System i

Networking
REXEC

Version 6 Release 1

System i

Networking
REXEC

Version 6 Release 1

Note

Before using this information and the product it supports, read the information in “Notices,” on page 21.

This edition applies to version 6, release 1, modification 0 of IBM i5/OS (product number 5761–SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

Contents

REXEC 1
PDF file for REXEC 1
Changing REXEC server attributes 1
REXEC command considerations 2
Selecting a command processor 2
REXEC connection usage 3
For i5/OS CL command processing 3
For Qshell and spawned path command processing 3
Spooled output considerations 3
REXEC client considerations 4
REXEC server jobs and job names 4
Creating REXEC server spooled job logs 4
Exit points for controlling REXEC server 4
Request validation exit point 6
VLRQ0100 exit point format 6
Server logon exit point 8
TCPL0100 exit point format 9
TCPL0300 exit point format 11
Server command processing selection exit point 14
RXCS0100 exit point format 14
Determining problems with REXEC 16
Materials required for reporting REXEC problems 18
Getting a copy of a REXEC server job log . . . 18
Tracing the REXEC server 18

Appendix. Notices 21
Programming interface information 22
Trademarks 23
Terms and conditions 23

REXEC

The Remote Execution (REXEC) server is a Transmission Control Protocol/Internet Protocol (TCP/IP) application that allows a client user to submit system commands to a remote system. The Remote Execution Protocol (REXEC) allows processing of these commands or programs on any host in the network. The local host then receives the results of the command processing.

The user's client program sends the user identifier, password, and command to run to the server. The server validates the user, runs the requested command, and returns the results of the command to the client.

Commands submitted to the System i™ host fall into the following categories:

i5/OS® command processor
You run i5/OS command processor commands by specifying QCAPCMD as the target of the client REXEC.

Qshell command interpreter (i5/OS option 30)
You can use the Qshell interpreter by specifying `qsh` as the target of client REXEC.

"Spawned paths"
You can run any i5/OS program in a "child" (spawned) job by specifying the complete path to the program or shell script as the target of the REXEC command.

Related information

Getting to know System i Navigator

PDF file for REXEC

You can view and print a PDF file of this information.

To view or download the PDF version of this document, select REXEC (about 287 KB).

Saving PDF files

To save a PDF on your workstation for viewing or printing:

1. Right-click the PDF link in your browser.
2. Click the option that saves the PDF locally.
3. Navigate to the directory in which you want to save the PDF.
4. Click **Save**.

Downloading Adobe Reader

You need Adobe® Reader installed on your system to view or print these PDFs. You can download a free copy from the Adobe Web site (www.adobe.com/products/acrobat/readstep.html)  .

Changing REXEC server attributes

The Change REXEC Attributes (CHGRXCA) command and Configure TCP/IP REXEC (CFGTCPRXC) command change the REXEC server attributes.

Note: You must have *IOSYSCFG special authority to make changes to the REXEC attributes with the CHGRXCA command.

Here are the ways to get to this command prompt:

- Specify the CHGRXCA command.
- Select Option 17 on the Configure TCP/IP Applications (CFGTCPAPP) display.

```
                      Change REXEC Attributes (CHGRXCA)

Type choices, press Enter.

Autostart server . . . . . .   *YES          *YES, *NO, *SAME
Number of initial servers  . . .  2           1-20, *SAME, *DFT
Inactivity timeout . . . . . .   300          1-2147483647, *SAME, *DFT
Coded character set identifier     00437        1-65533, *SAME, *DFT
```

Figure 1. Change REXEC Attributes (CHGRXCA)

The Configure TCP/IP REXEC (CFGTCPRXC) command provides a single interface for configuring attributes related to the Remote EXECution (REXEC) server and for running any other REXEC-related configuration commands.

REXEC command considerations

You should be aware of some restrictions when you use the REXEC command.

The REXEC server is restricted to running commands that are allowed in batch jobs. The command must have *BATCH as one of the *Where allowed to run* values.

The maximum length of a command that the REXEC server can process is 4000 bytes. Some REXEC clients limit the command to a smaller length.

For spawned paths, the program that runs in the child process must be either a program object in the QSYS.LIB file system (*PGM object) or a shell script. You must specify the path with the correct syntax for the file system in which the file exists.

For Qshell commands, you can put the same commands that you enter at an interactive command line into a noninteractive shell script.

Selecting a command processor

You can select different command processors for the REXEC server by using exit programs.

You can use the REXEC server command processing selection (QIBM_QTMX_SVR_SELECT) exit program to select which command processor the REXEC server uses to run the submitted command. (If you do not use an exit program, the REXEC server uses the control language (QCAPCMD) processor.) The following command processors are allowed.

- Control language (QCAPCMD)
- Qshell interpreter
- Spawned path (a shell script or program object)

Because data conversion is optional for the Qshell and spawn options, the exit program also selects whether the REXEC server performs ASCII-EBCDIC conversions on the stdin, stdout, and stderr streams.

REXEC connection usage

The REXEC protocol allows an REXEC client to specify whether to use one or two connections for returning data.

For i5/OS CL command processing

If you choose i5/OS CL command processing, you can choose to use one or two connections for returning data.

If you choose i5/OS CL command processing and two connections, normal output returns on the first connection and error output returns on the second connection. The REXEC server returns all spooled data that is written to the default printer file (*PRTF). This includes data that is written to the screen if the command is run in an interactive job. Any messages written to the job log return to the client on the second connection.

If the client specifies that all data returns on a single connection, the job log messages are returned first, followed by any spooled output.

For Qshell and spawned path command processing

For Qshell or spawned path command processing, the REXEC server, by default returns normal output on the first connection and error output on the second connection.

(The REXEC stdin, stdout, and stderr streams are mapped to file descriptors 0, 1, and 2. The QIBM_USE_DESCRIPTOR_STDIO environment variable is set to Y.) These options allow you to redirect input and output.

Choosing the Qshell command processor sets these environment variables:

- TERMINAL_TYPE=REMOTE
- PATH=/usr/bin
- LOGNAME= *user*, where *user* is the user profile
- HOME=*homedir*, where *homedir* is the user's home directory

The child job inherits any other environment variables that the exit program sets.

Spawned child processes are batch jobs or prestart jobs. They cannot do interactive I/O. See WebSphere® Development Studio: ILE C/C++ Programmer's Guide  for complete details on this support.

Spooled output considerations

The REXEC server overrides the default printer file (*PRTF) to capture spooled output. Any resulting spooled files are tagged with the user data field set to REXECSVR.

After the REXEC server runs the specified command, each spooled file with this user data tag is retrieved, returned to the client, and then deleted. If more than one spooled file is created, the files are processed in the order created, as determined by the spooled file number.

If the command or program run through REXEC performs its own print file override and changes the user data, the REXEC server is unable to capture and return the resulting spooled data.

Note: These considerations apply only to i5/OS CL commands.

REXEC client considerations

Different clients might use a single connection or two connections for returned data.

The REXEC client (RUNRMTCMD) uses a single connection for returned data, which is written to a spooled file on the client.

The UNIX®, OS/2®, and Windows® REXEC clients all use two connections, returning the normal output to the stdout stream and the error output to the stderr stream.

The VM REXEC client uses a single connection for the returned data, which is written to the console of the user.

REXEC server jobs and job names

To monitor and process requests from REXEC client users, start REXEC server jobs.

REXEC server jobs start when you run the STRTCP command and set the REXEC AUTOSTART parameter to *YES. You can also start REXEC server jobs by running the STRTCPSVR command with a SERVER parameter of *REXEC or *ALL. These jobs run in the QSYSWRK subsystem. Their purpose is monitoring and processing requests from REXEC client users. The format for the names of these jobs is QTRXC*nnnnn*, where *nnnnn* is a 5-digit decimal number.

To work with jobs in the QSYSWRK subsystem, including REXEC server jobs, specify the following command:

```
WRKSBSJOB SBS(QSYSWRK)
```

If you choose to have commands processed by the Qshell command interpreter, start Qshell by using the spawn() application programming interface (API) to create a child job.

If you choose to have commands interpreted as spawn path names, the REXEC server treats command strings as path names and passes them to the spawn() API. Spawned child processes are batch jobs or prestart jobs. Shell scripts are allowed for the child process. If you specify a shell script, the appropriate shell interpreter program is called. The shell script must be a text file and must contain this format on the first line of the file: `#!interpreter_path <options>`.

Creating REXEC server spooled job logs

To find the errors occurred in the REXEC session, you can create REXEC server spooled job logs.

The REXEC server automatically writes a server job log to a spooled file when it ends with an error.

To have a spooled job log produced at the end of each REXEC session and each time the REXEC server ends, use the CHGJOBD command as follows:

```
CHGJOBD JOBD(QTCP/QTMXRXCS) LOG(4 00 *SECLVL)
```

To obtain a spooled job log only when a server ends, use the CHGJOBD command as follows:

```
CHGJOBD JOBD(QTCP/QTMXRXCS) LOG(4 00 *NOLIST)
```

Exit points for controlling REXEC server

The experienced programmer can use exit programs to create customized processing while an application is running.

If the REXEC server finds a program registered to one of the exit points for the server, it calls that program using parameters that are defined by the exit point.

An exit point is a specific point in the REXEC program where control can pass to an exit program. An exit program is a program to which the exit point passes control.

For each exit point, there is an associated programming interface, called an exit point interface. The exit point uses this interface to pass information between the REXEC application and the exit program. Each exit point has a unique name. Each exit point interface has an exit point format name that defines how information is passed between the REXEC application and the customer-written exit program.

Different exit points can share the same exit point interface. When this is the case, multiple exit points can call a single exit program.

Exit point performance

The following table lists exit points that give you additional control over the REXEC server.

Table 1. TCP/IP exit points

TCP/IP exit points	Brief description	Exit points formats used
QIBM_QTMX_SERVER_REQ	The TCP/IP request validation exit point provides additional control for restricting an operation.	VLRQ0100
QIBM_QTMX_SVR_LOGON[1]	The TCP/IP server logon exit point provides additional control over authenticating a user and setting up the user's environment for the REXEC server.	TCPL0100 TCPL0300
QIBM_QTMX_SVR_SELECT	The REXEC server command processing selection exit point allows you to specify which command processor the REXEC server uses for interpreting and running your commands.	RXCS0100 format
[1]An exit point might have more than one format, but an exit program can only be registered for one of the exit point formats. Examine each of these formats, and then choose the one most appropriate for your system.		

Notes:

- The same interface format is used for request validation for the FTP client, FTP server, REXEC server, and TFTP server. This allows the use of one exit program for request validation of any combination of these applications.
- The same interface format is used for server log-on processing for the FTP server and REXEC server applications. This allows the use of one exit program to process log-on requests for both of these applications.

Related reference

"VLRQ0100 exit point format" on page 6
The interface that controls the parameter format for the exit point is VLRQ0100.

"TCPL0100 exit point format" on page 9
The exit point for REXEC server logon is QIBM_QTMX_SVR_LOGON. TCPL0100 is one of the interfaces that controls the parameter format for these exit points. This topic discusses the parameters of the TCPL0100 exit point format.

"TCPL0300 exit point format" on page 11
The exit point for REXEC server logon is QIBM_QTMX_SVR_LOGON. TCPL0300 is one of the interfaces that controls the parameter format for these exit points. This topic discusses the parameters of the TCPL0300 exit point format.

"RXCS0100 exit point format" on page 14
The exit point for REXEC server command processing selection is QIBM_QTMX_SVR_SELECT. The interface that controls the parameter format for the exit point is RXCS0100. This topic discusses the parameters of the RXCS0100 exit point format.

Related information

Use server exit programs

Request validation exit point

The request validation exit point can be used to restrict operations that can be performed by REXEC users.

The exit program should include the following elements:

- Exception handling
- Debugging
- Logging

Allowed and rejected commands:

The REXEC request validation exit program gives you control over whether to accept or reject an operation. Decisions made by exit programs are in addition to any validation that is performed by the REXEC server application.

Is there an exit program timeout feature?

There is no timeout for REXEC exit programs. If the exit program has an error or exception that it cannot handle, the REXEC server will stop the session.

VLRQ0100 exit point format

The interface that controls the parameter format for the exit point is VLRQ0100.

The following table shows the parameters and parameter format for the VLRQ0100 interface.

Table 2. Required parameter format for the VLRQ0100 exit point interface

Parameter	Description	Input or output	Type and length
1	Application identifier	Input	Binary (4)
2	Operation identifier	Input	Binary (4)
3	User profile	Input	Char (10)
4	Remote IP address	Input	Char (*)
5	Length of remote IP address	Input	Binary (4)
6	Operation-specific information	Input	Char (*)
7	Length of operation-specific information	Input	Binary (4)
8	Allow operation	Output	Binary (4)

Parameter descriptions

Application identifier

INPUT; BINARY(4) Identifies the TCP/IP application program that is making the request. Four different TCP/IP applications share the VLRQ0100 interface. These possible values are:

0 FTP client program

1 FTP server program

2 REXEC server program

3 TFTP server program

Operation identifier
Input; Binary(4) Indicates the operation (command) that the REXEC user wants (requests) to perform.

The following list shows the possible values when the application identifier indicates the REXEC server program.

0 Start session

1 Create directory/library

2 Delete directory/library

3 Set current directory/library

4 List files

5 Delete file

6 Send file

7 Receive file

8 Rename file

9 Execute CL command (The command must have *BATCH as one of the Where allowed to run values)

Notes:

- The 0 to 8 values represent control operations that are only for the FTP client and the FTP server.
- The hyphen symbol (-) represents control operations that the REXEC server exit does not recognize. To determine if a command has *BATCH as one of the *Where allowed to run* values you need to run the DSPCMD CL command, for example, DSPCMD CMD (DSPLIB).

User profile
INPUT; Char(10) The user profile for the REXEC session.

Remote IP address
INPUT; CHAR(*) The Internet Protocol (IP) address of the remote host system. For IPv4 connections, this string is in dotted-decimal format (123.45.67.89); for IPv6 connections, this string is in colon-delimited format (FE80::204:ACFF:FE7C:C84C). The remote host can be a client or a server that is based on the setting of the application identifier parameter.

Length of remote IP address
INPUT; BINARY(4) The length (in bytes) of the remote IP address.

Operation-specific information
INPUT; CHAR(*) Information that describes the requested operation. The contents of this field depend on the values of the operation identifier, and the application identifier.

For operation identifier 0 through 8 and application identifier 2
Not applicable for the REXEC server.

For operation identifier 9 and application identifier 2
The operation-specific information contains the i5/OS control language (CL) command that the user requests.

Length of operation-specific information
INPUT; BINARY(4) Indicates the length of the operation-specific information (parameter 6). The length is 0 when the exit point does not provide operation-specific information.

Allow operation
OUTPUT; BINARY(4) Indicates whether to allow or reject the requested operation. The following list shows the possible values.

-1 *Never* allow this operation identifier:

Reject this operation identifier unconditionally for the remainder of the current session.

The exit program will not receive this operation identifier again during the current session.

0 Reject the operation

1 Allow the operation

2 *Always* allow this operation identifier:

Allow this operation identifier unconditionally for the remainder of the current session.

The exit program will not receive this operation identifier again during the current session.

VLRQ0100 exit point format usage notes

VLRQ0100 is the exit point format that is used for REXEC request validation exit point.

Incorrect output parameters If the output returned for the Allow operation parameter is not valid, then the application rejects the requested operation and posts this message to the job log:

```
Data from exit program for exit point &1 is missing or not valid.
```

Exceptions

If the application encounters any exception when calling the exit program, it posts this message to the job log:

Exception encountered for exit program &1 in library &2 for exit &3

Related reference

"Exit points for controlling REXEC server" on page 4
The experienced programmer can use exit programs to create customized processing while an application is running.

Related information

FTP exit programs

Server logon exit point

You can control the authentication of users to a TCP/IP application server with the TCP/IP application server logon exit point.

This exit point allows server access based on the originating session's address. It also allows you to specify an initial working directory that is different from those that are in the user profile.

When you add an exit program to the exit point, the server calls the logon exit program each time a user attempts to log on. The exit program sets the return code output parameter to indicate whether the server continues the logon operation. Alternate return code settings are available for modifying the logon process and initializing directory information.

The i5/OS exit point for REXEC server logon is QIBM_QTMX_SVR_LOGON.

These are the two exit point formats available:

- The TCPL0100 exit point format controls the following logon operations:
  - Ability to accept or reject a logon
  - Control of the user profile, password, and current library
- The TCPL0300 exit point format extends the TCPL0200 format (the TCPL0200 format is not implemented by REXEC) so that you can use i5/OS enhanced password support and the additional parameters to enable coded character set identifier (CCSID) processing for password and directory name fields. In addition, when the user for the session has been authenticated with a client certificate, the exit program receives the client certificate.

Notes:

1. Only one exit program can be registered for the REXEC server logon exit point. You must decide which of the available exit point formats you want to use.
2. For all character parameters in exit point formats TCPL0100 and TCPL0300, and all character parameters without an associated CCSID in exit point format TCPL0300: Character data passed to the exit program is in the CCSID of the job. If the job CCSID is 65535, the character data is in the default CCSID of the job. Any character data that is returned by the exit program in these parameters is expected to be in this same CCSID.

TCPL0100 exit point format

The exit point for REXEC server logon is QIBM_QTMX_SVR_LOGON. TCPL0100 is one of the interfaces that controls the parameter format for these exit points. This topic discusses the parameters of the TCPL0100 exit point format.

Required parameter group

Parameter	Description	Input or output	Type and length
1	Application identifier	Input	Binary (4)
2	User identifier	Input	Char (*)
3	Length of user identifier	Input	Binary (4)
4	Authentication string	Input	Char (*)
5	Length of authentication string	Input	Binary (4)
6	Client IP address	Input	Char (*)
7	Length of client IP address	Input	Binary (4)
8	Return code	Output	Binary (4)
9	User profile	Output	Char (10)
10	Password	Output	Char (10)
11	Initial current library	Output	Char (10)

Parameter descriptions

Application identifier

INPUT; BINARY(4) Identifies the requested application server. The valid values are:

1 FTP server program

2 REXEC server program

User identifier
INPUT; CHAR(*) The user identification supplied by the client program. For the REXEC server, this parameter contains the user name of the user performing the REXEC request.

Length of user identifier
INPUT; BINARY(4) The length (in bytes) of the user identifier string.

Authentication string
INPUT; CHAR(*) The string (such as a password) supplied by the client program.

For the REXEC server, this parameter contains the password for the user that is specified in the User identifier parameter.

Length of authentication string
INPUT; BINARY(4) The length (in bytes) of the authentication string.

Client IP address
INPUT; CHAR(*) The Internet Protocol (IP) address from which the session originates. For IPv4 connections, this string is in dotted-decimal format (123.45.67.89); for IPv6 connections, this string is in colon-delimited format (FE80::204:ACFF:FE7C:C84C).

Length of client IP address
INPUT; BINARY(4) Indicates the length (in bytes) of the client IP address.

Return code
OUTPUT; BINARY(4) Indicates whether to accept or reject the logon operation, to perform password authentication, and whether to override the initial current library. The valid values are:

0 Reject the logon operation. Ignore the user profile, password, and initial current library output parameters.

1 Continue the logon operation with the specified user identifier and authentication string, and the user-specified initial current library. The user identifier becomes the user profile, and the authentication string becomes the password. The program ignores the user profile, password, and initial current library output parameters.

Note: For the logon to succeed, the password output parameter must match the password in the user profile.

2 Continue the logon operation with the specified user identifier and authentication string, and override the initial current library with the one specified by the initial current library parameter. The user identifier is the user profile. The authentication string is the password. Provide the initial current library output parameter. The program ignores the User profile and Password output parameters.

Note: For the logon to succeed, the password output parameter must match the password in the user profile.

3 Continue the logon operation. Override the user profile and password with those values you received from the output parameters of this exit program. Use the initial current library, which was specified by the user profile, that the exit program returns. The program ignores the initial current library output parameter.

Note: For the logon to succeed, the password output parameter must match the password in the user profile.

Attention: Never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

4 Continue the logon operation, which will override the user profile, password, and initial current library with output parameters of this exit program.

Note: For the logon to succeed, the Password output parameter must match the password in the user profile.

Attention: Never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

5 Accept the logon operation. Override the user profile with the profile returned in the User profile output parameter of this exit program. Use the initial current library, which was specified by the user profile, that the exit program returns. The program ignores the output parameters for the initial current library and password.

Note: Specifying this value overrides normal i5/OS password processing. The exit program must perform any required authentication.

6 Accept the logon operation. Override the user profile and initial current library with those that are returned in the output parameters of this exit program. Ignore the output parameter for password.

Note: Specifying this value overrides normal i5/OS password processing. The exit program must perform any required authentication.

User profile
OUTPUT; CHAR(10) The user profile to use for this session. This parameter must be left-aligned and padded with blanks.

Password
OUTPUT; CHAR(10) The password to use for this session. This parameter must be left-aligned and padded with blanks.

Initial current library
OUTPUT; CHAR(10) The initial current library to be established for this session. This parameter must be left-aligned and padded with blanks.

TCPL0100 format usage notes

REXEC server (application identifier 2):

1. 1. If the return code parameter is not valid, the REXEC server will not allow the operation. The REXEC server issues the message `Data from exit program for exit point &1 is missing or not valid` to the job log.
2. 2. If the REXEC server encounters any exception when calling the exit program, the REXEC server will not allow the operation. It issues the message `Exception encountered for REXEC exit program &1 in library &2 for exit point &3` to the job log.

Related reference

"Exit points for controlling REXEC server" on page 4
The experienced programmer can use exit programs to create customized processing while an application is running.

TCPL0300 exit point format

The exit point for REXEC server logon is QIBM_QTMX_SVR_LOGON. TCPL0300 is one of the interfaces that controls the parameter format for these exit points. This topic discusses the parameters of the TCPL0300 exit point format.

Required parameter group

Parameter	Description	Input or output	Type and length
1	Application identifier	Input	Binary (4)
2	User identifier	Input	Char (*)

Parameter	Description	Input or output	Type and length
3	Length of user identifier	Input	Binary (4)
4	Authentication string	Input	Char (*)
5	Length of authentication string	Input	Binary (4)
6	CCSID of authentication string	Input	Binary (4)
7	Client IP address	Input	Char (*)
8	Length of client IP address	Input	Binary (4)
9	Allow logon	Output	Binary (4)
10	User profile	Output	Char (10)
11	Password	Output	Char (*)
12	Length of password	Output	Binary (4)
13	CCSID of password	Output	Binary (4)
14	Initial current library	Input/Output	Char (10)
15	Initial home directory	Output	Char (*)
16	Length of initial home directory	Input/Output	Binary (4)
17	CCSID of initial home directory	Input/Output	Binary (4)
18	Application-specific information	Input/Output	Char (*)
19	Length of application-specific information	Input	Binary (4)

Parameter descriptions

Application identifier

INPUT; BINARY(4) Identifies the application server from which the request is being made. The valid values are:

1 FTP server program

2 REXEC server program

User identifier

INPUT; CHAR(*) The user identification supplied by the client program.

For the REXEC server, this parameter contains the user name of the user performing the REXEC request.

Length of user identifier

INPUT; BINARY(4) The length (in bytes) of the user identifier string.

Authentication string

INPUT; CHAR(*) The string (such as a password) supplied by the client program.

For the REXEC server, this parameter contains the password for the user that is specified in the User identifier parameter.

Length of authentication string

INPUT; BINARY(4) The length (in bytes) of the authentication string.

CCSID of authentication string
INPUT; BINARY(4) The CCSID of the authentication string parameter.

Client IP address
INPUT; CHAR(*) The Internet Protocol (IP) address from which the session originates. For IPv4 connections, this string is in dotted-decimal format (123.45.67.89); for IPv6 connections, this string is in colon-delimited format (FE80::204:ACFF:FE7C:C84C).

Length of client IP address
INPUT; BINARY(4) Indicates the length (in bytes) of the client IP address.

Allow logon
OUTPUT; BINARY(4) Indicates whether the logon operation should be accepted or rejected, and how password authentication is performed. The valid values are:

0 Reject the logon operation. Ignore all other output parameters.

1 Continue the logon operation with the specified user identifier and authentication string. The user identifier is the user profile, and the authentication string is the password. The current library and working directory is based on the settings of those output parameters. The application ignores the user profile and password output parameters.

Note: For the logon to succeed, the password output parameter must match the password in the user profile.

2 Continue the logon operation. Override the user profile and password with the returned values in the output parameters of this exit program. The application initializes the current library and working directory based on the settings of those output parameters.

Note: For the logon to succeed, the password output parameter must match the password in the user profile.

Attention: Never code passwords directly in an exit program. Encryption, for example, allows algorithmic password determination.

3 Accept the logon operation. Override the user profile with the profile returned in the User profile output parameter of this exit program. The program initializes the current library and working directory based on the settings of the output parameters. It ignores the Password output parameter.

Note: If the system is running at a security level of 20 or higher, specifying this value overrides normal i5/OS password processing. The exit program must perform any required authentication.

User profile
OUTPUT; CHAR(10) The user profile to use for this session. When required, this parameter must be left-aligned and padded with blanks.

Password
OUTPUT; CHAR(*) The password to use for this session. When required, the Length of password and CCSID of password parameters must also be specified, and this parameter must be left-aligned. When the QPWDLVL system value is set to 0 or 1, up to 10 characters can be specified. When the QPWDLVL system value is set to 2 or 3, up to 128 characters can be specified.

Length of password
OUTPUT; BINARY(4) The length (in bytes) of the password. When required, the valid range is 1 to 512 bytes.

CCSID of password
OUTPUT; BINARY(4) The CCSID of the password. This parameter must be set by the exit program when the password parameter is specified. The valid values are:

0 The CCSID of the job is used to determine the CCSID of the data to be converted. If the job CCSID is 65535, the CCSID from the default CCSID (DFTCCSID) job attribute is used.

1-65533
A valid CCSID in this range.

Initial current library
OUTPUT; CHAR(10) The initial current library to use for this session. When required, this parameter must be left-aligned and padded with blanks. This parameter is set to *CURLIB when the exit program is called. Use the current library that the user profile specifies.

Initial home directory
OUTPUT; CHAR(*) The initial setting of the home directory to use for this session. When specified, this parameter must be a valid absolute path name, and the Length of initial home directory and CCSID of initial home directory parameters must be set to the appropriate values.

Length of initial home directory
INPUT/OUTPUT; BINARY(4) The length of the initial home directory parameter returned by the exit program. This parameter initializes at zero when the application calls the exit program. If the exit program does not change the value of the parameter, the home directory is initialized to the home directory that the user profile specifies.

CCSID of initial home directory
OUTPUT; BINARY(4) The CCSID of the initial home directory. This parameter must be set by the exit program when the initial home directory is specified. The valid values are:

0 The CCSID of the job is used to determine the CCSID of the data to be converted. If the job CCSID is 65535, the CCSID from the default CCSID (DFTCCSID) job attribute is used.

1-65533
A valid CCSID in this range.

Application-specific information
INPUT/OUTPUT; CHAR(*) Information that is used to communicate application-specific logon settings.

Length of application-specific information
INPUT; BINARY(4) The length (in bytes) of the application-specific information.

Related reference

"Exit points for controlling REXEC server" on page 4
The experienced programmer can use exit programs to create customized processing while an application is running.

Server command processing selection exit point

The REXEC server command processing selection exit point allows you to specify which command processor the REXEC server uses for interpreting and running your commands.

RXCS0100 exit point format

The exit point for REXEC server command processing selection is QIBM_QTMX_SVR_SELECT. The interface that controls the parameter format for the exit point is RXCS0100. This topic discusses the parameters of the RXCS0100 exit point format.

The RXCS0100 exit program enables you to select:

- Which command processor runs the command that the REXEC client user provides.
- Whether the REXEC server converts data between ASCII and EBCDIC (for Qshell commands or spawn path names).

Note: Character data passes to the exit program in the coded character set identifier (CCSID) of the job. If the job CCSID is 65535, the server uses the default CCSID of the job.

Required parameter group

Parameter	Description	Input or output	Type and length
1	User profile	Input	Char (10)
2	Remote IP address	Input	Char (*)
3	Length of remote IP address	Input	Binary (4)
4	Command string	Input	Char (*)
5	Length of command string	Input	Binary (4)
6	Command processor identifier	Output	Binary (4)
7	Character conversion option	Output	Binary (4)

Parameter descriptions

User profile
INPUT; CHAR(10) The user profile under which the requested operations is run.

Remote IP address
INPUT; CHAR(*) The Internet protocol (IP) address of the REXEC client system. For IPv4 connections, this string is in dotted-decimal format (123.45.67.89); for IPv6 connections, this string is in colon-delimited format (FE80::204:ACFF:FE7C:C84C).

Length of remote IP address
INPUT; BINARY(4) Indicates the length (in bytes) of the remote IP address.

Command string
INPUT; CHAR(*) The command to be run as specified by the REXEC client.

Length of command string
INPUT; BINARY(4) Indicates the length (in bytes) of the command string.

Command processor identifier
OUTPUT; BINARY(4) Indicates the command processor that you want the server to use for interpreting and running the command. The following values are valid:

0 i5/OS control language. The server processes the command as an i5/OS control language (CL) command. This is the default value.

1 Qshell command. The Qshell command interpreter processes the command. The server uses the spawn() application programming interface (API) to call Qshell as a child job.

2 Spawn path name. The server treats the command name as a path name and passes it to the spawn() application programming interface (API), which runs as the child job.

Character conversion option
OUTPUT; BINARY(4) Indicates whether the REXEC server performs ASCII-EBCDIC character conversion for data that is passed on the stdin, stdout, and stderr streams. These values are valid:

0 Do not convert data. The server transfers all data on the stdin, stdout, and stderr streams without converting it.

1 Convert data. This is the default.

- The server converts data in the stdin stream from the ASCII CCSID that the CHGRXCA command specifies to the job CCSID. If the job CCSID is 65535, the server uses the default CCSID of the job.
- The server converts data in the stdout and stderr streams from the job CCSID to the ASCII CCSID that the CHGRXCA command specifies. If the job CCSID is 65535, the server uses the default CCSID of the job.

Usage notes

- If you add exit programs to both the QIBM_QTMX_SERVER_REQ and QIBM_QTMX_SVR_SELECT exit points, REXEC server first calls the exit program that you add to the QIBM_QTMX_SERVER_REQ exit point. If this program allows the operation, the server then calls the exit program that you add to the QIBM_QTMX_SVR_SELECT exit point.
- When you set the Command processor identifier parameter to 0 (i5/OS control language command), the conversion option is ignored. The server always performs character conversion for the CL commands.
- When you set the command processor identifier to 1 (Qshell Command), the server sets these environment variables:
  - TERMINAL_TYPE=REMOTE
  - PATH= /usr/bin:
  - LOGNAME= *user* (where *user* is the user profile)
  - HOME= *homedir* (where *homedir* is the user's home directory)

Related reference

"Exit points for controlling REXEC server" on page 4
The experienced programmer can use exit programs to create customized processing while an application is running.

Determining problems with REXEC

If you detect a problem when using the REXEC server, you can use the REXEC flow chart and cause lists to identify potential problems.

However, if you are having problems with general TCP/IP connectivity, it might be beneficial to first use the TCP/IP troubleshooting topic collection to identify basic TCP/IP problems. Use the REXEC flow chart for more localized problems.

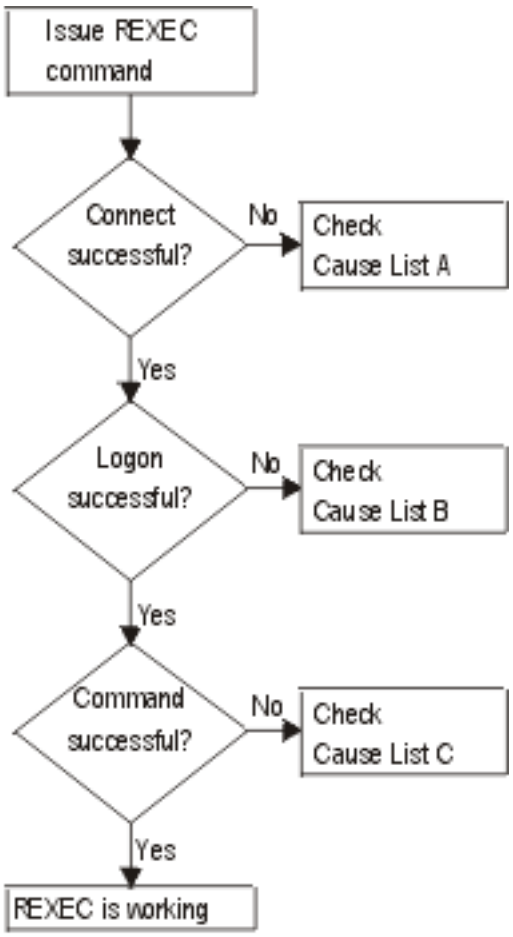

Figure 2. REXEC Server Problem Analysis

- Cause List A
  1. Check to see that the REXEC server is running. If not, start it with the STRTCPSVR SERVER(*REXEC) command.
  2. If the message "Connection refused" is returned to the REXEC client, check the exit program associated with the exit point QIBM_QTMX_SERVER_REQ. This exit program has either specified that the connection should be rejected, returned a value that is not correct for the Allow Operation parameter, or ended abnormally. Examine the REXEC server job log for messages. Resolve any problems with the exit program and install the corrected version.
- Cause List B
  1. Check your user ID and password by logging on to the system. If you are unable to do so, contact the system administrator to verify that your user ID and password are correct.
  2. Check the exit program associated with exit point QIBM_QTMX_SERVER_LOGON (if any). This exit program has either specified that the connection should be rejected, returned a value this is not correct for the Allow Operation parameter, or ended abnormally. Examine the REXEC server job log for messages.
- Cause List C
  1. Check for any job log messages returned to the REXEC client. Resolve any indicated problems and try the command again.
  2. If the message "Command Rejected" is returned to the REXEC client, check the exit program associated with the exit point QIBM_QTMX_SERVER_REQ. This exit program may be specifying that the command should be rejected, returning a value this is not correct for the Allow Operation

parameter, or ending abnormally. Examine the REXEC server job log for messages. Resolve any problems with the exit program and install the corrected version.

3. Verify that the correct ASCII CCSID is configured for the REXEC server. If not, set the correct CCSID with the CHGRXCA command.

Related information

TCP/IP troubleshooting

Materials required for reporting REXEC problems

When you report REXEC problems, you need to provide some specific information, such as the error data or server job logs.

Any REXEC problem reported to IBM® should include the following information:

- A communications trace from the time of the failure (Request TCP/IP data only) formatted for ASCII.
- If the REXEC server has logged software error data, submit this information.

  Note: The system value QSFWERRLOG must be set to *LOG for software error logging to take place. If an error occurs while QSFWERRLOG is set to *NOLOG, change the value to *LOG, try to re-create the error, and submit the logged software error data. If logged software error data is submitted, there is no need to perform a trace of the REXEC server.
- The QTCPIP and any REXEC server job logs.

Related information

Communications trace

Getting a copy of a REXEC server job log

To get a copy of the REXEC server job log, you need to create it.

To have the REXEC server save job logs, see "Creating REXEC server spooled job logs" on page 4.

Tracing the REXEC server

The REXEC server can be traced by creating a data area. Note that running the REXEC server with trace running may cause a significant performance impact.

To trace the REXEC server, follow these steps:

1. Create the data area using the following command:
   ```
   CRTDTAARA DTAARA(QUSRSYS/QTMXRXCDBG) TYPE(*LGL) LEN(1)
   ```
2. Perform the REXEC operation that you want to trace.
3. Delete the data area using the following command:
   ```
   DLTDTAARA DTAARA(QUSRSYS/QTMXRXCDBG)
   ```
4. Enter the following command to find the output queue:
   ```
   DSPSYSVAL QPRTDEV
   ```
   For example, the following display appears:

```
                          Display System Value
System value . . . . . :   QPRTDEV
Description  . . . . . :   Printer device description
Printer device . . . . :   PRT01          Name
```

Figure 3. Display System Value Display

   The printer device is also the name of the default system output queue.
5. Record the name of the printer device. In this example, PRT01 is the printer device.

6. Press F12 (Cancel) to return to the display where you entered the DSPSYSVAL command.
7. Type the following command:

   ```
   WRKOUTQ OUTQ(printer-device)
   ```

   Replace `printer-device` with the printer device recorded in the previous display. PRT01 is the output queue in this example. For example, the following display appears:

```
                           Work with Output Queue
Queue:   PRT01          Library:   QGPL           Status: RLS
Type options, press Enter.
  1=Send   2=Change   3=Hold   4=Delete   5=Display   6=Release   7=Messages
  8=Attributes        9=Work with printing status
Opt  File        User        User Data   Sts   Pages  Copies  Form Type   Pty
 _   QTCPPRT     QTCP        QTMSMTP     HLD     46   1       *STD         5
 _   QPSRVTRC    QSECOFR                 HLD     44   1       *STD         5
```

Figure 4. Work with Output Queue Display

8. Press F18 (Bottom) to get to the bottom of the spooled file list if More... appears on the display.
9. Find the last file named QPSRVTRC with the same user as the user who was logged on the REXEC server when the trace was created.
10. Press F11 (View 2) to view the date and time of the file you want to work with.
11. Verify that you are working with the most recent spooled file, QPSRVTRC.
12. Indicate in the problem report that the trace was tried and it failed. Send whatever trace information there is with the problem report.

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, IBM License Agreement for Machine Code, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

This REXEC publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of IBM i5/OS.

Trademarks

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

i5/OS
IBM
IBM (logo)
OS/2
System i
WebSphere

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

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.

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Printed in USA