OS/390 SecureWay Communications Server

IP User's Guide

Version 2 Release 8

GC31-8514-03

OS/390 SecureWay Communications Server

IP User's Guide

Version 2 Release 8

GC31-8514-03

Note:
Before using this information and the product it supports, be sure to read the general information under “Appendix L. Notices” on page 617.

Fourth Edition (September 1999)

This edition applies to OS/390 V2R8 (Program Number 5647-A01).

Publications are not stocked at the address given below. If you want more IBM publications, ask your IBM representative or write to the IBM branch office serving your locality.

A form for your comments is provided at the back of this document. If the form has been removed, you may address comments to:

IBM Corporation
Department CGMD
P.O. Box 12195
Research Triangle Park, North Carolina 27709
U.S.A.

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 RALVM17

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 is the fourth edition of this book, which has been revised and reissued for SecureWay® Communications Server for OS/390 V2R8. Like previous editions, this book describes how to use the applications available in IBM®® SecureWay Communications Server for OS/390® (CS for OS/390) to perform the following functions:

- Log on to a remote host
- Transfer data sets
- Send and receive electronic mail
- Print on remote printers
- Authenticate network users
- Display IBM GDDM®/MVS® graphics on X Window System workstations
- Run a command on another host
- Monitor the network
- Query name servers
- Manage network resources

SecureWay Communications Server for OS/390 is a part of the OS/390 family of products. For an overview and map of the documentation available for CS for OS/390 V2R8, refer to the *OS/390 Information Roadmap*.

Who Should Use This Book

This book is written for users and system administrators who want to use the applications that are available in CS for OS/390 V2R8.

Part 1 of this book provides information useful primarily to users, while Part 2 provides information useful to system administrators. Part 3 provides reference information useful to both.

Before using this book, you should be familiar with the IBM Multiple Virtual Storage (MVS) operating system, the IBM Time Sharing Option (TSO), and OS/390 UNIX System Services and the OS/390 UNIX shell. In addition, CS for OS/390 V2R8 should already be installed and customized for your network. For information about installing, refer to the *Program Directory*. For information about customizing, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

For more information about OS/390 UNIX® sendmail, go to the following URL:

http://www.sendmail.org

Where to Find More Information

The bibliography at the end of this book describes the books in the CS for OS/390 library, arranged according to task. The bibliography also lists the files and order numbers of books related to this book, or cited by name in this book.

Most licensed books were declassified in OS/390® V2R4 and are now included in the OS/390 Online Library Collection, SK2T-6700. The remaining licensed books appear in unencrypted BookManager® softcopy and PDF form on the OS/390 Licensed Product Library, LK2T-2499.

Where to Find Related Information on the Internet

You may find the following information helpful.

You can read more about VTAM, TCP/IP, OS/390, and IBM on these Web pages. For up-to-date information about Web addresses, please refer to informational APAR II11334.

Home Page	Web address
IBM SecureWay Communications Server product	http://www.software.ibm.com/network/commserver/
IBM SecureWay Communications Server support	http://www.software.ibm.com/network/commserver/support/
OS/390	http://www.ibm.com/os390/
OS/390 Internet Library	http://www.ibm.com/s390/os390/bkserv/
IBM Systems Center publications	http://www.ibm.com/redbooks
IBM Systems Center flashes	http://www.ibm.com/support/techdocs
VTAM and TCP/IP	http://www.software.ibm.com/network/commserver/about/csos390.html
IBM	http://www.ibm.com

For definitions of the terms and abbreviations used in this book, you can view or download the latest *IBM Networking Softcopy Glossary* at the following Web address:

http://www.networking.ibm.com/nsg/nsgmain.htm

Note: Any pointers in this publication to websites are provided for convenience only and do not in any manner serve as an endorsement of these websites.

How to Contact IBM Service

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

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 GC31-8514-03
SecureWay Communications Server for OS/390 V2R8

This edition contains new and changed information, indicated by vertical lines in the left margin.

New Information

- New OS/390 UNIX command information:
  - OS/390 UNIX oping -i *interface* parameter
  - OS/390 UNIX otracert -i *interface* parameter
  - OS/390 UNIX host command information
  - OS/390 UNIX hostname command information
  - OS/390 UNIX dnsdomainname command information
  - OS/390 UNIX domainname command synonym information
  - Synonyms now exist for the following OS/390 UNIX shell commands:

osnmp	the synonym is snmp
onetstat	the synonym is netstat
onslookup	the synonym is nslookup
otracert	the synonym is traceroute
orexec	the synonym is rexec
oping	the synonym is ping
orpcinfo	the synonym is rpcinfo

- New NETSTAT command information:
  - TSO NETSTAT VIPADYN and OS/390 UNIX onetstat -v commands
  - TSO NETSTAT SLAP and OS/390 UNIX onetstat -j commands
  - TSO NETSTAT REPORT additional parameters
- FTP RESTGet and NORESTGet parameters and descriptions

Changed Information

- The term eNetwork™ is replaced by SecureWay as part of our product name. The new name is SecureWay Communications Server.
- The Bibliography has been revised to show book number dash levels and delivery format.
- Changed NETSTAT information:
  - TSO NETSTAT ALL and OS/390 UNIX onetstat -A reports, updated to reflect the number of segments sent or received. Updated reports reflect the following changes:
    1. new segment counts for TCP connections
    2. TCP buffer size always display
    3. pending TCP receive indicator removed
  - TSO NETSTAT BYTEINFO and OS/390 UNIX onetstat -b reports, updated to support byte counts in kilobytes and megabytes when number of bytes is greater than 9,999,999

  - TSO NETSTAT CONFIG and OS/390 UNIX onetstat -f reports, updated to support multipath routing, Dynamic VIPA definitions, and the GLOBALCONFIG Profile statement
  - TSO NETSTAT DEVLINKS and OS/390 UNIX onetstat -d reports, updated to support the MPCOSA device/link
  - TSO NETSTAT ROUTE and OS/390 UNIX onetstat -y reports, updated to display 16-character link names
  - TSO NETSTAT and OS/390 UNIX onetstat example output, updated to display 8-byte connection ids
- FTP LOCSITE DCbdsn parameter description
- USCFxlate option added to the LPR command

Deleted Information

- SYSFTPD DD statement, removed from the FTP CLIENT SEARCH ORDERS, under the OS/390 UNIX SHELL column. This statement is invalid under the OS/390 UNIX shell.
- Reference to the PW.SRC file in the usage notes for the osnmp command. Bullet changed to indicate the osnmp command uses two configuration files.
- Four MIB objects, ibmMvsAtmOsasfChannelVtamId, ibmMvsTcpConnActiveOpen, ibmMvsTcpConnPendTcpRecv, and ibmMvsTcpConnBackoffCount, deleted from "Appendix D. Management Information Base (MIB) Objects" on page 571.

Summary of Changes
for GC31-8514-02
eNetwork Communications Server for OS/390 V2R7

This book was updated to document functional and service enhancements provided in CS for OS/390 V2R7.

The following information was new in this release:

- TSO NETSTAT CACHinfo and OS/390 UNIX onetstat -C options
- OSA-Express ARP output limitation noted in the TSO NETSTAT ARp option, and in the OS/390 UNIX onetstat -R option
- Command syntax and example output for the osnmp findname function
- SNMPv3 support
- pwtokey example output, added
- pwchange command, added to facilitate dynamic changes for user authentification and privacy keys

The following information changed in this release:

- TSO NETSTAT ALL and OS/390 UNIX onetstat -A output, updated to externalize new TCP connection sysplex cluster flag
- TSO NETSTAT CONFIG and OS/390 UNIX onetstat -f output, updated to reflect Dynamic XCF support
- MIB object table, updated to reflect support of SNMPv3 and the OMPROUTE Subagent
- SNMP capability statement, updated to show support of SNMPv3 and the OMPROUTE Version 2 MIB (RFC1850)
- pwtokey syntax, modified
- References to the /etc/snmpv2.conf file, changed to OSNMP.CONF (generic name indicating the file now has an augmented search order)

Summary of Changes
for GC31-8514-01
eNetwork Communications Server for OS/390 V2R6

This second edition of the book, updated to document service enhancements to CS for OS/390 V2R5, supported CS for OS/390 V2R6 and the OS/390 family of products.

The following information was new in this release:

- OS/390 UNIX sendmail and popper support

  CS for OS/390 V2R6 supports OS/390 UNIX sendmail to supplement the existing CS for OS/390 SMTP mail facility.

- Multicast Capability

  CS for OS/390 V2R6 adds host multicast datagram support to the existing support of the unicast datagram and broadcast datagram delivery mechanisms. Multicast is a way to distribute data to multiple nodes simultaneously. Host support implies implementation of the RFC 1112 standard.

Part 1. User Applications

Chapter 1. Getting Started

Transmission Control Protocol/Internet Protocol (TCP/IP) is a set of industry-standard protocols and applications that enable you to share data and computing resources with other computers, both IBM and non-IBM. By using TCP/IP commands at your workstation, you can perform tasks and communicate easily with a variety of other systems and workstations. SecureWay Communications Server for OS/390 (CS for OS/390) enables the user to interactively run TCP/IP applications (TCP/IP commands) from both the Time Sharing Option (TSO) and the OS/390 shell.

Understanding TCP/IP

TCP/IP is a set of protocols and applications that enable you to perform certain computer functions in a similar manner independent of the types of computers or networks being used. When you use TCP/IP, you are using a network of computers to communicate with other users, share data with each other, and share the processing resources of the computers connected to the TCP/IP network.

A computer network is a group of computer nodes electronically connected by some communication medium. Each node has the hardware and the programs necessary to communicate with other computer nodes across this communication medium. The node can be a PC, workstation, microcomputer, departmental computer, or large computer system. The size of the computer is not important. The ability to communicate with other nodes is important.

Computer networks enable you to share the data and computing resources of many computers. Applications, such as departmental file servers, rely on networking as a way to share data and programs.

Many forms of communication media are available today. Each is designed to take advantage of the environment in which it operates. Communication media consist of a combination of the physical network used to connect the computer nodes and the language, or protocol, they use to communicate with each other.

Physical Network

A physical network consists of electrical wiring and components, such as modems, bridges, controllers, access units, telephone lines, fiber optic cables, and co-axial cables. These are used to connect the computer nodes together. The physical network can connect two nodes in a single room or thousands of nodes communicating across large geographic areas. The most common networks in use today are Local Area Networks (LANs) and Wide Area Networks (WANs). LANs cover a limited distance, generally one or two floors or buildings, while WANs, using telecommunication facilities, are used for longer distances.

Protocols

Network protocols are the rules that define how information is delivered between nodes. They describe the sequence and contents of the data exchanged between nodes on the network. Network protocols determine how a computer node functions during communication with another node, how data is encoded to reach its destination safely, and what path it should follow. Protocols coordinate the flow of messages and can specify which node a message is destined for in the network. A

variety of protocols are used to take advantage of the characteristics of each of the physical network types. The most common protocols are Ethernet, 802.3, token ring, X.25, and System Network Architecture (SNA).

Network Devices

Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about network devices.

Addresses

A network address is a component of the communication network and is associated with both hardware and software. The address is the means by which the sending node selects the receiving node for data transfer. It is also used by the receiving node to recognize what data is destined for it. An address is a unique code assigned to every node on a network. But an address is formed differently for different protocols. The length, position, and method used to specify an address are unique for each protocol. A communication node using one protocol cannot recognize the address of another protocol.

Gateways

A network is designed and built using one physical network type and one protocol. All of the computers on that network can then communicate. Because computer nodes use different protocols based on which physical network type they are connected to, and because those using different protocols cannot communicate with each other, computers on different network types cannot share data and other computing resources.

A gateway is a network component that is physically connected to more than one network and can recognize the format of more than one network protocol. When required, a gateway translates the data transfer from the protocol of one network into the protocol of another. Computer nodes on two dissimilar networks can communicate through a gateway and share data and resources. A gateway between different physical networks is often called a router.

How TCP/IP Uses Networks

TCP/IP consists of a layered structure of protocols ranging from hardware-dependent programs to high-level applications. Each TCP/IP layer provides services to the layer above it and uses the services of the layer below it. The lowest layer, which is next to the physical layer, is not part of TCP/IP. This layer consists of existing protocols, such as Ethernet and token ring. TCP/IP uses the services of this layer to transport data across dissimilar networks, much like a gateway.

Local and Remote Nodes

A physical network is used by the hosts that reside on that network. Each host is a node on that network. A node is an addressable location in a communication network that provides host processing services. The intercommunication of these nodes gives rise to the concept of *local* and *remote* nodes. A *local node* pertains to a device, file, or host accessed directly from your system. A *remote node* pertains to a device, file, or host accessed by your system through the network.

Client and Server

A server is a computer or a program that contains data or provides services to be used by other computers on the network. Some of the common server types are file, print, and mail servers. They enable your computer to share the data, devices and resources of another computer. There are also servers that provide services to let you execute programs on a computer other than your own. This enables your computer to share the processing power of another computer. Servers are also known as daemons. Generally, a server runs continuously and can handle the requests of multiple clients simultaneously.

A client is a computer or a program that requests services or data from a server. A client could, for example, request that a file located at the server be sent across the network to the client.

TCP/IP Addresses

An address enables data to be routed to the chosen destination. Each destination in your network, as well as any other TCP/IP network you have access to, can be uniquely identified by its assigned address. A TCP/IP address is written in *dotted-decimal* notation. This scheme is numeric and consists of four groups separated by a period (.). For example, 9.67.1.100 represents a single host on a single network. 193.5.86.9 represents another host on another network. Most TCP/IP commands require you to include the address of the remote host where the server you wish to access resides. Each link (physical or virtual) on a host has an IP address.

Some commands use the C-function inet_addr() to convert dotted-decimal addresses entered through a terminal to the internally used 32-bit value. The design of this function enables certain ″short-cuts″ to be used in entering the data. Refer to the *OS/390 C/C++ User's Guide* explains the various ways the data may be input. The following commands use the inet_addr() function:

- FTP
- NETSTAT
- NSLOOKUP
- onetstat
- onslookup
- oping
- orpcinfo
- osnmp
- otracert
- pwtokey
- pwchange
- RPCINFO
- SNMP
- OSNMPD

For more information about the inet_addr() function, refer to the *OS/390 C/C++ Run-Time Library Reference*.

Network Names

An alternative to supplying a numeric address is to use the host name, rather than the address, in TCP/IP commands. (Each host may be assigned at least one name.) Your local host can resolve the name you supply in a command into the

correct numeric address. The names are translated using either a translation file or an application known as a *name server*. Your ability to use *network names* depends on how your TCP/IP network has been designed and which features have been installed.

Ports and Port Numbers

The use of *ports* and their identifying numbers are an extension to the addressing scheme. Once the address is used to deliver data to the desired host on the network, the *port number* is used to identify the process for which the data is used. This enables one host to provide more than one service.

How you define the port number depends on your configuration. Some applications make use of standard, or well-known, port numbers. Two applications at the same address cannot use the same port number. If you are configuring your system with multiple instances of TCP/IP on the same system, however, they will have different addresses and therefore the same port number can be used for the same function on each stack.

TCP/IP assumes the well-known port number unless you specifically specify otherwise when entering a TCP/IP command. A port number is entered as a decimal number on TCP/IP commands. For those cases when you are requesting the services of a user-developed server, you need to know the port number of that server.

Understanding What You Can Do

You can perform many functions from either of the two TCP/IP environments: TSO and OS/390 UNIX System Services (OS/390 UNIX). Table 1 shows the commands used in the two environments. The OS/390 UNIX commands are case-sensitive. All of the commands are described in detail in the following chapters.

Table 1. Commands Used in TSO and OS/390 UNIX

Task	TSO command	OS/390 shell command
Provides device, gateway, and local host status	NETSTAT	onetstat
Queries the name server to perform various tasks	NSLOOKUP	onslookup
Determines accessibility of a foreign node (remote node)	PING	oping
Executes a command on a remote host	REXEC	orexec
Displays server information	RPCINFO	orpcinfo
Provides diagnosis information for network problems	TRACERTE	otracert

In CS for OS/390, the OS/390 UNIX commands are in lowercase. The meaning of a parameter depends on whether it is in uppercase or lowercase..

CS for OS/390 commands provide a set of basic functions that include:

- Logging on to other hosts
- Transferring data sets and files between hosts
- Sending and receiving mail
- Using other hosts
- Printing to or from other hosts
- Testing network usability

Logging on to Other Hosts

The Telnet protocol provides a standardized interface that enables terminal devices and terminal-oriented processes on hosts that support TCP/IP to communicate with each other. The TELNET command enables you to log on to a remote host from TSO as though you are directly attached to that host. The MVS Telnet client does not run in the OS/390 UNIX environment.

Connecting to the OS/390 UNIX Telnet server from any client results in a session with the OS/390 shell as if the user had entered UNIX System Services from TSO in line mode or character mode. Once the OS/390 UNIX Telnet session has been established, you can enter any UNIX System Services command that can be issued from within the OS/390 shell.

See “Chapter 2. Logging on to a Host Using TELNET” on page 15, for more information about TELNET.

Transferring Data Sets between Hosts

When data is created or stored at one host but is processed by another host, some method for transferring the data between hosts is necessary. TCP/IP provides a command for transferring data sets and files between hosts. The FTP command moves or copies data sets and files between hosts that have either similar or dissimilar file systems. The FTP command provides subcommands that enable you to change the local and remote directories, set the transmission character code, list remote files, delete remote files, and send and receive files between hosts. You can use FTP to perform Structured Query Language (SQL) queries as well as submit jobs to JES for batch processing. It provides for security by requiring a user ID and password and enableing for remote logon and logoff. See “Chapter 3. Transferring Data Using the File Transfer Protocol (FTP)” on page 35, for a complete list of FTP functions.

Sending and Receiving Mail

The Simple Mail Transfer Protocol (SMTP) is a TCP/IP application that is used to transport electronic mail. Electronic mail enables you to send notes, messages, letters, or correspondence to others on the network. It is similar to sending a letter through the Post Office. You compose the message just as you would an ordinary letter, address the letter to one or more people and possibly carbon copy others. You enclose copies of the letter in envelopes, address them to the recipients, and give them to the delivery system. You expect the mail to be delivered to the correct address available for pickup when the recipient is ready. And you want any undeliverable mail returned to you. You can even keep a log of the mail you send and receive. The following commands are available to let you send and receive mail:

Command	Description
SMTPNOTE	Composes and sends mail from you to users on local or remote hosts. The TSO SMTPNOTE command helps you address the mail, set up a copy list, and enter the text of the message. The date, time, and your address are included automatically.
RECEIVE	Retrieves data that has been sent to you through the Job Entry Subsystem (JES). You use the RECEIVE command for your TCP/IP mail just as you do for other mail, messages, and data sets that are sent to you.

Command	Description
OS/390 UNIX sendmail	Composes and sends mail from you to users on local or remote hosts. The OS/390 UNIX Sendmail command helps you address the mail, set up a copy list, and enter the text of the message. The date, time, and your address are included automatically.

See “Chapter 4. Sending Electronic Mail Using SMTP Commands” on page 243, for more information about the SMTP mail facility.

See “Chapter 5. Sending Electronic Mail Using OS/390 UNIX sendmail and POP3 (Post Office Protocol)” on page 275, for more information about the OS/390 UNIX mail facility.

Using Other Hosts

Just as there are occasions when you want to transfer data to a host where it can be processed, there are also occasions when you want to process the data where it exists and send the processing results to another host. The data sets or files could be too large to transfer efficiently or all the data might be kept at one host for security reasons. The computing power necessary to perform some tasks could be more than your host is capable of or the only licensed copy of a required program might reside at some other host. TCP/IP provides a command that enables you to use the processing resources of other hosts. The remote commands (REXEC, orexec, and RSH) enable you to send any command that is valid on the remote host and receive the results at the local host. A user ID and password provide security checking at the remote host. The command sent to the remote host must not require user interaction to complete. See “Chapter 11. Monitoring the TCP/IP Network” on page 363, for more information about these commands.

Printing to or from Other Hosts

You can print reports, documents, listings, and so on completely independent of where the job or process that created them was executed by routing the data sets to a remote host for printing. Four TCP/IP commands are provided for remote printing:

Command	Description
LPR	Prints a data set on a remote printer. A variety of options enables you to specify how and where the data set is printed.
LPQ	Enables you to query a printer queue on a remote printer. You can query a printer queue for a specific job, a specific user ID, or all the jobs in a remote printer queue.
LPRM	Removes a job from a printer queue.
LPRSET	Specifies remote printer and remote host names when they are not specifically included in the Line Print commands.

Testing Network Usability

A set of tools is provided to assist in the operation and management of TCP/IP segments of the network. Some of these tools are complex and are used by system administrators. Others, such as the PING command, are available to, and are useful for, any user on the network.

The following commands let you test network availability, usability, and responsiveness. Some of these tools are available in the MVS environment, some

in the OS/390 UNIX environment, and others in both environments.

Command	Description
PING/ oping	Tests network connectivity of the local or remote host. It also measures the response time for a message to travel through the network to a remote host and return. See “Chapter 11. Monitoring the TCP/IP Network” on page 363, for more information about these commands.
NETSTAT/ onetstat	Shows local and remote addresses, routing tables, and statistics for hardware and software. See “Chapter 11. Monitoring the TCP/IP Network” on page 363, for more information about these commands.
RPCINFO/ orpcinfo	Determines which remote procedure call (RPC) servers are registered at different hosts in the network. It is used primarily by system administrators. See “Chapter 11. Monitoring the TCP/IP Network” on page 363, for more information about these commands.
DIG	Queries domain name servers. It is used primarily by system administrators to obtain information about the name servers operating in the network. See “Chapter 12. Querying the Domain Name System (DNS)” on page 433, for more detail on this command.
host	Queries domain name servers using the OS/390 UNIX resolver. It is used to look up host names and IP addresses. It can also be used by administrators to verify the configuration of the OS/390 UNIX resolver. See “Using the OS/390 UNIX host Command” on page 493 for more detail on this command.
hostname/ dsnsdomainname/ domainname	Displays the configured DNS host name or domain name for the local system. See “Using the OS/390 UNIX hostname Command” on page 495, “Using the OS/390 UNIX dnsdomainname Command” on page 497, or “Using the OS/390 UNIX domainname Command” on page 499 for more detail on these commands.
NSLOOKUP/ onslookup	Queries domain name servers. It is used primarily by system administrators to obtain information about the name servers operating in the network. See “Chapter 12. Querying the Domain Name System (DNS)” on page 433, for more detail on these commands.
NSUPDATE	Executes DNS update operations on host records. It is used primarily by system administrators to update information about the name servers operating in the network. See “Chapter 12. Querying the Domain Name System (DNS)” on page 433, for more detail on this command.
SNMP/osnmp	Provides a means for managing a multiprotocol, multivendor network environment. It enables management of network components, such as gateways, routers, and hosts. SNMP uses the NetView program to provide the end-user interface. OS/390 UNIX System Services osnmp uses the OS/390 shell to provide the end-user interface. See “Chapter 13. Managing TCP/IP Network Resources with SNMP” on page 501, for more detail on these commands.
TRACERTE/ otracert	Displays the route that a packet takes to reach the requested host. See “Chapter 11. Monitoring the TCP/IP Network” on page 363, for more information about these commands.

What You Need to Get Started

TCP/IP is a part of your OS/390 system. To use it you need a TSO user ID and password. If you are already a TSO user, you can begin using TCP/IP.

In order to use CS for OS/390 V2R8 applications, you must be authorized to use OS/390 UNIX System Services. For information about OS/390 UNIX, refer to the *OS/390 UNIX System Services User’s Guide*.

Ensure that you have the following before proceeding:

User IDs and passwords
You should have a user ID and password for each host you intend to use that requires user authorization and authentication. This includes most hosts you use. Some hosts on a TCP/IP network use a user ID of `anonymous` and a password of `guest` to permit all interested parties access to data sets contained at that host, but that is the exception and not the rule.

Host names
TCP/IP commands require that you know the name or dotted decimal TCP/IP address of the remote host you want to use.

Authorizations for data and programs
Your ability to access data sets and programs on remote hosts depends on the data security system used by that host. You might require authorization by the Resource Access Control Facility (RACF) or other security programs before you can gain access to data sets, commands, or other resources on remote hosts.

Electronic mail addresses
To send mail electronically, you need the TCP/IP mail address of the users you wish to send mail to and they need to know your electronic mail address to send you mail.

Printer names
You need the printer name and name of the remote host to which it is attached to print using TCP/IP.

Testing Commands with Loopback

In order to test your local machine, an address is reserved that always refers to your local host rather than any other hosts on a network. This class A network address is 127.0.0.1. You can also specify loopback as the host name. The only valid loopback address is 127.0.0.1.

You can use the loopback address with any TCP/IP command that accepts IP addresses, except TSO TRACERTE and OS/390 UNIX otracert. When you issue a command with the loopback address, the command is sent out from your local host's client and continues until it reaches the IP layer on your local host. The command is then sent on to your local host's server.

Note: Any command or data that you send using the loopback address never actually goes out on any network.

The loopback address is commonly used as the first step in diagnosing network problems. The information you receive indicates the state of your system and checks to ensure that the client and server code for the function you are testing is operating properly. You should see the same response as for a normal, successful command. If the client or server code is not operating properly, the same message that would be returned for an unsuccessful command is returned.

Obtaining Command Help

Commands typically support HELP or -? options that allow users to obtain on-line help. Additionally, end user commands supported in the OS/390 UNIX System Services shell environment support man pages. For example, typing **man oping** will display the manual pages for the oping command.

Chapter 2. Logging on to a Host Using TELNET

The TELNET protocol provides a standardized interface that enables terminal devices and terminal-oriented processes on hosts that support TCP/IP to communicate with each other.

The following subjects are covered in this chapter:
- Using the TELNET Command
- Using the TELNET Subcommands
- Suppressing Carriage Return and Line Feed
- Using TELNET 3270 DBCS Transform Mode

Using the TELNET Command

When you use the TELNET command to connect to a remote host running TCP/IP, the data displayed on your terminal is managed by TELNET for the remote host. As a result, the operation of your terminal may differ from what you are used to seeing when you are directly logged on to TSO or to another MVS application. For example, the remote host is running UNIX, DOS, OS/2®, VM, or another base operating system. You need to use the terminal operation procedures of the remote host operating system while you have a TELNET session with that remote host.

TELNET management of your terminal for the remote host can also cause operational differences. For example, the function keys described in “Using the TELNET Function Keys” on page 29 can result in different actions.

When the display data will not all fit on your screen, Linemode displays the HOLDING message in the lower right corner of your screen. If this message appears, press the CLEAR key to see the rest of the data.

If your TELNET session ends for any reason, the following message is displayed:

```
Session ended.  <ENTER> to return to TSO.
```

If you invoke the services of the MVS TELNET server from a non-MVS or a non-VM client, a transparent mode of operation might not be possible. If you use TELNET in line mode to access an MVS or VM TELNET server, all subsequent nested TELNET requests are automatically connected in line mode as a start-stop TTY terminal, and transparent (full-screen) operations are not possible.

When you return to TSO, a message explaining why the TELNET session ended is displayed. The following is an example of what is displayed when you return to TSO:

```
TELNET terminated -- Foreign host is no longer responding
```

Note: The OS/390 TELNET client does not support the Secure Sockets Layer (SSL) protocol.

TELNET Command

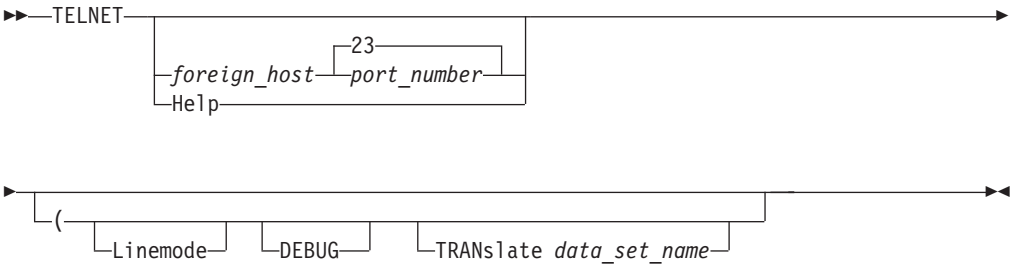
Purpose

The TELNET command enables you to log on to a foreign host that supports TCP/IP.

Notes:

1. For information on how to read syntax diagrams, see “Appendix J. How to Read a Syntax Diagram” on page 611.
2. The TSO TELNET command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format



Parameters

foreign_host
Specifies the name or IP address of the local or remote host. If you do not specify the name or IP address of the host, you are prompted for the *foreign_host*.

port_number
Specifies the port number to which you want to connect on the host. The default is well-known port 23.

Help
Provides a description of the TELNET command, its subcommands, and how it operates.

Linemode
Uses the line mode and prevents operation in the transparent mode.

In line mode, the foreign host output is displayed on your screen one line at a time, without full-screen capabilities.

Note: You cannot use the TELNET command to log on to an MVS host from an existing MVS line mode TELNET session. In this situation, the error message, `TELNET requires a 327x-Type terminal` is displayed.

In transparent mode, the foreign host full-screen capabilities are functional on your local terminal.

Transparent mode is the default.

DEBUG
Causes TELNET client-trace data, including the data transferred to and

received from the TELNET server, to be written to a data set defined by the DEBUGFIL DD statement in the user TSO LOGON procedure, or as specified by issuing the TSO ALLOC command.

The following is an example of the DEBUGFIL DD statement:

```
//DEBUGFIL  DD  DSN=USER28.TELNET.TRACE,DISP=OLD
```

The following is an example of the TSO ALLOC command:

```
ALLOC DDNAME(DEBUGFIL) DSNAME(USER28.TELNET.TRACE) OLD
```

TRANslate *data_set_name*
Specifies the name of a nonstandard translation table. If you specify this parameter, TELNET uses the translation table in the *user_id.data_set_name.*TCPXLBIN data set, rather than the standard translation tables *user_id.*TELNET.TCPXLBIN or *hlq.*TELNET.TCPXLBIN.

If *user_id.data_set_name.*TCPXLBIN does not exist, TELNET uses *hlq.data_set_name.*TCPXLBIN.

If *user_id.data_set_name.*TCPXLBIN and *hlq.data_set_name.*TCPXLBIN do not exist, or if they were incorrectly created, TELNET ends with an error message. A nonstandard translation table is used in line mode only.

Examples

- To log on to a host with an IP address of 1.1.2.3, enter:

  ```
  TELNET 1.1.2.3
  ```

  The following is displayed:

  ```
  System:
               READY
    User:     TELNET 1.1.2.3
  System:
               MVS TCP/IP TELNET V3R2
               Connecting to 1.1.2.3, port TELNET (23)
               ***
               Using Transparent Mode...

               Notes on using TELNET when in Transparent Mode:
               - To enter TELNET Command, Hit PA1
               ***
  ```

- If your user ID is `RON` and the translation table `RON.EXAMPLE.TCPXLBIN` is required rather than the standard one, you should enter:

  ```
  TELNET 1.1.2.3 (TRANslate EXAMPLE
  ```

  – If the remote host is neither MVS nor VM, and you specify a nonstandard translation table, a line-mode connection is automatically used.
  – If the remote host is an MVS or VM host and you specify a nonstandard translation table without the line-mode parameter, the nonstandard translation table is ignored.
  – If the host is an MVS or VM host and both the line-mode parameter and a nonstandard translation table are specified, the nonstandard translation table is used.

Usage

- The minimum abbreviation for each parameter is shown in uppercase letters.
- TELNET normally operates in transparent mode. In 3270 transparent mode, all full-screen capabilities of the remote host are functional at your local display

station, but only the **PA1** key is operational. In line mode, the remote host output is displayed on your screen one line at a time, without full-screen capabilities.

- The TELNET command supports IBM 3270-type display stations. Examples of supported display stations are:
  - IBM 3178 Display Station
  - IBM 3179 Display Station
  - IBM 3180 Display Station
  - IBM 3191 Display Station
  - IBM 3192 Display Station
  - IBM 3193 Display Station
  - IBM 3194 Display Station
  - IBM 3275 Display Station Model 2
  - IBM 3276 Control Unit Display Station Models 2, 3, and 4
  - IBM 3277 Display Station Model 2
  - IBM 3278 Display Station Models 2, 3, 4, and 5
  - IBM 3279 Color Display Station Models 2 and 3

Using the TELNET Subcommands

You must be in the TELNET environment to use the TELNET subcommands.

To invoke a TELNET subcommand while you are logged on to the foreign host, press the designated PF key. After you press the PF key, you will be prompted to enter a TELNET subcommand. You can enter TELNET subcommands in uppercase or lowercase. Table 2 lists the TELNET subcommands.

Table 2. TELNET Subcommands

Subcommand	Description	Page
AO	Stops the display of information	20
AYt	Queries the existence of the connection	21
Brk	Sends a **Break** or **Attn** keystroke	22
Help or ?	Displays help information	23
Ip	Interrupts the current process	24
Pa1	Sends a **PA1** keystroke in transparent mode	25
Quit	Disconnects from the foreign host	26
Synch	Clears the data path	27

Note: The minimum abbreviation for each subcommand is shown in uppercase letters.

AO—Terminate Output Display

Purpose

Use the AO (Abort Output) subcommand to stop the display of output.

Format

Parameters

There are no parameters for this subcommand.

Usage

The AO subcommand is used to clear any output that has already been produced, but has not been displayed on your terminal.

AYT—Query the Connection

Purpose

Use the AYT (Are You There) subcommand to query the existence of the connection.

Format

Parameters

There are no parameters for this subcommand.

Usage

- You can use the AYT subcommand to check for the existence of a TELNET connection. For example, if you feel that a command is taking longer than it should to complete, issue the AYT subcommand to test whether the connection is still active.
- If the connection exists and you are operating in transparent mode, the terminal makes a sound. If you are operating in line mode, you receive a message from the TELNET server.

BRK—Send a Break or Attention Keystroke to a Host

Purpose

Use the BRK subcommand to send a **Break** or Attention (**Attn**) keystroke to the remote session.

Format

Parameters

There are no parameters for this subcommand.

Usage

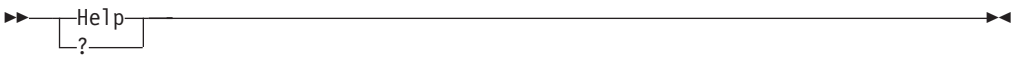
You can use the BRK subcommand to end a command without terminating the TELNET session.

HELP—Display Help Information

Purpose

Use the HELP (or ?) subcommand to access the help facility.

Format



Parameters

There are no parameters for this subcommand.

Usage

- After your TELNET connection is established, your display station screen is controlled by the foreign host, so the help you see is different for line mode and transparent mode. The help available in transparent mode is abbreviated because TELNET does not have control of the screen.
- When you invoke the HELP or ? subcommand in line mode, TELNET displays the help information one line after another, as in the following example:

```
Once connected, follow the log in and usage
procedures of the remote host.
To invoke one of several TELNET commands, hit a
PF key (PF4-12, PF16-24), and then enter any of
the following commands:
Help or ? -- Receive (this) assistance
AYT       -- Are You There?
AO        -- Abort Output
BRK       -- Break
IP        -- Interrupt Process
SYNCH     -- Clear data path, except for TELNET commands
Quit      -- Quit the TELNET session

The following PF settings are in force:
  PF1 or 13 -- Retrieve previous input line
  PF2 or 14 -- Scroll halfway up
  PF3 or 15 -- Turn off display of user-line; designed
               to be used before entering password
For control characters, enter ¢c or ′c where c is:
 "0": 00, a" - "z" or "A" - "Z": 0x01-0x1A
                         "2" - "6": 0x1B-0x1F
 "{": 0x5B, "}": 0x5D, "#": 0x7F
```

- When you invoke the HELP or ? subcommand in transparent mode, TELNET overwrites one line of the current screen with the help information, as in the following example:

```
 Valid TELNET cmds: AO,AYT,BRK,IP,PA1,QUIT,SYNCH.
```

IP—Interrupt the Process

Purpose

Use the IP subcommand to interrupt the current process running on the remote host.

Format

Parameters

There are no parameters for this subcommand.

Usage

You can use the IP subcommand if you want to stop a process that is in a loop, or when you want to stop a process that you inadvertently started.

PA1—Send the PA1 Keystroke to a Host

Purpose

Use the PA1 subcommand to send a **PA1** keystroke to the remote session in transparent mode.

Format

Parameters

There are no parameters for this subcommand.

Usage

- The PA1 subcommand operates only in transparent mode. This subcommand replaces the **PA1** attention key on the remote host.
- When there are nested TELNET sessions, use the **PA1** key to enter a TELNET subcommand in the first open TELNET session. To enter a TELNET subcommand in the second open TELNET session, send a PA1 subcommand from the first session.
- You would normally interrupt a PING command by pressing **PA1** . However, in a transparent mode TELNET session, this key is used to invoke a TELNET subcommand. You would issue a PA1 subcommand to interrupt the PING command instead.

QUIT—End the TELNET Session

Purpose

Use the QUIT subcommand to end the TELNET session.

Format

Parameters

There are no parameters for this subcommand.

Usage

- You should use the QUIT subcommand carefully because it can create an MVS error condition. If you do not reconnect within a timeout period, your TELNET session is cancelled.
- If you are logged on to an application on a remote host, and that application is defined as disconnectable to VTAM® and TCP/IP, you can use the QUIT subcommand to disconnect from the remote host without logging off the application.
- When you want to end a logon session with the host, use the logoff procedure of the host.

For more information about defining applications to VTAM and TCP/IP, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

SYNCH—Clear the Data Path

Purpose

Use the SYNCH subcommand to clear the data path.

Format

Parameters

There are no parameters for this subcommand.

Usage

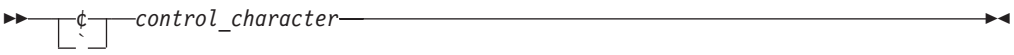
The SYNCH subcommand clears the data path to the host, except for any TELNET subcommands in the data path. This subcommand enables you to ensure that commands issued when the TELNET server is inactive are not executed when the TELNET server becomes active.

¢ and `—Send ASCII Control Characters to a Host in Line Mode

Purpose

Use the cent sign (¢) and the grave accent (`) in line mode to indicate a control character.

Format



Parameters

control_character
Indicates the ASCII control character that you want to send to the host. The purpose of each control character is specific to the remote host.

Examples

To send **Ctrl** - **p** , use either use either: ¢p or `p.

Usage

- If you want to use ¢ or ` without indicating a control character, you must enter these characters twice.
- The ASCII control characters are shown in Table 3.

Table 3. ASCII Control Characters

Character Input	ASCII Output
`A – `Z	01 – 1A (Ctrl-a – Ctrl-z)
`{	5B (left square bracket - [)
`}	5D (right square bracket -])
`2 – `6	1B – 1F
`#	7F (DEL)

Using the TELNET Function Keys

This section describes the functions that are assigned to PF keys when you invoke TELNET in transparent mode and line mode.

Transparent Mode Function Keys

In transparent mode, the only function key available is the **PA1** attention key. It is used to invoke a TELNET subcommand. If there is more than one nested TELNET session, the **PA1** key is used to invoke a TELNET subcommand for the first TELNET session.

See “PA1—Send the PA1 Keystroke to a Host” on page 25 for information about how to send the PA1 keystroke to the foreign host session.

Line Mode Function Keys

Table 4 describes the function keys that are available in line mode.

Table 4. TELNET Function Keys in Line Mode

Function Key	Description
PF4 – **PF12**, **PF16** – **PF24**	Enables you to invoke a TELNET subcommand. After pressing one of these function keys, enter a subcommand or enter Help to get a list of valid subcommands.
PF1, **PF13**	Retrieves the previous input line, except when the line was entered in hidden mode for security reasons.
PF2, **PF14**	Scrolls halfway up the screen.
PF3, **PF15**	Turns off input line display so data is not echoed to the screen. For example, use either of these keys before entering your password to keep it from being displayed.

Suppressing Carriage Return and Line Feed

It is useful if the command environment of the foreign host responds when you enter a single character, without the need for a carriage return and line feed after that character. This function is also useful when your cursor is at the end of the input field, but you want to continue the line without introducing a carriage return.

Figure 1 and Figure 2 on page 30 show the output of a BSD UNIX® program called MORE. This program displays one line or one page at a time. A carriage return character (CR) causes it to display one line, while a blank character causes it to display one page. If you are executing this program from an MVS host, use the grave accent (`) character to suppress the CR that is normally sent when you press Enter.

```
% more hosts.local
NET : 4.0.0.0 : SATNET :
NET : 6.0.0.0 : YPG-NET :
NET : 7.0.0.0 : EDN-TEMP :
NET : 8.0.0.0 : BBNCCNET :
NET : 9.0.0.0 : IBM :
NET : 10.0.0.0 : ARPANET :
NET : 12.0.0.0 : ATT :
NET : 13.0.0.0 : XEROX-NET :
NET : 14.0.0.0 : PDN :
NET : 15.0.0.0 : HP-INTERNET :
NET : 18.0.0.0 : MIT-TEMP :
NET : 21.0.0.0 : DDN-RVN :
NET : 23.0.0.0 : DDN-TC-NET :
NET : 24.0.0.0 : MINET :
NET : 25.0.0.0 : RSRE-EXP :
NET : 26.0.0.0 : MILNET :
NET : 27.0.0.0 : NOSC-LCCN-TEMP :
NET : 28.0.0.0 : WIDEBAND :
NET : 29.0.0.0 : MILX25-TEMP :
NET : 30.0.0.0 : ARPAX25-TEMP :
NET : 31.0.0.0 : UCDLA-NET :
NET : 35.0.0.0 : MERIT :
--More--(0%)  [HIT <ENTER> HERE.  ONE LINE IS DISPLAYED]
NET : 36.0.0.0 : SU-NET-TEMP :
--More--(0%)
```

Figure 1. Sample Output of the MORE Program—First Screen

```
NET : 39.0.0.0 : SRINET-TEMP :
--More--(0%)  [HIT <BLANK>, <ACCENT GRAVE>, <ENTER>]
NET : 39.0.0.0 : SRINET-TEMP :
NET : 41.0.0.0 : BBN-TEST-A :
NET : 42.0.0.0 : CAN-INET :
NET : 44.0.0.0 : AMPRNET :
NET : 46.0.0.0 : BBNET :
NET : 128.1.0.0 : BBN-TEST-B :
NET : 128.2.0.0 : CMU-NET :
NET : 128.3.0.0 : LBL-IP-NET1 :
NET : 128.4.0.0 : DCNET :
NET : 128.5.0.0 : FORDNET :
NET : 128.6.0.0 : RUTGERS :
NET : 128.7.0.0 : KRAUTNET :
NET : 128.8.0.0 : UMDNET :
NET : 128.9.0.0 : ISI-NET :
NET : 128.10.0.0 : PURDUE-CS-EN :
NET : 128.11.0.0 : BBN-CRONUS :
NET : 128.12.0.0 : SU-NET :
NET : 128.13.0.0 : MATNET :
NET : 128.14.0.0 : BBN-SAT-TEST :
NET : 128.15.0.0 : S1NET :
NET : 128.16.0.0 : UCLNET :
NET : 128.17.0.0 : MATNET-ALT :
--More--(1%)
```

Figure 2. Sample Output of the MORE Program—Second Screen

Using TELNET 3270 DBCS Transform Mode

When 3270 DBCS transform mode is configured for the MVS TELNET server, all new line-mode sessions to the server are introduced with a panel where you can select transform mode or line mode.

TELNET 3270 DBCS transform mode is used to provide 3270 DBCS emulation, while the 3270 processing is done only at the host end of the connection. This enables full-screen access from non-3270 terminals.

TELNET 3270 DBCS transform mode supports terminals of the VT100/VT220 family of terminals, including the following:

- VT100
- VT282

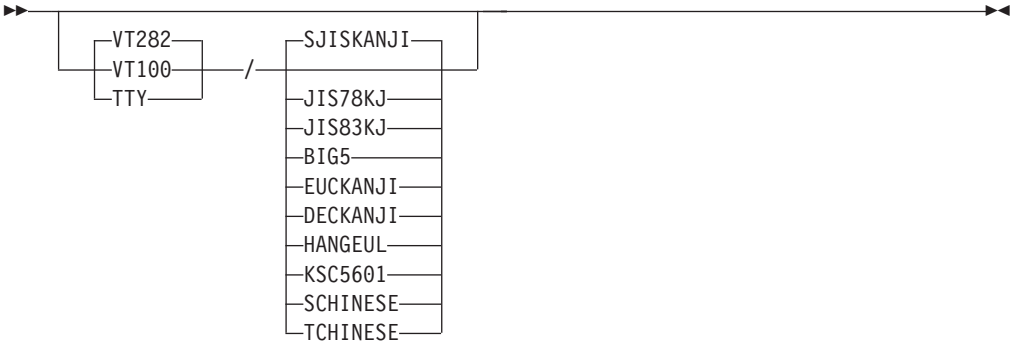
To log on to the server using 3270 DBCS transform mode, specify the `LINEMODE` option on the TELNET client command line. The following banner is displayed if transform mode is available:

```
IBM TCP/IP TELNET SERVER DBCS SERVICE START AT HH.MM.SS ON MM/DD/YY
KEY-IN YOUR TERMINAL TYPE & CONVERSION TYPE:
```

Terminal and Conversion Type

Format

When the DBCS banner appears, enter the required terminal and conversion type.



Parameters

VT100
VT100 terminal type, full-screen mode—no DBCS support

VT282
VT282 terminal type, full-screen mode—with DBCS support

TTY
Line mode—no DBCS support. This option bypasses operation in transform mode.

SJISKANJI
Shift JIS Kanji DBCS conversion

JIS78KJ
JIS Kanji 1978 DBCS conversion

JIS83KJ
JIS Kanji 1983 DBCS conversion

BIG5
Big-5 DBCS conversion

EUCKANJI
Extended UNIX code Kanji DBCS conversion

DECKANJI
DEC Kanji DBCS conversion

HANGEUL
Hangeul DBCS conversion

KSC5601
Korean Standard code KSC-5601 DBCS conversion

SCHINESE
Simplified Chinese DBCS conversion

TCHINESE
Traditional Chinese (5550) DBCS conversion

Usage

- Do not enter any spaces between the terminal type and the slash character (/), or between the slash character (/) and the conversion type. For example, to specify a VT282 terminal with Shift JIS Kanji DBCS conversion, enter the following:

  ```
  VT282/SJISKANJI
  ```

- If the conversion type is not specified, it defaults to the CODEKIND specified in the TNCBCSTM configuration data set. If neither terminal type nor conversion type is specified, the terminal type defaults to VT282 and the conversion type to the CODEKIND specified in the TNDBCSTM configuration data set.
- TELNET 3270 with DBCS transform mode supports a screen size of 24 by 80. Unpredictable results may occur when using a larger screen size.
- The maximum number of concurrent TELNET 3270 DBCS Transform connections is 250.

Context

For more information about using translation tables, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

For information about character sets, see “Character Set Cross Reference Table” on page 601.

For information about the TELNET extensions for terminals other than the 3270 family, see “Appendix H. TELNET Extensions” on page 601.

Chapter 3. Transferring Data Using the File Transfer Protocol (FTP)

The FTP command enables you to transfer data sets between your local host and any host that supports TCP/IP. Using the FTP command and its subcommands, you can sequentially access multiple hosts without leaving the FTP environment.

This chapter describes how to use the FTP command and its subcommands. Specifically, FTP enables you to:

- Establish a connection to a remote host or your local host
- Identify yourself to the host
- Obtain status and system information about the host
- Work with directories on the remote host
- Work with directories on your local host
- Transfer data sets to and from the host
- Pass TSO commands to your local host
- Invoke an OS/390 UNIX System Services command shell
- Set up an FTP proxy session between two FTP servers
- Send information to the local and remote hosts
- Obtain assistance for the FTP subcommands
- Submit FTP jobs in batch
- Use the EXEC interface
- Interface with JES
- Perform DB2® SQL queries
- Transfer DBCS or Unicode (UCS-2) data

Using the FTP Command

Before transferring files between your local host and a remote host, or using any other FTP functions, you must use the FTP command to enter the FTP environment. The FTP command can be used from either the TSO environment or the OS/390 UNIX shell.

FTP Command—Enter the FTP Environment

Purpose

Use the FTP command to enter the FTP environment.

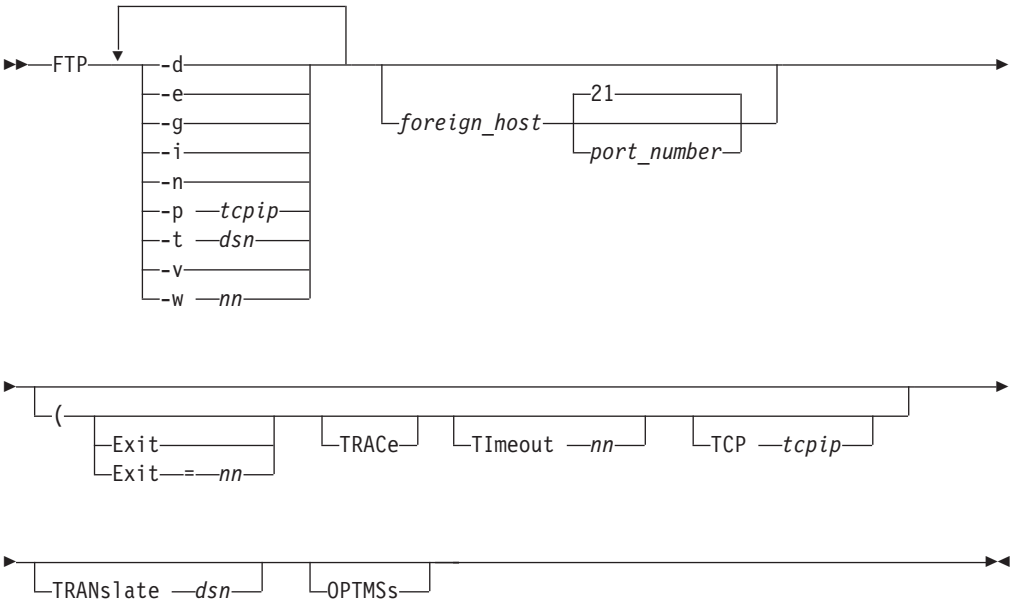
Note: In an OS/390 UNIX environment, you receive an error message if you use this form of the FTP command:

```
ftp 1.1.2.3 (trace
```

Instead, use the standard UNIX flag (for example, -d) or precede the left parenthesis with an escape character, such as the backslash (\):

```
ftp 1.1.2.3 \(trace
```

Format



Parameters

-d Starts the generation of tracing output. Equivalent to TRACE.

-e Terminates FTP, for certain FTP errors, with a nonzero MVS return code. Equivalent to EXIT.

-g Turns off metacharacter expansion (globbing). Equivalent to the GLOB subcommand.

-i Turns off interactive prompting for MDELETE, MGET, and MPUT subcommands. Equivalent to the PROMPT subcommand.

-n Inhibits automatic login. If you have defined a NETRC data set, it will not be used to log on to this session.

-p *tcpip*
Indicates the name of the TCP on the local host to which the FTP client should connect. This parameter is ignored if your system is not configured for multiple instances of TCP/IP. This is equivalent to TCP *tcpip*.

-t *dsn*
Specifies the name of a nonstandard translation table. Equivalent to TRANSLATE *dsn*.

-v Enables verbose mode. This parameter gives you extra information (such as message IDs) when running in OS/390 UNIX.

Note: If you are running the FTP client in TSO, you can have message numbers appended to client messages by using the profile option MSGID.

-w *nn*
Specifies the number of seconds to be used for the TIMEOUT parameters. Equivalent to TIMEOUT *nn*.

foreign_host
Specifies the name of the host to which you are connecting. Specify the host by its host name or its IP address. The host can be a remote host or your local host.

You are prompted for a host name if you do not specify a *foreign_host* with the FTP command. If you specify *foreign_host* incorrectly, or if the host is not accessible, you enter the FTP environment without connecting to a host. You should then use either the OPEN subcommand to attempt another connection with a host, or the QUIT subcommand (or Ctrl-C, in OS/390 UNIX) to exit the FTP environment.

port_number
Specifies the port number of the FTP server on the remote host. The default is well-known port 21. The maximum port number that can be specified is 65534. (This parameter should not be used unless you are sure there is a server listening on a port other than the well-known port 21 at the destination.)

Exit
Terminates FTP, for certain FTP errors, with a nonzero MVS return code. See Table 15 on page 80 for examples of these return codes.

Exit=*nn*
Terminates FTP with a nonzero return code of your choice when an FTP error occurs. Valid values are in the range 0–255.

TRACe
Starts the generation of tracing output. TRACe is used in debugging.

TImeout *nn*
Specifies the number of seconds to be used for the following TImeout parameters:

MyopenTime (default=60)
Defines the amount of time to wait for a session to open before terminating the attempt and reporting an error.

NOTE: This connection time interval is influenced by timers in the stack and the session attempt may be terminated well in advance of the time specified by TImeout.

DconnTime (default=120)
Defines the amount of time to wait after attempting to close a data transfer before terminating the connection and reporting an error.

CconnTime (default=30)
Defines the amount of time to wait after attempting to close a control connection before terminating the connection and reporting an error.

InactTime (default=120)
Defines the amount of time to wait with no activity for a session before terminating it and reporting an error. (This applies only if waiting for a server control response. If no command is pending, this timeout occurs only if the server session is terminated, for example, if the server times out.)

DataCtTime (default=120)
Defines the amount of time to wait after attempting to send or receive data before terminating the connection and reporting an error.

Note: Each send and receive uses a 185k buffer, so essentially 185k of data must be transferred within DataCtTime or the timer will pop. Those with slower connections will need to consider this when setting DataCtTime.

Numeric values in the range 15–720 are accepted for the Timeout parameter. All five of the Timeout parameters are set to the value you choose at invocation.

If you want to individually customize the values, you must use the FTP.DATA data set. If an FTP.DATA data set exists but you also specify a Timeout value at invocation, the Timeout parameter overrides the FTP.DATA data set.

If you specify an incorrect value for the Timeout parameter, FTP uses the default value.

TCP *tcpip*
Indicates the name of the TCP on the local host to which the FTP client should connect. This parameter is ignored if your system is not configured for multiple instances of TCP/IP.

Note: You must specify this value as a parameter, not as a value in the FTP.DATA data set. You may choose to specify this value with the TCPIPJOBNAME statement in the TCPIP configuration data file.

TRANslate *dsn*
Specifies the dataset name of a nonstandard translation table. If you specify this parameter, FTP uses the translation table in the *user_id.data_set_name.*TCPXLBIN data set, rather than the standard translation table provided with TCP/IP (*hlq*.STANDARD.TCPXLBIN). The *hlq*.STANDARD.TCPXLBIN data set is never used if you specify the TRANSLATE parameter.

If *user_id.data_set_name.*TCPXLBIN does not exist, FTP uses *hlq.data_set_name*. If *user_id.data_set_name.*TCPXLBIN and *hlq.data_set_name*.TCPXLBIN do not exist, or if they were incorrectly created, FTP ends with an error message.

If LOADDBCSTABLES is specified in *hlq*.TCPIP.DATA, *data_set_name* is used to determine which DBCS translation table to load. To load and use a customized DBCS translation table, an SBCS table must be defined as hlq.data_set_name*hlq.data_set_name*.TCPXLBIN or *user_id.data_set_name.*TCPXLBIN or the FTP request fails. A copy of *hlq*.STANDARD.TXPXLBIN can be used if you do not require a modified SBCS table.

Note: If the SBCS translation table that you need for data transfers does not support standard encodings for the portable character set, do not use the TRANSLATE parameter of the FTP command because this can adversly effect the EBCDIC-to-ASCII conversion of FTP commands. Instead, use the CTRLConn and SBDataconn statements in your local FTP.DATA to specify different SBCS tables for the control and data connections, or use the LOCSITE SBDataconn subcommand to change the SBCS translation for the data connection. For information on specifying these statements, see “Using Different SBCS Translation Tables for the Control and Data Connections” on page 68. If you also require a local DBCS translation table, you can name it *userid*.FTP.TCPdbBIN and find it in the client search order. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for information about the DBCS translation table search order.

OPTMSs

Use the OPTMSS parameter to enable TCP to build optimal size segments.

The option can be turned on in the following ways:

- `//STEP EXEC PGM=FTP,PARM='... (EXIT OPTMSS TRACE'`
- Via TSO: `FTP ... (EXIT OPTMSS TRACE`
- INPUT DD: `.open remote.server.ip.address (EXIT OPTMSS`

Usage

- When starting FTP in a TSO environment that includes support for the REXX programming language, you receive the following message:

  ```
  CSV003I Requested module IRXSTK not found
  ```

  This is a normal informational message when starting FTP in a TSO environment.
- If entered from OS/390 UNIX, the FTP flags (-d, -e, -g, -i, -n, -p, -t, -v, and -w) must be entered in lowercase. These options can be entered in lowercase or uppercase from TSO.

Context

- See “OPEN Subcommand—Connect to the FTP Server” on page 178 and “QUIT Subcommand—Leave the FTP Environment” on page 186 for more information about the OPEN and QUIT subcommands.
- See “FTP EXIT Return Codes” on page 79 for more information about the EXIT return codes.
- See “Changing Local Site Defaults Using FTP.DATA” on page 59 for information about the FTP.DATA data set.
- Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for information about the TCPIP.DATA data set or loading and customizing DBCS translation tables.

Logging on to FTP

If you correctly specify a foreign host with the FTP command, you are prompted to identify yourself. The following is a sample of the information that is displayed after you successfully invoke the FTP command with *foreign_host* correctly specified.

```
IBM FTP CS/390 V2R8   1997  314 01:11 UTC
Connecting to MVS1 9.67.58.227, port 21
220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
220 Connection closes if idle for more than 5 minutes.
NAME (<host>:tsouserid):
```

After successfully identifying yourself, you are prompted for a password if the foreign host requires a password. If you enter the password correctly, you are connected to the foreign host.

Note: You can use the data set NETRC to automatically log on to a remote host. For information about using NETRC, see "NETRC Data Set".

For the procedure to enter the FTP environment using the FTP command, see "Establishing and Exiting a Connection" on page 41 for more information.

Allocating FTP Input and Output Data Sets

When you invoke the FTP command, a check is made to see whether a data set is allocated to INPUT. If a data set is allocated, subcommands are read from that data set rather than from your terminal. Similarly, a check is also made to see whether a data set is allocated to OUTPUT. If so, all FTP prompts and replies are written to that data set rather than to your terminal.

The record length and block size of the output data set can be any size. If the logical record length of the output data set is less than 100 bytes, some messages could be truncated or wrapped around to the next line.

If you create INPUT and OUTPUT data sets, use the following guidelines:

- Specify the INPUT data set:
  - Record format=FB
  - Logical record length=80
  - Block size is a multiple of 80
- Specify the OUTPUT data set:
  - Record format=FB
  - Logical record length=160
  - Block size is a multiple of 160

NETRC Data Set

The *user_id*.NETRC data set (/$HOME/.netrc, in OS/390 UNIX) provides you with an alternative to specifying your *user_id* and *password* as FTP parameters when you want to FTP to a remote host. The following example shows you how to specify the *user_id*.NETRC data set:

```
MACHINE mvs1.tcp.raleigh.ibm.com LOGIN user28 PASSWORD user28
MACHINE 9.67.112.25 LOGIN user28
```

The keywords MACHINE, LOGIN, and PASSWORD do not have to be uppercase. The variables *user_ID* and *password* may be case-sensitive, depending on the remote host. (For example, UNIX or AIX® hosts require *user_ID* and *password* to be specified case-sensitive.)

To invoke the *user_id*.NETRC data set and automatically log on to the remote host named MVS1, enter the FTP command as shown in the following example:

```
  User:   ftp mvs1
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to MVS1 9.67.112.25, port 21
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will close if idle for more than 5 minutes.
           >>>USER user28
           331 Send password please.
           >>>PASS ********
           230 USER28 is logged on.  Working directory is "/u/user28".
           Command:
```

In order to bypass definitions in a *user_id*.NETRC data set for an ftp session, specify the -n parameter. For information about using the *user_id*.NETRC data set in a batch file, see “Submitting FTP Requests in Batch” on page 73.

Getting Help

The FTP Help subcommands are listed in Table 5.

Table 5. FTP Subcommands for Getting Help

Subcommand	Description	Page
?	Provides an introduction to using FTP	137
HElp	Displays help information for FTP	137

Establishing and Exiting a Connection

You normally establish a connection to a foreign host when you invoke the FTP command with a *foreign_host* specified. If you are not successful in specifying a foreign host, or if you need to connect to a different foreign host, use the subcommands listed in Table 6.

Table 6. FTP Subcommands for Establishing and Exiting a Connection

Subcommand	Description	Page
ACcount	Sends host-dependent account information	104
CLose	Disconnects from the foreign host	117
Open	Opens a connection to a foreign host	178
PAss	Supplies a password to the foreign host	179
QUIt	Leaves the FTP command environment	186
User	Identifies you to a foreign host	241

Example of Establishing and Exiting a Connection

This example shows how a single FTP session can be used to connect to the following multiple foreign hosts:

1. MVS Host: 192.9.2.1
2. VM Host: 192.9.2.4

```
  User:  ftp 192.9.2.4
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 192.9.2.4, port 21
           220-FTPSERVE at IBM VM V3R2 at APPLE.CORE.PIT, 09:28:02 EST
            MONDAY 09/08/97
           220 Connection will close if idle for more than 5 minutes.
           NAME (<host>:tsouserid):
  User:  vmuser
System:
           >>USER vmuser
           331 Sent password please.
           Password:
           >>>PASS ********
           230 VMUSER logged in; working directory = VMUSER 191

           Command:
  User:  account
System:
           Usage: ACCT account-information

           Command:
  User:  acct vmuser
System:
           >>>ACCT ********
           230 You now have write permission to VMUSER 191

           Command:
  User:  close
System:
           >>>QUIT
           221 Quit command received. Goodbye.

           Command:
```

```
 User:  open 192.9.2.1
System:
           Connecting to 192.9.2.1, port 21
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will close if idle for more than 5 minutes.

           Command:
  User:  user rtp
System:
           >>>USER rtp
           331 Send password please.
           Password:
           >>>PASS ********
           230 RTP is logged on.  Working directory is "/u/rtp".

           Command:
  User:  close
System:
           >>>QUIT
           221 Quit command received. Goodbye.

           Command:
  User:  quit
System:
           READY
```

Resource Access Control Facility (RACF) Considerations

When you first log on to an FTP session, the default working directory at the MVS server is the *user_id* you used to connect to the foreign host.

If your MVS server is running Resource Access Control Facility (RACF) 1.9 and your TSO user IDs are defined through RACF, the PREFIX defined in the PROFILE of your TSO user ID is used as the default working directory. If no PREFIX has been defined, the user ID is the working directory.

Note: To use FTP, your user ID must have an OMVS segment defined (as defaulted).

The PREFIX of a TSO user ID can be set or changed by using the TSO PROFILE command as follows:

1. Log on to TSO on the MVS system of the FTP server.
2. Set your new prefix using the TSO PROFILE command:

   `TSO PROFILE Prefix(`*`prefix`*`)`

   where *prefix* is any TSO prefix that you choose.

   Notes:

   a. You must enter both the opening and closing parentheses.
   b. At this point, the TSO prefix is defined for your current TSO session but is not known to RACF® or the FTP server until you log off and log on.
3. Log off to save the new default working directory name.

The TSO prefix should now be your default working directory whenever you log on to an FTP session on that FTP server. To make sure you set up the default working directory correctly, use the following steps.

1. Establish an FTP session to the FTP server.
2. Issue a PWD command. This should show the TSO prefix as your new default working directory. For information on using the PWD command, see “PWD Subcommand—Display the Current Working Directory” on page 185.

Notes:

1. To use the TSO PREFIX as your default working directory, you must have installed RACF Version 1.9 and you must define your TSO user IDs through RACF.
2. When you log on to an FTP session from an OS/390 UNIX shell, the default local working directory is the directory from where FTP was started.

Obtaining Status and System Information

To retrieve and display status information about the local host and remote host, use the subcommands listed in Table 7.

Table 7. FTP Subcommands for Obtaining Status and System Information

Subcommand	Description	Page
!	Passes a OS/390 UNIX System Services command to the OS/390 shell. This command must be issued while using FTP in the OS/390 shelll.	103
DEBug	Toggles internal debug options	119
LOCSTat	Displays FTP status information for the local host	160
NOop	Checks whether the foreign host is still responding	177
STAtus	Displays status information for the foreign host	214
SYstem	Displays the name of the foreign host's operating system	231
TSO	Passes a TSO command to the local host TSO environment	234

Working with Directories on the Remote Host

To obtain directory information on the remote host, use the subcommands listed in Table 8.

Table 8. FTP Subcommands for Working with Directories on the Remote Host

Subcommand	Description	Page
CD	Changes the working directory	112
CDUP	Changes to the parent of the current working directory	115
CWd	Changes the working directory (Synonymous with CD)	112
DIr	Lists the directory entries for files on the foreign host	124
LS	Lists the names of files on the foreign host	164
MKdir	Creates a directory on the foreign host	172
PWd	Displays the name of the active working directory on the foreign host	185
RMDir	Removes a directory on the foreign host	192

Examples of the CD Subcommand

This example shows how to change and choose remote working directories and how the FTP server enables you to switch between the MVS and Hierarchical File

System (HFS) environments. For more information on how to change the directory levels, see the information on the CD, CDUP, and LCD subcommands later in this chapter.

```
  User:  ftp 9.67.113.24 621
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 9.67.113.24, port 621
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will not timeout.
           USER(identify yourself to the host):
           NAME (<host>:tsouserid):
  User:  user121
System:
           >>>USER user121
           331 Send password please.
           Password:

           >>>PASS ********
           230 USER121 is logged on.  Working directory is "/u/user121".
           Command:
  User:  cd tcpip
System:
           >>>CWD tcpip
           250 HFS directory /u/user121/tcpip is the current working directory
           Command:
  User:  cd ..
System:
           >>>CWD ..
           250 HFS directory /u/user121 is the current working directory
           Command:
  User:  cd 'user121'
System:
           250 "'user121'" is working directory name prefix.
           Command:
```

Examples Showing the Differences between DIR and LS Output for HFS Directories

The examples in this section use the following Internet addresses:

MVSXA2: 9.67.113.25

MVSXA3: 9.67.113.24

The current host is MVSXA2 (9.67.113.25). An FTP command is issued from 9.67.113.25 to 9.67.113.24.

```
  User:  ftp 9.67.113.24
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 9.67.113.24, port 621
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will not timeout.
           NAME (<host>:tsouserid):
  User:  user121
System:
           >>>USER user121
           331 Send password please.
           Password:

           >>>PASS ********
           230 USER21 is logged on. Working directory is "/u/user121".
           Command:
```

```
  User:  dir
System:
           >>>PORT 9,67,112,25,4,25
           200 Port request OK.
           >>>LIST
           125 List started OK.
           total 2736
           drwxr-xr-x    2  USER121  SYS1        0  Nov 20  18:15  IBM
           -rwxr-xr-t    2  USER121  SYS1   389120  Feb  5  16:03  ftpdka
           -rwxr-xr-t    2  USER121  SYS1   962560  Feb  5  16:04  ftpsrvka
           -rw-r-----    1  USER121  SYS1    11648  Jan 20  14:30  g.s
           drwxr-x---    3  USER121  SYS1        0  Oct 21  17:50  msg
           -rw-r-----    1  USER121  SYS1     1458  Jan 10  19:25  s.k
           drwxr-x---    2  USER121  SYS1        0  Feb  6  15:59  tcpip
           drwxr-x---    2  USER121  SYS1        0  Feb  6  17:29  test
           250 List completed successfully.

           Command:
  User:  ls
System:
           >>>PORT 9,67,112,25,4,26
           200 Port request OK.
           >>>NLST
           125 List started OK.
           IBM
           ftpdka
           ftpsrvka
           g.s
           msg
           s.k
           tcpip
           test
           250 List completed successfully.
           Command:
```

DIR provides detailed information about the data sets under the remote working directory, while LS shows the data set names only.

Examples Showing the Differences between DIR and LS Output with DIRECTORYMode and DATASetmode for MVS

This section gives examples of issuing a DIR and LS command in both DIRECTORYMode and DATASetmode.

```
  User:  ftp 1.1.2.3
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 1.1.2.3, port 21
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will close if idle for more than 5 minutes.
           NAME (<host>:tsouserid):
  User:  mvsuser
System:
           >>>USER mvsuser
           331 Send password please.
           Password:
           >>>PASS ********
           230 MVSUSER is logged on.   Working directory is "/u/mvsuser"
           Command:
```

```
  User:  cd 'tcpv3'
System:
           >>>CWD 'tcpv3'
           257 "'TCPV3.'" is working directory name prefix.
           Command:
  User:  site directorymode
System:
           >>>SITE directorymode
           200 Site command was accepted
           Command:
  User:  dir
System:
           >>>PORT 1,1,2,2,4,39
           200 Port request OK.
           >>>LIST  125 List started OK.
           Volume Unit
       Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
           Pseudo Directory                                          ETC
           Pseudo Directory                                          FTP
           Pseudo Directory                                          HOSTS
           Pseudo Directory                                          NSMAIN
           Pseudo Directory                                          PROFILE
           Pseudo Directory                                          STANDARD
           Pseudo Directory                                          TCPIP
           Pseudo Directory                                          TCPIPL62
           Pseudo Directory                                          TELNET
           250 List completed successfully.

           Command:
```

```
  User:  site datasetmode
System:
           >>>SITE datasetmode
           200 Site command was accepted

           Command:
  User:  dir
System:
         >>>PORT 1,1,2,2,4,40
         200 Port request OK.
         >>>LIST
         125 List started OK.
         Volume Unit     Date  Ext Used Recfm Lrecl BlkSz Dsorg Dsname
         APCSPL 3380D  07/16/97  1    1  FB      80  8800  PS  ETC.RPC
         APCSPL 3380D  08/03/97  1    1  FB      80  3200  PS  ETC.SERVICES
         APCSPL 3380D  08/03/97  1    1  FB      80  3120  PS  FTP.DATA
         APCSPL 3380D  08/02/97  1    1  F      158   158  PS  HOSTS.ADDRINFO
         APCSPL 3380D  08/03/97  1    1  FB      80  3120  PS  HOSTS.LOCAL
         APCSPL 3380D  07/30/97  1    1  F       56    56  PS  HOSTS.SITEINFO
         APCSPL 3380D  07/15/97  1    1  FB      80  8800  PS  NSMAIN.CACHE
         APCSPL 3380D  07/28/97  1    1  FB      80  8800  PS  NSMAIN.DATA
         APCSPL 3380D  08/03/97  1    2  FB      80  3200  PS  PROFILE.TCPIP
         APCSPL 3380D  07/26/97  1    2  FB      80  3200  PS  PROFILE.TCPIP.XA2
         APCSPL 3380D  08/03/97  1    1  VB    5124  6160  PS  STANDARD.TCPKJBIN
         APCSPL 3380D  08/03/97  1   15  F      256   256  PS  STANDARD.TCPXLBIN
         APCSPL 3380D  08/03/97  1    1  FB      80  3120  PS  TCPIP.DATA
         APCSPL 3380D  06/29/97  1    2  FB      80  3200  PS  TCPIPL62.CONFIG
         APCSPL 3380D  07/29/97  1   15  F      256   256  PS  TELNET.TCPXLBIN
         250 List completed successfully.

         Command:
```

```
  User:  site directorymode
System:
           >>>SITE directorymode
           200 Site command was accepted

           Command:
  User:  ls
System:
           >>>PORT 1,1,2,2,4,41
          200 Port request OK.
           >>>NLST
           125 List started OK.
           ETC
           FTP
           HOSTS
           NSMAIN
           PROFILE
           STANDARD
           TCPIP
           TCPIPL62
           TELNET
           250 List completed successfully.

           Command:
```

```
  User:  site datasetmode
System:
           >>>SITE datasetmode
           200 Site command was accepted

           Command:
  User:  ls
System:
           >>>PORT 1,1,2,2,4,42
           200 Port request OK.
           >>>NLST
           125 List started OK.
           ETC.RPC
           ETC.SERVICES
           FTP.DATA
           HOSTS.ADDRINFO
           HOSTS.LOCAL
           HOSTS.SITEINFO
           NSMAIN.CACHE
           NSMAIN.DATA
           PROFILE.TCPIP
           PROFILE.TCPIP.XA2
           STANDARD.TCPKJBIN
           STANDARD.TCPXLBIN
           TCPIP.DATA
           TCPIPL62.CONFIG
           TELNET.TCPXLBIN
           250 List completed successfully.
           Command:
```

Working with Directories on the Local Host

To work with directories on the local host, use the subcommands listed in Table 9.

Table 9. FTP Subcommands for Working with Directories on the Local Host

Subcommand	Description	Page
LCd	Changes the current directory on the local host	143
LMkdir	Creates a PDS on the local host	145
LPwd	Displays the name of the active working directory on the local host	163

Your default working directory on the local host is set according to the environment in which the FTP client is invoked: $HOME in OS/390 UNIX, your MVS user ID in TSO.

The following examples show how to choose local working directories.

```
  User:  ftp 1.1.2.3
System:
            IBM FTP CS/390 V2R8   1997  314 01:11 UTC
            Connecting to 1.1.2.3, port 21
            220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
            220 Connection will close if idle for more than 5 minutes.
            NAME (<host>:tsouserid):
  User:  mvsuser
System:
            >>>USER mvsuser
            331 Send password please.
            Password:

            >>>PASS ********
            230 MVSUSER is logged on.  Working directory is "/u/mvsuser".

            Command:
```

```
  User:  lpwd
System:
            Local directory is MVSUSER.

            Command:
  User:  lcd tcpip
System:
            Local directory name set to MVSUSER.TCPIP.

            Command:
  User:  lpwd
System:
            Local directory is MVSUSER.TCPIP.

            Command:
  User:  lcd 'ftp.test'
System:
            Local directory name set to FTP.TEST.

            Command:
  User:  lpwd
System:
            Local directory is FTP.TEST.

            Command:
  User:  lcd ..
System:
            Local directory name set to FTP.

            Command:
  User:  lpwd
System:
            Local directory is FTP.
            Command:
```

Preparing the Environment

You can use the subcommands listed in Table 10 to prepare the environment before working with data.

Table 10. FTP Subcommands for Preparing the Environment

Subcommand	Description	Page
AScii	Sets the transfer type to ASCII	107
Binary	Sets the transfer type to IMAGE	110
BLock	Sets the data transfer mode to block mode	111
COMpress	Sets the data transfer mode to compressed mode	118
EBcdic	Sets the transfer type to EBCDIC	127
File	Sets the file structure to file	130
GLob	Toggles globbing (the expansion of metacharacters in file names) for the MDELETE, MGET, and MPUT subcommands	133
LOCSIte	Specifies information that is used by the local host to provide service specific to that host system	147
MOde	Specifies the mode or data format of the transfer	174
PROMpt	Toggles interactive prompting for MDELETE, MGET, and MPUT commands	180
QUOte	Sends an uninterpreted string of data	187

Table 10. FTP Subcommands for Preparing the Environment (continued)

Subcommand	Description	Page
RECord	Sets the file structure to record	189
SENDPort	Enables or disables automatic transmission of the FTP server PORT subcommand	195
SENDSite	Enables or disables automatic transmission of the SITE subcommand	196
SIte	Sends information to the foreign host using site-specific commands	197
STREam	Sets the data transfer mode to stream mode	228
STRucture	Sets the file transfer structure	229
SUnique	Toggles the storage methods	230
TYpe	Specifies the transfer type	235

Transferring Data

You can use the subcommands listed in Table 11 to work with and transfer data.

Table 11. FTP Subcommands for Transferring Data

Subcommand	Description	Page
APpend	Appends a data set on your local host to a file on the foreign host	105
DELEte	Deletes a single file on the foreign host	122
DELImit	Displays the delimiter character between the *file_name* and *file_type*	123
Get	Copies a file from the foreign host to your local host	131
MDelete	Deletes multiple files on the foreign host	166
MGet	Copies multiple files from the foreign host to your local host	168
MPut	Copies multiple files on your local host to the foreign host	175
PUt	Copies a file on your local host to the foreign host	183
REName	Renames a file on the foreign host	190
REStart	Restarts a checkpointed data transfer	191

How to Transfer Data

TCP/IP supports only the data transfer of a data set or file structured as a continuous sequence of data bytes. This ensures that the correct record format is preserved across MVS hosts. Information could be lost or altered during transmission if you use an incorrect transfer.

Table 12 shows how to set the transmission attributes for different host systems. For example, VM or MVS host systems use EBCDIC for internal character representation. A text file of ASCII data type contains displayable characters; a carriage return (X'0D') and line feed (X'0A') are used to delimit a line. A text file of EBCDIC data type contains displayable characters; the newline character (X'15') is used to delimit a line. A binary file contains a contiguous stream of bits with no line delimiters.

Table 12. Recommended Methods for Data Transfer

Transfer Between Host Types	Data Transfer Type	Data Transfer Mode
EBCDIC and EBCDIC — DBCS text data	IBMKANJI (EBCDIC)	Stream
EBCDIC and EBCDIC — text data	EBCDIC	Stream
EBCDIC and EBCDIC — DBCS binary data	IBMKANJI (EBCDIC)	Block
EBCDIC and EBCDIC — binary data	EBCDIC	Block
EBCDIC and ASCII — DBCS text data	SJISKANJI, EUCKANJI, JIS78KJ, JIS83KJ, HANGEUL, KSC5601, TCHINESE, BIG5, SCHINESE (ASCII)	Stream
ASCII and EBCDIC — text data	ASCII	Stream
ASCII and EBCDIC — DBCS binary data	Image (binary)	Stream
ASCII and EBCDIC — binary data	Image (binary)	Stream
ASCII-to-EBCDIC-to-ASCII — all data **Note:** The EBCDIC host is used for storage only. Data remains encoded in ASCII, therefore, the data cannot be used on the EBCDIC host.	Image (binary)	Stream

For more information about the DBCS data type keywords and examples, see “FTP with DBCS Support” on page 97.

For information about setting data transfer type, see “TYPE Subcommand—Set the Data Transfer Type” on page 235. For information about setting data transfer mode, see “MODE Subcommand—Set the Data Transfer Mode” on page 174.

Examples of GET and MGET Subcommands

GET and MGET enable you to obtain files from a remote host and send them to the local host. In this example, FTP subcommands are issued from MVSXA2 to MVSVIC03. See Table 11 on page 51 for other subcommands useful for working with and transferring data.

The following members exist in the data set USER121.FTP.EXAMPLE on MVSVIC03:

FILE1
FILE2
FILE3
FILE4
FILE5

The following is displayed when entering the FTP environment:

```
  User:  ftp 9.67.113.24 621
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 9.67.113.24, port 621
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will not timeout.
           NAME (<host>:tsouserid):
  User:  user121
System:
           >>>USER user121
           331 Send password please.
           Password:
           >>>PASS ********
           230 USER121 is logged on.  Working directory is "/u/user121".
           Command:
```

```
  User:  get 'user121.ftp.example(file1)' 'user121.ftp.example(file1)'
System:
           'USER121.FTP.EXAMPLE(FILE1)' IS AN non-EXISTENT PARTITIONED DATASET.
           USE LMKDIR TO CREATE IT. LOCAL FILE NOT FOUND
           COMMAND:
  User:  lmkdir 'user121.ftp.example'
System:
           USER121.FTP.EXAMPLE CREATED.
           COMMAND:
  User:  get 'user121.ftp.example(file1)' 'user121.ftp.example(file1)'
System:
           >>>PORT 9,67,112,25,4,9
           200 Port request OK.
           >>>RETR 'USER121.ftp.example(file1)'
           125 Sending data set USER121.FTP.EXAMPLE(FILE1) FIXrecfm 128
           250 Transfer completed successfully.
           3464 bytes transferred in 0.754 seconds. Transfer rate 4.59 Kbytes/sec.
           Command:
  User:  get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)'
  System:
           >>>PORT 9,67,112,25,4,34
           200 Port request OK.
           >>>RETR 'USER121.ftp.example(file2)'
           125 Sending data set USER121.FTP.EXAMPLE(FILE2) FIXrecfm 128
           250 Transfer completed successfully.
           3464 bytes transferred in 1.483 seconds. Transfer rate 2.34 Kbytes/sec.
           Command:
```

```
  User:  get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)'
System:
           Data set 'USER121.FTP.EXAMPLE(FILE2)' was not replaced.
           Local file already exists
           To replace it, use command with the (REPLACE option
           Command:
  User:  get 'user121.ftp.example(file2)' 'user121.ftp.example(file2)' (replace
System:
           >>>PORT 9,67,112,25,4,35
           200 Port request OK.
           >>>RETR 'user121.ftp.example(file2)'
           125 Sending data set USER121.FTP.EXAMPLE(FILE2)
           250 Transfer completed successfully.
           3464 bytes transferred in 0.767 seconds. Transfer rate 0.50 Kbytes/sec.
           Command:
  User:  lpwd
System:
           Local directory is USER121
           COMMAND:
  User:  mget 'user121.ftp.example(file3)' 'user121.ftp.example(file4)'
System:
           >>>PORT 9,67,112,25,4,10
           200 Port request OK.
           >>>NLST 'user121.ftp.example(file3)'
           125 List started OK.
           250 List completed successfully.
           >>>PORT 9,67,112,25,4,11
           200 Port request OK.
           >>>NLST 'user121.ftp.example(file4)'
           125 List started OK.
           250 List completed successfully.
           >>>PORT 9,67,112,25,4,12
           200 Port request OK.
           >>>RETR 'USER121.FTP.EXAMPLE(FILE3)'
           125 Sending data set USER121.FTP.EXAMPLE(FILE3)
           250 Transfer completed successfully.
           3993 bytes transferred in 0.745 seconds. Transfer rate 0.51 Kbytes/sec.
           >>>PORT 9,67,112,25,4,13
           200 Port request OK.
           >>>RETR 'USER121.FTP.EXAMPLE(FILE4)'
           125 Sending data set USER121.FTP.EXAMPLE(FILE4)
           250 Transfer completed successfully.
           7367 bytes transferred in 0.818 seconds. Transfer rate 9.01 Kbytes/sec.
           Command:
```

```
  User:  lpwd
System:
           Local directory is USER121.
           Command:
  User:  cd 'user121.ftp.example'
System:
           >>>CWD 'user121.ftp.example'
           250 "USER121.FTP.EXAMPLE" partitioned data set is working directory.
           Command:
  User:  pwd
System:
           >>>PWD
           257 "USER121.FTP.EXAMPLE" partitioned data set is working directory.
           Command:
  User:  mget file3 file4
System:
           >>>PORT 9,67,112,25,4,20
           200 Port request OK.
           >>>NLST file3
           125 List started OK.
           250 List completed successfully.
           >>>PORT 9,67,112,25,4,21
           200 Port request OK.
           >>>NLST file4
           125 List started OK.
           250 List completed successfully.
           >>>PORT 9,67,112,25,4,22
           200 Port request OK.
           >>>RETR FILE3
           125 Sending data set USER121.FTP.EXAMPLE(FILE3)
           250 Transfer completed successfully.
           3993 bytes transferred in 0.549 seconds. Transfer rate 0.46 Kbytes/sec.
           >>>PORT 9,67,112,25,4,23
           200 Port request OK.
           >>>RETR FILE4
           125 Sending data set USER121.FTP.EXAMPLE(FILE4)
           250 Transfer completed successfully.
           7367 bytes transferred in 0.936 seconds. Transfer rate 0.23 Kbytes/sec.
           Command:
  User:  quit
System:
           >>>QUIT
           221 Quit command received.  Goodbye.
           READY
```

```
  User:  ftp 9.67.113.24 621
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           Connecting to 9.67.113.24, port 621
           220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
           220 Connection will not timeout.
           NAME (<host>:tsouserid):
  User:  user121
System:
           >>>USER user121
           331 Send password please.
           Password:
           >>>PASS ********
           230 USER121 is logged on.  Working directory is "/u/user121".
           Command:
```

```
  User:  get '/u/user121/ftp.example/file1' 'user121.ftp.example(file1)'
System:
           >>>PORT 9,67,112,25,4,24
           200 Port request OK.
           >>>RETR '/u/user121/ftp.example/file1'
           125 Sending data set /u/user121/ftp.example/file1
           250 Transfer completed successfully.
           3464 bytes transferred in 1.391 seconds. Transfer rate 2.49 Kbytes/sec.
           Command:
  User:  lcd 'user121.ftp.example'
System:    Local directory name set to partitioned data set USER121.FTP.EXAMPLE.
           Command:
  User:  lpwd
System:    Local directory is partitioned data set USER121.FTP.EXAMPLE.
           Command:
  User:  cd '/u/user121/ftp.example'
System:    >>>CWD '/u/user121.ftp.example'
           250 HFS directory /u/user121/ftp.example is the current working
               directory
           Command:
  User:  pwd
System:    >>>PWD
           257 "/u/user121.ftp.example" is the HFS working directory.
           Command:
  User:  get file1
System:    >>>PORT 9,67,112,25,4,26
           200 Port request OK.
           >>>RETR file1
           125 Sending data set /u/user121/ftp.example/file1
           250 Transfer completed successfully.
           3464 bytes transferred in 1.059 seconds. Transfer rate 3.27 kbytes/sec.
           Command:
```

```
  User:  mget '/u/user121/ftp.example/file4' '/u/user121/ftp.example/file5'
System:
            >>>PORT 9,67,112,25,4,33
            200 Port request OK.
            >>>NLST '/u/user121/ftp.example/file4'
            125 List started OK
            250 List completed successfully.
            >>>PORT 9,67,112,25,4,34
            200 Port request OK.
            >>>NLST '/u/user121/ftp.example/file5'
            125 List started OK
            250 List completed successfully.
            >>>PORT 9,67,112,25,4,35
            200 Port request OK.
            >>>RETR /u/user121/ftp.example/file4
            125 Sending data set /u/user121/ftp.example/file4
            250 Transfer completed successfully.
            7367 bytes transferred in 1.324 seconds. Transfer rate 5.56
               kbytes/sec.
            200 Port request OK.
            >>>RETR /u/user121/ftp.example/file5
            125 Sending data set /u/user121/ftp.example/file5
            250 Transfer completed successfully.
            3464 bytes transferred in 0.951 seconds. Transfer rate 3.64
               kbytes/sec.
            Command:
```

The data set USER121.FTP.EXAMPLE on MVSXA2 now contains the following members:

FILE1

FILE2

FILE3

FILE4

FILE5

Notes:

1. You do not have a choice of names for the local file as a result of the MGET command.
2. The MGET command is not applicable for generation data groups (GDGs).

Examples of PUT and MPUT Subcommands

PUT and MPUT enable you to send files from a local host to a remote host. In this example, FTP subcommands are issued from MVSXA2 to MVSVIC03. The data set USER121.FTP.EXAMPLE on MVSXA2 contains the following members:

APPEND01

XA2FILE1

XA2FILE2

XA2FILE3

The data set USER121.FTP.EXAMPLE on MVSVIC03 contains the following members:

XA3FILE1

XA3FILE2

XA3FILE3

The following is displayed when entering the FTP environment:

```
  User:   ftp 1.1.2.3
System:     IBM FTP CS/390 V2R8   1997  314 01:11 UTC
            Connecting to 1.1.2.3, port 21
            220-FTPD1 IBM FTP CS/390 V2R8 AT MVS164, 20:12:38 ON 1997-12-02.
            220 Connection will close if idle for more than 5 minutes.
            NAME (<host>:tsouserid):
  User:   user121
System:     >>>USER user121
            331 Send password please.
            Password:
            >>>PASS ********
            230 user121 is logged on.  Working directory is "/u/user121"
            Command:
```

```
   User:  put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
 System:    >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
            200 Site command was accepted
            >>>PORT 1,1,2,2,4,48
            200 Port request OK.
            >>>STOR 'user121.ftp.example(f1from2)'
            125 Storing data set USER121.FTP.EXAMPLE(F1FROM2)
            250 Transfer completed successfully.
            390 bytes transferred in 1.117 seconds.
            Transfer rate 0.35 Kbytes/sec.
            Command:
  User:   put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
System:     >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
            200 Site command was accepted
            >>>PORT 1,1,2,2,4,49
            200 Port request OK.
            >>>STOR 'user121.ftp.example(f1from2)'
            125 Storing data set USER121.FTP.EXAMPLE(F1FROM2)
            250 Transfer completed successfully.
            390 bytes transferred in 0.680 seconds.
            Transfer rate 0.57 Kbytes/sec.
            Command:
```

```
  User:   sunique
System:     Store unique is ON
            Command:
  User:   put 'user121.ftp.example(xa2file1)' 'user121.ftp.example(f1from2)'
System:     >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
            200 Site command was accepted
            >>>PORT 1,1,2,2,4,50
            200 Port request OK.
            >>>STOU 'user121.ftp.example(f1from2)'
            125 Storing data set USER121.FTP.EXAMPLE(F1FROM21) ( unique name )
            250 Transfer completed successfully.
            390 bytes transferred in 1.085 seconds.
            Transfer rate 0.36 Kbytes/sec.
            Command:
  User:   sunique
System:     Store unique is OFF
            Command:
  User:   cd 'user121.ftp.example.'
System:     >>>CWD 'user121.ftp.example.'
            257 "'USER121.FTP.EXAMPLE.'" is working directory name prefix.
            Command:
  User:   lpwd
System:     Local directory is USER121.
            Command:
  User:   lcd 'user121.ftp.example'
System:     Local directory name set to PDS USER121.FTP.EXAMPLE.
            Command:
  User:   lpwd
System:     Local directory is partitioned data set USER121.FTP.EXAMPLE.
            Command:
```

```
  User:   mput xa2file2 xa2file3
System:     >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
            200 Site command was accepted
            >>>PORT 1,1,2,2,4,51
            200 Port request OK.
            >>>STOR XA2FILE2
            125 Storing data set USER121.FTP.EXAMPLE.XA2FILE2
            250 Transfer completed successfully.
            390 bytes transferred in 1.437 seconds.
            Transfer rate 0.27 Kbytes/sec.
            >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
            200 Site command was accepted
            >>>PORT 1,1,2,2,4,52
            200 Port request OK.
            >>>STOR XA2FILE3
            125 Storing data set USER121.FTP.EXAMPLE.XA2FILE3
            250 Transfer completed successfully.
            390 bytes transferred in 1.091 seconds.
            Transfer rate 0.36 Kbytes/sec.
            Command:
  User:   quit
System:     >>>QUIT
            221 Quit command received. Goodbye.
            READY
```

The data set USER121.FTP.EXAMPLE on MVSVIC03 now contains the following members:

F1FROM2

F1FROM21

XA3FILE1

XA3FILE2

XA3FILE3

MVSVIC03 now also has the following data sets:

USER121.FTP.EXAMPLE.XA2FILE2

USER121.FTP.EXAMPLE.XA2FILE3

Note: The MPUT command is not applicable for generation data groups (GDGs).

Changing Local Site Defaults Using FTP.DATA

The default values for the local site parameters are hard-coded in the FTP client module. You can change these default values by creating an FTP.DATA configuration data set.

Note: Unless otherwise indicated, FTP.DATA data set refers to both the /etc/ftp.data HFS file and the MVS data set FTP.DATA.

The FTP.DATA configuration data set is optional. As shown in Table 13, the FTP client uses one of the following search orders to obtain the local site parameter values:

Table 13. FTP Client Search Orders

TSO shell	OS/390 UNIX shell
1. SYSFTPD DD statement	1. $HOME/ftp.data
2. tso_prefix.FTP.DATA	2. userid.FTP.DATA
3. userid.FTP.DATA	3. /etc/ftp.data
4. /etc/ftp.data	4. SYS1.TCPPARMS(FTPDATA) data set
5. SYS1.TCPPARMS(FTPDATA) data set	5. tcpip_hlq.FTP.DATA file
6. tcpip_hlq.FTP.DATA file	

Table 14 describes the statements you can code in the FTP.DATA data set. It is not necessary to include all statements in the FTP.DATA data set, only those statements whose default values are to be changed. The hard-coded default is used for any statement not included in the FTP.DATA data set.

You can change several of the FTP local site parameters during the FTP session by using the LOCSITE subcommand. See “LOCSITE Subcommand—Specify Site Information to the Local Host” on page 147 for more information about using the LOCSITE subcommand to change the local site parameters.

You can also use the FTP.DATA data set to override the default site parameters for the FTP server. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information on overriding FTP server default local site parameters.

Table 14. FTP.DATA Data Set Statements for the FTP Client

Parameter	Value	Description
ASAtrans	true	Permits the client to translate characters in the first column of ASA files being transferred as print control characters, if recognized as such.
	false	Treats ASA file transfers as regular file transfers; that is, the ASA characters in column 1 are treated as part of the data and are not converted to print control character sequences. This is the hard-coded default.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
AUTOMount	true	Permits automatic mounting of volumes for data sets on volumes that are not mounted. This is the hard-coded default.
	false	Prevents automatic mounting of volumes for data sets on volumes that are not mounted.
AUTOTAPEMount	true	Permits automatic mounting of tape volumes for data sets on volumes that are not mounted. This is the hard-coded default.
	false	Prevents automatic mounting of tape volumes for data sets on volumes that are not mounted.
AUTORecall	true	Permits automatic recall of migrated data sets. This is the hard-coded default.
	false	Prevents automatic recall of migrated data sets.
BLKsize	*size*	Specifies the block size of a newly allocated data set. If BLKsize is not specified at all, the hard-coded default is 6233. Valid range is 0–32760, where BLKsize 0 is a special case that indicates that the operating system is to determine the blocksize for new data sets. Specifying a size value is optional. Specify BLKsize with no size value to enable the block size from a model DCB data set or SMS dataclass to be used. See also the DATAClass and DCbdsn parameters in this table. BLKsize is functionally equivalent to BLOCKSIze. The BLOCKSIze parameter is obsolete but it is accepted to provide compatibility with previous releases.
BLOCKSIze	*size*	Specifies the block size of a newly allocated data set. The hard-coded default is 6233. Valid range is 0–32760. BLOCKSIze is functionally equivalent to BLKsize. The BLOCKSIze parameter is obsolete but it is accepted to provide compatibility with previous releases.
BUfno	*number*	Specifies the number of access method buffers to be used when data is read from or written to a data set. Valid values are in the range 1–255. The hard-coded default is 5.
CCONNTIME	*seconds*	Defines the amount of time to wait after attempting to close a control connection before terminating it and reporting an error. The default is 30. The valid range for CCONNTIME is 0 (CCONNTIME not used) or 15-86400.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
CCTRANS	*dsname*	Specifies the SBCS translation table to be used for the control connection. FTP uses the translation table in the *user_id.dsname.*TCPXLBIN data set. If that data set does not exist, FTP uses the *hlq.dsname.*TCPXLBIN data set. FTP resolves the translation table to be used for the control connection by: 1. Data set specified in FTP subcommand with TRANSLATE option a. $HOME/*dsname*.TCPXLBIN (OS/390 UNIX environment only) b. *user_id.dsname*.TCPXLBIN c. *hlq.dsname*.TCPXLBIN 2. Data set specified in CTRLConn parameter in FTP.DATA data set 3. Data set specified in CCTRANS parameter in FTP.DATA data set 4. Default search order a. *user_id*.FTP.TCPXLBIN b. *hlq*.FTP.TCPXLBIN c. *user_id*.STANDARD.TCPXLBIN d. *hlq*.STANDARD.TCPXLBIN 5. 7-bit table build 6. FTP internal tables
CHKPTInt	*interval*	Specifies the checkpoint interval for the sending site in a file transfer request. This value is used to determine when checkpoint marker blocks are to be transmitted. The actual frequency of marker blocks is a function of the checkpoint interval value and the size of the buffers being used. If the checkpoint interval is 0, no checkpointing occurs and no marker blocks are transmitted. The default value is 0. **Note:** Use this parameter only if you are sending to a server that supports checkpointing.
CLIENTERRCODES	true	Convert FTP return codes into a set of codes defined in “FTP Client Error Codes” on page 83.
	false	Convert an FTP return code to the remainder of dividing the return code by 4096 to ensure a code in the range of 0-4095. This is the default. If FTP is called from a REXX exec, the return code will be the 5 digit xxyyy format, where xx is the subcommand code and yyy is the reply from the server.
CONDdisp	CATLG	Specifies that a data set is kept and cataloged when an FTP file transfer ends prematurely. This is the hard-coded default.
	DELETE	Specifies that a data set is deleted when a file transfer ends prematurely.
CTRLConn	7BIT	Indicates 7-bit ASCII is to be used on the control connection. This is the default if no TCPXLBIN data set is found.
	iconv_ascii	Indicates a name recognized by iconv to indicate the ASCII code page to be used for the control connection.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
DATAClass	*Class*	Specifies the data class for SMS created files as defined by your organization for the FTP client. To cancel the specification, use LOCSITE with the DATAC= parameter. See "Specifying Values for New Data Sets" on page 68 for more information about specifying attributes when allocating new data sets.
DATACTTIME	*seconds*	Defines the amount of time to wait after attempting to send or receive data before terminating the connection and reporting an error to the user. The default is 120. The valid range for DATACTTIME is 0 (DATACTTIME not used) or 15-86400.
DATATIMEOUT	*seconds*	Defines the amount of time to wait after attempting to send or receive data before terminating the connection and reporting an error to the user. The default is 1300. The valid range for DATATIMEOUT is 0 (DATATIMEOUT not used) to 86400. If coded with value greater than 86400, the timer is set to 86400.
DB2®	*name*	Specifies the name of the DB2 subsystem. The default name is DB2.
DB2PLAN	*planname*	Specifies the DB2 plan name to be opened by the FTP client. This name should match the name specified in the BIND job for the EZAFTPMQ DBRM used by FTP. If DB2PLAN is not present in FTP.DATA, the default plan name of EZAFTPMQ is used by the FTP client.
DCbdsn	*name*	Specifies the name of the MVS data set to be used as a model for allocation of new data sets. HFS file names are not allowed. There is no hard-coded default. **Note:** See "Generation Data Group Support" on page 70 for information about the relationship between DCbdsns and GDGs in the CFTP and OS/390 UNIX-FTP servers.
DCONNTIME	*seconds*	Defines the amount of time to wait after attempting to close a data transfer before terminating the connection and reporting an error. The default is 120. The valid range for DCONNTIME is 0 (DCONNTIME not used) or 15-86400.
Directory	*size*	Specifies the number of directory blocks to be allocated for the directory of a PDS. The hard-coded default is 27.
DIRECTORYMode	true	Specifies that only the data set qualifier immediately below the current directory is treated as an entry in the directory. In directory mode, this qualifier is the only one used by the MPUT, MGET, LS, and DIR subcommands.
	false	Specifies that all the data set qualifiers below the current directory are treated as entries in the directory. This is the hard-coded default.
FILEtype	*type*	Specifies the file type of the data set. The hard-coded default is SEQ. The description of each file type is: **Type** **Description** **SEQ** Sequential or partitioned data sets **SQL** SQL query function

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
INACTTIME	*seconds*	Specifies the amount of time to wait for an expected response from the server, on either the control or the data connection, before closing the session. Data transfer times that exceed this value will not cause session termination unless the time between data packet arrivals exceeds this value. The hard-coded default is 300. The valid range for INACTTIME is 0 (INACTTIME not used) or 15-86400.
LRecl	*length*	Specifies the record length of a newly allocated data set. Valid values are 0–32756. The default is 256. **Note:** You can also use a value of x to indicate the maximum possible value.
MGmtclass	*class*	Specifies the SMS management class to be assigned to newly allocated data sets. To cancel the specification, use LOCSITE subcommand with the MG= parameter. See "Specifying Values for New Data Sets" on page 68 for more information about specifying attributes when allocating new data sets.
MIGratevol	*volid*	Indicates the volume ID (volid) for migrated data sets using non-IBM storage management systems. If you do not specify MIGratevol, the default volid is MIGRAT.
MYOPENTIME	*seconds*	Defines the amount of time to wait for a session to open before terminating the attempt and reporting an error. The default is 60. The valid range for MYOPENTIME is 0 (MYOPENTIME not used) or 15-86400.
PRImary	*value*	Specifies the amount of direct access storage for primary allocation of new data sets. The hard-coded default is 1.
QUOtesoverride	true	Indicates that single quotes at the beginning of, or surrounding, a file name are interpreted to mean that the file name contained inside the single quotes should override the current working directory instead of being appended to the current working directory. Any single quotes inside the file name are treated as part of the file name. This is the hard-coded default.
	false	Specifies that a single quote at the beginning of a file name, as well as all other single quotes contained in the file name, is treated as part of the actual file name. The entire file name, including the leading single quote, is appended to the current working directory.
RDW	true	Specifies that RDWs are treated as if they were part of the record and not discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only. **Note:** RDW information is stored in a binary halfword. You should transfer files in binary mode to avoid translation problems that can occur if you transfer this binary field in EBCDIC or ASCII.
	false	Specifies that RDWs are discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only. This is the hard-coded default.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
RECfm	*format*	Specifies the record format of a data set. Valid record formats are: F, FA, FB, FBA, FBM, FBSA, FBSM, FM, FS, FSA, FSM, U, UA, UM, V, VA, VB, VBA, VBM, VBS, VBSA, VBSM, VM, VS, VSA, and VSM. The default is VB. The characters used to specify these record formats have the following meanings: **Format** **Description** **F** Fixed record length **V** Variable record length **U** Undefined record length **B** Blocked records **S** Spanned records (if variable) / Standard (if fixed) **A** Records contain ISO/ANSI control characters **M** Records contain machine code control characters
RESTGet	true	The checkpoint data set will be opened for a GET request. This is the default.
	false	The checkpoint data set will not be opened for a GET request.
RETpd	*days*	Specifies the number of days that a newly allocated data set should be retained. The default is 0, which means no retention period. If you do not specify a retention period, no retention period is assigned to newly allocated data sets. The maximum is 9999.
SBDataconn	*dsname*	Specifies the fully-qualified data set name or HFS file name that contains the EBCDIC-to-ASCII and ASCII-to-EBCDIC translate tables that are generated by the CONVXLAT utility.
	ebcdic_cp, *ascii_cp*	Specifies the names of EBCDIC and ASCII code pages that are recognized by iconv.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
SBTRANS	*dsname*	Specifies the SBCS translation table to be used for the data connection. This table is used for SBCS and DBCS data transfers. FTP uses the translation table in the *user_id.dsname.*TCPXLBIN data set. If the *user_id.dsname.*TCPXLBIN data set does not exist, FTP uses the *hlq.dsname.*TCPXLBIN data set. FTP resolves the translation table to be used for the data connection by: 1. Data set specified in FTP subcommand with TRANSLATE option a. $HOME/*dsname*.TCPXLBIN (OS/390 UNIX environment only) b. *user_id.dsname*.TCPXLBIN c. *hlq.dsname*.TCPXLBIN 2. Data set specified in SBDataconn parameter in FTP.DATA data set 3. Data set specified in SBTRANS parameter in FTP.DATA data set 4. Default search order a. *user_id*.FTP.TCPXLBIN b. *hlq*.FTP.TCPXLBIN c. *user_id*.STANDARD.TCPXLBIN d. *hlq*.STANDARD.TCPXLBIN 5. 7-bit table build 6. FTP internal tables
SECondary	*value*	Specifies the amount of direct access storage for secondary allocation of new data sets. The hard-coded default is 1.
SPACETYPE	*type*	Specifies whether newly allocated data sets are allocated in blocks, cylinders, or tracks. Valid values for *type* are BLOCK, CYLINDER, or TRACK. The hard-coded default is TRACK.
SPRead	true	Specifies that the output is in spreadsheet format when the file type is SQL.
	false	Specifies that the output is in report format rather than spreadsheet format when the file type is SQL. This is the hard-coded default.
SQLCol	*type*	Specifies what the column headings of the SQL output file are. The following list describes each type: **Type** **Description** **Names** Database column names are used for the column headings. The labels are ignored. This is the hard-coded default. **Labels** Labels are the DB2 SQL table column headings. If any of the columns do not have labels, FTP supplies a column heading in the form of `COLnnn`. **Any** The database column heading is the first choice for column heading. If there is no label, the database column name becomes the column heading.

Table 14. FTP.DATA Data Set Statements for the FTP Client (continued)

Parameter	Value	Description
STORclass	*class*	Specifies the storage class for the SMS-created files as defined by your organization for the FTP client. To cancel the specification, use LOCSITE with the STO= parameter. See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.
TRAILingblanks	true	Specifies that FTP preserves the trailing blanks that are in a fixed format data set when the data is transferred.
	false	Specifies that FTP does not preserve the trailing blanks that are in a fixed format data set when the data is transferred. This is the hard-coded default.
UCSHOSTCS	*code_set*	Specifies the EBCDIC code set to be used when converting to/from Unicode. If you do not specify a *code_set*, the current code set is used.
UCSSUB	true	In Unicode-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any Unicode character that cannot successfully be converted. Data transfer continues.
	false	In Unicode-to-EBCDIC conversion, the data transfer is terminated if any Unicode character cannot be converted into the EBCDIC code set. This is the hard-coded default.
UCSTRUNC	true	In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is allowed. The data transfer continues even if EBCDIC data is truncated. Warning: If data contains any double-byte characters, there is no guarantee that truncation will honor character boundaries or that records will end in shift-in state.
	false	In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is not allowed. The data transfer is aborted if the logical record length of the receiving data set is too small to contain the data after conversion to EBCDIC. This is the hard-coded default. **Note:** The setting of the CONDdisp statement determines what happens to the target data set if the transfer is aborted.
UMask	*octal_umask*	Defines the file mode creation mask that specifies the permission bits that must be OFF when a new file is created. The default is 027. **Note:** You cannot use FTP to create HFS files with execute permissions. If you require execute permissions, use the SITE CHMOD command after the file has been created.
UNITNAME	*type*	Specifies the unit *type* (for example, 3380) for allocation of new data sets on direct access devices. If UNITNAME is not specified, the unit type used for allocation is the system default.
VOLume	*name*	Specifies the volume serial number for allocation of new data sets. The value specified for *name* is case-sensitive. If VOLume is not specified, the volume serial number used for allocation is the system default.
WRAPrecord	true	Indicates that data is wrapped to the next record if there is no new line character.
	false	Indicates that data is truncated if there is no new line character. This is the default.

Sample FTP.DATA Data Set (FTCDATA)

The following is a sample of the contents of the FTP.DATA data set in the FTCDATA member of the *hlq*.SEZAINST data set.

```
;*********************************************************************
;                                                                    *
;   Name of File:            tcpip.SEZAINST(FTCDATA)                 *
;                                                                    *
;   Descriptive Name:         FTP.DATA  (for FTP Client)             *
;                                                                    *
;   SMP/E Distribution Name:  EZAFTPAC                               *
;                                                                    *
;   Licensed Materials - Property of IBM                             *
;   "Restricted Materials of IBM"                                    *
;   5647-A01                                                         *
;   (C) Copyright IBM Corp. 1997, 1999                               *
;   Status = CSV2R8                                                  *
;                                                                    *
;   This FTP.DATA file is used to specify default file and disk      *
;   parameters used by the FTP client.                               *
;                                                                    *
;   Note: For an example of an FTP.DATA file for the FTP server,     *
;   see the FTPSDATA example.                                        *
;                                                                    *
;   Syntax Rules for the FTP.DATA Configuration File:                *
;                                                                    *
;   (a) All characters to the right of and including a ; will be     *
;       treated as a comment.                                        *
;                                                                    *
;   (b) Blanks and  are used to delimit tokens.           *
;                                                                    *
;   (c) The format for each statement is:                            *
;                                                                    *
;       parameter value                                              *
;                                                                    *
;   (d) abbreviations are not permitted in the FTP.DATA file for     *
;       the FTP client.                                              *
;                                                                    *
;*********************************************************************
;
;
; File and disk parameters
;
AUTOMOUNT       TRUE       ; Nonmounted volumes mounted automatically
AUTORECALL      TRUE       ; Migrated HSM files recalled automatically
BLOCKSIZE       6144       ; Block size is 6144 bytes
;CTRLCONN        IBM-850    ; ascii code set for control connection
;DB2           D31             ; db2 subsystem name
;DB2PLAN       PLANNAME        ; db2 plan name for OE-FTP
;DCBDSN          MODEL.DCB  ; Data set name used as model for allocation
DIRECTORY       15         ; PDS allocated with 15 directory blocks
DIRECTORYMODE   FALSE      ; Use all qualifiers (Datasetmode)
FILETYPE        SEQ        ; File Type = SEQ (default)
LRECL           128        ; Logical record length is 128 bytes
;MGMTCLASS       TCPMGMT    ; SMS management class for new data sets
PRIMARY         5          ; Primary allocation is 5 tracks
RDW             FALSE      ; Do not retain RDWs as data
RECFM           FB         ; Fixed blocked record format
;RETPD           30         ; New data set expiration date is 30 days
;SBDATACONN   (IBM-1047,IBM-850) ; ebcdic/ascii code sets for data conn.
SECONDARY       2          ; Secondary allocation is 2 tracks
SPACETYPE       TRACK      ; Data sets allocated in tracks
;UNITNAME        SYSDA      ; Unit name used for allocation
;VOLUME          WRKLB2     ; Volume serial number for allocation
```

Using Different SBCS Translation Tables for the Control and Data Connections

If the SBCS table you need for SBCS or DBCS data transfer does not support the standard encodings for the portable character set, you should establish different translation tables for the control and data connections. This avoids unrecognized ASCII conversion of FTP commands.

You can specify different conversions for the control and data connections by:

- Using the CTRLConn or SBDataconn statements, or both, in your local FTP.DATA file
- Using the LOCSITE SBDataconn subcommand to change the conversion being used for the data connection

Specifying Values for New Data Sets

When allocating new data sets, there are several methods you can use to specify the data set attributes. You can individually use the storage attribute parameters with the SITE and LOCSITE commands or the *hlq*.FTP.DATA data set. Or, if your system administrator has used the Storage Management Subsystem to group together default attributes into named classes, you can specify those class names on the DATAClass, STORclass, and MGmtclass parameters.

Dynamic Allocation

FTP enables you to dynamically allocate a new physical sequential data set or a partitioned data set (PDS) for the purpose of transferring data to be written to that data set. The following optional allocation variables can be used by the client to override and turn off the hard-coded defaults that affect the allocation of the data set.

Variable	FTP.DATA parameter
allocation units	SPACETYPE
blocksize	BLKSIZE
data class	DATACLASS
directory blocks	DIRECTORY
logical record length	LRECL
management class	MGMTCLASS
model DCB values	DCBDSN
primary space	PRIMARY
record format	RECFM
retention period	RETPD
secondary space	SECONDARY
storage class	STORCLASS
unit	UNITNAME
volume serial number	VOLUME

Some of these allocation variables might provide duplicate information. For example, the model DCB may have a record format (RECfm) that differs from the

record format specified by a data class and from the one explicitly specified by the client. FTP passes all variables that are specified to dynamic allocation and lets it determine which of the specifications take precedence. The following list describes the exceptions to that policy:

- If neither the primary nor secondary space quantity is specified, the allocation units value is not sent.
- If the data set organization is physical sequential, directory blocks specification is not sent.
- Otherwise, all variables are sent to dynamic allocation where the order of precedence is:
  1. Any FTP.DATA, SITE, or LOCSITE parameters explicitly specified or defaulted
  2. Any attributes picked up from the model DCB and not otherwise explicitly specified
  3. Any attributes picked up from the data class and not previously derived from 1 or 2
  4. Any allocation defaults

Storage Management Subsystem (SMS)

An FTP client can specify one or more of the following Storage Management Subsystem (SMS) classes to manage characteristics that are associated with or assigned to data sets.

- Data class is an SMS construct that determines data set allocation attributes used by SMS for creation of data sets. The fields listed are available attributes that serve as a template for allocation. Each is **optional** and is overridden by any explicit specification of FTP allocation variables or by a model DCB (DCBDSN).

Variable	FTP.DATA parameter
directory blocks	DIRECTORY
logical record length	LRECL
primary space	PRIMARY
record format	RECFM
retention period	RETPD
secondary space	SECondary

  Note: If either primary or secondary space is explicitly specified, the primary and secondary values from data class are not used.

- Management class is an SMS construct that determines Data Facility Hierarchical Storage Manager (DFHSM) action for data set retention, migration, backup, and release of allocated but unused space. Management class replaces and expands attributes that otherwise would be specified. That is, management class may override any other specification of retention period.
- Storage class is a list of storage performance and availability services requests for an SMS-managed data set that SMS attempts to honor when selecting a volume or volumes for the data set. It may conflict with an explicit specification of volume and unit. If storage class is used, volume and unit parameters should be unspecified.

Using a DCBDSN Model to Create a New Data Set

To use a DCBDSN model to create a data set, do the following:

1. Issue the following command:

   ```
   SITE DCBDSN=data_set_name
   ```

   where *data_set_name* is the name of the data set to be used as a model to set the values of the logical record length (LRecl), the blocksize (BLKsize), the retention period (RETpd), and the record format (RECfm) of a new data set.
2. Issue the following command to enable the LRecl, BLKSIze, and RECfm of the model to be used:

   ```
   SITE LRECL BLKSIZE RETPD RECFM
   ```
3. Issue the following command to create the new data set with the values specified by the DCBDSN model:

   ```
   PUT data_set_name
   ```

   where *data_set_name* is the name of the new data set.

   Note: If you are using a non-MVS client that does not support the SITE command, you may be able to send the SITE command to the MVS server by using the QUOTE command. For example:

   ```
   QUOTE SITE DCBDSN=data_set_name
   ```

Generation Data Group Support

Generation data groups (GDGs) enable you to store multiple data sets, called generation data sets (GDSs) as versions of the GDG. You cannot use FTP to create a new GDG, but you can use it to create a new version (that is, a new GDS) or to transfer an existing version of an existing GDG.

The relationship between DCBDSN and GDGs is governed by MVS allocation rules rather than FTP usage rules. Therefore, when creating a new GDG [put 'sys1.proclib(jes2)' user77.mygdg(+1)], one of the following must be true:

- A valid MODEL or PATTERN DSCB (for FTP, DCBDSN) specification must be coded in the FTP.DATA file when the CFTP server is started.
- A valid SITE DCbdsn=*dataset_name* must be issued before a PUT command is issued.
- A data set having the same name as the GDG base must reside on the volume as the user catalog that contains the GDG definition. In this case, neither a SITE DCbdsn nor a DCBDSN argument in the FTP.DATA data file is required. Allocation detects that a GDG is being created and looks in the VTOC of the volume containing the USERCATALOG for a data set (uncataloged) that has the same name as the GDG BASE (see the sample GDG JCL on page 71).

  Notes:

  1. A model or pattern DSCB that is the same name as the GDG BASE cannot exist on an SMS managed volume. This is an SMS restriction and is documented in the DFP manuals pertaining to using data sets (generation data sets or generation data groups).
  2. Allocation does not generally have any requirements about the characteristics of a MODEL DSCB, (cannot be VSAM, must be on DASD). Most facilities create one model DSCB for the entire system and everyone uses that model. The system-wide model usually has no logical record length (LRecl), block

size (BLKsize), record format (RECfm), data set organization (DSORG) or retention period (RETpd) associated with it.

3. The existing CFTP and OS/390 UNIX-FTP servers require the MODEL DSCB to have a valid DSORG of physical sequential organization (PS). Otherwise the SITE command for the DCBDSN is ignored, and a message is issued indicating the DCBDSN was ignored.

The following restrictions apply:

- DCBDSN=USER.MYGDG(0)/ USER.MYGDG(-n), not supported
- DCBDSN=SYS1.PROCLIB(JES2) (specifying a member of a PDS, not valid)
- DCBDSN=SYS1.PROCLIB, valid
- The data set referenced on the DCBDSN, a DSORG of PS needed (FTP requirement)

Notes:

1. A DCBDSN is not required when issuing GETs to create a new GDG by way of the Pascal Client or issuing a PUT to create a new GDG to the Pascal Server.
2. If there are explicit values associated with LRecl, BLKsize, RECfm, or the SMS management equivalent parameters, these explicit parameters override the values associated with the model DSCB specified on the DCBDSN.

The following is a sample Job Control Language (JCL) to create a model and the GDG BASE:

```
USER77.MYGDG         -MODEL/PATTERN
VOL=SER=CPDLB1       -Volume having USERCATALOG, where USER77 is defined
(NAME(USER77.MYGDG)  -GDG BASE definition

//USER77X  JOB MSGLEVEL=(1,1),MSGCLASS=D,NOTIFY=USER77
//GDGA EXEC PGM=IDCAMS
//*
//GDGMOD   DD DSN=USER77.MYGDG,
//         VOL=SER=CPDLB1,
//            UNIT=SYSALLDA,
//            SPACE=(TRK,(0)),
//            DCB=(LRECL=80,RECFM=FB,BLKSIZE=6800,DSORG=PS),
//            DISP=(,KEEP)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEFINE GENERATIONDATAGROUP -
         (NAME(USER77.MYGDG)
         EMPTY
         NOSCRATCH
         LIMIT(255) )
```

GDG Examples

Before you specify a (+*nnn*) value to create a new GDS, issue the following command:

```
SITE DCBDSN=model
```

This subcommand specifies an MVS data set to be used as a model. The model must have a DSORG of PS. The other DCB characteristics of the data set are not checked.

Notes:

1. Failure to have a valid DCBDSN before trying to create a new GDS may cause FTP or ALLOCATION to fail or to return unpredictable results.

2. If you issue a SITE DCbdsn LRecl BLKsize command before the creation of a new data set, the LRecl and BLKsize parameters on the SITE command override the LRecl and BLKsize parameters on the DCbdsn command.

For more information about GDGs, refer to *MVS/DFP Version 3 Release 3: Using Data Sets*.

The following are sample FTP commands that access a GDG called JIMKEO.GDG.

Notes:

1. In the following examples, gdg (0), gdg (-1), and gdg (+1) specify which copy of the GDG you are using. 0 indicates the latest version, -1 indicates the previous version, and +1 indicates that a new version is created.
2. GDGALL is not supported by the C-FTP server. GDGALL processing occurs when the base name for the GDG is specified without a relative index value.
3. The MPUT and MGET commands are not applicable for GDGs.

The following example illustrates a PUT to the latest existing GDS. The working directory at the server is JIMKEO.

```
Command:
put my.gdg gdg(0)
>>>SITE FIXrecfm 150
200-Blocksize must be a multiple of lrecl for FB  data sets. Blocksize set to
6150.
200 Site command was accepted
>>>PORT 129,34,128,245,126,229
200 Port request OK.
>>>STOR gdg(0)
125 Storing data set JIMKEO.GDG.G0055V00
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 3.24 Kbytes/sec.
```

The following example illustrates a GET of the latest GDG:

```
Command:
get gdg(0) my.gdg2
>>>PORT 129,34,128,245,126,233
200 Port request OK.
>>>RETR gdg(0)
125 Sending data set JIMKEO.GDG.G0055V00 FIXrecfm 150
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 3.04 Kbytes/sec.
```

The following example illustrates a PUT to a new GDS [After the STOR is complete, this new version is referenced by (0)].

```
Command:
put my.gdg gdg(+1)
>>>SITE FIXrecfm 150
200 Site command was accepted
>>>PORT 129,34,128,245,126,234
200 Port request OK.
>>>STOR gdg(+1)
125 Storing data set JIMKEO.GDG.G0056V00
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 1.16 Kbytes/sec.
```

The following example illustrates a GET of the previous GDS into the local file called my.gdg3:

```
Command:
get gdg(-1) my.gdg3
>>>PORT 129,34,128,245,126,239
200 Port request OK.
>>>RETR gdg(-1)
125 Sending data set JIMKEO.GDG.G0055V00 FIXrecfm 150
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 2.77 Kbytes/sec.
```

The following example illustrates a GET that replaces the contents of my.gdg3 with the most recent GDS:

```
Command:
get gdg(0) my.gdg3 (replace
>>>PORT 129,34,128,245,126,243
200 Port request OK.
>>>RETR gdg(0)
125 Sending data set JIMKEO.GDG.G0056V00 FIXrecfm 150
250 Transfer completed successfully.
612 bytes transferred. Transfer rate 3.36 Kbytes/sec.
```

The following example illustrates changing the working directory:

```
Command:
cd gdg
>>>CWD gdg
257 "'JIMKEO.GDG.'" is working directory name prefix.
```

The following example shows the files created:

```
Command:
dir
>>>MODE s
200 Data transfer mode is Stream.
>>>PORT 129,34,128,245,127,12
200 Port request OK.
>>>LIST
125 List started OK.
Volume Unit    Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
STRG73 3380K   04/30/92   1    5 FB      150 32700  PS   G0003V00
STRG65 3380K   04/30/92   1    5 FB      150 32700  PS   G0006V00
STRG61 3380K   04/30/92   1    5 FB      150 32700  PS   G0010V00
STRG47 3380K   04/30/92   1    5 FB      150 32700  PS   G0015V00
STRG47 3380K   04/30/92   1    5 FB      150 32700  PS   G0021V00
STRG66 3380K   04/30/92   1    5 FB      150 32700  PS   G0028V00
STRG47 3380K   04/30/92   1    5 FB      150 32700  PS   G0036V00
STRG01 3380K   04/30/92   1    5 FB      150 32700  PS   G0045V00
STRG53 3380K   04/30/92   1    5 FB      150 32700  PS   G0055V00
STRG59 3380K   04/30/92   1    5 FB      150 32700  PS   G0056V00
250 List completed successfully.
>>>MODE b
200 Data transfer mode is Block.
Command:
```

Submitting FTP Requests in Batch

FTP is usually run interactively by starting and entering commands from your terminal. You can also run FTP as a batch job, but you must supply the job control language (JCL) file. You can use batch when you know what functions you want to perform, when you want a hard copy of the results, or when you want to perform an FTP function many times.

Notes:

1. FTP can be run in batch either by specifying data sets for input and output as in Figure 3 on page 75, or without referring to data sets for input and output as in Figure 5 on page 76.
2. The file containing the FTP commands cannot have sequence numbers in it. It must be saved *un*numbered. Input streams containing sequence numbers cause unpredictable results because the FTP client parses the job stream until the end of record (EOR).
3. A blank followed by a plus sign (+) at the end of an FTP subcommand line is used as a continuation indicator for all FTP subcommands except the QUOTE subcommand. When the continuation indicator is encountered at the end of an FTP subcommand line, the next line is appended to the subcommand. For example,

   ```
   PUT SOURCE.DS.NAME +
   DEST.DS.NAME
   ```

   is interpreted as:

   ```
   PUT SOURCE.DS.NAME DEST.DS.NAME
   ```
4. Do not attempt to use server parameters and options on this JCL, because it is acting as a client.
5. To have the FTP client perform DB/2 queries in a batch job, the DSNLOAD library must be in the link list or appear on a STEPLIB DD statement for the job.
6. Batch jobs require the *user_ID* to be case-sensitive. When connecting to a server such as a UNIX server where user IDs, passwords, directory names, and file names are case-sensitive, the data in the FTP batch job must be in the correct case.
7. Use the (EXIT parameter if you want FTP to display an error return code and then exit when certain errors are detected. See “FTP EXIT Return Codes” on page 79 for more information.

   Because MVS batch condition codes range from 0 to 4095 (X'FFF'), the step condition code displayed by the JCL will NOT match the exit return code that FTP displays in its message. What is displayed is the remainder of a division of the FTP exit return code by 4096. When viewed in hex, this is the same as the last three hex digits of the exit return code.

   For example, a PUT that fails and returns the message:

   ```
   "550       Open failed for ..."
   ```

   would get an exit return code of 27550. If you divide 27550 by 4096, the remainder of 2974 would be the JCL STEP condition code.

   It is easier to visualize this in hex, where decimal 27550 becomes hex 6B9E. The last three hex digits, B9E, is the decimal number 2974.

Figure 3 on page 75 shows an example of the JCL required to submit a batch job by referring to data sets for input and output.

```
//USER28F  JOB ,CARTER,MSGLEVEL=(1,1)
//FTPSTP1  EXEC PGM=FTP,REGION=2048K,
//              PARM='9.67.112.25 (EXIT TIMEOUT 20'
//NETRC    DD  DSN=ANYHLQ.NETRC,DISP=SHR
//SYSPRINT DD  SYSOUT=H
//OUTPUT   DD  SYSOUT=H
//INPUT    DD  *
type e
mode b
put idss.parts
/*
```

Figure 3. JCL to Run FTP in Batch Using Data Sets

The following apply to Figure 3:

Notes:

1. REGION=2048K is a minimum requirement. The requirement could increase depending on the block size of the data set being transmitted.
2. The first JCL statement is a standard job statement. The next JCL statement is an EXEC statement that specifies `PGM=FTP`, and a region parameter, because FTP might use more storage than your default region size.
3. For `PARM=`, you can specify any parameter that is valid when invoking FTP from your terminal. See “Using the FTP Command” on page 35 for more information. These parameters are only supported on the `PARM=` field of the EXEC card.

As shown in Figure 3, to run FTP in batch, you must have the following three DD statements:

SYSPRINT DD
: Specifies the data set where you want messages to be returned.

INPUT DD
: Specifies the data set where the FTP subcommands to be performed are located.

OUTPUT DD
: Specifies the data set where FTP is to place the client messages and server replies generated during the FTP session.

Note: The data set specified on the OUTPUT DD statement should have an LRecl of 160 with any block size that is a multiple of the LRecl. The data set specified on the INPUT DD statement should have an LRecl of 80 with any block size that is a multiple of the LRecl.

You can use the *user_id*.NETRC data set, as defined by the NETRC DD statement in Figure 3, to identify the user ID and password for a batch-processed remote login. You can also specify the user ID and password in the INPUT DD data set.

Figure 4 shows the records in an INPUT DD data set that contains the FTP commands to be executed.

```
HOSTNAME
USERID PASSWD
DIR
PUT MYFILE.LISTING
QUIT
```

Figure 4. Contents of an INPUT DD Data Set

The first line of Figure 4 on page 75 contains the name of the host that you want FTP to use. The second line contains the user ID followed by its password. The next three lines contain the FTP commands that you want FTP to perform. In this example, FTP is doing a directory listing of the server to which you are connecting. The example then instructs FTP to send a file to the server. The last line ends the connection.

Any client messages and server replies to the commands you execute appear in the OUTPUT DD data set. The SYSPRINT can contain some additional messages that relate to the execution of your FTP session.

Note: If you do not want your password to be copied to the output file, specify your user ID and password on separate input lines. See Figure 5 for an example.

Submitting Requests without Input and Output Data Sets

Figure 5 shows an easier way to submit a batch job, because you can avoid referring to data sets for input and output.

```
//USERIDX  JOB USERID,MSGLEVEL=(1,1),NOTIFY=USERID,MSGCLASS=H,TIME=9
//FTP   EXEC PGM=FTP,REGION=4096K
//INPUT DD *
nodeid
userid
password
CD
DIR
GET hostfile.name locfile.name
QUIT
//OUTPUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

Figure 5. JCL to Run FTP in Batch without Using Data Sets

Figure 6 shows step 1 creating a new GDS in batch and FTP getting the data set.

```
//USERIDX  JOB USERID,MSGLEVEL=(1,1),NOTIFY=USERID,MSGCLASS=H,TIME=9
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DSN=USER31.SOURCE.DATA,DISP=SHR     (MYDGD.G0008V00)
//SYSUT2   DD DSN=MYDGD(+1),DISP=(,CATLG),
              UNIT=SYSDA, SPACE=(TRK,(1,1)),DCB=(MODEL)
//FTP   EXEC PGM=FTP,REGION=4096K
//INPUT DD *
nodeid
userid
password
CD /u/joe
DIR
GET MYDGD(0) A.DATA.SET
QUIT
//OUTPUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

Figure 6. Job to Create a New GDS in Batch

Note: All the GDG allocation in batch must be complete before the start of FTP.

Using the EXEC Interface

The FTP EXEC interface enables you to execute FTP commands from an EXEC rather than interactively from a terminal. The FTP subcommands to be performed can be in a file (MVS data set or HFS file), or you can code them directly in the EXEC.

By default, the FTP session dialog is printed on the terminal. If you want the dialog sent to a data set rather than the terminal:

TSO Specify an OUTPUT data set as part of the ALLOC statement.

OS/390 UNIX System Services
Redirect the output to an HFS file when invoking the FTP command.

The following examples are written in REXX. Refer to the *REXX/MVS Reference* and *Using REXX and OS/390 UNIX System Services MVS* for more information about the REXX language.

Issuing FTP Subcommands from a File

Figure 7 is an example of an EXEC that issues FTP subcommands from a data set. In this example, the FTPIN1 data set is used for the FTP subcommands, and FTPOUT1 is used to store the FTP session dialog. This example must be invoked from TSO.

```
/*REXX*/
"ALLOC DA(FTPIN1) DD(INPUT) SHR REU"   /* Input will be from FTPIN1 */
if rc ¬= 0 then do
   say 'Error in ALLOC INPUT, rc = ' rc
   exit
end
"ALLOC DA(FTPOUT1) DD(OUTPUT) SHR REU" /* Output goes to FTPOUT1     */
if rc ¬= 0 then do
   say 'Error in ALLOC OUTPUT, rc = ' rc
   exit
end
"FTP YKTVSH"                           /* FTP to the YKTVSH host       */
"FREE DD(INPUT)"
"FREE DD(OUTPUT)"
EXIT
```

Figure 7. How to Issue the FTP Subcommands from a Data Set

Figure 8 on page 78 is an example of an EXEC that issues FTP subcommands from an HFS file. In this example, the name of the input file and output file are passed as arguments on the EXEC. For example,

```
EXAMPLE1 /u/user117/ftpin1  /u/user117/ftpout1
```

where `EXAMPLE1` is the name of the EXEC. This example must be invoked from the OS/390 UNIX shell.

```
/* rexx                                                              */
/*                                                                   */
/* Input: infile - HFS file containing FTP commands                  */
/*        outfile - HFS file to contain FTP output. If not specified */
/*                  output goes to terminal.                         */

parse arg infile outfile .          /* get command line input        */

if infile = '' then                 /* input file not specified      */
  do
    say 'Input file name is required.'
    exit 12                         /* return to UNIX System Services            */
  end
else
  input_file = '<' infile           /* redirect input from file      */

if outfile <> '' then
  output_file = '>' outfile         /* redirect output to file       */
else
  output_file = ''

address syscall "stat (infile) fstat."  /* test if input file exist  */
if fstat.0 = 0 then                 /* input file not found          */
  do
    say 'Input file:' infile 'not found.'
    exit 28                         /* return to UNIX System Services            */
  end

"ftp -v -p TCPIP" input_file output_file  /* invoke FTP client with
                                        input and output redirection  */

say "FTP client return code is:" rc  /* print client return code     */

exit 0                              /* return to UNIX System Services            */
```

Figure 8. How to Issue the FTP Subcommands from a Data Set

The following is an example of the input file (either the input data set FTPIN1 or the HFS file /u/user117/ftpin1).

```
krasik mvsftp
cd examples
put t.info t1.info
get t1.info t2.info (r
quit
```

Where:

krasik Is the user ID

mvsftp Is the password

t.info Is the file to be transferred

Issuing FTP Subcommands Directly from the EXEC Interface

Figure 9 on page 79 is an example of how to issue FTP subcommands directly from a REXX EXEC. The REXX stack is used to hold the FTP subcommands. This example runs in both the TSO and OS/390 UNIX environments.

Note: To use FTP in a OS/390 UNIX environment, TSO users must be authorized users or have a default OS/390 UNIX user ID.

```
/* rexx */

/* push commands on stack                                          */
QUEUE "YKTVSH"                      /* server address              */
QUEUE "krasik mvsftp"               /* userid/password             */
QUEUE "cd /tmp/examples/"
QUEUE "put t.info t1.info"
QUEUE "cd .."
QUEUE "cd dummy"
QUEUE "quit"

cmdargs = "-v -p TCPIP"            /* set ftp client arguments      */

parse source . . . . . . . . env .  /* check if running under UNIX System Services     */
                                    /* env='OpenMVS' if invoked from */
                                    /* UNIX System Services, otherwise env=' '         */

if env = "OpenMVS" then             /* running under UNIX System Services              */
  ADDRESS LINKMVS "FTP cmdargs"     /* invoke ftp client             */
                                    /* NOTE: ADDRESS LINKMVS is      */
                                    /* required since commands are   */
                                    /* on the rexx stack.            */
else                                /* running under TSO             */
  "FTP" cmdargs                     /* invoke ftp client             */

say "FTP client return code is:" rc  /* print client return code     */

exit 0                              /* return                        */
```

Figure 9. How FTP Subcommands Can Be Issued from an EXEC

Notes:

1. If data set DUMMY does not exist, FTP exits with a return code.
2. When using the REXX stack, ADDRESS LINKMVS must be used to invoke the FTP command in the OS/390 UNIX environment.

FTP EXIT Return Codes

FTP EXIT return codes are displayed (in message EZA1735I) when the (EXIT parameter is used on the FTP command and an error is detected. The return codes are composed of a subcommand code and a reply code. FTP EXIT return codes have the following format:

yyxxx

Where:

yy Represents the subcommand code, which is a number in the range 1–99. Each subcommand has an EXIT_IF_ERROR flag that determines whether FTP is exited when an error occurs if you specified EXIT on the FTP command. See Table 15 on page 80 for a description of the possible FTP subcommand codes.

xxx Represents the reply code that is sent from the server. The reply code is a 3-digit number. See Table 16 on page 82 for a description of the possible reply codes.

For example, the FTP EXIT return code 16550 indicates the following:

16 The GET command failed.

550 The reply code from the FTP server. The requested action was not taken; the file was not found or could not be accessed.

The FTP EXIT return code 4532 indicates the following:

4 The APPEND command failed.

532 The reply code from the FTP server. Need an account for storing files.

Note that 00 and 000 are valid values for *yy* and *xxx*. This means that the error occurred at a time when no FTP subcommand was being processed *yy*=00 or at a time when no REPLY had been received from the server for the current process. Message EZA1735I Error code is the internal FTP error code. Use this value and Table 17 on page 83 to find the cause of the error.

FTP Subcommand Codes

Table 15 lists the valid FTP subcommand codes. The “EXIT_IF_ERROR” column specifies whether an error causes FTP to end if you specified the EXIT parameter on the FTP command.

Table 15. FTP Subcommand Codes

Code Number	Subcommand	EXIT_IF_ERROR
00	No subcommand selected	Determined by internal FTP CLIENT ERROR CODE
1	AMBIGUOUS	true
2	?	false
3	ACCOUNT	true
4	APPEND	true
5	ASCII	true
6	BINARY	true
7	CD	true
8	CLOSE	true
9	TSO	true
10	OPEN	true
11	DEBUG	false
12	DELIMIT	false
13	DELETE	true
14	DIR	true
15	EBCDIC	true
16	GET	true
17	HELP	false
18	LOCSTAT	true
19	USER	true
20	LS	true
21	MDELETE	true
22	MGET	true
23	MODE	true

Table 15. FTP Subcommand Codes (continued)

Code Number	Subcommand	EXIT_IF_ERROR
24	MPUT	true
25	NOOP	true
26	PASS	true
27	PUT	true
28	PWD	true
29	QUIT	true
30	QUOTE	true
31	RENAME	true
32	SENDPORT	true
33	SENDSITE	false
34	SITE	false
35	STATUS	true
36	STRUCTURE	true
37	SUNIQUE	true
38	SYSTEM	true
39	TRACE	false
40	TYPE	true
41	LCD	true
42	LOCSITE	true
43	LPWD	true
44	MKDIR	true
45	LMKDIR	true
46	EUCKANJI	true
47	IBMKANJI	true
48	JIS78KJ	true
49	JIS83KJ	true
50	SJISKANJI	true
51	CDUP	true
52	RMDIR	true
53	HANGEUL	true
54	KSC5601	true
55	TCHINESE	true
56	RESTART	false
57	BIG5	true
58	BLOCK	true
59	COMPRESS	true
60	FILE	true
61	PROXY	true
62	RECORD	true
63	SCHINESE	true

Table 15. FTP Subcommand Codes (continued)

Code Number	Subcommand	EXIT_IF_ERROR
64	STREAM	true
65	GLOB	false
66	PROMPT	false
67	UCS2	true
99	UNKNOWN	true

FTP Reply Codes

When you enter an FTP command, TCP/IP displays the sequence of subcommands, if any, that are sent to the foreign host FTP server. In addition, the response from the FTP server is also displayed as a reply code. These replies ensure the synchronization of requests and actions during data transfer, and guarantee that the client always know the state of the foreign host FTP server.

Table 16 on page 82 lists the descriptions of the possible reply codes.

Note: The description of the reply codes is not the exact message that is displayed on your screen; the description describes the content of the reply code.

Table 16. FTP Reply Codes

Code	Description
000	FTP subcommand contains an incorrect parameter **Note:** A reply code of 000 is returned from the FTP client when it detects an incorrect parameter or an error occurred in the process such that no REPLY was received. For example, when the remote TCP/IP refuses to connect, the reply code will be 10000. When this occurs use FTP CLIENT ERROR CODES to determine the error. In this case, the FTP client does not send the command to the FTP server.
110	Restart marker reply
120	Service ready in *nnn* minutes
125	Data connection already open; transfer starting
150	File status okay; about to open data connection
200	Command okay
202	Command not implemented; not used on this host
208	Unable to delete data set because expiration date has not passed
211	System status, or system help reply
212	Directory status
213	File status
214	Help message
215	MVS is the operating system of this server
220	Service ready for new user
221	QUIT command received
226	Closing data connection; requested file action successful
230	User logged on; proceed
250	Requested file action okay, completed

Table 16. FTP Reply Codes (continued)

Code	Description
257	PATH NAME created
331	Send password please
332	Supply minidisk password using account
421	Service not available
425	Cannot open data connection
426	Connection closed; transfer ended abnormally
450	Requested file action not taken; file busy
451	Requested action abended; local error in processing
452	Requested action not taken; insufficient storage space in system
500	Syntax error; command unrecognized
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command not implemented for that parameter
530	Not logged on
532	Need account for storing files
550	Requested action not taken; file not found or could not be accessed
551	Requested action abended; page type unknown
552	Requested file action ended abnormally; exceeded storage allocation
553	Requested action not taken; file name not allowed
554	Transfer aborted; unsupported SQL statement

FTP Client Error Codes

Table 17 lists error codes used as condition codes for batch jobs if (EXIT was specified or as returned codes from REXX execs, whenever the FTP client detects one of the described errors and the FTPDATA statement CLIENTERRCODES is TRUE. These codes are always displayed in message EZA1735I Error codes.

Table 17. Internal Error Codes

Code	Error
01	FTP_INTERNAL_ERROR
02	FTP_SERVER_ERROR
03	FTP_CATALOG_NOT_FOUND
04	FTP_INVALID_PARAM
05	FTP_OPEN_IOSTREAM_FAILED
06	FTP_ALREADY_CONNECTED
07	FTP_USAGE
08	FTP_CONNECT_FAILED
09	FTP_TIMEOUT
10	FTP_SESSION_ERROR

Table 17. Internal Error Codes (continued)

Code	Error
11	FTP_LOGIN_FAILED
12	FTP_INPUT_ERR
13	FTP_INPUT_EOF
14	FTP_NOTFOUND
15	FTP_INVALID_ENVIRONMENT
16	FTP_NOTENABLED

For example, the internal error code 13 indicates that the internal error `FTP_INPUT_EOF` occurred, which means that there was an unexpected end-of-file in the input stream.

Interfacing with JES

The MVS Job Entry System (JES) enables you to perform the following functions:

- Submit jobs (consisting of JCL and data) to the job scheduler for execution
- Spool JCL messages and SYSOUT during execution
- Print the output
- View the output
- Delete job output

FTP server provides the following functions in its JES interface:

- Submitting a job
- Displaying the status of all the user's jobs
- Receiving the spool output of the job (JCL messages and SYSOUT)
- Deleting a job
- Submitting a job and automatically receiving output
- Terminating access to JES

Submitting a Job

A job consists of job control language (JCL) and data. To submit a job using FTP, you must do the following:

1. Create the JCL and data that you want to submit, using the editor on your client. The job name in the JCL must be USERID*x*, where *x* is a 1-character letter or number and USERID must be the user ID you use to log into the FTP server to submit the job. For example: `MYUSRIDA`. The output class for MSGCLASS and SYSOUT files contained in your JCL must specify a JES HOLD output class.

   Note: The maximum LRecl for the submitted job is 254 characters. JES scans only the first 72 characters of JCL.
2. Start a session with the FTP server on the MVS system to which you want to submit the job.
3. After you have entered your user ID and password, specify that you want to interface to JES with a site parameter by entering the following:

   ```
   SITE FILEtype=JES
   ```
4. To submit the JCL file you have created, enter the following:

```
PUT filename.filetype
```

The JCL is then submitted to the JES internal reader and waits for an initiator to start the job. The job is submitted under the user ID that you used when you logged on to the system.

The default for *filetype* is SEQ, and when you want to go back to normal FTP file transfer mode, enter the following:

```
SITE FILEtype=SEQ
```

Refer to the *JCL Reference Manual* for more information about using JCL.

Displaying the Status of a Job

This section describes client operation when the MVS server has been placed in FILEtype=JES mode with the SITE command. After you have submitted your job, you can determine whether it is waiting for execution, running, or finished. These subcommands display the status of all the jobs that are on the JES spool for your user ID. A display for the DIR subcommand might look like the following:

```
MYUSRIDA JOB05444 OUTPUT 3 spool Files
MYUSRIDB JOB05766 OUTPUT 6 spool Files
MYUSRIDC JOB05832 OUTPUT 6 spool Files
MYUSRIDD JOB05946 ACTIVE
MYUSRIDE JOB06021 INPUT                 -HELD-
```

The first column displays the job name. The second column displays the job ID, assigned by JES. This 8-character job ID, consisting of JOB followed by a 5-digit number assigned by JES, is the way that JES identifies your job. The third column displays the status of the job.

The following is a description of each status:

Status Description

INPUT The job was received, but not run yet.

HELD The JCL specified that the job is to be put on hold.

ACTIVE
The job is running.

OUTPUT
The job has finished and has output to be printed or retrieved. For each OUTPUT job, there are spool files that consist of JCL messages, JES messages, initiator and terminator messages, and SYSOUT. For jobs with a status of OUTPUT, the number of spool files for each job is specified in the DIR display output.

Note: The LS subcommand gives the same results as the DIR subcommand, but does not provide the number of spool files. Providing this spool information consumes a lot of computer resources. The server provides the job ID only as the result of an LS subcommand in order to support an MGET subcommand. Use the LS subcommand rather than the DIR subcommand when possible.

Receiving Spool Output

To retrieve the JCL messages, JES messages, initiator and terminator messages, and SYSOUT contained in the spool files, you must retrieve the spool files either individually or in a group.

Receiving Individual Spool Files

Retrieving the spool files one at a time enables you to see whether a job ran correctly before you retrieve the rest of the output, enabling you greater control over retrieving job information.

To retrieve the spool output, use the FTP GET subcommand while in the FILEtype=JES mode, specifying the job ID and the number of the spool file you want.

You can specify a short form of the job ID by entering the letter `J` followed by a 4- or 5-digit job number. For example:

```
GET JOB05444.1 JOB05444.FILE1 (REPLACE
GET JOB05766.6 ASSEMBLY.FILE6
GET JOB06235.2 (REPLACE
GET JOB00275.4
GET J7438.3
```

In these examples, *foreign_file* is specified first, followed by *local_file* (on your client machine) with the appropriate options, such as REPLACE. The first example requests that the first spool file for JOB05444 be transmitted and replace the file on your client named JOB05444.FILE1. The second command requests that the sixth spool file for JOB05766 be transmitted to your client with the name ASSEMBLY.FILE6.

If you have specified FILEtype=JES, you can use the MGET subcommand to receive output from multiple jobs without specifying them one at a time. For example you can enter:

```
MGET parameter
```

The FTP client requires an MGET subcommand parameter. The parameter is passed to the FTP server but is not used. The server returns all of the SYSOUT files for all of the jobs in the HELD queue for your user ID.

Notes:

1. On an MVS FTP server, *local_file* must be specified.
2. Truncation can cause a loss of data.
3. A GET command performed on an empty data set erases the contents of the existing local data set.
4. Receiving the output of a job does not remove the job output from the queue. To remove the job output from the queue, you must issue a DELETE command.

Receiving a Group of Spool Files

To retrieve all the spool files associated with the same job simultaneously into the same destination file, specify:

```
GET jobid.x
```

where *x* can be uppercase or lowercase. All the spool files are transferred together and put into file `jobid`.x. The following line appears between each retrieved JES spool file:

```
!! END OF JES SPOOL FILE !!
```

This enables you to easily find the end of each spool file.

You can also specify a data set name to send the files to, such as:

```
GET jobid.x data_set_name
```

All the spool files are put into the file named `file.name`. This eliminates the need to retrieve each spool file separately. For example,`GET J3456.X` retrieves all the spool files for JOB03456 and puts them in a file named J3456.X.

The command MGET with any parameter produces the same results as issuing GET *jobid.x* commands for each job that is associated with your user ID.

Notes:

1. In JES2, the spool files retrieved by GET and tallied by DIR must be in a hold queue (commonly class=H).
2. In JES3, the spool files must be in a hold queue reserved for external writers. Ask your system programmer for the class that says (HOLD=EXTWTR) in the JES3 installation stream.
3. The maximum record length that can be received at the server is 254 characters before the record is truncated.
4. Receiving the output of a job does not remove the job output from the queue. To remove the job output from the queue, you must issue a DELETE command.

Deleting a Job

You can delete a job before or during execution, or you can delete the output of a job before you have retrieved it. You do this by using the DELETE subcommand while in the FILEtype=JES mode and the job ID. You can specify either the 8-character job ID or a short form of the job ID by entering the letter `J` followed by a 4- or 5-digit job number. For example,

```
DELETE JOB05444
DELETE J3672
```

When you issue the DELETE command, all spool output related to a job is deleted.

The host returns the message `CANCEL SUCCESSFUL` after it deletes the job.

Submitting a Job and Automatically Receiving Output

You can submit a job using FTP and automatically receive your output. Rather than using the JCL you built on the FTP client, this function uses the JCL you have built on the FTP server. To submit a job using FTP, perform the following steps:

1. Create the JCL and data that you want to submit and save it on the MVS host where the FTP server resides. The JCL can reside in a sequential or partitioned data set. The job name in the JCL must be USERID*x*, were *x* is a 1-character letter or number. The output class for MSGCLASS and SYSOUT files contained in your JCL must specify a JES HOLD output class.
2. Start a session with the FTP server on the MVS system to which you want to submit the job.

3. After you have entered your user ID and password, specify that you want to interface to JES with a site parameter by entering the following:

   ```
   SITE FILEtype=JES
   ```

4. To submit the JCL file you have created, enter the following command:

   ```
   GET jclfilename.jclfiletype outputfilename.outputfiletype
   ```

   The *outputfilename.outputfiletype* defines the data set at the FTP client site that is to contain the HELD job output when the job completes.

The MVS FTP server reads the data set *jclfilename.jclfiletype* and submits it to the JES internal reader. It then sends the client the following two messages:

```
125 Submitting job outputfilename.outputfiletype FIXrecfm 80
125 When JOB05125 is done, will retrieve its output
```

Note: When submitting a job and automatically receiving the output, remember that your session is suspended. You should use care, based on the anticipated run time of your job, when using this function. If your session times out, you must restart FTP and manually retrieve your output. The time out is determined by the value of JESPUTGETTO in the server FTP.DATA data set and is usually 10 minutes.

Terminating Access to JES

The FTP default for FILEtype is SEQ. When you want to end access to JES and return to FTP in its normal file transfer mode, specify the following:

```
SITE FILEtype=SEQ
```

JES Examples

The following example shows the JCL file USER121.JCL.CNTL(SMFALL) being submitted to the JES. Before FTP commands are issued, only the data set USER121.FTP.EXAMPLE exists on MVSXA2.

```
  User:  ftp 9.67.113.24 621
System:
           IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           FTP.DATA FILE NOT FOUND.  USING HARDCODED DEFAULT VALUES.
           Connecting to 9.67.113.24, port 621
           220-FTPSERVE IBM CS/390 V2R8 at MVSVIC03.TCP.RALEIGH.IBM.COM, 19:03:08
           on 1997-09-17
           220 Connection will close if idle for more than 5 minutes.
           NAME (<host>:tsouserid):
  User:  user121
System:
           >>>USER user121
           331 Send password please.
           Password:
           >>>PASS ********
           230 user121 is logged on. Working directory is "/u/user121".
           Command:
```

```
  User:  site file=jes
System:
           >>>SITE file=jes
           200 Site command was accepted
           Command:
  User:  put 'user121.jcl.cntl(mvsjob)'
System:
           >>>SITE FIXrecfm 80 LRECL=80 RECFM=FB BLKSIZE=27920
           200 Site command was accepted
           >>>PORT 9,67,112,25,4,37
           200 Port request OK.
           >>>STOR 'user121.jcl.cntl(mvsjob)'
           125 Sending Job to JES Internal Reader FIXrecfm 80
           250-It is known to JES as JOB02189.
           250 Transfer completed successfully.
           1066 bytes transferred in 3.118 seconds. Transfer rate 0.34 Kbytes/sec.
           Command:
```

```
  User:  dir
System:
           >>>PORT 9,67,112,25,4,38
           200 Port request OK.
           >>>LIST
           125 List started OK.
           USER121A  JOB00067  INPUT
           250 List completed successfully.
           Command:
  User:  dir
System:
           >>>PORT 9,67,112,25,4,39
           200 Port request OK.
           >>>LIST
           125 List started OK.
           USER121A  JOB00067  ACTIVE
           250 List completed successfully.
           Command:
  User:  dir
System:
           >>>PORT 9,67,112,25,4,40
           200 Port request OK.
           >>>LIST
           125 List started OK.
           USER121A  JOB00067  OUTPUT    4 Spool Files
           250 List completed successfully.
           Command:
```

```
  User:  lcd 'user121.ftp.example.'
System:
           Local directory name set to USER121.FTP.EXAMPLE.
           Command:
  User:  lpwd
System:
           Local directory is USER121.FTP.EXAMPLE.

           Command:
  User:  dir
System:
           >>>PORT 9,67,112,25,4,41
           200 Port request OK.
           >>>LIST
           125 List started OK.
           USER121A  JOB00067  OUTPUT    4 Spool Files
           250 List completed successfully.
           Command:
  User:  get job00067.x spoolall
System:
           >>>PORT 9,67,112,25,4,42
           200 Port request OK.
           >>>RETR job00067.x
           125 Sending all SPOOL files for requested JOBID.
           250 Transfer completed successfully.
           5935 bytes transferred in 4.755 seconds. Transfer rate 1.25 Kbytes/sec.
           Command:
```

```
  User:  get job00067.1 spool1
System:
           >>>PORT 9,67,112,25,4,43
           200 Port request OK.
           >>>RETR job00067.1
           125 Sending data set USER121.USER121A.JOB00067.D000002.JESMSGLG
           250 Transfer completed successfully.
           1962 bytes transferred in 0.739 seconds. Transfer rate 2.65 Kbytes/sec.
           Command:
  User:  get job00067.2 spool2
System:
           >>>PORT 9,67,112,25,4,44
           200 Port request OK.
           >>>RETR job00067.3
           125 Sending data set USER121.USER121A.JOB00067.D000003.JESYSMSG
           250 Transfer completed successfully.
           1982 bytes transferred in 2.123 seconds. Transfer rate 0.93 Kbytes/sec.
           Command:
  User:  get job00067.3 spool3
System:
           >>>PORT 9,67,112,25,45
           200 Port request OK.
           >>>RETR job00067.3
           125 Sending data set USER121.USER121A.JOB00067.D000004.JESYSMSG
           250 Transfer completed successfully.
           1982 bytes transferred in 2.123 seconds. Transfer rate 0.93 Kbytes/sec.
           Command:
  User:  get job00067.4 spool4
System:
           >>>PORT 9,67,112,25,46
           200 Port request OK.
           >>>RETR job00067.4
           125 Sending data set USER121.USER121A.JOB00067.D000103.?
           250 Transfer completed successfully.
           1227 bytes transferred in 0.380 seconds. Transfer rate 3.23 Kbytes/sec.
           Command:
```

```
  User:  get job00067.5 spool5
System:
           >>>PORT 9,67,112,25,47
           200 Port request OK.
           >>>RETR job00067.5
           550 Index 5 is greater than number of spool files for JOB00067
           Command:
  User:  dir
System:
           >>>PORT 9,67,112,25,4,50
           200 Port request OK.
           >>>LIST
           125 List started OK.
           user121A  JOB00067  OUTPUT    4 Spool Files
           250 List completed successfully.
           Command:
  User:  delete job00067
System:
           >>>DELE job00067
           250 Cancel Successful
           Command:
  User:  dir
System:
           >>>PORT 9,67,112,25,4,51
           200 Port request OK.
           >>>LIST
           125 List started OK.
           No jobs found on Held queue
           250 List completed successfully.
           Command:
  User:  site filetype=seq
System:
           >>>SITE filetype=seq
           200 Site command was accepted
           Command:
  User:  quit
System:
           >>>QUIT
           221 Quit command received. Goodbye.
           READY
```

After executing the FTP commands, the following data sets now exist on MVSXA2:

USER121.FTP.EXAMPLE.SPOOLALL

USER121.FTP.EXAMPLE.SPOOL1

USER121.FTP.EXAMPLE.SPOOL2

USER121.FTP.EXAMPLE.SPOOL3

USER121.FTP.EXAMPLE.SPOOL4

Note: In most situations, the INPUT status is too fast to be captured by issuing DIR. However, if the ACTIVE or OUTPUT status of the job is captured, the INPUT status has been passed successfully.

Performing DB2 SQL Queries with FTP

FTP enables you to submit a Structured Query Language (SQL) SELECT query to the DB2 subsystem and receive the results of the SQL query. FTP can perform this function as either the server or the client.

For information on installing the SQL query function for the FTP client or server, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

SQL Data Types Supported by FTP

FTP access to SQL supports the following data types:

- DATE
- TIME
- TIMESTAMP
- VARCHAR (variable length, up to 254 characters)
- CHAR (fixed length, up to 254 characters)
- DECIMAL
- INTEGER (full word)
- SMALLINT (half word)
- FLOAT (single or double precision)
- LONG VARCHAR (**VARCHAR**(*n*), where *n* is greater than 254)
- GRAPHIC
- VARGRAPHIC
- LONG VARGRAPHIC

Mixed data (double-byte character set and single-byte character set) is supported in CHAR, VARCHAR, and LONG VARCHAR data types, but column alignment in the output file might not be maintained.

Creating the Input Data Set

Before performing a DB2 SQL query using FTP, you must create an MVS data set that contains the SQL query you want to perform.

You can create queries on the client and use the FTP PUT command to send the queries to the MVS system to be processed. Or, you can prepare a group of "stock" SQL queries on the MVS system and perform them regularly.

Note: FTP can process only one SQL query per file.

For example, a data set on an MVS system named `userid.SQL.IN` contains the following SQL query:

```
SELECT LASTNAME, EMPID, YEARS_EMPLOYED FROM EMPLOYEE_TABLE
WHERE YEARS_EMPLOYED > 25
```

You either created that data set on the MVS system with TSO, or you used the FTP PUT command to put the data set on the MVS system.

Setting the Characteristics for the SQL Query

After creating a data set to use for your query, you must log on to FTP and set the file type for the query:

```
SITE/LOCSITE FILEtype=SQL
```

There are several commands that are relevant to the client and server in SQL mode. The server commands use SITE, and the client commands use LOCSITE. The following list describes the commands:

SITE/LOCSITE DB2= Specifies the name of the DB2 subsystem that you want to perform your queries. See "Specifying the

DB2 Subsystem to Perform the Query" for more information about DB2 subsystems.

SITE/LOCSITE SPRead or NOSPRead
Specifies whether you want the output to be in spreadsheet or report format. See "Specifying the Output Format" on page 94 for more information about output format.

SITE/LOCSITE SQLCol= Specifies whether you want the column headings to use the DB2 column names or labels. Valid values include Names, Labels, or Any. See "Assigning Column Headings for the SQL Query Result Table" for more information about column headings.

To return to normal FTP processing after performing queries, or other processes, specify:

```
SITE/LOCSITE FILEtype=SEQ
```

On MVS systems, RECfm=VB is a recommended format that enables you to view the results of the SQL query. Issue the following command to specify that new data sets should be created with the RECfm=VB attribute:

```
SITE/LOCSITE RECFM=VB
```

To prevent the automatic sending of a SITE command that might override your SITE setting, toggle SENDSITE to OFF. For more information about the SENDSITE command, see "SENDSITE Subcommand—Toggle the Sending of Site Information" on page 196.

Specifying the DB2 Subsystem to Perform the Query

An MVS system can run several DB2 systems simultaneously, each known by a subsystem name of up to four characters. For example, you can have a DB2 test system called DB2T and a DB2 production system called DB2P.

FTP connects to a DB2 system to have it execute a DB2 query. You can specify what DB2 system FTP should connect to with the following SITE or LOCSITE parameter:

```
SITE/LOCSITE DB2=
```

For example, if you want the FTP server to have the DB2T system perform your queries, specify:

```
SITE DB2=DB2T
```

If you want the FTP client to have the DB2P system perform your queries, specify:

```
LOCSITE DB2=DB2P
```

The default DB2 system name is *DB2*. You can change the default with the DB2 parameter in the FTP.DATA data set. See "Changing Local Site Defaults Using FTP.DATA" on page 59 for more information about the FTP.DATA data set.

Assigning Column Headings for the SQL Query Result Table

When you create a DB2 table, you can assign descriptive labels to the table columns. For example, a column name could be XCM554, but the label could be

WEEKLY PAY. For information about assigning names and labels, refer to the DB2 DESCRIBE statement in the *DB2 SQL Reference*.

The SQLCol parameter of the SITE command enables you to specify whether you want names or labels to appear at the top of the columns in your output file. The default is Names. Issue the following command if you want a database column name to appear at the top of each column in your output file:

```
SITE/LOCSITE SQLCol=Names
```

Issue the following command if you want a label to appear at the top of each column:

```
SITE/LOCSITE SQLCol=Labels
```

If you specify the Labels parameter and a column in your query does not have a label defined in the database, the FTP server supplies a column heading. For more information about column headings, see “FTP-Supplied Column Headings”.

Issue the following command if you want either a label or a name to appear at the top of each column:

```
SITE/LOCSITE SQLCol=Any
```

If you specify the Any parameter, the label appears as the column heading. However, if the column does not have a label, the name appears at the top of the column.

FTP-Supplied Column Headings

The FTP client and server provide column headings in the result table when DB2 does not. This occurs when a result table contains expression columns or when labels are requested and a database column that appears in the result table does not have a label defined.

FTP builds a column heading for expression columns. For example,

```
Select employee, salary/52 from ABC.Staff
```

results in two columns. The first column gets its name from DB2, while the second column is built by the server. The server uses the heading `COL002` for the second column because it supports the SQL limit of 750 columns.

Specifying the Output Format

You have two choices for the format of your output data set: spreadsheet format and report format. The default is NOSPRead (report format), but you can change the default for your FTP server by changing the FTP.DATA data set. See “Changing Local Site Defaults Using FTP.DATA” on page 59 for more information.

Spreadsheet Format: You can have the output of the SQL query formatted to load directly into a spreadsheet program running on a PC or a workstation. To get the spreadsheet format, issue the following command:

```
SITE SPRead or LOCSITE SPRead
```

The SPRead format option puts a TAB character before the first character of each column entry, except the first column. See your spreadsheet program documentation for instructions about how to import the output of the SQL query.

Report Format: The NOSPRead format option puts one or more blank spaces between the columns, and it lists the SQL query, the column headings, and the resulting columns. Each section is separated with horizontal dashed lines. An output data set in NOSPRead, or report, format is easier to view and print.

To get the report format, issue the following command:

```
SITE NOSPRead or LOCSITE NOSPRead
```

The following is an example of the results contained in the NOSPRead format of the SQL.OUTPUT data set.

```
s---------+---------+---------+---------+---------+--------+---------+
SELECT * FROM DB2USER.PHONES
         WHERE FIRSTNAME LIKE 'BILL%'
         OR    FIRSTNAME LIKE 'WILL%'
h---------+---------+---------+---------+---------+---------+---------+
LASTNAME              FIRSTNAME       TIE  EXT  ALT  DEPT ROOM   NODE
d---------+---------+---------+---------+---------+---------+---------+
ACKERMAN              BILL            893  6266 7813 431  J2-A22 IBMABC
ADAMS                 WILLIAM J.      892  2202 1716 681  33-943 IBMABC
ASTERMAN              WILLIAM C.      893  7244 7813 222  J4-A44 IBMVM2
BENDER                WILLIAM R.      892  4217 4766 490  45-556 IBMVM2
```

A lowercase letter in the first position of each dashed line specifies what part of the output follows, enabling a program to read and interpret the contents. For example, `s` indicates that the SQL query follows, `h` indicates a header, `d` indicates that the rest of the data set is the actual data, and `e` indicates that an error message follows.

The width of the output data set depends on the width of the results from the DB2 query.

Submitting the Query

After you have created a data set that contains an SQL query, logged on to FTP, and set the appropriate SITE or LOCSITE parameters, you are ready to execute the contents of the data set. You can do this from either an FTP client or an FTP server.

Performing an SQL Query from an FTP Client

To have the FTP client perform SQL queries and have the results sent to an FTP server, specify:

```
LOCSITE FILEtype=SQL
```

Next, perform a PUT command specifying the name of the file on the client that contains the SQL query.

For example, if the client has a file named `userid.SQL.IN` that contains an SQL query, you can specify:

```
PUT SQL.IN SQL.OUT
```

The FTP client then submits the query found in SQL.IN to the DB2 subsystem on the client and sends the resulting rows of output to the server to be put into SQL.OUT on the server.

To return to normal FTP processing, specify:

```
LOCSITE FILEtype=SEQ
```

Performing an SQL Query from an FTP Server

To have the FTP server perform the query and have the results sent to the client, specify:

```
SITE FILEtype=SQL
```

Then perform a GET command specifying the name of the file on the server that contains the SQL query.

For example, if the server has a file named *userid*.SQL.IN that contains an SQL query, you can specify:

```
GET SQL.IN SQL.OUT
```

The FTP server then submits the query found in SQL.IN to the DB2 subsystem on the server and sends the resulting rows of output to the client to be put into SQL.OUT on the client.

Examples of SQL Query Output

This section shows examples of SQL query output using different options.

With NOSPRead and SQLCol=Names

The following output is from a query using NOSPRead and SQLCol=Names.

```
s--------+---------+---------+---------+---------+---------+---------+---------+
SELECT EMPLOYEE,AGE
FROM   ABC.STAFF
WHERE  AGE < 60
h--------+---------+---------+---------+---------+---------+---------+---------+
EMPLOYEE                  AGE
d--------+---------+---------+---------+---------+---------+---------+---------+
Steve Jasinski             23
Alison Cook                22
```

With SPRead and SQLCol=Names

The following output is from a query with SPRead and SQLCol=Names.

Note: The period symbol (.) represents a TAB character.

```
EMPLOYEE            .AGE
Steve Jasinski      . 23
Alison Cook         . 22
```

The following output examples are for the query:

```
SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)
```

With NOSPRead and SQLCol=Names

The following are output from queries using NOSPRead and SQLCol=Names.

```
s--------+---------+---------+---------+---------+---------+---------+---------+
SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)
h--------+---------+---------+---------+---------+---------+---------+---------+
```

```
TLA    SALARY
d--------+---------+---------+---------+---------+---------+---------+---------+
ACO  20050.00
SJJ  19040.00
```

and

```
s--------+---------+---------+---------+---------+---------+---------+---------+
SELECT * FROM ABC.STAFF
h--------+---------+---------+---------+---------+---------+---------+---------+
EMPLOYEE              TLA      AGE    SALARY
d--------+---------+---------+---------+---------+---------+---------+---------+
Steve Jasinski        SJJ       23  28040.00
Alison Cook           ACO       22  28040.00
Mark Ballam           MFB       63  87420.55
```

With NOSPRead and SQLCol=Labels

The following output is from a query using NOSPRead and SQLCol=Labels.

```
s--------+---------+---------+---------+---------+---------+---------+---------+
SELECT DISTINCT ABC.STAFF.TLA, ABC.STAFF.SALARY
FROM   ABC.STAFF, ABC.HOURS
WHERE  (ABC.STAFF.TLA = ABC.HOURS.TLA) AND
       (ABC.HOURS.TOTAL > 40)
h--------+---------+---------+---------+---------+---------+---------+---------+
EMPLOYEE'S INITIALS    SALARY
d--------+---------+---------+---------+---------+---------+---------+---------+
ACO                  20050.00
SJJ                  19040.00
```

With NOSPRead and SQLCol=Any

The following output is from a query using NOSPRead and SQLCol=Any.

```
s--------+---------+---------+---------+---------+---------+---------+---------+
SELECT * FROM ABC.STAFF
h--------+---------+---------+---------+---------+---------+---------+---------+
EMPLOYEE              EMPLOYEE'S INITIALS      AGE    SALARY
d--------+---------+---------+---------+---------+---------+---------+---------+
Steve Jasinski        SJJ                       23  28040.00
Alison Cook           ACO                       22  28040.00
Mark Ballam           MFB                       63  87420.55
```

FTP with DBCS Support

This section describes how to use FTP to exchange DBCS data sets between hosts supporting DBCS file transfer.

The MVS TCP/IP FTP server and client programs access data sets containing data that is usually in EBCDIC format. To transfer these data sets to or from an ASCII-based host requires translation tables. The transfer of DBCS data uses two tables—one for DBCS characters and one for SBCS characters.

Selecting a DBCS Translation Table

The LOADDBCSTABLES statement in *hlq*.TCPIP.DATA is used by both the FTP server and client to determine which DBCS translation table data sets can be loaded. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about the loading and customizing of DBCS translation tables for FTP.

The FTP server and client can be configured to load a number of DBCS translation tables. These are used during data set transfers to convert MVS host DBCS characters and non-MVS DBCS characters. The command

```
TYPE B n
```

or corresponding client subcommand is used to enter DBCS transfer mode and select a DBCS table.

Selecting an SBCS Translation Table

The SBCS table used to transfer DBCS data is the SBCS table that is established for the data connection.

SBCS tables are used by the control connection to transfer commands; they are also used by the data connection. Often the same SBCS table is used, but you may want to select a different table to be used for data transfers. How you specify the SBCS table for the data connection depends on whether the translation is to be done by the FTP server or the FTP client.

When the EBCDIC-to-ASCII translation is done by the FTP server, you can issue a SITE SBDataconn command to select the SBCS table to be used by the server for data transfers.

When the EBCDIC-to-ASCII translation is done by the FTP client, you can use the following parameters in your local FTP.DATA file to establish the SBCS tables:

CTRLConn Establishes the SBCS tables the client uses for control connections.

SBDataconn Establishes the SBCS tables the client uses for data connections.

Alternatively, you can use the TRANSLATE option of the FTP command to change the SBCS and DBCS translation table hierarchy for both the control and data connection. The TRANSLATE option results in the same SBCS table for both the control and the data connection.

Note: The TRANSLATE option can be used as long as the table maintains the integrity of the portable character set.

Another alternative when the FTP client is to perform the translation is to use the LOCSITE SBDataconn subcommand to change the SBCS table used by the client for the data connection.

DBCS Subcommands

DBCS data sets are transferred using the standard FTP subcommands PUT and GET. However, before the transfer commences, the current transfer type for the session must be set to the required DBCS type. To set the transfer type to DBCS for an FTP session, you must issue the appropriate FTP subcommand to the client or the server, depending on where the DBCS conversion is to be done. The FTP subcommands for DBCS support are listed in Table 18.

Table 18. FTP Subcommands for DBCS Support

Subcommand	Description	Page
BIG5	Sets the transfer type to BIG-5	108
EUckanji	Sets the transfer type to EUCKANJI	128

Table 18. FTP Subcommands for DBCS Support (continued)

Subcommand	Description	Page
HAngeul	Sets the transfer type to HANGEUL	135
Ibmkanji	Sets the transfer type to IBMKANJI	138
JIS78kj	Sets the transfer type to JIS78KJ	139
JIS83kj	Sets the transfer type to JIS83KJ	140
Ksc5601	Sets the transfer type to KSC5601	141
QUOte	Sends an uninterpreted string of data	187
SChinese	Sets the transfer type to SCHINESE	193
SJiskanji	Sets the transfer type to SJISKANJI	212
TChinese	Sets the transfer type to TCHINESE	232
TYpe	Specifies the transfer type	235

Server Commands and Client Subcommands

Table 19 shows examples of the server command that would be generated for each client subcommand:

Table 19. FTP TYPE Subcommand Aliases

Client Subcommand	Server Command	Description
BIG5	TYPE B 8	Big-5 transfer type
EUCKANJI	TYPE B 2	Extended Unix Code kanji transfer type
HANGEUL	TYPE B 5	Hangeul transfer type
IBMKANJI	TYPE F 1	IBM (EBCDIC) kanji transfer type
JIS78KJ	TYPE B 4 A	JIS 1978 kanji using ASCII shift-in transfer type
JIS78KJ (ASCII	TYPE B 4 A	ASCII shift-in escape sequence
JIS78KJ (JISROMAN	TYPE B 4 R	JISROMAN shift-in escape sequence
JIS78KJ (JISROMAN NOSO	TYPE B 4 R N	Pure DBCS data transfer
JIS83KJ	TYPE B 3 A	JIS 1983 kanji using ASCII shift-in transfer type
JIS83KJ (ASCII	TYPE B 3 A	ASCII shift-in escape sequence
JIS83KJ (JISROMAN	TYPE B 3 R	JISROMAN shift-in escape sequence
JIS83KJ (JISROMAN NOSO	TYPE B 3 R N	Pure DBCS data transfer
KSC5601	TYPE B 6	Korean Standard Code KSC-5601 transfer type
SCHINESE	TYPE B 9	Simplified Chinese transfer type
SJISKANJI	TYPE B 1	Shift JIS kanji transfer type
SJISKANJI (Sosi	TYPE B 1 S A	Shift-out/shift-in characters X'1E'/X'1F'
SJISKANJI (Sosi ASCII	TYPE B 1 S A	Shift-out/shift-in characters X'1E'/X'1F'

Table 19. FTP TYPE Subcommand Aliases (continued)

Client Subcommand	Server Command	Description
SJISKANJI (Sosi EBCDIC	TYPE B 1 S E	Shift-out/shift-in characters X'0E'/X'0F'
SJISKANJI (Sosi SPACE	TYPE B 1 S S	Shift-out/shift-in characters X'20'/X'20'
SJISKANJI (NOSO	TYPE B 1 N	Pure DBCS data transfer
TCHINESE	TYPE B 7	Traditional Chinese (5550) transfer type

Mapping DBCS Aliases to CCSIDs

The code sets supported by the DBCS for FTP options conform to standard coded character set identifiers (CCSIDs). Table 20 shows how CCSIDs map to DBCS keywords.

For more information about CCSIDs, refer to *Character Data Representation Architecture*.

Table 20. Mapping of DBCS Keywords to CCSIDs

DBCS Keyword	CCSID	Description
BIG5	00947	IBM Big-5 DBCS
EUCKANJI	00954	Japanese EUC (G0, G1 and G2 only)
HANGEUL	00926	Korean DBCS-PC
JIS78KJ	00955	JIS X0208–1978
JIS83KJ	05048	JIS X0208–1990
KSC5601	00951	IBM Korean Standard code
SCHINESE	01380	Simplified Chinese DBCS-PC
SJISKANJI	00301	Japanese DBCS-PC
TCHINESE	00927	Traditional Chinese DBCS-PC

FTP Subcommands

The FTP subcommands are listed in Table 21. The minimum abbreviation, a description, and the page reference for each subcommand are also included. You must be in the FTP environment to use the FTP subcommands. See "Using the FTP Command" on page 35 for more information.

Table 21. FTP Subcommands

Subcommand	Description	Page
?	Provides information to use FTP	137
!	Invokes a OS/390 UNIX System Services shell function. To use this command, you must be running the FTP client in the OS/390 shell.	103
ACcount	Sends host-dependent account information	104
APpend	Appends a data set on your local host to a file on the foreign host	105

Table 21. FTP Subcommands (continued)

Subcommand	Description	Page
AScii	Sets the transfer type to ASCII	107
BIG5	Sets the transfer type to BIG5. BIG is the minimum abbreviation for BIG5.	108
Binary	Sets the transfer type to IMAGE	110
BLock	Sets the data transfer mode to block mode. This is equivalent to specifying the MODE B subcommand.	111
CD	Changes the working directory	112
CDUp	Changes to the parent of the current working directory	115
CLose	Disconnects from the foreign host	117
COMpress	Sets the data transfer mode to compressed mode. This is equivalent to specifying the MODE C subcommand.	118
CWd	Changes the working directory (Synonymous with CD)	112
DEBug	Toggles or sets internal debug options	119
DELEte	Deletes a single file on the foreign host	122
DELImit	Displays the delimiter character between the *file_name* and *file_type*	123
DIr	Lists the directory entries for files on the foreign host	124
EBcdic	Sets the transfer type to EBCDIC	127
EUckanji	Sets the transfer type to EUCKANJI	128
File	Sets the file structure to file. This is equivalent to specifying the STRUCTURE F subcommand.	130
Get	Copies a file from the foreign host to your local host	131
GLob	Toggles globbing (the expansion of metacharacters in file names) for the MDELETE, MGET, and MPUT subcommands.	133
HAngeul	Sets the transfer type to HANGEUL	135
HElp	Displays help information for FTP	137
Ibmkanji	Sets the transfer type to IBMKANJI	138
JIS78kj	Sets the transfer type to JIS78KJ	139
JIS83kj	Sets the transfer type to JIS83KJ	140
Ksc5601	Sets the transfer type to KSC5601	141
LCd	Changes the current directory on the local host	143
LMkdir	Creates a PDS on the local host	145
LOCSIte	Specifies information that is used by the local host to provide service specific to that host system	147
LOCSTat	Displays FTP status information for the local host	160
LPwd	Displays the name of the active working directory on the local host	163
LS	Lists the names of files on the foreign host	164
MDelete	Deletes multiple files on the foreign host	166
MGet	Copies multiple files from the foreign host to your local host	168
MKdir	Creates a directory on the foreign host	172

Table 21. FTP Subcommands (continued)

Subcommand	Description	Page
MOde	Specifies the mode or data format of the transfer	174
MPut	Copies multiple files on your local host to the foreign host	175
NOop	Checks whether the foreign host is still responding	177
Open	Opens a connection to a foreign host	178
PAss	Supplies a password to the foreign host	179
PROMpt	Toggles interactive prompting for MDELETE, MGET, and MPUT commands. This function is similar to specifying the FTP command with the -i option, which turns off interactive prompting.	180
PROXy	Executes an FTP subcommand on a secondary control connection	181
PUt	Copies a file on your local host to the foreign host	183
PWd	Displays the name of the active working directory on the foreign host	185
QUIt	Leaves the FTP command environment	186
QUOte	Sends an uninterpreted string of data	187
RECord	Sets the file structure to record. This is equivalent to specifying the STRUCTURE R subcommand.	189
RENname	Renames a file on the foreign host	190
RESTart	Restarts a checkpointed data transfer	191
RMdir	Removes a directory	192
SChinese	Sets the transfer type to SCHINESE	193
SENDPort	Enables or disables automatic transmission of the FTP server PORT subcommand	195
SENDSite	Enables or disables automatic transmission of the SITE subcommand	196
SIte	Sends information to the foreign host using site-specific commands	197
SJiskanji	Sets the transfer type to SJISKANJI	212
STAtus	Displays status information for the foreign host	214
STREam	Sets the data transfer mode to stream mode. This is equivalent to specifying the MODE S subcommand.	228
STRucture	Sets the file transfer structure	229
SUnique	Toggles the storage methods	230
SYstem	Displays the name of the foreign host operating system	231
TChinese	Sets the transfer type to TCHINESE	232
TSO	Passes a TSO command to the local host TSO environment	234
TYpe	Specifies the transfer type	235
UCs2	Changes the data transfer type to Unicode UCS-2. UC is the minimum abbreviation for UCS2	240
User	Identifies you to a foreign host or changes your TSO user ID password	241

! Subcommand—Invoke an OS/390 UNIX System Services Function

Purpose

In an OS/390 UNIX environment, use the ! subcommand to invoke OS/390 UNIX functions.

Format

Parameters

shell_command
Specifying the ! subcommand with a shell command enables you to invoke OS/390 UNIX, perform the subcommand, and return to the FTP environment.

If no shell command is specified, the ! subcommand invokes OS/390 UNIX. There you can specify any number of shell commands before typing `exit` to return to the FTP environment.

ACCOUNT Subcommand—Supply Account Information

Purpose

Use the ACCOUNT subcommand to supply account information to a host.

Format

Parameters

account_information
Specifies the account information required by the host. Refer to your foreign-host FTP server documentation for the information required by that host.

Usage

- The MVS FTP server does not require any account information.
- You may have to use the ACCOUNT subcommand when the foreign host requires passwords for read and write access to its files or data sets. If you are not prompted by the foreign host for the passwords, use the ACCOUNT subcommand to send these passwords to the foreign host.

APPEND Subcommand—Append a Local Data Set

Purpose

Use the APPEND subcommand to append a local data set to a remote host.

Format

Parameters

local_data_set
The name of the data set on your local host to be appended.

destination_file
The name of the file on the remote host to which your data set is appended. If the destination file does not already exist at the remote host, a new file is created. If the server is an OS/390 UNIX server, the local file can be appended to as an HFS or an MVS data set.

Examples

In the following example, an FTP command is issued from MVSXA2 to MVSXA3. MVSXA2 has a data set MVSUSER.FTP.EXAMPLE with one member. The member, APPEND01, contains:

```
;
; THIS FILE ORIGINALLY RESIDED IN MVSXA2, AND
; WILL BE APPENDED TO A FOREIGN FILE IN MVSXA3.
;
```

MVSXA3 has a data set, MVSUSER.FTP.EXAMPLE, with one member, APPEND02. The member contains:

```
;
; THIS FILE ORIGINALLY RESIDED IN MVSXA3, AND
; WILL BE USED TO RECEIVE ANOTHER FILE FROM MVSXA2.
;
```

```
  User:  append
System:    Usage: APPEND localfile foreignfile
           Command:

  User:  lpwd
System:    Local directory is MVSUSER.
           Command:

  User:  append 'mvsuser.ftp.example(append01)' 'mvsuser.ftp.example(append02)'
 System:   >>>SITE FIXrecfm 128 Lrecl=128 Recfm=FB BlockSize=6144
           200 Site command was accepted
           >>>PORT 1,1,2,2,4,16
           200 Port request OK.
           >>>APPE 'mvsuser.ftp.example(append02)'
           125 Appending to data set MVSUSER.FTP.EXAMPLE(APPEND02)
           250 Transfer completed successfully.
           520 bytes transferred in 1.100 seconds.
           Transfer rate 0.47 Kbytes/sec.
           Command:
```

Usage

- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the APPEND subcommand, FTP may

truncate data records and you might lose data. If the data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.

- If the remote host is an MVS or VM host, and if the data set on the remote host has a fixed-record format, the format and record length of the data set on the remote host are always preserved.
- Records from the data set on your local host are truncated or padded with blank spaces when necessary.
- To append to a file on a remote host, you must define a working directory on that host, and you must have write privileges to the files in that directory.

Context

- See “CD Subcommand—Change the Directory on the Remote Host” on page 112 for more information about working with current directories.
- See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about naming conventions.
- APPEND can be used with the PROXY subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

ASCII Subcommand—Change the Data Transfer Type to ASCII

Purpose

Use the ASCII subcommand to change the data transfer type to ASCII.

Format

Parameters

There are no parameters for this subcommand.

Usage

The ASCII transfer type is used to transfer data to or from an ASCII host. ASCII is the default transfer type.

Context

For more information about transfer methods, see Table 12 on page 52.

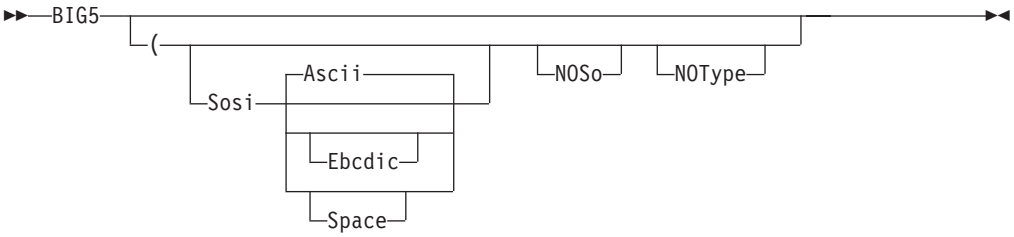
BIG5 Subcommand—Change the Data Transfer Type to BIG5

Purpose

Use the BIG5 subcommand to change the data transfer type to BIG5.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by the one of the following parameters – ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' andX'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (this is, data with no SBCS characters) and that the data is to be transferred to or from EBCDIC DBCS data that contains no shift-out or shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

- The BIG5 client subcommand is equivalent to the TYPE B 8 command.
- The minimum abbreviation for BIG5 is BIG.

Context

See “FTP with DBCS Support” on page 97 for more information.

BINARY Subcommand—Change the Data Transfer Type to Image

Purpose

Use the BINARY subcommand to change the data transfer type to image (binary).

Format

Parameters

There are no parameters for this subcommand.

Usage

Use the image transfer type to transfer data from an ASCII host to an EBCDIC host. When using the image transfer type, data is sent as contiguous bits packed into 8-bit bytes. Use the image transfer type for efficient storage and retrieval of data sets or files, and for the transfer of binary data.

Context

For more information about data transfer methods, see Table 12 on page 52.

BLOCK Subcommand—Set the Block Data Transfer Mode

Purpose

Use the BLOCK subcommand to set the data transfer mode to block mode. This is equivalent to specifying the MODE B subcommand. See “MODE Subcommand—Set the Data Transfer Mode” on page 174 for more information.

Format

Parameters

There are no parameters for this subcommand.

CD Subcommand—Change the Directory on the Remote Host

Purpose

Use the CD subcommand to change the working directory or file group on the remote host.

Format

Parameters

directory
Specifies the name of a file directory, a fully qualified data set, or a prefix on the remote host.

Examples

Changing the Directory of an MVS FTP Server: If the remote host is using TCP/IP for MVS, *directory* specifies either a common prefix for a group of data sets or the qualifiers of a Partitioned Data Set (PDS). If the remote server is the MVS OS/390 UNIX FTP server, *directory* can specify an HFS name.

When the CD subcommand is issued, the *directory* specified is appended to the current working directory. For example, if the current working directory is `TCPUSR14.TEST`, and you issue the CD subcommand:

```
CD FILES
```

the new working directory becomes `TCPUSR14.TEST.FILES`.

To override the existing directory rather than append to the directory, issue the *directory* parameter in single quotation marks. For example, if the current working directory is `TCPUSR14.TEST`, and you issued the CD subcommand:

```
CD 'FTP.FILES'
```

the new working directory would be `FTP.FILES`. If the subdirectory name contains white space, such as ″NEW SUBDIRECTORY″, then the syntax for the CD command would be as follows:

```
CD 'NEW SUBDIRECTORY'
```

The command syntax must specify the full subdirectory name (including the blank) delimited within single quotes.

If a PDS exists with the exact name of the current working directory, FTP considers the working directory to be that PDS. Otherwise, FTP considers the working directory to be a common prefix qualifier for sequential data sets.

If a PDS exists with the same name as the current working directory, but you want the current working directory to be treated as a common prefix for sequential data sets, specify the working directory with a period (.) at the end. For example, if a PDS named `TCPUSR14.TEST` exists, the subcommand:

```
CD 'TCPUSR14.TEST'
```

makes the PDS TCPUSR14.TEST the current working directory. A subsequent PUT of file name1 adds a member name1 to the TCPUSR14.TEST PDS. In contrast, the statement:

```
CD 'TCPUSR14.TEST.'
```

makes the current working directory, TCPUSR14.TEST., a prefix for sequential data sets. A subsequent PUT command used to copy data set name1 would create the sequential data set TCPUSR14.TEST.name1.

To back up one level of the current working directory, issue the CD subcommand with two periods (..) at the end. For example, if the working directory is jones.source, the statement,

```
CD ..
```

makes jones. the working directory. You can also use the CDUP command to back up one level of the current working directory. See "CDUP Subcommand—Change to the Parent of the Working Directory" on page 115 for more details.

The following sample commands and responses are displayed as a result of the CD subcommand.

For an MVS data set:

```
cd hsmtest
 >>>CWD hsmtest
 250 "'USER17.HSMTEST.'" is working directory name prefix.
 Command:
```

For an HFS file:

```
cd '/u/user121/A/B/C'
  >>>CWD '/u/user121/A/B/C'
 250 HFS directory /u/user121/A/B/C is the current working directory
  Command:
```

Changing the Directory of a VM FTP Server: If the remote host is using TCP/IP for VM, the directory can be specified in either of the following ways:

- *user_id minidisk_address*
- *user_id.minidisk_address*

For example, to access the 191 minidisk of user ID jones, enter one of the following:

- jones 191
- jones.191

Testing Throughput with *DEV.NULL: If you have an MVS FTP server, you can use the PUT command to copy a large number of files (or one large file) without having the files actually stored on an MVS server system. This is useful for testing purposes. Thus, you do not have to worry about allocating the disk space on the server system.

To use this function, first change the working directory to *DEV.NULL by using one of the following commands:

- CD *DEV.NULL
- CWD *DEV.NULL

This affects the working directory only for PUT commands. You then use the PUT command to copy the file to the server system. The input data set must be valid, and the output file can either be new or already exist. In either case, the file is not actually stored.

The following response shows information such as the number of bytes transferred and the rate of transfer.

To end the use of the *dev.null directory for PUT commands, issue another CHANGE Directory command.

```
 Command:
cd *dev.null
 >>>CWD *dev.null
 250-Working directory for PUT is NULL Device;
 250 for GET is HFS directory /u/user31
 Command:
 put a.b a.bbbbb
 >>>SITE VARrecfm Lrecl=128 Recfm=VB BlockSize=6144
 200 Site command was accepted
 >>>PORT 14,0,0,0,4,14
 200 Port request OK.
 >>>STOR a.bbbbb
 125 Storing data set in the Null directory (*dev.null).
 250 Transfer completed successfully.
 82 bytes transferred in 0.245 seconds. Transfer rate 0.33 Kbytes/sec.
 Command:
 quit
 >>>QUIT
 221 Quit command received. Goodbye.
```

Usage

You can also use the CWD and CW subcommands to change the current working directory. These subcommands are synonyms of the CD subcommand.

CDUP Subcommand—Change to the Parent of the Working Directory

Purpose

Use the CDUP subcommand as a special case of the CD subcommand to change the working directory to the next higher directory level. You can use it to simplify the implementation of programs for transferring directory trees between operating systems that have different syntaxes for naming the parent directory.

Format

Parameters

There are no parameters for this subcommand.

Examples

Change the working directory to the next higher directory level:

```
cd 'a.b.c.d'
  >>>CWD 'a.b.c.d'
 257 "'A.B.C.D.'" is working directory name prefix.
  Command: pwd

  >>>PWD
 257 "'A.B.C.D.'" is working directory
  Command: cdup

  >>>CDUP
 257 "'A.B.C.'" is working directory name prefix.
  Command: pwd

  >>>PWD
 257 "'A.B.C.'" is working directory
  Command: cdup

  >>>CDUP
 257 "'A.B.'" is working directory name prefix.
  Command: pwd

  >>>PWD
 257 "'A.B.'" is working directory
```

Change the working directory to the next higher directory level for an HFS file:

```
cd '/u/user121/A/B/C'
  >>>CWD '/u/user121/A/B/C'
 250 HFS directory /u/user121/A/B/C is the current working directory
  Command: pwd

  >>>PWD
 257 "/u/user121/A/B/C" is the HFS working directory
  Command: cdup

  >>>CDUP
 250 HFS directory /u/user121/A/B is the current working directory
  Command: pwd

  >>>PWD
 257 "/u/user121/A/B" is the HFS working directory
  Command: cdup

  >>>CDUP
  250 HFS directory /u/user121/A is the current working directory
  Command: pwd

  >>>PWD
 257 "/u/user121/A" is the HFS working directory.
 Command:
```

CLOSE Subcommand—Disconnect from a Remote Host

Purpose

Use the CLOSE subcommand to disconnect from the remote host and remain in FTP.

Format

Parameters

There are no parameters for this subcommand.

Usage

The FTP session remains active on your local host, but the session to the remote host is terminated. You can use the OPEN subcommand to establish a new session with either the same or a different remote host. If you establish a new session with the same remote host, values set by the SITE subcommand during the previous session are cleared. The remote host default values for the parameters of the SITE subcommand are used for the new session.

Context

- See “OPEN Subcommand—Connect to the FTP Server” on page 178 for information about the OPEN subcommand.
- CLOSE can be used with the PROXY subcommand to close a secondary control connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

COMPRESS Subcommand—Set the Compressed Data Transfer Mode

Purpose

Use the COMPRESS subcommand to set the data transfer mode to compressed mode. This is equivalent to specifying the MODE C subcommand. See “MODE Subcommand—Set the Data Transfer Mode” on page 174 for more information.

Format

Parameters

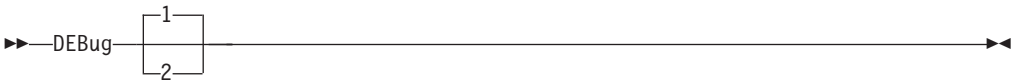
There are no parameters for this subcommand.

DEBUG Subcommand—Set Internal Debug Options

Purpose

Use the DEBUG subcommand to enable or disable internal debugging. You can enable two different levels of debugging: 1 (status messages), or 2 (detailed trace information).

Format



Parameters

n Specifies the level of debugging to toggle on: 1 or 2. Level 1 provides the same level of support as in previous TCP/IP releases; FTP displays in standard output or OUTPUT each command that is sent to the foreign host and the response that is received from the foreign host. Level 2 provides detailed trace information, primarily in the area of data flows.

Specifying DEBUG with no parameters toggles the debugging function. If the debugging toggle is off, DEBUG or DEBUG=1 sets the first level of debugging on. The following table summarizes the behavior of the DEBUG subcommand.

If This Level of Debugging is Set:	Specifying DEBUG Sets:	Specifying DEBUG 1 Sets:	Specifying DEBUG 2 Sets:
Toggle on (same as DEBUG=1)	Toggle off	DEBUG level 1	DEBUG level 2
DEBUG 1	Toggle off	DEBUG level 1	DEBUG level 2
DEBUG 2	Toggle off	DEBUG level 1	DEBUG level 2
Toggle off	Toggle on at DEBUG level 1	DEBUG level 1	DEBUG level 2

Examples

The following example shows sample client traces with DEBUG 1 and DEBUG 2 enabled.

```
************************************************************************
*                                                                      *
* Samples of client trace with debug 1 and debug 2                     *
*                                                                      *
* Notes:                                                               *
*                                                                      *
*  1. The sequence ". . . . . . . . . " indicates that trace           *
*     entries are deleted from the sample.                             *
*  2. In this trace, the difference is that for debug 2, the           *
*     text units for dynamic allocation are displayed as part          *
*     of extended trace data.                                          *
*                                                                      *
************************************************************************

************************************************************************
*                                                                      *
* Client trace with debug 1                                            *
*                                                                      *
```

```
*********************************************************************
 Command: get a abc
 . . . . . .
 CG0130 get: routine entered
 CG1127 rcvFile: routine entered...
 MV0772 seq_create_file: dsn=USER33.TEST.ABC
 MV0779 seq_create_file: SMS is active
 MV1361 seq_create_file: ddname=SYS00016
 MV1886 return_real_name: RC = 0 ERROR = 0 INFO = 0
 MV1898 return_real_name: sfip->dsname = USER33.TEST.ABC
 MV1373 seq_create_file: sfip->filename = USER33.TEST.ABC
 MV1396 seq create file: data set has recfm=50, lrecl=3000, blksize=5000
 CG1560 rcvFile: FASTIO pending for store_type = N
 CG1562 rcvFile: ... recfm=VB, lrecl=3000, blksize=5000
 CG1648 rcvFile: request FASTIO with recfmU override for record format VB
 MF0314 seq_open_file: setenv for _EDC_ZERO_RECLEN
 MF0364 seq_open_file: Using BSAM, O, SYS00016, , 20, 0
 SC0318 initDsConnection: data socket is 2
 SC0556 sendCmd: routine entered
 . . . . . .
*********************************************************************
*                                                                   *
* Client trace with debug 2 -- ma                                   *
*                                                                   *
*********************************************************************

 Command: get a abc
 . . . . . .
 CG0130 get: routine entered
 CG1127 rcvFile: routine entered...
 MV0772 seq_create_file: dsn=USER33.TEST.ABC
 MV0779 seq_create_file: SMS is active
 MV0832 TU_DSN  info...
 0008F32C  E34BC1C2  C3000000  00000000  00000000   *T.ABC...........*
 0008F33C  00000000  00000000  00000000  00000000   *................*
 0008F34C  00000000                                 *....           *
 MV0843 TU_STATS info...
 0008F208  00040001  00010400                       *........       *
 MV0850 TU_DISP  info...
 0008F210  00050001  00010200                       *........       *
 MV0857 TU_COND  info...
 0008F218  00060001  00010200                       *........       *
 MV0950 TU_PRIME  info...
 0008F17C  000A0001  00030000  0100                 *..........     *
 MV0959 TU_SECND  info...
 0008F186  000B0001  00030000  0101                 *..........     *
 MV0992 TU_SPACE  info...
 0008F1F8  00070000  E2C5D840  0000                 *....SEQ ..     *
 MV1180 TU_RECFM  info...
 0008F1D0  00490001  00015000                       *......&.       *
 MV1192 TU_LRECL  info...
 0008F1C0  00420001  00020BB8                       *........       *
 MV1201 TU_BLKSIZE  info...
 0008F1C8  00300001  00021388                       *.......h       *
 MV1241 TU_OPTCD    info...
 0008F1B8  00450001  00012000                       *........       *
 MV1287 TU_DSORG  info...
 0008F1D8  003C0001  00024000                       *...... .       *
 MV1296 TU_RLSE   info...
 0008F146  000D0000                                 *....           *
 MV1330 Dynamic allocation parameter list
 0008F308  14012000  00000000  0008F424  00000000   *..........4.....*
 0008F318  00000000                                 *....           *
 MV1361 seq_create_file: ddname=SYS00017
 MV1886 return_real_name: RC = 0 ERROR = 0 INFO = 0
 MV1887 RN PA=
```

```
000AA46C  14070000  00000000  000AA480  00000000   *.........u.....*
000AA47C  00000000                                 *....            *
MV1888 RN DD=
000AA456  00010001  0008E2E8  E2F0F0F0  F1F7856D   *......SYS00017e_*
000AA466  86899385  7A40                         *file:           *
MV1889 RN DSN=
000AA494  00050001  000FE4E2  C5D9F3F3  4BE3C5E2   *......USER33.TES*
000AA4A4  E34BC1C2  C340405C  4B4B4B4B  40404040   *T.ABC  *....    *
000AA4B4  40404040  40404040  5C150000  00000000   *        *.......*
000AA4C4  00000000                                 *....            *
MV1373 seq_create_file: sfip->filename = USER33.TEST.ABC
MV1396 seq create file: data set has recfm=50, lrecl=3000, blksize=5000
CG1560 rcvFile: FASTIO pending for store_type = N
CG1562 rcvFile: ... recfm=VB, lrecl=3000, blksize=5000
CG1648 rcvFile: request FASTIO with recfmU override for record format VB
MF0314 seq_open_file: setenv for _EDC_ZERO_RECLEN
MF0364 seq_open_file: Using BSAM, O, SYS00017, , 20, 0
SC0299 initDsConnection: routine entered
SC0318 initDsConnection: data socket is 2
SC0556 sendCmd: routine entered
. . . . . .
```

Usage

By default, DEBUG is off unless the TRACE parameter was specified on the FTP command.

DELETE Subcommand—Delete Files

Purpose

Use the DELETE subcommand to delete a file on the remote host.

Format

Parameters

foreign_file
Specifies the name of the file to be deleted on the remote host.

Context

See “Appendix A. Specifying Data Sets and Files” on page 545, for information about file naming conventions.

DELIMIT Subcommand—Display the File Name Delimiter

Purpose

Use the DELIMIT subcommand to display the character that is used as the delimiter between the file name and the file type.

Format

Parameters

There are no parameters for this subcommand.

Usage

- The DELIMIT subcommand should be used for information purposes only.
- You cannot change which character is used as the delimiter.

DIR Subcommand—Obtain a List of Directory Entries

Purpose

Use the DIR subcommand to obtain a list of directory entries or a list of files in a file group on the remote host, or a list of the members of the partitioned data set, as well as auxiliary information about the files.

Format

Parameters

name
Specifies the name of the directory or file group. The default is the current directory or file group.

DISK
Stores the results of the DIR subcommand as data set FTP.DIROUTP in the local current working directory. If the local current working directory is an MVS PDS, the member DIROUTP is stored. If the local current working directory is an HFS directory, the results are stored in a file named diroutp.

Examples

- List the data sets with a common high-level qualifier as the current working directory:

```
>>>PORT 9,67,58,227,4,62
200 Port request OK.
>>>LIST
125 List started OK.
Volume Unit    Referred Ext UsedRecfm Lrecl BlkSz Dsorg Dsname
WRKLB2 3380   1997/07/02  1    1 F       80    80  PS  TEST.A
WRKLB2 3380   1997/07/09  4    7 F       80    80  PO  TEST.PDS
WRKLB2 3380   1997/07/09  2    3 VB      60   256  PO  TEST.PDSV
WRKLB2 3380   1997/05/09  1    1 F       80    80  PO  TEST.DATA
WRKLB2 3380   1997/07/01 16   17 V      200  3120  PS  TEST.XYZ
WRKLB2 3380   1997/07/01 16   17 VB     200 32000  PS  TEST.XYZ1
250 List completed successfully.
Command:
```

List the files for an HFS directory:

```
cd '/u/user121/ftp.example'

 >>>CWD '/u/user121/ftp.example'
 250 HFS directory /u/user121/ftp.example is the current working directory
 Command:
dir
 >>>PORT 9,67,112,25,4,61
 200 Port request OK.
 >>>NLST
 125 List started OK
 total 64
 -rw-r-----    1 USER121  SYS1      6720 Feb  7 18:48 append02
 -rw-r-----    1 USER121  SYS1      3360 Feb  6 18:51 file1
 -rw-r-----    1 USER121  SYS1      3883 Feb  6 18:51 file2
 -rw-r-----    1 USER121  SYS1      3883 Feb  6 18:51 file3
 -rw-r-----    1 USER121  SYS1      7277 Feb  6 18:51 file4
 -rw-r-----    1 USER121  SYS1      3360 Feb  6 18:51 file5
  250 List completed successfully.
 Command:
```

- List the members of a partitioned data set containing load modules:

```
cd 'tcpv3.sezaxawl'
 >>>CWD 'tcpv3.sezaxawl'
 257 "'TCPV3.SEZAXAWL'" partitioned data set is working directory.
 Command: dir

 >>>PORT 9,67,112,25,4,27
 200 Port request OK.
 >>>LIST
 125 List started OK.
  Name      Size   TTR   Alias-of AC --------- Attributes --------- Amode Rmode
 @XADFTT1  0009F0 000A11 TEXTTR   00 FO                               31    ANY
 @XADFTT2  0009F0 000A11 TEXTTR   00 FO                               31    ANY
 @XADFTT3  0009F0 000A11 TEXTTR   00 FO                               31    ANY
 @XATXDRA  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXDSA  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXIFA  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXIFI  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXPSA  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXSCH  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXSTF  003998 00090C TEXTPOP  00 FO                               31    ANY
 @XATXZSL  005500 00080F TEXTACTI 00 FO                               31    ANY
 ACOMMAND  001420 00020A          00 FO                               31    ANY
 AFORM     001AC0 00021F          00 FO                               31    ANY
 ***
```

- List the members of a partitioned data set from a text library:

```
cd 'tcpv3.tcpip.profiles'
 >>>CWD 'tcpv3.tcpip.profiles'
 257 "'TCPV3.TCPIP.PROFILES'" partitioned data set is working directory.
 Command: dir

 >>>PORT 9,67,112,25,4,32
 200 Port request OK.
 >>>LIST
125 List started OK.
 Name     VV.MM    Created       Changed     Size  Init   Mod   Id
TST6MV1   01.05 1997/06/26 1996/07/10 06:38    16    16     0 USER34
TST6MV2   01.08 1997/05/23 1996/07/03 12:49    16    17     0 USER34
TST6MV3   01.19 1997/05/23 1996/07/10 06:34    16    17     0 USER34
TST6021   01.04 1997/03/04 1996/07/08 09:17    15    15     0 USER34
TST6121   01.10 1997/05/23 1996/07/10 06:26    16    17     0 USER34
250 List completed successfully.
 ***
```

Usage

- To make a file group the current working directory, use the CD command. The method you use to specify a directory or file group is host-dependent.

- You can use special characters for pattern matching when specifying the *name*. These characters depend on the host FTP server.
- The DIR subcommand provides a complete list of directory entries and gives additional information about the files.

Context

- See “Appendix A. Specifying Data Sets and Files” on page 545, for more information about pattern matching and about specifying data sets and files.
- To get a list containing only the file names in a directory, use the LS subcommand (see “LS Subcommand—Obtain a List of File Names” on page 164).
- To make a file group the current working directory, see “CD Subcommand—Change the Directory on the Remote Host” on page 112.
- To change the local directory, see “LCD Subcommand—Change the Local Working Directory” on page 143.

EBCDIC Subcommand—Change the Data Transfer Type to EBCDIC

Purpose

The EBCDIC subcommand enables you to change the data transfer type to EBCDIC.

Format

Parameters

There are no parameters for this subcommand.

Usage

The EBCDIC transfer type is used to transfer data to or from an EBCDIC host.

Context

For more information about transfer methods, see Table 12 on page 52.

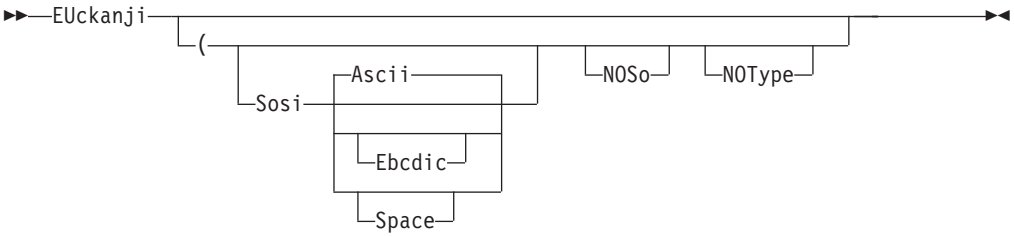
EUCKANJI Subcommand—Change the Data Transfer Type to EUCKANJI

Purpose

Use the EUCKANJI subcommand to change the data transfer type to Extended UNIX Code (EUC) kanji.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by one of the following parameters – ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (this is, data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The EUCKANJI client subcommand is equivalent to the TYPE B 2 server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

FILE Subcommand—Set the File Structure to File

Purpose

Use the FILE subcommand to set the file structure to file. This is equivalent to specifying the STRUCTURE F subcommand. See “STRUCTURE Subcommand—Set the File Structure” on page 229 for more information.

Format

Parameters

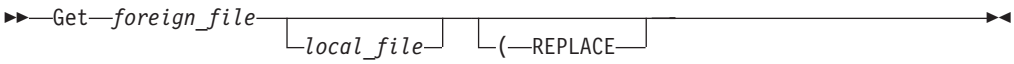
There are no parameters for this subcommand.

GET Subcommand—Copy Files

Purpose

Use the GET subcommand to copy a file from the remote host to your local host.

Format



Parameters

foreign_file
Specifies the name of the file to be retrieved from the remote host.

local_file
Specifies the name of the local file created as a result of the GET subcommand.

If the current local working directory is a PDS, *local_file* is the name of the member in the PDS. If the current local working directory is a data set prefix, the local file is a sequential data set with the *local_file* name appended to the current local working directory. If the current local working directory is an HFS directory, the local file is an HFS file in that directory.

You can override the usage of the current local working directory in the local file name by specifying *local_file* as a complete data set name enclosed in single quotation marks ('). If *local_file* is not specified, the *local_file* name is the same as the *foreign_file* name.

(REPLACE
Causes a data set on your local host to be overwritten, if it already exists. If the data set already exists, and you do not use the replace parameter, the existing data set is not overwritten. A message informing you of this is displayed.

If the data set already exists and you specify REPLACE, the data in the file is overwritten, but not reallocated. This means the local data set retains its existing characteristics.

Usage

- FTP uses either the characteristics of the local file, if it exists, or uses the values specified with the LOCSITE subcommand. Characteristics of the transmitted (foreign file) data set are unknown.

  When you use the GET subcommand, FTP may truncate data records and you might lose data, if:

  – You are creating a new data set at the client and the value of LRecl, as shown by the LOCSTAT command, is a value less than the LRecl of a received data set, FTP truncates the received data set.

  – The data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.

    You could also encounter truncated data records or lost data when you use GET with the REPLACE option.

Caution: A GET subcommand issued for an empty or nonexisting foreign file erases the contents of the existing local data set.

- If the name specified for *local_file* is not acceptable to your local host, the file is not transferred.
- To get a file from the remote host, you must have a defined working directory on that host and you must have read privileges to the files in this working directory.
- If the data set has been pre-located, you must specify DSORG=PS on the DCB statement in the JCL.

Context

- See “Appendix A. Specifying Data Sets and Files” on page 545, for more information about naming conventions.
- See “CD Subcommand—Change the Directory on the Remote Host” on page 112 and “ACCOUNT Subcommand—Supply Account Information” on page 104 for more information about working directories.
- GET can be used with the PROXY subcommand to transfer files from a host on a primary connection to a host on a secondary connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

GLOB Subcommand—Toggle Expansion of Metacharacters

Purpose

Use the GLOB subcommand to toggle globbing (the expansion of metacharacters in file names) for the MDELETE, MGET, and MPUT subcommands.

Format

Parameters

There are no parameters for this subcommand.

Examples

Assume that the files m1 and m1* exist in the directory /u/user33/mpp1.

```
 Command:
pwd
 >>> PWD
 257 "/u/user33/mpp1" is the HFS working directory

 Command:
lpwd
 Local directory name set to hierarchical file /u/user33

 Command:
prompt
 Interactive mode is off
```

```
 Command:
mget m1*
 >>> PORT 9,67,113,57,4,43
 200 Port request OK.
 >>> NLST m1*
 125 List started OK
 250 List completed successfully.
 >>> PORT 9,67,113,57,4,44
 200 Port request OK.
 >>> RETR m1
 125 Sending data set /u/user33/mpp1/m1
 250 Transfer completed successfully.
 200 bytes transferred in 0.050 seconds.  Transfer rate 4.00 ...
Kbytes/sec.
 >>> PORT 9,67,113,57,4,45
 200 Port request OK.
 >>> RETR m1*
 125 Sending data set /u/user33/mpp1/m1*
 250 Transfer completed successfully.
 200 bytes transferred in 0.020 seconds.  Transfer rate 10.00 ...
Kbytes/sec.
```

```
 Command:
delete /u/user33/m1
 >>> DELE /u/user33/m1
 250 /u/user33/m1 deleted.

 Command:
delete /u/user33/m1*
 >>> DELE /u/user33/m1*
 250 /u/user33/m1* deleted.

 Command:
glob
 Globbing off

 Command:
mget m1*
 >>> PORT 9,67,113,57,4,46
 200 Port request OK.
 >>> RETR m1*
 125 Sending data set /u/user33/mpp1/m1*
 250 Transfer completed successfully.
 200 bytes transferred in 0.010 seconds.  Transfer rate 20.00 ...
Kbytes/sec.
```

With globbing off, at most one file will match the pattern. Also, the NLST command is not sent to look for pattern matches.

Usage

GLOB acts as a toggle that turns metacharacter expansion on or off. By default, GLOB is on.

Context

For more information on globbing, refer to *MVS/ESA™ OS/390 UNIX System Services MVS User's Guide*.

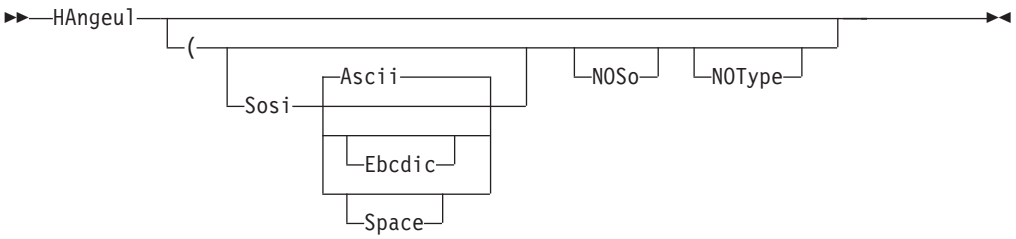
HANGEUL Subcommand—Change the Data Transfer Type to HANGEUL

Purpose

Use the HANGEUL subcommand to change the data transfer type to Hangeul.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by one of the following parameters – ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The HANGEUL client subcommand is equivalent to the TYPE B 5 server command.

Context

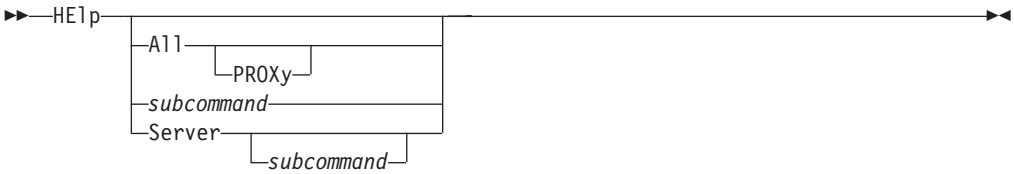
See “FTP with DBCS Support” on page 97 for more information.

HELP and ? Subcommands—Display Help Information

Purpose

Use the HELP command to get assistance with the FTP subcommands.

Format



Parameters

All
Displays a description of all subcommands.

subcommand
Displays a description of the specified subcommand. The subcommand name can be abbreviated to its minimum abbreviation.

Server
Displays the help that the foreign host offers for the specified subcommand.

If you do not specify a subcommand, FTP displays a list of the commands that the foreign host recognizes.

Usage

- If you enter the HELP subcommand without a parameter, you see the HELP FTP MENU, which lists the subcommands recognized by the FTP client and a description of the help information available.
- If you enter the ? subcommand by itself, you see introductory information about FTP.
-

Note: To receive help from a server on a secondary control connection, enter PROXY HELP SERVER. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

IBMKANJI Subcommand—Change the Data Transfer Type to IBMKANJI

Purpose

Use the IBMKANJI subcommand to change the data transfer type to IBM kanji.

Format

Parameters

(NOType
Suppresses sending of the TYPE command for host servers that do not support this data transfer type.

Usage

This subcommand causes no conversion to be performed on the transferred file. It has exactly the same effect as the EBCDIC TYPE command alias.

Context

See “FTP with DBCS Support” on page 97 for more information.

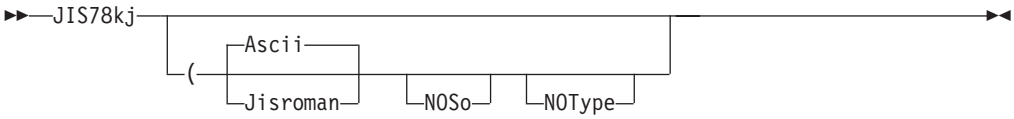
JIS78KJ Subcommand—Change the Data Transfer Type to JIS78KJ

Purpose

Use the JIS78KJ subcommand to change the data transfer type to JIS78KJ (1978 edition).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Ascii
Use ASCII shift-in escape sequence `ESC ( B` in the transferred data.

If neither ASCII nor JISROMAN is specified, the ASCII shift-in sequence is used.

Jisroman
Use JISROMAN shift-in escape sequence `ESC ( J` in the transferred data.

NOSo
Specifies that the data transferred is pure DBCS (data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

- The JIS78KJ or JIS78KJ (ASCII client subcommands are equivalent to the TYPE B 4 A server command.
- The JIS78KJ (JISROMAN client subcommand is equivalent to the TYPE B 4 R server command.
- The JIS78KJ (JISROMAN NOSO client subcommand is equivalent to the TYPE B 4 R N server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

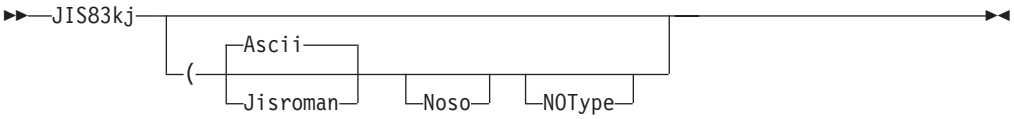
JIS83KJ Subcommand—Change the Data Transfer Type to JIS83KJ

Purpose

Use the JIS83KJ subcommand to change the data transfer type to JIS83KJ (1983 edition).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Ascii
Use ASCII shift-in escape sequence `ESC ( B` in the transferred data.

If neither ASCII nor JISROMAN is specified, the ASCII shift-in sequence is used.

Jisroman
Use JISROMAN shift-in escape sequence `ESC ( J` in the transferred data.

NOSo
Specifies that the data transferred is pure DBCS (this is, data with no SBCS characters) and that the data is to be transferred to/from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

- The JIS83KJ or JIS83KJ (ASCII client subcommands are equivalent to the TYPE B 3 A server command.
- The JIS78KJ (JISROMAN client subcommand is equivalent to the TYPE B 3 R server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

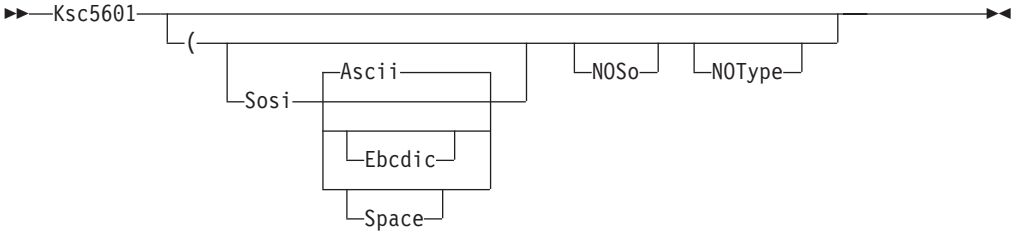
KSC5601 Subcommand—Change the Data Transfer Type to KSC-5601

Purpose

Use the KSC5601 subcommand to change the data transfer type to KSC-5601.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (this is, data with no SBCS characters) and that the data is to be transferred to/from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The KSC5601 client subcommand is equivalent to the TYPE B 6 server command. See “FTP with DBCS Support” on page 97 for more information.

LCD Subcommand—Change the Local Working Directory

Purpose

Use the LCD subcommand to change the current working directory on the local host.

Format

In a TSO Environment:

In an OS/390 UNIX System Services Environment:

Parameters

qualifier
Specifies either a common prefix for a group of sequential data sets or the qualifiers of a PDS.

Note: In an OS/390 UNIX environment, you can omit the qualifier on the LCD subcommand. Doing so changes the current working directory to your home directory. If you do not have a home directory, the working directory is not changed, and no message is issued.

Examples

- Change the local current working directory:

```
lcd ftp.test1

 Local directory name set to partitioned data set USER14.FTP.TEST1.
 Command:
```

- When the LCD subcommand is issued, *qualifier* is appended to the current local working directory. For example, if the current local working directory is `TCPUSR14.TEST` and you issue the LCD subcommand `LCD FILES`, the new working directory becomes `TCPUSR14.TEST.FILES`.
- To override the existing directory rather than append to the directory, issue the *qualifier* in single quotation marks. For example, if the current local working directory is `TCPUSR14.TEST` and you issued the LCD subcommand `LCD 'FTP.FILES'`, the new working directory is `FTP.FILES`.
- If a PDS exists with the exact name of the current local working directory, FTP considers the working directory to be that PDS. Otherwise, FTP considers the working directory to be a common prefix qualifier for sequential data sets.

  If a PDS exists with the same name as the current local working directory, but you want the current local working directory to be treated as a common prefix for sequential data sets, specify the working directory with a period (.) at the end. For example, if a PDS named `TCPUSR14.TEST` exists, the subcommand `LCD 'TCPUSR14.TEST'` makes the PDS `TCPUSR14.TEST` the current local working directory. A subsequent GET command used to copy data set `name1` would add

the member `name1` to the `TCPUSR14.TEST` PDS. In contrast, the statement `LCD 'TCPUSR14.TEST.'` would make the current local working directory `TCPUSR14.TEST.`, a prefix for sequential data sets. A subsequent GET command used to copy data set `name1` would create the sequential data set `TCPUSR14.TEST.name1`.

- To back up one level of the current local working directory, issue the LCD subcommand with two periods (..) at the end. For example, if the working directory is `jones.source`, the statement `LCD ..`makes `jones.` the working directory.

Usage

When you enter an FTP session, the working directory on the local host is set according to the environment in which the FTP client is invoked: $HOME in OS/390 UNIX, your MVS user ID in TSO.

LMKDIR Subcommand—Create a Directory on the Local Host

Purpose

Use the LMKDIR subcommand to create a PDS (or HFS directory) on the local host. This subcommand provides you with an easy way to create a directory in the local host for data transfer.

Format

Parameters

data_set
Specifies the name of the directory to be created.

Examples

In this example, before LMKDIR is issued, the local host had the following data sets:

- MVSUSER.ISPF.ISPPROF
- MVSUSER.JCL.CNTL
- MVSUSER.SMFTEST
- MVSUSER.TCPIP.DATA

```
  User:  ftp 1.1.2.3
System:    IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           220-EZAFTSRV IBM CS/390 V2R8 at EMU.ABC.OZ, 17:04:12 on 08/03/97
           220 Connection will close if idle for more than 5 minutes.
           NAME (<host>:tsouserid):

  User:  mvsuser
System:    >>>USER mvsuser
           331 Send password please.
           Password:
           >>>PASS ********
           230 MVSUSER is logged on.
           Command:
```

```
  User:  lpwd
System:    Local directory is MVSUSER.
           Command:
  User:  lcd ftp
System:    Local directory name set to MVSUSER.FTP.
           Command:
  User:  lmkdir example
System:    MVSUSER.FTP.EXAMPLE created.
           Command:
```

`MVSUSER.FTP.EXAMPLE` has now been created. You can get the same result directly with the `LMKDIR 'MVSUSER.FTP.EXAMPLE'` command.

After the LMKDIR was issued, the local host had the following data sets under MVSUSER:

- MVSUSER.FTP.EXAMPLE
- MVSUSER.ISPF.ISPPROF
- MVSUSER.JCL.CNTL

- MVSUSER.SMFTEST
- MVSUSER.TCPIP.DATA

Usage

- FTP provides no subcommand to display a list of local directory entries. You should use TSO ISPF facility to check whether the directory is created by the LMKDIR subcommand.
- If you are running FTP in an OS/390 UNIX environment, you can use the ! command to check the status of HFS directories.
- The *data_set* value is appended to the local current working directory to form the name of the created PDS. To override the local current working directory, specify:

  '*data_set*'

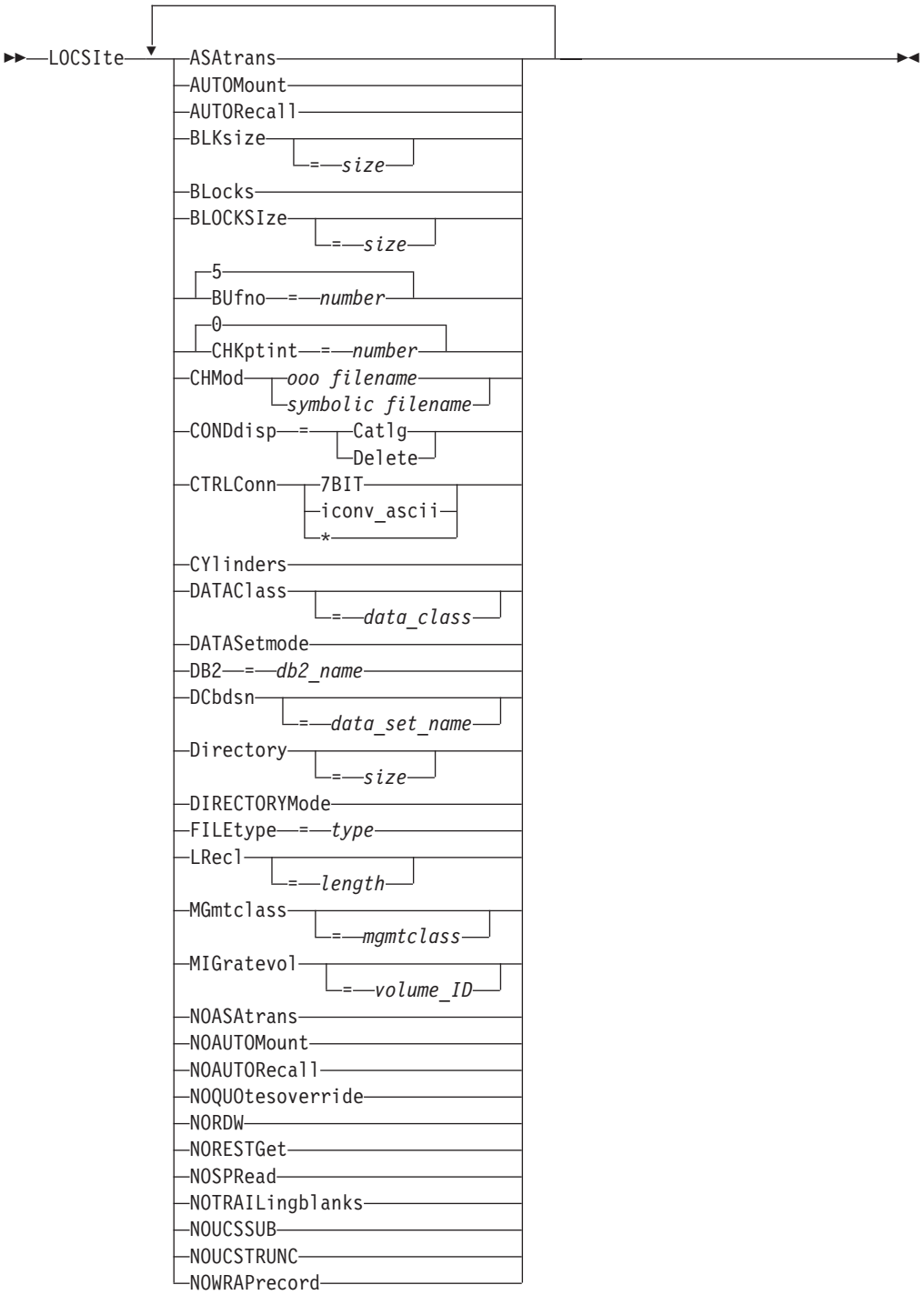
  The data set characteristics of the newly allocated PDS are determined by the settings of the local site variables. See “LOCSITE Subcommand—Specify Site Information to the Local Host” on page 147 and “LOCSTAT Subcommand—Display Local Status Information” on page 160 for information on setting and displaying the data set characteristics.

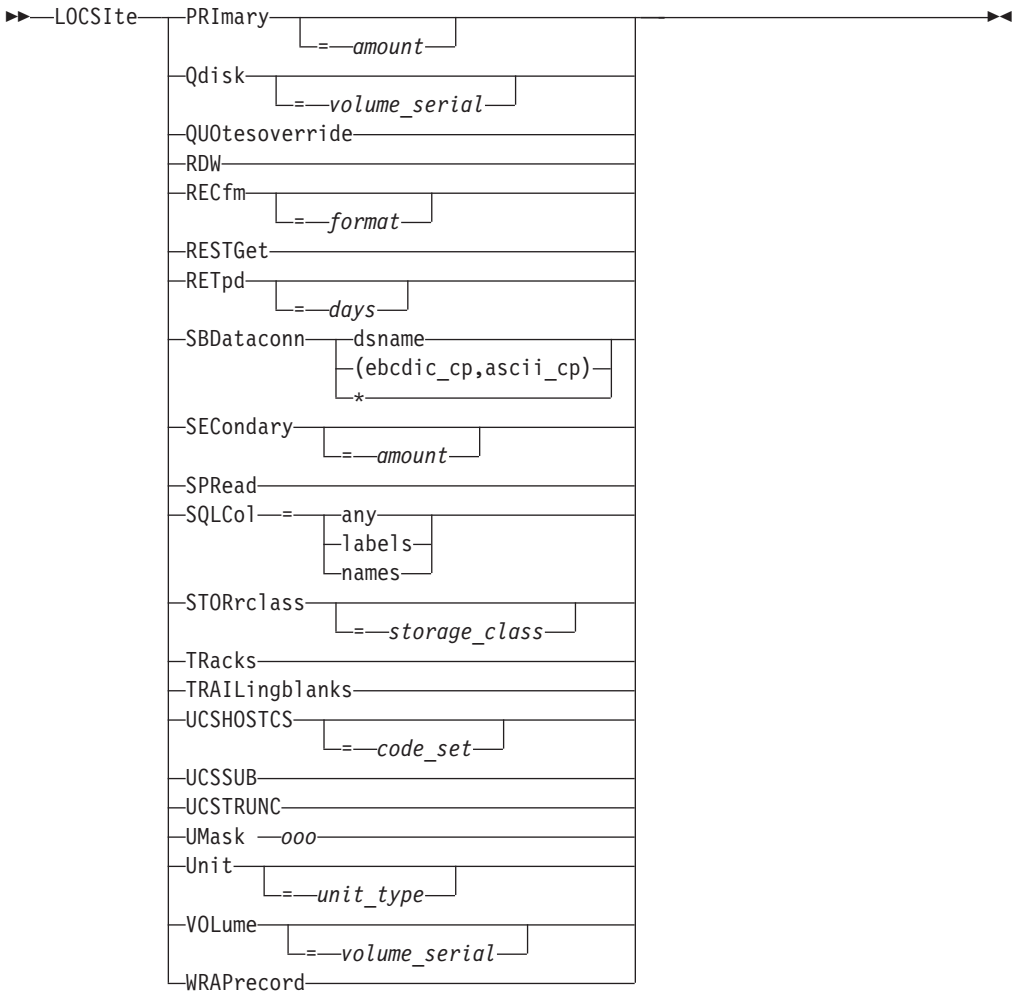
LOCSITE Subcommand—Specify Site Information to the Local Host

Purpose

Use the LOCSITE subcommand to specify information that is used by the local host to provide services specific to that host system.

Format





Parameters

ASAtrans
Permits the FTP client to interpret characters in the first column of ASA files being transferred as print control characters.

AUTOMount
Permits automatic mounting of volumes for data sets on volumes that are not mounted. If AUTOMount is specified and an unmounted volume is needed, a message is automatically issued to the MVS operator console requesting that the volume be mounted. The MVS operator must then mount the volume and reply to the message before FTP can proceed.

AUTORecall
Permits automatic recall of migrated data sets.

BLKsize
Specifies the block size of a newly allocated data set. BLKsize is functionally equivalent to BLOCKSIze. The BLOCKSIze parameter is obsolete, but it is accepted to provide compatibility with previous releases of OS/390 TCP/IP.

When specified without a *size*, no block size is used when allocating the new data set. When specified without a *size*, the equal sign (=) is optional.

Specify BLKsize with no value if you are also specifying DATAClass=*data_class* and you want the SMS data class to provide the BLKsize value, or if you are specifying DCbdsn=*data_set_name* and you want to use the blocksize from the DCBDSN data set. If BLKsize=*size* is specified with either the DATAClass or DCbdsn parameter, the value specified by the LOCSite BLKsize parameter overrides the DATAClass or DCbdsn blocksize.

Notes:

1. If you specify BLKsize without a size, FTP does not specify the block size when allocating new data sets.
2. Be especially careful specifying both BLKsize= and Blocks. While there are conditions where this is tolerated, if a valid BLKsize cannot be determined, the data set will not be created when the allocation is attempted.

size
: Specifies the block size of a newly allocated data set. The valid range is 0–32760.

  BLKsize=0 is a special case. When BLKsize=0 is specified, the operating system attempts to determine a blocksize for the new data set. FTP does not create the new data set unless the system is able to establish a nonzero blocksize.

BLocks
: Specifies that primary and secondary space allocations are in blocks.

  If both PRImary and SECondary are specified as zero, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the BLocks parameter is ignored.

BLOCKSIze
: Specifies the block size of a newly allocated data set. BLOCKSIze is functionally equivalent to BLKsize. BLOCKSIze is obsolete but it is accepted to provide compatibility with previous releases of OS/390 TCP/IP. See the BLKsize parameter for more information.

BUfno
: Specifies the number of access method buffers that is used when data is read from or written to a data set. The valid range is 1–35. The default value is 5.

CHKptint
: Specifies the checkpoint interval for the sending site in a file transfer request. If the checkpoint interval is zero, no checkpointing occurs and no marker blocks are transmitted. The default value is zero.

  CHKptint must be set by LOCSITE when the client site is the sending site for the file transfer (PUT command). Do not increase the checkpoint interval above zero for a PUT unless the receiving site supports the RESTART command and is able to process checkpoint markers in the file transfer data stream.

  CHKptint is valid only for block and compressed modes of data transfer.

  number
  : Specifies the checkpoint interval for the sending site in a file transfer request. This value is used to determine when checkpoint marker blocks are to be transmitted so that transmission can be restarted based on the information in the last marker.

A large checkpoint interval means that a large amount of data is sent between markers and therefore few markers are sent. A smaller checkpoint interval means that less data is sent in between markers and therefore more markers are sent.

The costs involved with using a nonzero checkpoint interval are:

- The markers themselves are transmitted, which means more bytes are being sent across the network (approximately 44 bytes per marker).
- Additional packets and acknowledgments may be required. For example, when the MVS client PUTs a file, a reply packet is sent from the server to the client and then the client must acknowledge that packet.

To estimate the appropriate checkpoint interval, use the following formula. You need to know the record length of the file you are transferring and how much data you think can be transmitted reliably.

```
           amount of data in interval
CHKPTINT = --------------------------
           record length of the file
```

Do not execute a CHKptint more often than once every 200KB. For example, if the file you are transferring has 80-byte records:

```
CHKPTINT = 200KB / 80 bytes
         = 200 * 1024 bytes / 80 bytes
         = 2560
```

CHMod

Changes the permission bits for a file.

ooo filename

ooo is the octal mask and *filename* is the name of the file whose mask is to be changed.

symbolic filename

symbolic is a value in the format accepted by the OS/390 UNIX CHMOD command (for example, a+r)

Note: Only *r, w,* and *x* permission bits are accepted in *symbolic*.

If *filename* does not begin with a slash character (/), it is appended to the current working directory. If *filename* does begin with a slash character (/), it is interpreted as a complete directory name.

The file name specified must be an HFS file name. The setting of QUOtesoverride is ignored and all quotes are treated as part of the file name.

The CHMOD keyword must be the only or last keyword on a LOCSITE subcommand.

CONDdisp

Specifies the disposition of the data set if a retrieve operation for a new data set ends before all of the data is written.

Catlg

Specifies that a data set is kept and cataloged when an FTP file transfer ends prematurely.

Delete
Specifies that a data set is deleted when an FTP file transfer ends prematurely.

Delete is ignored if the file transfer failed as a result of the FTP client being terminated or if the client has received checkpoint information during data transfer.

CTRLConn
Specifies the ASCII code page to be used for control connections. The valid subcommands are:

```
LOCSITE CTRLConn=7BIT
LOCSITE CTRLConn=iconv_ascii
LOCSITE CTRLConn=*
```

7BIT
Indicates 7-bit ASCII is to be used.

iconv_ascii
Is a name recognized by iconv to indicate an ASCII code page. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

***** Indicates that the ASCII used at initialization is to be used.

CYlinders
Specifies that primary and secondary space allocations are in cylinders.

If both PRImary and SECondary are specified as zero, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the CYlinders parameter is ignored.

DATAClass
Specifies the SMS data class, as defined by your organization, for the target host. Specifying DATAClass with no parameter value cancels the dataclass specification. The equal sign (=) is optional in this case.

See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

data_class
Specifies the SMS data class, as defined by your organization, for the target host. If values are specified for any of the following LOCSITE parameters, the values specified by the LOCSITE parameter overrides the value specified in the SMS dataclass:

- BLKsize
- Directory
- LRecl
- PRImary
- RECfm
- RETpd
- SECondary

If the DCbdsn parameter is specified on the LOCSITE subcommand, the LRecl, RECfm, BLOCKSIze, and RETpd (if specified) of the DCBDSN data set overrides the values specified in the data class.

If the MGmtclass parameter is specified on the LOCSITE subcommand, and the requested management class specifies a retention period, the retention period value of the management class may override the retention period value of the dataclass.

DATASetmode
Specifies that all the data set qualifiers below the current directory are treated as entries in the directory (disables DIRECTORYMode).

DB2
Specifies the name of the DB2 subsystem.

db2_name
The name of the DB2 subsystem.

DCbdsn
Specifies the name of the MVS data set to be used as a model for allocation of new data sets. Specifying DCbdsn with no parameter value cancels the DCbdsn specification.

data_set_name
Specifies the name of the data set. The file name must be an MVS data set name. HFS file names are not allowed on the DCbdsn parameter. The setting of QUOtesoverride is ignored. If the file name is enclosed in single quotes, it overrides the current working directory; otherwise it is appended to the current working directory.

Notes:

1. Specify LOCSITE RECfm, LRecl, and BLKsize parameters with no values to allow characteristics from the model DCB to be used.
2. To override the model characteristics of RECfm, LRecL, BLKsize, or RETpd, specify a value on the LOCSITE command.
3. If MGmtclass is specified, the RETpd value of the MGmtclasS may override the RETpd value.

Specifying a GDG data set with a relative index produces an error message. The following examples are unsupported specifications:

```
LOCSITE DCBDSN=MYGDG(0)
LOCSITE DCBDSN=MYGDG(–nnn)or
LOCSITE DCBDSN=MYGDG(+nnn)
```

See “Using a DCBDSN Model to Create a New Data Set” on page 70 for more information about DCbdsn.

Directory
Specifies the number of directory blocks to be allocated for the directory of a PDS.

Specify Directory=0 to allocate without specifying the number of directory blocks. Specify Directory=0 when you are also specifying DATAClass=*dataclass* and you want the SMS dataclass to provide the Directory *size*.

The *size* you specify with the Directory parameter overrides the DATAClass directory specification.

size
Specifies the number of directory blocks to be allocated for the directory of a PDS. The valid range is 1–16777215. A *size* of zero indicates that the directory blocks should be taken from the SMS data class.

DIRECTORYMode
Specifies that only the data set qualifier immediately below the current directory is treated as an entry in the directory. In directory mode, this data set qualifier is the only one used by the MPUT subcommand.

DIRECTORYMode has no effect on files residing in an HFS.

FILEtype
Specifies the file type of the data set.

type
The file type of the data set can be:

Type	Description
SEQ	Sequential or partitioned data sets
SQL	SQL query function

LRecl
Used to specify the logical record length (LRecl) of a newly allocated data set.

Specify LRecl with no value when you are also specifying DATAClass=*data_class* and you want the SMS dataclass to provide the LRecl value, or when you are specifying DCbdsn=*data_set_name* and you want to use the LRecl from the DCBDSN data set. If LRecl=*length* is specified with either DATAClass or DCbdsn, the length specified by the LOCSITE LRecl parameter overrides the DATAClass or DCbdsn LRecl.

length
Specifies the logical record length of a newly allocated data set. The valid range is 0–32760. A special value of x (LRecl=x) is also supported to indicate that a logical record length can exceed 32 760 for variable-length spanned records.

Specifying LRecl=0 has the same effect as specifying LRecl with no parameters.

MGmtclass
Used to specify the SMS management class as defined by your organization for the target host. Specifying MGmtclass with no *mgmtclass* cancels the mgmtclass specification. The equal sign (=) is optional in this case.

mgmtclass
Specifies the SMS management class as defined by your organization for the target host. If the mgmtclass specified has a setting for RETpd, the value specified by the mgmtclass may override the setting of the LOCSITE RETpd parameter, the RETpd value of a model data set if the DCbdsn parameter is specified, and the RETpd value defined in an SMS data class if DATAClass is specified. See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

MIGratevol
Specifies the volume ID for migrated data sets if they do not use IBM storage management systems. If you do not specify MIGratevol, the default *volume_serial* is MIGRAT.

volume_ID
The volume ID for migrated data.

NOASAtrans
Treats ASA file transfers as regular file transfers; that is, the ASA characters are treated as part of the data and are not converted to print control characters.

NOAUTOMount
Prevents automatic mounting of volumes for data sets on volumes that are not mounted.

NOAUTORecall
Prevents automatic recall of migrated data sets.

Note: A migrated data set can be deleted even though NOAUTORecall is specified, because migrated data sets are not recalled for deletion.

NOQUOtesoverride
A single quote at the beginning of the file name, as well as all other single quotes contained in the file name, is treated as part of the actual file name. The entire file name, including the leading single quote, is appended to the current working directory.

NORDW
Specifies that variable record descriptors (RDWs) are discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only.

NOSPRead
Specifies that the output is in report format rather than spreadsheet format when the file type is SQL.

NOTRAILingblanks
Specifies that the FTP client does not preserve the trailing blanks that are in a fixed format data set when the data is sent to a foreign host.

NOUCSSUB
In Unicode-to-EBCDIC conversion, the data transfer is terminated if any Unicode character cannot be converted into the EBCDIC code set.

NOUCSTRUNC
In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is not allowed. The data transfer is aborted if the logical record length of the receiving data set is too small to contain the data after conversion to EBCDIC.

Note: The setting of the CONDdisp parameter determines what happens to the target data set if the transfer is aborted.

NOWRAPrecord
Indicates that data is truncated if no new line character is encountered before the logical record length of the receiving file is reached.

PRImary
Used to specify the amount of tracks, blocks, or cylinders for primary allocation. When specified with a value of zero, no primary value is used when allocating the data set.

Specify PRImary with zero when you are also specifying DATAClass=*data_class* and you want the SMS dataclass to provide the PRImary *amount*.

To enable the SMS data class to determine the space allocation, both PRImary and SECondary must be specified as zero. The tracks, blocks, cylinders setting is ignored in this case. If PRImary with *amount* not equal to zero is specified

with DATAClass, the value specified by the LOCSITE PRImary parameter overrides the DATAClass space allocation.

amount
Specifies the amount of tracks, blocks, or cylinders for primary allocation. For allocating partitioned data sets, this is the amount that is allocated for the primary extent.

For allocating sequential data sets this is the maximum amount that is allocated for the primary extent. If a lesser amount is needed to hold the data being transferred, only the amount actually needed to hold the data is allocated. The valid range is 1–16777215.

Qdisk
Used to display statistics about available space on a volume. If Qdisk is entered without a specific *volume_serial*, statistics about available space are displayed for each volume that is defined with “Use Attribute=storage”.

volume_serial
Displays statistics about available space on a specific volume.

QUOtesoverride
Specifies single quotes at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory. This is the way single quotes are used in all previous MVS FTP servers, and is the default. Any single quotes inside the beginning and ending quote are treated as part of the file name.

QUOtesoverride indicates the usage of single quotes appearing at the beginning of, or surrounding, a file name. The setting of this keyword affects all FTP subcommands that have a path name as a parameter except keywords on the LOCSITE subcommand.

RDW
Specifies that variable record descriptors (RDWs) are treated as if they were part of the record and not discarded during FTP transmission of variable format data sets. This applies to transfers in stream mode only.

Note: RDW information is stored in a binary halfword. Transfer files in binary mode to avoid translation problems that can occur if you transfer this binary field in EBCDIC or ASCII.

RECfm
Used to specify the record format of a data set. When specified without the *format*, no record format is used when allocating the data set. The equal sign (=) is optional in this case.

Specify RECfm with no value when you are also specifying DATAClass=*data_class* and you want the SMS dataclass to provide the RECfm *format*, or when you are specifying DCbdsn=*data_set_name* and you want to use the record format from the DCBDSN data set.

If RECfm=*format* is specified with either DATAClass or DCbdsn, the value specified by the LOCSITE RECfm parameter overrides the DATAClass or DCbdsn record format.

format
Specifies the record format of a data set. Valid record formats are: F, FA, FB, FBA, FBM, FBS, FBSA, FBSM, FM, FS, FSA, FSM, U, UA, UM, V, VA,

VB, VBA, VBM, VBS, VBSA, VBSM, VM, VS, VSA, and VSM. The characters used to specify these record formats have the following meanings:

Code	Description
F	Fixed record length
V	Variable record length
U	Undefined record length
B	Blocked records
S	Spanned records (if variable) / standard records (if fixed)
A	Records contain ISO/ANSI control characters
M	Records contain machine code control characters

RESTGet

Allows opening the checkpoint data set for the GET request. This is the default when the RESTGet statement has not been added to the FTP.DATA file.

RETpd

Used to specify the number of days that a newly allocated data set should be retained.

Specify RETpd with no value when you are also specifying DATAClass=*data_class* or MGmtclass=*mgmtclass* and you want SMS to provide the RETpd value, or when you are specifying DCbdsn=*data_set_name* and you want to use the RETpd from the DCBDSN data set. If more than one of the LOCSITE parameters (RETpd, MGmtclass, DATAClass, or DCbdsn) are specified, the order of precedence (highest to lowest) is:

1. MGmtclass
2. RETpd
3. DCbdsn
4. DATAClass

If a retention period is associated with an SMS management or data class, or with a model DCBDSN data set, the value of the retention period may be overridden to another nonzero value, but it cannot be overridden to have no retention period specified for the newly created data sets.

days

Specifies the number of days that a newly allocated data set should be retained. The valid range is 0–9999. A value of zero indicates a retention period of zero days so that the data set expires the same day as it was created.

SBDataconn

Specifies the conversions between EBCDIC and ASCII code pages to be used for data transfers.

```
LOCSITE SBDataconn=dsname
LOCSITE SBDataconn=(ebcdic_cp,ascii_cp)
LOCSITE SBDataconn=*
```

dsname

Specifies the fully qualified name of an MVS data set or HFS file that contains the EBCDIC-to-ASCII and ASCII-to-EBCDIC translate tables generated by the CONVXLAT utility.

Notes:

1. The name must *not* be enclosed in quotes. If quotes appear, they are treated as part of the name. (QUOtesoverride is ignored.)

2. The HFS name is case-sensitive. The MVS name is not.
3. The name cannot begin with a left paren [(].
4. The SBDataconn keyword must be the only keyword or the last keyword on a LOCSITE subcommand.

ebcdic_cp
Specifies the name of an EBCDIC code page recognized by iconv. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

ascii_cp
Specifies the name of an ASCII code page recognized by iconv. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

***** Indicates the translate tables set up at initialization for the data connection must be used.

SECondary
Specifies the amount of tracks, blocks, or cylinders for secondary allocation.

Specify SECondary=0 when you are also specifying DATAClass=*dataclass* and you want the SMS dataclass to provide the SECondary value. To enable the SMS data class to determine the space allocation, both PRImary and SECondary must be specified as zero. The tracks/blocks/cylinders setting is ignored in this case. If SECondary is specified as other than zero with DATAClass, the value specified by the SITE SECondary parameter overrides the DATAClass space allocation.

amount
Specifies the amount of tracks, blocks, or cylinders for secondary allocation. The valid range is 0–16777215. If you specify an *amount* of zero, FTP allocates without specifying secondary space.

SPRead
Specifies that the output is in spreadsheet format when the file type is SQL.

SQLCol
Specifies the column headings of the SQL output file.

any
The label of the DB2 SQL table column heading is the first choice for column heading, but if there is no label, the name becomes the column heading.

labels
Labels are the DB2 SQL table column headings. If any of the columns do not have labels, FTP supplies a column heading in the form of `COLnnn`.

names
Uses the names of the DB2 SQL table column headings. The labels are ignored.

STOrclass
Specifies the SMS storage class as defined by your organization for the target host. Cancels the storage class specification, when specified without a *storage_class* parameter value. The equal sign (=) is optional in this case.

See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

storage_class
Specifies the SMS storage class as defined by your organization for the target host. The SMS storage class may override settings for the VOLume or Unit site parameters.

TRacks
Specifies that primary and secondary space allocations are in tracks.

If both PRImary and SECondary are specified as zero, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the TRacks parameter is ignored.

TRAILingblanks
Specifies that the FTP server preserves the trailing blanks in a fixed format data set when the data is sent to a foreign host.

UCSHOSTCS
Specifies the EBCDIC code set to be used when converting to and from Unicode. If you do not specify a *code_set*, the current code set is used.

code_set
Name of the EBCDIC code set to be used when converting to and from Unicode.

UCSSUB
In Unicode-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any Unicode character that cannot successfully be converted. Data transfer continues.

UCSTRUNC
In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is allowed. The data transfer continues even if EBCDIC data is truncated.

UMask
Defines the file mode creation mask. The file mode creation mask defines which permission bits are *not* to be set on when a file is created. When a file is created, the permission bits requested by the file creation are compared to the file mode creation mask, and any bits requested by the file creation which are disallowed by the file mode creation mask are turned off.

The format of the UMask keyword is UMask *ooo*.

When a file is created, the specified permission bits for the file are 666 (-rw-rw-rw-). If the file mode creation mask is 027, the requested permissions and the file mode creation mask are compared:

```
110110110   - 666
000010111   - 027
---------
11010000    - 640
```

The actual permission bits set for the file when it is created is 640 (-rw-r-----).

Notes:

1. The default value for UMask is 027.
2. You cannot use FTP to create HFS files with execute permissions. If you require execute permissions, use the LOCSITE CHMod command to change permissions after the file has been created.

Unit
Specifies the unit type for allocation of new data sets.

The setting for Unit may be overridden by the SMS storage class, if one is specified. Therefore, it is recommended that Unit not be coded if an SMS storage class is in use.

unit_type
The unit type (for example, 3380) for the allocation of new data sets on direct access devices. If *unit_type* is not specified, the unit type used for allocation is set back to the system default.

VOLume
Specifies the volume serial number for allocation of new data sets. If *volume_serial* is not specified, the volume serial number used for allocation is set back to the system default.

The setting for VOLume may be overridden by the SMS storage class, if one is specified. Therefore, VOLume should not be coded if an SMS storage class is in use.

volume_serial
The volume serial number for allocation of new data sets.

WRAPrecord
Indicates that data is wrapped to the next record if no new line character is encountered before the logical record length of the receiving file is reached.

Usage

- Because more than one parameter can be specified with the LOCSITE subcommand, *parameter* can be repeated many times, with each *parameter* separated by a blank space.
- Issue the HELP LOCSITE subcommand to display a list of services available on the local host.
- The site-dependent information set with the LOCSITE subcommand remains active until you issue a new LOCSITE subcommand. The new LOCSITE subcommand adds to or changes the parameters established by previous LOCSITE subcommands.
- If you specify one or more incorrect parameters with the LOCSITE subcommand, an error message specifying the incorrect parameter is displayed. All correct parameters are set, regardless of any incorrect parameters, and do not need to be reissued.

Context

- See “HELP and ? Subcommands—Display Help Information” on page 137 for more information on the HELP subcommand.
- To check the effect of the LOCSITE command on the attributes at the local host, see “LOCSTAT Subcommand—Display Local Status Information” on page 160.

LOCSTAT Subcommand—Display Local Status Information

Purpose

Use the LOCSTAT subcommand to display local status information.

Format

Parameters

There are no parameters for this subcommand.

Examples

The following example shows the output from a LOCSTAT subcommand.

```
Command: locstat
Trace: FALSE, Send Port: TRUE
Send Site with Put command: TRUE
Connected to:9.67.113.57, Port: 6321, logged in
Local Port: 1045
Proxy connection to:9.67.112.25, Port: 6121, logged in
Proxy Local Port: 1046
Data type:a, Transfer mode:s, Structure:f
Automatic recall of migrated data sets.
Automatic mount of direct access volumes.
Data set mode. (Do not treat each qualifier as a directory.)
Primary allocation 2 cylinders, Secondary allocation 3 cylinders
Partitioned data sets will be created with 23 blocks
FileType is SEQ (Sequential - the default).
Number of access method buffers is 1.
Mgmtclass for the new data sets is MGMT1
Storclass for the new data sets is STOR1
Dataclass for the new data sets is DATA1
RDW's from VB/VBS files are retained as part of data.
Retention period is 2
DB2 subsystem name is DB2
Volid of Migrated Data Sets is MIGRAT
Data sets will be allocated like data set 'USER33.TEST.S.A'
Trailing blanks in records read from RECFM F data sets are discarded.
Record format: FB, Lrecl: 80, Blocksize: 8000.
Data sets will be allocated on CPDLB0
Data sets will be allocated uding unit SYSDA
Wrapping data into next record.
Checkpoint interval is 10000
Prompting: ON, Globbing: ON
ASA control characters transferred as C control character sequences
New data sets deleted if a store operation terminates abnormally
Single quotes will override the current working directory
UMASK value is 027
Using 'USER33.FTP.DATA' for local site configuration parameteres

Command:
```

Usage

Table 22 on page 161 shows LOCSTAT subcommand line descriptions and the commands that they affect.

Table 22. Locstat Subcommand Line Description

Line	Description	Commands affecting
1	Trace setting (TRUE or FALSE)	ftp -d ftp (TRACe debug
1	SENDPORT setting (True or FALSE)	sendport
2	SENDSITE setting (TRUE of FALSE)	sendsite
3	IP address and port of foreign host	ftp host port open host port close
3	Login status	user close
4	Port number of the local host	
5	IP address and port of foreign host for a proxy connection	proxy open proxy close
5	Login status	proxy user prosy close
6	Port number of the local host for a proxy connection	
7	Data type: a (ASCII) e (EBCDIC) i (image) b (DBCS) u (unicode) Notes: a. If data type is b, a line that states the actual language that is being used will follow line 7. For example, Language: Big5 S E b. If data type is u, following line 7 are three lines that tell the host character set (locsi uchostcs=), the ucs substitution option (locsi ucssub|noucssub), the ucs truncation option (locsi ucstrunc|noucstrunc), and the byte order. For example, Host code set for TYPE U transfer: IBM-1047 UCS Substitution: OFF, UCS Truncation: OFF Byte Order: big-endian	 ascii ebcdic binary big5, and so forth ucs2
7	Transfer mode: s (stream) b (block) c (compressed)	 stream, mode s block, mode b compress, mode c
7	Structure: f (file) r (record)	 file, stru f record, stru r
8	AUTORecall setting – allow recalls or do not allow them	locsi autorecall locsi noautorecall
9	AUTOMount setting – allow dasd mounts or do not allow them	locsi automount locsi noautomount
10	Data set qualifier handling (dataset mode or directory mode)	locsi datasetmode locsi directorymode
11	Space allocation: primary extext	locsi primary=x

Table 22. Locstat Subcommand Line Description (continued)

Line	Description	Commands affecting
11	Space allocation: secondary extext	locsi secondary=x
11	Space allocation: spacetype:	locsi cylinders locsi tracks locsi blocks
12	Number of directory blocks for a new PDS	locsi directory=x
13	Filetype (SEQ or SQL)	locsi filetype=type
14	Number of access method buffers	locsi bufno=xxx
15	Management class (if one is specified)	locsi mgmtclass=m
16	Storage class (if one is specified)	locsi storclass=s
17	Data class (if one is specified)	locsi dataclass=d
18	Record descriptor words (RDWs) are kept or discarded	locsi rdw locsi nordw
19	Retention period in days for a new data set	locsi retpd=xxx
20	Name of the DB2 subsystem	locsi db2=name
21	Volid of migrated data sets	locsi migrat=vid
22	Data set used a model	locsi dcbdsn='XXX'
23	Whether trailing blanks are transferred on a store operation	locsi trailingbl locsi notrailingbl
24	For new data sets: record format	locsi recfm=format
24	For new data sets: logical record length	locsi lrecl=xxx
24	For new data sets: blocksize	locsi blksize=xxx
25	Volume used for new data sets	locsi volume=XXX
26	Unit used for new data sets	locsi unit=type
27	Whether data is wrapped to the next record	locsi wrap locsi nowrap
28	Interval between checkpoint markers	locsi chkptint=x
29	Prompting (ON or OFF) for the mdelete, mget, and mput subcommands	prompt
29	Globbing (ON or OFF) for the mdelete, mget, and mput subcommands	glob
30	ASA control characters as ASA control characters or as C control character sequences	locsi noasatrans locsi asatra
31	When a get transfer fails, delete a new data set or catalog it	locsi conddisp=Del locsi conddisp=Cat
32	Usage of quotes for a file name is do or do not override	locsi quotesover locsi noquotesover
33	File mode creation mask	locsi umask ooo
34	Name of the FTP.DATA file that is used	

For information on messages issued for the LOCSTAT command, see Table 23 on page 215.

LPWD Subcommand—Display the Current Working-Level Qualifier

Purpose

Use the LPWD subcommand to display the name of the current working directory on the local host.

Format

Parameters

There are no parameters for this subcommand.

Examples

Display the name of the current working directory:

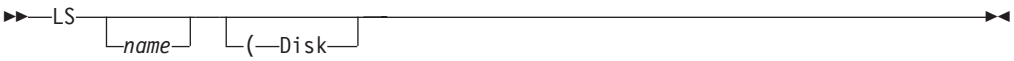
```
lpwd
 Local directory is partitioned data set USER14.FTP.TEST1.
 Command:
```

LS Subcommand—Obtain a List of File Names

Purpose

Use the LS subcommand to list only the names of a set of remote files, file group, or directory.

Format



Parameters

name
Specifies the set of remote files whose names are to be listed. The default is the entire current directory or file group.

Disk
Stores the results of the LS subcommand in the *user_id*.FTP.LSOUTPUT data set. The results are not displayed on the screen.

Note: If the local current working directory is an HFS directory, the results are stored in a file named `LSOUTPUT`.

Examples

The following is a sample response that is displayed as a result of the LS subcommand.

```
>>>PORT 9,67,58,227,4,63
200 Port request OK.
>>>NLST
125 List started OK.
A.X
CHR.TXT
OBEY.TCPIP
PROFILE.EXEC
SPF.ISPPROF
USERTRAN.TCPXLBIN
250 List completed successfully.
Command:
```

The following is a sample entry and response that is displayed as a result of the LS subcommand listing HFS files.

```
cd '/u/user121/ftp.example'

 >>>CWD '/u/user121/ftp.example'
 250 HFS directory /u/user121/ftp.example is the current working directory
 Command:
ls
 >>>PORT 9,67,112,25,4,62
 200 Port request OK.
 >>>NLST
 125 List started OK
 append02
 file1
 file2
 file3
 file4
 file5
 250 List completed successfully.
 Command:
```

Usage

- To make a file group the current working directory, use the CD subcommand. The method you use to specify a directory or file group is host-dependent.
- You can use special characters for pattern matching when specifying the *name*. These characters depend on the host FTP server.
- If the current local directory is a PDS, only a member named `LSOUTPUT` is created. If the current local directory is not a PDS, the local directory, not the user ID, is used as the high-level qualifier for the data set name.

Context

- See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about pattern matching and about specifying data sets and files.
- To make a file group the current working directory, see “CD Subcommand—Change the Directory on the Remote Host” on page 112.
- To get a list of complete directory entries with auxiliary information about the files, see “DIR Subcommand—Obtain a List of Directory Entries” on page 124.
- To change the local directory, see “LCD Subcommand—Change the Local Working Directory” on page 143.

MDELETE Subcommand—Delete Multiple Files

Purpose

Use the MDELETE subcommand to delete multiple files.

Format

Parameters

foreign_file
Specifies the name of the file to be deleted on the remote host.

Examples

The following is a sample entry and the response that is displayed as a result of the MDELETE subcommand for multiple HFS files.

```
cd '/u/user121/ftp.example'

 >>>CWD '/u/user121/ftp.example'
 250 HFS directory /u/user121/ftp.example is the current working directory
 Command:
 mdelete file1 file2 file3
 >>>PORT 9,67,112,25,4,75
  200 Port request OK.
 >>>NLST file1
 125 List started OK
 250 List completed successfully.
 >>>PORT 9,67,112,25,4,77
 200 Port request OK.
 >>>NLST file2
 125 List started OK
 250 List completed successfully.
 >>>PORT 9,67,112,25,4,76
 200 Port request OK.
 >>>NLST file3
 125 List started OK
 250 List completed successfully.
 >>>DELE file1
 250 /u/user121/ftp.example/file1 deleted.
 >>>DELE file2
 250 /u/user121/ftp.example/file2 deleted.
 >>>DELE file3
 250 /u/user121/ftp.example/file3 deleted.
 Command:
```

Usage

- Because more than one file can be deleted with the MDELETE subcommand, the *foreign_file* parameter of the MDELETE subcommand can be repeated many times, with each *foreign_file* separated by a blank space.
- If you specify one or more incorrect foreign files with the MDELETE subcommand, an error message specifying the incorrect foreign file is displayed. All correct foreign files are deleted, regardless of any incorrect foreign files, and the MDELETE subcommand does not need to be reissuedfor these files.
- HFS file names require special handling for certain special characters. All special characters that the operating system requires to be preceded by an escape

character in commands issued to the shell must be preceded by the backslash (\) escape character, except for the single quote (’), double quote (”), or blank ( ).

Context

See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about naming conventions.

MGET Subcommand—Copy Multiple Files

Purpose

Use the MGET subcommand to copy multiple files from a remote host to your local host and create a corresponding number of local files.

Format

Parameters

foreign_file
Specifies the name of the file to be retrieved from the remote host.

Because more than one file can be copied with the MGET subcommand, the *foreign_file* parameter of the MGET subcommand can be repeated many times, with each *foreign_file* separated by a blank space. You can use special characters for pattern matching when specifying the *foreign_file* with the MGET subcommand. These characters are dependent on the foreign host FTP server.

REPLACE
Causes a data set on your local host to be overwritten if it already exists. If the data set already exists, and you do not use the replace parameter, the existing data set is not overwritten. A message informing you of this is displayed.

If the data set already exists and you specify REPLACE, the data in the file is overwritten, but not reallocated. This means the local data set retains its existing characteristics.

Examples

The following is a sample entry and response that is displayed as a result of the MGET subcommand for multiple HFS files.

```
cd '/u/user121/ftp.example'

 >>>CWD '/u/user121/ftp.example'
 250 HFS directory /u/user121/ftp.example is the current working directory
 Command:
 mget file1 file2 file3
 >>>PORT 9,67,112,25,4,90
  200 Port request OK.
 >>>NLST file1
 125 List started OK
 250 List completed successfully.
 >>>PORT 9,67,112,25,4,91
 200 Port request OK.
 >>>NLST file2
 125 List started OK
 250 List completed successfully.
 >>>PORT 9,67,112,25,4,92
 200 Port request OK.
 >>>NLST file3
 125 List started OK
 250 List completed successfully.
 >>>PORT 9,67,112,25,4,93
 200 Port request OK.
 >>>RETR file1
 125 Sending data set /u/user121/ftp.example/file1
 250 Transfer completed successfully.
 3464 Bytes transferred in 1.031 seconds.  Transfer rate 3.36 kbytes/sec.
 >>>PORT 9,67,112,25,4,94
 200 Port request OK.
 >>>RETR file2
 125 Sending data set /u/user121/ftp.example/file2
 250 Transfer completed successfully.
 3993 Bytes transferred in 0.923 seconds.  Transfer rate 4.33 kbytes/sec.
 >>>PORT 9,67,112,25,4,95
 200 Port request OK.
 >>>RETR file3
 125 Sending data set /u/user121/ftp.example/file3
 250 Transfer completed successfully.
 3993 Bytes transferred in 0.791 seconds.  Transfer rate 5.05 kbytes/sec.
 Command:
```

The following is a sample entry and response that is displayed as a result of the MGET subcommand using a wildcard character in the file name.

```
Command:
mget file*
>>>PORT 9,67,113,57,5,123
200 Port request OK.
>>>NLST file*
125 List started OK
250 List completed successfully.
Mget file1 (Yes|No|Quit|Stop prompting)? s
>>>PORT 9,67,113,57,5,124
200 Port request OK.
>>>RETR file1
125 Sending data set /u/user31/file1
250 Transfer completed successfully.
164 bytes transferred in 0.310 seconds.  Transfer rate 0.53 Kbytes/sec.
>>>PORT 9,67,113,57,5,125
200 Port request OK.
>>>RETR file2
125 Sending data set /u/user31/file2
250 Transfer completed successfully.
164 bytes transferred in 0.270 seconds.  Transfer rate 0.61 Kbytes/sec.
>>>PORT 9,67,113,57,5,126
200 Port request OK.
>>>RETR file3
125 Sending data set /u/user31/file3
250 Transfer completed successfully.
164 bytes transferred in 0.280 seconds.  Transfer rate 0.59 Kbytes/sec.
Command:
```

Usage

- When you use the MGET subcommand, FTP may truncate data records and you might lose data, if:
  - You are creating a new data set at the client and the value of LRecl, as shown by the LOCSTAT command, is a value less than the LRecl of a received data set, FTP truncates the received data set.
  - The data set name already exists at the client and the logical record length (LRecl) of the data set at the client is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.

    You can encounter this situation when you use MGET with the REPLACE option.
- If the name specified for *foreign_file* is not acceptable to your local host, the file is not transferred. To get a file from the remote host, you must have a defined working directory on that host, and you must have read privileges to the files in this working directory.
- If you specify one or more incorrect foreign files with the MGET subcommand, an error message specifying the incorrect foreign file is displayed. All correct foreign files are retrieved, regardless of any incorrect foreign files, and do not need to be reissued.
- HFS file names require special handling for certain special characters. Except for single quote (’), double quote (”), or blank ( ), all special characters that the operating system requires to be preceded by an escape character in commands issued to the shell must be preceded by the backslash (\) escape character.
- The MGET command is not applicable to generation data groups (GDGs).
- MGET can be used with the PROXY subcommand to transfer files from a host on a primary connection to a host on a secondary connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

Context

- See “CD Subcommand—Change the Directory on the Remote Host” on page 112 for more information about working directories.
- See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about naming conventions.

MKDIR Subcommand—Create a Directory on the Remote Host

Purpose

Use the MKDIR subcommand to create a directory on the remote host.

Format

Parameters

directory
Specifies the name of the directory to be created.

Examples

In this example, a directory is created on the remote host (1.1.2.3 in this example). Both `EXAMPLE` and `FTP.EXAMPLE` are created in the remote host, showing the difference between specifying and omitting quotation marks in the directory name.

```
  User: ftp 9.67.113.24.621
System:    IBM FTP CS/390 V2R8   1997  314 01:11 UTC
           FTP.DATA FILE NOT FOUND.  USING HARDCODED DEFAULT VALUES.
           Connecting to 1.1.2.3, port 21
           220-EZAFTSRV IBM CS/390 V2R8 at EMU.ABC.OZ, 15:34:32 on 08/03/93
           220 Connection will not timeout.
           NAME (<host>:tsouserid):

  User:  user121
System:    >>>USER user121
           331 Send password please.
           Password:
  User:
           >>>PASS ********
           230 USER121 is logged on.  Working directory is '/u/user121".
           Command:
```

```
  User:  dir
System:    >>>PORT 9,67,112,25,4,96
           200 Port request OK.
           >>>LIST
           125 List started OK.
           total 2768
          -rwxr-xr-t    2  USER121  SYS1     389120  Feb  5  16:03  ftpdka
          -rwxr-xr-t    2  USER121  SYS1     962560  Feb  5  16:04  ftpsrvka
          -rw-r-----    1  USER121  SYS1      11648  Jan 20  14:30  g.s
          drwxr-x---    3  USER121  SYS1          0  Oct 21  17:50  msg
          -rw-r-----    1  USER121  SYS1       1458  Jan 10  19:25  s.k
          drwxr-x---    2  USER121  SYS1          0  Feb  6  15:59  tcpip
          drwxr-x---    2  USER121  SYS1          0  Feb  6  17:29  test
           250 List completed successfully.
           Command:

  User:  mkdir example
System:    >>>MKD example
           257 "/u/user121/example" created.
           Command:
```

```
  User:  dir
System:    >>>PORT 9,67,112,25,4,97
           200 Port request OK.
           >>>LIST
           125 List started OK.
           total 2768
           drwxr-xr--    2  USER121  SYS1          0  Feb  7  19:57  example
           -rwxr-xr-t    2  USER121  SYS1     389120  Feb  5  16:03  ftpdka
           -rwxr-xr-t    2  USER121  SYS1     962560  Feb  5  16:04  ftpsrvka
           -rw-r-----    1  USER121  SYS1      11648  Jan 20  14:30  g.s
           drwxr-x---    3  USER121  SYS1          0  Oct 21  17:50  msg
           -rw-r-----    1  USER121  SYS1       1458  Jan 10  19:25  s.k
           drwxr-x---    2  USER121  SYS1          0  Feb  6  15:59  tcpip
           drwxr-x---    2  USER121  SYS1          0  Feb  6  17:29  test
           250 List completed successfully.
           Command:

  User:  mkdir '/u/user121/ftp.example'
System:    >>>MKD '/u/user121/ftp.example'
           257 "/u/user121/ftp.example" created.
           Command:
```

```
  User:  dir
System:    >>>PORT 9,67,112,25,4,98
           200 Port request OK.
           >>>LIST
           125 List started OK.
           total 2800
           drwxr-x---    2  USER121  SYS1          0  Feb  7  19:57  example
           drwxr-x---    2  USER121  SYS1          0  Feb  7  19:57  ftp.example
           -rwxr-xr-t    2  USER121  SYS1     389120  Feb  5  16:03  ftpdka
           -rwxr-xr-t    2  USER121  SYS1     962560  Feb  5  16:04  ftpsrvka
           -rw-r-----    1  USER121  SYS1      11648  Jan 20  14:30  g.s
           drwxr-x---    3  USER121  SYS1          0  Oct 21  17:50  msg
           -rw-r-----    1  USER121  SYS1       1458  Jan 10  19:25  s.k
           drwxr-x---    2  USER121  SYS1          0  Feb  6  15:59  tcpip
           drwxr-x---    2  USER121  SYS1          0  Feb  6  17:29  test
           250 List completed successfully.
           Command:
```

Usage

- The MKDIR subcommand sends a request to the remote host FTP server to create a directory with name *directory* in the current working directory. When the request is sent to an MVS server, a PDS is created if either a fully qualified MVS data set name is entered (for example, 'USER33.TEST.PDS'), or if the current working directory is an MVS data set name. For example, if the current working directory is USER33.TEST, a command MKDIR PDS creates a PDS named USER33.TEST.PDS.
- For an MVS server, the data set characteristics used to allocate the new PDS are determined by the settings of the SITE parameters of the server.

Context

- See "SITE Subcommand—Send Site-Specific Information to a Host" on page 197 for information on setting the SITE parameters.
- See "STATUS Subcommand—Retrieve Status Information from a Remote Host" on page 214 for information on displaying the SITE parameters.

MODE Subcommand—Set the Data Transfer Mode

Purpose

Use the MODE subcommand to define how bits of data are to be transmitted.

Format

Parameters

B Sets the block mode. In block mode, data is transmitted as a series of data blocks, preceded by one or more header bytes. Block mode preserves the logical record boundaries of the data set or file. When MODE is set to B, the data transfer type must be EBCDIC.

Specifying MODE B is equivalent to specifying the BLOCK subcommand.

C Sets the compressed mode. In compressed mode, data is transmitted as a series of data blocks, preceded by one or more header bytes. Compressed mode preserves the logical record boundaries of the data set or file. In compressed mode, data is transmitted without repetitive characters and blanks. When MODE is set to C, the data transfer type must be EBCDIC.

Note: Because additional processing time is required for both the sender and receiver to compress or decompress the data, evaluate the time factor before you compress a file.

Specifying MODE C is equivalent to specifying the COMPRESS subcommand.

S Sets the stream mode. In stream mode, data is transmitted as a stream of bytes. Any data transfer type can be used with stream mode. Stream mode is efficient because data block information is not transferred.

Specifying MODE S is equivalent to specifying the STREAM subcommand.

Usage

- To use MODE C, the receiving host must support the compressed data mode.
- Data compression increases CPU processing costs even if the amount of data transferred is not large.

Context

- For the syntax of the BLOCK, COMPRESS, and STREAM subcommands, see “BLOCK Subcommand—Set the Block Data Transfer Mode” on page 111, “COMPRESS Subcommand—Set the Compressed Data Transfer Mode” on page 118, or “STREAM Subcommand—Set the Stream Data Transfer Mode” on page 228.
- For more information about transfer methods, see Table 12 on page 52.

MPUT Subcommand—Copy Multiple Data Sets to the Remote Host

Purpose

Use the MPUT subcommand to copy multiple data sets from your local host to the remote host.

Format

Parameters

local_data_set
Specifies the name of the file on your local host being sent to the remote host.

Because more than one data set can be copied with the MPUT subcommand, the *local_data_set* parameter of this subcommand can be repeated many times, with each *local_data_set* separated by a blank space. You can use the asterisk (*) character for pattern matching when specifying the *local_data_set* with the MPUT subcommand.

Examples

The following is a sample entry and response that is displayed as a result of the MPUT subcommand to send selected files.

```
Command:
mput file*
Mput FILE1 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,128
200 Port request OK.
>>>STOR FILE1
125 Storing data set /u/user31/temp/FILE1
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds.  Transfer rate 16.40 Kbytes/sec.
Mput FILE2 (Yes|No|Quit|Stop prompting)? no
Mput FILE3 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,129
200 Port request OK.
>>>STOR FILE3
125 Storing data set /u/user31/temp/FILE3
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds.  Transfer rate 16.40 Kbytes/sec.
Command:
dir
>>>PORT 9,67,113,57,5,130
200 Port request OK.
>>>LIST
125 List started OK
total 16
-rw-r-----   1 USER31   SYS1         162 Aug 14 13:20 FILE1
-rw-r-----   1 USER31   SYS1         162 Aug 14 13:21 FILE3
250 List completed successfully.
Command:
```

Usage

- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the MPUT subcommand, FTP may truncate data records and you might lose data, if:

  - You are creating a new file at the server and the value of LRecl, as shown by the STATUS subcommand, is a value less than the LRecl of the transmitted data set and SENDSITE has been set to OFF, FTP truncates the transmitted data set.
  - The data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.
- By default, if you use the MPUT subcommand, the remote host creates files with the same names as those specified in *local_data_set* and overwrites any existing files with those names.

  To put files on the remote host with unique file names, you must have set unique storage on before issuing the MPUT command. Use the SUNIQUE subcommand to change the storage method.
- To send a data set to the remote host, you must have a defined working directory on the remote host and write privileges to the files in this working directory.
- If you specify one or more incorrect parameters with the MPUT subcommand, an error message specifying the incorrect parameter is displayed. All correct files are transferred, regardless of any incorrect parameters, and do not need to be reissued.
- The MPUT command is not applicable to GDGs.

Context

- See “SUNIQUE Subcommand—Toggle the Storage Method” on page 230 for information about changing the storage method on the remote host.
- See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about naming conventions.
- MPUT can be used with the PROXY subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

NOOP Subcommand—Test the Connection

Purpose

Use the NOOP subcommand to determine whether the foreign host is still responding.

Format

Parameters

There are no parameters for this subcommand.

Examples

- If the foreign host is responding, you receive one of the following responses:

  `200 OK` or `200 NOOP command successful.`

- If the foreign host does not respond or is not connected, you receive an appropriate error message, such as:

```
421 no connection
to foreign server.
```

Usage

You can use the NOOP command to keep a connection alive that would otherwise be disconnected if it were idle for longer than the system time-out period.

OPEN Subcommand—Connect to the FTP Server

Purpose

Use the OPEN subcommand to open a connection to the remote host FTP server in the following situations:

- If, after closing a connection, you want to open another connection without leaving the FTP environment.
- If you were unable to open a connection when you specified a *foreign_host* with the FTP command.

Format

Parameters

host_name
Specifies the host name or IP address of the foreign host.

port_number
Identifies a port on the foreign host. The default is well-known port 21.

Usage

If you are already connected to a host, you must disconnect from the host before you can connect to a different host with the OPEN subcommand. The only exception to this is if you are using the PROXY OPEN command. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

Context

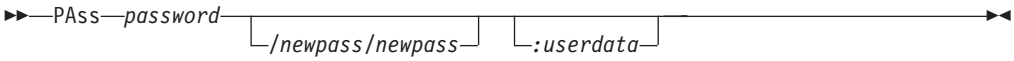
See “CLOSE Subcommand—Disconnect from a Remote Host” on page 117 for more information about closing a connection.

PASS Subcommand—Supply a Password

Purpose

Use the PASS subcommand to supply a password to a host.

Format



Parameters

password
Specifies your password on the remote host.

/newpass/newpass
Resets a password.

:userdata
An optional character string to be passed to the FTCHKPWD user exit routine. The user data must be separated from the password information by a colon (:) and can be any combination of up to 200 nonblank characters and numbers except the colon. Care should be taken when using the backslash character (\) in combination with other characters which might be interpreted as an escape sequence by the C compiler.

Usage

- The PASS subcommand must be preceded by the USER subcommand. For some sites, the password completes your identification for access control on the remote host.
- On MVS hosts, a PASS command is automatically generated by the USER command. If you do not enter the password as part of the user command, you are prompted for it.

Context

See “USER Subcommand—Identify Yourself to a Host or Change Your TSO User ID Password” on page 241 for more information.

PROMPT Subcommand—Toggle Interactive Prompting for M* Commands

Purpose

Use the PROMPT subcommand to toggle interactive prompting for MDELETE, MGET, and MPUT commands. Prompting is the default action unless the FTP session was started with the **-i** option which turns off interactive prompting.

Format

Parameters

There are no parameters for this subcommand.

Examples

The following example shows the MPUT command used with interactive prompting on.

```
Command:
mput file*
Mput FILE1 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,128
200 Port request OK.
>>>STOR FILE1
125 Storing data set /u/user31/temp/FILE1
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds.  Transfer rate 16.40 Kbytes/sec.
Mput FILE2 (Yes|No|Quit|Stop prompting)? no
Mput FILE3 (Yes|No|Quit|Stop prompting)? yes
>>>PORT 9,67,113,57,5,129
200 Port request OK.
>>>STOR FILE3
125 Storing data set /u/user31/temp/FILE3
250 Transfer completed successfully.
164 bytes transferred in 0.010 seconds.  Transfer rate 16.40 Kbytes/sec.
Command:
```

Context

See “Using the FTP Command” on page 35 for more information on the **-i** option.

PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections

Purpose

Use the PROXY subcommand to execute an FTP subcommand on secondary control connections. PROXY enables the FTP command to connect simultaneously to two remote FTP servers, and transfer files between those servers.

Format

Parameters

subcommand

Is the name of any FTP subcommand except those listed in the first note in "Usage" on page 182. The first PROXY *subcommand* should be OPEN, which establishes the secondary server connection.

The following subcommands behave differently when prefaced by the PROXY subcommand:

- OPEN establishes the secondary server connection.
- CLOSE closes the secondary server connection.
- GET and MGET transfer files from the host on the primary connection to the host on the secondary connection.
- PUT, MPUT, and APPEND transfer files from the host on the secondary connection to the host on the primary connection.

Examples

The following example shows a proxy open to establish connection to a secondary server.

```
 Command:
proxy open 9.67.113.57 6321
 Connecting to:  9.67.113.57 port: 6321.
 220-FTPDJG1 IBM CS/390 V2R8 at MVS164, 13:06:23 on 1997-08-14
 220 Connection will not timeout.
 NAME (9.67.113.57:USER33): user34
 >>>USER user34
 331 Send password please.
 PASSWORD:
 >>>PASS
 230 USER34 is logged on.  Working directory is "USER34.".
```

The following example shows the commands for a proxy transfer:

- PASV to the secondary server
- PORT to the primary server
- RETR to the primary server
- STOR to the secondary server

Two 250 replies are received by the client—one from each server.

```
 Command:
proxy get m1 mx
 >>>PASV
 227 Entering Passive Mode (9,67,113,57,5,121)
 >>>PORT 9,67,113,57,5,121
 200 Port request OK.
 >>>RETR m1
 125 Sending data set /u/user33/mpp1/m1
 >>>STOR mx
 125 Storing data set USER34.MX
 250 Transfer completed successfully.
 250 Transfer completed successfully.
 Command:
```

Usage

- The following subcommands are not valid proxy subcommands:
  - DEBUG
  - DELIMIT
  - GLOB
  - LCD
  - LPWD
  - PROMPT
  - QUIT
  - RESTART
  - SENDPORT
  - SENDSITE
  - SUNIQUE
  - TSO
- To receive help from a server on a secondary control connection, enter PROXY HELP SERVER.

PUT Subcommand—Copy Data Sets to the Remote Host

Purpose

Use the PUT subcommand to copy data sets from your local host to the remote host.

Format

Parameters

local_file
Specifies the name of the file on your local host being sent to the remote host.

foreign_file
Specifies the name that the delivered data set is given on the remote host. If the *foreign_file* name is not specified, the *foreign_file* name is the same as the *local_file* name.

Usage

- FTP maintains the attributes of a data set that is transmitted between a client and a server. However, when you use the PUT subcommand, FTP may truncate data records and you might lose data, if:
  - You are creating a new file at the server and the value of LRecl, as shown by the STATUS subcommand, is a value less than the LRecl of the transmitted data set and SENDSITE subcommand has been set to OFF, FTP truncates the transmitted data set.
  - The data set name already exists at the receiving site and the logical record length (LRecl) of the data set at the receiving site is less than the LRecl of the transmitted data set, FTP truncates the transmitted data set.
- When a PUT is issued FTP automatically sends a SITE subcommand containing record format information for the file or dataset. To toggle this off, you must first issue a SENDSITE subcommand. See “SENDSITE Subcommand—Toggle the Sending of Site Information” on page 196 for more detailed information.
- If the remote host already has a file with the name specified by *foreign_file*, the remote host overwrites the existing file. If the remote host does not have a file with the same name specified by *foreign_file*, the remote host creates a new file.
- To put files on the remote host with unique file names, you must have set unique storage on before issuing the PUT command. Use the SUNIQUE subcommand to change the storage method.
- To send a data set to the remote host, you must have a defined working directory on the remote host and write privileges to the files in this working directory.

Context

- See “SUNIQUE Subcommand—Toggle the Storage Method” on page 230 for information about changing the storage method on the remote host.
- See “Appendix A. Specifying Data Sets and Files” on page 545 for more information about naming conventions.

- PUT can be used with the PROXY subcommand to transfer files from a host on a secondary connection to a host on a primary connection. See “PROXY Subcommand—Execute FTP Subcommand on Secondary Control Connections” on page 181 for more information.

PWD Subcommand—Display the Current Working Directory

Purpose

Use the PWD subcommand to display the name of the current working directory on the remote host.

Format

Parameters

There are no parameters for this subcommand.

Examples

Display the name of the current working directory:

```
pwd
 >>>PWD
 257 "'USER17.HSMTEST.'" is working directory
 Command:
```

Display the name of the current HFS working directory:

```
pwd
 >>>PWD
 257 "/u/user121/example" is the HFS working directory.
 Command:
```

QUIT Subcommand—Leave the FTP Environment

Purpose

Use the QUIT subcommand to disconnect from the foreign host and end the FTP session.

Format

Parameters

There are no parameters for this subcommand.

Usage

- The QUIT subcommand ends the FTP session with the remote host and exits FTP on the local host. To establish a new session, use the FTP command.
- In an OS/390 UNIX environment, you can also press Ctrl-C to end an FTP session.
- When running with both a primary and a secondary server (by using the PROXY subcommand), the QUIT subcommand disconnects both sessions.

Context

See “Using the FTP Command” on page 35 for information about the FTP command.

QUOTE Subcommand—Send an Uninterpreted String of Data

Purpose

Use the QUOTE subcommand to send an uninterpreted string of data to the server port on the foreign host.

The QUOTE subcommand bypasses the FTP interface of your local host. You can use the QUOTE subcommand to send commands that the remote server understands, but that the local host does not understand.

Format

Parameters

string
Specifies the data to be sent verbatim to the remote host FTP server.

Examples

- For example, `QUOTE TYPE B 1` causes the FTP server to change its transfer type to Shift JIS kanji, without changing the transfer type in the FTP client. The client in this example should be set to the ASCII transfer type before the QUOTE subcommand is issued.
- The following examples show the screen display when setting the DBCS transfer type to JIS78KJ, shift-in JISROMAN, and then setting it to HANGEUL using EBCDIC SO/SI characters. The examples show an MVS TCP/IP FTP client connected to an MVS TCP/IP FTP server. All three methods of setting the DBCS transfer type are shown.

```
  User:  jis78kj (jisroman
System:    >>>TYPE b 4 r
           200-Representation type is kanji JIS 1978 shift-in JISROMAN
           200 Standard DBCS control used
           Command:
  User:  type b 4 r
System:    >>>TYPE b 4 r
           200-Representation type is kanji JIS 1978 shift-in JISROMAN
           200 Standard DBCS control used
           Command:
  User:  jis78kj (jisroman notype
System:    Command:
  User:  quote type b 4 r
System:    >>>type b 4 r
           200-Representation type is kanji JIS 1978 shift-in JISROMAN
           200 Standard DBCS control used
           Command:
  User:  hangeul (sosi ebcdic
System:    >>>TYPE b 5 s e
           200-Representation type is Hangeul
           200-SO/SI characters X'0E'/X'0F' used
           200 Data transfer is mixed SBCS/DBCS
           Command:
  User:  type b 5 s e
System:    >>>TYPE b 5 s e
           200-Representation type is Hangeul
           200-SO/SI characters X'0E'/X'0F' used
           200 Data transfer is mixed SBCS/DBCS
           Command:
  User:  hangeul (sosi ebcdic notype
System:    Command:
  User:  quote type b 5 s e
System:    >>>type b 5 s e
           200-Representation type is Hangeul
           200-SO/SI characters X'0E'/X'0F' used
           200 Data transfer is mixed SBCS/DBCS
           Command:
```

Usage

- No parsing or validity checking is performed on the character string you enter by FTP on your local host. If the character string you send to the FTP server is part of a required sequence of commands, you are required to provide this sequence correctly, or the results might be unpredictable.
- The QUOTE subcommand can be used to generate any of the DBCS TYPE commands supported by the server. This subcommand is used when the FTP server supports the DBCS TYPE command, but the FTP client does not.

RECORD Subcommand—Set the File Structure to Record

Purpose

Use the RECORD subcommand to set the file structure to record. This is equivalent to specifying the STRUCTURE R subcommand. See “STRUCTURE Subcommand—Set the File Structure” on page 229 for more information.

Format

Parameters

There are no parameters for this subcommand.

RENAME Subcommand—Rename Files

Purpose

Use the RENAME subcommand to rename a file on the remote host.

Format

Parameters

original_name
Specifies the current name of the file.

new_name
Specifies the new name of the file.

Usage

- For MVS data sets, if the file specified by the *new_name* already exists, the existing file is not replaced by the new file.
- For HFS files, if the file specified by *new_name* already exists, the existing file is replaced.

RESTART Subcommand—Restart a Checkpointed Data Transfer

Purpose

Use the RESTART subcommand to restart a checkpointed data transfer.

Format

Parameters

There are no parameters for this subcommand.

Usage

- The RESTART subcommand restarts the last checkpointed file transfer request at the point of the last valid checkpoint stored in the checkpoint data set, *userid*.FTP.CHKPOINT, for the TSO user *userid*. The file transfer environment, such as file transfer mode and type, which was in effect at the time of the last checkpointed file transfer request, must be reestablished before issuing the RESTART subcommand.
- The RESTART subcommand should be used when a checkpointed file transfer request fails because of some temporary condition such as the loss of the connection between the client and the server.
- If the FTP client is started under OS/390 UNIX, the checkpoint data set is named `ftp.checkpoint`, and the file is placed in your HFS directory.

Context

For more information about checkpointing a file transfer request, see the description of the CHKptint parameter on page 200 or page 150.

RMDIR Subcommand—Remove a Directory on the Remote Host

Purpose

Use the RMDIR subcommand to remove a directory on the remote host.

Format

Parameters

directory
Specifies the name of the directory to be removed.

Usage

- The RMDIR subcommand sends a request to the remote host FTP server to remove a directory with name *directory* from the current remote directory.
- The RMDIR subcommand can be used to delete a PDS.

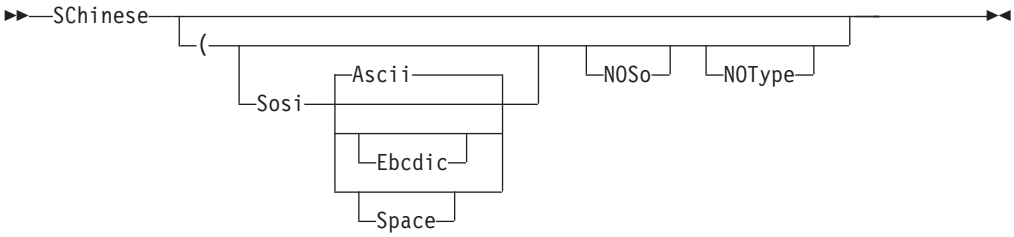
SCHINESE Subcommand—Change the Data Transfer Type to SCHINESE

Purpose

Use the SCHINESE subcommand to change the data transfer type to Simplified Chinese.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by the one of the following parameters — ASCII, EBCDIC, or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X’0E’ and X’0F’ to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X’20’ and X’20’ (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The SCHINESE client subcommand is equivalent to the TYPE B 9 server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

SENDPORT Subcommand—Toggle the Sending of Port Information

Purpose

Use the SENDPORT subcommand to toggle the automatic sending of the PORT command.

Format

Parameters

There are no parameters for this subcommand.

Usage

- By default, the SENDPORT subcommand is turned on when you start an FTP session. Each time you use the SENDPORT subcommand, it is turned alternately on and off.
- FTP uses a PORT command, by default, when establishing a connection for each data transfer. FTP does not send PORT commands for data transfer when you disable PORT commands by toggling the function off.
- SENDPORT is useful for communication with those FTP implementations that ignore PORT commands, but show (incorrectly) that the PORT command has been accepted.
- To determine if the sending of port information is enabled or disabled on your local host, use the LOCSTAT subcommand.

Context

See “LOCSTAT Subcommand—Display Local Status Information” on page 160 for more information about LOCSTAT subcommand.

SENDSITE Subcommand—Toggle the Sending of Site Information

Purpose

Use the SENDSITE subcommand to toggle the automatic sending of the SITE subcommands when sending a data set to a foreign host.

Format

Parameters

There are no parameters for this subcommand.

Usage

- By default, the SENDSITE subcommand is turned on when you start an FTP session. Each time you use the SENDSITE subcommand, it is turned alternately on and off.

  When turned on, FTP sends a SITE subcommand containing record format information for the file or data set, when you issue the PUT or MPUT subcommand.
- SENDSITE is useful when you want to PUT a file to the remote host and have the file created with the same characteristics as defined at the local host.
- If you are using either an SMS data class or a model DCB at your MVS server to provide the logical record length or record format, you must toggle the SENDSITE setting off at the client. Otherwise, the SITE information that is sent automatically by the client overrides the values provided by the SMS dataclass or model DCB.
- To determine if the sending of site information is enabled or disabled on your local host, use the LOCSTAT subcommand.

Context

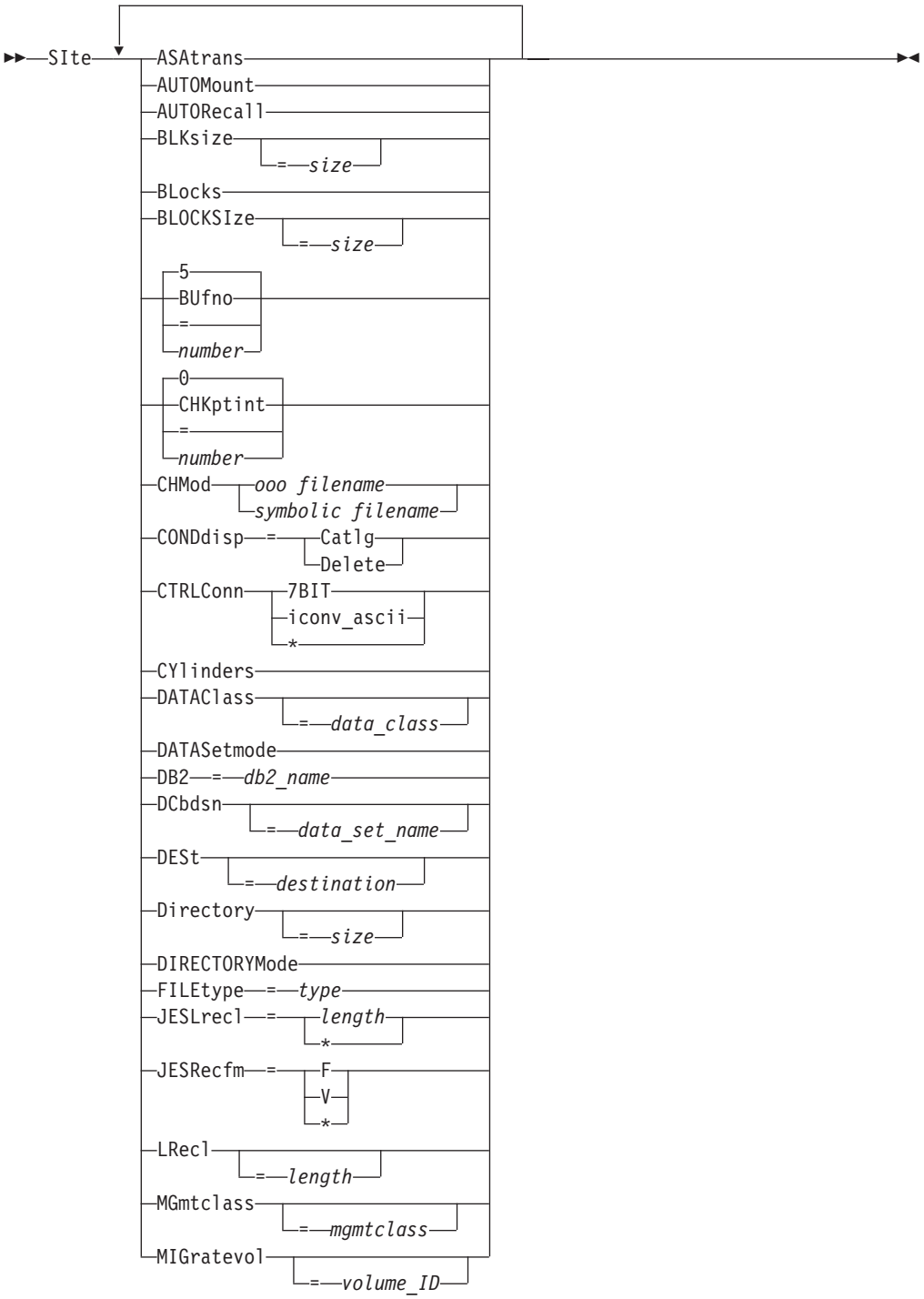
See “LOCSTAT Subcommand—Display Local Status Information” on page 160 for information about the LOCSTAT subcommand.

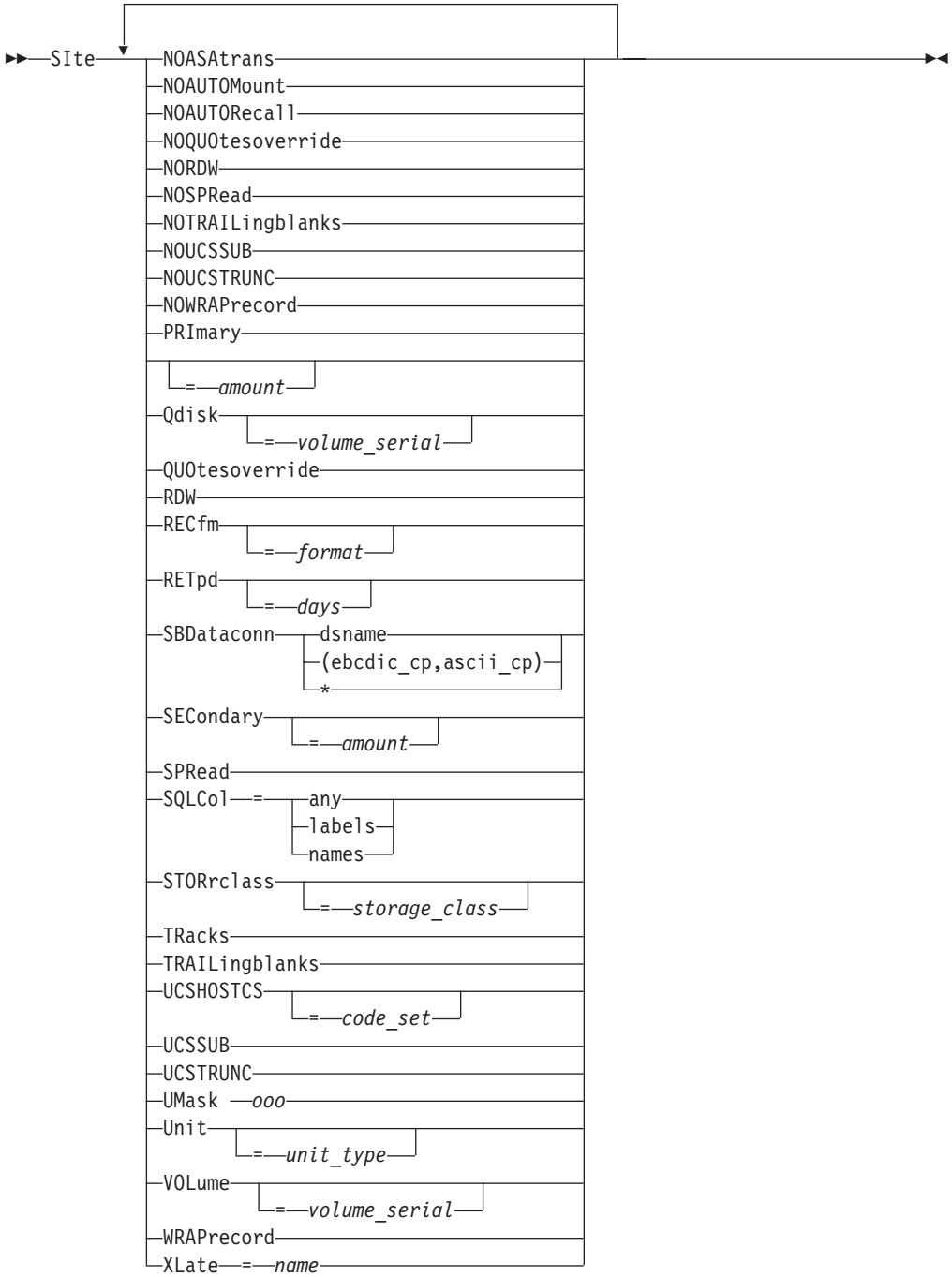
SITE Subcommand—Send Site-Specific Information to a Host

Purpose

Use the SITE subcommand to send information that is used by the remote host to provide services specific to that host system.

Format





Parameters

ASAtrans

Permits the FTP server to interpret the characters in the first column of ASA files being transferred as print control characters.

AUTOMount

Permits automatic mounting of volumes for data sets on volumes that are not mounted. If AUTOMount is specified and an unmounted volume is needed, a message is automatically issued to the MVS operator console requesting that

the volume be mounted. The MVS operator must then mount the volume and reply to the message before FTP can proceed.

AUTORecall
Permits automatic recall of migrated data sets.

BLKsize
Specifies the block size of a newly allocated data set. BLKsize is functionally equivalent to BLOCKSIze. The BLOCKSIze parameter is obsolete but it is accepted to provide compatibility with previous releases of OS/390 TCP/IP.

When specified without a *size*, no block size is used when allocating the new data set. When specified without a size, the equal sign (=) is optional.

Specify BLKsize with no value if you are also specifying DATAClass=*data_class* and you want the SMS dataclass to provide the BLKsize value, or if you are specifying DCbdsn=*data_set_name* and you want to use the blocksize from the DCBDSN data set. If BLKsize=*size* is specified with either the DATAClass or DCbdsn parameters, the value specified by the SITE BLKsize parameter overrides the DATAClass or DCbdsn blocksize.

Notes:

1. If you specify BLKsize without a size or with size of zero, FTP does not specify the block size when allocating new data sets.
2. Be especially careful specifying both BLKsize= and BLocks. While there are conditions where this is tolerated, if a valid BLKsize cannot be determined, the data set will not be created when the allocation is attempted.

size
Specifies the block size of a newly allocated data set. The valid range is 0–32760.

BLKsize=0 is a special case. When BLKsize=0 is specified, the operating system attempts to determine a blocksize for the new data set. FTP does not create the new data set unless the system is able to establish a nonzero blocksize.

BLocks
Specifies that primary and secondary space allocations are in blocks.

If both PRImary and SECondary are unspecified, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the BLocks parameter is ignored.

BLOCKSIze
Specifies the block size of a newly allocated data set. BLOCKSIze is functionally equivalent to BLKsize. BLOCKSIze is obsolete but it is accepted to provide compatibility with previous releases of OS/390 TCP/IP. See the BLKsize parameter for more information.

BUfno
Specifies the number of access method buffers that is used when data is read from or written to a data set. The valid range is 1–35. The default value is 5.

CHKptint
Specifies the checkpoint interval for the sending site in a file transfer request. If the checkpoint interval is zero, no checkpointing occurs and no marker blocks are transmitted. The default value is zero.

Use SITE to set CHKptint when the FTP server is the sending site (GET command). It will not be useful to change the checkpoint interval to more than zero in a GET unless the receiving site is able to process checkpoint markers in the file transfer data stream.

CHKptint is valid only for block and compressed modes of data transfer.

number
The checkpoint interval for the sending site in a file transfer request. This value is used to determine when checkpoint marker blocks are to be transmitted so that transmission can be restarted based on the information in the last marker.

A large checkpoint interval means that a large amount of data is sent in between markers and therefore few markers are sent. A smaller checkpoint interval means that less data is sent between markers and therefore more markers are sent.

The costs involved with using a nonzero checkpoint interval are:

- The markers themselves are transmitted, which means more bytes are being sent across the network (approximately 44 bytes per marker).
- Additional packets and acknowledgments may be required. For example, when the MVS client PUTs a file, a reply packet is sent from the server to the client and then the client must acknowledge that packet.

To estimate the appropriate checkpoint interval, use the following formula. You need to know the record length of the file you are transferring and how much data you think can be transmitted reliably.

```
           amount of data in interval
CHKPTINT = --------------------------
           record length of the file
```

Do not execute a CHKptint more often than once every 200KB. For example, if the file you are transferring has 80-byte records:

```
CHKPTINT = 200KB / 80 bytes
         = 200 * 1024 bytes / 80 bytes
         = 2560
```

CHMod
Changes the permission bits for a file.

ooo filename
ooo is the octal mask and *filename* is the name of the file whose mask is to be changed.

symbolic filename
symbolic is a value in the format accepted by the OS/390 UNIX CHMOD command (for example, a+r)

Note: Only *r, w,* and *x* permission bits are accepted in *symbolic*.

If the file name does not begin with a slash (/), it is appended to the current working directory. If the file name does begin with a slash (/), it is interpreted as a complete directory name.

The file name specified must be an HFS file name. The setting of QUOtesoverride is ignored and all quotes are treated as part of the file name.

The CHMod keyword must be the only or last keyword on a SITE subcommand.

CONDdisp
Specifies the disposition of the data set if a store operation for a new data set ends before all of the data is written.

Catlg
Specifies that a data set is kept and cataloged when an FTP file transfer ends prematurely.

Delete
Specifies that a data set is deleted when an FTP file transfer ends prematurely.

Delete is ignored if the file transfer failed as a result of the FTP server being terminated or if the server has received checkpoint information during data transfer.

CTRLConn
Specifies the ASCII code page to be used for control connections. The valid subcommands are:

```
SITE CTRLConn=7BIT
SITE CTRLConn=iconv_ascii
SITE CTRLConn=*
```

7BIT
Indicates 7-bit ASCII is to be used.

iconv_ascii
Is a name recognized by iconv to indicate an ASCII code page. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

***** Indicates that the ASCII used at initialization is to be used.

CYlinders
Specifies that primary and secondary space allocations are in cylinders.

If both PRImary and SECondary are unspecified, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the CYlinders parameter is ignored.

DATAClass
Used to specify the SMS data class, as defined by your organization, for the target host. Specifying DATAClass with no parameter value cancels the data class specification. The equal sign (=) is optional in this case.

See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

data_class
Specifies the SMS data class, as defined by your organization, for the target host. If values are specified for any of the following SITE parameters, the values specified by the SITE parameter overrides the value specified in the SMS data class:

- BLKsize
- Directory
- LRecl
- PRImary
- RECfm

- RETpd
- SECondary

If the DCbdsn SITE parameter is specified, the LRecl, RECfm, BLOCKSIze, and RETpd (if specified) of the DCBDSN data set overrides the values specified in the data class.

If the MGmtclass site parameter is specified, and the requested management class specifies a retention period, the retention period value of the management class may override the retention period value of the data class.

DATASetmode
Specifies that all the data set qualifiers below the current directory are treated as entries in the directory (disables DIRECTORYMode).

DB2
Specifies the name of the DB2 subsystem.

db2_name
The name of the DB2 subsystem.

DCbdsn
Specifies the name of the MVS data set to be used as a model for allocation of new data sets. Specifying DCbdsn with no parameter value cancels the DCbdsn specification.

data_set_name
Specifies the name of the data set. The file name must be an MVS data set name. HFS file names are not allowed on the DCbdsn parameter. The setting of QUOtesoverride is ignored. If the file name is enclosed in single quotes, it overrides the current working directory; otherwise it is appended to the current working directory.

Notes:

1. Specify SITE RECfm, LRecl, and BLKsize parameters with no values to allow characteristics from the model DCB to be used.
2. To override the model characteristics of RECfm, LRecL, BLKsize, or RETpd, specify a value on the SITE command.
3. Ensure that SENDSITE subcommand is toggled off. Otherwise, the SITE information that is sent automatically by the client overrides the values provided by the model DCB.
4. If MGmtclass is specified, the RETpd value of the MGmtclasS may override the RETpd value.

Specifying a GDG data set with a relative index produces an error message. The following examples are unsupported specifications:

```
SITE DCBDSN=MYGDG(0)
SITE DCBDSN=MYGDG(–nnn)or
SITE DCBDSN=MYGDG(+nnn)
```

See “Using a DCBDSN Model to Create a New Data Set” on page 70 for more information about DCbdsn.

DESt
Indicates the Network Job Entry (NJE) destination to which the files are routed

when you enter a PUT command. If specified without a *destination*, the destination resets and files are stored at the host system rather than sent to a remote network.

The SITE DESt subcommand enables you to send data sets (rather than storing them at the server) to other users on machines that are connected on an NJE network.

destination
Indicates the NJE destination to which the files are routed when you enter a PUT command. The value specified for destination can be:

- userID@nodeID
- nodeID.userID
- nodeID
- DestID

The file is sent over the NJE network to the specified destination.

This parameter is ignored if FILEtype=JES is set.

Directory
Specifies the number of directory blocks to be allocated for the directory of a PDS. When specified without the *size*, no directory value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify Directory without a *size* when you are also specifying DATAClass=dataclass and you want the SMS data class to provide the Directory *size*. If Directory=*size* is specified with DATAClass, the value specified by the SITE Directory parameter overrides the DATAClass directory specification.

size
Specifies the number of directory blocks to be allocated for the directory of a PDS. The valid range is 1—16777215.

DIRECTORYMode
Specifies that only the data set qualifier immediately below the current directory is treated as an entry in the directory. In directory mode, this data set qualifier is the only one used by the MGET, LS, and DIR subcommands.

DIRECTORYMode has no effect on files residing in an HFS.

FILEtype
Specifies the file type of the data set.

type
The file type of the data set can be:

Type	Description
SEQ	Sequential or partitioned data sets
SQL	SQL query function
JES	Remote job submission

JESLrecl
Specifies the logical record length (LRecl) for the Job Entry System (JES) internal reader at the foreign host.

length
The logical record length for the JES internal reader at the foreign host. The valid range is 1–254.

***** Indicates that the logical record length should be taken from the site LRecl parameter setting.

JESRecfm
Specifies the record format for the JES internal reader at the foreign host.

F Fixed record format

V Variable record format

***** Indicates that the record format should be taken from the SITE RECfm parameter setting.

LRecl
Specifies the logical record length (LRecl) of a newly allocated data set. When specified without a *length*, no LRecl is used when allocating the data set. The equal sign (=) is optional in this case.

Specify LRecl with no value when you are also specifying DATAC&=*data_class* and you want the SMS data class to provide the LRecl value, or when you are specifying DCbdsn=*data_set_name* and you want to use the LRecl from the DCBDSN data set. If LRecl=*length* is specified with either DATAClass or DCbdsn, the length specified by the SITE LRecl parameter overrides the DATAClass or DCbdsn LRecl.

length
Specifies the logical record length of a newly allocated data set. The valid range is 0—32760. A special value of x (LRecl=x) is also supported to indicate that a logical record length can exceed 32760 for variable-length spanned records.

Specifying LRecl=0 has the same effect as specifying LRecl with no parameters.

MGmtclass
Specifies the SMS management class as defined by your organization for the target host. Specifying MGmtclass with no *mgmtclass* cancels the management class specification. The equal sign (=) is optional in this case.

mgmtclass
Specifies the SMS management class as defined by your organization for the target host. If the mgmtclass specified has a setting for RETpd, the value specified by the mgmtclass may override the setting of the RETpd site parameter, the RETpd value of a model data set if the DCbdsn parameter is specified, and the RETpd value defined in an SMS data class if DATAClass is specified. See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

MIGratevol
Specifies the volume ID for migrated data sets if they do not use IBM Storage Management Subsystems. If you do not specify MIGratevol, the default *volume_serial* is MIGRAT.

volume_ID
The volume ID for migrated data.

NOASAtrans
Treats ASA file transfers as regular file transfers; that is, the ASA characters are treated as part of the data and are not converted to print control characters.

NOAUTOMount
Prevents automatic mounting of volumes for data sets on volumes that are not mounted.

NOAUTORecall
Prevents automatic recall of migrated data sets.

Note: A migrated data set can be deleted even though NOAUTORecall is specified, because migrated data sets are not recalled for deletion.

NOQUOtesoverride
Treats a single quote appearing at the beginning of the file name, as well as all other single quotes contained in the file name, as part of the actual file name. The entire file name, including the leading single quote, is appended to the current working directory.

NORDW
Specifies that Variable Record Descriptors (RDWs) are not treated as if they were part of the record and are discarded during FTP transmission of variable format data sets.

NORESTGet
Prevents opening the checkpoint data set for a GET request. Thus, checkpoint will not be active or recognized. Use this parameter when opening the checkpoint data set might cause a problem.

NOSPRead
Specifies that the output is in report format rather than spreadsheet format when the file type is SQL.

NOTRAILingblanks
Specifies that the FTP server does not preserve the trailing blanks that are in a fixed format data set when the data is sent to the foreign host.

NOUCSSUB
In Unicode-to-EBCDIC conversion, the data transfer is terminated if any Unicode character cannot be converted into the EBCDIC code set.

NOUCSTRUNC
In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is not allowed. The data transfer is aborted if the logical record length of the receiving data set is too small to contain the data after conversion to EBCDIC.

Note: The setting of the CONDdisp parameter determines what happens to the target data set if the transfer is aborted.

NOWRAPrecord
Indicates that data is truncated if no new line character is encountered before the logical record length of the receiving file is reached.

PRImary
Specifies the amount of tracks, blocks, or cylinders for primary allocation. When specified without an *amount*, no primary value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify PRImary with no value when you are also specifying DATAClass=*data_class* and you want the SMS data class to provide the PRImary *amount*.

To allow the SMS data class to determine the space allocation, both PRImary and SECondary must be specified with no value. The tracks, blocks, and cylinders setting is ignored in this case. If PRImary=*amount* is specified with DATAClass, the value specified by the SITE PRImary parameter overrides the DATAClass space allocation.

amount
Specifies the amount of tracks, blocks, or cylinders for primary allocation. For allocating partitioned data sets, this is the amount that is allocated for the primary extent.

For allocating sequential data sets this is the maximum amount that is allocated for the primary extent. If a lesser amount is needed to hold the data being transferred, only the amount actually needed to hold the data is allocated. The valid range is 1—16777215.

Qdisk
Used to display statistics about available space on a volume. If Qdisk is entered without a specific *volume_serial*, statistics about available space are displayed for each volume that is defined with “Use Attribute=storage”.

volume_serial
Displays statistics about available space on a specific volume.

QUOtesoverride
Specifies single quotes at the beginning and end of a file name should override the current working directory instead of being appended to the current working directory. This is the way single quotes are used in all previous MVS FTP servers, and is the default. Any single quotes inside the beginning and ending quote are treated as part of the file name.

QUOtesoverride indicates the usage of single quotes appearing at the beginning of, or surrounding, a file name. The setting of this keyword affects all FTP subcommands that have a path name as a parameter except keywords on the SITE subcommand.

RDW
Specifies that Variable Record Descriptors (RDWs) are treated as if they were part of the record and not discarded during FTP transmission of variable format data sets in stream mode.

Note: RDW information is stored in a binary halfword. Transfer files in binary mode to avoid translation problems that can occur if you transfer this binary field in EBCDIC or ASCII.

RECfm
Specifies the record format of a data set. When specified without the *format*, no record format is used when allocating the data set. The equal sign (=) is optional in this case.

Specify RECfm with no value when you are also specifying DATAClass=*data_class* and you want the SMS data class to provide the RECfm *format*, or when you are specifying DCbdsn=*data_set_name* and you want to use the record format from the DCBDSN data set.

If RECfm=*format* is specified with either DATAClass or DCbdsn, the value specified by the SITE RECfm parameter overrides the DATAClass or DCbdsn record format.

format
Specifies the record format of a data set. Valid record formats are: F, FA, FB, FBA, FBM, FBS, FBSA, FBSM, FM, FS, FSA, FSM, U, UA, UM, V, VA, VB, VBA, VBM, VBSA, VBSM, VBS, VM, VS, VSA, and VSM. The characters used to specify these record formats have the following meanings:

Code	Description
F	Fixed record length
V	Variable record length
U	Undefined record length
B	Blocked records
S	Spanned records (if variable) / standard records (if fixed)
A	Records contain ISO/ANSI control characters
M	Records contain machine code control characters

RETpd
Specifies the number of days that a newly allocated data set should be retained. When specified without the number of *days*, a retention period will not be specified when allocating new data sets. The equal sign (=) is optional in this case.

Specify RETpd with no value when you are also specifying DATAClass=*data_class* or MGmtclass=*mgmtclass* and you want SMS to provide the RETpd value, or when you are specifying DCbdsn=*data_set_name* and you want to use the RETpd from the DCBDSN data set. If more than one of the SITE parameters (RETpd, MGmtclass, DATAClass, or DCbdsn) are specified, the order of precedence (highest to lowest) is:

1. MGmtclass
2. RETpd
3. DCbdsn
4. DATAClass

If a retention period is associated with an SMS management or data class, or with a model DCBDSN data set, the value of the retention period may be overridden to another retention period, but it cannot be overridden to have no retention period specified for the newly created data sets.

days
Specifies the number of days that a newly allocated data set should be retained. The valid range is 0—9999. A value of zero indicates a retention period of zero days so that the data set expires the same day it was created.

Note: An attempt to either append or replace an existing data set with a retention period requires operator interaction to take place for permission to alter the data set. This is normal MVS behavior.

SBDataconn
Specifies the conversions between EBCDIC and ASCII code pages to be used for data transfers. Valid subcommands are:

```
SITE SBDataconn=dsname
SITE SBDataconn=(ebcdic_cp,ascii_cp)
SITE SBDataconn=*
```

dsname
Specifies the fully qualified name of an MVS data set or HFS file that contains the EBCDIC-to-ASCII and ASCII-to-EBCDIC translate tables generated by the CONVXLAT utility.

Notes:

1. The name must NOT be enclosed in quotes. If quotes appear, they are treated as part of the name. (QUOtesoverride is ignored.)
2. The HFS name is case-sensitive. The MVS name is not.
3. The name cannot begin with a left paren [(].
4. The SBDataconn keyword must be the only keyword or the last keyword on a SITE subcommand.
5. The translate tables being used for the data connection can also be changed by a SITE XLate subcommand.
6. SITE XLate and SITE SBDataconn are mutually exclusive.

ebcdic_cp
Specifies the name of an EBCDIC code page recognized by iconv. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

ascii_cp
Specifies the name of an ASCII code page recognized by iconv. For a list of code pages supported by iconv, refer to code set converters in the *C/C++ MVS Programming Guide*.

***** Indicates the translate tables set up at initialization for the data connection must be used.

SECondary
Specifies the amount of tracks, blocks, or cylinders for secondary allocation. When specified without the *amount* for the C server, no secondary value is used when allocating the data set. The equal sign (=) is optional in this case.

Specify SECondary with no value when you are also specifying DATAClass=*dataclass* and you want the SMS data class to provide the SECondary value. To allow the SMS data class to determine the space allocation, both PRImary and SECondary must be specified with no value. The tracks/blocks/cylinders setting is ignored in this case. If SECondary=*amount* is specified with DATAClass, the value specified by the SITE SECondary parameter overrides the DATAClass space allocation.

amount
Specifies the amount of tracks, blocks, or cylinders for secondary allocation. The valid range is 0—16777215.

SPRead
Specifies that the output is in spreadsheet format when the file type is SQL.

SQLCol
Specifies the column headings of the SQL output file.

any
The label of the DB2 SQL table column heading is the first choice for column heading, but if there is no label, the name becomes the column heading.

labels
Labels are the DB2 SQL table column headings. If any of the columns do not have labels, FTP supplies a column heading in the form of `COLnnn`.

names
Uses the names of the DB2 SQL table column headings. The labels are ignored.

STOrclass
Sspecifies the SMS storage class as defined by your organization for the target host. Cancels the storage class specification when specified without a *storage_class* parameter value. The equal sign (=) is optional in this case.

See “Specifying Values for New Data Sets” on page 68 for more information about specifying attributes when allocating new data sets.

storage_class
Specifies the SMS storage class as defined by your organization for the target host. The SMS storage class may override settings for the VOLume or Unit site parameters.

TRacks
Specifies that primary and secondary space allocations are in tracks. If both PRImary and SECondary are unspecified, and an SMS data class has been specified, the space allocation is determined by the SMS data class and the TRacks parameter is ignored.

TRAILingblanks
Specifies that the FTP server preserves the trailing blanks that are in a fixed format data set when the data is retrieved from a foreign host. —

UCSHOSTCS
Specifies the EBCDIC code set to be used when converting to and from Unicode. If you do not specify a *code_set*, the current code set is used.

code_set
Name of the EBCDIC code set to be used when converting to and from Unicode.

UCSSUB
In Unicode-to-EBCDIC conversion, the EBCDIC substitution character is used to replace any Unicode character that cannot successfully be converted. Data transfer continues.

UCSTRUNC
In Unicode-to-EBCDIC conversion, truncation of EBCDIC data is allowed. The data transfer continues even if EBCDIC data is truncated. **Caution:** If the EBCDIC data contains any double-byte data, truncation may not honor character boundaries and EBCDIC records may not end in Shift-in state.

UMask
Defines the file mode creation mask. The file mode creation mask defines which permission bits are *not* to be set on when a file is created. When a file is created, the permission bits requested by the file creation are compared to the file mode creation mask, and any bits requested by the file creation which are disallowed by the file mode creation mask are turned off.

The format of the UMask keyword is **UMASK** *ooo*.

When a file is created, the specified permission bits for the file are 666 (-rw-rw-rw-). If the file mode creation mask is 027, the requested permissions and the file mode creation mask are compared:

```
110110110   - 666
000010111   - 027
---------
11010000    - 640
```

The actual permission bits set for the file when it is created is 640 (-rw-r-----).

Notes:

1. The default value for UMask is 027.
2. You cannot use FTP to create HFS files with execute permissions. If you require execute permissions, use the LOCSITE CHmod command to change permissions after the file has been created.

Unit
Specifies the unit type for allocation of new data sets.

The setting for Unit may be overridden by the SMS storage class, if one is specified. Threrefore, Unit should not be coded if an SMS storage class is in use.

unit_type
The unit type (for example, 3380) for the allocation of new data sets on direct access devices. If *unit_type* is not specified, the unit type used for allocation is set back to the system default.

VOLume
Specifies the volume serial number for allocation of new data sets. If *volume_serial* is not specified, the volume serial number used for allocation is set back to the system default.

The setting for VOLume may be overridden by the SMS storage class, if one is specified. Therefore, VOLume should not be coded if an SMS storage class is in use.

The MVS FTP server identifies multiple commands issued with a single-site command by the white space. For example: site vol=fffff is a single-site command; however, site vol = fffff is treated by the server as 3 different commands.

volume_serial
The volume serial number for allocation of new data sets.

WRAPrecord
Indicates that data is wrapped to the next record if no new line character is encountered before the logical record length of the receiving file is reached.

XLate
Indicates the desired translate table to be used for the data connection. Valid subcommands are:

```
SITE XLate=name
SITE XLate=*
```

name
Specifies the name that corresponds to the desired translate table data set. The corresponding data set name is *hlq.name*.TCPXLBIN unless

environment variable _FTPXLATE_*name* = *dsn* was defined for the FTP server to override the data set name. In that case, *dsn* is the data set used.

***** Indicates the translate tables set up at initialization for the data connection are to be used.

Notes:

1. The translate tables being used for the data connection can also be changed by a SBDataconn subcommand.
2. SITE XLate and SITE SBDataconn are mutually exclusive.

Usage

- Because more than one parameter can be specified with the SITE subcommand, *parameter* can be repeated many times, with each *parameter* separated by a blank space.
- If the remote host is using TCP/IP, use the SITE subcommand to set data set routing and send data set allocation attributes to the host. The site-dependent information sent with the SITE subcommand remains active until you issue a new SITE subcommand. The new SITE subcommand adds to, or changes, the attributes established by previous SITE subcommands.
- If you specify one or more incorrect parameters with the SITE subcommand, an error message specifying the incorrect parameter is displayed. All correct parameters are set, regardless of any incorrect parameters, and do not need to be reissued.
- If the data sets are sent from an MVS host, use the SENDSITE subcommand to automatically invoke the SITE subcommand.

  If the local host does not support the SITE subcommand, site information can be sent to the MVS FTP server from the client using the QUOTE subcommand.
- Use the HELP SERVER SITE command to get information on the SITE parameters supported by a server.

Context

- To check the effect of the SITE subcommand on the attributes at the foreign host, see “STATUS Subcommand—Retrieve Status Information from a Remote Host” on page 214.
- See “SENDSITE Subcommand—Toggle the Sending of Site Information” on page 196 for more information about the SENDSITE subcommand.
- See “QUOTE Subcommand—Send an Uninterpreted String of Data” on page 187 for more information about the QUOTE subcommand.
- Refer to the *JCL Reference Manual* for more information about some of the SITE and LOCSITE parameters.

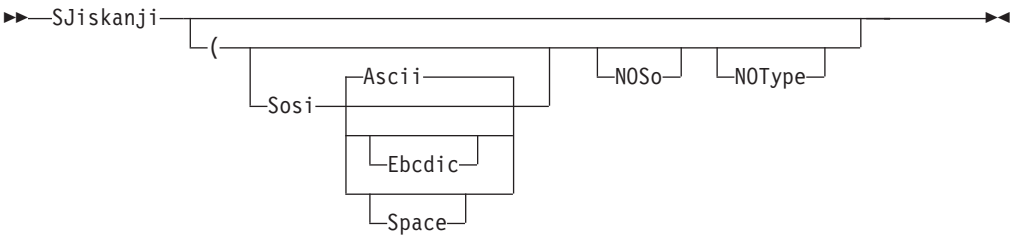
SJISKANJI Subcommand—Change the Data Transfer Type to SJISKANJI

Purpose

Use the SJISKANJI subcommand to change the data transfer type to SJISKANJI.

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Examples

To cause the FTP client to change its transfer type to Shift JIS kanji, without sending a TYPE command to the FTP server, use:

```
SJISKANJI (NOTYPE
```

The server in this example should be set to the ASCII transfer type before the (NOTYPE subcommand is issued.

Usage

- The SJISKANJI client subcommand is equivalent to the TYPE B 1 server command.
- The SJISKANJI (Sosi or SJISKANJI (Sosi ASCII client subcommands are equivalent to the TYPE B 1 S A server command.
- The SJISKANJI (Sosi EBCDIC client subcommand is equivalent to the TYPE B 1 S E server command.
- The SJISKANJI (Sosi SPACE client subcommand is equivalent to the TYPE B 1 S S server command.
- The SJISKANJI (NOSO client subcommand is equivalent to the TYPE B 1 N server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

STATUS Subcommand—Retrieve Status Information from a Remote Host

Purpose

Use the STATUS subcommand to retrieve current configuration information from the FTP server. This information includes the current settings of the configuration variables, which can be initialized in the FTP.DATA data set or changed using various FTP subcommands. For information on the parameters of the FTP.DATA data set, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

Format

Parameters

name
Specifies the file or directory for which status information is requested. The *name* parameter is not supported by the MVS FTP server.

Examples

Retrieve the status information from an FTP server:

```
status
 >>>STAT
 211-Server FTP talking to host 9.67.112.25, port 1084
 211-User: USER21  Working directory: /u/user121/example
 211-The control connection has transferred 11015 bytes
 211-There is no current data connection.
 211-The next data connection will be actively opened
 211-to host 9.67.112.25, port 1084,
 211-using Mode Stream, Structure File, type ASCII, byte-size 8
 211-Automatic recall of migrated data sets.
 211-Automatic mount of direct access volumes.
 211-ASA control characters in ASA files opened for text processing
 211-will be transferred as ASA control characters.
 211-Trailing blanks are removed from a fixed format
 211-data set when it is retrieved.
 211-Data set mode.  (Do not treat each qualifier as a directory.)
 211-Primary allocation 5 tracks.  Secondary allocation 2 tracks.
 211-Partitioned data sets will be created with 15 directory blocks.
 211-FileType SEQ (Sequential - default).
 211-Number of access method buffers is 5
 211-RDWs from variable format data sets are discarded.
 211-Site DB2 subsystem name is DB2
 211-Data not wrapped into next record.
 211-JESLRECL is 80
 211-JESRECFM is Fixed
 211-SMS is active.
 211-Mgmtclass for new data sets is TCPMGMT
 211-New data sets will be cataloged if a store operation terminates abnormally
 211-Single quotes will override the current working directory.
 211-UMASK value is 027
 211-Process id is 67108870
 211-Checkpoint interval is 0
 211-Record format VB, Lrecl: 256, Blocksize: 6233
 211 ***end of status***
Command:
```

Usage

- The retrieved status information can be a directory, a file, or general status information, such as a summary of activity. If *name* is omitted, general status information is retrieved.
- Table 23 lists the messages that are returned from the OS/390 UNIX FTP server for the STATUS command. Within the table, the STATUS messages are listed in the order they appear, followed by:
  - Their usage and variable values
  - Whether the message is always or conditionally displayed
  - Where the values are set during server initialization
  - Where the values can be changed during a client session

For further information on setting values for server initialization in the FTP.DATA data set, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

Table 23. STATUS Subcommand Output

Message/Explanation	Display Conditions
211-Server FTP talking to host *ip_address* port *port*	
The client remote host *ip_address* and *port*.	Always displayed.
211-User: *userid* Working directory: *directory*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
The *userid* that the client is currently logged in as and the *directory* the client is currently using. If the client has not entered a valid login (using the USER and PASS subcommand), *userid* is `* NONE *`. If the client is logged in as the anonymous user, *userid* is `Anonymous`. If the client has entered a valid user ID and password, *userid* is the user ID entered by the client on the USER subcommand. If the client currently has no working directory, *directory* is displayed as `* NONE *` If the current working directory is an HFS mount point, the working directory is displayed as that mount point. If the current working directory is an MVS high level qualifier, the working directory is displayed as that high level qualifier, followed by a period. If the current working directory is an MVS partitioned data set, the current working directory is displayed, prefaced by the words partitioned data set.	Always displayed. *userid* can be changed by the USER and PASS subcommands. *directory* can be changed by the CWD subcommand.
211-There is an RNFR pending for '*filename*'	
A rename is in progress for *filename*, and the RNTO subcommand has not been received yet.	Conditionally displayed when a rename is in progress.
211-The control connection has transferred *number* bytes	
The total number of bytes that have been sent or received over the control connection between the client and the server.	Always displayed.
211-The current data connection has transferred: *number* bytes 211-using Mode *mode*, Structure: *structure*, Type: *type*, byte-size 8	
The *number* of bytes sent or received over the data connection. The *mode* being used for the data transfer. The data set *structure* currently being used by the server for transfer. The data transfer *type* currently being used by the server for transfer.	Conditionally displayed when there is a currently active data connection. *mode* can be changed by the MODE subcommand. *structure* can be changed by the STRU subcommand. *type* can be changed by the TYPE subcommand.
211-There is no current data connection.	
There is currently no data connection between the client and the server.	Conditionally displayed when there is no data connection.
211-The next data connection is *open_mode* opened	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
Indicates how the next data connection is opened. If *open_mode* is actively, the server issues the connect request to the client, which should be in listen state on the client data port. If *open_mode* is passively, the server listens on the data port for the client to establish the data connection.	Always displayed.
211-to host *IP_address*, port *port*,	
The remote host *IP_address* and *port* the next data connection will connect to.	Conditionally displayed when the next data connection is opened actively.
211-using Mode *mode*, Structure *structure*, type *type*, byte-size 8	
The *mode* used for the next data transfer. The *structure* used by the server for file transfer. The data transfer *type* used by the server for file transfer.	
211-TYPE U data is converted to/from <codeset>	
For in-bound data transfer, data is converted from UCS-2 to the EBCDIC codeset <codeset>. For out-bound data transfer, EBCDIC data is converted to Unicode encoding using <codeset> to UCS-2 conversion.	Conditionally displayed when transfer type is UCS-2. <codeset> can be changed with the SITE UCSHOSTCS parameter.
211–UCS Substitution: <OFF|ON>, UCS Truncation <OFF|ON>	
UCS Substitution OFF: Inbound data transfer is aborted if data contains UCS-2 code point that cannot be converted into the EBCDIC codeset indicated on the previous line.	Conditionally displayed when transfer type is UCS-2.
UCS Substitution ON: Data transfer is completed using the EBCDIC substitution character whenever a UCS-2 code point cannot be converted.	Use of substitution can be changed using the SITE UCSTRUNC/NOUCSSUB parameters.
UCS Truncation OFF: Inbound data transfer is aborted if the record size of the target data set is not large enough to contain a line of data after it is converted to EBCDIC.	Use of truncation can be changed using the SITE UCSTRUNC/NOUCSSTRUNC parameters.
UCS Truncation ON: Data transfer is completed even if the EBCDIC data is truncated because the record size of the target data set is too small. WARNING: If the data contains double-byte characters, truncation may not honor character boundaries.	
211-Byte Order: big-endian	
Unicode (UCS-2) encoding is assumed to be in big-endian byte order.	Conditionally displayed when the data transfer type was set to UCS-2 by a TYPE U 2 or TYPE U 2 B command.
211-Byte Order: little-endian	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
Unicode (UCS-2) encoding is assumed to be in little-endian byte order.	Conditionally displayed when the data transfer type was set to UCS-2 by a TYPE U 2 L command.
211-Automatic recall of migrated data sets.	
Migrated data sets are recalled when they are accessed for data transfer or file rename. The AUTORecall setting does not apply when deleting an entire migrated data set. If a DELE subcommand is entered to delete a migrated data set, the data set is not recalled prior to deletion. However, if the DELE subcommand is entered to delete a member of a migrated partitioned data set, the data set is recalled.	Conditionally displayed when AUTORecall has been set to TRUE and has not been overridden with a SITE NOAUTORecall subcommand. Set by specifying either: • the AUTORecall statement in the FTP.DATA data set • the AUTORecall FTP parameter Changed by issuing the SITE subcommand with the AUTORecall parameter.
211-No Automatic recall of migrated data sets.	
Migrated data sets are not automatically recalled when accessed for data transfer or file rename. Subcommands issued for a migrated data set are rejected. NOTE: The NOAUTORecall setting does not apply when deleting an entire migrated data set. If a DELE subcommand is entered to delete a migrated data set, the data is not recalled prior to deletion; the data set is deleted even though the data set was migrated and NOAUTORecall was specified. However, if the DELE subcommand is used to delete a member of a migrated partitioned data set, the data set is not recalled and the DELE subcommand is rejected.	Conditionally displayed when AUTORecall has been set to FALSE and has not been overridden with a SITE AUTORecall subcommand. Set by specifying: • the AUTORecall statement in the FTP.DATA data set • the NOAUTORecall FTP parameter Changed by issuing the SITE subcommand with the AUTORecall parameter.
211-Automatic mount of direct access volumes.	
If a data set resides on a volume that is not mounted, the operator is automatically prompted to mount the necessary volume on the MVS system. The FTP server waits for the operator to mount the volume and reply to the prompt at the operator console before continuing.	Conditionally displayed when AUTORecall has been set to TRUE and has not been overridden with a SITE NOAUTOMount subcommand. Set during server initialization by specifying either: • the AUTOMount statement in the FTP.DATA data set • the AUTOMount FTP parameter Changed by issuing the SITE subcommand with the AUTOMount parameter.
211-No Automatic mount of direct access volumes.	
If a data set resides on a volume that is not mounted, the FTP subcommand is rejected.	Conditionally displayed when AUTOMount has been set to FALSE and has not been overridden with a SITE AUTOMount subcommand. Set by specifying either: • the AUTOMount statement in the FTP.DATA data set • the NOAUTOMount FTP parameter Changed by issuing the SITE subcommand with the NOAUTOMount parameter.
211-ASA control characters in ASA files opened for text processing will 211-be converted to C control character sequences during file transfer.	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
If the record format (RECfm) of the file indicates that the file contains ISO/ANSI control characters (RECfm with A), the ISO/ANSI control characters are converted to C control character sequences during file transfer. Files stored at the FTP server host are stored with the converted characters rather than the ISO/ANSI control characters.	Conditionally displayed when ASAtrans has been set to TRUE and has not been overridden by the SITE NOASAtrans subcommand. Set by specifying the ASAtrans statement in the FTP.DATA data set.
211-ASA control characters in ASA files opened for text processing 211-is transferred as ASA control characters.	
If the record format (RECfm) of the file indicates that the file control ISO/ANSI control characters (RECfm with A), the ISO/ANSI control characters remain unchanged during file transfer. Files stored at the FTP server host are stored with the ISO/ANSI control characters.	Conditionally displayed when ASAtrans has been set to FALSE and has not been overridden by the SITE ASAtrans subcommand. Set during server initialization by specifying the ASAtrans statement in the FTP.DATA data set. Changed during a client session by issuing the SITE subcommand with the NOASAtrans parameter.
211-Trailing blanks are removed from a fixed format 211-data set when it is retrieved.	
If a RETR subcommand is issued to retrieve a data set with a fixed record format, any trailing blanks at the end of each record in the data set are removed before the record is sent to the client.	Conditionally displayed when TRAILingblanks has been set to FALSE and has not been overridden with a SITE TRAILingblanks subcommand. Set by specifying the TRAILingblanks statement in the FTP.DATA data set. Changed during a client session by issuing the SITE subcommand with the NOTRAILingblanks parameter.
211-Trailing blanks are not removed from a fixed format 211-data set when it is retrieved.	
If a RETR subcommand is issued to retrieve a data set with a fixed record format, any trailing blanks at the end of each record of the data set are included with the record that is sent to the client.	Conditionally displayed when TRAILingblanks has been set to TRUE and has not been overridden by the SITE NOTRAILingblanks subcommand. Set during server initialization by specifying the TRAILingblanks statement in the FTP.DATA data set. Changed during a client session by issuing the SITE subcommand with the TRAILingblanks parameter.
211-Data set mode. (Do not treat each qualifier as a directory.)	
All the data set qualifiers below the current directory are treated as entries in the directory.	Conditionally displayed when DATASetmode has been set to TRUE and has not been overridden by the SITE subcommand with the DIRECTORYMode parameter. Set by specifying the DATASetmode statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the DATASetmode parameter.

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
211-Directory mode. (Treat each qualifier as a directory.)	
Only the data set qualifier immediately below the current directory is treated as an entry in the directory. Directory mode applies only to MVS PS and PDS data sets.	Conditionally displayed when DATASetmode has been set to FALSE, or DIRECTORYMode has been set to TRUE and has not been overridden by the SITE DATASetmode subcommand. Set by specifying the DATASetmode statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the DIRECTORYMode parameter.
211-Primary allocation *p_amt spacetype*. Secondary allocation *s_amt spacetype* - or - 211-Primary allocation *p_amt spacetype*. - or - 211-Secondary allocation *s_amt spacetype*	
These messages display the primary and secondary allocation information used when allocating new data sets. • *spacetype* indicates whether the primary and secondary space is allocated as BLocks, TRacks, or CYlinders • *p_amt* is the number of BLocks, TRacks, or CYlinders allocated for the primary extent. • *s_amt* is the number of BLocks, TRacks, or CYlinders allocated for each secondary extent. If primary or secondary, or both, space information is displayed, these settings override any space allocation information contained in an SMS DATAClass. Both primary and secondary allocation information must be unspecified for SMSDATA CLASS space allocation information to be used.	Only one line is displayed conditionally when values have been set. Set by specifying the FTP.DATA data set statement: • PRImary p_amt • SECondary s_amt • SPACETYPE Set by issuing the subcommand: • SITE PRImary=p_amt • SITE SECondary=s_amt • SITE with the BLocks, TRacks, or CYlinders parameter.
211-Partitioned data sets are created with *number* directory blocks.	
The number of directory blocks allocated for a new partitioned data set. If this is displayed in the STAT subcommand output, a value has been specified for Directory and this value overrides any value specified for directory blocks in an SMS DATAClass. To use the value specified for directory blocks in an SMS DATAClass, specify either Directory with no value in the FTP.DATA data set, or specify the SITE Directory subcommand with no value.	Conditionally displayed when a value has been specified for directory blocks and was not overridden by the SITE Directory subcommand. Set by specifying the Directory statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the Directory parameter.
211-FileType SEQ (Sequential - default)	
The server is currently operating in the SEQ (sequential) file type mode. FTP subcommands process MVS sequential or partitioned data sets, or partitioned data set members.	Conditionally displayed when FILEtype has been set to SEQ and has not been overridden by the SITE FILEtype=JES or SITE FILEtype=SQL subcommand. Set by specifying the FILEtype statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the FILEtype parameter.
211-FileType JES (MVS job spool) JES name is *jesname*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
The server is currently operating in the JES (MVS job spool) file type mode with *jesname* as the name of the JES subsystem. When the server file type is JES, FTP subcommands can submit, list, or delete JES jobs.	Conditionally displayed when FILEtype has been set to JES and has not been overridden by the SITE FILEtype=SEQ or SITE FILEtype=SQL subcommand. Set by specifying the FILEtype statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the FILEtype parameter.
211-FileType SQL (returns results of DB2 query)	
The server is currently operating in the SQL (DB2 query) file type mode. When the server file type is SQL, FTP subcommands can submit DB2 queries and retrieve the resulting DB2 output.	Conditionally displayed when FILEtype has been set to SQL and has not been overridden by the SITE FILEtype=SEQ or SITE FILEtype=JES subcommand. Set by specifying the FILEtype statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the FILEtype parameter.
211-Number of access method buffers is *number*	
The number of access method buffers that are used when data is read from or written to a data set. The valid range is 1 through 255.	Always displayed. Set by specifying the BUfno statement in the FTP.DATA data set. Default is 5. *number* can be changed by issuing the SITE subcommand with the BUfno parameter.
211-RDWs from variable format data sets are retained as part of the data.	
Variable Record Descriptors (RDW) are treated as if they are part of the record and not discarded during FTP transmission of variable format data sets in stream mode. RDW information is stored in binary halfword. Transfer files in binary mode to avoid translation problems that can occur if you transfer this binary field in EBCDIC or ASCII.	Conditionally displayed when RDW has been set to TRUE and has not been overridden with the SITE NORDW subcommand. Set during server initialization by specifying the RDW statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the RDW parameter.
211-RDWs from variable format data sets are discarded.	
NORDW specifies that the Variable Record Descriptors (RDW) are discarded during FTP transmission of variable format data sets.	Conditionally displayed when RDW has been set to FALSE and has not been overridden with the SITE RDW subcommand. Set during server initialization by specifying the RDW statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the NORDW parameter.
211-Retention period is *number_of_days*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
A retention period of *number_of_days* is assigned to a newly created data set. A retention period of 0 days is not the same as no retention period. If this line is not displayed, no retention period is assigned to newly created data sets. If this line is displayed, and *number_of_days* has been set to 0, a retention period of 0 days is assigned to newly created data sets. Consequently, newly created data sets expire the same day they are created. If this line is displayed, a retention period has been specified; this value overrides the retention period specified in an SMS data class. If an SMS management class has been specified, the retention period defined in the SMS management class may override the *number_of_days* specified by the FTP server.	Conditionally displayed when a value has been set for RETpd and has not been overridden by the SITE RETpd subcommand. Set by specifying the RETpd statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the RETpd parameter.
211-Site DB2 subsystem name is *db2name*	
SQL queries are sent to the DB2 subsystem indicated by *db2name*.	Always displayed. Set by specifying the DB2 statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the DB2 parameter.
211-Wrapping data into next record.	
When the FTP server stores data at the server host, data wraps to a new record if it reaches the logical record length before encountering a new line character.	Conditionally displayed when WRAPrecord has been set to TRUE and has not been overridden by the SITE NOWRAPrecord subcommand. Set by specifying the WRAPrecord statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the WRAPrecord parameter.
211-Data not wrapped into next record.	
When the FTP server stores data at the server host, data is truncated if it reaches the logical record length before encountering a new line character.	Conditionally displayed when WRAPrecord has been set to FALSE and has not been overridden by the SITE WRAPrecord subcommand. Set by specifying the WRAPrecord statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the NOWRAPrecord parameter.
211-SQL results sent in *report_format* format.	
The format of SQL query output.	Conditionally displayed when FILEtype is set to SQL. Set by specifying the SPRead statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with either the SPRead or NOSPRead parameter.

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
211-SQLCol (column headings) use *heading*	
The column headings for the SQL output.	Conditionally displayed when FILEtype is set to SQL. Set by specifying the SQLCol statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the SQLCol parameter.
211-JESLrecl is *lrecl*	
The logical record length for the JES internal reader at the server host system. If *lrecl* is *, the JES logical record length is the same as the logical record length used to allocate new data sets.	Always displayed. Set by specifying the JESLrecl statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the JESLrecl parameter.
211-JESRecfm is *recfm*	
The record format for the JES internal reader at the server host system. If *recfm* is *, is truncated if it the JES record format is the same as the record format used to allocate new data sets.	Always displayed. Set by specifying the JESRecfm statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the JESRecfm parameter.
211-Migrate Volid is *volid*	
The FTP server recognizes a data set with a volser of *volid* as a migrated data set.	Conditionally displayed when *volid* for a migrated data set is not MIGRAT. Set by specifying the MIGratevol statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the MIGratevol parameter.
211-Xlate name is *name*	
Conversion for the data connection is using translate tables from either the data set name defined by environment variable _FTPXLATE_*name* or the default data set name *hlq*.*name*.TCPXLBIN.	Conditionally displayed when an XLate specification is in effect. Set by: • XLATE*name* in FTP.DATA • SITE SLATE=*name* subcommand Can also be reset/changed by SITE SBDataconn subcommand.
211-CCXlate name is *name*	
Conversion for the control connection is using translate tables from either the data set name defined by environment variable _FTPXLATE_*name* or the default data set name *hlq*.*name*.TCPXLBIN.	Conditionally displayed when an CCXLATE specification is in effect. Set by: • CCXLATE*name* in FTP.DATA Can also be reset/changed by SITE CTRLConn subcommand.
211-NJE Destination is User *userid* Node *node* - or - 211-NJE Destination is *destination*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
Files transferred to the FTP Server are forwarded to the specified *userid* at the specified *node*, or to the specified *destination*, rather than stored at the host.	Conditionally displayed when a remote destination has been specified for server files. Set by one of the following: • The SITE DESt=dest subcommand has been issued. • The FTP.DATA data set specified DESt=dest and this setting has not been overridden with a SITE DESt subcommand.
211-SMS is active.	
The FTP server has determined that SMS is active on the server host MVS system.	Conditionally displayed when SMS is active.
211-SMS is not active.	
The FTP server has determined that SMS is not active on the server host MVS system.	Conditionally displayed when SMS is not active.
211-Mgmtclass for new data sets is *mgmtclass*	
The specified SMS Management class, *mgmtclass*, is used when allocating a new data set. If a retention period was defined in the specified SMS management class, this value may override the retention period specified by the FTP server.	Conditionally displayed when an SMS management class has been specified for the server and this setting has not been overridden by the SITE MGmtclass subcommand. Set by specifying the MGmtclass statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the MGmtclass parameter.
211-Storclass for new data sets is *storclass*	
SMS storage class used when allocating a new data set. The specified SMS storage class may override the FTP server settings for the volume and unit for new data set allocation.	Conditionally displayed when an SMS storage class has been specified for the server and this setting has not been overridden by the SITE STORclass subcommand. Set by specifying the STORclass statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the STORclass parameter.
211-Dataclass for new data sets is *dataclass*	
SMS Dataclass used when allocating a new data set. The message STATUS value for the following parameters overrides the corresponding parameter in the specified SMS data class: • BLKsize • Directory • LRecl • PRImary • RECfm • RETpd • SECondary	Conditionally displayed when an SMS data class has been specified for the server and this setting has not been overridden by the SITE DATAClass subcommand. Set by specifying the DATAClass statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the DATAClass parameter.
211-Data sets are allocated on *volser.*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
The volume on which new data sets are allocated. If no volume was specified for data set allocation, new data sets are allocated on the system default volume. If an SMS storage class has been specified, the SMS storage class may override the setting of *volser*.	Conditionally displayed when a volume has been specified for data set allocation and this setting was not overridden by the SITE VOLume subcommand. Set by specifying the VOLume statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the VOLume parameter.
211-Data sets are allocated using unit *unit_name.*	
The unit on which new data sets are allocated. If an SMS storage class has been specified, the SMS storage class may override the setting of *unit_name*	Conditionally displayed when a unit has been specified for data set allocation and the setting has not been overridden with a SITE Unit subcommand. Set by specifying the Unit statement of the FTP.DATA data set. Changed by issuing the SITE subcommand with the Unit parameter.
211-New data sets are *deleted* if a store operation terminates abnormally	
This line indicates the *disposition* of new data sets if the STOR, STOU, or APPE subcommand terminates while the data is being stored in a newly created data set at the server. If *disposition* is cataloged, the new data set is kept and cataloged. Data in the data set may be incomplete. If *disposition* is deleted, the server deletes the new data set.	Always displayed. Set by specifying the CONDdisp statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the CONDdisp parameter.
211-Single quotes overrides the current working directory.	
Single quotes appearing at the beginning and end of a file name are interpreted to mean that the file name contained inside the single quotes should override the current working directory instead of being appended to the current working directory.	Conditionally displayed when QUOtesoverride has been set to TRUE and has not been overridden with the SITE NOQUOtesoverride subcommand. Set by during server initialization by specifying the QUOtesoverride statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the QUOtesoverride parameter.
211-Single quotes are treated as part of the file name.	
A single quote at the beginning, as well as other single quotes contained in the file name, are treated as part of the actual file name. The entire file name including the leading single quote is appended to the current working directory.	Conditionally displayed when NOQUOtesoverride has been set to TRUE and has not been overridden with the SITE QUOtesoverride subcommand. Set by during server initialization by specifying the QUOtesoverride statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the NOQUOtesoverride parameter.
211-UMask value is *umask_value*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
umask_value is the octal mask that specifies which permission bits must be set OFF for a newly created file.	Always displayed. Set during server initialization by specifying the UMask statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the UMask parameter.
211-Process id is *pid number*	
The process identifier for the current client session logged in to the FTP server.	Always displayed.
211-Checkpoint interval is *interval*	
The *interval* at which the server sends checkpoint markers to the client when the client is retrieving a file. Checkpoint markers are sent only if the file transfer is done in EBCDIC in BLOCK or COMPRESSED mode. An *interval* of zero indicates that no checkpoint markers are sent.	Always displayed. Set by specifying the CHKptint statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the CHKptint parameter.
211-*'dsn'* is used as the DCBDSN model data set	
Data set used as a model DCB when allocating new data sets. Unless otherwise specified, the LRecl, RECfm, and BLKsize of *dsn* are used for the newly allocated data set. If the STAT subcommand output shows that a value has been specified for the LRecl, RECfm, or BLKsize parameters, these values override the information obtained from the model DCB data set. To use the BLKsize, LRecl, and RECfm of the model DCB data set, *dsn*, the LRecl, BLKsize, and RECfm parameters must be unspecified by one of the following methods: Specify the BLKsize, RECfm, and LRecl keywords in the FTP.DATA data set with no value Issue the SITE LRecl RECfm BLKsize subcommand with no values for the parameters.	Conditionally displayed when a model DCB has been specified for the server and this setting has not been overridden by issuing the SITE DCbdsn subcommand. Set by specifying the DCbdsn statement in the FTP.DATA data set. Changed by issuing the SITE subcommand with the DCbdsn parameter.
211 Record format *recfm*, Lrecl: *lrecl*, Blocksize: *blksize* - or - 211 Record format *recfm*, Lrecl: *lrecl* - or - - 211 Record format *recfm*, Blocksize: *blksize* - or - 211 Lrecl: *lrecl*, Blocksize: *blksize* - or - 211 Record format *recfm* - or - 211 Lrecl: *lrecl* - or - 211 Blocksize: *blksize*	

Table 23. STATUS Subcommand Output (continued)

Message/Explanation	Display Conditions
The logical record length, block size, and record format assigned to newly allocated data sets. If one of these lines is displayed, a value was specified for one or more of LRecl, BLKsize, and RECfm. Any values specified for LRecl, BLKsize, or RECfm override the corresponding value of a model DCB or SMS data class, if specified. If values for LRecl, BLKsize, or RECfm are not included in the message, no values have been specified and the value is obtained from a model DCB data set, or from an SMS data class, if one has been specified.	Conditionally displayed when values have been specified. Set by specifying the FTP.DATA data set statement: • LRecl • BLKsize • RECfm Changed by issuing: • the SITE subcommand with the LRecl parameter • the SITE subcommand with the BLKsize parameter • the SITE subcommand with the RECfm parameter
211 ***end of status***	
The FTP server has completed the command.	Always displayed.

STREAM Subcommand—Set the Stream Data Transfer Mode

Purpose

Use the STREAM subcommand to set the data transfer mode to stream mode. This is equivalent to specifying the MODE S subcommand. See “MODE Subcommand—Set the Data Transfer Mode” on page 174 for more information.

Format

Parameters

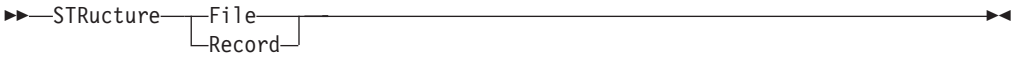
There are no parameters for this subcommand.

STRUCTURE Subcommand—Set the File Structure

Purpose

Use the STRUCTURE subcommand to set the file structure.

Format



Parameters

File
Sets the file structure to file. The file structure affects both the transfer mode and the interpretation and storage of the data set or file. With a file structure of F, the data being transferred is considered to be a continuous sequence of data bytes.

Record
Sets the file structure to record.

SUNIQUE Subcommand—Toggle the Storage Method

Purpose

Use the SUNIQUE subcommand to toggle the method of storing files on the foreign host.

Format

Use the SUNIQUE subcommand in the following format:

Parameters

There are no parameters for this subcommand.

Usage

- By default, SUNIQUE is toggled off, and FTP uses a store command (STOR) with the PUT and MPUT subcommands. If the foreign host already has a data set or file with the name specified by *foreign_file*, the foreign host overwrites the existing data set or file.
- If SUNIQUE is toggled on, FTP uses a store-unique command (STOU) with the PUT and MPUT subcommands, and prevents you from overwriting or erasing the existing data set or file on the foreign host. The created foreign data set or file is stored with a unique name. FTP sends the unique name of the created foreign data set or file to the local host, where the data set or file name is displayed on your terminal.

SYSTEM Subcommand—Display the Operating System Name

Purpose

Use the SYSTEM subcommand to display the name of the remote host operating system. The remote host must have also implemented the SYSTEM subcommand.

Format

Parameters

There are no parameters for this subcommand.

Usage

- If the MVS FTP server is the V3R2 FTP server, 215 MVS is the operating system of this server.
- If the MVS FTP server is the V3R1 Pascal FTP server, 215 OS/MVS is the operating system of this server.
- If the MVS FTP server is the V3R1 C FTP server, 215 MVS is the operating system of this server. FTP Server is the C-server.
- If the MVS FTP server is the V3R2 C FTP server, 215 MVS is the operating system of this server.
- If the MVS FTP server is the MVS OS/390 UNIX FTP server, 215 MVS is the operating system of this server. FTP Server is the C-server running on OS/390 UNIX System Services.

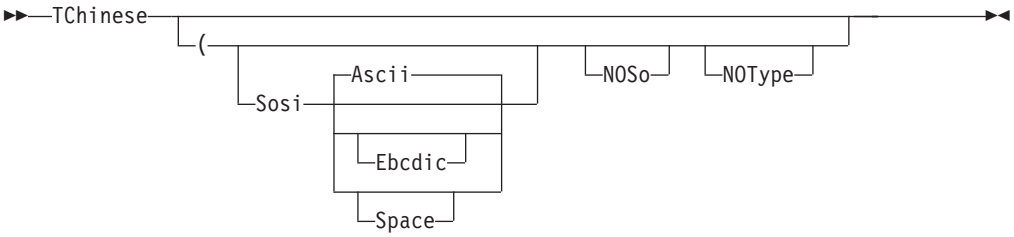
TCHINESE Subcommand—Change the Data Transfer Type to TCHINESE

Purpose

Use the TCHINESE subcommand to change the data transfer type to Traditional Chinese (5550).

MVS FTP uses the same SBCS translate table for single-byte or double-byte data transfers. If you require an alternate SBCS table for a double-byte transfer, use the SITE/LOCSITE SBDataconn or SITE XLate subcommand to have the server (or client) change the SBCS translation for the data connection.

Format



Parameters

Sosi
Transferred data contains the shift-out and shift-in characters specified by one of the following parameters — ASCII, EBCDIC or SPACE. If no parameter is specified, ASCII is used as the default.

If the S parameter is not specified at all, shift-out/shift-in characters are not used in the transferred data.

Ascii
When combined with the Sosi parameter, causes shift-out and shift-in characters X'1E' and X'1F' to be used to delimit DBCS strings in ASCII data.

Ebcdic
When combined with the Sosi parameter, causes shift-out and shift-in characters X'0E' and X'0F' to be used to delimit DBCS strings in ASCII data.

Space
When combined with the Sosi parameter, causes shift-out and shift-in characters X'20' and X'20' (ASCII spaces) to be used to delimit DBCS strings in ASCII data.

NOSo
Specifies that the data transferred is pure DBCS (data with no SBCS characters) and that the data is to be transferred to and from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

NOType
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when translation is to be done by the FTP client only.

Usage

The TCHINESE client subcommand is equivalent to the TYPE B 7 server command.

Context

See “FTP with DBCS Support” on page 97 for more information.

TSO Subcommand—Use TSO Commands

Purpose

Use the TSO subcommand to pass a Time Sharing Option (TSO) command to a local host TSO environment.

Format

Parameters

command_line
Specifies a TSO command. Do not use synonyms.

Usage

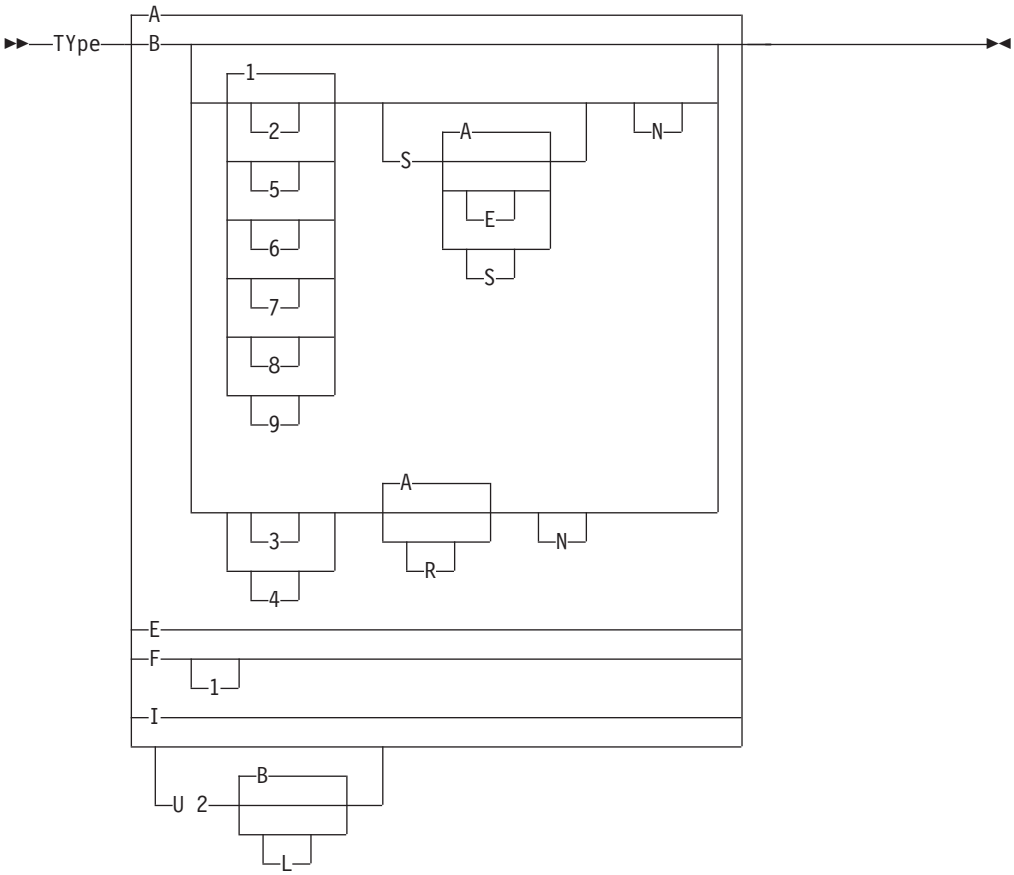
The TSO subcommand is not available from batch.

TYPE Subcommand—Set the Data Transfer Type

Purpose

Use the TYPE subcommand to set the data transfer type for the client and server at the same time with one command. FTP supports the ASCII, EBCDIC, image (binary), Unicode, and two DBCS data transfer types.

Format



Parameters

A Sets the transfer type as ASCII. Specifying the ASCII transfer type has the same effect as using the ASCII subcommand. The ASCII transfer type is intended for the transfer of data to or from an ASCII host. ASCII is the default transfer type.

B Sets the transfer type as DBCS. Specifying the B transfer type with the appropriate options has the same effect as using the BIG5, EUCKANJI, HANGEUL, JIS78KJ, JIS83KJ, KSC5601, SJISKANJI, SCHINESE, or TCHINESE subcommands. If B is specified alone, the second type parameter defaults to 1 and current transfer type is changed to Shift JIS kanji.

When you transfer double-byte data, the currently active SBCS translation table is used for an SBCS characters in the data set. If necessary, use the

SITE/LOCSITE SBDataconn or SITE XLate FTP subcommand to select an alternate SBCS translation table that is appropriate for your data before transferring your double-byte data.

B 1
Changes current transfer type to Shift JIS kanji.

B 2
Changes current transfer type to Extended UNIX Code kanji.

B 3
Changes current transfer type to JIS 1983 kanji.

B 4
Changes current transfer type to JIS 1978 kanji.

B 5
Changes current transfer type to Hangeul.

B 6
Changes current transfer type to Korean Standard Code KSC-5601, 1989 version.

B 7
Changes current transfer type to Traditional Chinese (5550).

B 8
Changes current transfer type to Big-5.

B 9
Changes current transfer type to Simplified Chinese.

S Transferred data contains shift-out and shift-in delimeters.

If S is specified alone, the second parameter defaults to A. Shift-out and shift-in characters X'1E' and X'1F' are used.

The S parameter can be used to control the use of shift-out (SO) and shift-in (SI) characters during DBCS data transfer for Big5, SChinese, Shift-JIS kanji, EUC kanji, Hangeul, KSC-5601, and TChinese.

If Sosi is not specified at all, shift-out/shift-in characters are not used in the transferred data.

S A
Use shift-out and shift-in characters X'1E' and X'1F' to delimit DBCS strings in the transferred data.

S E
Use shift-out and shift-in characters X'0E'and X'0F' to delimit DBCS strings in the transferred data.

S S
Use ASCII spaces (X'20') as shift-out and shift-in characters to delimit DBCS strings in the transferred data.

A Use ASCII shift-in escape sequence `ESC ( B`. This is the default. (Used for DBCS data types JIS 1983 kanji and JIS 1978 kanji only.)

R Use JISROMAN shift-in escape sequence `ESC ( J`. (Used for DBCS data types JIS 1983 kanji and JIS 1978 kanji only.)

N Indicates the transfer is to be pure DBCS data (data with no SBCS characters) and that the data is to be transferred to or from EBCDIC DBCS data that contains no shift-out/shift-in delimiters.

When data is transferred from the EBCDIC host, the entire data set is assumed to be EBCDIC DBCS with no SO/SI characters in the data. The data is then converted to the required ASCII type and if any SO/SI option has been specified for the transferred data then the corresponding SO/SI characters are used to delimit the ASCII DBCS strings.

When transferring data to the EBCDIC host, no SO/SI characters are inserted and if any SO/SI option is specified for the transferred data, the corresponding SO/SI characters are removed from the ASCII data and not replaced at the host. The length of data may change during transfer to and from the EBCDIC host when pure DBCS is specified with any SO/SI option. When pure DBCS is specified by itself, the length of data does not change. If N is not specified, the shift-out/shift-in characters X'0E' and X'0F' are used at the host.

E Sets the transfer type as EBCDIC. Specifying the EBCDIC transfer type has the same effect as using the EBCDIC subcommand. The EBCDIC transfer type is intended for efficient transfer between hosts that use EBCDIC for their internal character representation.

F Sets the transfer type as EBCDIC IBM kanji. Specifying the IBM kanji transfer type has the same effect as using the IBMKANJI subcommand.

F 1
Change current transfer type to IBM (EBCDIC) kanji.

I Sets the transfer type as image (binary). Specifying the image transfer type has the same effect as using the BINARY subcommand. With the image transfer type, data is sent as contiguous bits, packed into 8-bit bytes. The image transfer type is used for the efficient storage and retrieval of data sets or files, and for the transfer of binary data.

U 2
Sets the transfer type to Unicode UCS-2. TYPE U 2 has optional parameters:

B Specifies big-endian byte order for Unicode encoding. This is the default.

L Specifies little-endian byte order for Unicode encoding.

Examples

- Transfer text data from an EBCDIC host to an ASCII host:

```
  User:    ascii
System:      >>>TYPE A
             200 Representation type is ASCII.
             Command:
```

- Transfer binary data from an EBCDIC host to an ASCII host:

```
  User:    type i
System:      >>>TYPE I
             200 Representation type is IMAGE.
             Command:
```

- Transfer text data from an ASCII host to an EBCDIC host:

```
  User:     type a
System:      >>>TYPE A
             200 Representation type is ASCII.
             Command:
```

- Transfer binary data from an ASCII host to an EBCDIC host:

```
  User:     type i
System:      >>>TYPE I
             200 Representation type is ASCII.
             Command:
```

- Transfer text data from an EBCDIC host to an EBCDIC host:

```
  User:     type e
System:      >>>TYPE E
             200 Representation type is ASCII.
             Command:
```

- Transfer binary data from an EBCDIC host to an EBCDIC host:

```
  User:     type i
System:      >>>TYPE I
             200 Representation type is ASCII.
             Command:
```

- Set the transfer type to JIS 1983 kanji using the JISROMAN shift-in escape sequence `ESC ( J`:

  ```
  TYPE B 3 R
  ```

- Set the transfer type to Shift-JIS kanji using the EBCDIC SO/SI characters X'0E'/X'0F' in the transferred date:

  ```
  TYPE B 1 S E
  ```

Usage

- If no Sosi option is specified by the TYPE command for BIG5, SCHINESE, EUCKANJI, HANGEUL, KSC5601, SJISKANJI, or TCHINESE, standard DBCS control is used for the data transfer. This means that no SO/SI characters are placed in the ASCII data when transferring from the (EBCDIC) host to ASCII and the value of each ASCII character is used to determine if it is a single-byte character or part of a double-byte character when transferring to the host. For JIS 1983 kanji and JIS 1978 kanji, three-character escape sequences are always used to delimit DBCS strings in mixed SBCS/DBCS ASCII data. These escape sequences cannot be altered by using the S, S A, S E, or S S parameters.
- If no Sosi option is specified, the length of data may change as it is transferred to or from the EBCDIC host since EBCDIC DBCS types on the host contain SO/SI characters in mixed SBCS/DBCS data to determine which characters are part of a DBCS string. Any of the above SO/SI options (S, S A, S E or S S) may be used for mixed SBCS/DBCS data so that the length of data does not change when transferred to or from the EBCDIC host. Use of three-character escape sequences for JIS 1983 kanji and JIS 1978 kanji means that the length of data for these types alwayss change when transferring mixed SBCS/DBCS data to or from the EBCDIC host.
- Use ASCII spaces as SO/SI characters in the transferred data only for transfer from the EBCDIC host. Data can be transferred to the host when using this option but care must be taken as each ASCII space is interpreted as a shift-out or shift-in character and is replaced with the corresponding SO/SI character on the host.

Context

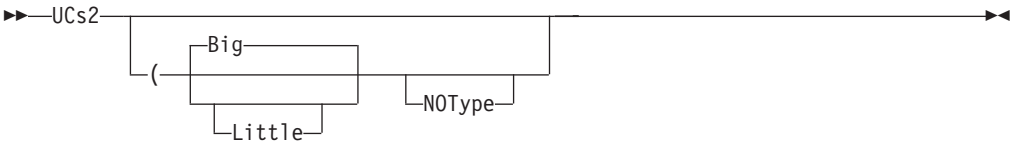
For more information about transfer methods, see Table 12 on page 52.

UCS2 Subcommand—Change Data Transfer Type to Unicode UCS-2

Purpose

Use the UCS2 subcommand to change the data transfer type to Unicode UCS-2.

Format



Parameters

BIG
Specifies big-endian byte order for the Unicode encoding. This is the default.

LITTLE
Specifies little-endian byte order for the Unicode encoding.

NOTYPE
Suppresses the sending of the corresponding TYPE command to the server. Use this parameter when you want translation to be done by the FTP client only.

Usage

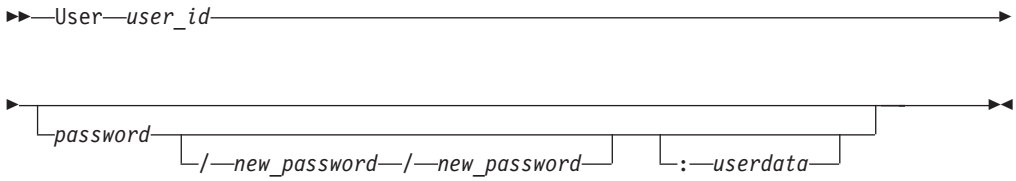
The UCS2 client subcommand is equivalent to the TYPE U 2 subcommand.

USER Subcommand—Identify Yourself to a Host or Change Your TSO User ID Password

Purpose

Use the USER subcommand to identify yourself to the host after opening a connection or, if the foreign host is an MVS FTP server, you can change your TSO user ID password.

Format



Parameters

user_id
Specifies your logon name on the host.

password
Specifies your current password on the host. If you do not supply a password when invoking the USER subcommand, you are prompted to enter a password if the host requires a logon password.

new_password / new_password
Specifies your new password on the host. You must enter the password twice.

:userdata
An optional character string to be passed to the FTCHKPWD user exit routine. The user data must be separated from the password information by a colon (:) and can be any combination of up to 200 nonblank characters and numbers—except the colon. Care should be taken when using the backslash character (\) in combination with other characters which might be interpreted as an escape sequence by the C compiler.

Usage

- If you enter your password incorrectly, you are not prompted to enter the password again. You must reissue the USER subcommand to enter the correct password.
- If you do not specify *password/new_password/new_password* on the USER subcommand, you can specify it when you are prompted for the password after entering the USER subcommand. You can specify the USER subcommand to change your TSO *user_id* password at any time during the FTP session.
- To avoid having your password print when issuing your user ID and password as part of a CLIST or batch job, enter your user ID and password on separate lines.
- Do not place any spaces between the passwords and the slashes (*/*), and the user data.
- You can use the NETRC data set to automatically log on to a remote host.

Context

For information about using NETRC, see “NETRC Data Set” on page 40.

Chapter 4. Sending Electronic Mail Using SMTP Commands

This chapter describes how to use the SMTPNOTE command, provided with CS for OS/390, to prepare and send electronic mail. Recipients of the mail can be users on your local host, users on network job entry (NJE), or users on TCP hosts. The SMTPNOTE command uses the Simple Mail Transfer Protocol (SMTP) to send the mail.

Interfaces to the SMTP address space are:

- SMTP mail can be sent and received interactively over a TCP/IP network. Mail from TCP/IP network sites destined for local MVS users (or users on a Network Job Entry (NJE) network attached to the local MVS system) arrives over this interface. All commands and data received and transmitted through this interface use ASCII characters.
- Interface from the Job Entry Subsystem (JES) spool, including any connected NJE nodes. SMTP commands can be written into a SYSOUT data set, with an external writer name of the SMTP address space. SMTP processes each of the commands in the data set in sequence, exactly as if it had been transmitted over a TCP/IP connection. This is how mail is sent from local MVS users to recipients on the TCP network. Batch SMTP data sets must contain commands and data in EBCDIC characters.

  For a description of batch SMTP in TSO utilities, see “Using Batch SMTP Command in TSO Utilities” on page 273. For examples of batch SMTP, see “Batch SMTP Examples” on page 272.

  Notes:

  1. This interface can be used to send mail only. Using the VERB command with this interface causes spool problems. For a description of the VERB command, refer to “VERB Command—Enable or Disable Verbose Mode” on page 270.
  2. With this interface, you might not be able to use certain SMTP options that require the NETDATA format. For details, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

IBM CS for OS/390 TCP/IP provides an SMTP gateway function, which can be used to transfer electronic mail between an NJE network and a TCP/IP network, as shown in Figure 10 on page 244.

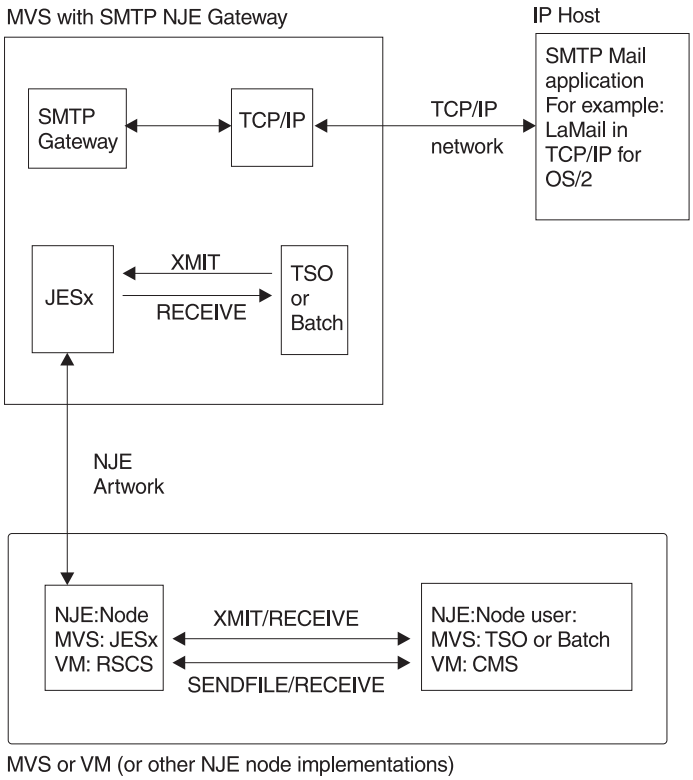


Figure 10. SMTP Gateway Overview

Using the SMTPNOTE Command from Your Terminal

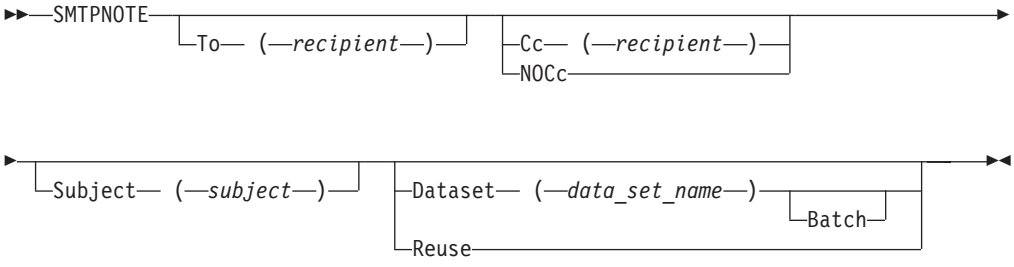
The SMTPNOTE command enables you to prepare the mail using the facilities of the Time Sharing Option (TSO) EDIT command, or to send mail prepared with a system editor of your choice.

SMTPNOTE Command—Send Electronic Mail to One or More Recipients on NJE or TCP Networks

Purpose

Use the SMTPNOTE command to send electronic mail to one or more recipients on NJE or TCP networks.

Format



Note: The minimum abbreviation for each parameter is shown in uppercase letters.

Parameters

To (*recipient***)**
Specifies a single recipient for the mail. If you do not specify the To parameter, you are prompted to enter a list of recipients. Enter the name of each recipient on a separate line and end the list with a blank line. You must specify at least one mail recipient.

Cc (*recipient***)**
Specifies a single copy recipient for the mail. If you do not specify the Cc parameter and you do not specify the NOCc or Batch parameters, you are prompted to enter a list of copy recipients.

If there are no recipients, press Enter. Otherwise, enter the name of each recipient on a separate line and end the list with a blank line.

NOCc
Specifies that no prompting for Cc take place.

recipient
Specifies the path address of the mail recipient. The format of *recipient* is equivalent to the path syntax, as described in RFC 821, without the **<** and **>** delimiters. See “Appendix G. Related Protocol Specifications (RFCs)” on page 595, for information about obtaining RFCs. This *recipient* parameter has one of the following formats:

user_id@*host_name*
User on a host in your domain, possibly on your local node.

user_id@*host_name* **.***domain*
User on a host in a specified domain.

*user_id***%***nje_host_name***@** *gateway_name***.***domain*
User on an NJE or RSCS node connected to a TCP network at *gateway_name*. See “Electronic Mail Gateway” on page 255 for more information.

@host1,@host2,...,@hostn **:***user_id @host_name*
User on a host that is not known by the local Domain Name Server, but that can be reached by following the path from *host1* to *hostn*.

Subject (*subject***)**
Specifies the subject of the note. If you do not specify the Subject parameter, you are prompted for the subject.

If there is no subject for the note, press Enter. If Subject is specified as a keyword, you cannot have embedded blanks. Otherwise, the *subject* can be any arbitrary string of characters, but is limited to a total length of 233 characters.

Batch
Specifies that no prompting for Cc take place.

Dataset (*data_set_name***)**
Indicates that the text of the mail is contained in a sequential data set. The data set can have any record format, be blocked or unblocked, and have records of up to 243 characters in length. The *data_set_name* is the name of the sequential data set containing the text of the note. It must be a valid data set name, and is fully qualified if it is contained within single quotation marks (').

Reuse
Causes SMTP to reuse the contents of a note that was previously canceled. If a note was not canceled, the Reuse parameter is ignored.

Usage

- When the To(), Cc(), Subject(), or Dataset() parameters are specified more than once, the last value specified is the one used by SMTPNOTE. No error messages are generated for duplicate parameters on the command line.
- SMTPNOTE no longer requires quotation marks around blanks, single quotation marks, semicolons, or commas, or triple quotation marks around data set names.

Preparing and Sending Mail

After you enter the SMTPNOTE command, you are prompted for the mail recipients (To:), the copy recipients (Cc:), and the subject of the note (Subject:), if they were not specified with the SMTPNOTE command. If you enter a list of recipients, enter the name of each recipient on a separate line and indicate the end of the list by entering a blank line.

After you answer the prompts, SMTPNOTE invokes the TSO EDIT command to enable you to prepare your note. Typically, the editor starts in INPUT mode. Enter the text of your note line by line. When your note is complete, enter a null line (that is, do not type anything when prompted), and press Enter. The editor switches to EDIT mode.

In EDIT mode, you can use all the functions of the editor. You can also return to INPUT mode, send the note, or cancel the note. For a complete description of the EDIT command, refer to the *IBM TSO Extensions Command Language Reference*.

If you invoke the SMTPNOTE command with the REUSE or DATASET parameter, you are immediately placed in EDIT mode. The contents of the previously canceled note, or the data set that you specify, are already part of the note. You can add to or change the data that is already present.

To send the note, enter **END SAVE** in EDIT mode, and then enter **SEND**. To cancel the note, enter **END SAVE** or **END NOSAVE** in EDIT mode, and then enter **CANCEL**. If you cancel a note, you can recover what you entered by invoking the SMTPNOTE command with the REUSE parameter. The recipients and subject of the note are not saved, and must be reentered.

Figure 11 is an example of preparing and sending mail.

```
READY
smtpnote
 TO:
irvine@mvs2.accounting
bekker@mvs2.accounting
mcgregr@mvs1.accounting

 CC:

 SUBJECT:
Travel Expenses

 ENTER "END SAVE" TO SAVE THE NOTE.
 FOR A COMPLETE LIST OF EDIT SUBCOMMANDS ENTER "HELP".
 INPUT
Could we please postpone the expense review because I
will be out of town the week of the 19th.

How about the 23rd?  Thank you, John.

 EDIT
end save
 ENTER "SEND" TO SEND THE NOTE.
 ENTER "CANCEL" TO TERMINATE WITHOUT SENDING THE NOTE.
send
 READY
```

Figure 11. Example of Preparing and Sending Mail

Receiving Mail

Use the TSO RECEIVE command to receive SMTP mail, as you would other mail and messages. Figure 12 shows an example of using the RECEIVE command.

```
 System:    READY

  User:  receive
System:    Dataset ** MESSAGE ** from SMTP on MVSXA2
           Received: from MVSXA2 by TREEFROG.ABC.OZ (IBM MVS SMTP V3R2)
             with BSMTP id 3088
           ;
           ***

           Wed, 28 Jul 93 16:41:31 EST
           Date:     28 Jul 93 16:37:56 LCL
           From:     MMC@TREEFROG.ABC.OZ
           To:       mvsuser%mvsxa3@TREEFROG.ABC.OZ
           Cc:       mmc%mvsxa3@TREEFROG.ABC.OZ
           Subject:  TSO RECEIVE Example

           This is an example of the TSO RECEIVE command.

           ********************************************************************
           INMR000I  No more files remain for the receive command to process.
```

Figure 12. TSO RECEIVE Command

If there is no message to be received, the following message is displayed:

```
INMR003I  You have no messages or data sets to receive.
***
```

For more information, refer to the *IBM TSO Extensions Interactive Data Transmission Facility: User's Guide*.

Undelivered Notes™

When SMTP cannot deliver a piece of mail, a nondelivery note explaining the reason for nondelivery is sent to the sender. Nondelivery can occur for several reasons, such as the destination host is unreachable, or the recipient does not have a user ID on the destination host. If a note cannot be delivered, the body of the original piece of mail is returned as part of the nondelivery notification. For an example of a nondelivery note, see Figure 13 on page 249. The nondelivery note is shown in bold.

```
Date: Mon, 9 Mar 92 08:23:54 EST
From: SMTP@MVS1.ACME.COM
To:   DANIEL@MVS1
Subject: Undeliverable Mail
MVS1.ACME.COM unable to deliver following mail to recipient(s):
      <MATT@SMTP-GATEWAY.IBM.COM>
MVS1.ACME.COM unable to connect for 3 days to host:
      SMTP-GATEWAY.IBM.COM
       ** Text of Mail follows **
Date: Mon, 9 Mar 92 08:22:36 EST
From: <DANIEL@MVS1.ACME.COM>
To:   <MATT@SMTP-GATEWAY.IBM.COM>
Subject:   ACME iron birdseed

Matt,

The shipment of ACME iron birdseed was shipped last Thursday.  Please
advise me if you have not received it, and I'll try to track it down.
Also, your ACME giant rock catapult is on back order; another customer
bought the last one yesterday.

Daniel
```

Figure 13. Example of a Nondelivery Note

If a recipient is unknown at the destination host, the destination host does not accept the mail for delivery, and a nondelivery note is forwarded to the sender. For an example of an unknown recipient note, see Figure 14. The unknown recipient note is shown in bold.

```
Date: Mon, 9 Mar 92 13:32:12 EST
From: SMTP@MVS1.ACME.COM
To:   DANIEL@MVS1
Subject: Undeliverable Mail
MVS1.ACME.COM unable to deliver following mail to recipient(s):
      <GEORGE@SMTP-GATEWAY.IBM.COM>
MVS1.ACME.COM received negative reply from host:
      SMTP-GATEWAY.IBM.COM
       ** Text of Mail follows **
Date: Mon, 9 Mar 92 08:22:36 EST
From: <DANIEL@MVS1.ACME.COM>
To:   <GEORGE@SMTP-GATEWAY.IBM.COM>
Subject:   Your retirement

George,

I recently learned that you will soon be retiring.  I just wanted to
wish you best of luck and thanks for all the great work you've done.
You were a great asset to your company, and I'm sure they will miss you.

Daniel
```

Figure 14. Example of an Unknown Recipient Note

Monitoring the Status of SMTP Using the SMSG Command

The SMSG command provides an interactive interface to the SMTP server to:

- Query the operating statistics of the SMTP server
- Query the SMTP mail delivery queues

- Perform privileged system administration tasks, such as shutting down the SMTP server and enabling or disabling various tracing and debugging options

Responses to commands are sent back to the originator of the command using MSG commands (or MSGNOH commands if SMTP is running with privilege class B).

SMSG SMTP Command for the General User

Purpose

As a general user, use the SMSG SMTP command to monitor the status of SMTP.

Format

Note: The minimum abbreviation for each parameter is shown in uppercase letters.

Parameters

HElp
Provides a list of valid SMTP SMSG commands.

QUeues
Provides a list of mail queued on the various SMTP mail delivery queues.

STats
Provides operating statistics about the SMTP server since the SMTP server was started.

Examples

The following example shows the output from the SMTP STATS command.

```
 System:    READY
  User:  user.smsg smtp stats
System:    +Msg from SMTP:  * Last Up Time: Wed, 28 Jul 93 16:33:32 EST
           +Msg from SMTP:  * Statistics  : 07/28
           +Msg from SMTP:  * From TCP    :     0
           +Msg from SMTP:  * From Spool  :     0
           +Msg from SMTP:  * BSMTP Logs  :     0
           +Msg from SMTP:  * Error Mail  :     0
           +Msg from SMTP:  * To Local    :     0
           +Msg from SMTP:  * To RSCS     :     0
           +Msg from SMTP:  * To TCP      :     0
           +Msg from SMTP:  * Passive Opns:     0
           +Msg from SMTP:  * Active Opens:     0
           READY
```

Usage

The statistics about SMTP include:

- The date that SMTP was last started.
- Statistics about mail handled by SMTP over the past two days, including:
  - The number of pieces of mail that arrived over TCP connections
  - The number of pieces of mail that arrived from spool (local or RSCS senders)
  - The number of pieces of mail generated in response to requests to VERBose batch SMTP connections
  - The number of pieces of mail generated to return error mail to the sender
  - The number of pieces of mail delivered to local recipients
  - The number of pieces of mail delivered to recipients on the RSCS network

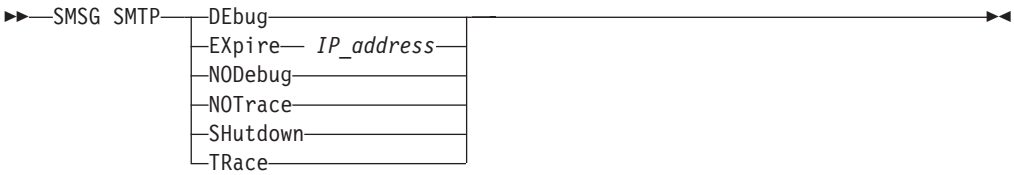
- The number of pieces of mail delivered to recipients on the TCP/IP network
- The number of TCP connections through which mail was received
- The number of TCP connections through which mail was delivered

SMSG SMTP Command for the Privileged User

Purpose

As a privileged user, use the SMSG SMTP command to monitor the status of SMTP.

Format



Note: The minimum abbreviation for each parameter is shown in uppercase letters.

Parameters

DEbug
Enables connection debugging and tracing information to the SMTP.DEBUG data set. This is the same as adding DEBUG to the SMTP.CONFIG data set.

EXpire *IP_address*
Causes the domain name resolution for mail queued for delivery to this IP address to expire.

NODebug
Disables connection debugging and tracing.

NOTrace
Disables resolver tracing.

SHutdown
Causes the SMTP server to shut down.

TRace
Enables resolver tracing. The output of the resolver trace is sent to the SMTP console. This is the same as adding TRACE RESOLVER to the *hlq*.TCPIP.DATA data set.

Examples

Note: If the screen remains inactive even after you issue the SMSG SMTP command, press Enter a few times until any one of the following messages is displayed.

- In this example, MVSUSER starts the debugging and tracing:

```
 System:    READY

  User:  smsg smtp debug
System:    +Msg from SMTP:  * Session Debugging Enabled
```

- Stop debugging and tracing:

```
 System:    READY

  User:  smsg smtp nodebug
System:    +Msg from SMTP:  * Session Debugging Disabled
           READY
```

- Begin resolver tracing:

```
 System:    READY

  User:  msg smtp trace
System:    +Msg from SMTP:  * Resolver Tracing Enabled
           READY
```

- Stop resolver tracing:

```
 System:    READY

  User:  smsg smtp notrace
System:    Msg from SMTP:  * Resolver Tracing Disabled
           READY
```

- Shut down the SMTP server:

```
System:    READY

  User:  smsg smtp shutdown
System:    +Msg from SMTP:  * ok, About to End SMTP
           READY
```

- Cause all mail queued for name resolution to expire:

```
 System:    READY

  User:  smsg smtp3 expire 9.67.112.25
System:    +Msg from SMTP3:  * No Mail Found for IP address: 9.67.112.25
           READY
```

Usage

Privileged user SMSG commands are accepted only from users specified in the SMSGAUTHLIST statement.

Context

Refer to *OS/390 SecureWay Communications Server: IP Configuration* for information about the TCPIP.DATA data set and for a description of the SMSGAUTHLIST configuration statement.

Electronic Mail Gateway

A system administrator can configure the SMTP server to run as a mail gateway between TCP network users and users located on an NJE or Remote Spooling Communications Subsystem (RSCS) network that is attached to the local host. Figure 15 is an example of a mail gateway.

In Figure 15, the following abbreviations are used:

A	The local MVS host running both the TCP/IP program and NJE.
B and C	Hosts attached to host A through an NJE network.
D and E	Hosts attached to host A through a TCP network.

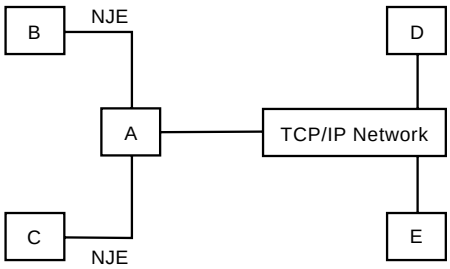


Figure 15. SMTP As a Mail Gateway

Users on hosts A, B, and C can send mail to users on TCP hosts D and E using the SMTPNOTE command. This method is described in “Using the SMTPNOTE Command from Your Terminal” on page 244.

Users on TCP hosts D and E can send mail to the users on host A using addresses in the following format:

user_id@A.domain

Where:

user_id	Is the user ID of the TSO user on host A
A.domain	Is the TCP host name of host A

The users on TCP hosts D and E can send mail to users on NJE hosts B and C using addresses in the following format:

user_id%NJEHost@A.domain

Where:

user_id	Is the user ID of the user on the host
NJEHost	Is the name of the NJE host (B or C)
A.domain	Is the TCP host name of host A

Path Address

Path addresses are rewritten according to the following rules:

- If the local part of a mailbox name includes a percent sign (%) and the domain of the mailbox is the host system, SMTP rewrites the address treating the portion of the local part to the right of the percent sign (%) as the real destination host. For example, the path address:

```
John%yourhost@ourhost.our.edu
```

is rewritten by SMTP running at ourhost.our.edu as:

```
John@yourhost
```

- Path addresses with source routes are accepted and rewritten to remove the domain name of the host system. For example, the path address:

```
@ourhost.our.edu,@next.host.edu:John@yourhost
```

is rewritten by SMTP running at ourhost.our.edu as:

```
@next.host.edu:John@yourhost
```

SMTP also optimizes a path address. For example,

```
@some.host.edu,@ourhost.our.edu,@next.host.edu:John@yourhost
```

is rewritten by SMTP running at ourhost.our.edu as:

```
@next.host.edu:John@yourhost
```

SMTP Commands

This section describes the SMTP commands as listed in Table 24.

Table 24. SMTP Commands

Subcommand	Description	Page
DATA	Define the following information as data	257
EXPN	Verify if a mailbox exists on the local host	259
HELO	Identify the domain name of the sending host to SMTP	260
HELP	Get help with SMTP commands	261
MAIL FROM	Specify the sender of the mail	262
NOOP	Return a `250 OK` return code when SMTP is responding	263
QUEU	Get information about mail queued at SMTP for delivery	264
QUIT	End an SMTP connection	266
RCPT TO	Specify the recipients of the mail	267
RSET	Reset the SMTP connection to the initial state	268
TICK	Insert an identifier into the batch SMTP response data set	269
VERB	Enable or disable verbose mode	270
VRFY	Verify if a mailbox exists on the local host	271

Data sets containing SMTP commands can be written to the JES spool as SYSOUT data sets. These SYSOUT data sets contain either punch or NETDATA records. Data sets originate from users on the same system as the SMTP address space or from users on any system connected to the host system through an NJE network.

DATA Command—Define the Following Information as Data

Purpose

Use the DATA command to define the following information as the data text of the mail body.

Format

Parameters

There are no parameters for this command.

Examples

After entering the DATA command, you receive the following message (response code 354) when you can transmit the body of your mail:

```
354 Enter mail body.  End new line with just a '.'
```

You end transmitting the body of your mail by entering a single ASCII period (.) on a line by itself.

The following message indicates success:

```
250 Mail Delivered
```

Usage

- Use the DATA command after a HELO command, a MAIL FROM command, and at least one RCPT TO command have been accepted.
- When receiving mail over a TCP connection, the ASCII period should be followed by the ASCII <carriage return> <line feed> character sequence. If any record in the body of the mail begins with a period, the sending SMTP program must convert the period into a pair of periods (··). When the receiving SMTP encounters a record that begins with two periods in the body of the mail, it discards the leading period. This convention permits the body of mail to contain records that would otherwise be interpreted as signaling the end of the body of mail. These rules must be followed over both TCP and batch SMTP connections.

  The SMTPNOTE command performs this period doubling on all mail spooled to SMTP. If the body of the mail in a batch SMTP command is not explicitly ended by a record with a single period, SMTP adds one.

  After a period has been received, the SMTP connection is reset to the initial state (the state before any sender or recipients have been specified). Additional MAIL FROM, RCPT TO, DATA, and other commands can now be sent.

  If no more mail is to be delivered, end the connection with the QUIT command. If a QUIT command is not found at the end of a batch SMTP command data set, it is implied.
- If SMTP runs out of local mail storage space, it sends a reply with a 451 code. If the length of the body of the mail exceeds MAXMAILBYTES (defined in SMTP configuration data sets to be 512KB), SMTP sends a reply with a 552 code. For more information about MAXMAILBYTES, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

- When mail arrives over a batch SMTP connection from a Remote Spooling Communications Subsystem (RSCS) network host, and 822 header rewriting is enabled (with the REWRITE822HEADER configuration option), then header fields are modified to ensure that all addresses are fully qualified domain names. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about header rewriting.

EXPN Command—Verify Whether a Mailbox Exists on the Local Host

Purpose

Use the EXPN command to verify whether a given mailbox exists on the local host.

Format

Parameters

mailbox
Specifies a user-defined identifier for a mailbox. This name can specify a single mailbox or a mailing list.

Examples

Verify whether mike is a mailbox on host abc.com:

```
 System:    READY

  User:  EXPN mike
System:    250 mike@abc.com
```

Verify whether users-hackers is a mailing list on host abc.com:

```
 System:    READY

  User:  EXPN users-hackers
System:    250-carol@abc.com
           250-greg@abc.com
           250-marsha@abc.com
           250 peter@abc.com
```

The hyphen (-) as the fourth character of a response indicates that the response is continued on the next line.

Usage

- The EXPN command operates exactly the same as the VRFY command.
- The EXPN command can verify the existence of one or more mailboxes on the system. The mailboxes are defined by configuration statements in the SMTP.SMTP.CONFIG data set.
- The MVS SMTP server verifies only TSO user IDs if RACF is installed on the local system. TSO user IDs that are verified as valid are accepted with a reply code of 250. If RACF is not installed, any character string up to eight characters is accepted with a reply code of 250.

HELO Command—Identify the Domain Name of the Sending Host to SMTP

Purpose

Use the HELO command to identify the domain name of the sending host to SMTP before a MAIL FROM command.

Format

Parameters

domain_name
Specifies the domain name of the sending host.

Usage

- The HELO command is sent once before a MAIL FROM command.
- If *domain_name* is not known, you can send messages, but the message `never heard of you` is returned. This message indicates that the HELO command is accepted, but that the host name could not be resolved either through the name server or through the host tables.
- HELO commands received over a TCP connection, which has a *domain_name* IP address that does not match the IP address of the *ForeignSocket* field of the connection information record, are accepted. However, the reply text indicates the mismatch. HELO commands received over a batch SMTP connection with a *domain_name* that does not match the host name of the origination point of the spool data set are accepted, but the reply text indicates the mismatch.

HELP Command—Get Help with SMTP Commands

Purpose

The HELP command returns a multiline reply with some general information about the SMTP commands.

Format

Parameters

command_name
Indicates any of the SMTP commands.

MAIL FROM Command—Specify the Sender of the Mail

Purpose

Use the MAIL FROM command to specify the sender of the mail.

Format

Parameters

sender_path_address
Specifies the full path address of the sender of the mail.

Note: The brackets (< >) surrounding the *sender_path_address* in the syntax diagram are required.

Usage

- The MAIL FROM command is used once after a HELO command.
- If the host system has never heard of the sender host, a positive reply is sent, but a message is logged in the SMTP console.

NOOP Command—Return a 250 OK Return Code When SMTP Is Responding

Purpose

Use the NOOP command to return a `250 OK` return code when SMTP is responding.

Format

Parameters

There are no parameters for this command.

QUEU Command—Get Information about Mail Queued at SMTP for Delivery

Purpose

Use the QUEU command to get information about mail queued at SMTP for delivery.

Format

Parameters

DATE
Causes information about the age of the mail to be returned. The default is to not return age information.

Usage

QUEU returns a multiline reply with information about mail queued at SMTP for delivery. The following information is returned about mail:

Spool Queue Contains mail that is destined for recipients on the local MVS system, or for recipients on an NJE system attached to the local MVS system. This queue is generally empty, because SMTP can deliver this mail quickly by spooling it to the local recipient or to the NJE address space for delivery to an NJE network recipient.

Active Indicates that if SMTP is currently transmitting to a TCP network destination, all the mail queued for that destination is shown to be active.

Queued All mail that arrives over a batch SMTP connection, and mail from TCP connections that is to be forwarded to another TCP network destination through source routing, is placed on the queued list. As soon as SMTP receives resources from the TCPIP address space, mail that is queued is *promoted* to active.

Retry Queue Mail is placed here after SMTP tries to transmit mail to each of the TCP network hosts, but was unable to either open a connection or complete delivery over the connection. After the number of minutes specified by RETRYINT, mail is promoted from the retry queue to the QUEUED state. For more information about the RETRYINT variable, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

Undeliverable Queue
Mail is placed here if SMTP cannot deliver mail to a local MVS recipient, or to a recipient on the NJE network attached to the local MVS system, because spool space on the local MVS system is full. After spool space has been increased and SMTP has been restarted, delivery is attempted again.

Resolution Queues
SMTP uses the following queues for processing queries to the

name server. If the SMTP server is configured to use the site tables rather than the name server, these queues are not used.

If the queue is empty, the word Empty appears to the right of the queue. If the queue contains queries, the queries appear on separate lines below the queue. However, due to the speed of the SMTP server, the output may show that the queue is active without containing any entries. In this case, the word Empty does not appear.

Resolver Process Queue
Is generally empty; it contains queries waiting to be sent to the SMTP resolver. After the query has been processed, it is put in the resolver send queue.

Resolver Send Queue
Contains queries waiting to be processed by the SMTP resolver. SMTP staggers the number of queries sent by the resolver to prevent the overloading of the network and the name server.

Resolver Wait Queue
Contains queries for which the SMTP resolver is waiting for responses. Queries remain in this queue for the period of time it takes to receive a reply from the name server. If a reply is not received, the queries are removed from this queue after the resolver timeout has occurred, and are placed in the resolver retry queue. If the query is successful, the query is placed in the resolver completed queue.

Note: The SMTP resolver timeout is specified by the RESOLVERTIMEOUT statement in the *hlq*.TCPIP.DATA data set.

Resolver Retry Queue
Contains queries that have previously failed, either because the name server did not reply, or the name server returned a temporary error that forced the SMTP resolver to retry the query. A temporary error occurs if, for example, the name server truncates a packet, or if the name server detects a processing error.

The RESOLVERRETRYINT statement specifies the number of minutes SMTP waits before retrying the query. The RETRYAGE statement specifies the number of days SMTP should continue to resolve the query before returning the mail to the sender.

Resolver Completed Queue
Contains queries that have been resolved and are waiting to be recorded into the mail. After the internet addresses are recorded, SMTP attempts to deliver the mail.

Resolver Error Pending Queue
Contains queries that the name server has returned without answers. The corresponding mail message is returned to the sender with an unknown recipient error.

QUIT Command—End an SMTP Connection

Purpose

Use the QUIT command to end an SMTP connection.

Format

Parameters

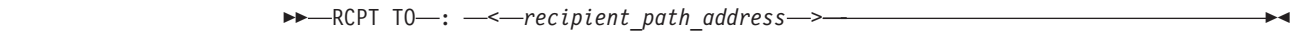
There are no parameters for this command.

RCPT TO Command—Specify the Recipients of the Mail

Purpose

Use the RCPT TO command to specify mail recipients.

Format



Parameters

recipient_path_address
Specifies the full path address of the mail recipient.

Note: The brackets (< >) surrounding *recipient_path_address* in the syntax diagram are required.

Usage

- You can use the RCPT TO command any number of times.
- You can use the RCPT TO command only after a MAIL FROM command has been issued.
- If the host system has never heard of the recipient host, the RCPT TO command receives a negative reply.
- If a name server is used for domain name resolution, MX records are used to resolve the recipient IP address before trying A records.

Context

For more information about MX and A records, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

RSET Command—Reset the SMTP Connection to the Initial State

Purpose

Use the RSET command to reset the SMTP connection to the initial state in which the sender and recipient buffers are erased and the connection is ready to begin a new mail transaction.

Format

Parameters

There are no parameters for this command.

TICK Command—Insert an Identifier into the Batch SMTP Response Data Set

Purpose

Use the TICK command, in combination with the VERB ON command, to insert an identifier into the batch SMTP response data set. This command is useful with mail systems that keep track of batch SMTP response data sets.

Format

Parameters

identifier
Specifies a string used to identify the origin of batch SMTP responses.

Usage

The TICK command has no effect when it is issued over a TCP connection to SMTP.

VERB Command—Enable or Disable Verbose Mode

Purpose

Use the VERB command to enable or disable verbose mode.

Format

Parameters

ON
Enables verbose mode. When enabled, the batch SMTP commands and associated replies are recorded in the batch SMTP response data set. Also, the batch SMTP response data set is sent back to the origination point of the batch SMTP command data set.

To avoid receiving spool errors, ensure that the origination point is a valid JES user and node on the SMTP sending system. The origination point information is taken from the TSO transmit (XMIT) command headers.

OFF
Disables verbose mode. When disabled, only the replies (not the commands) are recorded in the batch SMTP response data set. This is the default.

Usage

The VERB command has no effect when issued over a TCP connection to SMTP.

VRFY Command—Verify Whether a Mailbox Exists on the Local Host

Purpose

Use the VRFY command to verify whether a given mailbox exists on the local host.

Format

Parameters

mailbox
Specifies a user-defined identifier for a mailbox. This name can specify a single mailbox or a mailing list.

Examples

Verify whether mike is a mailbox on host abc.com:

```
System:    READY

 User:  VRFY mike
        250 mike@abc.com
```

Verify whether users-hackers is a mailing list on host abc.com:

```
 System:    READY

  User:  VRFY users-hackers
System:    250-carol@abc.com
           250-greg@abc.com
           250-marsha@abc.com
           250 peter@abc.com
```

The hyphen (-) as the fourth character of a response indicates that the response is continued on the next line.

Usage

- The VRFY command operates exactly the same as the EXPN command.
- The VRFY command can verify the existence of one or more mailboxes on the system. The mailboxes are defined by configuration statements in the SMTP.SMTP.CONFIG data set.
- The MVS SMTP server verifies only TSO user IDs if RACF is installed on the local system. TSO user IDs that are verified as valid are accepted with a reply code of 250. If RACF is not installed, any character string up to eight characters is accepted with a reply code of 250.

SMTP Responses

SMTP commands arrive over a TCP connection (to your terminal) or over a batch SMTP connection. With either connection, a response to each command is generated. All responses are prefixed with a 3-digit number. You can determine the response by inspecting the first digit of the response code:

First Digit	Description
2	Indicates a positive response. Command accepted.
3	Indicates a positive response. Send the data associated with the command.
4	Indicates a temporary negative response. Try again later.
5	Indicates a permanent negative response. The command has been rejected.

If SMTP commands arrive over a TCP connection, all responses (positive or negative) are returned over that TCP connection. If SMTP commands arrive over a batch SMTP connection, all responses are written to the batch SMTP response data set.

If verbose mode is enabled for a batch SMTP connection, SMTP returns the batch SMTP response data set to the origination point of the spool data set. The origination point is determined from the NETDATA header if the data set arrives in NETDATA format, or the MAIL FROM command if the data set arrives in punch format. If the batch SMTP connection is not in verbose mode, the batch SMTP response data set is not returned to the point of origin, and is discarded.

If an error occurs during the processing of commands over a batch SMTP connection, such as reception of a negative response (with a first digit of 4 or 5), an error report is mailed back to the sender. The sender is determined from the last MAIL FROM command received that was valid. If the sender cannot be determined from a MAIL FROM command, the sender is assumed to be the origination point of the batch SMTP command data set. The error report mailed to the sender includes the batch SMTP response data set and the text of the undeliverable mail.

All SMTP commands and data that arrive over TCP or batch SMTP connections are subject to the restrictions imposed by constants defined in the SMTPGLOB macro. Changes to the SMTPGLOB macro require recompiling the SMTP server. The default values of these constants are:

- Command lines must not exceed MaxCommandLine (512 characters).
- Data lines longer than MaxDataLine (1024 characters) are wrapped.
- Path addresses must not exceed MaxPathLength (256 characters).
- Domain names must not exceed MaxDomainName (256 characters).
- User names, the local part of a mailbox specification, must not exceed MaxUserName (256 characters).

Batch SMTP Examples

The following sections contain examples that demonstrate batch SMTP capabilities.

Sending Mail to a TCP Network Recipient

The following is an example of sending mail from an NJE network host to two TCP network recipients. The NJE network is BITNET, and the NJE host is named

YOURMVS. This batch SMTP data set is spooled to SMTP at OURMVS, which is running with the runtime arguments GATEWAY and NJEDOMAIN BITNET. OURMVS is a BITNET host and is also connected through a TCP network to the hosts *rsch.our.edu* and *ai.our.edu*.

```
HELO YOURMVS
MAIL FROM:<CAROL@YOURMVS>
RCPT TO:<msgs@rsch.our.edu>
RCPT TO:<alice@ai.our.edu>
DATA
Date: Thur, 26 Mar 92 21:48:57 EST
From: Carol <CAROL@YOURMVS>
To:   <msgs@rsch.your.edu>
Cc:   <alice@ai.your.edu>
Subject: update

Mike: Cindy stubbed her toe.  Bobby went to
      baseball camp.  Marsha made the cheerleading team.
      Jan got glasses.  Peter has an identity crisis.
      Greg made dates with 3 girls and couldn't
      remember their names.
.
QUIT
```

SMTP rewrites the `From:` line to reflect that the mail has been transferred from an NJE network (in this case, BITNET) to a TCP network. The TCP network recipients receive:

```
From: carol <CAROL%YOURMVS.BITNET@ourhost.our.edu>
```

Querying the SMTP Delivery Queues

The SMTP delivery queues can be queried by sending a data set with the SMTP commands VERB ON and QUEU to the SMTP address space. A batch SMTP response data set is returned with the result of the VERB ON command.

Using Batch SMTP Command in TSO Utilities

The batch SMTP commands are also used with the TSO transmit (XMIT) command and the IEBGENER utility in TSO/ISPF.

Use the TSO transmit (XMIT) command to spool a batch SMTP command sequential data set to SMTP. After you create the batch SMTP command sequential data set, use the following command to spool the data set to SMTP:

```
XMIT jesnode-name.smtp DA(batsmtp.text)
```

Where:

jesnode
: Is the JES nodename/hostname

smtp Is the SMTP address space name. The default is SMTP

batsmtp.text
: Is the batch SMTP commands data set

To code the batch SMTP commands as inline input for SYSUT1 and SYSUT2, create the following JCL using the IEBGENER utility on the TSO/ISPF application.

```
//BATSMTP  JOB (userid,nn),MSGCLASS=B,PRTY=12,MSGLEVEL=(2,1)
//IEBGENER EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSUT1 DD *
```

```
HELO YOURMVS
MAIL FROM:<CAROL@YOURMVS>
RCPT TO:<msgs@rsch.our.edu>
RCPT TO:<alice@ai.our.edu>
DATA
Date: Thur, 26 Mar 92 21:48:57 EST
.
.
.
QUIT
/*
//SYSUT2 DD SYSOUT=(B,smtp)
//*                  |   v
//*                  v  SMTP address space name for external writer
//*                 SYSOUT class
//SYSOUT DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//
```

SMTP with DBCS Support

Mail in EBCDIC DBCS Big-5, Kanji, Hangeul, Simplified Chinese, or Traditional Chinese can be sent to a remote host using SMTPNOTE, if the SMTP server is configured with the corresponding DBCS support. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about configuring DBCS support for the SMTP server.

Conversion of DBCS Mail

The transmission of DBCS mail by SMTP actually uses two different translation tables; one SBCS and one DBCS. SBCS characters in the mail headers and in the mail body are converted using *hlq*.STANDARD.TCPKJBIN, TCPHGBIN, TCPSCBIN, or TCPCHBIN.

Both SBCS and DBCS translation tables are required for the transmission of DBCS mail, as the mail can contain mixed-mode strings. A mixed-mode string contains both SBCS and DBCS data, delimited by shift-out and shift-in characters.

DBCS conversion is performed only on outgoing and incoming mail to and from other hosts. Mail spooled to SMTP by SMTPNOTE for the local host is delivered directly, without any DBCS code conversion.

Chapter 5. Sending Electronic Mail Using OS/390 UNIX sendmail and POP3 (Post Office Protocol)

This chapter briefly describes how to use OS/390 UNIX sendmail, provided with CS for OS/390, to prepare and send electronic mail using the facilities of the OS/390 shell. For a comprehensive discussion of sendmail, refer to the industry-accepted publication *sendmail* by O'Reilly & Associates, Inc. (ISBN 1-56592-222-0). That publication is known throughout the industry as the *bat book*, because of the fruit bat depicted on the cover. This chapter consistently refers to the *bat book* for further information.

You can also find more information about OS/390 UNIX sendmail on the following web sites:

- *http://www.sendmail.org*
- *http://www.s390.ibm.com/sample/sendmail*

Additional information about sendmail can also be found in the *sendmail Installation and Operation Guide* (sndmiaog.ps), which is received during the port of sendmail 8.8.7 from the sendmail web site. For more information about POP3 (Post Office Protocol), refer to RFC 1939.

Overview

OS/390 UNIX sendmail provides enhanced SMTP support, integrating with the existing SMTP mail server system to enable you to send mail across the internet. OS/390 UNIX sendmail replaces SMTPPROC as the primary SMTP server. OS/390 UNIX sendmail utilizes standard sendmail configuration and operation files. Consequently, you can simply use the existing mail user agent (MUA) interface to use OS/390 UNIX sendmail.

A POP3 server is used to deliver received mail to the MUA. OS/390 UNIX Popper is a POP3 server.

The simple mail architecture in which OS/390 UNIX sendmail and OS/390 UNIX Popper fit includes the following basic elements:

- A Mail User Agent (MUA), client software that a user invokes directly in order to send and receive mail. Mailx, Eudora, and Netscape Navigator are examples of MUAs.
- A Mail Transfer Agent (MTA), software that routes messages from a sender's system to the receiver's system. OS/390 UNIX sendmail is an MTA.
- A Mail Delivery Agent (MDA), server software that delivers received mail to a user's MUA. Popper is an MDA using the POP3 protocol.

Running sendmail by Hand

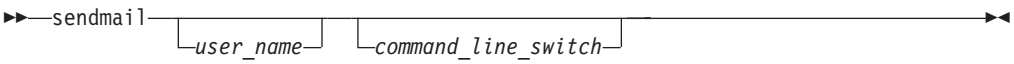
Most users do not run sendmail directly. Instead, they use one of the many mail user agents (MUAs) to compose a mail message. Those programs invisibly pass the mail message to sendmail, creating the appearance of instantaneous transmission. The sendmail program then takes care of delivery in its own fashion.

sendmail Command—Send File Contents

Purpose

Use the sendmail command to send the contents of a file.

Format



Parameters

See Table 25 on page 279 for details about the sendmail command-line switches.

Examples

To test run sendmail by hand, complete the following steps:

1. Create a file named *sendstuff* with the following contents:

   ```
   This is a one line message.
   ```

2. Mail this file to yourself with the following command line, where *you* is your login name:

   ```
   /usr/lib/sendmail you <sendstuff
   ```

You run sendmail directly by specifying its full path name. When you run sendmail, any command-line arguments that do not begin with a hyphen (-) are considered to be the names of the people to whom you are sending the mail message.

Note: The full path name may differ on your system. If so, simply specify the full sendmail path name as it is configured on your system.

The <sendstuff sequence causes the contents of the file that you have created (*sendstuff*) to be redirected into the sendmail program. The sendmail program treats everything it reads from its standard input (up to the end of the file) as the mail message to transmit.

Results

There are several ways to view the message that you just sent, depending on how your system is configured. You can type MAIL to view your mail, or use the *mh* package and type INC to receive and SHOW to view your mail. After viewing your mail, save the mail message to a file. The file will look something like this:

```
From you@Here.US.EDU  Fri Dec 13 08:11:44 1996
Received: (from you@localhost) by Here.US.EDU (8.8.4/8.8.4)
       id AA04599 for you; Fri, 13 Dec 96 08:11:44 -0700
Date: Fri, 13 Dec 96 08:11:43
From: you@Here.US.EDU (Your Full Name)
Message-Id: <9631121611.AA02124@Here.US.EDU>
To: you

This is a one line message.
```

This file begins with eight lines of text that were not in your original message. Those lines were added by sendmail and your local delivery program and are called the *header*.

The last line of the file is the original line from your *sendstuff* file. It is separated from the header by one blank line. The body of a mail message comes after the header and consists of everything that follows the first blank line.

Ordinarily, when you send mail with your MUA, the MUA adds a header and feeds both the header and the body to sendmail. This time, however, you ran sendmail directly and supplied only a body; the header was added by sendmail.

sendmail as a Daemon

Just as sendmail can transport mail message over a TCP/IP-based network, it can also receive mail that is sent to it over the network. To do this, it must be run in daemon mode. A daemon is a program that runs in the background independent of terminal control.

As a daemon, sendmail is run once, usually when your machine is booted. Whenever an email message is sent to your machine, the sending machine talks to the sendmail daemon that is listening on your machine.

The **-bd** command-line switch tells sendmail to run in daemon mode. The **-q1h** command-line switch tells sendmail to wake up once per hour and process the queue. Command-line switches are described in "OS/390 UNIX sendmail Commands" on page 279.

Invoking Sendmail

Via OS/390 UNIX

From an OS/390 user with root privledges, issue the following command:

```
sendmail -bd -q1h -v
```

where the `-bd` command-line switch tells sendmail to run in daemon mode. The `-q1h` command-line switch tells sendmail to wake up once per hour and process the queue. The `-v` command-line switch tells sendmail to run in verbose mode. This should only be used for debugging and results in trace records being logged to the syslogd daemon for logging to the syslogd daemon log file.

Note: Before sendmail can be invoked in this manner, tcpip must be active.

Via a Proc

Sendmail can be invoked through the BPXBATCH facility. Refer to the *OS/390 UNIX System Services User's Guide* for more information on the BPXBATCH facility.

Following is a sample JCL procedure for sendmail. Create a member on your system or recognized PROCLIB named sendmail. Use this sample as a base and customize it for your installation needs.

```
//SENDMAIL PROC
//*
//* Function: Start SENDMAIL Daemon
//*
//SENDMAIL EXEC PGM=BPXBATCH,REGION=4096K,TIME=NOLIMIT,
//         PARM='PGM /usr/sbin/sendmail -bd -q1h '
//*
//*        To save the standard output to an HFS file,
//*        define STDOUT as shown below:
//*
//STDOUT   DD PATH='/tmp/sendmail.stdout',
//         PATHOPTS=(OWRONLY,OCREAT,OAPPEND),
//         PATHMODE=(SIRUSR,SIWUSR)
//*
//*        To save standard error messages to an HFS file,
```

```
//*        define STDERR as shown below:
//*
//STDERR   DD PATH='/tmp/sendmail.stderr',
//         PATHOPTS=(OWRONLY,OCREAT,OAPPEND),
//         PATHMODE=(SIRUSR,SIWUSR)
//*
//CEEDUMP  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=132,BLKSIZE=132)
//         PEND
```

Note: Before sendmail can be invoked by the procedure, tcpip must be active.

Any user starting this procedure must have a uid(0).

If you want sendmail to be started automatically when the TCPIP address space is started, then include the name of the member containing the sendmail daemon catalogued procedure as one of the TCPIP AUTOLOG tasks.

Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information on the TCPIP AUTOLOG statement.

OS/390 UNIX sendmail Commands

Command-line switches are command-line arguments that begin with a hyphen (-) and precede the list of recipients (if any). The forms for the command-line switches, where *-Y* is a single letter, are:

-Y Boolean switch

-Yarg Switch with argument

All switches are single letters. A complete list is shown in Table 25.

Table 25. OS/390 Supported Command-Line sendmail Switches

Switch	Bat Book Reference	Version of sendmail	Description
-B	36.7.1	V8.1 and above	Specify message body type
-b	36.7.2	All versions	Set operating mode
-bD	36.7.4	V8.8 and above	Run as a daemon, but do not fork
-bd	36.7.5	All versions	Run as a daemon
-bH	36.7.6	V8.8 and above	Purge persistent host status
-bh	36.7.7	V8.8 and above	Print persistent host status
-bi	24.5.1	All versions	Initialize alias database
-bm	36.7.9	All versions	Be a mail sender
-bp	23.4	All versions	Print the queue
-bs	36.7.11	All versions	Run SMTP on standard input
-bt	38.1	All versions	Rule testing mode
-bv	36.7.13	All versions	Verify: do not collect or deliver
-C	36.7.15	All versions	Location of configuration file
-d	37.1	All versions	Enter debugging mode
-F	36.7.20	All versions	Set the sender's full name
-f	36.7.21	All versions	Set sender's address
-M	31.2	V8.7 and above	Define a macro on the command line

Table 25. OS/390 Supported Command-Line sendmail Switches (continued)

Switch	Bat Book Reference	Version of sendmail	Description
-m	34.8.39	All versions	Set the MeToo (m) option to true
-N	36.7.28	V8.8 and above	Specify DSN NOTIFY information
-n	24.6	All versions	Do not do aliasing
-O	34.1	V8.7 and above	Set a multi-character option
-o	34.1	All versions	Set a single-character option
-p	36.7.32	V8.1 and above	Set protocol and host
-q	23.6.1	All versions	Print the queue
-R	36.7.34	V8.8 and above	DSN what to return on a bounce
-t	36.7.38	All versions	Get recipients from message header
-U	36.7.39	V8.8 and above	This is the initial MUA to MTA submission
-V	36.7.40	V8.8 and above	Specify the ENVID string
-v	36.7.41	All versions	Run in verbose mode
-X	26.4	V8.2 and above	Log transactions
-x	36.7.43	V8.2 and above	Ignored

Alternative sendmail Command Names

The following commands or symbolic links produce the same results as the corresponding switch (described in Table 25 on page 279).

Table 26. OS/390 Supported Command-Line sendmail Aliases

Name	Bat Book Reference	Switch	Description
hoststat	36.1.1	-bh	Print persistent host status (V8.8 and above)
mailq	36.1.2	-bp	Print the queue contents
newaliases	36.1.3	-bi	Rebuild the *aliases* file
purgestat	36.1.4	-bH	Purge persistent host status (V8.8 and above)
smtpd	36.1.5	-bd	Run as a daemon

hoststat

The hoststat command, like the -bh switch, lists the status of the last mail transaction with all remote hosts. It functions only when the HostStatusDirectory option is set.

For each host that has its status saved, OS/390 UNIX sendmail prints the following information:

Hostname
Name of the host that OS/390 UNIX sendmail was connected to. It may not be the hostname specified for the recipient; it could be an MX record instead. If a message has multiple recipients, a separate status line is produced for each unique host that is tried. If this name is prefixed with an asterisk, the status file is locked and currently being updated.

How long ago

Shows how long ago this status record was updated. It is printed in the form: `DD+HH:MM:SS`. `DD` is the number of days. If the status was updated less than a day ago, the `DD+` is omitted. `HH` is hours, `MM` is minutes, and `SS` is seconds.

Results

Shows the results of the last connections attempt, failure, or success. If no reason was stored, the result prints as: `No status available`. If a result was stored, it prints in three parts: *smtp msg*

The *smtp* is the SMTP reply code. The *msg* is the text of the message generated by the other end or other program. An example of a result is:

```
 ------------- Hostname ------- How long ago ---------Results---------
 there.ufoa.edu                    00:00:51 250 PAA27153 Message acce
*books.ora.com                     07:43:39 250 GAA01255 Message acce
 prog.ammers.com                   06:55:08 No status available
 fbi.dc.gov                        03:28:53 Connection refused
```

In this example, the previous connections to *there.ufoa.edu* and *books.ora.com* were successful. The status for *books.ora.com* is currently being updated. The asterisk signifies that the file is locked. The host *prog.ammers.com* shows no status because connection to it could not be made. The last line in the example shows that the connection to *fbi.dc.gov* was refused by that host.

Note: The results are limited to 27 characters, unless the **-v** command-line flag is also used. In that case, results are limited to 79 characters, thus providing more complete information.

mailq

The **mailq** command, like the -bp switch, causes OS/390 UNIX sendmail to print the contents of the mail queue and then exit. The location of the queue is set with the QueueDirectory (Q) option. That location can be overridden from the command line, but if it is, OS/390 UNIX sendmail may give up its *root* privilege (unless it was run by *root*).

newaliases

The **newaliases** command, like the -bi switch, causes OS/390 UNIX sendmail to rebuild the *aliases* database files, print summary information, and then immediately exit. In this mode the -v command-line switch is automatically implied, and OS/390 UNIX sendmail runs in verbose mode.

The location of the *aliases* file is given in the configuration file with the AliasFile (A) option. That location can be overridden from the command line, but if it is, OS/390 UNIX sendmail gives up its *root* privilege (unless it was run by *root*).

purgestat

The **purgestat** command, like the -bH switch, causes OS/390 UNIX sendmail to clear (purge) all the host-status information that was being saved under the

HostStatusDirectory option directory. Clearing is done by removing all the directories under the HostStatusDirectory directory. The HostStatusDirectory directory is not removed.

smtpd

The **smtpd** command, like the -bd switch, causes OS/390 UNIX sendmail to run in the background as a daemon, listening for incoming SMTP mail. This mode of operation is usually combined with the **-q** command-line switch, which causes OS/390 UNIX sendmail to periodically process the queue.

Debugging Switches

Table 27 is a complete list of every debugging switch. The table also indicates which switches are useful for debugging mail problems.

Note: Some of these switches create long and complex output.

Table 27. Debugging Switches by Category

Category	Bat Book Reference	Useful for Mail Problems	Description
-d0.1	37.5.1	X	Print version information
-d0.4	37.5.2	X	Our name and aliases
-d0.10	37.5.3		Operating System defines
-d0.15	37.5.4	X	Dump delivery agents
-d0.20	37.5.5	X	Print network address of each interface
-d0.22	37.5.6		Show uname() failure
-d0.40	37.5.7		Show scanning of interfaces
-d0.44	37.5.8		Print addresses of strings
-d0.90	37.5.9	(obsolete)	Print first 10 rule sets
-d1.1	37.5.10		Show sender information
-d1.5	37.5.11		Dump the sender address
-d2.1	37.5.12		End with finis()
-d2.9	37.5.13		Show file descriptors with *dumpfd()*
-d3.1	37.5.14		Print the load average
-d3.5	37.5.15		Print load average
-d3.15	37.5.16		Print three load averages
-d3.20	37.5.17		Show offset for load average
-d3.30	37.5.18		Show result of decision to queue
-d4.80	37.5.19	X	Trace enoughspace()
-d5.4	37.5.20		Tick for queued events
-d5.5	37.5.21		Events set and cleared
-d5.6	37.5.22		Show events triggered
-d6.1	37.5.23	X	Show failed mail
-d6.5	37.5.24		The current error state
-d6.20	37.5.25		Show sender of return to sender

Table 27. Debugging Switches by Category (continued)

Category	Bat Book Reference	Useful for Mail Problems	Description
-d7.1	37.5.26		The queue file name
-d7.2	37.5.27		Show assigned queue file name
-d7.9	37.5.28		Dump file descriptor for the qf file
-d7.20	37.5.29		Show queue names being tried
-d8.1	37.5.30	X	Failure of MX search (low level)
-d8.2	37.5.31	X	Call to getcanonname(3)
-d8.3	37.5.32	X	Trace dropped local hostnames
-d8.5	37.5.33	X	Hostname being tried in getcanonname(3)
-d8.7	37.5.34	X	Yes/no response to -d8.5
-d8.8	37.5.35	X	MX lookup gets wrong type
-d8.20	37.5.36		Inconsistency in returned information
-d9.1	37.5.37		Canonify hostname and RFC1413 queries
-d9.3	37.5.38		Show raw RFC1413 reply
-d9.10	37.5.39		Show RFC1413 query being sent
-d10.1	37.5.40		Show recipient delivery
-d10.2	37.5.41		Dump controlling user's address
-d10.5	37.5.42		Show do not send to MeToo address
-d10.100	37.5.43		Predelivery file descriptor dump
-d11.1	37.5.44	X	Trace delivery
-d11.2	37.5.45	X	Show the uid/gid running as during delivery
-d11.20	37.5.46		Show tried D= directories
-d12.1	37.5.47	X	Show mapping of relative host
-d13.1	37.5.48	X	Show delivery
-d13.5	37.5.49		Show addresses that we should not send to
-d13.6	n/a		Trace envelope stripping, dropping, and moving
-d13.10	37.5.50		Trace sendenvelope()
-d13.20	37.5.51		Show final mode
-d13.21	n/a		Show final send queue
-d13.25	n/a		Watch owner deliveries
-d13.29	37.5.52		Show autoqueueing
-d13.30	37.5.53		Show envelopes being split
-d14.2	37.5.54		Show header field commas
-d15.1	37.5.55		Show network get request activity
-d15.2	37.5.56		Incoming connections
-d15.101	37.5.57		Kernel TCP debugging
-d16.1	37.5.58		Outgoing connections
-d16.101	37.5.59		Kernel TCP debugging
-d17.1	37.5.60		List MX hosts
-d17.9	37.5.61		Show randomizing MX records

Table 27. Debugging Switches by Category (continued)

Category	Bat Book Reference	Useful for Mail Problems	Description
-d18.1	37.5.62		Show SMTP replies
-d18.2	37.5.63		Show entry to MAIL From:
-d18.100	37.5.64		Pause on SMTP read error
-d19.1	37.5.65		Show ESMTP MAIL and RCPT parameters
-d20.1	37.5.66	X	Show resolving delivery agent: parseaddr()
-d21.1	37.5.67		Trace rewriting rules
-d21.2	37.5.68	X	Trace $& macros
-d21.3	37.5.69		Show subroutine calls
-d21.4	37.5.70		Result after rewriting by a rule
-d21.10	37.5.71		Announce failure
-d21.12	37.5.72		Announce success and show LHS
-d21.15	37.5.73		Show $digit replacement
-d21.35	37.5.74		Show token-by-token LHS matching
-d21.36	37.5.75		Trace class matching in the LHS
-d22.1	37.5.76	X	Trace tokenizing an address: prescan()
-d22.11	37.5.77	X	Show address before prescan
-d22.12	37.5.78		Show address after prescan
-d22.36	37.5.79		Show each token
-d22.101	37.5.80		Trace low-level state machine
-d24.4	37.5.81		Trace address allocation
-d24.5	37.5.82		Trace assembly of tokens
-d24.6	37.5.83		Show result of buildaddr()
-d25.1	37.5.84	X	Trace ″sendtolist″
-d26.1	37.5.85		Trace recipient queueing
-d26.8	37.5.86		Trace self-destructing addresses
-d26.10	37.5.87		Show full send queue in testselfdestruct
-d27.1	37.5.88	X	Trace aliasing
-d27.2	37.5.89	X	Include file, self-reference, error on home
-d27.3	37.5.90	X	Forwarding path and alias wait
-d27.4	37.5.91	X	Print not safe
-d27.5	37.5.92		Trace aliasing with printaddr()
-d27.8	37.5.93		Show setting up an alias map
-d27.9	37.5.94	X	Show uid/gid changes with :include: reads
-d27.14	37.5.95		Show controlling user that caused change in identity
-d27.20	37.5.96		Show how alias will be looked up in a map
-d28.1	37.5.97	X	Trace user database transactions
-d28.2	37.5.98		Show no match
-d28.4	37.5.99		Show result of lookup
-d28.8	37.5.100		Try hes_getmailhost()

Table 27. Debugging Switches by Category (continued)

Category	Bat Book Reference	Useful for Mail Problems	Description
-d28.16	37.5.101		MX records for forward host
-d28.20	37.5.102		Show udp lookup
-d28.80	37.5.103		Preview lookups
-d29.1	37.5.104		Special rewrite of local recipient
-d29.4	37.5.105	X	Trace fuzzy matching
-d29.5	37.5.106		Preview rule set 5
-d29.7	37.5.107		Show overaliasing fuzzy fallback
-d30.1	37.5.108		Trace processing of header
-d30.2	37.5.109		Eat from
-d30.3	37.5.110		Show a to-less header being added
-d30.35	37.5.111		Trace collect states
-d30.94	37.5.112		Trace collect states
-d31.2	37.5.113	X	Trace processing of headers
-d31.6	37.5.114		Is header known?
-d32.1	37.5.115		Show collected headers
-d32.2	37.5.116		Show ARPA mode with setsender
-d33.1	37.5.117		Watch crackaddr()
-d34.1	37.5.118		Watch header assembly for output
-d34.11	37.5.119	X	Trace header generation and skipping
-d35.9	37.5.120	X	Macro values defined
-d35.14	37.5.121		Macro identification
-d35.24	37.5.122		Macro expansion
-d36.5	37.5.123		Trace processing by stab()
-d36.9	37.5.124		Show hash bucket
-d36.90	37.5.125		Trace function applied to all symbols
-d37.1	37.5.126	X	Trace setting of options
-d37.8	37.5.127	X	Trace adding of words to a class
-d38.2	37.5.128	X	Show map opens and failures
-d38.3	37.5.129		Show passes
-d38.4	37.5.130	X	Show result of map open
-d38.9	37.5.131		Trace map closings and appends
-d38.10	37.5.132		Trace NIS search for end of aliases
-d38.12	37.5.133		Trace map stores
-d38.19	37.5.134	X	Trace switch map finds
-d38.20	37.5.135	X	Trace map lookups
-d38.44	37.5.136		Show nis_getcanonname() record
-d39.1	37.5.137		Display digit database mapping
-d40.1	37.5.138		Trace processing of the queue
-d40.3	37.5.139		Show envelope flags

Table 27. Debugging Switches by Category (continued)

Category	Bat Book Reference	Useful for Mail Problems	Description
-d40.4	37.5.140		Show qf file lines as they are read
-d40.8	37.5.141		Show reasons for failure
-d40.9	37.5.142		Show qf and lock file descriptors
-d40.32	37.5.143		Dump the send queue
-d41.1	37.5.144	X	Trace queue ordering
-d41.2	37.5.145		Cannot open qf
-d41.49	37.5.146		Show excluded (skipped) queue files
-d41.50	37.5.147		Show every file in the queue
-d42.2	37.5.148		Show connection checking
-d42.5	37.5.149		Trace caching and uncaching connections
-d43.1	37.5.150		Trace MIME conversions
-d43.3	37.5.151		See the final MIME boundary name
-d43.5	37.5.152		Watch search for boundaries
-d43.8	37.5.153		Show the calculations
-d43.35	37.5.154		Show boundary lines as emitted
-d43.36	37.5.155		Show content transfer encoding
-d43.40	37.5.156		Show parse of Content-Type: header
-d43.99	37.5.157		Print the leading/following comments
-d43.100	37.5.158		Mark collect() and putheader()
-d44.4	37.5.159		Trace safefile()
-d44.5	37.5.160	X	Trace writable()
-d45.1	37.5.161		Show envelope sender
-d45.3	37.5.162		Show saved domain
-d45.5	37.5.163		Show do not send to sender
-d46.9	37.5.164		Show xf file's descriptors
-d48.2	37.5.165	X	Trace calls to the check_rule sets
-d49.1	37.5.166		Trace checkcompat()
-d50.1	37.5.167		Show envelope being dropped
-d50.2	37.5.168		Show Booleans
-d50.10	37.5.169		Also show the send queue
-d51.4	37.5.170		Show queue entries being unlocked
-d51.104	37.5.171		Prevent unlink of xf file
-d52.1	37.5.172		Show isconnect from controlling TTY
-d52.100	37.5.173		Prevent disconnect from controlling TTY
-d53.99	37.5.174		Trace xclose()
-d54.1	37.5.175		Show error return and output message
-d54.8	37.5.176		Show message and flags
-d55.60	37.5.177		Show file locking
-d56.1	37.5.178		Persistent host status tracing

Table 27. Debugging Switches by Category (continued)

Category	Bat Book Reference	Useful for Mail Problems	Description
-d56.2	37.5.179		More persistent host status tracing
-d56.12	37.5.180		Perform a sanity check
-d56.80	37.5.181		Trace creating the path to the status file
-d56.193	37.5.182		Dump MCI record for the host
-d57.2	37.5.183		Monitor vsnprintf() overflows
-d59.1	37.5.184		XLA from contrib
-d60.1	37.5.185	X	Trace map lookups inside rewrite()
-d61.10	37.5.186		Trace gethostbyname()
-d62.1	37.5.187		Log file descriptors before and after all deliveries
-d62.8	37.5.188		Log file descriptors before each delivery
-d62.10	37.5.189		Log file descriptors after each delivery
-d80.1	37.5.190		Content-Length: header (Sun enhancement)
-d81.1	37.5.191		> option for remote mode (Sun enhancement)
-d91.100	37.5.192		Log caching and uncaching connections
-d99.100	37.5.193	X	Prevent backgrounding the daemon

Diagnosis Aids

OS/390 UNIX sendmail stores undeliverable messages in the QueueDirectory that is specified in the configuration file. The directory contains files named `dfxxxxxxxx` and there should be a matching `qfxxxxxxxx` file. The data file (df file) contains the message body that has been queued. There is a df file and a qf file for each message that has been queued.

Try the following in diagnosing problems with OS/390 UNIX sendmail.

- Look in the syslog.log for more information.

  Following is a sample OS/390 UNIX sendmail syslog.log:

  ```
  Mar 4 16:17:15 sendmail Y973078550 : EZZ7514I: sendmail starting
    .
    .
  Mar 4 16:17:47 sendmail Y486539289 : starting daemon (8.8.7):SMTP
  ```

- Use the -v (verbose) command-line switch.
- Check the qf file for queueing concerns.

  The qf file holds all the information that is needed to perform the delivery of a queued mail message. The qf file is line-oriented, containing one item of information per line. The single uppercase character (the code letter) specifies the contents of the line.
- Use the -X (trace log) command-line switch.

  The -X command-line switch can be used to record all input, output, SMTP traffic, and other significant transactions.

The qf File Internals

The complete list of qf code letters is shown in Table 28 on page 288.

Table 28. qf File Code Letters

Code	Bat Book Reference	Meaning	How Many
B	23.9.1	Body type	At most one
C	23.9.2	Controlling user	At most one per R line
D	23.9.3	Obsolete	Obsolete
E	23.9.4	Errors to	Many
F	23.9.5	Flag bits	Many
H	23.9.6	Header definition	Many
I	23.9.7	df file's inode number	Exactly one
K	23.9.8	Time last processed	Exactly one
M	23.9.9	Message (why queued)	At most one
N	23.9.10	Number times tried	At most one
P	23.9.11	Priority (current)	At most one
Q	23.9.12	Original recipient	At most one
R	23.9.13	Recipient address	Many
S	23.9.14	Sender address	Exactly one
T	23.9.15	Time created	Exactly one
V	23.9.16	Version	Exactly one
Z	23.9.17	DSN envelope ID	At most one
$	23.9.18	Restore macro value	At most one
.	23.9.19	End of qf file	At most one

Using the Mailstats Command

The OS/390 UNIX sendmail program provides the ability to gather information that can be used to produce valuable statistics. The StatusFile (S) option is used to specify a `sendmail.st` file into which delivery agent statistics can be saved. The Mailstats program prints a summary of those statistics by printing the `sendmail.st` file.

Mailstats Command—Printing Statistics

Purpose

Use the Mailstats command to print the statistics contained in the `sendmail.st` file.

Format

Parameters

-C *<conf filename>*
Specifies the name of the sendmail configuration file to be used to locate and analyze the OS/390 UNIX sendmail statistics file. If not specified, /etc/sendmail.cf is used as the default.

-s *<stat filename>*
Specifies the name of the OS/390 UNIX sendmail statistics file to be analyzed. If not specified, the statistics file is located on the StatusFile (S) option specified in the OS/390 UNIX sendmail configuration file.

-o Requests mailer names be omitted from the formatted output.

Results

The following example shows the result of a MAILSTAT command.

```
Statistics from Mon Oct 27 14:06:18 1997
 M msgsfr bytes_from msgsto bytes_to Mailer
 3   4        4K     4       4K       local
 5   1        1K     1       1K       esmtp
 ============================================================
 T   5        5K     5       5K
```

The first line of output shows the time the statistics file was begun. The M column shows the index into the internal array of delivery agents, and the Mailer shows the symbolic name. The lines that follow show the number of messages and the total size in kilobytes of the messages received (msgsfr) and sent (msgsto) for each delivery agent. The bottom line shows the totals.

Note: A delivery agent that has handled no traffic is excluded from the report.

Invoking Popper

Popper must be invoked by INETD, upon the initiation of a TCP connection to the POP3 port 110 (or any other specifically-configured port). You must invoke the Popper deamon using the following command:

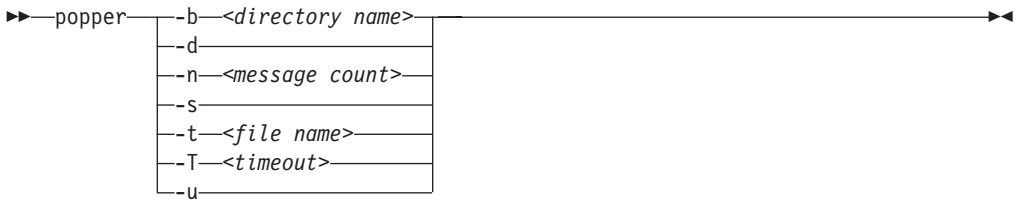
```
popper -d
```

Popper Command—Administering Received Mail

Purpose

If the receiver's MUA does not have direct access to the mail spool file, use Popper to access the mail spool on the local host. OS/390 Popper will be used when a POP3 server is needed.

Format



Parameters

The following command line options can be used when invoking Popper.

-b *<directory name>*
Specifies the name of the directory in which bulletins are found. If not specified, /usr/mail/bulletins is used as the default.

-d Requests additional debugging messages be turned on.

-n *<message count>*
Specifies the number of old bulletins to be delivered to new users. If not specified, no bulletins are delivered.

-s Requests statistics logging be turned on.

-t *<file name>*
Specifies a trace file for all message logging. If not specified, messages are logged via the syslog facility.

-T *<timeout>*
Specifies the time, in seconds, before an idle POP3 connection is terminated. The POP3 RFC specifies a minimum timeout of 600 seconds, but in practice such a long timeout does not work well. (When a connection gets aborted, the user is locked out of his mailbox for the timeout period.) If not specified, 120 seconds is used as the default timeout period.

-u Requests the user's mailbox be updated on abort. The POP3 RFC specifies that mailboxes should not be updated (that is, no messages should be deleted) if a connection is aborted abnormally. This option forces an update to occur despite the aborted connection. If not specified, no update will occur on aborted connections.

Chapter 6. Using Remote Printing

CS for OS/390 provides client and server support for remote printing. The remote printing application enables you to spool data sets remotely to a Line Printer Daemon (LPD). The Line Printer Requester (LPR) sends the spooled data set to a specified print server host and to a specified printer.

This chapter describes the remote printing commands listed in Table 29.

Note: Although this chapter describes the commands and parameters that are valid for the MVS LPR client, you may not get the same results from non-MVS servers, because those servers may not support the same commands and parameters.

Table 29. Remote Printing Commands

Command	Description	Page
LPQ	Request a list of the printer queue on a remote printer	294
LPR	Print to a remote printer	296
LPRM	Remove a job from the printer queue on a remote host	309
LPRSET	Set the default printer and host name	311

LPQ Command—Request a List of the Printer Queue on a Remote Printer

Purpose

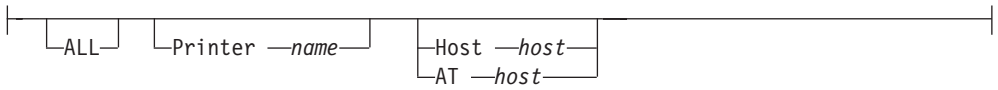
Use the LPQ command to request a list of the printer queue on a remote printer from the LPD server controlling that printer.

Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPQ command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format

Opt Parms 1



Opt Parms 2

Parameters

job_id
Specifies either a user ID (this must not start with a digit), or a job number in the remote printer queue. If you do not specify *job_id* with the LPQ command, all the jobs in the remote printer queue are listed.

Note: *job_id* is case-sensitive on some systems.

ALL
Prints a long report, which shows the source host and other print job information.

Printer *name*
Specifies the name of the printer for which you want the printer queue listed.

The printer name cannot contain an @ symbol.

Host *host*
Specifies the name or IP address of the printer host. AT® is a synonym for this option.

AT *host*
Specifies the name or IP address of the printer host. Host is a synonym for this parameter.

TRace
Turns on the trace details for interaction with the remote printer. TRace always overrides TYpe because TYpe is a subset of TRace. Use this option whenever you need to document a problem.

TYpe
Displays the progress of the command.

Version
Displays the version of the program.

Examples

- Query the printer `lp0` on the system `os2sys1` and print a short listing of the jobs that are queued for the `lp0` printer:

  ```
  LPQ (PRINTER lp0 HOST os2sys1
  ```

- If the LPRSET command was previously issued (LPRSET lp0@os2sys1), using the following LPQ command has the same effect as issuing the command in the previous example.

  ```
  LPQ
  ```

- Get a long listing of the jobs queued, including the name of the host that created the jobs:

  ```
  LPQ (PRINTER lp0 HOST os2sys1 ALL
  ```

- List the jobs for a user named `smith`:

  ```
  LPQ smith (PRINTER lp0 HOST os2sys1
  ```

- Get information only about job 123:

  ```
  LPQ 123 (PRINTER lp0 HOST os2sys1
  ```

Usage

- If the printer or host name are not specified in the LPQ command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can specify these variables with the LPRSET command. You can use these variables to set up a default printer and host to be used if you do not specify a printer or host.
- User names in a query are case-sensitive. For example, `smith` and `SMITH` are not the same names. Also for example, on UNIX systems, `lp0` and `LP0` can refer to different printers.
- Some systems do not answer with the job information when you use a job number for a job that was not produced by the querying system.
- You would not normally use the LPQ command to query an MVS system, because the LPD queue on MVS processes so quickly.

LPR Command—Print to a Remote Printer

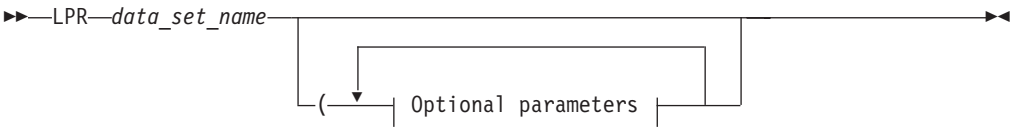
Purpose

Use the LPR command to print to a remote printer.

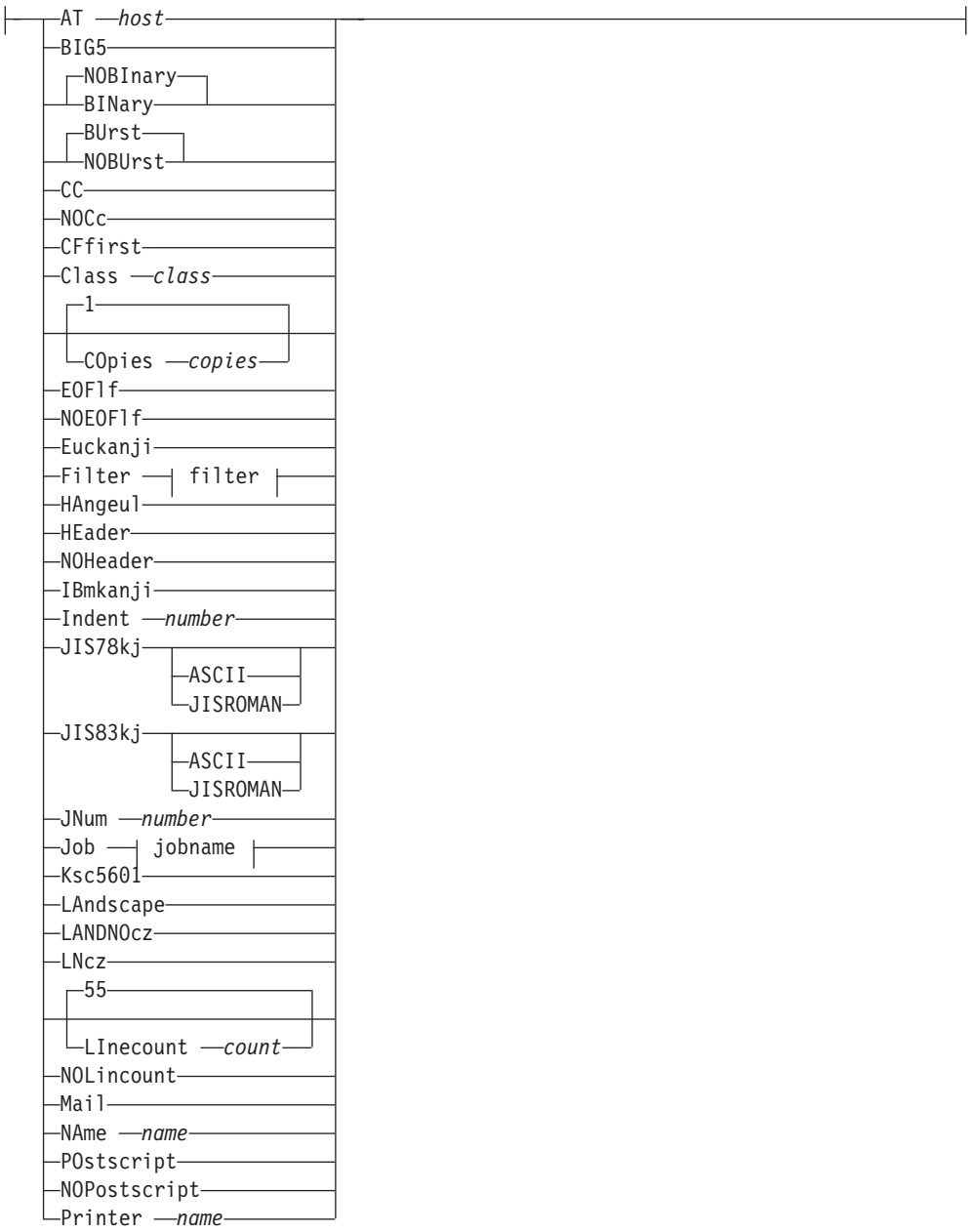
Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPR command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

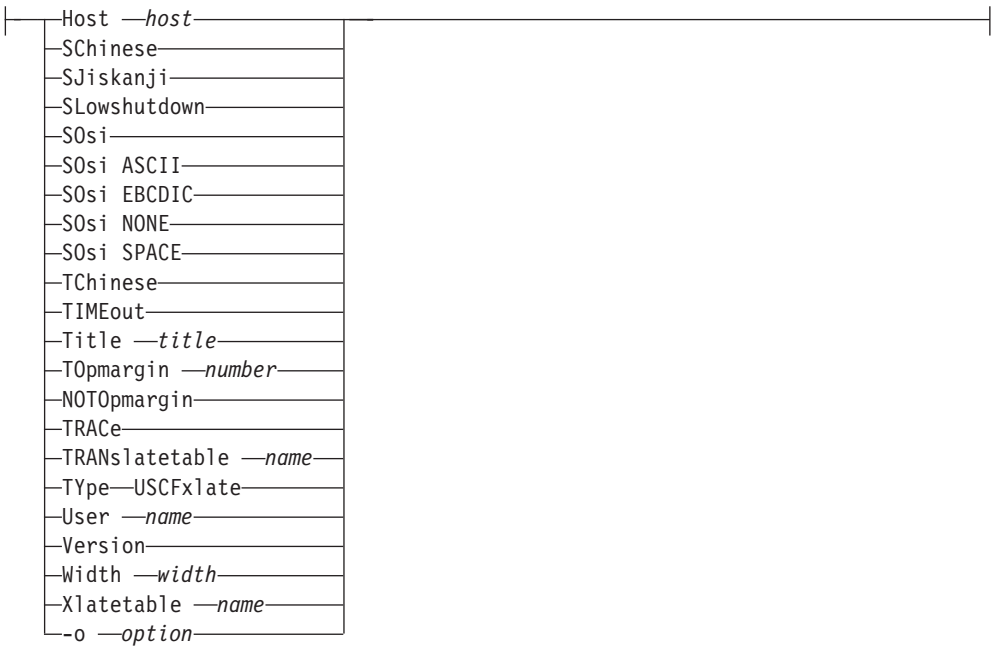
Format



Optional parameters:



More optional parameters:



filter:

jobname:

Parameters

data_set_name
Specifies the name of the data set to be printed.

AT *host*
Specifies the name or IP address of the printer host machine. Host is a synonym for this option.

BIG5
Converts data from Traditional Chinese host DBCS to Big-5 DBCS when transferring data to a remote system. LPR loads the BIG5 DBCS translation table from TCPCHBIN binary table data set.

BINary
Causes LPR to send the data without translation and without any indication of record boundaries. Use this option if the data set is already in ASCII.

Note: The MVS LPD server always converts data sets in ASCII to EBCDIC, and there is no option to disable this conversion.

NOBInary
Causes LPR to convert the data from EBCDIC to ASCII when it is sent to the remote system. This is the default.

BUrst
Causes a burst page to be printed on the remote printer. This is the default.

NOBUrst
Prevents a burst page from being printed on the remote printer.

CC
Causes the remote system to interpret the first character of each line as carriage control.

Records containing control characters that are not valid are deleted. If all records in the file are deleted, LPR processes the empty file.

CC is the default if the record format is FA, FBA, FBM, FM, VA, VBA, VBM, or VM. The characters used to specify these record formats have the following meanings:

F Fixed record length
V Variable record length
B Blocked records
A Records containing ISO/ANSI control characters
M Records containing machine code control characters

NOCc
Prevents the remote system from interpreting the first character of each line as an ASA carriage control.

CFfirst
Specifies that LPR will send the ControlFile describing the data before it sends the DataFile that contains the data. This option enables some LPD servers to print larger jobs since the data can be printed as it is received.

Specify this option when small jobs print, but large jobs don't.

Note: Even when this option is specified, the print job may still be too complex to print.

CLass *class*
Specifies the class name to the remote system. The class name is printed on the banner pages. The default is the sending host name.

To override the SYSOUT CLASS of a job, the CLASS value must be only one alphanumeric character. For example, CLASS c.

COpies *copies*
Specifies the number of copies to be printed. The default is one copy.

EOFIf
Causes an ASCII line feed after the last line of data of a file formatted by ANSI carriage control. Since ANSI carriage control positions the paper before the line

is printed, you cannot specify how the paper should be positioned after the last line prints. The normal network standard is to terminate every file with a line feed. Some LPDs do not print the last line if the line feed is not added. For example, in ANSI carriage control, a 'F1'X means eject to a new page before printing the lines. ANSI or ASA CC is used in files with RECFM=FBA.

EOFlf is the default for ANSI CC files. Specify NOEOFlf if your paper is positioned incorrectly and an extra line is printed because LPR adds a line feed.

This parameter does not impact files that are not ANSI CC. For example, BINary, POstscript, LAndscape and Machine CC files are not affected by EOFlf.

NOEOFlf
Inhibits adding the ASCII line feed after the last line of a file that is formatted by ANSI carriage control. Since ANSI carriage control positions the paper before the line is printed, how the paper should be positioned after the last line cannot be specified by the carriage control characters. The normal network standard is to terminate every file with a line feed. Specifying this parameter causes the last byte of data to be the last byte sent; a line feed is not added.

For example, an ANSI carriage control 'F1'X means eject to a new page before printing the line. ANSI or ASA CC is used, for example, in files with RECFM=FBA.

Specify NOEOFlf if your paper is positioned wrong because an extra line is being printed at the end of the file. Some applications specify the positioning and do not have the extra line feed sent by LPR.

For example, BINary, POstscript, and Machine CC files are not affected by this parameter. This option does not apply when you specify LAndscape.

Euckanji
Causes the data to be converted from EBCDIC DBCS to Extended UNIX Code kanji ASCII DBCS when it is sent to the remote system. LPR loads the EUCKANJI DBCS translation table from the TCPKJBIN binary translate table data set.

Filter *filter*
Specifies the type of processing to be done on the data by the remote system. The *filter* is written as a single letter. Both uppercase and lowercase letters are accepted, but uppercase letters are converted to lowercase.

Note: The filter values must also be defined in the SERVICE statement of the configuration data set for the MVS LPD server. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information on the LPD server configuration data set.

In addition to the following list of filter codes, there are filters supported by other servers described in RFC1179. (For a list of Requests for Comments (RFCs), see "Appendix G. Related Protocol Specifications (RFCs)" on page 595.)

Filter Code	Description
f	Print as a sequence of lines
l	Print, passing through all control characters
p	Print with pagination

Filter Code	Description
r	Print, interpreting the first column as FORTRAN carriage control characters. The supported IBM FORTRAN carriage control characters are `1`, `0`, `+`, and `-`. When using the MVS LPD server to print files with filter code r, the PAGESIZE parameter on the SERVICE statement for the printer you are printing to also affects pagination. See “Usage” on page 307 for further information.

HAngeul
Causes the data to be converted from EBCDIC DBCS to Hangeul ASCII DBCS when it is sent to the remote system. LPR loads the HANGEUL DBCS translation table from the TCPHGBIN binary translate table data set.

HEader
Causes a page header to be inserted by the client at the top of every printed page if the NOCc and NOBInary options are in effect. To cause the server to insert page headers, use `p` as the value of *filter* and specify the NOHeader option.

NOHeader
Prevents the client from inserting page headers.

IBmkanji
Causes the data to be sent without translation as IBM (EBCDIC) kanji. This parameter performs the same function as the BINary parameter.

Indent *number*
Specifies the number of columns the remote system indents the output when `f` or `p` is specified as the value of *filter*.

JIS78kj ASCII
Causes the data to be converted from EBCDIC DBCS to JIS 1978 kanji ASCII DBCS, using the ASCII shift-in escape sequence `ESC ( B`, when it is sent to the remote system. LPR loads the JIS78KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS78kj JISROMAN
Causes the data to be converted from EBCDIC DBCS to JIS 1978 kanji ASCII DBCS, using the JISROMAN shift-in escape sequence `ESC ( J`, when it is sent to the remote system. LPR loads the JIS78KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS83kj ASCII
Causes the data to be converted from EBCDIC DBCS to JIS 1983 kanji ASCII DBCS, using the ASCII shift-in escape sequence `ESC ( B`, when it is sent to the remote system. LPR loads the JIS83KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JIS83kj JISROMAN
Causes the data to be converted from EBCDIC DBCS to JIS 1983 kanji ASCII DBCS, using the JISROMAN shift-in escape sequence `ESC ( J`, when it is sent to the remote system. LPR loads the JIS83KJ DBCS translation table from the TCPKJBIN binary translate table data set.

JNum *number*
Specifies a specific job number for the print request, where *number* is a unique, three-digit number in the range 000–999. This job number is used by LPR to name the temporary data and control files, such as:

```
dFA123hostname   cFA123hostname
```

The JNum parameter is not valid from NPF; do not specify JNum in the NPF OPTIONS file.

If JNum is not specified for LPR, the three-digit number is randomly generated by LPR.

Job *jobname*
Specifies the job name to the remote system. The default name is the full data set name. The job name is printed on the banner pages.

The following parameters are available with the Job parameter and are entered as `PARAMETER=value`. When you use these parameters, you must separate them from Job or *jobname* by a blank and from other options by a comma and no blank. For example,

```
Job PASS=password,FOR=userid
```

DEST Sets the destination node. The default is the node on which the LPR client is running.

FOR Specifies a user ID other than the sending user ID for which the output is to be spooled. The default is the sender's ID.

FORM Identifies the form on which the data is printed. This is the equivalent of the form-name subparameter in the MVS SYSOUT parameter on the DD card. Therefore, the form name specified can be 1 through 4 alphanumeric or national ($, #, @) characters.

IDENTIFIER Sets the destination ID. The default is SYSTEM.

LINECOUNT Specifies a numeric field indicating the number of lines on a page. This option overrides the PAGESIZE parameter of LPD.

OTHERS Causes all subsequent options to be ignored. This option is ignored by the MVS LPD server.

PASS Specifies the password. The default is no password, which causes the job to fail if the RACF option is specified for the service.

PRIORITY Specifies the transmission priority. The default is 50.

Ksc5601
Causes the data to be converted from EBCDIC DBCS to Korean Standard Code KSC-5601 ASCII DBCS when it is sent to the remote system. LPR loads the KSC5601 DBCS translation table from the TCPHGBIN binary translate table data set.

LAndscape
Converts a non-PostScript data set to a PostScript data set for printing with print lines parallel to the long edge of the paper. If the remote printer can process PostScript output, the data set is printed in landscape format (rotated 90 degrees). Some non-printing EBCDIC characters below X'3F' are changed to blanks.

Note: An ASCII cntl-Z X'1A' is sent after the data. If this dos-EOF character causes problems, use the LANDNOcz option instead.

LANDNOcz/LNcz
Converts a non-PostScript data set to a PostScript data set for printing with print lines parallel to the long edge of the paper. If the remote printer can

process PostScript output, the data set is printed in landscape format (rotated 90 degrees). Some non-printing EBCDIC characters below X'3F' are changed to blanks.

Note: Normally, you should use the LAndscape option, unless the dos-EOF character (X'1A') sent after the file causes problems. LN and LNcz are abbreviations for LANDNOcz.

LInecount *count*
Determines the number of lines to be printed before a new heading is printed. This option is meaningful only for a data set that does not have the CC option specified either explicitly or by default.

The valid range for LInecount is 0 to the Pascal integer MAX number 2147483647. The default value is 55. To suppress printing a header before each new page, specify LInecount 0. Specifying LInecount 0 has the same effect as specifying the NOLinecount option.

NOLinecount
Prevents a header being printed before each new page. Specifying NOLinecount has the same effect as specifying LInecount 0.

Mail
Causes mail to be sent to the user when the printing operation ends (for those servers that support this).

NAme *name*
Specifies the job information to be provided by the remote system in response to a query. Only the *name* or *files* portion of the query is displayed. This option is not honored by all remote printing servers.

POstscript
Inserts the header required by some systems to recognize a PostScript data set.

NOPostscript
Prevents a PostScript data set from being recognized as a PostScript data set.

Printer *name*
Specifies the name of the printer on which you want the data set printed.

Host *host*
Specifies the name or IP address of the printer host machine. AT is a synonym for this option.

SChinese
Converts data from Simplified Chinese host DBCS to Simplified Chinese PC DBCS when transferring data to a remote system. LPR loads the SCHINESE DBCS translation table from TCPSCBIN binary table dataset.

SJiskanji
Causes the data to be converted from EBCDIC DBCS to Shift JIS kanji ASCII DBCS when it is sent to the remote system. LPR loads the SJISKANJI DBCS translation table from the TCPKJBIN binary translate table data set.

SLowshutdown
Causes slower TCP/IP connection termination after the job is sent to the printer. This option is rarely needed. This option is provided for print servers which discard the print job just successfully received when LPR uses the fast shutdown. Before using this option, check the print server for errors (such as spool file full) that may be causing the job to be discarded instead of printed.

SOsi
Determines how any EBCDIC DBCS shift-out (’0E’X) and shift-in (’0F’X) characters in the input file are handled. The ASCII, EBCDIC, or SPACE parameters specifies what is used as shift characters in the ASCII output from LPR. If SOsi is specified without a following parameter, ASCII is used as the default.

If you do not specify SOsi, shift-out/shift-in characters are not used in the ASCII data stream. Therefore, the EBCDIC DBCS shift characters are just removed during the translation to ASCII. This is the same as specifying SOsi NONE.

SOsi has no effect on DBCS translations JIS78KJ, JIS83KJ, and IBMKANJI. It is used with other DBCS translation such as BIG5, EUCKANJI, HANGEUL, KSC5601, SCHINESE, SJISKANJI, and TCHINESE.

SOsi ASCII
Specifies that DBCS data strings in the ASCII output are delimited by special shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out (’0E’X) becomes (’1E’X), and ’0F’X becomes ’1F’X.

SOsi EBCDIC
Specifies that DBCS data strings in the ASCII output are delimited by EBCDIC shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out (’0E’X) remains EBCDIC shift-out (’0E’X), and ’0F’X remains ’0F’X.

SOsi NONE
Specifies that DBCS data strings in the ASCII output are not delimited by any shift-out/shift-in characters. As the data is translated from EBCDIC to ASCII, any EBCDIC DBCS shift-out (’0E’X)

SOsi SPACE
Specifies that DBCS data strings in the ASCII output are delimited by ASCII space (’20’X) characters. As the data is translated from EBCDIC to ASCII, input EBCDIC shift-out (’0E’X) becomes ASCII space (’20’X) and EBCDIC shift-in (’0F’X) also becomes ’20’X.

TChinese
Causes the data to be converted from EBCDIC DBCS to Traditional Chinese (5550) ASCII DBCS when it is sent to the remote system. LPR loads the TCHINESE DBCS translation table from the TCPCHBIN binary translate table data set.

TIMEout
Specifies that LPR wait 5 minutes for an ACK or NACK from the LPD printer. If ACK or NACK does not arrive, LPR terminates the connection with Error Number=73. Since LPR waits for ACK’s in several places, this error can occur in different LPR messages, such as EZB1048E.

For some printers, the ACK is not returned until the job has printed. For these printers, specifying the TIMEout option can cause long jobs to stop printing.

When the TIMEout option is not specified, LPR waits as long as the TCP/IP connections exists. For some printers, this could tie up the PORT (and any NPF thread using LPR on that PORT) until an operator intervenes to fix the printer. For NPF users, TIMEout should be added as an LPR option for these printers. Refer to the *OS/390 eNetwork Communications Server: IP Network Print Facility* for information about configuring LPR options in NPF.

Title *title*
Specifies the title assigned to a data set printed with the FILTER `p` option.

TOpmargin *number*
Specifies the number of lines designated for the top margin.

NOTOpmargin
Indicates that blank lines are not inserted at the top of each page.

TRACe
Turns on the trace details for interaction with the remote printer. TRace always overrides TYpe because TYpe is a subset of TRace.

TRANslatetable *name*
Specifies that the translate table to be used by the client is found in the *name*.TCPXLBIN data set. Xlatetable is a synonym for this option.

If a DBCS conversion parameter is specified, *name* is used to determine which DBCS translation table to load. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about using translation tables for LPR.

TYpe
Displays the progress of the command as the data set is being processed.

USCFxlate
Specifies that a single byte translation table such as JPNKANA be used for the print data. The control file generated by LPR and sent to LPD contains upper and lower case alphanumeric characters. Specify this option if any of them are being translated incorrectly, causing the LPD to reject the print jobs.

User *name*
Specifies a name that overrides the user identification of the program that is requesting the print job, prints on the banner page, and becomes the user identification of the mail option. The *name* field cannot be longer than eight characters. If you do not enter the user name parameter, it defaults to the system user identification or to the job name.

Version
Displays the version of the program.

Width *width*
Specifies the line width of a data set printed with the FILTER options `f`, `l`, `p`, or `r`.

Xlatetable *name*
Specifies that the translate table to be used by the client is found in the *name*.TCPXLBIN data set. TRANslatetable is a synonym for this option.

If a DBCS conversion parameter is specified, *name* is used to determine which DBCS translation table to load. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about using translation tables for LPR.

-o *option*
Specifies an option that the control file in PSF for AIX (PSF/6000) or InfoPrint® uses to format the print job. Any -o option honored by PSF/6000 can be passed to the control file using this parameter of the LPR command. This parameter must be issued without a blank between the -o and the option.

Below are some sample options. None of these options are standard LDP options. They are not honored by most LPD print servers.

-ochars=GT15
Sets the value of the CHARS parameter to GT15.

-obin=2 Sets the input bin to 2 (use alternate input bin).

-opagedef=P13700
Sets the value of PAGEDEF parameter to P13700.

-oformdef=F1SEPA
Sets the value of the FORMDEF parameter to F1SEPA.

-ocopies=002 Sets the JCL COPIES count to 2. This parameter is not honored by most LPD servers. Other LPD servers ignore all control file information and print one copy. For these printers, issue the LPR command multiple times to get multiple copies.

IBM Print Services for AIX (S544-3878-03) contains detailed descriptions of the -o options for PSF/6000.

Examples

- Print the data set `TEST.LISTING` on a printer named `lp0` on the system `mvs1`:

  ```
  LPR TEST.LISTING (PRINTER lp0 HOST mvs1
  ```

- If `TEST.LISTING` has a record format that contains carriage control such as VBA, the first character of each line is interpreted as carriage control. To prevent the first character of each line from being interpreted as carriage control, use the following command:

  ```
  LPR TEST.LISTING (PRINTER lp0 HOST mvs1 NOCC
  ```

- If this LPRSET command was issued:

  ```
  LPRSET lp0@mvs1
  ```

  the following LPR command would also print the data set on printer LP0 on the host MVS1 and prevent the first character of each line from being interpreted as carriage control:

  ```
  LPR TEST.LISTING (NOCC
  ```

- Print the data set `TEST.LISTING` in landscape mode:

  ```
  LPR TEST.LISTING (LANDSCAPE
  ```

  The following example shows the PostScript attributes used in the LAndscape option:

  ```
  614 25 translate 90 rotate .88 .76 scale
  /n 1 def
  /fs 10 def
  /ls 11.2 def
  /ld ls 2 mul def
  /lt ls 3 mul def
  /t 740 fs sub def
  /y t def /ff t def /os 20 def
  /s 512 string def
  /Courier-Bold findfont
  fs scalefont setfont
  /p {n {copypage} repeat erasepage} def
  /i (%stdin) (r) file def
  /{/c i read not {p stop} if def
  c 26 eq {p stop} if
  /x 20 def
  /y c 43 eq {y /x os def}
  ```

```
{c 32 eq {y ls sub}
{c 48 eq {y ld sub}
{c 45 eq {y lt sub}
{c 49 eq {ff} {y} ifelse}
ifelse} ifelse} ifelse} ifelse def
/ff 0 def
y 65 le {p /y t def} if x y moveto
/os i s readline not {p stop} if dup show
length 0 eq {20} {20.72} ifelse def } loop
```

To understand these attributes you may need to reference a PostScript manual.

- If a data set TEST has a low-level qualifier of LISTPS (PostScript), use the following command to send TEST to a PostScript-capable printer without specifying the PostScript option:

  ```
  LPR TEST.LISTPS (PRINTER lp0 HOST mvs1
  ```

- Print a FORTRAN source program with 57 lines on each page:

  ```
  LPR TEST.FORTRAN (LINECOUNT 57
  ```

Usage

- When sending a print job to a printer that has RACF in its definition in the LPD.CONFIG data set, you must specify the password. If it is for a different user ID, you must specify that password and user ID as follows:

  ```
  Job PASS=password,FOR=userid
  ```

- If the printer or host name are not specified in the LPR command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can specify these variables with the LPRSET command.
- LPR normally issues messages only if there is an error. If you want to track the progress of the command, use the TYpe or TRACe parameter.
- You can use the LPR command to send PostScript data sets to a printer that can print documents in that language. LPR checks that no incompatible options were given, if it is a PostScript data set. You can override this check, if you want to print a PostScript program with the NOPostscript option. UNIX systems examine the first few characters of a data set (looking for %!) to determine if a data set is a PostScript data set. If you have PostScript data sets that do not contain the characters %!, use the POstscript parameter to add them.
- Carriage control is interpreted line by line. A data set can mix ASA and machine carriage control. Interpretation is done by converting the controls to the appropriate ASCII sequences, before the data set is sent to the remote system. Lines that have incorrect carriage control are not printed.
- When a data set is printed without carriage control, LPR adds a heading line that shows the name of the data set, the title of the system on which the LPR command is running, and a page number. You can specify the number of lines to be printed (excluding the 3 heading lines) with the LInecount parameter.
- When you specify a filter code, LPR ignores CC, HEader, NOCc, NOHeader, and TOpmargin. When a filter code of f, l, p, or r is specified, LPR stops paginating the data set it is printing. Instead, it sends the data in the data set as plain lines. The following list provides a description of these filter codes:

Filter Code	Description
f	Print as a sequence of lines
l	Print, passing through all control characters
p	Print with pagination

Filter Code	Description
r	Print, interpreting the first column as FORTRAN carriage control characters. The supported IBM FORTRAN carriage control characters are `1`, `0`, `+`, and `-`.

When using the MVS LPD server to print files with filter code r, the PAGESIZE parameter on the SERVICE statement for that printer in the LPD CONFIG file. The PAGESIZE parameter defaults to 60 if it is not specified. To prevent unwanted page ejects, change the value specified on the PAGESIZE parameter to a number greater than the actual number of lines on the pages being printed. For example, you can specify 100000 for the PAGESIZE parameter.

When you specify a filter code of `c`, `d`, `g`, `n`, `t`, or `v`, LPR transmits the data as a byte stream (as though you specified the BINary option).

LPRM Command—Remove a Job from the Printer Queue on a Remote Host

Purpose

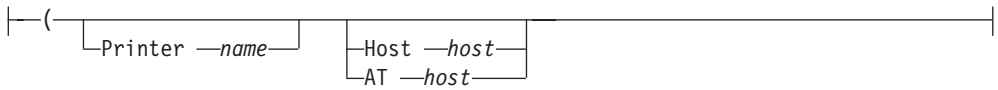
Use the LPRM command to remove a job from the printer queue on a remote host.

Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPRM command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format

Opt Parms 1



Opt Parms 2

Parameters

job_id
Specifies either a user ID (this must not start with a digit), or a job number in the remote printer queue. If you do not specify *job_id* with the LPRM command, your currently active job is removed.

Printer *name*
Specifies the name of the printer associated with the job.

Host *host*
Specifies the name or IP address of the printer host. AT is accepted as a synonym for HOST.

AT *host*
Specifies the name or IP address of the printer host. Host is a synonym for this option.

TRace
Turns on the trace details for interaction with the remote printer. TRace always overrides TYpe because TYpe is a subset of TRace.

TYpe
Displays the progress of the command.

Version
Displays the version of the program.

Examples

- Cancel job number 123 on the printer `lp0` on the local system `os2sys1`:

  ```
  LPRM 123 (PRINTER lp0 HOST os2sys1
  LPRM 123 (PRINTER lp0 AT os2sys1
  ```

  If the job is in the queue, it is removed. If the job is currently active, it is stopped.
- If the LPRSET command was previously issued (LPRSET lp0@os2sys1), using the following LPRM command has the same effect as issuing the command in the previous example:

  ```
  LPRM 123
  ```
- Cancel the currently active job:

  ```
  LPRM (PRINTER lp0 HOST os2sys1
  ```

Usage

- If the printer and host name are not specified in the LPRM command, the last LASTING.GLOBALV variables for PRINTER and PRTHOST in the *user_id*.LASTING.GLOBALV data set are used as the defaults. You can set these variables with the LPRSET command. You can use these variables to set up a default printer, which is used if you do not specify a printer.
- Removing the currently active job can depend on the number of jobs currently printing. If you have two jobs printing, and you use the LPRM command without the *job_id* parameter, the first job may finish, but you could inadvertently remove the second job instead.

LPRSET Command—Set the Default Printer and Host Name

Purpose

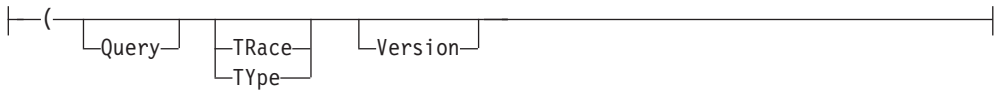
Use the LPRSET command to set the default printer and host name. The printer and host name can also be included in the line printer commands (LPR, LPQ, and LPRM).

Notes:

1. Do not use the forward slash character (/) in any parameter value for this command.
2. The TSO LPRSET command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format

Optional Parameters



Parameters

printer@host
Specifies the name of the printer and host to be used.

Query
Displays the current settings for the default printer and host.

TRace
Turns on the trace details for the recording of the printer and remote host name. TRace always overrides TYpe because TYpe is a subset of TRace.

TYpe
Displays the progress of the command.

Version
Displays the version of the program.

Examples

- Set the default printer and host as the printer `lp0` on the local system `mvs1`:

  ```
  LPRSET lp0@mvs1
  ```

- Display the current version of LPRSET:

  ```
  LPRSET (VERSION
  ```

- Display the current settings:

  ```
  LPRSET (QUERY
  ```

Usage

- When you use LPRSET to set the printer and host, a data set by the name *userid.LASTING.GLOBALV* (where *userid* is the TSO user ID) is created or updated. If this data set does not exist, you must be able to create this data set. *userid.LASTING.GLOBALV* is the name required by LPRSET.
- Printer names can be case sensitive. The printer name must be spelled the way the host uses it. For example, on UNIX systems, `lp0` and `LP0` can refer to different printers.

  Also be aware that ISPF panels default to uppercase unless otherwise specified.
- When you query the current settings on the same command that you set the default printer and host, the query of the current settings is done before the specified default printer and host are set. For example, if the printer was set to PRINT1@RALVM13, and you perform the query, `LPRSET PRINT2@RALVM13(Q`, the message returned says, `PRINTER is PRINT1@RALVM13`. However, it also changes the default printer to PRINT2. So if you query again, it would respond, `PRINTER is PRINT2@RALVM13`.

  If you want to set the default printer and host and see that the defaults were set in the same LPRSET command, use the TYPE parameter. For example, if you want to set the default printer and host to PRINT2 on RALVM13 and then see the results afterward, you would enter the following command.

  ```
  LPRSET PRINT2@RALVM13 (TYPE
  ```

Chapter 7. Authenticating Network Users Using Kerberos

This chapter describes the Kerberos name structures and Kerberos user commands. Examples of Kerberos user commands are also provided in this chapter.

Kerberos is a system that provides authentication and security services to users in a TCP/IP network environment.

Your Kerberos system administrator can provide you with your Kerberos name and password when you register with the Kerberos system.

For Kerberos administrator commands and commands for setting up the Kerberos servers, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

Understanding Kerberos Name Structures

Before using the Kerberos commands, you should be familiar with the structure of a Kerberos name. A Kerberos name consists of the following three parts:

Name Description

Principal name
Indicates the unique name of a user (client) or service.

Instance
Distinguishes among the variations of the principal name. An instance allows for the possibility that the same client or service can exist in several forms that require distinct authentication.

For client users, an *instance* can provide different identifiers for different privileges. For example, the `admin` instance provides special privileges to the users assigned to it.

For services, an instance usually specifies the host name of the machine that provides the service.

Realm Indicates the name of an administrative entity. The realm identifies each independent Kerberos site. The principal name and instance are qualified by the realm to which they belong, and are unique only within that realm. The realm is commonly the domain name.

Kerberos Commands

Before you can use the Kerberos commands, the MVSKERB server must be running on one of the hosts on your network. For information about how to set up the MVSKERB server, and how to use the Kerberos system administrator commands and utilities, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

You can enter the Kerberos commands at a TSO command prompt. The Kerberos user commands are listed in Table 30 on page 313.

Table 30. Kerberos Commands

Subcommand	Description	Page
KDESTROY	Destroy currently active Kerberos tickets	315

Table 30. Kerberos Commands (continued)

Subcommand	Description	Page
KINIT	Initiate a session with the Kerberos Authentication System	316
KLIST	List currently held Kerberos tickets	318
KPASSWD	Change passwords	319

KDESTROY—Delete Kerberos Ticket Data Sets

Purpose

Use the KDESTROY command to delete a currently active Kerberos tickets data set.

Format

Parameters

-f Deletes your active Kerberos authorization tickets data set without displaying a status message.

-q Eliminates the beep response to your terminal if the tickets are not deleted.

Usage

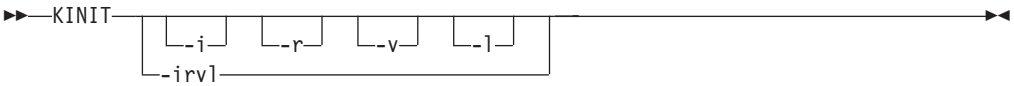
A status message indicating the success or failure of the operation is displayed on the screen. If KDESTROY cannot delete the ticket file, the system sends a beep response to your terminal.

KINIT—Connect to the Kerberos System

Purpose

Use the KINIT command to connect to the Kerberos System.

Format



Parameters

-i Prompts you for a Kerberos *instance*.

Use the -i parameter if you are registered in the Kerberos database with a nonnull *instance*. For example, you are registered as a services provider or a remote administrator, which has an `admin` *instance*.

-r Prompts you for a Kerberos *realm*.

Use the -r parameter if you want to log on to the Kerberos system in a *foreign realm*.

-v Specifies verbose mode for the Kerberos response. The response contains the name of the Kerberos *realm*, and a status message indicating the success or failure of your logon attempt.

-l Specifies the time-to-live of the ticket, if the time-to-live is shorter than the time-to-live specified in the client, server, or database.

Examples

The following is a sample of the information displayed after invoking the KINIT command.

```
Kerberos Initialization
Kerberos name:
<myname>
password:
<mypassword>
```

Usage

- The KINIT command attempts to initiate a session with the Kerberos system. This command enables you to obtain an initial ticket to communicate with the ticket-granting service.

  If the KINIT command is processed successfully, you are returned to a TSO command prompt and an initial ticket file is saved in the *user_id*.TMP.TKT0 data set. Your next step is to use the KLIST command to verify the initial ticket.
- The KINIT command may not function correctly if the clock on the local host is not synchronized with the clock on the host running the Kerberos authentication server.
- Users registered with a null *instance* who want to log on to the Kerberos system in their *local realm* must use the KINIT command without a parameter.

- The KINIT -irvl command requests the system to prompt you for an *instance* and a *realm* throughout your Kerberos session, and requests that Kerberos responses be displayed in verbose mode. Using the KINIT -irvl command has the same effect as using KINIT -i -r -v -l.

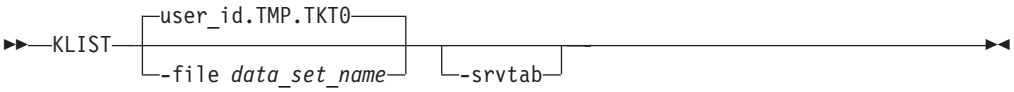
  When you use a Kerberos command with multiple parameters, do not enter spaces between the parameters; any entry (parameter) after a space is ignored.

KLIST—Display Your Current Tickets

Purpose

Use the KLIST command to list all Kerberos tickets or service keys.

Format



Parameters

-file *data_set_name*
Specifies the name of the data set to be used as the ticket data set. The default data set name is *user_id*.TMP.TKT0.

-srvtab
Lists the contents of the key data set called *user_id*.ETC.SRVTAB. The keys to all services provided by the same *instance* are in the key data set.

Examples

Display current Kerberos tickets:

```
Ticket file:user_id.TMP.TKT0
Principal:galina@univ.lab.chem

  Issued           Expires          Principal
Dec 20 13:49:16  Dec 20 21:49:16  krbtgt.univ.lab.chem@univ.lab.chem
```

Usage

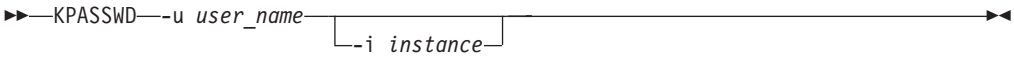
If the KLIST command is not processed successfully, an error message is displayed indicating the failure of the KLIST command.

KPASSWD—Change Your Password

Purpose

Use the KPASSWD command to change your Kerberos password.

Format



Parameters

-u *user_name*
Specifies the user.

-i *instance*
Specifies the *instance*, if you are registered with the Kerberos database with a nonnull *instance*.

Usage

When you use the KPASSWD command, Kerberos prompts you for your current password. After verifying that the current password is correct, Kerberos prompts you twice for the new password. After you enter the new password, a message is displayed indicating the success or failure of the change.

Chapter 8. Using GDDMXD/MVS with the X Window System

This chapter describes GDDMXD/MVS and the GDDMXD CLIST. This chapter also describes how to use GDDMXD/MVS user-specified options and keyboard functions. Problem determination information associated with GDDMXD/MVS is also described in *OS/390 SecureWay Communications Server: IP Diagnosis*.

Note: The feature HTCP38X is required for GDDMXD.

Overview of GDDMXD/MVS

GDDMXD/MVS is an interface that allows graphics from the IBM Graphical Data Display Manager/MVS to be displayed on workstations that support the X Window System. When GDDMXD/MVS is installed and activated, the data stream created by the GDDM application is translated to the X Window System protocol and transmitted by TCP/IP to the X Window System server for the display. If GDDMXD/MVS is installed and not activated, or has been made inactive, the GDDM application transmits data to its supported display device as if GDDMXD/MVS were not present.

Keyboard and Character Set Mappings

The following member names contain the described character sets:

Member Name	Description
GDXALTCS	A member of *hlq*.SEZAINST that contains a second character set (the 3270 alternate character sets).
GDXAPLCS	A member in the *hlq*.SEZAINST data set that contains sample keyboard mapping for APL2®.
KEYCODE	A member of *hlq*.SEZALINK that displays key codes. You can edit this data set to change keyboard mappings.

Executable Code

The following member names contain the described executable code or sample:

Member Name	Description
GDXLIOX0	A member of *hlq*.SEZALINK that contains executable code.
XDEFAULT	A member of *hlq*.SEZAINST that contains the sample XDEFAULT member for GDDMXD/MVS.

GDDM Application Limitations

When GDDMXD/MVS is inactive, there are no GDDM application restrictions. The following types of functions are not supported by GDDMXD/MVS:

- Multiple instances of the GDDM application
- Opening multiple display devices at one time
- Operator windows

GDDM Display Limitations

GDDMXD/MVS appears as an IBM 3179G device model to the GDDM application. When the HostRast option is active, the device model is an IBM 3279. The IBM

3179 Model G Color Graphics Display Station is used regardless of the display device nickname presented by the application.

The following are characteristics of the GDDMXD/MVS IBM 3179G display station:

Blinking Character Attribute
Ignores the blinking character attribute.

Detectable Fields
Ignores the detectable fields.

Character Display
Characters with an EBCDIC value of less than hex 40 are displayed as blanks.

Attached Graphics Cursor
When the graphics cursor is attached, the X Window System pointer device cursor for the GDDMXD window changes to a crosshair pattern and moves as the pointer device is moved.

Detached Graphics Cursor
When the graphics cursor is detached by the data stream or is not attached, the X Window System pointer device cursor for the GDDMXD window changes to an open arrow when the keyboard is unlocked, or changes to an X shape when the keyboard is locked.

Alphanumeric Cursor
When the graphics cursor is not enabled, the alphanumeric cursor can be repositioned by pointing the X Window System pointer device cursor for the GDDMXD window at the desired character location and pressing the pointing device button.

Pixel Spacing
When the HostRast option is not active, the vertical and horizontal pixel spacing of the actual display device that is obtained from the X Window System is supplied to the GDDM application. When the HostRast option is active, the pixel spacing of an IBM 3279 Color Display Station is supplied to the GDDM application.

Visual Appearance
For all programmed symbol and image data that is received from the GDDM application, each GDDM pixel is mapped to one X Window System display pixel, which causes a different appearance from the same data stream displayed on an IBM 3179G Color Graphics Display Station. This map process can also cause display differences in the placement of alphanumeric field data over the graphics display and in the appearance of the filled areas of the graphic display. When the HostRast option is active, aspect ratio distortion of the displayed graphics appears, unless the aspect ratio of the X Window System display is the same as the IBM 3279.

Color Mixing
GDDMXD/MVS supports only the overpaint foreground color mix mode. The initial color of the image area is black, and mixing with the actual background colors is not performed.

An exception is made for data passed by an image data order. In this exception, a combined foreground color mix mode is supported, if the series of begin image orders have exactly the same parameter values.

When the HostRast option is active, color mixing is performed by GDDM, and the preceding exception does not apply.

Default Vector Symbol Set
The Default Vector Symbol Set is not supported.

Using GDDMXD/MVS

Before GDDM data can be sent to an X Window System display, activate GDDMXD/MVS by invoking the GDDMXD CLIST. Make sure that you have already copied the GDDMXD CLIST to your system CLIST data set. If you have not, refer to the *OS/390 SecureWay Communications Server: IP Configuration* for instructions.

Note: If you do not want to run GDDM applications through the X Window System, do not enable GDDMXD/MVS.

To invoke the GDDMXD CLIST, use the GDDMXD command in the format described in “GDDMXD Command—Invoke the GDDMXD CLIST” on page 324.

GDDMXD Command—Invoke the GDDMXD CLIST

Purpose

Use the GDDMXD command to invoke the GDDMXD CLIST.

Format

Parameters

ON
Enables GDDMXD/MVS. GDDM output is sent to the X Window System display. The system responds with `GDDMXD/MVS active`.

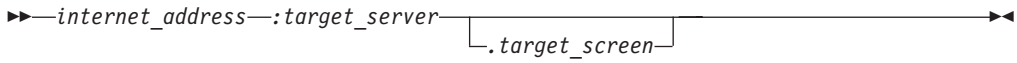
OFF
Disables GDDMXD/MVS. The system erases the data set that was created when GDDMXD/MVS was activated and responds with `GDDMXD/MVS inactive`.

Identifying the Target Display

Purpose

A TSO global variable is used by the X Window System to identify the IP address of the target display based on the contents of the data set *user_id*.XWINDOWS.DISPLAY.

Format



Parameters

internet_address
Specifies the IP address of the host machine on which the X Window System server is running.

:*target_server*
Specifies the number of the display server on the host machine.

.*target_screen*
Specifies the screen to be used on the *target_server*.

Examples

Examples of the contents of the *user_id*.XWINDOWS.DISPLAY data set.

- `charm.cambridg.ibm.com:0.0`
- `129.42.3.109:0.0`

GDDMXD Usage Notes

- When you use the TSO Session Manager with GDDMXD, enter a null line in the host session window after the GDDMXD graphics window disappears. This updates and activates the host session window.
- When you run PL/I GDDM applications, do not let the ISASIZE runtime option default to zero. This causes excessive allocation of storage below the 16 megabyte line, and causes a variety of storage allocation abends. Enter a runtime option for ISASIZE, such as ISASIZE(20K), to prevent storage allocation abends.
- Although GDDMXD provides its own device information to the GDDM application, normal GDDM device initialization occurs. A full-screen 3270 TSO session from a real or emulated 3270 terminal with 80 columns and 32 rows is required to invoke the GDDM application.

Resizing the GDDMXD Graphics Window

GDDMXD supports four graphics presentation space sizes. The size of the graphics presentation space used by GDDMXD is determined by the window width specified by the Geometry option in the *user_id.*X.DEFAULTS data set (see “User-Specified Options” on page 326 for more information). The size is also determined dynamically when you resize the GDDMXD graphics window. The relationship between the size of the graphics presentation space and the window width is shown in Table 31.

Table 31. Supported Graphics Presentation Space Sizes

Window Width (pixels)	GDDMXD Graphics Presentation Space (pixels)
< 650	480 horizontal by 352 vertical
>=650 to < 850	720 horizontal by 512 vertical
>= 850 to <= 1024	960 horizontal by 682 vertical
> 1024	1200 horizontal by 864 vertical

For graphics presentation space sizes other than the default size (720 pixels by 512 pixels), bit-mapped data, such as symbol sets and images, is expanded or compressed to meet the scaling requirements of the specified graphics presentation space.

You can expand bit-mapped data by duplicating rows and columns of the data. The resulting view can differ slightly from the default-size view. You can compress single-plane bit-mapped data by combining rows and columns of the data with a logical OR function. Because this might not yield acceptable results when a black on white image is viewed, the Compr option is provided to specify that a logical AND function be used to compress the data. See “Compr Option—Control the Technique Used to Compress Bit-Mapped Data” on page 330 for more information about using this option. You can compress multiplane bit-mapped data by eliminating certain rows and columns. Data compression produces a view that is different from the default-size view.

User-Specified Options

The user-specified options for GDDMXD/MVS are entries in a data set called *user_id.*X.DEFAULTS. The *user_id.*X.DEFAULTS data set is searched during initialization of GDDMXD/MVS.

Note: The values in the *user_id*.X.DEFAULTS data set are case-sensitive and must be entered as shown.

The options listed in Table 32 are supported by GDDMXD/MVS.

Table 32. GDDMXD/MVS Options

Option	Description	Page
ANFontn	Specify the X Window System font used for characters in the alphanumeric presentation space	328
CMap	Specify whether the default color map is loaded or bypassed	329
Compr	Control the technique used to compress bit-mapped data when a graphics window size of 480 by 352 pixels is specified	330
Enter	Overrides the default key mapping for Enter	331
GColornn	Specify a color name	332
Geometry	Specify the size and location of the initial GDDMXD graphics presentation space	333
GMCPnn	Override GDDM multicolor patterns with workstation color names	334
HostRast	Perform raster image processing at the System/370 host	335
NewLine	Override the default key mapping for NewLine	336
XSync	Request that the X Window System process one request at a time	337
ZWL	Tell GDDMXD/MVS to draw all lines using 0-width lines	338

ANFontn Option—Specify the X Window System Font Used for Characters in the Alphanumeric Presentation Space

Purpose: Use the ANFontn option to specify the X Window System font that GDDMXD should use to display characters in the alphanumeric presentation space of the GDDMXD window.

Format:

Parameters:

n Specifies presentation space size.

fontname
Specifies the name of the X Window System font.

Examples: The following are examples of ANFontn options.

```
gddmx*ANFont1: Rom8
gddmx*ANFont3: Rom14
```

Usage: Graphics mode 1 and 2 characters in the graphics presentation space are not affected by this option. The value of *n* is in the range 1–4 and defines the X Window System font for each of the four sizes of presentation space supported by GDDMXD. You can specify the ANFontn option for any, all, or none of the four values for *n*. The X Window System fonts specified should be fixed-space fonts that have characters that fit into the character box size required by each of the four presentation space sizes.

n	**Presentation Space**	**Character Box**	**Example Font**
1	480 x 352	6 x 11	Rom8
2	720 x 512	9 x 16	Rom11
3	960 x 682	12 x 21	Rom14
4	1200 x 864	15 x 27	Rom17

If you select a font that has characters that are larger than the character box size, the characters might overlap when displayed.

CMap Option— Specify Whether the Default Color Map is Loaded or Bypassed

Purpose: Use the CMap option to specify whether the default color map is loaded or bypassed.

Format:

Parameters:

Y Directs GDDMXD/MVS to load the default color map. This is the default.

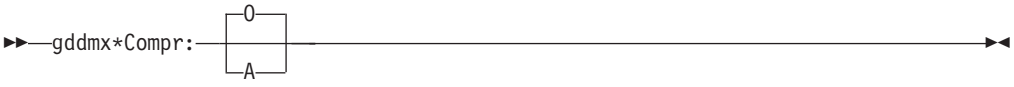
N Directs GDDMXD/MVS to bypass loading the default color map.

Usage: During initialization, GDDMXD/MVS issues the X Window System call, XInstallColormap, to load the default color map. If the CMap option is specified as N, the XInstallColormap call is not made. This option is for X Window System servers that load their own color map and do not want the clients to load any other color map.

Compr Option—Control the Technique Used to Compress Bit-Mapped Data

Purpose: Use the Compr option to control the technique used to compress bit-mapped data when a graphics window size of 480 by 352 pixels is specified.

Format:



Parameters:

A or a
Specifies that a logical AND function must be used when compressing bit-mapped data.

O or o
Specifies that a logical OR function must be used when compressing bit-mapped data. This is the default.

Enter Option—Override the Default Key Mapping for Enter

Purpose: The Enter option can be specified in the *user_id*.X.DEFAULTS data set to identify which X Window System Keysym is to be mapped to the Enter function. This option overrides the default mapping of the Keysym XK_Execute to the Enter function.

Format:

Parameters:

keysym_name
Specifies the X Window System Keysym representing the physical key. For standard Keysyms, the XK_ prefix is not included in specifying the option.

Examples: In the following example of the Enter option, the X Window System Keysym, XK_Return, is mapped to the Enter function.

```
gddmx*Enter:  Return
```

GColornn Option—Specify a Color Name

Purpose: GDDMXD/MVS provides a default mapping of GDDM colors to X Window System colors. Use the GColornn option to override a default color name or to specify a color if a default color name is not available by your X Window System server.

Format:

Parameters:

nn:
Specifies the GDDM color entry that is mapped.

c Specifies the X Window System color that is used as the GDDM color.

Usage: Table 33 lists the GDDM colors that GDDMXD/MVS maps to the X Window System.

Table 33. GColors

GColornn	GDDM Color	X Window System Color
GColor1	Blue	Blue
GColor2	Red	Red
GColor3	Magenta	Magenta
GColor4	Green	Green
GColor5	Turquoise	Turquoise
GColor6	Yellow	Yellow
GColor7	White	White
GColor8	Black	Black
GColor9	Dark Blue	Dark Slate Blue
GColor10	Orange	Orange
GColor11	Purple	Plum
GColor12	Dark Green	Dark Green
GColor13	Dark Turquoise	Dark Turquoise
GColor14	Mustard	Wheat
GColor15	Gray	Gray
GColor16	Brown	Brown

Examples: The following is an example of using a GColornn option to override a default color:

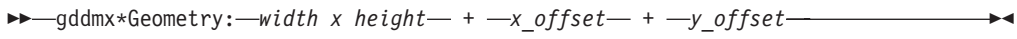
```
gddmx*GColor3:  Pink
```

In this example, specifying the `GColor3` entry in the *user_id*.X.DEFAULTS data set, maps the GDDM color, magenta, to the X Window System color, pink.

Geometry Option—Specify the Size and Location of the Initial GDDMXD Graphics Presentation Space

Purpose: Use the Geometry option to specify the size and location of the initial GDDMXD graphics presentation space.

Format:



Parameters:

width
Specifies the initial width of the GDDMXD graphics window. The *width* determines the initial size of the graphics presentation space.

height
Specifies the initial height of the GDDMXD graphics window.

x_offset
Specifies the location of the upper left corner of the window where *x_offset* is the horizontal offset from the upper left corner of the display.

y_offset
Specifies the location of the upper left corner of the window where *y_offset* is the vertical offset from the upper left corner of the display.

Examples: The following is an example of a Geometry option:

```
gddmx*Geometry:  750x600+20+20
```

GMCPnn Option—Override GDDM Multicolor Patterns with Workstation Color Names

Purpose: Use the GMCPnn option to override GDDM multicolor patterns with workstation color names.

Format:

Parameters:

nn:
Specifies the GDDM multicolor pattern.

c Specifies the color that is used with the defined GDDM multicolor pattern.

Examples: The following is an example of a GMCPnn option:

```
gddmx*GMCP126:  MediumBlue
```

In this example, the color medium blue is used when multicolor pattern 126 is specified by the GDDM application.

HostRast Option—Perform Raster Image Processing at the System/370 Host

Purpose: Use the HostRast option to perform raster image processing at the System/370 host.

Use the HostRast option when:
- Multiplane character symbol sets are required by the application
- GDDM color mixing is important to the application

The default device model for GDDMXD/MVS is an IBM 3179G with a mouse and the raster image processing is performed at the workstation.

Format:

Parameters:

Y Directs GDDMXD/MVS to use the IBM 3279 as a device model.

X Directs GDDMXD/MVS to use the IBM 3279 as a device model, and expand the pixel mapping to reduce aspect ratio distortion.

N Directs GDDMXD/MVS to use the IBM 3179G as a device model. This is the default.

Usage:
- The APL2 character set is not supported when the HostRast option is active.
- When the HostRast option is specified as Y, the GDDM application performs the raster image processing and transmits the picture as a series of characters whose pixel definitions have been transmitted to Programmed Symbol Sets. The picture is mapped exactly as an IBM 3279.
- If the ratio of horizontal to vertical pixel spacing is not the same as that of an IBM 3279, the aspect ratio can be distorted.

NewLine Option—Override the Default Key Mapping for NewLine

Purpose: The NewLine option can be specified in the *user_id*.X.DEFAULTS data set to identify which X Window System Keysym is to be mapped to the NewLine function. This option overrides the default mapping of the Keysym XK_Return to the NewLine function.

Format:

Parameters:

keysym_name
Specifies the X Window System Keysym representing the physical key. For standard Keysyms, the XK_ prefix is not included in specifying the option.

Examples: In the following example of the NewLine option, the X Window System Keysym, KP_Enter, is mapped to the NewLine function:

```
gddmx*NewLine:  KP_Enter
```

XSync Option—Request that the X Window System Process One Request at a Time

Purpose: The X Window System operates asynchronously. By the time an error has been detected, more requests could have been issued by the application.

Use the XSync option to request that the X Window System process one request at a time.

Format:

Parameters:

Y Directs GDDMXD/MVS to cause the X Window System to operate synchronously.

N Allows the X Window System to operate asynchronously. This is the default.

Usage: Be aware that system performance goes down when you use XSync=Y.

ZWL Option—Tell GDDMXD/MVS to Draw All Lines Using 0-Width Lines

Purpose: The X Window System supports a range of line widths. Because some X Window System servers draw wide lines at a slow rate, you can use the Zero Width Lines (ZWL) to tell GDDMXD/MVS to draw all lines using 0-width lines. The X Window System server uses the fastest process to draw the lines. The resulting line might not be exactly the same as if it had been drawn as a wide line.

Format:

Parameters:

Y Directs GDDMXD/MVS to use 0-width lines for all drawing.

N Directs GDDMXD/MVS not to use 0-width lines for all drawing. This is the default.

Keyboard Functions

The following sections detail different keyboard functions supported by GDDMXD/MVS.

GDDMXD/.MVS Keyboard Functions

When you enter input to the GDDM application by GDDMXD/MVS, use the following 3270 keyboard functions.

- All alphanumeric keys
- F1 – F24
  If F13 – F24 are not available, use Shift + F1 to Shift + F12
- Tab or Shift + Tab
- Directional arrows
- End key to erase to the end of the field
- Insert key and Delete key
- PA1 , PA2 , and PA3
- Enter key
- Newline key

Note: The Backspace key is treated as a cursor left key.

If you cannot locate these keys on your workstation, refer to your workstation X Window System documentation to determine the mapping of X Window System key symbol definitions to the physical keys.

GDDMXD/.MVS to X Window System Keyboard Functions

The following are the GDDMXD/MVS keyboard functions that translate to X Window System key symbol definitions. Key functions not listed are not supported.

GDDMXD/MVS Keyboard Function	X Window System Key Symbol
F1 – F12	XK_F1 – XK_F12
Tab	XK_Tab
Up	XK_Up
Down	XK_Down
Left	XK_Left
Right	XK_Right
End	XK_End
Insert	XK_Insert
Delete	XK_Delete
PA1	XK_Prior
PA2	XK_Next
PA3	XK_Home
Clear	XK_Pause

Enter XK_Execute

Newline XK_Return

APL2 character set toggle
XK_Backspace with state Mod1Mask

APL2 Character Set Keyboard

The APL2 character set is activated by simultaneously pressing the X Window System XK_Backspace key (usually the **Backspace** key) and the State Mod1Mask key (usually the **Alt** key). For example, if you use the IBM 101 Enhanced Keyboard, the APL2 character set is toggled on and off by pressing and holding the **Alt** key, and then pressing the **Backspace** key.

When the APL2 character set is active, the characters *APL* are displayed in the title bar of the GDDMXD/MVS window.

In the X Window System, a key code is assigned to each key on the keyboard. GDDMXD/MVS uses key codes in combination with modifier keys. For example, the **Shift** and **Alt** keys determine the data that should be passed back from GDDMXD/MVS to the X Window System application to identify the user's keystroke data.

GDDMXD looks for the data set *hlq*.SEZAINST(GDXALTCS) when it is initialized. Before using GDDMXD, copy the installed TCP/IP copy of *hlq*.SEZAINST(GDXALTCS) to *user_id*.GDXALTCS.PSS, or allocate the common installed copy of *hlq*.GDXALTCS.PSS to ddname GDXDACSP.

A default map for the APL2 character set is provided in GDDMXD/MVS, which corresponds to the IBM 101 Key Enhanced Keyboard. You can override this default map by creating a data set called GDXAPLCS.MAP to define the map for your workstation. When GDDMXD/MVS is initialized, the system searches for a data set called GDXAPLCS.MAP. If the GDXAPLCS.MAP data set exists, the data in the GDXAPLCS.MAP data set replaces the default mapping for all keys.

Setting Up *hlq*.GDXAPLCS.MAP

The following steps describe how to set up the GDXAPLCS.MAP data set.

1. Invoke the program KEYCODE from the TCP/IP load module library in *hlq*.SEZALINK to determine the key codes for the keyboard keys.

   When KEYCODE is executed from your workstation session to the host system, the key code is displayed for each key pressed at the workstation. Therefore, you can establish the association between a key and the character you want to generate.

   See "Appendix B. Mapping Values for the APL2 Character Set" on page 553, for more information about the mapping values that are defined in the GDXAPLCS.MAP data set.
2. Copy the *hlq*.GDXAPLCS.SAMPMAP installed with TCP/IP to *hlq*.GDXAPLCS.MAP.
3. Edit GDXAPLCS.MAP to establish the association between the key codes in the program KEYCODE and the character set and code values in Appendix B. Mapping Values for the APL2 Character Set.
4. GDDMXD looks for the data set *user_id*.GDXDAPLCS.MAP when it is initialized. If you want to use a different data set name, allocate the data set to ddname GDXDACSM.

Chapter 9. Executing Commands on a Remote Host

The Remote Execution Protocol (REXEC) and the Remote Shell Protocol (RSH) are remote execution clients that enable you to execute a command on a remote host and receive the results on the local host. You can execute either REXEC or RSH from the TSO command line or as a batch program. When executed as a batch program, the results are stored in a data set for later use.

To use REXEC, you must have a REXEC daemon running on the remote host. The REXEC client passes the user name, password, and command to the REXEC daemon. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

To use RSH, you must have a REXEC daemon running on the remote host. The RSH client passes the local user name, remote user name, and command to the RSH daemon. The remote user name may be in the form user/password when the RSH daemon is on an MVS host. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

This chapter describes how to use the REXEC and RSH clients.

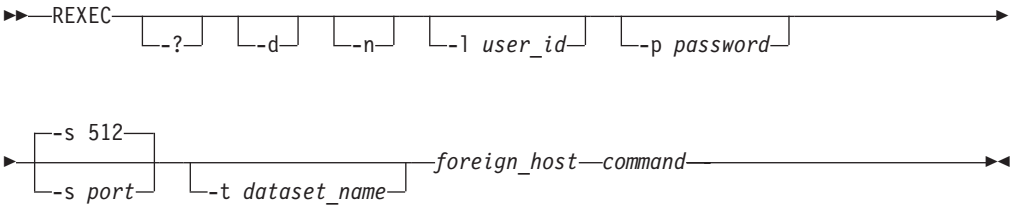
REXEC Command—Execute a Command on the Remote Host and Receive the Results on Your Local Host

Purpose

Use the REXEC command to execute a command on the remote host and receive the results on the local host.

Note: The TSO REXEC command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format



Note: The REXEC parameters -d, -l, -n, -p, -s, and -t are case-sensitive and must be entered in lowercase letters. The *user_id* and *password* parameters may be case-sensitive, depending on the operating system of the remote host.

Parameters

-? Displays the help message.

-d Activates debug tracing.

-n Prevents automatic logon, if the same foreign host name is defined in the NETRC data set.

-l *user_id*
Specifies the user ID on the foreign host.

-p *password*
Specifies the password for the user ID on the foreign host.

-s *port*
Specifies the TCP port number of the REXEC server on the foreign host. The default is port number 512.

-t *dataset_name*
Specifies the data set name of a translation table other than STANDARD. The data set name is *userid*.tran_table.TCPXLBIN. The default data set name is *userid*.STANDARD.TCPXLBIN. If this data set is not found, REXEC uses a compiled translation table.

foreign_host
Specifies the name or IP address of the foreign host to which you are sending the REXEC command. Specify the foreign host by its host name or IP address.

command
Specifies the command to be executed on the remote host.

Examples

Use the REXEC command without using NETRC.DATA:

```
READY  rexec -l user28 -p user28 -s 512 mvs1 lista

MVS TCP/IP REXEC V3R2
SYS1.HELP
GIM.SGIMCLS0
DSN230.DSNCLIST
USER.CLIST
BUILD.CLIST
SYS1.HRFCLST
USER28.RSHD5.JOB00160.D0000103.?
```

Usage

- If you omit the *user_id*, the *password*, or both when entering the REXEC command, the system prompts you to supply the parameters, if the foreign host is not specified with a user ID password in NETRC data set.
- When issuing a command to be executed on the remote host, do not place the command in quotes. Doing so may result in unexpected results.

  Note: There is no such restriction when using the OS/390 UNIX orexec command.
- A condition code of 12 will be set when an REXEC batch request encounters one of the following error conditions:
  - the client cannot connect to TCP/IP
  - the host name cannot be resolved
  - the translation table cannot be loaded

Using the NETRC Data Set

The NETRC data set provides you with an alternative to specifying the user_id and password as REXEC parameters. REXEC uses the following search order to find the NETRC data set to use:

1. NETRC DD statement
2. userid.NETRC.DATA
3. tso_prefix.NETRC
4. userid.NETRC

For information about using the NETRC data set in a batch file, see “Submitting REXEC and RSH Requests in Batch”.

If the user ID is specified on the -l parameter on the REXEC command, no NETRC data set are used. The user ID and password are case sensitive. If the user ID and password are not in the correct case, failures might occur when connecting to a non-mainframe REXEC server. The following is the format of the NETRC data set:

```
machine hostname login user_id password password
```

Note: You can omit your password in the NETRC data set. If you do, REXEC prompts you for your current password.

The following is an example of an NETRC data set:

```
machine mvs1 login user28 password abcdef
```

The following is a sample or the response that is displayed as a result of using the REXEC command and the NETRC data set.

```
 READY  rexec mvs1 lista
MVS TCP/IP REXEC V3R2
SYS1.HELP
GIM.SGIMCLS0
DSN230.DSNCLIST
USER.CLIST
BUILD.CLIST
SYS1.HRFCLST
USER28.RSHD5.JOB00161.D0000103.?
```

Submitting REXEC and RSH Requests in Batch

You usually run REXEC and RSH interactively by entering the command and then receiving the results at your terminal. However, you can also run REXEC and RSH as a batch job. To accomplish this, you must supply the necessary job control language (JCL) and submit it to the job entry subsystem (JES) using the TSO SUBMIT command.

The command format when submitted as a batch job is the same as the command format described in “REXEC Command—Execute a Command on the Remote Host and Receive the Results on Your Local Host” on page 342. You enter the command as a parameter on the EXEC statement. The results of the command executed on the remote host are stored on the local host according to how you define the SYSPRINT DD statement. The data set characteristics should be consistent with the output from the command you are executing at the remote host.

When you invoke the REXEC command, a check is made to see if a data set is allocated to INPUT If a data set is allocated, any input is read from that data set rather than from your terminal. Similarly, a check is made to see if data set is allocated to OUTPUT. If so, all REXEC output is written to that data set rather than to your terminal.

The REXECD server does not support output being returned to output DD statements other than SYSTSPRT or SYSPRINT.

When using the REXECD server, the procedure specified in the TSOPROC argument of the startup procedure must have the //SYSTSPRT DD statement appearing before any other output DD specifications in the procedure. For example, if the batch procedure specified was `TSOPROC=TESTJOB`, the following example would be the correct specification for the batch procedure for REXECD:

```
//TESTJOB  EXEC PGM=IKJEFT01,REGION=4M,DYNAM=30,REGION=4M
//STEPLIB   DD  DSN=A.LOADLIB,DISP=SHR
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD  SYSOUT=*
```

Adding a PARM argument to the EXEC JCL DD statement for commands to be submitted to batch might cause unpredictable output, characters to be lost, or output to be presented out of order.

If DDs are dynamically allocated, the order of output returned might be unpredictable. For example, the following TSO/E CLIST were invoked as the remote execution command, using the procedure TESTJOB:

```
PROC 0
TIME
ALLOC FI(OUT) SYSOUT(*)
OPENFILE OUT OUTPUT
SET &OUT = &STR(THIS; IS THE FIRST LINE)
PUTFILE OUT
SET &OUT = &STR(THIS; IS THE SECOND LINE)
PUTFILE OUT
TIME
SET &OUT = &STR(THIS; IS THE LAST LINE)
PUTFILE OUT
CLOSFILE OUT
FREE  FI(OUT)
EXIT
```

In this case, the output might be returned as:

```
THIS IS THE FIRST LINE
THIS IS THE SECOND LINE
THIS IS THE LAST LINE
output from time
output from time
```

In the following example, the TESTJOB procedure was modified to add the OUT DD statement:

```
//TESTJOB  EXEC PGM=IKJEFT01,REGION=4M,DYNAM=30,REGION=4M
//STEPLIB   DD  DSN=A.LOADLIB,DISP=SHR
//SYSTSPRT  DD  SYSOUT=*
//SYSPRINT  DD  SYSOUT=*
//OUT       DD  SYSOUT=*
```

The CLIST was also modified, as in the following example (notice that the `ALLOC FI(OUT)` and the `FREE FI(OUT)` have been removed):

```
PROC 0
TIME
OPENFILE OUT OUTPUT
SET &OUT = &STR(THIS; IS THE FIRST LINE)
PUTFILE OUT
SET &OUT = &STR(THIS; IS THE SECOND LINE)
PUTFILE OUT
TIME
SET &OUT = &STR(THIS; IS THE LAST LINE)
PUTFILE OUT
CLOSFILE OUT
EXIT
```

The output appears as in the following example:

```
output from time
output from time
THIS IS THE FIRST LINE
THIS IS THE SECOND LINE
THIS IS THE LAST LINE
```

Be aware that output being returned to the client from DDs other than SYSTSPRT or SYSPRINT might have characters truncated. This behavior has been observed only in the first line of new output files other than SYSTSPRT or SYSPRINT. If you use DDs other than SYSTSPRT or SYSPRINT, you might want to ensure that the first line of the output file contains a blank line, so that no data is lost.

Notes:

1. You can also use the NETRC data set name described in “Using the NETRC Data Set” on page 344 to specify the user ID and password. You can override the NETRC data set search order by specifying a NETRC DD statement in the batch job. The NETRC DD statement identifies the NETRC data set to be used. You must provide all REXEC command information by using the NETRC data set and the PARM keyword on the EXEC statement.
2. Submitting a long running command may cause the REXEC program to end abnormally with a 522 system abend code. This can be avoided by specifying TIME=1440 on the EXEC statement of the JCL you submit. Job step timing is suppressed, including the collection of SMF job time accounting information.
3. If the command to be executed on the remote host contains a slash (/), you must use a preceding slash (/) in the input stream of the NETRC data set or the PARM. For example, if the command to be executed is: `LS ./bin/temp/*` your input stream would be: `PARM='/-l userid -p pass_word host_name command'`
4. A condition code of 12 will be set when an REXEC batch request encounters one of the following error conditions:
   - the client cannot connect to TCP/IP
   - the host name cannot be resolved
   - the translation table cannot be loaded
5. A condition code of 12 will be set when an RSH batch request encounters an error condition in which the client cannot connect to TCP/IP or when the host name cannot be resolved.

The following example shows REXEC JCL Spooling Output to JES:

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//         PARM='-l userid -p password foreign_host command'
//SYSPRINT DD   SYSOUT=*
```

Note: The data set containing the JCL cannot have sequence numbers.

The following example shows the use of the *userid*.NETRC.DATA containing the user ID and password. The output is sent to a permanent data set for later use.

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//         PARM='foreign_host command'
//SYSPRINT DD   DSN=USERID.REXEC.SYSPRINT,DISP=(NEW,CATLG),
//         UNIT=3380,VOL=SER=MYVOL
```

Note: When running REXEC in batch, the user ID assigned to the job is used as the *user_id* in the NETRC data set.

The following example shows the use of the NETRC DD statement in batch. The NETRC DD statement can be used in batch to override the default userid.NETRC.DATA or userid.NETRC files.

```
//REXEC    JOB  USERID,MSGLEVEL=(1,1),NOTIFY=USERID
//STP1     EXEC PGM=REXEC,REGION=512K,
//         PARM='foreign_host command'
//SYSPRINT DD   SYSOUT=*
//NETRC    DD   DSN=TST.REXEC.NETRC,DISP=SHR
```

Note: The user ID and password are retrieved from TST.REXEC.NETRC instead of userid.NETRC.DATA or userid.NETRC.

RSH Command—Execute a Command on a Remote Host and Receive the Results on Your Local Host

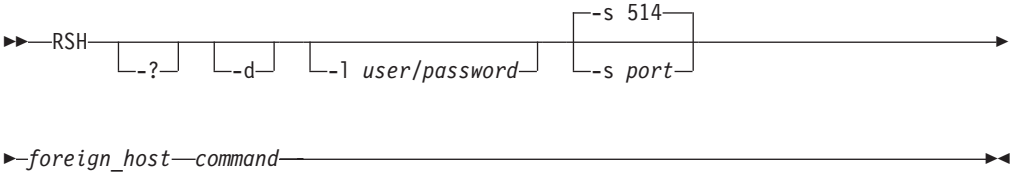
Purpose

The Remote Shell Protocol (RSH) is a remote execution client similar to REXEC that enables you to execute a command on a remote host and receive the results on your local host.

You can use the RSH command from TSO, or as a batch job.

Note: The TSO REXEC command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

Format



Note: The parameters for the RSH command could be case sensitive, depending on the remote host you are targeting.

Parameters

-? Lists the valid parameters for the RSH command.

-d Turns on debug tracing.

-l *user*/*password*
Specifies the remote user ID and password.

You must enter the slash (/) between the user ID and password if the target system is MVS. The character you must use can vary, depending on the target host. For VM hosts, you should use an at sign (@).

-s *port*
Specifies the server port. The default is 514.

foreign_host
Specifies the name of the remote host on which you are executing the command.

command
Specifies the command to be executed on the remote host.

Examples

- Execute a command on a remote host and receive the results on your local host:

```
 READY

rsh -l user28/user28 mvs1 lista

 SYS1.HELP
 BUILD.HELP
 GIM.SGIMCLS0
 ISR.V3R5M0.ISRCLIB
 DSN230.DSNCLIST
 DSN230.NEW.DSNCLIST
 DSN230.DSNAMACS
 USER.CLIST
 BUILD.CLIST
 SYS1.HRFCLST
 ISP.V3R5M0.ISPEXEC

 READY
```

- Use JCL to submit the RSH command as a batch job:

```
//USER28 JOB ,CARTER,MSGLEVEL=(1,1),NOTIFY=USER28
//RSH31  EXEC PGM=RSH,
//          REGION=800K,
//          PARM='/-d -l user28/user28 mvs1 lista'
//SYSPRINT DD  SYSOUT=*
//SYSTCPD  DD  DSN=USER28.TCPIP.DATA,DISP=SHR
//SYSIN    DD  DUMMY
```

Usage

- You must enter the required parameters on the command line. The RSH command does not prompt you for missing parameters, or enable you to use the NETRC.DATA set.
- A condition code of 12 will be set when an RSH batch request encounters an error condition in which the client cannot connect to TCP/IP or when the host name cannot be resolved.

RHOSTS.DATA Data Set

The *user_id*.RHOSTS.DATA data set provides you with an alternative to specifying RSH parameters *user_id* and *password* when you invoke the RSH command.

The *user_id*.RHOSTS.DATA data set contains host names and user IDs. Unlike NETRC.DATA, it is allocated and cataloged on the remote host.

The user ID specified in *user_id*.RHOSTS.DATA can be either the user ID you would otherwise specify as an RSH parameter or your logon ID on your local host.

If your user ID is the same at both the local and remote hosts, use this common ID to create RHOSTS.DATA. In this case, you do not need to include the *user_id/password* parameter on the RSH command, as shown in the following example:

```
rsh mvsone lista
```

If your user ID at the remote host is different from your user ID at the local host, use the user ID of the remote host to create RHOSTS.DATA. In this case, you can invoke the RSH command without the password, as follows:

```
rsh -l user28 mvsone lista
```

If you do not create the RHOSTS.DATA data set on the remote host, you must specify both the user ID and the password with the RSH command, as shown in the following example:

```
rsh -l user28/abcdef mvsone lista
```

The host names in the following examples are the official, fully qualified names of local hosts from which you want to run RSH. The user IDs are the logon IDs for those local hosts. Nicknames are not allowed.

```
local.host.name user_id
mvsthree.raleigh.ibm.com user30
mvsthree.raleigh.ibm.com user31
mvstfour.raleigh.ibm.com user30
```

Using Remote Execution Clients in an OS/390 UNIX Environment

OS/390 UNIX Remote Execution Protocol (OS/390 UNIX REXEC) is a remote execution client that you can use to execute a command on a remote host and receive the results on the local host.

You must have the OS/390 UNIX REXEC daemon (orexecd) running on the remote host to use OS/390 UNIX REXEC to run OS/390 UNIX commands. The OS/390 UNIX REXEC client passes the user name, password, and command to the OS/390 UNIX REXEC daemon. The daemon provides automatic logon and user authentication, depending on the parameters that you set.

You do not have to have the OS/390 UNIX REXEC deamon (orexecd) running on the remote host if you are not running OS/390 UNIX commands. The OS/390 UNIX REXEC client can use any REXEC daemon (not just OS/390 UNIX).

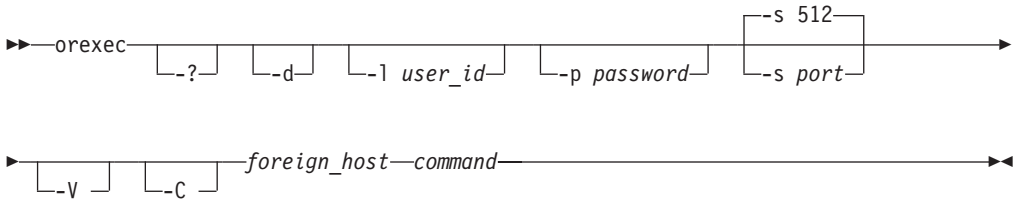
The OS/390 UNIX orexec/rexec Command—Execute a Command on the Remote Host

Purpose

Use the OS/390 UNIX orexec/rexec command to execute a command on the remote host and receive the results on the local host.

Note: rexec is a synonym for the orexec command in the OS/390 UNIX shell. rexec command syntax is the same as that for the orexec command.

Format



Note: Enter the orexec parameters -d, -l, -p, and -s in lowercase letters because they are case-sensitive. The *user_id* and *password* parameters may be case-sensitive, depending on the operating system of the remote host.

Parameters

-? Displays the help message.

-d Activates debug tracing.

-l *user_id*
Specifies the user ID on the foreign host.

-p *password*
Specifies the password for the user ID on the foreign host.

-s *port*
Specifies the TCP port number of the OS/390 UNIX REXECD server on the foreign host. The default is port number 512.

-V Writes the name and the Authorized Program Analysis Report (APAR) level to syslog.

-C Changes all messages to uppercase.

foreign_host
Specifies the name or IP address of the foreign host to which you are sending the orexec command. Specify the foreign host by its host name or IP address.

command
Specifies the command that is sent to the foreign host. The command is composed of one or more words. Coding is assigned after checking the prefixed parameters (-l, -p, -s) and assigning the remaining string as the command. The command you specify must not require a response from you to complete. orexec cannot interact with you after you enter data in the command format.

Examples

Use the orexec command to execute a command on a remote host:

```
orexec -l user28 -p user28 -s 512 mvs1 lista

SYS1.HELP
GIM.SGIMCLS0
DSN230.DSNCLIST
USER.CLIST
BUILD.CLIST
SYS1.HRFCLST
USER28.ORSHD5.JOB00160.D0000103.?
```

Usage

If you omit the *user_id*, the *password*, or both when entering the orexec command, the system prompts you to supply the parameters.

Chapter 10. Using the Network Database System

This chapter describes the Network Database System (NDB), that is used to access relational database systems in a TCP/IP environment.

The Network Database System enables:

- Workstation or mainframe users to issue Structured Query Language (SQL) statements interactively
- Database sharing between computers with different system architectures and operating systems

The NDB client command enables you to issue SQL statements that you can enter directly from the client machine to query or access data from a DB2 subsystem. You can use the NDBCLNT command from the following environments:

- DOS
- MVS
- OS/2
- AIX® on RS/6000®
- UNIX on Sun Microsystems
- VM

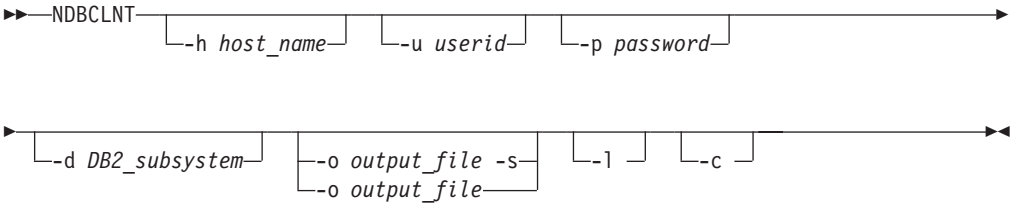
NDBCLNT Command—Issue SQL Statements to a DB2 Subsystem

Purpose

Use the NDBCLNT command to issue SQL statements to a DB2 subsystem.

Note: The SQL statement that you want to enter is not part of the NDBCLNT command. You enter your SQL statement after the NDBC prompt message is displayed.

Format



Parameters

-h *host_name*
Specifies the host on which the DB2 subsystem resides. You are prompted for a name if you do not enter one.

Note: For the NDBCLNT command, substituting the IP address for *host_name* is accepted only if doing so is supported by the remote procedure call (RPC) implementation of the TCP/IP product running on the client machine.

-u *userid*
Specifies the host user ID to be used to access the DB2 system. You are prompted for an ID if you do not enter one.

-p *password*
Specifies the password for the TSO *userid*. You are prompted for a password if you do not enter one.

Note: If you enter your password from the command line as part of the NDBCLNT invocation, the password is displayed. If you do not enter your password as part of the invocation, you are prompted for the password and the password is not echoed to the screen.

-d *DB2_subsystem*
Specifies the name of the DB2 subsystem to be accessed for this NDB session. You are prompted for a DB2 subsystem name if you do not enter one.

-o *output_file*
Specifies the file that stores the SQL statement result. Ensure that you use the correct naming convention for the system being used. This option is required if you specify the -s option.

-s Prevents output from being displayed on the terminal. If you do not specify the -o option, it is assumed, and you are prompted for the name of the output file.

-l Write listing of NDB session to output file, if one is specified. This includes the SQL statement and the outcome of that SQL statement. If no output file is specified, this option is ignored.

-c Specifies that the output from an SQL statement should be sent as one continuous stream. The default is to have only one buffer of data at a time displayed (enter the word **continue** to get the next buffer of data).

Examples

```
ndbclnt -h mvs1 -d db2a -u user1 -p user1

Port server has allocated NDB Prog: 536870945.
NDBC: create table empinfo (empno int,name char(15),salary dec(8,2),hiredate date)
SQL statement completed successfully

NDBC: insert into empinfo values (10001, 'Andersen', 23456.78,'01/02/1983')
1  rows were affected by this SQL statement

NDBC: insert into empinfo values (10204,'Jones',25890.21,'02/01/1983')
1  rows were affected by this SQL statement

NDBC: insert into empinfo values (10997,'Smith',27654.32,CURRENT DATE)
1  rows were affected by this SQL statement

NDBC: commit
SQL statement completed successfully

NDBC: select * from empinfo
      10001 Andersen          23456.78 1983-01-02
      10204 Jones             25890.21 1983-02-01
      10997 Smith             27654.32 1994-07-25

 NDBC: update empinfo set salary = salary * 1.50
3  rows were affected by this SQL statement

NDBC: select * from empinfo
      10001 Andersen          35185.17 1983-01-02
      10204 Jones             38835.31 1983-02-01
      10997 Smith             41481.48 1994-07-25

NDBC: rollback
SQL statement completed successfully

NDBC: select * from empinfo
      10001 Andersen          23456.78 1983-01-02
      10204 Jones             25890.21 1983-02-01
      10997 Smith             27654.32 1994-07-25

NDBC: drop table empinfo
SQL statement completed successfully

 NDBC: end
```

Usage

- If you specify more than once any of the NDB invocation options that require a parameter (for example, -h, -u, -p, -d, -o), the last occurrence is the one used by the NDB client code.
- The NDBCLNT command format and parameter descriptions are displayed when you specify any of the following:
  - No parameters
  - No valid parameters
  - Invalid parameters
- After entering the NDBCLNT command, you are prompted by the `NDBC:` prompt for either an SQL statement or an NDB command.

- The reply buffer used by the NDB clients and servers is 8192 bytes. The size of the reply buffer can be reset by changing the value of the definition `NDB_REPLY_LENGTH` at the beginning of the NDBCLT.C program of the sample client code, and then recompiling the NDB client.

SQL Statements

NDB accepts SQL statements up to 2048 characters long. NDB supports all SQL statements that DB2 allows to be interactively or dynamically prepared along with the SELECT statement.

If you are using NDBCLNT with an operating system that supports redirected input from a file, you must ensure that the last statement in the file is the NDB command END. Refer to the user's guide or online help facility for your workstation operating system for more information about using redirected input.

NDB Commands

The following list shows the NDB commands:

Command	Description
continue	Requests additional buffers of data when the SQL query returns more than one buffer of data.
end	Ends the NDB client session. END causes an implicit SQL commit work to take place. All changes made since the last commit or rollback (or since the beginning of the session, if no commits or rollbacks were issued) become permanent.
eexit	Emergency use only. Certain errors can cause the NDB client to be unable to continue processing NDB commands or SQL statements, including the NDB END command. These error conditions result in the same or other error conditions continually being reported by the NDB client. If this happens, issue the NDB emergency exit command (EEXIT) to immediately exit. Note that using (EEXIT) does not allow the DB2 thread to terminate. This will cause the database resources (for example, locks and buffers) to continue to be held. The NDB server and the port that the NDB client is currently using will also continue to be held until the address space containing this NDB server has been cancelled and restarted.

Return Codes for NDB

The valid NDB return codes and messages are:

Code	Message
1	Problem with RPC. Look at the error messages displayed before this one. Refer to the Remote Procedure Call (RPC) messages and codes information for the client platform from which you are running the NDB client.
-20000	RPC error. Unable to call the server. The NDB sample client code detected that the NDB client was unable to successfully send the SQL statement or NDB command to the NDB server and to receive a response. NDB uses RPC to handle requests between NDB clients and servers. Look at the error messages and RPC information that was displayed before this error. Refer to the Remote Procedure Call (RPC) messages and codes information for the client platform from which you are running the NDB client.

Code	Message
-20100	Invalid NDBC control block. Reason: <string>

The NDB server detected that the NDBC control block, used for communication between an NDB client and its assigned NDB Server, contained an incorrect value.

Listed below are the possible reasons why the error occurred:

- NDB version numbers do not match.

  The version number in the client code must match the version number in the server code. The NDB sample client code sets a field in the control block NDBC called `ndbver` to indicate the current version of the NDB client. The NDB server also contains a version number in its code.

  For TCP/IP for MVS 3.1, the version value is 2. For all previous releases of TCP/IP for MVS, the version value is 1. Since the version value for NDB changes when the NDBC control block is changed, one level (version, release and modification) of NDB for TCP/IP for MVS may not be compatible with another.
- NDB release numbers do not match.

  The release number in the client code must match the release number in the server code. The NDB sample client code sets a field in the control block NDBC called `ndbrel` to indicate the current release of the NDB client. The NDB server also contains a release number in its code.

  For TCP/IP for MVS 3.1 and all previous releases, the release value is 1. Since the version value for NDB changes when the NDBC control block is changed, one level (version, release and modification) of NDB for TCP/IP for MVS may remain compatible with another, but there are differences in the control block field values that affect NDB server processing.
- Eyecatcher not for NDBC control block.

  The NDB sample client code sets a field in the control block NDBC called `ndbcb` to indicate the eyecatcher of the main control block being used by the NDB client. The NDB server looks for this eyecatcher before processing information contained in the control block.

  For TCP/IP for MVS 3.1 and all previous releases, the control block eyecatcher is `NDBC`.
- NDB Server name is unrecognized.

  The NDB sample client code sets a field in the control block NDBC called `ndbsname` to indicate the NDB server name by which the NDB client code knows the NDB server. The NDB server did not find a recognized name in the `ndbsname` field of the NDBC control block. For TCP/IP for MVS 3.1 and all previous releases, the NDB server name is `netdbsrv`.

Fix the NDBC control block field that is causing the error and rebuild the NDB client.

-20101 You (or PUBLIC) do not have authority to run DB2 plan EZAND320.

For NDB to access data stored in DB2, the DBUITIL2 application program that acts as interface code, was stored in DB2. DBUTIL2 is in the form of a DB2 plan name EZAND320. To use NDB (which uses the EZAND320), you must be granted execute authority for EZAND320 by your database administrator. This authority can be granted to your specific user ID or to everyone by granting it to PUBLIC.

-20102 DB2 Subsystem is unavailable. DB2 not active at OPEN DB.

The NDB server detected that the DB2 subsystem specified at NDB server startup was not running at the time the NDB server tried to establish a connection with DB2. Contact your system programmer.

Code	Message
-20103	Error occurred while using DB2 CAF. An error occurred when the NDB server used the DB2 Call Attachment Facility (CAF) to establish a connection with DB2. Contact your system programmer.
-20104	Error occurred while trying to allocate storage. An error occurred when the NDB server allocated storage for a data structure used by DB2 or for an internal buffer. Contact your system programmer.
-20105	Single row of query data is larger than reply buffer. The length of one row of data to be returned from the current query was greater than the length of the NDB reply buffer (size of which is specified in the NDBC field NDBRPDLN). You can take either of the following actions: • Retrieve fewer columns per query either by issuing multiple queries or by making the query more restrictive in the number of columns requested • Increase the size of the NDB reply buffer by changing the `define` compiler directive for NDB_REPLY_LENGTH in the program NDBCLT.C and rebuilding the NDB client
-20108	Host userid or password not valid The NDB server has detected that the host user ID or host password did not pass the security check on the host system. Enter a valid user ID and password combination.
-20999	Should never occur error has occurred. During the execution of NDB server code, a programming error occurred. Contact your system programmer.
+xxx	Positive return codes, other than +25, are informational and warning codes from DB2.
-xxx	Negative return codes between -001 and -999 or that are less than or equal to -30000 are from DB2.

Refer to *OS/390 SecureWay Communications Server: IP and SNA Codes* for more information about TCP/IP return codes. Refer to *IBM Database 2: Messages and Codes* for more information about DB2 return codes.

Format of Output Displayed on the Client

The output from an SQL query (SELECT) is presented in row and column format. The NDB Server distinguishes between a null value that is retrieved from a table in DB2 and a null value that is the result of an arithmetic error in the SQL SELECT statement (for example, division by zero). A null data value is represented by dashes (------), while a null value due to an arithmetic error is represented by slashes (//////).

For more information on null values, refer to *IBM DATABASE2: Application Programming and SQL Guide*.

NDB supports the following SQL data types:

Data Type	Description
DECIMAL	All formats supported by DB2
FLOAT	Single and double precision
INTEGER	Long, or fullword, integer
SMALLINT	Short, or halfword, integer

Data Type	Description
CHAR	Fixed length, up to 254 characters
VARCHAR	Variable length, up to 254 characters
VARCHAR	Long variable length, over 254 characters
DATE	All formats supported by DB2
TIME	All formats supported by DB2
TIMESTAMP	All formats supported by DB2

For unsupported SQL data types, values are replaced with asterisks (******) unless the value is null. If the value is null, it is represented as previously described. Refer to the *IBM DATABASE 2: SQL Reference* for information about SQL data types.

Part 2. System Administrator Applications

Chapter 11. Monitoring the TCP/IP Network

This chapter describes how to use the following TCP/IP commands to obtain information from the network.

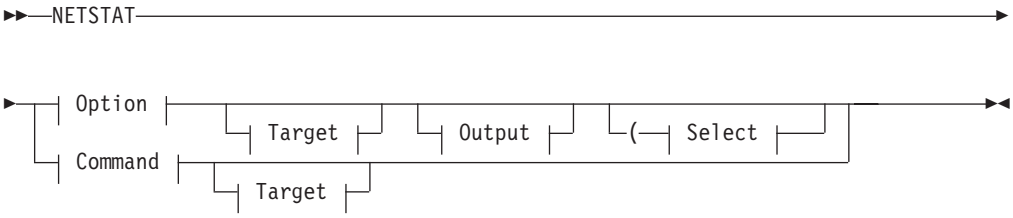
- The TSO NETSTAT and OS/390 UNIX onetstat commands provide information about the status of the local host, including information about TCP/IP connections, network clients, gateways, and devices. TSO NETSTAT and OS/390 UNIX onetstat also drop connections for users who have the MVS.VARY.TCPIP.DROP statement defined in their RACF profile.
- The TSO PING and OS/390 UNIX oping commands determine the accessibility of a foreign node.
- The TSO RPCINFO and OS/390 UNIX orpcinfo commands display the servers that are registered and operational with any Portmapper on your network.
- The TSO TRACERTE and OS/390 UNIX otracert commands let you debug network problems.

The TSO NETSTAT Command—Display Local Host Information

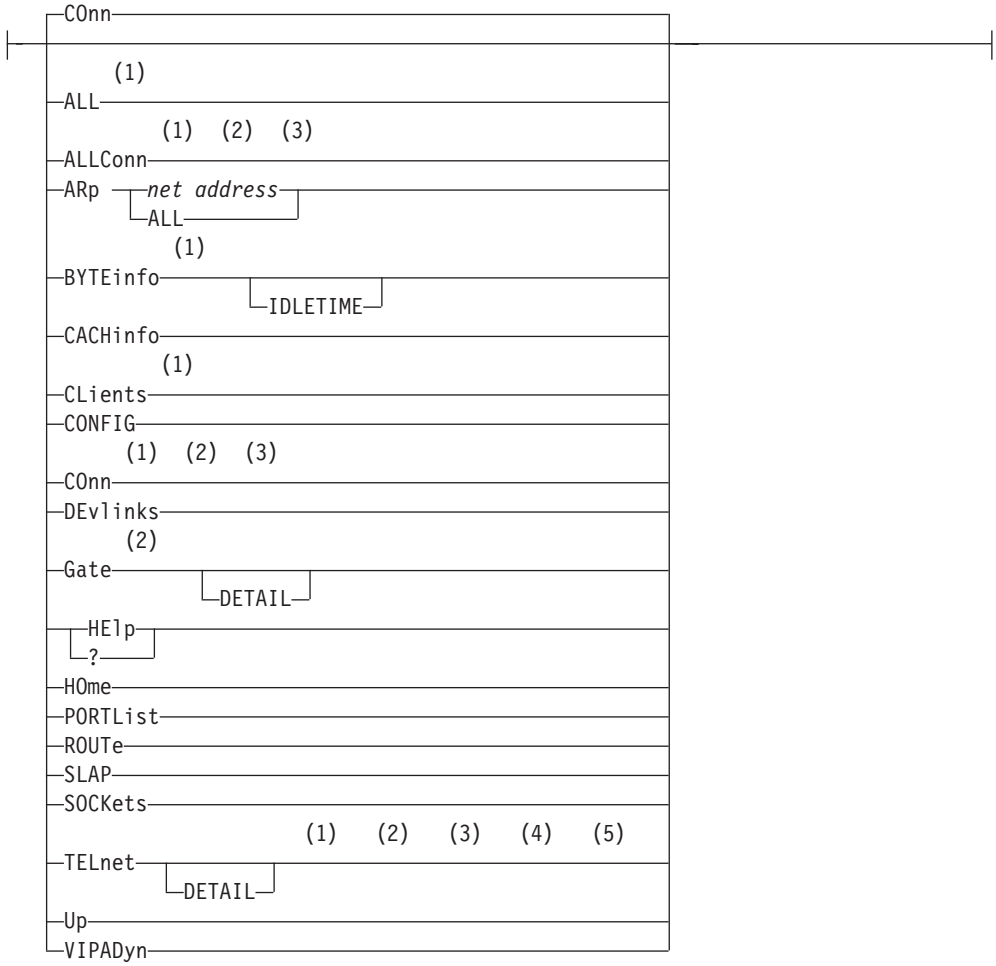
Purpose

Use the TSO NETSTAT command to display the network status of the local host.

Format



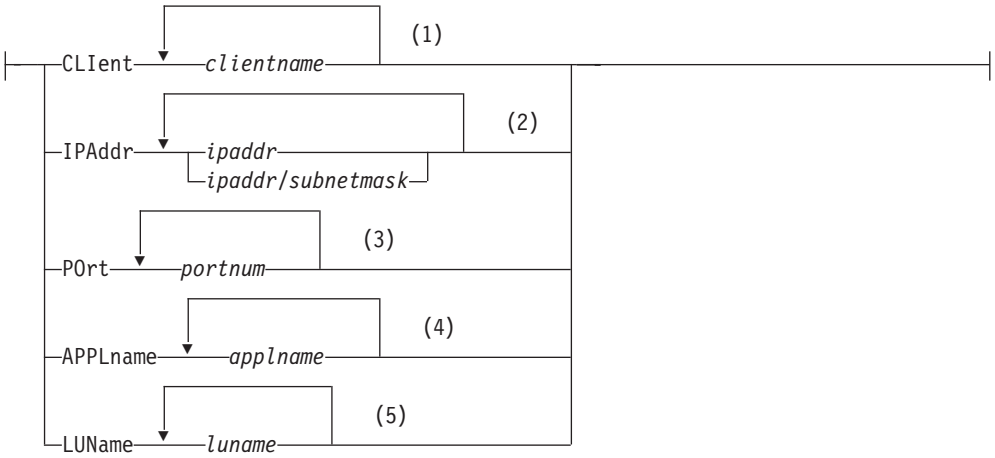
Option:



Target:

Output Option:

Select String:



Command:

Notes:

1 The CLIent select string is valid with ALL, ALLConn, BYTEinfo, COnn, CLients, and TELnet.

2 The IPAddr select string is valid only with ALLConn, COnn, Gate, and TELnet.

3 The POrt select string is valid only with ALLConn, COnn, and TELnet.

4 The APPLname select string is valid only with TELnet.

5 The LUName select string is valid only with TELnet.

Note: The minimum abbreviation for each parameter is shown in uppercase letters.

Parameters

ALL
Provides detailed information about TCP/IP connections. This option is useful

for debugging the TCP/IP address space. For more information about maintaining the TCP/IP address space, see *OS/390 SecureWay Communications Server: IP Configuration*.

If a UDP socket is being used for outgoing multicast data, this parameter displays the time-to-live value, indicates whether the datagrams are also sent to loopback, and displays the IP address of the link on which the datagrams are being sent. If a UDP socket is being used for incoming multicast data, this parameter lists the multicast groups by way of their IP addresses (up to a maximum of 20) for which data is being received and the IP address of the associated link.

ALLConn
Specifies that all information for all connections, including recently closed connections, is displayed.

ARp
Queries the ARP cache information.

net address
Queries the ARP cache for a given address.

ALL Queries all ARP cache entries.

Note: The ARP counter data will not reflect activity for any devices which support ARP offload. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* and the *OS/390 SecureWay Communications Server: SNA Network Implementation Guide* for more information.

BYTEinfo
Displays the byte-count information about each connection. This option is most useful when used in conjunction with the REPORT or STACK options.

The following information is given for each connection:
- Client ID
- Bytes sent on the connection
- Bytes received on the connection
- Local port
- Foreign socket
- State

IDLETIME
Displays the preceding information plus the idle time for each connection.

The idle time is displayed in the following format:

```
hours:minutes:seconds
```

CACHinfo
Displays information about Fast Response Cache Accelerator (Cache Accelerator) statistics. Statistics are displayed for each listening socket configured for Cache Accelerator support.

For each listening socket configured for Cache Accelerator support, the following information is displayed:

Client The user name of the application that bound the listening socket.

Socket The local IP address and port pair to that the listening socket is bound.

MaxCacheSize
The maximum number of 4K pages that may be used for storing cache objects by the Cache Accelerator for the given socket.

CurrCacheSize
The number of 4K pages currently being used for storing cache objects by the Cache Accelerator.

MaxNumObjects
The maximum number of cache objects that may be stored by the Cache Accelerator.

CurrNumObjects
The current number of cache objects stored by the Cache Accelerator.

NumConns The number of connections established through a listening socket which has been configured with Cache Accelerator support.

ConnsProcessed
The number of connections that have successfully completed an in-kernel transaction, resulting in a response being transmitted to the client. This counter is incremented at most once per connection.

Note: It is possible for a single connection to be processed by the Cache Accelerator for some cache entries and then deferred to the application for additional processing. If this occurs, the connection will be included in both ConnsProcessed and ConnsDeferred.

ConnsDeferred
The number of connections that require user-space application processing.

Note: This counter is not incremented due to the connection timeout expiration, even if the action taken is to defer the connection.

ConnsTimedOut
The number of times the connection timeout timer has expired.

RequestsProcessed
The number of connection requests that were at least partially processed by the Cache Accelerator.

Note: It is possible for a single connection to be processed by the Cache Accelerator for some cache objects and then deferred to the application for additional processing. If this occurs, the connection is included in both RequestsProcessed and RequestsDeferred.

IncompleteRequests
The number of times a request is received from the client where additional data is required to process the request. This counter may be incremented multiple times for a single connection.

NumCacheHits
The number of cache objects that were successfully located and transmitted to clients.

NumCacheMisses
The number of cache objects that were not successfully located and transmitted to clients.

NumUnprodCacheHits
The number of cache entries succesfully found within the cache but not transmitted to the client.

CLients
Provides the following information about each client:

- Client's authorization
- Elapsed time since the client was last used

CONFIG
Provides the following TCP/IP configuration data:

- Configured IP information
- Configured TCP information
- Configured UDP information
- SMF parameters
- GLOBALCONFIG Profile information

  Note: The TCPIPSTATS field shown under the SMF PARAMETERS section of the NETSTAT CONFIG output reflects the value of TcpIpStatistics or NoTcpIpStatistics specified on the SMFCONFIG statement in the TCP/IP Profile or Obeyfile. The TCPIPSTATS field shown under the GLOBAL CONFIGURATION section of the NETSTAT CONFIG output reflects the value from the ASSORTEDPARMS or GLOBALCONFIG statements in the TCP/IP Profile or Obeyfile.
- Data trace settings
- Dynamic VIPA information

COnn
Provides the following information about each active TCP/IP connection. An active connection is a connection that is not in the *closed* or *time-wait* state.

- Client name
- Client ID
- Local IP address
- Foreign IP address
- Connection state

COnn is the default parameter.

A connection progresses through a series of states during its lifetime. The following are the possible states for a TCP connection.

State	Description
LISTEN	Waiting for a connection request from any remote TCP and port
SYN-SENT	Waiting for a matching connection request after having sent a connection request
SYN-RECEIVED	Waiting for a confirming connection request acknowledgment after having both received and sent a connection request

State	Description
ESTABLISHED	Represents an open connection; this is the normal state for the data transfer phase of the connection
FIN-WAIT-1	Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request
FIN-WAIT-2	Waiting for a connection termination request from the remote TCP
CLOSE-WAIT	Waiting for a connection termination request from the local user
CLOSING	Waiting for a connection termination request acknowledgment from the remote TCP
LAST-ACK	Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP

For more information about the TCP connection states, refer to RFC793.

DEvlinks

Displays the following information about devices and defined links in the TCP/IP address space:

- Device name
- Device type
- Device number
- Link name
- Link type
- Status of the link

  The following list describes the possible link statuses:

Status	Description
Starting	A START of the device has been issued by the operator, and TCP/IP has been sent an Activation request to the Data Link Control (DLC) layer.
Enabling	DLC has acknowledged the Activation request from TCP/IP, and TCP/IP has requested DLC to allow TCP/IP to receive inbound connections for the device.
Sent SETUP	DLC has acknowledged the Activation request from TCP/IP, and TCP/IP has requested DLC to perform the initial I/O sequence with the device.
Connecting	DLC has accepted the Initial I/O Sequence request.
Connecting2	The control connection for a Common Link Access to Workstation protocol (CLAW) device has been established, and the second connection (on which IP traffic is carried) is being established.
Negotiating	The initial I/O sequence with the device is complete, and TCP/IP is performing additional link-layer initialization.
Ready	The initialization sequence with the device is complete. The device is now ready.
Sent CLEAR	The operator has issued a STOP of the device, and TCP/IP has sent a Deactivation request to DLC.
Deactivated	DLC has performed the first stage of an orderly device deactivation.
Not active	The device is not active. (The device has never been started, or has been stopped after having been started.)

- Net number

  Displays the link adapter number, if applicable.
- Queue size

  This field is significant only for links on LCS and ATM devices.
- ARP MAC addresses

This field is significant only for token-ring links.

- SR bridging capability

  This field is significant only for token-ring links.

- Multicast specific

  This field is significant only for multicast capable devices.

  If a link is being used to receive multicast data, then all the multicast groups, and the counts of receivers for each multicast group, are displayed. There is no limit to the number of multicast groups for which a link can receive data.

  – Multicast capability

    This field is always Yes for the following devices: CDLC, CLAW, CTC, and MPCPTP.

    For LCS and MPCIPA devices, the multicast capability is only known after the device is in the Ready state. If the device is not yet Ready, the multicast capability will be Unknown.

- Broadcast capability

  This field is significant only for links on LCS devices.

- Broadcast type

  This field is significant only for token-ring links.

- Number of bytes received
- Number of bytes transmitted
- BSD parameters
- Packet trace settings
- ATM specific information

  This field is significant only for ATM devices and links.

  Notes:

  1. No link-related information, packet trace settings, or BSD parameters are displayed for a device that has no link defined.
  2. The packet trace setting is displayed only when it is defined and set to ON.
  3. ATM specific information is displayed only for ATM devices that have links defined.
  4. The LOOPBACK device and link are displayed.

Gate

Provides the following information about each gateway:

- Address of the network
- First hop address
- Link name used by the first hop

  Note: Only the first 8 characters of the link name are displayed by this command. Issue the NETSTAT ROUTE command to see more than 8 characters of the link name.

- Packet size used by the first hop
- Subnet mask and subnet value

DETAIL

Displays the preceding information plus the metric associated with the cost of the use for the link, and displays the following MVS specific configured parameters for each gateway:

- Maximum retransmit time
- Minimum retransmit time
- Round trip gain
- Variance gain

- Variance multiplier

Note: The static routes associated with deleted interfaces in the routing table no longer appear in the reports generated with the NETSTAT Gate command. For details, refer to *OS/390 SecureWay Communications Server: IP Diagnosis*.

HElp or ?
Provides help information for the NETSTAT parameters.

HOme
Displays the HOME list. The IP address, link name, and primary interface flag are displayed for each entry in the list.

For more information about the home list, refer to *OS/390 SecureWay Communications Server: IP Configuration*.

PORTList
Displays the port reservation list:
- Port number
- Protocol
- Username
- Flags

  A Autolog

  D DelayAcks

  O Optimize MSS

  R Port is reserved by range

  S Share port
- Range

  This field is significant only for port entry reserved by range (flag R in the Flags field).

ROUTe
Displays routing information, such as:

Destination
The address of a destination host or network

Gateway
The gateway used in forwarding packets

Flags The state of the route:
U The route is up.
H The route is to a host rather than to a network.
G The route is a gateway.
D The route was created dynamically by a redirect.

Reference count
The current number of active users for the route

Interface
The link name for the route

Note: The static routes associated with deleted interfaces in the routing table no longer appear in the reports generated with the NETSTAT ROUTe command. For details, refer to *OS/390 SecureWay Communications Server: IP Diagnosis*.

SLAP
Displays all of the service policy definitions (policy rules, policy profiles, and service classes) and policy statistics data.

SOCKets
Displays information about each client using the socket interface. When you specify the NETSTAT SOCKets command, information about the client using the socket interface is displayed along with information about the sockets and associated connections owned by the client.

The following is the information displayed after invoking the SOCKets parameter:

Name The client address space name.

Subtask
The subtask identifier.

The subtask identifier is combined with the address space name to produce a unique identifier for the client.

- For socket programs written in the C language, the EBCDIC hexadecimal representation of an address within the program is used as the subtask identifier.

Type Displays one of the following socket types:

- `Stream` for stream (TCP) sockets
- `Dgram` for the stream (UDP) sockets

Bound to
Indicates the address and port to which the socket is bound. The output is in the format `internet address..bound port` where `internet address` is the address to which the socket is bound and `bound port` is the port number to which the socket is bound. Unbound TCP and UDP sockets are not displayed by NETSTAT CONN.

Connected to
Displays the address and port to which the socket is connected.

State Displays the TCP connection state for TCP sockets or *UDP* for UDP sockets.

Conn Displays the client identifier, which is a unique number assigned by TCP/UDP stack to uniquely identify a socket entity.

TELnet
Displays the status of the internal Telnet server.

DETAIL
Displays the logmode and Telnet protocol in use by each connection. If an application user ID was entered on the solicitor panel, it is displayed in the *TnUserId* field. Otherwise, the *TnUserId* field is blank.

Up
Provides the date and time that TCP/IP was started.

VIPADyn
Displays the current dynamic VIPA information for a local host.

TCp *tcpname*
Displays detailed information about the specified TCP/IP address space. You can use TCp *tcpname* with any other NETSTAT parameter to get information

about the specified TCP/IP address space; however, this parameter works only for TCP/IP address spaces having the same version as NETSTAT.

The *tcpname* is an 8-byte procedure name that is used to start the TCP/IP. When the 'S member.identifier' method of starting TCP/IP is used, the value specified for *identifier* must be used as *tcpname*.

REPort

Causes the output to be stored in a MVS data set. If there is no additional parameter specified, the output is stored in a data set named *tsoprefix*.NETSTAT.*option*. If NOPREFIX is set in the TSO user profile, then the data set name is NETSTAT.*option*. The data set is created and cataloged if it does not already exist. If the data set already exists, the output from the requested option replaces any existing data. The name of the data set depends on whether either of the following additional parameters were specified:

. DSN dsnname
Specifies the data set name in which the output will be stored. The dsnname can be either a fully qualified name surrounded by single quotation marks (for example, 'abc.xyz') or an unqualified name (for example, abc). If an unqualified name is specified, then the unqualified name will be prefixed with the TSO prefix value.

. HLQ hlqname
Specifies the high level qualifier for the data set in which the output will be stored. The resulting data set name will be *hlqname*.NETSTAT.*option*.

	tsoprefix unavailable	tsoprefix available
Nothing specified	NETSTAT.option	tsoprefix.NETSTAT.option
HLQ specified	hlq.NETSTAT.option	hlq.NETSTAT.option
Unqualified DSN	dsnname	tsoprefix.dsnname
Fully-qualified DSN	dsnname	dsnname

STACk
Causes the response to be placed in the program stack when NETSTAT is issued from a CLIST or a REXX EXEC. Title lines are not placed in the stack.

select_string
Specifies a character string used to limit option responses

CLIent *clientname*
Provides the response of ALL, ALLConn, BYTEinfo, COnn, CLient, and TELnet on the specified client name.

IPAddr *ipaddr*

IPAddr *ipaddr/subnetmask*
Provides the response of ALLConn, COnn, Gate, and TELnet on the specified IP address *ipaddr* or *ipaddr/subnetmask*. If *subnetmask* is not specified, the default of 255.255.255.255 is used.

Notes:

1. For the Gate option, *ipaddr* is the destination IP address, not the destination network address.
2. When filtering Gate responses on a specified *ipaddr*, the DEFAULT and DEFAULTNET routes are always displayed, as they apply to all IP addresses.

3. For ALLConn, CONN, and TELnet options, *ipaddr* can be either the local or remote IP address.

POrt *portnum*
Provides the response of ALLConn, COnn and TELnet on the specified port number.

Note: The port number can be either a local or remote port.

APPLname *applname*
Provides the response of TELnet on the specified VTAM application name.

LUName *luname*
Provides the response of TELnet on the specified LU name.

The *select_string* can be a complete string or a partial string using wildcard characters. A wildcard character can be an asterisk (*), which matches a null string or any character or character string, at the same position. A wildcard character can be a ?, which matches any single character at the same position. For example, a string ”searchee” matches with ”*ar?he*”, but the string ”searhee” does not match with ”*ar?he*”.

You can enter up to six *select_string* values. For IPAddr, each selected value can be up to 15 characters long, and for CLIents, APPLname and LUName, each selected value can be up to eight characters long.

If you want to use the wildcard character on the IPADDR select_string, you must specify the select value in the *ipaddr* format. The wildcard character is not accepted for the *ipaddr/subnetmask* format of IPAddr values.

If you specified *select_string*, it must be the last parameter on the NETSTAT command line.

DRop *n*
Drops the TCP/IP connection specified by *n*. You can determine the connection number from the CONN column in the NETSTAT COnn or NETSTAT TELnet display. If you drop the server’s *passive open* connection, the server immediately reissues the open request.

You can use this parameter only if your RACF profile contains the MVS.VARY.TCPIP.DROP definition.

Usage

- The time displayed in the header for each option is local time. The time field displayed in options ALL, BYTEinfo, CLients, and UP is Greenwich mean time (GMT).

Examples

This section contains examples of the response that is displayed as a result of issuing the NETSTAT command with each parameter. To help you find the examples, following is a list of the NETSTAT parameters and the page number where you will find the example:

- ALL on page 375
- ALLConn on page 377
- ARp *net address* on page 377

NETSTAT ALL

The NETSTAT ALL command displays information about TCP/IP connections. The client TCPCLIE1 in the following example shows information about a TCP connection, and the client UDPCLIE1 shows information about a UDP connection.

```
 READY
netstat all

 MVS TCP/IP NETSTAT CS V2R8        TCPIP Name: TCPCS            08:59:29
 Client Name: TCPCS                     Client Id: 00000008
 Local Socket: 127.0.0.1..1025          Foreign Socket: 127.0.0.1..1026
   Last Touched:       11:53:11          State:              Establsh
   BytesIn:            0000000644        BytesOut:           0000000840
   SegmentsIn:         0000000044        SegmentsOut:        0000000040
   RcvNxt:             1111472027        SndNxt:             1111472543
   ClientRcvNxt:       1111472027        ClientSndNxt:       1111472543
   InitRcvSeqNum:      1111471382        InitSndSeqNum:      1111471702
   CongestionWindow:   0000410006        SlowStartThreshold: 0000016384
   IncomingWindowNum:  1111504755        OutgoingWindowNum:  1111505275
   SndWl1:             1111471987        SndWl2:             1111472543
   SndWnd:             0000032732        MaxSndWnd:          0000032768
   SndUna:             1111472543        rtt_seq:            1111472507
   MaximumSegmentSize: 0000065483        OptMaxSegmentSize:  0000065483
   BackoffCount:                         Precedence:         Routine
   Round-trip information:
     Smooth trip time: 69.000             SmoothTripVariance: 113.000
   ReXmt:              0000000000        ReXmtCount:         0000000000
   DupACKs:            0000000000
   SockOpt:            00                TcpTimer:           00
   TcpSig:             04                TcpSel:             20
   TcpDet:             30                TcpPol:             00
   TcpClusterConnFlag: 82
   ReceiveBufferSize:  0
 ----
 Client name: 00010003               Client id : 10003
 Local Socket: 0.0.0.0..2025         Foreign Socket: *..*
   Last Touched:        3:35:05
   Precedence:          Routine
   BytesIn:             0000400000      BytesOut:     0000400000
   DgramIn:             0000000100      DgramOut:     0000000100
   MaxSendLim:          0000065000      MaxRecvLim:   0000065000
   SockOpt:             00
   Multicast Specific:
     TimeToLive:        0000000001      LoopBack:     Yes
     OutgoingIpAddress: 9.67.113.27
 ----
 Client name: 00010003               Client id : 10003
 Local Socket: 0.0.0.0..2025         Foreign Socket: *..*
   Last Touched:        3:35:05
   Precedence:          Routine
   BytesIn:             0000400000      BytesOut:     0000400000
   DgramIn:             0000000100      DgramOut:     0000000100
   MaxSendLim:          0000065000      MaxRecvLim:   0000065000
   SockOpt:             00
   Multicast Specific:
     Group              IncomingIpAddress
     -----              -----------------
     224.67.113.10      9.67.113.27
     225.67.113.10      9.67.116.91
```

The NETSTAT ALL command displays the following information for UDP sockets being used for outgoing multicast data:

- The time-to-live value
- Whether datagrams are sent to loopback
- The IP address of the link on which the datagrams are sent

The NETSTAT ALL command displays the following information for UDP sockets being used for incoming multicast data:

- The multicast groups by way of their IP addresses (up to a maximum of 20) for which data is being received
- The IP address of the associated link

NETSTAT ALLConn

The NETSTAT ALLConn command displays either closed or time-wait connections, whereas the COnn and BYTEinfo options default to display only active TCP/IP connections:

```
 READY
netstat allconn

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
 User Id  Conn        Local Socket       Foreign Socket     State
 -------  ----        ------------       --------------     -----
 INETD1   0000004B    0.0.0.0..623       0.0.0.0..0         Listen
 INETD1   0000004D    0.0.0.0..514       0.0.0.0..0         Listen
 INETD1   0000004C    0.0.0.0..513       0.0.0.0..0         Listen
 INETD1   0000004E    0.0.0.0..512       0.0.0.0..0         Listen
 INETD1   00000053    0.0.0.0..37        0.0.0.0..0         Listen
 INETD1   0000004F    0.0.0.0..7         0.0.0.0..0         Listen
 INETD1   00000052    0.0.0.0..13        0.0.0.0..0         Listen
 INETD1   00000051    0.0.0.0..19        0.0.0.0..0         Listen
 INETD1   00000050    0.0.0.0..9         0.0.0.0..0         Listen
 INETD1   0000004B    0.0.0.0..623       0.0.0.0..0         Listen
 OSNMPD   00000014    0.0.0.0..1027      0.0.0.0..0         Listen
 TCPCS    00000012    0.0.0.0..23        0.0.0.0..0         Listen
 TCPCS    0000000F    127.0.0.1..1026    127.0.0.1..1025    Establsh
 TCPCS    00000010    127.0.0.1..1025    127.0.0.1..1026    Establsh
 TCPCS    0000000C    0.0.0.0..1025      0.0.0.0..0         Listen
 TCPCS    00000045    127.0.0.1..23      127.0.0.1..1033    Establsh
 TCPCS    00000039    9.67.113.27..23    9.37.81.207..1096  ClosWait
 USER18   00000044    127.0.0.1..1033    127.0.0.1..23      Establsh
 INETD1   00000058    0.0.0.0..37        *..*               UDP
 INETD1   00000057    0.0.0.0..13        *..*               UDP
 INETD1   00000056    0.0.0.0..19        *..*               UDP
 INETD1   00000055    0.0.0.0..9         *..*               UDP
 INETD1   00000054    0.0.0.0..7         *..*               UDP
 OSNMPD   00000013    0.0.0.0..161       *..*               UDP
```

NETSTAT ARp

The NETSTAT ARp command queries the ARP cache for a given address:

```
 READY
netstat arp 9.67.112.25

  MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
  Querying ARP cache for address 9.67.112.25
  Link: TR1              IBMTR: 10005A0019F5
  Route info: 0000

 READY
```

Note: This report will not contain ARP cache data from devices which support ARP Offload. For more information about these devices, refer to the *OS/390 SecureWay Communications Server: IP Configuration* and the *OS/390 SecureWay Communications Server: SNA Network Implementation Guide*.

NETSTAT BYTEinfo

The NETSTAT BYTEinfo command displays byte-count information about each connection.

```
 READY
netstat byteinfo

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS             12:34:56
 03/15/1996          MVS TCP/IP Real Time Network Monitor
 User Id  B Out      B In       L Port  Foreign Socket         State
 -------  -----      ----       ------  --------------         -----
 INETD1   0000000000 0000000000 00623   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00514   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00513   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00512   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00037   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00007   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00013   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00019   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00009   0.0.0.0..0             Listen
 OSNMPD   0000000000 0000000000 01027   0.0.0.0..0             Listen
 TCPCS    0000000000 0000000000 00023   0.0.0.0..0             Listen
 TCPCS    0000000400 0000000436 01026   127.0.0.1..1025        Establsh
 TCPCS    0000000436 0000000400 01025   127.0.0.1..1026        Establsh
 TCPCS    0000000000 0000000000 01025   0.0.0.0..0             Listen
 TCPCS    0000025611 0000000898 00023   127.0.0.1..1033        Establsh
 TCPCS    0000000002 0000000000 00023   9.37.81.207..1096      ClosWait
 USER18   0000000898 0000025611 01033   127.0.0.1..23          Establsh
 INETD1   0000000000 0000000000 00037   *..*                   UDP
 INETD1   0000000000 0000000000 00013   *..*                   UDP
 INETD1   0000000000 0000000000 00019   *..*                   UDP
 INETD1   0000000000 0000000000 00009   *..*                   UDP
 INETD1   0000000000 0000000000 00007   *..*                   UDP
 OSNMPD   0000000216 0000000051 00161   *..*                   UDP
 Connections displayed: 23

 READY
```

You can redirect the preceding screen to a disk file by appending a REPORT subcommand. Use the BYTEinfo subcommand in the following format:

```
netstat byteinfo report
```

The data set `MVSUSER.NETSTAT.BYTEINFO` (`MVSUSER` is the user ID in this example) is created containing the screen output shown earlier.

You can also use NETSTAT BYTEinfo IDLETIME command to display the idle time for each connection:

```
 READY
netstat byteinfo idletime

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS         12:34:56
 03/15/1998          MVS TCP/IP Real Time Network Monitor
 User Id  B Out   B In    L Port  Foreign Socket        State    IdleTime
 -------  ------- ------- ------  --------------        -----    ----
 INETD1   0000000 0000000 00623   0.0.0.0..0            Listen   05:36:25
 INETD1   0000000 0000000 00514   0.0.0.0..0            Listen   05:36:25
 INETD1   0000000 0000000 00513   0.0.0.0..0            Listen   05:36:25
 Connections displayed: 3

 READY
```

Note: For the NETSTAT BYTEINFO IDLETIME display, the byte outbound (B Out) and byte inbound (B In) counts are in three forms:

nnnnnnn
Number range 0–9,999,999

nnnnnnK
Number range 10,000,000–999,999,499 (K = nnnnnn x 1000)

nnnnnnM
Number range 999,999,500–4,294,967,287 (M = nnnnnn x 1,000,000)

NETSTAT CLients

The NETSTAT CLients command displays information about clients:

```
 READY
netstat clients

  MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
  Current Clients:
  Client: INETD1
  Authorization: Autologged
  Last Touched:   4:01:17

  Client: TCPCS
  Authorization: None
  Last Touched:   3:14:47

 READY
```

NETSTAT CACHinfo

The NETSTAT CACHinfo command displays information about Cache Accelerator statistics:

```
READY
netstat cachinfo

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS      12:34:56
 Client: USER1147        Listening socket:  0.0.0.0..80
   MaxCacheSize:       0000000100  CurrCacheSize:      0000000001
   MaxNumObjects:      0000000010  CurrNumObjects:     0000000001
   NumConns:           0000000002  ConnsProcessed:     0000000001
   ConnsDeferred:      0000000001  ConnsTimedOut:      0000000000
   RequestsProcessed:  0000000001  IncompleteRequests: 0000000000
   NumCacheHits:       0000000002  NumCacheMisses:     0000000000
   NumUnprodCacheHits: 0000000000

READY
```

Information is displayed for each listening socket configured for Cache Accelerator support.

NETSTAT CONFIG

The NETSTAT CONFIG command displays information about TCP/IP configuration:

```
  READY
netstat config

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
 TCP Configuration Table:
 DefaultRcvBufSize:  00016384   DefaultSndBufSize:  00016384
 DefltMaxRcvBufSize: 00262114
 MaxReTransmitTime:  120.000    MinReTransmitTime:  0.500
 RoundTripGain:      0.125      VarianceGain:       0.250
 VarianceMultiplier: 2.000      MaxSegLifeTime:     120.000
 DefaultKeepAlive:   0.120      LogProtoErr:        00
 TcpFlags:           00

 UDP Configuration Table:
 DefaultRcvBufSize:  00065535  DefaultSndBufSize:  00065535
 CheckSum:           00000000  LogProtoErr:        01
 UdpFlags:           40

 IP Configuration Table:
 Forwarding: Pkt    TimeToLive: 00064    RsmTimeOut: 00060
 FireWall:   00000  ArpTimeout: 01200    MaxRsmSize: 00000
 IgRedirect: 00000  SysplxRout: 00000    DoubleNop:  00000
 StopClawEr: 00000  SourceVipa: 00000    VarSubnet:  00000
 MultiPath:   Conn  PathMtuDsc: 00000
 DynamicXCF: 00001
   IpAddr: 9.67.116.90      SubNet: 255.255.255.128  Metric: 00

 SMF Parameters:
 InitType: 00  TermType: 00  ClientType: 00  TcpIpStats: 00

 Global Configuration Information:
 TcpIpStats: 00

 Data Trace Setting:
 JobName:  TCP      TrRecCnt: 00000010    Length: FULL
 IpAddress: *       Subnet: *

  READY
```

Notes:

1. The possible values in the Multipath field are: No, Pkt, and Conn.
2. The possible values in the Forwarding field are: No, Yes, Pkt.
3. The TCPIP statistics field under SMF Parameters displays the subtype value used when creating the SMF type 118 record (if the value is non-zero). The TCPIP statistics field under Global Configuration Information indicates whether or not the TCP/IP stack will write statistics messages to the TCP/IP job log when TCP/IP is terminated.

NETSTAT COnn

The NETSTAT COnn command displays information about active TCP connections:

```
 READY
netstat conn

  MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
  User Id  Conn     Local Socket       Foreign Socket     State
  -------  ----     ------------       --------------     -----
  OSNMPD   00000014 0.0.0.0..1027      0.0.0.0..0         Listen
  USER18   00000044 127.0.0.1..1033    127.0.0.1..23      Establsh
  OSNMPD   00000013 0.0.0.0..161       *..*               UDP

  READY
```

NETSTAT DEvlinks

The NETSTAT DEvlinks command displays information about devices and defined links in the TCP/IP address space:

```
 READY
netstat devlinks

   MVS TCP/IP Netstat CS V2R8       TCPIP Name: TCPCS       12:34:56
   DevName: LOOPBACK         DevType: LOOPBACK  DevNum: 0000
     LnkName: LOOPBACK         LnkType: LOOPBACK  Status: Ready
       NetNum: 0  QueSize: 0   ByteIn: 0000053513 ByteOut: 0000053513
     BSD Routing Parameters:
       MTU Size: 00000            Metric: 00
       DestAddr: 0.0.0.0          SubnetMask: 0.0.0.0
     Multicast Specific:
       Multicast Capability: No
   DevName: LCS1              DevType: LCS       DevNum: 0120
     LnkName: TR1               LnkType: TR        Status: Ready
       NetNum: 1  QueSize: 0    ByteIn: 0000762810 ByteOut: 0000077579
       ArpMacAddress: Non-Canonical   SrBridgingCapability:  Yes
       BroadcastCapability: Yes       BroadcastType: All Rings
     BSD Routing Parameters:
       MTU Size: 00000              Metric: 00
       DestAddr: 0.0.0.0            SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
       Group             RefCnt
       -----             ------
       224.67.113.10     3
       225.67.113.10     5
  DevName: CTCD00           DevType: CTC       DevNum: 0D00
     LnkName: CTC1             LnkType: CTC       Status: Not Active
       NetNum: 1 QueSize: 0   ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000            Metric: 00
       DestAddr: 9.67.116.90      SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
   DevName: PTPDEV            DevType: MPC       DevNum: 0000
     LnkName: PTPLINK           LnkType: MPC       Status: Not Active
       NetNum: 0  QueSize: 0    ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000              Metric: 00
       DestAddr: 0.0.0.0            SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
   DevName: OSATRL90         DevType: ATM       DevNum: 0000
     LnkName: OSA90LINK1       LnkType: ATM       Status: Not Active
       NetNum: 0  QueSize: 0   ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000            Metric: 00
       DestAddr: 0.0.0.0          SubnetMask: 255.255.255.128
     ATM Specific:
       ATM portName:  OSA90
       ATM PVC Name:  STEPH            PVC Status: Not Active
       ATM LIS Name:  LIS1
       SubnetValue:   9.67.1.0         SubnetMask:    255.255.255.0
       DefaultMTU:    0000009180       InactvTimeOut: 0000000300
       MinHoldTime:   0000000060       MaxCalls:      0000001000
       CachEntryAge:  0000000900       ATMArpReTry:   0000000002
       ATMArpTimeOut: 0000000003       PeakCellRate:  0000000000
       NumOfSVCs:     0000000000
       ATMARPSV Name: ARPSV1
       VcType:        PVC              ATMaddrType:  NSAP
       ATMaddr:
       IpAddr:        0.0.0.0
     Multicast Specific:
       Multicast Capability: No
```

The Multicast Specific field is significant only for multicast-capable devices. If a link is being used to receive multicast data, then all the multicast groups, and the counts of receivers for each multicast group, are displayed. There is no limit to the number of multicast groups for which a link can receive data.

The Multicast Capability field is always Yes for the following devices: CDLC, CLAW, CTC, and MPCPTP. For LCS and MPCIPA devices, the multicast capability is only known after the device is in the Ready state. If the device is not yet Ready, the multicast capability is Unknown.

NETSTAT DRop

The NETSTAT DRop command drops the TCP connection specified by *n*:

```
 READY

netstat conn
  MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
  User Id  Conn     Local Socket       Foreign Socket     State
  -------  ----     ------------       --------------     -----
  PORTMP3  00010035 0.0.0.0..2220      0.0.0.0..0         Listen
  TSUSER1  00010020 0.0.0.0..1027      0.0.0.0..0         Listen
  TSUERS2  00010043 127.0.0.1..1033    127.0.0.1..23      Establsh
  PORTMP3  00021002 0.0.0.0..2221      *..*               UDP

 READY
netstat drop 10035

 Connection successfully dropped
  READY

netstat conn
  MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
  User Id  Conn     Local Socket       Foreign Socket     State
  -------  ----     ------------       --------------     -----
  TSUSER1  00010020 0.0.0.0..1027      0.0.0.0..0         Listen
  TSUERS2  00010043 127.0.0.1..1033    127.0.0.1..23      Establsh
  PORTMP3  00021002 0.0.0.0..2221      *..*               UDP

READY
```

NETSTAT Gate

The NETSTAT Gate command displays information about gateways:

```
netstat gate

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS            12:34:56
 Known gateways:
 NetAddress      FirstHop      Link     Pkt Sz Subnet Mask     Subnet Value
 ----------      --------      ----     ------ -----------     ------------
 Defaultnet      9.67.113.1    TR1      576    <none>
 9.0.0.0         <direct>      TR1      2000   0.255.255.128   0.67.113.0
 9.67.116.90     <direct>      CTC1     4000   HOST
 224.0.0.5       <direct>      TR1      2000   HOST

READY
```

You can also use DETAIL to display gateway information plus MVS-specific configured parameters.

```
netstat gate detail
 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS            12:34:56
 Known gateways:
 NetAddress      FirstHop      Link     Pkt Sz Subnet Mask     Subnet Value
 ----------      --------      ----     ------ -----------     ------------
 Defaultnet      9.67.113.1    TR1      576    <none>
   Metric:  -00000001
   MVS specific configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 9.0.0.0         <direct>      TR1      2000   0.255.255.128   0.67.113.0
   Metric:  -00000001
   MVS specific configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 9.67.116.90     <direct>      CTC1     4000   HOST
   Metric:  -00000001
   MVS Specific Configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 224.0.0.5       <direct>      TR1      2000   HOST
   Metric:  -00000001
   MVS Specific Configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000

READY
```

NETSTAT HElp

The NETSTAT HElp command displays help information for the NETSTAT parameters:

```
 READY
netstat help

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
 Usage: NETSTAT < Option | Command >  <Target>  <Output>  < ( Select >
 Option:
 ALL         - Everything about a connection
 ALLConn     - TCP/IP connections, include TIME-WAIT and CLOSED connections
 ARp         - Query ARP table or entry information
 BYTEinfo    - Current connection display
 CACHinfo    - Cache Accelerator data
 CLients     - Current clients
 CONFIG      - Configuration data
 COnn        - Active TCP/IP connection (Default option)
 DEvlinks    - Devices and links
 Gate        - Current known gateways
 HElp or ?   - Netstat information list
 HOme        - Home address list
 PORTList    - Display port reservation list
 ROUTe       - Display routing information in a standard fashion
 SLAP        - SLA Policy data
 SOCKets     - Socket interface users and their sockets
 TELnet      - Telnet connection information
 UP          - Date and time tcpip was last started
 VIPADyn     - Dynamic VIPA information
 Target option:
 TCp         - Displays detailed information about the specified
               TCPIP address space
 Output option:
 REPort      - Netstat information written to dataset name
               'tsoprefix.NETSTAT.option' or specified via DSN/HLQ
 STACk       - Netstat information written to REXX stack
 Select String:
 CLIent      - For All, ALLCON, BYTEinfo, CLIENT, CONN, and TELnet,
               select information on the specified client name
 IPAddr      - For ALLCON, CONN, GATE and TELnet,
               select information on the specified IP address
 POrt        - For ALLCON, CONN, and TELnet,
               select information on the specified port
 APPLname    - For TELNET, select information on the specified application name
 LUNname     - For TELNET, select information on the specified LU name
 Command:
 DRop        - Drop a TCP/IP connection

 READY
```

NETSTAT HOme

The NETSTAT HOme command displays the Home address list:

```
 READY
netstat home

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS       12:34:56
 Home address list:
 Address           Link          Flg
 -------           ----          ---
 9.67.113.27       TR1           P
 9.67.116.91       CTC1
 9.67.116.99       ETH2
 9.67.1.8          OSA90LINK1
 9.67.116.1        X25LINK
 9.67.116.2        CDLCLINK
 9.67.116.3        HCHLINK
 9.67.116.4        PTPLINK
 9.67.116.5        SNA1
 9.67.116.6        SNA62
 127.0.0.1         LOOPBACK

 READY
```

NETSTAT PORTList

The NETSTAT PORTList command displays the port reservation list:

```
 READY
netstat portlist

 MVS TCP/IP NETSTAT  CS V2R8      TCPIP Name: TCPCS       12:34:56
 Port# Prot User     Flags Range
 ----- ---- ----     ----- -----
 00020 TCP  OMVS     A
 00021 TCP  OMVS     A
 00025 TCP  SMTP     A
 00999 TCP  IOASRV   A
 02026 TCP  IOASNMP  AO
 03035 TCP  OMVS     AS
 05000 UDP  APPL1    AR    05000-05009
 05001 UDP  APPL1    AR    05000-05009
 05002 UDP  APPL1    AR    05000-05009
 05003 UDP  APPL1    AR    05000-05009
 05004 UDP  APPL1    AR    05000-05009
 05005 UDP  APPL1    AR    05000-05009
 05006 UDP  APPL1    AR    05000-05009
 05007 UDP  APPL1    AR    05000-05009
 05008 UDP  APPL1    AR    05000-05009
 05009 UDP  APPL1    AR    05000-05009

 READY
```

NETSTAT REPort

The NETSTAT REPort command causes output to be stored in the data set *tsoprefix*.NETSTAT.option. Use this command when you want to store the screen information produced from NETSTAT in a file used for later reference. For example, to store the output of the NETSTAT COnn command into a file, issue the following command:

```
netstat conn report
```

After you issue the preceding command, a data set called *tsoprefix*.NETSTAT.CONN is created which contains the following:

```
 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS      12:34:56
 User Id Conn      Local Socket           Foreign Socket    State
 ------- ----      ------------           --------------    -----
 TSUSER1 00010020  *..2150                *..*              Listen
 TSUSER2 00010043  *..2151                *..*              Listen
 PORTMP3 00021002  *..2221                *..*              UDP
```

NETSTAT ROUTe

The NETSTAT ROUTe command displays routing information.

```
 READY
netstat route

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPCS          12:34:56
 Destination      Gateway         Flags  Refcnt  Interface
 -----------      --------        -----  ------  ---------
 Defaultnet       9.67.113.1      UG     000001  TR1
 9.67.113.0       0.0.0.0         U      000000  TR1
 9.67.116.90      0.0.0.0         H      000000  CTC1
 224.0.0.5        0.0.0.0         UH     000000  TR1

 READY
```

NETSTAT SLAP

The NETSTAT SLAP command provides information about the service policy definitions and policy statistics:

```
MVS TCP/IP Netstat CS V2R8        TCPIP Name: TCPCS2          17:33:10
Policy:
  Profile:   DataPol
    Scope:             DataTraffic         Direction:        Both
    Permission:        Allowed             Protocol:         TCP
    LocalInterface:    0.0.0.0             Position:         0000000000
    SourceIpFrom:      0.0.0.0             SourceIpTo:       0.0.0.0
    SourcePortFrom:    008000              SourcePortTo:     008000
    DestIpFrom:        9.67.200.0          DestIpTo:         9.67.200.255
    DestPortFrom:      000000              DestPortTo:       000000
    ServiceClass:      DataCat
      Scope:           DataTraffic         OutgoingTOS:      C0
      Priority:        0                   Interface:        0.0.0.0
      MaxRate:         0001000000          MinRate:          0000010000
      MaxDelay:        0000000000          MaxConn:          0000000000
    Performance Information:
      FirstActTime:    20:28:12            LastMapTime:      20:31:08
      TotalBytesIn:    0000000000          TotalBytesOut:    0000125004
      BytesInDiscard:  0000000000          BytesOutDiscard:  0000000000
      TotalInPackets:  0000000216          TotalOutPackets:  0000000326
      ActConnMap:      0000000000          MaxConnLimit:     0000000000
      AcceptConn:      0000000002          DeniedConn:       0000000000
Policy:    JKF
  Profile:   CLPol
    Scope:             RSVP                Direction:        Outgoing
    Permission:        Allowed             Protocol:         0
    LocalInterface:    0.0.0.0             Position:         0000000000
    SourceIpFrom:      0.0.0.0             SourceIpTo:       0.0.0.0
    SourcePortFrom:    000000              SourcePortTo:     000000
    DestIpFrom:        0.0.0.0             DestIpTo:         0.0.0.0
    DestPortFrom:      000000              DestPortTo:       000000
    ServiceClass:      CLCat
      Scope:           DataTraffic         OutgoingTOS:      E0
      Priority:        0                   Interface:        0.0.0.0
      MaxRate:         0000000000          MinRate:          0000000000
      MaxDelay:        0000000000          MaxConn:          0000000000
    Performance Information:
      FirstActTime:    20:28:12            LastMapTime:      20:31:29
      TotalBytesIn:    0000000000          TotalBytesOut:    0000026960
      BytesInDiscard:  0000000000          BytesOutDiscard:  0000000000
      TotalInPackets:  0000000049          TotalOutPackets:  0000000072
      ActConnMap:      0000000000          MaxConnLimit:     0000000000
      AcceptConn:      0000000000          DeniedConn:       0000000000
```

NETSTAT SOCKets

The NETSTAT SOCKets command provides information about each client using the socket interface:

```
 READY
netstat sockets

 MVS TCP/IP NETSTAT CS V2R8        TCPIP Name: TCPCS          12:34:56
 Sockets interface status:
 Type   Bound to                Connected to          State    Conn
 ====   ========                ============          =====    ====
 Name: INETD1    Subtask: 006DB5B8
 Dgram  0.0.0.0..37             *..*                  UDP      00000058
 Dgram  0.0.0.0..13             *..*                  UDP      00000057
 Dgram  0.0.0.0..19             *..*                  UDP      00000056
 Dgram  0.0.0.0..9              *..*                  UDP      00000055
 Dgram  0.0.0.0..7              *..*                  UDP      00000054
 Stream 0.0.0.0..623            0.0.0.0..0            Listen   0000004B
 Stream 0.0.0.0..514            0.0.0.0..0            Listen   0000004D
 Stream 0.0.0.0..513            0.0.0.0..0            Listen   0000004C
 Stream 0.0.0.0..512            0.0.0.0..0            Listen   0000004E
 Stream 0.0.0.0..37             0.0.0.0..0            Listen   00000053
 Stream 0.0.0.0..7              0.0.0.0..0            Listen   0000004F
 Stream 0.0.0.0..13             0.0.0.0..0            Listen   00000052
 Stream 0.0.0.0..19             0.0.0.0..0            Listen   00000051
 Stream 0.0.0.0..9              0.0.0.0..0            Listen   00000050
 Name: OSNMPD    Subtask: 006DBA70
 Dgram  0.0.0.0..161            *..*                  UDP      00000013
 Stream 0.0.0.0..1027           0.0.0.0..0            Listen   00000014
 Name: TCPCS     Subtask: 00000000
 Stream 127.0.0.1..23           127.0.0.1..1033       Estblsh  00000045
 Stream 9.67.113.27..23         9.37.81.207..1096     ClosWait 00000039
 Name: TCPCS     Subtask: 006C57B0
 Stream 0.0.0.0..23             0.0.0.0..0            Listen   00000012
 Name: TCPCS     Subtask: 006D56F0
 Stream 127.0.0.1..1026          127.0.0.1..1025      Establsh 0000000F
 Name: TCPCS     Subtask: 006D5CF0
 Stream 0.0.0.0..1025           0.0.0.0..0            Listen   0000000C
 Stream 127.0.0.1..1025         127.0.0.1..1026       Establsh 00000010
 Name: USER18    Subtask: 006A3400
 Stream 127.0.0.1..1033         127.0.0.1..23         Establsh 00000044

 READY
```

NETSTAT STACk

The NETSTAT STACk command puts the response in the program stack when the command is issued from a CLIST or REXX exec. Title lines are not placed in the stack.

No information is displayed at the user's terminal.

NETSTAT TCp

The NETSTAT TCp command displays detailed information about the specified TCP/IP job. The output can be formatted in different ways, depending on the specific command you used. This command returns information about the specified TCP/IP address space.

The following example shows information about active TCP/IP connections for the TCP/IP address space TCPIPA.

```
netstat tcp tcpipa

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPIPA      12:34:56
 User Id  Conn     Local Socket          Foreign Socket     State
 -------  ----     ------------          --------------     -----
 CFTPSERV 00010040 0.0.0.0..5097         0.0.0.0..0         Listen
 MISCSRV  00010031 0.0.0.0..7            0.0.0.0..0         Listen
 MISCSRV  00010051 0.0.0.0..9            0.0.0.0..0         Listen
 TSUSER5  00010010 0.0.0.0..33333        0.0.0.0..0         Listen

 READY
```

The following example shows a NETSTAT TCp command issued for the TCP/IP address space TCPIPA, an address that is not the default. The name of the TCP/IP address space must be included in the command.

```
 READY
netstat arp 9.67.112.25 tcp TCPIPA

 MVS TCP/IP NETSTAT CS V2R8      TCPIP Name: TCPIPA           12:34:56
 Quering ARP cache for address 9.67.112.25
 Link:  TR1             IBMTR: 10005A0019F5
 Route info:  0000

 READY
```

NETSTAT TELnet

The NETSTAT TELnet command displays the status of the internal Telnet server:

```
 READY
netstat telnet

 MVS TCP/IP NETSTAT CS V2R8        TCPIP Name: TCPCS            12:34:56
 Internal Telnet Server Status:
 Conn     Foreign Socket       State    BytesIn  BytesOut ApplName LuName
 ----     --------------       -----    -------  -------- -------- ------
 00000045 127.0.0.1..1033      Establsh 00000898 00025611 TSO10002 TCPM1001

 READY
```

The NETSTAT TELnet DETAIL command displays additional server status information:

```
 READY
netstat telnet detail

 MVS TCP/IP NETSTAT CS V2R8        TCPIP Name: TCPCS            12:34:56
 Internal Telnet Server Status:
 Conn     Foreign Socket       State    BytesIn  BytesOut ApplName LuName
 ----     --------------       -----    -------  -------- -------- ------
 00000045 127.0.0.1..1033      Establsh 00000898 00025611 TSO10002 TCPM1001
   ModeName: NSX32702  TNProto:  TN3270    TnUserID:

 READY
```

NETSTAT Up

The NETSTAT Up command displays the date and time that TCP/IP was started:

```
 READY
netstat up

 MVS TCP/IP NETSTAT CS V2R8        TCPIP Name: TCPCS          12:34:56
 Tcpip started at 09:08:15 on 07/09/1997

 READY
```

NETSTAT VIPADyn

The NETSTAT VIPADyn command displays the current dynamic VIPA information for a local host:

```
 NETSTAT VIPADyn

   MVS TCP/IP NETSTAT CS V2R8     TCPIP Name: TCPCS       12:34:56

   IP Address       AddressMask      Status   Origination
   ----------       ----------       ------   -----------
   9.67.102.30      255.255.255.128  Active   VIPARange Bind
   9.67.102.31      255.255.128.0    Backup   VIPABackup
   9.67.102.32      <NONE>           Backup   VIPABackup
   9.67.113.27      255.255.255.0    Active   VIPADEFine
   9.67.113.28      255.255.0.0      Backup   VIPADEFine Contention
   9.67.116.10      255.255.0.0      Active   VIPARange IOCTL

READY
```

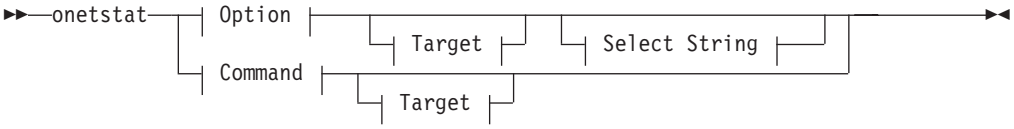
The OS/390 UNIX onetstat/netstat Command—Display Local Host Information

Purpose

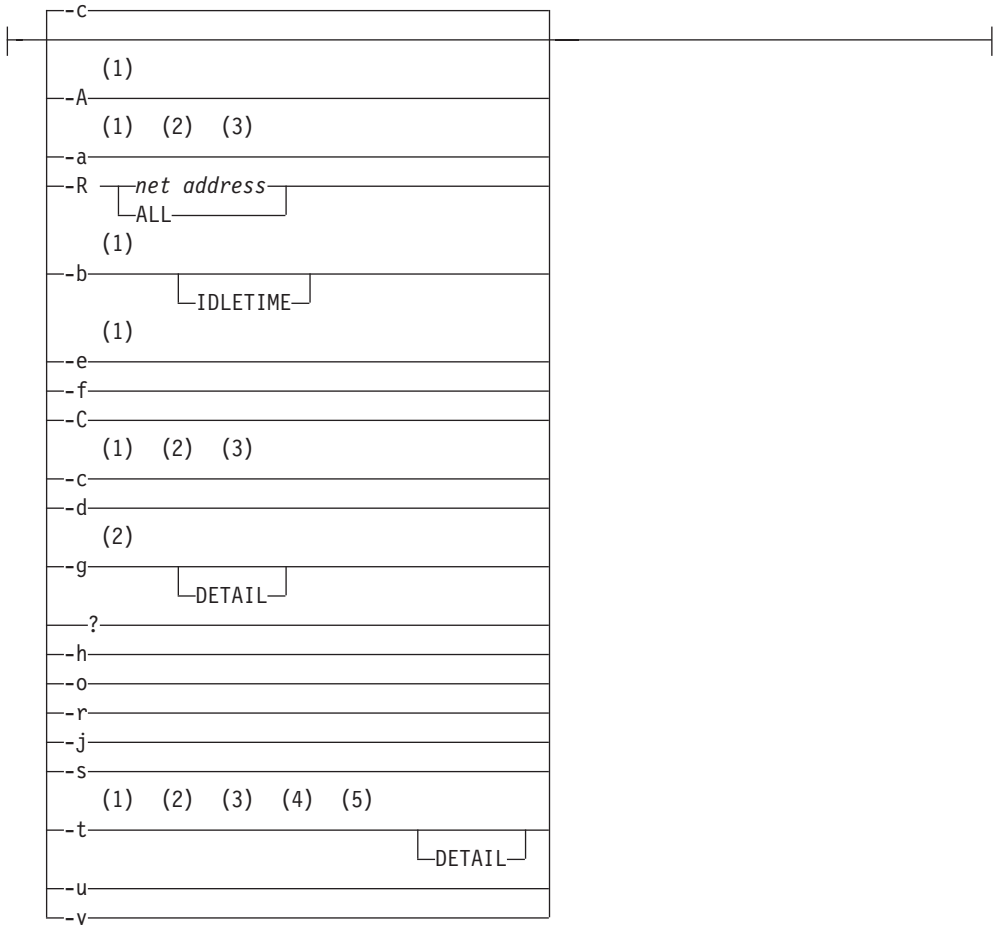
Use the OS/390 UNIX onetstat command to display the network status of the local host.

Note: netstat is a synonym for the onetstat command in the OS/390 UNIX shell. netstat command syntax is the same as that for the onetstat command.

Format

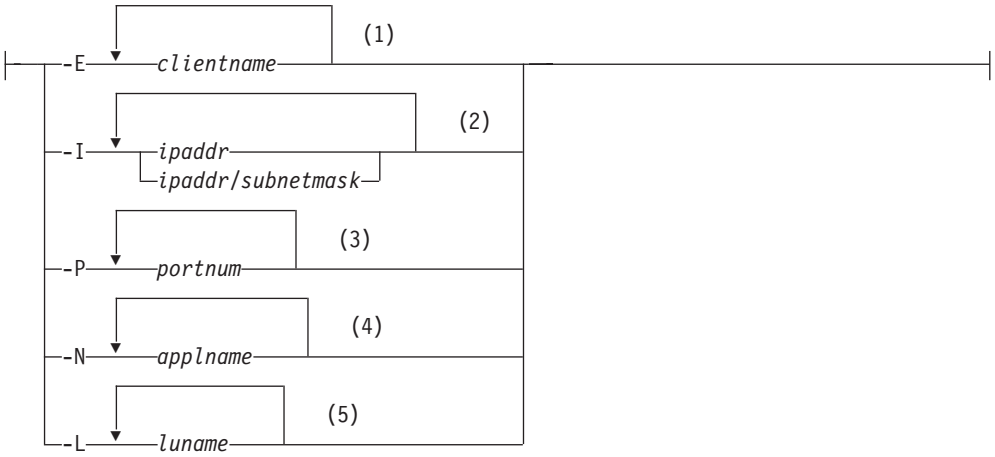


Option:

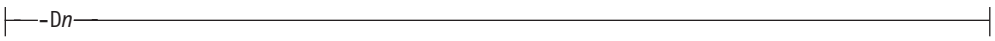


Target:

Select String:



Command:



Notes:

1 -E is valid only with -A, -a, -b, -e, -c and -t.

2 -I is valid only with -a, -c, -g and -t.

3 -P is valid only with -a, -c and -t.

4 -N is valid only with -t.

5 -L is valid only with -t.

Parameters

-A Provides detailed information about TCP/IP connections. This option is useful for debugging the TCP/IP address space. For more information about maintaining the TCP/IP address space, refer to *OS/390 SecureWay Communications Server: IP Configuration*.

If a UDP socket is being used for outgoing multicast data, this parameter displays the time-to-live value, indicates whether the datagrams are also sent to loopback, and displays the IP address of the link on which the datagrams are being sent. If a UDP socket is being used for incoming multicast data, this parameter lists the multicast groups by way of their IP addresses (up to a maximum of 20) for which data is being received and the IP address of the associated link.

-a Specifies that all information for all connections, including recently closed ones, is displayed.

-R Queries the ARP cache information.

net address
Queries the ARP cache for a given address.

ALL
Queries all ARP cache entries.

Note: This report will not contain ARP cache data from devices which support ARP Offload. For more information about these devices, refer to *OS/390 SecureWay Communications Server: IP Configuration* and the *OS/390 SecureWay Communications Server: SNA Network Implementation Guide*.

-b Displays the byte-count information about each connection. This option is most useful when used in conjunction with the redirect function (>).

The following information is given for each connection:
- Client name
- Bytes sent on the connection
- Bytes received on the connection
- Local port
- Foreign socket
- State

IDLETIME
Displays the preceding byte-count information plus the idle time for each connection.

The idle time is displayed in the following format:

```
hours:minutes:seconds
```

-e Provides the following information about each client:
- Authorization
- Elapsed time since the client was last used.

-f Provides the following TCP/IP configuration data:
- Configured IP information
- Configured TCP information
- Configured UDP information
- SMF parameters
- GLOBALCONFIG Profile information

Note: The TCPIPSTATS field shown under the SMF PARAMETERS section of the onetstat -f output reflects the value of TcpIpStatistics or NoTcpIpStatistics specified on the SMFCONFIG statement in the TCP/IP Profile or Obeyfile. The TCPIPSTATS field shown under the GLOBAL CONFIGURATION section of the onetstat -f output reflects the value from the ASSORTEDPARMS or GLOBALCONFIG statements in the TCP/IP Profile or Obeyfile.

- Data trace settings
- Dynamic VIPA information

-C Displays information about Cache Accelerator statistics. Statistics are displayed for each listening socket configured for Cache Accelerator support.

For each listening socket configured for Cache Accelerator support, the following information is displayed:

Client The user name of the application that bound the listening socket.

Socket The local IP address and port pair to that the listening socket is bound.

MaxCacheSize
The maximum number of 4K pages that may be used for storing cache objects by the Cache Accelerator for the given socket.

CurrCacheSize
The number of 4K pages currently being used for storing cache objects by the Cache Accelerator.

MaxNumObjects
The maximum number of cache objects that may be stored by the Cache Accelerator.

CurrNumObjects
The current number of cache objects stored by the Cache Accelerator.

NumConns The number of connections established via a listening socket which has been configured with Cache Accelerator support.

ConnsProcessed
The number of connections that have successfully completed an in-kernel transaction, resulting in a response being transmitted to the client. This counter is incremented at most once per connection.

Note: It is possible for a single connection to be processed by the Cache Accelerator for some cache entries and then deferred to the application for additional processing. If this occurs, the connection will be included in both ConnsProcessed and ConnsDeferred.

ConnsDeferred
The number of connections that require user-space application processing.

Note: This counter is not incremented due to the connection timeout expiration, even if the action taken is to defer the connection.

ConnsTimedOut
The number of times the connection timeout timer has expired.

RequestsProcessed
The number of connection requests that were at least partially processed by the Cache Accelerator.

Note: It is possible for a single connection to be processed by the Cache Accelerator for some cache objects and then deferred to the application for additional processing. If this occurs, the connection is included in both RequestsProcessed and RequestsDeferred.

IncompleteRequests
The number of times a request is received from the client where additional data is required to process the request. This counter may be incremented multiple times for a single connection.

NumCacheHits
The number of cache objects that were successfully located and transmitted to clients.

NumCacheMisses
The number of cache objects that were not successfully located and transmitted to clients.

NumUnprodCacheHits
The number of cache entries succesfully found within the cache but not transmitted to the client.

-c Provides the following information about each active TCP/IP connection. An active connection is a connection that is not in the *closed* or *time-wait* state.
- Client name
- Client ID
- Local IP address
- Foreign IP address
- Connection state

-c is the default parameter.

A connection progresses through a series of states during its lifetime. The following are the possible states for a TCP connection.

State	Description
LISTEN	Waiting for a connection request from any remote TCP and port
SYN-SENT	Waiting for a matching connection request after having sent a connection request
SYN-RECEIVED	Waiting for a confirming connection request acknowledgment after having both received and sent a connection request
ESTABLISHED	Represents an open connection; this is the normal state for the data transfer phase of the connection
FIN-WAIT-1	Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request
FIN-WAIT-2	Waiting for a connection termination request from the remote TCP
CLOSE-WAIT	Waiting for a connection termination request from the local user
CLOSING	Waiting for a connection termination request acknowledgment from the remote TCP
LAST-ACK	Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP

For more information about the TCP connection states, see RFC793.

-d Displays the following information about devices and defined links in the TCP/IP address space:
- Device name
- Device type
- Device number
- Link name

- Link type
- Status of link

  The following list describes the possible link statuses.

Status	Description
Starting	A START of the device has been issued by the operator, and TCP/IP has been sent an Activation request to the Data Link Control (DLC) layer.
Enabling	DLC has acknowledged the Activation reques from TCP/IP, and TCP/IP has requested DLC to allow TCP/IP to receive inbound connections for the device.
Sent SETUP	DLC has acknowledged the Activation request from TCP/IP, and TCP/IP has requested DLC to perform the initial I/O sequence with the device.
Connecting	DLC has accepted the Initial I/O Sequence request.
Connecting2	The control connection for a Common Link Access to Workstation protocol (CLAW) device has been established, and the second connection (on which IP traffic is carried) is being established.
Negotiating	The initial I/O sequence with the device is complete, and TCP/IP is performing additional link-layer initialization.
Ready	The initialization sequence with the device is complete. The device is now ready.
Sent CLEAR	A STOP of the device has been issued by the operator, and TCP/IP has sent a Deactivation request to DLC.
Deactivated	DLC has performed the first stage of an orderly device deactivation.
Not active	The device is not active. (The device has never been started, or has been stopped after having been started.)

- Net number

  This field is significant only for links on LCS and CTC devices.
- Queue size

  This field is significant only for links on LCS and ATM devices.
- ARP MAC addresses

  This field is significant only for token-ring links.
- SR bridging capability

  This field is significant only for token-ring links.
- Multicast specific

  This field is significant only for multicast capable devices.

  If a link is being used to receive multicast data, then all the multicast groups, and the counts of receivers for each multicast group, are displayed. There is no limit to the number of multicast groups for which a link can receive data.

  – Multicast capability

    This field is always Yes for the following devices: CDLC, CLAW, CTC, and MPCPTP.

    For LCS and MPCIPA devices, the multicast capability is only known after the device is in the Ready state. If the device is not yet Ready, the multicast capability is Unknown.
- Broadcast capability

  This field is significant only for links on LCS devices.
- Broadcast type

  This field is significant only for token-ring links.
- Number of bytes received
- Number of bytes transmitted
- BSD parameters

- Packet trace settings
- ATM specific information

  This field is significant only for ATM devices and links.

  Notes:

  1. No link related information, packet trace settings, or BSD parameters are displayed for a device that has no link defined.
  2. The packet trace setting is displayed only when it is defined and set to on.
  3. ATM specific information is displayed only for ATM devices which have links defined.
  4. The LOOPBACK devices and links are displayed.

-g Provides the following information about each gateway:
- Address of the network
- First hop address
- Link name used by the first hop

  Note: Only the first 8 characters of the link name are displayed by this command. Issue the onetstat -r command to see more than 8 characters of the link name.
- Packet size used by the first hop
- Subnet mask and subnet value

DETAIL

Displays the preceding information plus the metric associated with the cost of the use for the link, and displays the following MVS specific configured parameters for each gateway:
- Maximum retransmit time
- Minimum retransmit time
- Round trip gain
- Variance gain
- Variance multiplier

Note: The static routes associated with deleted interfaces in the routing table no longer appear in the reports generated with the onetstat -g command. For details, refer to *OS/390 SecureWay Communications Server: IP Diagnosis*.

-? Provides help information for the onetstat parameters.

-h Displays the HOME list. The IP address, link name, and primary interface information are displayed for each entry in the list.

For more information about the home list, refer to *OS/390 SecureWay Communications Server: IP Configuration*.

-o Displays the port reservation list:
- Port number
- Protocol
- username
- Flags

  A Autolog

  D DelayAcks

  O Optimize MSS

  R Port is reserved by range

S Share port

- Range

  This field is significant only for port entry reserved by range (flag R in the Flags field).

-r Displays routing information:

Destination
The address of a destination host or network

Gateway
The gateway used in forwarding packets

Flags The state of the route:
U The route is up.
H The route is to a host rather than to a network.
G The route is a gateway.
D The route was created dynamically by a redirect.

Reference count
The current number of active users for the route

Interface
The link name for the route

Note: The static routes associated with deleted interfaces in the routing table no longer appear in the reports generated with the onetstat -r command. For details, refer to *OS/390 SecureWay Communications Server: IP Diagnosis*.

-j Displays all of the service policy definitions (policy rules, policy profiles, and service classes) and policy stats data.

-s Displays information about each client using the socket interface. When you specify the onetstat -s command, information about the client using the socket interface is displayed along with information about the sockets and associated connections owned by the client.

The following is a list of the information displayed after invoking the -s parameter:

Name Displays the client's address space name.

Subtask
Displays the subtask identifier. The subtask identifier is combined with the address space name to produce a unique identifier for the client.

- For socket programs written in the C language, the EBCDIC hexadecimal representation of an address within the program is used as the subtask identifier.

Type Displays one of the following socket types:
- `Stream` for stream (TCP) sockets
- `Dgram` for datagram (UDP) sockets

Bound to
Indicates the address and port to which the socket is bound. The output is in the format `internet address..bound port` where `internet address` is the address to which the socket is bound and `bound port` is the port number to which the socket is bound. Unbound TCP and UDP sockets are not displayed by onetstat -c.

Connected to
Displays the address and port to which the socket is connected.

State Displays the TCP connection state for TCP sockets or UDP for UDP sockets.

Conn Displays the client identifier which is a unique number assigned by TCP/UDP stack to uniquely identify a socket entity.

-t Displays the status of the internal Telnet server.

DETAIL
Displays the logmode and Telnet protocol in use by each connection. If an application user ID was entered on the solicitor panel, it is displayed in the *TnUserId* field. Otherwise, the *TnUserId* field is blank.

-u Provides the date and time that TCP/IP was started.

-v Displays the current dynamic VIPA information for a local host.

-p *tcpname*
Displays detailed information about the specified TCP/IP address space. You can use -p *tcpname* with any other onetstat parameter to get information about the specified TCP/IP address space; however, this parameter works only for TCP/IP address spaces having the same version as onetstat.

The *tcpname* is an 8-byte procedure name that is used to start the TCP/IP. When the 'S member.identifier' method of starting TCP/IP is used, the value for *identifier* must be used as *tcpname*.

select_string
Specifies a character string used to limit the option responses

-E *clientname*
Provides the response of -A, -a, -b, -c, -e and -t on the specified client name.

-I *ipaddr*

-I *ipaddr/subnetmask*
Provides the response of -a, -c, -g and -t on the specified IP address *ipaddr* or *ipaddr/subnetmask*. If *subnetmask* is not specified, the default of 255.255.255.255 is used.

Notes:

1. For the -g option, *ipaddr* is the destination IP address not the destination network address.
2. When filtering -g responses on a specified *ipaddr*, the DEFAULT and DEFAULTNET routes are always displayed, as they apply to all IP addresses.
3. For -a, -c, and -t options, *ipaddr* can be either the local or remote IP address.

-P *portnum*
Provides the response of -a, -c and -t on the specified port number.

Note: The port number can be either a local or remote port.

-N *applname*
Provides the response of -t on the specified VTAM application name.

-L *luname*
Provides the response of -t on the specified LU name.

The *select_string* can be a complete string or a partial string using wildcard characters. A wildcard character can be an asterisk (*), which matches a null string or any character or character string, at the same position. A wildcard character can be a ?, which matches any single character at the same position. For example, a string ”searchee” matches with ”*ar?he*”, but the string ”searhee” does not match with ”*ar?he*”. See “Usage” for additional information.

You can enter up to six *select_string* values. For -I, each selected value can be up to 15 characters long, and for -E, -N and -L, each selected value can be up to eight characters long.

If you want to use the wildcard character on the -I *select_string*, you must specify the select value in the *ipaddr* format. The wildcard character is not accepted for the *ipaddr/subnetmask* format of -I values.

If you specified *select_string*, it must be the last parameter on the onetstat command line.

-D *n*
Drops the TCP/IP connection specified by *n*. You can determine the connection number from the -c column in the onetstat -c display. If you drop the server’s *passive open* connection, the server immediately reissues the open request.

You can use this parameter only if your RACF profile contains the MVS.VARY.TCPIP.DROP definition.

Usage

- The time displayed in the header for each option is local time. The time field displayed in options -A, -b, -e, and -u is Greenwich mean time (GMT).
- Care should be taken when you use a OS/390 UNIX MVS special character in a character string. It may cause an unpredictable result. To be safe, if you want to use a OS/390 UNIX MVS special character in a character string, the character string should be surrounded by single quotes. For example, to use an asterisk (*) in the IP address, 10.*.0.0 for -I *select_string* option, issue the command as:

  onetstat -g -I ’10..0.0’*

Examples

This section contains examples of the response that is displayed after issuing the onetstat command with each parameter. To help you find the examples, following is a list of the onetstat parameters and the page number where you will find the example:

- -c on page 407
- -d on page 407
- -D *n* on page 409
- -g on page 409
- -g DETAIL on page 410
- -? on page 410
- -h on page 411
- -o on page 411
- -r on page 412
- -j on page 412
- -s on page 413
- -p *tcpname* on page 414
- -t on page 415
- -u on page 415
- -v on page 415

onetstat -A

The onetstat -A command displays information about TCP/IP connections. The client TCPCLIE1 in the following example shows information about a TCP connection, and the client UDPCLIE1 shows information about a UDP connection.

```
onetstat -A

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS            08:59:29
 Client Name: 0000008                 Client Id: 00008
 Local Socket: 127.0.0.1..1025         Foreign Socket: 127.0.0.1..1026
   Last Touched:       11:53:11          State:              Establsh
   BytesIn:            0000000644        BytesOut:           0000000840
   SegmentsIn:         0000000044        SegmentsOut:        0000000040
   RcvNxt:             1111472027        SndNxt:             1111472543
   ClientRcvNxt:       1111472027        ClientSndNxt:       1111472543
   InitRcvSeqNum:      1111471382        InitSndSeqNum:      1111471702
   CongestionWindow:   0000410006        SlowStartThreshold: 0000016384
   IncomingWindowNum:  1111504755        OutgoingWindowNum:  1111505275
   SndWl1:             1111471987        SndWl2:             1111472543
   SndWnd:             0000032732        MaxSndWnd:          0000032768
   SndUna:             1111472543        rtt_seq:            1111472507
   MaximumSegmentSize: 0000065483        OptMaxSegmentSize:  0000065483
   BackoffCount:                         Precedence:         Routine
   Round-trip information:
     Smooth trip time: 69.000             SmoothTripVariance: 113.000
   ReXmt:              0000000000        ReXmtCount:         0000000000
   DupACKs:            0000000000
   SockOpt:            00                TcpTimer:           00
   TcpSig:             04                TcpSel:             20
   TcpDet:             30                TcpPol:             00
   TcpClusterConnFlag: 82
   ReceiveBufferSize:  0
 ----
 Client name: 00010003               Client id : 10003
 Local Socket: 0.0.0.0..2025         Foreign Socket: *..*
   Last Touched:        3:35:05
   Precedence:          Routine
   BytesIn:             0000400000      BytesOut:     0000400000
   DgramIn:             0000000100      DgramOut:     0000000100
   MaxSendLim:          0000065000      MaxRecvLim:   0000065000
   SockOpt:             00
   Multicast Specific:
     TimeToLive:        0000000001      LoopBack:     Yes
     OutgoingIpAddress: 9.67.113.27
 ----
 Client name: 00010003               Client id : 10003
 Local Socket: 0.0.0.0..2025         Foreign Socket: *..*
   Last Touched:        3:35:05
   Precedence:          Routine
   BytesIn:             0000400000      BytesOut:     0000400000
   DgramIn:             0000000100      DgramOut:     0000000100
   MaxSendLim:          0000065000      MaxRecvLim:   0000065000
   SockOpt:             00
   Multicast Specific:
     Group              IncomingIpAddress
     -----              -----------------
     224.67.113.10      9.67.113.27
     225.67.113.10      9.67.116.91
```

The onetstat -A command displays the following information for UDP sockets being used for outgoing multicast data:

- The time-to-live value
- Whether datagrams are sent to loopback
- The IP address of the link on which the datagrams are sent

The onetstat -A command displays the following information for UDP sockets being used for incoming multicast data:

- The multicast groups by way of their IP addresses (up to a maximum of 20) for which data is being received
- The IP address of the associated link

onetstat -a

The onetstat -a command displays either closed or time-wait connections, whereas the -c and -b options default to display only active TCP/IP connections:

```
onetstat -a

MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
User Id  Conn     Local Socket       Foreign Socket     State
-------  ----     ------------       --------------     -----
INETD1   0000004B 0.0.0.0..623       0.0.0.0..0         Listen
INETD1   0000004D 0.0.0.0..514       0.0.0.0..0         Listen
INETD1   0000004C 0.0.0.0..513       0.0.0.0..0         Listen
INETD1   0000004E 0.0.0.0..512       0.0.0.0..0         Listen
INETD1   00000053 0.0.0.0..37        0.0.0.0..0         Listen
INETD1   0000004F 0.0.0.0..7         0.0.0.0..0         Listen
INETD1   00000052 0.0.0.0..13        0.0.0.0..0         Listen
INETD1   00000051 0.0.0.0..19        0.0.0.0..0         Listen
INETD1   00000050 0.0.0.0..9         0.0.0.0..0         Listen
INETD1   0000004B 0.0.0.0..623       0.0.0.0..0         Listen
OSNMPD   00000014 0.0.0.0..1027      0.0.0.0..0         Listen
TCPCS    00000012 0.0.0.0..23        0.0.0.0..0         Listen
TCPCS    0000000F 127.0.0.1..1026    127.0.0.1..1025    Establsh
TCPCS    00000010 127.0.0.1..1025    127.0.0.1..1026    Establsh
TCPCS    0000000C 0.0.0.0..1025      0.0.0.0..0         Listen
TCPCS    00000045 127.0.0.1..23      127.0.0.1..1033    Establsh
TCPCS    00000039 9.67.113.27..23    9.37.81.207..1096  ClosWait
USER18   00000044 127.0.0.1..1033    127.0.0.1..23      Establsh
INETD1   00000058 0.0.0.0..37        *..*               UDP
INETD1   00000057 0.0.0.0..13        *..*               UDP
INETD1   00000056 0.0.0.0..19        *..*               UDP
INETD1   00000055 0.0.0.0..9         *..*               UDP
INETD1   00000054 0.0.0.0..7         *..*               UDP
OSNMPD   00000013 0.0.0.0..161       *..*               UDP
```

onetstat -R

The onetstat -R command queries the ARP cache for a given address or all ARP cache entries:

```
onetstat -R 9.67.112.25

  MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
  Querying ARP cache for address 9.67.112.25
  Link: TR1              IBMTR: 10005A0019F5
  Route info: 0000
```

Note: This report will not contain ARP cache data from devices which support ARP Offload. For more information about these devices, refer to *OS/390 SecureWay Communications Server: IP Configuration* and the *OS/390 SecureWay Communications Server: SNA Network Implementation Guide*.

onetstat -b

The onetstat -b command displays byte-count information about each connection:

```
onetstat -b

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS           12:34:56
 03/15/1996          MVS TCP/IP Real Time Network Monitor
 User Id  B Out      B In       L Port  Foreign Socket         State
 -------  -----      ----       ------  --------------         -----
 INETD1   0000000000 0000000000 00623   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00514   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00513   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00512   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00037   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00007   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00013   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00019   0.0.0.0..0             Listen
 INETD1   0000000000 0000000000 00009   0.0.0.0..0             Listen
 OSNMPD   0000000000 0000000000 01027   0.0.0.0..0             Listen
 TCPCS    0000000000 0000000000 00023   0.0.0.0..0             Listen
 TCPCS    0000000400 0000000436 01026   127.0.0.1..1025        Establsh
 TCPCS    0000000436 0000000400 01025   127.0.0.1..1026        Establsh
 TCPCS    0000000000 0000000000 01025   0.0.0.0..0             Listen
 TCPCS    0000025611 0000000898 00023   127.0.0.1..1033        Establsh
 TCPCS    0000000002 0000000000 00023   9.37.81.207..1096      ClosWait
 USER18   0000000898 0000025611 01033   127.0.0.1..23          Establsh
 INETD1   0000000000 0000000000 00037   *..*                   UDP
 INETD1   0000000000 0000000000 00013   *..*                   UDP
 INETD1   0000000000 0000000000 00019   *..*                   UDP
 INETD1   0000000000 0000000000 00009   *..*                   UDP
 INETD1   0000000000 0000000000 00007   *..*                   UDP
 OSNMPD   0000000216 0000000051 00161   *..*                   UDP
 Connections displayed: 23
```

You can redirect the preceding screen to a file by using the redirect function (>) in the following format:

```
onetstat -b > byteinfo
```

The file BYTEINFO is created in your home directory containing the screen output shown previously.

You can also use the onetstat -b IDLETIME command to display the idle time for each connection:

```
onetstat -b IDLETIME

 MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS         12:34:56
 03/15/1998          MVS TCP/IP Real Time Network Monitor
 User Id  B Out   B In    L Port  Foreign Socket        State     IdleTime
 -------  ------- ------- ------  --------------        -----     ----
 INETD1   0000000 0000000 00623   0.0.0.0..0            Listen    05:36:25
 INETD1   0000000 0000000 00514   0.0.0.0..0            Listen    05:36:25
 INETD1   0000000 0000000 00513   0.0.0.0..0            Listen    05:36:25
 Connections displayed: 3
```

Note: For the onetstat -b INDLETIME display, the byte outbound (B Out) and byte inbound (B In) counts are in three forms:

nnnnnnn
Number range 0–9,999,999

nnnnnnK
Number range 10,000,000–999,999,499 (K = nnnnnn x 1000)

nnnnnnM
Number range 999,999,500–4,294,967,287 (M = nnnnnn x 1,000,000)

onetstat -e

The onetstat -e command displays information about clients:

```
onetstat -e

  MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
  Current Clients:
  Client: INETD1
  Authorization: Autologged
  Last Touched:   4:01:17

  Client: TCPCS
  Authorization: None
  Last Touched:   3:14:47
```

onetstat -f

The onetstat -f command displays information about TCP/IP configuration:

```
onetstat -f

 MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
 TCP Configuration Table:
 DefaultRcvBufSize:  00016384   DefaultSndBufSize:  00016384
 DefltMaxRcvBufSize: 00262114
 MaxReTransmitTime:  120.000    MinReTransmitTime:  0.500
 RoundTripGain:      0.125      VarianceGain:       0.250
 VarianceMultiplier: 2.000      MaxSegLifeTime:     120.000
 DefaultKeepAlive:   0.120      LogProtoErr:        00
 TcpFlags:           00

 UDP Configuration Table:
 DefaultRcvBufSize:  00065535  DefaultSndBufSize:  00065535
 CheckSum:           00000000  LogProtoErr:        01
 UdpFlags:           40

 IP Configuration Table:
 Forwarding: Pkt    TimeToLive: 00064    RsmTimeOut: 00060
 FireWall:   00000  ArpTimeout: 01200    MaxRsmSize: 00000
 IgRedirect: 00000  SysplxRout: 00000    DoubleNop:  00000
 StopClawEr: 00000  SourceVipa: 00000    VarSubnet:  00000
 MultiPath:   Conn  PathMtuDsc: 00000
 DynamicXCF: 00001
   IpAddr: 9.67.116.90      SubNet: 255.255.255.128  Metric: 00

 SMF Parameters:
 InitType: 00  TermType: 00  ClientType: 00  TcpIpStats: 00

 Global Configuration Information:
 TcpIpStats: 00

 Data Trace Setting:
 JobName:  TCP      TrRecCnt: 00000010    Length: FULL
 IpAddress: *       Subnet: *
```

Notes:

1. The possible values in the Multipath field are: No, Pkt, and Conn.
2. The possible values in the Forwarding field are: No, Yes, Pkt.
3. The TCPIP statistics field under SMF Parameters displays the subtype value used when creating the SMF type 118 record (if the value is non-zero). The TCPIP statistics field under Global Configuration Information indicates whether or not the TCP/IP stack will write statistics messages to the TCP/IP job log when TCP/IP is terminated.

onetstat -C

The onetstat -C command displays information about Cache Accelerator statistics:

```
onetstat -C

 MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS      12:34:56
 Client: USER1147        Listening socket:  0.0.0.0..80
   MaxCacheSize:       0000000100  CurrCacheSize:      0000000001
   MaxNumObjects:      0000000010  CurrNumObjects:     0000000001
   NumConns:           0000000002  ConnsProcessed:     0000000001
   ConnsDeferred:      0000000001  ConnsTimedOut:      0000000000
   RequestsProcessed:  0000000001  IncompleteRequests: 0000000000
   NumCacheHits:       0000000002  NumCacheMisses:     0000000000
   NumUnprodCacheHits: 0000000000

 READY
```

Information is displayed for each listening socket configured for Cache Accelerator support.

onetstat -c

The onetstat -c command displays information about active TCP connections:

```
onetstat -c

  MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS       12:34:56
  User Id  Conn      Local Socket     Foreign Socket     State
  -------  ----      ------------     --------------     -----
  OSNMPD   00000014  0.0.0.0..1027    0.0.0.0..0         Listen
  USER18   00000044  127.0.0.1..1033  127.0.0.1..23      Establsh
  OSNMPD   00000013  0.0.0.0..161     *..*               UDP
```

onetstat -d

The onetstat -d command displays information about devices and defined links in the TCP/IP address space:

```
onetstat -d

   MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
   DevName: LOOPBACK         DevType: LOOPBACK  DevNum: 0000
     LnkName: LOOPBACK         LnkType: LOOPBACK  Status: Ready
       NetNum: 0  QueSize: 0   ByteIn: 0000053513 ByteOut: 0000053513
     BSD Routing Parameters:
       MTU Size: 00000           Metric: 00
       DestAddr: 0.0.0.0         SubnetMask: 0.0.0.0
     Multicast Specific:
       Multicast Capability: No
   DevName: LCS1              DevType: LCS       DevNum: 0120
     LnkName: TR1               LnkType: TR        Status: Ready
       NetNum: 1  QueSize: 0    ByteIn: 0000762810 ByteOut: 0000077579
       ArpMacAddress: Non-Canonical   SrBridgingCapability:  Yes
       BroadcastCapability: Yes       BroadcastType: All Rings
     BSD Routing Parameters:
       MTU Size: 00000             Metric: 00
       DestAddr: 0.0.0.0           SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
       Group             RefCnt
       -----             ------
       224.67.113.10     3
       225.67.113.10     5
  DevName: CTCD00           DevType: CTC       DevNum: 0D00
     LnkName: CTC1             LnkType: CTC       Status: Not Active
       NetNum: 1 QueSize: 0   ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000           Metric: 00
       DestAddr: 9.67.116.90     SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
   DevName: PTPDEV            DevType: MPC       DevNum: 0000
     LnkName: PTPLINK           LnkType: MPC       Status: Not Active
       NetNum: 0  QueSize: 0    ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000             Metric: 00
       DestAddr: 0.0.0.0           SubnetMask: 255.255.255.128
     Multicast Specific:
       Multicast Capability: Yes
   DevName: OSATRL90         DevType: ATM       DevNum: 0000
     LnkName: OSA90LINK1       LnkType: ATM       Status: Not Active
       NetNum: 0  QueSize: 0   ByteIn: 0000000000 ByteOut: 0000000000
     BSD Routing Parameters:
       MTU Size: 00000           Metric: 00
       DestAddr: 0.0.0.0         SubnetMask: 255.255.255.128
     ATM Specific:
       ATM portName:  OSA90
       ATM PVC Name:  STEPH            PVC Status: Not Active
       ATM LIS Name:  LIS1
       SubnetValue:   9.67.1.0         SubnetMask:    255.255.255.0
       DefaultMTU:    0000009180       InactvTimeOut: 0000000300
       MinHoldTime:   0000000060       MaxCalls:      0000001000
       CachEntryAge:  0000000900       ATMArpReTry:   0000000002
       ATMArpTimeOut: 0000000003       PeakCellRate:  0000000000
       NumOfSVCs:     0000000000
       ATMARPSV Name: ARPSV1
       VcType:        PVC              ATMaddrType:  NSAP
       ATMaddr:
       IpAddr:        0.0.0.0
     Multicast Specific:
       Multicast Capability: No
```

The Multicast Specific field is significant only for multicast-capable devices. If a link is being used to receive multicast data, then all the multicast groups, and the counts of receivers for each multicast group, are displayed. There is no limit to the number of multicast groups for which a link can receive data.

The Multicast Capability field is always Yes for the following devices: CDLC, CLAW, CTC, and MPCPTP. For LCS and MPCIPA devices, the multicast capability is only known after the device is in the Ready state. If the device is not yet Ready, the multicast capability is Unknown.

onetstat -D

The onetstat -D command drops the TCP connection specified by *n*:

```
onetstat -c
  MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS       12:34:56
  User Id  Conn      Local Socket     Foreign Socket     State
  -------  ----      ------------     --------------     -----
  PORTMP3  00010035  0.0.0.0..2220    0.0.0.0..0         Listen
  TSUSER1  00010020  0.0.0.0..1027    0.0.0.0..0         Listen
  TSUERS2  00010043  127.0.0.1..1033  127.0.0.1..23      Establsh
  PORTMP3  00021002  0.0.0.0..2221    *..*               UDP

onetstat -D 10035

 Connection successfully dropped

onetstat -c
  MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS       12:34:56
  User Id  Conn     Local Socket       Foreign Socket     State
  -------  ----     ------------       --------------     -----
  TSUSER1  00010020 0.0.0.0..1027      0.0.0.0..0         Listen
  TSUERS2  00010043 127.0.0.1..1033    127.0.0.1..23      Establsh
  PORTMP3  00021002 0.0.0.0..2221      *..*               UDP
```

onetstat -g

The onetstat -g command displays information about gateways:

```
onetstat -g

 MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS            12:34:56
 Known gateways:
 NetAddress      FirstHop      Link     Pkt Sz Subnet Mask     Subnet Value
 ----------      --------      ----     ------ -----------     ------------
 Defaultnet      9.67.113.1    TR1      576    <none>
 9.0.0.0         <direct>      TR1      2000   0.255.255.128   0.67.113.0
 9.67.116.90     <direct>      CTC1     4000   HOST
 224.0.0.5       <direct>      TR1      2000   HOST
```

You can use the DETAIL parameter to display gateway information plus MVS specific configured parameters:

```
onetstat -g detail
 MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS            12:34:56
 Known gateways:
 NetAddress      FirstHop      Link     Pkt Sz Subnet Mask     Subnet Value
 ----------      --------      ----     ------ -----------     ------------
 Defaultnet      9.67.113.1    TR1      576    <none>
   Metric:  -00000001
   MVS specific configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 9.0.0.0         <direct>      TR1      2000   0.255.255.128   0.67.113.0
   Metric:  -00000001
   MVS specific configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 9.67.116.90     <direct>      CTC1     4000   HOST
   Metric:  -00000001
   MVS Specific Configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
 224.0.0.5       <direct>      TR1      2000   HOST
   Metric:  -00000001
   MVS Specific Configured parameters:
     MaxReTransmitTime:  120.000   MinReTransmitTime: 0.500
     RoundTripGain:      0.125     Variance Gain:     0.250
     VarianceMultiplier: 2.000
```

onetstat -?

The onetstat -? command displays help information for the onetstat parameters:

```
onetstat -?

 MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS          12:34:56
 Usage: onetstat < Option | Command > < Target > < Select >
 Option:
 -A  -  Everything about a connection
 -a  -  TCP/IP connections, include TIME-WAIT and CLOSED connections
 -b  -  Current connection display
 -C  -  Cache Accelerator data
 -c  -  Active TCP/IP connections (Default option)
 -d  -  Devices and links
 -e  -  Current clients
 -f  -  Configuration data
 -g  -  Current known gateways
 -h  -  Home address list
 -j  -  SLA Policy data
 -o  -  Display port reservation list
 -R  -  Query ARP table or entry information
 -r  -  Display routing information in a standard fashion
 -s  -  Socket interface users and their sockets
 -t  -  Telnet connection information
 -u  -  Date and time tcpip was last started
 -v  -  Dynamic VIPA information
 -?  -  Onetstat information list
 Target option:
 -p  -  Displays detailed information about the specified TCPIP
        address space
 Select-String:
 -E     For -A, -a, -b, -c, -e and -t, select information on the specified
         client name.
 -I     For -a, -c, -g and -t, select information on the specified IP address
 -P     For -a, -c and -t, select information on the specified port.
 -N     For -t, select information on the specified application name
 -L     For -t, select information on the specified LU name
 Command:
 -D  -  Drop a TCP/IP connection
```

onetstat -h

The onetstat -h command displays the HOME list:

```
onetstat -h

 MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS       12:34:56
 Home address list:
 Address           Link          Flg
 -------           ----          ---
 9.67.113.27       TR1           P
 9.67.116.91       CTC1
 9.67.116.99       ETH2
 9.67.1.8          OSA90LINK1
 9.67.116.1        X25LINK
 9.67.116.2        CDLCLINK
 9.67.116.3        HCHLINK
 9.67.116.4        PTPLINK
 9.67.116.5        SNA1
 9.67.116.6        SNA62
 127.0.0.1         LOOPBACK
```

onetstat -o

The onetstat -o command displays port reservation list:

```
onetstat -o

 MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPCS       12:34:56
 Port# Prot User     Flags Range
 ----- ---- ----     ----- -----
 00020 TCP  OMVS     A
 00021 TCP  OMVS     A
 00025 TCP  SMTP     A
 00999 TCP  IOASRV   A
 02026 TCP  IOASNMP  AO
 03035 TCP  OMVS     AS
 05000 UDP  APPL1    AR    05000-05009
 05001 UDP  APPL1    AR    05000-05009
 05002 UDP  APPL1    AR    05000-05009
 05003 UDP  APPL1    AR    05000-05009
 05004 UDP  APPL1    AR    05000-05009
 05005 UDP  APPL1    AR    05000-05009
 05006 UDP  APPL1    AR    05000-05009
 05007 UDP  APPL1    AR    05000-05009
 05008 UDP  APPL1    AR    05000-05009
 05009 UDP  APPL1    AR    05000-05009
```

onetstat -r

The onetstat -r command displays routing information:

```
onetstat -r

 MVS TCP/IP onetstat CS V2R8     TCPIP Name: TCPCS          12:34:56
 Destination     Gateway         Flags  Refcnt  Interface
 -----------     --------        -----  ------  ---------
 Defaultnet      9.67.113.1      UG     000001  TR1
 9.67.113.0      0.0.0.0         U      000000  TR1
 9.67.116.90     0.0.0.0         H      000000  CTC1
 224.0.0.5       0.0.0.0         UH     000000  TR1
```

onetstat -j

The onetstat -j command provides information about service policy definitions and policy statistics:

```
 onetstat -j

MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS2           17:33:10
Policy:
  Profile:   DataPol
    Scope:             DataTraffic         Direction:        Both
    Permission:        Allowed             Protocol:         TCP
    LocalInterface:    0.0.0.0             Position:         0000000000
    SourceIpFrom:      0.0.0.0             SourceIpTo:       0.0.0.0
    SourcePortFrom:    008000              SourcePortTo:     008000
    DestIpFrom:        9.67.200.0          DestIpTo:         9.67.200.255
    DestPortFrom:      000000              DestPortTo:       000000
    ServiceClass:      DataCat
      Scope:           DataTraffic         OutgoingTOS:      C0
      Priority:        0                   Interface:        0.0.0.0
      MaxRate:         0001000000          MinRate:          0000010000
      MaxDelay:        0000000000          MaxConn:          0000000000
    Performance Information:
      FirstActTime:    20:28:12            LastMapTime:      20:31:08
      TotalBytesIn:    0000000000          TotalBytesOut:    0000125004
      BytesInDiscard:  0000000000          BytesOutDiscard:  0000000000
      TotalInPackets:  0000000216          TotalOutPackets:  0000000326
      ActConnMap:      0000000000          MaxConnLimit:     0000000000
      AcceptConn:      0000000002          DeniedConn:       0000000000
Policy:    JKF
  Profile:   CLPol
    Scope:             RSVP                Direction:        Outgoing
    Permission:        Allowed             Protocol:         0
    LocalInterface:    0.0.0.0             Position:         0000000000
    SourceIpFrom:      0.0.0.0             SourceIpTo:       0.0.0.0
    SourcePortFrom:    000000              SourcePortTo:     000000
    DestIpFrom:        0.0.0.0             DestIpTo:         0.0.0.0
    DestPortFrom:      000000              DestPortTo:       000000
    ServiceClass:      CLCat
      Scope:           DataTraffic         OutgoingTOS:      E0
      Priority:        0                   Interface:        0.0.0.0
      MaxRate:         0000000000          MinRate:          0000000000
      MaxDelay:        0000000000          MaxConn:          0000000000
    Performance Information:
      FirstActTime:    20:28:12            LastMapTime:      20:31:29
      TotalBytesIn:    0000000000          TotalBytesOut:    0000026960
      BytesInDiscard:  0000000000          BytesOutDiscard:  0000000000
      TotalInPackets:  0000000049          TotalOutPackets:  0000000072
      ActConnMap:      0000000000          MaxConnLimit:     0000000000
      AcceptConn:      0000000000          DeniedConn:       0000000000
```

onetstat -s

The onetstat -s command provides information about each client using the socket interface:

```
onetstat -s

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS          12:34:56
 Sockets interface status:
 Type   Bound to               Connected to          State    Conn
 ====   ========               ============          =====    ====
 Name: INETD1    Subtask: 006DB5B8
 Dgram  0.0.0.0..37            *..*                  UDP      00000058
 Dgram  0.0.0.0..13            *..*                  UDP      00000057
 Dgram  0.0.0.0..19            *..*                  UDP      00000056
 Dgram  0.0.0.0..9             *..*                  UDP      00000055
 Dgram  0.0.0.0..7             *..*                  UDP      00000054
 Stream 0.0.0.0..623           0.0.0.0..0            Listen   0000004B
 Stream 0.0.0.0..514           0.0.0.0..0            Listen   0000004D
 Stream 0.0.0.0..513           0.0.0.0..0            Listen   0000004C
 Stream 0.0.0.0..512           0.0.0.0..0            Listen   0000004E
 Stream 0.0.0.0..37            0.0.0.0..0            Listen   00000053
 Stream 0.0.0.0..7             0.0.0.0..0            Listen   0000004F
 Stream 0.0.0.0..13            0.0.0.0..0            Listen   00000052
 Stream 0.0.0.0..19            0.0.0.0..0            Listen   00000051
 Stream 0.0.0.0..9             0.0.0.0..0            Listen   00000050
 Name: OSNMPD    Subtask: 006DBA70
 Dgram  0.0.0.0..161           *..*                  UDP      00000013
 Stream 0.0.0.0..1027          0.0.0.0..0            Listen   00000014
 Name: TCPCS     Subtask: 00000000
 Stream 127.0.0.1..23          127.0.0.1..1033       Estblsh  00000045
 Stream 9.67.113.27..23        9.37.81.207..1096     ClosWait 00000039
 Name: TCPCS     Subtask: 006C57B0
 Stream 0.0.0.0..23            0.0.0.0..0            Listen   00000012
 Name: TCPCS     Subtask: 006D56F0
 Stream 127.0.0.1..1026         127.0.0.1..1025      Establsh 0000000F
 Name: TCPCS     Subtask: 006D5CF0
 Stream 0.0.0.0..1025          0.0.0.0..0            Listen   0000000C
 Stream 127.0.0.1..1025        127.0.0.1..1026       Establsh 00000010
 Name: USER18    Subtask: 006A3400
 Stream 127.0.0.1..1033        127.0.0.1..23         Establsh 00000044
```

onetstat -p

The onetstat -p command displays detailed information about the specified TCP/IP name. The output can be formatted in different ways, depending on the specific command you used. This command returns information about the TCP/IP address space that was specified on the command.

The following example shows information about active TCP/IP connections for the TCP/IP address space TCPIPA.

```
onetstat -p TCPIPA

 MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPIPA           12:34:56

User Id  Conn     Local Socket            Foreign Socket           State
-------  ----     ------------            --------------           -----
CFTPSERV 00010040 0.0.0.0..5097           0.0.0.0..0               Listen
MISCSRV  00010031 0.0.0.0..7              0.0.0.0..0               Listen
MISCSRV  00010051 0.0.0.0..9              0.0.0.0..0               Listen
TSUSER5  00010010 0.0.0.0..33333          0.0.0.0..0               Listen
```

The following example shows an onetstat -p command issued for the TCP/IP address space TCPIPA, an address that is not the default. The name of the TCP/IP address space must be included in the command.

```
onetstat -R  9.67.112.25 -p TCPIPA

MVS TCP/IP onetstat CS V2R8      TCPIP Name: TCPIPA           12:34:56
Querying ARP cache for address 9.67.112.25
Link: TR1               IBMTR: 10005A0019F5
Route info: 0000
```

onetstat -t

The onetstat -t command displays the status of the internal Telnet server:

```
onetstat -t

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS            12:34:56
 Internal Telnet Server Status:
 Conn     Foreign Socket       State    BytesIn  BytesOut ApplName LuName
 ----     --------------       -----    -------  -------- -------- ------
 00000045 127.0.0.1..1033      Establsh 00000898 00025611 TSO10002 TCPM1001
```

The onetstat -t DETAIL command displays additional server status information:

```
onetstat -t detail

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS            12:34:56
 Internal Telnet Server Status:
 Conn     Foreign Socket       State    BytesIn  BytesOut ApplName LuName
 ----     --------------       -----    -------  -------- -------- ------
 00000045 127.0.0.1..1033      Establsh 00000898 00025611 TSO10002 TCPM1001
   ModeName: NSX32702  TNProto:  TN3270    TnUserID:
```

onetstat -u

The onetstat -u command displays the date and time that TCP/IP was started:

```
onetstat -u

 MVS TCP/IP onetstat CS V2R8       TCPIP Name: TCPCS          12:34:56
 Tcpip started at 09:08:15 on 07/09/1997
```

onetstat -v

The onetstat -v command displays the current dynamic VIPA information for a local host:

```
onetstat -v

   MVS TCP/IP onetstat CS V2R8    TCPIP Name: TCPCS       12:34:56

   IP Address       AddressMask      Status   Origination
   ----------       ----------       ------   -----------
   9.67.102.30      255.255.255.128  Active   VIPARange Bind
   9.67.102.31      255.255.128.0    Backup   VIPABackup
   9.67.102.32      <NONE>           Backup   VIPABackup
   9.67.113.27      255.255.255.0    Active   VIPADEFine
   9.67.113.28      255.255.0.0      Backup   VIPADEFine Contention
   9.67.116.10      255.255.0.0      Active   VIPARange IOCTL
```

OS/390 UNIX Netstat Format Compared with TSO Netstat Format

The following table shows the equivalent OS/390 UNIX and TSO command formats.

Table 34. OS/390 UNIX and TSO Netstat Command Formats

Description	TSO Option	OS/390 UNIX Option
Options		
Displays detailed information about TCP/IP connection	ALL	-A
Displays information for all TCP/IP connections, including recently closed ones	ALLConn	-a
Displays ARP cache information	ARp *ipaddr*	-R *ipaddr*
Displays byte count information about each connection	BYTEinfo	-b
Displays Cache Acclerator information	CACHinfo	-C
Displays information about each client	CLients	-e
Displays configuration data	CONFIG	-f
Displays information about each active TCP/IP connection	COnn	-c
Displays information about devices and defined links in the TCP/IP address space	DEvlinks	-d
Displays information about each gateway	Gate	-g
Displays the HELP information	HElp	-?
Displays the HOME information	Home	-h
Displays the reserved port list	PORTList	-o
Displays routing information	ROUTe	-r
Displays information about service policy definitions and policy statistics	SLAP	-j
Displays information about each client using the socket interface	SOCKets	-s
Displays the status of the internal Telnet server	TELnet	-t
Displays the date and time that TCP/IP was started	Up	-u
Displays the current dynamic VIPA information for a local host	VIPADyn	-v
Target Options		
Displays information about the specified TCP/IP address space	TCp *tcpname*	-p *tcpname*
Output Options		
Causes the output to be stored in the data set userid.NETSTAT.option	REPort	n/a
Causes the output to be placed in the program stack	STACk	n/a
Select Strings		
Provides the responses of -A, -a, -b, -c, -e, and -t on the specified *client id*		-E *clientname*
Provides the responses of TSO option ALL, ALLConn, BYTEinfo, COnn, CLients, and TELnet on the specified *client id*	CLIent *clientname*	

Table 34. OS/390 UNIX and TSO Netstat Command Formats (continued)

Description	TSO Option	OS/390 UNIX Option
Provides the responses of -a, -c, -g, and -t on the specified *ipaddr*		-I *ipaddr*
Provides the responses of the TSO option ALLConn, COnn, Gate, and TELnet on the specified *ipaddr*	IPAddr *ipaddr*	
Provides the responses of -A, -c, and -t on the specified port number *portnum*		-P *portnum*
Provides the responses of TSO option ALL, COnn, and TELnet on the specified port number *portnum*	POrt *portnum*	
Provides the responses of -t on the specified VTAM Application name *applname*		-N *applname*
Provides the responses of TSO option TELnet on the specified VTAM Application name *applname*	APPLname *applname*	
Provides the responses of -t on the specified *luname*		-L *luname*
Provides the responses of TSO option TELnet on the specified *luname*	LUName *luname*	
Command		
Drops the TCP/IP connection specified by *n*	DRop *n*	-D *n*

The TSO PING Command—Send an Echo Request

Purpose

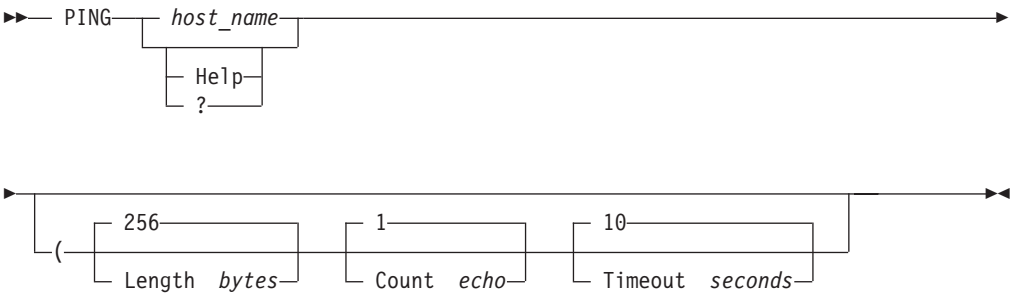
The PING command sends an echo request to a node to determine whether the computer is accessible.

When a response to a PING command is received, the elapsed time is displayed. The time does not include the time spent communicating between the user and the TCP/IP address space.

Note: The TSO PING command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND0D6 can occur.

For information about the remote PING function, which allows a user at one host to determine the response time between two remote hosts using SNMP, see "Chapter 13. Managing TCP/IP Network Resources with SNMP" on page 501.

Format



Note: The minimum abbreviation for each parameter is shown as uppercase letters.

Parameters

host_name
Specifies the local or remote host to which you want to send the echo request. If you omit the *host_name*, the system prompts you for a host name. The host name is either a character-string name or the IP address in dotted decimal format.

Help or ?
Provides help information about the PING command. You cannot place the HELP parameter on the PING command line with other parameters.

Length *bytes*
Sets the number of bytes of the echo request. If *bytes* is not specified, an error occurs. If you do not specify the Length parameter, the default of 256 is used. The number of bytes must be in the range 8—65507.

Count *echo*
Sets the number of echo requests that are sent to the host. If you do not specify the Count parameter, the default of 1 is used. If *echo* is not specified,

an error occurs. The number *echo* must be in the range 0-2^{31}—1, which is 2147483647. If *echo* is 0, the PING command sends echo requests continually. To stop the PING command, press **PA1** .

Timeout *seconds*
Sets the number of seconds that the PING command waits for a response. If you do not specify the Timeout parameter, the default of 10 seconds is used. If *seconds* is not specified, an error occurs. The number of *seconds* must be in the range 1—100.

Usage

- To stop or interrupt the PING command, press the **PA1** key.
- You can place more than one parameter on the PING command line; however, the HELP parameter is an exception and cannot be placed on the PING command line with other parameters.

The OS/390 UNIX oping/ping Command—Send an Echo Request

Purpose

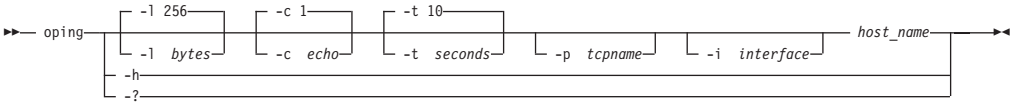
The OS/390 UNIX oping command sends an echo request to a foreign node (remote node) to determine whether the computer is accessible.

When a response to an oping command is received, the elapsed time is displayed. The time does not include the time spent communicating between the user and the TCP/IP address space.

Use the oping command to determine the accessibility of the foreign node.

Note: ping is a synonym for the oping command in the OS/390 UNIX shell. ping command syntax is the same as that for the oping command.

Format



Parameters

-l *bytes*
Sets the number of bytes of the echo request. If *bytes* is not specified, an error occurs. If you do not specify the -l parameter, the default of 256 is used. The number of bytes must be in the range 8—65507.

-c *echo*
Sets the number of echo requests that are sent to the host. If you do not specify the -c parameter, the default of 1 is used. If *echo* is not specified, an error occurs. The *echo* value must be in the range 0-2^{31}—1, which is 2147483647. If *echo* is 0, the oping command sends echo requests continually. To stop the oping command, see “Usage” on page 421.

-t *seconds*
Sets the number of seconds that the oping command waits for a response. If you do not specify the -t parameter, the default of 10 seconds is used. If *seconds* is not specified, an error occurs. The number of *seconds* must be in the range 1—100.

-p *tcpname*
Specifies the TCP/IP to be used.

The *tcpname* is an 8-byte procedure name that is used to start the TCP/IP. When the ’S member.identifier’ method of starting TCP/IP is used, the value specified for *identifier* must be used as *tcpname*.

-i *interface*
Specifies the local interface over which the oping packets will be sent. The interface is either a maximum 16-byte link name from a LINK Profile statement, or the IP address of the local interface from a HOME Profile statement. Local VIPA or LOOPBACK interfaces are invalid. This parameter is only valid when

the packets are sent by way of a CS for OS/390 TCP/IP stack. This parameter is independent of the source IP address used in the outbound packets.

Note: As a diagnostics aid in analyzing response times and path availability using a particular route, this parameter routes packets over specified interfaces regardless of the multipath settings in the IPCONFIG MULTIPATH Profile statement by bypassing the outbound path selection algorithm for the packets.

host_name
Specifies the local or remote host to which you want to send the echo request. If you omit the *host_name*, the system prompts you for a host name. The host name is either a character-string name or the IP address in the standard format of the host.

-h or -?
Provides help information about the oping command. You cannot place the -h or -? parameter on the oping command line with other parameters.

Usage

- To stop or interrupt the oping command, press Ctrl c . The interrupt key can be changed by using the OMVS ESCAPE command in the OS/390 UNIX shell, or the stty command for the RAW shell. For more information about OMVS and stty commands, refer to the *OS/390 UNIX System Services Command Reference*.
- You can place more than one parameter on the oping command line; however, the -h and -? parameters are exceptions and cannot be placed on the oping command line with other parameters.

TSO PING and OS/390 UNIX oping Command Return Codes

The following is a list of the return codes generated by the TSO PING and OS/390 UNIX oping commands:

Code	Description
0	Response
4	No response
8	TCP/IP address space failure (TSO PING only)
12	Socket API failure (OS/390 UNIX oping only)
100	Incorrect parameter

When a response to an TSO PING or OS/390 UNIX oping command is received, the elapsed time is displayed. The time does not include the time spent communicating between the user and TCP/IP address space.

Resolving TSO PING and OS/390 UNIX oping/ping Command Problems

A host may fail to respond even after several PING/oping commands for any of the following reasons:

- The host is not listening to the network.
- The host is inoperative, or some network or gateway leading from the user to the host is inoperative.
- The host is slow because of activity.
- The packet is too large for the host.

The echo request sent by the PING/oping command does not guarantee delivery. More than one PING/oping command should be sent before you assume that a communication failure has occurred.

Use additional PING/oping commands to communicate with other hosts in the network to determine the condition that is causing the communication failure. However, you should know the network topology to determine the location of the failure. Issue the PING/oping commands in the following order until the failure is located.

1. Send a PING/oping command to your local host.

   A successful PING/oping command sent to a different host on the same network as the original host suggests that the original host is down, or is not listening to the network.
2. Send a PING/oping command to a host other than your local host on your local network.
3. Send a PING/oping command to each intermediate node that leads from your local host to the remote host, starting with the node closest to your local host.

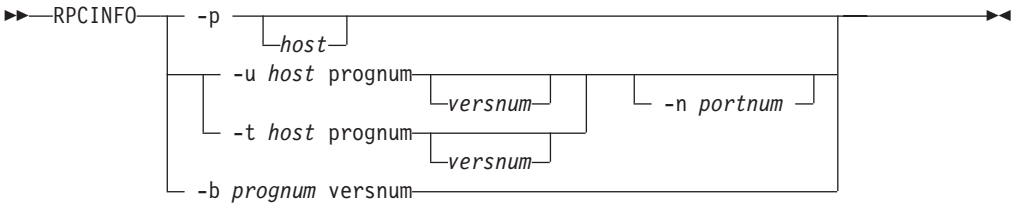
   If you cannot get echoes from any host on that network, the trouble is usually somewhere along the path to the remote hosts. Direct a PING/oping command to the gateway leading to the network in question. If the PING/oping command fails, continue to test along the network from the target, until you find the point of the communication breakdown.

The TSO RPCINFO Command—Display Server Information

Purpose

Use the RPCINFO command to display the servers that are registered and operational with any portmapper on your network. The RPCINFO command makes a remote procedure call (RPC) to an RPC server and displays the results.

Format



Parameters

-p *host*
Queries the portmapper on the specified host and prints a list of all registered RPC programs. If *host* is not specified, the system defaults to the local host name. For more information about how the local host name is defined, refer to *OS/390 SecureWay Communications Server: IP Configuration*.

-u *host prognum versnum*
Sends an RPC call to procedure zero of *prognum* on the specified host using UDP, and reports whether a response is received. The variable *prognum* is the name or number of the RPC program.

-n *portnum*
Specifies the port number to be used for the -t and -u options in place of the port number that is given by the portmapper.

-t *host prognum versnum*
Sends an RPC call to procedure zero of *prognum* on the specified host using TCP, and reports whether a response is received.

-b *prognum versnum*
Sends an RPC broadcast to procedure zero of the specified *prognum* and *versnum* using UDP, and reports all hosts that respond.

Usage

- The version number is required for the -b parameter. If a version is specified, RPCINFO attempts to call that version of the specified program. If a version is not specified, RPCINFO prints error information.
- RPCINFO -b (broadcast) displays information with the same network only. The broadcast packets do not pass through gateways.
- RPCINFO -b (broadcast) works only for the UDP transport services and does not find any TCP-based services.

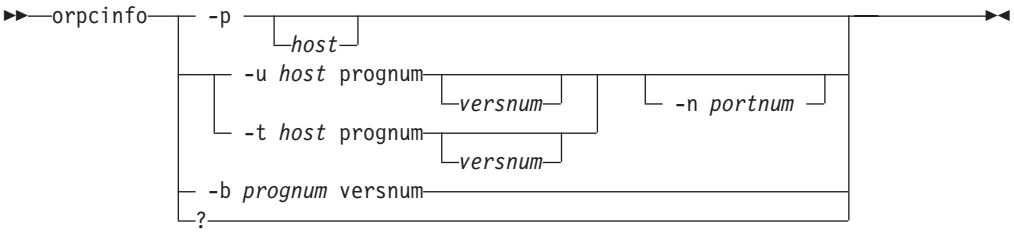
The OS/390 UNIX orpcinfo/rpcinfo Command—Display Server Information

Purpose

Use the orpcinfo command to display the servers that are registered and operational with any portmapper on your network. The orpcinfo command makes a remote procedure call (RPC) to an RPC server and displays the results.

Note: rpcinfo is a synonym for the orpcinfo command in the OS/390 UNIX shell. rpcinfo command syntax is the same as that for the orpcinfo command.

Format



Parameters

-p *host*
Queries the portmapper on the specified host and prints a list of all registered RPC programs. If *host* is not specified, the system defaults to the local host name. For more information about how the local host name is defined, refer to *OS/390 SecureWay Communications Server: IP Configuration*.

-u *host prognum versnum*
Sends an RPC call to procedure zero of *prognum* on the specified host using UDP, and reports whether a response is received. The variable *prognum* is the name or number of the RPC program.

-n *portnum*
Specifies the port number to be used for the -t and -u options in place of the port number that is given by the portmapper.

-t *host prognum versnum*
Sends an RPC call to procedure zero of *prognum* on the specified host using TCP, and reports whether a response is received.

-b *prognum versnum*
Sends an RPC broadcast to procedure zero of the specified *prognum* and *versnum* using UDP, and reports all hosts that respond.

-? Specifies the command help.

Usage

- The version number is required for the -b parameter. If a version is specified, RPCINFO attempts to call that version of the specified program. If a version is not specified, RPCINFO prints error information.

- OS/390 UNIX orpcinfo -b (broadcast) displays information with the same network only. The broadcast packets do not pass through gateways.
- OS/390 UNIX orpcinfo -b (broadcast) works only for the UDP transport services and does not find any TCP-based services.

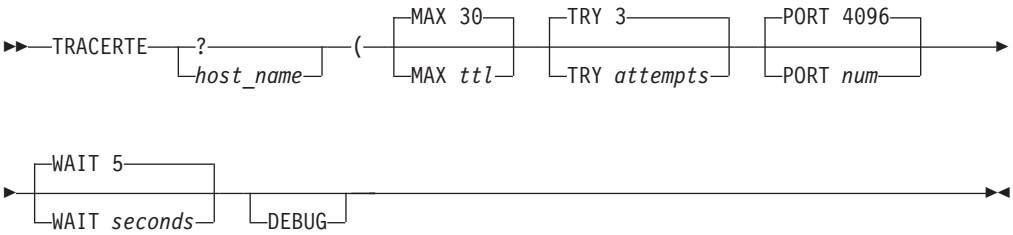
The TSO TRACERTE Command—Debug Network Problems

Purpose

TRACERTE is useful for debugging various network problems. TRACERTE sends UDP requests with varying TTL (time-to-live) values and then waits for the routers between the local and remote hosts to send TTL-exceeded messages.

Note: The TSO TRACERTE command uses the Pascal socket API, so VMCF must be started for the command to be successful. If VMCF is not started, an ABEND06D can occur.

Format



Parameters

host_name
Specifies the target of the TRACERTE command. This must be an IP address, or a host name that can be resolved.

MAX *ttl*
Specifies the maximum time to live (TTL). The range for valid values is 1–255. The default is 30.

TRY *attempts*
Specifies the number of attempts. The range for valid values is 1–20. The default is 3.

PORT *num*
Specifies the starting port number. The range for valid values is 2048–60000. The default is 4096.

WAIT *seconds*
Specifies how long to wait for a response. The range for valid values is 1–255. The default is 5 seconds.

DEBUG
Specifies that extra messages are to be printed.

Results

TRACERTE displays one line of output for every TTL value for which it sent a UDP probe packet. The format of the output is as follows:

```
TTL NAME (IP_ADDRESS) NUM ms FLAG
```

The values displayed are:

TTL	The time-to-live value (sometimes referred to as a hop count) used in the probe packets sent by TRACERTE.
NAME	If the source IP address in the received Internet Control Message Protocol (ICMP) response can be found in the host site tables, displays the name associated with the source IP address.
IP_ADDRESS	The source IP address from the received ICMP response.
NUM	The elapsed time between when the probe packet was sent out and when the ICMP response to that probe packet was received.
FLAG	This is an optional field. It is only present if one of the events described below occurs.

Flag	Indicates
*	No datagram was received before you timed out. The hop might not respond with ICMP.
F	The packet needs to be fragmented.
H	The destination host is unreachable.
N	The destination network is unreachable.
P	The destination protocol is unreachable.
Q	The destination host is reachable, but cannot accept the packet because the queue is full.
S	The route supplied for the message was incorrect.
?	An ICMP packet was received for the same target IP address which Traceroute is trying to reach, but the ICMP type in the packet is not destination unreachable, time exceeded, or source quench.

For a list of the ICMP types associated with the preceding Flags, see “Appendix I. ICMP Types and Codes” on page 609.

Examples

Note: In these examples, an asterisk (*) represents a lost packet.

- The second hop in this example does not send TTL-exceeded messages.

```
tracerte cyst.watson.ibm.com
Trace route to CYST.WATSON.IBM.COM (9.2.91.34)
1 (9.67.22.2) 67 ms 53 ms 60 ms
2 * * *
3 (9.67.1.5) 119 ms 83 ms 65 ms
4 (9.3.8.14) 77 ms 80 ms 87 ms
5 (9.158.1.1) 94 ms 89 ms 85 ms
6 (9.31.3.1) 189 ms 197 ms *
7 * * (9.31.16.2) 954 ms
8 (129.34.31.33) 164 ms 181 ms 216 ms
9 (9.2.95.1) 198 ms 182 ms 178 ms
10 (9.2.91.34) 178 ms 187 ms *
```

- Sometimes packets are lost (hops 6, 7, and 10).

```
Ready;
tracerte 129.35.130.09
Trace route to 129.35.130.09 (129.35.130.9)
1 (9.67.22.2) 61 ms 62 ms 56 ms
2 * * *
3 (9.67.1.5) 74 ms 73 ms 80 ms
4 (9.3.8.1) 182 ms 200 ms 184 ms
5 (129.35.208.2) 170 ms 167 ms 163 ms
6 * (129.35.208.2) 192 ms !H 157 ms !H
```

- The network was found, but no host was found. The packet could not route to that network.

```
Ready;
tracerte 129.45.45.45
Trace route to 129.45.45.45 (129.45.45.45)
1 (9.67.22.2) 320 ms 56 ms 71 ms
2 * * *
3 (9.67.1.5) 67 ms 64 ms 65 ms
4 (9.67.1.5) 171 ms !N 68 ms !N 61 ms !N
```

- TRACERTE uses the site tables for inverse name resolution rather than the domain name server. If a host name is found in the site table, it is printed along with its IP address.

```
tracerte EVANS
Trace route to EVANS (129.45.45.45)
1 BART (9.67.60.85) 20 ms 56 ms 71 ms
2 BUZZ (9.67.60.84) 55 ms 56 ms 54 ms
3 EVANS (9.67.30.25) 67 ms 64 ms 65 ms
```

Usage

- For TRACERTE to be authorized to use RAW sockets, member IKJTSOxx of SYS1.PARMLIB must be updated. Refer to *OS/390 SecureWay Communications Server: IP Migration* for more information.
- The range of port numbers TRACERTE uses are normally not valid but can be changed if the target host is using a nonstandard UDP port.
- TRACERTE requires that the HOSTS.ADDRINFO dataset be defined. Refer to *OS/390 SecureWay Communications Server: IP Configuration* for more information about using TRACERTE.
- TRACERTE does not support use of the loopback address.

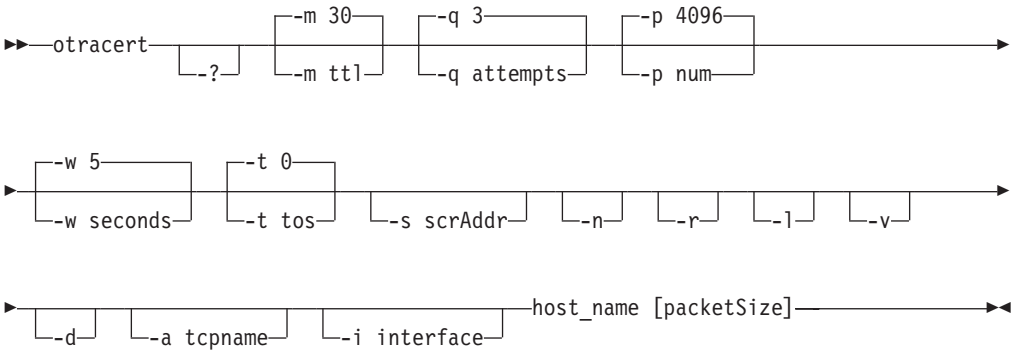
The OS/390 UNIX otracert/traceroute Command—Debug Network Problems

Purpose

This command is useful for debugging various network problems. The OS/390 UNIX otracert function sends UDP requests with varying TTL (time to live) values and then waits for the routers between the local and remote hosts to send TTL-exceeded messages.

Note: traceroute is a synonym for the otracert command in the OS/390 UNIX shell. traceroute command syntax is the same as that for the otracert command.

Format



Parameters

-? Specifies the command help.

-m *ttl*
Specifies the maximum time to live. The range for valid values is 1—255. The default is 30.

-q *attempts*
Specifies the number of times that a probe is sent with the same time-to-live value. This number reflects the total probe transmision (success or failure) per time-to-live increment. The range is 1—20. The default is 3.

-p *num*
Specifies the starting port number. The range for valid values is 2048—60000. The default is 4096.

-w *seconds*
Specifies how long to wait for a response. The range for valid values is 1—255. The default is 5 seconds.

-t *tos*
Specifies the type-of-service (tos) in the probe packets. The range for valid values is 0—255. The default is 0.

-s *scrAddr*
Specifies the source IP address. You must specify this address as an IP

number and not a host name. On hosts with more than one IP address, you can set the source address to the IP address for another one of this machine interface addresses.

-n Specifies the print hop address. This address is numeric and saves a nameserver address-to-name lookup for each gateway on the path.

-r Bypasses the normal routing tables and sends information directly to a host in an attached network. If the host is not in an adjacent network, an error is returned. You can use this option to ping a local host through an interface that has no active route.

-l Displays the time-to-live value from each received packet. This value can be used to help detect asymmetric routing.

-v Specifies that additional information is to be displayed.

-d Specifies that extra messages and other debugging information are to be displayed.

-a *tcpname*
Specifies the name of the TCP/IP stack to be used to send the probe packets. The *tcpname* is an 8-byte procedure name that is used to start TCP/IP. When the 'member.identifier' method of starting TCP/IP is used, the value specified for *identifier* must be used as tcpname.

When the -a option is not specified and OS/390 UNIX is configured for CINET, the CINET Prerouter selects the TCP/IP stack to which the request is routed.

-i *interface*
Specifies the local interface over which the otracert packets will be sent. The interface is either a maximum 16-byte link name from a LINK Profile statement, or the IP address of the local interface from a HOME Profile statement. Local VIPA or LOOPBACK interfaces are invalid. This parameter is only valid when the packets are sent by way of a CS for OS/390 TCP/IP stack. This parameter is independent of the -s parameter used as the source IP address in the outbound packets.

Note: As a diagnostics aid in analyzing response times and path availability using a particular route, this parameter routes packets over specified interfaces regardless of the multipath settings in the IPCONFIG MULTIPATH Profile statement by bypassing the outbound path selection algorithm for the packets.

host_name
Specifies the target of the otracert command. This must be an IP address, or a host name that can be resolved.

packetSize
Optional parameter that can be used to determine the size of a probe. The number of bytes specified is added to the probe packet and may affect the route of a probe. The packet size must be in the range 1—65495 (max IP packet, 65535 minus the size of probe packet header, 40).

Results

otracert displays one line of output for every TTL value for which it sent a UDP probe packet. The format of the output is as follows:

```
TTL NAME (IP_ADDRESS) NUM ms FLAG
```

The values displayed are:

TTL	The time-to-live value (sometimes referred to as a hop count) used in the probe packets sent by otracert.
NAME	If the source IP address in the received Internet Control Message Protocol (ICMP) repsonse can be found in the host site tables, displays the name associated with the source IP address.
IP_ADDRESS	The source IP address from the received ICMP response.
NUM	The elapsed time between when the probe packet was sent out and when the ICMP response to that probe packet was received.
FLAG	This is an optional field. It is only present if one of the events described below occurs.

Flag	Indicates
*	No datagram was received before you timed out. The hop might not respond with ICMP.
F	The packet needs to be fragmented.
H	The destination host is unreachable.
N	The destination network is unreachable.
P	The destination protocol is unreachable.
Q	The destination host is reachable, but cannot accept the packet because the queue is full.
S	The route supplied for the message was incorrect.
?	An ICMP packet was received for the same target IP address which otracert is trying to reach, but the ICMP type in the packet is not destination unreachable, time exceeded, or source quench.

For a list of the ICMP types associated with the preceding Flags, see “Appendix I. ICMP Types and Codes” on page 609.

Examples

In these examples, an asterisk (*) represents a lost packet.

- The second hop in this example does not send TTL-exceeded messages.

```
otracert cyst.watson.ibm.com
Trace route to CYST.WATSON.IBM.COM (9.2.91.34)
Use escape C sequence to interrupt
1 (9.67.22.2) 67 ms 53 ms 60 ms
2 * * *
3 (9.67.1.5) 119 ms 83 ms 65 ms
4 (9.3.8.14) 77 ms 80 ms 87 ms
5 (9.158.1.1) 94 ms 89 ms 85 ms
6 (9.31.3.1) 189 ms 197 ms *
7 * * (9.31.16.2) 954 ms
8 (129.34.31.33) 164 ms 181 ms 216 ms
9 (9.2.95.1) 198 ms 182 ms 178 ms
10 (9.2.91.34) 178 ms 187 ms *
```

- Sometimes packets are lost (hops 6,7, and 10).

```
otracert 129.35.130.09
Trace route to 129.35.130.09 (129.35.130.9)
Use escape C sequence to interrupt
1 (9.67.22.2) 61 ms 62 ms 56 ms
2 * * *
3 (9.67.1.5) 74 ms 73 ms 80 ms
4 (9.3.8.1) 182 ms 200 ms 184 ms
5 (129.35.208.2) 170 ms 167 ms 163 ms
6 * (129.35.208.2) 192 ms !H 157 ms !H
```

- The network was found, but no host was found. The packet could not route to that network.

```
otracert 129.45.45.45
Trace route to 129.45.45.45 (129.45.45.45)
Use escape C sequence to interrupt
1 (9.67.22.2) 320 ms 56 ms 71 ms
2 * * *
3 (9.67.1.5) 67 ms 64 ms 65 ms
4 (9.67.1.5) 171 ms !N 68 ms !N 61 ms !N
```

- OS/390 UNIX otracert uses the site tables for inverse name resolution rather than the domain name server. If a host name is found in the site table, it is printed along with its IP address.

```
otracert EVANS
Trace route to EVANS (129.45.45.45)
Use escape C sequence to interrupt
1 BART (9.67.60.85) 20 ms 56 ms 71 ms
2 BUZZ (9.67.60.84) 55 ms 56 ms 54 ms
3 EVANS (9.67.30.25) 67 ms 64 ms 65 ms
```

Usage

- The range of port numbers OS/390 UNIX otracert uses is normally not valid but can be changed if the target host is using nonstandard UDP port.
- OS/390 UNIX otracert does not support use of the loopback address.

Context

Refer to *OS/390 SecureWay Communications Server: IP Diagnosis* for information about using OS/390 UNIX otracert to debug network connectivity problems.

Chapter 12. Querying the Domain Name System (DNS)

This chapter describes the Domain Name System (DNS) domain names, domain name servers, resolvers, and resource records. It also provides descriptions of the NSLOOKUP, onslookup, NSUPDATE, and DIG commands used to query name servers, and the hostname, dnsdomainname, and domainname commands used to display the local DNS host name and domain name.

Overview of the Domain Name System

TCP/IP applications map domain names to 32-bit IP addresses to identify network nodes. Mapping must be consistent across the network to ensure interoperability. The DNS provides this mapping through network nodes called Domain Name Servers. The DNS can provide additional information about nodes and networks, including the TCP/IP services available at a node and the location of name servers in a network.

The DNS defines a special domain called `in-addr.arpa` to translate IP addresses to domain names. An `in-addr.arpa` name is composed of the reverse octet order of an IP address concatenated with the `in-addr.arpa` string. For example, a host named `Host1` has `9.67.43.100` as an IP address. The `in-addr.arpa` domain translates the `Host1` IP address `9.67.43.100` to `100.43.67.9.in-addr.arpa`.

The TCP/IP for MVS Domain Name System includes a name server and a resolver API for application programs. See “Domain Name Servers” on page 434 and “Resolvers” on page 435 for more information about these services. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for information about configuring these services.

For a complete description of the DNS, refer to RFC 1032, RFC 1033, RFC 1034, RFC 1035, RFC 1183, and RFC 1348, which define the Internet standard.

Domain Names

The DNS uses a hierarchical naming convention for naming hosts. Each host name is composed of domain labels separated by periods. Local network administrators have the authority to name local domains within an intranet. Each label represents an increasingly higher domain level within an intranet. The fully qualified domain name of a host connected to one of the larger intranets generally has one or more subdomains:

- *host.subdomain.subdomain.rootdomain*
- *host.subdomain.rootdomain*

Domain names often reflect the hierarchy level used by network administrators to assign domain names. For example, the domain name `eng.mit.edu.` is the fully qualified domain name, where `eng` is the host, `mit` is the subdomain, and `edu` is the highest level domain (root domain).

Figure 16 on page 434 is an example of the DNS used in the hierarchy naming structure across an intranet.

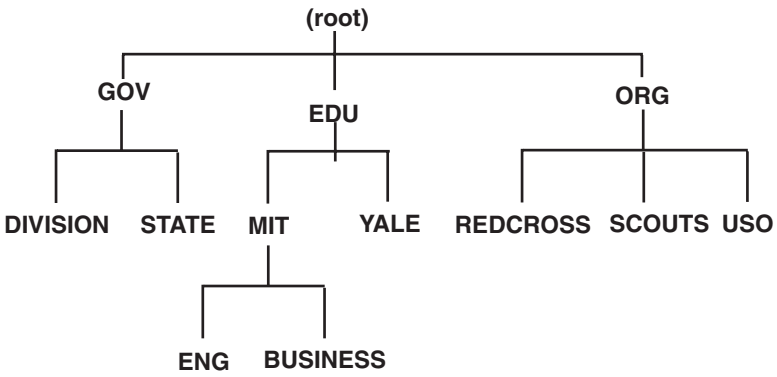


Figure 16. Hierarchical Naming Tree

You can refer to hosts in your domain by host name only; however, a name server requires a fully qualified domain name. The local resolver appends the domain name before sending the query to the Domain Name Server for address resolution.

Domain Name Servers

Domain Name Servers are designated network nodes that maintain a database of information about all nodes in a zone. A name server has a zone of authority that is a subnetwork, or a group of subnetworks, for which the name server maintains a database. The complete database is not kept by any one name server on a network. A name server is authoritative only within its zone of authority.

To minimize dependency on a particular node, the name server's database for a zone is replicated at several nodes. At least one of the nodes is designated as the primary name server. The others are secondary name servers. The zone data updates and maintenance are reflected in the primary name server. The secondary name servers update their database by contacting the primary name server at regular intervals. Both primary and secondary name servers are authoritative for a zone.

The zones of authority are arranged in a hierarchy based on the domain origin components. A special zone known as the *root* exists at the top of the domain name hierarchy in a network. The root zone contains a list of all the root servers. For example, in the Internet, the root name servers store information about nodes in the root domain, and information about the delegated domains, such as `com` (commercial), `edu` (education), and `mil` (military). The root name servers store the names of name servers for each of these domains, which in turn store the names of name servers for their delegated subdomains.

TCP/IP applications contact a name server whenever it is necessary to translate a domain name into an IP address, or when information is required about a domain. The name server performs the translation if it has the necessary information. If it does not have the necessary information, the name server can contact other name servers, which in turn can contact other name servers. This process is called a *recursive query*. Alternatively, a name server can simply return the address of another name server that might hold the requested information. This is called a *referral response* to a query. Name server implementations must support referrals, but are not required to perform recursive queries. See "Resolvers" on page 435 for more information about query responses.

Some name server implementations maintain a cache of query responses sent out to other name servers on behalf of clients. This improves the processing speed for queries about domain names outside the server's zone of authority. However, responses derived from cached information are not considered to be authoritative and are flagged in the response.

The Network Information Center (NIC) is responsible for network and user registration, including network number, top-level domain name assignment, and `in-addr.arpa` zone assignment. For more information contact:

```
Government Systems, Inc.
Attention:  Network Information Center
14200 Park Meadow Drive,  Suite 200
Chantilly, VA  22021
(internet address:  nic@nic.ddn.mil)
```

Resolvers

Programs that query a name server are called *resolvers*. Because many TCP/IP applications need to query the name server, a set of routines is usually provided for application programmers to perform queries. Under MVS, these routines are available in the TCP/IP application programming interface (API) for each supported language.

CS for OS/390 provides five programs for interactively querying a name server:

- NSLOOKUP
- onslookup
- NSUPDATE
- DIG
- host

When using NSLOOKUP and DIG queries, you should not use the NAMESRV TSO user ID. Because of a naming conflict of having NAMESRV as a procedure name as well as a TSO user ID, an IUCV return code 3 error can result.

See "Using the TSO NSLOOKUP Command" on page 439, and "Using the TSO DIG Command" on page 475 for more information about these programs.

Resolvers operate by sending query packets to a name server, either over the network or to the local name server.

A query packet contains the following fields:

- Domain name
- Query type
- A query class

"Resource Records" on page 436 lists valid query class (network class) and query type (data type) values. The name server attempts to match the three fields of the query packet to its database. For flexibility, the following wildcard query types are defined:

Type	Description
ANY	Indicates any record type for the domain name
AXFR	Indicates the query type used by secondary name servers to transfer all records in the zone (the query class is set to IN when using the AXFR query type)

Type	Description
MAILB	Indicates any mailbox records for the domain name

The name server can return the following query responses:

Response	Description
Authoritative	Is returned from a primary or secondary name server. The name server contains all the domain data used to define the zone for the specified query.
Non-authoritative	Is returned from a cache kept by a name server. The cache does not contain the domain data used to define the zone for the specified query.
Format Error	The name server found an error in the query packet sent by the resolver.
Name Error	No resource records of any type (including wild cards) exist for the domain name specified.
Negative	No records of the requested type were found for the domain name specified.
Not-implemented	The name server does not support the type of query requested.
Referral	Contains the addresses of other name servers that might be able to answer the query. A referral response is returned when a recursive query is not supported, not requested, or cannot be answered because of network connectivity.
Refused	The name server refuses to perform the specified operation. For example, some root name servers limit zone transfers to a set number of IP addresses.

Data from a name server is stored and distributed in a format known as a resource record. Resource record fields are described in detail in “Resource Records” on page 436. Each response from a name server can contain several resource records, which can contain a variety of information. The format of a response is defined in RFC 1035. It includes the following sections:

- A question section, echoing the query for which the response is returned.
- An answer section, containing resource records matching the query.
- An additional section, containing resource records that do not match the query, but might provide useful information for the client. For example, the response to a query for the host name of a name server for a specific zone includes the IP address of that name server in the additional section.
- An authority section, containing information specific to the type of response made to the query. If a referral is returned, this section contains the domain names of name servers that could provide an authoritative answer. If a negative response is returned indicating the name does not exist, this section contains a Start Of Authority (SOA) record defining the zone of authority of the responding name server.

Resource Records

Resource records are name server database records. These records contain the following fields:

Note: Experimental means there are no RFCs that define specific rules for these data types.

Field	Description

Domain name Identifies a network object. A network object can be a network, a specific node, a mailbox (for a network user's mail), or other objects addressable by the DNS.

Data type Indicates the type of data record. The following is a list of valid record types:

A Address record

The address record contains the dotted decimal notation IP address for the domain name identifying the record.

AFSDB
Andrew File System database location (experimental)

The AFS database location record enables mapping from a domain name to the name of an AFS cell database server. This record contains the domain name of a host that has a server for the cell named.

CNAME
Canonical name record

The canonical name record is used to provide alias or alternative name information for a domain name. The domain name specified in the first field of the record is an alternative to the canonical or real domain name specified in the data field.

HINFO
Host Information Record

This record type contains a text string specifying the processing unit type and operating system of a node.

ISDN ISDN address (experimental)

The ISDN address record contains the ISDN number of the owning domain, as well as any direct-dial-in number.

MB The mailbox record (experimental)

The mailbox record contains the domain name of a host machine to receive mail for the user-specified in the domain name field.

MG Mail group member record (experimental)

The mail group member record specifies the mail address of a person belonging to the mail group specified in the domain name field.

MINFO
Mailbox information record (experimental)

The mailbox information record specifies the mail addresses of the persons responsible for the mail group specified in the domain name field.

MR Mail rename name record (experimental)

The mail rename name record specifies a mailbox that is a rename of the mailbox specified in the domain name field.

MX Mail exchanger record

The mail exchanger record identifies a host that can act as a mail exchange for the domain specified in the domain name field. A mail exchange runs a mail agent that delivers or forwards mail for the domain name specified in the first field.

NS Name server record

The name server record contains the domain name of a name server for the current zone.

NULL Null resource record (experimental)

The null resource record contains any information less than 65 535 octets in length.

PTR Domain name pointer record

The domain name pointer record is mainly used to store data for the `in-addr.arpa` domain, and contains the domain name referenced by an IP address.

RP Responsible Person (experimental)

The Responsible Person record provides the ability to specify different responsible persons for different names in a zone. It contains the domain name that specifies the mail box for the responsible person.

RT Route Through (experimental)

The Route Through record provides a route-through binding for hosts that do not have their own direct wide-area-network addresses. It contains the route preference and the domain name of the host that serves as an intermediate in reaching the desired host.

SOA Start of authority record

The SOA record is unique to a zone and contains the administrative details of the zone, including:

- The domain name of the name server responsible for the zone
- The mail address of the user responsible for the zone
- The serial number of the zone database, which identifies the current revision of the data
- The refresh interval, which indicates the length of time, in seconds, you must allow between the refreshing of a database from a remote name server
- The retry interval, which indicates the length of time, in seconds, you must allow before retrying a failed refresh
- The expiration time to live (TTL), which indicates the maximum time, in seconds, for records to be valid in the zone database
- The minimum TTL, which indicates the minimum time, in seconds, for records to be valid in the zone database

TXT Text string record

The text string record contains descriptive text.

WKS Well-known services record

The well-known services record stores the protocol numbers of multiple services in a single record. Each of the defined TCP/IP services has a unique protocol number. Refer to RFC 1060 for more detailed information.

X25 X.25 address (experimental)

The X.25 address record contains the public-switched data network address of the named domain.

Network class

Valid network class values are:

IN The Internet (most Domain Name Systems support only the Internet [IN] class)

CS CSNET (obsolete)

CH CHAOS system (obsolete)

HS Hesiod class (Dyer 87)

Note: You can define the wildcard value ANY to match any of these classes.

TTL The number of seconds that a record is valid in a cache

Data Information appropriate for the data type indicated in the data type field, in the format defined for that specific data type

Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for examples of resource records.

Using the TSO NSLOOKUP Command

The NSLOOKUP command enables you to query name servers in order to accomplish the following tasks:

- Locate information about network nodes
- Examine the contents of a name-server database
- Establish the accessibility of name servers

NSLOOKUP has two modes of operation: interactive mode and command mode. Interactive mode enables you to repeatedly query one or more name servers for information about various hosts and domains and display that information on your terminal. Command mode displays the output from the query supplied as part of the command and then exits.

NSLOOKUP Configuration

The configuration options of NSLOOKUP determine the operation and results of your name server queries. You can configure NSLOOKUP operation using the following methods:

- TCP/IP client program configuration data set, TCPIP.DATA
- NSLOOKUP options data set, *user_id*.NSLOOKUP.ENV
- NSLOOKUP command options

For information about the TCPIP.DATA data set, refer to the *OS/390 SecureWay Communications Server: IP Configuration*. For information about the

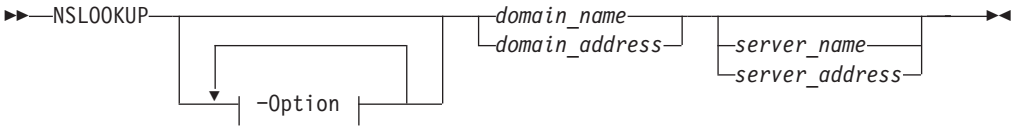
NSLOOKUP.ENV options data set and the NSLOOKUP command options, see “NSLOOKUP Options” on page 446.

NSLOOKUP—Query a Name Server in Command Mode

Purpose

Use the NSLOOKUP command to specify an individual query in command mode.

Format



Parameters

-Option
For a description of the NSLOOKUP options, see “NSLOOKUP Options” on page 446.

domain_name
Queries the name server for information about the current query type of *domain_name*. The default query type is A (address query).

domain_address
Reverses the components of the address and generates a pointer type (PTR) query to the name server for the `in-addr.arpa` domain mapping of the address to a domain name.

server_name
Directs the default name server to map *server_name* to an IP address and then use the name server at that IP address.

server_address
Specifies the IP address of the name server to be queried other than the default name server. A query for the address in the `in-addr.arpa` domain is initially made to the default name server to map the IP address to a domain name for the server.

Usage

The parameters and subcommands of NSLOOKUP are case sensitive and must be entered in lowercase. Parameter values and domain names are not case sensitive.

Context

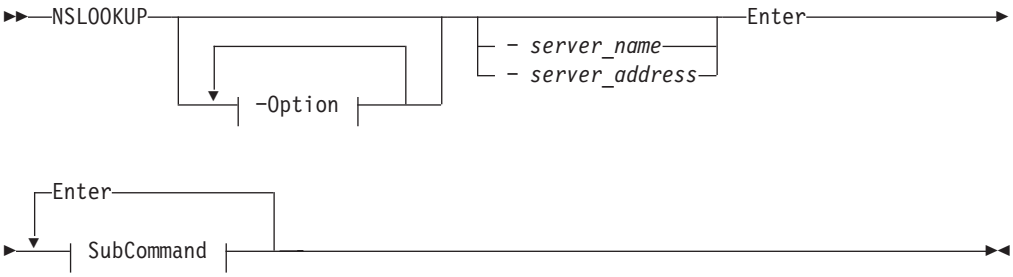
- See “NSLOOKUP Options” on page 446 for the complete list and description of NSLOOKUP options.
- See “NSLOOKUP—Issue Queries to Name Servers in Interactive Mode” on page 442 for the complete list and description of subcommand and query formats.

NSLOOKUP—Issue Queries to Name Servers in Interactive Mode

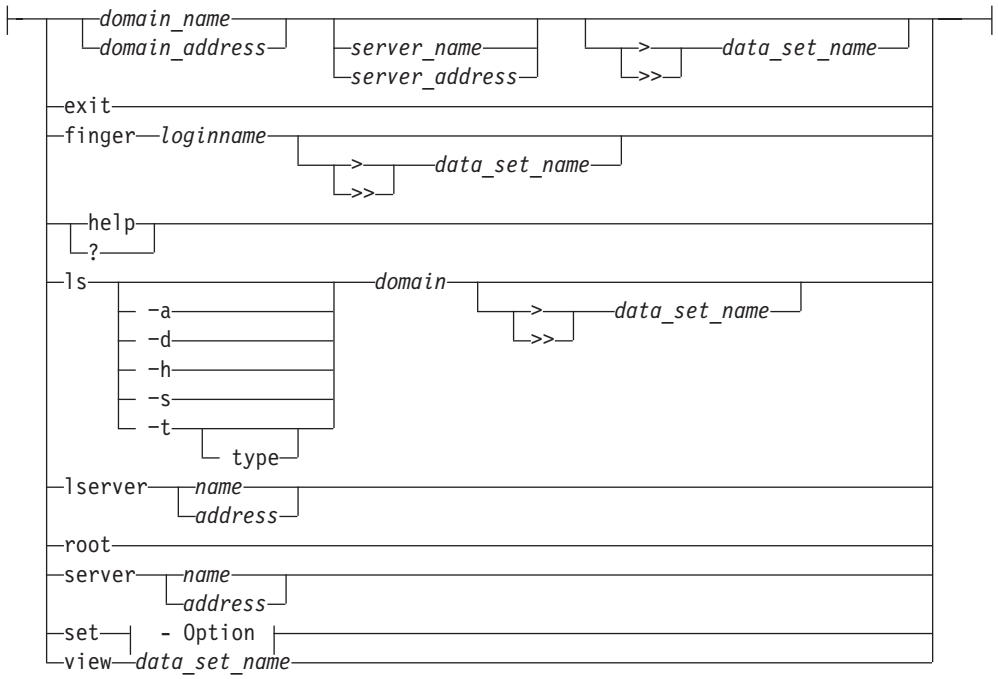
Purpose

Use the NSLOOKUP command to issue multiple queries in interactive mode. In interactive mode, an initial query is made to the selected name server to verify that the server is accessible. All subsequent interactive queries are sent to that server unless you specify another server using the *server* or *lserver* options.

Format



SubCommand:



Parameters

-Option
For a description of the NSLOOKUP options, see “NSLOOKUP Options” on page 446.

query
Directs NSLOOKUP to perform a query for the domain specified. The query requests all information about the domain name using the current class and

query (resource record) type. You can specify a server other than the current server to perform the domain name resolution.

Output can be placed in a data set for later viewing by specifying *data_set_name*. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least one space before and after the > or >> symbol.

Queries processed by NSLOOKUP that specify an address can give unexpected results. If the current query type is address (A) or domain-name pointer (PTR), NSLOOKUP generates a PTR type query for the specified address in the `in-addr.arpa` domain. This returns PTR records which define the host name for the specified address. If the current query type is neither of these two types, a query is performed using the current query type, with the domain name specified as the address given.

Text that does not conform to the defined options and follows the preceding syntax is treated as a domain query. NSLOOKUP does not issue a query for a domain name if the name is unqualified and is the same as one of the defined options.

exit
Exits from NSLOOKUP interactive mode.

finger *parms*
Extracts information from the finger server of the node found in the last address query. By default, this command returns a list of logged-in users for the node last found. You can find information about a particular user by specifying the login name of the user as a parameter. The *loginname* variable is case-sensitive and must be specified in the same case (upper or lower) as that used by the host.

Output can be placed in a data set for later viewing by specifying *data_set_name*. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least one space before and after the > or >> symbol.

An error occurs if the preceding subcommand was not a successful address query or finger operation. If the current host is not defined, querying the name server defines that name server to be the current host for a subsequent finger operation.

The finger option expects that the finger server is operating on the node found. An error occurs if the server is not operating or the node cannot be reached.

help or ?
Displays a brief summary of commands.

ls *parms*
Lists various information available for the domain. By default, the IP address of each node in the domain is listed.

To select resource records other than the default, specify one of the following options:

-a CNAME

-d ALL

-h HINFO

-s WKS

-t [*type***]**

Retrieves the resource record type specified in *type*. If no record type is specified with the -t option, the current default type is used.

See “Resource Records” on page 436 for information about valid query types.

The ls command expects the domain name specified in *domain* to be a zone. If the domain name specified refers to a host, an error message is printed and no information is given. This command should create a virtual circuit (TCP connection) with the current name server to service the request. An error message is printed if the virtual circuit cannot be established.

Output can be placed in a data set for later viewing by specifying *data_set_name*. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least 1 space before and after the > or >> symbol.

A number sign (#) is displayed at the terminal as every 50 lines are written to the data set to indicate the command is still executing.

lserver *parms*

Changes the current server. If *server_name* is specified, the IP address of *server_name* is determined using the initial server defined at command invocation.

An error occurs if the domain name cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the node specified; it simply changes a local variable storing the address of the default name server.

root

Changes the current server address to the address of the root server. The root server is `ns.nic.ddn.mil` by default, but can be changed using the root=*name* SET subcommand. This command is equivalent to lserver *name*.

An error occurs if the name of the root server cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the node specified; it simply changes a local variable storing the address of the default name server.

server *parms*

Changes the current server. If *name* is specified, the IP address of *name* is determined using the current server.

An error occurs if the domain name cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the address; it simply changes a local variable storing the address of the default name server.

set *option*
Changes internal state information values. See “NSLOOKUP Options” on page 446 for a description of the options.

view *data_set_name*
Sorts and lists the contents of *data_set_name* one screen at a time. An error occurs if the data set does not exist.

Usage

- You can query by entering the domain name of the node or subnetwork for which information is required. Define the data type of information to be retrieved using the SET *querytype=* option. You can define only one type of resource record for a domain name in a single query, unless the wildcard query type of ANY has been set. If an IP address is given instead of a domain name, a query for the address in the `in-addr.arpa` domain is made to map the IP address to a domain name.

  The domain name or address for the query can be followed by the domain name or IP address of a name server to contact for the query. If this is not specified, the current name server is used. For example, entering:

  ```
  toolah wurrup.fourex.oz
  ```

  queries the name server on `wurrup.fourex.oz` for information about the node `toolah`. When specifying domain names that include periods, the trailing period (indicating a fully qualified domain name) is optional. NSLOOKUP deletes the trailing period if it is present. If you are specifying a root domain, the domain name must have two trailing periods. For example, specify `mynode..` when the node `mynode` is in the root domain.
- The name server often requires a fully qualified domain name for queries. However, NSLOOKUP enables the specification of a default subnetwork domain using the SET *domain=* option, with the initial default obtained from the TCPIP.DATA data set. When the `defname` flag is enabled using the SET *defname* option, the default domain name specified by SET *domain=* is appended to all unqualified domain names. For example, if the default domain name is `fourex.oz` and the *defname* flag is enabled, a query for the name `toolah` automatically generates a query packet containing the domain name `toolah.fourex.oz`.
- A time-out error occurs if the name server is not running or is unreachable. A `Non-existent Domain` error occurs if any resource record type for the specified domain name is not available at the name server. A `Server Failed` error occurs when the local name server cannot communicate with the remote name server.
- NSLOOKUP may interpret typing or syntax errors in subcommands as queries. This results in a query being sent and the name server response printed. The response is usually `Non-existent Domain`, which indicates that the server could not find a match for the query.

NSLOOKUP Options

The configuration options of NSLOOKUP determine the operation and results of your name server queries. These options can be specified in command-mode queries, interactive-mode queries, or in the *user_id*.NSLOOKUP.ENV data set. When you include NSLOOKUP options with the initial NSLOOKUP command the - operand must immediately precede the option. If you specify NSLOOKUP options while in interactive mode the SET subcommand must precede the option. Specifying NSLOOKUP options in the *user_id*.NSLOOKUP.ENV data set is optional. Use the SET subcommand before the option if you want to reset the option value. The - operand is not valid preceding *options* in the *user_id*.NSLOOKUP.ENV data set.

For example, to specify a name server (NS) type record lookup for the domain name `fourex.oz` in command mode you enter:

```
nslookup -querytype=ns fourex.oz
```

To submit the same request using interactive mode enter the following sequence:

```
nslookup
enter
set querytype=ns
enter
fourex.oz
```

To make `querytype` of NS a default option for your NSLOOKUP commands place one of the following statement in the *user_id*.NSLOOKUP.ENV data set.

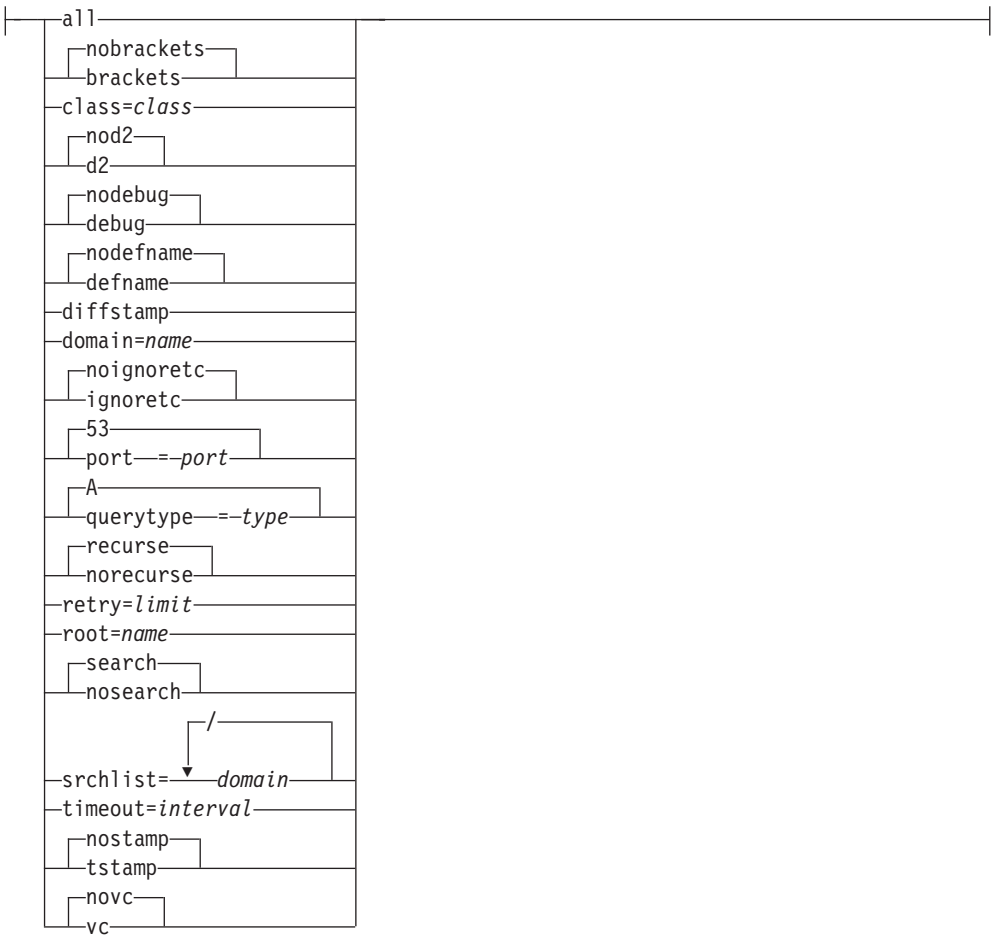
- *set querytype=ns*
- *querytype=ns*

The optional data set *user_id*.NSLOOKUP.ENV. contains only NSLOOKUP options and defines the NSLOOKUP defaults. If the *user_id*.NSLOOKUP.ENV data set exists, the NSLOOKUP options are read from the data set and executed before any queries are made. You must enter each option on a separate line. Blank lines are ignored.

The following is an example of the contents of the *user_id*.NSLOOKUP.ENV data set:

```
set domain=powers.oz
querytype=HINFO
set norecurse
vc
```

Option:



all
enables you to print the current values of the internal state variables. This option does not alter the internal state of NSLOOKUP.

brackets
Causes output to display with brackets.

nobrackets
Causes output to display using < and > instead of brackets. This option is for terminals that do not support brackets. This is the default.

class=*class*
Sets the class of information returned by queries. The class must be identified by its mnemonic. The minimum abbreviation for this option is *cl*.

d2
Directs NSLOOKUP to enable extra debugging mode. Using *d2* also enables debug mode.

Note: To obtain all alias names for a host when using reverse query, you must set the *d2* option.

nod2
Directs NSLOOKUP to disable extra debugging mode. The default is *nod2*.

debug
Directs NSLOOKUP to print debugging information for each query and its corresponding response. The minimum abbreviation is *deb* and *nodeb*.

nodebug
Directs NSLOOKUP to not print debugging information for each query and its corresponding response. This option also disables *d2*. The minimum abbreviation is *nodeb*. This is the default.

defname
Directs NSLOOKUP to append the default domain name to an unqualified domain name in a query.

The default domain name is initially obtained from the TCPIP.DATA data set, but can be changed using the domain=*name* option. The minimum abbreviation for this option is *def*.

nodefname
Directs NSLOOKUP to not append the default domain name to an unqualified domain name in a query.

If you specify this option, the domain name specified in the query is passed to the server without modification. This is the default. The minimum abbreviation for this option is *nodef*.

diffstamp
Displays the time, when it changes, before each output line.

domain=*name*
Sets the default domain name to *name*. Initially, the default domain name is obtained from the TCPIP.DATA data set. The validity of *name* is not verified. This option also updates the search list. The search list contains the domain specified and the parents of the default domain if it has at least two components in its name. For example, if the default domain is `wurrup.forex.oz` the search list contains `wurrup.forex.oz` and `forex.oz`. Use the SET *srchlist* command to specify a different search list. The minimum abbreviation for this option is *do*.

ignoretc
Directs NSLOOKUP on the handling of truncated responses. The name server indicates, in the response header, that the complete query response did not fit into a single UDP packet and has been truncated.

Specifying *ignoretc* directs NSLOOKUP to ignore the truncation condition when it is set in the response by the name server.

NSLOOKUP does not handle responses greater than 512 characters in length. Responses greater than 512 characters are truncated and the internal truncation flag is set. This condition is revealed only when the *debug* option is enabled. The minimum abbreviation for this option is *ig*.

noignorectc
Directs NSLOOKUP to automatically retry the query using a TCP connection when a response is sent with the truncation indicator set. This is the default. The minimum abbreviation for this option is *noig*.

port=*port*
Specifies the port number to use when contacting the name server. The Domain Name System is a well-known service and has been allocated port 53.

NSLOOKUP uses port 53 by default, but the port option enables you to specify another port to access. The minimum abbreviation for this option is *po*.

querytype=*type*
Specifies the type of information returned by queries. The initial query type is A (address information). See “Resource Records” on page 436 for the available query types.

NSLOOKUP cannot generate queries about type NULL. However, it can accept responses containing resource records of type NULL. In this case, NSLOOKUP displays the number of bytes returned in the NULL record. Global queries that return all resource records for a specific domain name are specified by the wildcard value ANY. The minimum abbreviation for this option is *q*.

The type=*type* option is accepted by NSLOOKUP as a synonym for the querytype=*type* option.

recurse
Directs NSLOOKUP to request a recursive query when querying a name server. The minimum abbreviation for this option is *rec*. This is the default.

norecurse
Specifies that a recursive query is not returned. The minimum abbreviation for this option is *norec*.

retry=*limit*
Specifies the number of times a request is resent. When a request is sent and the time-out period expires for a response, the request is resent until the value specified in *limit* has been exceeded. The value specified in *limit* determines the number of attempts made to contact the name server. The default value for *limit* is retrieved from the *hlq*.TCPIP.DATA data set.

Setting *limit* to 0 disables NSLOOKUP from contacting the name server. The result is an error message `no response from server`.

The retry algorithm for NSLOOKUP uses both the *limit* value and the time-out period. Each time a request is resent, the timeout period for the request is twice the time-out period used for the last attempt. The minimum abbreviation for this option is *ret*.

root=*name*
Specifies the name of a root server. The root server is `ns.nic.ddn.mil` by default.

search
Directs NSLOOKUP to enable the use of a search list. The minimum abbreviation for this option is *sea*.

nosearch
Directs NSLOOKUP to disable the use of a search list. The minimum abbreviation for this option is *nosea*.

srchlist=[*domain***/***domain***/...]**
Specifies one or up to three domain names to be appended to unqualified host names when attempting to resolve the host name. Each domain name specified is tried in turn until a match is found.

This option also directs the default domain to be set to the first domain name specified in the search list. The minimum abbreviation for this option is *srchl*.

timeout=*interval*
Specifies the number of seconds to wait before timing out of a request. The default for *interval* is retrieved from the *hlq*.TCPIP.DATA data set. The minimum abbreviation for this option is *t*.

tstamp
Prefixes the current time to each output line.

nostamp
Turns off the time display. This is the default.

vc Specifies to use a virtual circuit (TCP connection) to transport queries to the name server or datagrams (UDP). The default is retrieved from the *hlq*.TCPIP.DATA data set.

novc
Specifies to not use a virtual circuit to transport queries to the name server or datagrams. This option is the default.

NSLOOKUP Examples

This section contains examples of NSLOOKUP command-mode queries, and interactive-mode queries using the various options available for NSLOOKUP commands.

In Figure 17, the router wurrup has two IP addresses and there are two name servers, wurrup being the primary name server. This network is described by a single zone in the domain naming hierarchy stored in the name servers.

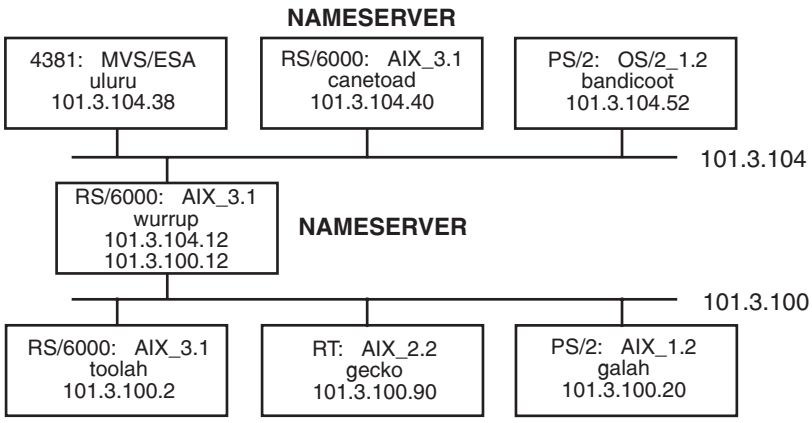


Figure 17. Hierarchical Naming Tree—A TCP/IP Network

The following are examples of how to use NSLOOKUP to extract information from a name server. The queries are executed from the MVS host uluru on the network described in Figure 17.

The following examples are command-mode queries.

- To make a simple address query:

```
  User:    nslookup toolah.fourex.oz wurrup.fourex.oz
System:  Server:  wurrup
         Address:   101.3.104.12

         Name:   toolah.fourex.oz
         Address:   101.3.100.2
```

- To specify a name server (NS) type record lookup:

```
  User:    nslookup -querytype=ns fourex.oz
System:  Server:  canetoad
         Address:   101.3.104.40

         fourex.oz  nameserver = wurrup.fourex.oz
         fourex.oz  nameserver = canetoad.fourex.oz
         wurrup.fourex.oz   internet address = 101.3.100.12
         wurrup.fourex.oz   internet address = 101.3.104.12
         canetoad.fourex.oz internet address = 101.3.104.40
```

- To specify a different default domain name to be appended to an unqualified domain name given as input:

```
  User:    nslookup -do=fourex.oz uluru
System:  Server:  canetoad.fourex.oz
         Address:  101.3.104.40

         Name:    uluru.fourex.oz
         Address:  101.3.104.38
```

- To specify a list of domain names to be appended in turn to the unqualified host name when attempting to resolve it:

```
  User:    nslookup -srchlist=nowhere.oz/fourex.oz uluru
System:  Server:  canetoad.fourex.oz
         Address:  101.3.104.40

         Name:    uluru.fourex.oz
         Address:  101.3.104.38
```

The following command places NSLOOKUP in interactive mode with wurrup as the default server.

```
User:
nslookup - wurrup
System:   Default Server:  wurrup
Address:   101.3.104.12
```

All following examples are in the interactive mode initiated in the preceding example.

- Show the default flag settings:

```
User:
set all
Default Server:  wurrup.fourex.oz
Address:  101.3.104.12

 Set options:
   nodebug             defname         nosearch        recurse
   nod2                novc            noignoretc      port=53
   querytype=A         class=IN        timeout=60      retry=1
   root=ns.nic.ddn.mil.
   domain=FOUREX.OZ
   brackets
   nostamp
   srchlist=FOUREX.OZ
```

- Perform a simple address query:

```
User:
toolah
System:    Server:  wurrup
Address:   101.3.104.12

Name:    toolah.FOUREX.OZ
Address:  101.3.100.2
```

- Set the query record type to HINFO, and perform another query:

```
User:
set q=HINFO
toolah
System:     Server:  wurrup
Address:   101.3.104.12

toolah.FOUREX.OZ   CPU = RS6000     OS = AIX3.1
```

- Find out the name servers available for a domain:

```
User:
set q=NS
fourex.oz
System:   Server:  wurrup
Address:   101.3.104.12

fourex.oz  nameserver = wurrup.fourex.oz
fourex.oz  nameserver = canetoad.fourex.oz
wurrup.fourex.oz   internet address = 101.3.100.12
wurrup.fourex.oz   internet address = 101.3.104.12
canetoad.fourex.oz internet address = 101.3.104.40
```

- Change the current server from wurrup to canetoad and make more queries:

```
User:   server canetoad
System:   Default Server:  canetoad.FOUREX.OZ
Address:   101.3.104.40

User:
set q=A
gecko
System:   Server:  canetoad.FOUREX.OZ
Address:   101.3.104.40

Name:    gecko.FOUREX.OZ
Address:  101.3.100.90
```

- Enable debugging and execute a simple query to see the result, and then disable debugging:

```
User:
set deb
wurrup
System:   Server:  canetoad.FOUREX.OZ
Address:   101.3.104.40

          res_mkquery(0, wurrup.FOUREX.OZ, 1, 1)
          ------------
          Got answer:
             HEADER:
                opcode = QUERY, id = 7, rcode = NOERROR
                header flags:  response, auth. answer, want recursion,
                recursion avail
                questions = 1, answers = 2, authority records = 0,
                additional = 0

             QUESTIONS:
                wurrup.FOUREX.OZ, type = A, class = IN
             ANSWERS:
             -> wurrup.FOUREX.OZ
                internet address = 101.3.104.12
                ttl = 9999999 (115 days 17 hours 46 mins 39 secs)
             -> wurrup.FOUREX.OZ
                internet address = 101.3.100.12
                ttl = 9999999 (115 days 17 hours 46 mins 39 secs)

          ------------
Name:    wurrup.FOUREX.OZ
         Addresses:  101.3.104.12, 101.3.100.12
User:
set nodeb
```

- Find all addresses in the `fourex.oz` domain using the *ls* option:

```
User:
ls fourex.oz
System:   [canetoad.FOUREX.OZ]
          fourex.oz                     server = wurrup.fourex.oz
          wurrup                         101.3.100.12
          wurrup                         101.3.104.12
          fourex.oz                     server = canetoad.fourex.oz
          canetoad                       101.3.104.40
          gecko                          101.3.100.90
          wurrup                         101.3.100.12
          wurrup                         101.3.104.12
          galah                          101.3.100.20
          bandicoot                      101.3.104.52
          toolah                         101.3.100.2
          canetoad                       101.3.104.40
          loopback                       127.0.0.1
          uluru                          101.3.104.38
```

- Find all aliases in the `fourex.oz` domain, then exit from NSLOOKUP interactive mode:

```
User:
ls -a fourex.oz
System:  [canetoad.FOUREX.OZ]
          localhost                      loopback.fourex.oz
          infoserver                     wurrup.fourex.oz
          pabxserver                     wurrup.fourex.oz
User:
exit
```

- To display a summary of available commands:

```
User:
help
System:
Commands:       (identifiers are shown in uppercase, <> means optional)
NAME            - print info about the host/domain NAME using default server
NAME1 NAME2     - as above, but use NAME2 as server
help or ?     - print info on common commands; see nslookup man for details
set OPTION      - set an option
    all         - print options, current server and host
    <no>debug   - print debugging information
    <no>d2      - print exhaustive debugging information
    <no>defname - append domain name to each query
    <no>recurse - ask for recursive answer to query
    <no>vc      - always use a virtual circuit
    domain=NAME - set default domain name to NAME
    srchlist=N1</N2/.../N6> - set domain to N1 and search list to N1,N2, etc.
    root=NAME   - set root server to NAME
    retry=X     - set number of retries to X
    timeout=X   - set initial time-out interval to X seconds
    querytype=X - set query type, e.g., A,ANY,CNAME,HINFO,MX,NS,PTR,SOA,WKS
    type=X      - synonym for querytype
    class=X     - set query class to one of IN (Internet), CHAOS, HESIOD or ANY
server NAME     - set default server to NAME, using current default server
lserver NAME    - set default server to NAME, using initial server
finger <USER>   - finger the optional NAME at the current default host
root            - set current default server to the root
ls <opt> DOMAIN ^> DATASET¦ - list addresses in DOMAIN
                  (optional: output to DATASET)
    -a          -  list canonical names and aliases
    -h          -  list HINFO (CPU type and operating system)
    -s          -  list well-known services
    -d          -  list all records
    -t TYPE     -  list records of the given type (e.g., A,CNAME,MX, etc.)
view DATASET  - sort an 'ls' output file and view it with more
exit          - exit the program
```

- To find information for all the users currently logged in on the node specified in the last address query:

```
User:
finger
System:
[canetoad.FOUREX.OZ]
Further output to be generated ....
```

- To set the default domain name to `fourex.oz`, use the command

  ```
  set domain=fourex.oz
  ```

  This command overrides the DOMAINORIGIN statement in the *tcpip*.TCPIP.DATA data set.
- To specify that the default domain name is to be appended to an unqualified domain name given in a query, use the SET *defname* command.
- To request that the query be resent three times if the time-out period expires for a response, use the SET *retry=3* command. Three (3) is the maximum valid value.

Using the OS/390 UNIX onslookup/nslookup Command

The OS/390 UNIX onslookup command enables you to query any name server to perform the following tasks from the OS/390 UNIX environment:

- Identify the location of name servers.
- Examine the contents of a name server database
- Establish the accessibility of name servers

To display a list of options, enter the following from the command line:

```
onslookup -?
```

Note: nslookup is a synonym for the onslookup command in the OS/390 UNIX shell. nslookup command syntax is the same as that for the onslookup command.

Note: onslookup messages are not documented in the SecureWay Communications Server for OS/390 library. Therefore, onslookup command messages do not give a message ID for debugging.

The onslookup command has two modes of operation: interactive mode and command mode. In both modes, the address of the default name server comes from the resolver configuration file.

In the following example, the default domain is raleigh.ibm.com, and the default name server is at 9.37.34.149. If that name server fails to respond, the one at 9.37.34.7 will be used.

```
domain   raleigh.ibm.com
nameserver   9.37.34.149
nameserver   9.37.34.7
```

onslookup Configuration

The configuration options of onslookup determine the operation and results of your name server queries. The values for onslookup options can be specified in more than one location, as shown in Table 35 on page 456. The locations from which the values for onslookup options can be specified are:

1. onslookup command options
2. .onslookuprc file in the home directory
3. environment variable
4. `/etc/resolv.conf`
5. TCPIP.DATA configuration data set

Values specified as onslookup command options have priority over values specified in the `.onslookuprc` file, which have priority over the value specified by the environmental variable, and so on. For example, the value specified by the *all* option in the onslookup command has priority over the value specified by the *all* option in the `.onslookuprc` file. Similarly, the value specified by *ResolverTimeout* in the `/etc/resolv.conf` file has priority over the value specified by `ResolverTimeout` in the TCPIP.DATA configuration data set.

The letters beside some settings indicate that the *terms* are functionally equivalent. For example, the term *domain* (the letter ″A″) is functionally equivalent to the terms *DomainOrigin* and *LOCALDOMAIN*. If two functionally equivalent settings are listed in the same file, the one listed last has priority. For example, if *domain* and *DomainOrigin* are both listed in the `/etc/resolv.conf` file and *domain* is listed last, the value specified by *domain* has priority.

Note: The `/etc/resolv.conf` file may contain all the data normally found in the TCPIP.DATA data set. If information is read from `/etc/resolv.conf`, then TCPIP.DATA is not used. To configure the `/etc/resolv.conf` file, you use certain directives. See “Resolv.conf Directives” on page 456 for detailed descriptions.

Table 35. Settings that Affect onslookup Operation

Settings	onslookup command options	.onslookup file in the home directory	/etc/resolv.conf	TCPIP.DATA configuration data set
all	x	x		
class	x	x		
no[d2]	x	x		
[no]debug	x	x		
[no]defname	x	x		
domain (A)	x	x	x	
[no]ignoretc	x	x		
port (B)	x	x		
querytype	x	x		
[no]recurse	x	x		
retry (C)	x	x		
root	x	x		
[no]search	x	x		
srchlist (D)	x	x		
timeout (E)	x	x		
[no]vc (F)	x	x		
search (D)			x	
nameserver (G)			x	
sortlist			x	
options debug			x	
options ndots			x	
DOMAINORIGIN (A)			x	x
NSINTERADD (G)			x	x
NSPORTADDR (B)			x	x
RESOLVEVIA (F)			x	x
RESOLVERTIMEOUT (E)			x	x
RESOLVERUDPRETRIES (C)			x	x

Note: The environment variable LOCALDOMAIN (A) also affects onslookup operation.

Resolv.conf Directives

The onslookup command uses a private resolver that is different than the OS/390 UNIX resolver used by other OS/390 UNIX socket programs. If name services are not used, onslookup requires the HFS file, `/etc/hosts`, for host table lookup. In addition, onslookup uses only the built-in translation table. For a complete discussion of resolver configuration files, refer to the *OS/390 SecureWay Communications Server: IP Migration*.

Resolver configuration directives for `/etc/resolv.conf` are:

domain *domain name*
Specifies the resolver's default domain if the host name is not fully qualified.

search *domain domain...*
Specifies an ordered list of domains for the resolver to search. The domain listed first is searched first.

nameserver *IP_address(es)*
Specifies the IP address or addresses of a particular name server to query. The addresses are queried in the order listed.

sortlist *subnet {/subnet_mask}...*
Specifies an ordered list of subnets and networks if the resolver receives more than one address as a result of a query. You can include one or more subnet masks or you can omit the mask to specify the entire network.

options *debug*
Starts the debugging option.

options ndots: *minimum_number_of_dots*
Specifies the minimum number of dots an argument must have before the search list is applied. If the argument has less than the specified number of dots, the search list is appended to the name before any queries are sent. If the argument has the same number of dots or more, the query is sent first just as the user typed it. If a positive response is not received, subsequent queries are sent with the search list appended. The default is one.

Entering the Command Line Mode

The command line mode displays or stores the output from the query supplied as part of the invocation string and then exits.

To enter the command line mode, provide a complete query with the onslookup command invocation string.

Note: The **onslookup help** command works only in the interactive mode.

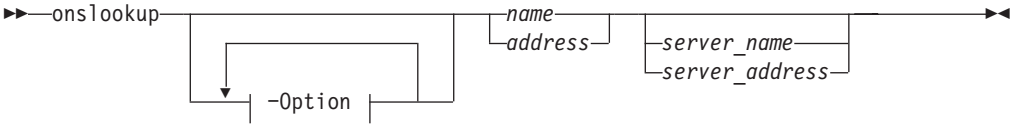
onslookup/nslookup (Command Mode)—Querying A Name Server in Command Mode

Purpose

Use the onslookup command to specify a single query.

Note: nslookup is a synonym for the onslookup command in the OS/390 UNIX shell. nslookup command syntax is the same as that for the onslookup command.

Format



Parameters

-Option
For a description of the onslookup options, see “onslookup Options” on page 465.

name
Queries the name server for the current query-type of name. The name typically represents a host name.

address
Reverses the components of the address and generates a pointer type (PTR) query to the name server for the in-addr.arpa domain mapping of the address to a domain name.

server_name
Directs the default name server to map *server_name* to an IP address and then use the name server at that address. This argument is optional. The default is the default name server found by the search order described in “onslookup Configuration” on page 455.

server_address
Specifies the IP address of the name server to be queried other than the default name server. A query for the address in the in-addr.arpa domain is initially made to the default name server to map the IP address to a domain name for the server. This argument is optional. The default is the default name server found by the search order described in “onslookup Configuration” on page 455.

Usage

Parameter values and domain names are not case sensitive.

Context

To display a list of options, enter the following from the command line:

```
onslookup -?
```

For a complete list and description of onslookup options, see “onslookup Options” on page 465.

Entering the Interactive Mode

Interactive mode enables you to query one or more name servers repeatedly for information about various hosts and domains, to display that information on your console, and, in some cases, to write response data to a file.

You can enter the interactive mode under the following conditions only:

- No arguments are supplied on command invocation. The default name server is used.
- The first argument is a hyphen (-), and the second argument is the host name or Internet address of a name server.

Note: The **onslookup help** command works only in the interactive mode.

onslookup/nslookup (Interactive Mode)—Issuing Multiple Queries to Name Servers in Interactive Mode

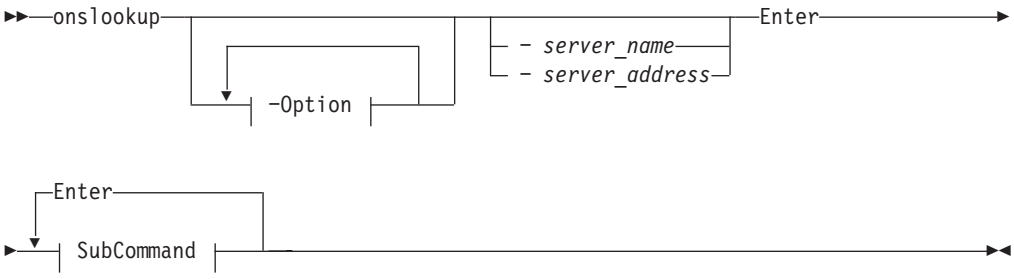
Purpose

Use the interactive mode to issue multiple queries. In interactive mode, an initial query is made to the selected name server to verify that the server is accessible. All subsequent interactive queries are sent to that server unless you specify another server using the *server* or *lserver* subcommands.

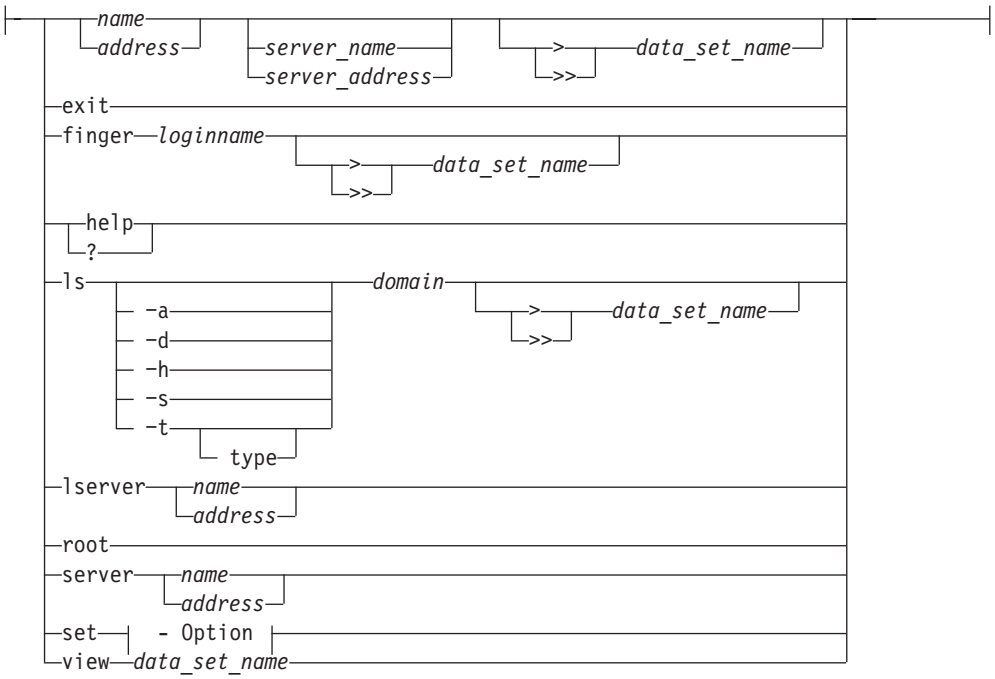
The command line length must be less than 256 characters. To treat a built-in command as a host name, precede it with an escape character (\). An unrecognized command is interpreted as a host name.

Note: nslookup is a synonym for the onslookup command in the OS/390 UNIX shell. nslookup command syntax is the same as that for the onslookup command.

Format



SubCommand:



Parameters

-Option
For a description of the onslookup options, see “onslookup Options” on page 465.

query
Perform a query for the *name* specified. (The *name* is typically a host name.) The query requests all information about the name using the current class and query (resource record) type. You can specify a server other than the current server to perform name resolution.

Output can be placed in a data set for later viewing by specifying *data_set_name*. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least one space before and after the > or >> symbol.

The use of *address* is only intended for A and PTR query types. If the current query type is address (A) or domain-name pointer (PTR), onslookup generates a PTR type query for the specified address in the `in-addr.arpa` domain. This returns PTR records, which define the host name for the specified address. If the current query type is neither of these two types, a query is performed using the current query type, with the domain name specified as the address given.

Text that does not conform to the defined options and follows the preceding syntax is treated as a domain query. onslookup does not issue a query for a domain name if the name is unqualified and is the same as one of the defined options unless the name is preceded by the escape character.

exit
Exits from onslookup interactive mode.

finger *parms*
Extracts information from the finger server of the node found in the last address query. By default, this command returns a list of logged-in users for the node last found. You can find information about a particular user by specifying the login name of the user as a parameter. The *loginname* variable is case-sensitive and must be specified in the same case (upper or lower) as that used by the host.

Output can be placed in a data set for later viewing by specifying *data_set_name*. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least one space before and after the > or >> symbol.

An error occurs if the preceding subcommand was not a successful address query or finger operation. If the current host is not defined, querying the name server defines that name server to be the current host for a subsequent finger operation.

The finger option expects that the finger server is operating on the node found. An error occurs if the server is not operating or the node cannot be reached.

help or -?
Displays a brief summary of commands.

Note: This command works only in interactive mode.

ls *parms*
Lists various information available for the domain. By default, the IP address of each node in the domain is listed.

To select resource records other than the default, specify one of the following options:

-a CNAME retrieves aliases and canonical names

-d ALL

-h HINFO

-s WKS

-t [*type***]**
Retrieves the resource record type specified in *type*. If no record type is specified with the -t option, the current default type is used.

See “Resource Records” on page 436 for information about valid query types.

The *ls* command expects the domain name specified in *domain* to be a zone. If the domain name specified refers to a host, an error message is printed and no information is given. This command should create a virtual circuit (TCP connection) with the current name server to service the request. An error message is printed if the virtual circuit cannot be established.

Output can be placed in a data set for later viewing by specifying *data_set_name* for a specified domain. The > *data_set_name* option places the output in *data_set_name* and overwrites the contents, if any, of the data set. The >> *data_set_name* option places the output in *data_set_name* and appends it to the contents, if any, of the data set. There must be at least 1 space before and after the > or >> symbol.

lserver *parms*
Changes the current server using the *original* name server to find the new name server. The parameter *name* or *address* is required.

An error occurs if the domain name cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the node specified; it simply changes a local variable storing the address of the default name server.

root
Changes the current server address to the address of the root server. The root server is `ns.nic.ddn.mil` by default, but can be changed using the *root=name* SET subcommand. This command is equivalent to *lserver name*.

An error occurs if the name of the root server cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the node specified; it simply changes a local variable storing the address of the default name server.

server *parms*
Changes the current server using the *current* name server to find the new name server. The parameter *name* or *address* is required.

An error occurs if the domain name cannot be mapped to an IP address. This option does not ensure that a name server can be reached at the address; it simply changes a local variable storing the address of the default name server.

set *option*
Changes internal state information values. See “NSLOOKUP Options” on page 446 for a description of the options.

view *data_set_name*
Sorts and lists the contents of *data_set_name* one screen at a time. An error occurs if the data set does not exist.

onslookup Options

The configuration options of onslookup determine the operation and results of name server queries. These options can be specified in command-mode queries, interactive-mode queries, or by the methods described in “onslookup Configuration” on page 455. In particular, see Table 35 on page 456.

When you include onslookup options with the initial onslookup command, the hyphen (-) operand must immediately precede the option. If you specify onslookup options while in interactive mode, the SET subcommand must precede the option.

For example, to specify a name server (NS) type record lookup for the domain name `fourex.oz` in command mode you enter:

```
onslookup -querytype=ns fourex.oz
```

To submit the same request using interactive mode, enter the following sequence:

```
onslookup
enter
set querytype=ns
enter
fourex.oz
```

The hyphen (-) operand is not valid preceding *options* in the `.onslookuprc` file. To make *querytype of NS* a default option for your onslookup commands place on of the following statements in the `.onslookuprc` file:

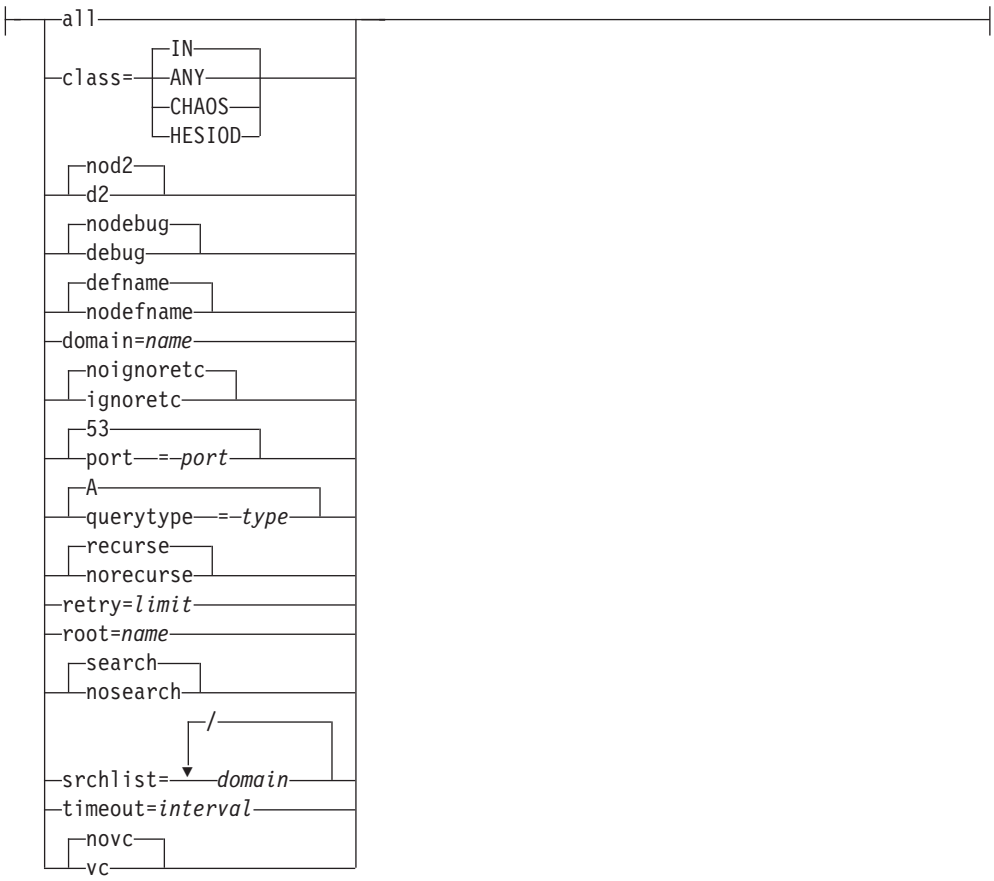
- `set querytype=ns`
- `querytype=ns`

The optional `.onslookuprc` file contains only onslookup options and defines the onslookup defaults. If the `.onslookuprc` file exists, the onslookup options are read from the file and executed before any queries are made. You must enter each option on a separate line. Blank lines are ignored.

The following is an example of the contents of the `.onslookuprc` file:

```
set domain=powers.oz
querytype=HINFO
set norecurse
vc
```

Option:



all enables you to print the current values of the internal state variables. This option does not alter the internal state of onslookup.

class=*class*
Sets the class of information returned by queries. The minimum abbreviation for this option is *cl*. The default is IN (Internet).

d2 Directs onslookup to enable extra debugging mode. Using *d2* also enables debug mode.

Note: Use this option to view all the information returned by the name server.

nod2
Directs onslookup to disable extra debugging mode. The default is *nod2*.

debug
Directs onslookup to print debugging information and the response to each query. The minimum abbreviation is *deb*.

Note: onslookup messages are not documented in the SecureWay Communications Server for OS/390 library. Therefore, onslookup command messages do not give a message ID for debugging.

nodebug
Directs onslookup not to print debugging information for each query and its corresponding response. This option also disables *d2*. The minimum abbreviation is *nodeb*. This is the default.

defname
Directs onslookup to append the default domain name to an unqualified domain name in a query. The minimum abbreviation for this option is *def*.

You can change the default domain name using the *domain=name* option.

nodefname
Directs onslookup not to append the default domain name to an unqualified domain name in a query.

If you specify this option, the domain name specified in the query is passed to the server without modification. The minimum abbreviation for this option is *nodef*.

domain=*name*
Sets the default domain name to *name*. The validity of *name* is not verified. This option also updates the search list to contain the specified domain and the two higher-level domains (parent and grandparent) of the default domain if it has at least two components. For example, if the default domain is

```
mugwump.wurrup.fourex.oz
```

the search list contains the following:

- `mugwump.wurrup.fourex.oz`
- `wurrup.fourex.oz`
- `fourex.oz`

Use the *set srchlist* subcommand and option to specify a different search list. The minimum abbreviation for the *domain* option is *do*.

ignoretc
Directs onslookup on the handling of truncated responses. When you specify this option, the name server indicates, in the response header, that the complete query response did not fit into a single UDP packet and has been truncated. The minimum abbreviation for this option is *ig*.

Specifying `ignoretc` directs onslookup to ignore the truncation when it is set in the response by the name server.

onslookup does not handle responses greater than 512 characters in length. Responses greater than 512 characters are truncated and the internal truncation flag is set. This condition is revealed only when the debug option is enabled.

noignorectc
Directs onslookup to automatically retry the query using a TCP connection when a response is sent with the truncation indicator set. This is the default. The minimum abbreviation for this option is *noig*.

port=*port*
Specifies the port number to use when contacting the name server. The Domain Name System is a well known service and has been allocated port 53. onslookup uses port 53 by default, but the port option enables you to specify another port to access. The minimum abbreviation for this option is *po*.

querytype=*type*
Specifies the type of information returned by queries. The default query type is A (address information). Using the wildcard value ANY specifies a global query that returns all resource records for a specific domain name. The minimum abbreviation for this option is *q*.

The *type=type* option is accepted by onslookup as a synonym for the *querytype=type* option.

recurse
Directs onslookup to request a recursive query when querying a name server. The minimum abbreviation for this option is *rec*. This is the default.

norecurse
Specifies that a recursive query is not returned. The minimum abbreviation for this option is *norec*.

retry=*limit*
Specifies the number of times a request is sent. When a request is sent and the timeout period expires for a response, the request is resent until the value specified in *limit* has been reached. The value specified in *limit* determines the total number of attempts made to contact the name server. The default value for *limit* is retrieved from the resolver configuration file. The minimum abbreviation for this option is *ret*.

Attempting to SET *retry* to zero results in a retry value of one. One is the minimum value allowed.

The retry procedure for onslookup uses both the *limit* value and the timeout period. Each time a request is resent, the timeout period for the request is twice the timeout period used for the last attempt.

root=*name*
Specifies the name of a root server. The root server is `ns.nic.ddn.mil` by default.

search
Directs onslookup to enable the use of a search list. The minimum abbreviation for this option is *sea*. This is the default.

nosearch
Directs onslookup not to use a search list. The minimum abbreviation for this option is *nosea*.

srchlist=[*domain***|***domain***|...]**
Specifies zero to six domain names to be appended to unqualified host names when attempting to resolve a host name. Each domain name is tried in turn until a match is found.

This option also sets the default domain to the first domain name specified in the search list. The minimum abbreviation for this option is *srchl*.

timeout=*interval*
Specifies the number of seconds to wait before a request times out. The minimum abbreviation for this option is *t*. The minimum value is 1.

vc Directs onslookup to use a virtual circuit (TCP connection) to transport queries to the name server or datagrams.

novc
Specifies to not use a virtual circuit to transport queries to the name server or datagrams. This option is the default.

Configuring Host Resolvers: onslookup Considerations

Programs that query a name server are called resolvers. Because many TCP/IP applications need to query the name server, a set of routines is usually provided for

application programmers to perform queries. Under MVS, these routines are available in the TCP/IP application programming interface (API) for each supported language or LE for OS/390 UNIX Sockets API.

The onslookup command uses a private resolver that is different from the OS/390 UNIX resolver used by other OS/390 UNIX socket programs. The onslookup command has the following functional differences:

- The HFS file, `/etc/hosts`, is required for host table lookup if name services do not exist. Following is a sample `/etc/hosts` file:

```
#
# OS/390 UNIX Resolver /etc/hosts file on mvss18oe.
#
# The format of this file is:
#
# Internet Address      Hostname   Aliases     # Comments
#
# Items are separated by any number of blanks and/or tabs.  A '#'
# indicates the beginning of a comment; characters up to the end of the
# line are not interpreted by routines which search this file.  Blank
# lines are allowed in this file.

9.24.104.126    mvs18oe mvsoe # OS/390 UNIX host
192.168.210.1   mvs18an       # AnyNet MVS host
192.168.210.8   mypcaa        # AnyNet gw host
9.24.104.79     mypc          # A workstation
```

- Only the built-in translation table is used.

For a complete discussion of resolver configuration files, refer to the *OS/390 SecureWay Communications Server: IP Migration*.

If the OS/390 UNIX name server will run on the host being configured, you need to configure the *first* name server (or NsInterAddr) directive in the resolver configuration file as the loopback address (127.0.0.1) or any address in your home list.

Diagnosing Problems

The onslookup program lets you query other name servers with the same query packet another name server would use. This is helpful in diagnosing lookup problems in TCP/IP UNIX System Services.

To turn debugging on at level 1, enter the following commands from the OS/390 shell:

```
onslookup
set debug
```

The onslookup program shows timeouts and displays response packets.

To turn the debug option off, enter the following command:

```
set nodebug
```

You can set the debugging option to level 2 by entering the following commands:

```
onslookup
set d2
```

The resolver shows the normal debugging information plus the query packets that were sent out. Turning on *d2* also turns on *debug*. Turning off *d2*, however, only turns off *d2* and *debug* remains on. To turn off both *d2* and *debug*, enter the subcommand SET *nodebug*.

Note: onslookup messages are not documented in the SecureWay Communications Server for OS/390 library. Therefore, onslookup command messages do not give a message ID for debugging.

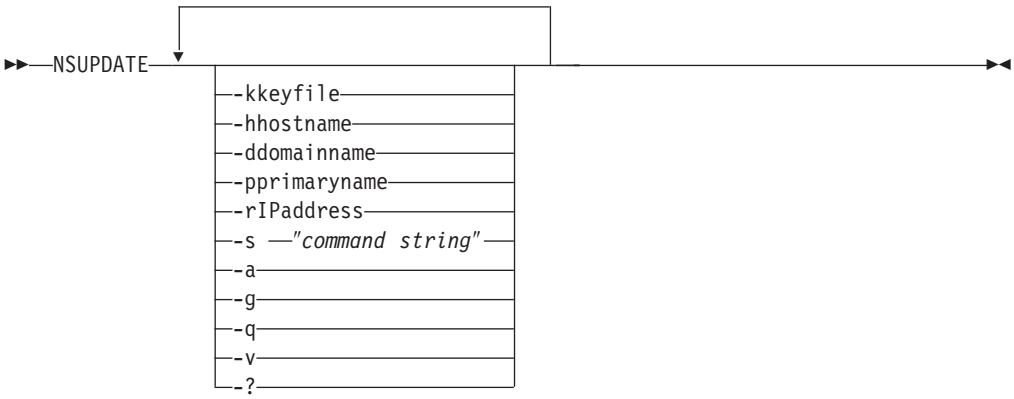
For help with onslookup commands from the command line, type `onslookup -?`.

Using the TSO NSUPDATE Command

Purpose

You can use NSUPDATE to create and execute DNS update operations on a host record. You can use this command in an interactive fashion (where you are prompted through a series of subcommands and associated input values), or if you know the sequence of operations and input values beforehand, you can use NSUPDATE in batch mode and specify a subcommand sequence in the -s command line parameter.

Format



Parameters

-kkeyfile
By default, NSUPDATE uses /etc/ddns.dat for storing and retrieving key information. Use the -k parameter to specify an alternate key file.

-hhostname
Specifies the name or alias of a remote host.

-ddomainname
Specifies the name of the dynamic domain for the host. Not needed if the -h parameter specifies a fully qualified host name.

-pprimaryname
Specifies the fully qualified host name of the DDNS server that is primary.

-rIPaddress
Specifies the IP address used to update PTR records. (You can not specify -r if you specify -h or -d.)

-s ″command string″
Command string is the string of subcommands with associated required input to be executed by NSUPDATE. Each subcommand is separated by a semicolon.

The following illustrates how the -s parameter is used to add an A Record with an expiration time of one hour (3600 seconds), and a signature 36 days (3110400 seconds):

```
NSUPDATE -h warpspeed.dynozone.sandbox -s a;9.67.96.10;s;36
>
The above example assumes that the key file entry for this host already exists
in /etc/ddns.dat and contains the fully-qualified name of the
primary DDNS server.
```

-a The administrator mode flag. Specifies that the records in the update are to be signed and authenticated.

-g Only generates a key file entry for the host, but does not register the key with the DDNS server.

-q Specifies quiet mode. When this option is used, no prompting or informational messages are displayed.

-v Used for debugging purposes. When used on the command line, this turns on verbose mode. When verbose mode is on, all the requests to, and responses from, the name server are displayed.

-? Displays help for NSUPDATE.

NSUPDATE Subcommands

The following subcommands can be used in the NSUPDATE command shell.

add Appends an ADD

delete Appends a DELETE

exists Appends an ADDNAMEEXIST

new Appends an ADDNAMENEW

sign Signs and sends the transaction(s)

ttl Sets the default TTL to be used in records on the update request. The default value is 4660.

quit Quits the program

NSUPDATE Examples

The following are example console sessions using **NSUPDATE** in interactive mode. All examples assume the administrator has already set up a zone key and that the private key component for the zone is included in the local /etc/ddns.dat.

How an Administrator Removes and Locks Out a Host Name

The following example demonstrates an administrator's input and the system responses when removing and locking out a host name. Administrator input is highlighted in bold:

1. Generate a New Key for the Host

```
[C:\]NSUPDATE -g -h warpspeed.dynozone.sandbox -p
netadmin.dynozone.sandbox
--- NSUPDATE Utility ---
---
Key Gen ...... succeeded ...
```

2. Delete a User's A and KEY RRs and add a New KEY RR for New, Administrator Generated Key

```
[C:\]NSUPDATE -a -h warpspeed.dynozone.sandbox -p
netadmin.dynozone.sandbox
--- NSUPDATE Utility ---

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> d
---
InitDDNSUpdate ...... succeeded ...
..rrtype (A,PTR,CNAME,MX,KEY,HINFO): a
....ip addr: *
DDNSUpdate_A (Delete *) ...succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> d
---
InitDDNSUpdate ...... succeeded ...
..rrtype (A,PTR,CNAME,MX,KEY,HINFO): key
DDNSUpdate_KEY DELETE *
succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> a
..rrtype (A,PTR,CNAME,MX,KEY,HINFO): key
```

```
DDNSUpdate_KEY (Add Flags 0000 Protocol 0 Algid 1
      Keylen 64 Key10-15Õ: AQPS80e7uGuuNIdA ...succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> s
..sig Expiration (secs from now, ENTER for 3600):
..sig KEY pad (ENTER for default of 3110400):
DDNSSignUpdate ...succeeded
DDNSFinalizeUpdate ...succeeded
DDNSSendUpdate ...succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> q

[C:\]
```

How an Administrator Creates an Alias for the Dynamic Zone

The following example demonstrates an administrator's input and the system responses when creating an alias for the dynamic zone. Administrator input is highlighted in bold:

```
[C:\]NSUPDATE -a -h ns-updates.dynozone.sandbox -p netadmin.dynozone.sandbox
--- NSUPDATE Utility ---

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> a
---
InitDDNSUpdate ...... succeeded ...
..rrtype (A,PTR,CNAME,MX,KEY,HINFO): cname
....hostname: netadmin
DDNSUpdate_CNAME (Add netadmin.dynozone.sandbox) ...succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> s
..sig Expiration (secs from now, ENTER for 3600):
..sig KEY pad (ENTER for default of 3110400):
DDNSSignUpdate ...succeeded
DDNSFinalizeUpdate ...succeeded
DDNSSendUpdate ...succeeded

Enter Action (Add,Delete,Exists,New,TTL,Send,Quit)
> q

[C:\]
```

Return Codes

Return codes are used to give the user feedback on the progress of his command. Following are the return codes, origination of the return codes, and explanations for the most common problems that you might encounter:

Return Code	Origin	Explanation
0	N/A	Successful.
-2	Local error	Input error.
-10	Local error	No key found in /etc/ddns.dat. A key is needed because either -f was specified or there is a KEY RR already in the name server data.
-11	Local error	Key in /etc/ddns.dat is not valid. Does not authenticate the user.

Return Code	Origin	Explanation
-12	Local error	No response received from the name server.
-1	Local error	Represents any other (local) error not specified above.
1	Server error	Format error. The name server was unable to interpret the request.
2	Server error	Server failure. The name server was unable to process this request because of a problem with the name server.
3	Server error	Name error. The domain name specified does not exist.
4	Server error	Not implemented. The name server does not support the specified operation code.
5	Server error	Refused. The name server refuses to perform the specified operation for security or policy reasons.
6	Server error	Alias error. A domain name specified in an update is an alias.
7	Server error	Name Exists error. A name already exists. This return code is only meaningful from a server in response to an ADDNAMENEW operation.
8	Server error	Record error. Indicates that a resource record (RR) does not exist. This return code is only meaningful from a server in response to a DELETE operation.
9	Server error	Zone error. Indicates that the update is to be performed on a zone for which the server is not authoritative, or that the records to be updated exist in more than one zone.
10	Server error	Ordering error. If an ordering mechanism is used (for example, a SIG RR or a SOA RR), this code indicates an ordering error. Time-signed problems are also indicated by this return code.

Using the TSO DIG Command

DIG is a program for querying Domain Name Servers, which enables you to:

- Exercise name servers
- Gather large volumes of domain name information
- Execute simple domain name queries

If you have a group of queries to be resolved, you must issue an NSLOOKUP command for each query. Compared to NSLOOKUP, the DIG command provides a larger range of options for controlling queries and screen output. You should use the DIG command only if the NSLOOKUP command cannot satisfy your request.

DIG Internal State Information

The internal state information of DIG determines the operation and results of your name server queries. You can configure the internal state information of DIG using the following methods, listed in order of preference:

1. TCP/IP client program configuration data set, *hlq*.TCPIP.DATA
2. DIG startup data set, *user_id*.DIG.ENV

3. Query options on the command line or in a batch data set

The *user_id*.DIG.ENV data set contains a list of query option defaults. This list is initialized from the *user_id*.DIG.ENV data set when DIG is invoked. The default values in *user_id*.DIG.ENV are used for all queries unless overridden by query flags on the command line. The defaults can be reset during a batch run by using the *-envset* flag on a batch data set line.

The *user_id*.DIG.ENV data set is created and updated using the *-envset* option, which writes the current defaults out to the data set after parsing the query options on the command line. The *-envset* option specified on the command line and the existing default values are saved in the *user_id*.DIG.ENV data set as the default environment for future invocations of DIG. The *user_id*.DIG.ENV data set is not reread when the environment is updated during batch queries and the *-envset* flag has no effect on subsequent queries in a batch data set. The *user_id*.DIG.ENV data set is written in nontext format, and cannot be viewed or edited.

DIG Command—Query Name Servers

Purpose

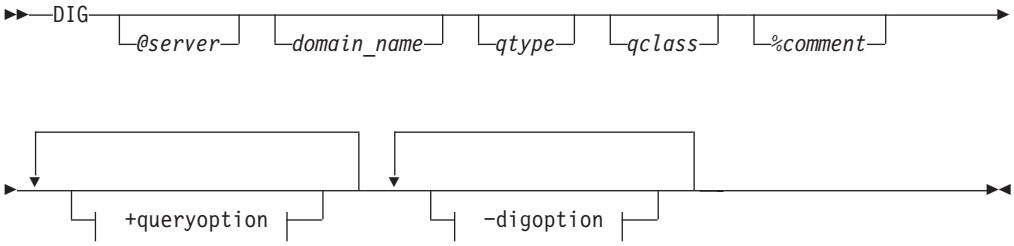
You can use DIG in command mode, where all options are specified on the invoking command line, or in batch mode, where a group of queries are placed in a data set and executed by a single invocation of DIG. DIG provides a large number of options for controlling queries and screen output, including most of the functions of NSLOOKUP.

You can create a data set for batch mode queries using the `-f` *`data_set`* option. The data set contains complete queries, one per line, that are executed in a single invocation of DIG. The keyword *DIG* is not used when specifying queries in a batch data set. Blank lines are ignored, and lines beginning with a number sign (#) or a semicolon (;) in the first column are comment lines.

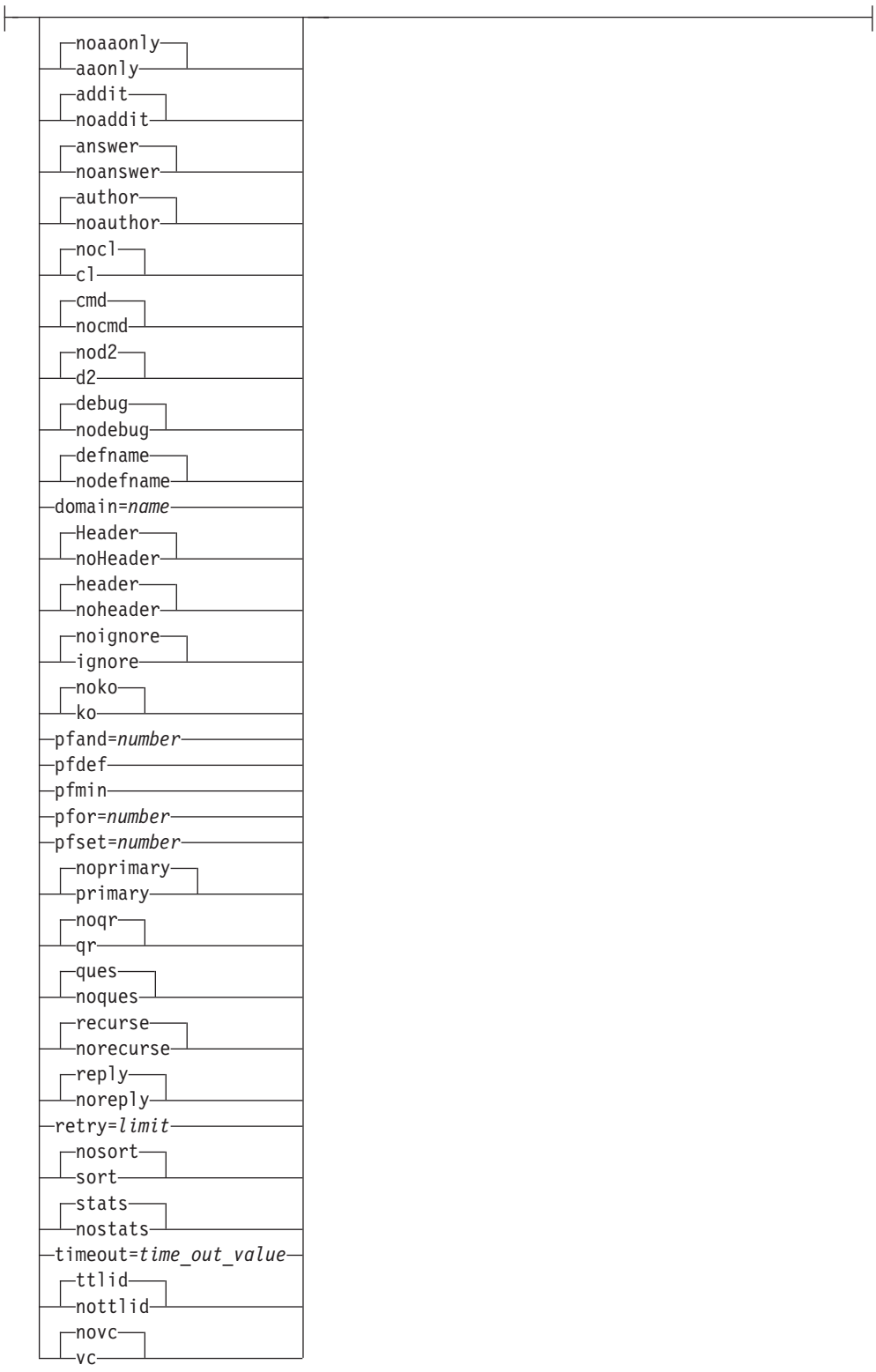
Options specified on the initial command line are in effect for all queries in the batch data set unless explicitly overridden. Several options are provided exclusively for use within batch data sets, giving greater control over DIG operation.

Some internal state information is retrieved from the *hlq*.TCPIP.DATA data set. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about the *hlq*.TCPIP.DATA data set.

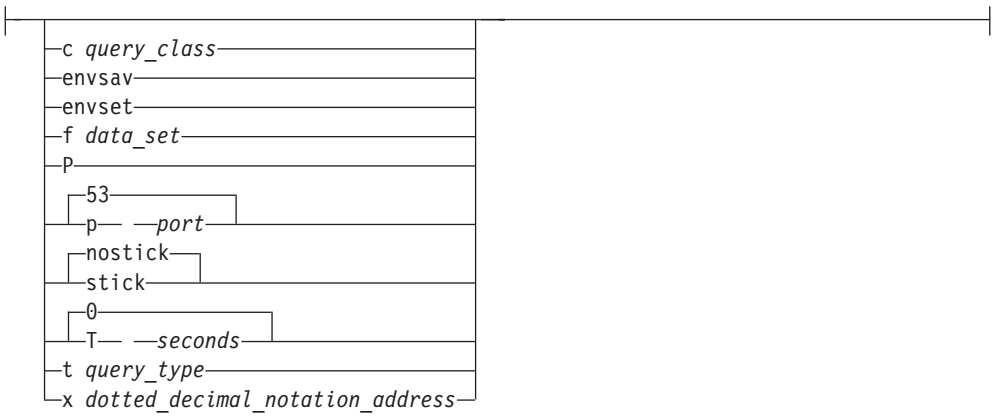
Format



+queryoption:


noaaonly
aaonly
addit
noaddit
answer
noanswer
author
noauthor
nocl
cl
cmd
nocmd
nod2
d2
debug
nodebug
defname
nodefname
domain=name
Header
noHeader
header
noheader
noignore
ignore
noko
ko
pfand=number
pfdef
pfmin
pfor=number
pfset=number
noprimary
primary
noqr
qr
ques
noques
recurse
norecurse
reply
noreply
retry=limit
nosort
sort
stats
nostats
timeout=time_out_value
ttlid
nottlid
novc
vc

−digoption:



Parameters

@server
Specifies the domain name or IP address of the name server to contact for the query. The default is the name server specified in the *hlq*.TCPIP.DATA data set.

If a domain name is specified, DIG uses the resolver library routines provided in the TCP/IP for MVS programming interface to map the name to an IP address.

domain_name
Specifies the name of the domain for which information is requested. If the domain name does not exist in the default domain specified in the *hlq*.TCPIP.DATA data set, you must specify a fully qualified domain name.

qtype
Specifies the type of query to be performed. DIG does not support the MAILA, MD, MF, and NULL query types. The wildcard query types are ANY, MAILB, and AXFR. See “Resource Records” on page 436 for more information about valid query types.

If the *qtype* option is omitted, the default query type is A (an address query).

qclass
Specifies which network class to request in the query. DIG recognizes only the IN, CHAOS, HESIOD, and ANY network classes.

%*comment*
Provides a means of including comments in a DIG command. Any characters following the percent (%) character up to the next space character (space or end-of-record) are ignored by DIG. This option is useful in batch data sets for annotating a command.

For example, using a dotted decimal notation IP address rather than a domain name removes any overhead associated with address mapping; however, this makes the command less readable. Therefore, in a batch data set you can include the domain name as a comment for readability.

+*queryoption*
Interprets the string following the plus sign (+) character as a query option. Query options have the format:

parameter[=*value*]

and are a superset of the SET subcommand options for NSLOOKUP.

aaonly
Accepts only authoritative responses to queries.

noaaonly
Accepts all responses to queries. This option is the default.

addit
Prints the additional section of the response. The additional section contains resource records that have not been explicitly requested, but could be useful. Refer to RFC 1035 for more information about this option. This option is the default.

noaddit
Does not print the additional section of the response.

answer
Prints the answer section of the response. The answer section contains the set of all resource records from the name server database that satisfy the query. This option is the default.

noanswer
Does not print the answer section of the response.

author
Prints the authoritative section of the response. The authoritative section contains resource records that specify the address of an authoritative name server for the query. This section is used when the name server queried cannot provide an authoritative answer. This option is the default.

noauthor
Does not print the authoritative section of the response.

cl Prints network class information for each of the resource records returned.

nocl
Does not print network class information for each of the resource records returned. This option is the default.

cmd
Echos the parsed options. This option is the default.

nocmd
Does not echo the parsed options.

d2
Prints the details of each query sent out to the network, including send time stamp and the timeout time stamp. When a server does not respond within the timeout period, DIG either sends the query to another server, or resends the query to the original server. The details of the query are visible when *d2* is set.

nod2
Does not print the details of each query sent out to the network. This option is the default.

debug
Directs DIG to print additional error messages. This option is the default.

nodebug
Directs DIG to not print additional error messages.

defname
Appends the default domain name to all unqualified domain names in a query. The default domain name is set by specifying the *+domain=name* option. This option is the default.

nodefname
Does not append the default domain name to all unqualified domain names in a query. This option causes the domain name specified to pass to the server without modification.

domain=*name*
Sets the default domain name to *name*. Initially the default domain name is obtained from the *hlq*.TCPIP.DATA data set. The validity of *name* is not verified. If the *defname* option is set, the domain name specified in *name* is appended to all unqualified domain names before the queries are sent to the name server.

Header
Prints the header line containing the operation code, returned status, and query identifier of each response. This option is distinct from the *header* option. This option is the default.

noHeader
Does not print the header line containing the operation code, returned status, and query identifier of each response.

header
Prints the query flags of each response. The query flags are defined in RFC 1035. This option is the default.

noheader
Does not print the query flags of each response.

ignore
Ignores truncation errors. Truncation errors occur when a response is too long for a single datagram.

noignore
Reports truncation errors. This option is the default.

ko
Keeps the virtual circuit open for queries in batch mode only. This option has no effect when used on the command line or when datagrams are used to transport queries (see the *novc* option).

noko
Does not keep the virtual circuit open for queries in batch mode only. This option is the default.

pfand=*number*
Performs a bitwise AND of the current print flags with the value specified in *number*. The number can be octal, decimal, or hexadecimal.

Note: To specify a number in octal, a zero is required in front of the number. To specify a number in hexadecimal, 0X is required in front of the number.

pfdef
Sets the print flags to their default values. The default print flag values are 0x2FF9. For query type AXFR, the print flag values are 0x24F9.

Note: To specify a number in octal, a zero is required in front of the number. To specify a number in hexadecimal, 0X is required in front of the number.

pfmin
Sets the print flags to the minimum default values. This option specifies that minimal information should be printed for each response. The minimum print flag values are `0xA930`.

Note: To specify a number in octal, a zero is required in front of the number. To specify a number in hexadecimal, 0X is required in front of the number.

pfor=*number*
Performs a bitwise OR of the current print flags with the value specified in *number*. The number can be octal, decimal, or hexadecimal.

Note: To specify a number in octal, a zero is required in front of the number. To specify a number in hexadecimal, 0X is required in front of the number.

pfset=*number*
Sets the print flags to the value specified in *number*. The number can be octal, decimal, or hexadecimal.

Note: To specify a number in octal, a zero is required in front of the number. To specify a number in hexadecimal, 0X is required in front of the number.

The print flags are represented by a 16-bit value. The following list describes the individual bits of the print flags in order of most-significant bit to least-significant bit.

0 Sort reply records

1 Unused

2 Display reply section

3 Display query section

4 Show basic header

5 Display time to live (TTL) in reply records

6 Show flags for query and reply

7 Show section headers with reply record totals

8 Show additional subsections

9 Show authoritative subsection

10 Show answer subsections

11 Show question subsections

12 Echo DIG command line

13 Display query class info in reply records

14 Unused

15 Display statistics

primary
Includes only the primary name server for the zone, or include the secondary name servers.

noprimary
Indicates that you should not use only the primary name server for the zone. This option is the default.

qr
Prints the outgoing query. The outgoing query consists of a header, question section and empty answer, additional, and authoritative sections. Refer to RFC 1035 for more information about outgoing queries.

noqr
Does not print the outgoing query. This option is the default.

ques
Prints the question section of a response. The question section contains the original query. This option is the default.

noques
Does not print the question section of a response.

recurse
Requests a recursive query when querying a name server. This option is the default.

norecurse
Specifies that a recursive query is not requested.

reply
Prints the response from the name server. This option is the default.

noreply
Does not print the response from the name server. When this option is disabled, other print flags that affect printing of the name server response are ignored and no sections of the response are printed.

retry=*limit*
Specifies the number of times a request is resent. When a request is sent and the timeout period expires for a response, the request is resent until the value specified in *limit* has been exceeded. The value specified in *limit* determines the number of attempts made to contact the name server. The default value for *limit* is retrieved from the *hlq*.TCPIP.DATA data set.

Setting *limit* to 0 disables DIG from contacting the name server. The result is an error message `no response from server`.

The retry procedure for DIG uses both the *limit* value and the timeout period. Each time a request is resent, the timeout period for the request is twice the timeout period used for the last attempt.

sort
Sorts resource records before printing. Records are sorted alphabetically on record type names.

nosort
Does not sort resource records before printing. This option is the default.

stats
Prints the query statistics including round trip time, time and date of query, size of query and response packets, and name of server used. This option is the default.

nostats
Does not print the query statistics.

timeout=*time_out_value*
Specifies the number of seconds to wait before timing out of a request. The default timeout value is retrieved from the *hlq*.TCPIP.DATA data set.

ttlid
Prints the time to live (TTL) for each resource record in a response. This option is the default.

nottlid
Does not print the TTL for each resource record in a response.

vc Uses a virtual circuit (TCP connection) to transport queries to the name server or datagrams. The default is retrieved from the *hlq*.TCPIP.DATA data set.

novc
Does not use a virtual circuit to transport queries to the name server or datagrams. This option is the default.

-*digoption*
Interprets the string following the hyphen (-) as a DIG option. The DIG options are either a parameter or a single character followed by a parameter.

c *query_class*
Specifies that the command-mode query or batch query retrieves resource records having the given network class. The *qclass* parameter, described on page 479, can also be used to specify the query class. In addition to the mnemonics, this option also accepts the equivalent numeric value that defines the class.

envsav
Directs DIG to save the environment specified on the current command line in the *user_id*.DIG.ENV data set. The DIG environment is described in "DIG Internal State Information" on page 475. This *hlq*.DIG.ENV data set initializes the default environment each time DIG is invoked.

envset
This option is valid for batch mode only. It directs DIG to set the default environment, see "DIG Internal State Information" on page 475, specified on the current line in the batch data set. This default environment remains in effect for all subsequent queries in the batch data set, or until the next line in the batch data set containing the *-envset* option is reached.

f *data_set*
Specifies a data set for DIG batch mode queries. The batch data set contains a list of queries that are to be executed in order. The keyword DIG is not used when specifying queries in a batch data set. Lines beginning with a number sign (#) or semicolon (;) in the first column are comment lines, and blank lines are ignored. Options that are specified on the original command line are in effect for all queries in the batch data set unless explicitly overwritten. The following is an example of a batch data set.

```
# A comment
; more comments
wurrup any in +noH =noqu -c IN

toolah +pfmin
```

Note: You must limit your query string to 99 characters to avoid error messages.

P Directs DIG to execute a PING command for response time comparison after receiving a query response. The last three lines of output from the following command are printed after the query returns:

```
PING server_name ( Length 56 Count 3
```

p *port*
Use the port number given when contacting the name server. The Domain Name System is a TCP/IP well-known service and has been allocated port 53. DIG uses 53 by default, but this option enables you to override the port assignment.

stick
Restores the default environment, see “DIG Internal State Information” on page 475, before processing each line of a batch data set. This flag is valid for batch mode only. If you set the *stick* option, queries in the batch data set are not affected by the options specified for preceding queries in the data set.

nostick
Causes the query option specified on the current line in the batch data set to remain in effect until the option is overridden by a subsequent query. The result of each query in the batch data set depends on the preceding queries. This option is the default.

T *seconds*
Specifies the wait time between successive queries when operating in batch mode. The default wait time is zero (do not wait).

t *query_type*
Specifies that the query retrieves resource records having the given resource record type. The *qtype* parameter on page 479 can also be used to specify the query type. In addition to the mnemonics, this parameter also accepts the equivalent numeric value that defines the type.

x *dotted_decimal_notation_address*
Simplifies the specification of a query for the `in-addr.arpa` domain. Normally these queries are made by specifying a query type of `PTR` for `nn.nn.nn.nn.in-addr.arpa`, where the four `nn` components are replaced by the dotted decimal notation IP address components in reverse order. This option enables you to make this query by simply specifying the dotted decimal notation IP address.

For example, the domain name corresponding to IP address `101.3.100.2` is found by a query for the domain name `2.100.3.101.in-addr.arpa`. You can use `DIG -x 101.3.100.2` rather than reversing the address and appending `in-addr.arpa`.

Examples

The following examples show how to use DIG to extract information from a name server. In Figure 18 on page 486, the router `wurrup` has two IP addresses, and

there are two name servers, wurrup being the primary name server. This network is described by a single zone in the domain naming hierarchy stored in the name servers.
In the examples, all queries are issued from the MVS uluru system.

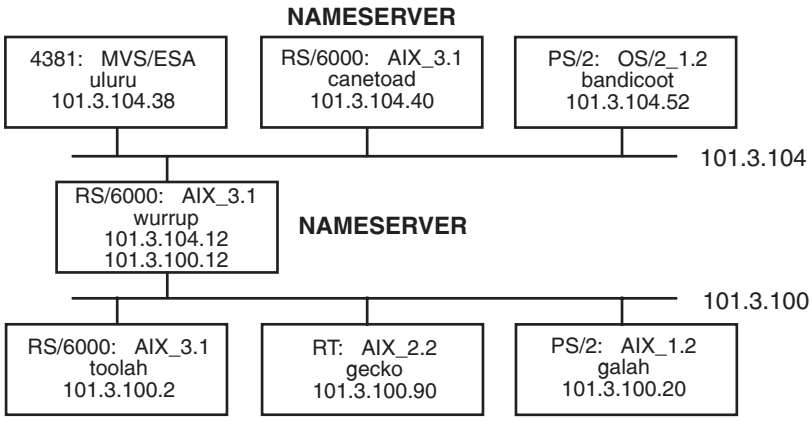


Figure 18. A TCP/IP Network

Create a default environment (default options) that gives minimal output from subsequent DIG commands:

```
System:
Ready
User:
DIG wurrup +noqu +noH +nohe +nocmd +noad +noau +nost +nocl
+nottl -envsav
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

The following queries show which part of the response output is controlled by each of the output control options. Each example enables or disables query options for tailoring output.

- Set the query type to ns, the query class to in, and print the additional section of the output:

  ```
  System:   Ready
  User:
  DIG fourex.oz ns in +ad
  System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 3
            ;; ANSWERS:
            fourex.oz NS      wurrup.fourex.oz
            fourex.oz NS      canetoad.fourex.oz
            ;; ADDITIONAL RECORDS:
            wurrup.fourex.oz  A       101.3.100.12
            wurrup.fourex.oz  A       101.3.104.12
            canetoad.fourex.oz        A       101.3.104.40
  ```

- Set the query type to ns, the query class to in, print the additional section of the output, but do not print the answer section:

```
System:   Ready
User:
DIG fourex.oz ns in +addit +noanswer
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 3
          ;; ADDITIONAL RECORDS:
          wurrup.fourex.oz  A       101.3.100.12
          wurrup.fourex.oz  A       101.3.104.12
          canetoad.fourex.oz        A       101.3.104.40
```

- Query a nonexistent domain and print the authoritative section of the output:

```
System:   Ready
User:
DIG noname +author
System:   ;; ->>HEADER<<- opcode: QUERY , status: NXDOMAIN, id: 3
          ; Ques: 1, Ans: 0, Auth: 1, Addit: 0
          ;; AUTHORITY RECORDS:
          fourex.oz SOA     wurrup.fourex.oz  adb.wurrup.fourex.oz (
                                  10003   ;serial
                                  3600    ;refresh
                                  300     ;retry
                                  3600000 ;expire
                                  86400 ) ;minim
```

In the previous example, the nonexistent domain name is *noname*.

- Use the default query options:

```
System:   Ready
User:
DIG wurrup
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Print the network class information:

```
System:   Ready
User:
DIG wurrup +cl
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  IN   A   101.3.104.12
          wurrup.FOUREX.OZ.  IN   A   101.3.100.12
```

- Echo the input query:

```
System:   Ready
User:
DIG wurrup +cmd
System:   ; <<>> DIG 2.0 <<>> wurrup +cmd
          ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Print the question section of the output:

```
System:   Ready
User:
DIG wurrup +qu
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.FOUREX.OZ, type = A, class = IN

          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Turn the header on:

```
System:   Ready
User:
DIG wurrup +H
System:   ;;>>HEADER<<- opcde: QUERY , status: NOERROR, id: 3
          ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Print the query flags:

```
System:   Ready
User:
DIG wurrup +he
System:   ;; flags: qr aa rd ra ; Ques: 1, Ans: 2, Auth: 0, Addit:
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Print the question section and the outgoing query:

```
System:   Ready
User:
DIG wurrup +qu +qr
System:   ; Ques: 1, Ans: 0, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.FOUREX.OZ, type = A, class = IN

          ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.FOUREX.OZ, type = A, class = IN

          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Print the query statistics including round-trip time:

```
System:   Ready
User:
DIG fourex.oz ns in +stats
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 3
          ;; ANSWERS:
          fourex.oz NS      wurrup.fourex.oz
          fourex.oz NS      canetoad.fourex.oz
          ;; Sent 1 pkts, answer found in time: 37 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:06:40 1992
          ;; MSG SIZE  sent: 24  rcvd: 116
```

- Print the TTL for each resource record:

```
System:   Ready
User:
DIG fourex.oz ns in +ttlid
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 3
          ;; ANSWERS:
          fourex.oz 9999999 NS      wurrup.fourex.oz
          fourex.oz 9999999 NS      canetoad.fourex.oz
```

- Enable extra debugging mode:

```
System:   Ready
User:
DIG wurrup +d2
System:   ;; res_mkquery(0, wurrup, 1, 1)
          ;; Querying server (# 1) address = 101.3.104.40
          ;; id = 3 - sending now: 4044656426 msec
          ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

The following examples show how options control the use and value of the default domain.

- Do not append the default domain name to unqualified domain names and print the question section of the response:

```
System:   Ready
User:
DIG wurrup +nodefname +qu
System:   ;;>>HEADER<<- opcde: QUERY , status: SERVFAIL, id: 3
          ; Ques: 1, Ans: 0, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup, type = A, class = IN
```

- Set the default domain name to fourexpd and print the question section of the response:

```
System:   Ready
User:
DIG wurrup +do=fourexpd +qu
System:   ;; ->>HEADER<<- opcode: QUERY , status: SERVFAIL, id: 3
          ; Ques: 1, Ans: 0, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.fourexpd, type = A, class = IN
```

- Set the query type to ns, the query class to in and sort the output:

```
System:   Ready
User:
DIG fourex.oz ns in +sort
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 3
          ;; ANSWERS:
          fourex.oz NS      canetoad.fourex.oz
          fourex.oz NS      wurrup.fourex.oz
```

- Query the domain at the address 101.3.100.20, and print the question section of the response:

```
System:   Ready
User:
DIG -x 101.3.100.20 +qu
System:   ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;    20.100.3.101.in-addr.arpa, type = ANY, class = IN

          ;; ANSWERS:
          20.100.3.101.in-addr.arpa.      PTR     galah.
```

- Retrieve resource records with a network class of ANY and print the question section of the response:

```
System:   Ready
User:
DIG wurrup -c any +qu
System:   ; Ques: 1, Ans: 2, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.FOUREX.OZ, type = A, class = ANY

          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
```

- Retrieve resource records with a query type of ANY and print the question section of the response:

```
System:   Ready
User:
DIG wurrup -t any +qu
System:   ; Ques: 1, Ans: 3, Auth: 0, Addit: 0
          ;; QUESTIONS:
          ;;      wurrup.FOUREX.OZ, type = ANY, class = IN

          ;; ANSWERS:
          wurrup.FOUREX.OZ.  A       101.3.104.12
          wurrup.FOUREX.OZ.  A       101.3.100.12
          wurrup.FOUREX.OZ.  HINFO   RS6000 AIX3.1
```

The following lists the batch data set, `test.digbat` used for this example. The default environment has been removed by discarding the *user_id*.DIG.ENV data set. The DIG command is omitted for all entries in the data set.

Note the effect of the *-envset* and *-stick* options on the output.

```
wurrup any in +noH +nohe +noqu +noad +noau -envset -stick
wurrup any in
toolah a in +d2
toolah a in
toolah a in +d2 -nostick
toolah a in
toolah a in +nod2
toolah a in
```

Specify the batch data set test.digbat:

```
System:  Ready
User:
DIG -f test.digbat

System:   ; <<>> DIG 2.0 <<>> DIG wurrup any in +noH +nohe +noqu +noad
          +noau -envset -stick
          ; Ques: 1, Ans: 3, Auth: 0, Addit: 0

          ;; ANSWERS:
          wurrup.FOUREX.OZ.  9999999 A       101.3.104.12
          wurrup.FOUREX.OZ.  9999999 A       101.3.100.12
          wurrup.FOUREX.OZ.  86400   HINFO   RS6000 AIX3.1

          ;; Sent 1 pkts, answer found in time: 20 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 95
```

```
System:   ; <<>> DIG 2.0 <<>> DIG wurrup any in
          ; Ques: 1, Ans: 3, Auth: 0, Addit: 0
          ;; ANSWERS:
          wurrup.FOUREX.OZ.  9999999 A       101.3.104.12
          wurrup.FOUREX.OZ.  9999999 A       101.3.100.12
          wurrup.FOUREX.OZ.  86400   HINFO   RS6000 AIX3.1
          ;; Sent 1 pkts, answer found in time: 112 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 95
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in +d2
          ;; res_mkquery(0, toolah, 1, 1)
          ;; Querying server (# 1) address = 101.3.104.40
          ;; id = 3 - sending now: 4046124888 msec
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A       101.3.100.2
          ;; Sent 1 pkts, answer found in time: 210 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A       101.3.100.2
          ;; Sent 1 pkts, answer found in time: 270 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in +d2 -nostick
          ;; res_mkquery(0, toolah, 1, 1)
          ;; Querying server (# 1) address = 101.3.104.40
          ;; id = 3 - sending now: 4046125037 msec
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A    101.3.100.2
          ;; Sent 1 pkts, answer found in time: 360 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in
          ;; res_mkquery(0, toolah, 1, 1)
          ;; Querying server (# 1) address = 101.3.104.40
          ;; id = 5 - sending now: 4046125101 msec
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A       101.3.100.2
          ;; Sent 1 pkts, answer found in time: 24 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in +nod2
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A       101.3.100.2
          ;; Sent 1 pkts, answer found in time: 19 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:57 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

```
System:   ; <<>> DIG 2.0 <<>> DIG toolah a in
          ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
          ;; ANSWERS:
          toolah.FOUREX.OZ.  9999999 A       101.3.100.2
          ;; Sent 1 pkts, answer found in time: 26 msec
          ;; FROM: FOUREXVM1 to SERVER: default -- 101.3.104.40
          ;; WHEN: Tue Mar 16 11:15:58 1992
          ;; MSG SIZE  sent: 31  rcvd: 47
```

Usage

The *queryoption* and *digoption* parameters are case sensitive and must be entered in lowercase. Domain names, query types, query classes, and the values associated with *queryoption* and *digoption* parameters are not case sensitive.

Using the OS/390 UNIX host Command

The OS/390 UNIX host command queries the configured name server to perform the following tasks:

- Identify the IP addresses associated with a specified DNS hostname
- Identify the DNS hostnames associated with a specified IP address

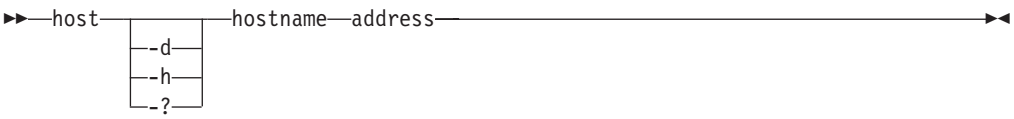
The host command, which uses the LE resolver, must be issued from within the OS/390 UNIX shell.

OS/390 UNIX host—Identify the Remote Host

Purpose

Use the OS/390 UNIX host command to identify the IP addresses associated with a specified DNS hostname or to identify the DNS hostnames associated with a specified IP address.

Format



Parameters

-d print trace messages for problem diagnosis

-h display the usage message

-? display the usage message

hostname
DNS hostname to look up

address
numeric address string to look up

Examples

The following example shows the command output.

```
host 204.146.18.33
EZZ8321I www.ibm.com has addresses 204.146.18.33

host www.ibm.com
EZZ8321I www.ibm.com has addresses 204.146.18.33
```

Using the OS/390 UNIX hostname Command

The OS/390 UNIX hostname command is used to display the fully-qualified DNS host name of the system. It supports several mechanisms for determining this information:

- LE resolver configuration (TCPIP.DATA) - default
- DNS lookup on the value returned by gethostname()
- gethostname()

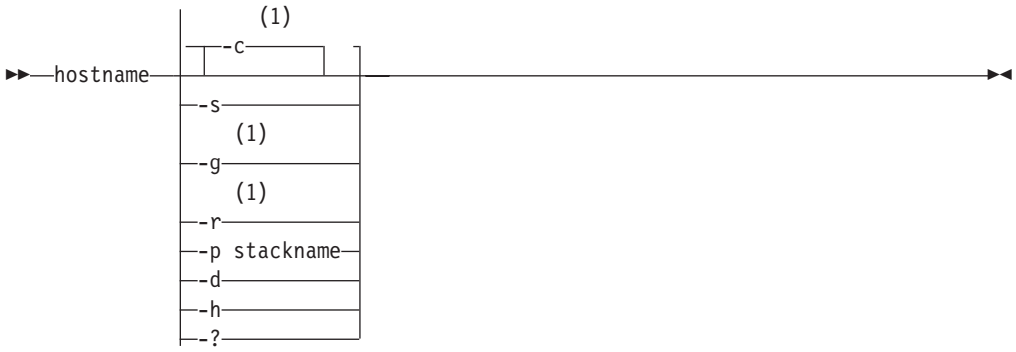
The OS/390 UNIX hostname command must be issued from within the OS/390 UNIX shell.

OS/390 UNIX hostname—Identify the Local Host

Purpose

Use the OS/390 UNIX hostname command to display the fully-qualified DNS host name of the local system.

Format



Notes:

1 Only one of the -c, -g, and -r can be specified.

Parameters

-c use the TCPIP.DATA configuration

-s print the short name of the host (without the DNS domain name)

-g use gethostname() result

-r use DNS lookup on gethostname() result

-p *stackname*
use this AF_INET stack

-d print trace messages for problem diagnosis

-h display the usage message

-? display the usage message

Usage

Only one of the -c, -g, and -r parameters can be specified.

Using the OS/390 UNIX dnsdomainname Command

The OS/390 UNIX dnsdomainname command is used to display the DNS domain name of the system. It supports several mechanisms for determining this information:

- LE resolver configuration (TCPIP.DATA) - default
- DNS lookup on the value returned by gethostname()
- gethostname()

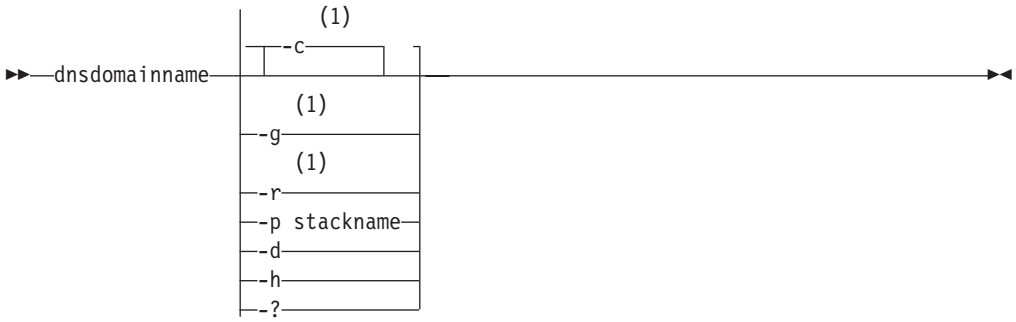
The OS/390 UNIX dnsdomainname command must be issued from within the OS/390 UNIX shell.

OS/390 UNIX dnsdomainname—Display the DNS Domain Name

Purpose

Use the OS/390 UNIX dnsdomainname to display the DNS domain name of the system.

Format



Notes:

1 Only one of the -c, -g, and -r can be specified.

Parameters

-c use the TCPIP.DATA configuration

-g use gethostname() result

-r use DNS lookup on gethostname() result

-p *stackname*
use this AF_INET stack

-d print trace messages for problem diagnosis

-h display the usage message

-? display the usage message

Usage

- Only one of the -c, -g, and -r parameters can be specified.
- If the DNS domain name cannot be retrieved, an error message will be displayed.

Using the OS/390 UNIX domainname Command

The OS/390 UNIX domainname command is a synonym for the OS/390 UNIX dnsdomainname command documented in the previous section. See “Using the OS/390 UNIX dnsdomainname Command” on page 497 for information on using this command.

Note: On some operating systems, the domainname command displays the system NIS/YP domain name, which may or may not be the same as the system DNS domain name. Portable shell scripts should use dnsdomainname rather than domainname if this distinction is important.

Chapter 13. Managing TCP/IP Network Resources with SNMP

This chapter describes how to use the Simple Network Management Protocol (SNMP) commands and details what support the CS for OS/390 SNMP agent and subagents provide.

Overview of SNMP

SNMP is a set of protocols that describe management data and the protocols for exchanging that data between heterogeneous systems. The protocols include both the description of the management data, defined in the Management Information Base (MIB), and the operations for exchanging or changing that information. By implementing common protocols, management data can be exchanged between different platforms with relative ease.

Three primary functional entities are defined in SNMP: managers, agents, and subagents. A manager is a management application that typically requests management data. This application could be a simple command-line interface (such as the OS/390 UNIX osnmp command and the Netview SNMP command supported in CS for OS/390) or a more complex application (such as a MIB browser or a performance monitoring tool). The agent is the server at the host that responds to requests for management data. On CS for OS/390, the server is called OSNMPD. The agent is often assisted in its support of management data by one or more subagents that are responsible for providing support for particular sets of management data.

The protocols used to exchange information between SNMP agents and managers is SNMP. The interface used to exchange information between the CS for OS/390 SNMP agent (and most other agents supported by IBM) and SNMP subagents is the Distributed Protocol Interface (DPI). The two management commands provided by CS for OS/390 are:

- NetView SNMP command
- OS/390 UNIX osnmp command

The NetView SNMP and OS/390 UNIX osnmp commands are management applications that can be used to monitor and control network elements. When you use NetView SNMP with TCP/IP, you require the NetView program to provide an end-user interface to the SNMP client. A NetView operator can use the SNMP command to communicate with SNMP agents. NetView acts like an SNMP client. The osnmp command acts like an SNMP client in the OS/390 shell.

Note: “Appendix D. Management Information Base (MIB) Objects” on page 571 lists the MIB objects supported by the SNMP agent and subagents shipped with CS for OS/390.

Overview of CS for OS/390 SNMP Version 3

SNMP Version 3 (SNMPv3), defined in RFCs 2271 through 2275 issued in January, 1998, is the standards-based solution to the previous weaknesses of SNMP security. The SNMPv3 architecture is modularized so that portions of it can be enhanced over time without requiring the entire architecture to be replaced. SNMPv3 defines a framework that consists of (among other things):

- A message processing model (SNMPv3)

- A security model (user-based security)
- An access control model (view-based access control)

The framework is structured so that multiple models can be supported concurrently and replaced over time. For example, although there is a new message format for SNMPv3, messages created with the SNMPv1 and SNMPv2 formats can still be supported. Similarly, the user-based security model can be supported concurrently with community-based security models previously used.

Sample Command Lists for NetView SNMP

Two sets of sample NetView command lists are supplied. One set is written in CLIST (SNMPCLST.README), and the other set is written in REXX (SNMPREXX.README). You should use CLIST if your host system does not support REXX.

You can issue SNMP requests with the two sets of sample command lists that are shipped with the TCP/IP product, with command lists that you write, or directly from the NetView command line. The SNMPCLST and the SNMPREXX commands invoke the main panel from which a NetView operator can execute most of the SNMP requests in full-screen mode. The SNMPCLST command uses a set of command lists written in the NetView Command List language; the SNMPREXX command uses a set of REXX command lists.

For more information about these sample command lists, refer to the SNMPCLST.README and SNMPREXX.README data sets on the product tape. These files are to be used as samples only and may not reflect the most recent MIB variable support.

The NetView SNMP Command

Purpose

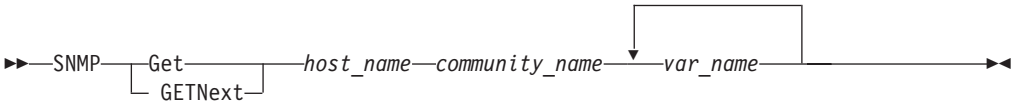
To issue an SNMP request from NetView, use the SNMP command. The SNMP command provides SNMP manager function with the NetView program to query SNMP agents for network management information.

The NetView SNMP command uses the SNMP Query Engine to issue SNMP requests to agents and to process SNMP responses returned by agents. The SNMP command supports issuance of SNMPv1 requests.

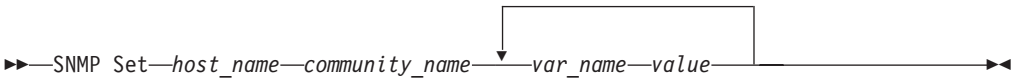
Note: The CS for OS/390 SNMP agent supports SNMPv1, SNMPv2c, and SNMPv3 requests.

Format

Getting MIB Variables



Setting the MIB Variables



Finding an ASN.1 Variable Name

Forwarding Traps

Stop Forwarding Traps

Pinging a Node

Parameters

asn.1_name
Specifies the MIB object, using its object identifier in ASN.1 notation. You can specify only one variable. Additional arguments are ignored.

community_name
Specifies the community name used to access the specified variables at the destination SNMP agent.

Note: Community names are case sensitive. SNMP commands issued from the NetView console are converted to uppercase. Those issued from REXX execs are not converted to uppercase.

filter_id
Specifies the trap filter ID.

When you request traps using the SNMP TRAPSON command, it returns a request number or *filter_id*, which the SNMP Query Engine associates with the TRAPSON request. To stop receiving traps, specify this *filter_id* in the TRAPSOFF request.

host_name
Specifies the destination host to which you want to send a request. The host can be specified with its name or with its IP address in dotted decimal notation.

net_desired
Specifies the network from which you want to receive traps.

net_mask
Specifies, in dotted decimal notation, the network mask to be evaluated with the IP address of incoming traps. The dotted decimal IP address is ANDed with this mask.

value
Specifies the value to be set by the SET function. On the Set command from the NetView console, a value is enclosed in single quotes, not double quotes. From the panels, you can specify no quotes, single quotes (’), or double quotes (”). No quotes and single quotes work the same. If you specify double quotes, you get double quotes as part of the value.

var_name
Specifies one or more MIB variable names to be retrieved or set. You can specify the textual names or ASN.1 notation (for example, sysDescr.0 or 1.3.6.1.2.1.1.1.0). The SNMP Query Engine can accept a maximum of 10 variables for each request.

All MIB variables that are defined as part of a sequence represent variables that can have more than one occurrence. These variables require an instance identifier appended to the end of the variable name to identify which occurrence of the variable is being requested.

SNMP request types

Get
Sends a request to an SNMP agent for a specific management information base (MIB) variable.

GETNext
Sends a request to an SNMP agent for the next MIB variable that lexicographically follows the *var_name* specified.

MIBvname
Requests the textual name of an ASN.1 MIB object.

PING
Obtains the minimum round-trip response time from the Query Engine to a specific node.

Set
Sends a request to an SNMP agent to set a specific MIB variable.

TRAPson
Requests that the SNMP Query Engine listen on well-known port 162 for SNMP traps and forward them to the NetView program, which displays trap information when they occur.

TRAPSOFf
Causes the SNMP Query Engine to stop listening on port 162 for SNMP traps and stop forwarding them to the NetView program.

Usage

- If you start and stop NetView, you must do the same to the SNMP Query Engine.
- When the SNMP command is issued from the NetView Command Facility command line, all input is translated to uppercase (standard NetView format) before it is sent to the SNMP Query Engine.
- When the SNMP command is issued from a CLIST, input is passed in whatever case it was passed from the CLIST (for example, mixed case).
- The textual names for the variables passed to the query engine are compared against the entries in the MIBDESC.DATA file. This comparison is not case sensitive.
- If multiple variables are specified with the GET, GETNext, or SET commands, they are all packaged in one SNMP PDU to be sent to the agent.
- If multiple SNMP requests are issued, the responses may not be received in the same order the requests are issued.
- The SNMP agent can receive SNMP requests over any interface.
- The SNMP Query Engine treats numbers with leading zeros as octal numbers. Therefore, do not use leading zeros.
- If an SNMP request is issued with the wrong community name, it could receive multiple AUTHENTICATION FAILURE traps with the same *filter_id* but different time stamps from the same host. This is because the SNMP Query Engine retries the request if a response is not received from the host, and each attempt causes the host to generate an AUTHENTICATION FAILURE trap.

Return Codes

The following table lists the return codes generated by SNMP.

Return Code	Description
1	Error from DSIGET, cannot continue
2	Incorrect function specified
3	Missing SNMP function

Return Code	Description
4	Not enough parameters
5	Missing variable name
6	Missing variable value
7	Missing or incorrect host name
8	Missing community name
9	SNMPIUCV not active
10	Error from DSIMQS
11	Incorrect *net_mask*/desired network
12	Missing/Incorrect trap *filter_id*
1001+	All return codes above 1000 indicate that the command was successful.

Examples

- **Retrieving the MIB variable**

  For example, if you know:

```
hostname            -  anyhost
IP address          -  129.34.222.72
community name      -  public
variable name       -  sysDescr.0
asn.1 variable name -  1.3.6.1.2.1.1.1.0
variable name       -  sysObjectID.0
asn.1 variable name -  1.3.6.1.2.1.1.2.0
variable name       -  sysUpTime.0
asn.1 variable name -  1.3.6.1.2.1.1.3.0
```

  You can issue the following SNMP GET commands:

```
snmp get 129.34.222.72 public 1.3.6.1.2.1.1.1.0
snmp get 129.34.222.72 public sysDescr.0
snmp get anyhost public 1.3.6.1.2.1.1.1.0
snmp get anyhost public sysDescr.0
snmp get anyhost public sysObjectID.0
snmp get anyhost public sysUpTime.0
snmp get anyhost public sysDescr.0 sysObjectID.0 sysUpTime.0
```

  After the last SNMP GET command is completed, you get a message similar to the following:

```
SNM050I SNMP Request 1001 from NETOP accepted, sent to Query Engine
```

  When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form:

```
SNM040I SNMP Request 1001 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.1.1.0
SNM043I Variable value type: 9
SNM044I Variable value: AIX 2.2.1 SNMP Agent Version 1.0
SNM042I Variable name: 1.3.6.1.2.1.1.2.0
SNM043I Variable value type: 3
SNM044I Variable value: 1.3.6.1.4.1.2.1.1
SNM042I Variable name: 1.3.6.1.2.1.1.3.0
SNM043I Variable value type: 8
SNM044I Variable value: 98800
SNM049I SNMP Request 1001 end of response
```

- **Retrieving the next MIB variable**

  For example, if you know:

```
hostname            -  anyhost
IP address          -  129.34.222.72
community name      -  public
variable name       -  ifAdminStatus (in ifTable)
asn.1 variable name -  1.3.6.1.2.1.2.2.1.7
```

You can issue an SNMP GETNext command in one of the following ways:

```
snmp getnext 129.34.222.72 public 1.3.6.1.2.1.2.2.1.7.0
snmp getnext 129.34.222.72 public ifAdminStatus.0
snmp getnext anyhost public 1.3.6.1.2.1.2.2.1.7.0
snmp getnext anyhost public ifAdminStatus.0
```

The GETNext command is completed in the same manner as the GET command, and you receive an asynchronous response similar to the following:

```
SNM040I SNMP Request 1001 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.2.2.1.7.1
SNM043I Variable value type: 1
SNM044I Variable value: 1
SNM049I SNMP Request 1001 end of response
```

In this example, the first instance of the variable has a status of 1 or greater (ends in `7.1`).

You can then issue another GETNext command in one of the following ways:

```
snmp getnext 129.34.222.72 public 1.3.6.1.2.1.2.2.1.7.1
snmp getnext 129.34.222.72 public ifAdminStatus.1
snmp getnext anyhost public 1.3.6.1.2.1.2.2.1.7.1
snmp getnext anyhost public ifAdminStatus.1
```

The GETNext command is completed in the same manner as the GET command, and you receive an asynchronous response similar to the following:

```
SNM040I SNMP Request 1002 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.2.2.1.7.2
SNM043I Variable value type: 1
SNM044I Variable value: 1
SNM049I SNMP Request 1002 end of response
```

In this example, the second instance of the variable has a status of 1 or greater (ends in `7.2`).

You can then issue another GETNext command in one of the following ways:

```
snmp getnext 129.34.222.72 public 1.3.6.1.2.1.2.2.1.7.2
snmp getnext 129.34.222.72 public ifAdminStatus.2
snmp getnext anyhost public 1.3.6.1.2.1.2.2.1.7.2
snmp getnext anyhost public ifAdminStatus.2
```

The GETNext command is completed in the same manner as the GET command, and you receive an asynchronous response similar to the following:

```
SNM040I SNMP Request 1003 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.2.2.1.8.1
SNM043I Variable value type: 1
SNM044I Variable value: 1
SNM049I SNMP Request 1003 end of response
```

- **Setting the MIB variable**

  For example, if you know:

```
hostname            -  anyhost
IP address          -  129.34.222.72
community name      -  publicw
variable name       -  ifAdminStatus
asn.1 variable name -  1.3.6.1.2.1.2.2.1.7.1
     (instance 1)
```

You can then issue an SNMP SET command in one of the following forms to set the administrative status of the first interface in the ifTable (first instance) to test:

```
snmp set 129.34.222.72 publicw 1.3.6.1.2.1.2.2.1.7.1 3
snmp set 129.34.222.72 publicw IfAdminStatus.1 3
snmp set anyhost publicw 1.3.6.1.2.1.2.2.1.7.1 3
snmp set anyhost publicw ifAdminStatus.1 3
```

After the command is completed, you receive a message similar to the following:

```
SNM050I SNMP Request 1001 from NETOP accepted, sent to Query Engine
```

When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form:

```
SNM040I SNMP Request 1001 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.2.7.1
SNM043I Variable value type: 1
SNM044I Variable value: 3
SNM049I SNMP Request 1001 end of response
```

If a SET request is attempted against an object for which the target agent/subagent does not allow SETs, you receive:

```
– noSuchName for SNMPv1 requests
```

"Appendix D. Management Information Base (MIB) Objects" on page 571, identifies the objects supported by the CS for OS/390 SNMP agent and subagents and the level of access supported for each object.

Note: The variable being set must be present in the MIBDESC.DATA data set for the Query Engine to determine the syntax to use when encoding the SNMP PDU.

- **Receiving a trap**

  The SNMP TRAPSON command permits the specification of a filtering condition, that enables the Query Engine to perform filtering. The SNMP TRAPSON command assigns a unique request number to each filter (also called a *filter_id*) and returns this number in a message and in the return code. This *filter_id* is the argument to an SNMP TRAPSOFF command, which is used to stop receiving traps that pass this filter.

  For example, if you know:

```
IP address          -  129.34.222.72
net mask            -  255.255.255.255
```

  You can issue the following SNMP TRAPSON commands:

```
snmp trapson
snmp trapson 255.255.255.255 129.34.222.72
```

  The first command receives all traps (the default is a mask of 0 and a desired network of 0). The second command receives traps only from the specific host 129.34.222.72.

After the command is completed, you receive a message similar to the following:

```
SNM050I SNMP Request 1001 from NETOP accepted, sent to Query Engine
```

The number returned in the message (1001 in the previous example) is used as the *filter_id*. This *filter_id* is displayed in the header message of traps passed by this filter. The *filter_id* is used in the TRAPSOFF command to turn the filter off.

When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form to indicate that the TRAPSON request was accepted:

```
SNM040I SNMP Request 1001 from NETOP Returned the following response:
SNM045I Major error code: 0
SNM046I Minor error code: 0
SNM047I Error index: 0
SNM048I Error text: no error
SNM049I SNMP Request 1001 end of response
```

When traps arrive, the NetView program displays each trap with a multiline message in the following form. This multiline message is sent to the NetView operator who is designated as the authorized receiver (AUTH MSGRECVR=YES in the operator profile); it might not show up on the console of the operator who issues the TRAPSON command.

```
SNM030I SNMP request 1001 received following trap:
SNM031I Agent Address: 129.34.222.34
SNM032I Generic trap type: 4
SNM033I Specific trap type: 0
SNM034I Time stamp: 472600
SNM035I Enterprise Object ID: 1.3.6.1.4.1.2.1.1
SNM039I SNMP request 1001 End of trap data
```

After the TRAPSON command has been issued, traps can start to arrive asynchronously. They can even arrive after the operator who issued the TRAPSON command has logged off. Often, a TRAPSON command is issued by a CLIST, and the received trap data triggers another CLIST to handle the trap data. Therefore, the messages in the range SNM030—SNM039 are sent to the authorized receiver. For a NetView operator to see the traps, the operator must have the following statement in the NetView Operator profile:

```
AUTH MSGRECVR=YES
```

However, only one operator receives the message. The message also goes to the log file, so you can always browse the log file to see trap data. Additionally, you can assign trap messages to go to a specific operator using the NetView ASSIGN operator command.

In the response to the SNMP TRAPSON request, not all lines need to be present; but the first line is always message SNM040I, and the last line is always message SNM049I.

For the multiline trap message, not all lines need to be present; but the first line is always message SNM030I, and the last line is always message SNM039I.

Additional messages (SNM036I—SNM038I) could be present if the trap has additional data.

If a variable value is too long, message SNM038 might not fit on an 80-character line. If this happens, the value is split and multiple SNM038 messages are displayed.

The SNMP trap data always displays the variable name in ASN.1 notation. You can use SNMP MIBVNAME to obtain the textual name for the variable.

A trap always shows the agent address in the form of an IP address in dotted decimal notation.

You can issue multiple TRAPSON requests, with either the same or a different filter. If a trap passes multiple filters, the trap is sent to the NetView program multiple times. However, in the NetView program, the header and trailer lines (messages SNM030I and SNM039I) of the duplicate trap are different, because they contain the *filter_id* (request number) by which the trap was forwarded. Different types of traps from different hosts can have the same *filter_id*, if these traps pass the same trap filter.

The SNMP Query Engine can forward only those traps that it receives. Each agent has a trap destination table, which lists all the hosts that should receive that agent's traps. The host name of your system should be in the trap destination table of all agents from which you want to receive traps.

- **Stop listening for traps**

  For example, if you know the *filter_id* is 1001, you can issue the following SNMP TRAPSOFF command to tell the SNMP Query Engine to quit sending traps that would pass filter 1001:

  ```
  snmp trapsoff 1001
  ```

  The command completes with a message similar to the following:

  ```
  SNM050I SNMP Request 1001 from NETOP accepted, sent to Query Engine
  ```

  When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form to indicate that the TRAPSOFF request was accepted.

  ```
  SNM040I SNMP Request 1002 from NETOP Returned the following response:
  SNM045I Major error code: 0
  SNM046I Minor error code: 0
  SNM047I Error index: 0
  SNM048I Error text: no error
  SNM049I SNMP Request 1002 end of response
  ```

  Only one *filter_id* for each SNMP TRAPSOFF command can be passed. Extraneous arguments are ignored.

- **Finding the name of an ASN.1 variable**

  For example, if you have a trap that tells you:

```
SNM030I SNMP request 1001 received following trap:
SNM031I Agent Address: 129.34.222.34
SNM032I Generic trap type: 2
SNM033I Specific trap type: 0
SNM034I Time stamp: 472600
SNM035I Enterprise Object ID: 1.3.6.1.4.1.2.1.1
SNM036I Variable name: 1.3.6.1.2.1.2.2.1.1
SNM037I Variable value type: 1
SNM038I Variable value: 2
SNM039I SNMP request 1001 End of trap data
```

You can issue the following SNMP MIBVNAME command to find the textual MIB variable name:

```
snmp mibvname 1.3.6.1.2.1.2.2.1.1
```

The command completes with a message similar to the following:

```
SNM050I SNMP Request 1002 from NETOP accepted, sent to Query Engine
```

When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form:

```
SNM040I SNMP Request 1002 from NETOP Returned the following response:
SNM042I Variable name: 1.3.6.1.2.1.2.2.1.1
SNM043I Variable value type: 9
SNM044I Variable value: ifIndex
SNM049I SNMP Request 1002 end of response
```

Only one ASN.1 variable name can be passed for each SNMP MIBVNAME command. Additional parameters are ignored.

- **Pinging a node**

  For example, if you know:

  ```
  nodename            -  anynode
  IP address          -  129.34.222.72
  ```

  You can issue the following SNMP PING commands:

  ```
  SNMP PING ANYNODE
  SNMP PING 129.34.222.72
  ```

  The command completes with a message similar to the following:

  ```
  SNM050I SNMP Request 1001 from NETOP accepted, sent to Query Engine
  ```

  When the response arrives in the NetView program (asynchronously), it displays the response as a multiline message in the following form:

  ```
  SNM040I SNMP Request 1001 from NETOP Returned the following response:
  SNM042I Variable name: 1.3.6.1.4.1.2.2.1.3.2.129.34.222.72
  SNM043I Variable value type: 1
  SNM044I Variable value: 26
  SNM049I SNMP Request 1001 end of response
  ```

  The Query Engine issues one PING (an ICMP echo on a raw socket) and returns the value in milliseconds in an IBM-defined SNMP variable minRTT. Because only one PING is issued, this is also the average and the maximum response time.

  If the PING does not respond, the Query Engine retries twice, once after one second and again after two seconds (Query Engine default retry mechanism). If a

response is not received after all retries have been exhausted, a variable value of -1 is returned to indicate that a reply was not received.

The 129.34.222.72 in the example for the SNMP PING command represents an instance of the IBM variable minRTT.

Only one node name can be passed for each SNMP PING command.

SNMP uses ICMP Echo to send a PING command to the remote host. No SNMP PDU exchange with the remote host occurs. Therefore, a successful SNMP PING indicates only that the remote host is active and reachable. It does not indicate that the SNMP agent at the remote host is active, or that the SNMP manager can send requests to the SNMP agent if it is active.

Usage

- The SNMP response always displays the variable name in ASN.1 notation. You can use SNMP MIBVNAME to obtain the textual name for the variable.
- If you issue a GET for multiple variables, messages SNM042—SNM044 are displayed for each variable.
- When you issue a GET for multiple variables, they are returned in the same sequence as requested. In the example on page 506, GET was issued for `sysDescr.0 sysObjectID.0 sysUpTime.0.`. The same 3 variables are returned in the response.
- If an error was detected, messages SNM042–SNM044 might not be present. You can get (in addition to other messages) error messages in the following forms (all as part of multiline message SNM040I):

```
SNM045I Major error code: n
SNM046I Minor error code: y
SNM047I Error index: z
SNM048I Error text: message text
```

- If a variable value is too long, message SNM044 might not fit on an 80-character line. If this happens, the value is split and multiple SNM044 messages are displayed.
- According to RFC 1157, a message exchanged between SNMP entities (including version identification and community name) can be as small as 484 octets. If you specify up to 10 variables in a GET/GETNext command, the names could be short enough to send the GET command to the SNMP agent, but the response could be too long to fit in the message. As a result, you receive a `tooBig` error.
- If one (or more) of the variables requested results in an error, all variables listed after the first variable in error are ignored, and data is not returned for them.
- To correctly retrieve the next variable for the GETNext command, you must specify an instance identifier as part of the variable name. If the variable has only one occurrence, or if you want the first occurrence of a table variable is desired, you should use .0 as the instance identifier.
- The GETNext command is used to interrogate a table (for example, the interface table) or an array. You can issue a GETNext command at the start of a table (use instance 0.0). The first element in the table is returned. The process continues in a loop, performing GETNext requests on the previously obtained variable name, until the name of the variable returned no longer has the same prefix as the one at the start of the table. This condition occurs when the GETNext request returns a variable that is in the next group.

Context

For information about the variable ibmMvsRPingResponseTime, which enables you to send remote PING commands, see “SNMP Remote PING” on page 533.

For a list of variables supported by the CS for OS/390 IP agent, see “Appendix D. Management Information Base (MIB) Objects” on page 571.

Host Name Resolution

When a NetView SNMP request uses a symbolic host name rather than an IP address, the SNMP Query Engine uses the standard gethostbyname() function to look up the IP address of that host. The IP address is then saved in an in-memory cache for future reference. Use of this cache improves the performance of subsequent requests for the same host. Refer to the *OS/390 C/C++ Run-Time Library Reference* for more information about gethostbyname().

Note: Because the cache cannot be refreshed, if the mapping between host names and IP addresses changes, you must restart the SNMP Query Engine (the SQESERV module) to rebuild the cache. You must also restart the SNMP Query Engine after a host name is added to the name server data base.

Major and Minor Error Codes and SNMP Value Types

The following are the possible major and minor error codes and variable value types that can be returned in a NetView SNMP response or trap.

- The major error code can have one of the following values:

Value	Major Error Code
0	No error detected
1	SNMP agent reported error
2	Internally detected error

- The minor error code can have one of the following values when the major error code indicates that an SNMP agent detected an error (1):

Value	SNMP Agent Detected Minor Error Code
0	No error
1	Too big
2	No such name
3	Incorrect value
4	Read only
5	General error

- The minor error code can have one of the following values when the major error code indicates that an internal error was detected (2):

Value	Internal Minor Error Code
0	No error
1	Protocol error
2	Out of memory
3	No response–all retries failed
4	Some I/O error occurred
5	Illegal request
6	Unknown host specified
7	Unknown MIB variable
8	No such filter
9	Too many variables specified

- If the major error code indicates that an SNMP agent detected the error (1), the error index indicates the position of the first variable in error.
- The variable value type is one of the following (as specified in RFC 1155 and RFC 1156):

Value	Value Type
0	Text representation
1	Number (integer, signed)
2	Binary data string
3	Object identifier
4	Empty (no value)
5	Internet address
6	Counter (unsigned)
7	Gauge (unsigned)
8	Time ticks (1/100ths seconds)
9	Display string

Note: The binary data string is displayed in the NetView program as a contiguous string of hexadecimal characters (for example, X'0123' is displayed as `0123`).

The OS/390 UNIX osnmp/snmp Command

Purpose

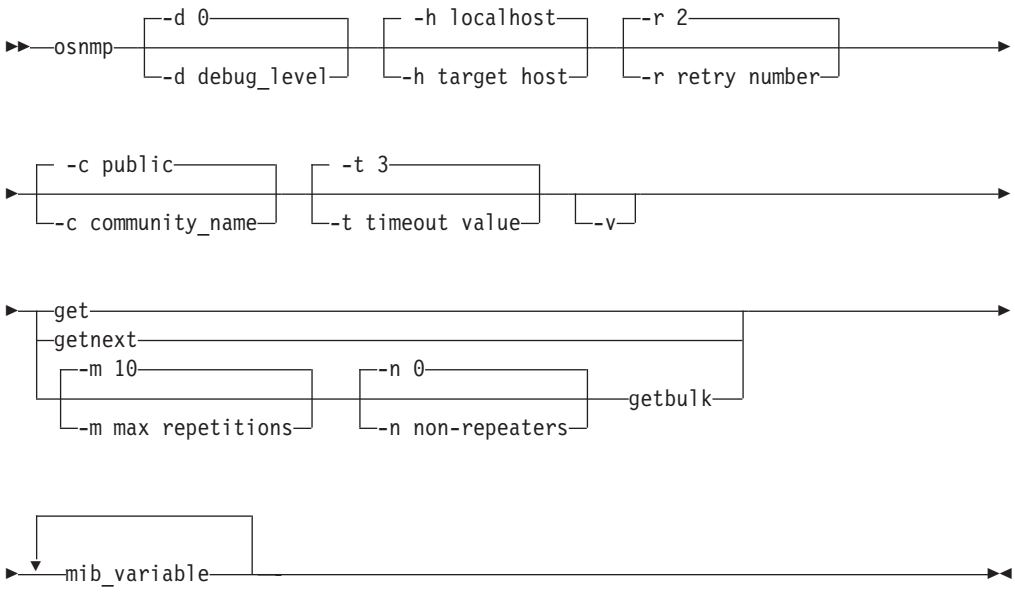
The OS/390 UNIX osnmp command provides the SNMP manager function from the OS/390 shell to query SNMP agents for network management information.

Use the osnmp command to issue SNMP requests to agents and to process SNMP responses returned by agents. The OS/390 UNIX osnmp commands supports issuance of SNMPv1, SNMPv2c, and SNMPv3 requests.

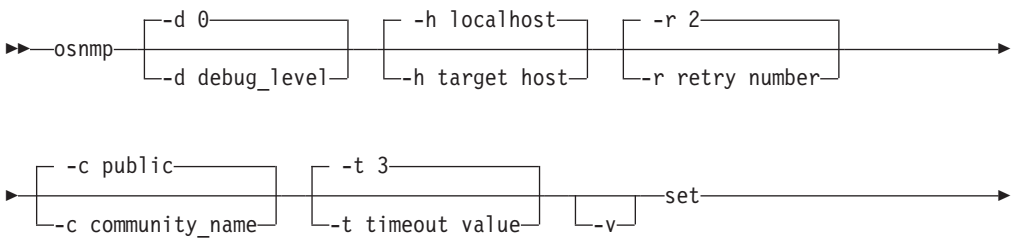
Note: snmp is a synonym for the osnmp command in the OS/390 UNIX shell. snmp command syntax is the same as that for the osnmp command.

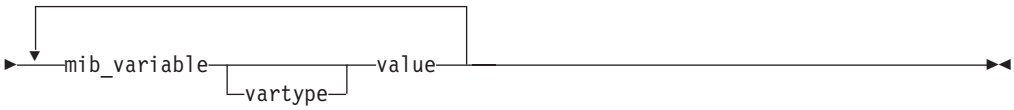
Format

Getting MIB Variables

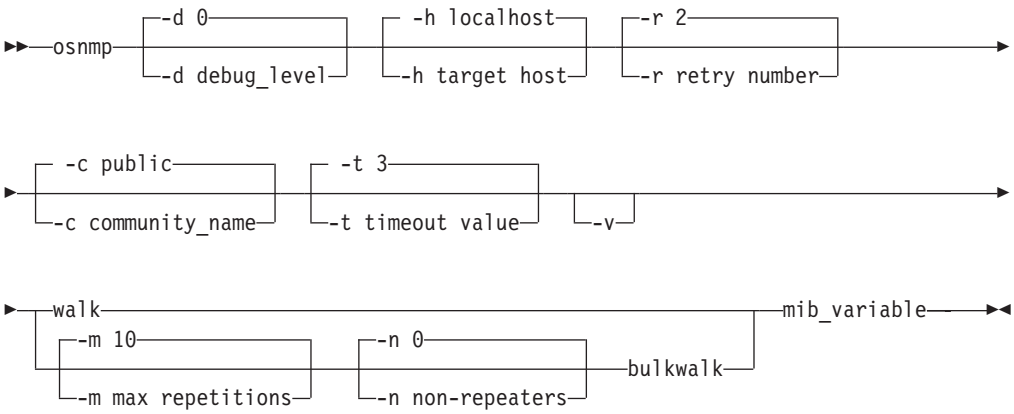


Setting the MIB Variables



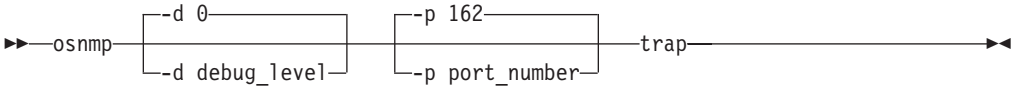


Walking the MIB Tree

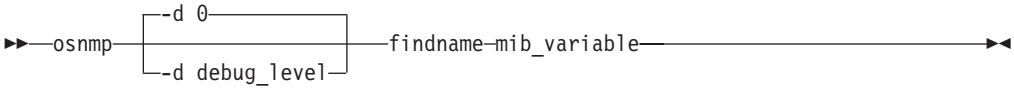


Displaying osnmp Help

Receiving a Trap



Finding a MIB Variable Name



Parameters

-c *community_name*
Specifies the community name used to access the specified variables at the destination SNMP agent. If you do not specify a community name, the default name is *public*. Community names are not required when using the user-based security model.

Note: Community names are case sensitive.

-d *debug_level*
Specifies the debug level. The default level is 0, which means no debug. The higher the debug level, the greater the amount of messages that are displayed. The debug levels are 0–4.

-h *target host*
Specifies the target host to which you want to send a request. This can be an internet protocol address, a host name, or a winSNMP name in the OSNMP.CONF configuration file. If you do not specify a host, the default is your local host.

-r *retry number*
Specifies the maximum number of times to retry the command if it timed out. The default is 2.

-t *timeout value*
Specifies the amount of time (in seconds) that the osnmp command waits for a reply from the SNMP agent. The default is 3.

-m *max repetitions*
Only applies to getbulk and bulkwalk requests. This is ignored if the function request is not a getbulk or bulkwalk. Maximum repetitions is the number of lexicographic successors to be returned for each variable binding pair after the first ”-n number” successors. For example, starting with successor ”-n number”+1, return ”-m number” of successors for each variable binding pair. The default is 10.

-n *non-repeaters*
Only applies to getbulk or bulkwalk requests. This is ignored if the function request is not a getbulk or bulkwalk. non-repeaters is the number of variable binding pairs (name/value), starting with the first, for which only a single successor is returned. The default is 0.

-? Displays help information.

mib_variable
Specifies the Management Information Base (MIB) object, using its object descriptor (textual name), object identifier in ASN.1 notation, or a combination of the two. When used with walk and bulkwalk, this is the MIB object prefix. A prefix can be any leading portion of the complete object identifier. When used with findname, this is the object identifier in ASN.1 notation.

-p *port_number*
Specifies the number of the port that listens for traps. If a port number is not specified, the osnmp trap function listens on the well-known port 162, the default port for osnmp traps.

-v Specifies that the output from a request should be displayed using verbose output, for example textual name instead of the MIB object identifier.

vartype
Specifies the type of value being set. To complete an SNMP SET request, the SMI_type must be known. If no type is specified, osnmp searches first the /etc/mibs.data file and then the compiled MIB to determine the type. If the variable is not found, an error is returned. If a *vartype* is specified, the *vartype* takes precedence over any type that may be assigned in the MIB. The *vartype* and value must be compatible. For example, if you specify a type of “number” and a *value* of “foo”, an error is returned because ”foo” is not a number. *vartype* is not case sensitive. Valid variable types are:

- bitstring

- counter
- counter32
- counter64
- display or displaystring
- integer
- integer32
- ipaddress
- gauge
- gauge32
- nsapaddress
- null
- objectidentifier or OID
- octetstring
- opaque
- opaqueascii
- timeticks
- uinteger

value
Specifies the value to be set by the SET function. If white space is needed in the value, you must enclose the value in double quotes ("). If you want to set a variable to a value that is also a type, you must specify the type.

SNMP request types

bulkwalk
Issues a GETBULK request for a specified prefix, then continues to issue GETBULK requests for as long as there are variables that match the specified prefix.

findname
Sends a request that a search be done to obtain the textual name, for a given *mib_variable* input, whose internal ASN.1 value best matches the input ASN.1 value. The search first checks the /etc/mibs.data file, and if a matching textual name is not found, continue with the compiled MIB. Only one *mib_variable* is allowed per osnmp findname invocation.

get
Sends a request to an SNMP agent for a specific management information base (MIB) variable. osnmp then waits for a response or times out.

getbulk
Obtains the value of the variables in the MIB tree specified by the OID or MIB variable name. A single getbulk performs the same function as a series of getnexts with fewer data exchanges between the osnmp command and the SNMP agent.

getnext
Sends a request to an SNMP agent for the next MIB variable that lexicographically follows the *mib_variable* specified. osnmp then waits for a response or times out.

set
Sends a request to an SNMP agent to set a specific MIB variable. osnmp then waits for a response or times out.

trap
Listens for SNMP traps and displays trap information when they occur. Uses the default, well-known port 162 or the port number specified on the *-p* option. The osnmp trap function continues to listen for traps until the process is killed or canceled.

walk
Issues a getnext request for a specified prefix, then continues to issue getnext requests for as long as there are variables that match the specified prefix. A prefix can be any leading portion of the complete object identifier.

Usage

- The set operation is not supported on all MIB objects. The set operation may be rejected if the agent or subagents managing the MIB object does not support SET.
- getbulk and bulkwalk are SNMPv2 functions. If the target agent only supports SNMPv1, the target agent ignores your request. As a result, your request times out.
- The function keywords are not case sensitive. The "-" options and variable names and values are case sensitive.
- In order to issue the osnmp trap command, you must be in superuser mode if the use of the low port numbers is restricted by the UDPCONFIG statement in the TCP/IP profile. Low port access is required in order to bind to well-known port 162. If you are not in superuser mode, you receive error `EZZ3301I Error return from bind() : EDC5111I Permission denied`.

  For more information about the UDPCONFIG statement, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.
- In order to listen to traps from NetView SNMP and OS/390 UNIX osnmp at the same time, use the *-p port_number* parameter on the osnmp command. Only one management application at an IP address can listen on a port at a time. Specifying *-p* on the osnmp trap command enables a port other than well-known port 162 to be used. Both ports must be configured as agent trap destinations.
- An osnmp command that is not authenticated (by using an acceptable community name or user name) will time out.
- The osnmp command uses two configuration files: /etc/mibs.data and OSNMP.CONF. Sample files are shipped in the/usr/lpp/tcpip/samples directory. For information about these configuration files, see the *OS/390 SecureWay Communications Server: IP Configuration*.
- The osnmp command supports sending SNMPv1, SNMPv2c, and SNMPv3 requests. The file osnmp uses to determine whether it should send an SNMPv1, SNMPv2c, or SNMPv3 request is the OSNMP.CONF file. If the target specified by way of the *-h* parameter matches a winSNMP name in the OSNMP.CONF file, osnmp sends the request using the parameters specified on the entry. If the *-h* parameter is not specified, then the request will be sent as an SNMPv1 request. If the *-h* parameter is specified and is not found in the OSNMP.CONF file, the following error message is issued:

  ```
  EZZ3306I Error converting <name> to Entity
  ```

Examples

- **Getting the MIB variable**

  The following requests MIB object sysName.0:

```
osnmp get sysName.0
1.3.6.1.2.1.1.5.0 = MVS SNMP
```

The following requests MIB object myName.0, where myName is defined in the /etc/mibs.data file to be the same object identified by sysName.0:

```
osnmp get myName.0
1.3.6.1.2.1.1.5.0 = MVSX SNMPv2 Agent
```

- **Getting the next MIB variable**

  The following requests the next logical MIB object:

```
osnmp getnext udp
1.3.6.1.2.1.7.1.0 = 653
```

  The following requests the next logical object, using the *-v* option to have value displayed with textual name instead of object identifier:

```
osnmp -v getnext udp
udpInDatagrams.0 = 653
```

- **Setting the MIB variable**

  The following sets MIB object sysName.0 to a value of 'MVSX SNMPv2 Agent':

```
osnmp set sysName.0  "MVSX SNMPv2 Agent"
1.3.6.1.2.1.1.5.0 = MVSX SNMPv2 Agent
```

- **Walking the MIB tree**

  The following returns by name all objects beginning with the same object identifier prefix:

```
osnmp -v walk udp
udpInDatagrams.0 = 653
udpNoPorts.0 = 22
udpInErrors.0 = 0
udpOutDatagrams.0 = 678
udpLocalAddress.0.0.0.0.7 = 0.0.0.0
udpLocalAddress.0.0.0.0.9 = 0.0.0.0
udpLocalAddress.0.0.0.0.13 = 0.0.0.0
udpLocalAddress.0.0.0.0.19 = 0.0.0.0
udpLocalAddress.0.0.0.0.37 = 0.0.0.0
udpLocalAddress.0.0.0.0.161 = 0.0.0.0
udpLocalAddress.0.0.0.0.5020 = 0.0.0.0
udpLocalPort.0.0.0.0.7 = 7
udpLocalPort.0.0.0.0.9 = 9
udpLocalPort.0.0.0.0.13 = 13
udpLocalPort.0.0.0.0.19 = 19
udpLocalPort.0.0.0.0.37 = 37
udpLocalPort.0.0.0.0.161 = 161
udpLocalPort.0.0.0.0.5020 = 5020
```

- **Walking the tree using bulkwalk**

  The following returns by name all objects beginning with the same object identifier prefix, but with fewer data packages to be exchanged between the osnmp command and the SNMP agent:

```
$ osnmp -h loopback -v -m 10 bulkwalk udp
udpInDatagrams.0 = 724
udpNoPorts.0 = 22
udpInErrors.0 = 0
udpOutDatagrams.0 = 749
udpLocalAddress.0.0.0.0.7 = 0.0.0.0
udpLocalAddress.0.0.0.0.9 = 0.0.0.0
udpLocalAddress.0.0.0.0.13 = 0.0.0.0
udpLocalAddress.0.0.0.0.19 = 0.0.0.0
udpLocalAddress.0.0.0.0.37 = 0.0.0.0
udpLocalAddress.0.0.0.0.161 = 0.0.0.0
udpLocalAddress.0.0.0.0.5021 = 0.0.0.0
udpLocalPort.0.0.0.0.7 = 7
udpLocalPort.0.0.0.0.9 = 9
udpLocalPort.0.0.0.0.13 = 13
udpLocalPort.0.0.0.0.19 = 19
udpLocalPort.0.0.0.0.37 = 37
udpLocalPort.0.0.0.0.161 = 161
udpLocalPort.0.0.0.0.5021 = 5021
```

- **Getting multiple MIB variables**

  The following requests multiple MIB objects using the getbulk request type. The getbulk request type returns the next logical object for one or more MIB objects listed on the command. In the following example, the *-n* option indicates that only one next logical object is requested for the first two variables (sysLocation and ifTable). For all the other objects in the list (tcp, udp, and icmp), the *-m* option indicates that 5 repetitions are requested.

  The getbulk request type is an SNMPv2 function. The *-h* parameter identifies a host, loopback, defined in the OSNMP.CONF file as an agent that supports SNMPv2 or SNMPv3.

```
 $ osnmp -h loopback -v -n 2 -m 5 getbulk sysLocation ifTable tcp udp icmp
sysLocation.0 = Research Triangle Park, NC
ifIndex.1 = 1
tcpRtoAlgorithm.0 = 4
udpInDatagrams.0 = 782
icmpInMsgs.0 = 22
tcpRtoMin.0 = 0
udpNoPorts.0 = 22
icmpInErrors.0 = 0
tcpRtoMax.0 = 120
udpInErrors.0 = 0
icmpInDestUnreachs.0 = 22
tcpMaxConn.0 = -1
udpOutDatagrams.0 = 807
icmpInTimeExcds.0 = 0
tcpActiveOpens.0 = 1
udpLocalAddress.0.0.0.0.7 = 0.0.0.0
icmpInParmProbs.0 = 0
```

- **Finding the name of an ASN.1 variable**

  The following sends a request that a search be done to obtain the textual name, for a given *mib_variable* input, whose internal ASN.1 value best matches the input ASN.1 value. The search begins with the /etc/mibs.data file and, if not found, continues with the compiled MIB. Only one *mib_variable* is allowed per osnmp findname invocation.

```
osnmp findname 1.3.6.1.2.1.6.13.1.2
1.3.6.1.2.1.6.13.1.2 found as: tcpConnLocalAddress
```

```
osnmp findname 1.3.6.1.2.1.6.13.1.2.0
1.3.6.1.2.1.6.13.1.2.0 found as: tcpConnLocalAddress.0
```

```
osnmp findname 1.3.6.1.2.
1.3.6.1.2. found as: mgmt
```

Creating User Keys

Authentication

Authentication is generally required for SNMPv3 requests to be processed (unless the security level requested is 'noAuth'). When authenticating a request, the SNMP agent verifies that the authentication key sent in an SNMPv3 request can be used to create a message digest that matches the message digest created from the authentication key defined for the user.

The osnmp command uses the authentication key found on an entry in the OSNMP.CONF configuration file. It needs to correlate with the authentication key specified on a USM_USER entry for that user in the agent SNMPD.CONF configuration file.

As an alternative to storing authentication keys in the client configuration file, the osnmp command allows user passwords to be stored. If the osnmp command is configured with a password, the code will generate an authentication key (and privacy key if requested) for the user. These keys must, of course, produce the same authentication values as the keys configured for the USM_USER in the agent's SNMPD.CONF file or configured dynamically with SNMP SET commands. Note, however, the use of passwords in the client configuration file is considered less secure than the use of keys in the configuration file.

The authentication key is generated from two pieces of information:

- The specified password
- The identification of the SNMP agent at which the key will be used. If the agent is an IBM agent and its engineID was generated using the vendor-specific engineID formula, the agent may be identified by IP address or hostname. Otherwise, the engineID must be provided as the agent identification.

A key that incorporates the identification of the agent at which it will be used is called a localized key. It can be used only at that agent. A key that does not incorporate the engineID of the agent at which it will be used is called nonlocalized.

Keys stored in the osnmp command configuration file, OSNMP.CONF, are expected to be nonlocalized keys. Keys stored in the SNMP agent's configuration file, SNMPD.CONF, can be either localized or nonlocalized, though the use of localized keys is considered more secure.

Encryption

Encryption is supported as a separately-orderable product, where export laws allow. Keys used for encryption are generated using the same algorithms as are used for authentication. However, key lengths may differ. For example, an HMAC-SHA authentication key is 20 bytes long, but a localized encryption key used with HMAC-SHA will be only 16 bytes long.

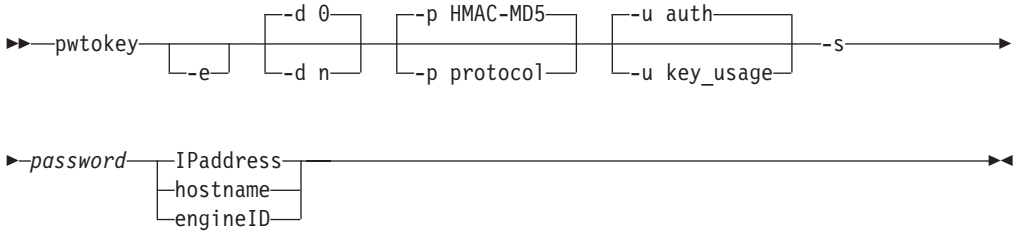
Using the pwtokey Facility

Purpose

CS for OS/390 provides a facility called pwtokey that allows conversion of passwords into localized and nonlocalized authentication and privacy keys. The pwtokey procedure takes as input a password and an identifier of the agent and generates authentication and privacy keys. Since the procedure used by the pwtokey facility is the same algorithm used by the osnmp command, the person configuring the SNMP agent can generate appropriate authentication and privacy keys to put in the SNMPD.CONF file for a user, given a particular password and the IP address at which the agent will run.

To convert passwords into authentication and privacy keys, issue the following command from OS/390 UNIX to use the pwtokey facility.

Format



Parameters

-e This flag indicates that the agent for which the key is being defined is identified by engineID rather than by IP address or host name.

-d *n*
This flag indicates what level of debug information is desired. Debug tracing is either on or off, so a value of 1 causes debug tracing to be generated to the screen of the command issuer (sysout), and a value of 0 specifies that no debug tracing be generated. Debug tracing is off (0) by default.

-p *protocol*
This flag indicates the protocols for which the keys should be generated. Valid values are:

HMAC-MD5
Generates keys for use with the HMAC-MD5 authentication protocol

HMAC-SHA
Generates keys for use with the HMAC-SHA authentication protocol

all Generates both HMAC-MD5 and HMAC-SHA keys

The default is that keys for the HMAC-MD5 protocol are generated.

-u *key_usage*
This flag indicates the usage intended for the key. Valid values are:

auth An authentication key

priv A privacy key

all Both authentication and privacy keys

Note: There is no difference between a key generated for authentication and a key generated for privacy. However, the length of privacy keys depends on whether the key is localized or not.

-s This flag indicates that output data should be displayed with additional spaces to improve readability. By default, data is displayed in a condensed format to facilitate cut-and-paste operations on the keys into configuration files or command lines.

password
Specifies the text string to be used in generating the keys. The *password* must be in the range of 8–255 characters long. In general, while any printable characters can be used in the passwords, the OS/390 UNIX shell may interpret some characters rather than passing them to the pwtokey command. Include passwords in single quotes to avoid interpretation of the characters by the OS/390 UNIX shell.

Note: This password is not related to the community name (or ”password”) used with community-based security (SNMPv1 and SNMPv2c). This password is used only to generate keys for user-based security, an entirely different security scheme.

ipaddress
Specifies the IP address in dotted decimal notation of the SNMP agent at which the key will be used on an SNMP request.

hostname
Specifies the SNMP agent at which the key will be used on an SNMP request.

engineID
Specifies the engineID of the SNMP agent at which the key will be used. The engineID is determined at SNMP agent initialization from the SNMPD.BOOTS file. The engineID must be a string of 1–32 octets (2–64 hex digits). The default is that the agent identification is not an engineID.

Examples

Sample output from the pwtokey command:

```
# pwtokey testpassword 9.67.113.79
Display of 16 byte HMAC-MD5 authKey:
 775b109f79a6b71f94cca5d22451cc0e

Display of 16 byte HMAC-MD5 localized authKey:
 de25243d5c2765f0ce273e4bcf941701
```

pwtokey generates two keys – one that is localized (has been tailored to be usable only at the agent identified) and one that has not been localized. Typically, the localized key is used in the configuration for the SNMP agent. The nonlocalized key is used in the configuration for the osnmp command.

If pwtokey is invoked requesting HMAC-SHA keys for both authentication and privacy, the output looks like this:

```
# pwtokey -p HMAC-SHA -u all testpassword 9.67.113.79
Display of 20 byte HMAC-SHA authKey:
 b267809aee4b8ef450a7872d6e348713f04b9c50

Display of 20 byte HMAC-SHA localized authKey:
 e5438092d1098a43e27e507e50d32c0edaa39b7c

Display of 20 byte HMAC-SHA privKey:
 b267809aee4b8ef450a7872d6e348713f04b9c50

Display of 16 byte HMAC-SHA localized privKey:
 e5438092d1098a43e27e507e50d32c0e
```

The output for the privacy keys is the same as the output for the authentication keys, except that the localized privacy key has been truncated to 16 bytes, as is required for DES.

Note: If encryption is used, it is more secure to use different passwords for authentication and privacy.

Usage

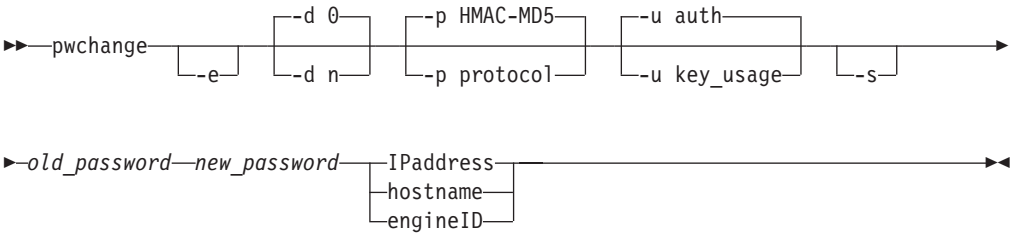
If the IP address or the hostname is specified, the SNMP agent must be an IBM agent. The engineID will be created using a vendor-specific formula that incorporates the IP address of the agent and an enterprise ID representing IBM.

Using the pwchange Facility

Purpose

The pwchange command is provided to facilitate dynamic changes of user authentication and privacy keys. Dynamic configuration of authentication and privacy keys is done by doing SET commands to objects of syntax keyChange. The keyChange syntax provides a way of changing keys without requiring that the actual keys (either new or old) be flowed directly across the wire, which would not be secure. Instead, if an object, such as usmUserAuthKeyChange (for example) is to be set, the keyChange value must be derived from the old and new passwords and the engineID of the agent at which the key will be used. The pwchange command is used to generate the keyChange values.

Format



Parameters

-e This flag indicates that the agent for which the keychange value is being defined is identified by engineID rather than by IP address or host name.

-d *n*
This flag indicates what level of debug information is desired. Debug tracing is either on or off: 1 causes debug tracing to be generated to the screen of the command issuer (sysout). Debug tracing is off (0) by default.

-p *protocol*
This flag indicates the protocols for which the keychange values should be generated. Valid values are:

HMAC-MD5
Generates keychange values for use with the HMAC-MD5 authentication protocol

HMAC-SHA
Generates keychange values for use with the HMAC-SHA authentication protocol

all Generates both HMAC-MD5 and HMAC-SHA keychange values

The default is that keychange values for the HMAC-MD5 protocol are generated.

-u *key_usage*
This flag indicates the usage intended for the keychange value. Valid values are:

auth An authentication keychange value

priv A privacy keychange value

all Both authentication and privacy keychange values

Note: There is no difference between a keychange value generated for authentication and a keychange value generated for privacy. However, the length of privacy keychange values depends on whether the keychange value is localized.

-s This flag indicates that output should be displayed with additional spaces to improve readability. By default, data is displayed in a condensed format to facilitate cut-and-paste operations on the keychange values onto command lines in shell scripts.

old_password
Specifies the password that was used in generating the key originally. The *password* must be between eight and 255 characters long.

new_password
Specifies the password that will be used in generating the new key. The *password* must be between eight and 255 characters long.

ipaddress
Specifies the IP address in dotted decimal notation of the agent at the destination host at which the key is to be used.

hostname
Specifies the destination host at which the key is to be used.

engineID
Specifies the engine ID (1–32 octets, 2–64 hex digits) of the destination host at which the key is to be used. The engineID must be a string of 1–32 octets (2–64 hex digits). The default is that the agent identification is not an engineID.

Usage

The pwchange command generates different output, depending on which protocol and what key usage is selected. Keychange values are typically twice as long as the key to be changed.

Examples

Sample pwchange output:

```
# pwchange oldpassword newpassword 9.67.113.79
Dump of 32 byte HMAC-MD5 authKey keyChange value:
  3eca6ff34b59010d262845210a401656
  78dd9646e31e9f890480a233dbe1114d
```

The value to be set should be passed as a hex value:

```
osnmp set usmUserAuthKeyChange.12.0.0.0.2.0.0.0.0.9.67.113.79.2.117.49
\'3eca6ff34b59010d262845210a40165678dd9646e31e9f890480a233dbe1114d\'h
```

Note: The backslash in the preceding example is required before the single quotation mark to enable OS/390 UNIX to correctly interpret the hexadecimal value.

(The index of the usmUserTable is made up of the engineID and the ASCII representation of the user name; in this case it is 2 characters long and translates to 117.49.)

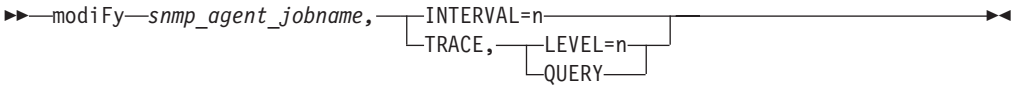
Note: pwchange incorporates a random component in generating keys and keyChange values. The output from multiple commands with the same input does not produce duplicate results.

Modifying SNMP Agent Parameters

Purpose

Some SNMP agent initialization parameters can be modified while the agent is executing using the MVS MODIFY command. The MODIFY command can also be used to display the current level of SNMP agent tracing.

Format



Parameters

INTERVAL
Specifies an integer in the range of 0–10 which indicates the maximum number of minutes before committed configuration changes to the SNMPD.CONF file will get written out. A value of 0 means that the changes will be written out at the time the SET committed.

TRACE
Indicates SNMP agent tracing is to be queried or changed.

LEVEL
Specifies an integer in the range of 0–255 which indicates the level of agent tracing. This corresponds to the -d parameter at agent initialization. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for additional guidance on setting the trace level.

QUERY
Requests that the current level of SNMP agent tracing be displayed.

Management Data Supported

The following sections describe the type of management data supported by the CS for OS/390 SNMP agent and subagents and how this data can be used to support network management. The SNMP agent supports objects related to the agent's configuration and the subagents connected to it. The subagents shipped with CS for OS/390 are:

- The TCP/IP subagent
- The OMPRoute subagent
- The SLA subagent

The agent and subagents support many MIB objects defined as standard objects in RFCs. Additionally, the SNMP agent and the TCP/IP subagent support nonstandard MIB objects, called enterprise-specific objects. The complete list of MIB objects supported by the SNMP agent and subagents is in "Appendix D. Management Information Base (MIB) Objects" on page 571.

SNMP MIB Support

The CS for OS/390 SNMP agent and subagents support for nonstandard MIB variables is defined in several files shipped with the product. These files are installed into the HFS in the /usr/lpp/tcpip/samples directory:

- mvstcpip.caps

  This file is the CS for OS/390 SNMP Capability Statement. It contains the formal SNMPv2 definition of the MIBs supported by the SNMP agent and subagents shipped with CS for OS/390.
- mvstcpip.mi2

  Contains the formal SNMPv2 syntax (SMI) of the IBM MVS Enterprise Specific MIB extension. This is supported by the TCP/IP subagent.
- mvstcpip.mib

  Contains the formal SNMPv1 syntax (SMI) of the IBM MVS Enterprise Specific MIB extension. This is supported by the TCP/IP subagent.
- samib.mib

  Contains the formal SNMPv1 syntax (SMI) for the subagent MIB (saMIB) objects. This is supported by the SNMP agent.
- samib.mi2

  Contains the formal SNMPv2 syntax (SMI) for the subagent MIB (saMIB) objects. This is supported by the SNMP agent.
- slapm.mib

  Contains the formal SNMPv1 syntax (SMI) for SLAPM-MIB objects. This is supported by the SLA subagent.
- slapm.mi2

  Contains the formal SNMPv2 syntax (SMI) for SLAPM-MIB objects. This is supported by the SLA subagent.
- slapm.txt

  Contains the draft of the SLAPM-MIB RFC that describes the MIB.
- rfc1592b.mib

  Contains the SNMPv1 syntax (SMI) for the additional information that expands the implementation of RFC 1592 in CS for OS/390. This is supported by the SNMP agent.

- rfc1592b.mi2

  Contains the SNMPv2 syntax (SMI) for the additional information that expands the implementation of RFC 1592 in CS for OS/390. This is supported by the SNMP agent.
- ibm3172.mi2

  Contains the SNMPv2 syntax (SMI) for the 3172 enterprise specific MIB objects. This is supported by the SNMP agent.
- ibm3172.mib

  Contains the SNMPv1 syntax (SMI) for the 3172 entrprise specific MIB objects. This is supported by the SNMP agent.

TCP/IP Subagent

In addition to the standard MIB objects, the TCP/IP subagent provides extensive support for enterprise-specific MIB objects to facilitate:

- Retrieval of response time data between two remote hosts
- Retrieval of TCP/IP configuration, interface, and TCP/UDP connection data
- Support for the IBM 3172 Interconnect Controller
- Support for ATM management

SNMP Remote PING

SNMP remote PING is a function of the TCP/IP subagent that gives an SNMP manager the ability to obtain the round-trip response time for an ICMP echo request message (PING) from an SNMP agent to a destination IP address.

The SNMP remote PING function is a valuable tool in an enterprise network that provides centralized management services because it gives a third-party (SNMP manager) system the ability to request that a PING operation be performed on a remote system running OS/390. The remote system must be running the SNMP agent and the TCP/IP subagent.

For example, if there are three hosts (A, B, and C) as shown in Figure 19 on page 534, you can obtain the response time between the two remote hosts. In this example, your host is running the SNMP manager function (Host A), Host B is running the SNMP agent and TCP/IP subagent functions, and Host C is some arbitrary remote host. The standard PING function enables Host A to obtain the round-trip response time from A to B and from A to C, but not from B to C. With the SNMP remote PING function on the TCP/IP subagent, Host A can obtain the round-trip response time from B to C.

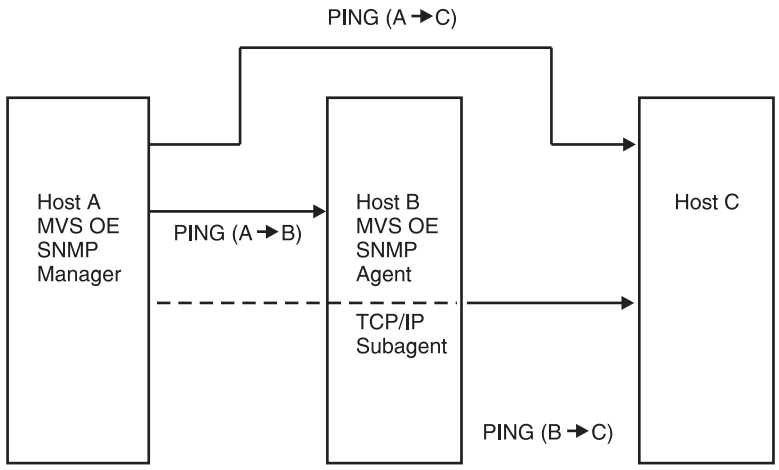
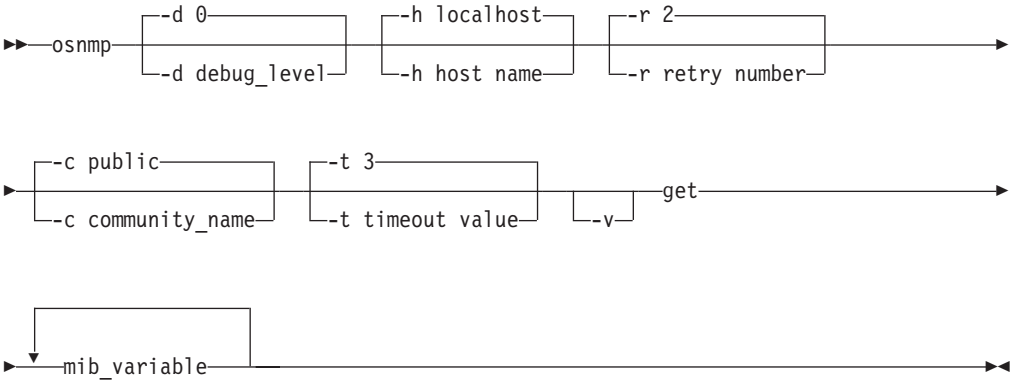


Figure 19. SNMP Remote PING Function

With the SNMP remote PING function, you can specify the size of the packet, in bytes, that is sent in the ICMP echo request message and the time period, in seconds, to wait for that ICMP echo request message to return from the requested destination address.

Format

To send a remote PING command, use the SNMP GET command with the *ibmMvsRPingResponseTime mib_variable*, or use the osnmp GET command in the following format. You are requesting the variable from the remote system.



Parameters

mib_variable

Specifies one or more MIB variable names to be retrieved. You can specify the names in textual form or ASN.1 notation.

For the remote ping object, a three-part index is required, with each part separated by dots (.), as in the following example:

```
osnmp -h host_name get ibmMvsRPingResponseTime.packet_size.time_out.ip_address
```

Note: To find a description of the other paramters, see “Parameters” on page 517

The following list describes the instances of *var_name*:

Instance	Description
ibmMvsRPingResponseTime	Specifies the remote PING command
packet_size	Specifies the packet size of the PING request
time_out	Specifies the timeout value, in seconds, for the PING request
ip_address	Specifies the IP address of the remote host that receives the PING request

Example

The following is an example of using GET to perform a remote PING:

```
osnmp -h mvs1 -c mvs150 get ibmMvsRPingResponseTime.2048.5.9.37.33.175
```

where:

```
host_name = mvs1

community_name  = mvs150

mib_variable  = ibmMvsRPingResponseTime.2048.5.9.37.33.175  where:

          packet size     = 2048 bytes
          time-out        = 5 seconds
          ip_address      = 9.37.33.175
```

The expected response is as follows:

```
1.3.6.1.4.1.2.6.19.2.2.1.1.1.4.2048.5.9.37.33.175=33
```

The variable value in the previous example is a positive value (33) indicating a successful response. The variable number, when positive, is the round-trip response time, in milliseconds, from the SNMP agent host system to the requested destination IP address.

The variable value can be a negative integer indicating that a failure has occurred. A negative integer is a result of the SNMP agent or TCP/IP subagent detecting either an internal error, an incorrect MIB instance format (f.e.a.b.c.d), an ICMP echo request timeout, an incorrect packet size value (f), an incorrect timeout value (e) or an incorrect destination IP address (a.b.c.d). See Table 36 for a description of what the variable value can represent.

Table 36. SNMP Get Command Responses for Variable Value

Description	Returned value	Condition	Valid input
Round-Trip Response Time	>0 (milliseconds)	Success	N/A
Internal error	-1	Failure	N/A
ICMP echo request timed out	-2	Failure	N/A
Incorrect packet size (f)	-4	Failure	0, 16–4096 (bytes)
Incorrect time-out (e)	-5	Failure	0, 3–15 (seconds)
Unknown destination IP address (a.b.c.d)	-6	Failure	Dotted decimal IP address
Incorrect MIB instance format (f.e.a.b.c.d)	-7	Failure	Packet size.timeout.IP address in dotted decimal format

Note: The packet size (f) and the timeout (e) in the *mib_variable* part of the osnmp get command can have a value of 0 indicating that the default values of 16 bytes and 15 seconds, respectively, are used.

The MIB variable *ibmMVSRPingResponseTime* is defined to access the SNMP Remote PING function and is included in the IBM MVS Enterprise Specific MIB. This variable exists under the SNMP subtree `ibmRemotePingEntry`.

The ASN.1 for the SNMP subtree `ibmRemotePingEntry` is `1.3.6.1.4.1.2.6.19.2.2.1.1.1`, and the ASN.1 for the SNMP Remote PING variable `ibmMvsRPingResponseTime` is `1.3.6.1.4.1.2.6.19.2.2.1.1.1.4`.

Interface Layering

In the SNMP framework, the most fundamental MIB Table is the Interfaces table. RFC 1213 provides the latest SNMPv1 format of this table. The TCP/IP subagent has also implemented RFC 2233. For more information, see “Appendix D. Management Information Base (MIB) Objects” on page 571 for a list of supported ifMIB objects. RFC 2233 expands what is defined by RFC 1213 by:

- Extending the Interface table, ifTable and its ifEntry definition, through the ifXTable and its ifXEntry definition
- Providing a new table, ifStackTable, that shows how interfaces are layered

The TCP/IP subagent’s interface layering implementation is explained by the following example, where the following DEVICE and LINK Profile statements apply:

```
DEVICE OSA1 ATM PORTNAME ATMPORT1
LINK ATMLINK1 ATM OSA1
DEVICE LCS1 LCS 100
LINK TR1 IBMTR 0 LCS1
LINK ETH1 ETHERNET 0 LCS1
```

The following interface entries would be created:

ifIndex	ifType	Description
1	53 (propVirtual)	LOOPBACK device
2	24 (softwareLOOPBACK)	LOOPBACK link
3	53 (propVirtual)	ATM Device OSA1
4	49 (aal5)	ATMPORT1’s aal5 layer
5	37 (atm)	ATMPORT1’s atm layer
6	59/60 (atmlane8023/8025)	ATM LAN Emulation
7	59/60 (atmlane8023/8025)	ATM LAN Emulation
8	1 (other)	ATM Link ATMLINK1
9	53 (propVirtual)	LCS Device LCS1
10	9 (iso88025TokenRing)	LCS Link TR1
11	6 (ethernetCsmacd)	LCS Link ETH1

ifTypes are assigned by the Internet Assigned Numbers Authority (IANA) to indicate the type of interface. In CS for OS/390, a DEVICE has a corresponding interface entry with its LINKs defined as interface entries stacked below it. The ifStackTable is used to reflect the relationship between interfaces. Reference the ifMib for a detailed explanation of how the ifStackTable is used to reflect interface

relationships. Essentially a DEVICE is stacked above its LINKs. Its ifEntry and ifXEntry counters reflect the sum of its LINKs.

In the previous example, a LOOPBACK DEVICE and LINK ifEntry were created when the links were not explicitly defined. TCP/IP automatically generated these entries. The ifEntry and ifXEntry counters can be retrieved from either the LOOPBACK LINK or DEVICE interface entry to determine LOOPBACK activity. The counters shown for a DEVICE interface entry equals that of a subordinate LINK when there is only one LINK defined for the DEVICE.

When an ATM DEVICE is defined, two subordinate interface entries are created below it, AAL5 and ATM. AAL5 and ATM are UNI defined layers that exist physically in an ATM Port. The ifEntry and ifXEntry counters reflect traffic though the port.

If the ATM DEVICE is configured for LAN Emulation mode, two additional subordinate layers may be created after the AAL5 and ATM layers. These additional layers represent emulated link interfaces. The counter data for all these ATM subordinate layers is obtained directly from OSA/SF. See “ATM Considerations” on page 538 for more information.

IBM 3172 Enterprise-Specific MIB Variables

The IBM 3172 Interconnect Controller maintains a set of enterprise-specific MIB variables. The SNMP agent can act as a proxy agent to retrieve these variables from the 3172. You can issue either a GET or GETNext command to retrieve the 3172 variables. The 3172 variable names can be included in a GET/GETNext command that also contains standard MIB variable names. See “Appendix D. Management Information Base (MIB) Objects” on page 571, for a description of the 3172 enterprise-specific MIB variables.

The 3172 variables are referenced by a single element instance identifier, for example, `(.1, .2, .3)`.This identifier is the interface index, ifIndex, assigned to the LAN Channel System (LCS) device and links by TCP/IP. TCP/IP assigns ifIndex values to its devices and links based on where they are physically located in the PROFILE.TCPIP. The following example shows the Profile statements and the ifIndex values that would be assigned:

```
                                                ifIndex
                                                -------
DEVICE LCS1     LCS          120     NETMAN        3
DEVICE CTCD00   CTC D00                            4
LINK CTC1       CTC 1 CTCD00 IFSPEED   12345       5
LINK TR1  IBMTR      1 LCS1                        6
```

For objects which pertain to the entire 3172, the instance identifier is the ifIndex of the LCS device. In the example above, this is an ifIndex of 3.

For counter objects related to a specific link interface, the instance identifier is the ifIndex of that link. In the example above, this is an ifIndex of 6.

If a GET command is issued for a counter object using an instance identifier of a link that does not support the 3172 objects, a response of NO SUCH INSTANCE is returned from the SNMP agent.

If a GETNext command is issued the links that do not support the 3172 objects are skipped and the NEXT link that does support the 3172 objects is returned.

If an error occurs accessing a 3172 variable from the 3172 (either an error return code is received from the 3172 or no response is received from the 3172), an error code of GEN ERROR is returned to the client in the SNMP response PDU for that variable. An error message containing more specific information about the error that occurred is written to the syslog daemon if SNMP subagent tracing has been activated by the ITRACE Profile statement. Several of the potential error conditions reference the 3172 MIB variable by the 3172 attribute index. See “Appendix E. IBM 3172 Attribute Index” on page 589, for a list of the 3172 attribute indices and the corresponding MIB variable names.

ATM Considerations

The interface table values for the ATM device and link interfaces are based on the TCP/IP stack use of the ATM for data transport. The interface table values for the ATM AAL5 and ATM layers are retrieved from OSA/SF. The values in the following IBM MVS enterprise-specific MIB tables are retrieved from OSA/SF. The values in the following ipoaMIB (RFC2320) tables are based on the TCP/IP stack's use of the ATM device.

ATM MIB Objects

CS for OS/390 SNMP provides several MIB tables that form the basis of ATM Management support:

- osasfChannelTable

  An entry in this table is created for every ATM DEVICE interface. Each ATM DEVICE statement represents one ATM OSA-2 adapter card externally through SNMP. This table is indexed by the ifIndex of the ATM DEVICE. The osasfChannelTable is from the IBM MVS Enterprise Specific MIB.

  Note: Each 390 host has a maximum of 12 OSA ATM adapter cards. Initially an ATM OSA-2 card can support only one ATM port.

- osasfPvcTable

  This table is created at the same time that the osasfChannelTable is created. Indexing is by the ifIndex of the AAL5 layer and pvcName. One entry is created for every PVC defined for a given ATM Port. Each port channel has a limit of 256 PVCs. The osasPvcTable is from the IBM MVS Enterprise Specific MIB.

- osasfPortTable

  An entry in this table is created for every ATM DEVICE interface. Indexing is by the ifIndex of the AAL5 interface layer. The osasfPortTable is from the IBM MVS Enterprise Specific MIB.

- atmInterfaceConfTable from RFC 1695 MIB

  One entry in this table is created for every ATM Link interface. It is, however, indexed by the ifIndex of the AAL5 interface entry that corresponds to the ATM DEVICE.

- ibmMvsAtmSnaLeTable

  One entry in this table is created for every ATM LAN Emulation interface where the ATM OSA-2 port is configured for SNA and LAN Emulation mode. Indexing is by the ifIndex of the ATM LAN Emulation interface.

- ibmMvsAtmLecConfigTable

  One entry in this table is created for every ATM LAN Emulation interface, where the ATM OSA-2 port is configured for LAN Emulation mode. This table is modeled after the LEC Config Table from the LAN Emulation MIB defined by the ATM Forum. The ifIndex of the ATM LAN Emulation interface does the indexing.

- ibmMvsAtmLecStatusTable

  One entry in this table is created for every ATM LAN Emulation interface, where the ATM OSA-2 port is configured for LAN Emulation mode. This table is modeled after the LEC Status Table from the LAN Emulation MIB defined by the ATM Forum. The ifIndex of the ATM LAN Emulation interface does the indexing.
- ibmMvsAtmLecStatisticsTable

  One entry in this table is created for every ATM LAN Emulation interface, where the ATM OSA-2 port is configured for LAN Emulation mode. This table is modeled after the LEC Statistics Table from the LAN Emulation MIB defined by the ATM Forum. The ifIndex of the ATM LAN Emulation interface does the indexing.
- ibmMvsAtmLecServerTable

  One entry in this table is created for every ATM LAN Emulation interface, where the ATM OSA-2 port is configured for LAN Emulation mode. This table is modeled after the LEC Server Table from the LAN Emulation MIB defined by the ATM Forum. The ifIndex of the ATM LAN Emulation interface does the indexing.
- ibmMvsAtmLecMacAddressTable

  One entry in this table is created for every ATM LAN Emulation interface, where the ATM OSA-2 port is configured for LAN Emulation mode. This table is modeled after the LEC Mac Address Table from the LAN Emulation MIB defined by the ATM Forum. The ifIndex of the ATM LAN Emulation interface does the indexing.
- Interface Table Data

  ifTable and ifXTable data are retrieved from OSA/SF for the AAL5, ATM, and LAN Emulation interfaces subordinate to an ATM DEVICE interface. All the data for the AAL5, ATM, and LAN Emulation layers is retrieved at the same time.

  ifTable and ifXTable data for ATM DEVICE and LINK interfaces used by TCP/IP for data transport is retrieved from TCP/IP.

  Interface Table Data is from the ifMIB (RFC2233).
- ipoaLisTable

  An entry in this table is created for every ATMLIS statement whose LIS name is referenced on an ATM LINK statement. The ipoaLisTable is from the ipoaMIB (RFC2320).
- ipoaLisIfMappingTable

  An entry in this table is created for every ATM LINK statement, which includes a LIS name. The ipoaLisIfMappingTable is from the ipoaMIB (RFC2320).
- ipoaArpClientTable

  An entry in this table is created for every local IP address that is assigned to an ATM interface (that is, for every LINK ATM statement on a DEVICE ATM). The ipoaArpClientTable is from the ipoaMIB (RFC2320).
- ipoaArpRemoteServerTable

  An entry in this table is created for every TCP/IP link to an ATMARP Remote server. The ipoaArpRemoteServerTable is from the ipoaMIB (RFC2320).
- ipoaVcTable

  An entry in this table is created for each ATM VC connection. The ipoaVcTable is from the ipoaMIB (RFC2320).
- ipoaConfigPvcTable

  An entry in this table is created for each ATM VC connection, which is a permanent VC. The ipoaConfigPvcTable is from the ipoaMIB (RFC2320).

ATM Port IP Address Assignment

CS for OS/390 SNMP provides a method for assigning an IP address to the ATM Port. The ATM OSA-2 Port reports the IP address as its atmfMyIpNmAddress as specified by the ATM Forum User-Network Interface (UNI) Specification. UNI defines an Interim Local Management Interface (ILMI) layer that provides a MIB that can be accessed directly over an ATM Network by way of an SNMP Request.

To specify an IP address for an ATM Port, use the osnmp set command against the ibmMvsAtmOsasfPortIpAddress MIB Object. Once an IP address is set, the ATM Port remembers the IP address and it does not have to be reset. Make sure you issue the osnmp set command on the MVS image where the managing OSA/SF for the ATM OSA-2 device is running. For information about the *osnmp set* command, see "The OS/390 UNIX osnmp/snmp Command" on page 516.

ATM Trap Notification from OSA/SF

Asynchronous events are forwarded from OSA/SF to SNMP subagents. These events are converted to traps and sent to the osnmp agent associated with the TCP/IP instance receiving the notification for forwarding. The traps supported for ATM Management are:

- ATM Port enabled—LinkUp Trap
- ATM Port disabled—LinkDown Trap
- Permanent Virtual Circuit (PVC) creation—ibmMvsAtmOsasfAtmPvcCreate Trap
- Permanent Virtual Circuit (PVC) deletion—ibmMvsAtmOsasfAtmPvcDelete Trap

The LinkUp and LinkDown traps for ATM Ports are generated by the Subagent for the AAL5 layer interface.

An `ibmMvsAtmOsasfAtmPvcCreate` notification is generated when OSA/SF sends an asynchronous notification to a subagent that a PVC was created for a given ATM Port. An `ibmMvsAtmOsasfAtmPvcDelete` notification is generated when a PVC is deleted.

Note: The TCP/IP subagent discards any notification received for an ATM port that is not properly defined through an ATM DEVICE statement.

OMPRoute Subagent

The OMPRoute subagent provides an alternative to DISPLAY commands for displaying Open Shortest Path First (OSPF) protocol configuration and state information. The subagent implements the Management Information Base (MIB) variables defined in Request for Comments (RFC) 1850.

SLA Subagent

The SLA subagent provides support for the Service Level Agreement Performance Monitor MIB (SLAPM-MIB) objects. This MIB provides information on defined service policies (both service categories and service policy rules), and performance statistics for TCP and UDP connections that map to active policies. It can monitor individual or multiple TCP connections that map to a policy for minimum rate, maximum rate, and maximum delay. The monitor table entries can be configured to send SNMP traps when traffic flows deviate from the monitored values and when

they return to normal. Refer to the *OS/390 SecureWay Communications Server: IP Configuration* for more information about monitor tables.

Part 3. Appendixes

Appendix A. Specifying Data Sets and Files

This appendix describes the file-naming formats for the MVS, AIX, UNIX, AS/400, OS/2, and VM operating systems. Examples of each format are provided to show how the files appear to a TCP/IP user who is logged on to the different operating systems.

MVS Data Sets

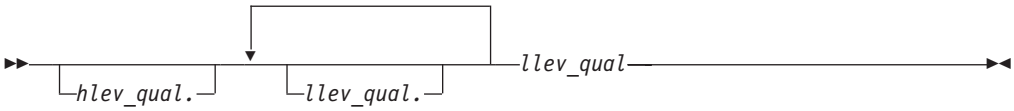
FTP subcommands can require a data set or file name. The format used to name a data set depends on the host system. Some systems limit the length of a data set name, and some systems are case sensitive.

Data set names in MVS consist of one or more names, called qualifiers, each from one to eight characters long, that are delimited from one another by periods.

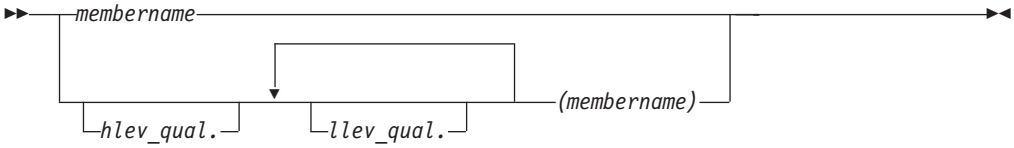
The leftmost qualifier in the data set name is the high-level qualifier. The rightmost qualifier in the data set name is the low-level qualifier. Partitioned data sets may be further qualified with a member name in the rightmost position. Qualifiers lying between them are called intermediate-level qualifiers.

For example, in the data set name `dog.bulldog.winston`, `dog` is the high-level qualifier, `bulldog` is the intermediate-level qualifier, and `winston` is the low-level qualifier.

Specify TSO sequential data sets in the following format:



Specify TSO partitioned data sets in the following format:



hlev_qual
: Specifies the high-level qualifier of the data set. The default is the current working directory. If you specify this parameter, the complete data set name must be enclosed within single quotation marks (').

llev_qual
: Specifies the low-level qualifier of the data set. You must specify this qualifier for sequential data sets.

membername
Specifies the member name of a partitioned data set (PDS). You must include parentheses around *membername* only when you also specify *llev_qual* or *hlev_qual*.

Sequential Data Sets

A sequential data set is a single file that can be allocated with any record length specified. The naming requirements for a sequential data set on an MVS host are minimal, and most of the requirements apply to any data set name under MVS.

The naming requirements for a sequential data set are:

- No part of the name can start with a numeric.
- No part of the name can be more than 8 characters in length.
- Each part of the name is separated by a period.
- A sequential data set name can have a minimum of 2 and a maximum of 44 characters.
- If single quotation marks are not used when specifying the data set name, the MVS system appends the current working directory as the first part of the name.

The following examples show the naming conventions for sequential data sets on an MVS host.

To access the sequential data set `KC00852.SEQ.NAMES`, the user, with the current working directory KC00852, enters one of the following:

- `'KC00852.SEQ.NAMES'`
- `SEQ.NAMES`

Either of these formats is acceptable for accessing a sequential data set.

Partitioned Data Sets

A partitioned data set (PDS) is a group of files contained in a library. The individual files that make up a PDS are called members. You can access an entire PDS or any individual member of a PDS.

The naming requirements for a partitioned data set are:

- No part of the name can start with a numeric.
- No part of the name can be more than eight characters in length.
- Each part of the name is separated by a period.
- If single quotation marks are not used when specifying the PDS name, the MVS system appends the current working directory as the first part of the name.

The difference between a sequential and partitioned data set specification is that the partitioned data set user accesses the directory of members in the PDS, and the sequential data set user accesses an individual file.

The following examples show the naming conventions for partitioned data sets on an MVS host.

To access the partitioned data set `KC00852.PDS.NAMES`, the user, with the current working directory ‘KC00852’, enters one of the following:

- `'KC00852.PDS.NAMES'`

- `PDS.NAMES`

Either of these formats is acceptable to access a partitioned data set.

Note: You can use the special character asterisk (*) as a global name character (wildcard) for pattern matching when you specify a data set name, with the following restrictions:

- The asterisk must be the last, or only, character specified for a level of qualifier.
- When the data set name is enclosed in quotation marks, you cannot use the asterisk as a wildcard in the high-level qualifier of the data set name. Data set names not enclosed in quotation marks will use the setting of the current directory as the high-level qualifier.
- You can use the asterisk more than once in the complete data set name, but the asterisk must be the last character for each level of data set name qualifier.
- If you specify a member name, you cannot use an asterisk anywhere in the data set name.
- If you use an asterisk as all or part of the member name, you cannot use an asterisk anywhere else in the data set name.

To access an individual member of a PDS, the member name is entered in parentheses.

To access the member `PROPER` in the PDS `KC00852.PDS.NAMES`, the user, with the current working directory KC00852, enters one of the following:

- `'KC00852.PDS.NAMES(PROPER)'`
- `PDS.NAMES(PROPER)`

Either of these formats is acceptable to access an individual member of a partitioned data set.

Transferring Data between Partitioned and Sequential Data Sets

When transferring data between partitioned and sequential data sets, ensure that the *local_file* and *foreign_file* parameters of the FTP subcommands are compatible with the type of data set you are transferring to or from. For example, if your local working directory is a partitioned data set and you want to GET the sequential file TEST.FILE1, you cannot issue the subcommand GET TEST.FILE1 to retrieve the file, because this subcommand by default tries to use the local PDS member name TEST.FILE1, which is not a valid member name.

To keep the local and remote file names compatible with the type of data set used, do one of the following:

- Specify both the local and remote file names

  For example:

  ```
  GET TEST.FILE TESTFL1
  PUT TESTPDS(FILE1) FILE1
  ```

- Change the directory to the lowest level qualifier.

  For example, to transfer between the PDS ‘USER14.TESTPDS(NAME1)’ and the sequential data set ‘USER17.SEQ.NAME1’, do the follwoing:

  1. Change the local directory (LCD) to ‘USER14.TESTPDS’
  2. Change the directory (CD) to ‘USER17.SEQ’

3. Enter one of the following:
   – `GET NAME1` to get 'USER17.SEQ.NAME1' as 'USER14.TESTPDS(NAME1)'
   – `PUT NAME1` to put 'USER14.TESTPDS(NAME1)' as 'USER17.SEQ.NAME1'

Data Transfer Methods

You must use the appropriate transmission attributes to preserve the content and structure of the data when you transfer data sets or files between two hosts. Use the FTP MODE subcommand to specify how the bits of data are to be transmitted, and the FTP TYPE subcommand to define the way that data is represented during the data transfer.

See "MODE Subcommand—Set the Data Transfer Mode" on page 174 for information about the MODE subcommand, and "TYPE Subcommand—Set the Data Transfer Type" on page 235 for information about the TYPE subcommand.

TCP/IP supports only the data transfer of a data set or file structured as a continuous sequence of data bytes. This ensures that the correct record format is preserved across MVS hosts.

Table 37 shows how to set the transmission attributes for different host systems. IBM mainframe operating systems (VM or MVS) are identified as EBCDIC transfer types. Systems with ASCII storage are identified as ASCII transfer types. A text file of an ASCII transfer type contains standard, displayable characters; a carriage return (ASCII X'0D' and EBCDIC X'15'), and line feed characters (ASCII X'0A' and EBCDIC X'25'). A text file of an EBCDIC transfer type contains standard, displayable characters only. A binary file can contain any characters.

Table 37. Recommended Methods for Data Transfer

Transfer Between Host Types	Transfer Type	Mode
EBCDIC to EBCDIC—text data	EBCDIC	Stream
EBCDIC to EBCDIC—binary data	EBCDIC	Block
EBCDIC to ASCII—text data	ASCII	Stream
ASCII to EBCDIC—text data	ASCII	Stream
ASCII to EBCDIC—binary data	Image (binary)	Stream
ASCII to EBCDIC to ASCII—all data	Image (binary)	Stream

Note: The EBCDIC host is used for storage only. Data is not used on the EBCDIC host.

Transferring PDS Directory Information

When a PDS member is transmitted in block or compressed data transfer mode with a representation type of EBCDIC, the user data associated with the PDS member is also transferred to the directory on the target host. This transfer occurs only when using an MVS client.

AIX and UNIX Files

For the Advanced Interactive Executive (AIX) and UNIX operating systems, data is stored in files. Related files are stored in a directory. CS for OS/390 files are UNIX files.

Specify AIX and UNIX files in the following format:

directory
Specifies a directory name. Directories contain the names of files, other directories, or both.

filename
Specifies a file name. It can be up to 14 characters long.

The complete name of an AIX and UNIX file contains the directory name and the file name. The following is an example:

```
/mailfiles/cooks
```

Where:
mailfiles
The directory name.
cooks The file name.

In the AIX and UNIX operating systems, you specify the first slash (/) only when you begin at the root directory. If you are specifying a file in the current directory, enter only the file name. For example, if you are in the current directory *mailfiles* and you want to access the *cooks* file, specify:

```
cooks
```

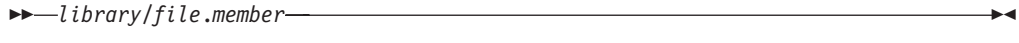
The directory name and file name can each be up to 14 characters in length. The AIX and UNIX operating systems distinguishes between uppercase and lowercase letters in file names.

A directory name and file name should not include characters such as backslash (\), ampersand (&), and period (.) that have a special meaning to the shell.

AS/400 Operating System

For the AS/400 operating system, data is stored in files.

Specify AS/400 files in the following format:



library
A library name. Libraries contain the names of programs, files, and commands.

file.member
The file name.

In the AS/400 operating system, files can have one or more members. Each file can consist of data records, source programs, or database definitions.

The FTP subcommand PUT is used to copy a local file member into a file at the remote host. The following is an example:

```
PUT PDS.DATA(MBR1)  LIB1/FILEA.MBR1
```

In this example, the PUT subcommand copies the file member `MBRA` in file `FILEA` into library `TCPA` at the local host to `MBRA` in `FILEA` in library `TCPB` at the remote host. If the member already exists at the remote host, it is overwritten.

OS/2 Files

OS/2 files are stored on a storage device such as a diskette or hard disk. A storage device is identified by a drive letter. Within the storage device, files can be assigned to a directory, which is a group of related files.

The complete name of an OS/2 file contains the storage device identifier, directory name, file name, and an extension. The following is an example of a complete name for an OS/2 file:

```
C:\WP\MAIL.LST
```

In this example, the device identifier is C: and the directory name is WP, which could be a group of word processing files. The file name is MAIL, and the file extension is .LST.

In the OS/2 operating system, the device identifier is a drive letter that is assigned by the file system, followed by a colon. The drive letter, such as A, B, or C, can represent either a physical device or a logical device. If a device identifier is not specified, the default is the device from which the system was booted or the current drive.

You assign the directory name, which consists of a character string of from 1 to 8 characters, preceded by a backslash. A directory name is optional. If a directory name is not specified, the system searches the current directory.

Normally, you also assign the file name, which consists of a character string of from 1 to 8 characters. A file name is required.

The file extension is a character string that consists of 1–3 characters, preceded by a period. A file extension is optional.

Note: The above rules reflect the restrictions on the FAT file system on OS/2. Other file systems on OS/2® are less restrictive.

If you use an incorrect file name, the system gives you an error message.

The following are examples of different OS/2 file names that are valid.

```
TEMP
START.BAT
A:COMMAND.COM
C:ABC\ABCPC.HLP
```

VM Files

Data is stored in files on VM hosts. Specify VM files in the following format:

filename
Specifies the file name.

filetype
Specifies the file type.

Note: The file mode is not accepted by foreign VM hosts; it is taken to be the file mode associated with the current working directory. The file mode is not used in TCP/IP commands.

For example, if you want to specify a file named `accounts` with a file type `cprog`, enter the following:

```
accounts.cprog
```

Where *filename* is `accounts` and *filetype* is `cprog`.

All VM file specifications are treated as if they are entered in uppercase. The file name and the file type consist of from 1 to 8 alphanumeric characters. Other valid characters are ($), number sign (#), at sign (@), plus (+), hyphen (-), and underscore (_).

You can use the special character asterisk (*) for pattern matching.

Appendix B. Mapping Values for the APL2 Character Set

This appendix lists the GDDMXD/MVS default mapping values for the APL2 character set. However, If the *hlq*.GDXAPLCS.MAP data set exists, the default mapping values are overridden.

Each entry in the *hlq*.GDXAPLCS.MAP data set (alternative character set) contains the mapping for a particular physical key that corresponds to three characters. The characters correspond to the physical key by:

- Pressing the key alone
- Pressing the key and the **Shift** key simultaneously
- Pressing the key and the **Alt** key simultaneously

The *hlq*.GDXAPLCS.MAP data set entries must contain the following 7 single-byte hexadecimal values entered as EBCDIC characters:

- Value 1 is the hexadecimal keycode for the physical key.
- Values 2, 4, and 6 identify whether the character is in the primary or alternative character set for the emulated 3179G. If the character is in the primary set, the value is 0; if the character is in the alternative set, the value is 8.
- Values 3, 5, and 7 specify the EBCDIC code of the character in the character set.

The combination of values 2 and 3 define the bytes that describe the character when the key corresponding to the keycode is pressed alone.

The combination of values 4 and 5 define the bytes that describe the character when the key corresponding to the keycode and the **Shift** key are pressed simultaneously.

The combination of values 6 and 7 define the bytes that describe the character when the key corresponding to the keycode and the **Alt** key are pressed simultaneously.

Table 38 lists the mapping values for the APL2 character set.

Table 38. Mapping Values for the APL2 Character Set

Character Name	Character Set Value	EBCDIC Value	Default Keycode
Quad Jot	8	73	**9** + **Shift**
Quad Slope	8	CE	**9** + **Alt**
1	0	F1	**A**
Diaeresis	8	72	**A** + **Shift**
Down Tack Up Tack	8	DA	**A** + **Alt**
2	0	F2	**B**
Overbar	8	A0	**B** + **Shift**
Del Tilde	8	FB	**B** + **Alt**
3	0	F3	**C**
<	0	4C	**C** + **Shift**
Del Stile	8	DC	**C** + **Alt**
4	0	F4	**D**

Table 38. Mapping Values for the APL2 Character Set (continued)

Character Name	Character Set Value	EBCDIC Value	Default Keycode
Not Greater	8	8C	D + Shift
Delta Stile	8	DD	D + Alt
5	0	F5	E
=	0	7E	E + Shift
Circle Stile	8	CD	E + Alt
6	0	F6	F
Not Less	8	AE	F + Shift
Circle Slope	8	CF	F + Alt
7	0	F7	10
>	0	6E	10 + Shift
Circle Bar	8	ED	10 + Alt
8	0	F8	11
Not Equal	8	BE	11 + Shift
Circle Star	8	FD	11 + Alt
9	0	F9	12
Down Caret	8	78	12 + Shift
Down Caret Tilde	8	CB	12 + Alt
0	0	F0	13
Up Caret	8	71	13 + Shift
Up Caret Tilde	8	CA	13 + Alt
+	0	4E	14
-	0	60	14 + Shift
!	8	DB	14 + Alt
Times	8	B6	15
Divide	8	B8	15 + Shift
Quad Divide	8	EE	15 + Alt
Q	0	D8	19
?	0	6F	19 + Shift
Q Underbar	8	58	19 + Alt
W	0	E6	1A
Omega	8	B4	1A + Shift
W Underbar	8	66	1A + Alt
E	0	C5	1B
Epsilon	8	B1	1B + Shift
E Underbar	8	45	1B + Alt
R	0	D9	1C
Rho	8	B3	1C + Shift
R Underbar	8	59	1C + Alt
T	0	E3	1D

Table 38. Mapping Values for the APL2 Character Set (continued)

Character Name	Character Set Value	EBCDIC Value	Default Keycode
Tilde	8	80	1D + Shift
T Underbar	8	63	1D + Alt
Y	0	E8	1E
Up Arrow	8	8A	1E + Shift
Y Underbar	8	68	1E + Alt
U	0	E4	1F
Down Arrow	8	8B	1F + Shift
U Underbar	8	64	1F + Alt
I	0	C9	20
Iota	8	B2	20 + Shift
I Underbar	8	49	20 + Alt
O	0	D6	21
Circle	8	9D	21 + Shift
O Underbar	8	56	21 + Alt
P	0	D7	22
Star	0	5C	22 + Shift
P Underbar	8	57	22 + Alt
Left Arrow	8	9F	23
Right Arrow	8	8F	23 + Shift
Quad Quote	8	DE	23 + Alt
Left Brk Right Brk	8	CC	24
Iota Underbar	8	74	24 + Shift
Delta Underbar	8	FC	24 + Alt
Equal Underbar	8	E1	25
Epsilon Underbar	8	E1	25 + Shift
Diaeresis Dot	8	75	25 + Alt
A	0	C1	27
Alpha	8	B0	27 + Shift
A Underbar	8	41	27 + Alt
S	0	E2	28
Up Stile	8	8D	28 + Shift
S Underbar	8	62	28 + Alt
D	0	C4	29
Down Stile	8	8E	29 + Shift
D Underbar	8	44	29 + Alt
F	0	C6	2A
Underbar	0	6D	2A + Shift
F Underbar	8	46	2A + Alt
G	0	C7	2B

Table 38. Mapping Values for the APL2 Character Set (continued)

Character Name	Character Set Value	EBCDIC Value	Default Keycode
Del	8	BA	2B + Shift
G Underbar	8	47	2B + Alt
H	0	C8	2C
Delta	8	BB	2C + Shift
H Underbar	8	48	2C + Alt
J	0	D1	2D
Jot	8	AF	2D + Shift
J Underbar	8	51	2D + Alt
K	0	D2	2E
Quote	0	7D	2E + Shift
K Underbar	8	52	2E + Alt
L	0	D3	2F
Quad	8	90	2F + Shift
L Underbar	8	53	2F + Alt
Left Bracket	8	AD	30
(	0	4D	30 + Shift
Down Tack Jot	8	FE	30 + Alt
Right Bracket	8	BD	31
)	0	5D	31 + Shift
Up Tack Jot	8	EF	31 + Alt
Z	0	E9	36
Left Shoe	8	9B	36 + Shift
Z Underbar	8	69	36 + Alt
X	0	E7	37
Right Shoe	8	9A	37 + Shift
X Underbar	8	67	37 + Alt
C	0	C3	38
Up Shoe	8	AA	38 + Shift
C Underbar	8	43	38 + Alt
V	0	E5	39
Down Shoe	8	AB	39 + Shift
V Underbar	8	65	39 + Alt
B	0	C2	3A
Down Tack	8	AC	3A + Shift
B Underbar	8	42	3A + Alt
N	0	D5	3B
Up Tack	8	BC	3B + Shift
N Underbar	8	55	3B + Alt
M	0	D4	3C

Table 38. Mapping Values for the APL2 Character Set (continued)

Character Name	Character Set Value	EBCDIC Value	Default Keycode
Stile	0	4F	3C + Shift
M Underbar	8	54	3C + Alt
,	0	6B	3D
;	0	5E	3D + Shift
Up Shoe Jot	8	DF	3D + Alt
period	0	4B	3E
:	0	7A	3E + Shift
Slope Bar	8	EB	3E + Alt
/	0	61	3F
\	0	E0	3F + Shift
Slash Bar	8	EA	3F + Alt
Space	0	40	45

Appendix C. SNMP Capability Statement

This appendix includes the SNMP agent and subagents capability statement for CS for OS/390.

The SNMP capability statement defines the MIBs supported by the SNMP Agent, osnmpd, and the MIBs supported by the subagents shipped as part of CS for OS/390.

This information is in the HFS directory /usr/lpp/tcpip/samples. The file name is mvstcpip.caps.

```
-----------------------------------------------------------------------
-- March    30, 1999
--
-- Sara Haggar  IBM vnet:      shaggar at ibmusm20
--              Internet:      shaggar@us.ibm.com
--    IBM SecureWay Communications Server for OS/390 development
--    Research Triangle Park, NC
--
-- Program name : SecureWay Communications Server for OS/390
--                Capabilities ASN.1 Description file
-- Requires: SecureWay Communications Server for OS/390
--           Version 2 Release 8
-- Description : Defines the MIBs supported by the SNMP Agent,
--               osnmpd, and the MIBs supported by the subagents
--               shipped as part of SecureWay Communications Server.
--               This file is installed in the HFS as part of the
--               product install at:
--
--                      /usr/lpp/tcpip/samples
--
--                as mvstcpip.caps.
-----------------------------------------------------------------------

 IBMTCPIPMVS-CAPS DEFINITIONS ::= BEGIN

 IMPORTS
    enterprises, MODULE-IDENTITY
          FROM SNMPv2-SMI

    AGENT-CAPABILITIES
          FROM SNMPv2-CONF;

 ibmTcpIpMvsCaps MODULE-IDENTITY
      LAST-UPDATED "9903300000Z"
      ORGANIZATION "IBM SecureWay Communications Server Development"
      CONTACT-INFO
          "        Sara Haggar

           Postal: International Business Machines Corporation
                   P.O. Box 12195
                   Dept. G86A/Bldg. 501
                   Research Triangle Park, NC 27709
                   USA

              Tel: +1 919 254 7597

           E-mail: shaggar@us.ibm.com"
      DESCRIPTION
          "The SecureWay Communications Server for OS/390 capabilities
           statements.

           Licensed Materials - Property of IBM
```

```
            Restricted Materials of IBM
            5647-A01 (C) Copyright IBM Corp. 1997, 1999
            US Government Users Restricted Rights -
            Use, duplication or disclosure restricted by
            GSA ADP Schedule Contract with IBM Corp."
        REVISION "9903300000Z"
        DESCRIPTION
               "Changes in this revision
                - Added ibmTCPIPmvsTcpGroup4
                - Added ibmTCPIPmvsAtmSupportGroup3"
        REVISION "9902150000Z"
        DESCRIPTION
               "Changes in this revision
                - Changed product name from eNetwork Communications
                  Server to SecureWay Communications Server
        REVISION "9811240000Z"
        DESCRIPTION
               "Changes in this revision
                - Added statement to document the MIB support
                  provided by the new subagent, pagtsnmp.
                - Added ibmTCPIPmvsSystemGroup5"
        REVISION "9807130000Z"
        DESCRIPTION
               "Changes in this revision
                - Added SNMPV3 support
                - Removed support for SNMPv2-USEC-MIB"
        REVISION "9806120000Z"
        DESCRIPTION
               "Added OSPF-MIB support"
        REVISION "9805120000Z"
        DESCRIPTION
               "Changes in this revision
                - Added ibmTCPIPmvsSystemGroup4
                - Added ibmTCPIPmvsInterfacesGroup3"
        REVISION "9804150000Z"
        DESCRIPTION
               "Added IPOA-MIB support"
        REVISION "9803050000Z"
        DESCRIPTION
               "Changes in this revision
                - Added copyright
                - Changed CONTACT-INFO"
    ::= { ibmAgentCapabilities 7 }

    ibm                  OBJECT IDENTIFIER ::= { enterprises 2 }
    ibmAgentCapabilities OBJECT IDENTIFIER ::= { ibm 11 }

    ibmTcpIpMvsAgtCaps AGENT-CAPABILITIES
       PRODUCT-RELEASE  "IBM SecureWay Communications Server for OS/390
                         Version 2 Release 8 SNMP Agent"
       STATUS           current
       DESCRIPTION      "SecureWay Communications Server for OS/390 Agent"

       SUPPORTS             SNMPv2-MIB       -- RFC 1907
           INCLUDES         { systemGroup, snmpGroup, snmpSetGroup,
                              snmpBasicNotificationsGroup,
                              snmpCommunityGroup }
           VARIATION        coldStart
               DESCRIPTION  "A coldStart trap is generated on all
                            reboots."

       SUPPORTS             DPI20-MIB        -- RFC 1592
           INCLUDES          { dpiGroup }
           VARIATION        dpiPathNameForUnixStream
               DESCRIPTION  "This object was added to the dpiMib
                                defined by RFC1592 in order to support
                                AF_UNIX DPI connections. It's SMI
```

```
                              definition is:

                            dpiPathNameForUnixStream OBJECT-TYPE
                            SYNTAX      DisplayString
                            MAX-ACCESS  read-only
                            STATUS      current
                            DESCRIPTION
                               'The full path name for a connection via an
                                AF_UNIX stream connection. The empty value
                                means the agent has no DPI AF_UNIX support.'
                            ::= { dpiPort 3 }

                                  replace the single quotes with double
                                  quotes in the DESCRIPTION of this object
                                  when compiling."

-- This MIB was posted to the agentx mailing list in the IETF.
-- A copy of this MIB is installed as samib.mi2 in HFS at
-- /usr/lpp/tcpip/samples as part of installing the
-- SecureWay Communications Server for OS/390.
SUPPORTS             SUBAGENT-MIB
    INCLUDES          { saTableGroup, saTreeGroup }

SUPPORTS             snmpFrameworkMIB
    INCLUDES          { snmpEngineGroup }

SUPPORTS             snmpMPDMIB
    INCLUDES          { snmpMPDGroup }

SUPPORTS             snmpTargetMIB
    INCLUDES          { snmpTargetBasicGroup,
                        snmpTargetResponseGroup,
                        snmpTargetCommandResponderGroup }
    VARIATION         snmpTargetAddrName
        SYNTAX        DisplayString (SIZE(1..32))
        DESCRIPTION   "Supported as DisplayString instead of as
                       an snmpAdminString"
    VARIATION         snmpTargetAddrTagList
        SYNTAX        SnmpTagValue
        DESCRIPTION   "Only single-value tagList is supported"
    VARIATION         snmpTargetAddrParams
        SYNTAX        DisplayString (SIZE(1..32))
        DESCRIPTION   "Supported as DisplayString instead of as
                       an snmpAdminString"
    VARIATION         snmpTargetParamsName
        SYNTAX        DisplayString (SIZE(1..32))
        DESCRIPTION   "Supported as DisplayString instead of as
                       an snmpAdminString"
    VARIATION         snmpTargetParamsSecurityName
        SYNTAX        DisplayString
        DESCRIPTION   "Supported as DisplayString instead of as
                       an snmpAdminString"

SUPPORTS             snmpNotificationMIB
    INCLUDES          { snmpNotifyGroup }
    VARIATION         snmpNotifyName
        SYNTAX        DisplayString (SIZE(1..32))
        DESCRIPTION   "Supported as DisplayString instead of as
                       an snmpAdminString"
    VARIATION         snmpNotifyType
        ACCESS        read-only
        SYNTAX        INTEGER { trap(1) }
        DESCRIPTION   "Inform type notifications are not supported.
                       Write access not supported."

SUPPORTS             snmpUsmMIB
    INCLUDES          { usmMIBBasicGroup }
```

```
        VARIATION        usmUserName
            SYNTAX       DisplayString (SIZE(1..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        usmUserSecurityName
            SYNTAX       DisplayString
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"

    SUPPORTS             snmpVacmMIB
        INCLUDES          { vacmBasicGroup }
        VARIATION        vacmContextName
            SYNTAX       DisplayString (SIZE(0..32))
            DESCRIPTION  "Only the null context is supported"
        VARIATION        vacmSecurityName
            SYNTAX       DisplayString (SIZE(1..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmGroupName
            SYNTAX       DisplayString (SIZE(1..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmAccessContextPrefix
            SYNTAX       DisplayString (SIZE(0..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmAccessReadViewName
            SYNTAX       DisplayString (SIZE(0..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmAccessWriteViewName
            SYNTAX       DisplayString (SIZE(0..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmAccessNotifyViewName
            SYNTAX       DisplayString (SIZE(0..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"
        VARIATION        vacmViewTreeFamilyName
            SYNTAX       DisplayString (SIZE(1..32))
            DESCRIPTION  "Supported as DisplayString instead of as
                          an snmpAdminString"

    ::= { ibmTcpIpMvsCaps 1 }

ibmTcpIpMvsDpiSaCaps AGENT-CAPABILITIES
    PRODUCT-RELEASE  "IBM SecureWay Communications Server for OS/390
                      Version 2 Release 8 DPI Subagent"
    STATUS           current
    DESCRIPTION      "SecureWay Communications Server for OS/390
                      DPI Subagent"

    -- Our enterprise specific MIB. Its SMI definition, mvstcpip.mi2,
    -- is shipped with SecureWay Communications Server for OS/390 and
    -- installed in the HFS at: /usr/lpp/tcpip/samples
    SUPPORTS       IBMTCPIPMVS-MIB
        INCLUDES {  ibmTCPIPmvsPingGroup,
                    ibmTCPIPmvsSystemGroup5,
                    ibmTCPIPmvsTcpGroup4,
                    ibmTCPIPmvsUdpGroup2,
                    ibmTCPIPmvsInterfacesGroup3,
                    ibmTCPIPmvsPortGroup,
                    ibmTCPIPmvsRoutingGroup,
                    ibmTCPIPmvsAtmSupportGroup3,
                    ibmTCPIPmvsAtmNotificationGroup,
                    ibmTCPIPmvsAtmLeGroup }
```

```
SUPPORTS             IF-MIB   -- RFC 2233
    INCLUDES         { ifGeneralInformationGroup,
                       ifStackGroup2,
                       ifPacketGroup,
                       ifHCFixedLengthGroup }
    VARIATION        ifAdminStatus
        SYNTAX       INTEGER { up(1), down(2) }
        DESCRIPTION  "Test mode (testing(3)) not supported. The
                      set operation is not allowed to a loopback
                      or Virtual IP Address (VIPA) device
                      interface or to any but a aal5 (atm port
                      upper layer) link associated interface."
    VARIATION        ifOperStatus
        SYNTAX       INTEGER { up(1), down(2) }
        DESCRIPTION  "Information limited to up or down. Don't
                      support testing(3), unknown(4), dormant(5),
                      notPresent(6), nor lowerLayerDown(7)."
    VARIATION        ifMtu
        DESCRIPTION  "For ATM LAN Emulation interfaces configured
                      for token ring, this value is the maximum
                      data frame size minus 54 octets for
                      encapsulation. For ATM LAN Emulation
                      interfaces not configured for token ring,
                      this value is the maximum dataframe size."
    VARIATION        ifLastChange
        DESCRIPTION  "Use time that TCP/IP was started instead of
                      sysUpTime to calculate this value, since
                      sysUpTime respesents time relative to the
                      agent's IPL not TCP/IP's."
    VARIATION        ifPromiscuousMode
        ACCESS        read-only
        DESCRIPTION  "Write access is not required, nor supported."
    VARIATION        ifStackStatus
        SYNTAX       INTEGER { active(1) } -- subset of RowStatus
        ACCESS       read-only
        DESCRIPTION  "Write access is not required, nor supported.
                      Only one enumerated values for the RowStatus
                      textual convention is supported."
    VARIATION        ifStackLastChange
        DESCRIPTION  "Not supported"
    VARIATION        ifCounterDiscontinuityTime
        DESCRIPTION  "Use time that TCP/IP was started instead of
                      sysUpTime to calculate this value, since
                      sysUpTime respesents time relative to the
                      agent's IPL not TCP/IP's."

SUPPORTS             IP-MIB   -- RFC 2011
    INCLUDES         { ipGroup, icmpGroup }
    VARIATION        ipReasmTimeout
        ACCESS        read-write
        DESCRIPTION  "This implementation of the TCP/IP
                      protocols allows this configuration
                      parameter to be changed."
    VARIATION        ipNetToMediaIfIndex
        ACCESS        read-only
        DESCRIPTION  "Write access not supported."
    VARIATION        ipNetToMediaPhysAddress
        ACCESS        read-only
        DESCRIPTION  "Write access not supported."
    VARIATION        ipNetToMediaNetAddress
        ACCESS        read-only
        DESCRIPTION  "Write access not supported."
    VARIATION        ipNetToMediaType
        ACCESS        read-only
        DESCRIPTION  "Write access not supported."
    VARIATION        ipAddrTable
        DESCRIPTION  "Not all existing instances can be supported
```

```
                       because the index is an IP address and
                       CS for OS/390 allows the same IP address
                       to be defined for multiple interfaces."

-- RFC 1354 defines this MIB using SNMPv1 SMI, to compile
-- this import the SNMPv2 version from RFC 2096 and disable
-- compiler checks ACCESS clauses on referenced items, since
-- RFC 2096 obsoletes RFC 1354.
SUPPORTS             IP-FORWARD-MIB
    INCLUDES         { ipForwardMultiPathGroup }
    VARIATION        ipForwardMask
         ACCESS   read-only
         DESCRIPTION "Write access not supported."
    VARIATION        ipForwardPolicy
         DESCRIPTION "Not used in this release. Will always return
                      a zero."
    VARIATION        ipForwardIfIndex
         ACCESS   read-only
         DESCRIPTION  "write access not supported."
    VARIATION        ipForwardType
         ACCESS   read-only
         DESCRIPTION  "write access not spported."
    VARIATION        ipForwardInfo
         ACCESS   read-only
         DESCRIPTION "write access not supported.
                      Will always return a zero"
    VARIATION        ipForwardNextHopAS
         ACCESS   read-only
         DESCRIPTION "write access not supported.
                      Will always return a zero."
    VARIATION       ipForwardMetric1
         ACCESS   read-only
         DESCRIPTION
            "An alternate routing metric  for  this  route."
    VARIATION       ipForwardMetric2
         ACCESS   read-only
         DESCRIPTION "not supported"
    VARIATION      ipForwardMetric3
         ACCESS   read-only
         DESCRIPTION "not supported"
    VARIATION     ipForwardMetric4
         ACCESS   read-only
         DESCRIPTION "not supported"
    VARIATION     ipForwardMetric5
         ACCESS    read-only
         DESCRIPTION "not supported"

SUPPORTS             TCP-MIB   -- RFC 2012
    INCLUDES         { tcpGroup }

SUPPORTS             UDP-MIB   -- RFC 2013
    INCLUDES         { udpGroup }
    VARIATION        udpTable
        DESCRIPTION  "Not all existing instances can be supported
                      because the index is the local address
                      and port.  If the socket option SO_REUSEADDR
                      is specified on a setsockopt() for a UDP
                      listener, then CS for OS/390 allows more than
                      one listener to bind to the same multicast
                      IP address and port."

SUPPORTS             ATM-MIB      -- RFC 1695
    INCLUDES         { atmInterfaceConfGroup }
    VARIATION        atmInterfaceConfVpcs
        DESCRIPTION  "For OSA purposes this object is defined
                      as the number of active VPCs (PVCs and
                      SVCs)."
```

```
    VARIATION        atmInterfaceConfVccs
        DESCRIPTION  "For OSA purposes this object is defined
                      as the number of active VCCs (PVCs and
                      SVCs)."
    VARIATION        atmInterfaceIlmiVpi
        DESCRIPTION  "The VPI value of the VCC supporting the
                      ILMI at this ATM interface. If the values
                      of atmInterfaceVpi and atmInterfaceVci
                      are both equal to zero, than the ILMI is not
                      supported at this atm interface. Only valid
                      value is currently 0."
    VARIATION        atmInterfaceIlmiVci
        DESCRIPTION  "The VPI value of the VCC supporting the
                      ILMI at this ATM interface. If the values
                      of atmInterfaceVpi and atmInterfaceVci
                      are both equal to zero, than the ILMI is not
                      supported at this atm interface. Only valid
                      value is currently 16."
    VARIATION        atmInterfaceAddressType
        DESCRIPTION  "The type of primary ATM address configured
                      for use at this ATM interface. Only valid
                      value on current OSA is 1."

SUPPORTS             IBM3172-MIB      -- IBM 3172 MIB
    INCLUDES         { ibm3172Group }

SUPPORTS             IPOA-MIB         -- IP over ATM MIB RFC 2320
    INCLUDES         { ipoaGeneralGroup}
    VARIATION     ipoaLisTrapEnable
         DESCRIPTION  "This implementation does not support
                       this object."
    VARIATION     ipoaLisDefaultMtu
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
    VARIATION     ipoaLisDefaultEncapsType
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set. Object can only
                       be llcsnap."
    VARIATION     ipoaLisInactivityTimer
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set. Smallest value
                       is 10 seconds. Default value is 300.
                       A zero continues to indicate
                       no time out in effect."
    VARIATION     ipoaLisMinHoldingTime
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
    VARIATION     ipoaLisQDepth
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
    VARIATION     ipoaLisMaxCalls
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
    VARIATION     ipoaLisCacheEntryAge
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
    VARIATION     ipoaLisRetries
         ACCESS    read-only
         DESCRIPTION  "This implementation does not allow
                       this object to be set."
```

```
        VARIATION     ipoaLisTimeout
             ACCESS    read-only
             DESCRIPTION  "This implementation does not allow
                           this object to be set. Our default is
                           3 seconds."
        VARIATION     ipoaLisDefaultPeakCellRate
             ACCESS    read-only
             DESCRIPTION  "This implementation does not allow
                           this object to be set."
        VARIATION     ipoaLisRowStatus
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaLisIfMappingStatus
             ACCESS    read-only
             DESCRIPTION  "This implementation does not support
                           remote creation."
        VARIATION     ipoaArpClientAtmAddr
             ACCESS    read-only
             DESCRIPTION  "This implementation does not support
                           setting this object."
        VARIATION     ipoaArpClientRowStatus
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaArpSrvrTable
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaArpRemoteSrvrRowStatus
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaArpRemoteSrvrAdminStatus
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaArpRemoteSrvrOperStatus
             DESCRIPTION  "This implementation does not support
                           this object."
        VARIATION     ipoaVcNegotiatedEncapsType
             DESCRIPTION  "always llcsnap."
        VARIATION     ipoaConfigPvcDefaultMtu
             ACCESS    read-only
             DESCRIPTION  "This implementation does not support
                           a set to this object."
        VARIATION     ipoaConfigPvcRowStatus
             DESCRIPTION  "This implementation does not support
                           this object."

    ::= { ibmTcpIpMvsCaps 2 }

ibmTcpIpMvsOspfCaps AGENT-CAPABILITIES
    PRODUCT-RELEASE  "IBM SecureWay Communications Server for OS/390
                      Version 2 Release 8 OSPF Subagent"
    STATUS           current
    DESCRIPTION      "SecureWay Communications Server for OS/390
                      OSPF Subagent"


    SUPPORTS            OSPF-MIB   -- RFC 1850
        INCLUDES        { ospfBasicGroup,
                          ospfAreaGroup,
                          ospfStubAreaGroup,
                          ospfLsdbGroup,
                          ospfIfGroup,
                          ospfIfMetricGroup,
                          ospfVirtIfGroup,
                          ospfNbrGroup,
                          ospfVirtNbrGroup,
                          ospfExtLsdbGroup,
                          ospfAreaAggregateGroup }
```

```
VARIATION     ospfRouterId
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfAdminStat
    SYNTAX      STATUS { 1 }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation always has at least one
                 interface enabled."
VARIATION     ospfAdminStat
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfASBdrRtrStatus
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfTOSSupport
    SYNTAX      TruthValue { false(2) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation does not support
                 type-of-service routing."
VARIATION     ospfStubTOS
    SYNTAX      TOSType { 0 }
    DESCRIPTION "This implementation only supports TOS
                 set to 0."
VARIATION     ospfExtLsdbLimit
    SYNTAX      Integer32 { -1 }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation does not have a limit
                 on maximum number of non-default
                 AS-external-LSAs entries."
VARIATION     ospfMulticastExtensions
    SYNTAX      Integer32 { 0 }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation does not support
                 multicast forwarding."
VARIATION     ospfExitOverflowInterval
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 Overflow State."
VARIATION     ospfDemandExtensions
    SYNTAX      TruthValue { true(1) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This router always suppots demand routing."
VARIATION     ospfImportAsExtern
    SYNTAX      INTEGER { importExternal(1),
                          importNoExternal(2) }
    DESCRIPTION "This implementation only supports these
                 import AS external link-state advertisement."
VARIATION     ospfImportAsExtern
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfAreaSummary
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfAreaStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION     ospfStubMetric
    ACCESS       read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfStubStatus
    ACCESS      not-implemented
```

```
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION     ospfStubMetricType
    SYNTAX      INTEGER { comparableCost(2),
                          nonComparable(3) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation only supports these
                 types of metric advertised as a default
                 route."
VARIATION     ospfLsdbType
    SYNTAX      INTEGER { routerLink(1), networklink(2),
                  summaryLink(3), asSummaryLink(4) }
    DESCRIPTION "This implementation only supports these
                 types of links."
VARIATION     ospfAddressLessIf
    SYNTAX      Integer32 { 0 }
    DESCRIPTION "This implementation only supports Interfaces
                 with IP addresses."
VARIATION      ospfIfAreaId
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfType
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfAdminStat
    SYNTAX      Status { 1 }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation only supports the value
                 formed on the interface, and the interface
                 will be advertised as an internal route
                 to some area."
VARIATION      ospfIfRtrPriority
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfTransitDelay
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfRetransInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfHelloInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfRtrDeadInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfPollInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfAuthKey
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION      ospfIfMulticastForwarding
    SYNTAX      INTEGER { blocked(1) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation does not support
                 multicast forwarding."
VARIATION      ospfIfDemand
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
```

```
VARIATION      ospfIfAuthType
    SYNTAX      INTEGER { none(0), simplePassword(1) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation only supports these
                 values."
VARIATION     ospfifMetricAddressLessIf
    SYNTAX      Integer32 { 0 }
    DESCRIPTION "This implementation only supports Interfaces
                 with IP addresses."
VARIATION      ospfIfMetricValue
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfIfMetricStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION      ospfIfMetricTOS
    SYNTAX      TOSType { 0 }
    DESCRIPTION "This implementation only supports value of 0."
VARIATION      ospfVirtIfTransitDelay
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfVirtIfRetransInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfVirtIfHelloInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfVirtIfRtrDeadInterval
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfVirtIfAuthKey
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfVirtIfStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION      ospfVirtIfAuthType
    SYNTAX      INTEGER { none(0), simplePassword(1) }
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported.
                 This implementation only supports these
                 values."
VARIATION     ospfNbrAddressLessIndex
    SYNTAX      InterfaceIndex { 0 }
    DESCRIPTION "This implementation only supports Interfaces
                 with IP addresses."
VARIATION      ospfNbrPriority
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION      ospfNbmaNbrStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION      ospfVirtNbrOptions
    SYNTAX      Integer32 { 0 }
    DESCRIPTION "This implementation only supports value of 0."
VARIATION      ospfAreaAggregateStatus
    ACCESS      not-implemented
    DESCRIPTION "This implementation does not support
                 this object."
VARIATION      ospfAreaAggregateEffect
    ACCESS      read-only
    DESCRIPTION "Write access is not required, nor supported."
VARIATION     ospfAreaAggregateLsdbType
```

```
            SYNTAX      INTEGER { summaryLink(3) }
            DESCRIPTION "This implementation only supports summary
                         link Lsdb Type."
    ::= { ibmTcpIpMvsCaps 3 }

 ibmTcpIpMvsSlapmCaps AGENT-CAPABILITIES
    PRODUCT-RELEASE  "IBM SecureWay Communications Server for OS/390
                      Version 2 Release 8 Pagtsnmp DPI Subagent"
    STATUS           current
    DESCRIPTION      "Pagtsnmp SLAPM-MIB DPI Subagent"

    -- A copy of this MIB is installed as slapm.txt in HFS at
    -- /usr/lpp/tcpip/samples as part of installing the
    -- SecureWay Communications Server for OS/390.
    SUPPORTS             SLAPM-MIB  -- <draft-white-slapm-mib-03.txt>
        INCLUDES         { slapmBaseGroup,
                           slapmNotGroup,
                           slapmEndSystemGroup,
                           slapmEndSystemNotGroup }
        VARIATION   slapmPolicyCountQueries
            DESCRIPTION  "Not supported. A value of zero is
                          always returned."
        VARIATION   slapmPolicyCountAccesses
            DESCRIPTION  "Not supported. A value of zero is
                          always returned."
        VARIATION   slapmPolicyCountSuccessAccesses
            DESCRIPTION  "Not supported. A value of zero is
                          always returned."
        VARIATION   slapmPolicyCountNotFounds
            DESCRIPTION  "Not supported. A value of zero is
                          always returned."
    ::= { ibmTcpIpMvsCaps 4 }

END
```

Appendix D. Management Information Base (MIB) Objects

This appendix lists the objects defined by the Management Information Base (MIB) which are supported by the SNMP agent and subagents on the CS for OS/390 and the maximum access allowed.

Note: If an SNMP SET (write) is attempted against a variable for which the maximum access is read-only, an error code is returned. For an SNMPv2 request, the error code is noAccess or notWritable.

The object types are defined using the following fields:

Object Descriptor
A textual name for the object type, along with its corresponding OBJECT IDENTIFIER.

Object Identifier
The name for the object type, using ASN.1 notation.

Supported by Support by the agent or subagents. If support is by one of the subagents, the subagent is named. Supported subagents include:

- TCP/IP
- OMPRoute
- SLA

Defined by The location of the description of the object.

The SNMP agent provides support of the following enterprise-specific MIBs:

- Subagent MIB
- Extensions to the DPI20 MIB defined by RFC 1592

The TCP/IP subagent provides support of the following enterprise-specific MIBs:

- IBM 3172 MIB
- IBM TCP/IP MVS Enterprise Specific MIB (which includes Remote Ping)

Copies of the SMI syntax for the above-mentioned MIBs are installed in the HFS directory /usr/lpp/tcpip/samples as:

- mvstcpip.mi2 (SNMPv2 SMI)
- mvstcpip.mib (SNMPv1 SMI)
- saMIB.mi2 (SNMPv2 SMI)
- saMIB.mib (SNMPv1 SMI)
- slapm.mi2 (SNMPv2 SMI)
- slapm.mib (SNMPv1 SMI)
- slapm.txt (draft RFC)
- rfc1592b.mi2 (SNMPv2 SMI)
- rfc1592b.mib (SNMPv1 SMI)
- ibm3172.mi2 (SNMPv2 SMI)
- ibm3172.mib (SNMPv1 SMI)

Access Allowed

- Read-only (R/O)
- Read-write (R/W)
- Read-create (R/C)
- Write-only (W/O)
- Not-accessible (N/A)

Table 39 shows the MIB objects supported by CS for OS/390 IP SNMP agent and subagents.

Table 39. MIB Objects

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
sysDescr	1.3.6.1.2.1.1.1	Agent	RFC1907	R/O
sysObjectID	1.3.6.1.2.1.1.2	Agent	RFC1907	R/O
sysUpTime	1.3.6.1.2.1.1.3	Agent	RFC1907	R/O
sysContact	1.3.6.1.2.1.1.4	Agent	RFC1907	R/W
sysName	1.3.6.1.2.1.1.5	Agent	RFC1907	R/W
sysLocation	1.3.6.1.2.1.1.6	Agent	RFC1907	R/W
sysServices	1.3.6.1.2.1.1.7	Agent	RFC1907	R/O
sysORLastChange	1.3.6.1.2.1.1.8	Agent	RFC1907	R/O
sysORTable	1.3.6.1.2.1.1.9	Agent	RFC1907	N/A
sysOREntry	1.3.6.1.2.1.1.9.1	Agent	RFC1907	N/A
sysORIndex	1.3.6.1.2.1.1.9.1.1	Agent	RFC1907	N/A
sysORID	1.3.6.1.2.1.1.9.1.2	Agent	RFC1907	R/O
sysORDescr	1.3.6.1.2.1.1.9.1.3	Agent	RFC1907	R/O
sysORUpTime	1.3.6.1.2.1.1.9.1.4	Agent	RFC1907	R/O
ifTable	1.3.6.1.2.1.2.2	TCP/IP	RFC2233	N/A
ifEntry	1.3.6.1.2.1.2.2.1	TCP/IP	RFC2233	N/A
ifIndex	1.3.6.1.2.1.2.2.1.1	TCP/IP	RFC2233	R/O
ifDescr	1.3.6.1.2.1.2.2.1.2	TCP/IP	RFC2233	R/O
ifType	1.3.6.1.2.1.2.2.1.3	TCP/IP	RFC2233	R/O
ifMtu	1.3.6.1.2.1.2.2.1.4	TCP/IP	RFC2233	R/O
ifSpeed	1.3.6.1.2.1.2.2.1.5	TCP/IP	RFC2233	R/O
ifPhysAddress	1.3.6.1.2.1.2.2.1.6	TCP/IP	RFC2233	R/O
ifAdminStatus	1.3.6.1.2.1.2.2.1.7	TCP/IP	RFC2233	R/W
ifOperStatus	1.3.6.1.2.1.2.2.1.8	TCP/IP	RFC2233	R/O
ifLastChange	1.3.6.1.2.1.2.2.1.9	TCP/IP	RFC2233	R/O
ifInOctets	1.3.6.1.2.1.2.2.1.10	TCP/IP	RFC2233	R/O
ifInUcastPkts	1.3.6.1.2.1.2.2.1.11	TCP/IP	RFC2233	R/O
ifInNUcastPkts	1.3.6.1.2.1.2.2.1.12	TCP/IP	RFC2233	R/O
ifInDiscards	1.3.6.1.2.1.2.2.1.13	TCP/IP	RFC2233	R/O
ifInErrors	1.3.6.1.2.1.2.2.1.14	TCP/IP	RFC2233	R/O
ifInUnknownProtos	1.3.6.1.2.1.2.2.1.15	TCP/IP	RFC2233	R/O
ifOutOctets	1.3.6.1.2.1.2.2.1.16	TCP/IP	RFC2233	R/O
ifOutUcastPkts	1.3.6.1.2.1.2.2.1.17	TCP/IP	RFC2233	R/O
ifOutNUcastPkts	1.3.6.1.2.1.2.2.1.18	TCP/IP	RFC2233	R/O
ifOutDiscards	1.3.6.1.2.1.2.2.1.19	TCP/IP	RFC2233	R/O
ifOutErrors	1.3.6.1.2.1.2.2.1.20	TCP/IP	RFC2233	R/O
ifOutQLen	1.3.6.1.2.1.2.2.1.21	TCP/IP	RFC2233	R/O
ifSpecific	1.3.6.1.2.1.2.2.1.22	TCP/IP	RFC2233	R/O
ipForwarding	1.3.6.1.2.1.4.1	TCP/IP	RFC2011	R/W
ipDefaultTTL	1.3.6.1.2.1.4.2	TCP/IP	RFC2011	R/W
ipInReceives	1.3.6.1.2.1.4.3	TCP/IP	RFC2011	R/O
ipInHdrErrors	1.3.6.1.2.1.4.4	TCP/IP	RFC2011	R/O
ipInAddrErrors	1.3.6.1.2.1.4.5	TCP/IP	RFC2011	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ipForwDatagrams	1.3.6.1.2.1.4.6	TCP/IP	RFC2011	R/O
ipInUnknownProtos	1.3.6.1.2.1.4.7	TCP/IP	RFC2011	R/O
ipInDiscards	1.3.6.1.2.1.4.8	TCP/IP	RFC2011	R/O
ipInDelivers	1.3.6.1.2.1.4.9	TCP/IP	RFC2011	R/O
ipOutRequests	1.3.6.1.2.1.4.10	TCP/IP	RFC2011	R/O
ipOutDiscards	1.3.6.1.2.1.4.11	TCP/IP	RFC2011	R/O
ipOutNoRoutes	1.3.6.1.2.1.4.12	TCP/IP	RFC2011	R/O
ipReasmTimeout	1.3.6.1.2.1.4.13	TCP/IP	RFC2011	R/W
ipReasmReqds	1.3.6.1.2.1.4.14	TCP/IP	RFC2011	R/O
ipReasmOKs	1.3.6.1.2.1.4.15	TCP/IP	RFC2011	R/O
ipReasmFails	1.3.6.1.2.1.4.16	TCP/IP	RFC2011	R/O
ipFragOKs	1.3.6.1.2.1.4.17	TCP/IP	RFC2011	R/O
ipFragFails	1.3.6.1.2.1.4.18	TCP/IP	RFC2011	R/O
ipFragCreates	1.3.6.1.2.1.4.19	TCP/IP	RFC2011	R/O
ipAddrTable	1.3.6.1.2.1.4.20	TCP/IP	RFC2011	N/A
ipAddrEntry	1.3.6.1.2.1.4.20.1	TCP/IP	RFC2011	N/A
ipAdEntAddr	1.3.6.1.2.1.4.20.1.1	TCP/IP	RFC2011	R/O
ipAdEntIfIndex	1.3.6.1.2.1.4.20.1.2	TCP/IP	RFC2011	R/O
ipAdEntNetMask	1.3.6.1.2.1.4.20.1.3	TCP/IP	RFC2011	R/O
ipAdEntBcastAddr	1.3.6.1.2.1.4.20.1.4	TCP/IP	RFC2011	R/O
ipAdEntReasmMaxSize	1.3.6.1.2.1.4.20.1.5	TCP/IP	RFC2011	R/O
ipNetToMediaTable	1.3.6.1.2.1.4.22	TCP/IP	RFC2011	N/A
ipNetToMediaEntry	1.3.6.1.2.1.4.22.1	TCP/IP	RFC2011	N/A
ipNetToMediaIfIndex	1.3.6.1.2.1.4.22.1.1	TCP/IP	RFC2011	R/O
ipNetToMediaPhysAddress	1.3.6.1.2.1.4.22.1.2	TCP/IP	RFC2011	R/O
ipNetToMediaNetAddress	1.3.6.1.2.1.4.22.1.3	TCP/IP	RFC2011	R/O
ipNetToMediaType	1.3.6.1.2.1.4.22.1.4	TCP/IP	RFC2011	R/O
ipRoutingDiscards	1.3.6.1.2.1.4.23	TCP/IP	RFC2011	R/O
ipForward	1.3.6.1.2.1.4.24	TCP/IP	RFC1354	N/A
ipForwardNumber	1.3.6.1.2.1.4.24.1	TCP/IP	RFC1354	R/O
ipForwardTable	1.3.6.1.2.1.4.24.2	TCP/IP	RFC1354	N/A
ipForwardEntry	1.3.6.1.2.1.4.24.2.1	TCP/IP	RFC1354	N/A
ipForwardDest	1.3.6.1.2.1.4.24.2.1.1	TCP/IP	RFC1354	R/O
ipForwardMask	1.3.6.1.2.1.4.24.2.1.2	TCP/IP	RFC1354	R/O
ipForwardPolicy	1.3.6.1.2.1.4.24.2.1.3	TCP/IP	RFC1354	R/O
ipForwardNextHop	1.3.6.1.2.1.4.24.2.1.4	TCP/IP	RFC1354	R/O
ipForwardIfIndex	1.3.6.1.2.1.4.24.2.1.5	TCP/IP	RFC1354	R/O
ipForwardType	1.3.6.1.2.1.4.24.2.1.6	TCP/IP	RFC1354	R/O
ipForwardProto	1.3.6.1.2.1.4.24.2.1.7	TCP/IP	RFC1354	R/O
ipForwardAge	1.3.6.1.2.1.4.24.2.1.8	TCP/IP	RFC1354	R/O
ipForwardInfo	1.3.6.1.2.1.4.24.2.1.9	TCP/IP	RFC1354	R/O
ipForwardNextHopAS	1.3.6.1.2.1.4.24.2.1.10	TCP/IP	RFC1354	R/O
ipForwardMetric1	1.3.6.1.2.1.4.24.2.1.11	TCP/IP	RFC1354	R/O
ipForwardMetric2	1.3.6.1.2.1.4.24.2.1.12	TCP/IP	RFC1354	R/O
ipForwardMetric3	1.3.6.1.2.1.4.24.2.1.13	TCP/IP	RFC1354	R/O
ipForwardMetric4	1.3.6.1.2.1.4.24.2.1.14	TCP/IP	RFC1354	R/O
ipForwardMetric5	1.3.6.1.2.1.4.24.2.1.15	TCP/IP	RFC1354	R/O
icmpInMsgs	1.3.6.1.2.1.5.1	TCP/IP	RFC2011	R/O
icmpInErrors	1.3.6.1.2.1.5.2	TCP/IP	RFC2011	R/O
icmpInDestUnreachs	1.3.6.1.2.1.5.3	TCP/IP	RFC2011	R/O
icmpInTimeExcds	1.3.6.1.2.1.5.4	TCP/IP	RFC2011	R/O
icmpInParmProbs	1.3.6.1.2.1.5.5	TCP/IP	RFC2011	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
icmpInSrcQuenchs	1.3.6.1.2.1.5.6	TCP/IP	RFC2011	R/O
icmpInRedirects	1.3.6.1.2.1.5.7	TCP/IP	RFC2011	R/O
icmpInEchos	1.3.6.1.2.1.5.8	TCP/IP	RFC2011	R/O
icmpInEchoReps	1.3.6.1.2.1.5.9	TCP/IP	RFC2011	R/O
icmpInTimestamps	1.3.6.1.2.1.5.10	TCP/IP	RFC2011	R/O
icmpInTimestampReps	1.3.6.1.2.1.5.11	TCP/IP	RFC2011	R/O
icmpInAddrMasks	1.3.6.1.2.1.5.12	TCP/IP	RFC2011	R/O
icmpInAddrMaskReps	1.3.6.1.2.1.5.13	TCP/IP	RFC2011	R/O
icmpOutMsgs	1.3.6.1.2.1.5.14	TCP/IP	RFC2011	R/O
icmpOutErrors	1.3.6.1.2.1.5.15	TCP/IP	RFC2011	R/O
icmpOutDestUnreachs	1.3.6.1.2.1.5.16	TCP/IP	RFC2011	R/O
icmpOutTimeExcds	1.3.6.1.2.1.5.17	TCP/IP	RFC2011	R/O
icmpOutParmProbs	1.3.6.1.2.1.5.18	TCP/IP	RFC2011	R/O
icmpOutSrcQuenchs	1.3.6.1.2.1.5.19	TCP/IP	RFC2011	R/O
icmpOutRedirects	1.3.6.1.2.1.5.20	TCP/IP	RFC2011	R/O
icmpOutEchos	1.3.6.1.2.1.5.21	TCP/IP	RFC2011	R/O
icmpOutEchoReps	1.3.6.1.2.1.5.22	TCP/IP	RFC2011	R/O
icmpOutTimestamps	1.3.6.1.2.1.5.23	TCP/IP	RFC2011	R/O
icmpOutTimestampReps	1.3.6.1.2.1.5.24	TCP/IP	RFC2011	R/O
icmpOutAddrMasks	1.3.6.1.2.1.5.25	TCP/IP	RFC2011	R/O
icmpOutAddrMaskReps	1.3.6.1.2.1.5.26	TCP/IP	RFC2011	R/O
tcpRtoAlgorithm	1.3.6.1.2.1.6.1	TCP/IP	RFC2012	R/O
tcpRtoMin	1.3.6.1.2.1.6.2	TCP/IP	RFC2012	R/O
tcpRtoMax	1.3.6.1.2.1.6.3	TCP/IP	RFC2012	R/O
tcpMaxConn	1.3.6.1.2.1.6.4	TCP/IP	RFC2012	R/O
tcpActiveOpens	1.3.6.1.2.1.6.5	TCP/IP	RFC2012	R/O
tcpPassiveOpens	1.3.6.1.2.1.6.6	TCP/IP	RFC2012	R/O
tcpAttemptFails	1.3.6.1.2.1.6.7	TCP/IP	RFC2012	R/O
tcpEstabResets	1.3.6.1.2.1.6.8	TCP/IP	RFC2012	R/O
tcpCurrEstab	1.3.6.1.2.1.6.9	TCP/IP	RFC2012	R/O
tcpInSegs	1.3.6.1.2.1.6.10	TCP/IP	RFC2012	R/O
tcpOutSegs	1.3.6.1.2.1.6.11	TCP/IP	RFC2012	R/O
tcpRetransSegs	1.3.6.1.2.1.6.12	TCP/IP	RFC2012	R/O
tcpConnTable	1.3.6.1.2.1.6.13	TCP/IP	RFC2012	N/A
tcpConnEntry	1.3.6.1.2.1.6.13.1	TCP/IP	RFC2012	N/A
tcpConnState	1.3.6.1.2.1.6.13.1.1	TCP/IP	RFC2012	R/W
tcpConnLocalAddress	1.3.6.1.2.1.6.13.1.2	TCP/IP	RFC2012	R/O
tcpConnLocalPort	1.3.6.1.2.1.6.13.1.3	TCP/IP	RFC2012	R/O
tcpConnRemAddress	1.3.6.1.2.1.6.13.1.4	TCP/IP	RFC2012	R/O
tcpConnRemPort	1.3.6.1.2.1.6.13.1.5	TCP/IP	RFC2012	R/O
tcpInErrs	1.3.6.1.2.1.6.14	TCP/IP	RFC2012	R/O
tcpOutRsts	1.3.6.1.2.1.6.15	TCP/IP	RFC2012	R/O
udpInDatagrams	1.3.6.1.2.1.7.1	TCP/IP	RFC2013	R/O
udpNoPorts	1.3.6.1.2.1.7.2	TCP/IP	RFC2013	R/O
udpInErrors	1.3.6.1.2.1.7.3	TCP/IP	RFC2013	R/O
udpOutDatagrams	1.3.6.1.2.1.7.4	TCP/IP	RFC2013	R/O
udpTable	1.3.6.1.2.1.7.5	TCP/IP	RFC2013	N/A
udpEntry	1.3.6.1.2.1.7.5.1	TCP/IP	RFC2013	N/A
udpLocalAddress	1.3.6.1.2.1.7.5.1.1	TCP/IP	RFC2013	R/O
udpLocalPort	1.3.6.1.2.1.7.5.1.2	TCP/IP	RFC2013	R/O
snmpInPkts	1.3.6.1.2.1.11.1	Agent	RFC1907	R/O
snmpInBadVersions	1.3.6.1.2.1.11.3	Agent	RFC1907	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
snmpInBadCommunityNames	1.3.6.1.2.1.11.4	Agent	RFC1907	R/O
snmpInBadCommunityUses	1.3.6.1.2.1.11.5	Agent	RFC1907	R/O
snmpInASNParseErrs	1.3.6.1.2.1.11.6	Agent	RFC1907	R/O
snmpEnableAuthenTraps	1.3.6.1.2.1.11.30	Agent	RFC1907	R/W
snmpSilentDrops	1.3.6.1.2.1.11.31	Agent	RFC1907	R/O
snmpProxyDrops	1.3.6.1.2.1.11.32	Agent	RFC1907	R/O
ospf	1.3.6.1.2.1.14	omproute	RFC1850	N/A
ospfGeneralGroup	1.3.6.1.2.1.14.1	omproute	RFC1850	N/A
ospfRouterId	1.3.6.1.2.1.14.1.1	omproute	RFC1850	R/O
ospfAdminStat	1.3.6.1.2.1.14.1.2	omproute	RFC1850	R/O
ospfVersionNumber	1.3.6.1.2.1.14.1.3	omproute	RFC1850	R/O
ospfAreaBdrRtrStatus	1.3.6.1.2.1.14.1.4	omproute	RFC1850	R/O
ospfASBdrRtrStatus	1.3.6.1.2.1.14.1.5	omproute	RFC1850	R/O
ospfExternLsaCount	1.3.6.1.2.1.14.1.6	omproute	RFC1850	R/O
ospfExternLsaCksumSum	1.3.6.1.2.1.14.1.7	omproute	RFC1850	R/O
ospfTOSSupport	1.3.6.1.2.1.14.1.8	omproute	RFC1850	R/O
ospfOriginateNewLsas	1.3.6.1.2.1.14.1.9	omproute	RFC1850	R/O
ospfRxNewLsas	1.3.6.1.2.1.14.1.10	omproute	RFC1850	R/O
ospfExtLsdbLimit	1.3.6.1.2.1.14.1.11	omproute	RFC1850	R/O
ospfMulticastExtensions	1.3.6.1.2.1.14.1.12	omproute	RFC1850	R/O
ospfDemandExtensions	1.3.6.1.2.1.14.1.14	omproute	RFC1850	R/O
ospfAreaTable	1.3.6.1.2.1.14.2	omproute	RFC1850	N/A
ospfAreaEntry	1.3.6.1.2.1.14.2.1	omproute	RFC1850	N/A
ospfAreaId	1.3.6.1.2.1.14.2.1.1	omproute	RFC1850	R/O
ospfImportAsExtern	1.3.6.1.2.1.14.2.1.3	omproute	RFC1850	R/O
ospfSpfRuns	1.3.6.1.2.1.14.2.1.4	omproute	RFC1850	R/O
ospfAreaBdrRtrCount	1.3.6.1.2.1.14.2.1.5	omproute	RFC1850	R/O
ospfAsBdrRtrCount	1.3.6.1.2.1.14.2.1.6	omproute	RFC1850	R/O
ospfAreaLsaCount	1.3.6.1.2.1.14.2.1.7	omproute	RFC1850	R/O
ospfAreaLsaCksumSum	1.3.6.1.2.1.14.2.1.8	omproute	RFC1850	R/O
ospfAreaSummary	1.3.6.1.2.1.14.2.1.9	omproute	RFC1850	R/O
ospfStubAreaTable	1.3.6.1.2.1.14.3	omproute	RFC1850	N/A
ospfStubAreaEntry	1.3.6.1.2.1.14.3.1	omproute	RFC1850	N/A
ospfStubAreaId	1.3.6.1.2.1.14.3.1.1	omproute	RFC1850	R/O
ospfStubTOS	1.3.6.1.2.1.14.3.1.2	omproute	RFC1850	R/O
ospfStubMetric	1.3.6.1.2.1.14.3.1.3	omproute	RFC1850	R/O
ospfStubMetricType	1.3.6.1.2.1.14.3.1.5	omproute	RFC1850	R/O
ospfLsdbTable	1.3.6.1.2.1.14.4	omproute	RFC1850	N/A
ospfLsdbEntry	1.3.6.1.2.1.14.4.1	omproute	RFC1850	N/A
ospfLsdbAreaId	1.3.6.1.2.1.14.4.1.1	omproute	RFC1850	R/O
ospfLsdbType	1.3.6.1.2.1.14.4.1.2	omproute	RFC1850	R/O
ospfLsdbLsid	1.3.6.1.2.1.14.4.1.3	omproute	RFC1850	R/O
ospfLsdbRouterId	1.3.6.1.2.1.14.4.1.4	omproute	RFC1850	R/O
ospfLsdbSequence	1.3.6.1.2.1.14.4.1.5	omproute	RFC1850	R/O
ospfLsdbAge	1.3.6.1.2.1.14.4.1.6	omproute	RFC1850	R/O
ospfLsdbChecksum	1.3.6.1.2.1.14.4.1.7	omproute	RFC1850	R/O
ospfLsdbAdvertisement	1.3.6.1.2.1.14.4.1.8	omproute	RFC1850	R/O
ospfIfTable	1.3.6.1.2.1.14.7	omproute	RFC1850	N/A
ospfIfEntry	1.3.6.1.2.1.14.7.1	omproute	RFC1850	N/A
ospfIfIpAddress	1.3.6.1.2.1.14.7.1.1	omproute	RFC1850	R/O
ospfAddressLessIf	1.3.6.1.2.1.14.7.1.2	omproute	RFC1850	R/O
ospfIfAreaId	1.3.6.1.2.1.14.7.1.3	omproute	RFC1850	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ospfIfType	1.3.6.1.2.1.14.7.1.4	omproute	RFC1850	R/O
ospfIfAdminStat	1.3.6.1.2.1.14.7.1.5	omproute	RFC1850	R/O
ospfIfRtrPriority	1.3.6.1.2.1.14.7.1.6	omproute	RFC1850	R/O
ospfIfTransitDelay	1.3.6.1.2.1.14.7.1.7	omproute	RFC1850	R/O
ospfIfRetransInterval	1.3.6.1.2.1.14.7.1.8	omproute	RFC1850	R/O
ospfIfHelloInterval	1.3.6.1.2.1.14.7.1.9	omproute	RFC1850	R/O
ospfIfRtrDeadInterval	1.3.6.1.2.1.14.7.1.10	omproute	RFC1850	R/O
ospfIfPollInterval	1.3.6.1.2.1.14.7.1.11	omproute	RFC1850	R/O
ospfIfState	1.3.6.1.2.1.14.7.1.12	omproute	RFC1850	R/O
ospfIfDesignatedRouter	1.3.6.1.2.1.14.7.1.13	omproute	RFC1850	R/O
ospfIfBackupDesignatedRouter	1.3.6.1.2.1.14.7.1.14	omproute	RFC1850	R/O
ospfIfEvents	1.3.6.1.2.1.14.7.1.15	omproute	RFC1850	R/O
ospfIfAuthKey	1.3.6.1.2.1.14.7.1.16	omproute	RFC1850	R/O
ospfIfMulticastForwarding	1.3.6.1.2.1.14.7.1.18	omproute	RFC1850	R/O
ospfIfDemand	1.3.6.1.2.1.14.7.1.19	omproute	RFC1850	R/O
ospfIfAuthType	1.3.6.1.2.1.14.7.1.20	omproute	RFC1850	R/O
ospfIfMetricTable	1.3.6.1.2.1.14.8	omproute	RFC1850	N/A
ospfIfMetricEntry	1.3.6.1.2.1.14.8.1	omproute	RFC1850	N/A
ospfIfMetricIpAddress	1.3.6.1.2.1.14.8.1.1	omproute	RFC1850	R/O
ospfIfMetricAddressLessIf	1.3.6.1.2.1.14.8.1.2	omproute	RFC1850	R/O
ospfIfMetricTOS	1.3.6.1.2.1.14.8.1.3	omproute	RFC1850	R/O
ospfIfMetricValue	1.3.6.1.2.1.14.8.1.4	omproute	RFC1850	R/O
ospfVirtIfTable	1.3.6.1.2.1.14.9	omproute	RFC1850	N/A
ospfVirtIfEntry	1.3.6.1.2.1.14.9.1	omproute	RFC1850	N/A
ospfVirtIfAreaId	1.3.6.1.2.1.14.9.1.1	omproute	RFC1850	R/O
ospfVirtIfNeighbor	1.3.6.1.2.1.14.9.1.2	omproute	RFC1850	R/O
ospfVirtIfTransitDelay	1.3.6.1.2.1.14.9.1.3	omproute	RFC1850	R/O
ospfVirtIfRetransInterval	1.3.6.1.2.1.14.9.1.4	omproute	RFC1850	R/O
ospfVirtIfHelloInterval	1.3.6.1.2.1.14.9.1.5	omproute	RFC1850	R/O
ospfVirtIfRtrDeadInterval	1.3.6.1.2.1.14.9.1.6	omproute	RFC1850	R/O
ospfVirtIfState	1.3.6.1.2.1.14.9.1.7	omproute	RFC1850	R/O
ospfVirtIfEvents	1.3.6.1.2.1.14.9.1.8	omproute	RFC1850	R/O
ospfVirtIfAuthKey	1.3.6.1.2.1.14.9.1.9	omproute	RFC1850	R/O
ospfVirtIfAuthType	1.3.6.1.2.1.14.9.1.11	omproute	RFC1850	R/O
ospfNbrTable	1.3.6.1.2.1.14.10	omproute	RFC1850	N/A
ospfNbrEntry	1.3.6.1.2.1.14.10.1	omproute	RFC1850	N/A
ospfNbrIpAddr	1.3.6.1.2.1.14.10.1.1	omproute	RFC1850	R/O
ospfNbrAddressLessIndex	1.3.6.1.2.1.14.10.1.2	omproute	RFC1850	R/O
ospfNbrRtrId	1.3.6.1.2.1.14.10.1.3	omproute	RFC1850	R/O
ospfNbrOptions	1.3.6.1.2.1.14.10.1.4	omproute	RFC1850	R/O
ospfNbrPriority	1.3.6.1.2.1.14.10.1.5	omproute	RFC1850	R/O
ospfNbrState	1.3.6.1.2.1.14.10.1.6	omproute	RFC1850	R/O
ospfNbrEvents	1.3.6.1.2.1.14.10.1.7	omproute	RFC1850	R/O
ospfNbrLsRetransQLen	1.3.6.1.2.1.14.10.1.8	omproute	RFC1850	R/O
ospfNbmaNbrPermanence	1.3.6.1.2.1.14.10.1.10	omproute	RFC1850	R/O
ospfNbrHelloSuppressed	1.3.6.1.2.1.14.10.1.11	omproute	RFC1850	R/O
ospfVirtNbrTable	1.3.6.1.2.1.14.11	omproute	RFC1850	N/A
ospfVirtNbrEntry	1.3.6.1.2.1.14.11.1	omproute	RFC1850	N/A
ospfVirtNbrArea	1.3.6.1.2.1.14.11.1.1	omproute	RFC1850	R/O
ospfVirtNbrRtrId	1.3.6.1.2.1.14.11.1.2	omproute	RFC1850	R/O
ospfVirtNbrIpAddr	1.3.6.1.2.1.14.11.1.3	omproute	RFC1850	R/O
ospfVirtNbrOptions	1.3.6.1.2.1.14.11.1.4	omproute	RFC1850	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ospfVirtNbrState	1.3.6.1.2.1.14.11.1.5	omproute	RFC1850	R/O
ospfVirtNbrEvents	1.3.6.1.2.1.14.11.1.6	omproute	RFC1850	R/O
ospfVirtNbrLsRetransQLen	1.3.6.1.2.1.14.11.1.7	omproute	RFC1850	R/O
ospfVirtNbrHelloSuppressed	1.3.6.1.2.1.14.11.1.8	omproute	RFC1850	R/O
ospfExtLsdbTable	1.3.6.1.2.1.14.12	omproute	RFC1850	N/A
ospfExtLsdbEntry	1.3.6.1.2.1.14.12.1	omproute	RFC1850	N/A
ospfExtLsdbType	1.3.6.1.2.1.14.12.1.1	omproute	RFC1850	R/O
ospfExtLsdbLsid	1.3.6.1.2.1.14.12.1.2	omproute	RFC1850	R/O
ospfExtLsdbRouterId	1.3.6.1.2.1.14.12.1.3	omproute	RFC1850	R/O
ospfExtLsdbSequence	1.3.6.1.2.1.14.12.1.4	omproute	RFC1850	R/O
ospfExtLsdbAge	1.3.6.1.2.1.14.12.1.5	omproute	RFC1850	R/O
ospfExtLsdbChecksum	1.3.6.1.2.1.14.12.1.6	omproute	RFC1850	R/O
ospfExtLsdbAdvertisement	1.3.6.1.2.1.14.12.1.7	omproute	RFC1850	R/O
ospfAreaAggregateTable	1.3.6.1.2.1.14.14	omproute	RFC1850	N/A
ospfAreaAggregateEntry	1.3.6.1.2.1.14.14.1	omproute	RFC1850	N/A
ospfAreaAggregateAreaID	1.3.6.1.2.1.14.14.1.1	omproute	RFC1850	R/O
ospfAreaAggregateLsdbType	1.3.6.1.2.1.14.14.1.2	omproute	RFC1850	R/O
ospfAreaAggregateNet	1.3.6.1.2.1.14.14.1.3	omproute	RFC1850	R/O
ospfAreaAggregateMask	1.3.6.1.2.1.14.14.1.4	omproute	RFC1850	R/O
ospfAreaAggregateEffect	1.3.6.1.2.1.14.14.1.6	omproute	RFC1850	R/O
ifXTable	1.3.6.1.2.1.31.1.1	TCP/IP	RFC2233	N/A
ifXEntry	1.3.6.1.2.1.31.1.1.1	TCP/IP	RFC2233	N/A
ifName	1.3.6.1.2.1.31.1.1.1.1	TCP/IP	RFC2233	R/O
ifInMulticastPkts	1.3.6.1.2.1.31.1.1.1.2	TCP/IP	RFC2233	R/O
ifInBroadcastPkts	1.3.6.1.2.1.31.1.1.1.3	TCP/IP	RFC2233	R/O
ifOutMulticastPkts	1.3.6.1.2.1.31.1.1.1.4	TCP/IP	RFC2233	R/O
ifOutBroadcastPkts	1.3.6.1.2.1.31.1.1.1.5	TCP/IP	RFC2233	R/O
ifHCInOctets	1.3.6.1.2.1.31.1.1.1.6	TCP/IP	RFC2233	R/O
ifHCOutOctets	1.3.6.1.2.1.31.1.1.1.10	TCP/IP	RFC2233	R/O
ifLinkUpDownTrapEnable	1.3.6.1.2.1.31.1.1.1.14	TCP/IP	RFC2233	R/W
ifHighSpeed	1.3.6.1.2.1.31.1.1.1.15	TCP/IP	RFC2233	R/O
ifPromiscuousMode	1.3.6.1.2.1.31.1.1.1.16	TCP/IP	RFC2233	R/O
ifConnectorPresent	1.3.6.1.2.1.31.1.1.1.17	TCP/IP	RFC2233	R/O
ifAlias	1.3.6.1.2.1.31.1.1.1.18	TCP/IP	RFC2233	R/W
ifCounterDiscontinuityTime	1.3.6.1.2.1.31.1.1.1.19	TCP/IP	RFC2233	R/O
ifStackTable	1.3.6.1.2.1.31.1.2	TCP/IP	RFC2233	N/A
ifStackEntry	1.3.6.1.2.1.31.1.2.1	TCP/IP	RFC2233	N/A
ifStackStatus	1.3.6.1.2.1.31.1.2.1.3	TCP/IP	RFC2233	R/O
ipoaLisTable	1.3.6.1.2.1.10.46.1.2.	TCP/IP	RFC2320	N/A
ipoaLisEntry	1.3.6.1.2.1.10.46.1.2.1	TCP/IP	RFC2320	N/A
ipoaLisSubnetAddr	1.3.6.1.2.1.10.46.1.2.1.1	TCP/IP	RFC2320	R/O
ipoaLisDefaultMtu	1.3.6.1.2.1.10.46.1.2.1.2	TCP/IP	RFC2320	R/O
ipoaLisDefaultEncapsType	1.3.6.1.2.1.10.46.1.2.1.3	TCP/IP	RFC2320	R/O
ipoaLisInactivityTimer	1.3.6.1.2.1.10.46.1.2.1.4	TCP/IP	RFC2320	R/O
ipoaLisMinHoldingTime	1.3.6.1.2.1.10.46.1.2.1.5	TCP/IP	RFC2320	R/O
ipoaLisQDepth	1.3.6.1.2.1.10.46.1.2.1.6	TCP/IP	RFC2320	R/O
ipoaLisMax Calls	1.3.6.1.2.1.10.46.1.2.1.7	TCP/IP	RFC2320	R/O
ipoaLisCacheEntryAge	1.3.6.1.2.1.10.46.1.2.1.8	TCP/IP	RFC2320	R/O
ipoaLisRetries	1.3.6.1.2.1.10.46.1.2.1.9	TCP/IP	RFC2320	R/O
ipoaLisTimeout	1.3.6.1.2.1.10.46.1.2.1.10	TCP/IP	RFC2320	R/O
ipoaLisDefaultPeakCellRate	1.3.6.1.2.1.10.46.1.2.1.11	TCP/IP	RFC2320	R/O
ipoaLisActiveVcs	1.3.6.1.2.1.10.46.1.2.1.12	TCP/IP	RFC2320	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ipoaLisIfMappingTable	1.3.6.1.2.1.10.46.1.3	TCP/IP	RFC2320	N/A
ipoaLisIfMappingEntry	1.3.6.1.2.1.10.46.1.3.1	TCP/IP	RFC2320	N/A
ipoaLisIfMappingRowStatus	1.3.6.1.2.1.10.46.1.3.1.1	TCP/IP	RFC2320	R/O
ipoaArpClientTable	1.3.6.1.2.1.10.46.1.4	TCP/IP	RFC2320	N/A
ipoaArpClientEntry	1.3.6.1.2.1.10.46.1.4.1	TCP/IP	RFC2320	N/A
ipoaArpClientAtmAddr	1.3.6.1.2.1.10.46.1.4.1.1	TCP/IP	RFC2320	R/O
ipoaArpClientSrvrInUse	1.3.6.1.2.1.10.46.1.4.1.2	TCP/IP	RFC2320	R/O
ipoaArpClientInArpInReqs	1.3.6.1.2.1.10.46.1.4.1.3	TCP/IP	RFC2320	R/O
ipoaArpClientInArpOutReqs	1.3.6.1.2.1.10.46.1.4.1.4	TCP/IP	RFC2320	R/O
ipoaArpClientInArpInReplies	1.3.6.1.2.1.10.46.1.4.1.5	TCP/IP	RFC2320	R/O
ipoaArpClientInArpOutReplies	1.3.6.1.2.1.10.46.1.4.1.6	TCP/IP	RFC2320	R/O
ipoaArpClientInArpInvalidInReqs	1.3.6.1.2.1.10.46.1.4.1.7	TCP/IP	RFC2320	R/O
ipoaArpClientInArpInvalidOutReqs	1.3.6.1.2.1.10.46.1.4.1.8	TCP/IP	RFC2320	R/O
ipoaArpClientArpInReqs	1.3.6.1.2.1.10.46.1.4.1.9	TCP/IP	RFC2320	R/O
ipoaArpClientArpOutReqs	1.3.6.1.2.1.10.46.1.4.1.10	TCP/IP	RFC2320	R/O
ipoaArpClientArpInReplies	1.3.6.1.2.1.10.46.1.4.1.11	TCP/IP	RFC2320	R/O
ipoaArpClientArpOutReplies	1.3.6.1.2.1.10.46.1.4.1.12	TCP/IP	RFC2320	R/O
ipoaArpClientArpInNaks	1.3.6.1.2.1.10.46.1.4.1.13	TCP/IP	RFC2320	R/O
ipoaArpClientArpOutNaks	1.3.6.1.2.1.10.46.1.4.1.14	TCP/IP	RFC2320	R/O
ipoaArpClientArpUnknownOps	1.3.6.1.2.1.10.46.1.4.1.15	TCP/IP	RFC2320	R/O
ipoaArpClientArpNoSrvrResps	1.3.6.1.2.1.10.46.1.4.1.16	TCP/IP	RFC2320	R/O
ipoaArpRemoteSrvrTable	1.3.6.1.2.1.10.46.1.6	TCP/IP	RFC2320	N/A
ipoaArpRemoteSrvrEntry	1.3.6.1.2.1.10.46.1.6.1	TCP/IP	RFC2320	N/A
ipoaArpRemoteSrvrIpAddr	1.3.6.1.2.1.10.46.1.6.1.4	TCP/IP	RFC2320	R/O
ipoaVcTable	1.3.6.1.2.1.10.46.1.7	TCP/IP	RFC2320	N/A
ipoaVcEntry	1.3.6.1.2.1.10.46.1.7.1	TCP/IP	RFC2320	N/A
ipoaVcType	1.3.6.1.2.1.10.46.1.7.1.3	TCP/IP	RFC2320	R/O
ipoaVcNegotiatedEncapsType	1.3.6.1.2.1.10.46.1.7.1.4	TCP/IP	RFC2320	R/O
ipoaVcNegotiatedEncapsMtu	1.3.6.1.2.1.10.46.1.7.1.5	TCP/IP	RFC2320	R/O
ipoaConfigPvcTable	1.3.6.1.2.1.10.46.1.8	TCP/IP	RFC2320	N/A
ipoaConfigPvcEntry	1.3.6.1.2.1.10.46.1.8.1	TCP/IP	RFC2320	N/A
ipoaConfigPvcDefaultMtu	1.3.6.1.2.1.10.46.1.8.1.4	TCP/IP	RFC2320	R/O
atmInterfaceConfTable	1.3.6.1.2.1.37.1.2	TCP/IP	RFC1695	N/A
atmInterfaceConfEntry	1.3.6.1.2.1.37.1.2.1	TCP/IP	RFC1695	N/A
atmInterfaceMaxVpcs	1.3.6.1.2.1.37.1.2.1.1	TCP/IP	RFC1695	R/O
atmInterfaceMaxVccs	1.3.6.1.2.1.37.1.2.1.2	TCP/IP	RFC1695	R/O
atmInterfaceConfVpcs	1.3.6.1.2.1.37.1.2.1.3	TCP/IP	RFC1695	R/O
atmInterfaceConfVccs	1.3.6.1.2.1.37.1.2.1.4	TCP/IP	RFC1695	R/O
atmInterfaceMaxActiveVpiBits	1.3.6.1.2.1.37.1.2.1.5	TCP/IP	RFC1695	R/O
atmInterfaceMaxActiveVciBits	1.3.6.1.2.1.37.1.2.1.6	TCP/IP	RFC1695	R/O
atmInterfaceIlmiVpi	1.3.6.1.2.1.37.1.2.1.7	TCP/IP	RFC1695	R/O
atmInterfaceIlmiVci	1.3.6.1.2.1.37.1.2.1.8	TCP/IP	RFC1695	R/O
atmInterfaceAddressType	1.3.6.1.2.1.37.1.2.1.9	TCP/IP	RFC1695	R/O
atmInterfaceAdminAddress	1.3.6.1.2.1.37.1.2.1.10	TCP/IP	RFC1695	R/O
atmInterfaceMyNeighborIpAddress	1.3.6.1.2.1.37.1.2.1.11	TCP/IP	RFC1695	R/O
atmInterfaceMyNeighborIfName	1.3.6.1.2.1.37.1.2.1.12	TCP/IP	RFC1695	R/O
slapmSpinLock	1.3.6.1.3.88.1.1.1	SLA	SLAMIB	R/W
slapmPolicyPurgeTime	1.3.6.1.3.88.1.1.6	SLA	SLAMIB	R/W
slapmPolicyTrapEnable	1.3.6.1.3.88.1.1.7	SLA	SLAMIB	R/W
slapmPolicyTrapFilter	1.3.6.1.3.88.1.1.8	SLA	SLAMIB	R/W
slapmPolicyStatsTable	1.3.6.1.3.88.1.2.1	SLA	SLAMIB	N/A
slapmPolicyStatsEntry	1.3.6.1.3.88.1.2.1.1	SLA	SLAMIB	N/A

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
slapmPolicyStatsSystemAddress	1.3.6.1.3.88.1.2.1.1.1	SLA	SLAMIB	N/A
slapmPolicyStatsPolicyName	1.3.6.1.3.88.1.2.1.1.2	SLA	SLAMIB	N/A
slapmPolicyStatsTrafficProfileName	1.3.6.1.3.88.1.2.1.1.3	SLA	SLAMIB	N/A
slapmPolicyStatsOperStatus	1.3.6.1.3.88.1.2.1.1.4	SLA	SLAMIB	R/O
slapmPolicyStatsActiveConns	1.3.6.1.3.88.1.2.1.1.5	SLA	SLAMIB	R/O
slapmPolicyStatsTotalConns	1.3.6.1.3.88.1.2.1.1.6	SLA	SLAMIB	R/O
slapmPolicyStatsFirstActivated	1.3.6.1.3.88.1.2.1.1.7	SLA	SLAMIB	R/O
slapmPolicyStatsLastMapping	1.3.6.1.3.88.1.2.1.1.8	SLA	SLAMIB	R/O
slapmPolicyStatsInOctets	1.3.6.1.3.88.1.2.1.1.9	SLA	SLAMIB	R/O
slapmPolicyStatsOutOctets	1.3.6.1.3.88.1.2.1.1.10	SLA	SLAMIB	R/O
slapmPolicyStatsConnectionLimit	1.3.6.1.3.88.1.2.1.1.11	SLA	SLAMIB	R/O
slapmPolicyStatsCountAccepts	1.3.6.1.3.88.1.2.1.1.12	SLA	SLAMIB	R/O
slapmPolicyStatsCountDenies	1.3.6.1.3.88.1.2.1.1.13	SLA	SLAMIB	R/O
slapmPolicyStatsInDiscards	1.3.6.1.3.88.1.2.1.1.14	SLA	SLAMIB	R/O
slapmPolicyStatsOutDiscards	1.3.6.1.3.88.1.2.1.1.15	SLA	SLAMIB	R/O
slapmPolicyStatsInPackets	1.3.6.1.3.88.1.2.1.1.16	SLA	SLAMIB	R/O
slapmPolicyStatsOutPackets	1.3.6.1.3.88.1.2.1.1.17	SLA	SLAMIB	R/O
slapmPolicyStatsMinRate	1.3.6.1.3.88.1.2.1.1.20	SLA	SLAMIB	R/O
slapmPolicyStatsMaxRate	1.3.6.1.3.88.1.2.1.1.21	SLA	SLAMIB	R/O
slapmPolicyStatsMaxDelay	1.3.6.1.3.88.1.2.1.1.22	SLA	SLAMIB	R/O
slapmPolicyMonitorTable	1.3.6.1.3.88.1.2.2	SLA	SLAMIB	N/A
slapmPolicyMonitorEntry	1.3.6.1.3.88.1.2.2.1	SLA	SLAMIB	N/A
slapmPolicyMonitorOwnerIndex	1.3.6.1.3.88.1.2.2.1.1	SLA	SLAMIB	N/A
slapmPolicyMonitorSystemAddress	1.3.6.1.3.88.1.2.2.1.2	SLA	SLAMIB	N/A
slapmPolicyMonitorPolicyName	1.3.6.1.3.88.1.2.2.1.3	SLA	SLAMIB	N/A
slapmPolicyMonitorTrafficProfileName	1.3.6.1.3.88.1.2.2.1.4	SLA	SLAMIB	N/A
slapmPolicyMonitorControl	1.3.6.1.3.88.1.2.2.1.5	SLA	SLAMIB	R/C
slapmPolicyMonitorStatus	1.3.6.1.3.88.1.2.2.1.6	SLA	SLAMIB	R/O
slapmPolicyMonitorInterval	1.3.6.1.3.88.1.2.2.1.7	SLA	SLAMIB	R/C
slapmPolicyMonitorIntTime	1.3.6.1.3.88.1.2.2.1.8	SLA	SLAMIB	R/O
slapmPolicyMonitorCurrentInRate	1.3.6.1.3.88.1.2.2.1.9	SLA	SLAMIB	R/O
slapmPolicyMonitorCurrentOutRate	1.3.6.1.3.88.1.2.2.1.10	SLA	SLAMIB	R/O
slapmPolicyMonitorMinRateLow	1.3.6.1.3.88.1.2.2.1.11	SLA	SLAMIB	R/C
slapmPolicyMonitorMinRateHigh	1.3.6.1.3.88.1.2.2.1.12	SLA	SLAMIB	R/C
slapmPolicyMonitorMaxRateHigh	1.3.6.1.3.88.1.2.2.1.13	SLA	SLAMIB	R/C
slapmPolicyMonitorMaxRateLow	1.3.6.1.3.88.1.2.2.1.14	SLA	SLAMIB	R/C
slapmPolicyMonitorMaxDelayHigh	1.3.6.1.3.88.1.2.2.1.15	SLA	SLAMIB	R/C
slapmPolicyMonitorMaxDelayLow	1.3.6.1.3.88.1.2.2.1.16	SLA	SLAMIB	R/C
slapmPolicyMonitorMinInRateNotAchieves	1.3.6.1.3.88.1.2.2.1.17	SLA	SLAMIB	R/O
slapmPolicyMonitorMaxInRateExceeds	1.3.6.1.3.88.1.2.2.1.18	SLA	SLAMIB	R/O
slapmPolicyMonitorMaxDelayExceeds	1.3.6.1.3.88.1.2.2.1.19	SLA	SLAMIB	R/O
slapmPolicyMonitorMinOutRateNotAchieves	1.3.6.1.3.88.1.2.2.1.20	SLA	SLAMIB	R/O
slapmPolicyMonitorMaxOutRateExceeds	1.3.6.1.3.88.1.2.2.1.21	SLA	SLAMIB	R/O
slapmPolicyMonitorCurrentDelayRate	1.3.6.1.3.88.1.2.2.1.22	SLA	SLAMIB	R/O
slapmPolicyMonitorRowStatus	1.3.6.1.3.88.1.2.2.1.23	SLA	SLAMIB	R/C
slapmSubcomponentTable	1.3.6.1.3.88.1.2.3	SLA	SLAMIB	N/A
slapmSubcomponentEntry	1.3.6.1.3.88.1.2.3.1	SLA	SLAMIB	R/O
slapmSubcomponentRemAddress	1.3.6.1.3.88.1.2.3.1.1	SLA	SLAMIB	N/A
slapmSubcomponentRemPort	1.3.6.1.3.88.1.2.3.1.2	SLA	SLAMIB	N/A
slapmSubcomponentLocalAddress	1.3.6.1.3.88.1.2.3.1.3	SLA	SLAMIB	N/A
slapmSubcomponentLocalPort	1.3.6.1.3.88.1.2.3.1.4	SLA	SLAMIB	N/A
slapmSubcomponentProtocol	1.3.6.1.3.88.1.2.3.1.5	SLA	SLAMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
slapmSubcomponentSystemAddress	1.3.6.1.3.88.1.2.3.1.6	SLA	SLAMIB	R/O
slapmSubcomponentPolicyName	1.3.6.1.3.88.1.2.3.1.7	SLA	SLAMIB	R/O
slapmSubcomponentTrafficProfileName	1.3.6.1.3.88.1.2.3.1.8	SLA	SLAMIB	R/O
slapmSubcomponentLastActivity	1.3.6.1.3.88.1.2.3.1.9	SLA	SLAMIB	R/O
slapmSubcomponentInOctets	1.3.6.1.3.88.1.2.3.1.10	SLA	SLAMIB	R/O
slapmSubcomponentOutOctets	1.3.6.1.3.88.1.2.3.1.11	SLA	SLAMIB	R/O
slapmSubcomponentTcpOutBufferedOctets	1.3.6.1.3.88.1.2.3.1.12	SLA	SLAMIB	R/O
slapmSubcomponentTcpInBufferedOctets	1.3.6.1.3.88.1.2.3.1.13	SLA	SLAMIB	R/O
slapmSubcomponentTcpReXmts	1.3.6.1.3.88.1.2.3.1.14	SLA	SLAMIB	R/O
slapmSubcomponentTcpRoundTripTime	1.3.6.1.3.88.1.2.3.1.15	SLA	SLAMIB	R/O
slapmSubcomponentTcpRoundTripVariance	1.3.6.1.3.88.1.2.3.1.16	SLA	SLAMIB	R/O
slapmSubcomponentInPdus	1.3.6.1.3.88.1.2.3.1.17	SLA	SLAMIB	R/O
slapmSubcomponentOutPdus	1.3.6.1.3.88.1.2.3.1.18	SLA	SLAMIB	R/O
slapmSubcomponentApplName	1.3.6.1.3.88.1.2.3.1.19	SLA	SLAMIB	R/O
slapmSubcomponentMonitorStatus	1.3.6.1.3.88.1.2.3.1.20	SLA	SLAMIB	R/O
slapmSubcomponentMonitorIntTime	1.3.6.1.3.88.1.2.3.1.21	SLA	SLAMIB	R/O
slapmSubcomponentMonitorCurrentInRate	1.3.6.1.3.88.1.2.3.1.22	SLA	SLAMIB	R/O
slapmSubcomponentMonitorCurrentOutRate	1.3.6.1.3.88.1.2.3.1.23	SLA	SLAMIB	R/O
dpiPort	1.3.6.1.4.1.2.2.1.1.0	Agent	RFC1592	R/O
dpiPortForTCP	1.3.6.1.4.1.2.2.1.1.1.0	Agent	RFC1592	R/O
dpiPortForUDP	1.3.6.1.4.1.2.2.1.1.2.0	Agent	RFC1592	R/O
dpiPathNameForUnixStream	1.3.6.1.4.1.2.2.1.1.3.0	Agent	RFC1592B	R/O
saDefaultTimeout	1.3.6.1.4.1.2.4.12.1	Agent	SAMIB	R/W
saMaxTimeout	1.3.6.1.4.1.2.4.12.2	Agent	SAMIB	R/W
saAllowDuplicateIDs	1.3.6.1.4.1.2.4.12.3	Agent	SAMIB	R/W
saNumber	1.3.6.1.4.1.2.4.12.4	Agent	SAMIB	R/O
saAllPacketsIn	1.3.6.1.4.1.2.4.12.5	Agent	SAMIB	R/O
saAllPacketsOut	1.3.6.1.4.1.2.4.12.6	Agent	SAMIB	R/O
saTable	1.3.6.1.4.1.2.4.12.7	Agent	SAMIB	N/A
saEntry	1.3.6.1.4.1.2.4.12.7.1	Agent	SAMIB	N/A
saIndex	1.3.6.1.4.1.2.4.12.7.1.1	Agent	SAMIB	R/O
saIdentifier	1.3.6.1.4.1.2.4.12.7.1.2	Agent	SAMIB	R/O
saDescription	1.3.6.1.4.1.2.4.12.7.1.3	Agent	SAMIB	R/O
saStatus	1.3.6.1.4.1.2.4.12.7.1.4	Agent	SAMIB	R/W
saStatusChangeTime	1.3.6.1.4.1.2.4.12.7.1.5	Agent	SAMIB	R/O
saProtocol	1.3.6.1.4.1.2.4.12.7.1.6	Agent	SAMIB	R/O
saProtocolVersion	1.3.6.1.4.1.2.4.12.7.1.7	Agent	SAMIB	R/O
saProtocolRelease	1.3.6.1.4.1.2.4.12.7.1.8	Agent	SAMIB	R/O
saTransport	1.3.6.1.4.1.2.4.12.7.1.9	Agent	SAMIB	R/O
saTransportAddress	1.3.6.1.4.1.2.4.12.7.1.10	Agent	SAMIB	R/O
saTimeout	1.3.6.1.4.1.2.4.12.7.1.11	Agent	SAMIB	R/W
saMaxVarBinds	1.3.6.1.4.1.2.4.12.7.1.12	Agent	SAMIB	R/O
saPacketsIn	1.3.6.1.4.1.2.4.12.7.1.13	Agent	SAMIB	R/O
saPacketsOut	1.3.6.1.4.1.2.4.12.7.1.14	Agent	SAMIB	R/O
saTreeTable	1.3.6.1.4.1.2.4.12.8	Agent	SAMIB	N/A
saTreeEntry	1.3.6.1.4.1.2.4.12.8.1	Agent	SAMIB	N/A
saTsubtree	1.3.6.1.4.1.2.4.12.8.1.1	Agent	SAMIB	R/O
saTpriority	1.3.6.1.4.1.2.4.12.8.1.2	Agent	SAMIB	R/O
saTindex	1.3.6.1.4.1.2.4.12.8.1.3	Agent	SAMIB	R/O
saTstatus	1.3.6.1.4.1.2.4.12.8.1.4	Agent	SAMIB	R/W
saTtimeout	1.3.6.1.4.1.2.4.12.8.1.5	Agent	SAMIB	R/W
ibm3172Descr	1.3.6.1.4.1.2.6.1.1.1.1	TCP/IP	ibm3172MIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibm3172Contact	1.3.6.1.4.1.2.6.1.1.1.2	TCP/IP	ibm3172MIB	R/O
ibm3172Location	1.3.6.1.4.1.2.6.1.1.1.3	TCP/IP	ibm3172MIB	R/O
ibm3172ifNumber	1.3.6.1.4.1.2.6.1.1.1.4	TCP/IP	ibm3172MIB	R/O
ibm3172ifTrapEnable	1.3.6.1.4.1.2.6.1.2.1.1	TCP/IP	ibm3172MIB	R/O
ibm3172ifInChanOctets	1.3.6.1.4.1.2.6.1.3.1.1	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutChanOctets	1.3.6.1.4.1.2.6.1.3.1.2	TCP/IP	ibm3172MIB	R/O
ibm3172ifInChanBlocks	1.3.6.1.4.1.2.6.1.3.1.3	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutChanBlocks	1.3.6.1.4.1.2.6.1.3.1.4	TCP/IP	ibm3172MIB	R/O
ibm3172ifInLANOctets	1.3.6.1.4.1.2.6.1.4.1.1	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutLANOctets	1.3.6.1.4.1.2.6.1.4.1.2	TCP/IP	ibm3172MIB	R/O
ibm3172ifInLANFrames	1.3.6.1.4.1.2.6.1.4.1.3	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutLANFrames	1.3.6.1.4.1.2.6.1.4.1.4	TCP/IP	ibm3172MIB	R/O
ibm3172ifInLANErrors	1.3.6.1.4.1.2.6.1.4.1.5	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutLANErrors	1.3.6.1.4.1.2.6.1.4.1.6	TCP/IP	ibm3172MIB	R/O
ibm3172ifInLANDiscards	1.3.6.1.4.1.2.6.1.4.1.7	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutLANDiscards	1.3.6.1.4.1.2.6.1.4.1.8	TCP/IP	ibm3172MIB	R/O
ibm3172ifBlkRcvOctets	1.3.6.1.4.1.2.6.1.5.1.1	TCP/IP	ibm3172MIB	R/O
ibm3172ifBlkXmitOctets	1.3.6.1.4.1.2.6.1.5.1.2	TCP/IP	ibm3172MIB	R/O
ibm3172ifBlkRcvFrames	1.3.6.1.4.1.2.6.1.5.1.3	TCP/IP	ibm3172MIB	R/O
ibm3172ifBlkXmitBlocks	1.3.6.1.4.1.2.6.1.5.1.4	TCP/IP	ibm3172MIB	R/O
ibm3172ifInBlkErrors	1.3.6.1.4.1.2.6.1.5.1.5	TCP/IP	ibm3172MIB	R/O
ibm3172ifInBlkDiscards	1.3.6.1.4.1.2.6.1.5.1.6	TCP/IP	ibm3172MIB	R/O
ibm3172ifDblkRcvOctets	1.3.6.1.4.1.2.6.1.6.1.1	TCP/IP	ibm3172MIB	R/O
ibm3172ifDblkXmitOctets	1.3.6.1.4.1.2.6.1.6.1.2	TCP/IP	ibm3172MIB	R/O
ibm3172ifDblkRcvBlocks	1.3.6.1.4.1.2.6.1.6.1.3	TCP/IP	ibm3172MIB	R/O
ibm3172ifDblkXmitFrames	1.3.6.1.4.1.2.6.1.6.1.4	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutDblkErrors	1.3.6.1.4.1.2.6.1.6.1.5	TCP/IP	ibm3172MIB	R/O
ibm3172ifOutDblkDiscards	1.3.6.1.4.1.2.6.1.6.1.6	TCP/IP	ibm3172MIB	R/O
ibm3172ifDeviceNumber	1.3.6.1.4.1.2.6.1.7.1.1	TCP/IP	ibm3172MIB	R/O
ibmRemotePingTable	1.3.6.1.4.1.2.6.19.2.2.1.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmRemotePingEntry	1.3.6.1.4.1.2.6.19.2.2.1.1.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsRPingResponseTime	1.3.6.1.4.1.2.6.19.2.2.1.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsSubagentCacheTime	1.3.6.1.4.1.2.6.19.2.2.2.1	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsIgnoreRedirect	1.3.6.1.4.1.2.6.19.2.2.2.2	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsArpCacheTimeout	1.3.6.1.4.1.2.6.19.2.2.2.3	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsTcpKeepAliveTimer	1.3.6.1.4.1.2.6.19.2.2.2.4	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsTcpReceiveBufferSize	1.3.6.1.4.1.2.6.19.2.2.2.5	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsTcpSendBufferSize	1.3.6.1.4.1.2.6.19.2.2.2.6	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsUdpChecksum	1.3.6.1.4.1.2.6.19.2.2.2.7	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsIplDateAndTime	1.3.6.1.4.1.2.6.19.2.2.2.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsNoUdpQueueLimit	1.3.6.1.4.1.2.6.19.2.2.2.9	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsSoMaxConn	1.3.6.1.4.1.2.6.19.2.2.2.10	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsTcpipProcname	1.3.6.1.4.1.2.6.19.2.2.2.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpipAsid	1.3.6.1.4.1.2.6.19.2.2.2.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsSourceVipaEnabled	1.3.6.1.4.1.2.6.19.2.2.2.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsOsasfSysplexName	1.3.6.1.4.1.2.6.19.2.2.2.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsOsasfHostName	1.3.6.1.4.1.2.6.19.2.2.2.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsOsasfProductVersion	1.3.6.1.4.1.2.6.19.2.2.2.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPrimaryInterfaceIfIndex	1.3.6.1.4.1.2.6.19.2.2.2.17	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsIpMaxReassemblySize	1.3.6.1.4.1.2.6.19.2.2.2.18	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpRestrictLowPorts	1.3.6.1.4.1.2.6.19.2.2.2.19	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsUdpRestrictLowPorts	1.3.6.1.4.1.2.6.19.2.2.2.20	TCP/IP	ibmTCPIPmvsMIB	R/W

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibmMvsUdpSendBufferSize	1.3.6.1.4.1.2.6.19.2.2.2.21	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsUdpRecvBufferSize	1.3.6.1.4.1.2.6.19.2.2.2.22	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsTcpipStatisticsEnabled	1.3.6.1.4.1.2.6.19.2.2.2.23	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsFirewallEnabled	1.3.6.1.4.1.2.6.19.2.2.2.24	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsMaximumRetransmitTime	1.3.6.1.4.1.2.6.19.2.2.2.25	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsMinimumRetransmitTime	1.3.6.1.4.1.2.6.19.2.2.2.26	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsRoundTripGain	1.3.6.1.4.1.2.6.19.2.2.2.27	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsVarianceGain	1.3.6.1.4.1.2.6.19.2.2.2.28	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsVarianceMultiplier	1.3.6.1.4.1.2.6.19.2.2.2.29	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsSendGarbageEnabled	1.3.6.1.4.1.2.6.19.2.2.2.30	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpMaxReceiveBufferSize	1.3.6.1.4.1.2.6.19.2.2.2.31	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsPathMtuDscEnabled	1.3.6.1.4.1.2.6.19.2.2.2.33	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsMultipathType	1.3.6.1.4.1.2.6.19.2.2.2.34	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsIpForwarding	1.3.6.1.4.1.2.6.19.2.2.2.35	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsDeviceType	1.3.6.1.4.1.2.6.19.2.2.3.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceBaseNumber	1.3.6.1.4.1.2.6.19.2.2.3.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceIoBufferSize	1.3.6.1.4.1.2.6.19.2.2.3.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceAutoRestart	1.3.6.1.4.1.2.6.19.2.2.3.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceNetmanEnabled	1.3.6.1.4.1.2.6.19.2.2.3.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceHostClawName	1.3.6.1.4.1.2.6.19.2.2.3.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceWorkstationClawName	1.3.6.1.4.1.2.6.19.2.2.3.1.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceReadBuffers	1.3.6.1.4.1.2.6.19.2.2.3.1.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceReadSize	1.3.6.1.4.1.2.6.19.2.2.3.1.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceWriteBuffers	1.3.6.1.4.1.2.6.19.2.2.3.1.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceWriteSize	1.3.6.1.4.1.2.6.19.2.2.3.1.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceProcname	1.3.6.1.4.1.2.6.19.2.2.3.1.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceIncomingSvcEnabled	1.3.6.1.4.1.2.6.19.2.2.3.1.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceLuName	1.3.6.1.4.1.2.6.19.2.2.3.1.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsDeviceRouterStatus	1.3.6.1.4.1.2.6.19.2.2.3.1.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkType	1.3.6.1.4.1.2.6.19.2.2.3.2.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkDeviceIndex	1.3.6.1.4.1.2.6.19.2.2.3.2.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkAdapterAddr	1.3.6.1.4.1.2.6.19.2.2.3.2.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkNumber	1.3.6.1.4.1.2.6.19.2.2.3.2.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkIbmtrCanonical	1.3.6.1.4.1.2.6.19.2.2.3.2.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkIbmtrBcast	1.3.6.1.4.1.2.6.19.2.2.3.2.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkMcast	1.3.6.1.4.1.2.6.19.2.2.3.2.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkChecksumEnabled	1.3.6.1.4.1.2.6.19.2.2.3.2.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsLinkMcastRefCount	1.3.6.1.4.1.2.6.19.2.2.3.3.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortNumberLow	1.3.6.1.4.1.2.6.19.2.2.4.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortNumberHigh	1.3.6.1.4.1.2.6.19.2.2.4.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortProtocol	1.3.6.1.4.1.2.6.19.2.2.4.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortProcName	1.3.6.1.4.1.2.6.19.2.2.4.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortAutoLoggable	1.3.6.1.4.1.2.6.19.2.2.4.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortDelayAcks	1.3.6.1.4.1.2.6.19.2.2.4.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortOptMaxSegmentSize	1.3.6.1.4.1.2.6.19.2.2.4.1.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsPortSharePort	1.3.6.1.4.1.2.6.19.2.2.4.1.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayMaximumRetransmitTime	1.3.6.1.4.1.2.6.19.2.2.5.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayMinimumRetransmitTime	1.3.6.1.4.1.2.6.19.2.2.5.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayRoundTripGain	1.3.6.1.4.1.2.6.19.2.2.5.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayVarianceGain	1.3.6.1.4.1.2.6.19.2.2.5.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayVarianceMultiplier	1.3.6.1.4.1.2.6.19.2.2.5.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsGatewayDelayAcks	1.3.6.1.4.1.2.6.19.2.2.5.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
osasfChannelTable	1.3.6.1.4.1.2.6.19.2.2.6.1	TCP/IP	ibmTCPIPmvsMIB	N/A
osasfChannelEntry	1.3.6.1.4.1.2.6.19.2.2.6.1.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmOsasfChannelNumber	1.3.6.1.4.1.2.6.19.2.2.6.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelType	1.3.6.1.4.1.2.6.19.2.2.6.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelSubType	1.3.6.1.4.1.2.6.19.2.2.6.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelMode	1.3.6.1.4.1.2.6.19.2.2.6.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelHwModel	1.3.6.1.4.1.2.6.19.2.2.6.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelState	1.3.6.1.4.1.2.6.19.2.2.6.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelShared	1.3.6.1.4.1.2.6.19.2.2.6.1.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelNumPorts	1.3.6.1.4.1.2.6.19.2.2.6.1.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelDeterNodeDesc	1.3.6.1.4.1.2.6.19.2.2.6.1.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelControlUnitNumber	1.3.6.1.4.1.2.6.19.2.2.6.1.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelCodeLevel	1.3.6.1.4.1.2.6.19.2.2.6.1.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelEcLevel	1.3.6.1.4.1.2.6.19.2.2.6.1.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelCurLparName	1.3.6.1.4.1.2.6.19.2.2.6.1.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelCurLparNum	1.3.6.1.4.1.2.6.19.2.2.6.1.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelManParnName	1.3.6.1.4.1.2.6.19.2.2.6.1.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelManParnNum	1.3.6.1.4.1.2.6.19.2.2.6.1.1.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelDate	1.3.6.1.4.1.2.6.19.2.2.6.1.1.17	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelTime	1.3.6.1.4.1.2.6.19.2.2.6.1.1.18	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfChannelFlashLevel	1.3.6.1.4.1.2.6.19.2.2.6.1.1.19	TCP/IP	ibmTCPIPmvsMIB	R/O
osasfPortTable	1.3.6.1.4.1.2.6.19.2.2.6.2	TCP/IP	ibmTCPIPmvsMIB	N/A
osasfPortEntry	1.3.6.1.4.1.2.6.19.2.2.6.2.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmOsasfPortNumber	1.3.6.1.4.1.2.6.19.2.2.6.2.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortType	1.3.6.1.4.1.2.6.19.2.2.6.2.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortHardwareState	1.3.6.1.4.1.2.6.19.2.2.6.2.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortMediaType	1.3.6.1.4.1.2.6.19.2.2.6.2.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortUniType	1.3.6.1.4.1.2.6.19.2.2.6.2.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortUniVersion	1.3.6.1.4.1.2.6.19.2.2.6.2.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortNetPrefix	1.3.6.1.4.1.2.6.19.2.2.6.2.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortNetPrefixPrefix	1.3.6.1.4.1.2.6.19.2.2.6.2.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortNetPrefixStatus	1.3.6.1.4.1.2.6.19.2.2.6.2.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortCodeLoadStatus	1.3.6.1.4.1.2.6.19.2.2.6.2.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortMacAddrBurntIn	1.3.6.1.4.1.2.6.19.2.2.6.2.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortMacAddrActive	1.3.6.1.4.1.2.6.19.2.2.6.2.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortMaxPcmConnections	1.3.6.1.4.1.2.6.19.2.2.6.2.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortPcmName	1.3.6.1.4.1.2.6.19.2.2.6.2.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortAAL5InPackets	1.3.6.1.4.1.2.6.19.2.2.6.2.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortAAL5OutPackets	1.3.6.1.4.1.2.6.19.2.2.6.2.1.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPortIpAddress	1.3.6.1.4.1.2.6.19.2.2.6.2.1.17	TCP/IP	ibmTCPIPmvsMIB	R/W
osasfPvcTable	1.3.6.1.4.1.2.6.19.2.2.6.3	TCP/IP	ibmTCPIPmvsMIB	N/A
osasfPvcEntry	1.3.6.1.4.1.2.6.19.2.2.6.3.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmOsasfPvcName	1.3.6.1.4.1.2.6.19.2.2.6.3.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcBestEffort	1.3.6.1.4.1.2.6.19.2.2.6.3.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcFwdPeakCellRate	1.3.6.1.4.1.2.6.19.2.2.6.3.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcBwdPeakCellRate	1.3.6.1.4.1.2.6.19.2.2.6.3.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcFwdsustainCellRate	1.3.6.1.4.1.2.6.19.2.2.6.3.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcBwdsustainCellRate	1.3.6.1.4.1.2.6.19.2.2.6.3.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcFwdCellBurstSize	1.3.6.1.4.1.2.6.19.2.2.6.3.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcBwdCellBurstSize	1.3.6.1.4.1.2.6.19.2.2.6.3.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcVpi	1.3.6.1.4.1.2.6.19.2.2.6.3.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcVci	1.3.6.1.4.1.2.6.19.2.2.6.3.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibmMvsAtmOsasfPvcFwdMaxAal5PduSize	1.3.6.1.4.1.2.6.19.2.2.6.3.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmOsasfPvcBwdMaxAal5PduSize	1.3.6.1.4.1.2.6.19.2.2.6.3.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeTable	1.3.6.1.4.1.2.6.19.2.2.6.4	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmSnaLeEntry	1.3.6.1.4.1.2.6.19.2.2.6.4.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmSnaLeLlcTi	1.3.6.1.4.1.2.6.19.2.2.6.4.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeLlcT1	1.3.6.1.4.1.2.6.19.2.2.6.4.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeLlcT2	1.3.6.1.4.1.2.6.19.2.2.6.4.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeMaxStations	1.3.6.1.4.1.2.6.19.2.2.6.4.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeMaxSaps	1.3.6.1.4.1.2.6.19.2.2.6.4.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeMaxIn	1.3.6.1.4.1.2.6.19.2.2.6.4.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeMaxOut	1.3.6.1.4.1.2.6.19.2.2.6.4.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeCrsGroupAddress	1.3.6.1.4.1.2.6.19.2.2.6.4.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeCrsUserData	1.3.6.1.4.1.2.6.19.2.2.6.4.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeClientEnableState	1.3.6.1.4.1.2.6.19.2.2.6.4.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeBestEffortPeakRate	1.3.6.1.4.1.2.6.19.2.2.6.4.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeMaxLECConnections	1.3.6.1.4.1.2.6.19.2.2.6.4.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeTrEnableLoadBalancing	1.3.6.1.4.1.2.6.19.2.2.6.4.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeTrLoadBalancing	1.3.6.1.4.1.2.6.19.2.2.6.4.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmSnaLeTrSessionDelay	1.3.6.1.4.1.2.6.19.2.2.6.4.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigTable	1.3.6.1.4.1.2.6.19.2.2.6.5	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecConfigEntry	1.3.6.1.4.1.2.6.19.2.2.6.5.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecConfigMode	1.3.6.1.4.1.2.6.19.2.2.6.5.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigLanType	1.3.6.1.4.1.2.6.19.2.2.6.5.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMaxDataFrameSize	1.3.6.1.4.1.2.6.19.2.2.6.5.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigLanName	1.3.6.1.4.1.2.6.19.2.2.6.5.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigLesAtmAddress	1.3.6.1.4.1.2.6.19.2.2.6.5.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlTimeout	1.3.6.1.4.1.2.6.19.2.2.6.5.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMaxUnknownFrameCount	1.3.6.1.4.1.2.6.19.2.2.6.5.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecUnknownFrameTime	1.3.6.1.4.1.2.6.19.2.2.6.5.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecVccTimeoutPeriod	1.3.6.1.4.1.2.6.19.2.2.6.5.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMaxRetryCount	1.3.6.1.4.1.2.6.19.2.2.6.5.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecAgingTime	1.3.6.1.4.1.2.6.19.2.2.6.5.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecForwardDelayTime	1.3.6.1.4.1.2.6.19.2.2.6.5.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecExpectedArpResponseTime	1.3.6.1.4.1.2.6.19.2.2.6.5.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecFlushTimeout	1.3.6.1.4.1.2.6.19.2.2.6.5.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecPathSwitchingDelay	1.3.6.1.4.1.2.6.19.2.2.6.5.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecLocalSegmentID	1.3.6.1.4.1.2.6.19.2.2.6.5.1.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendType	1.3.6.1.4.1.2.6.19.2.2.6.5.1.17	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendAvgRate	1.3.6.1.4.1.2.6.19.2.2.6.5.1.18	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendPeakRate	1.3.6.1.4.1.2.6.19.2.2.6.5.1.19	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConnectionCompleteTimer	1.3.6.1.4.1.2.6.19.2.2.6.5.1.20	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecStatusTable	1.3.6.1.4.1.2.6.19.2.2.6.6	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecStatusEntry	1.3.6.1.4.1.2.6.19.2.2.6.6.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecPrimaryAtmAddress	1.3.6.1.4.1.2.6.19.2.2.6.6.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecID	1.3.6.1.4.1.2.6.19.2.2.6.6.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecInterfaceState	1.3.6.1.4.1.2.6.19.2.2.6.6.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecLastFailureRespCode	1.3.6.1.4.1.2.6.19.2.2.6.6.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecLastFailureState	1.3.6.1.4.1.2.6.19.2.2.6.6.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecProtocol	1.3.6.1.4.1.2.6.19.2.2.6.6.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecVersion	1.3.6.1.4.1.2.6.19.2.2.6.6.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecTopologyChange	1.3.6.1.4.1.2.6.19.2.2.6.6.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecconfigServerAtmAddress	1.3.6.1.4.1.2.6.19.2.2.6.6.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibmMvsAtmLecConfigSource	1.3.6.1.4.1.2.6.19.2.2.6.6.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecActualLanType	1.3.6.1.4.1.2.6.19.2.2.6.6.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecActualMaxDataFrameSize	1.3.6.1.4.1.2.6.19.2.2.6.6.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecActualLanName	1.3.6.1.4.1.2.6.19.2.2.6.6.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecAtmAddress	1.3.6.1.4.1.2.6.19.2.2.6.6.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecProxyClient	1.3.6.1.4.1.2.6.19.2.2.6.6.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecStatisticsTable	1.3.6.1.4.1.2.6.19.2.2.6.7	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecStatisticsEntry	1.3.6.1.4.1.2.6.19.2.2.6.7.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecArpRequestsOut	1.3.6.1.4.1.2.6.19.2.2.6.7.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecArpRequestsIn	1.3.6.1.4.1.2.6.19.2.2.6.7.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecArpRepliesOut	1.3.6.1.4.1.2.6.19.2.2.6.7.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecArpRepliesIn	1.3.6.1.4.1.2.6.19.2.2.6.7.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlFramesOut	1.3.6.1.4.1.2.6.19.2.2.6.7.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlFramesIn	1.3.6.1.4.1.2.6.19.2.2.6.7.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecSvcFailures	1.3.6.1.4.1.2.6.19.2.2.6.7.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecServerTable	1.3.6.1.4.1.2.6.19.2.2.6.8	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecServerEntry	1.3.6.1.4.1.2.6.19.2.2.6.8.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecConfigDirectInterface	1.3.6.1.4.1.2.6.19.2.2.6.8.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigDirectVPI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecConfigDirectVCI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDirectInterface	1.3.6.1.4.1.2.6.19.2.2.6.8.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDirectVPI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDirectVCI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDistributeInterface	1.3.6.1.4.1.2.6.19.2.2.6.8.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDistributeVPI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecControlDistributeVCI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendInterface	1.3.6.1.4.1.2.6.19.2.2.6.8.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendVPI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastSendVCI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastFwdInterface	1.3.6.1.4.1.2.6.19.2.2.6.8.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastFwdVPI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.14	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMulticastFwdVCI	1.3.6.1.4.1.2.6.19.2.2.6.8.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsAtmLecMacAddressTable	1.3.6.1.4.1.2.6.19.2.2.6.9	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecMacAddressEntry	1.3.6.1.4.1.2.6.19.2.2.6.9.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsAtmLecMacAddress	1.3.6.1.4.1.2.6.19.2.2.6.9.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmTcpipMvsTcpConnTable	1.3.6.1.4.1.2.6.19.2.2.7.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmTcpipMvsTcpConnEntry	1.3.6.1.4.1.2.6.19.2.2.7.1.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsTcpConnLastActivity	1.3.6.1.4.1.2.6.19.2.2.7.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnBytesIn	1.3.6.1.4.1.2.6.19.2.2.7.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnBytesOut	1.3.6.1.4.1.2.6.19.2.2.7.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnIpTos	1.3.6.1.4.1.2.6.19.2.2.7.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOptions	1.3.6.1.4.1.2.6.19.2.2.7.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOutBuffered	1.3.6.1.4.1.2.6.19.2.2.7.1.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnUsrSndNxt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSndNxt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSndUna	1.3.6.1.4.1.2.6.19.2.2.7.1.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOutgoingPush	1.3.6.1.4.1.2.6.19.2.2.7.1.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOutgoingUrg	1.3.6.1.4.1.2.6.19.2.2.7.1.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOutgoingWinSeq	1.3.6.1.4.1.2.6.19.2.2.7.1.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnInBuffered	1.3.6.1.4.1.2.6.19.2.2.7.1.1.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnRcvNxt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.17	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnUsrRcvNxt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.18	TCP/IP	ibmTCPIPmvsMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibmMvsTcpConnIncomingPush	1.3.6.1.4.1.2.6.19.2.2.7.1.1.19	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnIncomingUrg	1.3.6.1.4.1.2.6.19.2.2.7.1.1.20	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnIncomingWinSeq	1.3.6.1.4.1.2.6.19.2.2.7.1.1.21	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnReXmt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.22	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnMaxSndWnd	1.3.6.1.4.1.2.6.19.2.2.7.1.1.23	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnReXmtCount	1.3.6.1.4.1.2.6.19.2.2.7.1.1.24	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnCongestionWnd	1.3.6.1.4.1.2.6.19.2.2.7.1.1.25	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSSThresh	1.3.6.1.4.1.2.6.19.2.2.7.1.1.26	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnRoundTripTime	1.3.6.1.4.1.2.6.19.2.2.7.1.1.27	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnRoundTripVariance	1.3.6.1.4.1.2.6.19.2.2.7.1.1.28	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnInitSndSeq	1.3.6.1.4.1.2.6.19.2.2.7.1.1.29	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnInitRcvSeq	1.3.6.1.4.1.2.6.19.2.2.7.1.1.30	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSendMSS	1.3.6.1.4.1.2.6.19.2.2.7.1.1.31	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSndWl1	1.3.6.1.4.1.2.6.19.2.2.7.1.1.32	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSndWl2	1.3.6.1.4.1.2.6.19.2.2.7.1.1.33	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSndWnd	1.3.6.1.4.1.2.6.19.2.2.7.1.1.34	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnRcvBufSize	1.3.6.1.4.1.2.6.19.2.2.7.1.1.36	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnResourceName	1.3.6.1.4.1.2.6.19.2.2.7.1.1.37	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSubtask	1.3.6.1.4.1.2.6.19.2.2.7.1.1.38	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnResourceId	1.3.6.1.4.1.2.6.19.2.2.7.1.1.39	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnSockOpt	1.3.6.1.4.1.2.6.19.2.2.7.1.1.40	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTcpTimer	1.3.6.1.4.1.2.6.19.2.2.7.1.1.41	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTcpSig	1.3.6.1.4.1.2.6.19.2.2.7.1.1.42	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTcpSel	1.3.6.1.4.1.2.6.19.2.2.7.1.1.43	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnRttSeq	1.3.6.1.4.1.2.6.19.2.2.7.1.1.44	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTcpDet	1.3.6.1.4.1.2.6.19.2.2.7.1.1.46	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTcpPol	1.3.6.1.4.1.2.6.19.2.2.7.1.1.47	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnTargetAppl	1.3.6.1.4.1.2.6.19.2.2.7.1.1.48	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnLuName	1.3.6.1.4.1.2.6.19.2.2.7.1.1.49	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnClientUserID	1.3.6.1.4.1.2.6.19.2.2.7.1.1.50	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnLogMode	1.3.6.1.4.1.2.6.19.2.2.7.1.1.51	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnProto	1.3.6.1.4.1.2.6.19.2.2.7.1.1.52	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnDupacks	1.3.6.1.4.1.2.6.19.2.2.7.1.1.53	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOptMaxSegmentSize	1.3.6.1.4.1.2.6.19.2.2.7.1.1.54	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnClusterConnFlag	1.3.6.1.4.1.2.6.19.2.2.7.1.1.55	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnInSegs	1.3.6.1.4.1.2.6.19.2.2.7.1.1.56	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsTcpConnOutSegs	1.3.6.1.4.1.2.6.19.2.2.7.1.1.57	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmTcpipMvsUdpTable	1.3.6.1.4.1.2.6.19.2.2.8.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmTcpipMvsUdpEntry	1.3.6.1.4.1.2.6.19.2.2.8.1.1	TCP/IP	ibmTCPIPmvsMIB	N/A
ibmMvsUdpLastAct	1.3.6.1.4.1.2.6.19.2.2.8.1.1.1	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpTos	1.3.6.1.4.1.2.6.19.2.2.8.1.1.2	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpIpOpts	1.3.6.1.4.1.2.6.19.2.2.8.1.1.3	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpDgramIn	1.3.6.1.4.1.2.6.19.2.2.8.1.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpBytesIn	1.3.6.1.4.1.2.6.19.2.2.8.1.1.5	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpDgramOut	1.3.6.1.4.1.2.6.19.2.2.8.1.1.6	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpBytesOut	1.3.6.1.4.1.2.6.19.2.2.8.1.1.7	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpResourceName	1.3.6.1.4.1.2.6.19.2.2.8.1.1.8	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpSubtask	1.3.6.1.4.1.2.6.19.2.2.8.1.1.9	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpResourceId	1.3.6.1.4.1.2.6.19.2.2.8.1.1.10	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpSockOpt	1.3.6.1.4.1.2.6.19.2.2.8.1.1.11	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpSendLim	1.3.6.1.4.1.2.6.19.2.2.8.1.1.12	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpRecvLim	1.3.6.1.4.1.2.6.19.2.2.8.1.1.13	TCP/IP	ibmTCPIPmvsMIB	R/O

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
ibmMvsUdpEntryState	1.3.6.1.4.1.2.6.19.2.2.8.1.1.14	TCP/IP	ibmTCPIPmvsMIB	R/W
ibmMvsUdpMcastTTL	1.3.6.1.4.1.2.6.19.2.2.8.1.1.15	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpMcastLoopback	1.3.6.1.4.1.2.6.19.2.2.8.1.1.16	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpMcastLinkAddr	1.3.6.1.4.1.2.6.19.2.2.8.1.1.17	TCP/IP	ibmTCPIPmvsMIB	R/O
ibmMvsUdpMcastRecvLinkAddr	1.3.6.1.4.1.2.6.19.2.2.8.2.1.4	TCP/IP	ibmTCPIPmvsMIB	R/O
snmpSetSerialNo	1.3.6.1.6.3.1.1.6.1	Agent	RFC1907	R/O
snmpEngineID	1.3.6.1.6.3.10.2.1.1	Agent	RFC2271	R/O
snmpEngineBoots	1.3.6.1.6.3.10.2.1.2	Agent	RFC2271	R/O
snmpEngineTime	1.3.6.1.6.3.10.2.1.3	Agent	RFC2271	R/O
snmpEngineMaxMessageSize	1.3.6.1.6.3.10.2.1.4	Agent	RFC2271	R/O
snmpUnknownSecurityModels	1.3.6.1.6.3.11.2.1.1	Agent	RFC2272	R/O
snmpInvalidMsgs	1.3.6.1.6.3.11.2.1.2	Agent	RFC2272	R/O
snmpUnknownPDUHandlers	1.3.6.1.6.3.11.2.1.3	Agent	RFC2272	R/O
snmpTargetSpinLock	1.3.6.1.6.3.12.1.1	Agent	RFC2273	R/W
snmpTargetAddrTable	1.3.6.1.6.3.12.1.2	Agent	RFC2273	N/A
snmpTargetAddrEntry	1.3.6.1.6.3.12.1.2.1	Agent	RFC2273	N/A
snmpTargetAddrName	1.3.6.1.6.3.12.1.2.1.1	Agent	RFC2273	N/A
snmpTargetAddrTDomain	1.3.6.1.6.3.12.1.2.1.2	Agent	RFC2273	R/C
snmpTargetAddrTAddress	1.3.6.1.6.3.12.1.2.1.3	Agent	RFC2273	R/C
snmpTargetAddrTimeout	1.3.6.1.6.3.12.1.2.1.4	Agent	RFC2273	R/C
snmpTargetAddrRetryCount	1.3.6.1.6.3.12.1.2.1.5	Agent	RFC2273	R/C
snmpTargetAddrTagList	1.3.6.1.6.3.12.1.2.1.6	Agent	RFC2273	R/C
snmpTargetAddrParams	1.3.6.1.6.3.12.1.2.1.7	Agent	RFC2273	R/C
snmpTargetAddrStorageType	1.3.6.1.6.3.12.1.2.1.8	Agent	RFC2273	R/C
snmpTargetAddrRowStatus	1.3.6.1.6.3.12.1.2.1.9	Agent	RFC2273	R/C
snmpTargetParamsTable	1.3.6.1.6.3.12.1.3	Agent	RFC2273	N/A
snmpTargetParamsEntry	1.3.6.1.6.3.12.1.3.1	Agent	RFC2273	N/A
snmpTargetParamsName	1.3.6.1.6.3.12.1.3.1.1	Agent	RFC2273	N/A
snmpTargetParamsMPModel	1.3.6.1.6.3.12.1.3.1.2	Agent	RFC2273	R/C
snmpTargetParamsSecurityModel	1.3.6.1.6.3.12.1.3.1.3	Agent	RFC2273	R/C
snmpTargetParamsSecurityName	1.3.6.1.6.3.12.1.3.1.4	Agent	RFC2273	R/C
snmpTargetParamsSecurityLevel	1.3.6.1.6.3.12.1.3.1.5	Agent	RFC2273	R/C
snmpTargetParamsStorageType	1.3.6.1.6.3.12.1.3.1.6	Agent	RFC2273	R/C
snmpTargetParamsRowStatus	1.3.6.1.6.3.12.1.3.1.7	Agent	RFC2273	R/C
snmpUnavailableContexts	1.3.6.1.6.3.12.1.4	Agent	RFC2273	R/O
snmpUnknownContexts	1.3.6.1.6.3.12.1.5	Agent	RFC2273	R/O
snmpNotifyTable	1.3.6.1.6.3.13.1.1	Agent	RFC2273	N/A
snmpNotifyEntry	1.3.6.1.6.3.13.1.1.1	Agent	RFC2273	N/A
snmpNotifyName	1.3.6.1.6.3.13.1.1.1.1	Agent	RFC2273	N/A
snmpNotifyTag	1.3.6.1.6.3.13.1.1.1.2	Agent	RFC2273	R/C
snmpNotifyType	1.3.6.1.6.3.13.1.1.1.3	Agent	RFC2273	R/C
snmpNotifyStorageType	1.3.6.1.6.3.13.1.1.1.4	Agent	RFC2273	R/C
snmpNotifyRowStatus	1.3.6.1.6.3.13.1.1.1.5	Agent	RFC2273	R/C
usmStatsUnsupportedSecLevels	1.3.6.1.6.3.15.1.1.1	Agent	RFC2274	R/O
usmStatsNotInTimeWindows	1.3.6.1.6.3.15.1.1.2	Agent	RFC2274	R/O
usmStatsUnknownUserNames	1.3.6.1.6.3.15.1.1.3	Agent	RFC2274	R/O
usmStatsUnknownEngineIDs	1.3.6.1.6.3.15.1.1.4	Agent	RFC2274	R/O
usmStatsWrongDigests	1.3.6.1.6.3.15.1.1.5	Agent	RFC2274	R/O
usmStatsDecryptionErrors	1.3.6.1.6.3.15.1.1.6	Agent	RFC2274	R/O
usmUserSpinLock	1.3.6.1.6.3.15.1.2.1	Agent	RFC2274	R/W
usmUserTable	1.3.6.1.6.3.15.1.2.2	Agent	RFC2274	N/A
usmUserEntry	1.3.6.1.6.3.15.1.2.2.1	Agent	RFC2274	N/A

Table 39. MIB Objects (continued)

Object Descriptor	Object Identifier	Supported by	Defined by	Access Allowed
usmUserEngineID	1.3.6.1.6.3.15.1.2.2.1.1	Agent	RFC2274	N/A
usmUserName	1.3.6.1.6.3.15.1.2.2.1.2	Agent	RFC2274	N/A
usmUserSecurityName	1.3.6.1.6.3.15.1.2.2.1.3	Agent	RFC2274	R/O
usmUserCloneFrom	1.3.6.1.6.3.15.1.2.2.1.4	Agent	RFC2274	R/C
usmUserAuthProtocol	1.3.6.1.6.3.15.1.2.2.1.5	Agent	RFC2274	R/C
usmUserAuthKeyChange	1.3.6.1.6.3.15.1.2.2.1.6	Agent	RFC2274	R/C
usmUserOwnAuthKeyChange	1.3.6.1.6.3.15.1.2.2.1.7	Agent	RFC2274	R/C
usmUserPrivProtocol	1.3.6.1.6.3.15.1.2.2.1.8	Agent	RFC2274	R/C
usmUserPrivKeyChange	1.3.6.1.6.3.15.1.2.2.1.9	Agent	RFC2274	R/C
usmUserOwnPrivKeyChange	1.3.6.1.6.3.15.1.2.2.1.10	Agent	RFC2274	R/C
usmUserPublic	1.3.6.1.6.3.15.1.2.2.1.11	Agent	RFC2274	R/C
usmUserStorageType	1.3.6.1.6.3.15.1.2.2.1.12	Agent	RFC2274	R/C
usmUserStatus	1.3.6.1.6.3.15.1.2.2.1.13	Agent	RFC2274	R/C
vacmContextTable	1.3.6.1.6.3.16.1.1	Agent	RFC2275	N/A
vacmContextEntry	1.3.6.1.6.3.16.1.1.1	Agent	RFC2275	N/A
vacmContextName	1.3.6.1.6.3.16.1.1.1.1	Agent	RFC2275	R/O
vacmSecurityToGroupTable	1.3.6.1.6.3.16.1.2	Agent	RFC2275	N/A
vacmSecurityToGroupEntry	1.3.6.1.6.3.16.1.2.1	Agent	RFC2275	N/A
vacmSecurityModel	1.3.6.1.6.3.16.1.2.1.1	Agent	RFC2275	N/A
vacmSecurityName	1.3.6.1.6.3.16.1.2.1.2	Agent	RFC2275	N/A
vacmGroupName	1.3.6.1.6.3.16.1.2.1.3	Agent	RFC2275	R/C
vacmSecurityToGroupStorageType	1.3.6.1.6.3.16.1.2.1.4	Agent	RFC2275	R/C
vacmSecurityToGroupStatus	1.3.6.1.6.3.16.1.2.1.5	Agent	RFC2275	R/C
vacmAccessTable	1.3.6.1.6.3.16.1.4	Agent	RFC2275	N/A
vacmAccessEntry	1.3.6.1.6.3.16.1.4.1	Agent	RFC2275	N/A
vacmAccessContextPrefix	1.3.6.1.6.3.16.1.4.1.1	Agent	RFC2275	N/A
vacmAccessSecurityModel	1.3.6.1.6.3.16.1.4.1.2	Agent	RFC2275	N/A
vacmAccessSecurityLevel	1.3.6.1.6.3.16.1.4.1.3	Agent	RFC2275	N/A
vacmAccessContextMatch	1.3.6.1.6.3.16.1.4.1.4	Agent	RFC2275	R/C
vacmAccessReadViewName	1.3.6.1.6.3.16.1.4.1.5	Agent	RFC2275	R/C
vacmAccessWriteViewName	1.3.6.1.6.3.16.1.4.1.6	Agent	RFC2275	R/C
vacmAccessNotifyViewName	1.3.6.1.6.3.16.1.4.1.7	Agent	RFC2275	R/C
vacmAccessStorageType	1.3.6.1.6.3.16.1.4.1.8	Agent	RFC2275	R/C
vacmAccessStatus	1.3.6.1.6.3.16.1.4.1.9	Agent	RFC2275	R/C
vacmViewSpinLock	1.3.6.1.6.3.16.1.5.1	Agent	RFC2275	R/W
vacmViewTreeFamilyTable	1.3.6.1.6.3.16.1.5.2	Agent	RFC2275	N/A
vacmViewTreeFamilyEntry	1.3.6.1.6.3.16.1.5.2.1	Agent	RFC2275	N/A
vacmViewTreeFamilyViewName	1.3.6.1.6.3.16.1.5.2.1.1	Agent	RFC2275	N/A
vacmViewTreeFamilySubtree	1.3.6.1.6.3.16.1.5.2.1.2	Agent	RFC2275	N/A
vacmViewTreeFamilyMask	1.3.6.1.6.3.16.1.5.2.1.3	Agent	RFC2275	R/C
vacmViewTreeFamilyType	1.3.6.1.6.3.16.1.5.2.1.4	Agent	RFC2275	R/C
vacmViewTreeFamilyStorageType	1.3.6.1.6.3.16.1.5.2.1.5	Agent	RFC2275	R/C
vacmViewTreeFamilyStatus	1.3.6.1.6.3.16.1.5.2.1.6	Agent	RFC2275	R/C

Appendix E. IBM 3172 Attribute Index

This appendix shows the 3172 attributes and their corresponding MIB variables.

Table 40. MIB Variable Cross-Reference Table

3172 Attribute	MIB Variable
01	= ibm3172Descr
02	= ibm3172Contact
03	= ibm3172Location
04	= ibm3172ifNumber
10	= ibm3172ifTrapEnable
11	= ifDescr
12	= ifType
13	= ifPhysAddress
14	= ifOperStatus
20	= ibm3172ifChanCounters
21	= ibm3172ifInChanOctets
22	= ibm3172ifOutChanOctets
23	= ibm3172ifInChanBlocks
24	= ibm3172ifOutChanBlocks
30	= ibm3172ifLANCounters
31	= ibm3172ifInLANOctets
32	= ibm3172ifOutLANOctets
33	= ibm3172ifInLANFrames
34	= ibm3172ifOutLANFrames
35	= ibm3172ifInLANErrors
36	= ibm3172ifOutLANErrors
37	= ibm3172ifInLANDiscards
38	= ibm3172ifOutLANDiscards
40	= ibm3172ifBlkCounters
41	= ibm3172ifBlkRcvOctets
42	= ibm3172ifBlkXmitOctets
43	= ibm3172ifBlkRcvFrames
44	= ibm3172ifBlkXmitBlocks
45	= ibm3172ifInBlkErrors
46	= ibm3172ifInBlkDiscards
50	= ibm3172ifDblkCounters
51	= ibm3172ifDblkRcvOctets
52	= ibm3172ifDblkXmitOctets
53	= ibm3172ifDblkRcvBlocks
54	= ibm3172ifDblkXmitFrames
55	= ibm3172ifOutDblkErrors

Table 40. MIB Variable Cross-Reference Table (continued)

3172 Attribute	MIB Variable
56	= ibm3172ifOutDblkDiscards

Appendix F. SNMP Trap Types

This appendix lists the generic and enterprise-specific trap types that can be received by SNMP.

SNMP Generic Trap Types

Table 41 lists the generic trap types that can be received by SNMP.

Table 41. Generic Trap Types

Value	Type	Description
0	coldStart	A coldStart trap signifies that the sending protocol entity is reinitializing itself so that the agent's configuration or the protocol entity implementation can be altered.
1	warmStart	A warmStart trap signifies that the sending protocol entity is reinitializing itself so that neither the agent configuration nor the protocol entity implementation can be altered.
2	linkDown	A linkDown trap signifies that the sending protocol entity recognizes a failure in one of the communication links represented in the agent's configuration. A Trap-PDU of type linkDown contains, as the first element of its variable-bindings, the name and value of the ifIndex instance for the affected interface.
3	linkUp	A linkUp trap signifies that the sending protocol entity recognizes that one of the communication links represented in the agent's configuration has come up. A Trap-PDU of type linkUp contains, as the first element of its variable-bindings, the name and value of the ifIndex instance for the affected interface.
4	authenticationFailure	An authenticationFailure trap signifies that the sending protocol entity is the addressee of a protocol message that is not properly authenticated.
5	egpNeighborLoss	An egpNeighborLoss trap signifies that an EGP neighbor for whom the sending protocol entity was an EGP peer has been marked down and the peer relationship no longer exists. The Trap-PDU of the egpNeighborLoss contains, as the first element of its variable-bindings, the name and value of the egpNeighAddr instance for the affected neighbor.

Table 41. Generic Trap Types (continued)

Value	Type	Description
6	enterpriseSpecific	An enterpriseSpecific trap signifies that the sending protocol entity recognizes that some enterprise-specific event has occurred. The specific-trap field identifies the particular trap that occurred.

SNMP Enterprise-Specific Trap Types

Table 42 lists the enterprise-specific trap types generated by subagents shipped with CS for OS/390. All enterprise-specific traps are generated with a trap value of 6.

Table 42. MVS Enterprise Trap Types

Subagent	Type	Description
TCP/IP	ibmMvsAtmOsasfAtmPvcDelete	This trap is generated when OSA/SF sends a asyn notification to the TCP/IP DPI Subagent that a PVC was deleted for a given OSA Port. This notification contains the corresponding ibmMvsAtmOsasfPortName instance. Representation of this contains the port's (aal5 layer interface) 'ifIndex.pvcNameOctetCount. pvcNameInASCIINvt'.
TCP/IP	ibmMvsAtmOsasfAtmPvcCreate	This trap is generated when OSA/SF sends a asyn notification to the TCP/IP DPI Subagent that a PVC was created for a given OSA Port. This notification contains the corresponding ibmMvsAtmOsasfPortName instance. Representation of this contains the port's (aal5 layer interface) 'ifIndex.pvcNameOctetCount. pvcNameInASCIINvt'.
SLA	slapmMonitoredEventNotAchieved	This notification is generated when a monitored event is not achieved with respect to threshold. This applies only towards monitoring a policy traffic profile as an aggregate via an associating slapmPolicyStatsEntry. The value of slapmPolicyMonitorControl can be examined to determine what is being monitored. The first slapmPolicyMonitorStatus value supples the current monitor status while the 2nd value supplies the previous status.
SLA	slapmMonitoredEventOkay	This notification is generated when a monitored event has improved to an acceptable level. This applies only towards monitoring a policy traffic profile as an aggregate by way of an associating slapmPolicyStatsEntry. The value of slapmPolicyMonitorControl can be examined to determine what is being monitored. The first slapmPolicyMonitorStatus value supplies the current monitor status while the 2nd value supplies the previous status.

Table 42. MVS Enterprise Trap Types (continued)

Subagent	Type	Description
SLA	slapmPolicyProfileDeleted	A slapmPolicyProfileDeleted notification is sent when a slapmPolicyStatsEntry is deleted if the value of slapmPolicyTrapEnable is enabled (1).
SLA	slapmPolicyMonitorDeleted	A slapmPolicyMonitorDeleted notification is sent when a slapmPolicyMonitor is deleted if the value of slapmPolicyTrapEnable is enabled (1).
SLA	slapmSubcomponentMonitored EventNotAchieved	This notification is generated when a monitored value does not achieve a threshold specification. This applies only towards monitoring the individual components of a policy traffic profile. The value of the corresponding slapmPolicyMonitorControl can be examined to determine what is being monitored. The first slapmSubcomponentMonitorStatus value supplies the previous status.
SLA	slapmSubcomponentMonitored EventOkay	This notification is generated when a monitored value has reached an acceptable level.

Appendix G. Related Protocol Specifications (RFCs)

This appendix lists the related protocol specifications for TCP/IP. The internet protocol suite is still evolving through requests for comments (RFC). New protocols are being designed and implemented by researchers and are brought to the attention of the internet community in the form of RFCs. Some of these protocols are so useful that they become recommended protocols. That is, all future implementations for TCP/IP are recommended to implement these particular functions or protocols. These become the *de facto* standards, on which the TCP/IP protocol suite is built.

Many features of TCP/IP for MVS are based on the following RFCs:

RFC	Title and Author
768	*User Datagram Protocol* J.B. Postel
791	*Internet Protocol* J.B. Postel
792	*Internet Control Message Protocol* J.B. Postel
793	*Transmission Control Protocol* J.B. Postel
821	*Simple Mail Transfer Protocol* J.B. Postel
822	*Standard for the Format of ARPA Internet Text Messages* D. Crocker
823	*DARPA Internet Gateway* R.M. Hinden, A. Sheltzer
826	*Ethernet Address Resolution Protocol or Converting Network Protocol Addresses to 48.Bit Ethernet Address for Transmission on Ethernet Hardware* D.C. Plummer
854	*Telnet Protocol Specification* J.B. Postel, J.K. Reynolds
855	*Telnet Option Specification* J.B. Postel, J.K. Reynolds
856	*Telnet Binary Transmission* J.B. Postel, J.K. Reynolds
857	*Telnet Echo Option* J.B. Postel, J.K. Reynolds
858	*Telnet Suppress Go Ahead Option* J.B. Postel, J.K. Reynolds
859	*Telnet Status Option* J.B. Postel, J.K. Reynolds
860	*Telnet Timing Mark Option* J.B. Postel, J.K. Reynolds
861	*Telnet Extended Options —List Option* J.B. Postel, J.K. Reynolds
862	*Echo Protocol* J.B. Postel
863	*Discard Protocol* J.B. Postel
864	*Character Generator Protocol* J.B. Postel
877	*Standard for the Transmission of IP Datagrams over Public Data Networks* J.T. Korb
885	*Telnet End of Record Option* J.B. Postel
903	*Reverse Address Resolution Protocol* R. Finlayson, T. Mann, J.C. Mogul, M. Theimer
904	*Exterior Gateway Protocol Formal Specification* D.L. Mills
919	*Broadcasting Internet Datagrams* J.C. Mogul
922	*Broadcasting Internet Datagrams in the Presence of Subnets* J.C. Mogul

950 *Internet Standard Subnetting Procedure* J.C. Mogul, J.B. Postel

952 *DoD Internet Host Table Specification* K. Harrenstien, M.K. Stahl, E.J. Feinler

959 *File Transfer Protocol* J.B. Postel, J.K. Reynolds

974 *Mail Routing and the Domain Name System* C. Partridge

1006 *ISO Transport Service on tiop of the TCP Version 3* M.T.Rose, D.E. Cass

1009 *Requirements for Internet Gateways* R.T. Braden, J.B. Postel

1013 *X Window System Protocol, Version 11: Alpha Update* R.W. Scheifler

1014 *XDR: External Data Representation Standard* Sun Microsystems Incorporated

1027 *Using ARP to Implement Transparent Subnet Gateways* S. Carl-Mitchell, J.S. Quarterman

1032 *Domain Administrators Guide* M.K. Stahl

1033 *Domain Administrators Operations Guide* M. Lottor

1034 *Domain Names—Concepts and Facilities* P.V. Mockapetris

1035 *Domain Names—Implementation and Specification* P.V. Mockapetris

1042 *Standard for the Transmission of IP Datagrams over IEEE 802 Networks* J.B. Postel, J.K. Reynolds

1044 *Internet Protocol on Network System's HYPERchannel: Protocol Specification* K. Hardwick, J. Lekashman

1055 *Nonstandard for Transmission of IP Datagrams over Serial Lines: SLIP* J.L. Romkey

1057 *RPC: Remote Procedure Call Protocol Version 2 Specification* Sun Microsystems Incorporated

1058 *Routing Information Protocol* C.L. Hedrick

1073 *Telnet Window Size Option* D. Waitzman

1079 *Telnet Terminal Speed Option* C.L. Hedrick

1091 *Telnet Terminal-Type Option* J. VanBokkelen

1094 *NFS: Network File System Protocol Specification* Sun Microsystems Incorporated

1096 *Telnet X Display Location Option* G. Marcy

1112 *Host Extensions for IP Multicasting* S. Deering

1118 *Hitchhikers Guide to the Internet* E. Krol

1122 *Requirements for Internet Hosts—Communication Layers* R.T. Braden

1123 *Requirements for Internet Hosts—Application and Support* R.T. Braden

1155 *Structure and Identification of Management Information for TCP/IP-Based Internets* M.T. Rose, K. McCloghrie

1156 *Management Information Base for Network Management of TCP/IP-Based Internets* K. McCloghrie, M.T. Rose

1157 *Simple Network Management Protocol (SNMP)* J.D. Case, M. Fedor, M.L. Schoffstall, C. Davin

1179 *Line Printer Daemon Protocol* The Wollongong Group, L. McLaughlin III

1180 *TCP/IP Tutorial* T.J. Socolofsky, C.J. Kale

1183 *New DNS RR Definitions* C.F. Everhart, L.A. Mamakos, R. Ullmann, P.V. Mockapetris, (Updates RFC 1034, RFC 1035)

1184 *Telnet Linemode Option* D. Borman

1187 *Bulk Table Retrieval with the SNMP* M.T. Rose, K. McCloghrie, J.R. Davin

1188 *Proposed Standard for the Transmission of IP Datagrams over FDDI Networks* D. Katz

1191 *Path MTU Discovery* J. Mogul, S. Deering

1198 *FYI on the X Window System* R.W. Scheifler

1207 *FYI on Questions and Answers: Answers to Commonly Asked "Experienced Internet User" Questions* G.S. Malkin, A.N. Marine, J.K. Reynolds

1208 *Glossary of Networking Terms* O.J. Jacobsen, D.C. Lynch

1213 *Management Information Base for Network Management of TCP/IP-Based Internets: MIB-II* K. McCloghrie, M.T. Rose

1215 *Convention for Defining Traps for Use with the SNMP* M.T. Rose

1228 *SNMP-DPI Simple Network Management Protocol Distributed Program Interface* G.C. Carpenter, B. Wijnen

1229 *Extensions to the Generic-Interface MIB* K. McCloghrie

1230 *IEEE 802.4 Token Bus MIB IEEE 802 4 Token Bus MIB* K. McCloghrie, R. Fox

1231 *IEEE 802.5 Token Ring MIB IEEE 802.5 Token Ring MIB* K. McCloghrie, R. Fox, E. Decker

1267 *A Border Gateway Protocol 3 (BGP-3)* K. Lougheed, Y. Rekhter

1268 *Application of the Border Gateway Protocol in the Internet* Y. Rekhter, P. Gross

1269 *Definitions of Managed Objects for the Border Gateway Protocol (Version 3)* S. Willis, J. Burruss

1270 *SNMP Communications Services* F. Kastenholz, ed.

1323 *TCP Extensions for High Performance* V. Jacobson, R. Braden, D. Borman

1325 *FYI on Questions and Answers: Answers to Commonly Asked "New Internet User" Questions* G.S. Malkin, A.N. Marine

1340 *Assigned Numbers* J.K. Reynolds, J.B. Postel

1348 *DNS NSAP RRs* B. Manning

1350 *TFTP Protocol* K.R. Sollins

1351 *SNMP Administrative Model* J. Davin, J. Galvin, K. McCloghrie

1352 *SNMP Security Protocols* J. Galvin, K. McCloghrie, J. Davin

1353 *Definitions of Managed Objects for Administration of SNMP Parties* K. McCloghrie, J. Davin, J. Galvin

1354 *IP Forwarding Table MIB* F. Baker

1356 *Multiprotocol Interconnect on X.25 and ISDN in the Packet Mode* A. Malis, D. Robinson, R. Ullmann

1372 *Telnet Remote Flow Control Option* D. Borman, C. L. Hedrick

1374 *IP and ARP on HIPPI* J. Renwick, A. Nicholson

1381 *SNMP MIB Extension for X.25 LAPB* D. Throop, F. Baker

1382 *SNMP MIB Extension for the X.25 Packet Layer* D. Throop

1387 *RIP Version 2 Protocol Analysis* G. Malkin

1388 *RIP Version 2 — Carrying Additional Information* G. Malkin

1389 *RIP Version 2 MIB Extension* G. Malkin

1390 *Transmission of IP and ARP over FDDI Networks* D. Katz

1393 *Traceroute Using an IP Option* G. Malkin

1397 *Default Route Advertisement In BGP2 And BGP3 Versions of the Border Gateway Protocol* D. Haskin

1398 *Definitions of Managed Objects for the Ethernet-Like Interface Types* F. Kastenholz

1540 *IAB Official Protocol Standards* J.B. Postel

1571 *Telnet Environment Option Interoperability Issues* D. Borman

1572 *Telnet Environment Option* S. Alexander

1577 *Classical IP and ARP over ATM* M. Laubach

1583 *OSPF Version 2* J. Moy

1592 *Simple Network Management Protocol Distributed Protocol Interface Version 2.0* B. Wijnen, G. Carpenter, K. Curran, A. Sehgal, G. Waters

1594 *FYI on Questions and Answers: Answers to Commonly Asked "New Internet User" Questions* A.N. Marine, J. Reynolds, G.S. Malkin

1695 *Definitions of Managed Objects for ATM Management Version 8.0 Using SMIv2* M. Ahmed, K. Tesink

1723 *RIP Version 2 — Carrying Additional Information* G. Malkin

1850 *OSPF Version 2 Management Information Base* F. Baker, R. Coltun

1901 *Introduction to Community-Based SNMPv2* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1902 *Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1903 *Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1904 *Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1905 *Protocols Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1906 *Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1907 *Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1908 *Coexistence between Version 1 and Version 2 of the Internet-Standard Network Management Framework* J. Case, K. McCloghrie, M. Rose, S. Waldbusser

1939 *Post Office Protocol-Version 3* J. Myers, M. Rose

2011 *SNMPv2 Management Information Base for the Internet Protocol Using SMIv2* K. McCloghrie

2012 *SNMPv2 Management Information Base for the Transmission Control Protocol Using SMIv2* K. McCloghrie

2013 *SNMPv2 Management Information Base for the User Datagram Protocol Using SMIv2* K. McCloghrie

2205 *Resource ReSerVation Protocol (RSVP) Version 1*R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin

2233 *The Interfaces Group MIB Using SMIv2* K. McCloghrie, F. Kastenholz

2271 *An Architecture for Describing SNMP Management Frameworks* D. Harrington, R. Presuhn, B. Wijnen

2272 *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)* J. Case, D. Harrington, R. Presuhn, B. Wijnen

2273 *SNMP Applications* David B. Levi, Paul Meyer, Bob Stewart

2274 *User-Based Security Model (USM) for Version 3 of the Simple Network Management Protocol (SNMPv3)* U. Blumenthal, B. Wijnen

2275 *View-Based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)* B. Wijnen, R. Presuhn, K. McCloghrie

2320 *Definitions of Managed Objects for Classical IP and ARP over ATM Using SMIv2* M. Greene, J. Luciani, K. White, T. Kuo

These documents can be obtained from:

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

Many RFCs are available online. Hard copies of all RFCs are available from the NIC, either individually or by subscription. Online copies are available using FTP from the NIC at the following Web address: http://www.rfc-editor.org/rfc.html

Use FTP to download the files, using the following format:

```
RFC:RFC-INDEX.TXT
RFC:RFCnnnn.TXT
RFC:RFCnnnn.PS
```

where:

nnnn Is the RFC number.
TXT Is the text format.
PS Is the PostScript format.

You can also request RFCs through electronic mail, from the automated NIC mail server, by sending a message to `service@nic.ddn.mil` with a subject line of `RFC nnnn` for text versions or a subject line of `RFC nnnn.PS` for PostScript versions. To request a copy of the RFC index, send a message with a subject line of `RFC INDEX`.

For more information, contact `nic@nic.ddn.mil`.

Appendix H. TELNET Extensions

This appendix describes the Telnet 3270 DBCS Transform special operations. Sections are included to describe:

- The special character set cross reference
- Special key operation
- PF and PA key operation
- Sense codes for special key operation

Character Set Cross Reference Table

Table 43 describes the language, codefiles, and character sets for the Telnet 3270 DBCS Transform extended language support.

Table 43. TCP/IP Character Set Cross Reference

Keyword	Codefiles	Description	Character Set	Description	CCSID[1]	CPGID[2]
KANJI						
JIS78KJ	J8EETA J8EATE	JIS 8 Bit English SBCS	ASCII	JIS X0201 8 Bit	none	none
			EBCDIC	English SBCS	none	none
	J8KETA J8KATE	JIS 8 Bit Katakana SBCS	ASCII	JIS X0201 8 Bit	none	none
			EBCDIC	Katakana SBCS	none	none
	JIS78ETA JIS78ATE	JIS 1978 Kanji DBCS	ASCII	JIS X0208 1978	00955	00955
			EBCDIC	Japanese Host DBCS	00300	00300
JIS83KJ	J8EETA J8EATE	JIS 8 Bit English SBCSI	ASCIII	JIS X0201 8 Bit	none	none
			EBCDIC	English SBCS	none	none
J8KETA J8KATE	JIS 8 Bit Katakana SBCS	ASCII	JIS X0201 8 Bit	none	none	
		EBCDIC	Katakana SBCS	none	none	
	JIS83ETA JIS83ATE	JIS 1983 Kanji DBCS	ASCII	JIS X0208 1990	00952	00952
			EBCDIC	Japanese Host DBCS	00300	00300
SJISKANJI	A8EETA A8EATE	8 Bit English SBCS ASCII	ASCII	ISO/ANSI Multilingual	00819	00819
			EBCDIC	Japanese Latin Host SBCS	01027	01027
	A8KETA A8KATE	8 Bit Katakana SBCS	ASCII	ISO/ANSI Multilingual	00819	00819
			EBCDIC	Japanese Katakana Host SBCS	00290	00290
	JIS83ETA JIS83ATE	JIS 1983 Kanji DBCS	ASCII	JIS X0208 1990	00959	none
			EBCDIC	Japanese Host DBCS	00300	00300
DECKANJI	SJDCEETA SJDCEATE	DEC English SBCS	ASCII		none	none
			EBCDIC		none	none
	SJDCKETA SJDCKATE	DEC Katakana SBCS	ASCII		none	none
			EBCDIC		none	none

Table 43. TCP/IP Character Set Cross Reference (continued)

Keyword	Codefiles	Description	Character Set	Description	CCSID[1]	CPGID[2]
	JDECETA JDECATE	DEC Kanji DBCS	ASCII		none	none
			EBCDIC	Japanese Host DBCS	none	none
EUCKANJI	SJECEETA SJECEATE	8 Bit English SBCS	ASCII	JIS X0201 8 Bit	none	none
			EBCDIC	Japanese Latin Host SBCS	01027	01027
	SJECKETA SJECKATE	8 Bit Katakana SBCS	ASCII	JIS X0201 8 Bit	none	none
			EBCDIC	Japanese Katakana Host SBCS	00290	00290
	JEUCETA JEUCATE	Japanese Extended Unix DBCS	ASCII	JIS X0208 1990	00952	00952
			EBCDIC	Japanese Host DBCS	00300	00300
HANGEUL						
KSC5601	SKSHETA SKSHATE	Korean Standard Code KSC 5601 SBCS	ASCII	KSC 5601 SBCS	01088	01088
			EBCDIC	Korean Host SBCS	00833	00833
	KSHETA KSHATE	Korean Standard Code KSC 5601 DBCS	ASCII	KSC 5601 DBCS	00951	00951
			EBCDIC	Korean Host DBCS	00834	00834
HANGEUL	SHANETA SHANATE	Hangeul SBCS	ASCII	Korean PC SBCS	00891	00891
			EBCDIC	Korean Host SBCS	00833	00833
	HANETA HANATE	Hangeul DBCS	ASCII	Korean PC DBCS	00926	00926
			EBCDIC	Korean Host DBCS	00834	00834
TCHINESE						
TCHINESE	STCHETA STCHATE	Traditional Chinese SBCS	ASCII	T-Chinese PC SBCS	00904	00904
			EBCDIC	CECP Host SBCS	00037	00037
	TCHETA TCHATE	Traditional Chinese DBCS	ASCII I	T-Chinese PC DBCS	00927	00927
			EBCDIC	T-Chinese Host DBCS	00835	00835
BIG5	SBG5ETA SBG5ATE	Big-5 Chinese SBCS I	ASCII I	Big-5 Chinese PC SBCS	01114	01114
			EBCDIC	CECP Host SBCS	00037	00037
	BG5ETA BG5ATE	Big-5 Chinese DBCS	ASCII	Big-5 PC DBCS	00947	00947
			EBCDIC	T-Chinese Host DBCS	00835	00835
SCHINESE						
SCHINESE	SSCHETA SSCHATE	Simplified Chinese SBCS	ASCII	S-Chinese PC SBCS	01115	01115
			EBCDIC	S-Chinese Host SBCS	00836	00836
	SCHETA SCHATE	Simplified Chinese DBCS	ASCII	S-Chinese PC DBCS	01380	01380
			EBCDIC	S-Chinese Host DBCS	00837	00837

[1]Coded Character Set ID
[2]Code Page Group ID

Special Key Operation for TELNET

Table 44 contains information about the operation of special keys for the TELNET function when you are using a terminal that is not part of the 3270 family.

Use these key combinations if you are using TELNET and your terminal does not have the key that you want to use. For example, if you want to clear your screen and your terminal does not have an ERASE INPUT key, press CTRL and Y together to get the same result.

To use the TELNET extensions, TCP/IP must be configured using the DBCSTRANSFORM option. For more information, refer to the *OS/390 SecureWay Communications Server: IP Configuration*.

Table 44. Special Key Conversions

Function Name	Input Keys	Function Description
Duplicate	Ctrl + D	Press CTRL and D together. This combination enters the Duplicate control code in the screen buffer and a TAB is performed on the screen.
Field Mark	Ctrl + K	Press CTRL and K together. This combination enters the Field Mark control code in the screen buffer and displays it as a blank on the screen.
Redisplay	Ctrl + V	Press CTRL and V together. This combination redisplays the contents of the screen buffer on your screen.
Erase Input	Ctrl + Y	Press CTRL and Y together. This combination erases all characters in the unprotected fields on the screen and replaces them with blanks. The cursor is placed at the first unprotected character position on the screen.
Erase EOF	Ctrl + Y	Press CTRL and X together. This combination erases all characters in an unprotected field from the cursor position to the end of the field and replaces them with blanks. If the cursor is on a protected field, the screen is inhibited and no characters are erased.
Delete One Character.	Del	The DEL deletes the character at the cursor position, if the field is unprotected. The cursor does not move. All characters in the unprotected field to the right of the cursor are shifted one position to the left and blank characters are add ed at the end of the field.
Alphanumeric or Alphanumeric-Kana	Ctrl + B	Press CTRL and B together. This combination is a toggle switch that redisplays the screen by switching between Alphanumeric and Alphanumeric-Kana mode.

Table 44. Special Key Conversions (continued)

Function Name	Input Keys	Function Description
Field Forward Tab	Ctrl + F	Press CTRL and F together. This combination moves the cursor to the first character position in the next unprotected field. If the screen is unformatted or there are no unprotected fields on the screen, the cursor is placed in the first position on the screen.
Field Backward Tab	Ctrl + A F, or Ctrl + A Ctrl + F	Press CTRL and A together then press F, or Press CTRL and A together then press CTRL and F together. This combination moves the cursor to the first character position in the previous unprotected field. If the screen is unformatted or there a re no unprotected fields on the screen, the cursor is placed in the first position on the screen.
Home	CSI P, or ESC [P	Press CSI then press P, or Press ESC then press [then press P. This combination moves the cursor to the first character position in the first unprotected field on the screen. If the screen is unformatted or there are no unprotected fie lds on the screen, the cursor is placed in the first character position on the screen.
Move Cursor Up	CSI A, or ESC [A	Press CSI then press A, or press ESC then press [then press P. This combination moves the cursor up one line in the same column. If the cursor is on the first line on the screen, it moves to the last line on the screen.
Move Cursor Down	CSI B, or ESC [B	Press CSI then press B, or press ESC then press [, then press B. This combination moves the cursor down one line in the same column. If the cursor is on the last line on the screen, it moves to the first line on the screen.
Move Cursor Right	CSI C, or ESC [C	Press CSI then press C, or press ESC then press [, then press C. This combination moves the cursor one character to the right. If the cursor is in the last column in a line, it moves to the first position in the next line on the screen. If the cursor is in the last position on the screen, moves to the first position on the screen.
Move Cursor Left	CSI D, or ESC [D	Press CSI then press D, or press ESC then press [, then press D. This combination moves the cursor one character to the left. If the cursor is in the first column in a line, it moves to the last position in the previous line on the screen. If the cursor is in the first position on the screen, moves to the last position on the screen.

Table 44. Special Key Conversions (continued)

Function Name	Input Keys	Function Description
Backspace One Character	Ctrl + H	Press CTRL and H together. This combination deletes one character before the cursor position in an unprotected field. The cursor moves one position to the left and all characters in the field shift one position to the left. If the cursor is on a protected field the screen is inhibited.
Reset	Ctrl + R	Press CTRL and R together. This combination releases the screen inhibit condition. When the screen is inhibited, only the RESET and MASTER RESET key combinations remain active.
Master Reset	Ctrl + A M, or Ctrl + A Ctrl + M	Press CTRL and R together then press M, or Press CTRL and R together then press CTRL and M together This combination produces the same results that you get if you key in REDISPLAY followed by RESET.
Clear	Ctrl + L	Press CTRL and L together. This combination fills the screen with blanks and places the cursor at the first character position on the screen. MASTER RESET key combinations remain active.
Enter	Ctrl + M	Press CTRL and M together. This combination sends the data on the screen to the host system.
CSI stands for Control Sequence Indicator.		

Operation of PF and PA Keys

This section describes the PF and PA keys for TELNET operations when you are using a terminal that is not part of the 3270 family.

When you press a key combination, a code that represents the 3270 equivalent symbol is sent to the application with which you have established TELNET communication. The application that you are using controls how these codes are used.

PF Key	Input Keys
PF1	Press ESC then press 1
PF2	Press ESC then press 2
PF3	Press ESC then press 3
PF4	Press ESC then press 4
PF5	Press ESC then press 5
PF6	Press ESC then press 6
PF7	Press ESC then press 7

PF8	Press ESC then press 8
PF9	Press ESC then press 9
PF10	Press ESC then press 0
PF11	Press ESC then press .
PF12	Press ESC then press =
PF13	Press ESC ESC then press 1
PF14	Press ESC ESC then press 2
PF15	Press ESC ESC then press 3
PF16	Press ESC ESC then press 4
PF17	Press ESC ESC then press 5
PF18	Press ESC ESC then press 6
PF19	Press ESC ESC then press 7
PF20	Press ESC ESC then press 8
PF21	Press ESC ESC then press 9
PF22	Press ESC ESC then press 0
PF23	Press ESC ESC then press .
PF24	Press ESC ESC then press =
PA Key	**Input Keys**
PA1	Press Ctrl + P together then press 1
PA2	Press Ctrl + P together then press 2
PA3	Press Ctrl + P together then press 3

Sense Codes for Special Key Operation

Table 45 describes the sense codes that are returned with the error messages for special key operations when you are using TELNET.

For information about error messages, see *OS/390 SecureWay Communications Server: IP Messages*.

Table 45. Sense Codes

Sense Code	Problem Description
8001	The command data length is less than 0 bytes.
8002	There is not enough data in an ERASE/WRITE or an ERASE/WRITE ALTERNATE command.
8003	There is not enough data in A WRITE command.
8004	There is not enough data in A WRITE STRUCTURED FIELD command.
21001	WCC is not a character. The error occurred in a WRITE command.
21002	There is not enough data in a START FIELD subcommand. The error occurred in a WRITE command.
21003	There is not enough data in a START FIELD EXTENDED subcommand. The error occurred in a WRITE command.

Table 45. Sense Codes (continued)

Sense Code	Problem Description
21004	There is not enough data in a MODIFY FIELD subcommand. The error occurred in a WRITE command.
21005	There is not enough data in a SET BUFFER ADDRESS subcommand. The error occurred in a WRITE command.
21006	There is not enough data in a REPEAT TO ADDRESS subcommand. The error occurred in a WRITE command.
21007	There is not enough data in an ERASE UNPROTECTED TO ADDRESS subcommand. The error occurred in a WRITE command.
22001	The specified attribute is not a character. subcommand. The error occurred in a START FIELD subcommand.
22002	A current buffer addressing error occurred. The error occurred in a START FIELD subcommand.
23001	An addressing error for the current buffer occurred. The error occurred in a START FIELD EXTENDED subcommand.
23002	The specified attribute is not a character. The error occurred in a START FIELD EXTENDED subcommand.
23003	The specified attribute is not an acceptable attribute type. The error occurred in a START FIELD EXTENDED subcommand.
24001	A current buffer addressing error occurred. The error occurred in a MODIFY FIELD subcommand.
24002	The specified attribute is not a character. The error occurred in a MODIFY FIELD subcommand.
24003	The specified attribute is not an acceptable attribute type. The error occurred in a MODIFY FIELD subcommand.
25001	The specified address is incorrect. The error occurred in a SET BUFFER ADDRESS subcommand.
26001	A current buffer addressing error occurred. The error occurred in a PROGRAM TAB subcommand.
28001	A current buffer addressing error occurred. The error occurred in a REPEAT TO ADDRESS subcommand.
28002	The specified address is incorrect. The error occurred in a REPEAT TO ADDRESS subcommand.
29001	A current buffer addressing error occurred. The error occurred in an ERASE UNPROTECTED TO ADDRESS subcommand.
29002	The specified address is incorrect. The error occurred in an ERASE UNPROTECTED TO ADDRESS subcommand.
30001	A current buffer addressing error occurred. The error occurred in the Write Data Process.
36001	Another structured field appeared after the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command.
36002	The length is too long or too short in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command.
36003	A reserved character is incorrect in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command.
36004	The type is incorrect in the READ PARTITION field. The error occurred in a WRITE STRUCTURED FIELD command.

Table 45. Sense Codes (continued)

Sense Code	Problem Description
38001	The length is too long or too short in the ERASE/RESET field. The error occurred in a WRITE STRUCTURED FIELD command.
38002	The Partition ID is incorrect in the ERASE/RESET field. The error occurred in a WRITE STRUCTURED FIELD command.
39001	The length is too long or too short in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command.
39002	The Partition ID is incorrect in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command.
39003	The Reply Mode is incorrect in the SET REPLY mode. The error occurred in a WRITE STRUCTURED FIELD command.
40001	The 3270 outbound data stream contains a command that is not in this list. • WRITE • ERASE/WRITE • ERASE/WRITE ALTERNATE • ERASE ALL UNPROTECTED The error occurred in a WRITE STRUCTURED FIELD command.
50001	SO/SI is on longer a pair. SET REPLY mode. The error occurred in the GRFTOMAP Process.

Appendix I. ICMP Types and Codes

Table 46 lists the Internet Control Message Protocol (ICMP) types and codes from *TCP/IP Illustrated, Volume 1 The Protocols*, by W. Richard Stevens.

Table 46. ICMP Types and Codes

Type	Code	Description
0	0	echo reply
3		destination unreachable:
	0	network unreachable
	1	host unreachable
	2	protocol unreachable
	3	port unreachable
	4	fragmentation needed
	5	source route failed
	6	destination network unknown
	7	destination host unknown
	8	source host isolated
	9	destination network administratively prohibited
	10	destination host administratively prohibited
	11	network unreachable for TOS
	12	host unreachable for TOS
	13	communication administratively prohibited by filtering
	14	host precedence violation
	15	precedence cutoff in effect
4	0	source quench
5		redirect:
	0	redirect for network
	1	redirect for host
	2	redirect for type of service and network
	3	redirect for type of service and host
8	0	echo request
9	0	router advertisement
10	0	router solicitation
11		time exceeded:
	0	time-to-live equals 0 during transmit
	1	time-to-live equals 0 during reassembly
12		parameter problem:
	0	IP header bad
	1	required option missing
13	0	timestamp request
14	0	timestamp reply
15	0	information request

Table 46. ICMP Types and Codes (continued)

Type	Code	Description
16	0	information reply
17	0	address mask request
18	0	address mask reply

Appendix J. 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:

►► 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.

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.

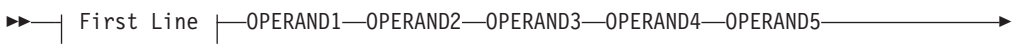
Parameters are classified as keywords or variables. Keywords are displayed in uppercase letters and can be entered in uppercase or lowercase. For example, a command name is a keyword.

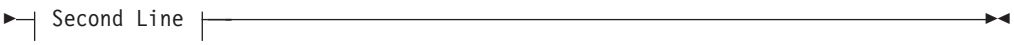
Variables are italicized, appear in lowercase letters, and represent names or values you supply. For example, a data set is a variable.

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.

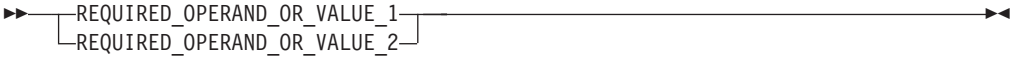




Required operands: Required operands and values appear on the main path line.

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.



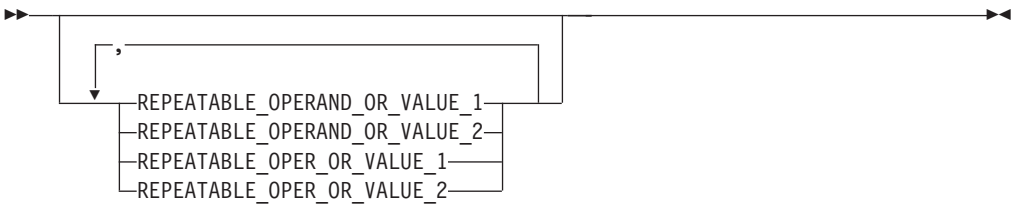
Optional values: Optional operands and values appear below the main path line.

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.

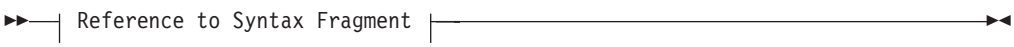
Nonalphanumeric characters: If a diagram shows a character that is not alphanumeric (such as parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code OPERAND=(001,0.001).

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 K. Information Apars

This appendix lists information apars for IP-related books.

Notes:

1. Information apars contain updates to previous editions of the manuals listed below. Books updated for V2R8 contain all the updates except those contained in the information apars that may be issued after V2R8 books went to press.
2. Information apars are predefined for CS for OS/390 V2R8 and may not contain updates.

IP Information Apars

Table 47 lists information apars for IP-related books.

Table 47. IP Information Apars

Title	CS for OS/390 2.8	CS for OS/390 2.7	CS for OS/390 2.6	CS for OS/390 2.5	TCP/IP 3.3	TCP/IP 3.2
High Speed Access Service User's Guide (GC31-8676)	ii11629	ii11566	ii11412	ii11181		
IP API Guide (SC31-8516)	ii11635	ii11558	ii11405	ii11144		
IP CICS Sockets Guide (SC31-8518)	ii11626	ii11559	ii11406	ii11145		ii10825 ii10330
IP Configuration (SC31-8513)	ii11620	ii11555 ii11637	ii11402 ii11619	ii11159	ii10633	
IP Diagnosis (SC31-8521)	ii11628	ii11565	ii11411	ii11160 ii11414	ii10637	
IP Messages Volume 1 (SC31-8517)	ii11630	ii11562	ii11408		Messages and Codes ii10635	
IP Messages Volume 2 (SC31-8570)	ii11631	ii11563	ii11409			
IP Messages Volume 3 (SC31-8674)	ii11632	ii11564	ii11410	ii11158		
IP Migration (SC31-8512)	ii11618	ii11554	ii11401			
IP Network Print Facility (SC31-8522)	ii11627	ii11561	ii11407	ii11150		

Table 47. IP Information Apars (continued)

Title	CS for OS/390 2.8	CS for OS/390 2.7	CS for OS/390 2.6	CS for OS/390 2.5	TCP/IP 3.3	TCP/IP 3.2
IP Programmer's Reference (SC31-8515)	ii11634	ii11557	ii11404		ii10636	
IP and SNA Codes (SC31-8571)		Was created by VTAM V2R6	ii11361	ii11146		
IP User's Guide (GC31-8514)	ii11625	ii11556	ii11403	ii11143	ii10634	

Appendix L. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. 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. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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, North Carolina 27709-2195
U.S.A

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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs

conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, photographs and color illustrations may not appear.

Some updates of this book, such as an update between OS/390 releases, might be available only in softcopy. You can obtain softcopy from the OS/390 Online Library Collection (SK2T-6700), the OS/390 PDF Library Collection (SK2T-6718), or the OS/390 Internet Library (http://www.ibm.com/s390/os390/). To order the latest hardcopy edition that is available, you might need to order a lower suffix (dash) level.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

ACF/VTAM
Advanced Peer-to-Peer Networking
AFP
AD/Cycle
AIX
AIX/ESA
AnyNet
APL2
APPN
AS/400
AT
BookManager
BookMaster
CBPDO
C/370
CICS
CICS/ESA
C/MVS
Common User Access
C Set ++
CT
CUA
DATABASE 2
DatagLANce
DB2
DFSMS
DFSMSdfp
DFSMShsm
DFSMS/MVS
Domino
DRDA
eNetwork
Enterprise Systems Architecture/370
ESA/390
ESCON
ES/3090
ES/9000
ES/9370
EtherStreamer
Extended Services
FAA
Micro Channel
MVS
MVS/DFP
MVS/ESA
MVS/SP
MVS/XA
MQ
Natural
NetView
Network Station
Nways
Notes
NTune
NTuneNCP
OfficeVision/MVS
OfficeVision/VM
Open Class
OpenEdition
OS/2
OS/390
Parallel Sysplex
Personal System/2
PR/SM
PROFS
PS/2
RACF
Resource Measurement Facility
RETAIN
RFM
RISC System/6000
RMF
RS/6000
S/370
S/390
SAA
SecureWay
Slate
SP
SP2
SQL/DS
System/360

FFST
FFST/2
FFST/MVS
First Failure Support Technology
GDDM
Hardware Configuration Definition
IBM
IBMLink
IMS
IMS/ESA
InfoPrint
Language Environment
LANStreamer
Library Reader
LPDA
MCS
System/370
System/390
SystemView
Tivoli
TURBOWAYS
UNIX System Services
Virtual Machine/Extended Architecture
VM/ESA
VM/XA
VSE/ESA
VTAM
WebSphere
XT
400
3090
3890

Lotus, Freelance, and Word Pro are trademarks of Lotus Development Corporation in the United States, or other countries, or both.

Tivoli and NetView are trademarks of Tivoli Systems Inc. in the United States, or other countries, or both.

The following terms are trademarks of other companies:

ATM is a trademark of Adobe Systems, Incorporated.

BSC is a trademark of BusiSoft Corporation.

CSA is a trademark of Canadian Standards Association.

DCE is a trademark of The Open Software Foundation.

HYPERchannel is a trademark of Network Systems Corporation.

UNIX is a registered trademark in the United States, other countries, or both and is licensed exclusively through X/Open Company Limited.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks of Intel Corporation in the United States, other countries, or both. For a complete list of Intel trademarks, see http://www.intel.com/tradmarx.htm.

Bibliography

SecureWay Communications Server for OS/390 Publications

Following are descriptions of the books in the SecureWay Communications Server for OS/390 library. The books are arranged in the following categories:

- Softcopy Information
- Planning
- Resource Definition, Configuration, and Tuning
- Operation
- Customization
- Writing Application Programs
- Diagnosis
- Messages and Codes
- APPC Application Suite.
- Multiprotocol Transport Networking (MPTN) Architecture publications

The complete set of unlicensed books in this section can be ordered using a single order number, SBOF-7011.

Updates to books are available on RETAIN. See "Appendix K. Information Apars" on page 615 for a list of the books and the INFOAPARS associated with them.

Some books are available in both hard- and soft-copy, or soft-copy only. The following abbreviations follow each order number:

HC/SC	Both hard- and soft-copy are available
SC	Only soft-copy is available

Related Publications

For information about OS/390 products, refer to *OS/390 Information Roadmap* (GC28-1727-03 [HC/SC]). The Roadmap describes what level of documents are supplied with each release of CS for OS/390, as well as describing each OS/390 publication.

Firewall

OS/390 Firewall Technologies Guide and Reference (SC24-5835-03 [HC/SC])

OSA-Express

S/390 Open Systems Adapter-Express Customer's Guide and Reference (SA22-7403-01 [HC/SC])

Softcopy Information

- *OS/390 Online Library Collection* (SK2T-6700).

  This collection contains softcopy unlicensed books for OS/390, Parallel Sysplex products, and S/390 application programs that run on OS/390. This collection is updated quarterly with any new or updated books that are available for the product libraries included in it.
- *OS/390 PDF Library Collection* (SK2T-6718).

  This collection contains the unlicensed books for OS/390 Version 2 Release 6 in Portable Document Format (PDF).
- *OS/390 Licensed Product Library* (LK2T-2499).

  This library contains unencrypted softcopy licensed books for OS/390 Version 2. If any of the books in this library are changed, it is updated quarterly. The OS/390 Licensed Product Library for Version 1 (LK2T-6702) is still available, but is no longer updated.
- *System Center Publication IBM S/390 Redbooks Collection* (SK2T-2177).

  This collection contains over 300 ITSO redbooks that apply to the S/390 platform and to host networking arranged into subject bookshelves.

Planning

OS/390 SecureWay Communications Server: SNA Migration (SC31-8622-03 [HC/SC]). This book is intended to help you plan for SNA, whether you are migrating from a previous version or installing SNA for the first time. This book also identifies the optional and required modifications needed to enable you to use the enhanced functions provided with SNA.

OS/390 SecureWay Communications Server: IP Migration (SC31-8512-03 [HC/SC]). This book is intended to help you plan for IP, whether you are migrating from a previous version or installing IP for the first time. This book also identifies the optional and required modifications needed to enable you to use the enhanced functions provided with IP.

Resource Definition, Configuration, and Tuning

OS/390 SecureWay Communications Server: IP Configuration (SC31-8513-03 [HC/SC]). This book is for people who want to configure, customize, administer, and maintain IP. Familiarity with MVS operating system, IP protocols, and IBM Time Sharing Option (TSO) is recommended.

OS/390 SecureWay Communications Server: SNA Network Implementation Guide (SC31-8563-03 [HC/SC]). This book presents the major concepts involved in implementing a SNA network. Use this book in conjunction with the *OS/390 SecureWay Communications Server: SNA Resource Definition Reference*.

OS/390 SecureWay Communications Server: SNA Resource Definition Reference (SC31-8565-03 [HC/SC]). This book describes each SNA definition statement, start option, and macroinstruction for user tables. It also describes NCP definition statements that affect SNA. The information includes:

- IBM-supplied default tables (logon mode and USS)
- Major node definitions
- User-defined tables and filters
- SNA start options.

Use this book in conjunction with the *OS/390 SecureWay Communications Server: SNA Network Implementation Guide*.

OS/390 eNetwork Communications Server: SNA Resource Definition Samples (SC31-8566-00 [HC/SC]). This book contains sample definitions to help you implement SNA functions in your networks, and includes sample major node definitions.

OS/390 eNetwork Communications Server: AnyNet SNA over TCP/IP (SC31-8578-00 [SC]). This guide provides information to help you install, configure, use, and diagnose SNA over TCP/IP.

OS/390 eNetwork Communications Server: AnyNet Sockets over SNA (SC31-8577-00 [SC]). This guide provides information to help you install, configure, use, and diagnose Sockets over SNA. It also provides information to help you prepare application programs to use sockets over SNA.

Operation

OS/390 SecureWay Communications Server: IP User's Guide (GC31-8514-03 [HC/SC]). This book is for people who want to use TCP/IP for data communication activities such as FTP and Telnet. Familiarity with MVS operating system and IBM Time Sharing Option (TSO) is recommended.

OS/390 SecureWay Communications Server: SNA Operation (SC31-8567-03 [HC/SC]). This book serves as a reference for programmers and operators requiring detailed information about specific operator commands.

OS/390 SecureWay Communications Server: Quick Reference (SX75-0121-03 [HC/SC]). This book contains essential information about SNA and IP operator commands.

OS/390 eNetwork Communications Server: High Speed Access Services User's Guide (GC31-8676-01 [SC]). This book is for end users and system administrators who want to use applications using a High Speed Access Services connection available in CS for OS/390.

Customization

OS/390 SecureWay Communications Server: SNA Customization (LY43-0110-01 [SC]). This book enables you to customize SNA, and includes:

- Communication network management (CNM) routing table
- Logon-interpret routine requirements
- Logon manager installation-wide exit routine for the CLU search exit
- TSO/SNA installation-wide exit routines
- SNA installation-wide exit routines

OS/390 eNetwork Communications Server: IP Network Print Facility (SC31-8522-00 [SC]). This book is for system programmers and network administrators who need to prepare their network to route SNA, JES2, or JES3 printer output to remote printers using TCP/IP.

Writing Application Programs

OS/390 SecureWay Communications Server: IP Application Programming Interface Guide (SC31-8516-03 [SC]). 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.

OS/390 SecureWay Communications Server: IP CICS Sockets Guide (SC31-8518-01 [SC]). 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.

OS/390 eNetwork Communications Server: IP IMS Sockets Guide (SC31-8519-00 [SC]). 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.

OS/390 SecureWay Communications Server: IP Programmer's Reference (SC31-8515-03 [SC]). 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. Familiarity with the MVS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.

OS/390 eNetwork Communications Server: SNA Programming (SC31-8573-01 [SC]). This book describes how to use SNA macroinstructions to send data to and receive data from (1) a terminal in either the same or a different domain, or (2) another application program in either the same or a different domain.

OS/390 eNetwork Communications Server: SNA Programmers LU 6.2 Guide (SC31-8581-00 [SC]). This book describes how to use the SNA LU 6.2 application programming interface for host application programs. This book applies to programs that use only LU 6.2 sessions or that use LU 6.2 sessions along with other session types. (Only LU 6.2 sessions are covered in this book.)

OS/390 eNetwork Communications Server: SNA Programmers LU 6.2 Reference (SC31-8568-00 [SC]). This book provides reference material for the SNA LU 6.2 programming interface for host application programs.

OS/390 eNetwork Communications Server: CSM Guide (SC31-8575-00 [SC]). This book describes how applications use the communications storage manager.

OS/390 eNetwork Communications Server: CMIP Services and Topology Agent Guide (SC31-8576-01 [SC]). This book describes the Common Management Information Protocol (CMIP) programming interface for application programmers to use in coding CMIP application programs. The book provides guide and reference information about CMIP services and the SNA topology agent.

Diagnosis

OS/390 SecureWay Communications Server: IP Diagnosis (SC31-8521-03 [HC/SC]). This book explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the TCP/IP product code. It explains how to gather information for and describe problems to the IBM Software Support Center.

OS/390 SecureWay Communications Server: SNA Diagnosis Volume 1: Techniques and Procedures (LY43-0079-03 [HC/SC]) and *OS/390 SecureWay Communications Server: SNA Diagnosis Volume 2: FFST Dumps and the VIT* (LY43-0080-02 [HC/SC]). These books help you identify a SNA problem, classify it, and collect information about it before you call the IBM Support Center. The information collected includes traces, dumps, and other problem documentation. Volume 1 includes information about the following:

- Command syntax for running traces and collecting and analyzing dumps
- Procedures for collecting documentation (SNA, TSO)
- VIT analysis tool
- Channel programs
- Flow diagrams
- Procedures for locating buffer pools
- CPCB operation codes
- Storage and control block ID codes
- Offset names and locations for SNA buffer pools.

Volume 2 includes information about the following:

- VIT entries
- SNA internal trace
- FFST dumps and probes

OS/390 SecureWay Communications Server: Data Areas Volume 1 (LY43-0111-03 [SC]). This book

describes SNA data areas and can be used to read a SNA dump. It is intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA.

OS/390 SecureWay Communications Server: Data Areas Volume 2 (LY43-0112-03 [SC]). This book describes SNA data areas and can be used to read a SNA dump. It is intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA.

Messages and Codes

OS/390 SecureWay Communications Server: SNA Messages (SC31-8569-03 [HC/SC]). This book describes the following types of messages and other associated information:

- Messages:
  - ELM messages for logon manager
  - IKT messages for TSO/SNA
  - IST messages for SNA network operators
  - ISU messages for sockets-over-SNA
  - IVT messages for the communications storage manager
  - IUT messages
  - USS messages
- Other information that displays in SNA messages:
  - Command and RU types in SNA messages
  - Node and ID types in SNA messages
- Supplemental message-related information

OS/390 SecureWay Communications Server: IP Messages Volume 1 (EZA) (SC31-8517-03 [HC/SC]). This volume contains TCP/IP messages beginning with EZA.

OS/390 SecureWay Communications Server: IP Messages Volume 2 (EZB) (SC31-8570-03 [HC/SC]). This volume contains TCP/IP messages beginning with EZB.

OS/390 SecureWay Communications Server: IP Messages Volume 3 (EZY-EZZ-SNM) (SC31-8674-03 [HC/SC]). This volume contains TCP/IP messages beginning with EZY, EZZ, and SNM.

OS/390 SecureWay Communications Server: IP and SNA Codes (SC31-8571-03 [HC/SC]). This book describes codes and other information that display in CS for OS/390 messages.

APPC Application Suite

OS/390 eNetwork Communications Server: APPC Application Suite User's Guide (GC31-8619-00 [SC]). This book documents the end-user interface (concepts, commands, and messages) for the AFTP, ANAME, and APING facilities of the APPC application suite. Although its primary audience is the end user, administrators and application programmers may also find it useful.

OS/390 eNetwork Communications Server: APPC Application Suite Administration (SC31-8620-00 [SC]). This book contains the information that administrators need to configure the APPC application suite and to manage the APING, ANAME, AFTP, and A3270 servers.

OS/390 eNetwork Communications Server: APPC Application Suite Programming (SC31-8621-00 [SC]). This book provides the information application programmers need to add the functions of the AFTP and ANAME APIs to their application programs.

Multiprotocol Transport Networking (MPTN) Architecture Publications

Following are selected publications for MPTN:

Networking Blueprint Executive Overview (GC31-7057)

Multiprotocol Transport Networking: Technical Overview (GC31-7073)

Multiprotocol Transport Networking: Formats (GC31-7074)

Redbooks

The following Redbooks may help you as you implement CS for OS/390.

- *OS/390 eNetwork Communication Server V2R7 TCP/IP Implementation Guide Volume 1: Configuration and Routing* (SG24–5227–01).

  This book provides examples of how to configure the base TCP/IP stack, routing daemons and the TELNET server. This book

also provides information about national language support (NLS), routing, OSPF, network interfaces, diagnosis, multicasting, OS/390 UNIX System Services and security in an OS/390 UNIX System Services environment.

- *OS/390 eNetwork Communication Server V2R7 TCP/IP Implementation Guide Volume 2: UNIX Applications* (SG24–5228–01).

  This book provides information about implementing applications that run in the OS/390 UNIX environment, such as FTP, SNMP, BIND-based name server, DHCP, and SENDMAIL. This book also provides configuration samples and describes the implementation process.

- *OS/390 eNetwork Communication Server V2R7 TCP/IP Implementation Guide Volume 3: MVS Applications* (SG24–5229–01).

  This book provides information about TCP/IP applications that run in a legacy MVS environment, including CICS/IMS Sockets, and printing (NPF, LPR, and LPD.)

- *TCP/IP in a Sysplex* (SG24–5235–01).

  The main goals of a Parallel Sysplex are high availability and high performance. This book demonstrates how these goals can be achieved in the particular environment of SecureWay Communications Server for OS/390 and its TCP/IP applications. This book describes the WLM/DNS functions, the Network Dispatcher and the Dynamic VIPA.

- *SNA and TCP/IP Integration* (SG24–5291–00).

  This book provides information about integrating current SNA network with future TCP/IP and Web-based communication requirements. This book concentrates on routing techniques.

- *SNA in a Parallel Sysplex Environment* (SG24–2113–01).

  This book provides information about implementing a VTAM-based network on a Parallel Sysplex.

- *Subarea to APPN Migration : VTAM and APPN Implementation* (SG24–4656–01).

  This book is the first of two volumes. This book provides information about the migration of a subarea network to an APPN network. Some knowledge of SNA subarea networks and familiarity with the functions, terms and data flows of APPN networks is assumed.

- *Subarea to APPN Migration : HPR and DLUR Implementation* (SG24–5204–00).

  This book is the second of two volumes. This book provides information about the coverage of a network using HPR, DLUR and APPN/HPR routers. Some knowledge of SNA subarea networks and familiarity with the functions, terms and data flows of APPN networks is assumed.

Bibliography

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Readers' Comments — We'd Like to Hear from You

OS/390 SecureWay Communications Server
IP User's Guide
Version 2 Release 8

Publication No. GC31-8514-03

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
GC31-8514-03

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

IBM Corporation
Information Development
Department CGMD / Bldg 500
P.O. Box 12195
Research Triangle Park, NC
27709-9990

Fold and Tape **Please do not staple** Fold and Tape

GC31-8514-03

Cut or Fold
Along Line

File Number: S390-50
Program Number: 5647-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GC31-8514-03