Using ssh to test connectivity between an IBM MQ Client in one host and a Queue Manager in another host (netstat, ncat and telnet)

https://www.ibm.com/support/pages/node/482611

Date last updated: 10-May-2024

Angel Rivera
IBM MQ Support
https://www.ibm.com/products/mq/support
Find all the support you need for IBM MQ

++ Objective ++

You have an IBM MQ Client Application that is trying to connect to a remote queue manager, but the application is receiving errors indicating that a queue manager is not found or that is not available.

For example,
- Errors like: AMQ9213E, AMQ9524E, AMQ9202E or AMQ9508E
- Reason codes like 2059 (MQRC_Q_MGR_NOT_AVAILABLE)

Please note that this list is not a comprehensive list of errors, but just a sample of some of the more common ones.

++ Causes ++

a) One likely cause is that the two machines cannot communicate at the basic level (not MQ involved). Perhaps the remote host is not running, or that there is a DNS problem with the host name or IP address.

b) The MQ Client needs 3 pieces of information to attempt to connect to the MQ queue manager:
- The hostname or IP address
- The port number for the MQ Listener
- The Server-Connection channel

May be the IP address or hostname was not properly specified by the MQ client.
Or the port number might be incorrect.

c) The queue manager might not be running at the desired host.

d) The queue manager could be running, but the corresponding listener is not running at the specified port.

e) There may be some third party application (like a firewall or network threat protection service) not allowing the communication to flow in or out of the two machines.

++ Diagnosing The Problem ++

For this tutorial let's assume that we have these 2 hosts:

- Host-1 for the queue manager: riggioni1
  Queue manager name: QM93LNX
  Port: 1415

- Host-2 for the client application: suvereto1
  For illustration purposes, the connection information is specified by means of MQSERVER:
  export MQSERVER='SYSTEM.DEF.SVRCONN/TCP/riggioni1(1415)'

You can use the TCP/IP connectivity tools:
- "ping" to determine whether there is basic TCP/IP connectivity.
- "ssh" to see if a possible MQ Listener is running in the remote port.

The steps are shown below (with an illustration command):

- Step 1) Ensure that the Listener is running in the queue manager

- Step 2) Use "ping" to test for basic connectivity
  ping riggioni1

- Step 3) Use the "ssh" command to test the remote port
  ssh riggioni1 -p 1414

Interpreting the results:

3.a) ssh: connect to host riggioni1 port 1414: Connection refused
Conclusion:
The host could be reached, but nobody is listening in the specified port!
Definitely, the MQ Listener is NOT running on that port.

3.b) kex_exchange_identification: Connection closed by remote host
Conclusion:
Somebody is listening! But it is NOT a guarantee that the MQ Listener is actually using it. It could be another application!

3.c) ssh: connect to host port 1421: No route to host
Conclusion:
The host CANNOT be reached or IS NOT responding.
Thus, the status of the port cannot be asserted, and thus, we cannot reach yet a conclusion about the status of the MQ Listener.

- Step 4) Additional/Optional MQ connection test: sample amqscnxc

```
amqscnxc -x 'riggioni1.fyre.ibm.com(1415)' -c SYSTEM.DEF.SVRCONN QM93LNX
```

Note:
The original technote used the deprecated tool "telnet", but this tool is no longer shipped with recent versions of Windows or Linux.

++ Step 1) Ensure that the Listener is running in the queue manager

In Host-1, you need to make sure that you have a Queue manager running and a Listener running on the MQ server.
Take notice regarding the port that the listener is listening on (the default is port 1414).

The following article could be helpful:
https://www.ibm.com/support/pages/node/7028389
Verifying that an IBM MQ Listener runmqlsr is running in Windows and Linux

++ Step 2) Use "ping" to test for basic connectivity

In Host-2, where you have your MQ Client application, you need to use the TCP/IP tool "ping" to see if there is basic connectivity from Host-2 to Host-1:

You need to specify exactly the short host name or the fully qualified domain name of the host, as you have it specified in the MQSERVER.

The following shows a successful command:

+ Linux:

In Linux, you must use Ctrl-C to end the ping command.

```
mqm@suvereto1.fyre.ibm.com: /home/mqm
$ ping riggioni1
PING riggioni1.fyre.ibm.com (9.46.66.142) 56(84) bytes of data.
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=1 ttl=63 time=0.265 ms
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=2 ttl=63 time=0.248 ms
^C
```

Or you can specify to only 4 attempts (like in Windows), via the -c option:

```
$ ping -c 4 riggioni1
PING chamonix1 (9.46.110.220) 56(84) bytes of data.
PING riggioni1.fyre.ibm.com (9.46.66.142) 56(84) bytes of data.
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=1 ttl=63 time=0.265 ms
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=2 ttl=63 time=0.248 ms
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=1 ttl=63 time=0.265 ms
64 bytes from riggioni1.fyre.ibm.com (9.46.66.142): icmp_seq=2 ttl=63 time=0.248 ms
--- riggioni1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3073ms
rtt min/avg/max/mdev = 0.424/0.446/0.483/0.026 ms
```

+ Windows:

The ping command is more user-friendly: it only does 4 attempts and then terminates. That is, there is no need to do Control-C.
For example:

```
C:\> ping suvereto1
Pinging suvereto1.fyre.ibm.com [9.46.77.193] with 32 bytes of data:
Reply from 9.46.77.193: bytes=32 time=65ms TTL=54
Reply from 9.46.77.193: bytes=32 time=65ms TTL=54
Reply from 9.46.77.193: bytes=32 time=70ms TTL=54
Reply from 9.46.77.193: bytes=32 time=69ms TTL=54
Ping statistics for 9.46.77.193:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 65ms, Maximum = 70ms, Average = 67ms
```

++ ATTENTION:

If the "ping" command fails, such as due to a typo or due to a DNS TCP configuration problem, then you need to address this issue and ensure that the ping command is successful.
ONLY AFTER the issue has been addressed, then you can proceed with the rest of this article.

++ INCORRECT EXAMPLES

Some incorrect examples are:
** A typographical error:

+ Linux:
$ ping rigioni1
ping: rigioni1: Name or service not known

+ Windows:
C:\> ping rigioni1
Ping request could not find host riggioni1. Please check the name and try again.

** The host is not running. You start the ping, but after a minute nothing happens and you terminate with Control-C. Then you notice the statement of 100% packet loss

+ Linux:
$ **ping volterra1**
PING volterra1.fyre.ibm.com (9.46.80.212) 56(84) bytes of data.
^C
--- volterra1.fyre.ibm.com ping statistics ---
19 packets transmitted, 0 received, 100% packet loss, time 18438ms

+ Windows:
C:\> **ping volterra1**
Pinging volterra1.fyre.ibm.com [9.46.80.212] with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 9.46.80.212:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

++ Step 3) Use the "ssh" command to test the remote port

The "ssh" command provides an additional test that "ping" cannot provide: ssh tries to connect to a port that you can specify. In addition, ssh is more thorough in its method of connectivity.

Ping is like calling a phone number and checks only if there is a ringtone at the other end, while ssh is like calling a phone number and checks is someone picking up and answering.

NOTE: There are many other utilities for testing port availability, but because most users have already "ssh" installed on a client machine, "ssh" will be explored as a tool for testing connectivity in this Technote.

+ ATTENTION: Entries will be added to the error log and informational FDC files can be generated! (But they will not hurt the queue manager!)

When you test this method to a MQ server, the queue manager will dump errors in its error log and if configured to do so, then the queue manager can generated an informational FDC, since the data coming in over the communication is not MQ formatted data. See the following technote for details:

https://www.ibm.com/support/pages/node/133357
IBM MQ FDC with Probe IDs CO618001 CO052000, Major Errorcode rrcE_BAD_DATA_RECEIVED

A sample of the errors in the error log of the queue manager is:

AMQ9207E: The data received from host '9.46.74.79' on channel '????' is not valid.
EXPLANATION:
Incorrect data format received from host '9.46.74.79' over TCP/IP. It may be that an unknown host is attempting to send data.
An FFST file might be generated containing the invalid data received. It will not be generated if this is the beginning of a conversation with the remote side, and the format is a simple known format (example: a GET request from an HTTP web browser). If you want to override this, to cause FFST files to be written for any bad data, including simple known formats, then set the environment variable AMQ_BAD_COMMS_DATA_FDCS=TRUE and restart the queue manager.
The channel name is '????'; in some cases it cannot be determined and so is shown as '????'.

AMQ9492E: The TCP/IP responder program encountered an error.
EXPLANATION:
The responder program was started but detected an error.
The host name was '9.46.74.79'; in some cases the host name cannot be determined and so is shown as '????'.

+ In Host-1, the runmqlsr listener is running in port 1415, and there are no applications listening in port 1414

```
mqm@riggioni1.fyre.ibm.com: /var/mqm/qmgrs/QM93LNX/errors
$ netstat -apn | grep 1415
tcp6       0      0 :::1415                 :::*                    LISTEN      15335/runmqlsr

$ netstat -apn | grep 1414
(notice that the output is null - empty)
```

+ Using ssh in Host-2 for port 1414 in Host-1 (where there is no application that is listening)

Note: the reply should be very fast!

```
mqm@suvereto1.fyre.ibm.com: /home/mqm
$ ssh riggioni1 -p 1414
ssh: connect to host riggioni1 port 1414: Connection refused
```

Conclusion:
The host could be reached, but nobody is listening in the specified port!
Definitely, the MQ Listener is NOT running on that port.

Note about telnet:

This is the equivalent error from telnet:

```
$ telnet riggioni1 1414
   telnet: connect: A remote host refused an attempted connect operation.
```

Note from ncat:

This is the equivalent error from ncat:

```
$ ncat -v riggioni1 1414
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Connection refused.
```

+ Using ssh in Host-2 for port 1415 in Host-1 (where runmqlsr is listening)

Note: it took 2-3 seconds to get a reply, which provided an indirect hint that there was some processing going on.

```
mqm@suvereto1.fyre.ibm.com: /home/mqm
$ ssh riggioni1 -p 1415
kex_exchange_identification: Connection closed by remote host
```

Conclusion:
Somebody is listening!
But it is NOT a guarantee that the MQ Listener is actually using it. It could be another application that is listening in the port!

Note about telnet:

This is the equivalent error from telnet:

```
$ telnet riggioni1 1415
Trying...
Connected to paqlnd1.nfcu.net.
Escape character is '^]'.
```

Note from ncat:

This is the equivalent error from ncat.
You will need to do Ctrl-C to escape.

```
$ ncat -v riggioni1 1415
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Connected to 9.46.85.142:1415.
^C
```

+ Using ssh in Host-2 for port 1414 in Host-3

For completeness, this is what happens when the host is down or there is a firewall issue.

$ ssh unreachablehost -p 1414
ssh: connect to host port 1421: No route to host

Conclusion:
The host CANNOT be reached or IS NOT responding.
Thus, the status of the port cannot be asserted, and thus, we cannot reach yet a conclusion about the status of the MQ Listener.

The most likely problems are:
- The host you are trying to connect to is not turned on or is not connected to the network.
- The host is behind a firewall and the firewall is blocking SSH connections.

+ Note: If you are in Host-2 (MQ Client application) and if you know the password for the MQ Administrator in Host-1, then you could issue the following command ssh. You will be prompted to enter the password.

```
mqm@suvereto1.fyre.ibm.com: /home/mqm
$ ssh mqm@riggioni1 "ps -ef" | grep runmqlsr
The authenticity of host 'riggioni1 (9.46.66.142)' can't be established.
ECDSA key fingerprint is SHA256:sb4sb82MgZMvH9C46pthq6XizW0WnRg710IgT8QxdjQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'riggioni1,9.46.66.142' (ECDSA) to the list of known hosts.
mqm@riggioni1's password:
mqm       15335   15198  0 09:57 ?        00:00:00 /opt/mqm/bin/runmqlsr -r -m QM93LNX -t
TCP -p 1415
```

++ Step 4) Additional/Optional MQ connection test: sample amqscnxc

There is an additional MQ test you can run: the amqscnxc sample connect program included with the MQ samples. You need to have the MQ samples module installed to get this. Source code for the utility is also provided with the samples.

The following test will connect to the queue manager and respond that you are connected/or fail:

Generic format:
  amqscnxc -x ConnName -c SvrconnChannelName QMgrName

Example:
mqm@suvereto1.fyre.ibm.com: /home/mqm
$ amqscnxc -x 'riggioni1.fyre.ibm.com(1415)' -c SYSTEM.DEF.SVRCONN QM93LNX
Sample AMQSCNXC start
Connecting to queue manager QM93LNX
using the server connection channel SYSTEM.DEF.SVRCONN
on connection name riggioni1.fyre.ibm.com(1415).
Connection established to queue manager QM93LNX
Sample AMQSCNXC end

++ Notes about ncat

Some customers like to use the Linux tool "ncat".
For more information see:
https://nmap.org/ncat/guide/ncat-usage.html

In a test Linux machine, the ncat command was installed by the following rpm fileset:

mqm@rochefort1.fyre.ibm.com: /home/mqm
$ **which ncat**
/usr/bin/ncat

$ **rpm -qf /usr/bin/ncat**
nmap-ncat-7.91-12.el9.x86_64

+++ end +++