CICS® Universal Clients Configuration

CICS Universal Client Security

CICS® Universal Clients Configuration

CICS Universal Client Security

Contents

Overview

CICS servers may require that a userid and password be provided by the CICS Universal Client before they permit a client connection, terminals to be installed, or transactions to be run. This is dependent upon the server and protocol security settings. The userid and password are sent to the server in the FMH header of the transaction attach request for each conversation. A userid and password are also required when a signon transaction is invoked on a signon capable terminal. In this instance, the userid and password are flowed to the server as part of the 3270 datastream.

Note: The CICS Universal Client may be running as a standalone CICS Universal Client or as part of the CICS Transaction Gateway (CTG).

The CICS Universal Client has no security manager and so does not support userid authentication. It is therefore recommended that you configure your CICS server client connections so that incoming attach requests must specify a userid and password. For mainframe servers, specify **AttachSec = Verify** in the CICS connection definition. **AttachSec = Identify**, which indicates that userid, but not password, are required, is **not** supported for client connections.

Default connection settings

The CICS Universal Client maintains a default userid and password per server connection, which may be set by any of the following methods:

- CICSCLI security commands:

  ```
  cicscli /c=servername /u=userid /p=password
  ```

  On Unix platforms, the servername parameter can also be specified with the /s option.
- From C, PL/I or COBOL, use the ESI function **CICS_SetDefaultSecurity**. Note that this call is not available from the CICS Transaction Gateway APIs.

  From C++, use the **makeSecurityDefault** method of the **CclConn** or **CclTerminal** class.

  From COM, use the **MakeSecurityDefault** method of the **Connect** or **Terminal** COM class.
- If running on a Windows platform, via the Network Provider Interface (NPI); refer to the *CICS Universal Client for Windows Administration* book for details.
- If running on a Windows or OS/2 platform, via a CICS Universal Client security popup.

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: When the CICS Client is running as a service on a Windows platform, the default userid and password values entered by the currently logged on user, by whatever method, are retained even when that user logs off and are reused when a subsequent user logs on.

Security popups

The CICS Universal Client will display a security popup to allow the user to enter a userid and password if **ALL** of the following are true:

- The client platform is Windows or OS/2.
- Popups are enabled in the initialization file (ctg.ini).
- Under Windows, the client is not running as a service, or is running as a service that is enabled to interact with the desktop.
- The server requires that a userid and password are flowed in the transaction attach request.
- No default security has yet been set for the server connection **or** default security has been set but the settings produce a security error on the server, for example, password expired.
- The transaction being invoked by the client is CCIN (client install) or CTIN (terminal install).
- A terminal userid and password have not been supplied on a terminal install request.

Values entered via a security popup, once verified, are used to set the default userid and password values for that server connection.

Security popups are not displayed directly by ECI or EPI transaction requests, although they may be prompted by these requests, if the request causes a server connection to be established, or a terminal to be installed.

It is recommended that popups are disabled on gateway machines.

ECI security

An application may provide a userid and password on an ECI request and these values will override any default values set for the server connection as follows.

- C, PL/I or COBOL 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.
- Java client programs:

  Set the userid and password parameters when constructing an **ECIRequest** object.

EPI terminal security

From Version 3.1 onwards, the CICS Universal Client also maintains a userid and password per installed terminal. These values will override any default values set for the server connection. They can be set by one of the following methods:

- C, PL/I or COBOL 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.
- COM programs:

  Use the **AlterSecurity** method of the **Terminal** COM class. This can only be used for signon incapable created terminals.
- Java client programs:

  Set the userid and password parameters when constructing an **EPIRequest** object via the **addTerminal** or **addTerminalAsync** method. Or, use the **alterSecurity** method of the **EPIRequest** class.

Terminal security can **NOT** be set when using the following APIs:

- CTG EPI support classes
- CTG EPI bean classes
- Terminal Servlet
- CICS Connectors

Terminal security would normally only be required if using signon incapable terminals.

Signon capable and signon incapable terminals

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

If a terminal resource is installed as **signon capable**, the application or user is responsible for starting a signon transaction; the userid and password once entered are embedded in the 3270 data.

- Transactions started before the signon transaction will execute with the authorities granted to the default userid defined for the server. A check is also done against the userid associated with the connection to see whether the CICS Universal Client itself has authority to access the resource.
- Transactions started after the signon transaction will execute with the authorities granted to the authenticated userid. For transactions attempting to access resources, security checking is done against the signed-on user's userid and the userid associated with the connection. If a server supports signon timeout and a client terminal is left idle so that the timeout expires, the user is signed off without notification and the next transaction will run against the default userid.

If the terminal resource is installed as **signon incapable**, the userid and password are authenticated for each transaction started for the terminal resource.

Prior to Version 3.1.0 of CICS Universal Clients and Gateways, whether a terminal was signon capable or not was dependant upon the server implementation. Client terminals installed on distributed CICS servers were signon capable and terminals installed on mainframe CICS and CICS/400 servers were signon incapable.

In Version 3.1.0, new EPI function was introduced to allow the signon capability of a terminal to be specified by one of the following methods.

- C, PL/I or COBOL programs:

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

  Set the signon capability parameter when constructing a **CclTerminal** class object.
- COM programs:

  Use the **SetTermDefns** method of the **Terminal** COM class.
- Java client programs:

  Set the signon capability parameter when constructing an **EPIRequest** object via the **addTerminal** or **addTerminalAsync** method.

The signon 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 signon capability parameter in the terminal install (CTIN) response.

The signon capability of a terminal can **NOT** be specified when using the following APIs:

- CTG EPI support classes
- CTG EPI bean classes
- CTG Terminal Servlet
- CICS Connectors

If you are using one of these interfaces, the EPI functionality available is unchanged from release Version 3.0.x, that is, you cannot specify a userid and password per terminal, or specify the signon capability.

To use any of the new EPI functionality, you must ensure that you have applied the relevant server APAR, see "Required APARS" on page 7.

Refer to the *CICS Family Client/Server Programming* book for further information.

Mainframe CICS servers

These servers support both signon capable and incapable terminals, provided that they are at the prerequisite maintenance level (see "Required APARS" on page 7). A terminal install request that does not specify any signon capability, for example from **CICS_EpiAddTerminal**, will result in a signon incapable terminal being installed.

For signon 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 via a signon transaction are flowed to the server as part of the 3270 datastream and they will therefore appear in a client trace.

  If you are planning to use signon capable terminals, it is recommended that you specify **UseDfltUser = Yes** in the CICS CONNECTION definition, or ensure that a default connection userid and password are set by the system administrator for the client. Otherwise, the user may be prompted to enter a userid and password for the CTIN terminal install request and then be required to run CESN to signon to the terminal as well.

- Before the user has signed on, transactions will run under the default userid for the CICS server. After signon, transactions will run under the signed-on userid.

For signon incapable terminals without terminal security:

- Use the **CICS_EpiAddTerminal** call
- A connection userid and password will be required regardless of the setting of the **UseDfltUser** in the CICS connection definition on the server.
- Transactions will run under the userid specified in the corresponding FMH attach request.

For signon 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.

If you wish to use one of the APIs that does not support the new EPI functionality, then you can use CRTE through a middle tier system to get signon capable terminal-like functionality.

TXSeries and CICS OS/2 servers

These servers do not support the signon capability parameter in a terminal install (CTIN) request but will tolerate it if the required APAR is applied (see “Required APARS” on page 7). A terminal install request will always result in a signon capable terminal being installed, regardless of the signon capability requested.

CICS/400 servers

These servers do not support the signon capability parameter in a terminal install (CTIN) request . A terminal install request will result in a signon incapable terminal being installed, regardless of the signon capability requested.

CICSTERM

In v3.10 and later, the default behavior of CICSTERM has changed; it now attempts to install a signon capable terminal.

Use the option ’-a’ to request the old default CICSTERM behaviour as in releases prior to Version 3.1.0. The resulting CTIN request in this case will not contain the signon capability parameter.

CICSTELD

CICSTELD does not support the new EPI functionality.

Required APARS

The server APAR fixes to support the terminal signon capability are as follows:

CICS/ESA v4.1	PQ30167
CICS TS for OS/390 v1.2 & 3	PQ30168
CICS TS for VSE/ESA v1.1	PQ30170
TXSeries	IY03691

CICS TS for OS/390 V1.3 servers

If the server does not have the required APAR applied and the ’-a’ option is not specified on CICSTERM, the installed terminal will give unpredictable results.

TXSeries servers

If the server does not have the required APAR applied and the ’-a’ option is not specified on CICSTERM, the Client will display the message:

```
CL7053E Errors found while communicating with server
```

and the message:

```
CCL3105 Inbound CICS datastream error (CTIN, 4, 0)
```

will be written to CICSCLI.LOG.

On the server, the message:

```
ERZ042004E/0112: An invalid request was received from client
```

will be written to CSMT.out and console.msg will include:

```
ERZ014016E/0036: Transaction CTIN Abend A42B
```

Appendix. Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS
CICS/ESA
CICS/VSE
CICS/400
IBM
OS/2
OS/390

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.