

[print](#) | [close](#)

## APIs by Example: Making the Connection with Server Authentication Entries

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 05/10/2007 (All day)

Exchanging and accessing data between systems is often a requirement in today's application programming assignments. Different options are readily available on the System i to accomplish this task, provided that a TCP/IP connection is configured between the systems in question. DDM (Distributed Data Management) files, DDM data queues and DRDA (Distributed Relational Database Architecture) using SQL are some of the built-in facilities that come to mind.

Apart from setting up and configuring the connection between the systems, there's yet another challenge included when going down that road: How do you authenticate the user profile under which the connection will be established? Again, different options exist, but one of the most secure and flexible is offered by server authentication entries.

A server authentication entry is added for a local user profile. It specifies a remote server as the main attribute and contains the authentication information (user ID and optionally, a password) that is sent to the remote server when the local user profile is used to establish a connection to the remote server. One user profile can have any number of server authentication entries, but only one per server.

Also, please note that to allow passwords to be stored encrypted with server authentication entries, the system value Retain Server Security Data (QRETSVRSEC) must be set to '1'. Before performing any adjustments to this system value, please be sure to read the documentation carefully. A link is provided below.

For example, if user profile CHUCK on SYSTEMA wants to use user ID CHUCK2 and password 'After\_8' every time he establishes a connection to SYSTEMB, the Add Server Authentication Entry (ADDSVRAUTE) command on SYSTEMA would look like this:

```
ADDSVRAUTE  USRPRF(CHUCK)
              SERVER(SYSTEMB)
              USRID(CHUCK2)
              PASSWORD(After_8)
```

The above setup applies to all connects of type DDM, where the relational database server name was used to identify the remote system. This means the Relational Database (RDB) parameter was specified for the remote location on the Create DDM File (CRTDDMF) or Create Data Queue (CRTDTAQ) TYPE(\*DDM) CL command.

Specifying the RDB parameter causes the system to use the relational database directory to identify the remote system when establishing the connection. You can verify and configure the relational database directory using the Work with RDB Directory Entry (WRKRDBDIRE) command. For this setup to work, you must ensure that the same server name is specified for both the server authentication entry and the RDB directory entry.

For non-RDB DDM files and data queues, IBM has provided a special workaround that allows you to use the server authentication entry method: Specify the special server name QDDMSERVER (all uppercase) for the server name on the ADDSVRAUTE command.

With this method, each user profile on a system can have only one non-RDB DDM authentication entry, but you can overcome this limitation by using a special application or communication user profile for each remote system. By special application or communication user profile, I refer to a user profile created specifically for

running either an application service or a communication process, and nothing else. The service or process is run in an individual job using the dedicated user profile and is accessed through a data queue as demonstrated in "APIs by Example: Data Queue APIs and CL Commands, Part 5" ([article ID 54098](#)).

Using the SQL CONNECT statement either interactively from the Start SQL Interactive Session (STRSQL) command's statement entry panel, or in an embedded SQL statement in a program, works the same way as the DDM RDB-method. If you leave out the USER (user profile) USING (password) clause from the SQL CONNECT statement, the system will pick up this information from the server authentication entry directory, if available.

You don't have to provide this information explicitly or by means of host variables for interactive SQL and embedded SQL. The system takes care of this requirement once the correct information has been added to server authentication entry directory, and the password is encrypted and stored safely in an internal table.

You can use the Display Server Authentication Entries (DSPSVRAUTE) command to see a given user profile's server authentication entries. The Change Server Authentication Entry (CHGSVRAUTE) command is available to change a specific authentication entry's user ID or password, taking effect immediately. There's also a Remove Server Authentication Entry (RMVSVRAUTE) command to remove one or all of a user's server authentication entries.

APIs similar in function to the CL commands are also available. For more details, and for detailed IBM documentation describing different scenarios where server authentication entries can be useful, please follow the links at the end of this article.

Because i5/OS does not currently provide a command to list all or a subset of a system's server authentication entries, I provide one here. Based on the Open List of Authorized Users (QGYOLAUS) and Retrieve Server Authentication Entries (QsyRetrieveServerEntries) APIs, as well as some UIM programming, the task is quite simple to overcome. Here's the Work with Server Authentication Entries (WRKSVRAUTE) command prompt:

```

                                Work with Server Auth Entries (WRKSVRAUTE)
Type choices, press Enter.
User profile . . . . . *ALL          Name, generic*, *ALL

Output . . . . . *          *, *PRINT

```

You can specify a user profile name, a generic user profile name, or the special value \*ALL, and whether the output should be displayed in a work-with panel or printed with the current job's spooled output. The work-with panel looks like this:

```

                                Work with Server Authentication Entries
WYNDHAMW
                                04-05-07
16:20:04
User profile . . .   *ALL          Position to . . .

Type options, press Enter.

    2=Change    4=Remove    5=Display user profile auth entries

    User

Opt  profile      Server      CCSID  User ID      CCSID
Password
    CHUCK         WEBSERVER    037   CHUCK2      037    *YES
    DAN           APPDEV001    037   APPUSR      037    *YES

```

WEBAPPUSR	QDDMSERVER	037	WEBUSR	037	*YES
					Bottom
Parameters or command					
===>					
F3=Exit	F4=Prompt	F5=Refresh	F6=Add server auth entry	F9=Retrieve	
F11=Display text	F12=Cancel	F22=Display entire name	F24=More keys		

The above panel lets you change the user profile selection criteria and specify a 'Position to' user profile value. The available function keys include F6 (to add a server authentication entry using the ADDSVRAUTE command), F20 (to jump to and work with the relational database directory), and F22 (to display the entire name of either a server name or a user ID exceeding the space available in the list panel).

As always, detailed online help is provided for both the command and the work-with panel. Please be aware that, depending on the number of user profiles on your system and system resources available, the initial run of the WRKSVRAUTE command might take some time to complete. Subsequent updates using the refresh function key or user profile selection criteria normally complete much faster, once the user profile list has been primed.

#### **This APIs by Example includes the following sources:**

```
CBX173  -- RPGLE  -- Work with Server Authentication Entries - CPP

CBX173E -- RPGLE  -- Work with Server Authentication Entries - UIM Exit Program
CBX173H -- PNLGRP -- Work with Server Authentication Entries - Help
CBX173P -- PNLGRP -- Work with Server Authentication Entries - Panel Group

CBX173V -- RPGLE  -- Work with Server Authentication Entries - VCP

CBX173X -- CMD    -- Work with Server Authentication Entries

CBX173M -- CLP    -- Work with Server Authentication Entries - Build command
```

To create all above objects, compile and run CBX173M. Compilation instructions are found in the source headers, as usual.

#### **IBM Server Authentication Entry related documentation:**

Add Server Auth Entry (ADDSVRAUTE) CL command:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/cl/addsvraute.htm>

Retain Server Security Data (QRETSVRSEC) system value:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/rzakz/rzakzqretsvrsec.htm>

Clear Server Security Data (CLRSVRSEC) CL command:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/cl/clrsvrsec.htm>

Application requester security in a TCP/IP network:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/ddm/rbae5sourcesecurity.htm>

Application server security in a TCP/IP network:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/ddm/rbae5targetsecurity.htm>

Explicit connection management:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/ddp/rba1excon.htm>

DDM connection authorization failure:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/ddm/rbae5connauth.htm>

DRDA connect authorization failure:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/ddp/rbal1drdaconauth.htm>

Protection strategies in a distributed relational database:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/ddp/rbal1rdbpro.htm>

Configuring DDM Data Queue Support over TCP/IP:

[https://www-](https://www-912.ibm.com/s_dir/slkbase.NSF/7250f367f6396d2f86256a4f007973d5/e95c5072635554dd86256e980068aada?OpenDocument)

[912.ibm.com/s\\_dir/slkbase.NSF/7250f367f6396d2f86256a4f007973d5/e95c5072635554dd86256e980068aada?OpenDocument](https://www-912.ibm.com/s_dir/slkbase.NSF/7250f367f6396d2f86256a4f007973d5/e95c5072635554dd86256e980068aada?OpenDocument)

Assigning a Default User Profile for IBM® DRDA® over TCP/IP:

[https://www-](https://www-912.ibm.com/s_dir/slkbase.NSF/643d2723f2907f0b8625661300765a2a/9e61c67ade70359986256afd007cc9f1?OpenDocument)

[912.ibm.com/s\\_dir/slkbase.NSF/643d2723f2907f0b8625661300765a2a/9e61c67ade70359986256afd007cc9f1?OpenDocument](https://www-912.ibm.com/s_dir/slkbase.NSF/643d2723f2907f0b8625661300765a2a/9e61c67ade70359986256afd007cc9f1?OpenDocument)

Distributed Data Management V5R4:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/////ddm/rbae5.pdf>

Distributed Database Programming V5R4:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic//ddp/rbal1.pdf>

**This article demonstrates the following Server Authentication Entry APIs APIs:**

Retrieve Server Authentication Entries (QSYRTVSE/QsyRetrieveServerEntries) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/qsyrvtse.htm>

Server Authentication Entry APIs documentation:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/netsec1a.htm>

**You can retrieve the source code for this API example from:**

[http://www.pentontech.com/IBMContent/Documents/article/54655\\_202\\_WrkSvrAutE.zip](http://www.pentontech.com/IBMContent/Documents/article/54655_202_WrkSvrAutE.zip).

**Source URL:** <http://iprodeveloper.com/rpg-programming/apis-example-making-connection-server-authentication-entries>