Exercise 4. Connecting queue managers

Estimated time

01:00

Overview

In this exercise, you create channels between two queue managers. You use the IBM MQ sample programs to test the connection between the queue managers.

Objectives

After completing this exercise, you should be able to:

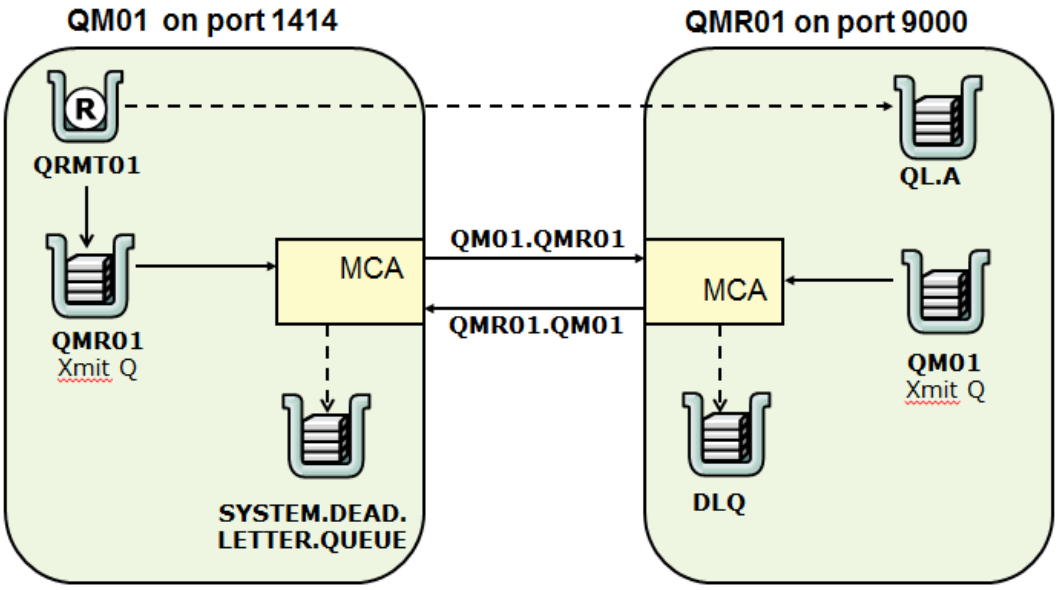
- Configure a distributed network of two or more interconnected queue managers
- Use IBM MQ commands to create the channels and supporting objects to implement distributed queuing
- Use the IBM MQ sample programs to test the connection between the queue managers

Introduction

In this exercise, you must configure two queue managers. One of queue managers (QM01) is a local queue manager, and the other queue manager (QMR01) **simulates** a queue manager on a remote server.

In Part 1 of this exercise, you create a server-connection channel on QMR01 so that you can manage the queue manager as if it is a queue manager on another server. You learn more about server-connection channels later in this course.

In this exercise, you configure two message channels between the queue managers so that messages can flow in each direction. The local queue manager is QM01 that you created in Exercise 1. You create queue manager QMR01 on the simulated remote server in this exercise. The following figure summarizes the configuration requirements.



In IBM MQ, channel authentication and connection authentication are enabled by default for all new queue managers. By default, channel authentication and connection authentication prevent remote connections to the queue manager from MQ Explorer. In one step of this exercise, you disable channel authentication and connection authentication on the queue manager to allow connections to remote queue managers from MQ Explorer.

The exercise uses TCP/IP as the communications protocol.

Requirements

- IBM MQ and IBM MQ Explorer
- The queue manager QM01 and queues that were created in Exercise 1
- Familiarity with the sample programs that were described in Exercise 3
- The IBM MQ sample programs **amqsput**, **amqsget**, **amqsreq**, and **amqsech**
- A text editor

Exercise instructions

Part 1: Create a remote queue manager

In this part of the exercise, you create queue manager QMR01.

You gain experience with the steps that are required to connect to a queue manager on a remote server from MQ Explorer. You connect to this queue manager from MQ Explorer as if it is on a remote server.

__ 1. Start the queue manager QM01 that you created in Exercise 1 unless it is already running.

__ 2. Using MQ Explorer, create a queue manager that is named **QMR01** that uses a queue that is named **DLQ** for a dead-letter queue and a listener that uses port number 9000.

__ 3. Using MQ Explorer, create the local queue DLQ for the queue manager dead-letter queue (defined in the previous step).

__ 4. In IBM MQ, channel authentication and connection authentication are enabled by default for all new queue managers. By default, channel authentication and connection authentication prevent remote connections to the queue manager from MQ Explorer.

Use MQSC to disable channel authentication and connection authentication on QMR01 to allow a remote connection from MQ Explorer.

__ a. In a command window, type: **runmqsc QMR01**

__ b. Display the queue manager properties. Type: **DIS QMGR**

__ c. Verify that the CHLAUTH attribute is set to **ENABLED** and that the CONNAUTH attribute is set to **SYSTEM.DEFAULT.AUTHINFO.IDPWOS**. These settings are the default values.

__ d. Disable channel authentication. Type:

```
ALTER QMGR CHLAUTH(DISABLED)
```

__ e. Verify that the CHLAUTH attribute is now set to **DISABLED**.

__ f. For this exercise, disable client connection authentication. Type:

```
ALTER QMGR CONNAUTH(' ')
```

Note

A space character is between the single quotation mark characters in the **CONNAUTH** attribute of the **ALTER QMGR** command.

__ g. Refresh the security cache for the queue manager. Type:

```
REFRESH SECURITY
```

__ h. End the MQSC session. Type: **END**

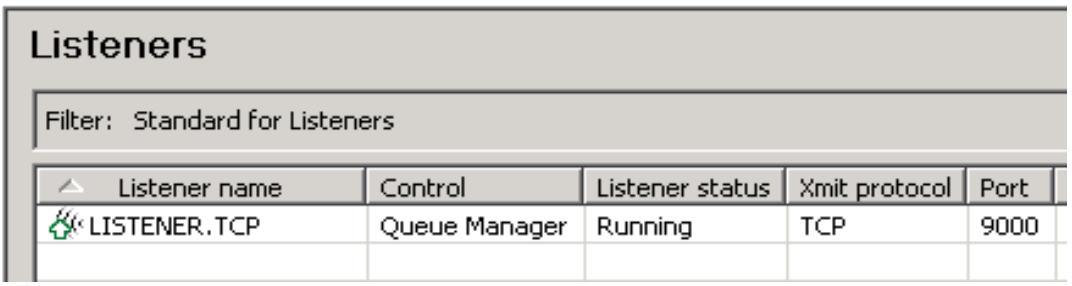
Important

The steps to disable channel and connection authentication are not standard practice. Authentication is disabled in this exercise because security configuration is covered in a later unit.

__ 5. Verify that the listener is running on QMR01.

__ a. In the **MQ Explorer - Navigator** view, click the **Listeners** folder under QMR01.

__ b. Verify that the listener that is named **LISTENER.TCP** is in the list, that the status is **Running**, and that the listener is running on port 9000.



__ 6. In this step, you configure the queue manager QMR01 so that you can connect to it from MQ Explorer as if the queue manager is on a remote server.

In MQ Explorer, create a server connection channel (SVRCONN) on the queue manager QMR01.

__ a. In the **MQ Explorer - Navigator** view, right-click the queue manager QMR01 and then click **Remote Administration**.

__ b. In the **Remote Administration** window, click **Create** to create the SYSTEM.ADMIN.SVRCONN channel.

Click **OK** on the confirmation window and then click **Close**.

Information

You can display the SYSTEM channels in MQ Explorer by clicking **Channels** in the **MQ Explorer - Navigator** view and then clicking the **Show System Objects** icon in the **Channels** view.

__ 7. Connect to the queue manager QMR01 from MQ Explorer as if it is running on a remote server.

__ a. In the **MQ Explorer - Navigator** view, right-click **Queue Managers** and then click **Add Remote Queue Manager**.

__ b. In the **Add Queue Manager** window, type: **`QMR01`**

__ c. Select **Connect directly** and then click **Next**.

__ d. For the **Host name or IP address**, type: **`localhost`**

__ e. For the **Port number**, type: **`9000`**

__ f. Verify that the server-connection channel is SYSTEM.ADMIN.SVRCONN.

__ g. Click **Finish.**

The queue manager should now be visible in the **MQ Explorer - Navigator** view as **QMR01 on 'localhost(9000)'**.

Part 2: Define the channels and listener on the local queue manager

In this part of the exercise, you define a sender channel and a receiver channel on queue manager, QM01.

__ 1. Using a text editor, create an MQSC command file that is named **ConnectLocal.mqsc** to define the MQ objects that are required for a connection between QM01 and QMR01.

__ a. Define a message sender channel.

- Channel name = **QM01.QMR01** (where `QM01` is the local queue manager name and `QMR01` is the remote queue manager)
- Protocol = **TCP/IP**
- Network address of QMR01 = **localhost(9000)** (where `localhost` is the host name and 9000 is the listener port)
- Transmission queue name = **QMR01** (the same as the name of the remote queue manager)

Type:

```
DEF CHL(QM01.QMR01) CHLTYPE(SDR) REPLACE +
TRPTYPE(TCP) CONNAME('localhost(9000)') +
XMITQ(QMR01)
```

__ b. Define a receiver channel. The attributes must match the sender channel of the remote queue manager QMR01. Type:

```
DEF CHL(QMR01.QM01) CHLTYPE(RCVR) REPLACE +
TRPTYPE(TCP)
```

__ c. Create a transmission queue with the same name as the remote queue manager QMR01. Type:

```
DEF QL(QMR01) REPLACE USAGE(XMITQ)
```

__ 2. Use **runmqsc** to run the script file **ConnectLocal.mqsc** against the local queue manager QM01. Redirect the output to a text file that is named **LocalReport.txt**.

__ a. Type: **runmqsc QM01 < ConnectLocal.mqsc > LocalReport.txt**

Note

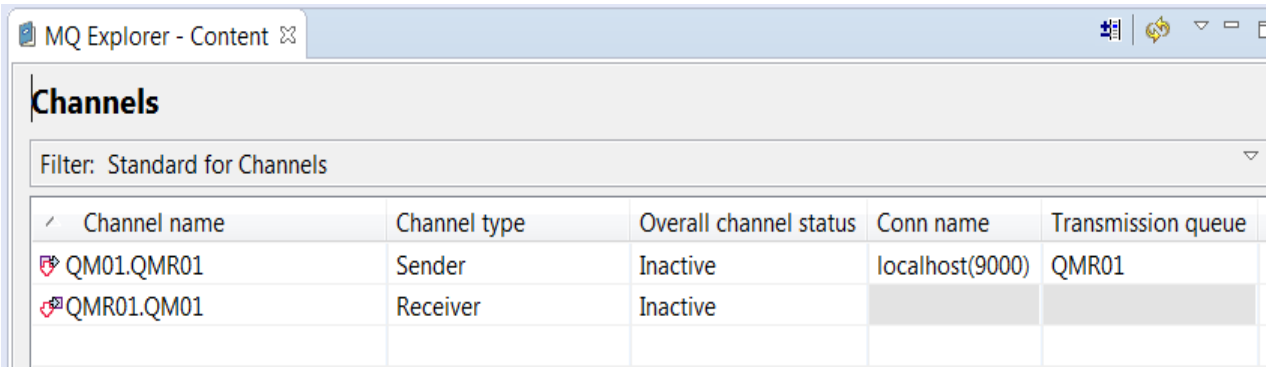
If you did not save the **ConnectLocal.mqsc** file in the current directory, you must either change directories or specify the path name. For example:

```
runmqsc QM01 < /labfiles/Lab04/ConnectLocal.mqsc > /labfiles/Lab04/LocalReport.txt
```

__ 3. Examine the **LocalReport.txt** file and verify that all commands ran successfully.

__ 4. Use MQ Explorer to verify that QM01 now has two channels:

- A sender channel that is named **QM01.QMR01**
- A receiver channel that is named **QMR01.QM01**



__ 5. Verify that QM01 now has a transmission queue that is named `QMR01`.

__ 6. When you created QM01 in Exercise 1, you did not create or start a listener. In this step, use the `DEFINE LISTENER` command to create a listener on port 1414 for the queue manager QM01.

Use the `CONTROL(QMGR)` option in the `DEFINE LISTENER` command to associate the listener service with the queue manager. The listener service automatically starts when the queue manager starts and stops when the queue manager stops.

In MQSC for QM01, type:

```
DEF LISTENER(LISTENER.TCP) TRPTYPE(TCP) PORT(1414) CONTROL(QMGR)
```

__ 7. Use the `START LISTENER` command to start the listener on QM01.

In MQSC for QM01, type:

```
START LISTENER(LISTENER.TCP)
```

Alternatively, you can start the MQ listener by using the **runmqlsr** command. For example:

```
runmqlsr -t TCP -p 1414 -m QM01
```

If you use the **runmqlsr** command, you must use the **endmqlsr** command to manually stop the service.

Linux

If you want to start the MQ listener in the background on Linux, type:

```
nohup runmqlsr -m QM01 -t tcp -p 1414 2>&1 &
```

You can verify that the listener is running by typing the following command in a terminal window:

```
ps -ef | grep runmqlsr
```

__ 8. To eliminate any security errors and security configuration in this exercise, disable the use of channel authentication and connection authentication on QM01.

In MQSC for QM01, type:

```
ALTER QMGR CHLAUTH(DISABLED)
ALTER AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPWOS) +
AUTHTYPE(IDPWOS) CHCKCLNT(NONE)
REFRESH SECURITY
```

Important

The steps to disable channel and connection authentication are not standard practice. Authentication is disabled in this exercise because security configuration is covered in a later unit.

Part 3: Define the channels on the remote queue manager

In this part of the exercise, you define the sender channel, receiver channel, and transmission queue for QMR01.

__ 1. Using a text editor, create an MQSC command file that is named **ConnectRemote.mqsc** to define the MQ objects that are required for a connection between your local queue manager and the remote queue manager.

__ a. Define a sender channel that matches the receiver channel on QM01.

- Channel name = **QMR01.QM01**
- Protocol = **TCP/IP**
- Network address of QM01 = **localhost(1414)**
- Transmission queue name = **QM01**

Type:

```
DEF CHL(QMR01.QM01) CHLTYPE(SDR) REPLACE +
TRPTYPE(TCP) CONNAME('localhost(1414)') +
XMITQ(QM01)
```

__ b. Define a receiver channel. The attributes must match the sender channel QM01. Type:

```
DEF CHL(QM01.QMR01) CHLTYPE(RCVR) REPLACE +
TRPTYPE(TCP)
```

__ c. Create a transmission queue for QM01. Type:

```
DEF QL(QM01) REPLACE USAGE(XMITQ)
```

__ 2. Use **runmqsc** to run the script file **ConnectRemote.mqsc** against the remote queue manager QMR01. Redirect the output to a text file that is named **RemoteReport.txt**.

Type: **runmqsc QMR01 < ConnectRemote.mqsc > RemoteReport.txt**

__ 3. Use the report and MQ Explorer to verify that QMR01 now has two channels:

- A receiver channel that is named **QM01.QMR01**
- A sender channel that is named **QMR01.QM01**

Also, verify that QMR01 now has a transmission queue that is named **QM01**.

Part 4: Test and start the connection

__ 1. To test the channel definitions, open another command window, and use the **runmqsc** command to ping the message channel from the QM01 (the sender). Type:

```
runmqsc QM01
PING CHL(QM01.QMR01)
```

Check for successful completion.

__ 2. In MQSC for QM01, use the **START CHANNEL** command to start the sender channel and verify that it is working.

Type: **START CHL(QM01.QMR01)**

__ 3. Using MQSC for QM01, verify that the channel is running.

Type: **DIS CHSTATUS(*)**

You should see that the channel QM01.QMR01 is running, the remote queue manager name is QMR01, and the transmission queue is QMR01.

Part 5: Test queues in a distributed environment

In this part of the exercise, you create the queues for testing queue managers in a distributed environment and then use the MQ sample programs to put and get messages.

__ 1. On QMR01, define a local queue that is named **QL.A**.

In MQSC for QMR01, type: **DEF QL(QL.A) REPLACE**

__ 2. In MQSC, define the application queues on QM01.

__ a. Delete and redefine the local queue **QL.A**. Type:

```
DEF QL(QL.A) REPLACE
```

__ b. Verify that you defined a transmission queue that is named QMR01. Type:

```
DIS QL(QMR01)
```

Verify that the USAGE attribute is set to **XMITQ**.

__ c. Create a remote queue definition that is named **QRMT01** that points to the local queue QL.A on QMR01 and uses QMR01 for the transmission queue.

Type the following command:

```
DEF QR(QRMT01) REPLACE +
RNAME(QL.A) RQMNAME(QMR01) +
XMITQ(QMR01)
```

__ d. Verify the remote queue definition. Type:

```
DIS QR(QRMT01)
```

__ 3. Use the sample program **amqsput** to send messages from queue manager QM01 to the queue QL.A on queue manager QMR01.

In a command window, type:

```
amqsput QRMT01 QM01
```

__ 4. Use MQ Explorer to verify that the messages you sent in Step 3 are on queue QL.A on QMR01.

You can also use MQSC to check the CURDEPTH attribute of the target queue QL.A on QMR01.

__ 5. If the messages do not arrive on the target, investigate the possible causes. Check the following items:

__ a. Is the transmission queue on QM01 empty?

__ b. Is the sender channel on QM01 running?

__ c. Is the dead-letter queue of the remote queue manager QMR01 empty?

__ d. Inspect the error logs in both queue managers.

Exercise cleanup

__ 1. Clear the messages on QL.A on the QMR01 queue manager.

__ 2. Leave the queue managers running. You use these queue managers in the next exercise.

End of exercise

Exercise review and wrap-up

Having completed this exercise, you should be able to:

- Create the objects that are required for distributed queuing
- Configure and refresh the started MQ listener
- Start message channels manually
- Use sample applications to test the message flow