webMethods EntireX

EntireX RPC Server for CICS® IPIC

Version 10.9

April 2023

This document applies to webMethods EntireX Version 10.9 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: XX-IPICRPC-109-20230403

Table of Contents

1 About this Documentation

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

In addition, you can also access the cloud product documentation via **https://www.softwareag.cloud**. Navigate to the desired product and then, depending on your solution, go to “Developer Center”, “User Center” or “Documentation”.

Product Training

You can find helpful product training material on our Learning Portal at **https://knowledge.softwareag.com**.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at **https://tech-community.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

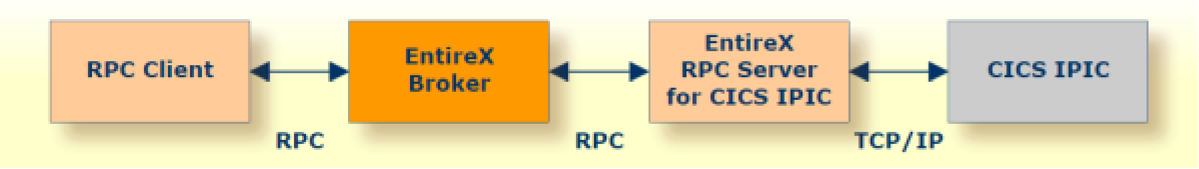
Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Introduction to the RPC Server for CICS IPIC

The EntireX RPC Server for CICS® IPIC allows standard RPC clients to communicate with CICS programs running on IBM CICS® without installing EntireX components on the mainframe (Zero Footprint). The CICS interface types Channel Container and DFHCOMMAREA are supported.

Overview

The RPC Server for CICS IPIC acts on one side as an RPC server and on the other side as a client for CICS IPIC. The RPC Server for CICS IPIC is a Java-based component that can run on a different host to the one where CICS is running. This allows it to operate with a zero footprint of EntireX on the CICS host.



For local extraction, all source files have to be stored locally on the same machine where the Designer is running.

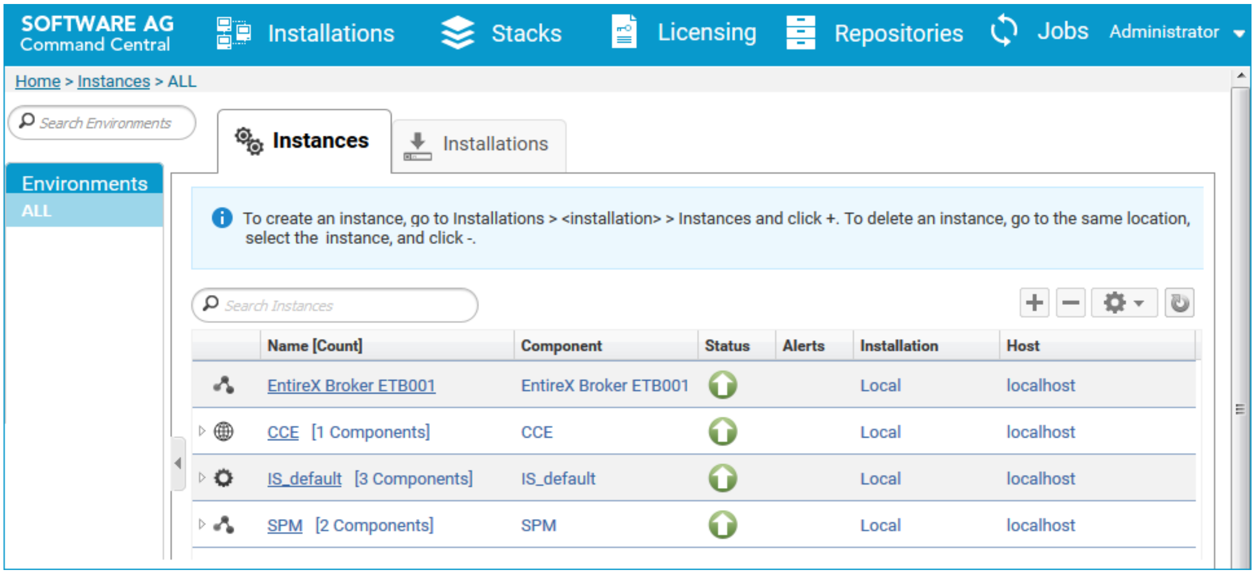
- For existing CICS COBOL programs, use the *IDL Extractor for COBOL* to extract the *Software AG IDL File* in the IDL Editor documentation for the RPC clients.
- For existing CICS PL/I programs, use the *IDL Extractor for PL/I* to extract the *Software AG IDL File* in the IDL Editor documentation for the RPC clients.

Remote extraction requires an RPC server running under z/OS with Extractor Service (Batch | IMS).

- For COBOL, see *Step 2: Select a COBOL Extractor Environment or Create a New One* in the IDL Extractor for COBOL documentation.
- For PL/I, see *Extract Software AG IDL File from a Remote PL/I RPC Environment* in the IDL Extractor for PL/I documentation.

Administration using Command Central

Software AG Command Central is a tool that enables you to manage your Software AG products remotely from one location. Command Central offers a browser-based user interface, but you can also automate tasks by using commands to remotely execute actions from a terminal or custom script (for example CI servers such as Jenkins, or generic configuration management tools such as Puppet or Chef).



To create an instance, go to Installations > <installation> > Instances and click +. To delete an instance, go to the same location, select the instance, and click -.

Name [Count]	Component	Status	Alerts	Installation	Host
EntireX Broker ETB001	EntireX Broker ETB001	⬆		Local	localhost
CCE [1 Components]	CCE	⬆		Local	localhost
IS_default [3 Components]	IS_default	⬆		Local	localhost
SPM [2 Components]	SPM	⬆		Local	localhost

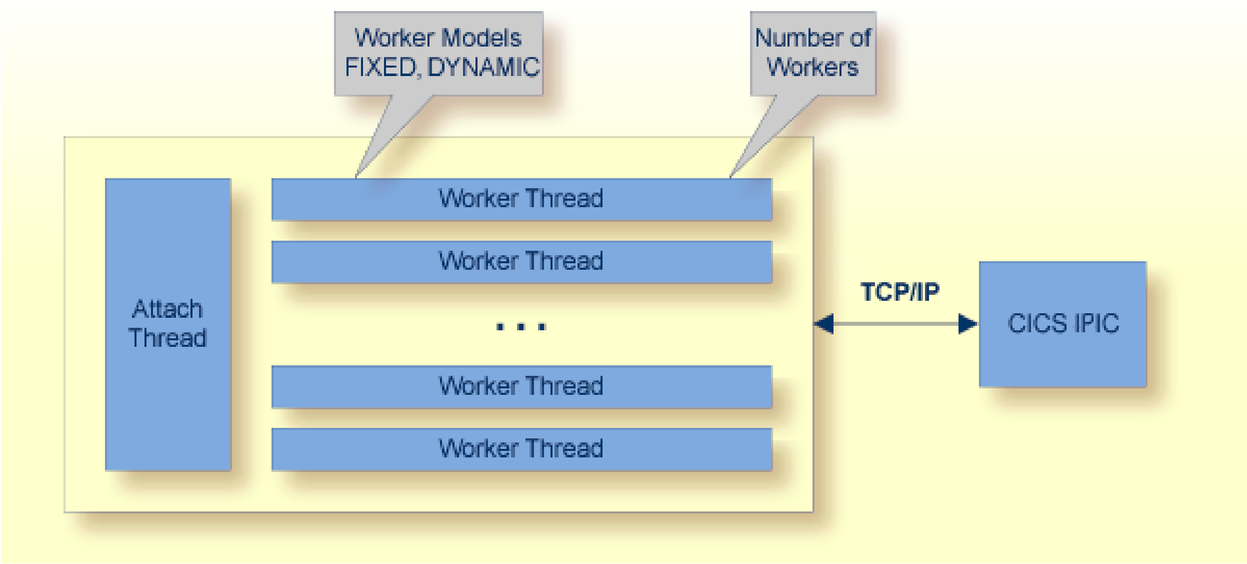
Command Central can assist with the following configuration, management, and monitoring tasks:

- Infrastructure engineers can see at a glance which products and fixes are installed, where they are installed, and compare installations to find discrepancies.
- System administrators can configure environments by using a single web user interface or command-line tool. Maintenance involves minimum effort and risk.
- Release managers can prepare and deploy changes to multiple servers using command-line scripting for simpler, safer lifecycle management.
- Operators can monitor server status and health, as well as start and stop servers from a single location. They can also configure alerts to be sent to them in case of unplanned outages.

The Command Central graphical user interface is described under *Administering the RPC Server for CICS IPIC using the Command Central GUI*. For the command-line interface, see *Administering the RPC Server for CICS IPIC using the Command Central Command Line*.

The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Worker Models



RPC requests are worked off inside the RPC server in worker threads. If you are using RPC conversations, each RPC conversation requires its own thread during the lifetime of the conversation. The RPC Server for CICS IPIC can adjust the number of worker threads to the number of parallel requests. The RPC server provides two worker models:

- `FIXED`
  The *fixed* model creates a fixed number of worker threads. The number of worker threads does not increase or decrease during the lifetime of an RPC server instance.
- `DYNAMIC`
  The *dynamic* model creates worker threads depending on the incoming load of RPC requests.

For configuration with the Command Central GUI, see ***Worker Scalability*** under *Configuration > Server*.

For technical details, see property `entirex.server.fixedservers` under *Administering the RPC Server for CICS IPIC*.

3 Administering the RPC Server for CICS IPIC using the Command Central GUI

This chapter describes how to administer the EntireX RPC Server for CICS® IPIC, using the Command Central graphical user interface.

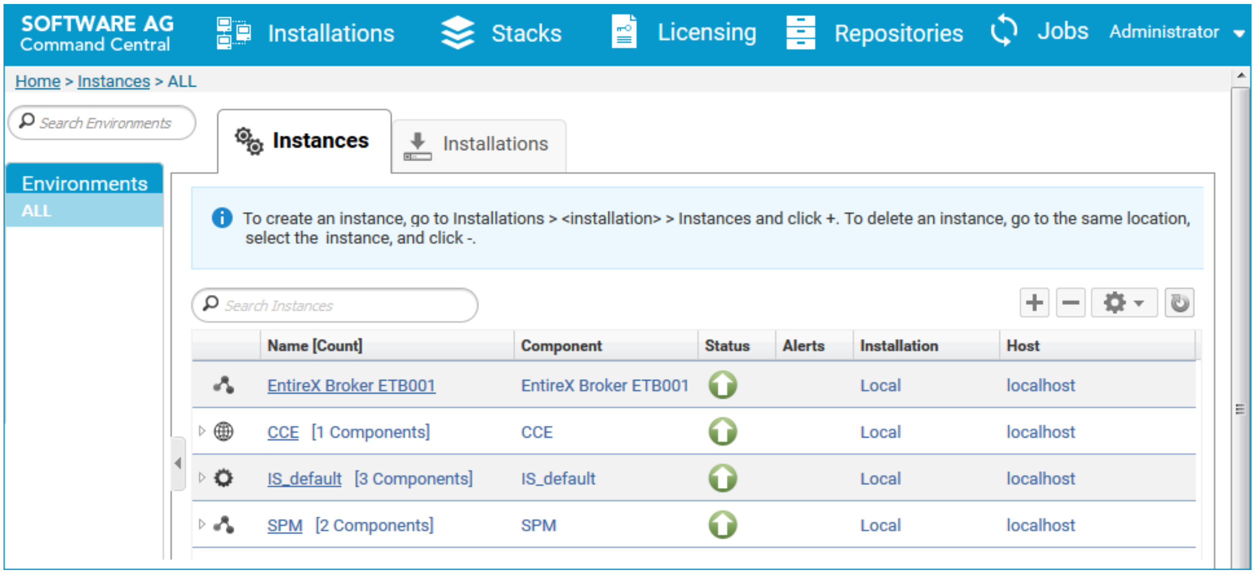
See also *Administering the RPC Server for CICS IPIC using the Command Central Command Line*. The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Logging in to Command Central

Open an Internet browser and specify the URL of the Command Central Server as follows: *http://<Command_Central_host>:<Command_Central_port>*. This takes you to the Command Central **Login** page.

On Windows you can also get to the **Login** page from the Command Central Start Menu entry.

Provide your user credentials in the **Login** page and click **Log In**. This takes you to the page **Home > Instances**:



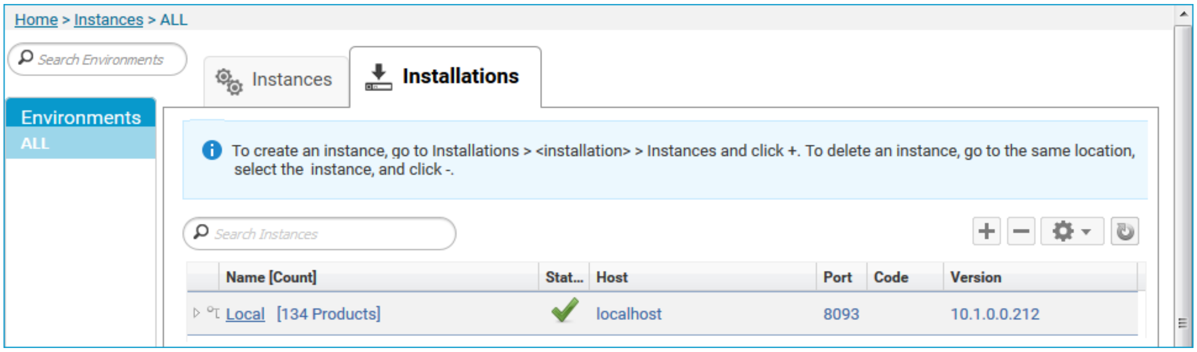
To create an instance, go to Installations > <installation> > Instances and click +. To delete an instance, go to the same location, select the instance, and click -.

Name [Count]	Component	Status	Alerts	Installation	Host
EntireX Broker ETB001	EntireX Broker ETB001	⬆		Local	localhost
CCE [1 Components]	CCE	⬆		Local	localhost
IS_default [3 Components]	IS_default	⬆		Local	localhost
SPM [2 Components]	SPM	⬆		Local	localhost

Creating an RPC Server Instance

To create an RPC Server for CICS IPIC instance

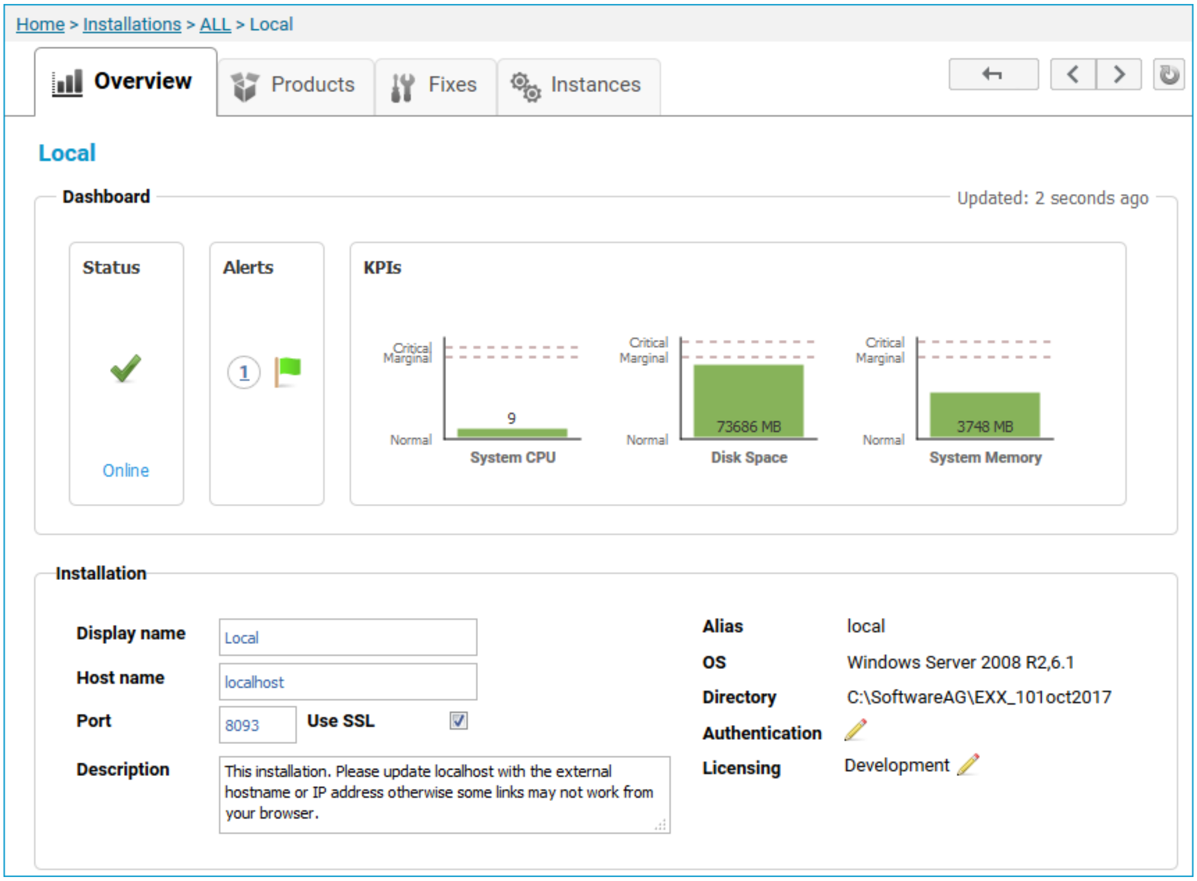
1 In the Command Central home page, click the **Installations** tab.



To create an instance, go to Installations > <installation> > Instances and click +. To delete an instance, go to the same location, select the instance, and click -.

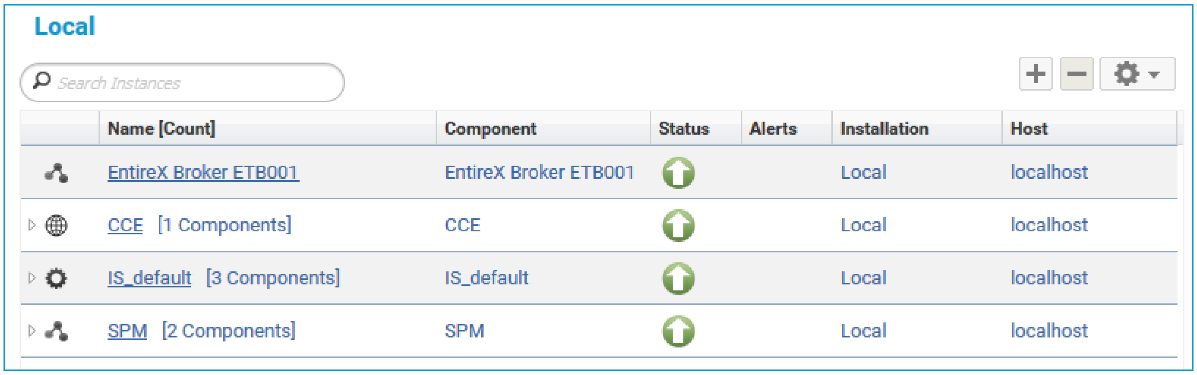
Name [Count]	Stat...	Host	Port	Code	Version
Local [134 Products]	✓	localhost	8093		10.1.0.0.212

2 Click on the desired installation, for example **Local**, where you want to add an RPC Server for CICS IPIC instance.



Display name	Local
Host name	localhost
Port	8093
Use SSL	☑
Description	This installation. Please update localhost with the external hostname or IP address otherwise some links may not work from your browser.
Alias	local
OS	Windows Server 2008 R2,6.1
Directory	C:\SoftwareAG\EXX_101oct2017
Authentication	
Licensing	Development

3 Click the **Instances** tab.

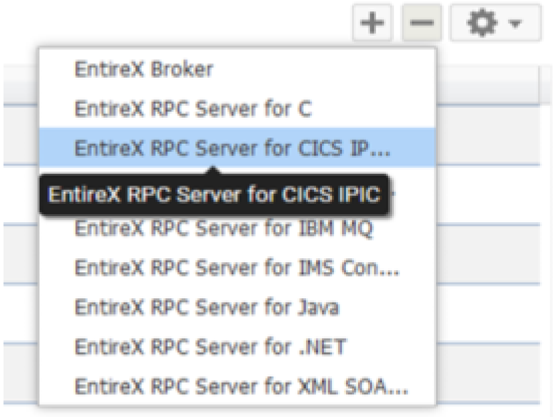


Local

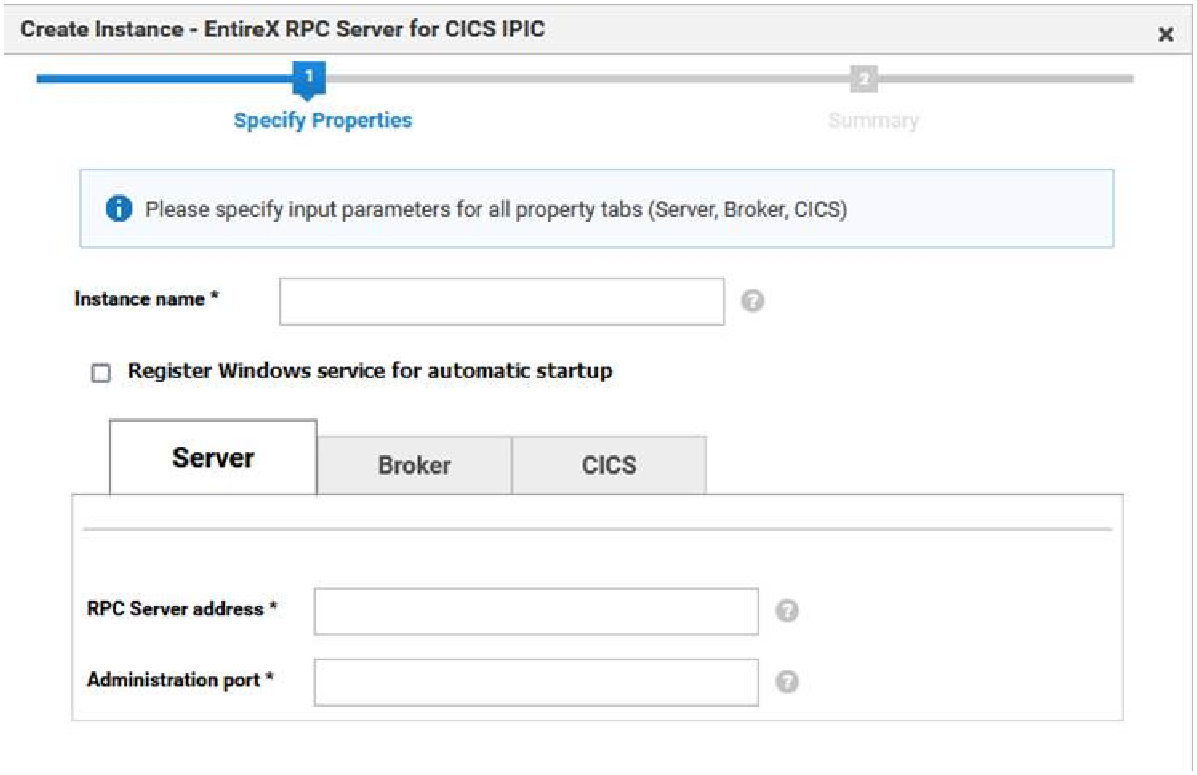
Search Instances

Name [Count]	Component	Status	Alerts	Installation	Host
EntireX Broker ETB001	EntireX Broker ETB001	⇧		Local	localhost
CCE [1 Components]	CCE	⇧		Local	localhost
IS_default [3 Components]	IS_default	⇧		Local	localhost
SPM [2 Components]	SPM	⇧		Local	localhost

4 Click the + button in the upper right corner above the list and choose **EntireX RPC Server for CICS® IPIC**.


EntireX Broker
EntireX RPC Server for C
EntireX RPC Server for CICS IP...
EntireX RPC Server for CICS IPIC
EntireX RPC Server for IBM MQ
EntireX RPC Server for IMS Con...
EntireX RPC Server for Java
EntireX RPC Server for .NET
EntireX RPC Server for XML SOA...

5 In the **Create Instance** wizard, fill in the fields in the main screen and in the **Server**, **Broker** and **CICS** tabs.



Create Instance - EntireX RPC Server for CICS IPIC

Please specify input parameters for all property tabs (Server, Broker, CICS)

Instance name *

□ Register Windows service for automatic startup

RPC Server address *

Administration port *

Main Screen

Parameter	Description
Instance name	Required. Name of the runtime component, for example "`MyRpcServer`".
Register Windows Service for automatic startup	Optional. Register Windows Service for automatic startup. Default is not checked. If this parameter is checked, the RPC server can be controlled by the Windows Service Control Manager.

Server Tab

Parameter	Description
RPC Server address	Required. The case-sensitive RPC server address has the format: `CLASS/SERVER/SERVICE`.
Administration port	Required. The administration port in range from 1025 to 65535.

Broker Tab

Parameter	Description
Connection	
Transport	Transport over TCP or SSL. Default is `TCP`.
Broker host	Required. EntireX Broker host name or IP address. See *Using the Broker ID in Applications* in the RPC Programming documentation.
Broker port	Required. Port number in range from 1025 to 65535.
SSL trust store	Optional. Specifies the location of the SSL trust store.
Credentials	
User	Optional. The user ID for secured access to the broker.
Password	Optional. The password for secured access to the broker.

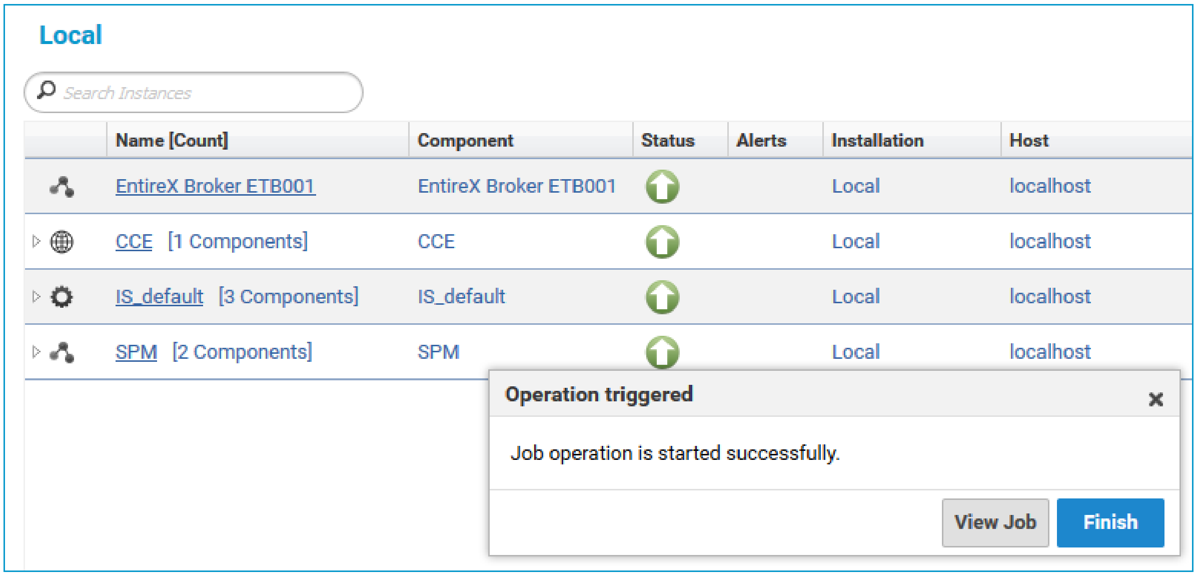
CICS Tab

Here you can modify the CICS IPIC settings for the RPC Server for CICS IPIC. As a prerequisite CICS IPIC must be configured, see *Preparing IBM CICS for IPIC*.

Parameter	Description
Connection	
Transport	Use TCP or SSL to communicate with CICS IPIC.
CICS host	Host name or IP address where CICS is running.
CICS port	TCP or SSL port number (1-65535) of CICS IPIC.
CICS transaction ID	Transaction ID (1-4 chars). Name of the CICS mirror transaction that will receive transactions.
CICS encoding	Specify the appropriate EBCDIC encoding used by your CICS installation. Default is codepage `cp037` with full Latin-1 character set.
Credentials	
RACF user ID	The user ID (max. 8 chars) as defined in your underlying mainframe security system (e.g. RACF).
RACF password	Password/passphrase as defined in your underlying mainframe security system (e.g. RACF).

6 Press **Next** to get to the **Summary** page to verify your input.

7 Press **Finish**.



Local

	Name [Count]	Component	Status	Alerts	Installation	Host
	EntireX Broker ETB001	EntireX Broker ETB001	⬆		Local	localhost
▷	CCE [1 Components]	CCE	⬆		Local	localhost
▷	IS_default [3 Components]	IS_default	⬆		Local	localhost
▷	SPM [2 Components]	SPM	⬆		Local	localhost

Operation triggered

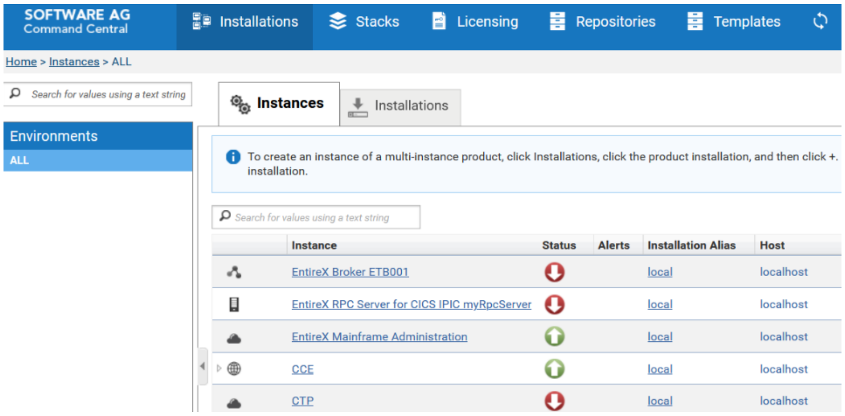
Job operation is started successfully.

The new instance *myRpcServer* appears in the list.

Configuring an RPC Server Instance

To configure an RPC Server for CICS IPIC instance

1 In the Command Central home page, click the **Instances** tab.

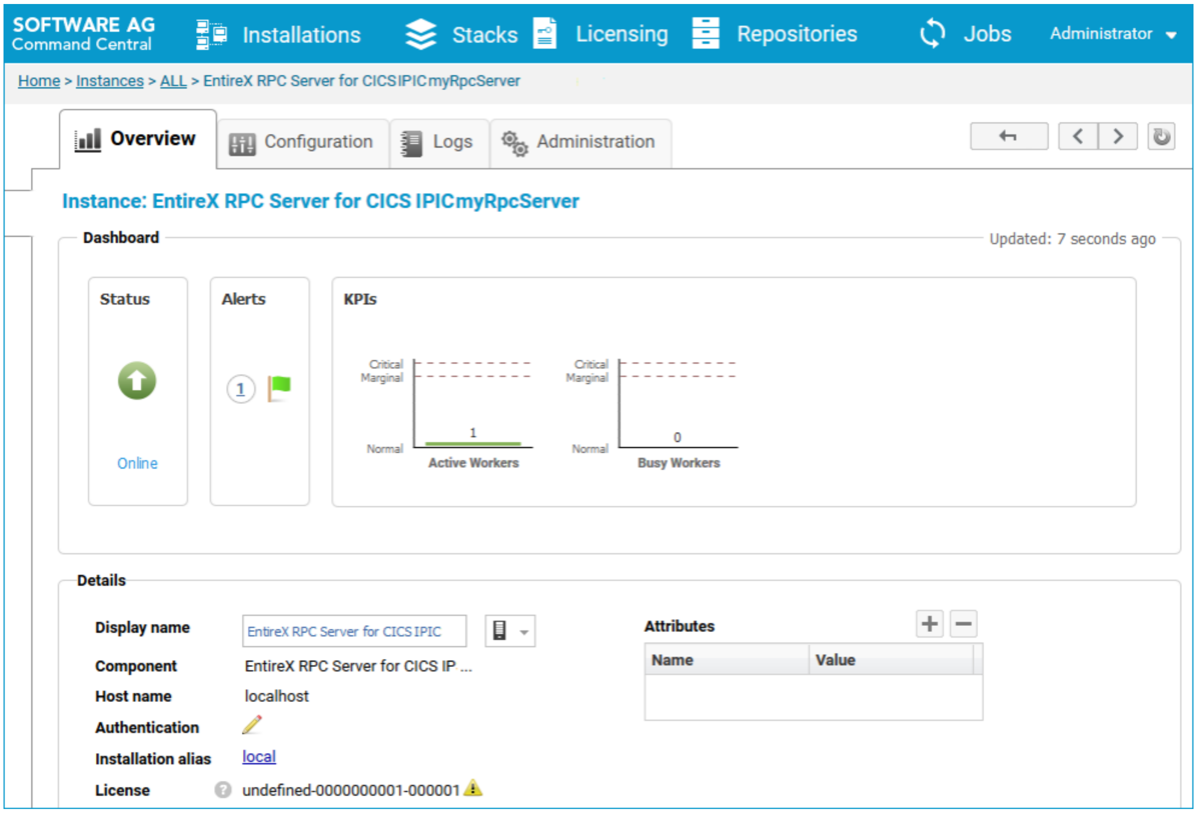


Home > Instances > ALL

To create an instance of a multi-instance product, click Installations, click the product installation, and then click +. installation.

Instance	Status	Alerts	Installation Alias	Host
EntireX Broker ETB001	↓		local	localhost
EntireX RPC Server for CICS IPIC myRpcServer	↓		local	localhost
EntireX Mainframe Administration	↑		local	localhost
CCE	↑		local	localhost
CTP	↓		local	localhost

2 Click on the link associated with this instance to select the RPC server instance you want to configure.



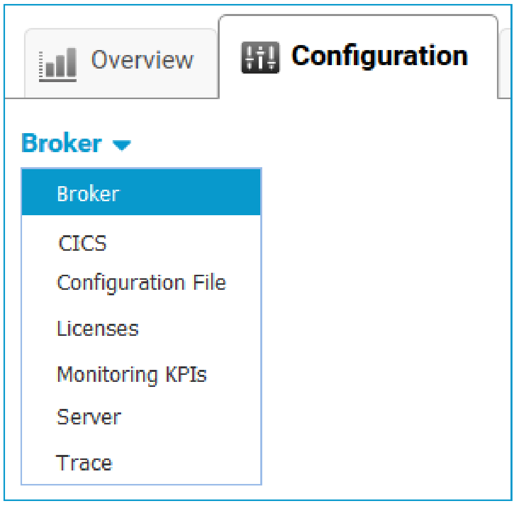
Home > Instances > ALL > EntireX RPC Server for CICSIPICmyRpcServer

Instance: EntireX RPC Server for CICS IPICmyRpcServer

Updated: 7 seconds ago

Display name	EntireX RPC Server for CICS IPIC
Component	EntireX RPC Server for CICS IP ...
Host name	localhost
Authentication	
Installation alias	local
License	undefined-0000000001-000001

3 Click the **Configuration** tab. EntireX supports the following configuration types, which are presented in a drop-down box when you click the down arrow below the **Configuration** tab label:



Note: All configuration changes require a restart of the instance to take effect.

4 Click **Edit** to modify the parameters on your selected configuration type.

5 Click **Test** to check the correctness of your input or **Apply** to save your changes.

The configuration options are described in more detail below:

- Broker
- CICS
- Configuration File
- Licenses
- Monitoring KPIs
- Server
- Trace Level

Broker

Parameter	Description
Connection	
Transport	Transport over TCP or SSL. Default is `TCP`.
Broker host	Required. EntireX Broker host name or IP address. See *Using the Broker ID in Applications* in the RPC Programming documentation.
Broker port	Required. Port number in range from 1025 to 65535.
SSL trust store	Optional. Specifies the location of the SSL trust store.

Parameter	Description
SSL trust password	Optional. The password of the SSL trust store.
SSL verify server	Optional. The RPC server as SSL client checks the identity of the broker as SSL server.
FIPS-140 mode	Optional. Enable FIPS-140 compliant SSL communication. Default is `no`.
Credentials	
User	Optional. The user ID for secured access to the broker.
Password	Optional. The password for secured access to the broker.

CICS

Here you can modify the CICS IPIC specific parameters.

Parameter	Description
Connection	
Transport	Use TCP or SSL to communicate with CICS IPIC.
CICS host	Host name or IP address where CICS is running.
CICS port	TCP or SSL port number (1-65535) of CICS IPIC.
CICS transaction ID	Transaction ID (1-4 chars). Name of the CICS mirror transaction that will receive transactions.
CICS application ID	Client Application ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
CICS network ID	Client Network ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
CICS encoding	Specify the appropriate EBCDIC encoding used by your CICS installation. Default is codepage `cp037` with full Latin-1 character set.
CICS SSL trust store	Specifies the location of the SSL trust store.
CICS SSL trust password	The password of the SSL trust store.
CICS SSL verify server	The RPC server as SSL client checks the identity of CICS as SSL server.
CICS socket timeout	Timeout (in seconds) for CICS. Default is 20 seconds.
CICS send sessions	Number of simultaneous transactions (1-999) allowed over the CICS connection. See your IBM documentation for more information.
Credentials	
RACF user ID	The user ID (max. 8 chars) as defined in your underlying mainframe security system (e.g. RACF).
RACF password	Password/passphrase as defined in your underlying mainframe security system (e.g. RACF).
Use PassTicket (1)	Use PassTicket instead of password.
Application name	Required if PassTicket is to be used instead of a password. Application name (1-8 chars) as defined in your RACF system. This property is ignored if RACF password is set.

Parameter	Description
Secured signon key	Required if PassTicket is to be used instead of a password. Secured signon key as defined in your RACF system. Must be exactly 16 characters long.

Notes:

1. By default, PassTickets can only be used once during a one-second time interval. They are protected against replay. This limits the workload to one request per second.

   Caution: IBM provides a mechanism to bypass the PassTicket's replay mechanism, allowing higher workloads, but this introduces security risks. The option to bypass the PassTicket's replay mechanism should only be used in secure environments where access to generated PassTickets is limited to a secure or internal network. See your IBM documentation for more information.

Configuration File

Here you can view/edit the configuration file of the RPC Server for CICS IPIC.

Licenses

Here you can view/set the license file in the EntireX installation. For details see *Import Product License Keys for Instances or Components* under *Getting Started with Command Central* in the separate Command Central documentation under Software AG Suite & Cross-Product Guides.

Note: The license file is used for all EntireX instances in this installation.

Monitoring KPIs

Here you can modify margins of monitored key performance indicators (KPIs) available for the RPC Server for CICS IPIC: Active Workers and Busy Workers.

Key performance indicators (KPIs) enable you to monitor the health of your RPC Server for CICS IPIC. The following KPIs help you administer, troubleshoot, and resolve performance issues:

KPI	Setting
Absolute number of Active Workers	`entirex.generic.kpi.1.max=20`
Critical alert relative to maximum	`entirex.generic.kpi.1.critical=0.95`
Marginal alert relative to maximum	`entirex.generic.kpi.1.marginal=0.80`
Absolute number of Busy Workers	`entirex.generic.kpi.2.max=20`
Critical alert relative to maximum	`entirex.generic.kpi.2.critical=0.95`
Marginal alert relative to maximum	`entirex.generic.kpi.2.marginal=0.80`

Do not change the other properties!

Server

Here you can specify the RPC Server settings.

Parameter	Description
RPC Server	
RPC Server address	Required. The case-sensitive RPC server address has the format: `CLASS/SERVER/SERVICE`.
Administration port	Required. The administration port in range from 1025 to 65535.
Reconnection attempts	Required. Number of reconnection attempts to the broker. When the number of attempts is reached and a connection to the broker is not possible, the RPC Server stops.
Worker Scalability	
Worker model	You can either have a fixed or dynamic number of workers. Default is `dynamic` (`true`). For more information see ***Worker Models***.
Fixed number	Required. Fixed number of workers. Must be a number in range from 1 to 255.
Minimum number	Required. Minimum number of workers. Must be a number in range from 1 to 255.
Maximum number	Required. Maximum number of workers. Must be a number in range from 1 to 255.

Trace Level

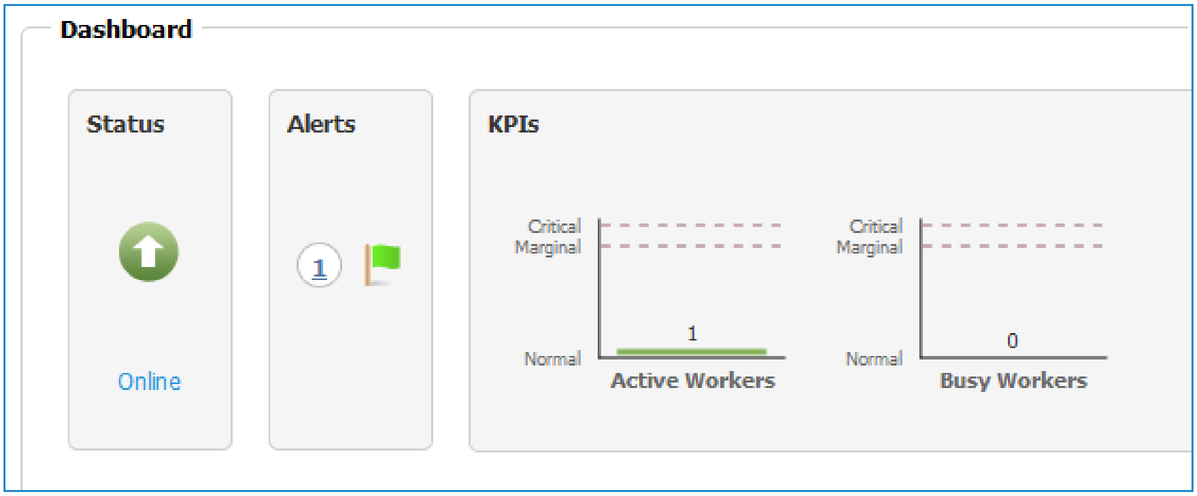
Here you can set the trace level of the RPC Server for CICS IPIC.

Parameter	Value	Description
Trace level	`0-3`	One of the following levels: 0 - None - No trace output (default). 1 - Standard - Minimal trace output. 2 - Advanced - Detailed trace output. 3 - Support - Support diagnostic. Use only when requested by Software AG Support.

Viewing the Runtime Status

To view the runtime status of the RPC server instance

- In the Command Central **Home** page, click the **Instances** tab and select the RPC Server for CICS IPIC instance for which you want to see the runtime status (same as Step 1 under *Configuring a Broker Instance*).



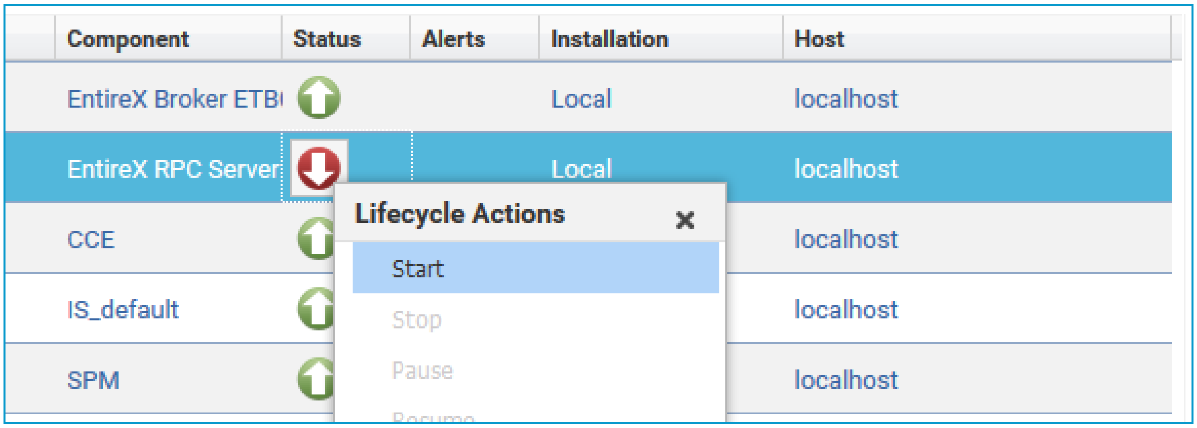
The visual key performance indicators (KPIs) and alerts enable you to monitor the RPC Server for CICS IPIC's health.

KPI	Description
Active Workers	Number of active workers.
Busy Workers	Number of busy workers.

Starting an RPC Server Instance

To start an RPC Server for CICS IPIC instance from the Instances tab

1 In the Command Central home page, click the **Instances** tab.

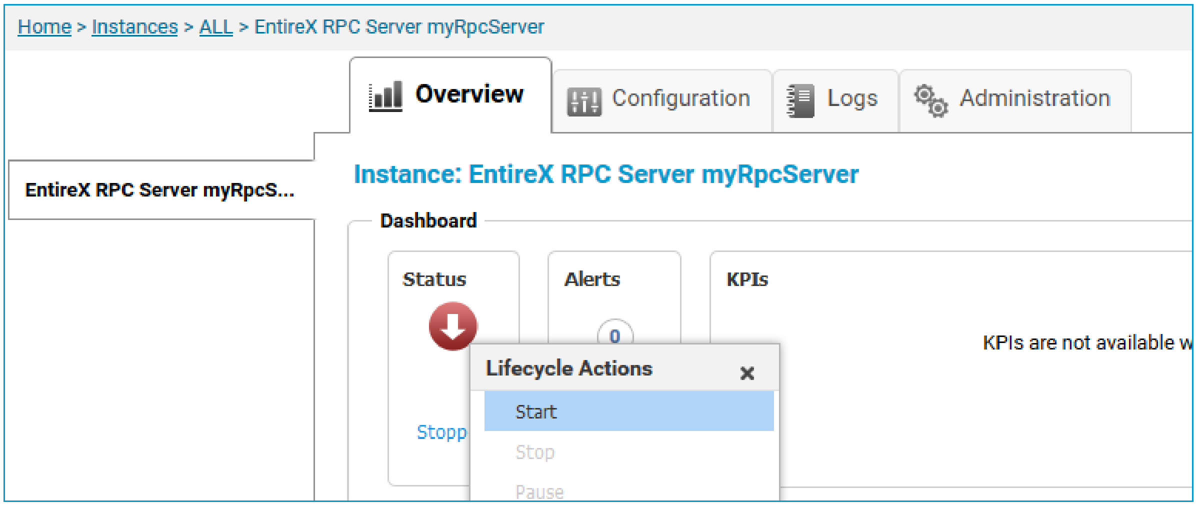


Component	Status	Alerts	Installation	Host
EntireX Broker ETB	⇧		Local	localhost
EntireX RPC Server	⇩		Local	localhost
CCE	⇧			localhost
IS_default	⇧			localhost
SPM	⇧			localhost

2 Select the status, and from the context menu choose **Start**.

To start an RPC Server for CICS IPIC instance from its Overview tab

1 In the Command Central home page, click the **Instances** tab and select the RPC Server for CICS IPIC instance you want to start (same as Step 1 under *Configuring a Broker Instance*).



Home > Instances > ALL > EntireX RPC Server myRpcServer

Instance: EntireX RPC Server myRpcServer

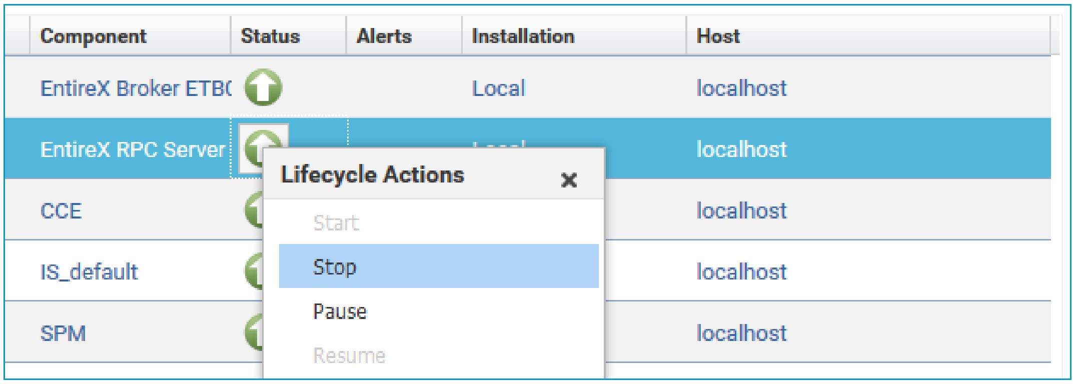
KPIs are not available w

2 Select the status, and from the context menu choose **Start**.

Stopping an RPC Server Instance

To stop an RPC Server for CICS IPIC instance from the Instances tab

1 In the Command Central home page, click the **Instances** tab.

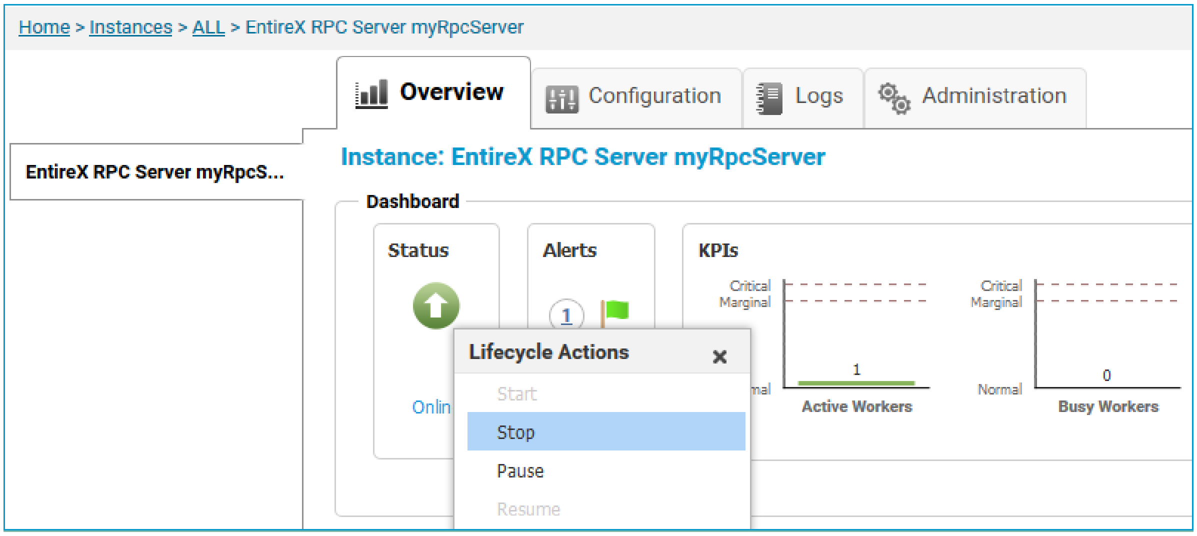


Component	Status	Alerts	Installation	Host
EntireX Broker ETB(	↑		Local	localhost
EntireX RPC Server	↑		Local	localhost
CCE				localhost
IS_default				localhost
SPM				localhost

2 Select the status, and from the context menu choose **Stop**.

To stop an RPC Server for CICS IPIC instance from its Overview tab

1 In the Command Central home page, click the **Instances** tab and select the RPC Server for CICS IPIC instance you want to stop (same as Step 1 under *Configuring a Broker Instance*).



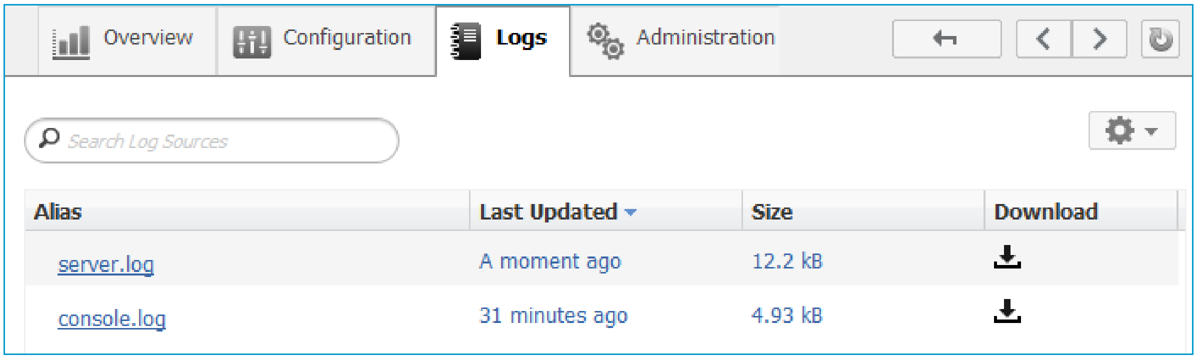
2 Select the status, and from the context menu choose **Stop**.

Inspecting the Log Files

To inspect the log files of an RPC Server for CICS IPIC instance

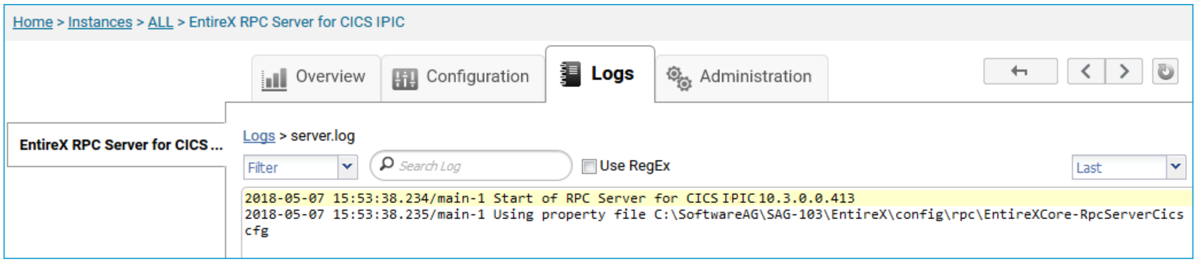
1 In the Command Central home page, click the **Instances** tab, then click the link associated with the RPC Server for CICS IPIC instance for which you want to inspect the log files (same as Step 1 under *Configuring a Broker Instance*).

2 Click the **Logs** tab:



Alias	Last Updated	Size	Download
server.log	A moment ago	12.2 kB	⬇
console.log	31 minutes ago	4.93 kB	⬇

3 In the **Alias** column, click the link of the log file you want to inspect, for example *server.log*:


Home > Instances > ALL > EntireX RPC Server for CICS IPIC
Overview
Configuration
Logs
Administration
EntireX RPC Server for CICS ...
Logs > server.log
Filter
Search Log
Use RegEx
Last

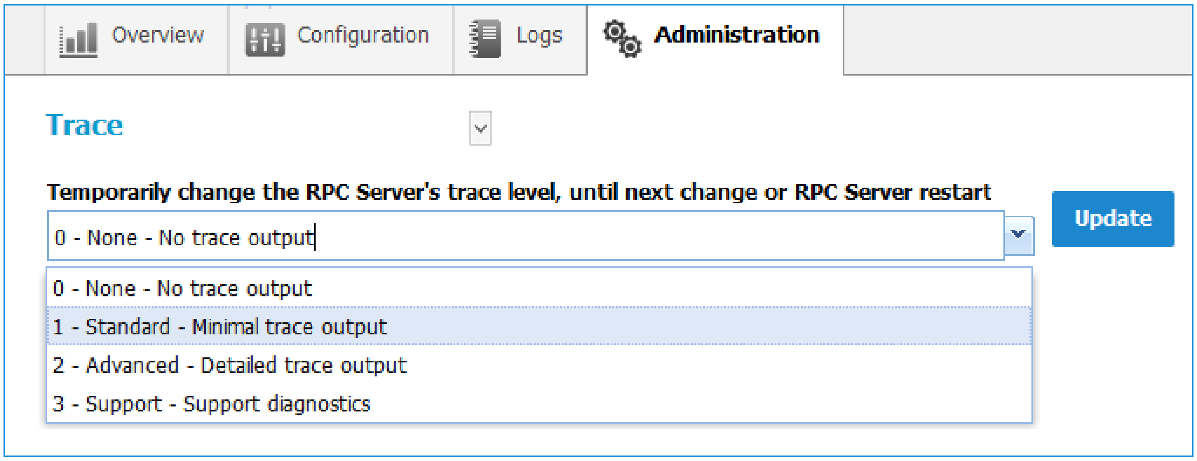
```
2018-05-07 15:53:38.234/main-1 Start of RPC Server for CICS IPIC 10.3.0.0.413
2018-05-07 15:53:38.235/main-1 Using property file C:\SoftwareAG\SAG-103\EntireX\config\rpc\EntireXCore-RpcServerCics
cfg
```

Changing the Trace Level Temporarily

To temporarily change the trace level of an RPC Server for CICS IPIC instance

1 In the Command Central home page, click the **Instances** tab then click the link associated with the RPC Server for CICS IPIC instance for which you want change the trace level temporarily (same as Step 1 under *Configuring a Broker Instance*).

2 In the **Administration** tab, select the trace level and press **Update**.



Temporarily change the RPC Server's trace level, until next change or RPC Server restart

0 - None - No trace output

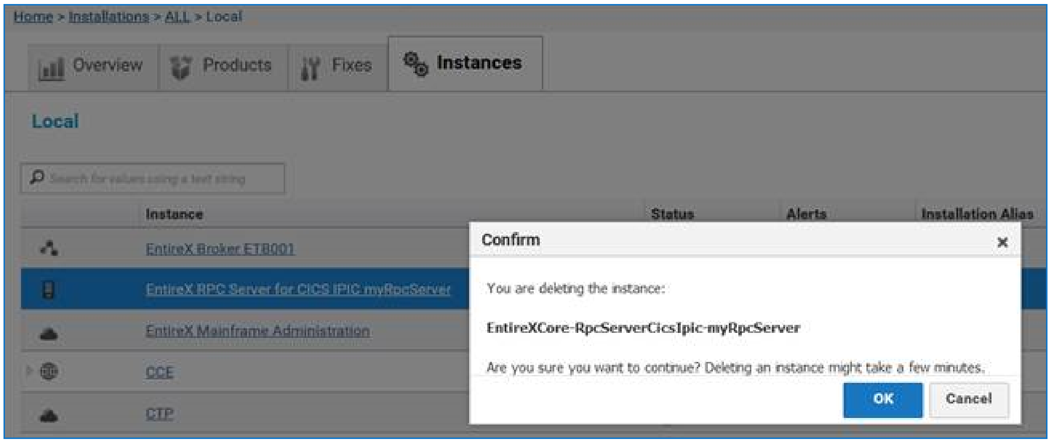
0 - None - No trace output
1 - Standard - Minimal trace output
2 - Advanced - Detailed trace output
3 - Support - Support diagnostics

Note: If you want to set the trace level permanently, see *Trace Level* under *Configuring an RPC Server Instance*.

Deleting an RPC Server Instance

To delete an RPC Server for CICS IPIC instance

1 In the list of EntireX RPC Server for CICS® IPIC instances for your selected installation (for example Local), select the instance you want to delete and click the − button in the upper right corner above the list.


Home > Installations > ALL > Local
Overview
Products
Fixes
Instances
Local
Search for values using a text string
Instance
Status
Alerts
Installation Alias
EntireX Broker ETB001
EntireX RPC Server for CICS IPIC myRpcServer
EntireX Mainframe Administration
CCE
CTP
Confirm
×
OK
Cancel

You are deleting the instance:

EntireXCore-RpcServerCicsIpic-myRpcServer

Are you sure you want to continue? Deleting an instance might take a few minutes.

2 Click **OK** to confirm the uninstall of this RPC Server for CICS IPIC instance.

3 In the next window, click **Finish**. The selected instance is removed from the list.

4 Administering the RPC Server for CICS IPIC using the Command Central Command Line

This chapter describes how to administer the EntireX RPC Server for CICS® IPIC, using the Command Central command-line interface.

Administering the RPC Server for CICS IPIC using the Command Central GUI is described under ***Administering the RPC Server for CICS IPIC using the Command Central GUI***. The core Command Central documentation is provided separately and is also available under **Guides for Tools Shared by Software AG Products** on the Software AG documentation website.

Creating an RPC Server Instance

The following table lists the parameters to include when creating an EntireX RPC instance, using the Command Central `create instances` commands.

Command	Parameter	Value	Opt/ Req	Description
`sagcc create instances`	*`node_alias`*	*`name`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`type`	`RpcServerCicsIpic`	R	Required. EntireXCore instance type of RPC server. Must be "`RpcServerCicsIpic`".
	`product`	`EntireXCore`	R	Required. Must be set to "`EntireXCore`".
	`instance.name`	*`name`*	R	Required. Name of the runtime component, for example "`MyRpcServer`".
	`install.service`	`true` | `false`	O	Optional. Register Windows Service for automatic startup. Default is `false`. If this parameter is `true`, the RPC server can be controlled by the Windows Service Control Manager.
	`server.address`	*`class/server/service`*	R	Required. The case-sensitive RPC server address has the format: `CLASS/SERVER/SERVICE`.
	`server.adminport`	`1025-65535`	R	Required. The administration port in range from 1025 to 65535.
	`broker.transport`	`ssl` | `tcp`	R	Transport over TCP or SSL. Default is `TCP`.
	`broker.host`	*`name`*	R	Required. EntireX Broker host name or IP address. See *Using the Broker ID in Applications* in the RPC Programming documentation.

Command	Parameter	Value	Opt/ Req	Description
	broker.port	1025-65535	R	Required. Port number in range from 1025 to 65535.
	broker.user	*user*	O	Optional. The user ID for secured access to the broker.
	broker.password	*password*	O	Optional. The password for securec access to the broker.
	cics.ipic.host	*name*	R	Host name or IP address where CICS is running.
	cics.ipic.port	1-65535	R	TCP or SSL port number (1-65535) o CICS IPIC.
	cics.ipic.applid	*application_id*	O	Client Application ID (max. 8 chars to match target IPCONN. See your IBM documentation for more information.
	cics.ipic.networkid	*network_id*	O	Client Network ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information
	cics.ipic.transaction	*transaction*	R	Transaction ID (1-4 chars). Name of the CICS mirror transaction that wil receive transactions.
	entirex.bridge.targetencoding	cp037 | *codepage*	R	Specify the appropriate EBCDIC encoding used by your CICS installation. Default is codepage cp037 with full Latin-1 character set
	cics.ipic.userid	*user_id*	O	The user ID (max. 8 chars) as definec in your underlying mainframe security system (e.g. RACF).
	cics.ipic.password	*password*	O	Password/passphrase as defined in your underlying mainframe security system (e.g. RACF). The password or passphrase is encrypted and written to the property cics.ipic.password.e ■ To change the password or passphrase, set the new passworc in the properties file. ■ To disable password, passphrase and/or secured signon key encryption, set entirex.server.passwordencrypt=no

Example

- To create a new instance for an installed EntireX of the type "RpcServerCicsIpic", with name "MyRpcServer", with server address "RPC/SRV1/CALLNAT", using administration port 5757, with broker host name "localhost", listening on broker port 1971, transmitting CICS request with the transport over "tcp", to host "cicsHost", via port "5822", with transaction ID "CPMI", application ID "CICSAPPL", network ID "CICSNETW", coded character set ID "37" and encoding "cp037" and user transaction ID "myId", in the installation with alias name "local":

  ```
  sagcc create instances local EntireXCore type=RpcServerCicsIpic
  instance.name=MyRpcServer server.address=RPC/SRV1/CALLNAT server.adminport=5757
  broker.host=localhost broker.port=1971 cics.sl.transport=TCP cics.sl.host=cicsHost
  cics.sl.port=5822 cics.ipic.transaction=CPMI cics.ipic.applid=CICSAPPL
  cics.ipic.networkid=CICSNETW, entirex.bridge.targetencoding=cp037
  ```

  Information about the creation job - including the job ID - is displayed.

Configuring an RPC Server Instance

Here you can administer the parameters of the RPC Server for CICS IPIC. Any changes to parameters will be used the next time you start the RPC server.

- Broker
- CICS
- Configuration File
- Monitoring KPIs
- Server
- Trace

Broker

Here you can administer the parameters used for communication between the RPC Server for CICS IPIC and EntireX Broker.

- Parameters
- Displaying the Broker Settings of the RPC Server
- Updating the Broker Settings of the RPC Server

Parameters

Parameter	Value	Opt/ Req	Description
`BrokerTransport`	`TCP` | `SSL`	R	Transport over TCP or SSL. Default is `TCP`.
`BrokerHost`	*name*	R	Required. EntireX Broker host name or IP address. See *Using the Broker ID in Applications* in the RPC Programming documentation.
`BrokerPort`	`1025-65535`	R	Required. Port number in range from 1025 to 65535.
`BrokerUser`	*user*	O	Optional. The user ID for secured access to the broker.
`BrokerPassword`	*password*	O	Optional. The password for secured access to the broker.
`BrokerEncoding`	*codepage*	R	Required. Encoding used for the communication between the RPC server and EntireX Broker.
`BrokerSslTrustStore`	*filename*	O	Optional. Specifies the location of the SSL trust store.
`BrokerSslTrustStorePassword`	*password*	O	Optional. The password of the SSL trust store.
`BrokerSslVerifyServer`	`true` | `false`	O	Optional. The RPC server as SSL client checks the identity of the broker as SSL server.

Parameter	Value	Opt/ Req	Description
`BrokerFipsMode`	`yes | no`	O	Optional. Enable FIPS-140 compliant SSL communication. Default is `no`.

Displaying the Broker Settings of the RPC Server

Command	Parameter	Opt/ Req	Description
`sagcc get configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`BROKER`".
	`-o` *file*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the Broker parameters of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
BROKER
```

Example 2

- To store the Broker parameters in the file *broker.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
BROKER -o broker.json
```

Resulting output file in JSON format:

```
{
"BrokerHost":"localhost",
"BrokerPort":"1971",
"BrokerTransport":"TCP",
"BrokerUser":"testuser",
"BrokerPassword":"",
"BrokerEncoding":"Cp1252",
"BrokerSslTrustStore":"",
"BrokerSslVerifyServer":"true"
"BrokerFipsMode":"no"
}
```

Updating the Broker Settings of the RPC Server

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`BROKER`".
	`-i` *file*	O	Optional. Specifies the file from where you want the input read.

Example

- To load the Broker parameters of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" from the file *broker.json* in the current working directory:

  ```
  sagcc update configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
  BROKER -i broker.json
  ```

  See **Example 2** above for sample input file.

CICS

Here you can modify the CICS-specific configuration parameters.

Parameters

Command	Parameter	Opt/ Req	Description
`CicsIpicTransport`	`ssl | tcp`	R	Use TCP or SSL to communicate with CICS IPIC.
`CicsIpicHost`	*name*	R	Host name or IP address where CICS is running.
`CicsIpicPort`	`1-65535`	R	TCP or SSL port number (1-65535) of CICS IPIC.
`CicsIpicTransaction`	*transaction*	R	Transaction ID (1-4 chars). Name of the CICS mirror transaction that will receive transactions.
`CicsIpicEncoding`	`cp037` | *codepage*	R	Specify the appropriate EBCDIC encoding used by your CICS installation. Default is codepage `cp037` with full Latin-1 character set.
`CicsIpicApplicationId`	*application_id*	O	Client Application ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
`CicsIpicNetworkId`	*network_id*	O	Client Network ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
`CicsIpicSslTrustStore`	*file_path*	O	Specifies the location of the SSL trust store.
`CicsIpicSslTrustStorePassword`	*password*	O	The password of the SSL trust store.
`CicsIpicVerifyServer`	`true | false`	O	The RPC server as SSL client checks the identity of CICS as SSL server.
`CicsIpicTimeout`	`20` | *n*	O	Timeout (in seconds) for CICS. Default is 20 seconds.
`CicsIpicUser`	*user*	O	The user ID (max. 8 chars) as defined in your underlying mainframe security system (e.g. RACF).
`CicsIpicPassword`	*password*	O	Password/passphrase as defined in your underlying mainframe security system (e.g. RACF).

Command	Parameter	Opt/ Req	Description
CicsIpicPassTicket [(1)]	true | false	O	Use PassTicket instead of password.
CicsIpicApplicationName	*application_name*	O	Required if PassTicket is to be used instead of a password. Application name (1-8 chars) as defined in your RACF system. This property is ignored if RACF password is set.
CicsIpicSecuredSignonKey	*key_name*	O	Required if PassTicket is to be used instead of a password. Secured signon key as defined in your RACF system. Must be exactly 16 characters long.
CicsIpicSendSessions	1-999	R	Number of simultaneous transactions (1-999) allowed over the CICS connection. See your IBM documentation for more information.

Notes:

1. By default, PassTickets can only be used once during a one-second time interval. They are protected against replay. This limits the workload to one request per second.

   Caution: IBM provides a mechanism to bypass the PassTicket's replay mechanism, allowing higher workloads, but this introduces security risks. The option to bypass the PassTicket's replay mechanism should only be used in secure environments where access to generated PassTickets is limited to a secure or internal network. See your IBM documentation for more information.

Displaying the CICS IPIC Specific Parameters

Command	Parameter	Opt/ Req	Description
sagcc get configuration data	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "EntireXCore-RpcServerCicsIpic-".
	instanceid	R	Required. Must be "CICS-IPIC".
	-o *file*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the CICS IPIC specific parameters of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CICS_IPIC
```

Example 2

- To store the CICS IPIC specific parameters in the file *cics.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CICS_IPIC -o cics.json
```

Resulting output file in JSON format:

```
{
    "CicsIpicTransport": "TCP",
    "CicsIpicHost": "ibm2",
    "CicsIpicPort": "1234",
    "CicsIpicEncoding": "cp037",
    "CicsIpicSslTrustStore": "",
    "CicsIpicSslVerifyServer": "true",
    "CicsIpicTimeout": "20",
    "CicsIpicUser": "",
    "CicsIpicPassTicket": "",
    "CicsIpicApplicationName": "",
    "CicsIpicSecuredSignonKey": "",
    "CicsIpicTransaction": "CPMI",
    "CicsIpicApplId": "CICSAPPL",
    "CicsIpicNetworkId": "CICSNETW"
    }
```

Updating the CICS IPIC Specific Parameters

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`CICS-IPIC`".
	`-i` *file*	O	Optional. Specifies the file from where you want the input read.

Example

- To modify the CICS IPIC parameters, get the file *cics.json* with the `get` command. Edit the parameters in this file, and update the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" with the following command:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CICS_IPIC -i cics.json
```

See **Example 2** above for sample input file.

Configuration File

Here you can administer the configuration file of the RPC Server for CICS IPIC. Any changes will take effect after the next restart.

- Displaying the Content of the RPC Server Configuration File
- Updating the Content of the RPC Server Configuration File

Displaying the Content of the RPC Server Configuration File

Command	Parameter	Opt/ Req	Description
`sagcc get configuration data`	*`node_alias`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	`instanceid`	R	Required. Must be "`CONFIGURATION`".
	`-o` *`file`*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the configuration file of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local":

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CONFIGURATION
```

Example 2

- To store the contents of the configuration file in the text file *configuration.txt* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CONFIGURATION -o configuration.txt
```

Updating the Content of the RPC Server Configuration File

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`CONFIGURATION`".
	`-i` *file*	O	Optional. Specifies the file from where you want the input read.

Example

- To load the contents of configuration file *configuration.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
CONFIGURATION -i configuration.json
```

Monitoring KPIs

Here you can administer margins of monitored key performance indicators (KPIs) available for the RPC Server for CICS IPIC: Active Workers and Busy Workers.

Parameters

Key performance indicators (KPIs) enable you to monitor the health of your RPC Server for CICS IPIC. The following KPIs help you administer, troubleshoot, and resolve performance issues:

KPI	Setting
Absolute number of Active Workers	`entirex.generic.kpi.1.max=20`
Critical alert relative to maximum	`entirex.generic.kpi.1.critical=0.95`
Marginal alert relative to maximum	`entirex.generic.kpi.1.marginal=0.80`
Absolute number of Busy Workers	`entirex.generic.kpi.2.max=20`
Critical alert relative to maximum	`entirex.generic.kpi.2.critical=0.95`
Marginal alert relative to maximum	`entirex.generic.kpi.2.marginal=0.80`

Do not change the other properties!

Displaying the Monitoring KPIs

Command	Parameter	Opt/ Req	Description
`sagcc get configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`EXX-MONITORING-KPIS`".
	`-o` *file*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the monitoring KPI properties of RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
MONITORING-KPI
```

Example 2

- To store the monitoring KPI properties in the file *my.properties* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
MONITORING-KPI -o my.properties
```

Resulting output file in text format:

```
entirex.entirex.spm.version=10.9.0.0.473
entirex.generic.kpi.1.critical=0.95
entirex.generic.kpi.1.id=\#1
entirex.generic.kpi.1.marginal=0.80
entirex.generic.kpi.1.max=20
entirex.generic.kpi.1.name=Active Workers
entirex.generic.kpi.1.unit=
entirex.generic.kpi.1.value=0
entirex.generic.kpi.2.critical=0.95
entirex.generic.kpi.2.id=\#2
entirex.generic.kpi.2.marginal=0.80
entirex.generic.kpi.2.max=20
entirex.generic.kpi.2.name=Busy Workers
entirex.generic.kpi.2.unit=
entirex.generic.kpi.2.value=0
```

Updating the Monitoring KPIs

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`EXX-MONITORING-KPIS`".
	`-i` *file*	O	Optional. Specifies the file from where you want the input read.

Example

- To load the contents of file *my.properties* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
MONITORING-KPI -i my.properties
```

Server

Here you can administer the parameters defining the registration name, the administration port and the behavior of the RPC Server for CICS IPIC.

- Parameters
- Displaying the Server Settings
- Updating the Server Settings

Parameters

Parameter	Value	Opt/ Req	Description
ServerAddress	*class/server/service*	R	Required. The case-sensitive RPC server address has the format: CLASS/SERVER/SERVICE.
ServerAdminport	1025-65535	R	Required. The administration port in range from 1025 to 65535.
ReconnectionAttempts	*n*	R	Required. Number of reconnection attempts to the broker. When the number of attempts is reached and a connection to the broker is not possible, the RPC Server stops.
WorkerScalability	true | *false*	R	You can either have a fixed or dynamic number of workers. Default is dynamic (true). For more information see ***Worker Models***.
FixNumber	1-255	R	Required. Fixed number of workers. Must be a number in range from 1 to 255.
MinWorkers	1-255	R	Required. Minimum number of workers. Must be a number in range from 1 to 255.
MaxWorkers	1-255	R	Required. Maximum number of workers. Must be a number in range from 1 to 255.

Displaying the Server Settings

Command	Parameter	Opt/ Req	Description
sagcc get configuration data	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "EntireXCore-RpcServerCicsIpic-".
	instanceid	R	Required. Must be "SERVER".
	-o *file*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the server parameters of RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
SERVER
```

Example 2

- To store the server parameters in the file *server.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
SERVER -o server.json
```

Resulting output file in JSON format:

```
{
"ServerAddress":"RPC/SRV1/CALLNAT",
"ServerAdminport":"4711",
"ReconnectionAttempts":"15",
"WorkerScalability":"true",
"FixNumber":"5",
"MinWorkers":"1",
"MaxWorkers":"10"
}
```

Updating the Server Settings

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`SERVER`".
	`-i` *file*	O	Optional. Specifies the file from where you want the input read.

Example

- To load the server parameters from the file *server.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
SERVER -i server.json
```

See **Example 2** above for sample input file.

Trace

Here you can set the trace level of the RPC Server for CICS IPIC.

- Parameters
- Displaying the Trace Level
- Updating the Trace Level

Parameters

Command	Parameter	Opt/ Req	Description
`TraceLevel`	`0 | 1 | 2 | 3`	R	One of the following levels: 0 - None - No trace output (default). 1 - Standard - Minimal trace output. 2 - Advanced - Detailed trace output. 3 - Support - Support diagnostic. Use only when requested by Software AG Support.

Displaying the Trace Level

Command	Parameter	Opt/ Req	Description
`sagcc get configuration data`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	instanceid	R	Required. Must be "`TRACE`".
	`-o` *file*	O	Optional. Specifies the file where you want the output written.

Example 1

- To display the trace level of RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
TRACE
```

Example 2

- To store the trace level in the file *trace.json* in the current working directory:

```
sagcc get configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
TRACE -o trace.json
```

Resulting output file in JSON format:

```
{
"TraceLevel":"0"
}
```

Updating the Trace Level

Command	Parameter	Opt/ Req	Description
`sagcc update configuration data`	*`node_alias`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	`instanceid`	R	Required. Must be "`TRACE`".
	`-i` *`file`*	O	Optional. Specifies the file from where you want the input read.

Example

- To load the trace level parameters from the file *trace.json* in the current working directory:

```
sagcc update configuration data local EntireXCore-RpcServerCicsIpic-MyRpcServer
TRACE -i trace.json
```

See **Example 2** above for sample input file.

Displaying the EntireX Inventory

Listing all Inventory Components

The following table lists the parameters to include, when listing all EntireX instances, using the Command Central `list inventory` commands.

Command	Parameter	Opt/ Req	Description
`sagcc list inventory components`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".

Example

- To list inventory components of instance EntireX in the installation with alias name "local":

```
sagcc list inventory components local EntireXCore*
```

A list of all EntireX RPC Server runtime components will be displayed.

Viewing the Runtime Status

The following table lists the parameters to include when displaying the state of an EntireX component, using the Command Central `get monitoring` commands.

Command	Parameter	Opt/ Req	Description
`sagcc get monitoring state`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is `"EntireXCore-RpcServerCicsIpic-"`.

Example

- To display state information about the RPC Server for CICS IPIC:

```
sagcc get monitoring state local EntireXCore-RpcServerCicsIpic-MyRpcServer
```

Runtime status and runtime state will be displayed.

- Runtime *status* indicates whether a runtime component is running or not. Examples of a runtime status are `ONLINE` or `STOPPED`.
- Runtime *state* indicates the health of a runtime component by providing key performance indicators (KPIs) for the component. Each KPI provides information about the current use, marginal use, critical use and maximum use.

Starting an RPC Server Instance

The following table lists the parameters to include when starting an EntireX RPC Server for CICS® IPIC, using the Command Central `exec lifecycle` commands.

Command	Parameter	Opt/ Req	Description
`sagcc exec lifecycle start`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".

Example

- To start the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local":

```
sagcc exec lifecycle start local EntireXCore-RpcServerCicsIpic-MyRpcServer
```

Information about the job - including the job ID - will be displayed.

Stopping an RPC Server Instance

The following table lists the parameters to include when stopping an EntireX RPC Server for CICS® IPIC, using the Command Central `exec lifecycle` commands.

Command	Parameter	Opt/ Req	Description
`sagcc exec lifecycle stop`	*node_alias*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	componentid	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".

Example

- To stop the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local":

```
sagcc exec lifecycle stop local EntireXCore-RpcServerCicsIpic-MyRpcServer
```

Information about the job - including the job ID - will be displayed.

Inspecting the Log Files

Here you can administer the log files of the RPC Server for CICS IPIC. The following table lists the parameters to include when displaying or modifying parameters of the RPC server, using the Command Central `list` commands.

- List all RPC Server Log Files
- Getting Content from or Downloading RPC Server Log Files

List all RPC Server Log Files

Command	Parameter	Opt/ Req	Description
`sagcc list diagnostics logs`	*`node_alias`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".

Example

- To list the log files of RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" on stdout:

```
sagcc list diagnostics logs local EntireXCore-RpcServerCicsIpic-MyRpcServer
```

Getting Content from or Downloading RPC Server Log Files

Command	Parameter	Opt/ Req	Description
`sagcc get diagnostics logs`	*`node_alias`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	`full | tail | head`	O	Optional. Shows full log file content, or only tail or head.
	`export -o` *`file`*	O	Optional. Creates a zip file of the logs.

Example 1

- To list the tail of the log file content in the current working directory:

```
sagcc get diagnostics logs local EntireXCore-RpcServerCicsIpic-MyRpcServer
server.log tail
```

Example 2

- To create a zip file *myfile.zip* of the logs:

```
sagcc get diagnostics logs local EntireXCore-RpcServerCicsIpic-MyRpcServer export
-o myfile.zip
```

Changing the Trace Level Temporarily

Here you can temporarily change the trace level of a running RPC server. The following table lists the parameters to include when displaying or modifying parameters of an EntireX component, using the Command Central `exec administration` command. The change is effective immediately; there is no need to restart the RPC server.

Note: If you want to set the trace level permanently, see *Trace* under *Configuring an RPC Server Instance*.

Displaying the Trace Level of a Running RPC Server

Command	Parameter	Opt/ Req	Description
`sagcc exec administration`	`component`	R	Required. Specifies that a component will be administered.
	`node_alias`	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`Trace`	R	Required. Specifies what is to be administered.
	`load tracelevel=?`	R	Required. Get the trace level.
	`-f xml|json`	R	Required. Specifies XML or JSON as output format.

Example 1

- To display the current trace level of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" in JSON format on stdout:

```
sagcc exec administration component local EntireXCore-RpcServerCicsIpic-MyRpcServer
Trace load tracelevel=? -f json
```

Example 2

- To display the current trace level of the RPC Server for CICS IPIC "MyRpcServer" in the installation with alias name "local" in XML format on stdout:

```
sagcc exec administration component local EntireXCore-RpcServerCicsIpic-MyRpcServer
Trace load tracelevel=? -f xml
```

Updating the Trace Level of a Running RPC Server

Command	Parameter	Opt/ Req	Description
`sagcc exec administration`	`component`	R	Required. Specifies that a component will be administered.
	`node_alias`	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".
	`Trace`	R	Required. Specifies what is to be administered.
	`update tracelevel`	R	Required. Update temporarily the trace level of a running RPC server.
	`-f xml|json`	R	Required. Specifies XML or JSON as output format.

Example

- To change the current trace level of the running RPC Server with the name "MyRpcServer" in the installation with alias name "local":

```
sagcc exec administration component local EntireXCore-RpcServerCicsIpic-MyRpcServer
Trace update tracelevel=2 -f json
```

Deleting an RPC Server Instance

The following table lists the parameters to include when deleting an EntireX RPC Server instance, using the Command Central `delete instances` commands.

Command	Parameter	Opt/ Req	Description
`sagcc delete instances`	*`node_alias`*	R	Required. Specifies the alias name of the installation in which the runtime component is installed.
	`componentid`	R	Required. The component identifier. The prefix is "`EntireXCore-RpcServerCicsIpic-`".

Example

- To delete an instance of an EntireX RPC Server for CICS® IPIC with the name "MyRpcServer" in the installation with alias name "local":

```
sagcc delete instances local EntireXCore-RpcServerCicsIpic-MyRpcServer
```

Information about the deletion job - including the job ID - is displayed.

5 Administering the RPC Server for CICS IPIC with Local Scripts

The EntireX RPC Server for CICS® IPIC allows standard RPC clients to communicate with CICS programs running on IBM CICS® without installing EntireX components on the mainframe (Zero Footprint). The CICS interface types Channel Container and DFHCOMMAREA are supported.

Customizing the RPC Server

The following are used to set up the RPC Server for CICS IPIC:

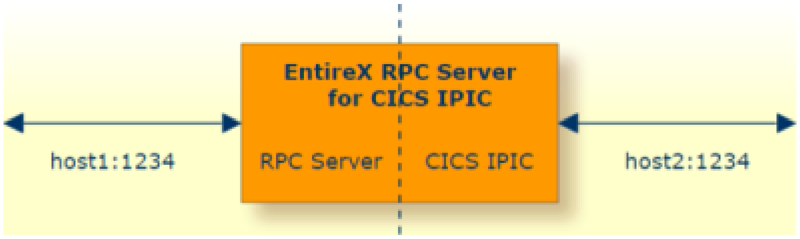
- Configuration File
- Start Script

Configuration File

The default name of the configuration file is *entirex.cicsipic.properties*. The RPC Server for CICS IPIC searches for this file in the current working directory.

You can set the name of the configuration file with `-Dentirex.server.properties=<your file name>` with "/" as file separator.

The configuration file contains the configuration for both parts of the RPC Server for CICS IPIC.



Configuring more than one RPC Server

If you configure more than one RPC Server for CICS IPIC that connect to the same broker, the following items must be distinct:

- the trace output file (property `entirex.server.logfile`)
- the log for the Windows Service (property `entirex.server.serverlog`)

Start Script

The start script for the RPC Server for CICS IPIC is called *cicsipicserver.bsh* (Linux) or *cicsipicserver.bat* (Windows) and is provided in the *bin* folder of the installation directory. You may customize this file.

Configuring the RPC Server Side

The RPC Server for CICS IPIC uses the properties that start with "`entirex.server`" for configuring the RPC server side.

Alternatively to the properties, you can use the command-line options. These have a higher priority than the properties set as Java system properties, and these have higher priority than the properties in the configuration file.

Property Name	Command-line Option	Default	Explanation
`entirex.server.brokerid`	`-broker`	`localhost`	Broker ID.
`entirex.server.serveraddress`	`-server`	`RPC/SRV1/CALLNAT`	Server address.
`entirex.server.userid`	`-user`	`RPCServer`	The user ID for access to the broker.
`entirex.server.fixedservers`		`no`	`NO` The number of worker threads balances between what is specified in `entirex.server.minservers` and what is specified in `entirex.server.maxservers`. This is done by a so-called attach thread. At startup, the number of worker threads is the number specified in `entirex.server.minservers`. A new worker thread starts if the broker has more requests than there are worker threads waiting. If more than the number specified in `entirex.server.minservers` are waiting for requests, a worker thread stops if its receive call times out. The timeout period is configured with `entirex.server.waitserver`. See worker model `DYNAMIC`. `YES` The number of worker threads specified in `entirex.server.minservers` is started and the server can process this number of parallel requests. See worker model `FIXED`.
`entirex.server.minservers`		`1`	Minimum number of server threads.

Property Name	Command-line Option	Default	Explanation
`entirex.server.maxservers`		`32`	Maximum number of server threads.
`entirex.server.restartcycles`	`-restartcycles`	`15`	Number of restart attempts if the Broker is not available. This can be used to keep the RPC Server for CICS IPIC running while the Broker is down for a short time.
`entirex.server.password`	`-password`	`yes`	The password for secured access to the broker. The password is encrypted and written to the property `entirex.server.password.e`. ■ To change the password, set the new password in the properties file. ■ To disable password encryption, set `entirex.server.passwordencrypt=no`. Default=`yes`.
`entirex.server.security`	`-security`	`no`	Valid values: `no` | `yes` | `auto` | name of `BrokerSecurity` object.

`entirex.server.compresslevel`	`-compresslevel`	`0` (no compression)	Enter the text or the numeric value:
			`BEST_COMPRESSION` `9`
			`BEST_SPEED` `1`
			`DEFAULT_COMPRESSION` `-1` (*mapped to* `6`)
			`DEFLATED` `8`
			`NO_COMPRESSION` `0`
			`N` `0`
			`Y` `8`

`entirex.server.waitattach`		`600S`	Wait timeout for the attach server thread.
`entirex.server.waitserver`		`300S`	Wait timeout for the worker threads.
`entirex.timeout`		`20`	TCP/IP transport timeout.
	`-help`		Display usage of the command-line parameters.
`entirex.server.logfile`	`-logfile`	standard output	Name of the log file.

Property Name	Command-line Option	Default	Explanation
`entirex.trace`	`-trace`	0	Trace level. 0 No tracing, default. 1 Trace all broker calls and other major actions. 2 Dump the send and receive buffer. 3 Dump the buffers sent to the broker and received from the broker.

Configuring the CICS IPIC Side

These properties are used to configure the connection to CICS IPIC.

Alternatively, you can use the command-line options. These have a higher priority than the properties set as Java system properties, and these have higher priority than the properties in the configuration file.

Name	Default Value	Explanation
`cics.ipic.host`		Host name or IP address where CICS is running.
`cics.ipic.port`		TCP or SSL port number (1-65535) of CICS IPIC.
`cics.ipic.transaction`	`CPMI`	Transaction ID (1-4 chars). Name of the CICS mirror transaction that will receive transactions.
`cics.ipic.networkid`		Client Network ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
`cics.ipic.applid`		Client Application ID (max. 8 chars) to match target IPCONN. See your IBM documentation for more information.
`cics.ipic.sendsessions`	`100`	Number of simultaneous transactions (1-999) allowed over the CICS connection. See your IBM documentation for more information.
`entirex.bridge.targetencoding`	`cp037`	Specify the appropriate EBCDIC encoding used by your CICS installation. Default is codepage `cp037` with full Latin-1 character set.
`cics.ipic.sockettimeout`	`10000`	Socket timeout for connection to CICS IPIC (in milliseconds).
`cics.ipic.userid`		The user ID (max. 8 chars) as defined in your underlying mainframe security system (e.g. RACF).
`cics.ipic.password`	`yes`	Password/passphrase as defined in your underlying mainframe security system (e.g. RACF). The password or passphrase is encrypted and written to the property `cics.ipic.password.e`. ■ To change the password or passphrase, set the new password in the properties file. ■ To disable password, passphrase and/or secured signon key encryption, set entirex.server.passwordencrypt=no.
`cics.ipic.sslparams`		SSL parameters. Same syntax as Broker ID.

Name	Default Value	Explanation
`cics.ipic.application.name` [1]		Required if PassTicket is to be used instead of a password. Application name (1-8 chars) as defined in your RACF system. This property is ignored if RACF password is set.
`cics.ipic.secured.signonkey` [1]	`yes`	Required if PassTicket is to be used instead of a password. Secured signon key as defined in your RACF system. Must be exactly 16 characters long. The secured signon key is encrypted and written to the property `cics.slipic.secured.signonkey.e`. - To change the secured signon key, set the new secured signon key in the properties file. - To disable password, passphrase and/or secured signon key encryption, set entirex.server.passwordencrypt=no. This property is ignored if `cics.ipic.password` is set.

Notes:

1. By default, PassTickets can only be used once during a one-second time interval. They are protected against replay. This limits the workload to one request per second.

   Caution: IBM provides a mechanism to bypass the PassTicket's replay mechanism, allowing higher workloads, but this introduces security risks. The option to bypass the PassTicket's replay mechanism should only be used in secure environments where access to generated PassTickets is limited to a secure or internal network. See your IBM documentation for more information.

Using SSL/TLS with the RPC Server

To use SSL with the RPC Server for CICS IPIC, you need to configure two sides:

- **CICS Side**
  See parameter *`cics.sl.sslparams`*.
- **RPC Server Side**
  RPC servers can use Secure Sockets Layer/Transport Layer Security (SSL/TLS) as the transport medium. The term “SSL” in this section refers to both SSL and TLS. RPC-based servers are always SSL clients. The SSL server can be either the EntireX Broker or Broker SSL Agent. For an introduction see *SSL/TLS, HTTP(S), and Certificates with EntireX* in the platform-independent Administration documentation.

To use SSL

1. To operate with SSL, certificates need to be provided and maintained. Depending on the platform, Software AG provides sample certificates, but we strongly recommend that you create your own. See *SSL/TLS Sample Certificates Delivered with EntireX* in the EntireX Security documentation.

2. Set up the RPC Server for CICS IPIC for an SSL connection.

   Use the *URL-style Broker ID* with protocol `ssl://` for the Broker ID. If no port number is specified, port 1958 is used as default. Example:

   ```
   ssl://localhost:22101?trust_store=C:\SoftwareAG\EntireX\etc\ExxCACert.p12&trust_passwd=ExxCACert&verify_server=no
   ```

   If the SSL client checks the validity of the SSL server only, this is known as *one-way SSL*. Two SSL parameters must be specified on the SSL client side: `trust_store` and `trust_passwd`. The mandatory `trust_store` parameter specifies the file name of a PKCS#12 certificate store that must contain the certificate chain of the trusted certificate authority (CA) that issued the SSL server's certificate.
   To unlock this certificate store, the password has to be set with SSL parameter `trust_passwd`. By default a check is made that the certificate of the SSL server is issued for the hostname specified in the Broker ID. The common name of the subject entry in the server's certificate is checked against the hostname. If they do not match, the connection will be refused. You can disable this check with SSL parameter `verify_server=no`.

   If the SSL server additionally checks the identity of the SSL client, this is known as *two-way SSL*. In this case the SSL server requests a client certificate (the parameter `verify_client=yes` is defined in the configuration of the SSL server). Two additional SSL parameters must be specified on the SSL client side: `key_store` and `key_passwd`. This keystore must contain the private key of the SSL client. The password that protects the private key is specified with `key_passwd`.

   The ampersand (&) character cannot appear in the password.

   SSL parameters are separated by ampersand (&). See also *SSL/TLS Parameters for SSL Clients*.

3. Make sure the SSL server to which the RPC side connects is prepared for SSL connections as well. The SSL server can be EntireX Broker or Broker SSL Agent. See:

   - *Running Broker with SSL/TLS Transport* in the platform-specific Administration documentation
   - Broker SSL Agent in the platform-specific Administration documentation

Starting the RPC Server

To start the RPC Server for CICS IPIC

- Use the *Start Script*.

Stopping the RPC Server

To stop the RPC Server for CICS IPIC

- Use the command `stopService`. See *Stop Running Services* in Command Central's Command-line Interface.

  Or:

  Stop the service using Command Central's Graphical User Interface. See *Stopping a Service*.

  Or:

  Use the command-line utility `etbcmd`. See `ETBCMD` under *Broker Command-line Utilities* in the platform-specific Administration documentation.

  Or:

  Use `CTRL-C` in the session where you started the RPC server instance.

  Or:

  Under Linux, enter command `kill -process-id`.

Pinging the RPC Server

To ping the RPC Server for CICS IPIC

- Enter the following command:

```
java -classpath "$EXXDIR/classes/entirex.jar" ↩
com.softwareag.entirex.rpcping.RPCServerPing -p <admin_port>
```

where `admin_port` is the number of the administration port.

The `ping` command returns "`0`" if the server is reachable, and "`1`" if the server cannot be accessed.

Running an EntireX RPC Server as a Windows Service

For general information see *Running an EntireX RPC Server as a Windows Service* in the Windows Administration documentation.

To run the RPC Server for CICS IPIC as a Windows Service

1 Customize the ***Start Script*** according to your system installation.

See also ***Starting the RPC Server***.

2 Test your RPC server to see whether it will start if you run your script file.

3 Use the *EntireX RPC Service Tool* and install the `RPCService` with some meaningful extension, for example `MyServer`. If your ***Start Script*** is *cicsipicserver.bat*, the command will be

```
RPCService -install -ext MyServer ↩
-script install_path\EntireX\bin\cicsipicserver.bat
```

The log file will be called *RPCservice_MyServer.log*.

4 In **Windows Services** menu (**Control Panel** > **Administrative Tools** > **Services**) select the service: `Software AG EntireX RPC Service [MyServer]` and change the property `Startup Type` from "Manual" to "Automatic".

Application Identification

The application identification is sent from the RPC Server for CICS IPIC to the Broker. It is visible with Broker Command and Information Services.

The identification consists of four parts: name, node, type, and version. These four parts are sent with each Broker call and are visible in the trace information.

For the RPC Server for CICS IPIC, these values are:

Identification Part	Value
Application name	`ANAME=RPC Server for CICS IPIC`
Node name	`ANODE=<`*`host name`*`>`
Application type	`ATYPE=Java`
Version	`AVERS=10.9.0.0`

6 Preparing IBM CICS for IPIC

This chapter describes how to configure an autoinstalled IPCONN for your IPIC connection. The most common scenarios are described in the IBM documentation of CICS Transaction Gateway for Multiplatforms:

⚠ **Important:** If the terms and concepts in this chapter are unfamiliar to you, ask an appropriate CICS system programmer. Only authorized personnel should make changes to mainframe computer systems.

Secure Autoinstalled IPIC Connection

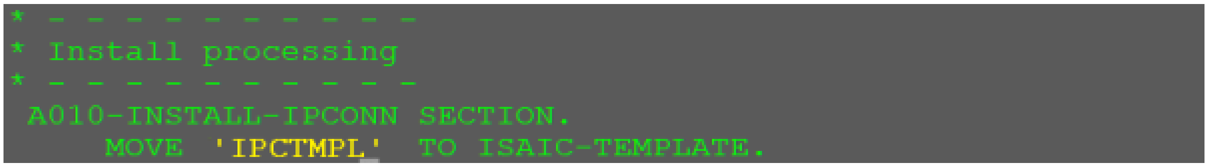
This section describes how to configure a secure autoinstalled IPCONN for your IPIC connection (similar to SC01 scenario in CICS Transaction Gateway for Multiplatforms). With this configuration, the CICS endpoint checks the security.

- Configuring CICS
- Installation Verification CICS
- Off-host Configuration

Configuring CICS

In this configuration you define a TCPIPSERVICE, a template IPCONN and provide an IPCONN autoinstall user program in CICS.

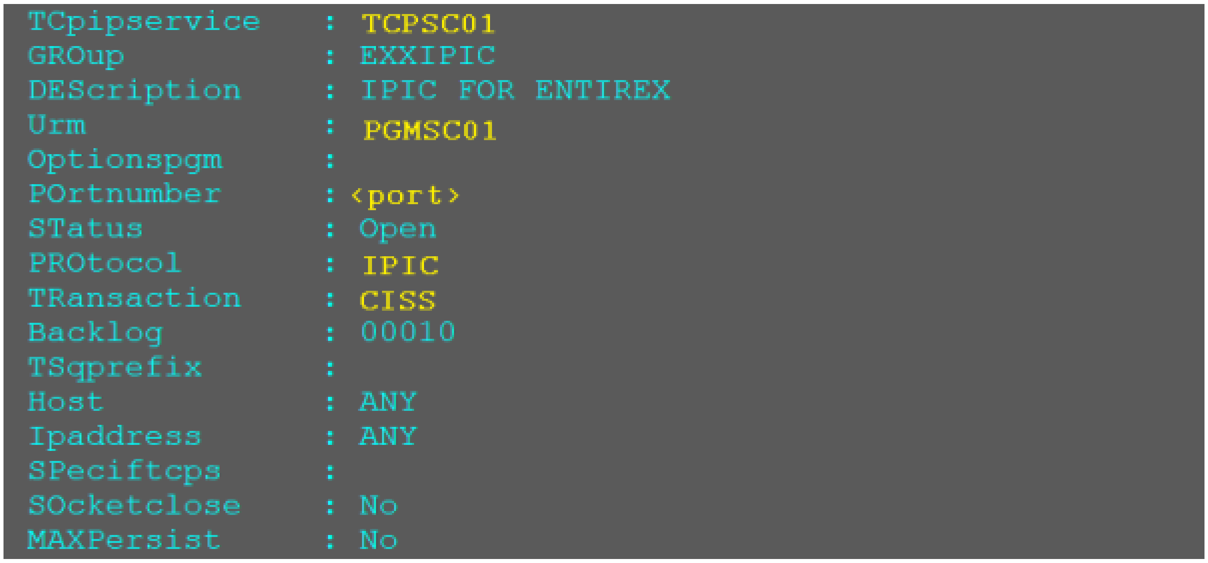
- To provide the IPCONN autoinstall user program, use the CICS sample program DFHISCIP (Note 1) and add in the A010-INSTALL-IPCONN section a MOVE statement telling CICS the name of your IPCONN template.



```
* - - - - - - - - - -
* Install processing
* - - - - - - - - - -
 A010-INSTALL-IPCONN SECTION.
     MOVE 'IPCTMPL'  TO ISAIC-TEMPLATE.
```

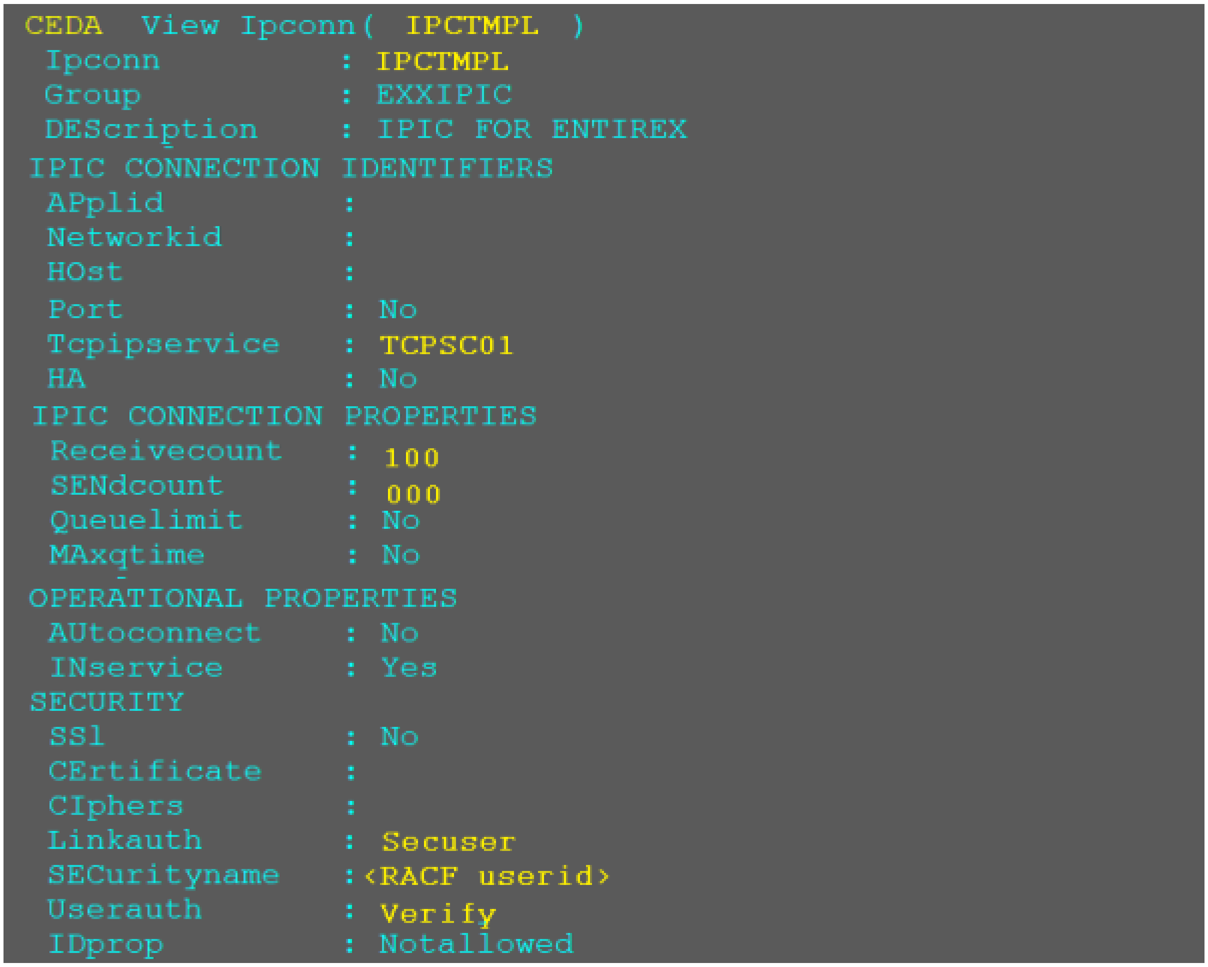
  Compile, link-edit (giving a module name of your choice e.g. PGMSC01) and provide it as any other CICS program in the CICS RPL chain. CEDA DEFINE and CEDA INSTALL it.

- In the TCPIPSERVICE resource, define Urm to point to your IPCONN autoinstall user program, set the POrtnmuber, PROtocol to IPIC and TRansaction to CISS. CEDA INSTALL the TCPIPSERVICE.



```
TCpipservice    : TCPSC01
GROup           : EXXIPIC
DEScription     : IPIC FOR ENTIREX
Urm             : PGMSC01
Optionspgm      :
POrtnumber      : <port>
STatus          : Open
PROtocol        : IPIC
TRansaction     : CISS
Backlog         : 00010
TSqprefix       :
Host            : ANY
Ipaddress       : ANY
SPeciftcps      :
SOcketclose     : No
MAXPersist      : No
```

- Define an `IPCONN` resource as a template. Set `Tcpipservice`, `Receivecount` to the number of parallel `IPCONN` sessions, `SENdcount` to zero, `INservice` to yes. For Security set `SECurityname` to a RACF user ID authorized to establish IPIC connections, `Linkauth` to `Secuser` and `Userauth` to `Verify`. `CEDA INSTALL` the `IPCONN` resource.



```
CEDA  View Ipconn(  IPCTMPL  )
 Ipconn              : IPCTMPL
 Group               : EXXIPIC
 DEScription         : IPIC FOR ENTIREX
IPIC CONNECTION IDENTIFIERS
 APplid              :
 Networkid           :
 HOst                :
 Port                : No
 Tcpipservice        : TCPSC01
 HA                  : No
IPIC CONNECTION PROPERTIES
 Receivecount        : 100
 SENdcount           : 000
 Queuelimit          : No
 MAxqtime            : No
OPERATIONAL PROPERTIES
 AUtoconnect         : No
 INservice           : Yes
SECURITY
 SSl                 : No
 CErtificate         :
 CIphers             :
 Linkauth            : Secuser
 SECurityname        :<RACF userid>
 Userauth            : Verify
 IDprop              : Notallowed
```

Notes:

1. `DFHISCIP` is written in COBOL. There are also CICS samples available for PL/I, C and Assembler. Refer to your IBM documentation.

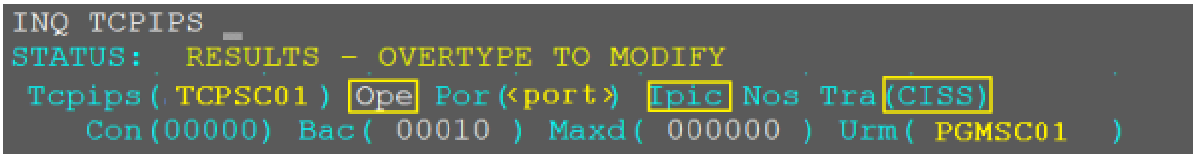
Installation Verification CICS

To verify the installation in CICS

1 Check TCP/IP with command

```
CEMT INQ TCPIPS
```

2 Check status is open, the port number is correct, protocol is `IPIC`, and `Urm` shows your `IPCONN` autoinstall program.

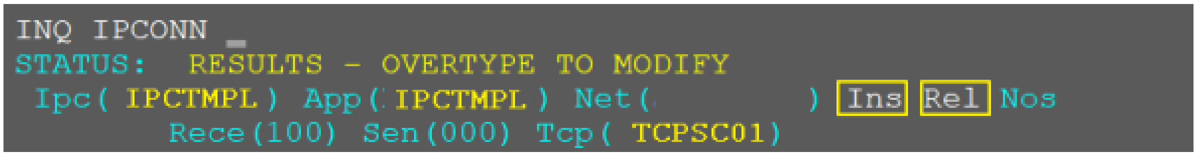


```
INQ TCPIPS _
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Tcpips( TCPSC01 ) Ope Por(<port>) Ipic Nos Tra(CISS)
    Con(00000) Bac( 00010 ) Maxd( 000000 ) Urm( PGMSC01    )
```

3 Check `IPCONN` with command

```
CEMT INQ IPCONN
```

4 Check the `IPCONN` attributes are `INService` and `RELeased`.



```
INQ IPCONN _
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Ipc( IPCTMPL ) App( IPCTMPL ) Net(         ) Ins Rel Nos
         Rece(100) Sen(000) Tcp( TCPSC01)
```

Off-host Configuration

The parameters for `APPLID` and `NETWORKID` should be left empty in your off-host configuration.

RACF Passwords, RACF Passphrases and RACF PassTickets are supported. For details see *CICS IPIC Configuration* using Command Central (GUI | Command Line), or using Local Scripts.

Autoinstalled IPIC Connection

This section describes how to configure autoinstalled IPIC connections *without* security implemented in the CICS endpoint. It uses the default connection settings for IPIC autoinstalled connections (similar to `SC08` scenario in CICS Transaction Gateway for Multiplatforms). With this configuration, it is assumed that the EntireX Adapter or RPC Server for CICS IPIC is running in the green zone (protected network) and a calling instance has already successfully passed security checks. It covers the following topics:

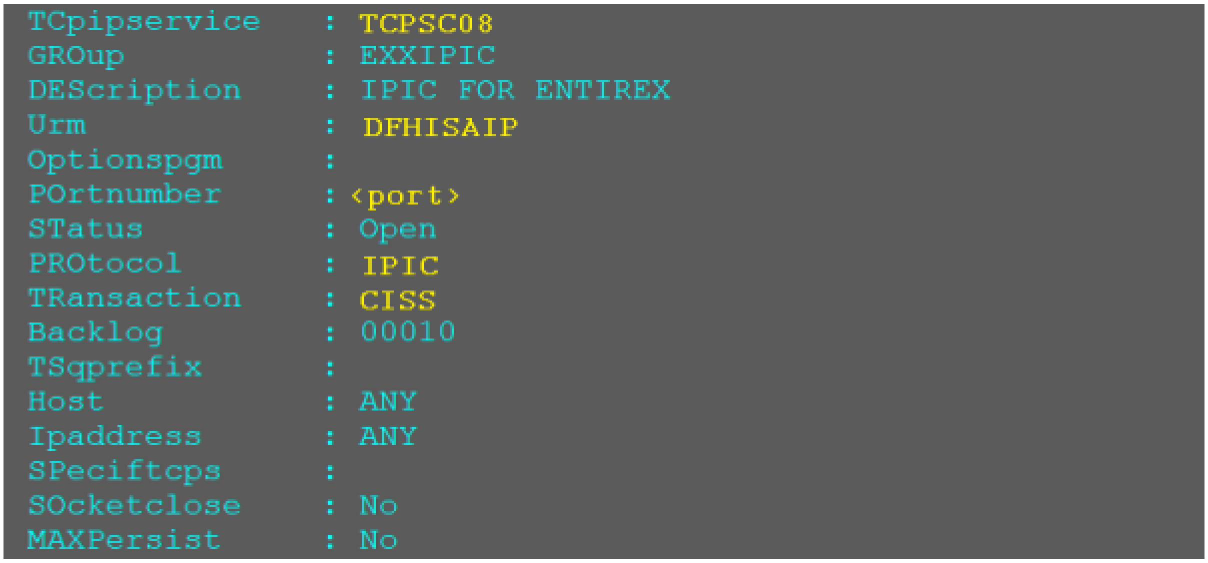
- Configuring CICS
- Installation Verification CICS

- Off-host Configuration

Configuring CICS

In this configuration you define a CICS `TCPIPSERVICE` resource only.

- In the `TCPIPSERVICE` resource, define `Urm` to point to `DFHISAIP`, set the port, PROTOCOL to `IPIC` and transaction to `CISS`. `CEDA INSTALL` the `TCPIPSERVICE`.



```
TCpipservice     : TCPSC08
GROup            : EXXIPIC
DEScription      : IPIC FOR ENTIREX
Urm              : DFHISAIP
Optionspgm       :
POrtnumber       :<port>
STatus           : Open
PROtocol         : IPIC
TRansaction      : CISS
Backlog          : 00010
TSqprefix        :
Host             : ANY
Ipaddress        : ANY
SPeciftcps       :
SOcketclose      : No
MAXPersist       : No
```

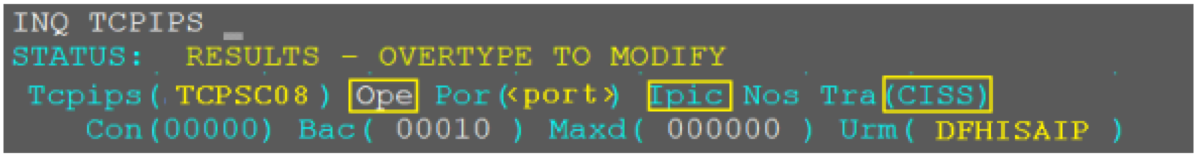
Installation Verification CICS

To verify the installation in CICS

1. Check TCP/IP with command

   ```
   CEMT INQ TCPIPS
   ```

2. Check the status is open, the port number is correct, protocol is `IPIC`, and `Urm` shows program `DFHISAIP`.



```
INQ TCPIPS
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Tcpips( TCPSC08 ) Ope Por(<port>) Ipic Nos Tra(CISS)
    Con(00000) Bac( 00010 ) Maxd( 000000 ) Urm( DFHISAIP )
```

Off-host Configuration

The parameters for `APPLID` and `NETWORKID` should be left empty in your off-host configuration. For details see *CICS IPIC Configuration* using Command Central (GUI | Command Line), or using Local Scripts.

More Information

For more information on CICS `TCPIPSERVICE` and IPCONN resource definitions on your IBM mainframe, see the IBM documentation of CICS Transaction Server and CICS Transaction Gateway for Multiplatforms.

Error Handling

This table describes the handling of errors in the CICS IPIC connection or the RPC Server for CICS IPIC.

Problem	Handling
A CICS program sends abend code in response.	The CICS session is closed and the next call opens a different session.
The TCP/IP connection is lost with a SocketTimeoutException.	The CICS session is closed and the next call opens a different session.