SYBASE®

Installation and Administration Guide

Mainframe Connect™ Server Option

15.0

[IBM CICS]

DOCUMENT ID: DC36510-01-1500-01

LAST REVISED: August 2007

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

The Mainframe Connect™ Server Option for CICS *Installation and Administration Guide* describes how to install and configure the Server Option for CICS. It also addresses system administration.

Note If you want to go directly to the installation instructions, skip to Chapter 2, "Planning Your Installation."

Audience

The guidelines and instructions in this book are intended for those who install, configure, and maintain Sybase® mainframe components on an IBM z/Series mainframe computer. This book refers to anyone performing these tasks as the Server Option administrator.

To use this book, you should have a working knowledge of system administration for your environment.

How to use this book

This table shows how this book is organized.

To	See
Understand the Server Option	Chapter 1, "Introduction to the Server Option"
Plan the Server Option installation	Chapter 2, "Planning Your Installation"
Install the Server Option	Chapter 3, "Installation and Configuration"
Understand Server Option security	Chapter 4, "Security"
Set up tracing and accounting	Chapter 5, "Tracing and Accounting"
Customize the Server Options	Appendix A, "Customization Options"
Reference translation tables	Appendix B, "Translation Tables"
Install IBM CICS sockets interface	Appendix C, "Setting Up the CICS Sockets Interface"
Set up Secure Sockets Layer (SSL)	Appendix D, "Setting Up Secure Sockets Layer Protocol"
Understand gateway-less considerations	Appendix E, "Gateway-less Considerations"
Understand network considerations	Appendix F, "Network considerations"
Troubleshoot problems with client access to data	Appendix G, "Troubleshooting"

Related documents

To install and use the Server Option, you may need to refer to the following documentation:

- Mainframe Connect Server Option *Programmers Reference for PL/1*
- Mainframe Connect Server Option *Programmers Reference for COBOL*
- Mainframe Connect Server Option *Programmers Reference for Remote Stored Procedures*
- Mainframe Connect Client Option *Programmers Reference for PL/1*
- Mainframe Connect Client Option *Programmers Reference for COBOL*
- Mainframe Connect Client Option *Programmers Reference for C*
- Mainframe Connect Client Option *Programmers Reference for Client Services Applications*
- Mainframe Connect Client Option and Server Option *Messages and Codes*
- Mainframe Connect DirectConnect for z/OS Option *Installation Guide*
- Mainframe Connect DirectConnect for z/OS Option *Users Guide for DB2 Access Services*
- Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*
- Enterprise Connect Data Access and Mainframe Connect *Server Administration Guide6*

Other sources of information

Use the Sybase Getting Started CD, the SyBooks™ CD, and the Sybase Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

  Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.

- The Sybase Product Manuals Web site is an online version of the SyBooks CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

  To access the Sybase Product Manuals Web site, go to Product Manuals at http://www.sybase.com/support/manuals/.

Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

❖ **Finding the latest information on product certifications**

1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
2. Select Products from the navigation bar on the left.
3. Select a product name from the product list and click Go.
4. Select the Certification Report filter, specify a time frame, and click Go.
5. Click a Certification Report title to display the report.

❖ **Creating a personalized view of the Sybase Web site (including support pages)**

Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
2. Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ **Finding the latest information on EBFs and software maintenance**

1. Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.
2. Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.
3. Select a product.

4 Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

   Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as a Technical Support Contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click Edit Roles to add the “Technical Support Contact” role to your MySybase profile.

5 Click the Info icon to display the EBF/Maintenance report, or click the product description to download the software.

Conventions

The Server Option uses 8-character function names; other versions of Server-Library use longer names. This book uses the long version of Server-Library names with this exception: the 8-character version is used in syntax statements. For example, in a syntax statement, "CTBCMDPROPS" is written "CTBCMDPR." You can use either version in your code.

Syntax statements that display options for a command look like this:

```
COMMAND [object_name, [ {TRUE | FALSE} ] ]
```

Table 1 explains the syntax conventions used in this guide.

Table 1: Syntax conventions

Symbol	
( )	When you see parentheses, include them as part of the command.
{ }	Braces indicate that you must choose at least one of the enclosed options. Do not type the braces when you type the option.
[ ]	Brackets indicate that you can choose one or more of the enclosed options, or none. Do not type the brackets when you type the options.
|	The vertical bar indicates that you can select only one of the options shown. Do not type the bar in your command.
,	The comma indicates that you can choose one or more of the options shown. Separate each choice by using a comma as part of the command.

This book uses the following style conventions:

This type of information	Looks like this
Gateway-Library function names	TDINIT, TDRESULT
Client-Library™ function names	CTBINIT, CTBRESULTS

This type of information	Looks like this
Other executables (DB-Library™ routines, SQL commands) in text	the dbrpcparam routine, a select statement
Directory names, path names, and file names	*/usr/bin directory*, *interfaces* file
Variables	*n* bytes
Datatypes	datetime, float
Sample code	01 BUFFER PIC S9(9) COMP SYNC
User input	*01 BUFFER PIC X(n)*
Client-Library and Gateway-Library function argument names	*BUFFER*, *RETCODE*
Names of objects stored on the mainframe	SYCTSAA5
Symbolic values used with function arguments, properties, and structure fields	CS-UNUSED, FMT-NAME, CS-SV-FATAL
Client-Library property names	CS-PASSWORD, CS-USERNAME
Client-Library and Gateway-Library datatypes	CS-CHAR, TDSCHAR

All other names and terms are in regular typeface.

Accessibility features

This document is available in an HTML version that is specialized for accessibility. You can navigate the HTML with an adaptive technology such as a screen reader, or view it with a screen enlarger.

The HTML documentation has been tested for compliance with U.S. government Section 508 Accessibility requirements. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Note You might need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool.

For information about how Sybase supports accessibility, see Sybase Accessibility at http://www.sybase.com/accessibility. The Sybase Accessibility site includes links to information on Section 508 and W3C standards.

If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

CHAPTER 1

Introduction to the Server Option

What is the Server Option?

The Server Option is an application programming interface (API) enabling the creation of mainframe applications for use with Sybase client applications. Server Option applications can retrieve and update data stored in mainframe resources like the following:

- DB2 UDB and other relational database management systems (RDMSs)
- Transient Storage (TS) queues
- Transient Data (TD) queues
- VSAM files

Any resource that is accessible from your CICS region is also accessible by a Server Option application. The Server Option is available for CICS, and IMS and MVS.

Note For information on how the Server Option functions in the IMS TM and native MVS environments, see the Mainframe Connect Server Option for IMS and MVS *Installation and Administration Guide*.

Architecture

The Server Option runs on an IBM z/Series or plug-compatible mainframe computer. The Server Option uses either the Logical Unit 6.2 (LU 6.2) or Transmission Control Protocol/Internet Protocol (TCP/IP) communications protocol with a host transaction processor, such as CICS, as a communications front end.

The Server Option supports both three-tier (gateway-enabled) and two-tier (gateway-less) environments. When installing and using the Server Option, follow the instructions in this book for your environment.

Three-tier (gateway-enabled)

In a Server Option network configuration using a three-tier (gateway-enabled) SNA environment, the DirectConnect for z/OS Option acts as the gateway between LAN-based clients and the server. The DirectConnect for z/OS Option routes requests and replies between the client and server using two components: access services and the Transaction Router Service (TRS).

For more information on the DirectConnect for z/OS Option and its components, see the DirectConnect for z/OS Option documentation.

Two-tier (gateway-less)

The two-tier environment allows LAN clients to directly log in to the Server Option, which eliminates the need for a DirectConnect for z/OS Option gateway. However, server-to-server communication is not possible in a two-tier (gateway-less) environment. Other drawbacks of the two-tier environment include the client being limited to accessing a single CICS region, loss of the ability to group transactions, and loss of gateway security features.

If you have standardized on TCP for connectivity between LAN clients and z/OS, you can use the two-tier environment.

Functionality

Server Option applications can receive requests from LAN clients and Client Option applications in either of these ways:

- *In a three-tier environment*, using DirectConnect for z/OS Option access service or Transaction Router Service (TRS).
- *In a two-tier environment*, using TCP. See "Two-tier (gateway-less)" on page 2 for more information on two-tier environments.

This section describes:

- Requests in a three-tier environment
- Requests in a two-tier environment

Requests in a three-tier environment

In a Server Option network configuration using a three-tier (gateway-enabled) SNA environment, the DirectConnect for z/OS Option accepts requests from LAN-based clients and routes them to the appropriate server.

Server Option applications receive requests from LAN clients through either of these DirectConnect for z/OS Option components:

- DirectConnect for z/OS Option DB2 access service
- TRS

DirectConnect for z/OS Option DB2 access service

An access service is a logical server application, used with an access service library, that enables a LAN client to communicate with Server Option applications. Each DirectConnect for z/OS Option server can have multiple DB2 access services.

For more information about DB2 access services, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for DB2 Access Services*.

TRS

The Transaction Router Services (TRS) software allows Sybase clients running on workstations and sharing a local area network (LAN) to access mainframe data and applications. The TRS listener waits for and accepts client requests and routes them to the mainframe, using transaction and connection information that the DirectConnect for z/OS Option administrator provides during configuration.

TRS treats all client requests like remote procedure calls (RPCs). TRS maps each request to a specific mainframe transaction. On receiving a client request, TRS invokes the corresponding mainframe transaction. The CICS transaction processor runs the transaction and returns results to TRS, which forwards the results to the requesting client.

For details, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

Configuration in a three-tier architecture

The mainframe and TRS configuration parameters must be coordinated to permit communication with one another. When configuring a mainframe region to communicate with TRS, coordinate the following mainframe configuration values with TRS:

- For CICS Logical Unit 6.2 (LU 6.2):
  - CICS connection and session definitions
  - Virtual Telecommunications Access Method (SNA)
  - Network Control Program (NCP)
  - SNA, using your TRS platform SNA support program
- For CICS TCP/IP:
  - TCP/IP for z/OS port definitions
  - Sybase listener configuration values

Note The configuration values are provided in the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

Sybase listener in three-tier (gateway-enabled) environments

Note The Sybase listener default transaction name is SY01. You may choose a different name.

The Sybase listener listens on a specific port for any incoming TCP connect request. When a request arrives, the listener performs security or logon processing for the request, then passes the TCP socket to the transaction received from the client. There can be more than one instance of a listener in a single CICS region, and each instance listens on a different port number. For three-tier transactions, the listener issues a command to start the transaction specified by the DirectConnect for z/OS Option and pass the associated socket. Then, the Server Option takes the socket through the Server Option API and manages the connection.

Requests in a two-tier environment

Routing client requests in a two-tier (gateway-less) environment requires use of the Server Option socket handler (SYSH). The socket handler is a conversational transaction that matches client application procedure calls to Server Option RPCs, thereby enabling a client application to access the Server Option directly (in two-tier mode) without needing to route transactions through a gateway.

For each client application logged in to the Server Option, a socket handler transaction instance receives client requests and manages the connection between the client application and the Server Option. In addition, CICS activates the socket handler transaction instance to handle a transaction abend. An active socket handler instance terminates when its associated client application logs out.

Configuration in a two-tier environment

Since the same mainframe transactions support both gateway-enabled and gateway-less access, configuring a workstation for gateway-less mainframe access is similar to configuring for gateway-enabled access.

To set up for gateway-less environments, migrate the LAN TRS RPC mapping definition files to the *SYRPCFIL* file on the mainframe. This is accomplished by using a CICS transaction called SYRP, which maps LAN RPC names to CICS transaction names. *SYRPCFIL* is a VSAM file that stores the RPC mapping entries.

The SYRP transaction is a panel-driven interface that allows the user to add, delete, change, and display entries in the *SYRPCFIL* file.

Long-running transactions

The Server Option supports user-defined, long-running transactions. Do not confuse the socket handler connection management with the "long transaction" modes that are handled by the language transactions.

For more information about long-running transactions in the Server Option, see the Mainframe Connect Server Option *Programmers Reference* for the appropriate programming language. PL/1 and COBOL versions of this guide are available.

CHAPTER 2

Planning Your Installation

Choosing a network driver

The Server Option provides added flexibility and easy installation for sites configured to run both SNA and TCP/IP by supporting the concurrent use of multiple network drivers. Programs can invoke network drivers from the same Server Option and Client Option common code base, and the appropriate network driver loads dynamically during program execution.

Note For information on network considerations, see Appendix F, "Network considerations."

General criteria for choosing a driver

The choice of a network driver depends on your network type and operating environment.

CICS environment

The following drivers are supported in the CICS environment:

- TCP/IP for an IBM network
- LU 6.2 for an SNA network

The following table indicates which drivers can be used by the Server Option for CICS in two-tier and three-tier environments.

Driver	Gateway-enabled	Gateway-less
LU62CICS	X	
TCPCICS	X	X

Choosing between a CPI-C/LU 6.2 driver and a TCP/IP driver

If your network uses only SNA or only TCP/IP, you must choose the driver that supports your network protocol. Performance is also a consideration in choosing a network driver.

SNA performance

Because LAN operating systems and applications do not support the SNA protocol, networks using SNA require a gateway to communicate between the mainframe and the LAN. This added layer of communication adds a burden to performance. Consequently, communication may be somewhat slower than communication in a two-tier, or gateway-less, architecture.

TCP/IP performance

Because the TCP/IP protocol is commonly recognized and supported among LAN operating systems and applications, it is not limited to a gateway-enabled architecture and can also be used in a gateway-less architecture. The added layer of a three-tier, gateway-enabled architecture may cause somewhat slower communication for TCP/IP than in a two-tier, gateway-less architecture.

Planning the installation

This section describes the installation media and pre-installation tasks.

Installation media

The Server Option is distributed on CD or in downloadable form.

Note For information on obtaining the latest EBFs for the Server Option, see *Release Bulletin* for the product.

Pre-installation tasks

Installation requires completing the following pre-installation tasks, which are explained in the following subsections. You should skip those tasks that do not pertain to the option or options you have chosen to install.

1. Verify the platforms, components and distributed software
2. Determine JCL and system information
3. Determine CICS and DB2 UDB information
4. Determine compiler information
5. Determine Server Option information
6. Determine FTP information
7. Plan the security requirements
8. Identify the change control requirements
9. Back up the release libraries (upgrades only)
10. Determine the library names
11. Verify the connectivity

Task list

Perform these tasks before you begin installation.

1. Verify the platforms, components and distributed software

See the Mainframe Connect Server Option for CICS *Release Bulletin*.

2. Determine JCL and system information

Determine the following information to be used in the installation procedure:

- JCL jobcard values – used in the final installation jobs run in TSO.
- High-level qualifier – used as a prefix for data sets generated during installation.
- Volume serial number – indicates where generated data sets are cataloged.
- Unit parameter value – indicates the device requirements for cataloging generated data sets.
- Work unit – for the use of temporary work data sets.
- Customer CICS, IMS, and MVS LOADLIBs – pre-cataloged partitioned data sets (PDSs) or partitioned data sets extended (PDSE), into which configuration modules and sample programs are to be linked.

3. Determine CICS and DB2 UDB information

Determine the following information if you intend to install a component that uses CICS or DB2 UDB:

- High-level qualifier for CICS system data sets.
- RDO data set name (DSN) – the name of the CICS RDO (DFHCSD) containing the application resource definitions used by your CICS region.
- RDO group list – the RDO group list used by your CICS region when executing an initial start.
- The CICS region APPLID – the VTAM APPLID for your CICS region.
- DB2 system data sets high-level qualifier – the high-level qualifier used for DB2 system data sets.
- DB2 exit data set name (DSN).
- DB2 DSN.

4. Determine compiler information

Determine the following information if you intend to install an API component:

- LE370 high-level qualifier – used for the Language Environment 370.

- COBOL compiler name – the module used to execute COBOL in your environment.
- COBOL compiler LOADLIB – the system LOADLIB where your COBOL compiler module resides.
- PL/1 compiler name – the module used to execute PL/1 in your environment.
- PL/1 compiler LOADLIB – the system LOADLIB where your PL/1 compiler module resides.
- C compiler data sets high-level qualifier – the high-level qualifier used for C.
- TCP/IP data sets high-level qualifier.

5. Determine Server Option information

Determine the following information for use in installing the Server Option:

- TCP address space name.
- Remote server name – the name by which your Server Option applications will refer to the remote server.
- Remote TCP host name – the DSN name for the remote server.
- Remote server TCP host port – the TCP/IP port used by the remote server.

6. Determine FTP information

Determine the following information needed to establish an FTP connection to your mainframe:

- User ID.
- Password.
- Mainframe host name.
- Control port number – the listener port used by your mainframe FTP server, usually 21.
- TCP address space name.
- Volume serial number or unit – either a volume serial number (VOL=SER) and unit assignment for FTP to use or allow FTP to use default values.
- Log path name – indicates where FTP log information is to be written.

7. Plan the security requirements

Review your security requirements with your security administrator. You may also need to consult with your network administrator.

8. Identify the change control requirements

Create a change control plan that includes:

- All the tasks that need to be considered for installation
- The different groups that need to be aware of the environment change, for example, field personnel and groups involved in administering applications, z/OS, security, change control, and scheduling
- A schedule, including cut-off dates for specific tasks

9. Back up the release libraries (upgrades only)

If you are upgrading an existing release of the Server Option, Sybase strongly recommends that you back up the entire set of release libraries before beginning this installation.

10. Determine the library names

The shipped library names are unique for this release. If you are upgrading, decide whether you want to use your current library names. If this is a new release, you still might want to consider how to name the files.

You do not need to remove previous releases from your Sybase libraries because default names shipped with this release create an entirely unique set of release libraries. You can change them, however, based on naming standards at your site.

Note When the upgrade is complete and tested, be sure to replace the old LOADLIB name or add the new LOADLIB name to the DFHRPL concatenation for the selected CICS regions, as described in the installation instructions.

If you are going to continue to use the old Sybase library names, delete all members before installing the new ones with the new version.

11. Verify the connectivity

Use the standard LAN ping utility to ensure connectivity between z/OS and the workstation running Adaptive Server® Enterprise (ASE).

CHAPTER 3

Installation and Configuration

Installing and configuring Server Option for CICS

Note Be sure you completed the tasks in Chapter 2, “Planning Your Installation.”

Licensing information

The Server Option for CICS requires a permanent authorization key. However, Sybase includes a temporary key, which is valid for 30 days, within the order at installation time. To avoid interrupting your operations, call Customer Service at 1-800-8Sybase (1-800-879-2273), select Option 3, then select Option 3 again, and request a permanent key.

When speaking with Customer Service, be sure to have this information ready:

- Product name
- Order number
- For the machine you are using:
  - Serial number
  - Machine type
  - Model number
- A valid e-mail address

Note Please allow seven business days for the key to be generated and sent to you.

The two procedures in this subsection describe the installation steps necessary to install all Mainframe Connect options from the installation program and to complete the installation for the Server Option for CICS. You should skip those installation steps that do not pertain to the option or options you have chosen to install.

Note The installation program runs only on Windows.

❖ **To install from the installer wizard**

1 Start the installation program from CD by executing *setupwin.exe*, which is in the root directory.

The initial dialog box displays the options available for installation. Click Next and Back to navigate through the wizard. To cancel the installation, click Cancel. Click Next to proceed.

2 Accept the terms of the user-license agreement by selecting your country in the drop-down list and selecting the option to indicate that you agree with the terms. Click Next.

3 Select the components you want to install and click Next.

Note If you are installing the Server Option for CICS or the DB2 UDB Option for CICS, those runtime components will be automatically selected as you proceed to the next window.

4 Provide the following JCL and system information:

- *JCL Line 1-3* – a valid jobcard, used to run the final installation jobs in TSO.
- *High Level Qualifier* – the high-level qualifier, used as a prefix for all data sets generated during installation.
- *Volume* – the volume serial number that indicates where generated data sets are cataloged.
- *Unit* – the unit parameter value that indicates the device requirements for cataloging generated data sets.
- *Work Unit* – for the use of temporary work data sets.

- *Customer CICS, IMS, and MVS Loadlibs* – pre-cataloged partitioned data sets (PDSs) or partitioned data sets extended (PDSE) into which configuration modules and sample programs are to be linked. For CICS, this data set should be in the DFHRPL configuration ahead of other Sybase libraries.

Click Next.

5 If you are installing an option that uses CICS, DB2, or IMS, provide this information where it applies. Otherwise, skip to the next step.

- *CICS system datasets hlq* – the high-level qualifier for CICS system data sets used to locate *SDFHLOAD* and other CICS libraries.
- *RDO Dataset*– the name of the CICS RDO (DFHCSD) containing the application resource definitions used by your CICS region.
- *RDO Group List* – the RDO group list used by your CICS region when executing an initial start.
- *CICS Region Applid* – the VTAM APPLID for your CICS region.
- *DB2 system datasets hlq* – the DB2 system data sets high-level qualifier used for DB2 system data sets.
- *DB2 Exit Dataset* – the name of the DB2 exit data set used by your DB2 region.
- *DB2 DSN Name* – the data set name (DSN) of your DB2 region.
- *IMS datasets hlq* – the high-level qualifier for IMS system data sets used to locate IMS libraries.

Click Next.

6 If you are installing an API component, provide this compiler information, which is used to configure JCL for compiling sample programs. Otherwise, skip to the next step.

- *LE/370 datasets hlq* – the LE370 high-level qualifier used for the Language Environment 370, used here to locate data sets like CEELKED.
- *COBOL Compiler Name*– the module used to execute COBOL in your environment.
- *COBOL Compiler Loadlib* – the system loadlib in which your COBOL compiler module resides.

- *PLI Compiler Name* – the module used to execute PLI in your environment.
- *PLI Compiler Loadlib* – the system loadlib in which your PLI compiler module resides.
- *C compiler datasets hlq* – the high-level qualifier used for C, used to locate data sets like SBCCMP.
- *TCP/IP datasets hlq* – the TCP/IP data sets high-level qualifier used to locate data sets like SEZATCP.

Click Next.

7 If you are installing the Client Option for CICS, provide this information for configuring a host connection definition for the Client Option. Otherwise, skip to the next step.

- *TCP Address Space Name* – the name of your TCP/IP region.
- *Server Name* – the name by which your Client Option applications refers to the remote server.
- *Server TCP Host Name* – the DNS name for the remote server.
- *Server TCP Host Port* – the TCP/IP port used by the remote server.

Click Next.

8 If you are installing the Server Option for CICS or the DB2 UDB Option for CICS, provide this information for configuring a TCP/IP listener for these options. Otherwise, skip to the next step.

- *TCP Address Space Name* – the name of your TCP/IP region.
- *Listener Port* – the port on which the option listens.

Click Next.

9 Click Next until the installer displays the information you entered in steps 5 through 8. Review this information and, if necessary, click Back to return to previous windows and make corrections.

10 Click Next until the wizard displays a dialog box for FTP information. Provide the following data for establishing an FTP session to your mainframe:

- *Userid* – the mainframe user ID for the FTP session.
- *Password* – the password for the FTP session.
- *Mainframe Host Name*– the mainframe DNS name.

- *FTP Port* – the control port used by your mainframe FTP server, usually 21.
- *VOL/UNIT Assignment* – either a volume serial number and unit assignment for FTP, or allow FTP to use default values.

  Note If you specify a volume serial number that does not exist, FTP suspends operation until the mainframe responds to a message requesting that the volume be mounted.

- *Log FTP Commands* – indicates where FTP log information is to be written. This log information may be useful in troubleshooting FTP problems.

The installation program will create JCL and upload the selected components to your mainframe when you click Next.

11 Close the installation program.

To complete the installation of your Mainframe Connect components, review and submit JCL from TSO. If you are installing multiple components, Sybase strongly suggests you install in this sequence:

1 Client Option for CICS

2 Server Option for CICS

3 DB2 UDB Option for CICS

4 Any other options

❖ **To complete the installation**

1 Locate the installation JCL for the Server Option for CICS in *hlq.OSC150.CICS.JCL*, where *hlq* is the high-level qualifier you specified in step 5.

2 Run the following jobs in the order they are described here, where *x* is an integer that denotes the order in which the job is to be run in the overall sequence of jobs. Ignore jobs that are not present or relevant to the option you are installing.

- *IxRECV* – runs IKJEFT01 to use the TSO RECEIVE command to build and populate the product libraries.

- *IxRDO* – runs the CICS Resource Definition Utility, DFHCSDUP, to define the transaction, program, and file entries for the Server Option for CICS. If your CICS region has had a previous version of the Server Option, you may need to uncomment or change the DELETE and REMOVE entries at the top of the RDO input.
- *IxVSAM* – allocates the VSAM data sets used for error and trace logging.
- *IxTPSEC* – for listener security options. For information about setting up the CICS sockets interface, see Appendix C, "Setting Up the CICS Sockets Interface."
- *IxHOST*– assembles and links the Server Option for CICS customization module, character sets, and remote host definitions. You may rerun this job at any time to change configuration and character sets or to add, remove, or modify remote host definitions and license keys.

  Note The Client Option for CICS installation has its own *IxHOST* job. If you are installing both the Client and Server Options for CICS, you should run the *IxHOST* job contained in *hlq.OSC150.CICS.JCL*.

- *IxRPC*– adds definitions to the SYRPCFIL dataset, which is created during the installation of the Server Option and required for the DB2 UDB Option.
- *IxDELETE*– deletes the data sets in the TSO XMIT format used for the installation.

3 Run these jobs if you want to compile and link-edit the sample applications provided with the Server Option for CICS:

- *SAPASMD* – assembles and links sample assembler language applications that use the Server Option API and access DB2 UDB.
- *SAPASMV* – assembles and links sample assembler language applications that use the Server Option API and access VSAM.
- *SAPCOBD* – compiles and links sample COBOL language applications that use the Server Option API and access DB2.
- *SAPCOBV* – compiles and links sample COBOL language applications that use the Server Option API and access VSAM.
- *SAPPLID* – compiles and links sample PL/1 language applications that use the Server Option API and access DB2.

- *SAPPLIV* – compiles and links sample PL/1 language applications that use the Server Option API and access VSAM.
- *SRSPASM1*– compiles and links sample assembler language applications that use the RSP API.
- *SRSPASM2* – compiles and links sample assembler language applications that use the RSP API and access DB2.
- *SRSPCOB1* – compiles and links sample COBOL language applications that use the RSP API.
- *SRSPCOB2* – compiles and links sample COBOL language applications that use the RSP API and access DB2.

Libraries and samples

For a list and description of the libraries, sample programs, JCL, and transactions for your product, see the *CONTENTS* member of the JCL data set.

CHAPTER 4 # Security

Understanding Server Option security

Security for Server Option processing is implemented at several levels and according to the method used to access CICS.

Implementing security in the Server Option is a complex task, and Sybase recommends that you read this chapter before installing the Server Option.

For information about:

- ASE security, refer to the Adaptive Server Enterprise *System Administration Guide*.
- Mainframe security, refer to documentation provided with CICS, IMS TM, or the appropriate mainframe security system.
- Security for DB2 access service requests through the DirectConnect for z/OS Option, refer to the Mainframe Connect DirectConnect for z/OS Option *Users Guide for DB2 Access Services*.
- Secure Sockets Layer (SSL) implementation and configuration, refer to Appendix D, “Setting Up Secure Sockets Layer Protocol.”

Server Option security is implemented through the Sybase component architecture and through the connectivity used to communicate between components.

Security in the Sybase architecture

Sybase components can provide security at the levels described in the following subsections:

- Client workstation
- ASE
- TRS
- Vendor SNA support software
- Mainframe

Note You should coordinate efforts to set up and maintain security between these components, and communicate changes when they occur.

Client workstation

Most workstations have a secure login that verifies the identity and authorization of the user by requiring a unique user ID and password. This client user ID, password, and profile information may be passed to ASE and the DirectConnect for z/OS Option.

ASE

Adaptive Server Enterprise (ASE) performs security checks on requests through ASE and may grant or deny a user permission to call a remote procedure. Your TRS administrator can apply this security to all requests by specifying the -D (indirect access) parameter when starting TRS. This parameter requires all client requests to pass through ASE. For further details, refer to the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

Note You must have a three-tier architecture to route transactions in server-to-server mode (for example, through an ASE to the mainframe).

TRS

This section addresses:

- External security systems
- Defining security
- Overriding security
- Conversation-level security

External security systems

Most mainframe-based security systems, such as RACF, are based on user login information. The system employs user ID and password information, restricting transaction access to authorized users.

If the communications support at your TRS platform sends login access information to the mainframe, you can use any security system that works with CICS. Otherwise, use one or more of the security methods documented in this chapter.

Defining security

Implementing security in TRS entails defining both client logins and RPCs.

Defining a client login

Under TRS security, every client login must be defined to TRS. For each client, this login definition indicates:

- The client login ID and password
- A host login ID and password (optional)
- Lists of the connections and host transactions available to clients using the login

Your TRS administrator can restrict client access to specific host resources by working with mainframe systems programmers and security administrators, and also by carefully defining user IDs, host IDs, transactions, and connections.

Defining an RPC

When the TRS administrator defines an RPC to TRS, these security options are available:

- None
- User ID

- Both (user ID and password)

By defining an RPC to TRS, the TRS administrator sets security parameters for transactions associated with the RPC. Each option above indicates the type of login information passed to the mainframe when a client calls an RPC.

Verification

In enforcing TRS security, the mainframe verifies that the requesting client has authorization to access the specified transaction. If no proper authorization exists, the mainframe returns an error message.

Overriding security

Your TRS administrator can override TRS security by setting the Security configuration property to No in the TRS configuration file. This allows you to map users to transaction groups that allow specific RPCs. For more information about what the security parameter does, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

Conversation-level security

You can set up conversation-level security, a process by which TRS passes client login information to the mainframe when it allocates a conversation. Under conversation-level security, the following can be passed to the host:

- A pre-defined host ID and password, which can be set up in the login definition
- A separate ID and password attached to the transaction group of the client

Vendor SNA support software

The SNA support software of the vendor may send login information to the host in FMH-5 fields with client requests. This allows you to use external security products that require client login information.

Mainframe

Security at the mainframe level concerns all components residing on the mainframe and components that interact with the mainframe, including CICS, the DirectConnect for z/OS Option, your database, the Server Option, and the DB2 UDB Option for CICS.

CICS

CICS works with security systems like RACF to verify transaction requests against the user ID and password. The authorization ID passed to DB2 UDB from CICS is system-dependent, based on the security requirements at your installation. You specify the authorization ID in the CICS RCT table with the AUTHID parameter.

DirectConnect for z/OS Option

If the communications software of your DirectConnect for z/OS Option platform supports passing login information to the mainframe, you can use an external mainframe security product, such as RACF, that requires client login information.

DBMS

Your mainframe DBMS may have additional security mechanisms.

Server Option

You can customize the Server Option to specify whether an access code is required to retrieve client passwords. See Appendix A, “Customization Options.”

DB2 UDB Option for CICS

Note The transaction name for the DB2 UDB Option for CICS is AMD2.

The security requirements are as follows:

- The current user must have execute privileges on the DB2 UDB plan for the Catalog RPCs.
- The DB2 UDB CURRENT SQLID must be the same for the AMD2 transaction and plans, and for Catalog RPCs.

The shipped default authority for AMD2 and Catalog RPCs is AUTH=(AMD2). You must change the default to set up security. When you do, be sure to keep AMD2 and Catalog RPCs in synchronization.

Note For specific security information about the DB2 UDB Option for CICS, refer to the Mainframe Connect DB2 UDB Options for CICS and IMS *Installation and Administration Guide*.

Connectivity security

In addition to security provided by components in the Sybase architecture, Server Option security can be provided through LU 6.2 and TCP/IP connectivity.

LU 6.2 security for CICS

Note TCP/IP security for CICS is discussed in "TCP/IP security for CICS" on page 31.

Conversation-level security with LU 6.2 requires changes to CICS and SNA definitions. The CICS definitions and how to set up conversation-level security are discussed in the following subsections. Implementation of this security requires mainframe and TRS coordination.

For corresponding TRS requirements, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*. For more information about CICS LU 6.2 security, see the IBM *CICS-RACF Security Guide*.

This section contains the following subsections:

- Setting CICS definitions for conversation-level security
- Setting up conversation-level security with LU 6.2

Setting CICS definitions for conversation-level security

This section addresses the three parts to conversation-level security.

LU 6.2 bind-time security

Bind-time security is controlled by the Bindsecurity parameter on the CICS RDO Connection Definition. If Bindsecurity is set to YES, CICS applies LU 6.2 bind-time security to determine whether a requested session is authorized.

CICS uses a password to verify session authorization. The password supplied in CICS must match the password defined on the workstation. The SECURITY Bindpassword parameter in the CICS Connection Definition supplies the password.

Refer to the documentation for the SNA support on your remote system for information about defining the bind password.

CICS link security

CICS link security is required for conversation-level security to CICS. Use link security to define CICS security values on the LU 6.2 session. To specify link security, specify a valid user ID in the SECURITY SEcurityname parameter of the CICS Connection Definition.

When the session is bound after checking bind-time security, CICS checks the External Security Manager to see if the user ID is valid. If it is valid, CICS uses that user ID for the session authorization.

User security

For individual users, the SECURITY ATtachsec parameter in the CICS connection definition determines what type of security is active for a connection. Table 4-1 shows the options:

Table 4-1: User security ATtachsec options

ATtachsec option	Result
LOCAL	CICS does not require a user ID from the remote system and ignores any sent. User security is set to the link security.
IDENTIFY	CICS requires a user ID on every attach request. CICS internal security or an external security manager verifies the user ID.
VERIFY	CICS requires a user ID and password on every attach request. Your security manager verifies both.

Setting up conversation-level security with LU 6.2

Setting up a successful security system for use with the Server Option in a CICS LU 6.2 environment requires careful synchronization between SNA, CICS, and TRS. These steps are explained in the following subsections:

1. Define security in the SNA logmode entry
2. Specify the mode
3. Specify a link security user ID
4. Coordinate the modename parameter and the SNA logmode entry

1. Define security in the SNA logmode entry

To allow an LU to support conversation-level security, you must set the PSERVIC parameter on the SNA logmode entry. Assign each LU a logmode corresponding to the desired level of security.

The 10th byte of PSERVIC determines security as follows:

- x’00’ – LOCAL

- x'12' – IDENTIFY
- x'10' – VERIFY

See Table 4-1 for descriptions of LOCAL, IDENTIFY, and VERIFY.

2. Specify the mode

In your network definition to SNA, specify the mode you defined in the Logmode entry. You can apply the Logmode entry to a specific LU statement, or apply it globally through the PU statement.

```
SYBPU1    PU    CUADDR=041,DLOGMOD=M6P1024V,MAXBFRU=11,                  +
                 USSTAB=ISTINCDT,DELAY=0,SECNET=YES,ISTATUS=ACTIVE,      +
                 XID=YES,PUTYPE=2,VPACING=0,PACING=0
SYBLU01   LU    1        LOCADDR=0
```

3. Specify a link security user ID

In the CICS Connection Definition, set SEcurityname to specify a valid user ID, which will be used to determine the session authorization. Also, set the ATtachsec parameter, as shown in this example:

```
 OBJECT CHARACTERISTICS
        CEDA  View
        Connection:        SYB1
        Group:             SYBCONS
        DEscription:
    CONNECTION IDENTIFIERS
        Netname:           SYBLU01
        INDsys:
    REMOTE ATTRIBUTES
        REMOTESystem:
        REMOTEName:
    CONNECTION PROPERTIES
        ACcessmethod:      SNA        SNA | IRc | INdirect | Xm
        Protocol:          Appc       Appc | Lu61
        SInglesess:        No         No | Yes
        DAtastream:        User       User | 3270 | SCs |STrfield |Lms
        RECordformat:      U          U | Vb
    OPERATIONAL PROPERTIES
+         AUtoconnect:     All        No | Yes | All
+         INService:       Yes        Yes | No
    SECURITY
        SEcurityname:      SYBUSER
        ATtachsec:         Verify     Local | Identify | Verify
        Bindpassword:                 PASSWORD NOT SPECIFIED
```

4. Coordinate the modename parameter and the SNA logmode entry

In the CICS session definition under SESSION IDENTIFIERS, make sure the MOdename parameter matches the logmode in the SNA Logmode Entry. Based on this example (see steps 1 and 2), MOdename would be M6P1024V.

At the DirectConnect for z/OS Option, the TRS administrator sets up TRS for conversation-level security, along with other TRS security, based on site requirements. For details, see the chapter on security in the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

TCP/IP security for CICS

The Server Option for CICS supports IBM TCP/IP.

Note For listener security options, see "Customizing the SYBTPSEC configuration module" in Appendix C, "Setting Up the CICS Sockets Interface."

Authorizing surrogate users

The Sybase TCP/IP listener issues a CICS VERIFY PASSWORD command to validate the user ID and password passed by the client. Then, the listener issues the START TRANSACTION command to start a surrogate transaction, which uses the CICS INQUIRE command to validate the user ID against the RPC request and start the transaction.

The user ID used by the Sybase listener must have sufficient authorization to execute the VERIFY and START TRANSACTION commands. See the appropriate IBM documentation on RACF security for more information.

Sybase listener security checking

The Sybase listener performs security checking for users connecting both through a three-tier, gateway-enabled, and a two-tier, gateway-less environment. This section explains which user ID is associated with the Sybase listener and the processing for both of these scenarios.

User ID associated with the listener

Use the SIT PLTIUSR parameter to assign a user ID to your PLT programs. All PLT programs run under the transaction ID CPLT. If XUSER=YES in the SIT, surrogate authorization is checked before the CPLT transaction ID is attached. The CICS region *userid* must be authorized as a surrogate for the PLTIUSR *userid*. If a value is not specified for the PLTIUSR parameter, no surrogate checking is done, and PLT programs run under the authorization of the CICS region *userid*.

Three-tier, gateway-enabled processing

The Sybase listener uses the client user ID and password as input to the EXEC CICS VERIFY PASSWORD command. Verification proceeds as follows:

- If the user ID and password are valid, the client transaction is started with the USERID parameter.
- If surrogate checking is active, the user ID under which the Sybase listener was started is checked to see if it is authorized to the *USERID*.DFHSTART profile, where user ID (in this case) is the user ID passed up from the client.
- If the password has expired, the Sybase listener checks to see if the client RPC is the PEM RPC called SYB_PEM. If so, the transaction is started, and the client may change the password.
- If any other type of error results from VERIFY PASSWORD, the client receives an error notification, and a message is sent to the CICS log.
- If security is not on in this region (SEC=NO in the SIT), the client transaction is started without the USERID parameter.

Two-tier, gateway-less processing

The Sybase listener uses the client user ID and password as input to the EXEC CICS VERIFY PASSWORD command. Verification proceeds as follows:

- If the user ID and password are valid, the Sybase listener starts the Sybase Sockets Handler (SYSH) transaction with the USERID parameter.
- If surrogate checking is active, the user ID under which the Sybase listener was started is checked to see if it is authorized to the *USERID*.DFHSTART profile, where USERID (in this case) is the user ID passed up from the client. Then, the SYSH transaction starts the client transaction using the START command with the USERID parameter.

- If the password has expired, the Sybase listener sets a flag and starts SYSH with the USERID parameter. Then, SYSH checks to see if the client RPC is the PEM RPC called SYB_PEM. If so, the corresponding transaction is started with the USERID parameter. This allows the client to change the password.
- If any other type of error occurs on the VERIFY PASSWORD, the Sybase listener sets a flag, and the socket handler is started without the USERID parameter. If a security error flag is set, the socket handler notifies the client of the error, and a message is sent to the CICS log. The client transaction does not run.
- If security is not on in this region (SEC=NO in the SIT), the SYSH transaction is started without the USERID parameter. Then, SYSH starts the client's transaction without the USERID parameter.

CHAPTER 5

Tracing and Accounting

Tracing

Server Option provides tracing functions for tracing program activity, either for all transactions (global tracing) or for individual transactions (specific tracing). Server Option writes header and data information to the error log under CICS as VSAM ESDS file API traces, which trace calls from the client application to Server Option using the CICS auxiliary (“aux”) trace facility.

Server Option trace functions

Server Option trace functions allow you to do three types of tracing:

- API tracing, which traces Server Option calls
- Tabular Data Stream™ (TDS) header tracing
- TDS data tracing

You can enable and disable any kind of transaction tracing globally or specifically, using these Server Option functions:

- TDSETLOG – to turn tracing on or off and, in CICS, to change the name of the trace log
- TDINFLOG – to determine whether tracing is enabled and, in CICS, to name the trace log
- TDSETSPT, TDLSTSPT, and TDINFSPT – to enable, disable, and retrieve information about specific tracing

- TDWRTLOG – to write your own record or add a system entry to the trace log file

For complete descriptions and examples of these functions, see the Mainframe Connect Server Option *Programmers Reference* for the appropriate language. PL/1 and COBOL versions of this guide are available.

Trace log

Under CICS, the Server Option trace facility stores header and data trace information in a VSAM *ESDS* file. As installed, this file is named *SYTDLOG1*.

The Server Option appends TDS trace records to the log until it becomes full. When the log is full, all subsequent attempts to write to that log are rejected. To make room for new records, do *one* of the following:

- Archive or delete trace records,
- Change the name of the trace/error log, or
- Change the underlying VSAM file assigned to this name.

The CICS auxiliary datasets record API activity in CICS. Depending on how the trace facility is set up, the dataset either fills up and fails to record further information, or it wraps, overlaying information. To retrieve the auxiliary trace output, you can dump the file using a CICS utility or a third-party vendor package designed for this purpose. Refer to CICS documentation for details about the auxiliary trace facility.

Trace log layout

Table 5-1 shows the layout of the trace log used under CICS.

Note The structures of the trace log have the same layout in CICS as in IMS TM, although the log headers differ slightly.

Table 5-1: Trace log layout

Position	Field name	Field type	Field description
1-2	log-length	Unsigned 2-byte integer	Record length. The total length of this record. (Maximum size is 640.)

Position	Field name	Field type	Field description
3	log-type	Unsigned byte	Type of record: • The *trace* type can be an error record (TDS-ERR-REC; the value is 1) or a trace record (TDS-TRACE; the value is 2). • The *accounting* log type value is 0.
4	log-direction	Unsigned byte	Communication state: Shows whether the mainframe is in send or receive mode.
5-68	log-key	Unsigned byte (max. length = 30) Unsigned byte (length=8) Unsigned byte (length = 24) Unsigned 2-byte integer	Structure containing the following: user_id: The server login ID of the client, from the login packet. trace_resid: Trace resource ID. In CICS, this keeps track of who is doing the tracing. datetime: Date and time the SNA conversation or TCP/IP session started (TDACCEPT was issued). uniquekey: Reserved for future use to ensure record has unique key.
69-82	log-txp-name	Unsigned byte	Transaction name.
82-112	log-conn-id	Unsigned byte	Connection ID. Name by which the connection is known to TRS.
113	log-connp	Unsigned byte	Name of associated TDPROC structure.
114-117	log-error-rc	4-byte integer	Value returned to the RETURN-CODE parameter of a Server Option function (TDS-xxxx). See the Mainframe Connect Client Option and Server Option *Messages and Codes* for more information about return codes.
118-121	log-err-type	4-byte integer	Type of error detected.
122-123	log-err-reserved	2-byte integer	Reserved for future use.
124-125	log-data-length	2-byte integer	Length of the data to be logged.
126-637	log-data	Unsigned byte	Data, including the packet header and the data. For TDWRTLOG, this is the message being written to the log.
637-640	log_filler	Unsigned byte (length=36)	Filler, to fill out record to 640 bytes.

Using the tracing facility

This section describes how to use the Server Option tracing facility.

❖ **To use the trace facility**

1 Call TDSETLOG and perform these steps for global or specific tracing:

For this type of tracing	Do this
Global	1 Set the trace flag to the TRACE ALL RPCS option. 2 Set the flag for each desired kind of tracing to TRUE. **Note** If you want to enable tracing for the entire program, TDSETLOG must precede TDACCEPT.
Specific	1 Set the trace flag to the TRACE SPECIFIC RPCS option. 2 Set the flag for each desired type of tracing to TRUE.
Both global and specific	1 Perform the previous steps for global and specific tracing. 2 If you are developing under CICS and want to enable API tracing, give the auxiliary trace log a CICS ID.

Note You can use TDINFLOG at any time to check the value of the settings.

2 For each transaction for which you want trace activity, call TDSETSPT and perform these steps:

1 Identify the transaction.

2 Set the transaction trace flag to TRUE.

3 Set the trace options flags for the types of tracing desired.

Note You can enable tracing for up to eight transactions at a time.

Trace table for individual transactions

When you enable tracing for an individual transaction, TDSETSPT adds the transaction to a *trace table*, which can contain up to eight entries. For examples, see “Specific tracing example” on page 39.

When you disable tracing for a transaction, its position in the trace table becomes available for another transaction. If all eight positions are in use, you can trace more transactions only if you turn tracing off for one of the transactions in the list or set global tracing on.

You can query the trace table one of two ways:

- Call TDINFSPT to determine if tracing is enabled for a specific transaction. You specify the transaction ID, and TDINFSPT returns the trace flag setting, or
- Call TDLSTSPT to get a list of all transactions for which tracing is currently enabled. TDLSTSPT returns this list as an array.

Specific tracing example

The following example shows how to enable or disable tracing for specific transactions. It also shows how TDSETSPT calls affect the contents of the trace table. TDS packet tracing is initially turned on for eight specific transactions. Tracing continues for the specified functions until a TDSETSPT call turns tracing off for those functions, or until TDSETLOG disables tracing entirely.

This example does not show exact syntax or arguments; it merely indicates which flags and transactions are set. See the sample program in the appropriate Mainframe Connect Server Option *Programmers Reference* for an example of exact coding. PL/1 and COBOL versions of this guide are available.

```
*--------------------------------------------------------------------*
*  First, initialize your environment and set on specific tracing.   *
*--------------------------------------------------------------------*
CALL 'TDINIT' ...
CALL 'TDSETLOG' ... (global flag: OFF,
      API flag: ON,
      header flag: OFF,
      data flag: OFF)...

*--------------------------------------------------------------------*
* Enable packet tracing (option 01) for a specific transaction.     *
*--------------------------------------------------------------------*
CALL 'TDSETSPT' ... (trace flag: ON,
     trace option: 01,
     tran ID: MYT1)...

*--------------------------------------------------------------------*
* Use the same parameter values except the transaction ID           *
* in the next seven TDSETSPT calls.                                 *
```

```
*-------------------------------------------------------------------*
CALL 'TDSETSPT' ... (tran ID: MYT2)...
CALL 'TDSETSPT' ... (tran ID: MYT3)...
CALL 'TDSETSPT' ... (tran ID: MYT4)...
CALL 'TDSETSPT' ... (tran ID: MYT5)...
CALL 'TDSETSPT' ... (tran ID: MYT6)...
CALL 'TDSETSPT' ... (tran ID: MYT7)...
CALL 'TDSETSPT' ... (tran ID: MYT8)...

*-------------------------------------------------------------------*
* With tracing on, begin to accept and process client requests.    *
*-------------------------------------------------------------------*
CALL 'TDACCEPT'
.
.
.
```

At this point, the trace table looks like this:

Table 5-2: Sample trace table (1)

Transaction ID	Tracing flag
MYT1	TRUE
MYT2	TRUE
MYT3	TRUE
MYT4	TRUE
MYT5	TRUE
MYT6	TRUE
MYT7	TRUE
MYT8	TRUE

Later, you decide to turn on tracing for one more transaction, MYT9:

```
*-------------------------------------------------------------------*
* Try to turn on packet tracing for MYT9.                           *
*-------------------------------------------------------------------*
CALL 'TDSETSPT' ... (trace flag: ON,
          trace option: 01,
          tran ID: MYT9)...
*-------------------------------------------------------------------*
* The operation fails, and you get a return code of SOS,            *
* indicating that the trace table is full.                          *
* The contents of the trace table do not change.                    *
* To make room in the table for MYT9, you decide to                 *
* turn tracing off for MYT0.                                        *
*-------------------------------------------------------------------*
```

```
CALL 'TDSETSPT'...(trace flag: OFF,
          trace option: 01,
          tran ID: MYT0)...
*-------------------------------------------------------------------*
* The operation fails, and you get a return code                    *
* of ENTRY NOT FOUND, indicating that there is no such              *
* transaction listed in the trace table.                            *
* The contents of the trace table do not change.                    *
*-------------------------------------------------------------------*
* Since you apparently don't have an up-to-date list of the         *
* contents of the trace table, you use TDLSTSPT to survey           *
* all entries.                                                      *
*                                                                   *
* TDLSTSPT returns an array containing eight elements, each         *
* containing the transaction ID of an entry in the trace table      *
* for which tracing is TRUE.                                        *
*-------------------------------------------------------------------*
CALL 'TDLSTSPT' ...
*-------------------------------------------------------------------*
* You decide to turn tracing off for MYT3.                          *
*-------------------------------------------------------------------*
CALL 'TDSETSPT' ...(trace flag: OFF,
          trace option: 08,
          tran ID: MYT3)...
*-------------------------------------------------------------------*
* The operation succeeds;  the return code is OK.                   *
*-------------------------------------------------------------------*
```

The trace table now looks like this:

Table 5-3: Sample trace table (2)

Transaction ID	Tracing flag
MYT1	TRUE
MYT2	TRUE
MYT3	FALSE
MYT4	TRUE
MYT5	TRUE
MYT6	TRUE
MYT7	TRUE
MYT8	TRUE

Note The third position in the trace table is now considered empty.

When you try again to turn tracing on for MYT9, TDSETSPT moves it into the open position in the trace table.

```
*-------------------------------------------------------------------*
 * Try to enable tracing for MYT9.                                  *
 *-------------------------------------------------------------------*
 CALL 'TDSETSPT' ...(trace flag: ON,
            trace option: 01,
            tran ID: MYT9)
 .
 .
 .
```

The trace table now looks like this:

Table 5-4: Sample trace table (3)

Transaction ID	Tracing flag
MYT1	TRUE
MYT2	TRUE
MYT9	TRUE
MYT4	TRUE
MYT5	TRUE
MYT6	TRUE
MYT7	TRUE
MYT8	TRUE

Still later, you decide to turn on tracing for MYT2:

```
*-------------------------------------------------------------------*
 * Try to enable tracing for MYT2.                                  *
 *-------------------------------------------------------------------*
 CALL 'TDSETSPT' ... (trace flag: OFF,
            trace option: 01,
            tran ID: MYT2)...
 *-------------------------------------------------------------------*
 * The operation fails. You get a TDS DUPLICATE ENTRY return code, as*
 * tracing is already enabled for the transaction-no action needed.  *
 *-------------------------------------------------------------------*
```

Accounting

The Server Option allows you to record accounting information at the mainframe and at TRS. Mainframe-based accounting is independent of TRS-based accounting. For example, when the TRS accounting facility records packet is received, it is recording the number of packets sent from the mainframe to TRS. However, when the mainframe accounting facility records packet is received, it is recording the number of packets sent from TRS to the mainframe.

Accounting can be enabled at TRS, at the mainframe, or both. For information on TRS accounting, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*. This section describes accounting at the mainframe.

Note The mainframe accounting facility uses elapsed time.

Server Option accounting functions

To enable mainframe server accounting information, call TDSETACT in your Server Option program. TDSETACT begins recording when your program issues a TDACCEPT and continues until the program issues TDFREE. Use TDINFACT to learn whether accounting recording is enabled and the name of the accounting log file.

See the appropriate Mainframe Connect Server Option *Programmers Reference* for complete descriptions and examples of these functions. PL/1 and COBOL versions of this guide are available.

Accounting log

Under CICS, the Server Option accounting functions store information in a VSAM ESDS accounting log file. As installed, this file is named *SYTACCT1*. The Server Option appends accounting records to that file until it becomes full; all subsequent attempts to write accounting records to that file are rejected. To make room in the file, do *one* of the following:

- Archive or delete accounting records
- Change the name of the accounting log

- Change the underlying VSAM file assigned to this name

Accounting log layout

Table 5-5 shows the layout of the accounting log used under CICS.

Note The structures of the accounting log have the same layout in CICS as in IMS TM, although the log headers differ slightly.

Table 5-5: Accounting log layout

Position	Field name	Field type	Field description
1-2	acct-length	Unsigned 2-byte integer	Record length. The total length of this accounting record. (Maximum size of a CICS record is 256.)
3	acct-type	Unsigned byte	Type of record. For the accounting log, this type is always TDS-ACCT-REC.
4	acct-direction	Unsigned byte	Reserved for future use.
5-68			Structure containing the following:
	acct-key	Unsigned byte (max. length = 30)	user_id: Client's server login ID, from the login packet.
		Unsigned byte (length = 24)	trace_resid: Trace resource ID. In CICS, this keeps track of who is doing the tracing.
		Unsigned byte (length=8)	datetime: Date and time the SNA conversation or TCP/IP session started. (TDACCEPT was issued.)
		Unsigned 2-byte integer	uniquekey: Reserved for future use to ensure record has unique key.
69-82	acct-txp-name	Unsigned byte	Transaction name.
83-112	acct-server-id	Unsigned byte	TRS name. Name of the TRS sending the current request.
113-142	acct-conn-id	Unsigned byte	Connection ID. Name by which the connection is known to TRS.
143-144	Filler	Unsigned byte	Filler to allow next entries to be full words.
145-148	acct-tot-secs-wall	4-byte integer	Elapsed wall clock time, in seconds, during the SNA conversation or TCP/IP session.

Position	Field name	Field type	Field description
149-152	acct-tot-fracsecs-wall	Unsigned 4-byte integer	Elapsed wall clock time, in milliseconds during the SNA conversation or TCP/IP session.
153-156	acct-tot-secs-cpu	4-byte integer	CPU time used, in seconds during the SNA conversation or TCP/IP session.
157-160	acct-tot-fracsecs-cpu	Unsigned 4-byte integer	CPU time used, in milliseconds, during the SNA conversation or TCP/IP session.
161-164	acct-tot-sent-bytes	4-byte integer	Total number of TDS bytes sent during an SNA conversation or TCP/IP session.
165-168	acct-tot-sent-packets	4-byte integer	Total number of TDS packets sent during an SNA conversation or TCP/IP session.
169-172	acct-tot-sent-msgs	4-byte integer	Total number of TDS messages sent during an SNA conversation or TCP/IP session.
173-176	acct-tot-sent-rows	4-byte integer	Total number of TDS rows sent during an SNA conversation or TCP/IP session.
177-180	acct-tot-sent-requests	4-byte integer	Total number of RPCs or SQL requests sent during an SNA conversation or TCP/IP session. For the Server Option, this is always 0.
181-184	acct-tot-rcvd-bytes	4-byte integer	Total number of TDS bytes received during an SNA conversation or TCP/IP session.
185-188	acct-tot-rcvd-packets	4-byte integer	Total number of TDS packets received during an SNA conversation or TCP/IP session.
189-192	acct-tot-rcvd-msgs	4-byte integer	Total number of TDS messages received during an SNA conversation or TCP/IP session.
193-196	acct-tot-rcvd-rows	4-byte integer	Total number of TDS rows received during an SNA conversation or TCP/IP session. For the Server Option, this is always 0.
197-200	acct-tot-rcvd-requests	4-byte integer	Total number of RPCs or SQL requests received during an SNA conversation or TCP/IP session.

Position	Field name	Field type	Field description
201-204	acct-tot-rcvd-cancels	4-byte integer	Total number of Cancels or Attentions received during an SNA conversation or TCP/IP session.
205-208	acct-reserved1	4-byte integer	Reserved for future use.
209-212	acct-reserved2	4-byte integer	Reserved for future use.
213-216	acct-reserved3	4-byte integer	Reserved for future use.
217-220	acct-reserved4	4-byte integer	Reserved for future use.
221-236	acct_fill	Unsigned byte (length=36)	Filler, to fill out record to 256 bytes.

APPENDIX A

Customization Options

Overview

You can customize Sybase mainframe access components to meet the requirements at your site. The customization load module SYGWXCPH is a table created by assembling and linking five macros:

- SYGWMCST – a global customization macro.
- SYGWMCXL – a character set conversion macro.
- SYGWDRIV – specifies which dynamic network drivers are used at the site.
- SYGWHOST – provides mapping between Sybase Server names and TCP/IP addresses or host names.
- SYGWLKEY – a license key macro.

The SYGWXCPH table is shared by the Client Option and the Server Option.

Customizing global options (SYGWMCST)

SYGWMCST, one of the macros in table SYGWXCPH, provides options for customizing the Client Option and the Server Option. Some Server Option parameters are used only for customizing the DB2 UDB Option for CICS. You can customize SYGWMCST using the provided JCL member.

Table A-1 describes SYGWMCST parameters. Except where noted, these apply to both the Client Option for CICS and Server Option for CICS.

Table A-1: Complete list of SYGWMCST parameters

Parameter	Default	Format	Purpose
ACCESSCODE	blank	Up to 32 characters	Defines an access code, which is then compared to the access code supplied by Server Option programs using TDGETUSR. If the access codes do not match, the client password is not returned to the caller of Server Option programs using TDGETUSR. See the appropriate Mainframe Connect Server Option *Programmers Reference* for details on TDGETUSR.
ACCESSCODESW *(Server Option only)*	N	Y or N	Turns on/off access code comparison (see ACCESSCODE value). When ACCESSCODESW=N (default), the logged-in password is always returned to the caller of Server Option programs using TDGETUSR. When ACCESSCODESW=Y, the logged-in password is returned only if the access code passed to TDGETUSR matches the access code specified in SYGWMCST ACCESSCODE.
CHARSETSRV	iso_1	Up to 32 characters	Specifies the default character set that the Client Option or Server Option uses internally. The valid values are iso_1 and utf8. **Note** The value utf8 is valid only if USEIBMUNICODE=Y.
DEBUGSW	N	Y or N	Specifies whether or not debugging messages, used in troubleshooting, should be displayed in the system log.

Parameter	Default	Format	Purpose
DECPOINT *(Server Option only)*	'.'	Either a decimal point or comma delimited by single quotation marks	Decimal point indicator, used only with the DB2 UDB Option for CICS.
DEFLTPROTOCOL	TCP	TCP	Specifies the default network driver protocol.
DQUOTETRAN *(Server Option only)*	Y	Y or N	Used only with the DB2 UDB Option for CICS. Make this setting consistent with your DB2 configuration. When DQUOTETRAN=Y (default), double quotes are translated to single quotes in incoming SQL text. If you are using an ODBC driver, set DQUOTETRAN=N. **Note** If you are using double-byte or multi-byte characters for DB2 metadata, set DQUOTETRAN=N.
IMSLOGTYPE *(IMS TM only)*	A1	A value greater than or equal to A0	Specifies a log type. IMS TM reserves values less than A0.
LONGVARTRUNC	N	Y or N	Indicates whether to truncate LongVarChar and VarBinary. *For CICS only:* Coordinate this setting with the DirectConnect for z/OS Option TRS. If either this parameter or the TRS TruncateLV configuration property is set for truncation, truncation occurs. If you do not want truncation, set this parameter to N and make sure the TRS TruncateLV configuration property is set to No. See the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

Parameter	Default	Format	Purpose
MVSDDNAME *(IMS TM and MVS only)*	blank	From 1 to 8 characters	The DD name of the MVS Open Client and Open Server log file. If this parameter is left blank (the default), MVS transactions are not logged. If you enter a DD name of 1-8 characters, MVS transactions are logged. The name specified here must match a DD name specified in each MVS transaction profile job. MVSDDNAME must match a DD name specified in the JCL for one of the following: • An MVS job • An MVS started task • The MVS transaction profile (if run in an APPC initiator as a transaction)
NATLANGUAGESRV	us_english	Up to 32 characters	Designates the default national language used by the Client Option or Server Option. Also, see the CHARSETSRV property.
ROWLIMIT *(Server Option only)*	0 (zero)		Used only by the DB2 UDB Option for CICS. When ROWLIMIT=0, there is no limit to the number of rows that can be sent. ROWLIMIT=*n*, where *n* indicates the global limit of rows that can be sent.
USEIBMUNICODE	N	Y or N	Specifies whether or not Unicode support for a particular z/OS installation is enabled through the IBM conversion environment and services. • If USEIBMUNICODE=Y, IBM support is used for character set conversions. • If USEIBMUNICODE=N, conversion is accomplished through the product-supplied translation tables. **Note** If USEIBMUNICODE=Y, all character sets that are to be used at a particular site must have entries created with the SYGWMCXL macro. See "Using the IBM z/OS conversion environment and services" on page 51.

Using the IBM z/OS conversion environment and services

Unicode support in the Client Option and Server Option is based on Unicode support provided by IBM z/OS, including the conversion environment and conversion services. When the conversion environment and services are installed and set up, the Client Option and Server Option can convert character streams from one Coded Character Set Identifier (CCSID) to another. This functionality is provided in addition to the support for language and character sets offered in previous versions.

❖ **To install IBM Unicode support**

1 Create an *IMAGE* member in SYS1.PARMLIB using the CUNMIUTL utility.

2 Copy the *CUNIMG01* member from WORK.IMAGE to SYS1.PARMLIB.

3 Use this comand to load the *CUNIMG01* member into z/OS:

```
SET UNI=01
```

4 Use this command to display the current active image and the character set conversions defined for that image:

```
DISPLAY UNI, ALL
```

5 To enable Unicode support for the Client Option and Server Option, set USEIBMUNICODE=Y. The USEIBMUNICODE parameter is specified in the SYGWMCST macro in the SYGWXCPH customization module.

For more information on installing Unicode support for IBM z/OS, see "Support for Unicode Using Conversion Services" (SA22-7649-07).

Customizing mainframe character set conversion options (SYGWMCXL)

SYGWMCXL is the character set conversion macro in the SYGWXCPH table. The following considerations apply when using the SYGWMCXL macro:

- When Unicode support is disabled (USEIBMUNICODE=N) and the original translation method is used, SYGWMCXL can be used to override supplied SBCS translation tables or to define new SBCS translation tables.

- When Unicode support is enabled (USEIBMUNICODE=Y), SYGWMCXL is used to create definition entries for the character sets to be used in the Client Option or Server Option conversions at a particular installation. These entries are created in addition to system-generated entries.

Note All EBCDIC-to-ASCII and ASCII-to-EBCDIC translation for Client Option or the Server Option occurs on the mainframe.

Overriding the supplied SBCS translation tables

For SBCS, shipped character sets are called *predefined*, and the character sets you define are called *user-definable*.

Predefined character sets

Predefined SBCSs shipped with the product include:

SBCS	Definition
ascii_8	Default used for logins and for IBM cp1027 (code page 1027) support
cp437 (code page 437)	Used by IBM PCs
cp850 (code page 850)	IBM/Microsoft Multilingual Character Set, used by IBM PCs
iso_1 (ascii 0819)	International ISO standard, 8-bit character set for many systems, and the default for Adaptive Server Enterprise on several platforms
mac (Macintosh Roman)	Default used by Macintosh systems
roman8	Default Hewlett-Packard proprietary character set

Warning! Unpredictable failures can occur if the character set names are changed from lowercase to uppercase.

User-defined character sets

You can change all attributes for user character sets. The SBCS settings of the parameters for SYGWMCXL are:

Table A-2: SYGWMCXL parameters for SBCS

Parameter	Value
A2E	Optional ASCII-to-EBCDIC translate overrides
E2A	Optional EBCDIC-to-ASCII translate overrides
CHARSET	Name of the SBCS
CHARSETBYTES	S for SBCS
TYPE	Valid types: • INITIAL • ENTRY (default) • FINAL

If there is no override entry for a predefined character set, a default entry is generated with the appropriate translation tables and other attributes for that character set. A total of 99 character sets, including custom-generated character set entries, is supported.

The minimum translate customization entries are:

```
SYGWMCXL TYPE=INITIAL
SYGWMCXL TYPE=FINAL
```

These entries generate all of the predefined SBCSs.

Defining new SBCS translation tables

For SBCSs, you can modify the translation tables shipped with the product and create new translation tables with names you define.

Warning! Do not use the shipped table names for the tables you create.

If you create new tables for the Server Option in a three-tier environment, you must coordinate with the person responsible for the Sybase client. The client uses the names of the tables you create to issue logins to the DirectConnect for z/OS Option TRS.

When you finish customizing the SBCS translation tables, rebuild the SYGWXCPH module, and load the new module for your revisions to take effect. Instructions are provided in “Building a global customization module (SYGWXCPH)” on page 61.

Overriding defaults and creating new tables on the mainframe

The SYGWMCXL macro generates translation tables to convert between ASCII and EBCDIC character sets. Default translation tables are generated for the following ASCII character sets:

- ascii_8
- cp437
- cp850
- iso_1
- mac
- roman8

> **Note** Unpredictable failures can occur if the character set names are changed from lowercase to uppercase.

These default tables also provide the "base" for any character set changes or new tables you want to define. For details on the base translate tables, see Appendix B, "Translation Tables"

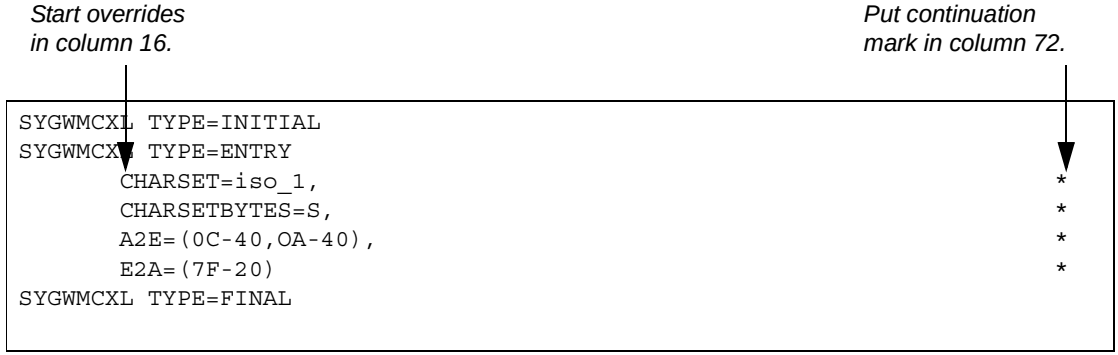
You can change all attributes for user character sets. An entry is added to the translate table, specifying the appropriate character set attributes. Two examples follow for overriding defaults.

Overriding ASCII-to-EBCDIC defaults

The first example, Figure A-1, shows how to use A2E and E2A macro parameters to override the ASCII-to-EBCDIC defaults. You can use uppercase or lowercase to define the parameters.

When you override the ASCII-to-EBCDIC defaults, the appropriate base table is picked up as a template for the character overrides or user-defined character sets, thus generating a default table. In Figure A-1, the client is using us_english, which is not predefined.

Figure A-1: Using A2E and E2A example



This example converts both of the following:

- ASCII form feeds (x'0C') and line feeds (x'0A') to EBCDIC spaces (x'40')
- EBCDIC DELs (x'7F') to ASCII space (x'20')

Creating a new table

The next example shows how to modify the default character set, iso_1, for Hebrew, creating a new table:

```
* These SYGWMCXL macro calls modify the iso_1 character set
* to Hebrew.
*
  SYGWMCXL        TYPE=INITIAL
  SYGWMCXL TYPE=ENTRY,                        *
     CHARSET=(unique_name),                   *
     CHARSETBYTES=S,                          *
     A2E=(E0-41,E1-42,E2-43,E3-44,E4-45,E5-46,E6-47,E7-48,E8-*
     49,E9-51,EA-52,EB-53,EC-54,ED-55,EE-56,EF-57,F0-58,F1-59*
     ,F2-62,F3-63,F4-64,F5-65,F6-66,F7-67,F8-68,F9-69,FA-71),*
     E2A=(41-E0,42-E1,43-E2,44-E3,45-E4,46-E5,47-E6,48-E7,49-*
     E8,51-E9,52-EA,53-EB,54-EC,55-ED,56-EE,47-EF,58-F0,59-F1*
     ,62-F2,63-F3,64-F4,65-F5,66-F6,67-F7,68-F8,69-F9,71-FA)
  SYGWMCXL TYPE=FINAL
*
* Assembler END is required.
*
END
```

For the CHARSET parameter, specify a unique name. This generates a new user-defined table. Provide the name to the appropriate person at the Sybase client site. The client login packet uses this name.

Defining new character set entries

In using the IBM Unicode conversion environment and services, the SYGWMCXL macro is used to create definition entries for all the character sets that will be used at a particular site and that are not already defined as system character sets. Table A-3 describes the parameters used in the SYGWMCXL macro to create a definition entry:

Table A-3: SYGWMCXL macro parameters

Parameter	Value
CHARSET	The name of the SBCS or DBCS character set.
CHARSET BYTES	An S to denote SBCS, or a D to denote DBCS.
CCSID	The CCSID for the character set.
CHARSETYPE	The type of character set. A denotes ASCII, and E denotes EBCDIC.
CHARSIZE	The maximum length of a character, from 1 to 4 bytes.
PAD	The padding character. The value of this parameter depends on the character set type. For ASCII, the padding character is 20. For EBCDIC, the padding character is 40.

Note If USEIBMUNICODE=Y, all character sets that are to be used at a particular site must have entries created with the SYGWMCXL macro.

The following examples illustrate definitions for Russian and Japanese EBCDIC character sets, which are code pages 1025 and 939, respectively.

Example: code page 1025

```
SYGWMCXL TYPE=ENTRY,
      CHARSET=Russian,CHARSETBYTES=S,
      CCSID=1025,CHARTYPE=E,CHARSIZE=1,PAD=40
```

Example: code page 939

```
SYGWMCXL TYPE=ENTRY,
      CHARSET=cp939,CHARSETBYTES=D,
      CCSID=939,CHARTYPE=E,CHARSIZE=2,PAD=40
```

In addition to the default ASCII SBCS translation tables, these names are used to generate system entries for ASCII DBCS character sets:

- *sjis* – Japanese code page cp943 or cp932

- *eucjis* – Japanese code page cp33722
- *cp950* – traditional Chinese Big5 or cp950
- *cp936* – simplified Chinese GBK or cp936

If you use any of these names, you do not need to create a new definition.

Customizing dynamic network drivers (SYGWDRIV)

SYGWDRIV, a macro in the SYGWXCPH table, defines the dynamic network drivers for the Client Option or the Server Option.

Note If you are using a TCP/IP driver, you must also configure the SYGWHOST macro.

CICS network drivers

Table A-4 shows the default drivers that are shipped with the Client Option or Server Option, depending on the environment:

Table A-4: CICS network drivers

Driver	Load module name	Comments
LU 6.2	LU62CICS	Uses CICS LU 6.2 API
IBM TCP/IP	TCPCICS	Uses IBM EZACICAL API
CPIC	CPICCICS	Uses CICS CPIC Support

The CICS JCL member *IxHOST* contains these macro definitions, which set up support for all three network drivers:

```
        SYGWDRIV TYPE=INITIAL
*
        SYGWDRIV TYPE=ENTRY,ENV=CICS,NETD=LU62
        SYGWDRIV TYPE=ENTRY,ENV=CICS,NETD=CPIC
        SYGWDRIV TYPE=ENTRY,ENV=CICS,NETD=TCP
*
        SYGWDRIV TYPE=FINAL
```

Using the CPI-C CICS network driver

If you use the CPI-C CICS driver, you must use CEDA to define an entry in the CICS PARTNER Table. Due to an IBM requirement, each Partner entry must be exactly 8 characters in length and use A-Z, 0-9. If your actual server name is not 8 characters, put an alias for it in your *interfaces* file. For example:

Figure A-2: CEDA window

```
OBJECT CHARACTERISTICS                              CICS RELEASE = 0410

      CEDA View PARTner( MYSERVER )
            PARTner      : MYSERVER
            Group        : GROUP42
            Description  : SIDE INFO ENTRY TO GET TO mymcg
      REMOTE LU NAME
            NETName      : U6T42P0M
            NETWork      :
      SESSION PROPERTIES
            Profile      : SYOCPROF
      REMOTE TP NAME
            Tpname       :
            Xtpname      : 94A8948387



                                              SYSID=CICS APPLID=CICS41

PF1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Enter the PARTner and Remote TP name field values as follows:

- PARTner – This must be *exactly* 8 characters long. An alias for the 8-character name should be added to the *interfaces* file if necessary.
- Remote TP name – If the name of your server is in uppercase, enter it in the Tpname field. If the name of your server is in lowercase, enter the EBCDIC hexadecimal name in the Xtpname field.

Note If you enter a lowercase name in the Tpname field, CEDA changes it to uppercase and an erroneous entry is passed.

Customizing the TCP/IP driver (SYGWHOST)

The SYGWHOST macro is part of the SYGWXCPH global customization module. This macro is used only for the Client Option in connections from the mainframe to other applications. It is required only if you are using a TCP/IP driver, in which case you must configure SYGWHOST to define the mapping between Sybase server names and TCP/IP addresses or host names. Do not depend on the default shipped with the installation to work in your environment.

Macro formats

There are three macro formats: TYPE=INITIAL, TYPE=ENTRY, and TYPE=FINAL.

Note For the Server Option, only the TYPE=INITIAL and TYPE=FINAL macros are required. For the Client Option, only the TYPE=ENTRY macro is required.

TYPE=INITIAL

The format of TYPE=INITIAL is:

```
SYGWHOST TYPE=INITIAL
```

TYPE=ENTRY

The format of TYPE=ENTRY is:

```
SYGWHOST TYPE=ENTRY
    IBMTCPADRSPCNAME=&&TCP,
    LISTENER=(LAN,CICS,IMS)
    LSTNPORT=99999,
    SERVERNAME=sybase10,
    HOSTNAME=myhost
```

TYPE=FINAL

The format of TYPE=FINAL is:

```
SYGWHOST TYPE=FINAL
```

Macro parameters

There are six parameters in the SYGWHOST macro:

Parameter	Definition
HOSTNAME	The name of the host on which the Sybase server resides. The maximum length of the host name is 24 characters. If a value is provided for the IPADDR parameter, the HOSTNAME parameter is ignored, and no DNS search is performed.
IBMTCPADDRSPACE	Designates the name of the IBM TCP/IP address space. This parameter can be specified as either of the following: • A hard-coded value of up to 8 characters. • A system symbolic name. System symbolic names are defined in the IEASYMxx PARMLIB member and are limited to seven characters preceded by "&&". For example, the symbolic name "SYBTCP" would be designated as follows: `IBMTCPADDRSPACE=&&SYBTCP` Symbolic names allow the use of a common SYGWXCPH configuration module across multiple LPARs, even if each LPAR has a different TCP address space name. The default address space name is TCPIP.
IPADDR	The IP address of the host on which the Sybase server resides. If a value is provided for this parameter, the HOSTNAME parameter is ignored.
LISTENER	One of the following: • *LAN* if the listen port is for a LAN-based server (default) • *CICS* if the listen port is for an CICS Server Option listener • *IMS* if the listen port is for an IMS TM Server Option listener
LSTNPORT	The listen port of the server specified by SERVERNAME.
SERVERNAME	The 1-30 byte name of a Sybase server.

Defining license keys (SYGWLKEY)

The SYGWLKEY macro is part of the SYGWXCPH global customization module. It is used to define the customer license key that is verified at runtime.

There are two parameters in this macro:

Parameter	Definition
PRODUCT	The product related to the license key, either the Client Option, the Server Option, or the DB2 UDB Option. Valid values are OCC, OSC, or DB2.
KEY	Defines the license key given for a product. The license key is a 23-character numeric value.

This example of SYGWLKEY defines license keys for four Mainframe Connect options in the order they are listed: Client Option for CICS, Server Option for CICS, Server Option for IMS and MVS, and DB2 UDB Option for CICS:

```
SYGWLKEY TYPE=INITIAL
SYGWLKEY TYPE=ENTRY,PRODUCT=OCC,KEY=19320-00000-10$*#-#19$B
SYGWLKEY TYPE=ENTRY,PRODUCT=OSC,KEY=19300-00000-10E2G-4K##6
SYGWLKEY TYPE=ENTRY,PRODUCT=OSC,KEY=19315-00000-2$#0$-4A#49
SYGWLKEY TYPE=ENTRY,PRODUCT=DB2,KEY=26875-00239-2$$$A-#AR#H
SYGWLKEY TYPE=FINAL
```

Building a global customization module (SYGWXCPH)

The installation process in Chapter 3, “Installation and Configuration,” creates the *IxTCP* job (where *x* is an integer that denotes the order in which the job is to be run in the overall sequence of jobs). The *IxTCP* job can be run to create a basic version of the SYGWXCPH global customization module, which contains these macros:

- SYGWMCST
- SYGWMCXL
- SYGWDRIV
- SYGWHOST
- SYGWLKEY
- TDSGLOB, a relocatable object module

APPENDIX B

Translation Tables

Note This appendix shows the default settings for the ASCII-EBCDIC and EBCDIC-ASCII translation tables before any user overrides.

Understanding the ASCII-EBCDIC and EBCDIC-ASCII translation tables

Note The translation tables shown here are used in date conversion only if Unicode support is disabled and USEIBMUNICODE=N.

The four pairs of default, or “base,” tables are:

- ASCII_8
- ISO_1
- cp437
- cp 850

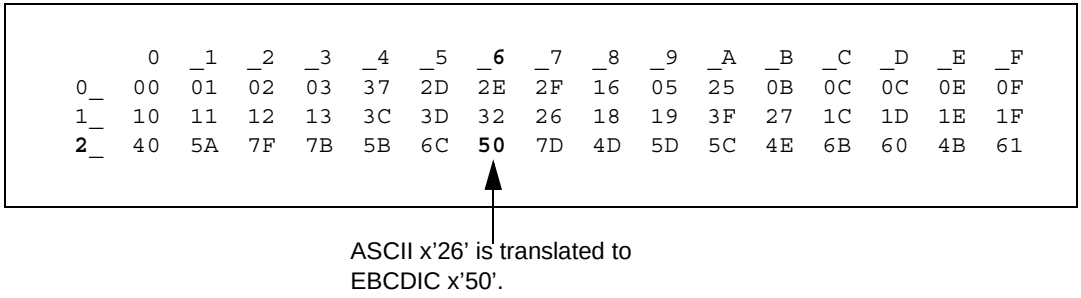
Each pair includes a table for ASCII-to-EBCDIC translation, and one for EBCDIC-to-ASCII translation.

Note As supplied, all ASCII character sets translate to and from EBCDIC code page 500 (iso_1) on the mainframe by default.

For the ASCII-to-EBCDIC tables, find the leftmost hexadecimal ASCII digit to the left of the table as a digit followed by an underscore. Find the rightmost hexadecimal ASCII digit on top of the table as a digit preceded by an underscore.

Here is an example from the default table in the section "ASCII_8, ASCII-to-EBCDIC translation table" on page 66.

Figure B-1: Example from the ASCII_8, ASCII-to-EBCDIC translation table



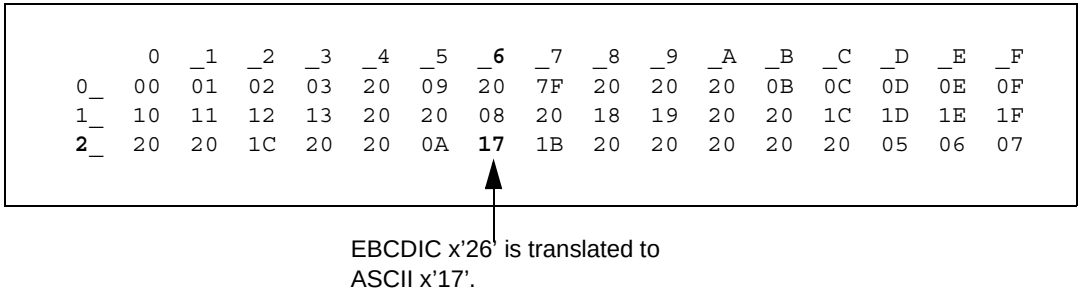
	0	_1	_2	_3	_4	_5	**_6**	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	37	2D	2E	2F	16	05	25	0B	0C	0C	0E	0F
1_	10	11	12	13	3C	3D	32	26	18	19	3F	27	1C	1D	1E	1F
2_	40	5A	7F	7B	5B	6C	**50**	7D	4D	5D	5C	4E	6B	60	4B	61

To locate ASCII x'26', find row 2_ to the left of the table, and proceed along that row to the column headed by _6. At the intersection is x'50'. Therefore, ASCII x'26' is translated to EBCDIC x'50'.

For the EBCDIC-to-ASCII tables, find the leftmost hexadecimal EBCDIC digit to the left of the table as a digit followed by an underscore. Find the rightmost hexadecimal EBCDIC digit on top of the table as a digit preceded by an underscore.

Here is an example from the default table in the section "ASCII_8, ASCII-to-EBCDIC translation table" on page 66.

Figure B-2: Example from the ASCII_8, EBCDIC-to-ASCII translation table



To locate EBCDIC x'26', find row 2_on the left side of the table; then proceed along that row to the column headed by _6. At the intersection is x'17'. Therefore, EBCDIC x'26' is translated to ASCII x'17'.

Warning! If you create a new table from a default table, give the new table a unique name and coordinate with the appropriate person at the Sybase client site. The client can use the name to issue logins to TRS.

Default ASCII_8 translation tables

The ASCII-to-EBCDIC translation tables in this section are the base tables for these predefined system SBCSs:

- ascii_8
- roman8
- mac
- ibmascii

Use these tables as the base ASCII-to-EBCDIC translation table for user-definable character sets.

ASCII_8, ASCII-to-EBCDIC translation table

Figure B-3: ASCII_8, ASCII-to-EBCDIC translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	37	2D	2E	2F	16	05	25	0B	0C	0D	0E	0F
1_	10	11	12	13	3C	3D	32	26	18	19	3F	27	1C	1D	1E	1F
2_	40	5A	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3_	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4_	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5_	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	AD	E0	BD	5F	6D
6_	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7_	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	8B	6A	9B	A1	07
8_	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
9_	90	91	92	93	94	95	96	97	98	99	9A	4A	9C	9D	9E	9F
A_	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	5F	AB	AC	AD	AE	AF
B_	B0	B1	B2	4F	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BC
C_	AB	C1	C2	C3	BF	8F	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
D_	D0	D1	D2	D3	D4	D5	D6	D7	D8	BB	AC	DB	DC	DD	DE	DF
E_	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
F_	F0	9E	AE	8C	F4	F5	F6	F7	A1	AF	FA	FB	FC	FD	9F	FF

ASCII_8, EBCDIC-to-ASCII translation table

Figure B-4: ASCII_8, EBCDIC-to-ASCII translation table

	0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	20	09	20	7F	20	20	20	0B	0C	0D	0E	0F
1_	10	11	12	13	20	20	08	20	18	19	20	20	1C	1D	1E	1F
2_	20	20	1C	20	20	0A	17	1B	20	20	20	20	20	05	06	07
3_	20	20	16	20	20	20	20	04	20	20	20	20	14	15	20	1A
4_	20	20	20	20	20	20	20	20	20	20	9B	2E	3C	28	2B	B3
5_	26	20	20	20	20	20	20	20	20	20	21	24	2A	29	3B	AA
6_	2D	2F	20	20	20	20	20	20	20	20	7C	2C	25	5F	3E	3F
7_	20	20	20	20	20	20	20	20	20	60	3A	23	40	27	3D	22
8_	20	61	62	63	64	65	66	67	68	69	20	7B	F3	20	20	C5
9_	20	6A	6B	6C	6D	6E	6F	70	71	72	20	7D	20	20	F1	FE
A_	20	7E	73	74	75	76	77	78	79	7A	20	C0	DA	5B	F2	F9
B_	20	20	20	20	20	20	20	20	20	20	20	D9	BF	5D	20	C4
C_	7B	41	42	43	44	45	46	47	48	49	20	20	20	20	20	20
D_	7D	4A	4B	4C	4D	4E	4F	50	51	52	20	20	20	20	20	20
E_	5C	20	53	54	55	56	57	58	59	5A	20	20	20	20	20	20
F_	30	31	32	33	34	35	36	37	38	39	20	20	20	20	20	20

Default ISO_1 translation tables

The ASCII-to-EBCDIC translation tables in this section are the base table for the predefined system iso_1 character set.

ISO_1 ASCII-to-EBCDIC translation table

Figure B-5: ISO_1 ASCII-to-EBCDIC translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	37	2D	2E	2F	16	05	25	0B	0C	0D	0E	0F
1_	10	11	12	13	3C	3D	32	26	18	19	3F	27	1C	1D	1E	1F
2_	40	4F	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3_	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4_	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5_	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	4A	E0	5A	5F	6D
6_	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7_	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	C0	BB	D0	A1	07
8_	20	21	22	23	24	15	06	17	28	29	2A	2B	2C	09	0A	1B
9_	30	31	1A	33	34	35	36	08	38	39	3A	3B	04	14	3E	FF
A_	41	AA	B0	B1	9F	B2	6A	B5	BD	B4	9A	6A	BA	CA	AF	BC
B_	90	8F	EA	FA	BE	A0	B6	B3	9A	DA	9B	8B	B7	C7	B9	AB
C_	64	65	62	66	63	67	9E	69	74	71	72	73	78	75	76	77
D_	AC	69	ED	EE	EB	EF	EC	BF	80	FD	FE	FB	FC	AD	AE	59
E_	44	45	42	46	43	47	9C	48	54	51	52	53	58	55	56	57
F_	8C	49	CD	CE	CB	CF	CC	E1	70	DD	DE	DB	DC	8D	8E	DF

ISO_1 EBCDIC-to-ASCII translation table

Figure B-6: ISO_1 EBCDIC-to-ASCII translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	9C	09	86	7F	97	8D	8E	0B	0C	0D	0E	0F
1_	10	11	12	13	9D	85	08	87	18	19	92	8F	1C	1D	1E	1F
2_	80	81	82	83	84	0A	17	1B	88	89	8A	8B	8C	05	06	07
3_	90	91	16	93	94	95	96	04	98	99	9A	9B	14	15	9E	1A
4_	20	A0	E2	E4	E0	E1	E3	E5	E7	F1	5B	2E	3C	28	2B	21
5_	26	E9	EA	EB	E8	ED	EE	EF	EC	DF	5D	24	2A	29	3B	5E
6_	2D	2F	C2	C4	C0	C1	C3	C5	C7	D1	A6	2C	25	5F	3E	3F
7_	F8	C9	CA	CB	C8	CD	CE	CF	CC	60	3A	23	40	27	3D	22
8_	D8	61	62	63	64	65	66	67	68	69	AB	BB	F0	FD	FE	B1
9_	B0	6A	6B	6C	6D	6E	6F	70	71	72	AA	BA	E6	B8	C6	A4
A_	B5	7E	73	74	75	76	77	78	79	7A	A1	BF	D0	DD	DE	AE
B_	A2	A3	A5	B7	A9	A7	B6	BC	BD	BE	AC	7C	AF	A8	B4	D7
C_	7B	41	42	43	44	45	46	47	48	49	AD	F4	F6	F2	F3	F5
D_	7D	4A	4B	4C	4D	4E	4F	50	51	52	B9	FB	FC	F9	FA	FF
E_	5C	F7	53	54	55	56	57	58	59	5A	B2	D4	D6	D2	D3	D5
F_	30	31	32	33	34	35	36	37	38	39	B3	DB	DC	D9	DA	9F

Default cp437 (code page 437) translation tables

The ASCII-to-EBCDIC translation tables in this section are the base tables for the predefined system cp 437 (code page 437) character set.

cp437 ASCII-to-EBCDIC translation table

Figure B-7: cp437 ASCII-to-EBCDIC translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	37	2D	2E	2F	16	05	25	0B	0C	0D	0E	0F
1_	10	11	12	13	B6	B5	32	26	18	19	1C	27	07	1D	1E	1F
2_	40	4F	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3_	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4_	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5_	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	4A	E0	5A	5F	6D
6_	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7_	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	C0	BB	D0	A1	3F
8_	68	DC	51	42	43	44	47	48	52	53	54	57	56	58	63	67
9_	71	9C	9E	CB	CC	CD	DB	DD	DF	EC	FC	B0	B1	B2	3E	B4
A_	45	55	CE	DE	49	69	9A	9B	AB	9F	BA	B8	B7	AA	8A	8B
B_	3C	3D	62	6A	64	65	66	20	21	22	70	23	72	73	74	BE
C_	76	77	78	80	24	15	8C	8D	8E	FF	06	17	28	29	9D	2A
D_	2B	2C	09	0A	AC	AD	AE	AF	1B	30	31	FA	1A	33	34	35
E_	36	59	08	38	BC	39	A0	BF	CA	3A	FE	3B	04	CF	DA	14
F_	EE	8F	46	75	FD	EB	E1	ED	90	EF	B3	FB	B9	EA	BD	41

cp437 EBCDIC-to-ASCII translation table

Figure B-8: cp437 EBCDIC-to-ASCII translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	EC	09	CA	1C	E2	D2	D3	0B	0C	0D	0E	0F
1_	10	11	12	13	EF	C5	08	CB	18	19	DC	D8	1A	1D	1E	1F
2_	B7	B8	B9	BB	C4	0A	17	1B	CC	CD	CF	D0	D1	05	06	07
3_	D9	DA	16	DD	DE	DF	E0	04	E3	E5	E9	EB	B0	B1	9E	7F
4_	20	FF	83	84	85	A0	F2	86	87	A4	5B	2E	3C	28	2B	21
5_	26	82	88	89	8A	A1	8C	8B	8D	E1	5D	24	2A	29	3B	5E
6_	2D	2F	B2	8E	B4	B5	B6	8F	80	A5	B3	2C	25	5F	3E	3F
7_	BA	90	BC	BD	BE	F3	C0	C1	C2	60	3A	23	40	27	3D	22
8_	C3	61	62	63	64	65	66	67	68	69	AE	AF	C6	C7	C8	F1
9_	F8	6A	6B	6C	6D	6E	6F	70	71	72	A6	A7	91	CE	92	A9
A_	E6	7E	73	74	75	76	77	78	79	7A	AD	A8	D4	D5	D6	D7
B_	9B	9C	9D	FA	9F	15	14	AC	AB	FC	AA	7C	E4	FE	BF	E7
C_	7B	41	42	43	44	45	46	47	48	49	E8	93	94	95	A2	ED
D_	7D	4A	4B	4C	4D	4E	4F	50	51	52	EE	96	81	97	A3	98
E_	5C	F6	53	54	55	56	57	58	59	5A	FD	F5	99	F7	F0	F9
F_	30	31	32	33	34	35	36	37	38	39	DB	FB	9A	F4	EA	C9

Default cp850 (code page 850) translation tables

The EBCDIC-to-ASCII translation tables in this section are the base tables for the predefined system cp 850 (code page 850) character set.

cp850 ASCII-to-EBCDIC translation table

Figure B-9: cp850 ASCII-to-EBCDIC translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	37	2D	2E	2F	16	05	25	0B	0C	0D	0E	0F
1_	10	11	12	13	3C	3D	32	26	18	19	1C	27	07	1D	1E	1F
2_	40	4F	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3_	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4_	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5_	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	4A	E0	5A	5F	6D
6_	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7_	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	C0	BB	D0	A1	3F
8_	68	DC	51	42	43	44	47	48	52	53	54	57	56	58	63	67
9_	71	9C	9E	CB	CC	CD	DB	DD	DF	EC	FC	70	B1	80	BF	FF
A_	45	55	CE	DE	49	69	9A	9B	AB	AF	BA	B8	B7	AA	8A	8B
B_	2B	2C	09	21	28	65	62	64	B4	38	31	34	33	B0	B2	24
C_	22	17	29	06	20	2A	46	66	1A	35	08	39	36	30	3A	9F
D_	8C	AC	72	73	74	0A	75	76	77	23	15	14	04	6A	78	3B
E_	EE	59	EB	ED	CF	EF	A0	8E	AE	FE	FB	FD	8D	AD	BC	BE
F_	CA	8F	1B	B9	B6	B5	E1	9D	90	BD	B3	DA	FA	EA	3E	41

cp850 EBCDIC-to-ASCII translation table

Figure B-10: cp850 EBCDIC-to-ASCII translation table

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	00	01	02	03	DC	09	C3	1C	CA	B2	D5	0B	0C	0D	0E	0F
1_	10	11	12	13	DB	DA	08	C1	18	19	C8	F2	1A	1D	1E	1F
2_	C4	B3	C0	D9	BF	0A	17	1B	B4	C2	C5	B0	B1	05	06	07
3_	CD	BA	16	BC	BB	C9	CC	04	B9	CB	CE	DF	14	15	FE	7F
4_	20	FF	83	84	85	A0	C6	86	87	A4	5B	2E	3C	28	2B	21
5_	26	82	88	89	8A	A1	8C	8B	8D	E1	5D	24	2A	29	3B	5E
6_	2D	2F	B6	8E	B7	B5	C7	8F	80	A5	DD	2C	25	5F	3E	3F
7_	9B	90	D2	D3	D4	D6	D7	D8	DE	60	3A	23	40	27	3D	22
8_	9D	61	62	63	64	65	66	67	68	69	AE	AF	D0	EC	E7	F1
9_	F8	6A	6B	6C	6D	6E	6F	70	71	72	A6	A7	91	F7	92	CF
A_	E6	7E	73	74	75	76	77	78	79	7A	AD	A8	D1	ED	E8	A9
B_	BD	9C	BE	FA	B8	F5	F4	AC	AB	F3	AA	7C	EE	F9	EF	9E
C_	7B	41	42	43	44	45	46	47	48	49	F0	93	94	95	A2	E4
D_	7D	4A	4B	4C	4D	4E	4F	50	51	52	FB	96	81	97	A3	98
E_	5C	F6	53	54	55	56	57	58	59	5A	FD	E2	99	E3	E0	E5
F_	30	31	32	33	34	35	36	37	38	39	FC	EA	9A	EB	E9	9F

APPENDIX C

Setting Up the CICS Sockets Interface

Understanding the CICS sockets interface

Mainframe Connect Server Option for CICS contains modules for implementing IBM CICS sockets interface. Before running this option, verify that a CICS region has the IBM CICS sockets interface installed and configured. For details, refer to *z/OS Communications Server: IP CICS Sockets Guide*.

Using the IBM CICS sockets interface and the Sybase TCP/IP listener for Mainframe Connect Server Option for CICS requires:

- Installing IBM and Sybase RDO definitions for the required programs and transactions.
- Concatenating the sockets load library.
- Configuring listeners. You can install any number of Sybase TCP/IP listeners and run them in a single CICS region. You can also configure each listener differently.
- Enabling CICS sockets interface automatically or by using the EZAO command.

The IBM CICS sockets interface uses IBM TRUE, which eliminates compatibility issues and provides some advantages:

- An individual listener can handle a socket pool of up to 1000 sockets.

- A new listener can be added without requiring system intervention.
- An individual listener can be used as both gateway and gatewayless.

Installing and configuring the CICS sockets interface

❖ **To install and configure the CICS sockets interface**

1 Add the IBM CICS sockets RDO entries to your CICS region. You can find input for DFHCSDUP in *tcphlq.SEZAINST(EZACICCT)*, where *tcphlq* refers to the high-level qualifier of your system's IBM TCP/IP configuration.

Note The Sybase listener program (SYBLSTNR) must run in the CICS key and have concurrency set to THREADSAFE. The listener TRANID (SY01) must have TASKDATALOC=ANY and TASKDATAKEY=CICS. When you add listener definitions, Sybase suggests that you copy the SY01 transaction definition.

---

2 Define the EZACONFG data set to contain the configuration data for CICS sockets and load the system configuration parameters. You can find IBM-supplied JCL for this in *tcphlq.SEZAINST(EZACICFG)*.

3 Add *tcphlq.SEZATCP* into the DFHRPL concatenation for the CICS region.

4 Add the configuration information for the first Sybase listener (SY01) into the EZACONFG data set, using an online transaction or the IBM EZACICD utility, as shown:

```
EZACICD TYPE=LISTENER,  Create Listener Record       X
   APPLID=CICSDEV1  APPLID of CICS                   X
   TRANID=SY01,   Use transaction ID                 X
   PORT=3044,     Use port number 3044               X
   BACKLOG=40,    Set backlog value to 40            X
   ACCTIME=30,    Set timeout value to 30 seconds    X
   NUMSOCK=100,   Support 99 concurrent connections  X
   SECEXIT=SYBTPSEC,                                 X
   USERID=DFHCICS,                                   X
   IMMED=YES        Start listener immediately
```

For more information about EZACICD, refer to the *z/OS Communications Server: IP CICS Sockets Guide*.

The Sybase TCP/IP listener uses the following IBM parameters:

APPLID	The CICS region APPLID where the listener is run.
TRANID	The transaction ID defined for the listener, which must have its own unique transaction ID. Sybase recommends using SY01, SY02, SY03 and so on.
PORT	The listener port number.
IMMEDIATE	The listener starts automatically when CICS sockets is started.
BACKLOG	Backlog value for the listener.
NUMSOCK	The size of the listener's socket pool ("Max Sockets").
ACCTIME	The time-out value of the accept select logic. The listener normally waits until action is pending for any of the sockets that it is currently managing. After processing pending actions, it also checks to see if a request is pending to shut down the listener. If the ACCTIME value is reached, the listener checks for pending shutdown requests.
SECEXIT	The Sybase security extensions configuration module name. (See "Customizing the SYBTPSEC configuration module" on page 77.)
USERID	The user ID that starts the listener. The listener issues the EXTRACT EXIT and SET FILE commands, and starts transactions with a surrogate user ID. This user ID must have authority to use EXTRACT EXIT and SET FILE commands and the authority to start transactions.

5 Assemble and link the SYBTPSEC configuration module.

Customizing the SYBTPSEC configuration module

You can find the JCL used to assemble and link the SYBTPSEC configuration module in *OSC150.CICS.JCL(IxTPSEC)*.

This table lists the macro fields, their defaults, and their meanings:

SECURITY	Security verification: Y – The user ID and password are verified when a language request or RPC is started. For gatewayless connections, the connection does occur, but a security error can occur when the first request is processed. Invalid user IDs are then rejected. H (default) – The user ID and password are verified immediately when a gatewayless connection is established. **Note** There is no difference between SEC=Y and SEC=H when you are running transactions from a gateway. When running gatewayless, SEC=Y emulates a gateway transaction by not returning any security errors until the client executes a language request or RPC. The setting SEC=H returns all user ID and password errors at connect time and drops the connection, as do ASE servers. However, there are rare instances in which the interaction of RACF and CICS prevents a user ID or password error from being returned to a gatewayless client when SEC=Y is used. When this occurs, the gatewayless handler transaction SYSH ends abnormally without closing the socket. Therefore, Sybase suggests that you use SEC=H setting. U – No password verification is performed. CICS assumes that the incoming user ID is correct and has the authority to run. **Note** For this setting, CICS versions prior to CICS/TS 3.1 cannot detect if a user ID is revoked. T – Use CICS Terminal Security. This setting results in additional transaction processing that may reduce the listener efficiency. N – No user ID and password verification occurs.
GWTRAN	The handler transaction called for gateway connections. Default is SYGH.
GWLTRAN	The handler called for gatewayless transactions. Default is SYSH.
TERMON	The sign-on transaction used for terminal security. Default is SYSO.
TERMOFF	The sign off transaction used for terminal security. Default is SYSF.
PING	The transaction used for SYBPING. Default is SYPG.
PWTRAN	The transaction ID used by the SYBPEM (password change) RPC. Default is SYPM.

If you require different listeners to have different parameters, you must assemble and link the SYBTPSEC macro under different configuration module names. When configuring a listener, you set the value of the SECEXIT parameter to a specific configuration module name. You also must add an RDO program definition (by copying that of SYBTPSEC) for each new name used.

CICS sockets interface control

The following commands control the CICS sockets interface:

- EZAC for configuration
- EZAO for control

Use EZAC to configure listeners. Each listener is identified by its transaction ID. The following example shows the output of the EZAC,DISplay,LISTENER command:

```
APPLID ===> CICSDEV1   APPLID of CICS System
TRANID ===> SY01       Transaction Name of Listener
PORT ===> 03044        Port Number of Listener
AF ===> INET           Listener Address Family
IMMEDIATE ===> NO      Immediate Startup   Yes|No
BACKLOG ===> 020       Backlog Value for Listener
NUMSOCK ===> 100       Number of Sockets in Listener
ACCTIME ===> 060       Timeout Value for ACCEPT
GIVTIME ===> 000       Timeout Value for GIVESOCKET
REATIME ===> 000       Timeout Value for READ
MINMSGL ===> 004       Minimum Message Length
TRANTRN ===> YES       Translate TRNID     Yes|No
TRANUSR ===> YES       Translate User Data Yes|No
SECEXIT ===> SYBTPSCY  Name of Security Exit
GETTID  ===> NO        Get AT-TLS ID  (YES|NO)
USERID  ===> DFHCICS   Listener User ID
WLM group 1 ===>       Workload Manager Group Name 1
WLM group 2 ===>       Workload Manager Group Name 2
WLM group 3 ===>        Workload Manager Group Name 3
```

Use the EZAO command to start or stop the entire sockets interface for the region, or to start and stop individual listeners.

If you want the CICS sockets interface to start automatically when CICS is initialized, you must add the EZACIC20 program to the second stage of the start-up PLT and the first stage of the shutdown PLT. Any sockets defined with IMMEDIATE=YES are automatically started.

The CICS sockets interface creates the required work areas for each listener only at start-up. If a listener is created while the interface is running, the interface must be stopped and restarted using EZAO STOP CICS and EZAO START CICS. If not, the Sybase listener ends abnormally with code SB01, indicating that the required work area does not exist. Changes to existing listeners (such as port numbers, backlog, and so on) require only stopping and starting that listener using EZAO STOP LIST(SYxx) and EZAO START LIST(SYxx).

For detailed information about the EZAC and EZAO commands, see the *z/OS Communications Server: IP CICS Sockets Guide*.

APPENDIX D

Setting Up Secure Sockets Layer Protocol

Understanding Secure Sockets Layer

Mainframe Connect Client and Server Options for CICS support Secure Sockets Layer (SSL) session-based security. SSL is the standard for securing the transmission of sensitive information (such as credit card numbers, stock trades, and banking transactions) over the Internet. These sections describe the SSL protocol and how it works.

Description of features

SSL provides these features:

- Authentication for clients and servers, with practical emphasis on the server
- Data confidentiality (encryption)
- Verification that a transaction was sent by the client and that the identical transaction was received by the server

To provide efficient authentication and encryption, SSL combines private-key and public-key technologies.

Public-key cryptography (asymmetric)

Public-key (asymmetric) cryptography is based on the Public Key Infrastructure (PKI) method of encryption, in which two different keys are used for encrypting and decrypting operations: one is public, the other is private. This means that an operation encrypted by one key can only be decrypted by the other key. The result is that public keys can be seen by all, yet privacy is still possible.

The highly-used RSA algorithm works for both encryption and decryption operations. This method solves critical key exchange issue, but the algorithms it uses require large key sizes and often result in slow CPU-bound operations.

Private-key cryptography (symmetric)

In private-key (symmetric) cryptography, the sender and receiver use the same key for both encryption and decryption operations on the same data. Key size is very important: the longer the key, the stronger it is. Currently, 1024, 2048 is the recommended length. The private-key (symmetric) method is 1000 times faster than the public-key (asymmetric) method.

How SSL provides security

This section describes how SSL provides and signs certificates to provide security.

Authentication and encryption

SSL starts with a “handshake,” during which the client authenticates the server, and the server optionally authenticates the client. Handshake negotiations are based on the public-key cryptography: The client and server agree on how to encrypt and decrypt data, such as using cipher suites and session keys. Also, the format to transmit encrypted data is defined in the handshake.

Certificate Authorities (CA)

When securing communications, both client and server use X.509 certificates. The client must verify the server's certificate based on the certificate of the Certificate Authority (CA) that signed the certificate or based on a self-signed certificate from the server. (The client verification is optional.) Then, the client and the server use the negotiated session keys and begin encrypted communication, using Private Key cryptography.

the main fields in a X.509 V3 certificate are:

- Issuer
- Subject fields (an X.500 Distinguished Name, commonName field)
- Algorithm identifier
- Subject's public key
- Period of validity

- Extensions
- Digital signature

Generating a certificate

The process of generating a certificate involves these basic steps:

1 Generate the public-private key pair.

2 Store the private key securely (as password-encrypted).

3 Generate a certificate signing request (CSR) in PKCS#10 format for the server certificate.

4 Present the CSR to the CA.

5 Receive the signed certificate from the CA.

6 Store the certificate.

You can use different tools to generate test (self-signed) certificates.

Setting up SSL in Sybase products

As of version 12.5, Sybase added SSL support to some Sybase products, such as ASE, Open Client, Open Server, Replication Server, and jConnect. Currently, some Sybase products use Certicom SSLPlus and Security Builder libraries.

Note Be aware that ASE 12.5 uses SSLPlus, versions 3.1.4/5 and 5.1.4, while ASE 15.0 uses SSLPlus, version 5.1.4.

ASE can use SSL for CIS and Site-handler. Also, ASE and Replication Server require licensing to use SSL.

Some Sybase products have SSL configuration options and can be easily configured to use SSL. Others provide special APIs or require some programming. The following section provides instructions for specific setups.

Setting up SSL in ASE and Open Client

Follow these procedures to configure and enable ASE and Open Client with SSL.

❖ **To set up SSL in ASE**

1 Obtain an ASE license for the SSL feature, either ASE_ASM or ASE_SECDIRS (SySAM).

2 Obtain a certificate for the server (plus any CA).

Use the Sybase-provided utilities certreq and certauth to respectively generate and sign server certificates or CA (self-signed) certificates.

3 Install the server certificate with the private key appended. The default location and naming scheme for the server certificate is *$SYBASE/$SYBASE_ASE/certificates/<server>.crt*.

Note The <*server*> name must match the server name from the interfaces file, as well as the CommonName from the server certificate.

4 Install CA certificates for the server at this location: *$SYBASE/$SYBASE_ASE/certificates/<server>.txt*.

5 Use the sp_ssladmin addcert command to let the server know about the certificate location.

6 To enable SSL in the server, issue this statement:

```
sp_configure "enable ssl", 1
```

7 Modify directory services so that listening ports use SSL by adding the keyword ssl to the appropriate interfaces file entries.

❖ **To set up SSL in Open Client**

1 Install client copies of CA certificates. You must concatenate your *<server>.txt* file to the *trusted.text* file, or simply create a new *trusted.text* file with the *<server>.txt* contents.

Client default locations are:

- For UNIX: *$SYBASE/config/trusted.txt*
- For Windows: *%sybase%\ini\trusted.txt*

2 Enable SSL on the port in Directory Servers, for example, the interfaces file.

Setting up SSL in IBM z/OS

Note The term SSL is used to describe both SSL and TLS protocols.

SSL protocol runs above the TCP/IP protocol and below higher-level protocols such as HTTP.

IBM SSL support runs as part of the TCP/IP stack under UNIX System Services (USS).

Levels of authentication and encryption available with TLS/SSL security are:

- Server authentication only
- Client authentication level 1
- Client authentication level 2
- Client authentication level 3

Note For consistency with other Sybase products, Sybase implements server authentication only.

For server authentication to work, the server must have a private key and associated Server certificate in the server key database file. To manage the keys and certificates needed for SSL support, you can use the gskkyman utility, provided by the System SSL, or RACF Common Keyring support. The server certificate and the CA certificates are stored in a key ring, also called a key database.

Here are some considerations when using RACF:

- References to RACF apply to any other System Authorization Facility (SAF)-compliant security products that contain the required support.
- For RACF support, all key rings and certificates are stored in the RACF database. There are no separate key database or stash files.

IBM provides the mainframe applications with two options for implementing SSL support:

- System SSL, which runs on top of the TCP/IP stack and provides interfaces to write both client and server applications.
- Application Transparent - Transport Layer Security (AT-TLS), which provides application-transparent secured connections for both client and server. Internally, it uses System SSL interfaces.

See the following subsections for descriptions of each of these options.

Using System SSL on z/OS

System SSL provides APIs associated with either an SSL environment layer or a secure socket connection layer.

- The SSL environment layer defines the general attributes of the environment (such as database file name, time-out, and so on).
- Secure socket connection layer defines the attributes associated with each secure connection. In addition, secure socket connection layers has read and write function calls.

First, the SSL application must create the SSL environment layer. Then, one or more secure socket connection layers can be associated with the SSL environment. Each layer has four general function calls:

- open
- attribute_set
- initialize
- close

The open function calls return a handle, either an environment handle or a secure socket connection handle, that must be passed as a parameter on subsequent function calls. Read and write functions are full-duplex; however, only one read and one write call can be in progress at one time for any secure socket connection handle.

In addition to using the SSL programming interfaces in the application, a key database must be created for the SSL application. This key database, which contains certificate information, can be an HFS file built and managed by the gskkyman utility or a RACF key ring.

System SSL uses the Integrated Cryptographic Service Facility (ICSF), if it is available. ICSF provides hardware cryptographic support that will be used instead of the System SSL software algorithms.

For System SSL to use the hardware support, the ICSF-started task must be running, and the application user ID must be authorized to the key and certificate handling resources in the RACF CSFSERV class. RACF can also be used to control access to ICSF services.

Note SSL applications must call SSL APIs from a C program because they are C APIs.

Configuring AT-TLS

AT-TLS consolidates TLS implementation in one location, reducing or eliminating application development overhead, maintenance, and parameter specification. AT-TLS is based on z/OS System SSL, and transparently implements it in the TCP layer of the stack.

Applications that are taking advantage of the AT-TLS can be separated into three different types: basic, aware, and controlling. The type is based on whether the application is aware of the service, and if so, the amount of control that the application is given over the security functions. The SIOCTTLSCTL.ioctl function call provides the interface for the application to query or control AT-TLS.

Basic application

A basic application is unaware that AT-TLS is encrypting or decrypting data.

Aware application

An aware application is aware of AT-TLS and can query information such as AT-TLS status, partner certificate, and derived RACF user ID without any advanced setting in AT-TLS policy.

Controlling application

A controlling application is aware of AT-TLS and needs to control the secure session. It must have the ApplicationControlled parameter in the AT-TLS policy set to ON.

All of these types of applications send and receive unencrypted text data while encrypted data flows over the network.

Follow this procedure to configure AT-TLS policies.

❖ **To configure AT-TLS**

1 Provide the TCP/IP stack with the AT-TLS policies required to negotiate secure connections.

AT-TLS policies are configured in the Policy Agent (described in the next section) using a set of configuration statements and parameters coded into a flat file. You can create the flat file using one of two methods:

- Using manual configuration, coding all the required statements in an HFS file or MVS data set, or
- Using z/OS Network Security Configuration Assistant, which is a standalone Windows application that requires no network connectivity or setup. You can download the GUI from the Web site for the Communication Server Family downloadable tool.

2 Enable AT-TLS through the TTLS parameter on the TCPCONFIG statement in PROFILE.TCPIP.

When AT-TLS is enabled and a newly established connection is first used, the TCP layer of the stack searches for a matching AT-TLS policy installed from the Policy Agent. If no policy is found, the connection is made without AT-TLS involvement.

Configuring AT-TLS policies in the Policy Agent

The Policy Agent component is responsible for implementing policy decisions that control network security and traffic prioritization for the z/OS environment. When initiated, the Policy Agent reads the configuration files, parses the policies, and stores the policy definitions in the TCP/IP stack, which then operates based on the policies. When the policy rule is true, one set of actions is initiated; when it is false a different set of actions is initiated.

The Policy Agent main configuration file points to other policy files that contain specific policies for TCP/IP images. It can contain a TcpImage statement that identifies the z/OS UNIX file or MVS data set that contains the policy to be received by a stack. On its end, the TCP/IP image policy file can contain a TTLSConfig statement that identifies the z/OS UNIX file or MVS data set that contains the AT-TLS policy.

Types of configuration files

There are several types of configuration files:

- Main configuration files, determined by using a standard search order
- Common IPSec configuration files
- Common AT-TLS configuration files
- Image configuration files
- Image AT-TLS configuration files

Rule conditions for a connection

Within the AT-TLS policy file, a TTLSRule statement defines a set of conditions that are compared against the connection being checked. When a match is found, policy look-up stops, and the connection is assigned the actions associated with the rule.

The rule conditions apply to connect parameters as follows:

- LocalAddr
- RemoteAddr
- LocalPortrange
- RemotePortrange
- Jobname
- Userid

Direction and at least one other condition must be specified. The TTLSRule statement can reference up to three action statements. In a simple implementation for AT-TLS, these configuration statements should be defined:

- TTLSGroupAction, which must specify TTLSEnabled=ON. The AT-TLS group action represents a single Language Environment process and enclave, and initializes one instance of the System SSL DLL.
- TTLSEnvironmentAction, which must specify a key ring and the handshake role. The AT-TLS environment action initializes a System SSL environment within the Language Environment process that was created to represent an AT-TLS group action.
- TTLSConnectionAction, which specifies attributes for a subset of connections. It is not required for a simple implementation.

❖ **To start the Policy Agent**

- You can start the Policy Agent, which runs as a UNIX process, using one of two methods:
  - From the z/OS shell, where its executable resides in */usr/lpp/tcpip/sbin*, or
  - As a started task using the PAGENT command on an MVS console. You can find a sample started task procedure for PAGENT in TCPIP.SEZAINST(EZAPAGSP).

Note To start Policy Agent from z/OS, you need security product authorization definition (for RACF or any other product).

Policy Agent search order for configuration file information

The Policy Agent search order for accessing the main configuration file (PAGENT.CONF information) is:

1 File or data set specified with the -c startup option

2 File or data set specified with the PAGENT_CONFIG_FILE environment variable

3 The *etc/pagent.conf* file

Policy Agent environment variables

These environment variables are used to tailor the Policy Agent to a particular installation:

- PAGENT_CONFIG_FILE, which points to the main configuration file or data set
- PAGENT_LOG_FILE, which points to the log file
- PAGENT_LOG_FILE_CONTROL, which controls the number and size of log files.

You might also need to define these:

- TZ, which defines the local time zone, even if it is defined in */etc/profile*.
- LIBPATH, which points to the dynamic link libraries (DLLs) needed to act as an LDAP client.

AT-TLS support in Client and Server Options for CICS

In the CICS sockets implementation, transaction security environments are not visible to AT-TLS support. The CICS job and all its transactions appear to the stack as a single server application. As a result, all AT-TLS policy look-up, System SSL key ring authorization checks, and ICSF private key authorization checks are processed using the identity of the CICS job.

The connection that is established, whether active or passive, can perform SSL handshake processing as either the client or the server. All of the connections established by a single CICS job can share the session ID cache in the SSL environment. The CICS job should use a private key ring with a Server certificate, and the key ring used must contain the chain of the root certificates it needs to validate the Server certificate it presents to the client.

Mainframe Connect Client and Server Options for CICS take advantage of the AT-TLS security support, provided that the following conditions are true:

- The TCP/IP stack supports AT-TLS.
- An AT-TLS Policy configuration matches identifiers of the CICS application that will use it, for example, the status of the application as a listener or a client, the IP addresses, and the ports that will be used for communication.
- Digital certificates and key rings are created for these applications.

Configuring a z/OS client or server system

On each z/OS system where a server or client application is to implement AT-TLS security, you need to perform these basic tasks:

1 Create a key ring

2 Create Policy Agent files

3 Add AT-TLS configuration

4 Add statements to the *TTLSConfig* policy file

5 Set up *INITSTACK* access control

6 Enable AT-TLS.

The following subsections present an example of configuration tasks performed to ensure SSL secure communication for the following network participants, all of which use self-signed digital certificates:

- A z/OS CICS server named “CICSDEV1”
- A z/OS CICS client
- A Windows ASE server named “ase1”
- A Windows Open Client client

1. Create a key ring

The task of creating a key ring involves all steps for generating and managing digital certificates. To do so, you use RACF commands. Be sure that you have RACF authority.

Note You need the SPECIAL attribute to issue the RACDCERT command. (GROUP-SPECIAL is not sufficient.)

❖ **To create a key ring**

1 To activate certificate and key ring classes, use these commands:

- SETROPTS CLASSACT(DIGTRING)
- SETROPTS CLASSACT(DIGTCERT)

2 To refresh after you make changes, use these commands:

- SETROPTS RACLIST(DIGTRING) REFRESH
- SETROPTS RACLIST(DIGTCERT) REFRESH

3 To give access to the appropriate resources, use these commands:

- RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
- RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
- PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)ID(KGUEOR) ACCESS(READ)
- PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)ID(KGUEOR) ACCESS(READ)
- PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)ID(DFHCICS) ACCESS(CONTROL)

4 To define a key ring, generate a self-signed certificate, and connect it to the key ring, use these commands:

- RACDCERT ID(DFHCICS) ADDRING(CICSDEV1)
- RACDCERT ID(DFHCICS) GENCERT SUBJECTSDN(CN('CICSDEV1') OU('MFI') O('MFI') C('US')) WITHLABEL ('CICSDEV1CERT') TRUST SIZE(1024)
- RACDCERT ID(DFHCICS) CONNECT (ID(DFHCICS) RING(CICSDEV1) LABEL('CICSDEV1CE RT') DEFAULT)
- SETROPTS RACLIST (DIGTRING DIGTCERT) REFRESH

5 To ensure the Windows SSL client connection to the “CICSDEV1” server, export the newly-created, self-signed certificate to a data set using this command:

RACDCERT EXPORT(LABEL('CICSDEV1CERT')) ID(DFHCICS) DSN(CICSDEV1.CERT)

This puts the certificate contents into the PS data set called KGUEOR.CICSDEV1.CERT using the default FORMAT(CERTB64) and code page 1047.

6 FTP the contents of that file to the Windows client's host with ASCII conversion.

7 Paste the certificate to the client's CA list in the *%sybase%\ini\trusted.txt* file.

8 Add ",ssl" to the CICSDEV1 server entry in the client's *sql.ini* file.

9 To ensure that the CICS client (Client Option for CICS) connects to the SSL server named "ase1," FTP the "ase1" certificate file from the *$SYBASE/$SYBASE_ASE/certificates/ase1.txt* file to the KGUEOR.ASE1CERT data set, with ASCII conversion.

10 Add the sent FTP "ase1" CA certificate to the RACF database with a TRUST status using this command:

```
RACDCERT ID(DFHCICS) ADD('KGUEOR.ASE1CERT')
WITHLABEL('ASE1CERT') TRUST
```

11 Connect the newly added certificate to the key ring and refresh:

```
RACDCERT ID(DFHCICS) CONNECT (ID(DFHCICS) RING(CICSDEV1)
LABEL('ASE1CERT'))

SETROPTS RACLIST (DIGTRING DIGTCERT) REFRESH
```

Note Be aware of certificates encoding, which is generally in binary or text formats. Binary formats must be transported in their exact binary format, without any conversion. In contrast, text formats (such as Base64) must be transported as text. When transporting for an ASCII system, be sure that the ASCII-to-EBCDIC translation is performed.

2. Create Policy Agent files

Note For all Policy Agent configuration file examples, only the content relevant to the AT-TLS configuring is presented here.

The main TCP/IP configuration file is in */etc/pagent.conf3*. It contains two TcpImage statements for defining policies on stacks *TCPIP* and *TCPIPT*:

```
#   Path: /etc/pagent.conf3

#   This is a sample PAGENT.CONF Policy Agent main configuration file
#   applied to stacks TCPIP and TCPIPT
LogLevel 15
```

```
#      AT-TLS config needed for TCPIPT stack
TcpImage TCPIPT /etc/TCPIPT.policy # FLUSH PURGE 600  params can be used
TcpImage TCPIP /etc/TCPIP.policy # No AT-TLS policy applied to this stack
```

3. Add the AT-TLS configuration

The policy file for the *TCPIPT* stack, */etc/TCPIPT.policy*, contains a TTLSConfig statement pointing to the AT-TLS configuration file for that stack:

```
# This is a sample TCP/IP image configuration file
# Path: /etc/TCPIPT.policy
# This is a sample TCP/IP image configuration file
# applied to stack TCPIPT
LogLevel 15
TTLSConfig /etc/pagent_TTLS.conf3
```

4. Add statements to the TTLSConfig policy file

The TCPIPT stack configuration file */etc/pagent_TTLS.conf3* has all the AT-TLS configuration statements for that stack:

```
#  Path: /etc/pagent_TTLS.conf3
#  This is a sample AT-TLS configuration file  for
#  stack TCPIPT, server CICSDEV1.
#  Server port #3042 is defined as SSL and port #3042 - as non-SSL
# Client is to connect to a remote SSL port #6005.
####  Inbound definitions   ####
#
#
TTLSRule                   DEV1SY02    # Listener on port #3042
{
LocalPortRange            3042
Userid                    DFHCICS
Direction                 Inbound
TTLSGroupActionRef        TTLSGRP1
TTLSEnvironmentActionRef TTLSENV1
}

TTLSGroupAction TTLSGRP1
{
TLSEnabled On
Trace 1             # Log Errors to TCP/IP job log
}
TTLSEnvironmentAction TTLSENV1
{
 HandshakeRole       Server
```

```
 EnvironmentUserInstance 1
 TTLSKeyRingParms
 {
 Keyring           CICSDEV1    # Refers to RACF database
 }
}

# Port #3043 is not AT-TLS configured
TTLSRule                   DEV1SY03 # Listener on port #3043
{
LocalPortRange           3043
Userid                   DFHCICS
Direction                Inbound
TTLSGroupActionRef       TTLSGRP2
}

TTLSGroupAction TTLSGRP2
{
TTLSEnabled Off
Trace 1             # Log Errors to TCP/IP job log
}

TTLSEnvironmentAction TTLSENV2
{
 HandshakeRole       Server
 EnvironmentUserInstance 3
 TTLSKeyRingParms
 {
 Keyring           CICSDEV1    # Refers to RACF database
 }
}
#
#### Outbound definitions   ####
#
TTLSRule                   DEV1Client     # Client
{
RemotePortRange          6005          # Remote port to connect to
Userid                   DFHCICS
Direction                Outbound
TTLSGroupActionRef       TTLSGRP3
TTLSEnvironmentActionRef TTLSENV3
}
TTLSGroupAction TTLSGRP3
{
TTLSEnabled On
Trace 1             # Log Errors to TCP/IP job log
```

```
}

TTLSEnvironmentAction TTLSENV3
{
 HandshakeRole       Client
 EnvironmentUserInstance 1
 TTLSKeyRingParms
 {
 Keyring           CICSDEV1    # Refers to RACF database
 }
}
```

5. Set up INITSTACK access control

You need to define the security product authorization for PAGENT.

When using AT-TLS, z/OS will not allow any socket-based applications to start before PAGENT is up and running. This restriction is needed to verify that all the security policies are enforced. However, some essential applications need to start before PAGENT. For these applications, you need to define a resource INITSTACK profile in the SERVAUTH class. The resource name consists of the following parts:

- EZB is the constant.
- INITSTACK is the constant for this resource type.
- *sysname* is the system name.
- *tcpprocname* is the TCP/IP proc name.

When TCPCONFIG TTLS is defined in the initial *TCPIP.PROFILE*, the INITSTACK profile must be defined. Policy Agent—and any socket -based programs it requires—must be given permission to this resource.

Note Be sure that the program name is the name used to invoke the program—*not* the module name.

Most TCP/IP applications are invoked by ALIAS name. This example lists both names:

```
SETROPTS CLASSACT(SERVAUTH)
SETROPTS RACLIST (SERVAUTH)
SETROPTS GENERIC (SERVAUTH)
RDEFINE SERVAUTH EZB.INITSTACK.*.TCP*       UACC(NONE) PERMIT
EZB.INITSTACK.*.TCP* CLASS(SERVAUTH) ID(*) ACCESS(READ)-
```

```
         WHEN(PROGRAM(PAGENT,EZAPAGEN)
SETROPTS GENERIC(SERVAUTH) REFRESH
SETROPTS RACLIST(SERVAUTH) REFRESH
SETROPTS WHEN(PROGRAM) REFRESH
```

6. Enable AT-TLS

To enable AT-TLS functionality for a stack, define the TCPCONFIG TTLS parameter in the TCPIP.PROFILE.

7. Run SSL secure connections

After you complete all previous configuration tasks, you are ready to start the Policy Agent. Here is the example of the PAGENT procedure used:

```
//PAGENT   PROC PARMS='-c /etc/pagent.conf3 -l /tmp/pagent.log4'
//PAGENT   EXEC PGM=PAGENT,REGION=0K,TIME=NOLIMIT,
// PARM=('POSIX(ON) ALL31(ON)',
//     'ENVAR("LIBPATH=/usr/lib"',
//     '"TZ=MST7MDT6")/&PARMS')
//STDENV DD PATH='/etc/pagent.env',PATHOPTS=(ORDONLY)
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=132,BLKSIZE=132)
```

where */etc/pagent.env* contains these definitions:

- LIBPATH=/usr/lib
- TZ=MST7MDT6

After PAGENT starts successfully, start the stack TCPIPT and the CICS region called “CICSDEV1” on that stack. If the server called “ase1” is running, you are ready to test your connections.

For more information

Refer to the IBM *z/OS Communication Server: IP Configuration Guide* for more information about these topics:

- For an overview of the policy-based networking and components, see the chapter called “Policy-based Networking.”

- For information on the AT-TLS security application and configuration, see the chapter called “AT-TLS Data Protection.”
- For an overview of using digital certificates, generating them, and handling them, see the appendix called “TLS/SSL Security.”

For detailed information on Policy Agent statements and policy applications, see the chapter called “Policy Agent Statements and Policy Applications” in the *z/OS Communication Server: IP Configuration Reference*.

For more information about using digital certificates, generating them, and handling them, refer to the chapter called “RACF and Digital Certificates” in the *z/OS Security Server RACF Security Administrator Guide*.

For detailed information about using the RACDCERT (RACF Digital Certificate) command, refer to the *z/OS Security Server RACF Command Language Reference*.

For diagnosing AT-TLS or Policy Agent problems, refer to the appropriate chapters in the *z/OS Communication Server: IP Diagnosis Guide*.

For information on using digital certificates with other security products, refer to related documentation.

For more information on how to obtain digital certificates, configure and use SSL connections with the Sybase products, see the appropriate Sybase product documentation.

APPENDIX E

Gateway-less Considerations

Note Although much of the information in this guide addresses the use of the Server Option in a three-tier, gateway-enabled environment, the Server Option can be used in a two-tier, gateway-less environment. This appendix discusses issues you should consider when using the Server Option in a two-tier environment.

Introduction

Working with the Server Option in a two-tier environment, client applications can access and update data stored in mainframe resources without having to interact with a gateway component like the DirectConnect for z/OS Option. These clients include both Sybase and user-written applications, including the following:

- Open Client applications
- PowerBuilder® applications
- ASE/CIS
- Replication Server®
- EAServer
- jConnect applications

Trade-offs

There are both advantages and disadvantages to using the Server Option in a two-tier environment over a three-tier environment.

Advantages of a three-tier environment

The features and functionality available with the DirectConnect for z/OS Option are available to the three-tier user of the Server Option and unavailable in a two-tier environment. These features include:

- DirectConnect for z/OS Option access service features
- DirectConnect for z/OS Option TRS features
- General DirectConnect for z/OS Option functionality

DirectConnect for z/OS Option access service features

These DirectConnect for z/OS Option access service features are available to the three-tier user:

- *Datatype translation* – datatypes used by your client applications are matched to those used on the mainframe.
- *SQL transformation* – the SQL used by your client applications matches the SQL used by mainframe applications.

DirectConnect for z/OS Option TRS features

These DirectConnect for z/OS Option TRS features are available to the three-tier user:

- *IMS/MVS access* – IMS/MVS data is available to the client in a three-tier environment.
- *Transaction mapping by security definition* – transactions can be mapped from client to mainframe by the security definition.
- *Transaction mapping by user ID* – transactions can be mapped from client to mainframe by the user ID of the client.

General DirectConnect for z/OS Option functionality

These DirectConnect for z/OS Option features are available to the three-tier user:

- *Logging, tracing, and accounting* – your environment can use DirectConnect for z/OS Option facilities for logging, tracing, and accounting.
- *GUI configuration and multi-user management* – any GUI configuration and user-management tools available with the DirectConnect for z/OS Option are available to the three-tier user.
- *SNA connectivity* – the SNA connectivity protocol can only be used through a gateway, which provides TCP/IP-to-SNA protocol conversion.
- *Access to multiple CICS regions* – client applications in a three-tier environment may access multiple CICS regions at one time.
- *Transaction grouping* – transactions can be grouped in a three-tier environment.
- *Security* – additional levels of security can be implemented by the gateway in a three-tier environment.

Two-tier advantages

A primary advantage of a two-tier environment is the increased throughput associated with a simpler architecture. Also, less administrative attention is required to install and maintain components in a two-tier environment.

Database connectivity

Sybase and user-written clients can access the Client and Server Options using any of the standard database connectivity drivers, including the ASE ODBC Driver by Sybase and Sybase jConnect™ for JDBC™ drivers.

Using RPCs in a two-tier environment

The use of RPCs in a two-tier, gateway-less environment is accomplished by using the SYRP CICS transaction to map LAN RPC names to CICS transaction names. These mappings are stored in the *SYRPCFIL* file, which resides on the mainframe.

Accessing DB2 UDB with CSPs

The Server Option in a two-tier, gateway-less environment supports the use of CSPs in the DB2 UDB Option for CICS to access the DB2 UDB catalog. For information on how to use CSPs, refer to the Mainframe Connect DB2 UDB Option for CICS *Installation and Administration Guide*.

APPENDIX F

Network considerations

Understanding network communication definitions

Use this overview to understand network communication topics and issues.

- System Application Architecture (SAA)
- Common Programming Interface (CPI)
- APPC/MVS
- Systems Network Architecture (SNA)
- LU 6.2
- Advanced Program-to-Program Communications (APPC)
- Common threads between APPC/MVS, CICS, and IMS TM
- Transmission Control Protocol/Internet Protocol (TCP/IP)

System Application Architecture (SAA)

SAA is composed of selected software interfaces, conventions, and protocols designed to provide a framework for developing distributed applications. The benefits of SAA are portability, consistency, and connectivity. The components of SAA are specifications for the key application interface points:

- Common user access

- Common communication support
- Common Programming Interface (CPI), explained in the following section

Systems Network Architecture (SNA)

SNA is an IBM Network Architecture composed of software interfaces, protocols, and operational sequences used for network configuration, operation, and communication.

LU 6.2

LU 6.2 is the SNA Logical Unit Type 6.2, which supports general communication between programs in a distributed environment. LU 6.2 is characterized by peer-to-peer communications support, comprehensive end-to-end error processing, optimized data transmission flow, and a generic API.

The LU 6.2 system is layered functionally. It can be represented by a set of finite-state machines, each of which has a finite number of states and a set of rules that govern the transition from one state to another. These finite state machines govern the behavior of LU 6.2 devices by guaranteeing that a given input always produces the same output.

Advanced Program-to-Program Communications (APPC)

APPC is peer-level data communication support based on the SNA LU 6.2 protocols.

APPC/MVS

APPC/MVS is an SNA application that extends APPC support to the z/OS operating system. APPC/MVS provides full LU 6.2 capacity to z/OS applications to allow communication with other applications across a distributed SNA network.

APPC/MVS provides programming support by providing an API based on the CPI-C interface. This interface is implemented in a lower-level API that is z/OS-specific:

- CPI-C calls all begin with CM. For example, CMALLC (Allocate).
- z/OS calls all begin with ATB. For example, ATBSEND (Send_data).

The CPI-C calls are portable to non-z/OS platforms. ATB calls are not portable to non-z/OS platforms.

Common threads between APPC/MVS, CICS, and IMS TM

All inbound transactions require a scheduler and are scheduled as follows:

- *z/OS* – in z/OS, the ASCH address space schedules inbound transactions in initiators under its control. The ASCH use of initiators is similar to that of JES (Job Entry Subsystem), which schedules jobs in initiators under its control.
- *IMS TM* – in IMS TM, the Control region schedules inbound transactions using message regions under its control. The Control region use of message regions is similar to the ASCH use of initiators.
- *CICS* – CICS schedules inbound transactions as tasks within its own address space. CICS differs from z/OS and IMS TM in that it does not schedule transactions in a separate address space.

Outbound transactions are handled as follows:

- *z/OS* – outbound transaction names are mapped to an SNA logical unit using a file called the Side Information File.
- *IMS TM* – outbound transaction names are mapped to an SNA logical unit using a file called the Side Information File.
- *CICS* – for CPI-C, transaction names are mapped through the PARTNER table, which is set up using the Resource Definition Online (RDO) facility.

  LU 6.2 uses connection and session tables and the RDO facility.

Common Programming Interface (CPI)

The SAA CPI specifies the languages and services used to develop applications across SAA environments. The elements of the CPI specification are divided into two parts:

1 *Processing logic*, which consists of these three components:

   - High-level language (HLL): COBOL, C, Fortran, RPG

- Procedure language: REXX
- Application generator: Cross Systems Product/Application Development (CSP/AD)

2 *Services*, which consists of these three components:

- Communication Interface or CPI-C: API for writing APPC applications.
- Database Interface: Structured Query Language (SQL)
- Dialog Interface: Interactive System Productivity Facility (ISPF)

Transmission Control Protocol/Internet Protocol (TCP/IP)

TCP/IP is a set of protocols supporting network communications.

The two major interfaces for network programming using TCP/IP are AT TLI (Transport Library Interface) and the BSD Sockets Interface. AT TLI is older than the more recent BSD Sockets Interface.

Only IBM TCP/IP is supported; it uses the BSD Sockets Interface.

CICS LU 6.2 sample networks

Note If you are using CICS TCP/IP, skip this section.

This section includes two samples of mainframe-TRS networks on which Sybase components can run.

The first sample represents an IBM Token-Ring network running single sessions only, in which conversation-level security is supported. Configuration examples throughout this book use the names and other values from this sample. TRS manuals use the same sample network in their configuration instructions.

The second sample network represents an SDLC non-switched line that supports parallel sessions.

Warning! These samples are generic. You must make appropriate changes, based on requirements for your site and recommendations from your IBM representative.

Sample Token-Ring network

This section covers SNA and CICS definitions for a sample Token-Ring network and contains the following topics:

- SNA entries
- CICS definitions

The sample illustrated in this section uses the following names:

Table F-1: Token-Ring specification names

Token-Ring specification	Name
Host LU (CICS Region)	CICSSYB
Remote LU Name	*SYBLU02* SYBLU03 SYBLU04
Remote LU Addresses	2, 3, 4
SNA Logon Mode Table Name	SYSTABV
Sessions	*Single sessions*
RU Size	1024
Pacing Setting	5
Synch Level	Confirm
Security	*Verify*

SNA entries

This section contains the SNA Logmode entry and network definition statements. It includes the following subsections:

- APPL definition statement
- Logmode entry
- Network definition (PU and LU statements)

APPL definition statement

This statement defines CICS region CICSSYB to SNA:

```
CICSAPPL VBUILD TYPE=APPL            APPLICATION MAJOR NODE
 CICSSYB  APPLAUTH=(ACQ,VPACE),PARSESS=YES,SONSCIP=YES,VPACING=5,*
    EAS=50,APPC=NO,ACBNAME=CICSSYB
```

Logmode entry

This statement defines mode table SYBTABV to SNA:

```
 LOGMODE ENTRY:
  ---------------  SYBTABV   MODETAB
      EJECT
      TITLE 'M6S1024V'
 *-------------------------------------------------------------*
 *  LU 6.2, SINGLE SESSIONS, RU_SIZE(1024), SYNCH_LEVEL(CONFIRM),
 *  SECURITY(VERIFY)
 *-------------------------------------------------------------*
 M6S1024V   MODEENT LOGMODE=M6S1024V,FMPROF=X'13',TSPROF=X'07',  +
    PRIPROT=X'B0',SECPROT=X'B0',COMPROT=X'78A5',          +
    RUSIZES=X'8787',TYPE=X'00',                           +
    PSNDPAC=X'05',SRCVPAC=X'05',SSNDPAC=X'05',            +
    PSERVIC=X'060200000000000000102C00'
 *
    MODEEND
    END
```

Network definition (PU and LU statements)

The following statement defines your network to SNA:

```
 TRGRPSYB VBUILD TYPE=LOCAL
  *
  SYBPU1 PU   CUADDR=041,DLOGMOD=M6S1024V,MAXBFRU=11,SSCPFM=FSS,   +
    USSTAB=ISTINCDT,DELAY=0,SECNET=YES,ISTATUS=ACTIVE,   +
    MODETAB=SYBTABV
 *
 SYBLU02 LU   LOCADDR=2
 SYBLU03 LU   LOCADDR=3
 SYBLU04 LU   LOCADDR=4
```

CICS definitions

This section explains the CICS definitions for the sample Token-Ring network and includes the following subsections:

- CICS APPLID
- System Initialization Table (SIT)
- Connection definition
- Session definition

CICS APPLID

Define the CICS APPLID to SNA, as shown under “APPL definition statement” on page 108.

System Initialization Table (SIT)

Set these parameters as follows for CICS version 3.x and later:

- ISC=YES
- TCP=YES
- SEC=YES, if using an external security manager
- SNA=YES

Connection definition

Use the following sample as a model. Change it, as appropriate to your site, noting the requirements listed after the sample:

```
OBJECT CHARACTERISTICS
 CEDA  View Connection( SYB2 )
  Connection     : SYB2
  Group          : SYBCONN
  DEscription    : ARAPAHOE
 CONNECTION IDENTIFIERS
  Netname        : SYBLU02
  INDsys         :
 REMOTE ATTRIBUTES
  REMOTESYSTem   :
  REMOTEName     :
  REMOTESYSNet   :
 CONNECTION PROPERTIES
  ACcessmethod   : SNA               SNA | IRc | INdirect | Xm
  PRotocol       : Appc              Appc | Lu61 | Exci
  Conntype       :                   Generic | Specific
  SInglesess     : Yes               No | Yes
  DAtastream     : User              User | 3270 | SCs | STrfield | Lms
  RECordformat   : U                 U | Vb
```

```
   Queuelimit     : No                 No | 0-9999
   Maxqtime       : No                 No | 0-9999
  OPERATIONAL PROPERTIES
   AUtoconnect    : No                 No | Yes | All
   INService      : Yes                Yes | No
  SECURITY
   SEcurityname   : SYBUSER
   ATtachsec      : Verify             Local | Identify | Verify | Persistent
                                       | Mixidpe
   BINDPassword   :                    PASSWORD NOT SPECIFIED
   BINDSecurity   : No                 No | Yes
   Usedfltuser    : No                 No | Yes
  RECOVERY
   PSrecovery     : Sysdefault         Sysdefault | None
   Xlnaction      : Keep               Keep | Force
```

Requirements include the following:

- The connection and session are related by the Connection parameter in the OBJECT CHARACTERISTICS and SESSION IDENTIFIERS definitions.
- The Netname parameter corresponds to an LU name defined to SNA.
- Set the SInglesess parameter to Yes for dependent LUs.
- Set AUtoconnect to No for single sessions.
- For conversation-level security:
  - Set the SEcurityname parameter to a user ID specified in the CICS sign-on table, or use the default sign-on table entry and enter a valid RACF ID.
  - Set the ATtachsec parameter to Verify.

Session definition

Use the following sample as a model. Change it as appropriate to your site, noting the requirements listed after the sample:

```
OBJECT CHARACTERISTICS
 CEDA  View Sessions( SESLU02 )
  Sessions       : SESLU02
  Group          : SYBCONN
  DEscription    : ARAPAHOE
 SESSION IDENTIFIERS
  Connection     : SYB2
```

```
 SESSName       :
 NETnameq       :
 MOdename       : MVSMODE
SESSION PROPERTIES
 Protocol       : Appc              Appc | Lu61 | Exci
 MAximum        : 001   000         0-999
 RECEIVEPfx     :
 RECEIVECount   :                   1-999
 SENDPfx        :
 SENDCount      :                   1-999
 SENDSize       : 01024             1-30720
 RECEIVESize    : 01024             1-30720
 SESSPriority   : 000               0-255
 Transaction    :
OPERATOR DEFAULTS
 OPERId         :
 OPERPriority   : 000               0-255
 OPERRsl        : 0                                                 0-24 ...
 OPERSecurity   : 1                                                 1-64 ...
PRESET SECURITY
 USERId         :
OPERATIONAL PROPERTIES
 Autoconnect    : No                No | Yes | All
 INservice      :                   No | Yes
 Buildchain     : Yes               Yes | No
 USERArealen    : 000               0-255
 IOarealen      : 00000   00000      0-32767
 RELreq         : No                No | Yes
 DIscreq        : No                No | Yes
 NEPclass       : 000               0-255
RECOVERY
 RECOVOption    : Sysdefault        Sysdefault | Clearconv | Releasesess
                                    | Uncondrel | None
 RECOVNotify    : None              None | Message | Transaction
```

Requirements include the following:

- The connection and session are related by the Connection parameter in the OBJECT CHARACTERISTICS and SESSION IDENTIFIERS definitions.
- On the SESSION definition, set MAximum=001 000. This setting both defines one session for the LU and sets CICS as the contention loser, improving performance during session initiation (BIND).

- Set SENDSize and RECEIVESize to match the value specified in the RUSIZES parameter of the chosen logmode.
- AUtoconnect determines whether CICS attempts to bind sessions when the connection is established. Set this parameter to No.

Sample SDLC non-switched line with parallel sessions

This section contains the following subsections:

- SNA entries
- CICS definitions

You need all of the following SNA, NCP, and CICS definitions to define a Remote SDLC Non-Switched Line supporting parallel sessions.

SNA entries

This section contains the SNA Logmode entry and NCP SDLC Group definition. It contains the following subsections:

- Logmode entry
- NCP SDLC group definition

Logmode entry

The following statement defines mode table SYBTABV to SNA:

```
 LOGMODE ENTRY:
 ---------------
 SYBTABV  MODETAB
          EJECT
          TITLE 'M6P1024V'
 *-------------------------------------------------------------------*
 * LU 6.2, PARALLEL SESSIONS, RU_SIZE(1024), SYNCH_LEVEL(CONFIRM),
 * SECURITY(VERIFY)
 *-------------------------------------------------------------------*
 M6P1024 MODEENT
LOGMODE=M6P1024,FMPROF=X'13',TSPROF=X'07',                          *
    PRIPROT=X'B0',SECPROT=X'B0',COMPROT=X'78A5',                        *
    RUSIZES=X'8787',TYPE=X'00',                      *
    PSNDPAC=X'05',SRCVPAC=X'05',SSNDPAC=X'05',                        *
    PSERVIC=X'060200000000000000102F00'                        *
```

```
   EJECT
   MODEEND
   END
```

The value for RUSIZES matches that used for SENDSize and RECEIVESize. See “Session definition” on page 116.

NCP SDLC group definition

Use the following sample as a model. Change it as appropriate to your site, noting the requirements listed after the sample:

```
******* NCP: REMOTE SNA LU 6.2 SDLC NON-SWITCHED *******

*****************************************************************
SDLC NON-SWITCHED LINES GROUP
*****************************************************************

*
NSWGRP GROUP CLOCKNG=DIRECT,          SCANNER PROVIDES CLOCKING
             +
  DIAL=NO,         NO SWITCHED LINES IN THIS GROUP               +
  DISCNT=NO,         (V) SNASNA               +
  ISTATUS=ACTIVE,         INITIAL STATUS ACTIVE               +
  LNCTL=SDLC,         SDLC LINE CONTROL               +
  NEWSYNC=NO,         DO NOT SUPPLY NEW SYNC SIGNAL               +
  PAUSE=(0,0),         0 SEC BETWEEN SERVICE CYCLES               +
  PU=YES,         (V) SNA               +
  REPLYTO=0.3,         .3 SEC REPLY TIMEOUT               +
  RETRIES=(5,0,2),         5 RETRIES PAUSE 0 SEC. FOR 3 TIMES               +
  SERVLIM=8,         NUMBER OF SCANS OF THE SOT               +
  SPEED=19200,         LINE SPEED               +
  TYPE=NCP         NCP MODE ONLY LINE GROUP
*-----------------------------------------------------------------------
* LIC 01 PORT 01;
*-----------------------------------------------------------------------
L01P01 LINE    ADDRESS=01,         LINE ADDRESS ON 3745               +
  ISTATUS=ACTIVE,         INITIAL STATUS               +
  MAXPU=1,                        +
  MAXDATA=1024,                        +
  MODETAB=SYBTABV,                        +
  NRZI=NO
*
S01P01 SERVICE ORDER=(NSWPU1),ORDER IN WHICH                        +
    MAXLIST=1         DEVICES ARE SERVICED
*
```

```
 NSWPU1 PU
ADDR=01,DLOGMOD=M6P1024V,SSCPFM=FSS,USSTAB=ISTINCDT,
+
     XID=YES,PUTYPE=2
 NSWLU101 LU LOCADDR=0,RESSCB=128,PACING=5,LOGAPPL=CICSSYB
 *---------------------------------------------------------------------
```

Requirements include the following:

- The MAXDATA value should match the RUSIZES defined on the chosen DLOGMOD.
- To define an independent LU to NCP, refer to appropriate IBM documentation for NCP resource definition. Requirements are:
  - On the PU definition, set XID=YES.
  - Define an independent LU (ILU) as LOCADDR=0. You can define more than one ILU per PU.
  - The RESSCB parameter defines the number of Boundary Session Control Blocks (BSBs) reserved for this ILU. You need one BSB for each session in which the ILU participates.

Also consider that the LOGAPPL parameter is used for error recovery in case the PU becomes inactive. If you specify LOGAPPL, then when the PU is reactivated, SNA reestablishes the SNASVCMG service session between this LU and the application defined to LOGAPPL.

CICS definitions

To define an independent LU to CICS, use the models in the following subsections:

- Connection definition
- Session definition

Connection definition

Use the following sample as a model. Change it as appropriate to your site, noting the requirements listed after the sample:

```
  OBJECT CHARACTERISTICS
   CEDA  View Connection( ILU1 )
    Connection     : ILU1
    Group          : SYBCONN
```

```
 DEscription    : ARAPAHOE
CONNECTION IDENTIFIERS
 Netname        : NSWLU101
 INDsys         :
REMOTE ATTRIBUTES
 REMOTESYSTem   :
 REMOTEName     :
 REMOTESYSNet   :
CONNECTION PROPERTIES
 ACcessmethod   : SNA               SNA | IRc | INdirect | Xm
 PRotocol       : Appc              Appc | Lu61 | Exci
 Conntype       :                   Generic | Specific
 SInglesess     : No                No | Yes
 DAtastream     : User              User | 3270 | SCs | STrfield | Lms
 RECordformat   : U                 U | Vb
 Queuelimit     : No                No | 0-9999
 Maxqtime       : No                No | 0-9999
OPERATIONAL PROPERTIES
 AUtoconnect    : All               No | Yes | All
 INService      : Yes               Yes | No
SECURITY
 SEcurityname   : SYBUSER
 ATtachsec      : Verify            Local | Identify | Verify | Persistent
                                    | Mixidpe
 BINDPassword   :                   PASSWORD NOT SPECIFIED
 BINDSecurity   : No                No | Yes
 Usedfltuser    : No                No | Yes
RECOVERY
 PSrecovery     : Sysdefault        Sysdefault | None
 Xlnaction      : Keep              Keep | Force
```

Requirements to allow an independent LU to support parallel sessions include the following:

- Set the SInglesess parameter to No.
- Set the AUtoconnect parameter to All. If CICS goes down, this parameter tells CICS to reestablish the SNASVCMG sessions with this LU.
- For conversation-level security:
  - Set the SEcurityname parameter to a user ID specified in the CICS sign-on table, or use the default sign-on table entry and enter a valid RACF ID.
  - Set the ATtachsec parameter to Verify.

Session definition

Use this sample as a model. Change it as appropriate to your site, noting the requirements listed after the sample:

```
OBJECT CHARACTERISTICS
 CEDA  View Sessions( SILU1 )
  Sessions       : SILU1
  Group          : SYBCONN
  DEscription    : ARAPAHOE
 SESSION IDENTIFIERS
  Connection     : ILU1
  SESSName       :
  NETnameq       :
  MOdename       : M6P1024V
 SESSION PROPERTIES
  Protocol       : Appc               Appc | Lu61 | Exci
  MAximum        : 128   000          0-999
  RECEIVEPfx     :
  RECEIVECount   :                    1-999
  SENDPfx        :
  SENDCount      :                    1-999
  SENDSize       : 01024              1-30720
  RECEIVESize    : 01024              1-30720
  SESSPriority   : 000                0-255
  Transaction    :
 OPERATOR DEFAULTS
  OPERId         :
  OPERPriority   : 000                0-255
  OPERRsl        : 0                                                  0-24 ...
  OPERSecurity   : 1                                                  1-64 ...
 PRESET SECURITY
  USERId         :
 OPERATIONAL PROPERTIES
  Autoconnect    : No                 No | Yes | All
  INservice      :                    No | Yes
  Buildchain     : Yes                Yes | No
  USERArealen    : 000                0-255
  IOarealen      : 00000   00000      0-32767
  RELreq         : No                 No | Yes
  DIscreq        : No                 No | Yes
  NEPclass       : 000                0-255
 RECOVERY
  RECOVOption    : Sysdefault         Sysdefault | Clearconv | Releasesess
                                      | Uncondrel | None
  RECOVNotify    : None               None | Message | Transaction
```

Requirements include the following:

- To allow an independent LU to support parallel sessions:
  - Set the MOdename parameter to a logmode that supports parallel sessions, that is, where bits 6 and 7 of the 11th byte of the PSERVIC are each set to 1.
  - Change MAximum to xxx yyy, where xxx = maximum number of sessions, yyy = number of contention winners. For improved performance, set CICS to be the contention loser (yyy=000).
- Set SENDSize and RECEIVESize to match the value specified in the RUSIZES parameter of the chosen logmode.
- Set AUtoconnect to No. AUtoconnect determines whether CICS attempts to bind sessions during start-up.

APPENDIX G

Troubleshooting

Note For troubleshooting information about TRS, see the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*. For explanations of specific error messages, see the Mainframe Connect Client Option and Server Option *Messages and Codes*.

Where to start troubleshooting

At the client, the DirectConnect for z/OS Option workstation, and mainframe levels, check components systematically to locate the problem. Depending on your setup, you may want to check for problems in this sequence:

1 Connectivity

2 Client application

3 Client LAN

4 Client network setup

5 Major outage

6 DirectConnect for z/OS Option workstation

7 Connection from the client to the DirectConnect for z/OS Option workstation

8 Connection from the DirectConnect for z/OS Option workstation to the mainframe

For any of these problems, the appropriate system administrator should use normal troubleshooting procedures. For example:

- Record specific information on the error messages, including:
  - Error message number
  - Associated SNA sense codes or SNA Services error codes
  - Time the error occurred
  - The client or user affected
- Refer to the appropriate documentation, as needed.
- Perform the recommended action.
- Continue the process until the problem is resolved.

Common problems and suggested solutions

Problems can often be traced to configuration errors or to network, line, modem, or adaptor outages.

This section contains the following subsections:

- Configuration errors
- Mainframe network operational failure
- Network session or line failures

Configuration errors

This section contains the following subsections:

- Cannot establish session
- Session established but transaction does not run
- SDLC line or Token-Ring not up
- SDLC Link and PU are active but LU is not active

Configuration errors are often the cause of communications failure. To resolve these errors, you need the following information, which was created when the network was installed and successfully implemented:

- CICS RDO definition
- SNA /NCP definitions for the LU and associated logmode
- SDLC or Token-Ring connection charts to the mainframe
- TCP/IP connection charts to clients
- For CICS LU 6.2, workstation SNA configuration
- Sybase interface files for clients and TRS
- Sybase security definitions, including client logins, connection groups, and transaction groups

Verify that this information is the same as it was before the error occurred. If it is not, determine whether a recent change is contributing to the problem.

Cannot establish session

Cause

Check for the following:

- Mismatched LU definitions between SNA and workstation
- Mismatched modenames
- Incorrect SNA MODETAB and APPLID macros
- Incorrect CICS RDO definitions
- Incorrect TCP/IP addresses or host names

Suggested action

- Check the CICS system log on the mainframe for messages.
- Correct the spelling.
- Coordinate with the TRS administrator to check connection and modename profiles, using the utility shipped with the product.

Session established but transaction does not run

Cause

Check for the following:

- CICS RDO definition errors
- RACF security error
- Incorrect transaction ID in the TRS RPC table
- Incorrect entries in the SYRP transaction (gateway-less)

Suggested action

- Check the CICS system log on the mainframe for messages.
- Verify definitions.
- Coordinate with the TRS administrator for correct security and transaction ID setups.
- Check the SYRP transaction entries.

SDLC line or Token-Ring not up

Cause

Address is incorrectly configured with NCP (assumes correct line or modem setup).

Suggested action

Check both ends of the SDLC station or Token-Ring address configuration.

SDLC Link and PU are active but LU is not active

Cause

Check for the following:

- SNA and DirectConnect for z/OS Option LU definition errors
- An incorrect SSCPID value in the local LU profile

Suggested action

Use the SDLC trace and error log facilities to find the error.

Mainframe network operational failure

On the mainframe, there are two frequent causes of operational errors:

- The CICS or SNA operator put the resource out of service with the vary command.
- SNA placed the line, physical unit (PU), or LU into a non-operating (INOP) state because of a network outage.

In these cases, one of the following occurs:

- The TRS administrator sees SNA Services time-out and connection failure messages when trying to start the DirectConnect for z/OS Option.
- The requesting client sees an SNA Services message indicating that the system could not start the RPC.

When you are contacted about such messages, reactivate the necessary mainframe resources.

Network session or line failures

This section explains what happens when line, adapter, or modem outages occur and how to prevent them.

This section contains the following subsections:

- When these errors occur
- Preventing these errors

When these errors occur

Line, adapter, or modem outages result in error messages at the SNA console and at the DirectConnect for z/OS Option. The DirectConnect for z/OS Option records the message and, when possible, sends a similar error message to any affected clients.

Preventing these errors

Intermittent hardware errors and line degradation problems disrupt processing and may be difficult to find. It helps to check periodically for these problems. For example:

- To check for hardware errors, use the SNA error logs. Report errors to IBM Service.
- To check for line degradation, use SNA to periodically report the SDLC line statistics. Examine the statistics for a significant number of re-transmissions or idle detect timeouts. Line degradation results in random SDLC line failures or very slow response to the client, even during a moderate processing load.

Troubleshooting at each component

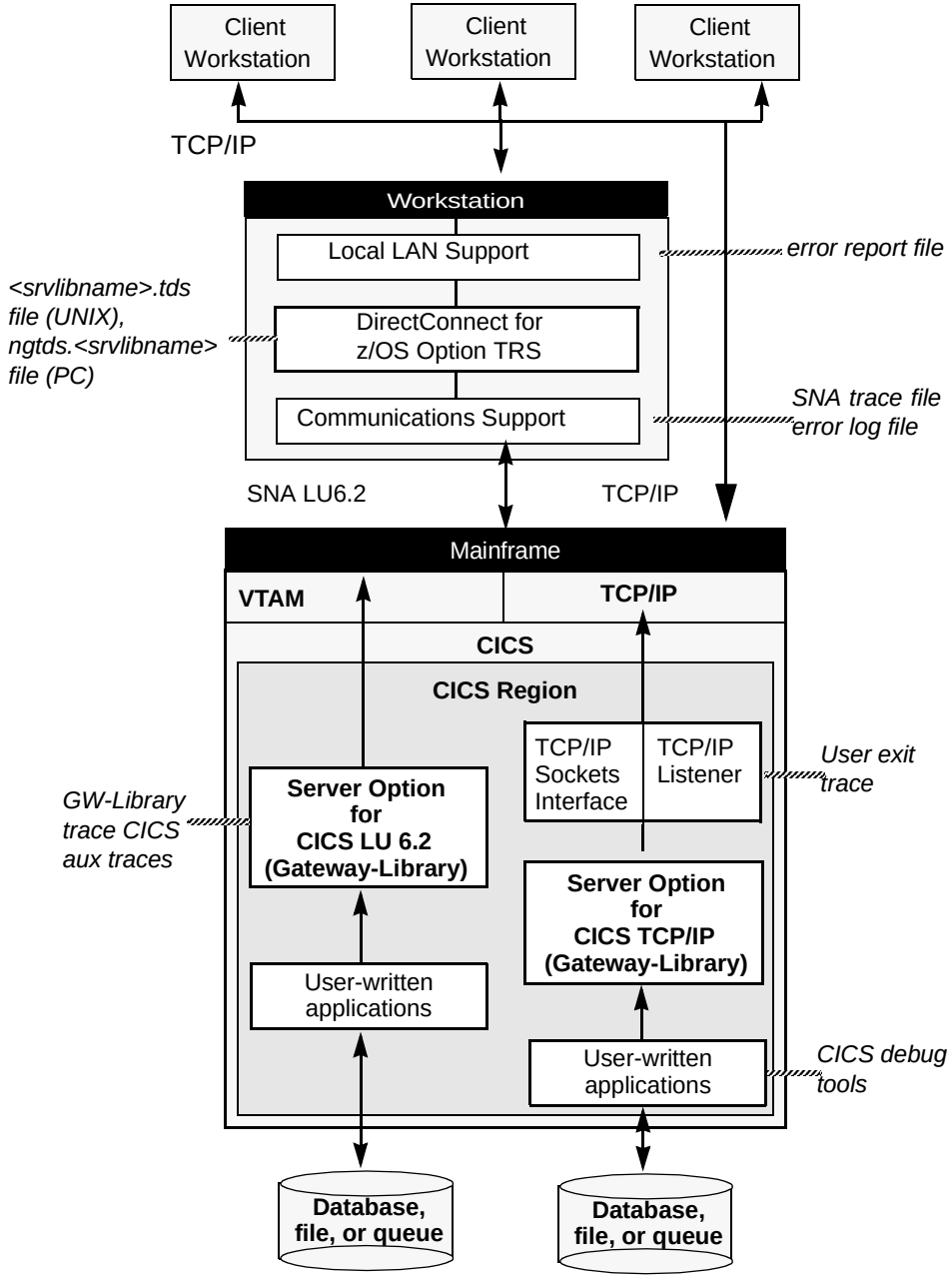
This section explains troubleshooting at each of the Sybase components. It includes the following subsections:

- Server Option support
- TRS support
- DirectConnect for z/OS Option communications with the mainframe
- Mainframe communications support
- Gateway-Library support (Server Option user-written applications and RPCs)
- DB2 UDB Option for CICS support
- Gateway-less support

Server Option support

Server Option support consists of several components on the IBM z/Series mainframe and the DirectConnect for z/OS Option platform, as the following diagram shows. On the z/OS Option platform, support is aided by information from LAN error report files, TDS and SNA trace files, and connectivity error log files. On the IBM z/Series mainframe, Server Option GW-Library and CICS aux traces, TCP/IP listener traces, and CICS debug tools help in troubleshooting.

Figure G-1: Components for Server Option support



TRS support

TRS does the following:

- Receives requests from client applications
- Converts the requests to the appropriate communications protocol call
- Sends the requests to the mainframe

Each instance of TRS has a unique service name, which clients use to select a service for communication. Each instance of TRS has its own set of configuration information, defined globally in the TRSL configuration file.

As shown in the previous figures, TRS uses the following files:

- *<srvlibname>.tds* for tracing Sybase TDS traffic between TRS on UNIX platforms and mainframe SNA
- *ngtds.<srvlibname>* for tracing Sybase TDS traffic between TRS on PC platforms and mainframe SNA
- *svr.log* for logging TDS traffic between TRS and client workstations, and for recording errors

The Transaction Router Service Library (TRSL) SNATraceFile configuration property specifies the file to which the SNA side of the TRS trace file is written. Formatted TDS traces and errors are logged and written to this file.

The associated TRSL name is appended to these files so that the TRS administrator can differentiate the log for each instance of TRS. For more information, refer to the Mainframe Connect DirectConnect for z/OS Option *Users Guide for Transaction Router Services*.

The DirectConnect for z/OS Option server logs TDS traffic between TRS and client workstations, and records errors. For more information, refer to the Enterprise Connect Data Access and Mainframe Connect *Server Administration Guide*.

DirectConnect for z/OS Option communications with the mainframe

This section contains the following subsections:

- SNA LU 6.2
- TCP/IP

TRS depends on the communications support of the server it runs on to communicate with the mainframe transaction processor.

SNA LU 6.2

The LAN communications server, such as SNA Services for AIX, uses the SNA trace file to record SDLC/SNA traffic between the workstation and mainframe. The vendor's trace utility extracts this file.

For AIX, the error log file records errors that SNA Services detects. The IBM error log report utility extracts this information. Refer to the appropriate Microsoft SNA Server documentation for details on Windows.

TCP/IP

For CICS TCP/IP environments, third party TCP/IP trace facilities provide a way of obtaining low-level TCP/IP traces between the mainframe and TRS.

For AIX, the error log file records errors that TCP/IP Services detects. The IBM error log report utility extracts this information.

Mainframe communications support

Mainframe-based communications support provides the "transport" level of function. Depending on the mainframe communications software installed, Gateway-Library uses SNA/NCP or TCP/IP for z/OS.

This section contains the following subsections:

- SNA/NCP
- TCP/IP for z/OS

SNA/NCP

For CICS LU 6.2 environments, you can use the SNA General Trace Facility (GTF) files to trace SDLC/SNA traffic between TRS and the mainframe. The IBM TAP utility extracts this information.

TCP/IP for z/OS

For CICS IBM TCP/IP environments, you can use the Netstat facility to check the status of TCP/IP connections, as well as to make them inactive if problems occur. You can use the IBM TCP/IP trace facility to trace traffic between TRS and the mainframe. Also, check the CICS message user log for any system or Sybase messages.

Gateway-Library support (Server Option user-written applications and RPCs)

Note Skip this section if you are not using Server Option RPCs.

The Gateway-Library is a set of functions available for writing applications to enable mainframe environments to communicate with clients attached to TRS. These functions convert client calls into the TDS needed to communicate with TRS and its clients.

Stubs provide access to the Gateway-Library functions. These stubs are a set of object libraries that application programmers can include in job steps used to link-edit programs they create.

Gateway-Library tracing functions enable you to trace program activity globally, for all transactions, or specifically, for individual transactions. Based on the transaction processor, tracing functions provide:

- AAPI tracing for Gateway-Library calls, using the CICS auxiliary (aux) facility
- TDS header tracing, using the CICS Error Log
- TDS data tracing, using the CICS Error Log

Table G-1 shows the tracing functions:

Table G-1: Gateway-Library tracing functions

Function	Description
TDINFLOG	Determines what types of tracing are set
TDINFSPT	Indicates whether tracing is on or off for a transaction and returns the transaction ID
TDLSTSPT	Lists all transactions for which tracing is enabled
TDSETLOG	Turns system-wide tracing options on or off
TDSETSPT	Turns tracing on or off for a specific transaction
TDWRTLOG	Writes a user message or system entry

You can use standard CICS debugging tools or third party debugging tools to debug user-written applications.

For more information, see:

- Chapter 5, “Tracing and Accounting,” which describes the logging processes.

- The Mainframe Connect Server Option *Programmers Reference* for the appropriate programming language, which describes Gateway-Library tracing functions. PL/1 and COBOL versions of this guide are available.

For CICS, Gateway-Library tracing stores information about the TDS traffic between the mainframe and workstation in the VSAM ESDS file, *SYTDLOG1*. This information includes any errors detected in the traffic.

Remember that some TDS calls fill up internal TDS buffers before sending them out to the network. For example, a TDSNDROW or TDSNDMSG call does not cause execution of a corresponding CICS EXEC SEND call unless the TDS buffer becomes full.

Warning! To avoid losing records, periodically archive or delete the trace records on *SYTDLOG1*. Trace records are appended to this file until it is full; then the records are rejected.

DB2 UDB Option for CICS support

The DB2 UDB Option for CICS provides alternatives to creating Server Option applications:

- The AMD2 transaction for the DB2 UDB Options for CICS and for IMS, which will automatically process client SQL language requests using the DB2 UDB dynamic SQL facilities.
- A set of Catalog RPCs, an interface for accessing DB2 UDB catalog information that includes DB2 UDB tables and views. This interface resides on the LAN-side of the DirectConnect for z/OS Option in a DB2 UDB Option for CICS configuration.

Using this product, client applications can communicate directly with TRS or with another server that communicates with TRS, such as Adaptive Server Enterprise (ASE).

Tracing

For the DB2 UDB Option for CICS, you can use the standard TRS tracing support. See “TRS support” on page 127.

Logging

For further information about logging with DB2 UDB Option for CICS installations, refer to the Mainframe Connect DB2 UDB Option for CICS *Installation and Administration Guide*.

Gateway-less support

In a gateway-less architecture, the client application communicates directly with the mainframe. Consequently, you must use a TDS capture application or a sniffer application for tracing where a gateway trace might otherwise be used. At the mainframe, you can use the Sybase DEBUGSW option and listener trace flags to cause messages to be written to the CICS message user log or a CICS AUX trace.

Coordinating troubleshooting efforts

This section contains the following subsections:

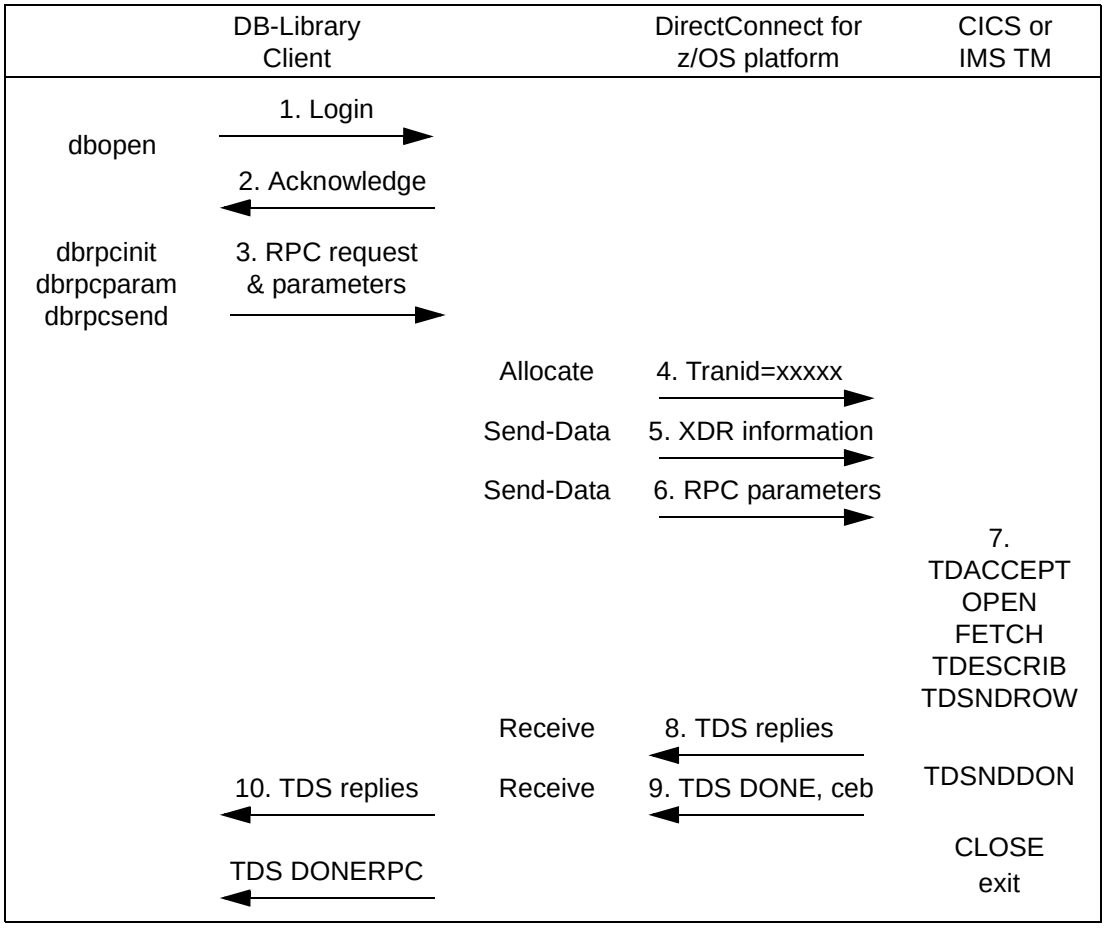
- Processing flow and requirements
- Process flow during attention sequences
- Browse applications

System administrators at the mainframe, TRS, and client level need to coordinate troubleshooting efforts. To help you with your analysis, this section describes the processing flow from the client through TRS to the mainframe.

Processing flow and requirements

The following diagram shows the processing flow:

Figure G-2: Client-to-TRS-to-mainframe processing flow



The following steps describe the sequence shown in Figure G-2 and highlight the requirements:

1 If TRS has started, the client opens a LAN connection to a designated DirectConnect for z/OS Option server and logs in. This message may appear:

```
Server name not found in interface file
```

If so, make sure that:

- The client interfaces file is set up correctly.
- The client Sybase path variable (SYBASE) is defined correctly.

- The DirectConnect for z/OS Option server is specified in the *DSQUERY* variable.

2 When it receives the client login information, the DirectConnect for z/OS Option checks security:

- If security is enabled, the DirectConnect for z/OS Option ensures that the client is authorized. If the client is not authorized, this error appears:

```
Security Violation: Login denied (no login entry)
```

- If the client is authorized or security is disabled, the DirectConnect for z/OS Option acknowledges the login.

3 When the client application needs to invoke an RPC or language request on the mainframe, the client sends a request to TRS over the logged-in LAN connection.

4 TRS receives the request and performs a table look-up to find the mainframe session and Server Option transaction ID to use. The RPC and connection must be in the table. If security is enabled, the client must be authorized to use the RPC and connection to the mainframe. If the table look-up and security check are successful, the line is up, and the session is active, TRS allocates a conversation with the named transaction.

If a failure occurs during this process, SNA Services writes one of the following error messages to both the TRS log and the client:

```
Security Violation: Access to RPC 'xxxx' denied.
```

The client is not authorized or is not listed correctly.

```
Request Rejected: No host connections are
available.
```

Connections to the mainframe are unavailable.

```
Request Rejected: Remote procedure 'xxxx' not
found.
```

The RPC name was entered incorrectly or the name is not in the lookup table.

5 TRS sends the client External Data Representation (XDR) information to the mainframe.

6 TRS sends the client RPC parameters to the mainframe, and then waits for a reply from the transaction.

7 On the mainframe, the transaction processor initiates the named transaction, and the transaction issues the Server Option Gateway-Library calls. These calls read the client XDR information and RPC parameters. The transaction also performs associated processing, such as issuing static SQL DB2 UDB requests or reading VSAM or other database data.

8 The transaction issues Gateway-Library calls that send results back to the client. These calls perform required data conversions, generate the TDS reply data stream, and send out reply data.

9 TRS receives the TDS reply packet and forwards it to the client, which continues until the Server Option transaction issues a TDSNDDON call.

If a failure occurs during this process, the LAN SNA software writes an error message to the DirectConnect for z/OS Option server log. It also writes this error message to the client:

```
Unexpected EOF from Adaptive Server Enterprise
```

(The mainframe is acting as a ASE.) If in use, the Gateway-Library tracing functions also record errors in this process.

10 When the request is complete, the transaction exits and the conversation terminates. A long-running transaction (also called a user-defined transaction) can remain active through multiple requests before the conversation ends. If a long-running transaction terminates before it should, determine whether appropriate client support is set up. For example:

- The client may be set up to disconnect after invoking the transaction and before the transaction ends.
- ASE logs out after sending a client request and, therefore, does not support long-running transactions.

For more information on identifying problems, see “Common problems and suggested solutions” on page 120.

Process flow during attention sequences

Any of the following actions results in an attention sequence:

- Database-Library issues a dbcancel() command.
- An isql user cancels processing while the server is sending results.
- An APT program or form issues a closesql command.

- A Data Workbench user exits a form while the server is sending results.

When an attention sequence is issued, the process flow is as follows:

1 Database-Library issues an attention packet to TRS, then discards anything else received until it receives a TDS DONE packet with the attention Ack bit on.

2 TRS converts the attention packet into a SNA SIGNAL command, issuing an LU 6.2 request-to-send verb. TRS then discards any results received from the mainframe until it receives a TDS DONE packet with the attention Ack bit on.

3 At the mainframe, the Server Option receives the attention signal and passes it to the Gateway-Library (RPC).

4 Gateway-Library passes back a return code, indicating TDS_CANCEL_RECEIVED, on all subsequent TDSNDROW, TDSNDMSG, and TDSETPRM calls from an application. Any data associated with TDSNDROW or TDSNDMSG calls is discarded until the application issues a TDSNDDON call.

   For details on these calls, see the Mainframe Connect Server Option *Programmers Reference* for the appropriate programming language. PL/1 and COBOL versions of this guide are available.

5 When the application issues a TDSNDDON call, Server Option support sends a TDS DONE packet with the attention Ack bit on. This ends the attention sequence.

Browse applications

Sybase architecture uses a “streaming mode” of data transfer. Rather than sending a short block of data and waiting for a reply, the mainframe continuously sends data until the client stops accepting it. When the client stops accepting data, normal SNA pacing functions suspend data transfer.

For applications that select a small set of data, process it, then request the next block of data, it is best to use RPC parameters to specify the ID of a set of records. If the client RPC parameters are set up as return parameters, and the Gateway-Library TDSETPRM specifies the ID of the desired set of records, Server Option support returns the updated RPC value to the client. The client can use this value to invoke the next set of records.

Glossary

accept — Establishment of a SNA or TCP/IP connection between Mainframe Connect Server Option and Mainframe Connect DirectConnect for z/OS Option.

access service — The named set of properties, used with an access service library, to which clients connect. Each DirectConnect server can have multiple services.

access code — A number or binary code assigned to programs, documents, or folders that allows authorized users to access them.

access service library — A service library that provides access to non-Sybase data contained in a database management system or other type of repository. Each such repository is called a "target." Each access service library interacts with exactly one target and is named accordingly. See also **service library**.

ACSLIB — See **access service library**.

Adaptive Server Enterprise — The server in the Sybase client/server architecture. It manages multiple databases and multiple users, tracks the actual location of data on disks, maintains mapping of logical data description to physical data storage, and maintains data and procedure caches in memory.

Adaptive Server Enterprise/Component Integration Services — Includes a variation of ASE that provides a Transact-SQL interface to various sources of external data. Component Integration Services allows ASE to present a uniform view of enterprise data to client applications.

administrative service library — A service library that provides remote management capabilities and server-side support. It supports a number of remote procedures, invoked as RPC requests, that enable remote DirectConnect server management. See also **remote procedure call**, **service library**.

ADMLIB — See **administrative service library**.

Advanced Interactive Executive — The IBM implementation of the UNIX operating system. The RISC System/6000, among other workstations, runs the AIX operating system.

advanced program-to-program communication — Hardware and software that characterize the LU 6.2 architecture and its implementations in products. See also **logical unit 6.2**.

AIX See **Advanced Interactive Executive.**

AMD2 The component of the Mainframe Connect DB2 UDB Option that allows clients to submit SQL statements to DB2 UDB. It is a CICS transaction that receives SQL statements sent from Mainframe Connect DirectConnect for z/OS Option and submits them to DB2 UDB, using the DB2 UDB dynamic SQL facility. It also receives the results and messages from DB2 UDB and returns them to Mainframe Connect DirectConnect for z/OS Option.

American Standard Code for Information Interchange The standard code used for information interchange among data processing systems, data communication systems, and associated equipment. The code uses a coded character set consisting of 7-bit coded characters (including a parity check, 8 bits).

API See **application program interface**.

APPC See **advanced program-to-program communication**.

application program interface The programming language interface between the user and Mainframe Connect Client Option or Mainframe Connect Server Option. The API for Mainframe Connect Client Option is Client-Library. The API for Mainframe Connect Server Option is Gateway-Library.

ASCII See **American Standard Code for Information Interchange**.

ASE See **Adaptive Server Enterprise**.

ASE/CIS See **Adaptive Server Enterprise/Component Integration Services**.

batch A group of records or data processing jobs brought together for processing or transmission.

bind In the Sybase environment, this term has different meanings depending on the context:

- In CICS, it is an SNA command used to establish a connection between LUs, or a TCP/IP call that connects an application to a port on its system.
- In DB2 UDB, it compiles the Database Request Module, the precompiler product that contains SQL statements in the incoming request, and produces an access plan, a machine code version of the SQL statements that specifies the optimal access strategy for each statement.
- In the mainframe access product set, it establishes a connection between a TRS port and a CICS or IMS region.

bulk copy transfer A transfer method in which multiple rows of data are inserted into a table in the target database. Compare with **destination-template transfer** and **express transfer**.

call level interface A programming style that calls database functions directly from the top level of the code. Contrast with **embedded SQL.**

catalog A system table that contains information about objects in a database, such as tables, views, columns, and authorizations.

catalog RPC A component of the Mainframe Connect DB2 UDB Option that allows clients to access DB2 UDB system catalogs. It uses an interface compatible with the catalog interface for the ODBC API.

catalog stored procedure A procedure used in SQL generation and application development that provides information about tables, columns, and authorizations.

character set A set of specific (usually standardized) characters with an encoding scheme that uniquely defines each character. ASCII is a common character set.

CICS See **Customer Information Control System**.

CICS region The instance of CICS.

client In client/server systems, the part of the system that sends requests to servers and processes the results of those requests. See also **client/server**. Compare with **server**.

client application Software responsible for the user interface that sends requests to applications acting as servers. See also **client/server**.

Client-Library A library of routines that is part of Mainframe Connect Client Option.

client request An RPC or language request sent by a client to a server.

client/server An architecture in which the client is an application that handles the user interface and local data manipulation functions, and the server is an application providing data processing access and management. See also **client application**.

Client Services Application A customer-written CICS program initiated on the host that uses the API to invoke the Mainframe Connect Client Option as a client to the Server Option server or to ASE. See also **application program interface**, **Client Services for CICS**.

Client Services for CICS
A Sybase host API that invokes the Mainframe Connect Server Option as a client to an access service for DB2 UDB or ASE. See also **application program interface**, **Customer Information Control System**, **Client Services Application**, **Mainframe Connect Server Option**.

clustered index
An index in which the physical order and the logical (indexed) order is the same. Compare with **nonclustered index**.

code page
An assignment of graphic characters and control function meanings to all code points.

commit
A process that makes permanent all changes made to one or more database files since the initiation of the application program, the start of an interactive session, or the last commit or rollback operation. Compare with **rollback**.

Common Programming Interface
Specifies the languages and services used to develop applications across SAA environments. The elements of the CPI specification are divided into two parts: processing logic and services.

configuration file
A file that specifies the characteristics of a system or subsystem.

configuration set
A section into which service library configuration files are divided.

conversion
The transformation between values that represent the same data item but which belong to different datatypes. Information can be lost due to conversion, because accuracy of data representation varies among different datatypes.

connection
A network path between two systems. For SNA, the path connects a logical unit (LU) on one machine to an LU on a separate machine. For TCP/IP, the path connects TCP modules on separate machines.

connection router
A program provided with Mainframe Connect Client Option that directs requests to particular remote servers. Mainframe system programmers use the connection router to define remote servers and server connections to Mainframe Connect Client Option.

Connection Router Table
A memory-resident table maintained by a Mainframe Connect Client Option system programmer that lists servers and the connections that a Client-Library transaction can use to access them.

control section
The part of a program specified by the programmer to be a relocatable unit, all elements of which are to be loaded into adjoining main storage locations.

control statement
In programming languages, a statement that is used to alter the continuous sequential execution of statements. A control statement can be a conditional statement or an imperative statement.

conversation-level security — The passing of client login information to the mainframe by TRS when it allocates a conversation.

CSA — See **Client Services Application**.

CSP — See **catalog stored procedure**.

cursor — In SQL, a named control structure used by an application program to point to a row of data.

Customer Information Control System — An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs.

DASD — See **direct access storage device**.

data definition statement — An IBM mainframe statement used to relate a name with a file.

data definition language — A language for describing data and data relationships in a database.

data set name — The term or phrase used to identify a data set.

database management system — The term or phrase to identify a data set.A computer-based system for defining, creating, manipulating, controlling, managing, and using databases.

database operation — A single action against the database. For Mainframe Connect DirectConnect for z/OS Option, a database operation is usually a single SQL statement. One or more database actions can be grouped together to form a request. See also **request**.

Database 2 — An IBM relational database management system.

datatype — A keyword that identifies the characteristics of stored information on a computer.

DB-Library — A Sybase and Microsoft API that allows client applications to interact with ODS applications. See also **application program interface**.

DBMS — See **database management system**.

DB2 UDB — See **Database 2**.

DDL — See **data definition language**.

DD statement — See **data definition statement**.

default language — The language that displays a user's prompts and messages.

destination-template transfer A transfer method in which source data is briefly put into a template where the user can specify that some action be performed on it before execution against a target database. See also **transfer**. Compare with **bulk copy transfer** and **express transfer**.

direct access storage device A device in which access time is effectively independent of the location of the data.

direct request A request sent directly from a client workstation through Transaction Router Service to the DirectConnect server without going through ASE. Contract with **indirect request**.

direct resolution A type of service name resolution that relies upon a client application specifying the exact name of the service to be used. See also **service name resolution**. Compare with **service name redirection**.

DirectConnect Manager A Java application from Sybase that can be used in Windows and UNIX environments. It provides remote management capabilities for DirectConnect products, including starting, stopping, creating, and copying services.

Server Option server The component of Mainframe Connect DirectConnect for z/OS Option that provides general management and support functions to service libraries.

dll See **dynamic link library**.

DSN See **data set name**.

dynamic link library A file containing executable code and data bound to a program at load time or runtime, rather than during linking.

dynamic SQL The preparation and processing of SQL source statements within a program while the program runs. The SQL source statements are contained in host-language variables rather than being coded directly into the application program. Contrast with **static SQL**.

ECDA See **Enterprise Connect Data Access**.

ECDA Option for ODBC A Sybase solution that allows client applications to access ODBC data. It combines the functionality of the ECDA Option for ODBC architecture with ODBC to provide dynamic SQL access to target data, as well as the ability to support stored procedures and text and image pointers.

ECDA Option for Oracle A Sybase solution that provides Open Client access to Oracle databases. When used in combination with ASE, it provides many of the features of a distributed database system, such as location transparency, copy transparency, and distributed joins.

embedded SQL SQL statements that are embedded within a program and are prepared in the process before the program runs. After it is prepared, the statement itself does not change, although values of host variables specified within the statement might change.

end user A person who connects to a DirectConnect server using an application to access databases and perform transfers. See also **transfer**.

Enterprise Connect Data Access An integrated set of software applications and connectivity tools that allow access to data within a heterogeneous database environment, such as a variety of LAN-based, non-Sybase data sources, as well as mainframe data sources.

environment variable A variable that describes how an operating system runs and the devices it recognizes.

exit routine A user-written routine that receives control at predefined user exit points.

express transfer A form of bulk copy transfer that uses ODBC bulk APIs to improve performance when transferring bulk data between data sources. Because it uses the same syntax as bulk copy transfer, no modification of applications is required.

external call interface A CICS client facility that allows a program to call a CICS application as if the calling program had been linked synchronously from a previous program instead of started from a terminal.

External Security Manager An add-on security package for the z/OS mainframe, licensed by Computer Associates.

FCT See **forms control table**.

forms control table An object that contains the special processing requirements for output data streams received from a host system by a remote session.

gateway Connectivity software that allows two or more computer systems with different network architectures to communicate.

Gateway-Library A library of communication, conversion, tracing, and accounting functions supplied with Mainframe Connect Server Option.

globalization The combination of internationalization and localization. See **internationalization**, **localization**.

global variable A variable defined in one portion of a computer program and used in at least one other portion of the computer program. Contrast with **local variable**.

handler A routine that controls a program's reaction to specific external events, for example, an interrupt handler.

host The mainframe or other machine on which a database, an application, or a program resides. In TCP/IP, this is any system that is associated with at least one Internet address. See also **Transmission Control Protocol/Internet Protocol**.

host ID In Mainframe Connect Server Option, the ID that the TRS passes to the mainframe with a client request. The host ID is part of the client login definition at the TRS.

host password In Mainframe Connect Server Option, the password that the client passes to the mainframe with a client request.

host request library A DB2 UDB table that contains host-resident SQL statements that can be executed dynamically. See also **host-resident request**.

host-resident request A SQL request that resides in a DB2 UDB table called the host request library. See also **host request library**.

IMS See **Information Management System**.

indirect request A client request that is routed through a stored procedure on a SQL Server, which forwards the request to TRS as an RPC. Compare with **direct request**.

Information Management System A database/data communication system that can manage complex databases and networks.

interfaces file An operating system file that determines how the host client software connects to a Sybase product. An *interfaces* file entry contains the name of any Server Option server and a list of services provided by that server.

internationalization The process of extracting locale-specific components from the source code and moving them into one or more separate modules, making the code culturally neutral so it can be localized for a specific culture. See also **globalization**. Compare with **localization**.

iSeries Previously known as the AS/400 computer, supports Linux, UNIX, and Windows platforms.

keyword A word or phrase reserved for exclusive use by Transact-SQL.

language RPC The name TRS uses to represent a client's language request. TRS treats a language request as a remote procedure call (RPC) and maps it to a language transaction at the remote server.

language transaction
The server transaction that processes client language requests. The Mainframe Connect DB2 UDB Option language transaction for CICS is AMD2, which uses the DB2 UDB dynamic SQL facilities to process incoming SQL strings. The Mainframe Connect DB2 UDB Option for IMS uses SYRT by default.

linkage
In computer security, combining data or information from one information system with data or information from another system with the intention to derive additional information; for example, the combination of computer files from two or more sources.

linkage editor
A computer program that creates load modules from one or more object modules or creates load modules by resolving cross references among the modules, and if necessary, adjusts those addresses.

link-edit
To create a loadable computer program by using a linkage editor. See also **linkage editor**.

localization
The process of preparing an extracted module for a target environment, in which messages are displayed and logged in the user's language. Numbers, money, dates, and time are represented using the user's cultural convention, and documents are displayed in the user's language. See also **globalization**.

local variable
A variable that is defined and used only in one specified portion of a computer program. Contrast with **global variable**.

logical unit
A type of network addressable unit that enables a network user to gain access to network facilities and communicate remotely. A connection between a TRS and a CICS region is a connection between logical units.

logical unit 6.2
A type of logical unit that supports general communication between programs in a distributed processing environment. See also **advanced program-to-program communication**.

login ID
In Mainframe Connect Server Option, the ID that a client user uses to log in to the system.

login packet
Client information made available to Mainframe Connect Server Option. The client program sets this information in a login packet and sends it to TRS, which forwards it to the mainframe.

long-running transaction
A transaction that accepts more than one client request. Whereas short transactions end the communication after returning results to a client, a long-running transaction can await and process another request. Compare with **short transaction**.

LU 6.2
See **logical unit 6.2**.

mainframe access products — Sybase products that enable client applications to communicate with mainframes in a client/server environment. See **client/server**.

Mainframe Connect — The Sybase product set that provides access to mainframe data.

Mainframe Connect Client Option — A Sybase product that, using Client-Library, allows mainframe clients to send requests to SQL Server, Open Server, the Mainframe Connect DB2 UDB Option and Mainframe Connect Server Option. Mainframe Connect Client Option provides capability for the mainframe to act as a client to LAN-based resources in the CICS or the IMS and MVS environment.

Mainframe Connect DB2 UDB Option — A Sybase mainframe solution that provides dynamic access to DB2 UDB data. It is available in the CICS or IMS environment. See also **Customer Information Control System**, **Database 2**, **Multiple Virtual Storage**.

Mainframe Connect Server Option for z/OS Option — A Sybase Open Server application that provides access management for non-Sybase databases, copy management (transfer), and remote systems management.

Mainframe Connect Server Option — A Sybase product that provides capability for programmatic access to mainframe data. It allows workstation-based clients to execute customer-written mainframe transactions remotely. It is available for the CICS and the IMS and MVS environments

Multiple Virtual Storage — An IBM operating system that runs on most System/370 and System/390 mainframes. It supports 24-bit addressing up to 16 megabytes.

network protocol — A set of rules governing the way computers communicate on a network.

nonclustered index — An index that stores key values and pointers to data. Compare with **clustered index**.

null — Having no explicitly assigned value. NULL is not equivalent to 0 or to blank.

ODBC — See **Open Database Connectivity**.

ODS — See **Open Data Services**.

Open Client — A Sybase product that provides customer applications, third-party products, and other Sybase products with the interfaces required to communicate with Open Client and Open Server applications.

Open Data Services — A product that provides a framework for creating server applications that respond to DB-Library clients.

Open Database Connectivity — A Microsoft API that allows access to both relational and non-relational databases. See also **application program interface**.

Open Server — A Sybase product that provides the tools and interfaces required to create a custom server. Clients can route requests to the Server Option server through an Open Server configured to meet specific needs, such as the preprocessing of SQL statements.

parameter — A variable that is given a constant value for a specified application and can denote the application. Compare with **property**.

Partner Certification Reports — Sybase publications that certify third-party or Sybase products to work with other Sybase products.

Password Expiration Management — An IBM password management program with CICS Version 3.3 through an optional program temporary fix, and as an integral part of CICS with version 4.1 and higher.

PEM — See **Password Expiration Management**.

PL/1 — See **Programming Language /1**.

primary database — The database management system that the DirectConnect server is always connected to. It is implied in the transfer statement.

Programming Language/1 — A programming language designed for use in a wide range of commercial and scientific computer applications.

property — A setting for a server or service that defines the characteristics of the service, such as how events are logged. Compare with **parameter**.

protocol — The rules for requests and responses used to manage a network, transfer data, and synchronize the states of network components.

query — A request for data from a database, based upon specified conditions.

Registry — The part of the Windows operating system that holds configuration information for a particular machine.

relational database — A database in which data is viewed as being stored in tables consisting of columns (data items) and rows (units of information).

relational operators — Operators supported in search conditions.

relops — See **relational operators**.

remote procedure call — A call to execute a stored procedure on a remote server. For Mainframe Connect Server Option, an RPC is a direct request from a client to TRS. For Mainframe Connect Client Option, a Client-Library transaction that calls a procedure on a remote server acts like an RPC.

remote stored procedure — A customer-written CICS program using an API that resides on the mainframe and communicates with Mainframe Connect DB2 UDB Option. See also **Customer Information Control System**, **stored procedure**. Compare with **Client Services Application**.

remote systems management — A feature that allows a system administrator to manage multiple DirectConnect servers and multiple services from a client.

Replication Server — A Sybase SQL Server application that maintains replicated data and processes data transactions received from a data source.

request — One or more database operations an application sends as a unit to the database. Depending upon the response, the application commits or rolls back the request. See also **commit**, **rollback**, **unit of work**.

resource table — A main storage table that associates each resource identifier with an external logical unit (LU) or application program.

rollback — An instruction to a database to back out of changes requested in a unit of work. Compare with **commit**.

router — An attaching device that connects two LAN segments, which use similar or different architectures, at the Open System Interconnection (OSI) reference model network layer. Contrast with **gateway**.

RPC — See **remote procedure call.**

RSP — See **remote stored procedure**.

SAA — See **System Application Architecture**.

secondary connection — The connection specified in the transfer statement. It represents anything that can be accessed using Mainframe Connect Client Option, such as ASE or another access service.

secondary database — In transfer processing, the supported database that is specified in the transfer statement. Compare with **primary database**.

server — A functional unit that provides shared services to workstations over a network. See also **client/server**. Compare with **client**.

server process ID — A positive integer that uniquely identifies a client connection to the server.

service — A functionality available in Mainframe Connect DirectConnect for z/OS Option. It is the pairing of a service library and a set of specific configuration properties.

service library — In Mainframe Connect DirectConnect for z/OS Option, a set of configuration properties that determine service functionality. See also **access service library**, **administrative service library**, **Transaction Router Service library**, **transfer service library**.

service name redirection — A type of service name resolution that allows a system administrator to create an alternative mechanism to map connections with services. See also **service name resolution**. Compare with **direct resolution**.

service name redirection file — The default name of the file used for the service name redirection feature. See **service name redirection**.

service name resolution — The DirectConnect server mapping of an incoming service name to an actual service. See also **direct resolution**, **service name redirection**.

session — A connection between two programs or processes. In APPC communications, sessions allow transaction programs to have conversations between the partner LUs. See also **advanced program-to-program communication**.

short transaction — A mainframe transaction that ends the communication when it finishes returning results to the client. Compare with **long-running transaction**.

SNA — See **Systems Network Architecture**.

SNRF — See **service name redirection file**.

SPID — See **server process ID**.

SQL — See **structured query language**.

SQLDA — See **SQL descriptor area**.

sqledit — A utility for creating and editing *sql.ini* files and file entries.

sql.ini — The interfaces file containing definitions for each Server Option server to which a workstation can connect. The file must reside on every client machine that connects to ASE.

SQL descriptor area — A set of variables used in the processing of SQL statements.

SQL stored procedure — A single SQL statement that is statically bound to the database. See also **stored procedure**.

static SQL — SQL statements that are embedded within a program and prepared during the program preparation process before the program runs. Compare with **dynamic SQL**.

stored procedure — A collection of SQL statements and optional control-of-flow statements stored under a particular name. Adaptive Server stored procedures are called "system procedures." See also **remote stored procedure**, **system procedures**.

structured query language — An IBM industry-standard language for processing data in a relational database.

stub — A program module that transfers remote procedure calls (RPCs) and responses between a client and a server.

SYRT — The component of Mainframe Connect DB2 UDB for IMS that allows clients to submit SQL language requests to DB2 through IMS.

System Administrator — The person in charge of server system administration, including installing and maintaining DirectConnect servers and service libraries.

System Application Architecture — An IBM proprietary plan for the logical structure, formats, protocols, and operational sequences for transmitting information units through networks and controlling network configuration and operation. See also **advanced program-to-program communication**.

system procedures — A stored procedure that ASE supplies for use in system administration. System procedures serve as shortcuts for retrieving information from system tables, or a mechanism for accomplishing database administration. See also **stored procedure**.

Systems Network Architecture — An IBM proprietary plan for the structure, formats, protocols, and operational sequences for transmitting information units through networks. See also **advanced program-to-program communication**.

table — An array of data or a named data object that contains a specific number of unordered rows. Each item in a row can be unambiguously identified by means of one or more arguments.

Tabular Data Stream — A Sybase application-level protocol that defines the form and content of relational database requests and replies.

target — A system, program, or device that interprets, rejects, satisfies, or replies to requests received from a source.

target database — The database to which the DirectConnect server transfers data or performs operations on specific data.

TCP/IP — See **Transmission Control Protocol/Internet Protocol**.

TDS — See **Tabular Data Stream**.

transaction A unit of processing initiated by a single request. A transaction consists of one or more application programs that, when executed, accomplish a particular action. In Mainframe Connect Server Option, a client request (RPC or language request) invokes a mainframe transaction. In Mainframe Connect Client Option, a mainframe transaction executes a stored procedure on a remote server.

transaction processing A sequence of operations on a database that is viewed by the user as a single, individual operation.

Transaction Router Service A Mainframe Connect DirectConnect for z/OS Option program used when the mainframe acts as a transaction server to route requests from remote clients to the Mainframe Connect Server Option and return results to the clients.

Transaction Router Service library A service library that facilitates access to remote transactions, allowing customers to execute transactions from virtually any mainframe data source. See also **service library**.

Transact-SQL A Sybase-enhanced version of the SQL database language used to communicate with ASE.

transfer A Mainframe Connect DirectConnect for z/OS Option feature that allows users to move data or copies of data from one database to another.

transfer service library A service library that provides copy management functionality. See also **service library**.

Transmission Control Protocol/Internet Protocol A set of communication protocols that supports peer-to-peer connectivity functions for both local and wide area networks.

trigger A form of stored procedure that automatically executes when a user issues a change statement to a specified table.

TRS See **Transaction Router Service**.

TRS library See **Transaction Router Service library**.

T-SQL See **Transact-SQL**.

unit of work One or more database operations grouped under a commit or rollback. A unit of work ends when the application commits or rolls back a series of requests, or when the application terminates. See also **commit**, **rollback**, **transaction**.

user ID User identification. The ID number by which a user is known in a specific database or system.

variable An entity that is assigned a value. Mainframe Connect Server Option for z/OS Option has two kinds of variables: *local* and *global.*

view An alternate representation of data from one or more tables. A view can include all or some of the columns contained the table or tables on which it is defined.

Virtual Storage Access Method An IBM-licensed program that controls communication and the flow of data in an SNA network.

Virtual Telecommunications Access Method IBM mainframe software that allows communication on an SNA network between mainframes and allows the mainframe to have multiple sessions per connection.

VSAM See **Virtual Storage Access Method.**

VTAM See **Virtual Telecommunications Access Method**.

wildcard A special character that represents a range of characters in a search pattern.

zSeries IBM family of hardware products, including servers and connectivity devices.

Index

A

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

W

X

Z