CICS Universal Client

UNIX and Linux Client Administration

Version 7.1

SC34-6755-02

CICS Universal Client

UNIX and Linux Client Administration

Version 7.1

SC34-6755-02

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 139.

Third Edition (February 2008)

This edition applies to Version 7.1 of CICS Universal Client for Linux, program number 5724-J09. It will also apply to all subsequent versions, releases, and modifications until otherwise indicated in new editions.

This edition replaces SC34-6653. Technical changes to the text are indicated by a vertical line to the left of the change.

Contents

About this book

This information describes the planning, installation, configuration, and operation of the IBM® CICS® Universal Client product.

You should be familiar with the operating system on which your CICS Universal Client runs.

This information is organized as shown in the following table:

"What's new on the CICS Universal Client" on page ix	Functional changes made in this version of CICS Universal Client.
Chapter 1, "CICS Universal Client overview," on page 1	Overview of CICS Universal Client and the functions it provides.
Chapter 2, "Planning your installation," on page 5	Planning your installation, including the hardware and software you need to run CICS Universal Client.
Chapter 3, "Installing CICS Universal Client," on page 13	How to install CICS Universal Client.
Chapter 4, "Configuring CICS server connections," on page 21	How to set up communication links between CICS Universal Client and CICS servers.
Chapter 5, "Configuring CICS Universal Client," on page 31	How to configure your CICS Universal Client.
Chapter 6, "Client security," on page 55	How to provide a user ID and password when connecting to a CICS server.
Chapter 7, "Performance," on page 59	How to tune your CICS Universal Client, and other system components, to achieve the best possible performance.
Chapter 8, "Operating the Client daemon," on page 61	How to operate the Client daemon.
Chapter 9, "Terminal Emulation," on page 71	How to use the cicsterm and cicsprnt programs.
Chapter 10, "Problem determination and problem solving," on page 81	Problem determination and problem solving.
Chapter 11, "Migration," on page 101	Migrating from an earlier version of CICS Universal Client.
"Data conversion," on page 105	Data conversion when using the Client daemon and a CICS server

Installation path

The term <install_path> is used in file paths to represent the directory where you installed the product. The default location for new installations is:

UNIX® platforms
`/opt/IBM/cicsuc`

Linux® platforms
`/opt/ibm/cicsuc`

If you are upgrading, the installation process keeps your existing directory structure.

What's new on the CICS Universal Client

CICS Universal Client Version 7.1 delivers extended interoperability in the following areas:

- TCP/IP network optimizations are provided to improve response times for Java clients connecting to CICS through the Gateway daemon.
- Support for fully qualified SNA partner LU names, provides for easier configuration of SNA clients in an APPN network, simplifying the migration of TCP62 connections to Enterprise Extender for customers who continue to require APPC interconnectivity in a TCP/IP network.

Removed and changed function

For information on removed and changed function, see "Migrating to Version 7.1" on page 101.

Chapter 1. CICS Universal Client overview

The CICS Universal Client enables users to access transactions and programs on the entire family of CICS application servers from their desktop. It can communicate with multiple CICS servers using a variety of protocols; see "Communication with CICS servers" on page 10. You use the Configuration Tool to determine the settings for client operation, and identify the associated servers and protocols used for communication (see "Using the Configuration Tool" on page 31).

The external access interfaces (ECI, EPI, ESI)

The external interfaces allow non-CICS applications to access and update CICS resources by calling CICS programs or by initiating CICS transactions. When used in conjunction with CICS communication, the external interfaces enable non-CICS programs to access and update resources on any CICS system. This method of using the external interfaces supports such activities as the development of graphical user interface (GUI) front ends for CICS applications and it allows the integration of CICS systems and non-CICS systems.

External Call Interface (ECI)

The ECI enables a user application to call a CICS program synchronously or asynchronously. It enables the design of new applications to be optimized for client/server operation, with the business logic on the server and the presentation logic on the client.

External Presentation Interface (EPI)

The EPI enables a user application to act as a logical 3270 terminal and to control a CICS 3270 application. It enables modern technologies, such as graphical or multimedia interfaces, to be used with traditional CICS 3270 applications.

External Security Interface (ESI)

The ESI enables user applications to verify and change passwords for specified user IDs that are managed by an external security manager (ESM) on a CICS server.

For more information on the external access interfaces, see *CICS Transaction Gateway: Programming Guide* and *CICS Transaction Gateway: Programming Reference*.

3270 terminal emulation (cicsterm)

CICS 3270 terminal emulation enables a CICS Universal Client system to operate as a 3270 display for CICS applications, without needing a separate 3270 emulator product. Multiple CICS 3270 emulation sessions can run to one or more CICS servers.

You can use mapping files to customize the screen color attributes and keyboard settings of the client emulator, for example, to comply with company standard keyboard layouts.

CICS Client terminal definitions are autoinstalled on the CICS server and do not have to be predefined.

3270 printer support (cicsprnt)

CICS 3270 printer support allows you to define a printer terminal on a CICS Universal Client system. This support enables CICS applications running on a server to send output to a printer attached to the CICS Universal Client.

You can send output to a physical printer or you can specify a command to process the data into a format more suitable for special-purpose printers.

CICS 3270 printer support uses CICS 3270 terminal emulation. Table 1 on page 10 in topic "Communication with CICS servers" on page 10 tells you which CICS servers currently support CICS 3270 emulation and hence CICS 3270 client printer support.

Client control

Commands exist to:

- **Control the client process**

  You can:
  - Start or stop the client process
  - Turn client trace on or off
  - Specify the client components to be traced
  - Set up security by specifying user IDs and passwords for a CICS server
  - List connected servers
  - Enable and disable the display of pop-up messages
  - Perform controlled restarts of the client process
- **Control terminal emulation**

  You can:
  - Start and stop the terminal emulator
  - Specify the initial transaction
  - Define the terminal characteristics
  - Specify the name of the keyboard and screen color mapping files
  - Define the command used to process print requests
  - Specify the name of a file used for appending print requests.
- **Control client printer operation**

  You can:
  - Start and stop the client printer emulator
  - Specify the initial transaction to be run against the client printer
  - Define the printer terminal characteristics
  - Define the command used to process print requests
  - Specify the name of a file used for appending print requests.

Dynamic Logical Partitioning (DLPAR, AIX only)

The CICS Universal Client for AIX® is a DLPAR-safe application. This means that it does not fail as a result of DLPAR operations. Its performance might suffer when resources are removed, and it might not scale when new resources are added, but the program still works as expected.

Note: The CICS Universal Client does not manage I/O devices such as network adapters. Before removing such devices dynamically, ensure that they are no longer being used by the CICS Universal Client, or any of its underlying network components.

Chapter 2. Planning your installation

This information helps you to plan the installation of the CICS Universal Client by listing the hardware and software that you need. The CICS servers to which CICS Universal Client can connect are also listed.

If you are using an earlier version of the product see Chapter 11, “Migration,” on page 101 for a discussion of migration issues.

See the product readme file for any late changes to hardware and software requirements:

- If you are installing from CD, the readme file is in the root directory:

Operating system	README file
AIX	README.AIX
Linux on Intel®	README.LNX
Linux on POWER™	README.pLNX
Linux on zSeries®	README.zLNX
Solaris	README.SOL

- If you downloaded the product, the packaged file that you downloaded contains the product code and the readme file for your operating system.

When you install the CICS Universal Client, the readme file is copied into your installation directory as readme.txt.

Hardware requirements

CICS Universal Client requires the following hardware:

- An IBM eServer™ zSeries system or S/390® system supported by Linux
- An Intel 32-bit system supported by Linux
- An IBM eServer pSeries®, iSeries®, or Open Power™ system supported by Linux
- An IBM eServer pSeries system supported by AIX
- A Sun SPARC system supported by Sun Solaris
- An HP PA-RISC 1.1 or 2.0 system supported by HP-UX
- An HP Itanium system supported by HP-UX

Supported software

For the latest details about supported software, visit the Web site at `http://www.ibm.com/software/cics/cuc/support`.

The CICS Universal Client works with the following products:

Supported operating systems

CICS Universal Client is supported on the following operating systems:

- AIX V5.3. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels.

- AIX V6.1. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels.
- Linux on Intel - Novell Linux Desktop 9. CICS Universal Client is supported as a 32-bit application on the 32-bit kernel.
- Linux on Intel - SUSE Linux Enterprise Desktop 10. CICS Universal Client is supported as a 32-bit application on the 32-bit kernel.
- Linux on Intel - SUSE Linux Enterprise Server 9, with SP2 minimum service level. CICS Universal Client is supported as a 32-bit application on both the 32 and 64-bit kernels. Linux distributions that use OEM code from SUSE are also supported.
- Linux on Intel - SUSE Linux Enterprise Server 10. CICS Universal Client is supported as a 32-bit application on both the 32 and 64-bit kernels.
- Linux on Intel - Red Hat Enterprise Linux V3 with a minimum service level of Update 6. CICS Universal Client is supported as a 32-bit application on both the 32 and 64-bit kernelst. Only the NPTL threading model is supported.
- Linux on Intel - Red Hat Enterprise Linux V4 with a minimum service level of Update 2. CICS Universal Client is supported as a 32-bit application on both the 32 and 64-bit kernels.
- Linux on Intel - Red Hat Enterprise Linux V5.CICS Universal Client is supported as a 32-bit application on both the 32 and 64-bit kernels.
- Linux on POWER - SUSE Linux Enterprise Server V9. Linux distributions that use OEM code from SUSE are also supported. CICS Universal Client is supported as a 32-bit application on the 64-bit kernel.
- Linux on POWER - SUSE Linux Enterprise Server V10. CICS Universal Client is supported as a 32-bit application on the 64-bit kernel.
- Linux on POWER - Red Hat Enterprise Linux V4 with a minimum service level of Update 4. Only the NPTL threading model is supported. CICS Universal Client is supported as a 31-bit application on the 31-bit kernel.
- Linux on POWER - Red Hat Enterprise Linux V5. Only the NPTL threading model is supported. CICS Universal Client is supported as a 31-bit application on the 64-bit kernel.
- Linux on POWER - Red Hat Enterprise Linux V5. CICS Universal Client is supported as a 32-bit application on the 64-bit kernel.
- Linux on zSeries - SUSE Linux Enterprise Server V9. Linux distributions that use OEM code from SUSE are also supported. CICS Universal Client is supported as a 31-bit application on the 64-bit kernel.
- Linux on zSeries - SUSE Linux Enterprise Server V10. CICS Universal Client is supported as a 31-bit application on the 64-bit kernel.
- Linux on zSeries - Red Hat Enterprise Linux V4. CICS Universal Client is supported as a 32-bit application on the 64-bit kernel.
- Solaris V9. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels. Only SPARC hardware is supported.
- Solaris V10. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels. Only SPARC hardware is supported.
- HP-UX 11i v2. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels. Both PA-RISC and Itanium hardware is supported.
- HP-UX 11i v3. CICS Universal Client is supported as a 32-bit application on both 32-bit and 64-bit kernels. Both PA-RISC and Itanium hardware is supported.

The CICS Universal Client is a 32-bit application, and so can run on a 32-bit operating system, or on 64-bit operating system that can natively run 32-bit applications. Where available, both NPTL and LT threading models are supported unless otherwise stated; NPTL is preferred.

All UNIX and Linux operating systems require the Korn shell to be installed, so that scripts supplied with the product can run.

| **Note:** CICS Universal Client does not support Security-Enhanced Linux.

Supported CICS servers

The CICS Universal Client is supported by the following CICS servers:

- CICS Transaction Server for z/OS® V2.2:
  - Apply APAR PQ92943 and PK17427 for EXCI.
  - Apply APAR PK08496 for mixed-case password support.
  - If connecting over TCP/IPapply APAR PQ75803 and PQ82124.
  - Apply APAR PQ70543 for better recovery for EPI and cicsterm requests made over SNA, after a failure of CICS or the network.
- CICS Transaction Server for z/OS V2.3:
  - Apply APAR PQ92943 and PK17427 for EXCI.
  - Apply APAR PK08496 for mixed-case password support.
  - If connecting over TCP/IPto apply APAR PQ81772 and PQ82124.
- CICS Transaction Server for z/OS V3.1:
  - Apply APAR PK17426 for EXCI.
- CICS Transaction Server for z/OS V3.2:
  - Apply APAR PK51587 for z/OS CICS security violation.
- CICS/VSE 2.3:
  - Apply APAR PQ30169 for EPI support.
- CICS Transaction Server for VSE/ESA™ 1.1.1:
  - If connecting over TCP/IP apply APAR PQ80212 and APAR PQ52342.
  - Apply APAR PQ30170 for EPI support.
- TXSeries® V5.1 (Windows®, AIX, Solaris, HP-UX).
- TXSeries V6.0 and V6.1 (AIX and Linux on Intel PRPQ).
- CICS Transaction Server for Windows V5.0.
- CICS Transaction Server for iSeries V5.3 and V5.4..

Support for SNA communications

Customers wishing to use SNA communications must install one of the following products:

AIX
IBM Communications Server V6.3 for AIX APAR IY94042 is required

Linux on zSeries
IBM Communications Server for Linux V6.2.2. APAR IY94042 is supplied with V6.2.2.1

Linux on POWER
IBM Communications Server for Linux V6.2.2. APAR IY94042 is supplied with V6.2.2.1

Linux on Intel
IBM Communications Server for Linux V6.2.2. APAR IY94042 is supplied with V6.2.2.1

Solaris
SNAP-IX V7 for Solaris

SNA communications connections are not available when using other UNIX and Linux operating systems.

Support for TCP/IP communications

The operating system provides TCP/IP support.

Support for compilers and application development tools

CICS Universal Client supports the following compilers and application development tools:

AIX

- IBM COBOL complier for AIX Version 2
- Micro Focus Server Express V4.0, COBOL compiler for AIX
- XL C/C++ Enterprise Edition 7.0 for AIX
- XL C/C++ Enterprise Edition 8.0 for AIX
- XL C/C++ Enterprise Edition 9.0 for AIX

HP-UX

- ANSI C complier for HP
- aC++ complier for HP

Linux on Intel

- GNU C/C++ compiler (gcc) 3.3, 3.4, and 4.1

Linux on POWER

- GNU C/C++ compiler (gcc) 3.3, 3.4, and 4.1
- XL C/C++ Enterprise Edition V7.0, V8.0, and V9.0 for Linux

Linux on zSeries

- GNU C/C++ compiler (gcc) 3.3, 3.4 and 4.1

Solaris

- Sun ONE Studio 10 and 11

Support for other tools

CICS Universal Client supports the following tools:

- Adobe Acrobat Reader 6.0
- JAWS Screen Reader V6.0. (Version 6.0 is a minimum requirement)

GPL licence and copyright issues on Linux

This product uses the following libraries from the glibc package: libnsl.so, libm.so, libdl.so, ld.so, libc.so and libpthread.so. Refer to the glibc package on your machine for the various copyright statements and licensing terms for these libraries.

This product also uses libncurses.so from the ncurses package. Again, refer to this package on your machine for the copyright statement and licensing terms applicable to this library.

IMPORTANT: Your use of the libstdc++ or egcs-c++ packages is subject to the GNU GPL licence terms which could require you to provide source code in certain circumstances. Note that IBM will not supply source code, for example for the CICS Gateways C++ libraries.

CICS server PTF requirements

See the README file for the latest details on APARs and PTFs applicable to the CICS Universal Client.

Terminal sign-on capability

CICS servers require APAR fixes to support the terminal sign-on capability available in this release of CICS Universal Client; see “Supported software” on page 5 for details of the APAR relating to your CICS server. If the server does not have the required APAR applied and the ’-a’ option is not specified when the cicsterm command is issued, the installed terminal will give unpredictable results.

Timeout support

Any TXSeries or CICS Transaction Server on the Windows and UNIX operating systems must include the appropriate PTF level to provide complete support for timeouts. See “Supported software” on page 5 for details.

If the server does not have the required APAR applied and the ’-a’ option is not specified when the cicsterm command is issued, the CTIN transaction abends with code A42B. You see the following:

- The CICS Universal Client displays the message:
  ```
  CCL7053E Errors found while communicating with server
  ```
- This message is written to cicscli.log:
  ```
  CCL3105 Inbound CICS data stream error (CTIN,4, 0)
  ```
- On the server, this message is written to CSMT.out:
  ```
  ERZ042004E/0112: An invalid request was received from client
  ```
- console.msg includes this message:
  ```
  ERZ014016E/0036: Transaction CTIN Abend A42B
  ```

Terminal Sign-on capability

CICS servers require APAR fixes to support the terminal sign-on capability function available in this release; see “Supported software” on page 5 for details of the APAR relating to your CICS server. If the server does not have the required APAR applied and the ’-a’ option is not specified when the cicsterm command is issued, the installed terminal will give unpredictable results.

Timeout support

Any TXSeries or CICS Transaction Server on the Windows and UNIX operating systems must include the appropriate PTF level to provide complete support for timeouts. See “Supported software” on page 5 for details.

If the server does not have the required APAR applied and the ’-a’ option is not specified when the cicsterm command is issued, the CTIN transaction abends with code A42B. You see the following:

- The CICS Universal Client displays the message: `CCL7053E Errors found while communicating with server`
- This message is written to cicscli.log:

  `CCL3105 Inbound CICS data stream error (CTIN,4, 0)`
- On the server, the message: `ERZ042004E/0112: An invalid request was received from client` is written to CSMT.out
- console.msg includes this: `ERZ014016E/0036: Transaction CTIN Abend A42B`

Communication with CICS servers

You can use these protocols to communicate with CICS servers:

TCP/IP
TCP/IP is a widely used, robust, suite of protocols, particularly useful in connecting heterogeneous networks.

SNA The CICS Universal Client uses Advanced Program-to-Program Communication (APPC) to provide SNA LU 6.2 communication. APPC is an implementation of the SNA LU 6.2 protocol that provides device-independent application-to-application communication over LU 6.2 sessions.

The protocols that you can use for the various client/server connections are shown in Table 1. This table lists the protocols that you can use to connect your CICS Universal Client to your CICS server. It also shows whether EPI emulation and ECI are supported for your combination of CICS server and CICS Universal Client operating system. ● indicates that the protocol or feature is supported; ○ indicates that it is not supported.

Table 1. Protocols and functions supported. If a protocol or feature is supported by the CICS Universal Client only on certain operating systems, the operating systems are listed.

SERVER	TCP/IP	SNA	ECI	EPI See note	ESI
CICS Transaction Server for z/OS V3.2	● (ECI only)	AIX Linux Solaris ●	●	●	●
CICS Transaction Server for z/OS V2.2, V2.3 and V3.1	● (ECI only)	AIX Linux Solaris ●	●	●	●
CICS Transaction Server for VSE/ESA V1.1.1	● (ECI only)	AIX Linux Solaris ●	●	●	●
CICS/VSE® V2.3	○	AIX Linux Solaris ●	●	●	●

Table 1. Protocols and functions supported (continued). If a protocol or feature is supported by the CICS Universal Client only on certain operating systems, the operating systems are listed.

SERVER	TCP/IP	SNA	ECI	EPI See note	ESI
TXSeries V5.1, V6.0 and V6.1	● (ECI and EPI only)	○	●	●	○
CICS Transaction Server for Windows V5.0	●	○	●	●	○
CICS Transaction Server for iSeries V5.3	●	AIX Linux Solaris ●	●	●	●
CICS Transaction Server for i5/OS® V5.4	●	AIX Linux Solaris ●	●	●	●

Note: EPI also incorporates CICS 3270 terminal emulation and CICS 3270 client printer support.

Restrictions on CICS Transaction Server for iSeries

The following restrictions apply:

- DBCS languages are supported when communicating using ECI but not EPI.
- Sign-on capable terminals are not supported.
- You cannot start the CEDA transaction from a client terminal.
- You cannot use PF1 to obtain CICS online help from a client terminal.

Why use a particular protocol?

As shown in table Table 1 on page 10, some protocols can be used only for certain types of client/server connection. Access to 3270-based CICS transactions might require an SNA connection. SNA network connections are the recommended approach, because they can be routed over IP networks using the SNA Remote API Client or Enterprise Extender and they do not require complex AnyNet® definitions.

For information on continuing AnyNet support in z/OS Communications Server, refer to the information in the z/OS Statements of direction announcement, Software Announcement 203-266, dated October 7, 2003, and the z/OS V1.7 preview announcement, Software Announcement 205-034, dated February 15, 2005.

DBCS multibyte characters

Some characters in certain code pages are represented with 3 or more bytes. The CICS Universal Client does not support multibyte characters that are longer than 2 bytes. If you try to display such characters on a CICS terminal, you will get unpredictable results.

If you are running on a locale that is unique to AIX, you might experience problems when connecting to certain CICS servers. The table below lists the relevant client/server combinations.

CICS Client code page	CICS Server operating system	CICS Server code page
ja_JP (33722)	OS/2®	932
ja_JP (33722)	Windows NT®	932
ja_JP (33722)	AIX	932
ko_KR (970)	OS/2	949
ko_KR (970)	Windows	949
zh_TW (964)	OS/2	950
zh_TW (964)	AIX	950
zh_CN (1383)	OS/2	1381
zh_CN (1383)	Windows NT	1381

Server code page support

Some CICS servers do not support all the code pages that are supported by the operating system on which CICS Universal Client is running.

If the code page of an ECI application is different from the code page of the server, data conversion must be performed at the CICS server. This restriction applies for EBCDIC CICS servers such as CICS Transaction Server for z/OS. For more information, see “Data conversion,” on page 105, the CICS server documentation, and the Redbook *Java Connectors for CICS*.

Chapter 3. Installing CICS Universal Client

To install CICS Universal Client follow the steps in these topics:

Choosing what to install

You can choose either a complete or custom installation.

Complete installation

Installs all features:

- Program files
- Development kit, consisting of the following features:
  - Programming samples (installed to `<install_path>/samples/`)
  - Toolkit (includes Javadoc information)

into the default installation directory (see "Installation path" on page vii).

Custom installation

To perform a custom installation:

- Select the features to install

Preparing to install CICS Universal Client

Before you install CICS Universal Client, perform the following steps as appropriate:

1. Check that your operating system is supported; see "Supported software" on page 5 for supported operating systems.
2. Log on as root. Check the root user's umask to ensure that files created during the installation are readable by the user IDs that will run the CICS Transaction Gateway. A umask of 077 restricts access to the root user that installed the CICS Transaction Gateway; a umask of 022 allows all users to run the CICS Transaction Gateway.
3. **Linux operating systems:** Issue the following command from the command line:
   ```
   umask 0022
   ```
4. Close down any programs that are running, including the following:
   - Client daemon (issue the `cicscli -x` command)
   - Configuration Tool
5. The installation process can upgrade from the CICS Universal Client Version 5.1 or later. Remove any other versions of the CICS Universal Client or CICS Transaction Gateway before installing this product. Remove any beta versions of the product.

Known issues

- On the Linux on Intel platform, if a folder created during installation cannot be removed by the uninstallation program, a warning might be displayed. This warning will be issued if the folder contains a file that has been modified or is new, for example ctg.ini. This warning can be ignored.

Installing with the InstallShield wizard

Follow these steps to install CICS Universal Client with the InstallShield wizard:

1. Insert the CD into the drive. If installing from a network drive, connect to the drive.
2. Click **Next**, then type `d:\windows\installer.exe`, where d: is your CD or network drive. A message box informs you that a Java™ virtual machine is being installed. This is used only by the installation and uninstallation processes; refer to the license for restrictions on the use of this JRE. Be prepared for a pause before the next screen is displayed.
3. Issue a command like the following: `/<cdrom>/<platform>/installer`, where *<cdrom>* is the name of your mounted cd drive and *<platform>* is your operating system.
4. A welcome screen is displayed. Click **Next**.
5. If you are upgrading, you see the Upgrade warning screen. Click **Next** to continue.
6. The license agreement is displayed. Read the agreements, click the appropriate radio button, and then click **Next**. You must accept all license agreements that are shown to continue the installation. License files are stored in the <install_path>/license subdirectory. If you run a console-based installation in a DBCS locale, the License agreement is displayed in English and not the local language. Installation using a Graphical User Interface is not affected.
7. The README file is displayed, giving details of any late changes to the product. Read it and then click **Next**.
8. Choose the type of installation required. Select **Complete** or **Custom** as appropriate, and then click **Next**.

   If you chose a complete installation, go to step 10. If you chose a custom installation, continue at step 9.
9. A screen showing the features that can be installed is displayed. Select the check box for each feature that you want to install.
10. (Resume here if you chose a complete installation in step 8.) Information about the choices you have made is displayed on your screen. Read the information and then click **Next** to start installing. The installation process keeps you informed of progress.
11. Read the information about configuring the product and click **Finish** to end the installation process.

Installing from the console

To install from the command line, take the following steps:

1. Insert the CD into the drive. If installing from a network drive, connect to the drive.
2. Issue a command like the following:

   ```
   /<cdrom>/<platform>/installer -console
   ```

   where *<cdrom>* is the name of your mounted cd drive and *<platform>* is your operating system.

You cannot cancel a command-line installation after the installer has started to copy files.

Installing silently

You can silently install CICS Universal Client without going through any of the installer screens. The installation process takes its input from a response file; no user input is required. If you do not provide a response file, all default options are used.

Creating a response file

Use one of the following methods to create a response file:

- Use the supplied sample file installResponseSamp.txt. Edit this file as appropriate.
- Create a response file by installing the CICS Universal Client and recording the responses that you enter. Issue the following command:

  ```
  installer -options-record full_path_name/response_file_name
  ```

  This command stores your responses in file *`response_file_name`*. Note that there is no space between -options and -record.

Using the response file to install silently

To install silently, take the following steps:

1. Make the response file and the install image available to the computer on which the CICS Universal Client is to be installed.
2. Issue the following command:

   ```
   installer -options "full_path_name/response_file_name" -silent -V licenseAccepted=true
   ```

Installing silently with no response file

To install silently without using a response file, issue the following command:

```
installer -silent -V licenseAccepted=true
```

All options will take default values.

Actions after installation

After installing CICS Universal Client you may do the following:

Installing the Java runtime environment

You need Java to run the Configuration Tool.

Install the JRE from the following location:

```
/<cdrom>/JRE/<platform>
```

where `<cdrom>` is the name of the mounted CD drive that contains the CD, and `<platform>` is the name of your operating system.

Setting LD_LIBRARY_PATH for Linux C++ shared library support

Your default CICS Universal Client installation on all Linux systems requires the libstdc++ shared library /usr/lib/libstdc++.so.5. Alternative CICS Universal Client shared library files that are installed require the libstdc++ shared library /usr/lib/libstdc++.so.6. A default installation of Red Hat Enterprise Linux V4 or V5, or SUSE Linux Enterprise Server V10 will not install /usr/lib/libstdc++.so.5, so use the CICS Universal Client alternative shared library files on these Linux systems.

To use the alternative shared libraries, define **LD_LIBRARY_PATH**. For example, if the install_path is /opt/ibm/cicstg, issue the following command:

```
export LD_LIBRARY_PATH=/opt/ibm/cicstg/lib/stdc++6
```

Installing the Java runtime environment

You need Java to run the Configuration Tool.

Install the JRE from the following location:

```
/<cdrom>/JRE/<platform>
```

where `<cdrom>` is the name of the mounted CD drive that contains the CD, and `<platform>` is the name of your operating system.

Installing a shared library on Red Hat Enterprise Linux V3

Shared library /usr/lib/libstdc++-libc6.2-2.so.3 is not installed by default for Red Hat Enterprise Linux (RHEL) 3.0, but is required by the run-time environment. The Configuration Tool will not work correctly without it. The rpm that contains this library is compat-libstdc++-7.3-2.96.122.i386.rpm. To install the library, at a shell prompt type:

```
 rpm -ivh compat-libstdc++-7.3-2.96.122.i386.rpm
```

Adding features after installation

To add features after installation:

1. Follow the steps in “Installing with the InstallShield wizard” on page 14 as far as step 8 on page 14.
2. Select **Custom** as the type of installation and select the features that you want to install.
3. Complete the remaining steps in the installation process to add the new features.

Removing features

1. Run <install_path>/bin/ctguninst
2. Click **Next** on the Welcome screen.
3. The uninstaller screen is displayed, with all features selected. Leave selected the features that you want to remove; deselect any features that you want to keep. Click **Next**.
4. Read the summary information and then click **Next** to remove the features.
5. Click **Finish** when prompted.

Updating CICS Universal Client

After installing CICS Universal Client, you can update the product or obtain corrective service software (“fixes”).

For information about corrective service software, visit the Web site at `www.ibm.com/software/cics/cuc` and follow the **Support** link.

Uninstalling the CICS Universal Client

To uninstall the CICS Universal Client, run the `ctguninst` script as the root user.

Uninstalling from the console

Issue the following command:

```
ctguninst -console
```

Uninstalling silently

Issue the following command:

```
ctguninst [-options "response_file_name"] -silent
```

You do not have to provide a response file. If you do not, the whole product is uninstalled.

Uninstalling from the console

Issue the following command:

```
ctguninst -console
```

Uninstalling silently

Issue the following command:

```
ctguninst [-options "response_file_name"] -silent
```

You do not have to provide a response file. If you do not, the whole product is uninstalled.

Diagnosing problems with the installation and uninstallation logs

Errors and warnings generated during the installation or uninstallation of the CICS Universal Client are recorded in the appropriate log file. The log files, if created, are at the following locations:

Installation
/tmp/cicsucInstall.log

Uninstallation
/tmp/cicsucUninstall.log

If a log file already exists, any new messages are appended to the end of the file.

Use the log to diagnose any problems that might have caused an installation or uninstallation to fail, especially in silent mode which has no user interaction with the program.

National language support

After installation, you can change the language in which user messages are displayed by entering the `ctgmsgs` command. To do this:

1. Stop the CICS Universal Client.
2. Change the locale of the machine to the locale in which messages are to be displayed.
3. Run the `ctgmsgs` command:

   ```
   ctgmsgs XX <code set>
   ```

   where *XX* is the two character message language. To obtain a list of available languages, enter the command without parameters. Figure 1 on page 18 shows the output:

```
This utility is used to change the language of user messages.
-------------------------------------------------------------

    Language               Locale          Code Set
    ---------------------  --------------  ----------
en  US English             en_US           ISO-8859-1
                           EN_US           UTF-8
fr  French                 fr_FR           ISO-8859-1
                           FR_FR           UTF-8
de  German                 de_DE           ISO-8859-1
                           DE_DE           UTF-8
it  Italian                it_IT           ISO-8859-1
                           IT_IT           UTF-8
es  Spanish                es_ES           ISO-8859-1
                           ES_ES           UTF-8
tr  Turkish                tr_TR           ISO-8859-9
                           TR_TR           UTF-8
ja  Japanese               ja_JP           EUCJP
                           JA_JP           UTF-8
ko  Korean                 ko_KR           EUCKR
                           KO_KR           UTF-8
zh  Simplified Chinese     zh_CN           EUCCN
```

Figure 1. Output from the ctgmsgs command

This also shows the locale and code set associated with the language.

4. Restart the CICS Universal Client.

Using an alternative code set on AIX

On AIX issue the command smitty iconv to use an alternate code set.

This provides the Convert Flat File screen, on which you can enter parameters:

```
                             Convert Flat File

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* CURRENT FILE / DIRECTORY name                      []
* CURRENT CODE set                                   []                       +
* NEW FILE / DIRECTORY name                          []
* NEW CODE set                                       []                       +















F1=Help             F2=Refresh          F3=Cancel           F4=List
Esc+5=Reset         F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Fill in this screen as follows:

CURRENT FILE / DIRECTORY name
Enter <install_path>/bin/cclmsg.txt

CURRENT CODE set
Enter the code set for the language you selected with ctgmsgs.

NEW FILE / DIRECTORY name
Enter the name of the new file.

NEW CODE set
Enter the alternative code set. You can view a list of the alternative code sets using the smitty lang command, which is used to set the language environment. For example, the code set IBM-850 is an alternative for the US English code set ISO8859-1.

When you have done the conversion, overwrite the cclmsg.txt file with the new file.

Using an alternative code set on other operating systems

To use an alternate code set, use the iconv routine for the flat file <install_path>/bin/cclmsg.txt. For example, to convert <install_path>/bin/cclmsg.txt from code set ISO8859-1 to code set ISO-850 enter:

```
iconv -f ISO8859-1 -t ISO-850 <install_path>/bin/cclmsg.txt > cclmsg.new
```

When you have done this conversion, you can overwrite the cclmsg.txt file with the new file:

```
mv cclmsg.new <install_path>/bin/cclmsg.txt
```

Using X-Window System from a remote system

When using X-Window System from a remote system, for example, to access the Configuration Tool, you must set up the DISPLAY environment variable to allow the application to display its windows on that system.

On the display system (that is the one that will display the windows), enter the command:

```
xhost +appl
```

where *appl* is the network name of the system being used to run the application.

On the application system, before you run the application, enter the command:

```
export DISPLAY=disp:0.0
```

where *disp* is the host name or IP address of the system where the windows will be displayed (followed by a colon and the display id—normally 0.0). The application windows are then displayed on the *disp* system.

Chapter 4. Configuring CICS server connections

After you have installed the CICS Universal Client and set up your CICS servers for communication, your next step is to set up the communication links between the CICS Universal Client and your CICS servers.

After you have set up your communications links, continue at Chapter 5, "Configuring CICS Universal Client," on page 31, which tells you how to make the required entries in the configuration file.

Related information

"Sample configuration documents" on page 115

Configuring TCP/IP

The TCP/IP stack on your local machine should already be correctly configured. Contact your system administrator if you encounter problems.

Verifying the TCP/IP installation

To verify that the CICS Universal Client can communicate with CICS servers, use the TCP/IP PING command to check the route to the CICS server:

```
ping [machine address | name]
```

To start PING, enter a command like the following:

```
ping 192.113.36.200
```

where 192.113.36.200 is the IP address of your CICS server. If you are using a *Domain Name Server (DNS)*, you can specify the symbolic host name rather than the IP address of the server.

If the statistics message shows a value other than 0% packet loss, it is possible that TCP/IP is not correctly configured:

- Check for TCP/IP definition errors.
- Check the physical connection to the network.

The PING command differs slightly, depending on your operating system. For more information, refer to the documentation supplied with your operating system.

Refer to "TCP/IP settings" on page 43 for information on TCP/IP settings.

Configuring TCP/IP on CICS Transaction Server for z/OS

CICS Universal Client can send ECI requests over TCP/IP to CICS Transaction Server for z/OS V2.2 and later. To perform this configuration:

1. Set the SIT parameter TCPIP=YES.
2. Install the following:
   - CICS-supplied transient data queue CIEO, in group DFHDCTG
   - Transaction CIEP in group DFHIPECI
   - Program DFHIEP in group DFHIPECI

3. Define the TCP/IP address and host name for the z/OS system. By default they are defined in the PROFILE.TCPIP and TCPIP.DATA data sets.
4. Add a TCP/IP listener to CICS. Use the following CEDA command to define a TCPIPSERVICE in a group:

   ```
   CEDA DEF TCPIPSERVICE(service-name) GROUP(group-name)
   ```

   Ensure that the group in which you define the service is in the startup GRPLIST, so that the listener starts when CICS is started. Key fields are explained as follows:

   POrtnumber
   The port on which the TCP/IP service listens.

   PRotocol
   The protocol of the service is ECI.

   TRansaction
   The transaction that CICS runs to handle incoming ECI requests. Set it to CIEP.

   Backlog
   The number of TCP/IP requests that are queued before TCP/IP starts to request incoming requests.

   Ipaddress
   The IP address (in dotted decimal form) on which the TCPIPSERVICE listens. For configurations with more than one IP stack, specify ANY to make the TCPIPSERVICE listen on all addresses.

   SOcketclose
   Whether CICS should wait before closing the socket after issuing a receive for incoming data on that socket. NO is recommended for ECI connections, to ensure that the connection from the Client daemon always remains open.

   ATtachsec
   Specifies the level of attach-time security required for TCP/IP connections.
5. Use the following command to install the TCPIPSERVICE definition:

   ```
   CEDA INS TCPIPSERVICE(service-name) GROUP(group-name)
   ```

Next steps

1. Use the Configuration Tool to create a server definition; see “The Configuration Tool interface” on page 32.
2. Configure the server definition; see “Configuring Server settings” on page 40.

Verifying the TCP/IP installation

To verify that the CICS Universal Client can communicate with CICS servers, use the TCP/IP PING command to check the route to the CICS server:

```
ping [machine address | name]
```

To start PING, enter a command like the following:

```
ping 192.113.36.200
```

where 192.113.36.200 is the IP address of your CICS server. If you are using a *Domain Name Server (DNS)*, you can specify the symbolic host name rather than the IP address of the server.

If the statistics message shows a value other than 0% packet loss, it is possible that TCP/IP is not correctly configured:

- Check for TCP/IP definition errors.
- Check the physical connection to the network.

The PING command differs slightly, depending on your operating system. For more information, refer to the documentation supplied with your operating system.

Refer to "TCP/IP settings" on page 43 for information on TCP/IP settings.

Configuring TCP/IP on CICS Transaction Server for z/OS

CICS Universal Client can send ECI requests over TCP/IP to CICS Transaction Server for z/OS V2.2 and later. To perform this configuration:

1. Set the SIT parameter TCPIP=YES.
2. Install the following:
   - CICS-supplied transient data queue CIEO, in group DFHDCTG
   - Transaction CIEP in group DFHIPECI
   - Program DFHIEP in group DFHIPECI
3. Define the TCP/IP address and host name for the z/OS system. By default they are defined in the PROFILE.TCPIP and TCPIP.DATA data sets.
4. Add a TCP/IP listener to CICS. Use the following CEDA command to define a TCPIPSERVICE in a group:

   ```
   CEDA DEF TCPIPSERVICE(service-name) GROUP(group-name)
   ```

   Ensure that the group in which you define the service is in the startup GRPLIST, so that the listener starts when CICS is started. Key fields are explained as follows:

   POrtnumber
   The port on which the TCP/IP service listens.

   PRotocol
   The protocol of the service is ECI.

   TRansaction
   The transaction that CICS runs to handle incoming ECI requests. Set it to CIEP.

   Backlog
   The number of TCP/IP requests that are queued before TCP/IP starts to request incoming requests.

   Ipaddress
   The IP address (in dotted decimal form) on which the TCPIPSERVICE listens. For configurations with more than one IP stack, specify ANY to make the TCPIPSERVICE listen on all addresses.

   SOcketclose
   Whether CICS should wait before closing the socket after issuing a receive for incoming data on that socket. NO is recommended for ECI connections, to ensure that the connection from the Client daemon always remains open.

   ATtachsec
   Specifies the level of attach-time security required for TCP/IP connections.

5. Use the following command to install the TCPIPSERVICE definition:

```
CEDA INS TCPIPSERVICE(service-name) GROUP(group-name)
```

Next steps

1. Use the Configuration Tool to create a server definition; see "The Configuration Tool interface" on page 32.
2. Configure the server definition (see "Configuring Server settings" on page 40.)

Configuring SNA

You need to install and configure an SNA communications server, such as IBM Communications Server. To set up communication over SNA define the following in your SNA communications product.

- The **local node characteristics** that are common to all SNA users at the workstation.
- A **local logical unit (LU)** for the CICS Universal Client.
- A **partner logical unit** for each CICS server with which the CICS Universal Client will communicate.
- One or more **modes** to specify sets of session properties that are used in binding SNA sessions.
- A **transaction program (TP)** for the CRSR transaction. You need this if the following apply:
  - The CICS servers support terminal emulation, and
  - You require automatic transaction initiation (ATI) against the CICS Universal Client terminals.

The terms used to describe these definitions vary with the product used to provide support. The terms used above are the ones used by IBM Communications Server.

SNA links to the CICS Universal Client support data synchronization levels (sync levels) 0 and 1.

Overview of SNA configuration definitions

To configure an SNA connection, you need to make entries on VTAM®, CICS, Communications Server, and the CICS Universal Client. Table 2 shows the entries that you need to make.

Table 2. Matching definitions for SNA

VTAM	**CICS Transaction Server**	**IBM Communications Server**	**ctg.ini**	**Example**
NETID	—	First part of fully qualified LU name in Partner LU	—	ABC3XYZ4
PU	—	Control Point alias in Node Definition	—	IYAMR021
LU	Netname	LU Name/LU alias in independent LU Type 6.2	Local LU name	IYAMT210
XID	—	Last five digits of Node identifier in Node Definition	—	05d**316fc**

Table 2. Matching definitions for SNA (continued)

VTAM	CICS Transaction Server	IBM Communications Server	ctg.ini	Example
Token Ring destination address	—	Adjacent node MAC address in Link Station	—	400045121088
Ethernet port address	—	MAC address		020070000428
Enterprise Extender IP address	—	Communication end point		192.113.36.200
APPL	APPLID	Second part of fully qualified LU name in Partner LU	—	IYCQST34
LogMode	Modename	Name in Mode	Mode name	LU62PS
—	—	—	Partner LU name	IYCQST34

Note:

1. The NETID is the VTAM network name. It is defined for your VTAM network in the VTAM start options.
2. The PU is the name of the Communications Server physical unit (PU). It is named in the VTAM switched major node.
3. The LU is the independent LU6.2 used by the CICS Universal Client. It must also be defined to VTAM in a switched Major Node.
4. The XID (or node ID) is configured in the VTAM switched major node using IDBLK and IDNUM. It is used in the XID exchange to activate the link station.
5. For IBM Communications Server, Token Ring, Ethernet and Enterprise Extender are all valid communication link station types. Choose one or more as required. For more details about link station configuration refer to the Communications Server Task Guides.
6. The APPL is is the CICS APPLID and also the VTAM APPL. It is defined in the VTAM application major node used by the CICS region.
7. The LogMode is the mode group used to control LU6.2 session properties. It must be defined in a VTAM logon mode table, which must be named on the VTAM APPL definition.
8. The Host system control point name is the CP for the PU of the VTAM front end processor.

More information on connecting the CICS Transaction Gateway to CICS Transaction Server for z/OS is in *CICS Transaction Gateway V5 - The WebSphere Connector for CICS, SG24-6133*. The information in the book is relevant to connections from the CICS Universal Client over SNA.

Configuring IBM Communications Server for Linux

Communications Server for Linux requires a number of environment variables to be defined. Refer to the README file supplied with the server and ensure that all variables are correctly set, including *LD_PRELOAD*. Do not set the *LD_PRELOAD* variable globally.

To ensure that messages can be written to error logs, the user who starts the Client daemon process must be a member of the 'sna' group. Read the man page for `ld.so` for information about security issues associated with LD variables.

Using IBM Communications Server for Linux Remote API client 6.3

IBM Communications Server for Linux Remote API Client 6.3 and above does not require the *LD_PRELOAD* environment variable of the Linux streams (LiS) library.

Set *LD_PRELOAD* to an empty string; for example:

```
EXPORT LD_PRELOAD=""
```

Configuring IBM Communications Server for ATI

To enable ATI against Communications Server client terminals, define the transaction program CRSR on the Server. Follow the instructions in the Administration Guide. To define the transaction program using X-Window System:

1. Start the administration application, **xsnaadmin**.
2. Select **Services—>APPC—> Transaction Programs**.
3. Select **TP invocation**.
4. Click **Add** or, on Linux, **New**.
5. Enter `CRSR` as the Application TP.
6. Indicate Parameters are for invocation on any LU Queue incoming allocates and enter the path to the executable as: `/usr/bin/cclclnt`
7. Set **Arguments** to `CRSR`
8. Set **Userid** to `root` and **Group** to `system`.
9. Close the TP definition window.
10. In the **Local LU advanced parameters**, ensure that the Attach routing computer host name is entered.
11. On the remote API client, ensure that invoked_tps = YES is set in the sna_clnt.net configuration file.

Defining SNA connections on CICS Transaction Server for z/OS

To define SNA connections from CICS do the following:

1. Specify the SIT parameter ISC=YES.
2. Install CSD groups DFHCLNT and DFHISC.
3. Create and install CICS connection and sessions definitions, as described later in this information.

Defining the location of the remote system

Define the location of the remote CICS system and the parameters of the connection to it. Use CEDA to set the following:

ACcessmsmethod
Set to `Vtam`.

PRotocol
Set to `Appc`.

Singlesess
Set to `No`.

AUtoconnect
Specify whether CICS is to bind sessions (drive CNOS) when the connection is installed. Set this to `Yes`.

ATtachsec
This defines the settings for SNA LU6.2 conversation-level security. Set it to one of the following:

- `Verify` to flow a user ID and password from the CICS Universal Client.
- `LOCAL` if you do not want to flow a user ID and password.

Netname
Set to the LU name of the partner LU6.2.

If you change and reinstall the CICS connection definition, you must stop and restart the connection.

Defining CICS sessions

For each CICS connection definition, define one or more session definitions to specify the SNA mode groups to be used within that connection.

Connection
Set to the name of the associated connection definition.

MOdename
Specify the mode group as defined in a VTAM LOGMODE. The modename must be unique among the sessions definitions that relate to one connection definition.

Protocol
Set to `APPC`.

MAximum
Specify the maximum number of sessions that are to be supported. The first value is the maximum number of sessions that can be supported. The second value specifies the number of contention-winner sessions (CICS-owned sessions). These values are negotiated during change number of sessions (CNOS) flows, when the sessions are actually bound; the negotiated values depend on the settings specified in the partner SNA node.

Set the first value to be at least as big as the MAXREQUESTS parameter in the ctg.ini file, to prevent a bottleneck to throughput. Set the second value to `001`, to ensure that START requests are shipped serially from the server to the client.

Autoconnect
Set to determine whether the sessions for this mode group will be bound when the connection is installed. Set it to one of the following:

- `Yes` to bind only contention-winner sessions
- `All` to bind all sessions

Configuring CICS connection autoinstall

Autoinstall of connections is particularly useful when dealing with many similar connections or when you are unsure of the LU names (netnames) to be used. To configure autoinstall of connection definitions:

1. Update the default autoinstall program from DFHZATDX to DFHZATDY, by specifying the SIT parameter AIEXIT=DFHZATDY. Alternatively, write your own autoinstall user-replaceable module based on the samples provided.
2. Configure model definitions. The supplied DFHZATDY autoinstall program uses the template CBPS. CBPS is supplied in DFHAI62 group; copy it to your own group and modify it accordingly. The parameters in the connection

definition template are the same as for a static definition (see "Defining the location of the remote system" on page 26), except that the netname is not needed.

The parameters for the sessions definition are the same as those listed in "Defining CICS sessions" on page 27, except that the Connection parameter must refer to the CBPS connection definition.

If you use the supplied connection autoinstall program (DFHZATDY), the connection name generated is based on the last four characters of the Netname. To change the connection name, create your own user-replaceable module from the sample provided in CICSTS22.CICS.SDFHSAMP.

Data conversion

The ECI and EPI allow non-CICS applications running in a client system to access CICS facilities and data managed by a CICS server system. Character data might need to be converted as it is passed between client and server; for example, data is encoded in ASCII on a CICS Universal Client system and in EBCDIC on a CICS mainframe server system. The server system performs data conversion. For more information, see "Data conversion," on page 105.

Configuring message queues

In UNIX and Linux systems, the Client daemon communicates internally using message queues. In systems other than AIX, the default configuration settings for these queues are too small to allow for large client data flows (such as 3270 maps or user COMMAREAs). Some symptoms of this problem are:

- An ECI program gives return code -3 (ECI_ERR_NO_CICS)
- A cicsterm locks when a large map is sent to it
- A large number of concurrent ECI requests significantly degrades performance

Change the configuration settings of the message queues to allow for large client data flows. The way that you do this depends on your operating system.

Message queues on HP-UX

The following settings are recommended:

```
msgssz  32     Message Segment Size
msgmnb  65535  Max Number of Bytes on Message Queue
msgmax  65535  Message Max Size (bytes)
msgseg  16384  Number of Segments Available for Messages
```

Set these values by using the **SAM** utility:

1. Type `sam` at the command prompt.
2. Select **Kernel Configuration**, **Configurable Parameters**.

   This displays a list of kernel parameters that you can change.
3. Select a parameter, either by clicking on it with the mouse or by moving the cursor to it and pressing the enter key.
4. Select **Actions**,**Modify configurable parameter**.
5. Enter the new value for the parameter in the **Formula/Value** field, and then select **OK**.

   If the value you entered is not valid, SAM displays a window explaining the error.

6. When you have made all of the required changes, select **Actions**, **Process New Kernel**.

   SAM displays a window asking for confirmation; select **Yes**.

SAM then compiles the kernel and displays a window asking if you want to replace the old kernel before restarting the system. You must restart the system for the changes to take effect.

Message queues on Linux

You are recommended to place the following settings in file /etc/sysctl.conf:

```
kernel.msgmni=128              #Max # of msg queue identifiers
kernel.msgmnb=163840           #Size of message queue
kernel.msgmax=40960            #Max size of a message
```

- On Red Hat, the new settings are used after you restart the computer.
- On SuSE, issue the command `chkconfig boot.sysctl on`, and then reboot.
- To check that the new settings have been applied, issue the command `sysctl -a`.

The `MSGMNI` variable determines the maximum number of message queue identifiers system wide. This is typically set to 128 which is sufficient for the normal number of concurrent requests expected to be processed.

Message queues on Solaris

The following settings are recommended:

```
set msgsys:msginfo_msgmax = 65535  Maximum size of System V message.
set msgsys:msginfo_msgmnb = 65535  Maximum number of bytes that can be on any
                                   one message queue.
set msgsys:msginfo_msgssz = 32     Specifies size of chunks system uses to
                                   manage space for message buffers.
                                   Obsolete since the Solaris 8 release.
set msgsys:msginfo_msgseg = 16384  Number of msginfo_msgssz segments the system
                                   uses as a pool for available message memory.
                                   Total memory available for messages is
                                   msginfo_msgseg * msginfo_msgssz.
                                   Obsolete since the Solaris 8 release.
set semsys:seminfo_semmni = 4096   Maximum number of semaphore identifiers.
set msgsys:msginfo_msgtql = 10000  The maximum number of queue entries that
                                   can be in the system at the same time.
                                   A low value can adversely affect
                                   system performance, or cause the
                                   client to freeze. IBM recommends that
                                   you set this value to the maximum (10000),
                                   or at least double the maximum number of
                                   concurrent requests. Stress load your
                                   system, and then use the ipcs -qa command
                                   to determine the setting.
```

Set these values by changing the entries in the `/etc/system` file. See Solaris system notes for information on changing this file.

Chapter 5. Configuring CICS Universal Client

This information describes how to configure your CICS Universal Client.

Related information

"Sample configuration documents" on page 115

"Redbooks" on page 115

Chapter 4, "Configuring CICS server connections," on page 21

Before you start to configure your system

This information covers what you must do before running the Configuration Tool.

In particular, make sure that communication links with your CICS servers are correctly set up; see Chapter 4, "Configuring CICS server connections," on page 21.

Gathering information

Have the following information available:

- The protocol you will use to connect to the CICS server (TCP/IP or SNA)
- The host name or IP address of the server
- The port number at the server to which the Client daemon should connect

See "Overview of SNA configuration definitions" on page 24, for information on matching definitions for SNA.

Using the Configuration Tool

Use the Configuration Tool to configure the CICS Universal Client.

To use the Configuration Tool remotely, export the display using the commands described in "Using X-Window System from a remote system" on page 19. To start the Configuration Tool, enter the **ctgcfg** command.

Storing configuration details

Configuration details are stored by default in the ctg.ini file in the <install_path>/bin subdirectory. You are recommended to use the Configuration Tool to update this file.

You can specify a different location for the configuration file, and optionally change its name, by setting the CICSCLI environment variable. If this variable is set, at startup the Configuration Tool loads the file referenced by CICSCLI. If the Configuration Tool fails to find the configuration file, it creates a file that includes a partial definition for a new server named `A_Server`. You must edit and save this file before use. The tool does not automatically create a directory if the directory referenced by CICSCLI does not exist.

The CICS Universal Client log indicates the name and location of the INI file. The CICS Universal Client will not run if a configuration file is not found.

Mixed case values

The configuration tool allows parameter values to be entered in mixed case and also writes the values as mixed case to the configuration file, `ctg.ini`. For both the Gateway daemon and the Client daemon, some values are folded to upper case at runtime.

Running the Configuration Tool for a different operating system

Proceed as follows:

1. Install the CICS Universal Client on the workstation on which you want to run the Configuration Tool.
2. To edit an existing configuration, copy ctg.ini to the <install_path>/bin subdirectory on the workstation, using FTP in ASCII mode.
3. Issue the following command to display help about the Configuration Tool:
   ```
   ctgcfg -?
   ```
4. Issue the following command:
   ```
   ctgcfg -PLAT OSCODE
   ```

   where *OSCODE* represents the operating system that the Configuration Tool should emulate, and is one of the values returned by the command that you issued at step 3.
5. Make the required entries and click **Save** to create or update the ctg.ini file.
6. Use FTP in ASCII mode to transfer the file to your system.

The Configuration Tool interface

The screen has four input areas:

1. Menu bar
2. Toolbar
3. Navigation panel
4. Settings panel

A fifth area, the status bar, shows which configuration file and operating system you are using. The status bar is display only.

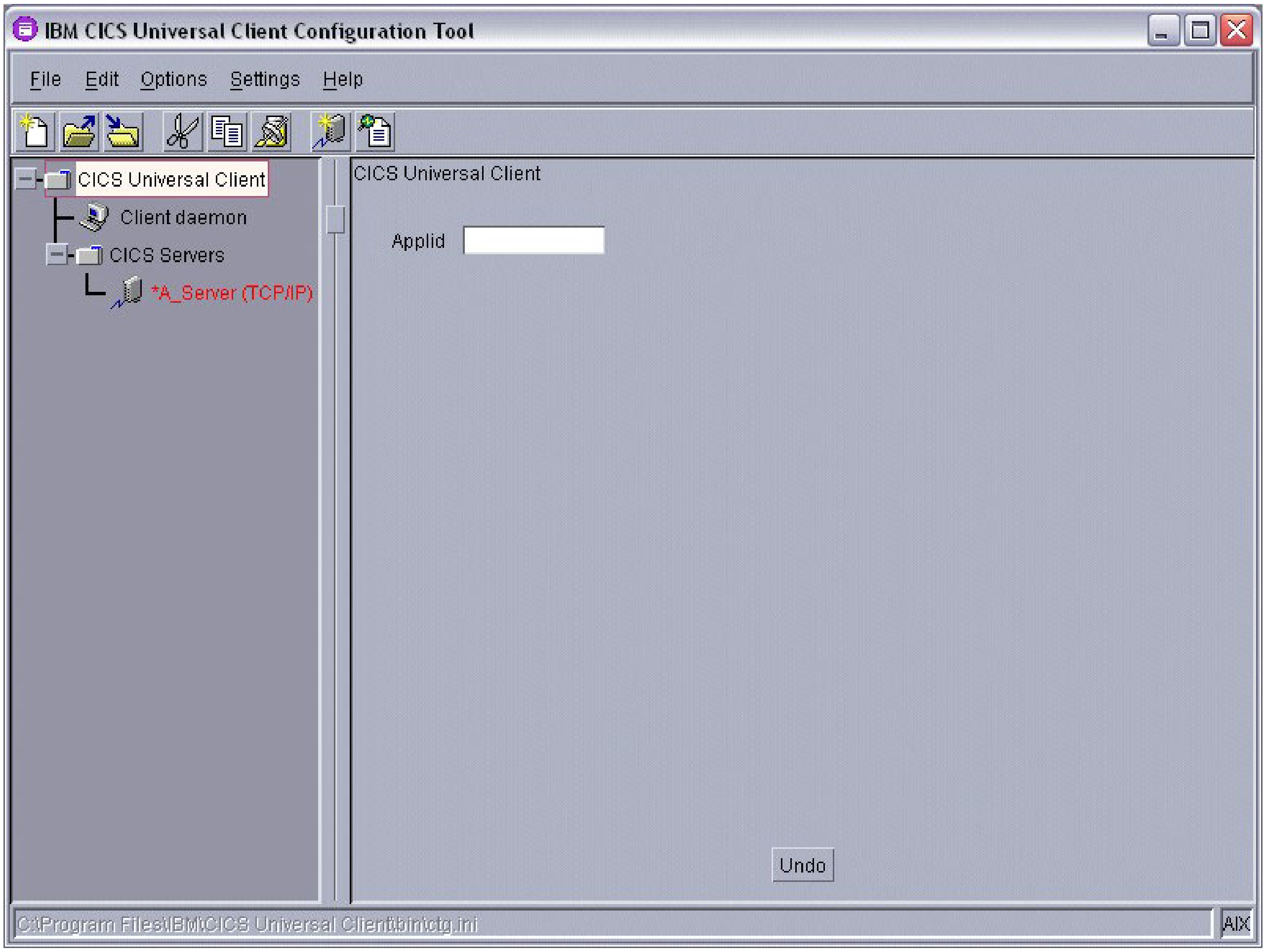


Figure 2. The Configuration Tool

Note: Screen captures later in this chapter show the Configuration Tool running for the Windows operating system. On your platform, the Configuration Tool might look slightly different.

Navigation panel

The navigation panel (see Figure 2) allows you to navigate through all the settings in your configuration. The tree has the following root node:

Client daemon
Contains the Client daemon.

CICS servers
Contains a list of CICS server names and the network protocol used to connect to that server.

Menu bar

The menu bar contains the **File, Edit, Options, Settings**, and **Help** menus.

The **File** menu has the following options:

New Creates a new configuration.

Open Opens an existing configuration.

Save Saves the current configuration. By default, the file name is ctg.ini and is saved to the <install_path>/bin subdirectory.

Save As
Allows you to override the default path and name of the configuration file.

Exit Exits the Configuration Tool. If you have made any alterations you are asked whether you want to save the configuration.

The **Edit** menu has **Cut**, **Copy** and **Paste** options, which perform the standard text functions.

The **Options** menu has the following options:

Trace Settings
Displays the Trace Settings dialog.

New Server
Defines a new server, named `A_Server`, that uses the TCP protocol.

Delete Server
Deletes a server node from the navigation panel.

The **Settings** menu has these options:

Color Changes the background color on panels and fields. If you change the color, all fields (except check boxes and radio buttons) take on the same background color as the main panel.

Font Changes the typeface, size, style, and color of text. You can also change the color of warning text.

Save on exit

If this menu option is selected, your choices are stored in CTGCFG.INI, for use in future sessions. A copy of CTGCFG.INI is created for each user in the user's home directory (~/.ctg).

Notes:

1. The file is not deleted if the CICS Universal Client is uninstalled. It remains on the system unless manually deleted and will be reused if the product is reinstalled.
2. To restore the default settings, close the Configuration Tool, delete file CTGCFG.INI, and then restart the Configuration Tool.
3. When you close the Configuration Tool, a message is displayed if file CTGCFG.INI cannot be written, even if you have deselected the **Save on exit** menu option. Click **OK** to clear the message and close the Configuration Tool.

The **Help** menu has the following options:

Context Help
Displays online help information according to the current screen context; for example, for a particular configuration setting.

Contents
Displays the contents list for online help.

Keys Help
Displays information on keyboard alternatives and other information to help users with a disability to use the software. See also "Accessibility features for CICS Universal Client" on page 117.

Index Displays a subject index for online help.

About Displays the version number and other information about the Configuration Tool.

Toolbar

The toolbar provides some of the function available from the menu bar in a set of icons. These icons have *hover help*, so that, when you place the cursor over them, a text box describing the option appears.

Settings panels

When you select nodes in the navigation panel, the relevant settings panel is displayed. The settings in these panels correspond to parameters in the ctg.ini file.

On each settings panel, an **Undo Changes** button allows you to undo changes you have made.

Configuring the product setting

The opening screen of the CICS Universal Client configuration tool displays the product settings panel for the for the CICS Universal Client.

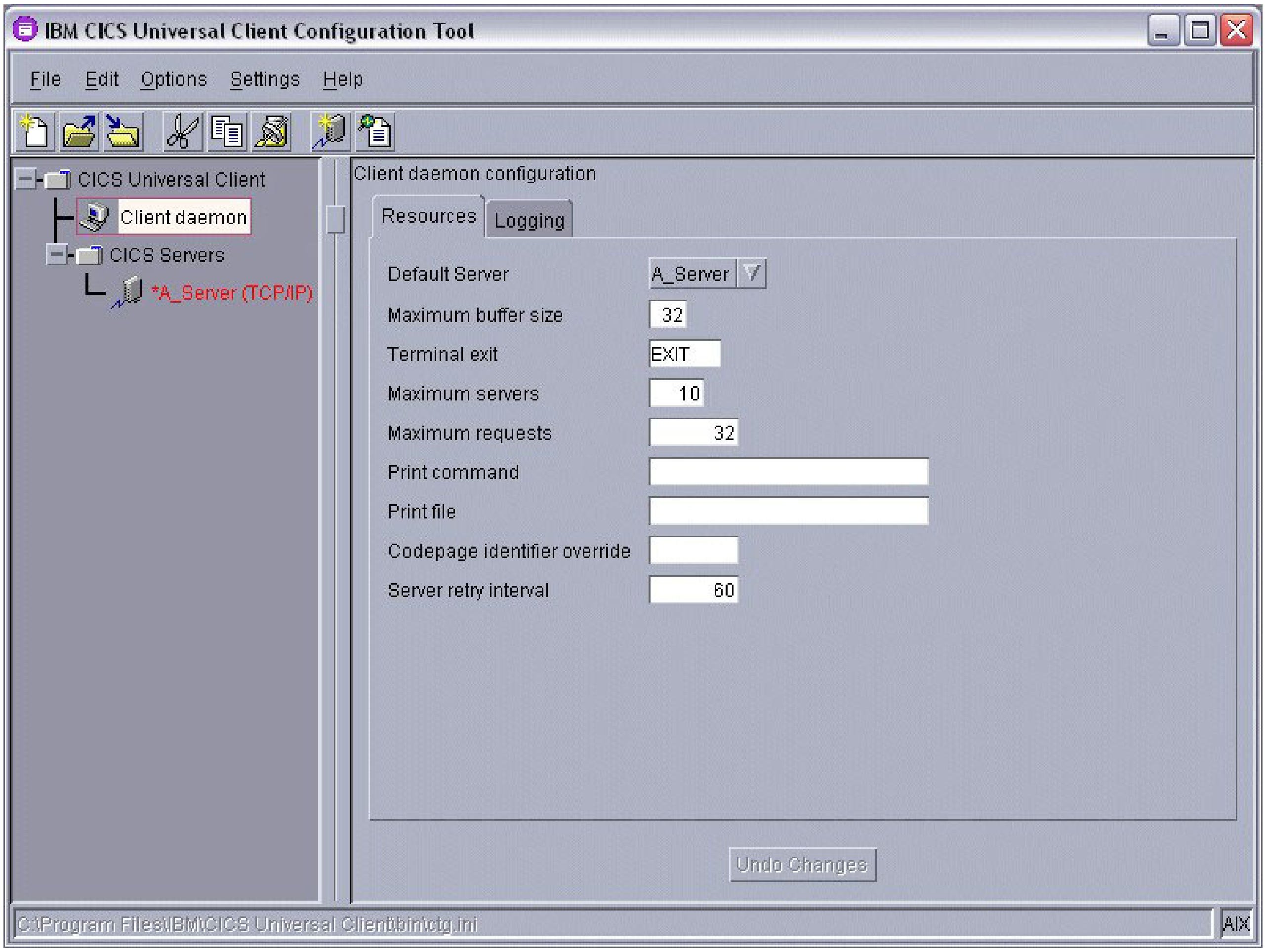
APPLID

The APPLID is used to identify the instance of the CICS Universal Client on server connections and tasks in a CICSplex.

Enter up to 8 characters, or leave this field blank. There is no restriction on the characters that can be used in APPLID, but to ensure that the APPLID is valid for use in all scenarios, it is advisable to restrict the characters used to be in the range A through Z, and 0 through 9. The character strings entered are converted to uppercase for SNA and TCP/IP connections.

The name must be unique within the CICS server system. The value of blank automatically generates a name that is guaranteed to be unique.

Configuring Client daemon settings


IBM CICS Universal Client Configuration Tool
File
Edit
Options
Settings
Help
CICS Universal Client
Client daemon
CICS Servers
*A_Server (TCP/IP)
Client daemon configuration
Resources
Logging
Undo Changes
C:\Program Files\IBM\CICS Universal Client\bin\ctg.ini
AIX

Default Server	A_Server
Maximum buffer size	32
Terminal exit	EXIT
Maximum servers	10
Maximum requests	32
Print command	
Print file	
Codepage identifier override	
Server retry interval	60

Figure 3. Client settings

Select the **Client** node to display the Client Configuration panel. The settings map to the parameters in the CLIENT section of the ctg.ini file. The panel has these tabs:

Resources
Logging

Default Server

Select the default server from the drop-down list.

Maximum buffer size

Enter a number of kilobytes, in the range 4 through 32. The default is 32 KB. If you enter a value outside the permitted range, the Configuration Tool warns you. If the value entered is too low, it substitutes the minimum. If the value entered is too high, it substitutes the maximum. If a non-numeric value is present in the configuration file, it substitutes the minimum value.

This value specifies the size of the transmission buffers in which application or terminal data will flow. The value should be large enough to cater for the largest possible COMMAREA or terminal input/output area (TIOA) to be used. The maximum COMMAREA size might be less than the Maximum buffer size, because certain protocols have an overhead of 512 bytes.

Leave this setting at the default unless the CICS Universal Client is running on a machine that is short of memory.

For details of the corresponding entry in the configuration file, see MAXBUFFERSIZE ("CLIENT section of the configuration file" on page 47).

Terminal exit

Enter a character string of between 1 and 4 characters. The default is EXIT.

The string, when entered at a terminal emulator at any time and place where a transaction name can be entered, causes the terminal emulator to terminate. The string must not contain any blank characters.

The string is case-sensitive. If a terminal emulator has uppercase translation in its CICS terminal definition, enter this string in uppercase.

For details of the corresponding entry in the configuration file, see TERMINALEXIT ("CLIENT section of the configuration file" on page 47).

Maximum servers

Enter a value in the range 1 through 256. The default is 10. If you enter a value outside the permitted range, the Configuration Tool warns you. If the value entered is too low, it substitutes the minimum. If the value entered is too high, it substitutes the maximum. If a non-numeric value is present in the configuration file, it substitutes the minimum value.

This value specifies the maximum number of servers that can be accessed concurrently from the client.

For details of the corresponding entry in the configuration file, see MAXSERVERS ("CLIENT section of the configuration file" on page 47).

Maximum requests

Enter a value in the range 1 through 32. The default is 32. If you enter a value outside the permitted range, the Configuration Tool warns you. If the value entered is too low, it substitutes the minimum. If the value entered is too high, it substitutes the maximum. If a non-numeric value is present in the configuration file, it substitutes the minimum value.

This value specifies the maximum number of concurrent items that might be executing on the Client daemon, where an item is one of:

- A request to install or uninstall a terminal emulator
- A request to install or uninstall an EPI terminal
- A transaction invoked by a terminal
- An ECI unit of work
- An ESI unit of work

It is used to detect runaway conditions where an application could, in error, submit an excessive number of requests to a server. The actual limit might be less than this setting if other operating system limits (for example, memory constraint or communication sessions), come into effect.

For details of the corresponding entry in the configuration file, see MAXREQUESTS ("CLIENT section of the configuration file" on page 47).

Print command

Enter a character string, from 1 to 256 characters long.

The specified string is a command specific to the operating system under which the CICS Universal Client is running. When a request to print is received, the Client daemon generates a temporary print file with a unique name.

The parameter string is appended with the temporary file name, and the resultant command executed. This allows, for example, print requests to be copied to a file, directed to a local printer, formatted for inclusion into documentation, and so on.

It is the responsibility of the Print command to delete the temporary print file after it has finished processing it.

Use a shell script with the Print command (for example, **lpr**) followed by the command to delete (**rm**).

See also the **Print file** description for more information.

For details of the corresponding entry in the configuration file, see PRINTCOMMAND ("CLIENT section of the configuration file" on page 47).

Print file

Enter a character string, 1 to 256 characters long.

This option applies only if you make no entry in the **Print command** setting.

The specified string identifies a file to which output from print requests received at the Client daemon is directed. Each print request is appended to the end of the current file.

This setting acts only as a default. The terminal and print emulators provide options to override this value. (See "cicsterm command reference" on page 73 and "cicsprnt command reference" on page 77).

For details of the corresponding entry in the configuration file, see PRINTFILE ("CLIENT section of the configuration file" on page 47).

Code page identifier override

Enter a value indicating a Coded Character Set Identifier (CCSID) to override your local code page identifier.

Use this setting if your platform has been updated for euro support, and the CICS Server has euro support. For example, for Latin-1 countries, use a CCSID value of 858 to indicate that the code page 850 includes euro support. For code page 1252, specify a CCSID value of 5348.

Note:

1. Regardless of the value in the **Code page identifier override** setting, `cicsterm` always displays characters based on the local code page of the workstation.
2. If you use the CCSID to change the code page identifier, data that is already stored on the server might be modified when retrieved by the CICS Universal Client, if it includes characters for which the code points produce different characters.

3. **AIX operating system:** On AIX 4.3.2 and above, support for Ja_JP locale uses CCSID=943. If the CICS Server does not support this CCSID, enter a code page identifier override of 932, or add the statement ’CCSID=932’ to the Client Section of the configuration file ctg.ini.

For details of the corresponding entry in the configuration file, see CCSID (“CLIENT section of the configuration file” on page 47).

Server retry interval

Enter a number between 1 and 3600, to specify the time in seconds between attempts by the Client daemon to reconnect to a server. The default is 60 seconds.

When the Client daemon becomes aware that a server to which it was connected is no longer active, it attempts to reconnect to the server one second after it becomes inactive. If it fails it keeps trying, at the interval specified here.

Set the interval to 0 (zero) to disable automatic re-connection attempts.

For details of the corresponding entry in the configuration file, see SRVRETRYINTERVAL (“CLIENT section of the configuration file” on page 47).

Log terminal installations and deletions

Select this check box to log server connection and disconnection events.

For details of the corresponding entry in the configuration file, see TERMINSTLOGGING (“CLIENT section of the configuration file” on page 47).

Error and warning log file

Enter the name of the log file to be used for problem diagnosis.

If not specified, the log file name defaults to cicscli.log in the /var/cicscli subdirectory.

For details of the corresponding entry in the configuration file, see LOGFILE (“CLIENT section of the configuration file” on page 47).

Use the same file for information messages

Clear this check box to send information messages to a separate log.

By default, information messages are logged to the “Error and warning log file”; clear the Use the same file for information messages check box to enable the “Information log file” field.

This field has no corresponding entry in the configuration file.

Information log file

Enter the name of the file to which information messages will be logged.

By default, information messages are logged to the “Error and warning log file”; clear the “Use the same file for information messages” field, and then complete this field, to change the destination. If you do not specify a path to the file, the log is created in the following directory:

```
/var/cicscli
```

For details of the corresponding entry in the configuration file, see LOGFILEINFO (“CLIENT section of the configuration file” on page 47).

Changing settings for Client daemon logging

This information describes how to specify a destination for error and other messages from the Client daemon.

By default error, warning, and informational messages are output to the “Error and warning log file” on page 39. Terminal installations and deinstallations are not logged by default. Take the steps below to change the settings.

1. Launch the Configuration Tool and navigate to the **Logging** tab of the **Client daemon** node.
2. Complete the fields on the tab as appropriate. You can choose to:
   - Log terminal installations and deinstallations.
   - Change the name of the Error and warning log file from the default cicscli.log.
   - Specify a different log for information messages.
3. Save the configuration file.

Related tasks

“Viewing the information log”
This information describes how to view messages that have been output to the information log.

Related information

“Using the Configuration Tool” on page 31
Use the Configuration Tool to configure the CICS Universal Client.

Viewing the information log

This information describes how to view messages that have been output to the information log.

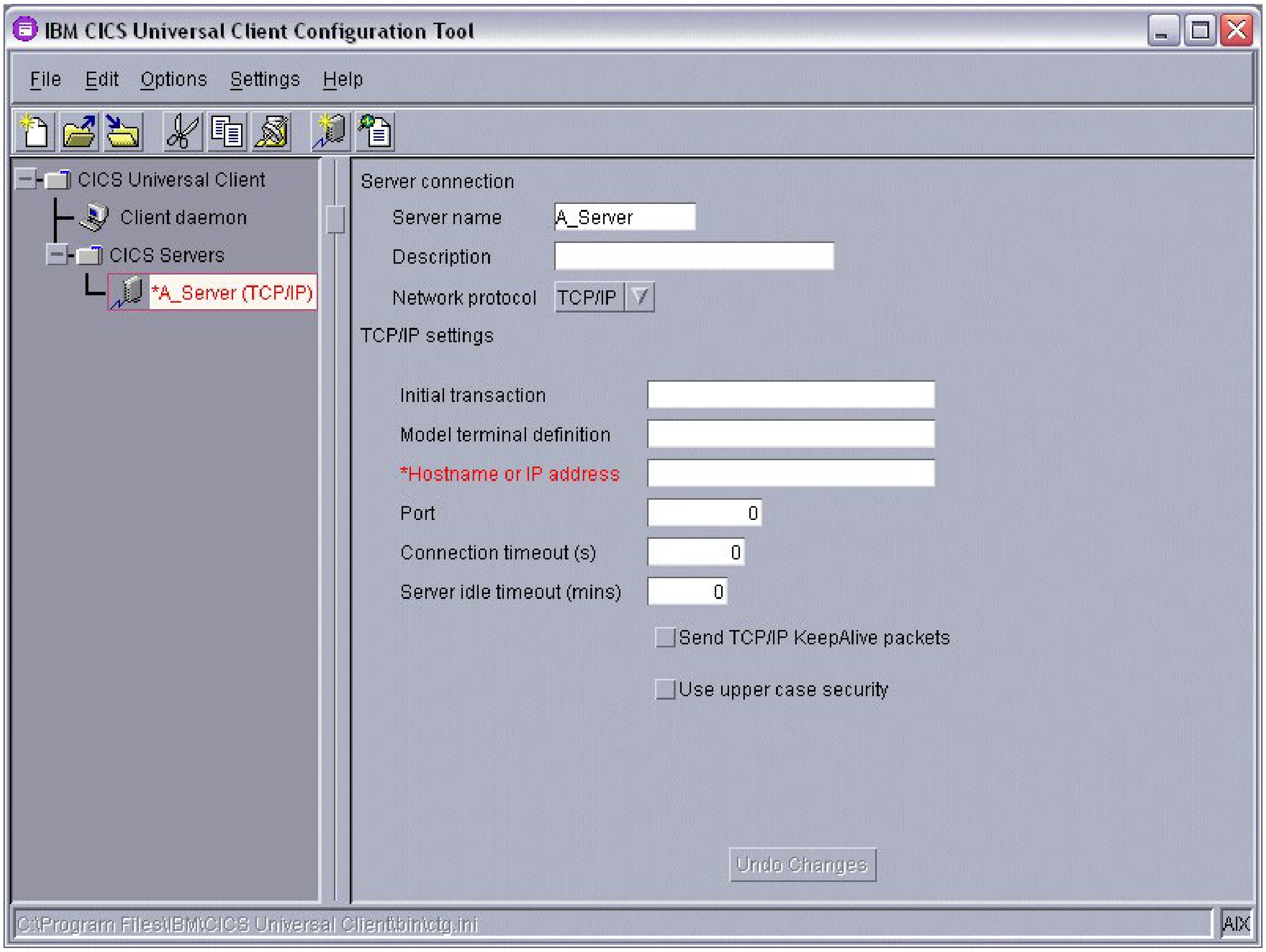
1. Determine the location of the information log file. By default, information messages are output to the “Error and warning log file” on page 39, but the destination can be changed through the “Information log file” on page 39 field in the Configuration Tool.
2. View the file using any standard text editor.

Related tasks

“Changing settings for Client daemon logging”
This information describes how to specify a destination for error and other messages from the Client daemon.

Configuring Server settings

To configure a new CICS Server, select **New Server** from the **Options** menu, or from the toolbar, or right click on the CICS Servers entry in the Navigation Panel.



Server connection

Server name: A_Server

Description:

Network protocol: TCP/IP

TCP/IP settings

Initial transaction:

Model terminal definition:

*Hostname or IP address:

Port: 0

Connection timeout (s): 0

Server idle timeout (mins): 0

☐ Send TCP/IP KeepAlive packets

☐ Use upper case security

Figure 4. Server settings

Select a server in the navigation panel to display the Server connection panel. The settings map to the parameters in a Server section of the configuration file.

Multiple identical server definitions are not permitted in the configuration file of the CICS Universal Client. The combination of fields that identify a server, and must therefore be unique is as follows:

Protocol	**Fields in ctg.ini**
TCP/IP	Hostname or IP address and Port.

This ensures that every connection that the CICS region and the network protocol see is represented by a unique server definition in the configuration file. If you have existing Client applications that use different server names to send requests to the same CICS region, you need to write a user exit to redirect requests. This is demonstrated in sample user exits `ecix2.c` and `epix2.c`; see `<install_path>/samples/samples.txt`.

Server name

Enter a name of between 1 and 8 characters. This provides a name that is independent of the communications protocol for the server, local to the Client daemon.

Any requests to access the server from ECI, EPI, ESI, or terminal emulators should use this name.

For details of the corresponding entry in the configuration file, see SECTION SERVER ("SERVER section of the configuration file" on page 48).

Description

Enter a description for the server of between 1 and 60 characters. This description is optional.

This description is returned on EPI or ECI list systems calls.

For details of the corresponding entry in the configuration file, see DESCRIPTION ("SERVER section of the configuration file" on page 48).

Initial transaction

Enter a transaction identifier of between 1 and 128 characters.

This string is case-sensitive and identifies the initial transaction (and any parameters) to be run when the terminal emulator connects to the server. If you do not enter anything, no initial transaction is run. The first four characters, or the characters before the first blank in the string are taken as the transaction. The remaining data is passed to the transaction on its invocation.

Ensure that the transaction does not require terminal input.

For details of the corresponding entry in the configuration file, see INITIALTRANSID ("SERVER section of the configuration file" on page 48).

Model terminal definition

Enter a string of between 1 and 16 characters.

The string is case-sensitive and specifies the name of a model terminal definition at the server, identifying the characteristics of terminals to be autoinstalled from the client. If the model cannot be located at the server, or you do not enter anything, a default terminal definition is used. This default is server-specific.

The interpretation of the Model terminal definition setting is server-specific. For example, for a TXSeries for AIX server, the value is 1 to 16 characters, and is the DevType for a CICS terminal definition entry to be used as the model.

For details of the corresponding entry in the configuration file, see MODELTERM ("SERVER section of the configuration file" on page 48).

Use uppercase security

Select this check box to specify that the CICS Universal Client converts to uppercase any user ID or password from an ECI application or from a user prompt. This setting is disabled by default.

If the UPPERCASESECURITY entry is missing from the configuration file, the default behaviour is for UPPERCASESECURITY to be enabled.

For details of the corresponding entry in the configuration file, see UPPERCASESECURITY ("SERVER section of the configuration file" on page 48).

Network protocol

Select the protocol to be used for the server connection from the list:

- TCP/IP
- SNA

The protocol settings on the panel change according to the protocol you select.

Server idle timeout (mins)

Enter an integer in the range 1 through 1080, to specify in minutes the period of inactivity after which the connection between the Client daemon and the CICS server is closed. A value of 0 means that no timeout is applied. The default value is 0.

The field is available if one of the following network protocols is selected:

- TCP/IP
- SNA

The **Server idle timeout (mins)** period is counted from when the number of outstanding conversations (units of work) on the connection is zero. The connection is automatically reestablished when a Client application sends the next ECI, EPI or ESI request.

When a server connection times out, the Client daemon behaves as if the `cicscli -x=<servername>` command had been issued. In particular, any value specified in the "Server retry interval" on page 39 configuration field is ignored, and no attempt is made to reestablish the connection. The Server retry interval setting is re-enabled if the connection is re-established.

For details of the corresponding entry in the configuration file, see SRVIDLETIMEOUT (ctg.ini: SERVER section `sectionserver). If the network protocol for the server in question is not supported for this field, the Client daemon ignores any entry in the configuration file for the **Server idle timeout (mins)** field.

Related information

"Shutting down the Client daemon normally" on page 62

TCP/IP settings

Hostname or IP address: Enter the character or numeric TCP/IP identifier for the host on which the CICS server is running. For example, cicssrv2.company.com (host name) or 192.113.36.200 (IP Address).

Host names are mapped to IP addresses either by the name server or in the `hosts` system file. It is better to use a host name in case the IP address changes.

The `hosts` system file is in the /etc directory.

For details of the corresponding entry in the configuration file, see NETNAME ("SERVER section of the configuration file" on page 48).

Port: Enter a numeric value in the range 0 through 65 535 defining the port number at the server to which the Client daemon should connect. The default value is 0. If you enter a value outside the permitted range, the Configuration Tool warns you. If the value entered is too low, it substitutes the minimum. If the value entered is too high, it substitutes the maximum. If a non-numeric value is present in the configuration file, it substitutes the minimum value.

A value of 0 indicates that the services system file should be used to locate the port number for the CICS service using the TCP protocol.

The services system file is in the /etc directory.

If no entry can be located in the services system file, a value of 1435 is assumed. This is the port assigned to the Client daemon in the TCP/IP architecture.

For details of the corresponding entry in the configuration file, see PORT ("SERVER section of the configuration file" on page 48).

Connection timeout: Enter a value in the range 0 through 3600, specifying the maximum time in seconds that establishing a connection is allowed to take; the default value of 0 means that no limit is set by the Client daemon. If you enter a value outside the permitted range, the Configuration Tool warns you. If the value entered is too low, it substitutes the minimum. If the value entered is too high, it substitutes the maximum. If a non-numeric value is present in the configuration file, it substitutes the minimum value.

A timeout occurs if connection establishment takes longer than the specified time. The TCP/IP socket is closed and the return code passed back to the client application is either ECI_ERR_NO_CICS or CICS_EPI_ERR_FAILED.

Cleanup processing happens after a timeout only if the server has support for this function installed.

For details of the corresponding entry in the configuration file, see CONNECTTIMEOUT ("SERVER section of the configuration file" on page 48).

Send TCP/IP KeepAlive packets: Select this check box if you want TCP/IP to periodically send keepalive messages to the server to check the connection. The CICS Universal Client uses the interval specified by your operating system.

For details of the corresponding entry in the configuration file, see TCPKEEPALIVE ("SERVER section of the configuration file" on page 48).

SNA settings

Note: Use the Configuration Tool to configure SNA on the AIX, Linux and Solaris operating systems.

Local LU name: Enter the **Local LU name** as it is known to the SNA Communications Server.

On **UNIX and Linux operating systems:** Enter an alias name of eight-characters or less to be used when connecting to the server.

For details of the corresponding entry in the configuration file, see LOCALLUNAME ("SERVER section of the configuration file" on page 48).

Partner LU name: Enter the LU name of the CICS server as it is defined for the SNA network.

- If you selected **Use Partner LU alias name**, enter an alias name of eight characters or less.
- If you did not select **Use Partner LU alias name**, enter a qualified name of up to 17 characters, for example, `ABC3XYZ4.PQRS1234`

Use Partner LU alias name: Select the **Use Partner LU alias name** box to specify the **Partner LU name** is an alias.

On **UNIX and Linux operating systems:** The default is for the check box to be selected.

Mode name: Enter between 1 and 8 characters specifying the mode name to be used when connecting to the server.

Enter `*` if you want the mode name to be filled with spaces.

Trace settings

To configure the trace settings, select the **Trace Settings** option from the **Tools** menu.

Trace Settings

Select check boxes to specify the components that will be traced when tracing is turned on.

Select all Client trace components
Select all Client trace components.

Client API level 1
The client API layer (level 1).

Client API level 2
The client API layer (level 1 and 2).

CICSCLI command line
The cicscli command interface.

CICSTERM and CICSPRINT
cicsterm and cicsprnt emulators.

CPP classes
The C++ class libraries.

Client daemon
The Client daemon.

Transport layer
Interprocess communication.

Protocol drivers level 1
Protocol drivers (for example, TCP). This traces data sent and received and provides supplementary information about failures.

Protocol drivers level 2
This traces internal flows through the protocol drivers and interactions with other software components. This enhanced tracking level currently has the same functionality as level 1.

If you enable tracing without specifying the components in either the cicscli command or in ctg.ini, a default set of components is traced:

- Protocol drivers

- The Client daemon
- Client API level 1

Selecting any of the check boxes in the Configuration Tool overrides the default set of components.

For details of the corresponding entry in the configuration file, see TRACE ("CLIENT section of the configuration file" on page 47).

Client trace file

Enter the path name of a trace file to which Client trace messages will be written, if tracing is enabled.

You do not have to enter an extension for the file name, because a file of type .BIN is always generated (or .WRP if the trace file wraps).

If no path is specified, the trace is written as follows:

```
/var/cicscli/cicscli.bin
```

To minimize any performance impact, the trace file is written out in binary format. To read it, convert the file to ASCII using the `cicsftrc` command.

For details of the corresponding entry in the configuration file, see TRACEFILE ("CLIENT section of the configuration file" on page 47).

Client trace file wrap size (KB)

Enter a value in the range 0 through 2 000 000.

If you use the memory mapped tracing option (cicscli -b), the size of the trace files is determined by this field, which specifies the total amount of space in KB reserved for trace data files. Subsequent trace entries will continue to be written from the beginning of the file. **Client trace file wrap size (KB)** must be greater than 0 if memory mapped tracing is to be used; the default value of 0 disables wrapping. If its value is between 1 and 99, a value of 100 is used instead to guarantee an adequate minimum trace size.

For details of the corresponding entry in the configuration file, see MAXWRAPSIZE ("CLIENT section of the configuration file" on page 47).

Applying your changes to the system

You must restart the Client daemon for any changes to the configuration file to take effect.

Sample configuration and mapping files

The following files are supplied in the <install_path>/bin subdirectory:

ctgsamp.ini
A sample configuration file. (The default configuration file name is ctg.ini.)

cicskey.ini
The keyboard mapping file.

cicskeysamp.ini
A sample keyboard mapping file.

cicscol.ini
The color mapping file.

cicscolsamp.ini
A sample color mapping file.

It is recommended that you create your own customized versions of these files with different names, because installing service updates might overwrite the files and cause any customization to be lost.

Reference your customized files through the following environment variables:

File	Environment variable
configuration file	CICSCLI
keyboard mapping file	CICSKEY
color mapping file	CICSCOL

Editing the configuration file

The default name for the configuration file is ctg.ini. This configuration file name can be changed to another name if required. A sample configuration file is supplied: `<install_path>/bin/ctgsamp.ini`. To create a new configuration, copy this file to `<install_path>/bin/ctg.ini` and edit the copy.

The configuration file is a text file consisting of a number of sections. Each section starts with `SECTION <NAME>` as the first entry on a line, and ends with `ENDSECTION`.

This information maps entries in the configuration file to the corresponding fields in the Configuration Tool. For an explanation of the fields see the Configuration chapter in the Administration book.

If you use the number sign (#) character to denote a comment, either:

- place it at the start of a line; or
- precede it by a space or tab character

because some valid names (for example modename) can start with the # character.

Some lines of the INI file are longer than 72 characters; take care when editing them.

CLIENT section of the configuration file

The CLIENT section of the configuration file defines the Client daemon settings.

There must be one CLIENT section of the configuration file. Entries correspond to fields in the Client Configuration and Trace settings panels of the Configuration Tool. The section name also stores the value entered in the Application ID field of the Configuration Tool.

Table 3. SECTION CLIENT

Entry in the INI file	**Configuration Tool field**
SECTION CLIENT	The Application ID entry is deprecated, it has been replaced by the “APPLID” on page 35 parameter which now takes precedence. Existing configuration files with the Application ID parameter set are still supported for compatibility with earlier versions. **Note:** SECTION CLIENT must be set to * when creating a new configuration file.
CCSID	“Code page identifier override” on page 38
LOGFILE	“Error and warning log file” on page 39
LOGFILEINFO	“Information log file” on page 39
MAXBUFFERSIZE	“Maximum buffer size” on page 36
MAXREQUESTS	“Maximum requests” on page 37
MAXSERVERS	“Maximum servers” on page 37
MAXWRAPSIZE	“Client trace file wrap size (KB)” on page 46
PRINTCOMMAND	“Print command” on page 38
PRINTFILE	“Print file” on page 38
SRVRETRYINTERVAL	“Server retry interval” on page 39
TERMINALEXIT	“Terminal exit” on page 37
TERMINSTLOGGING	“Log terminal installations and deletions” on page 39
TRACEFILE	“Client trace file” on page 46
TRACE	A comma-separated list of components to trace, as entered on the “Trace Settings” on page 45 panel. Possible entries are as follows:
ALL	Trace everything
API	Client API level 1
API.2	Client API level 2 (also traces level 1)
TRN	Transport layer
DRV	Protocol drivers level 1
DRV.2	Protocol drivers level 2 (also traces level 1)
CLI	CICSCLI command line
CCL	Client daemon
EMU	CICSTERM and CICSPRINT
CPP	CPP classes

SERVER section of the configuration file

The SERVER section of the configuration file defines a server to which the Client daemon may connect.

There may be more than one SERVER section. The first server listed is the default. The section name also stores the value entered in the **Server name** field of the Configuration Tool.

Table 4. SECTION SERVER

Entry in the INI file	**Configuration Tool field**
SECTION SERVER	“Server name” on page 41
DESCRIPTION	“Description” on page 42
INITIALTRANSID	“Initial transaction” on page 42
MODELTERM	“Model terminal definition” on page 42
UPPERCASESECURITY	“Use uppercase security” on page 42. Set to **Y** to enable, otherwise set to **N**.
PROTOCOL	Network protocol. Valid entries are `SNA` (AIX and Linux on zSeries operating systems only), `TCPIP`. The Network protocol must correspond to an entry in the DRIVER section; see “DRIVER section of the configuration file.”

Other entries in this section depend upon the protocol selected.

Table 5. SECTION SERVER: additional entries for the TCP protocol

Entry in the INI file	**Configuration Tool field**
NETNAME	“Hostname or IP address” on page 43
PORT	“Port” on page 43
CONNECTTIMEOUT	“Connection timeout” on page 44
SRVIDLETIMEOUT	“Server idle timeout (mins)” on page 43
TCPKEEPALIVE	“Send TCP/IP KeepAlive packets” on page 44. Set to **Y** to enable, otherwise set to **N**.

Table 6. SECTION SERVER: additional entries for the SNA protocol

Entry in the INI file	**Configuration Tool field**
LOCALLUNAME	“Local LU name” on page 44
NETNAME	“Partner LU name” on page 44
PARTNERLUALIAS	“Use Partner LU alias name” on page 45. Set to **Y** to enable, otherwise set to **N**. On Windows operating systems this option should only be set to Y when LOCALLUALIAS is also set to Y.
MODENAME	“Mode name” on page 45
SRVIDLETIMEOUT	“Server idle timeout (mins)” on page 43

DRIVER section of the configuration file

This DRIVER section of the configuration file defines a communications protocol library used to communicate with a server.

There should be one DRIVER section for each Network protocol used with your servers; see “SERVER section of the configuration file” on page 48. Entries take the following form:

```
SECTION DRIVER=<protocol>
    DRIVERNAME=<library>
ENDSECTION
```

Table 7. DRIVER section

Valid entries for *<protocol>*	**Corresponding** *<library>*
SNA	CCLIBMSN
TCPIP	CCLIBMIP

The DRIVER section has no corresponding section in the Configuration Tool. The Configuration Tool automatically selects the correct protocol drivers.

PRODUCT section of the configuration file

The PRODUCT section of the configuration file defines the APPLID for the Client daemon.

Table 8. SECTION PRODUCT

Entry in the INI file	**Configuration Tool field**
SECTION=Product	
APPLID	Enter an alphanumeric identifier, in the range A-Z and 0-9, up to 8 characters in length. This identifies the client connection to the Gateway daemon. The APPLID is not case sensitive.

The following template shows the INI file definition:

```
SECTION=Product
  APPLID = cccccccc

ENDSECTION
```

Customizing cicsterm

The following topics cover customizing cicsterm for keyboard mapping and screen colors.

Keyboard mapping for cicsterm

The keyboard mapping for terminal emulator operation is defined in a keyboard mapping file. A sample key mapping file is supplied with your CICS Universal Client; see “Sample configuration and mapping files” on page 46 for details. It is recommended that you create your own customized mapping file.

The keyboard mapping file can be specified by:

- The -k option of the cicsterm command, which identifies a keyboard mapping file with a particular terminal (see “cicsterm command reference” on page 73).
- The CICSKEY environment variable. For example:

```
export CICSKEY=/var/cicscli/mykeys.ini
```

If you do not specify otherwise, a file name of cicskey.ini in the <install_path>/bin subdirectory is assumed.

You can change the keyboard mapping file at any time, although changes do not take effect until the next time the terminal emulator is started.

Keyboard mapping file syntax

This section describes the syntax of the keyboard mapping file. A statement must be provided for each key that is needed, because there are no default assignments (except for the alphabetic and numeric keys). There is no case sensitivity. Each binding must be on a separate line, and of the following form:

```
BIND 3270function [modifier+] key [;comment|#comment]
```

For example, to map the 3270 function EraseEof to the Ctrl+Delete keys pressed together the binding is as follows:

```
bind     EraseEof             Ctrl+Delete ;erase to end of field
```

The keyboard mapping file

In the mapping file, *3270function* can be any one of the following:

```
backspace       pa1           pf1     pf13
backtab         pa2           pf2     pf14
clear           pa3           pf3     pf15
cursordown                    pf4     pf16
cursorleft      printscreen   pf5     pf17
cursorright     reset         pf6     pf18
cursorselect    tab           pf7     pf19
cursorup                      pf8     pf20
delete          ignore        pf9     pf21
enter                         pf10    pf22
eraseeof                      pf11    pf23
eraseinput                    pf12    pf24
home
insert
newline
```

The value of `ignore` is provided to permit unwanted control keys on the keyboard to be ignored. (Unexpected glyphs are not generated.)

The *Modifier* can be either:

```
Ctrl
Shift
```

The *Key* can be any one of the keys shown in Table 9, (but some combinations of modifier+key are not supported — see “Key combinations” on page 52):

Table 9. Keys that you can map

Group	Keys
Escape key	`Escape`
Function keys	`f1 f2 f3 f4 f5 f6 f7 f8 f9 f10 f11 f12`
Numeric keys	`0 1 2 3 4 5 6 7 8 9`

Alphabetic keys	`a b c d e f g h i j k l m`
	`n o p q r s t u v w x y z`

Tab key	`Tab`

Movement keys	`newline backspace`
	`insert home pageup`
	`delete end pagedown`
	`up left down right`

Table 9. Keys that you can map (continued)

Group	Keys

Keypad keys	`keypad/ keypad* keypad-`
	`keypad7 keypad8 keypad9`
	`keypad4 keypad5 keypad6 keypad+`
	`keypad1 keypad2 keypad3`
	`keypad0 keypad. keypadenter`

Note: For the Client daemon, the `Ctrl/Act`, `Print Screen`, `Scroll Lock` and `Pause` keys are not available. The 3270 function Clear is assigned to the `Esc` key by default.

Key combinations: The following combinations of modifier and key can be mapped:

No modifier
All keys available for mapping.

Ctrl modifier
Only function keys, movement keys, alphabetic keys, tab key, and keypad keys can be mapped.

Shift modifier
Only function keys, numeric keys, tab key, and alphabetic keys can be mapped.

Customizing the screen colors for cicsterm

Screen colors and attributes are defined in a color mapping file. A sample is provided for you to tailor; see "Sample configuration and mapping files" on page 46 for details. It is recommended that you create your own customized mapping file.

The color mapping file can be identified by either of these methods:

- The -c option of the cicsterm command, which identifies a color mapping file with a particular server.
- The CICSCOL environment variable: For example:

```
export CICSCOL=/var/cicscli/mycols.ini
```

If you do not specify otherwise, a file name of cicscol.ini in the <install_path>/bin subdirectory is assumed.

A color mapping file provides alternative representations in hardware environments where it is not possible exactly to replicate 3270 screen attributes, for example, blinking or underscore. The color mapping file therefore defines how 3270 screen attributes are emulated on the client hardware.

If the color mapping file specifies a mapping for an attribute, this mapping is used even if the hardware upon which the CICS Universal Client is running actually supports the screen attribute.

If an application requests a 3270 field to be displayed with, for example, underscore, and no emulation setting has been specified, and the hardware cannot display underscore, the field is displayed without any highlighting at all.

The color mapping file is optional. However, for most hardware environments a mapping file is required if blinking or underscore support is required by the emulator.

You can change the color mapping file at any time, although changes do not take effect until the next time the terminal emulator is started.

Color mapping syntax

The syntax of the color mapping file is as follows. Each binding must be on a separate line, in this form:

```
BIND 3270attrib fg_color[/bg_color] [;comment|#comment]
```

Specifying a space between fg_color and /bg_color causes bg_color to be ignored.

The file is not case-sensitive.

The color mapping file

In the color mapping file, *3270attrib* can be any one of the following:

```
normal_protected    intensified_protected
normal_unprotected  intensified_unprotected

default    blinking_default    underscored_default
blue       blinking_blue       underscored_blue
green      blinking_green      underscored_green
cyan       blinking_cyan       underscored_cyan
red        blinking_red        underscored_red
magenta    blinking_magenta    underscored_magenta
white      blinking_white      underscored_white
yellow     blinking_yellow     underscored_yellow

default_highlight

operator_information_area
```

Each of *fg_color* and *bg_color* (foreground color and background color) can be any one of the following:

```
black       light_gray
blue        light_blue
brown       yellow
cyan        light_cyan
green       light_green
magenta     light_magenta
red         light_red
gray        white
```

If *bg_color* is omitted, a default value of black is taken.

Checking your configuration

After you have configured your system, check that it is configured correctly.

1. Start the CICS Universal Client.
2. Run one of the sample programs supplied. See file samples.txt, in the <install_path>/bin subdirectory, for details, including compilation instructions and information on compiler considerations.

   Related information

   "Starting the Client daemon" on page 61

Chapter 6. Client security

This information describes how to provide a user ID and password when connecting to a CICS server.

Client security overview

CICS servers might require the Client daemon to supply a user ID and password before they permit the following:
- A client connection
- Terminals to be installed
- Transactions to be run

This depends on the server and protocol security settings. The user ID and password are sent to the server of the transaction attach request for each conversation. A user ID and password are also required when a sign-on transaction is invoked on a sign-on capable terminal. In this instance, the user ID and password are flowed to the server as part of the 3270 data stream.

User IDs and passwords must not contain DBCS characters.

If no user ID is passed by a CICS Universal Client user application, and no default is set by the CICS Universal Client, the transaction is run using the mainframe CICS server's default user ID and password if the **Usedfltuser** parameter on the CICS server connection definition is set to Yes. If this parameter is set to No, security is enforced by the host CICS server and a user ID and password will need to be supplied. In each case, transactions execute in the server with the authorities assigned to the user ID authenticated.

Because the Client daemon has no security manager, it does not support user ID authentication. Configure your CICS server client connections so that incoming attach requests must specify a user ID and password. For mainframe servers, specify **AttachSec = Verify** in the CICS connection definition. **AttachSec = Identify**, which indicates that a user ID, but not password, is required, is **not** supported for client connections.

Default connection settings

The Client daemon maintains a default user ID and password for each server connection, which can be set by any of the following methods:
- CICSCLI security commands:
  ```
  cicscli -c=servername -u=userid -p=password
  ```
  The servername parameter can also be specified with the -s option.
- From C use the ESI function **CICS_SetDefaultSecurity**.
- From C++, use the **makeSecurityDefault** method of the **CclConn** or **CclTerminal** class.

These default values are used when required on all subsequent transaction requests for that server, provided that no values have been passed on the ECI request itself, or have been set for the specific EPI terminal against which the transaction will run.

Note: If the Client daemon is running in an environment where it survives user logoff, the default user ID and password values entered by the current user are retained even when that user logs off, and are subsequently reused as required.

ECI security

An application can provide a user ID and password on an ECI request; these values override any default values set for the server connection.

C programs
Set eci_userid and eci_password or eci_userid2 and eci_password2 in the ECI parameter block.

C++ programs
Set the userid and password parameters when constructing a server connection object - **CclConn**.

COM programs
Set the userid and password via the Details method on the Connect object.

EPI terminal security

The Client daemon also maintains a user ID and password for each installed terminal. These values override any default values set for the server connection. .

Terminal security is normally required only if using sign-on incapable terminals.

Changing the user ID and password

The user ID and password may be changed at any time, as follows:

- C programs:

  Set UserId and Password in the **CICS_EpiAttributes_t** structure on a **CICS_EpiAddExTerminal** call. Or, use the EPI function **CICS_EpiSetSecurity**. This call would typically be used to change the terminal security settings if, for example, the user's password had expired.
- C++ programs:

  Set the userid and password parameters when constructing a **CclTerminal** class object. Or, use the **alterSecurity** method of the **CclTerminal** class.

Password expiry management

For CICS clients, the management of expired passwords can be handled by the ESI functions **CICS_ChangePassword** and **CICS_VerifyPassword**.

The ESI functions can be used only with CICS servers that support password expiry management (PEM). See "Supported software" on page 5 for information on supported servers. Refer to the documentation for your CICS server for information on PEM support.

To use PEM, the Client daemon must be connected to the CICS server over SNA. An External Security Manager such as Resource Access Control Facility (RACF®), must also be available to the CICS server. ESI calls can be included within your ECI or EPI application. Only CICS servers returned by the **CICS_EciListSystems** and **CICS_EpiListSystems** functions are valid.

Sign-on capable and sign-on incapable terminals

Sign-on capable terminals allow sign-on transactions, either CICS-supplied (CESN) or user-written, to be run, whereas sign-on incapable terminals do not allow these transactions to be run.

If a terminal resource is installed as **sign-on capable**, the application or user is responsible for starting a sign-on transaction; the user ID and password, once entered, are embedded in the 3270 data. If the terminal resource is installed as **sign-on incapable**, the user ID and password are authenticated for each transaction started for the terminal resource.

Specifying the sign-on capability of a terminal

Prior to Version 3.1.0 of Clients and Gateways, whether a terminal was sign-on capable or not depended upon the server implementation. Client terminals installed on distributed CICS servers were sign-on capable; terminals installed on mainframe CICS and CICS Transaction Server for iSeries servers were sign-on incapable.

In Version 3.1.0, extended EPI function was introduced to allow the sign-on capability of a terminal to be specified by one of the following methods.

- C programs:

  Use **CICS_EpiAddExTerminal** and set the sign-on capability parameter in the **CICS_EpiAttributes_t** structure.
- C++ programs:

  Set the sign-on capability parameter when constructing a **CclTerminal** class object.

The sign-on capability of the installed terminal is returned in the terminal attributes. This will be set to SIGNON_UNKNOWN if the server does not return a sign-on capability parameter in the terminal install (CTIN) response.

To use any of the extended EPI functionality, ensure that you have applied the relevant server APAR; see “CICS server PTF requirements” on page 9.

Mainframe CICS servers

These servers support both sign-on capable and incapable terminals, provided that they are at the prerequisite maintenance level. A terminal install request that does not specify any sign-on capability, for example from **CICS_EpiAddTerminal**, results in a sign-on incapable terminal being installed.

For sign-on capable terminals:

- Use the **CICS_EpiAddExTerminal** call specifying a **SignonCapability** of CICS_EPI_SIGNON_CAPABLE.
- You do not need to set the userid and password fields on the **CICS_EpiAddExTerminal** call or use **CICS_EpiSetSecurity**, provided that you specify **UseDfltUser = Yes** in the CICS connection definition on the server.
- A userid and password entered through a sign-on transaction are flowed to the server as part of the 3270 data stream and they will therefore appear in a client trace.

  Specify **UseDfltUser = Yes** in the CICS CONNECTION definition, or ensure that the system administrator sets a default connection userid and password for the client. Otherwise, the add terminal request might fail with an

EPI_ERR_SECURITY return code. The default user ID must have sufficient privileges to allow the CTIN transaction to run.
- Before the user has signed on, transactions run under the default userid for the CICS server. After sign-on, transactions run under the signed-on userid.

For sign-on incapable terminals without terminal security:
- Use the **CICS_EpiAddTerminal** call
- A connection userid and password are required regardless of the setting of the **UseDfltUser** in the CICS connection definition on the server.
- Transactions run under the userid specified in the corresponding FMH attach request.

For sign-on incapable terminals with terminal security:
- Use the **EpiAddExTerminal** call specifying a **SignonCapability** of CICS_EPI_SIGNON_INCAPABLE.
- Set the userid and password fields on the **CICS_EpiAddExTerminal** call.
- Specify **UseDfltUser = No** in the CICS connection definition on the server to enforce security.
- Use **CICS_EpiSetSecurity** in conjunction with **CICS_VerifyPassword** and **CICS_ChangePassword** to change the security settings for an existing terminal.
- The userid and password are flowed to the server in the FMH of the attach request and will not appear in a client trace.
- Transactions will run under the userid specified in the corresponding FMH attach request.

To use one of the APIs that does not support the extended EPI functionality, use CRTE through a middle tier system to get sign-on capable terminal-like functionality.

CICS Transaction Server for iSeries servers

These servers do not support the sign-on capability parameter in a terminal install (CTIN) request. A terminal install request always results in a sign-on incapable terminal being installed.

Chapter 7. Performance

This information helps you to:

- Understand the factors that affect CICS Universal Client performance
- Achieve the best performance from your system

System considerations

CICS server transactions

The design of your CICS server transactions affects the performance of your system. The response time through the CICS Universal Client might increase if your transactions:

- Need to wait for shared resources, for example data sets or applications, to become available
- Make remote links to other CICS systems
- Are unnecessarily complex

Refer to the performance and tuning documentation for your CICS server system for information on how to get the best performance.

Extended logical units of work

Take care when extending a logical unit of work across multiple program link calls that might span a long time period (for example, user thinking time). The logical unit of work holds various locks, and other CICS resources, on the server. This might cause delays to other users who are waiting for the same locks and resources.

Also, each logical unit of work occupies one CICS non-facility task for the duration of its execution. This means that you must define enough free tasks in the CICS server to service the maximum expected number of concurrent calls.

Refer to *CICS Transaction Gateway: Programming Guide* for more information.

Tracing and performance

Tracing the Client reduces performance significantly. Where possible, try to measure response times, without using tracing, through the different parts of your system, to find where delays are happening. For example, you could measure response times at the user application, or through the facilities provided by CICS and WebSphere® Application Server.

If you need to trace the Client, take the following steps to lessen the impact on performance:

- Use memory mapped trace whenever possible; see “Memory mapped tracing” on page 84. This should be suitable in most cases, unless it was not possible to flush the buffers, for example.
- Start by specifying the API.2, CCL and DRV.1 components. If this does not isolate the problem, include extra components as necessary.

IBM does not recommend the use of the full CICS Universal Client trace to monitor performance in a production environment.

Performance monitoring tools

The following performance monitoring tools are available on the AIX operating system. Similar tools are available on other UNIX and Linux operating system:

vmstat
Gives statistics about virtual memory, disks, traps, and processor activity. Use it to determine system loading.

iostat
Gives processor statistics and I/O statistics for tty, disks, and CD-ROM drives.

sar (system activity report)
Collects, reports, or saves system activity information.

netstat
Shows network status.

IBM Performance Toolbox
Provides some useful performance tools that are integrated into a graphical user interface.

Refer also to the performance and tuning documentation for TCP/IP, CICS Transaction Server for z/OS, and TXSeries, and the documentation supplied with your operating system.

Chapter 8. Operating the Client daemon

This information describes how to operate the Client daemon.

The cicscli command

The cicscli command is used to start and shut down the Client daemon, check the availability of servers, and set other options.

The Client daemon starts automatically when you invoke any of the client functions such as EPI, ECI, or 3270 terminal emulation; you do not need to explicitly start the Client daemon, but you must to use the cicscli command to shut it down.

The following table shows the functions that you can perform with the cicscli command, and the options associated with each function:

Function	Options
Start the Client daemon, and start communication with CICS servers	-s
Shut down the Client daemon	-i or -x
Restart the Client daemon	-j or -y
Create autostart parameters for the Client daemon	-r
Start client tracing	-d
Use memory mapped tracing	-b
Stop client tracing	-o
Specify the client components to be traced	-m
Set up security	-c, -u, and -p
List connected CICS servers	-l
Disable all output messages	-q
Wait for confirmation before completion	-w
Display information about the version and build level of the Client daemon	-v

The following sections provide examples of using the cicscli command. Full details of the command syntax are in “cicscli command reference” on page 65.

Starting the Client daemon

To start the Client daemon, enter:

```
cicscli -s
```

To start the Client daemon and start communication with a CICS server, enter:

```
cicscli -s=servername
```

where *servername* is the name of a CICS server.

Starting connections with additional servers

To start a connection to a CICS server when the Client daemon is already running, enter:

```
cicscli -s=servername
```

where *servername* is the name of a CICS server.

Note: If you change and reinstall the CICS connection definition, you must stop and restart the connection.

Shutting down the Client daemon normally

To shut down the Client daemon for all connected servers, after all outstanding units of work have completed, enter:

```
cicscli -x
```

To shut down the session with a particular server, enter:

```
cicscli -x=servername
```

where *servername* is the name of a CICS server. This stops only the session with the named server; it does not shut down the Client daemon or connections to other servers.

Shutting down the Client daemon immediately

To shut down the Client daemon for all connected servers, without completing outstanding units of work, enter:

```
cicscli -i
```

To shut down the session with a particular server, enter:

```
cicscli -i=servername
```

where *servername* is the name of a CICS server. This stops only the session with the named server; it does not shut down the Client daemon or connections to other servers.

If the Client daemon does not shut down completely, this might be because the Client daemon process, cclclnt, remains active. To stop this process, enter the command

```
kill -2 pid
```

where *pid* is the numeric process id of cclclnt.

Do not use the `kill -9` command as this stops a process without allowing its resources to be released; those resources remain active until you restart the system.

Restarting the Client daemon normally

To shut down the Client daemon for all connected servers, after all outstanding units of work have completed, and then start it again, enter:

```
cicscli -y
```

`cicscli -y` is equivalent to `cicscli -x` followed by `cicscli -s`. This does not re-establish server connections or trace settings.

Restarting the Client daemon immediately

To shut down the Client daemon for all connected servers, without completing outstanding units of work, and then start it again, enter:

```
cicscli -j
```

`cicscli -j` is equivalent to `cicscli -i` followed by cicscli -s. This does not re-establish server connections or trace settings.

Starting client tracing

To start tracing the Client daemon, enter:

```
cicscli -d
```

To trace the Client daemon from the startup sequence, enter the -s and -d options together:

```
cicscli -d -s
```

The Client daemon writes trace entries to the cicscli.bin file in the `/var/cicscli` subdirectory. New trace entries overwrite any existing entries in the trace file. If required, make a backup copy of the old trace file before you start tracing.

Performance while tracing is on can be improved by using *memory mapped tracing*. With memory mapped tracing, data is stored initially in memory, and flushed to disk by the operating system's paging mechanism. For more information, see "Memory mapped tracing" on page 84. For important security information, see "Security considerations for trace and log files" on page 64.

To use memory mapped tracing, do the following:

1. Turn on wrapping trace by setting the **Client trace file wrap size (KB)** field in the Configuration Tool to a value greater than 0; see "Client trace file wrap size (KB)" on page 46 for details of the field setting.
2. When you turn on tracing, specify the `-b` option as well as the other options, for example:

   ```
   cicscli -d -b
   ```

   or

   ```
   cicscli -d -m=component_list -b
   ```

Use the cicsftrc utility to format the trace file; see "Formatting the binary trace file" on page 84.

Specifying the trace components

To specify which client components to trace, enter, for example:

```
cicscli -m=TRN,API.2
```

For information on which components you can trace, see "cicscli command reference" on page 65.

Stopping client tracing

To stop tracing the Client daemon, enter:

```
cicscli -o
```

Trace stops automatically if you enter the `cicscli -x command.`

Security considerations for trace and log files

The Client daemon restricts access to the client trace and log files. By default, these are normal files, not links, called cicscli.bin, and cicscli.log, in the /var/cicscli subdirectory. You can specify names for these files using the Configuration Tool.

The trace file

Normally, the file permissions on cicscli.bin allow only the owner, and group to write to the file, and only the owner to read it. However, a user who has write access to the /var/cicscli subdirectory can delete cicscli.bin regardless of the file permissions.

If however you use the `-b` option (see "Starting client tracing" on page 63), every process is given read access to the trace files.

If you do not want unauthorized users to have access to the cicscli.bin file, do not give them write access to the /var/cicscli subdirectory. For example, a command such as:

```
chmod 755 /var/cicscli
```

allows users to see files in /var/cicscli subdirectory but not to create, delete, or move them.

The Client daemon prevents you from starting tracing if an unauthorized user has deleted and recreated cicscli.bin.

Security and the error and warning log file

The file permissions on cicscli.log allow only the owner (root) and group to read and write the file.

To improve security further:

- Set the permissions on the /var/cicscli subdirectory to restrict general access
  ```
  chmod 0711 /var/cicscli
  ```
  This means users cannot even see which files are in this directory.
- Allow ECI and EPI programs, and terminals, to start the Client daemon, but allow only the root user to perform all other client administration. To do this, restrict access to the <install_path>/cicscli binary file, and allow general read and execute access to the /var/cicscli subdirectory.

Setting up security for the Client daemon

You can use the following commands after first starting the Client daemon.

To set a user ID to use when accessing server *servername*, enter:

```
    cicscli -c=servername -u=userid
```

where *userid* is the user ID and *servername* is the server name. (You are prompted for the password.)

To set a password to use when accessing server *servername*, enter:

```
    cicscli -c=servername -p=password
```

where *servername* is the name of the server and *password* is the password.

You can enter the -u and -p options together.

You can also specify the security parameters when you start the Client daemon or make a connection to a server:

```
cicscli -s=servername -u=userid -p=password
```

Displaying information about the version of the Client daemon

To display information about the version and build level of your Client daemon, enter the command

```
cicscli -v
```

Listing the connected servers

To list all the servers being used by the Client daemon, and their status, enter:

```
cicscli -l
```

The status of connected servers is updated as a result of requests being flowed and protocol specific events. The status returned is the last known state of connected servers, which might not be the same as the current state.

Disabling the display of messages from the Client daemon

To disable the display of all messages produced by the command, enter, for example:

```
cicscli -s -q
```

Destination for error messages

By default, the Client daemon sends error messages to the log file cicscli.log in the /var/cicscli subdirectory.

On AIX, you can redirect these messages to another target device or file by using the **swcons** command. You cannot use the `-e` option to send messages to the console.

Displaying command options for the Client daemon

To display the options of the cicscli command, enter:

```
cicscli -?
```

---

cicscli command reference

This section provides a reference guide for the cicscli command.

All client control commands have options identified by a leading dash (-).

- On Linux platforms you can replace the dash by a forward slash (/) for all options.
- On AIX platforms you can replace the dash by a forward slash (/) for all options except the **?** option, where you must use a dash sign.

All options of the form `-x=variable` may contain spaces in the variable part, if it is enclosed in double quotes. Double quotes within variables must be entered as \" , that is with a backslash preceding the double quote.

```
cicscli [[-s[=servername]]|[-x[=servername]]|[-i[=servername]]|[-j]|[-y] ]
[-l]
[[-d[=nnn] [-b] [-m[=components]]]|-o]
[-n|-e]
```

```
[-c=servername [-u[=userid]] [-p[=password]]]
[-w|-q]
[-v]
[-y|-j]
```

The options are:

-b Uses memory mapped trace. This improves performance significantly compared with standard file I/O, but has implications for security. This option is valid only if specified when tracing is switched on, and requires wrapping trace to be enabled. See "Starting client tracing" on page 63 for more details.

-c=*servername*
Identifies the name of the server to which security information in the form of a user ID and password is to be associated. Some CICS servers require the user to provide security information to the server before interacting with the server. The Client daemon prompts the user at the workstation for a user ID and password, unless you provide them by using the -u and -p options.

-d=[*nnn***]**
Starts debug tracing for the Client daemon. If tracing is required while the Client daemon is starting up, this option may be specified along with the -s option.

nnn is the maximum size of data areas to be traced in bytes. The range is 1 through 32 767 bytes, and the default value is 512 bytes. If you are producing a trace for your support organization, set `-d` to at least 1000 to ensure that all relevant data is included.

The Client daemon writes trace entries to the cicscli.bin file in the /var/cicscli subdirectory. New trace entries overwrite any existing entries in the trace file. If required, make a backup copy of the old trace file before you start tracing.

Use the cicsftrc utility to format the trace file. The resulting file, cicscli.trc, is an ASCII file that you can read with a text editor. For more information see "Formatting the binary trace file" on page 84.

-e Enables the display of Client daemon error and security messages in pop-up windows. This option has no effect on UNIX and Linux operating systems; pop-up messages are written to the error log.

-i Shuts down the Client daemon immediately. The Client daemon does not wait for outstanding units of work to complete. Stopping the Client daemon in this way can result in a loss of data at connected servers. To stop the Client daemon normally, use the -x option.

-j Shuts down the Client daemon immediately and then restarts it.

A restart involves shutting down the Client daemon, waiting for it to shut down, and then starting it again. cicscli -j is equivalent to cicscli -i followed by cicscli -s. Server connections are not re-established when the Client daemon is restarted.

The Client daemon does not wait for units of work to complete. This might result in loss of data. To restart the Client daemon normally, use the -y option.

-l Displays a list of all connected servers. For each server, the netname of the server as it is known to the Client daemon is also displayed, as well as the state of the connection to the server and the connection protocol.

-m=[*components***]**

Specifies a comma-separated list of identifiers for components that will be traced when tracing starts. You can specify any of the following components:

ALL	All components. Use it if performance allows, and consider using the binary formatting tool to filter information. See "Formatting the binary trace file" on page 84 for details.
API.1	The client API layer (level 1). This traces the boundary between the Client application and the Client daemon.
API.2	The client API layer (level 1 and 2). This gives level 1 plus additional parameter tracing.
API	Synonymous with API.1.
CCL	The Client daemon.
CPP	The C++ class libraries.
CLI	The cicscli command interface.
DEF	The default components, that is the API, CCL, and DRV.1 components.
DRV	Synonymous with DRV.1.
DRV.1	Protocol driver tracing level 1. This traces data sent and received and provides supplementary information about failures.
DRV.2	Protocol driver tracing level 2. This traces internal flows through the protocol drivers and interactions with other software components. It is currently supported only by the CCLTCP62 protocol driver.
EMU	cicsterm and cicsprnt emulators.
TRN	The TRN component traces the internal interprocess transport between Client processes. Use it if entries in the Client log refer to functions such as FaarqGetMsg, FaarqPutMsg, or FaarqStart. TRN is the most verbose tracing component.

- If you are not sure whether a problem is inside the Client daemon, and you want to minimize the impact on performance, specify the DRV.1 and API components. You can also specify these components to help to determine which product is causing the system to lock up.
- Specify the API and CCL components if you believe that the problem is within the Client daemon.
- If you want to diagnose an application error, and are not interested in the Client daemon, specify only the API.2 and CPP tracepoints. The trace contains less information, and is easier to understand.

The -m option does not start tracing in the Client daemon; it simply specifies the components to trace. You cannot use -m while the Client daemon is not running, so in this case you must specify the -s option as well. Consider using wrapping trace (-b) for improved performance:

```
cicscli -m=component_list -b
```

If you specify -m with no parameters, a list of the possible component identifiers is displayed, with an 'x' beside each component that it is currently enabled for tracing.

You can also specify settings for trace components using the Configuration Tool (see "Trace settings" on page 45 in the Configuration chapter for details). Any component tracing specified using cicscli overrides that specified using the Configuration Tool. If component tracing is not specified either by the cicscli command or by using the Configuration Tool, a default set of components is traced, namely: DRV.1, CCL, and API. Any component tracing specified using the Configuration Tool overrides the default set of components.

Note that the cicscli -d=*nnn* command is used to set the maximum size of the data areas to be traced. The trace data might be truncated if you set *nnn* lower than the size of data expected.

-n
Disables the display of Client daemon error and security messages in pop-up windows. This option has no effect on UNIX and Linux operating systems; pop-up messages are written to the error log.

-o
Stops tracing if it is already active.

-p=*password*
Sets the current password to be used when accessing the server specified by the -c option. This password is used if the server requires a password (and user ID) before running transactions on the client's behalf.

For ECI applications, any user ID and password specified in the ECI parameter block override values set by the cicscli command.

Specifying -p or -p= (that is, no password is specified) resets the associated password to a null value.

-q
Disables the display of all messages produced by the cicscli command.

-s
Starts the Client daemon. No attempt is made to initiate communication with a server unless -s=*servername* is specified. In this case, the Client daemon also connects to the server using information specified in the configuration file. The *servername* must correspond to an entry in the configuration file.

-u=*userid*
Sets the current user ID to be used when accessing the server specified by the -c option. This user ID is used if the server requires a user ID (and password) before running transactions for the Client daemon.

If you do not provide the -p option, you are prompted for the password.

For ECI applications, any user ID and password specified in the ECI parameter block override values set by the cicscli command.

Specifying -u or -u= (that is, no user ID is specified) resets the associated user ID to a null value.

-v
Displays information about the version and build level of the Client daemon.

-w
Prompts the user, before the command completes, to press the Enter key, to confirm that information and error messages output to the screen have been read.

-x
Shuts down the Client daemon normally. If -x=*servername* is specified, the connection to the server is terminated when all outstanding units of work on the specified server have completed. If other server connections are active, these remain unchanged.

If -x is specified without a server name, the Client daemon waits for all outstanding units of work to complete, terminates all connections to servers, and ends the control process. Using -x or -x=*servername* is the preferred way of stopping the Client daemon.

-y Restarts the Client daemon normally.

A restart involves shutting down the Client daemon, waiting for it to shut down, and then starting it up again. cicscli -y is equivalent to cicscli -x followed by cicscli -s. Server connections are not reestablished when the Client daemon is restarted.

Using -y is the preferred way of restarting the Client daemon.

-? Causes the command syntax to be displayed.

Changing the system time

You can change the system time while the CICS Universal Client is running. If you change the system time, an active process responds to the changed time and not to the elapsed time.

When setting clocks forward or back, the behavior of timeout settings will change. When the clock is set back the elapsed time for a timeout may be increased. When the clock is set forward a timeout may expire earlier. The maximum elapsed time for a timeout will be the original timeout value plus the value of the change in time. For example, if the current time is 19:00, and a timeout is set to expire 5 minutes from now; the effect of setting the clock back by 1 hour is to increase the value of the timeout by 1 hour. The total elapsed time for the timeout is 1 hour and 5 minutes.

The following time-outs are effected by this change:
- ECI time-outs
- EPI install timeout
- EPI read timeout
- Client daemon connection timeout.
- Client daemon load manager region timeout
- Client daemon server retry interval
- Gateway daemon idle timeout
- Gateway daemon ping timeout
- Gateway daemon close timeout
- Server idle time-outs
- Worker thread available thread timeout
- EXCI timeout

Chapter 9. Terminal Emulation

This information describes how to use the cicsterm and cicsprnt programs.

Summary of Terminal Emulation commands

cicsterm
This command starts a terminal emulation session with particular options.

cicsprnt
This command starts a printer terminal session with particular options.

The cicsterm command

The cicsterm command controls 3270 terminal emulation. You can start emulator sessions, specify terminal emulator characteristics, and the names of the keyboard mapping and color mapping files.

You can have multiple terminal emulation sessions running simultaneously.

The Client daemon does not support 3270 field outlining for the cicsterm command.

Window resizing is not supported. If you resize the window in which cicsterm is running the display becomes distorted.

The CICS Transaction Server interprets cicsterm as a remote terminal. The execution diagnostic facilities CEDF and CEDX have some restrictions on remote terminals. Refer to your CICS Transaction Server documentation for details.

Using cicsterm

The following table shows the functions that you can perform with the cicsterm command, and the parameters associated with each function:

Function	Parameters
Start a 3270 terminal emulator	-s and -r
Specify the initial transaction	-t
Specify the name of the keyboard mapping file	-k
Specify the name of the color mapping file	-c
Define the 3270 terminal emulator characteristics	-n and -m
Specify a terminal emulator that is sign-on incapable	-a
Determine the print file processing	-p
Specify a file to which print files are appended	-f
Wait for confirmation before completing	-w
Inhibit all output messages	-q

Issue a single cicsterm command, with all the parameters you require, as shown in the following example:

```
cicsterm -s=CICSTSW -t=CESN -k=mykeys.ini -c=mycols.ini
         -n=cicsv123 -f=clprint.txt -q
```

In this example:

-s=CICSTSW
A 3270 terminal emulator is started for the server `CICSTSW`.

-t=CESN
The initial transaction is `CESN`.

-k=mykeys.ini
The keyboard mapping file is `mykeys.ini`.

-c=mycols.ini
The color mapping file is `mycols.ini`.

-n=cicsv123
The 3270 terminal emulator characteristics are defined by the terminal definition `cicsv123`.

-f=clprint.txt
The print file will be appended to the file `clprint.txt`.

-q The display of messages output by the command is disabled.

All cicsterm parameters are optional. If you enter the cicsterm command without any parameters, defaults are taken from the configuration file. Full details of the parameters are given in “cicsterm command reference” on page 73.

Stopping a terminal emulator

To stop a terminal emulator, enter the string specified by the Terminal exit configuration setting from an empty terminal screen, and then press Enter. The default string is `EXIT`

cicsterm and user exits

You can use cicsterm to drive EPI user exits.

The EPI user exits, and how cicsterm can use them, are described in *CICS Transaction Gateway: Programming Guide*.

cicsterm and RETURN TRANSID IMMEDIATE

When an application running from a cicsterm session issues either of the commands:

```
EXEC CICS RETURN TRANSID(name) IMMEDIATE
EXEC CICS RETURN TRANSID(name) IMMEDIATE INPUTMESSAGE(data-area)
```

the transaction named in the TRANSID option starts immediately without any user input. When the INPUTMESSAGE option is specified, the contents of the *data-area* are passed to the new transaction, and the screen is not updated with the *data-area* contents.

Issuing these EXEC CICS commands from cicsterm does not result in the StartTranExit or ReplyExit user exits being driven. See *CICS Transaction Gateway: Programming Guide* for more information.

Using clients for X-Window System

There are some known problems with certain clients for X-Window System (such as Exceed). These include corruption of the text on the title bar of the window that you are trying to display, for example, with the Configuration Tool. In some cases the title bar might be missing. The problems are with the client used, not the CICS Universal Client.

There are two ways to solve this problem:

1. Start a window manager before launching the Configuration Tool.

   The window manager HWM is shipped with Exceed.
2. Alter the Exceed configuration:
   a. Launch Xconfig
   b. Select **screen definition**
   c. Set the Window Manger to *native*
   d. Make sure **Use Native WM for Embedded Clients** is checked

   If you use this second way you cannot run any other window manager.

Keyboard mapping in cicsterm

Keyboard mapping in cicsterm is governed by the terminal type that you are using. Many terminal types do not support all of the function keys that can be used in a CICS application. If cicsterm does not recognize some of the key mappings defined by the <install_path>/bin/cicskey.ini file, try using a different terminal type. For example, on AIX systems use *aixterm* in preference to *xterm*.

cicsterm command reference

The dash (-) may be replaced with the forward slash (/) character as follows:

- On Linux platforms you can replace the dash by a forward slash (/) for all parameters.
- On AIX platforms you can replace the dash by a forward slash (/) for all parameters except the **?** parameter, where you must use a dash sign.

```
cicsterm [-s[=servername]|-r[=servername]]
[-t=[initialtransid]]
[-k=keyfile]
[-c[=colorfile]]
[-m[=modelname]]
[-a]
[-n=netname]
[-p=printcmd|-f=printfile]
[-q|-w]
[-?]
```

The options are:

-a
Specifies that the terminal emulator is *not* sign-on capable. By default, cicsterm creates terminal emulators that are sign-on capable.

For more information on sign-on capability, see *CICS Transaction Gateway: Programming Guide*.

-c=*colorfile*
Identifies the name of a color mapping file to be used with the emulator; see "Customizing the screen colors for cicsterm" on page 52 for more details. If you omit this parameter, the environment variable CICSCOL is

assumed to identify the color mapping file. If CICSCOL is not defined, a file name of cicscol.ini in the <install_path>/bin subdirectory is assumed.

If the parameter is specified as -c=, that is, the color mapping file name is omitted, the emulator runs without any color definitions.

-f=*printfile*
Specifies the name of a file to which the output of print requests is appended. If you do not specify a full path, *printfile* is created in the <install_path>/bin directory. If the name of the file contains embedded blanks, it must be surrounded by double quotes (“). Any double quotes within the name of the file must be entered as backslash double quote (\“).

If neither the -f or -p parameters is specified, the **Print command** or **Print file** configuration settings define the command, file, or default action to take with print requests.

-k=*keyfile*
Identifies the name of a keyboard mapping file to be used with the emulator; see “Keyboard mapping for cicsterm” on page 50 for more details. If this parameter is omitted, the environment variable CICSKEY is assumed to identify the key mapping file. If CICSKEY is not defined, a file name of cicskey.ini in the <install_path>/bin subdirectory is assumed.

-m=*modelname*
Specifies the name of a Model terminal definition, as known at the server to which the emulator is to connect, to be used to define the terminal characteristics. If neither this parameter nor -n=netname is specified, any Model terminal definition value from the configuration file is used. If no Model terminal definition value has been specified in the configuration file, the server’s default terminal definition is assumed.

If the parameter is specified as -m= (that is, the modelname is omitted), any Model terminal definition value specified in the configuration file is ignored, and the server’s default terminal definition is assumed.

This option is case-sensitive.

-n=*netname*
Specifies the name of a particular terminal definition at the server that this emulator is to be installed as. The precise interpretation of netname varies between servers.

This option is case-sensitive.

-p=*printcmd*
Specifies an operating system command used to process the temporary print file generated when print requests are received by the terminal emulator. If the command contains embedded blanks, enclose it in double quotes (“). Any double quotes within the command must be entered as backslash double quote (\“).

The temporary print file is post-processed by appending the file name to the command, and executing the resultant command. Thus print output may simply be copied to a local printer, copied into a permanent file, processed further for inclusion into a document, and so on. If the temporary file is to be processed by a print command, the command is responsible for deleting the temporary file.

If neither the -f or -p parameters is specified, the **Print command** or **Print file** configuration settings define the command, file, or default action to take with print requests.

-q Disables the display of all messages output by the command.

-s=*servername* **or -r=***servername*
Specifies the name of the server that the terminal emulator is to be connected to. This server name must correspond to an entry in the configuration file.

You can specify -s, or -r, but not both. If neither parameter is specified, the first server entry in the configuration file is used.

If the parameter is specified as -s or -r (that is, no server name is provided), and the configuration file identifies more than one potential server to which the Client daemon can connect, the user is prompted to select from a list of available servers. These prompts are generated even if messages have been disabled (the -q parameter is specified).

If there is only one potential server identified in the configuration file, that server is used and the user is not prompted.

-t=*initialtransid*
Identifies the initial transaction to be invoked for this terminal. If this option is omitted, any initial transaction specified in the configuration file is run. The string can be up to 128 characters long, specifying both a transaction name, and parameters to be passed to the transaction. The transaction name is the first four characters or the characters up to the first blank in the string. The rest of the string is the parameter data.

If the parameter is specified as -t= (that is, the initialtransid is omitted), any initial transaction specified in the configuration file is ignored.

This option is case-sensitive.

Ensure that transaction that you specify here does not require terminal input to complete.

-w Prompts the user to press the Enter key, to confirm that messages output to the screen (both informational and error) have been read, before the command completes.

-? Causes the parameter syntax to be listed; any other options specified are ignored.

The cicsprnt command

The cicsprnt command controls 3270 printer terminal emulation.

An application running on a server can direct output to a printer in one of the following ways:

- An application running from a terminal can initiate printing by sending a map or data with the PRINT indicator set.
- A user can start a 3270 Print Terminal Emulator, at a client, using the cicsprnt command. A 3270 Print Terminal Emulator must be started for a netname or model terminal definition predefined in the server's terminal tables. Output is directed to such a device by starting a transaction against the printer device.
- At client workstations you can use the PrintScreen key, as defined by the keyboard mapping file. However, any lines that contain only null characters are not printed. For a 'blank' line to be printed, it must contain at least one space character.

Using cicsprnt

The following table shows the functions that you can perform with the cicsprnt command, and the parameters associated with each function:

Function	Parameters
Start a 3270 print terminal emulator.	-s and -r
Specify the initial transaction.	-t
Define the 3270 printer terminal emulator characteristics.	-n and -m
Determine the print file processing.	-p
Specify a file to which print files are appended. The default location is <install_path>`/bin`.	-f
Wait for confirmation before completing.	-w
Inhibit all output messages.	-q
Issue one print job per transaction.	-j
Display help	-?

Issue the cicsprnt command once with all the parameters you require, as shown in this example:

```
cicsprnt -s=CICSTSW -n=P123 -t=XPRT -f=clprint.txt -q
```

In this example:

-s=CICSTSW
A 3270 print terminal emulator is started for the server `CICSTSW`.

-n=P123
The 3270 print terminal emulator characteristics are defined by the terminal definition `P123`

-t=XPRT
The initial transaction is `XPRT`.

-f=clprint.txt
The print file to which print requests are appended is `clprint.txt`, in the default location (<install_path>`/bin`).

-q The display of messages output by the command is disabled.

You must specify at least one of these parameters:

-n=*netname*
-m=*modelname*

All other parameters are optional, and defaults are taken from the configuration file. Full details of the parameters are given in “cicsprnt command reference” on page 77.

If the system running the Client daemon supports DBCS, it is assumed that the printer attached to the processor also supports DBCS. Conversely, if the system does not support DBCS, the Client daemon will not send DBCS data to the printer.

cicsprnt and user exits

You can use cicsprnt to drive EPI user exits.

The EPI user exits, and how cicsprnt can use them, are described in *CICS Transaction Gateway: Programming Guide*.

cicsprnt and RETURN TRANSID IMMEDIATE

Unlike cicsterm, cicsprnt does not support these commands:

```
EXEC CICS RETURN TRANSID(name) IMMEDIATE
EXEC CICS RETURN TRANSID(name) IMMEDIATE INPUTMESSAGE(data-area)
```

For more information, refer to “cicsterm and RETURN TRANSID IMMEDIATE” on page 72.

cicsprnt command reference

The dash (-) may be replaced with the forward slash (/) character as follows:

- On Linux platforms you can replace the dash by a forward slash (/) for all parameters.
- On AIX platforms you can replace the dash by a forward slash (/) for all parameters except the **?** parameter, where you must use a dash sign.

```
cicsprnt -m=modelname|-n=netname
[-s[=servername]|-r[=servername]]
[-t=[initialtransid]]
[-p=printcmd|-f=printfile]
[-q|-w]
[-j]
[-?]
```

The options are:

-f=*printfile*
: Specifies the name of a file to which the output of print requests is appended. If you do not specify a full path, *printfile* is created in the <install_path>/bin directory. If the name of the file contains embedded blanks, it must be surrounded by double quotes (“). Any double quotes within the name of the file must be entered as backslash double quote (\“).

If neither of the -f or -p parameters is provided, the **Print command** or **Print file** setting in the configuration file defines the command, file, or default action to take with print requests.

-j
: Specifies that cicsprnt should concatenate all EXEC CICS SEND PRINT commands issued on a server transaction into a single print job. This print job is issued when the transaction terminates. Otherwise cicsprnt will generate a separate print job for every EXEC CICS SEND PRINT command issued for a server transaction.

-m=*modelname*
: Specifies the name of a model terminal definition, as known at the server to which the 3270 Print Terminal emulator is to connect, to be used to define the terminal characteristics. If this parameter is not specified, any Model terminal definition value from the configuration file is used. If no Model terminal definition value has been specified in the configuration file, the server’s default terminal definition is assumed.

You must specify either the -m or the -n option, or both.

This option is case-sensitive

-n=*netname*
: Specifies the name of a particular terminal definition at the server that this

3270 Print Terminal emulator is to be installed as. The precise interpretation of netname varies between servers. For example, on TXSeries for AIX it is a netname.

You must specify either the -m or the -n option, or both.

This option is case-sensitive.

-p=*printcmd*
Specifies a command used to process the temporary print file generated when print requests are received by the terminal emulator.

If the command contains embedded blanks, the command must be surrounded by double quotes ("). Any double quotes within the command must be entered as backslash double quote (\").

If neither of the -f or -p parameters is specified, the **Print command** or **Print file** setting in the configuration file defines the command, file, or default action to take with print requests.

The temporary print file is post-processed by appending the file name to the command, and executing the resultant command. Thus print output may simply be copied to a local printer, copied into a permanent file, processed further for inclusion into a document, and so on. If the temporary file is to be processed by a print command, the command is responsible for deleting the temporary file.

-q Disables the display of all messages output by the command.

-s=*servername* **or -r=***servername*
Specifies the name of the server that the printer is to be connected to. This servername must correspond to an entry in the configuration file. You can specify -s, or -r, but not both.

If neither parameter is specified, the first server entry in the configuration file is used.

If the parameter is specified as -s or -r (that is, no servername is provided) then, if the configuration file identifies more than one potential server to which the Client daemon can connect, the user is prompted to select from a list of available servers. These prompts are generated even if the -q parameter is specified.

If there is only one potential server identified in the configuration file that server is used and the user is not prompted.

-t=*initialtransid*
Identifies the initial transaction to be invoked for this printer. If this option is omitted, any initial transaction specified in the configuration file is run. The string may be up to 128 characters long, specifying both a transaction name, and parameters to be passed to the transaction. The transaction name is the first four characters or the characters up to the first blank in the string. The rest of the string is the parameter data.

If the parameter is specified as -t= (that is, the initialtransid is omitted), any initial transaction specified in the configuration file is ignored.

Note: Be careful that transactions that you specify either here or in the configuration file do not require terminal input to complete.

This option is case-sensitive.

-w Prompts the user, before the command completes, to press the Enter key, to confirm that messages output to the screen (both informational and error) have been read.

-? Causes the parameter syntax to be listed; any other options specified are ignored.

Chapter 10. Problem determination and problem solving

Most problems in a CICS Universal Client system are caused by one of the following:

- User errors
- Programming errors
- Configuration errors

Problem determination: preliminary checks

Before investigating a problem, check whether there is an obvious cause:

- Has the system run successfully before now?
- Have you made any changes to the configuration of the system or added new features or programs?
- Is the environment configured correctly?
- Can the Client daemon connect to the CICS server?

  Use the command:

  ```
  cicscli -s=servername
  ```

  followed by the command:

  ```
  cicscli -l
  ```

  If the connection is good, this command returns the name of the server that you specified in *servername*.
- Are there any messages explaining the failure?
- Can you re-create the failure?
- Do any of the available statistics help to explain the failure?

What to do next

If you think the problem is with the CICS Universal Client, see “Diagnosing common problems” on page 88. Here you will find information on diagnosing common configuration and other errors, as well as advice on the documentation you need to collect in specific situations.

See “Program support” on page 96 for information on how to contact IBM. See “Documenting problems” on page 97 for the information that you will be asked to provide.

If you think the problem is in another product, follow the problem determination procedures provided with that product. For CICS servers, see the *CICS Problem Determination Guide*.

Problems installing the product

This information describes problems that you might meet when installing or uninstalling the CICS Universal Client.

A message indicates that the CICS Universal Client is running

You might see this message when migrating from an earlier version, or when installing the product. Check the following components, and stop them as necessary:

Client daemon
This can be seen in the process list as `cclclnt`. See "The cicscli command" on page 61 for information on stopping the Client daemon.

Configuration Tool
Close the Configuration Tool.

Installation fails with a ServiceException message

Symptoms
The following message is seen during installation:

```
>WARNING: could not write using log service: ServiceException:
(error code = 200; message = "/tmp/cicstgInstall.log
(Permission denied)";
severity = 0; exception = >[java.io.FileNotFoundException:
/tmp/cicstgInstall.log (Permission denied)])
```

Cause A log file from a previous installation of the CICS Universal Client was created by the root user ID. The current installation is being attempted by a non-root user ID that cannot update the log file; this causes the installation to fail.

Solution
Only root can install the CICS Universal Client. Change to the root user ID and rerun the installation.

Silent installation fails

If a silent installation does not work correctly, it might not send an error to the console that produced the problem. Appending `-is:log <`*`file_name`*`>` to the installer command causes details of any errors to be output to a file. For example, to install silently using response file `responseFile`, and log errors to `logFile`, enter the following command:

```
installer -options "responseFile" -silent -is:log logFile.txt
```

Problems with the Client daemon

Messages

There are two types of message in the Client daemon:

1. Messages displayed to the user
2. Errors written to the Client daemon error log and trace file

The *CICS Transaction Gateway: Messages* book explains all error and warning messages.

Error log messages

Any errors on the client workstation that are not caused by incorrect use of the API are written to the Client daemon error log.

The log file is named in the configuration file. The default name is cicscli.log. The log file is written to the /var/cicscli directory.

Help information for all messages is provided in two ASCII text files in the <install_path>/bin subdirectory. You can view these using any standard text editor:

cclmsg.hlp
Help for the end user messages, in the language you chose at installation time.

ccllog.hlp
Help for the trace and log messages. This is provided in English only.

Errors resulting from incorrect use of the APIs simply result in error return codes to the application. It is the responsibility of the application to notify the end user of the error and provide guidance on corrective actions.

Messages from the Client daemon process

Pop-up messages are not displayed on UNIX and Linux systems; they are written to the error log instead.

Tracing

Client daemon tracing is a useful problem determination tool for communication problems. You can use the trace functions to collect detailed information on the execution of a certain function or transaction. A trace can show how the execution of a particular activity is affected by, for example, the execution of other tasks in a CICS system. Each trace entry has a time stamp, which provides information on the time taken to perform certain activities.

To learn how to turn tracing on, see “Starting client tracing” on page 63.

For information on specifying the components of the Client daemon to be traced, see the cicscli -m command.

The output from the trace function is a binary trace file called, by default, cicscli.bin in the /var/cicscli subdirectory. You can specify a different name for this file, using the Configuration Tool. However, you cannot change the .BIN extension. Using the **Client trace file wrap size (KB)** configuration setting, you can specify that the binary trace file should wrap into a second trace file, and you can also specify the maximum size of these files.

To read the trace, run the cicsftrc utility to convert the binary file or files into a text file. This text file is called cicscli.trc by default. The default trace files are:

cicscli.bin
The binary trace file produced by running the Client daemon trace.

cicscli.wrp
The second binary trace file if wrapping of client trace is enabled.

cicscli.trc
The name of the text trace file produced when the binary trace file is converted to a text file using the cicsftrc utility.

cicscli.bak
The backup file of the binary trace file. A backup file is produced from any existing .BIN file when you turn tracing on.

cicscli.bbk
The backup of the first binary trace file if memory mapped tracing is enabled.

cicscli.wbk*n*
The backup of subsequent binary trace files, if memory mapped tracing is enabled, where *n* is the number of the original .WRP file.

See "Formatting the binary trace file" for information on the trace conversion utility.

Wrapping the Client daemon trace

You can control the size of the binary trace file by specifying that it wraps into a second trace file. Use the **Client trace file wrap size (KB)** configuration setting to turn on wrapping trace; specify the maximum size of the wrapping trace (in kilobytes). If this value is 0 (the default), wrapping trace is not turned on.

When wrapping trace is turned on with standard I/O tracing, two files (called cicscli.bin and cicscli.wrp) are used. Each file can be up to half the size of the **Client trace file wrap size (KB)** value.

Memory mapped tracing

If you enable wrapping trace, you can use the `-b` switch when you issue the `cicscli` command to turn tracing on. This specifies that memory mapped trace files should be used.

With memory mapped tracing, the operating system's paging mechanism is used to swap data between memory and the trace file. This improves performance significantly when compared to standard file I/O, because the trace file is opened and written to less frequently. Because the operating system is responsible for flushing data to disk, data is not normally lost if an application terminates unexpectedly. However, if the operating system itself fails, data can be lost, and the trace file can be corrupted. If you are diagnosing problems where the server fails and needs to be restarted, use standard I/O tracing instead of memory mapped tracing.

If you use memory mapped tracing, the size of the trace files is limited to 10 MB, and in addition to cicscli.bin and cicscli.wrp, you might see a series of files of the form cicscli.wrp1, cicscli.wrp2...cicscli.wrp*n*, where *n* is the number of files needed to hold the total amount of trace data specified in the Client trace file wrap size (KB) field of the Configuration Tool; see "Client trace file wrap size (KB)" on page 46 for the maximum amount of data that can be specified. The trace formatter finds all files in the sequence when you format the binary files. Memory mapped tracing uses up to 10 MB of virtual storage.

See "Starting client tracing" on page 63 for details of how to issue the command, and "Security considerations for trace and log files" on page 64 for important information on security.

Formatting the binary trace file

If you are using memory mapped tracing, note that data is not always written to disk immediately. As a consequence, turn tracing off before you format the binary trace file, to ensure that all data is flushed to disk.

You use the Binary Trace Formatter utility cicsftrc to convert the binary trace file cicscli.bin to ASCII text. The utility has the following parameters:

-m=*list of components*
Specifies that only trace points from the listed components are written to the

text file. The components you can specify are the same as for cicscli -m; see the cicscli -m command. If -m is not specified, all trace points in the binary trace are written to the text file.

-w[=*filename*]
Indicates that there are two or more binary trace files to format and then concatenate (that is, the binary files were created with a wrapping trace). If no file name is specified with the -w parameter, cicsftrc assumes that the name of the second trace file is cicscli.wrp.

-n Indents entry and exit points in the test trace file to make it more readable. By default, indentation is turned off.

-d Specifies detailed trace formatting. If you are using EPI calls, cicsterm or CICSPRINT, an approximation of the screen layout will be included in the trace.

-i=*filename*
Specifies the name of the input (binary) trace file, which is cicscli.bin by default.

-o=*filename*
Specifies the name of the output (text) trace file. If no -o parameter is specified, the name of the text trace file is assumed to be cicscli.trc.

-f Overwrite any existing files.

-s Do summary trace formatting. Summary trace formatting is controlled by a template file (cclsumtr.txt), which is read in at initialization time. It formats key trace points, and shows for example the flow of user API calls, the progress of calls through the Client daemon, and network flows to the server. For the most detailed results, specify the API.2 component when you define the components to be traced. Summary tracing provides an overview; use it as requested by your IBM support organization.

Note: Versions of the CICS Universal Client earlier than 5.0.1 cannot format binary trace files produced by this version.

Figure 5 shows an example summary trace.

```
       -->Sample of API summary trace taken with API.2 and DRV options.

[Process ,Thread  ]     Time       API Summary                   CCLCLNT Summary               Comms Summary
=============================================================================================================
...
...
...
[000000bf,0000017c] 12:08:32.190 --->[7315] CCL3310 ECI Call type ECI_SYNC, UOW=0
[00000089,000000a4] 12:08:32.290                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=89
[00000089,00000063] 12:08:32.400                                              <-R-[4418] CCL4412 TCP/IP (to STEMPLAR) receive data: Length=12
[00000089,0000018b] 12:08:32.511                                              <-R-[4418] CCL4412 TCP/IP (to STEMPLAR) receive data: Length=29
[00000089,000000a4] 12:08:32.521                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=94
[00000089,000000a4] 12:08:32.531                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=94
[00000089,000000a4] 12:08:32.541                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=94
[00000089,000000a4] 12:08:32.541                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=94
[00000089,000000a4] 12:08:32.551                                              -S->[4410] CCL4411 TCP/IP (to STEMPLAR) send data: Length=94
[00000089,00000168] 12:08:32.581                                              <-R-[4418] CCL4412 TCP/IP (to STEMPLAR) receive data: Length=12
[00000089,0000017e] 12:08:32.601                                              <-R-[4418] CCL4412 TCP/IP (to STEMPLAR) receive data: Length=31
[000000bf,0000018e] 12:08:32.621     [7364] CCL3350   Event Service Thread got a request REQUEST_TYPE_ECI_1
[000000bf,0000008a] 12:08:32.671 <---[7316] CCL3311 ECI Call type ECI_SYNC, UOW=0 got rc=ECI_NO_ERROR {Time in API = 0.821 seconds}
```

Figure 5. Sample of API summary trace taken with API.2 and DRV options

Points to note:

1. {Time in API} shows the amount of time that the client API call took to complete. This can help when investigating performance problems; see Chapter 7, “Performance,” on page 59 for more information.
2. The API Summary column refers to client API code inside the user application process. It tracks when user requests enter and

leave the client API code. `--->` and `<---` show the program entering and leaving the Client daemon API.

3. CCLCLNT is the background Client daemon process. You get entries here only if you specify the CCL component.
4. The Comms Summary tracks when Client daemon calls enter and leave the network. `-S->` shows a request being sent to the network; `<-R-` shows a reply being received.

If a user application is making EPI calls, or using cicsterm or cicsprnt, the trace formatter puts an approximation of the screen into the trace. Figure 6 shows a 3270 screen capture from a formatted trace file, taken from the CECI transaction. It is an aid to problem determination, not a completely accurate representation of the screen. See "Formatting the binary trace file" on page 84 for details of how to format the trace file.

```
Command = Erase/Write, so clearing main screen
 Command2 = Read Modified
 WCC = 0x32 ( Free Kbd,80 char)
 Set Buffer Address to (1,2)
 Insert Cursor @ (1,2)
 Set Buffer Address to (1,1)
 Start Field Extended (Unprotected,Alphanumeric,Display,not-pen-detectable,Foreground Colour Green)
 Data : ' '
 Insert Cursor @ (1,3)
 Set Buffer Address to (2,1)
  Data : 'User        '
  .....
  .....
  .....
  .....
 Set Buffer Address to (24,49)
 Start Field Extended (Autoskip (Prot+Num),Display,not-pen-detectable,Foreground Colour Turquoise)
 Data : '9'
 Set Buffer Address to (24,51)
 Start Field Extended (Unprotected,Alphanumeric,Intense,pen-detectable,Foreground Colour Red)
 Data : 'Messages    '
             1         2         3         4         5         6         7         8
    12345678901234567890123456789012345678901234567890123456789012345678901234567890
 >+----------------------------------------------------------------------------------+
01| -                                                                                |
02| ´STATUS. . :´Enter one of the following:                                         |
03|     ´                                                                            |
04| ´ABend       EXtract     READPrev    WAit                                        |
05| ´ADdress     FEpi        READQ       WRITE                                       |
06| ´ALlocate    FOrmattime  RECeive     WRITEQ                                      |
07| ´ASKtime     FREE        RELease     Xctl                                        |
08| ´ASSign      FREEMain    RESetbr                                                 |
09| ´Bif         Getmain     RETRieve                                                |
10| ´CAncel      Handle      RETUrn                                                  |
11| ´CHange      IGnore      REWrite                                                 |
12| ´CONNect     INquire     SENd                                                    |
13| ´CONVerse    ISsue       SET                                                     |
14| ´DELAy       LInk        SIGNOFf                                                 |
15| ´DELETE      LOad        SIGNON                                                  |
16| ´DELETEQ     PErform     START                                                   |
17| ´DEQ         POP         STARTBr                                                 |
18| ´DUmp        POSt        SUspend                                                 |
19| ´ENDbr       PUsh        SYncpoint                                               |
20| ´ENQ         READ        Unlock                                                  |
21| ´ENTer       READNext    Verify                                                  |
22|     ´                                                                            |
23| ´PF´1-Help        ´2-HEX         ´3-End         ´4-EIB         ´5-Variables      |
24|    ´6-User                                      ´9-Messages                      |
  +----------------------------------------------------------------------------------+
  | 1BþC000                   STEMPLAR                                               |
  +----------------------------------------------------------------------------------+
    12345678901234567890123456789012345678901234567890123456789012345678901234567890
             1         2         3         4         5         6         7         8
```

Figure 6. Screen capture from a formatted trace file

The formatter lists the commands which built the screen, and shows an approximation of the screen.

Format of trace entries

The entries in the Client daemon trace file have the following format:

```
time [process id,thread id] [number] component trace message
data
```

where:

time
: The time the entry was written, to millisecond accuracy.

[*process id, thread id*]
: Process id is a unique number that the operating system uses to identify a process. Thread id is a unique number that the operating system uses to identify a thread within a particular process.

[*number*]
: A number that uniquely identifies the particular trace entry. This helps your support organization in the diagnosis of serious problems.

[*component*]
: The component of the product to which this entry applies.

trace message
: The trace message number and text.

data
: Some trace entries include a dump of key data blocks in addition to the trace message.

Sample Client daemon trace: Figure 7 shows the trace information recorded during the successful connection of a Client daemon to a CICS server using the TCP/IP protocol. The trace was generated using the commands:

```
cicscli -s=cicstcp -dcicscli
 -o
```

```
17:03:57.580 [0000080c,00000908] [1007] TRC:CCL1042 *** CICS Client for Windows v7 Service Level 00 - service trace started ***
17:03:57.590 [0000080c,00000908] [2183] CCL:CCL2048 Maximum trace data size set to 512
17:03:57.600 [0000080c,00000908] [2114] CCL:CCL2142 GetNextTimeout timeout is 0 seconds
17:03:58.612 [0000080c,00000908] [2030] CCL:CCL2106 Comms Event : LINK-UP
17:03:58.622 [0000080c,00000908] [2019] DRV:CCL2055 Connection with server established (linkID=1)
17:03:58.622 [0000080c,00000908] [2035] CCL:CCL2109 Send server TCS data
17:03:58.632 [0000080c,00000908] [3214] CCL:CCL3251 Comms Allocate request (LinkId=1, Tran='CCIN')
17:03:58.632 [0000080c,00000908] [3217] CCL:CCL3238 Comms Allocate completed (LinkId=1, ConvId=1, Rc=0)
17:03:58.632 [0000080c,00000908] [2127] CCL:CCL2143 CommsBegin - OK
17:03:58.632 [0000080c,00000908] [3100] CCL:CCL3113 CCIN install request: ApplId='*       ', Code page=819
...
...
...
17:04:00.625 [0000080c,00000908] [3102] CCL:CCL3114 CCIN install response: ApplId='@0Z8AAAA', Code page=8859-1, Rc=0
17:04:00.625 [0000080c,00000908] [3241] CCL:CCL3255 Comms Complete request (ConvId=1)
17:04:00.635 [0000080c,00000908] [3244] CCL:CCL3246 Comms Complete completed (ConvId=1, Rc=0)
17:04:00.635 [0000080c,00000908] [3218] CCL:CCL3252 Comms Deallocate request (ConvId=1)
17:04:00.645 [0000080c,00000908] [3221] CCL:CCL3239 Comms Deallocate completed (ConvId=1, Reason=0, Rc=0)
17:04:00.645 [0000080c,00000908] [2042] CCL:CCL2114 Processed TCS Reply - Server connected
17:04:00.645 [0000080c,00000908] [2114] CCL:CCL2142 GetNextTimeout timeout is 0 seconds
17:04:00.655 [0000080c,00000908] [2114] CCL:CCL2142 GetNextTimeout timeout is 0 seconds
17:04:00.664 [0000080c,00000908] [1004] TRC:CCL1043 *** Service trace ended ***
```

Figure 7. Sample Client daemon trace

Message ID
: **Explanation**

CCL1042
: Start of trace message. Each time a trace is started, a backup of the old trace file is created, and the trace file is overwritten. You can delete the file when required. Check the time stamp to ensure that you are reading the correct trace.

CCL2048
Maximum trace data size is at the default size of 512 bytes. You can modify this size by specifying the size value in the start command for the client trace; see the cicscli -d command.

CCL3251
The client sends a CCIN transaction to the server to install its connection definition on the server.

CCL3238
A conversation ID has been successfully allocated. If more than one conversation is active, use the conversation ID to distinguish between them.

CCL3113
The client sends a CCIN transaction to the server with Appl ID set to * to install its application. The Appl ID is specified in the configuration file as `Client=*`. This requests the server to dynamically generate an Appl ID that is unique within the CICS server system.

CCL3114
This shows the dynamically generated Appl ID.

CCL1043
End of trace message.

Figure 8 shows trace information recorded when we tried to connect to a CICS server over TCP/IP using an invalid port number. The port number specified in the configuration file file was not defined in the services file of the server. Hence, the connection could not be established.

```
16:16:41.562 [0000093c,000008ec] [1007] TRC:CCL1042 *** CICS Client for Windows v6 Service Level 00 - service trace started ***
16:16:41.572 [0000093c,000008ec] [2183] CCL:CCL2048 Maximum trace data size set to 512
16:16:41.582 [0000093c,000008ec] [2114] CCL:CCL2142 GetNextTimeout timeout is 0 seconds
16:16:41.612 [0000093c,000008ec] [2114] CCL:CCL2142 GetNextTimeout timeout is -1 seconds
16:16:41.622 [0000093c,000008ec] [3207] CCL:CCL3249 Comms Open request (Server=CICSTCP, Driver=CCLIBMIP)
16:16:41.622 [0000093c,000008ec] [4408] DRV:CCL4413 CCL4413 TCP/IP (to CICSTCP) address=192.113.36.200, port=1089, socket=3
16:16:41.622 [0000093c,000008ec] [3210] CCL:CCL3236 Comms Open completed (Server=CICSTCP, LinkId=1, Rc=0)
16:16:41.633 [0000093c,000008ec] [2114] CCL:CCL2142 GetNextTimeout timeout is 3660 seconds
16:16:41.633 [0000093c,000008ec] [1004] TRC:CCL1043 ***Service trace ended ***
```

Figure 8. Client daemon trace: using an invalid port number

Message ID
Explanation

CCL4413
Shows the port number used for this connection request.

You must check your definitions in the SIT on the server, the configuration file on the workstation, and the services file for the port number specified.

You must provide a valid port number or use the default value.

Diagnosing common problems

This section provides information to help you solve some common problems with the Client daemon. The problems are discussed under the following headings:

- “Starting clients and terminals” on page 89
- “TCP/IP communication problems” on page 89
- “The Client daemon stops responding” on page 90
- “CICS server problem determination” on page 92
- “Communication problem determination” on page 93

For information on error messages, refer to *CICS Transaction Gateway: Messages*.

Starting clients and terminals

The following are solutions to problems that can occur when starting clients and terminals:

A cicsterm request has gone to the wrong server

If you do not specify the -s=*servername* option on the cicsterm command, the Client daemon sends a cicsterm request to either of:

- the default server
- the first server listed in the configuration file, even if it is not yet activated.

The *servername* should be as specified in the configuration file.

Problems loading protocol drivers

The message ″Cannot load protocol driver″ in the cicscli.log file means that you are trying to use a protocol driver that does not exist. Check your configuration file file to make sure that the driver name is correct, and is supported in this release.

The Client daemon can connect to the server, but cicsterm cannot

In other words, cicscli -s=servername connects successfully, but cicsterm -s=servername does not. Check the following:

- Is the CTIN transaction defined on the server?
- Does cicsterm-a successfully install a terminal that is not sign-on capable? cicsterm attempts to install a sign-on capable terminal by default. If the -a option works, the server probably does not support sign-on capable terminals.

CICS servers require APAR fixes to support terminal sign-on capability; see “Supported software” on page 5. Refer to the CICS Universal Client README file for the latest details and check the PTFs for the CICS servers.

See “APARs and fixes” on page 99 for general information about APARs.

An XTERM session stops working after the CICS Universal Client is upgraded

The problem is caused because some files might have changed location between versions. The solution is to open a new XTERM window.

TCP/IP communication problems

The following are solutions to problems that can occur when communicating over TCP/IP:

Message CCL4404 TCP/IP (to ’CICSTCP’) unable to resolve name: RC=2

The CICS server, in this example CICSTCP, could not be resolved by the TCP/IP protocol driver. Ensure that your domain name server and router address information is correct, and that any names and IP addresses in the TCP/IP etc/hosts file are correct.

Use the TCP/IP `ping` or `nslookup` commands to see if TCP/IP can resolve the hostname.

CCIN not recognized, CTIN not recognized

The CCIN transaction installs your Client daemon definition on the CICS server. The CTIN transaction installs your client terminal definition on the CICS server. These transactions must be available on the CICS server if it supports the EPI, because the EPI implies CICS 3270 terminal emulation and CICS 3270 printer emulation. For information on which CICS servers support EPI and hence CICS 3270 emulation, see Table 1 on page 10. You can ignore these messages if your CICS server does not support the EPI.

A cicsterm command for a mainframe CICS server failed

cicsterm and cicsprnt use CICS 3270 emulation. Some mainframe CICS servers do not support CICS 3270 emulation. For information on which CICS servers support CICS 3270 emulation, see Table 1 on page 10.

APPC communication problems

The following are solutions to problems that can occur when communicating over APPC:

Error message CCL4678E received

Linux generates this message if environment variable LD_PRELOAD is not defined. Refer to “Configuring IBM Communications Server for Linux” on page 25 for configuration details. For security reasons LD_ variables are not inherited by the root user.

SNA error log: The SNA error log can help your support organization to diagnose problems. In a default installation, the log is located as follows:

AIX: `/var/sna/sna.err`

Linux on zSeries: `/var/opt/ibm/sna/sna.err`

Linux on POWER: `/var/opt/ibm/sna/sna.err`

Linux on Intel: `/var/opt/ibm/sna/sna.err`

Solaris: `/var/opt/sna/sna.err`

The user who starts the Client daemon should be a member of the **sna** group, otherwise the Client daemon cannot write to the SNA error log; in this situation the Communications Server for Linux on zSeries writes a message to /var/log/warn.

The Client daemon stops responding

This section discusses what to do if you think the Client daemon has stopped responding, for example because:
- API, ECI and EPI calls do not complete, or
- cicsterm hangs

Check that your CICS Transaction Server is working correctly, for example by looking at the CICS log. Insufficient storage on the CICS region may cause CICS Universal Client to hang.

To test whether CICS Universal Client has stopped responding, issue cicscli -l from the command line. If the call hangs then CICS Universal Client is not responding.

Try to reproduce the problem with tracing turned on. Take a client trace with as many components as possible active. As a minimum you need the API, DRV and CCL tracepoints; add TRN if possible. (Use wrapping trace if you do not want the trace file to get too large.) The summary trace shows whether the client or user application has stopped responding, and gives an indication as to whether the problem is in the client or server.

If you can reproduce the problem, have details of how to reproduce it available for your support organization. If you cannot reproduce the problem, have available details of the circumstances that led up to the hang, for example:

- Does it hang only when the system is under heavy load?
- Does it hang only when there are many concurrent users?
- Does it hang only under a particular configuration, or after a known sequence of events?

All UNIX and Linux systems

Run `ps -ef` and `ipcs -qa` on the system, piping the output to a file. This lists the following:

- Processes on the system
- All UNIX and Linux IPC queues
- Amount of data in the queues
- Owner of the queues

This information is useful because the client uses IPC queues for internal communication.

Check also that your message queues are correctly configured; see “Configuring message queues” on page 28.

AIX operating system

Use the System Management Interface Tool (SMIT) to take an operating system trace of the failure.

Use the `syscalls event set` and `ipcs` related trace options. Format it to show pids, tids, current system calls, and elapsed time options. Consider increasing the buffer file settings.

dbx tool

You can attach the dbx tool to a process that has locked, to find out what the process is doing. (Check that dbx is installed on your system, and that you have authority to connect to a process.) This example shows how to attach dbx to the Client daemon `cclclnt`. You can adapt the example to find out why a user application has locked.

Important: Attaching dbx to a process sometimes causes the process to lock or terminate. Use it only if you are sure the process has locked.

Type the following command to find out the process id (pid):

```
ps -ef | grep cclclnt
```

Information about the process is displayed:

```
    root 26864 27348   3 11:06:51  pts/2  0:00 grep cclclnt
bartfast 28266     1   0 11:06:46  pts/0  0:00 cclclnt
```

Using the information provided by the ps command, type this to attach dbx to the process:

```
dbx -a 28266 ./cclclnt
```

You should now see something like this:

```
Waiting to attach to process 28266 ...
Successfully attached to cclclnt.
Type 'help' for help.
reading symbolic information ...warning: no source compiled with -g

stopped in _pthread_ksleep at 0xd0139164 ($t2)
0xd0139164 (_pthread_ksleep+0x9c) 80410014        lwz   r2,0x14(r1)
```

To get a stack back trace of the current thread, issue the **where** command:

```
_pthread_ksleep(??, ??, ??, ??, ??) at 0xd0139164
_pthread_event_wait(??) at 0xd01395c0
_cond_wait_local(??, ??, ??) at 0xd0135494
_cond_wait(??, ??, ??) at 0xd0135998
pthread_cond_timedwait(??, ??, ??) at 0xd0136368
OsEventTimedWait() at 0xd00a78b4
.() at 0x100005b8
_pthread_body(??) at 0xd012f358
```

To list all threads in the cclclnt process, type `thread` :

```
 thread  state-k     wchan    state-u    k-tid   mode held scope function
 $t1     wait      0xc0000100 running    21793     k   no   sys
>$t2     run                  blocked    32425     k   no   sys  _pthread_ksleep
```

To make thread 1 the current thread, type `thread current 1`.

To get a stack back trace of thread 1, type `where`. The screen looks like this:

```
.() at 0x1000c784
.() at 0x10000ae0
.() at 0x10000ae0
.() at 0x100003f4
```

To close dbx, type `quit`.

Solaris

Run **truss** against the hung process; see the operating system documentation for relevant parameters. Specify options to follow forked calls, trace arguments to system calls, and show the environment strings to operating system calls.

Ensure that the `/etc/system` file allows sufficient queue entries; see "Configuring message queues" on page 28. If the value is too low, the client may freeze while waiting for a queue entry to become available.

Solaris system notes:

1. Whenever you change the `/etc/system` file, you must restart your system for the changes to take effect.
2. Changes to your system may not work if your workstation has less than 32MB of memory; the recommended minimum is 64MB.

CICS server problem determination

The most important facilities for problem determination on CICS servers are:

- Traces

  - auxiliary
  - internal
- Dumps
- CICS message logs
- Statistics information
- Monitoring information
- Execution diagnostic facility (EDF)
- CICS-supplied transactions, CEBR and CECI
- Independent software vendor (ISV) tools

Information about these facilities is given in the *Problem Determination* books for the individual products (see “CICS Transaction Server publications” on page 116). You may need to contact the server system administrator for more information, for example, about a CICS server error log.

The following shows the error message prefixes for CICS products:

CCL	CICS Universal Client
CTG	CICS Transaction Gateway
DFH	CICS on CICS on System/390® and z/OS
ERZ	CICS Transaction Server for Windows, and TXSeries
AEG	CICS Transaction Server for iSeries.

CPMI tasks lock in CICS Transaction Server for z/OS: If a CPMI (or equivalent mirror transaction) task locks in your CICS region, it is possibly because you have reached the MAXTASKS limit of your CICS region. This prevents the mirror program from sending data back to the Client daemon, which appears to hang. The problem typically occurs when CICS has a high number of concurrent transactions. A solution is to put the mirror transaction in a TranClass that has a MAXACTIVE lower than the MAXTASKS of your CICS region. All new requests to the CICS region are queued, and CICS can continue to process current requests. The value you specify for MAXACTIVE depends on your installation, and other tasks running in the region.

Communication problem determination

Even a small telecommunications network is a very complex system in which all components depend on one another. If one component fails and presents incorrect information to the other components, the latter may fail even more severely than the former. Sometimes the failure may be considerably delayed, and the error indicator may be lost before the error is detected. Thus, it is sometimes difficult to analyze a problem in the communication part of a system.

The Client daemon generates various messages associated with the use of the supported communication protocols and the associated products. Refer to *CICS Transaction Gateway: Messages* for a list of these messages and their explanations.

The communication products themselves generate error messages. For details of these, and information on troubleshooting, you should refer to the documentation for the communication product. The following sections summarize useful commands and utilities for problem determination.

TCP/IP-providing products: TCP/IP provides the following diagnostic tools:

arp Displays and modifies the IP-to-Ethernet or token ring physical address translation tables used by address resolution protocol (ARP).

hostname
Displays the host name of your workstation.

ifconfig
Displays all current TCP/IP network configuration values. This is helpful if you have to find out if an IP interface is active.

netstat
Displays protocol statistics and current TCP/IP network connections. This helps you obtain information about your own IP interfaces, for example, to list IP addresses and TCP/IP routing tables in use at your workstation.

nslookup
Displays information from Domain Name System (DNS) name servers.

ping Verifies connections to a remote computer or computers.

traceroute
Traces the TCP/IP path to a requested destination. It is useful for determining whether there is a problem with one of the intermediate nodes.

For more information on these utilities, refer to the online help for TCP/IP.

APPC-providing products: This section describes problem determination in products involved in APPC communication.

VTAM Buffer Trace: You can use VTAM buffer tracing to record the flow of data between logical units in the CICS environment. The trace entries include the netname of the terminal (logical unit) to which they relate. For details on VTAM buffer tracing and other VTAM problem determination functions, see the appropriate book in the VTAM library.

The APING utility: In an APPC environment, you can use the APING utility to test a connection. APING exchanges data packets with a partner computer over APPC and times how long the data transfer takes. It can be used to get a first estimate of the session setup time between two computers and the throughput and turnaround time on that APPC session. You can use APING to determine whether a session can be set up between two computers and to display extensive error information if session allocation fails. APING consists of two transaction programs: APING, which runs on the client side, and APINGD, which runs on the server side.

Checking client and host configuration: If the SNA product on the Client daemon machine, and the host are both correctly configured, you should be able to activate the connection between them. Follow the steps below to activate the connection; you do not need to start the client first.

1. Start the node on the SNA product on the Client daemon machine.
2. Use the CEMT transaction to acquire the connection. Type `CEMT I CON`. The screen will look similar to this:

```
 I CON
 STATUS:  RESULTS - OVERTYPE TO MODIFY
  Con(SIMD) Net(PYKS88BA)     Ins Rel Vta Appc
  Con(SYSB) Net(IYCKZCCV)     Ins Rel Vta Appc




                                                SYSID=SYSA APPLID=IYCKZCCU
  RESPONSE: NORMAL                         TIME:  12.25.04  DATE: 03.01.01
PF 1 HELP       3 END       5 VAR        7 SBH 8 SFH 9 MSG 10 SB 11 SF
```

3. Overtype Rel with `ACQ`, to acquire the connection.

If you cannot acquire the connection, SNA is wrongly configured on the product or the host.

IBM Communications Server for AIX and Linux: IBM Communications Server writes a log file as follows:

AIX systems
/var/sna/sna.err

Linux on zSeries systems
/var/opt/ibm/sna/sna.err

Linux on POWER systems
/var/opt/ibm/sna/sna.err

Linux on Intel systems
/var/opt/ibm/sna/sna.err

The IBM support organization needs the file if you have SNA errors that you cannot resolve. For information about sense codes, use the following management command:

```
sna -getsense sensecode
```

Mirror transaction does not time out

The default ECI mirror transaction (CPMI or CSMI) uses a PROFILE of DFHCICSA. This profile has an RTIMOUT value of NO. This means that if an ECI request is suspended in CICS during an extended unit of work, the mirror transaction never times out. To enable timeout in these situations:

1. Define a new mirror transaction with SPURGE=YES, and which specifies a PROFILE containing an RTIMOUT value other than NO.
2. Specify this new mirror transaction as the Transid in the ECIRequest using a call type of ECI_SYNC_TPN.

In these situations, if no response is received from the client after the timeout period has elapsed, CICS purges the mirror transaction and rolls back the associated unit of work.

Problems with Client applications

On Linux systems, your application might fail to start with a message like the following:

```
relocation error: ./ecib1: undefined symbol:
__6CclBufPCcQ26CclBuf12DataAreaType
```

This can occur with C++ applications compiled with GNU C/C++ compiler (gcc) 2.96. This compiler is not supported; recompile your application with a supported compiler listed at “Supported software” on page 5.

Telnet clients

If you run the CICS Universal Client **cicsterm** command from a Telnet prompt, certain Telnet clients can cause problems with the display. For example, your Telnet session might truncate message lines over a certain length. This is usually a problem with the Telnet client that you are using, or the terminal type that you are emulating. Currently there is no solution to this problem.

Problems running the Configuration Tool

If the Configuration Tool does not display all characters correctly, check the console where CICS Universal Client was invoked. If the operating system has issued warning messages about fonts, check that all fonts needed for code page for the current locale have been installed. Refer to the documentation supplied with your operating system and JRE for information on which font packages are required for which locales.

Program support

If you need to contact your IBM support organization, first refer to the Service and Support card supplied with the product for details of the support available to you, or visit our Web site at: www.ibm.com/software/cics/cuc and follow the **Support** link.

This section provides guidance on reporting problems, detailing the information you should collect to assist your support organization. It also explains how the problem is handled through to resolution and the provision of a fix.

Reporting problems

Before reporting your problem to the support organization try to ensure that the error is actually occurring within your CICS Universal Client system, even if you are unsure whether the problem is due to the CICS Universal Client itself.

In practice, many errors reported to support organizations turn out to be user errors, or they cannot be reproduced, indicating just how difficult it can be to determine the precise cause of a problem. User errors are mainly caused by faults in application programs or by errors in setting up systems.

The ability of the support organization to resolve your problem quickly depends on the quality of the information you provide to them. For this reason it is good practice to collect and organize problem data before making the initial contact.

Summarize your problem and the documentation/information you collect on a problem reporting sheet. This sheet provides a valuable summary to the support organization, and a copy will be useful for your own records.

Level-1 support

If this is your first contact regarding this problem a unique incident number will be assigned. A Problem Management Record (PMR) is opened on the RETAIN® database system, where all activity associated with your problem is recorded. The PMR remains "open" until it is solved.

Make a note of the incident number on your copy of the problem reporting sheet. You will be expected to quote the incident number in all future correspondence relating to this problem.

Your initial communication with the support organization is through a Level-1 representative, who uses the keywords that you have provided to search the RETAIN database for problems with similar symptoms. If your problem is found to be one already known to the support organization, and a fix has been devised for it, you will be advised of the availability of corrective service software that you can install to overcome the problem, as described in "Obtaining the fix" on page 99. If the RETAIN search is unsuccessful, the problem is passed on to a Level-2 representative who will contact you for more information about your problem, to start a more detailed investigation of the cause.

Level-2 support

Let the Level-2 representative know if any of the following events occurred before the problem appeared:

1. Changes in level of the CICS Universal Client, compiler or associated licensed programs
2. Corrective service software and fixes applied to workstation software
3. Problem Tracking Fixes (PTFs) applied to other associated products
4. Additional features used
5. Application programs changed
6. Unusual operator action.

At this stage you may be asked to supply more details or documentation. If this happens, the PMR is updated to show any documentation you have submitted.

Based on the information you supply, the investigation then carried out can determine whether the cause of your problem is new to the support organization, or is already known.

If the problem is a valid new one, an APAR may be raised, to be dealt with by the CICS service team, as described in "APARs and fixes" on page 99. If, however, the problem is already known, and a fix has been developed, you can obtain it as described in "Obtaining the fix" on page 99.

Documenting problems

In a communication environment, where many clients may be connected to several servers, you must obtain information from both the client and the server sides.

To facilitate problem determination, try to reduce your environment to only one client workstation and one server to narrow down the range of possibilities that may cause the error.

Listed below are the types of information most likely to be needed by the support organization. The list is not exhaustive; it covers the most commonly requested information.

- The version of the CICS Universal Client, including details of any service releases applied.
- The utility or programming language that gives the problem, for example:
  - ECI, EPI, ESI
  - C, C++

  For languages, include the version.
- The network protocols being used.

- The end-to-end configuration, and product stacks at each step of the way, for example, WebSphere Application Server async ECIRequest class over SSL to the Gateway daemon on Solaris using TCP/IP to CICS Transaction Server 2.3.
- The operating system on which the problem occurs. If the Client application and CICS Universal Client are on different operating systems, provide details of both. Details of the CICS server are also useful.
- A minimum recreate scenario, if possible.
- Details of any log messages at the time the problem occurred.
- A short description of the problem in terms of CICS Universal Client calls, including why you think it is a problem with the CICS Universal Client. For example, a specific API call hangs and never returns, or an unexpected return code on a specific API call.

Locating and compiling information

The advice listed below will help you compile the information required. If you are in any doubt about how to gather any of the items listed above, wait until you can seek advice from your support organization.

- Try to establish which program in your system seems to be the cause of the problem. As you are reading this book, it is likely that you already believe the CICS Universal Client to be the problem source.

  You also need to provide the version and release number of the product, for example Version 7.1, together with the number of any PTF or other fix applied, for example UN*nnnnn* (where *nnnnn* is a valid PTF number).
- You should find details of your compiler level on the product media labels or the associated documentation. Alternatively, check the panel displayed on the screen at compile time.
- Give the problem a severity level. Severity levels have the following meanings:
  - Severity level 1 indicates that you are unable to use a program, resulting in a critical condition that needs immediate attention.
  - Severity level 2 indicates that you are able to use the program, but that operation is severely restricted.
  - Severity level 3 indicates that you are able to use the program, with limited functions, but the problem is not critical to your overall operation.
  - Severity level 4 indicates that you are able to use the program with the problem causing negligible hindrance.
- Also, try to classify the error and give a brief description of the problem. Include keywords associated with the problem, such as ABEND, WAIT, LOOP, PERFORMANCE, INCORROUT, and MESSAGE, corresponding to the problem classification types used in the RETAIN database. Strings containing other keywords are also useful. These are not predefined, and might include such items as a message or message number, an abend code, any parameters known to be associated with the problem, or, for example, STARTUP, INITIALIZATION, or TRANSIENT DATA.
- Finally, you should include your address, relevant contact names, and details of the other products at your installation.

Keep a copy of the problem reporting sheet which summarizes all the information relevant to the problem, together with any available documentation such as dumps, traces and output from programs, translators, and compilers.

Submitting the documentation

If you are asked to provide documentation, it will help the support organization during their investigation if you adhere to the following rules when preparing it:

- Provide as much information as possible in softcopy format.
- Write any notes and documents in English.
- If you are asked to supply any additional trace output, provide an unformatted binary version, rather than the viewable output from the formatter.
- If you upload files to a mainframe system, do so in binary rather than ASCII format.

APARs and fixes

After a reported problem is confirmed as being both new and valid, an *authorized program analysis report* (APAR) is raised from the Problem Management Record (PMR) and becomes a permanent record describing your error, and its solution. An APAR is the means of reporting to the appropriate product service team any problems you find with an IBM program.

The APAR process

The first step in the APAR process is that a Level-2 representative from the support organization enters an APAR containing a description of your problem into the RETAIN system. If you have a means of getting round the problem, details of this are entered as well. Your name is also entered, so that your support organization knows who to contact if the service team need to ask anything further about the APAR documentation.

At this stage, you are given an APAR number. This number is always associated with the APAR and its resolution and, if a code change is required, is associated with the fix as well.

The service team may ask for additional backup documentation, normally via the Level-2 representative. The specific documentation required will vary from problem to problem, and depends on what information has already been supplied during the PMR stage.

During the investigation, you can ask your support organization at any time about how your APAR is progressing, particularly if it is a problem of high severity.

Obtaining the fix

When the service team has found a fix for your problem, you may be asked to test it on your system. If so, you may be sent a Program Temporary Fix (PTF) in the form of individual replacement modules. This is normally done through the Level-2 organization and feedback from your testing will be requested.

Each PTF may contain several APAR fixes. If an individual APAR fix within a PTF is found to be in error, it may still be advisable to apply the PTF, to obtain the other APAR fixes.

In time, formal corrective service software is made available, which you can order through your support organization. Traditionally, corrective service software has been supplied on Corrective Service Diskettes (CSD) ordered by CSD number. It may now be supplied on CD-ROM or via the Internet.

Fixes for CICS Universal Client are available from the CICS Support web page at:

`www.ibm.com/software/ts/cics/support/`

Chapter 11. Migration

This information deals with issues that can arise when you move from an earlier version of the CICS Universal Client, or a product related to the CICS Universal Client.

Migrating to Version 7.1

This information details any special action you need to take if migrating from an earlier version of the product.

Installation location

If you are upgrading from Version 5, Version 6 or Version 7, the product is installed into your existing directory structure, replacing the older version.

Removal of TCP62 support

The announcement of CICS Transaction Gateway for z/OS V6.1 indicated that the releases of CICS Transaction Gateway V7.0 and CICS Universal Client V7.0, would be the last releases that would contain TCP62 support for the AnyNet protocol communicating with remote CICS systems using SNA over TCP/IP protocol encapsulation. Accordingly, this capability has been removed from the V7.1 level of the products. For continued use of SNA over TCP/IP, it is necessary to migrate TCP62 server definitions to SNA and implement another IBM Communications Server TCP/IP protocol encapsulation solution, such as Enterprise Extender or Remote API client support.

For information on migrating from TCP62 to Enterprise Extender support refer to the IBM publication *Migrating an SNA connection from TCP62 to Enterprise Extender - GC34-6889-00*.

For information on the removal of AnyNet support from z/OS Communications Server, refer to the z/OS and z/OS.e statements of direction announcement, Software Announcement 203-266, dated October 7, 2003, and the z/OS V1.7 preview announcement, Software Announcement 205-034, dated February 15, 2005.

Removal of SNA Server configuration setting LUALIASNAMES

On Windows operating systems LUALIASNAMES has been replaced by LOCALLUALIAS and PARTNERLUALIAS.

On Unix and Linux operating systems LUALIASNAMES has been replaced by PARTNERLUALIAS.

After migration if LUALIASNAMES is found in the configuration file the Client daemon will attempt to start the SNA Server connection and will generate a warning in the Client daemon log file saying that the deprecated entry LUALIASNAMES exists.

MAXSERVERS parameter

The MAXSERVERS parameter, a setting in the CLIENT section of the configuration file for SNA and TCP/IP protocols is deprecated, and might be removed in a future release.

Support for mixed case passwords

CICS Transaction Server for z/OS version 2.2 and higher supports mixed-case passwords. To use mixed-case passwords, ensure that the “Use uppercase security” on page 42 field in the Configuration Tool is cleared.

Information log

Some Client information messages are now logged. You can configure a separate log for information messages; see “Changing settings for Client daemon logging” on page 40.

Migration considerations at Version 6

This information applies if you are migrating from Version 5 to Version 7.1.

Supported ECI and EPI versions

Applications can be built only with the following ECI and EPI versions:

ECI_VERSION_1A
CICS_EPI_VERSION_200

Existing applications built with the following versions are still supported at run time:

CICS_EPI_VERSION_100
CICS_EPI_VERSION_101
ECI_VERSION_1

Recompiling these applications will give a compiler error.

Change to maxrequests configuration setting

The maximum value permitted for this setting is now 32. If the configuration file specifies a value greater than this an error is logged, and the maximum value is used. See “Maximum requests” on page 37 for details of the configuration setting.

Configuration file: change of default name

The default configuration file is ctg.ini. In versions of the product before version 6 it was CTG.INI. The CICS Universal Client accepts either ctg.ini or CTG.INI. If both files are present the lowercase version is used; take one of the following steps:

- Specify the configuration file to be used; see “Using the Configuration Tool” on page 31 for information on how to do this.
- Delete file CTG.INI.

If you have specified the configuration file, for example by setting the CICSCLI environment variable, the exact file name specified is used.

User exits: change of file name

The user exits are now lower case:

Old name	New name
CICSEPIX	cicsepix
CICSEPIX	cicsepix

The CICS Universal Client first looks for a lowercase exit, and then for an uppercase exit. If the objects are not found, no exit processing occurs.

Linux C++ applications

Recompile any applications that have been compiled with GNU C/C++ compiler (gcc) 2.96 with a supported compiler; see "Supported software" on page 5 for supported compilers.

Appendix. Data conversion

Data conversion when using the Client daemon and a CICS server

The ECI and EPI allow non-CICS applications running in a client system to gain access to CICS facilities and data managed by a CICS server system.

Character data may have to be converted as it is passed between client and server; for example data is encoded in ASCII on a client system and in EBCDIC on a CICS mainframe server system. Data conversion is performed by the server system.

The possible ASCII and EBCDIC encoding schemes are described in detail in SC09-2190, the Character Data Representation Architecture Reference and Registry (CRDA). In summary, each encoding scheme is identified by a Coded Character Set Identifier (CCSID) which defines a set of graphic characters and the Code Page Global Identifier (CPGID) which specifies the code points used to represent the graphic characters.

The data managed by the server system may be accessed from several client systems each of which uses a different ASCII encoding scheme. To support such access, each client system must be able to supply a CCSID 'tag' in order that the data is converted correctly.

Supported conversions

The method used to perform data conversion depends on the server platform. The range of data conversions supported also depends on the platform. The following information is taken from *Communicating from CICS on System/390*, and is provided as a guide; check the current copy of the book for full information. ASCII and EBCDIC CCSIDs are assigned to geographic or linguistic groups.

Data conversion is supported between ASCII and EBCDIC where both CCSIDs belong to the same group. The groups are:

Arabic

Baltic Rim
Latvia, Lithuania

Cyrillic
Eastern Europe: Bulgaria, Russia, Yugoslavia

Estonian
Estonia

Greek
Greece

Hebrew
Israel

Japanese
Japan

Korean
Korea

Latin-1 and Latin-9
USA, Western Europe, and many other countries

Latin-2
Eastern Europe: Albania, Czech Republic, Hungary, Poland, Romania, Slovakia, Yugoslavia, Former Yugoslavia

Latin-5
Turkey

Simplified Chinese
People's Republic of China

Traditional Chinese
Taiwan

Vietnamese
Vietnam

The following tables list the CCSIDs supported for each group. For each CCSID, they show:

- The value to be specified for the CLINTCP or SRVERCP keyword.
- The code page identifier or identifiers (CPGIDs).
- The current CICS on System/390 products that support the CCSID. Three levels of support are defined—"Base", "T01", and "T02".

  Base
  - CICS Transaction Server for z/OS
  - CICS Transaction Server for OS/390® (all releases)
  - CICS/ESA® Version 4 Release 1
  - CICS/VSE
  - CICS/VSE Version 2 Release 3

  T01
  - CICS Transaction Server for z/OS
  - CICS Transaction Server for OS/390 Release 3
  - CICS Transaction Server for OS/390 Release 1 or Release 2 plus APAR PQ19018
  - CICS/ESA Version 4 Release 1 plus APAR PQ18896
  - CICS/VSE
  - CICS/VSE Version 2 Release 3 plus APAR PQ19019

  T02
  - CICS Transaction Server for z/OS

Arabic

Table 10. Arabic, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
864	Base	00864	00864	PC data: Arabic
1089 8859-6	Base	01089	01089	ISO 8859-6: Arabic
1256	T01	01256	01256	MS Windows: Arabic

Table 10. Arabic, Client CCSIDs (continued)

CLINTCP	in	CCSID	CPGID	Comments
5352	T02	05352	05352	MS Windows: Arabic, version 2 with euro
17248	T02	17248	00864	PC Data: Arabic with euro

Table 11. Arabic, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
420	Base	00420	00420	Host: Arabic
16804	T02	16804	00420	Host: Arabic with euro

Note: Data conversion does not change the direction of Arabic data.

Baltic Rim

Table 12. Baltic Rim, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
901	T02	00901	00901	PC data: Latvia, Lithuania; with euro
921	T01	00921	00921	PC data: Latvia, Lithuania
1257	T01	01257	01257	MS Windows: Baltic Rim
5353	T02	05353	05353	MS Windows: Baltic Rim, version 2 with euro

Table 13. Baltic Rim, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
1112	T01	01112	01112	Host: Latvia, Lithuania
1156	T02	01156	01156	Host: Latvia, Lithuania; with euro

Cyrillic

Table 14. Cyrillic, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
808	T02	00808	00808	PC data: Cyrillic, Russia; with euro
848	T02	00848	00848	PC data: Cyrillic, Ukraine; with euro
849	T02	00849	00849	PC data: Cyrillic, Belarus; with euro
855	Base	01235	00855	PC data: Cyrillic
866	Base	00866	00866	PC data: Cyrillic, Russia
872	T02	00872	00872	PC data: Cyrillic with euro
915 8859-5	Base	00915	00915	ISO 8859-5: Cyrillic
1124	T02	01124	01124	8-bit: Cyrillic, Belarus
1125	T02	01125	01125	PC Data: Cyrillic, Ukraine
1131	T02	01131	01131	PC Data: Cyrillic, Belarus
1251	T01	01251	01251	MS Windows: Cyrillic
5347	T02	05347	05347	MS Windows: Cyrillic, version 2 with euro

Table 15. Cyrillic, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
1025	Base	01025	01025	Host: Cyrillic multilingual
1123	T02	01123	01123	Host: Cyrillic Ukraine
1154	T02	01154	01154	Host: Cyrillic multilingual; with euro
1158	T02	01158	01158	Host: Cyrillic Ukraine; wtih euro

Estonian

Table 16. Estonian, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
902	T02	00902	00902	PC data: Estonia with euro
922	T01	00922	00922	PC data: Estonia
1257	T01	01257	01257	MS Windows: Baltic Rim
5353	T02	05353	05353	MS Windows: Baltic Rim, version2 with euro

Table 17. Estonian, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
1122	T01	01122	01122	Host: Estonia
1157	T01	01157	01157	Host: Estonia with euro

Greek

Table 18. Greek, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
813 8859-7	Base	00813	00813	ISO 8859-7: Greece
869	Base	00869	00869	PC data: Greece
1253	T01	01253	01253	MS Windows: Greece
4909	T02	04909	00813	ISO 8859-7: Greece with euro
5349	T02	05349	01253	MS Windows: Greece, version 2 with euro
9061	T02	09061	00869	PC Data: Greece with euro

Table 19. Greek, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
875	Base	00875	00875	Host: Greece
4971	T02	04971	00875	Host: Greece with euro

Hebrew

Table 20. Hebrew, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
856	Base	00856	00856	PC data: Hebrew

Table 20. Hebrew, Client CCSIDs (continued)

CLINTCP	in	CCSID	CPGID	Comments
862	T02	00862	00862	PC data: Hebrew (migration)
867	T02	00867	00867	PC Data: Hebrew with euro
916 8859-8	Base	00916	00916	ISO 8859-8: Hebrew
1255	T01	01255	01255	MS Windows: Hebrew
5351	T02	05351	05351	MS Windows: Hebrew, version 2 with euro

Table 21. Hebrew, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
424	Base	00424	00424	Host: Hebrew
803	T02	00803	00803	Host: Hebrew (Character Set A)
4899	T02	04899	00803	Host: Hebrew (Character Set A) with euro
12712	T02	12712	00424	Host: Hebrew with euro and new sheqel

Note: Data conversion does not change the direction of Hebrew data.

Japanese

Table 22. Japanese, ASCII

CLINTCP	in	CCSID	CPGID	Comments
932	Base	00932	1. 00897 2. 00301	1. PC data: SBCS 2. PC data: DBCS including 1880 user-defined characters
942	Base	00942	1. 01041 2. 00301	1. PC data: Extended SBCS 2. PC data: DBCS including 1880 user-defined characters
943	T01	00943	1. 00897 2. 00941	1. PC data: SBCS 2. PC data: DBCS for Open environment including 1880 IBM user-defined characters
954 EUCJP	Base	00954	1. 00895 2. 00952 3. 00896 4. 00953	1. G0: JIS X201 Roman 2. G1: JIS X208-1990 3. G1: JIS X201 Katakana 4. G1: JIS X212
5050	T02	05050	1. 00895 2. 00952 3. 00896 4. 00953	1. G0: JIS X201 Roman 2. G1: JIS X208-1990 3. G1: JIS X201 Katakana 4. G1: JIS X212

Table 23. Japanese, EBCDIC

SRVERCP		CCSID	CPGID	Comments
930	Base	00930	1. 00290 2. 00300 3. 00290 4. 00300	1. Katakana Host: extended SBCS 2. Kanji Host: DBCS including 4370 user-defined characters 3. Katakana Host: extended SBCS 4. Kanji Host: DBCS including 1880 user-defined characters

Table 23. Japanese, EBCDIC (continued)

SRVERCP		CCSID	CPGID	Comments
931	Base	00931	1. 00037 2. 00300	1. Latin Host: SBCS 2. Kanji Host: DBCS including 4370 user-defined characters
939	Base	00939	1. 01027 2. 00300 3. 01027 4. 00300	1. Latin Host: extended SBCS 2. Kanji Host: DBCS including 4370 user-defined characters 3. Latin Host: extended SBCS 4. Kanji Host: DBCS including 1880 user-defined characters
1390	T02	01390	1. 00290 2. 00300	1. Katakana Host: extended SBCS; with euro 2. Kanji Host: DBCS including 6205 user-defined characters
1399	T02	01399	1. 01027 2. 00300	1. Latin Host: extended SBCS; with euro 2. Kanji Host: DBCS including 4370 user-defined characters; with euro

Korean

Table 24. Korean, ASCII

CLINTCP	in	CCSID	CPGID	Comments
934	Base	00934	1. 00891 2. 00926	1. PC data: SBCS 2. PC data: DBCS including 1880 user-defined characters
944	Base	00944	1. 01040 2. 00926	1. PC data: Extended SBCS 2. PC data: DBCS including 1880 user-defined characters
949	Base	00949	1. 01088 2. 00951	1. IBM KS Code - PC data: SBCS 2. IBM KS code - PC data: DBCS including 1880 user-defined characters
970 EUCKR	Base	00970	1. 00367 2. 00971	1. G0: ASCII 2. G1: KSC X5601-1989 including 1880 user-defined characters
1363	T01	01363	1. 01126 2. 01362	1. PC data: MS Windows Korean SBCS 2. PC data: MS Windows Koran DBCS including 11172 full Hangul

Table 25. Korean, EBCDIC

SRVERCP	in	CCSID	CPGID	Comments
933	Base	00933	1. 00833 2. 00834	1. Host: Extended SBCS 2. Host: DBCS including 1880 user-defined characters and 11172 full Hangul characters
1364	T02	01364	1. 00833 2. 00834	1. Host: Extended SBCS 2. Host: DBCS including 1880 user-defined characters and 11172 full Hangul characters

Latin-1 and Latin-9

Table 26. Latin-1, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
437	Base	00437	00437	PC data: PC Base; USA, many other countries
819 8859-1	Base	00819	00819	ISO 8859-1: Latin-1 countries
850	Base	00850	00850	PC data: Latin-1 countries
858	T01	00858	00858	PC data: Latin-1 countries; with euro
923	T01	00923	00923	ISO 8859-15: Latin-9
924	T02	00924	00924	ISO 8859-15: Latin-9
1047	T02	01047	01047	Host: Open Edition Latin-1
1252	T01	01252	01252	MS Windows: Latin-1 countries
5348	T01	05348	01252	MS Windows: Latin-1 countries, version 2 with euro

Table 27. Latin-1 and Latin-9, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
037	Base	00037	00037	Host: USA, Canada (ESA), Netherlands, Portugal, Brazil, Australia, New Zealand
273	Base	00273	00273	Host: Austria, Germany
277	Base	00277	00277	Host: Denmark, Norway
278	Base	00278	00278	Host: Finland, Sweden
280	Base	00280	00280	Host: Italy
284	Base	00284	00284	Host: Spain, Latin America (Spanish)
285	Base	00285	00285	Host: United Kingdom
297	Base	00297	00297	Host: France
500	Base	00500	00500	Host: Belgium, Canada (AS/400®), Switzerland, International Latin-1
871	Base	00871	00871	Host: Iceland
924	T01	00924	00924	Host: Latin-9
1047	T01	01047	01047	Host: Open Edition Latin-1
1140	T01	01140	01140	Host: USA, Canada (ESA), Netherlands, Portugal, Brazil, Australia, New Zealand; with euro
1141	T01	01141	01141	Host: Austria, Germany; with euro
1142	T01	01142	01142	Host: Denmark, Norway; with euro
1143	T01	01143	01143	Host: Finland, Sweden; with euro
1144	T01	01144	01144	Host: Italy; with euro
1145	T01	01145	01145	Host: Spain, Latin America (Spanish); with euro
1146	T01	01146	01146	Host: United Kingdom; with euro
1147	T01	01147	01147	Host: France; with euro
1148	T01	01148	01148	Host: Belgium, Canada (AS/400), Switzerland, International Latin-1; with euro
1149	T01	01149	01149	Host: Iceland; with euro

Note: Conversions are supported between non euro-supported CCSIDs and euro-supported CCSIDs. These should be used with care because:

- The international currency symbol in each non euro-supported EBCDIC CCSID (for example, 00500) has been replaced by the euro symbol in the equivalent euro-supported EBCDIC CCSID (for example, 01148).
- The dotless *i* in non euro-supported ASCII CCSID 00850 has been replaced by the euro symbol in the equivalent euro-supported ASCII CCSID 00858.

Latin-2

Table 28. Latin-2, ASCII

CLINTCP	in	CCSID	CPGID	Comments
852	Base	00852	00852	PC data: Latin-2 multilingual
912 8859-2	Base	00912	00912	ISO 8859-2: Latin-2 multilingual
1250	T01	01250	01250	MS Windows: Latin-2
5346	T02	05346	01250	MS Windows: Latin-2, version 2 with euro
9044	T02	09044	00852	PC data: Latin-2 multilingual with euro

Table 29. Latin-2, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
500	T02	00500	00500	Host: International Latin-1
870	Base	00870	00870	Host: Latin-2 multilingual
1148	T02	01148	01148	Host: International Latin-1 with euro
1153	T02	01153	01153	Host: Latin-2 multilingual with euro

Note: Conversions are supported for some combinations of Latin-2 ASCII CCSIDs and Latin-1 EBCDIC CCSIDs.

Latin-5

Table 30. Latin-5, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
857	Base	00857	00857	PC data: Latin-5 (Turkey)
920 8859-9	Base	00920	00920	ISO 8859-9: Latin-5 (ECMA-128, Turkey TS-5881)
1254	T01	01254	01254	MS Windows: Turkey
5350	T02	05350	01254	MS Windows: Turkey, version 2 with euro
9049	T02	09049	00857	PC data: Latin-5 (Turkey) with euro

Table 31. Latin-5, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
1026	Base	01026	01026	Host: Latin-5 (Turkey)
1155	T02	01155	01155	Host: Latin-5 (Turkey) with euro

Simplified Chinese

Table 32. Simplified Chinese, ASCII

CLINTCP	in	CCSID	CPGID	Comments
946	Base	00946	1. 01042 2. 00928	1. PC data: Extended SBCS 2. PC data: DBCS including 1880 user-defined characters
1381	Base	01381	1. 01115 2. 01380	1. PC data: Extended SBCS (IBM GB) 2. PC data: DBCS (IBM GB) including 31 IBM-selected, 1880 user-defined characters
1383 EUCCN	T01	01383	1. 00367 2. 01382	1. G0: ASCII 2. G1: GB 2312-80 set
1386	T01	01386	1. 01114 2. 01385	1. PC data: S-Chinese GBK and T-Chinese IBM BIG-5 2. PC data: S-Chinese GBK

Table 33. Simplified Chinese, EBCDIC

SRVERCP	in	CCSID	CPGID	Comments
935	Base	00935	1. 00836 2. 00837	1. Host: Extended SBCS 2. Host: DBCS including 1880 user-defined characters
1388	T02	01388	1. 00836 2. 00837	1. Host: Extended SBCS 2. Host: DBCS including 1880 user-defined characters
9127	T02	09127	1. 00836 2. 00837	1. Host: Extended SBCS 2. Host: DBCS including 1880 user-defined characters

Traditional Chinese

Table 34. Traditional Chinese, ASCII

CLINTCP	in	CCSID	CPGID	Comments
938	Base	00938	1. 00904 2. 00927	1. PC data: SBCS 2. PC data: DBCS including 6204 user-defined characters
948	Base	00948	1. 01043 2. 00927	1. PC data: Extended SBCS 2. PC data: DBCS including 6204 user-defined characters
950 BIG5	Base	00950	1. 01114 2. 00947	1. PC data: SBCS (IBM BIG5) 2. PC data: DBCS including 13493 CNS, 566 IBM selected, 6204 user-defined characters
964 EUCTW	Base	00964	1. 00367 2. 00960 3. 00961	1. G0: ASCII 2. G1: CNS 11643 plane 1 3. G1: CNS 11643 plane 2
1370	T02	01370	1. 01114 2. 00947	1. PC data: Extended SBCS; with euro 2. PC data: DBCS including 6204 user-defined characters; with euro

Table 35. Traditional Chinese, EBCDIC

SRVERCP	in	CCSID	CPGID	Comments
937	Base	00937	1. 00037 2. 00835	1. Host: Extended SBCS 2. Host: DBCS including 6204 user-defined characters
1371	T02	01371	1. 01159 2. 00835	1. Host: Extended SBCS; with euro 2. Host: DBCS including 6204 user-defined characters; with euro

Vietnamese

Table 36. Vietnamese, Client CCSIDs

CLINTCP	in	CCSID	CPGID	Comments
1129	T02	01129	01129	ISO-8: Vietnamese
1163	T02	01163	01163	ISO-8: Vietnamese with euro
1258	T02	01258	01258	MS Windows: Vietnamese
5354	T02	05354	01258	MS Windows: Vietnamese, version 2 with euro

Table 37. Vietnamese, Server CCSIDs

SRVERCP	in	CCSID	CPGID	Comments
1130	T02	01130	01130	Host: Vietnamese
1164	T02	01164	01164	Host: Vietnamese with euro

Unicode data

CICS on System/390 provides limited support for Unicode-encoded character data. The support allows workstations to share UCS-2 or UTF-8 encoded data with the System/390 provided, that no conversion is required.

Table 38. Unicode

CLINTCP SRVERCP	in	CCSID	CPGID	Comments
1200 UCS-2	T01	01200	01400	UCS-2 level 3, maximal (growing) character set
1208 UTF-8	T01	01200	01400	UTF-8 based on UCS-2 level 3, maximal (growing) character set
13488	T01	13488	01400	UCS-2 level 1 (level 3 tolerant), subset (fixed) character set

The product library and related literature

This information lists books on the CICS Universal Client, and related topics.

CICS Universal Client books

- *CICS Universal Client: UNIX and Linux Administration*, SC34-6962-00

  This book describes the administration of the CICS Universal Client for the Linux operating system.
- *CICS Transaction Gateway: Messages*, SC34-6964-00

  This online book lists and explains the error messages that can be generated by the CICS Transaction Gateway or the CICS Universal Client.
- *CICS Transaction Gateway: Programming Reference*, SC34-6966-00

  This book provides information on the APIs of the programming languages supported by the CICS Transaction Gateway and the CICS Universal Client.
- *CICS Transaction Gateway: Programming Guide*, SC34-6965-00

  This introduction to programming for the CICS Transaction Gateway and the CICS Universal Client provides the information that you need to allow user applications to use CICS facilities in a client/server environment.

Sample configuration documents

Several sample configuration documents are available in portable document format (PDF). These documents give step-by-step guidance for configuring CICS Transaction Gateway for communication with CICS servers, using various protocols. They provide detailed instructions that extend the information in the CICS Transaction Gateway library. Although written for the CICS Transaction Gateway, they also contain information relevant to the CICS Universal Client.

Visit the following Web site:

`www.ibm.com/software/cics/cuc`

and follow the **Library** link.

Redbooks

The following International Technical Support Organization (ITSO) Redbook publication contains many examples of client/server configurations. Although these were written for the CICS Transaction Gateway they also contain information applicable to the CICS Universal Client.

- *CICS Transaction Gateway V5 - The WebSphere Connector for CICS, SG24-6133*
- *Revealed! Architecting Web Access to CICS, SG24-5466*
- *Enterprise JavaBeans for z/OS and OS/390 CICS Transaction Server V2.2, SG24-6284*
- *Java Connectors for CICS: Featuring the J2EE Connector Architecture, SG24-6401*. This book provides information on developing J2EE applications.
- *Systems Programmer's Guide to Resource Recovery Services (RRS), SG24-6980-00*. This book provides information on using RRS in various scenarios.

- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide, SG24-6517-00*. This book provides information on using Communications Server for z/OS V1R2, including load balancing.
- *Redpaper: Transactions in J2EE, REDP-3659-00*. This redpaper provides a discussion of transactions in the J2EE environment, including one- and XA transactions.

You can obtain ITSO Redbooks® from a number of sources. For the latest information, see:

`www.ibm.com/redbooks/`

Other Useful Books

CICS Transaction Server publications

CICS Transaction Server for z/OS RACF Security Guide, SC34-6249

CICS interproduct communication

The following books describe the intercommunication facilities of the CICS server products:

- *CICS Family: Interproduct Communication*, SC34-6267
- *CICS Transaction Server for Windows V5.0 Intercommunication*, SC34-6209
- *CICS Transaction Server for z/OS CICS External Interfaces Guide*, SC34-6449
- *CICS Transaction Server for z/OS: Intercommunication Guide*, SC34-6448
- *CICS/VSE 2.3: Intercommunication Guide*, SC33-0701
- *CICS Transaction Server for iSeries V5R2: Intercommunication*, SC41-5456
- *TXSeries 5.1: CICS Intercommunication Guide*, SC09-4462

The first book above is a CICS family book containing a platform-independent overview of CICS interproduct communication.

CICS problem determination books

The following books describe the problem determination facilities of the CICS server products:

- *Transaction Server for Windows V5.0: Problem Determination*, GC34-6210
- *CICS Transaction Server for z/OS V3.1 CICS Problem Determination Guide*, SC34-6441
- *CICS/VSE 2.3 Problem Determination Guide*, SC33-0716
- *CICS Transaction Server for iSeries V5R2: Problem Determination*, SC41-5453
- *TXSeries V5.1: CICS Problem Determination Guide*, SC09-4465

You can find information on CICS products at the following Web site:

`www.ibm.com/software/cics/cuc`

Obtaining books from IBM

For information on books you can download, visit our Web site at:

`www.ibm.com/software/cics/cuc`

and follow the **Library** link.

Accessibility features for CICS Universal Client

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products successfully. The CICS Universal Client supports keyboard-only operation. Topics on the following pages give details of accessibility features.

Visit the IBM Accessibility Center for more information about IBM's commitment to accessibility.

Installation

The InstallShield wizard is not fully accessible to screen readers. An alternative is to use the -console option; see "Installing from the console" on page 14.

Documentation

See the Eclipse information center for an HTML version of the documentation.

Configuration Tool accessibility

See "Editing the configuration file" on page 47for information on how to configure the CICS Universal Client by editing the configuration file. This is the recommended way if you use a screen reader. f

The configuration file uses the number sign (#) character to denote a comment; consider configuring your screen reader accordingly.

Components

Each component in the Configuration Tool has a name and description for screen readers.

Keys

You can use the following keys to operate the Configuration Tool:

Alt, Space
Press and release the **Alt** key, then press **Space**, to open the window menu. This allows you to move, size, maximize or minimize the window.

Ctrl+key
Certain actions have shortcuts assigned to them. Hold down the `Ctrl` key and type the letter to do the action.

Action	**Ctrl+key**
Activate a link.	Spacebar
Copy the selected value	C except in the following locales: **Locale** **Key** **German** K **Turkish** P

Action	Ctrl+key
Create a new configuration	N
Create a new server definition	W except in the following locales: **Locale** **Key** **German** N **Italian** V **Spanish** V **Turkish** S
Cut the selected value	U except in the following locales: **Locale** **Key** **Italian** G **Spanish** O **Turkish** E
Cycle between the navigation panel, objects on the settings panel, and the buttons	Use this key combination on panels that contain a table. Otherwise use **Tab**.
Next link or other focusable object	T
Open an existing configuration	O except in the following locales: **Locale** **Key** **German** F **Italian** A **Spanish** B **Turkish** A
Paste	P except in the following locales: **Locale** **Key** **French** L **German** F **Italian** I **Turkish** P
Previous link or other focusable object	Shift+T
Save the current configuration	S except in the following locales: **Locale** **Key** **Spanish** G **Italian** L

Action	Ctrl+key
Scroll left	Page Up
Scroll right	Page Down

Arrows (left and right)

1. (Applies if the menus are active.) Move to a different menu.
2. (Applies if the buttons are active.) Cycle through the buttons.
3. (Applies if the navigation panel is active.) If a node contains subnodes, the left arrow collapses the node, the right arrow expands it. If a node cannot be expanded further, pressing the right arrow moves down to the next node. If a subnode is selected, pressing the left arrow moves to the parent node.

Arrows (up and down)

1. (Applies if the navigation panel is active.) Move up and down through the navigation panel.
2. (Applies if the buttons are active.) Cycle through the buttons.

Escape
Closes the menu.

F2 Select and change an editable number field in a table.

F6 (Applies if the navigation panel or the settings panel is active.) Toggles between the navigation panel and the settings panel.

F8 (Applies if the navigation panel or the settings panel is active.) Allows you to gain control of the split between the navigation panel and settings panel areas. After pressing F8, use the arrow keys to move the split:

- Press the left arrow key to move the split left (decrease the width of the navigation panel, increase the width of the settings panel).
- Press the right arrow key to move the split right (increase the width of the navigation panel, decrease the width of the settings panel).

F10
Opens the file menu.

Page Up
Scroll up.

Page Down
Scroll down.

Scroll bars
You cannot use the keyboard to control scroll bars in the navigation panel. Use the up and down arrow keys to navigate the tree; settings for the selected item are shown in the settings panel.

Shift+Tab
If the cursor is on a field in the settings panel other than the first field, moves backwards through the fields. Otherwise, toggles between the navigation panel and the settings panel.

Space

1. Activates a button.
2. Selects a check box.
3. Selects a node in the tree.

Tab
Cycles through the tree, fields in the settings panel, and the buttons.

Customizing colors and fonts

Use the **Settings** option on the menu to change colors and fonts.

Fields that contain errors are displayed by default in the font warning color, preceded by an asterisk. Use the **Settings->Font** option on the menu to change the warning color.

cicsterm

Although cicsterm is accessible, it relies on the application that is being processed to define an accessible 3270 screen. Keyboard mapping depends on the terminal type that you are using; see "Keyboard mapping in cicsterm" on page 73 for details.

The bottom row of cicsterm contains status information. The following list shows this information, as it appears from left to right:

Status
For example, **1B** is displayed while cicsterm is connecting to a server. Displayed at columns 1 – 3.

Terminal name
Also referred to as *LU Name*. Columns 4 – 7.

Action
For example, **X-System**, indicating that you cannot enter text in the terminal window because cicsterm is waiting for a response from the server. Columns 9 – 16.

Error number
Errors in the form CCLNNNN, relating to the CICS Universal Client. Columns 17 – 24.

Server name
The server to which cicsterm is connected. Columns 27 – 35.

Upper case
An up arrow is displayed when the Shift key is pressed. Column 42.

Caps Lock
A capital A is displayed when Caps Lock is on. Column 43.

Insert on
The caret symbol (^) is displayed if text will be inserted, rather than overwriting existing text. If you have difficulty seeing the caret, change the font face and size, or use a screen magnifier to increase the size of the status line. Column 52.

Cursor position
The cursor position, in the form ROW/COLUMN, where ROW is a two-digit number, and COLUMN a three-digit number. The top left of the screen is 01/001. Column 75–80.

Note: You might need to change the default behavior of your screen reader if it reads only the last digit of the cursor position. Customize your screen reader to specify that columns 75–80 of the status row are to be treated as one field. This will cause the full area to be read when any digit changes.

Glossary

This glossary defines special terms used in the CICS Universal Client library.

3270 emulation
The use of software that enables a client to emulate an IBM 3270 display station or printer, and to use the functions of an IBM host system.

abnormal end of task (abend)
The termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve.

Advanced program-to-program communication (APPC)
An implementation of the SNA/SDLC LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs. The Client daemon uses APPC to communicate with CICS server systems.

APAR See *Authorized program analysis report*.

API Application programming interface.

applet A small application program that performs a specific task and is usually portable between operating systems. Often written in Java, applets can be downloaded from the Internet and run in a Web browser.

application identifier
The name by which a CICS system is known in a network of interconnected CICS systems. CICS Transaction Gateway application identifiers do not need to be defined in SYS1.VTAMLST. The CICS APPLID is specified in the APPLID system initialization parameter.

application programming interface (API)
A functional interface that allows an application program that is written in a high-level language to use specific data or functions of the operating system or another program.

APPLID
See *application identifier*.

ARM See *automatic restart management*.

Authorized program analysis report (APAR)
A request for correction of a defect in a current release of an IBM-supplied program.

ATI See *automatic transaction initiation*.

attach In SNA, the request unit that flows on a session to initiate a conversation.

Attach Manager
The component of APPC that matches attaches received from remote computers to accepts issued by local programs.

autoinstall
A method of creating and installing resources dynamically as terminals log on, and deleting them at logoff.

automatic restart manager

A z/OS recovery function that can improve the availability of specific batch jobs or started tasks, and therefore result in faster resumption of productive work. Acronym: ARM.

automatic transaction initiation (ATI)

The initiation of a CICS transaction by an internally generated request, for example, the issue of an EXEC CICS START command or the reaching of a transient data trigger level. CICS resource definition can associate a trigger level and a transaction with a transient data destination. When the number of records written to the destination reaches the trigger level, the specified transaction is automatically initiated.

bean

A definition or instance of a JavaBeans™ component. See also *JavaBeans*.

BIND command

In SNA, a request to activate a session between two logical units (LUs).

business logic

The part of a distributed application that is concerned with the application logic rather than the user interface of the application. Compare with *presentation logic*.

CA

See *certificate authority*.

callback

A way for one thread to notify another application thread that an event has happened.

certificate authority

In computer security, an organization that issues certificates. The certificate authority authenticates the certificate owner’s identity and the services that the owner is authorized to use. It issues new certificates and revokes certificates from users who are no longer authorized to use them.

change-number-of-sessions (CNOS)

An internal transaction program that regulates the number of parallel sessions between the partner LUs with specific characteristics.

CICS connectivity components

A generic reference to the Client daemon, EXCI, and the IPIC protocol.

CICS on System/390

A generic reference to the products CICS Transaction Server for z/OS, CICS for MVS/ESA™, CICS Transaction Server for VSE/ESA, and CICS/VSE.

CICS TS

Abbreviation of CICS Transaction Server.

class

In object-oriented programming, a model or template that can be instantiated to create objects with a common definition and therefore, common properties, operations, and behavior. An object is an instance of a class.

classpath

In the execution environment, an environment variable keyword that specifies the directories in which to look for class and resource files.

Client API

The Client API is the interface used by Client applications to invoke services in CICS using the Client daemon. See External Call Interface, External Presentation Interface, and External Security Interface.

Client application
The client application is a user application written in a supported programming language, other than Java, that uses the Client API.

Client daemon
The Client daemon, process cclclnt, exists only on UNIX, Windows, and Linux. It manages network connections to CICS servers. It processes ECI, EPI, and ESI requests, sending and receiving the appropriate flows from the CICS server to satisfy the application requests. It uses the CLIENT section of ctg.ini for its configuration.

client/server
Pertaining to the model of interaction in distributed data processing in which a program on one computer sends a request to a program on another computer and awaits a response. The requesting program is called a client; the answering program is called a server.

CNOS See *Change-Number-of-Sessions*.

code page
An assignment of hexadecimal identifiers (code points) to graphic characters. Within a given code page, a code point can have only one meaning.

color mapping file
A file that is used to customize the 3270 screen color attributes on client workstations.

communication area (COMMAREA)
A communication area that is used for passing data both between programs within a transaction and between transactions.

configuration file
A file that specifies the characteristics of a program, system device, server or network.

connection
In data communication, an association established between functional units for conveying information.

In Open Systems Interconnection architecture, an association established by a given layer between two or more entities of the next higher layer for the purpose of data transfer.

In TCP/IP, the path between two protocol application that provides reliable data stream delivery service.

In Internet, a connection extends from a TCP application on one system to a TCP application on another system.

control table
In CICS, a storage area used to describe or define the configuration or operation of the system.

conversation
A connection between two programs over a session that allows them to communicate with each other while processing a transaction.

conversation security
In APPC, a process that allows validation of a user ID or group ID and password before establishing a connection.

daemon
A program that runs unattended to perform continuous or periodic systemwide functions, such as network control. A daemon may be launched automatically, such as when the operating system is started, or manually.

data link control (DLC)
A set of rules used by nodes on a data link (such as an SDLC link or a token ring) to accomplish an orderly exchange of information.

DBCS See *double-byte character set*.

dependent logical unit
A logical unit that requires assistance from a system services control point (SSCP) to instantiate an LU-to-LU session.

deprecated
Pertaining to an entity, such as a programming element or feature, that is supported but no longer recommended, and that might become obsolete.

digital certificate
An electronic document used to identify an individual, server, company, or some other entity, and to associate a public key with the entity. A digital certificate is issued by a certificate authority and is digitally signed by that authority.

digital signature
Information that is encrypted with an entity's private key and is appended to a message to assure the recipient of the authenticity and integrity of the message. The digital signature proves that the message was signed by the entity that owns, or has access to, the private key or shared secret symmetric key.

distributed application
An application for which the component application programs are distributed between two or more interconnected processors.

distributed processing
The processing of different parts of the same application in different systems, on one or more processors.

distributed program link (DPL)
A link that enables an application program running on one CICS system to link to another application program running in another CICS system.

domain
In the Internet, a part of a naming hierarchy in which the domain name consists of a sequence of names (labels) separated by periods (dots).

domain name
In TCP/IP, a name of a host system in a network.

domain name server
In TCP/IP, a server program that supplies name-to-address translation by mapping domain names to internet addresses. Synonymous with name server.

dotted decimal notation
The syntactical representation for a 32-bit integer that consists of four 8-bit numbers written in base 10 with periods (dots) separating them. It is used to represent IP addresses.

double-byte character set (DBCS)
A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires 2 bytes, the typing, display, and printing of DBCS characters requires hardware and programs that support DBCS. Contrast with *single-byte character set.*

DPL See *distributed program link*.

EBCDIC
See *Extended binary-coded decimal interchange code*.

ECI See *external call interface*.

EJB See *Enterprise JavaBeans*.

emulation program
A program that allows a host system to communicate with a workstation in the same way as it would with the emulated terminal.

emulator
A program that causes a computer to act as a workstation attached to another system.

encryption
The process of transforming data into an unintelligible form in such a way that the original data can be obtained only by using a decryption process.

enterprise bean
A Java component that can be combined with other resources to create J2EE applications. There are three types of enterprise beans: entity beans, session beans, and message-driven beans.

Enterprise JavaBeans
A component architecture defined by Sun Microsystems for the development and deployment of object-oriented, distributed, enterprise-level applications (J2EE).

environment variable
A variable that specifies the operating environment for a process. For example, environment variables can describe the home directory, the command search path, the terminal in use, and the current time zone.

EPI See *external presentation interface*.

ESI See *external security interface*.

Ethernet
A local area network that allows multiple stations to access the transmission medium at will without prior coordination, avoids contention by using carrier sense and deference, and resolves contention by using collision detection and transmission. Ethernet uses carrier sense multiple access with collision detection (CSMA/CD).

EXCI See *External CICS Interface*.

external call interface (ECI)
A facility that allows a non-CICS program to run a CICS program. Data is exchanged in a COMMAREA as for normal CICS interprogram communication.

Extended binary-coded decimal interchange code (EBCDIC)
A coded character set of 256 8-bit characters developed for the representation of textual data.

extended logical unit of work (extended LUW)
A logical unit of work that is extended across successive ECI requests to the same CICS server.

External CICS Interface (EXCI)
The EXCI is an MVS™ application programming interface provided by CICS Transaction Server for z/OS that enables a non-CICS program to call a CICS program and to pass and receive data using a COMMAREA or container. The CICS application program is invoked as if linked-to by another CICS application program.

external presentation interface (EPI)
A facility that allows a non-CICS program to appear to CICS as one or more standard 3270 terminals. 3270 data can be presented to the user by emulating a 3270 terminal or by using a graphical user interface.

external security interface (ESI)
A facility that enables client applications to verify and change passwords for user IDs on CICS servers.

firewall
A configuration of software that prevents unauthorized traffic between a trusted network and an untrusted network.

gateway
A device or program used to connect two systems or networks.

host A computer that is connected to a network (such as the Internet or an SNA network) and provides an access point to that network. The host can be any system; it does not have to be a mainframe.

host address
An IP address that is used to identify a host on a network.

host ID
In TCP/IP, that part of the Internet address that defines the host on the network. The length of the host ID depends on the type of network or network class (A, B, or C).

host name
In the Internet suite of protocols, the name given to a computer. Sometimes, host name is used to mean the fully qualified domain name; other times, it is used to mean the most specific subname of a fully qualified domain name. For example, if mycomputer.city.company.com is the fully qualified domain name, either of the following may be considered the host name: mycomputer.city.company.com, mycomputer.

hover help
Information that can be viewed by holding a mouse over an item such as an icon in the user interface.

HTTP See *Hypertext Transfer Protocol*.

HTTPS
See *Hypertext Transfer Protocol Secure*.

Hypertext Transfer Protocol
In the Internet suite of protocols, the protocol that is used to transfer and display hypertext and XML documents.

Hypertext Transfer Protocol Secure
A TCP/IP protocol that is used by World Wide Web servers and Web browsers to transfer and display hypermedia documents securely across the Internet.

iKeyman
A tool for maintaining digital certificates for JSSE.

independent logical unit
A logical unit (LU) that can both send and receive a BIND, and which supports single, parallel, and multiple sessions. See *BIND*.

Internet Architecture Board
The technical body that oversees the development of the internet suite of protocols known as TCP/IP.

Internet Protocol (IP)
In TCP/IP, a protocol that routes data from its source to its destination in an Internet environment.

interoperability
The capability to communicate, execute programs, or transfer data among various functional units in a way that requires the user to have little or no knowledge of the unique characteristics of those units.

IP Internet Protocol.

IP address
A unique address for a device or logical unit on a network that uses the IP standard.

J2EE See *Java 2 Platform Enterprise Edition*

J2EE Connector architecture (JCA)
A standard architecture for connecting the J2EE platform to heterogeneous enterprise information systems (EIS).

Java An object-oriented programming language for portable interpretive code that supports interaction among remote objects.

Java 2 Platform Enterprise Edition (J2EE)
An environment for developing and deploying enterprise applications, defined by Sun Microsystems Inc. The J2EE platform consists of a set of services, application programming interfaces (APIs), and protocols that allow multitiered, Web-based applications to be developed.

JavaBeans
As defined for Java by Sun Microsystems, a portable, platform-independent, reusable component model.

Java Client application
The Java client application is a user application written in Java, including servlets and enterprise beans, that uses the Gateway classes.

Java Development Kit (JDK)
The name of the software development kit that Sun Microsystems provided for the Java platform, up to and including v 1.1.x. Sometimes used erroneously to mean the Java platform or as a generic term for any software developer kits for Java.

Java Native Interface (JNI)
A programming interface that allows Java code running in a Java virtual machine to work with functions that are written in other programming languages.

Java Runtime Environment (JRE)
A subset of the Java Software Development Kit (SDK) that supports the execution, but not the development, of Java applications. The JRE comprises the Java Virtual Machine (JVM), the core classes, and supporting files.

Java Secure Socket Extension (JSSE)
A Java package that enables secure Internet communications. It implements a Java version of the Secure Sockets Layer (SSL) and Transport Layer Security (TSL) protocols and supports data encryption, server authentication, message integrity, and optionally client authentication.

Java virtual machine (JVM)
A software implementation of a processor that runs compiled Java code (applets and applications).

JDK See *Java development kit (JDK)*.

JCA See *J2EE Connector Architecture (JCA)*.

JNI See *Java Native Interface (JNI)*.

JRE See *Java Runtime Environment*

JSSE See *Java Secure Socket Extension (JSSE)*.

JVM See *Java Virtual Machine (JVM)*.

keyboard mapping
A list that establishes a correspondence between keys on the keyboard and characters displayed on a display screen, or action taken by a program, when that key is pressed.

key ring
In the JSSE protocol, a file that contains public keys, private keys, trusted roots, and certificates.

logical unit (LU)
In SNA, a port through which an end user accesses the SNA network in order to communicate with another end user and through which the end user accesses the functions provided by system services control points (SSCP). An LU can support at least two sessions, one with an SSCP and one with another LU, and may be capable of supporting many sessions with other logical units. See *network addressable unit, primary logical unit, secondary logical unit*.

logical unit 6.2 (LU 6.2)
A type of logical unit that supports general communications between programs in a distributed processing environment.

The LU type that supports sessions between two applications using APPC.

logical unit of work (LUW)
A recoverable unit of work performed within CICS.

LU-LU session
In SNA, a session between two logical units (LUs) in an SNA network. It provides communication between two end users, or between an end user and an LU services component.

LU-LU session type 6.2
In SNA, a type of session for communication between peer systems. Synonymous with APPC protocol.

LUW See *logical unit of work*.

managed mode
Describes an environment in which connections are obtained from connection factories that the J2EE server has set up. Such connections are owned by the J2EE server.

medium access control (MAC) sublayer
One of two sublayers of the ISO Open Systems Interconnection data link layer proposed for local area networks by the IEEE Project 802 Committee on Local Area Networks and the European Computer Manufacturers Association (ECMA). It provides functions that depend on the topology of the network and uses services of the physical layer to provide services to the logical link control (LLC) sublayer. The OSI data link layer corresponds to the SNA data link control layer.

method
In object-oriented programming, an operation that an object can perform. An object can have many methods.

mode In SNA, a set of parameters that defines the characteristics of a session between two LUs.

name server
In TCP/IP, synonym for Domain Name Server. In Internet communications, a host that translates symbolic names assigned to networks and hosts into Internet addresses.

network address
In SNA, an address, consisting of subarea and element fields, that identifies a link, link station, or network addressable unit (NAU). Subarea nodes use network addresses; peripheral nodes use local addresses. The boundary function in the subarea node to which a peripheral node is attached transforms local addresses to network addresses and vice versa. See also *network name*.

network addressable unit (NAU)
In SNA, a logical unit, a physical unit, or a system services control point. The NAU is the origin or the destination of information transmitted by the path control network. See also *logical unit, network address, network name*.

network name
In SNA, the symbolic identifier by which end users refer to a network addressable unit (NAU), link station, or link. See also *network address*.

node type
In SNA, a designation of a node according to the protocols it supports and the network addressable units (NAUs) it can contain. Four types are defined: 1, 2, 4, and 5. Type 1 and type 2 nodes are peripheral nodes; type 4 and type 5 nodes are subarea nodes.

nonmanaged mode
An environment in which the application is responsible for generating and configuring connection factories. The J2EE server does not own or know about these connection factories and therefore provides no Quality of Service facilities.

object In object-oriented programming, a concrete realization of a class that consists of data and the operations associated with that data.

object-oriented (OO)
Describing a computer system or programming language that supports objects.

pacing
A technique by which a receiving station controls the rate of transmission of a sending station to prevent overrun.

parallel session
In SNA, two or more concurrently active sessions between the same two LUs using different pairs of network addresses. Each session can have independent session parameters.

PING In Internet communications, a program used in TCP/IP networks to test the ability to reach destinations by sending the destinations an Internet Control Message Protocol (ICMP) echo request and waiting for a reply.

partner logical unit (PLU)
In SNA, the remote participant in a session.

partner transaction program
The transaction program engaged in an APPC conversation with a local transaction program.

PLU See *primary logical unit* and *partner logical unit*.

port An endpoint for communication between devices, generally referring to a logical connection. A 16-bit number identifying a particular Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) resource within a given TCP/IP node.

presentation logic
The part of a distributed application that is concerned with the user interface of the application. Compare with *business logic*.

primary logical unit (PLU)
In SNA, the logical unit that contains the primary half-session for a particular logical unit-to-logical unit (LU-to-LU) session. See also *secondary logical unit*.

protocol boundary
The signals and rules governing interactions between two components within a node.

Resource Access Control Facility (RACF)
An IBM licensed program that provides access control by identifying users to the system; verifying users of the system; authorizing access to protected resources; logging detected unauthorized attempts to enter the system; and logging detected accesses to protected resources.

region In workload management on CICS Universal Client for Windows, an instance of a CICS server.

remote procedure call (RPC)
A protocol that allows a program on a client computer to run a program on a server.

request unit (RU)
In SNA, a message unit that contains control information such as a request code, or function management (FM) headers, end-user data, or both.

request/response unit
A generic term for a request unit or a response unit. See also *request unit* and *response unit*.

response file
A file that contains predefined values that is used instead of someone having to enter those values one at a time. See *CID methodology*.

response unit (RU)
A message unit that acknowledges a request unit; it may contain prefix information received in a request unit.

Resource Recovery Services (RRS)
A z/OS facility that provides two-phase sync point support across participating resource managers.

rollback
An operation in a transaction that reverses all the changes made during the unit of work. After the operation is complete, the unit of work is finished. Also known as a backout.

RU Request unit. Response unit.

RPC See *remote procedure call*.

SBCS See *single-byte character set*.

secondary logical unit (SLU)
In SNA, the logical unit (LU) that contains the secondary half-session for a particular LU-LU session. Contrast with primary logical unit. See also *logical unit*.

Secure Sockets Layer (SSL)
A security protocol that provides communication privacy. SSL enables client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery. SSL applies only to internet protocols, and is not applicable to SNA.

servlet
A Java program that runs on a Web server and extends the server's functionality by generating dynamic content in response to Web client requests. Servlets are commonly used to connect databases to the Web.

session limit
In SNA, the maximum number of concurrently active logical unit to logical unit (LU-to-LU) sessions that a particular logical unit (LU) can support.

single-byte character set (SBCS)
A character set in which each character is represented by 1 byte. Contrast with double-byte character set.

sign-on capable terminal
A sign-on capable terminal allows sign-on transactions, either CICS-supplied (CESN) or user-written, to be run. Contrast with sign-on incapable terminal.

SIT See *system initialization table*.

SNA sense data
An SNA-defined encoding of error information In SNA, the data sent with a negative response, indicating the reason for the response.

SNASVCMG mode name
The SNA service manager mode name. This is the architecturally-defined

mode name identifying sessions on which CNOS is exchanged. Most APPC-providing products predefine SNASVCMG sessions.

socket A network communication concept, typically representing a point of connection between a client and a server. A TCP/IP socket will normally combine a host name or IP address, and a port number.

SSL See *Secure Sockets Layer (SSL)*.

SSLight
An implementation of SSL, written in Java, and no longer supported by CICS Transaction Gateway.

system initialization table
A table containing parameters used to start a CICS control region.

System Management Interface Tool (SMIT)
An interface tool of the AIX operating system for installing, maintaining, configuring, and diagnosing tasks.

standard error
In many workstation-based operating systems, the output stream to which error messages or diagnostic messages are sent.

subnet
An interconnected, but independent segment of a network that is identified by its Internet Protocol (IP) address.

subnet address
In Internet communications, an extension to the basic IP addressing scheme where a portion of the host address is interpreted as the local network address.

sync point
A logical point in the execution of program where the changes made by the program are consistent and complete, and can be committed. The output, which has been held up to that point, is sent to its destination, the input is removed from the message queues, and updates are made available to other applications. When a program terminates abnormally, CICS recovery and restart facilities do not backout updates prior to the last completed sync point.

Systems Network Architecture (SNA)
An architecture that describes the logical structure, formats, protocols, and operational sequences for transmitting information units through the networks and also the operational sequences for controlling the configuration and operation of networks.

System SSL
An implementation of SSL, no longer supported by CICS Transaction Gateway on z/OS.

TCP62 SNA logical unit type 62 (LU62) protocol encapsulated in TCP/IP. This allows APPC applications to communicate over a TCP/IP Network without changes to the applications.

TCP/IP
See *Transmission Control Protocol/Internet Protocol*.

TCP/IP load balancing
The ability to distribute TCP/IP connections across target servers.

terminal emulation
The capability of a microcomputer or personal computer to operate as if it were a particular type of terminal linked to a processing unit and to access data. See also *emulator, emulation program*.

thread
A stream of computer instructions that is in control of a process. In some operating systems, a thread is the smallest unit of operation in a process. Several threads can run concurrently, performing different jobs.

timeout
A time interval that is allotted for an event to occur or complete before operation is interrupted.

TLS
See *Transport Layer Security (TLS)*.

token-ring network
A local area network that connects devices in a ring topology and allows unidirectional data transmission between devices by a token-passing procedure. A device must receive a token before it can transmit data.

trace
A record of the processing of a computer program. It exhibits the sequences in which the instructions were processed.

transaction program
A program that uses the Advanced Program-to-Program Communications (APPC) application programming interface (API) to communicate with a partner application program on a remote system.

Transmission Control Protocol/Internet Protocol (TCP/IP)
An industry-standard, nonproprietary set of communications protocols that provide reliable end-to-end connections between applications over interconnected networks of different types.

Transport Layer Security (TLS)
A security protocol that provides communication privacy. TLS enables client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery. TLS applies only to internet protocols, and is not applicable to SNA. TLS is also known as SSL 3.1.

type 2.0 node
A node that attaches to a subarea network as a peripheral node and provides a range of end-user services but no intermediate routing services.

type 2.1 node
An SNA node that can be configured as an endpoint or intermediate routing node in a network, or as a peripheral node attached to a subarea network.

Uniform Resource Locator (URL)
A sequence of characters that represent information resources on a computer or in a network such as the Internet. This sequence of characters includes (a) the abbreviated name of the protocol used to access the information resource and (b) the information used by the protocol to locate the information resource.

unit of recovery (UR)
A defined package of work to be performed by the RRS.

unit of work (UOW)
A recoverable sequence of operations performed by an application between

two points of consistency. A unit of work begins when a transaction starts or at a user-requested sync point. It ends either at a user-requested sync point or at the end of a transaction.

user session
Any APPC session other than a SNASVCMG session.

verb
A reserved word that expresses an action to be taken by an application programming interface (API), a compiler, or an object program.

In SNA, the general name for a transaction program's request for communication services.

Web browser
A software program that sends requests to a Web server and displays the information that the server returns.

Web server
A software program that responds to information requests generated by Web browsers.

wide area network (WAN)
A network that provides communication services to a geographic area larger than that served by a local area network or a metropolitan area network, and that may use or provide public communication facilities.

wrapping trace
A configuration in which the **Maximum Client wrap size** setting is greater than 0. The total size of Client daemon binary trace files is limited to the value specified in the **Maximum Client wrap size** setting. With standard I/O tracing, two files, called `cicscli.bin` and `cicscli.wrp`, are used; each can be up to half the size of the **Maximum Client wrap size**.

XA requests
An XA request is any request sent or received by the CICS Transaction Gateway in support of an XA transaction. These requests include the XA commands commit, complete, end, forget, prepare, recover, rollback, and start.

XA transaction
A global transaction that adheres to the X/Open standard for distributed transaction processing (DTP.)

Index

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX
AnyNet
AS/400
CICS
CICS/400
CICS/ESA
CICS/VSE
DB2
Domino
Hummingbird
IBM
IBM
IBMLink
IMS
iSeries
MQSeries
MVS
MVS/ESA
Notes
OS/2
OS/390
POWER
pSeries
RACF
Redbooks
RETAIN
RMF
RS/6000
SAA
SP2
System/390
Tivoli
TXSeries

VisualAge
VSE/ESA
VTAM
WebSphere
z/OS
zSeries

Microsoft®, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Intel, Intel Inside® (logos), MMX and Pentium® are trademarks of Intel Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product or service names may be trademarks or service marks of others.

Sending your comments to IBM

If you especially like or dislike anything about this book, use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Limit your comments to the information in this book and the way in which the information is presented.

To ask questions, make comments about the functions of IBM products or systems, or to request additional publications, contact your IBM representative or your IBM authorized remarketer.

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

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

  User Technologies Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom
- By fax:
  - +44 1962 842327 (if you are outside the UK)
  - 01962 842327 (if you are in the UK)
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication title and order number
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5724-J09

SC34-6755-02

Spine information:

CICS Universal Client UNIX and Linux Client Administration Version 7.1