# Introducing IBM MQ Replicated Data Queue Manager

## Introduction

This lab provides a demonstration of a new approach to Disaster Recovery in MQ on Linux, with the follow-ing key features:

* Use of Distributed Replicated Block Device (DRBD) storage rather than network shared storage
* This is still using a Replicated Data Queue Manager (RDQM):
  + Takeover will be manual not automatic
  + Both asynchronous and synchronous replication is supported
  + An RDQM is active on one node only at any one time
  + Each node can run different active RDQMs
  + An individual DR RDQM is created to use one style of replication and it cannot be changed without recreating the RDQM
  + In 9.0.5 an RDQM can be either HA or DR but not both \* As only two nodes are involved it will be possible to get into a split-brain situation, but on-ly if a user has chosen to promote a DR Secondary and start a DR RDQM when the DR network is disconnected and the DR RDQM is still running, or is also started, where it was Primary.

The goals for RDQM-DR are:

* Allow an RDQM to be created which is configured to replicate its data to a single Secondary in-stance at a given IP address \*
  + Asynchronous replication is supported provided the latency is no more than 50ms for a round trip time
  + Synchronous replication is subject to the same 5ms limits on latency as it is for HA
* Allow manual control of when a DR Secondary becomes a DR Primary and can then run the RDQM In this lab, instructions are provided to show the setup for both.For this lab exercise node 02 has had some of the ‘pre-configuration’ already performed.

### Summary of lab environment

* 3 RHEL 7.4 x86\_64 systems running in Skytap:
  + miqmp This will be our primary node
  + miqms This will be a secondary node
* VMWare Workstation virtual networks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Subnet** | **DHCP** |
| VMnet3 | Host-only | 10.0.3.0 | No |
| VMnet4 | Host-only | 10.0.4.0 | No |
| VMnet8 | Host-only | 10.0.0.0 | No |

* Network interfaces:

|  |  |  |  |
| --- | --- | --- | --- |
| **Interface Purpose** | **Interface Name** | **miqmp (Primary node)** | **miqms (Secondary node)** |
| Administration | ens33 | 10.0.0.1 | 10.0.0.2 |
| DR Replication | Ens37 | 10.0.4.1 | 10.0.4.2 |
| MQ Fixed IP | ens36 | 10.0.3.1 | 10.0.3.2 |

DR interfaces are used as follows:\*

1. DR Replication - for synchronous / asynchronous data replication (the higher the bandwidth the better and the lower the latency the better)

|  |  |
| --- | --- |
|  | hosts **miqmp**, **miqms** are tied to the Administration IP addresses above |

### Pre-configuration steps

The following steps are necessary for configuring RDQM. They have already been completed on the VMs. See the Appendix for the commands to execute the installation and prerequisites prior to configuring RDQM.

* Although not required for this Lab, the following Pacemaker dependencies required for RDQM HA have already been installed. This list should be sufficient for a standard installation of RHEL 7.4 Server or Workstation. For your own environment setup, if you are using some other installation then additional packages may be needed:
  + OpenIPMI-modalias.x86\_64
  + OpenIPMI-libs.x86\_64
  + libyaml.x86\_64
  + PyYAML.x86\_64
  + libesmtp.x86\_64
  + net-snmp-libs.x86\_64
  + net-snmp-agent-libs.x86\_64
  + openhpi-libs.x86\_64
  + libtool-ltdl.x86\_64
  + perl-TimeDate
* Extract and Install MQ 9.0.5
* The code is provided as a compressed tar file in the directory /home/student/Downloads.
* Install the MQ and RDQM code
* RDQM is a single feature which now supports HA and/or DR (but not at the same time for a single queue manager). The RDQM support requires the Server and Runtime packages. Run the installation script.
* Configure the RedHat firewall
* If there is a firewall between the nodes in the HA group, then the firewall must allow traffic between the nodes on a range of ports.
* Firewall (firewalld) enabled, and ports 1500 & 1501 will be defined during the lab.
* Configure the OS storage settings
* If the system uses SELinux in a mode other than permissive, you must run the following command:

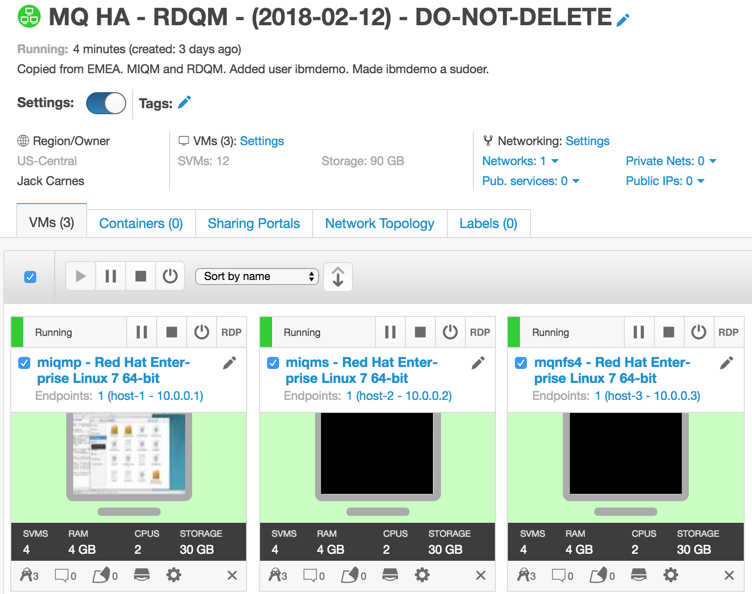
*semanage permissive -a drbd\_t*

* Configure groups
  1. To create, delete, or configure replicated data queue managers (RDQMs) you must use a user ID that belongs to both the mqm and haclient groups.
  2. If want to allow a normal user in the mqm group to create RDQM instances etc., you need to grant the user access to the certain commands via sudo. The user will also need to be part of the mqm group.
  3. You will add the mqm user to the root and haclient group. Then add root, student, and ibmdemo to the mqm and haclient groups.
* The following groups set up:
  + - **mqm** to allow user to run specific MQ commands,
    - A normal user "ibmdemo" has been defined.
* Create the Logical Group for the QM data
  1. Each node requires a volume group named drbdpool. The storage for each replicated data queue manager is allocated as a separate logical volume per queue manager from this volume group. For the best performance, this volume group should be made up of one or more physical volumes that correspond to internal disk drives (preferably SSDs).

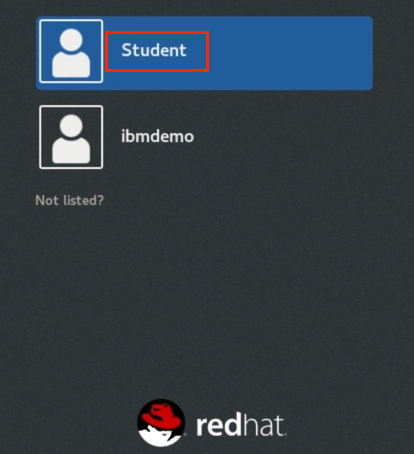
The above steps must be completed on each node before RDQM can be configured. At this point you are ready to begin RDQM configuration.

### Setup the RHEL image (pre-configured on SkyTap):

In the Skytap environment, there are 3 virtual machines miqmp, miqms, mqnfs4 and should be in a paused state.



1. Click the run button to resume the VMs.
2. Log on to VM miqmp as user Student, using password Passw0rd!.



## Configure RDQM-DR

A primary instance of a disaster recovery queue manager is created on one server. A secondary instance of the same queue manager must be created on another server, which acts as the recovery node. Data is replicated between the queue manager instances. The replication of the data between the two nodes is handled by DRBD.

Unlike the High Availability solution, there is no heartbeat detection between the two nodes. If the primary queue manager node is lost, the secondary instance can be manually made into the primary instance, the queue manager started, and work resumed.

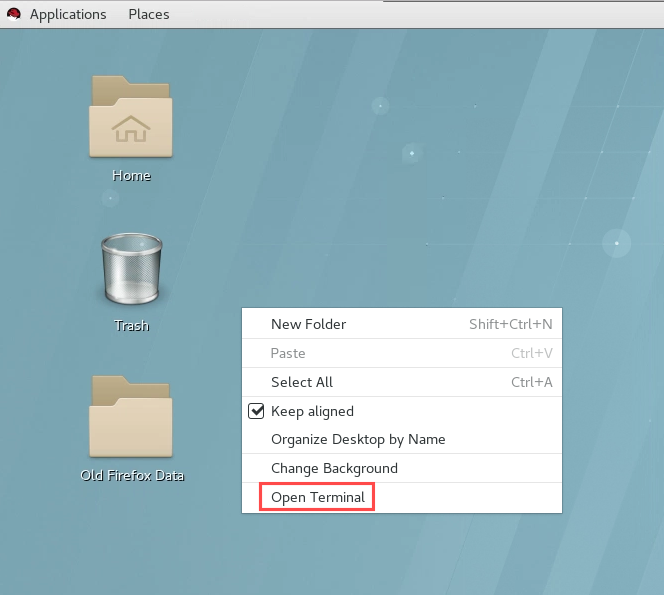
Data replication between primary and secondary queue managers can be done synchronously or asynchronously. If the asynchronous option is selected, operations such as PUT or GET complete and return to the application before the data is replicated to the secondary queue manager. Asynchronous replication means that, following a recovery situation, some messaging data might be lost. But the secondary queue manager will be in a consistent state, and able to start running immediately, even if it is started at a slightly earlier part of the message stream.

You will configure a DR RDQM that uses asynchronous replication.

### Create the DR RDQM

You will create a DR with asynchronous replication. You must first create a primary RDQM DR queue manager. Then you will create a secondary instance of the same queue manager on another node. The primary and secondary instances must have the same name and be allocated the same amount of storage.

1. You should be logged on as *student* on miqmp.
2. Right-click on the desktop and select *Open Terminal.*



1. Switch to root user with the command:

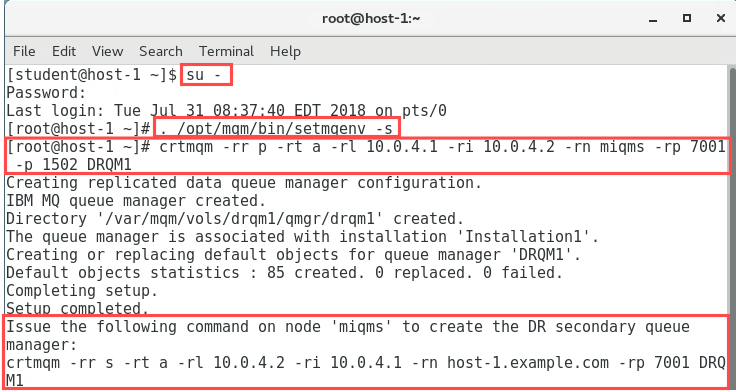
* *su -*

1. Enter the *Passw0rd!* for student's password.
2. Enter the command to set the MQ environment.

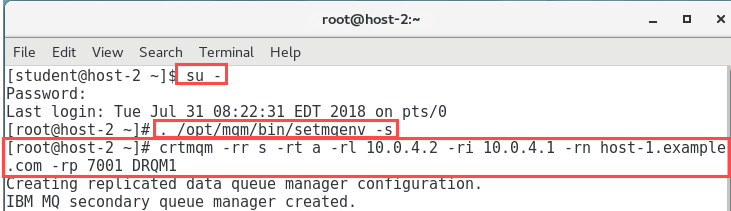
*. /opt/mqm/bin/setmqenv -s*

1. Create a primary queue manager on node miqmp. It will use asynchronous replication. The local IP for DR replication is 10.0.0.1. The recovery IP used for replication on the secondary instance is 10.0.4.0. Replication will take place using port 7001. The queue manager will be DRQM1.
2. In the terminal window, create the primary node:

*crtmqm -rr p -rt a -rl 10.0.4.1 -ri 10.0.4.2 -rn miqmp -rp 7001 -p 1502 DRQM1*

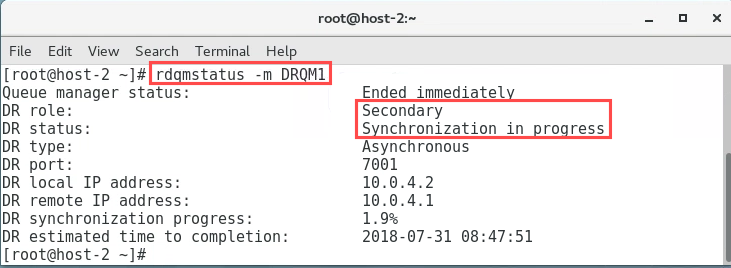


1. Notice at the end the command needed to create the secondary instance is provided for you.
2. Switch to the miqms VM. Login as *student / Passw0rd!*
3. As you did on miqmp, switch user to root.
4. Create a secondary instance of the queue manager on node miqms. In the termninal window, enter the command which was provided for you when you ran the crtmqm command on miqmp.

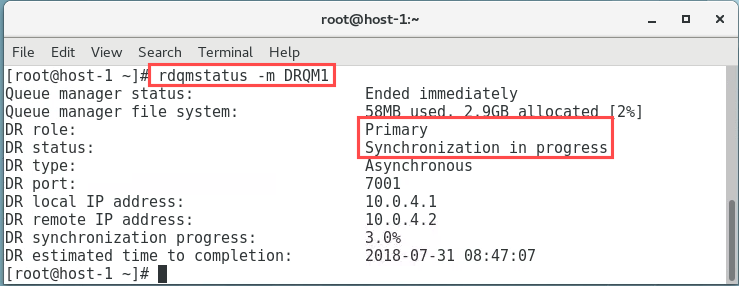
* *crtmqm -rr s -rt a -rl 10.0.4.2 -ri 10.0.4.1 -rn host-1.example.com -rp 7001 DRQM1*
* 

1. Check the status on both nodes with the command to ensure they are correct. On node miqmp use the command:

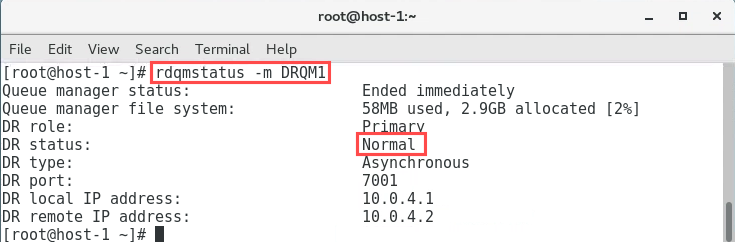
* *rdqmstatus -m DRQM1*

On miqms:  
  


On miqmp:



1. Issue the command again until it shows synchronisation is complete. When initial synchronisation has completed, it should look similar to the following:

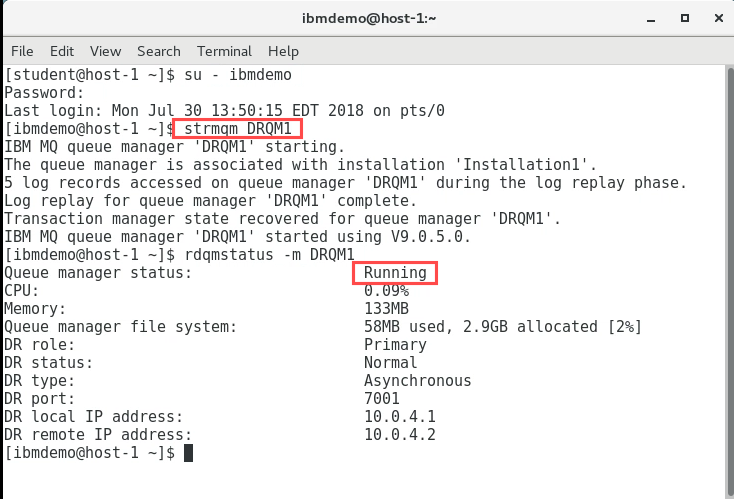
* 

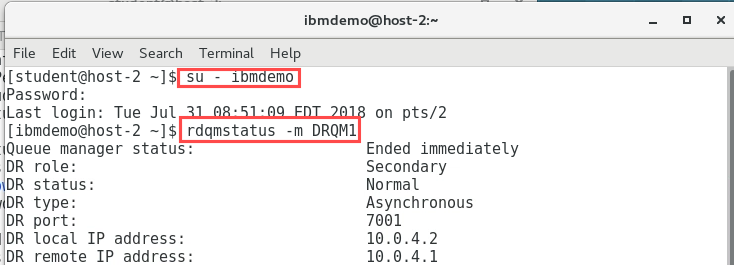
1. On the node with the secondary instance, the output should initially look similar to the following:
2. On node miqmp, start the queue manager with the following command:

* *strmqm DRQM1*

1. Now check the status on both nodes to ensure they are correct, using the command:

* *rdqmstatus -m DRQM1*

On miqmp, the output will initially look similar to the following:  
  


As the node with the secondary instance, only runs the queue manager when DR is needed, the output will be unchanged and look as it did previously.  
  


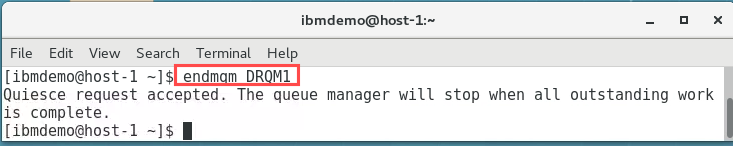
## Test the DR Secondary

Now that the DR nodes have been set up, you will test the secondary DR queue manager.

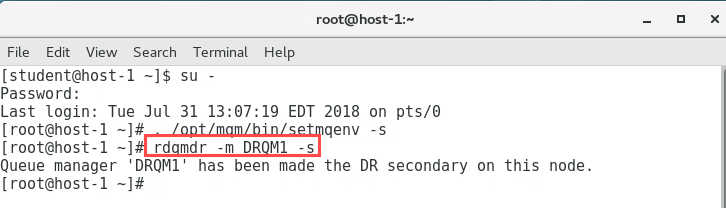
### Make the Primary instance the Secondary node

Only one node can be the Primary. Therefore, before another node can be designated the Primary, the original Primary needs to be designated the Secondary.

1. On node miqmp, stop the queue manager:

* *endmqm DRQM1*
* 

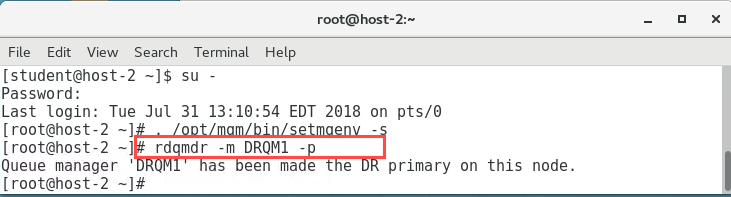
1. Designate node miqmp, as the secondary using the rdqmdr command:

* *rdqmdr -m RDQM1 -s*
* 

### Make the Secondary node the Primary instance

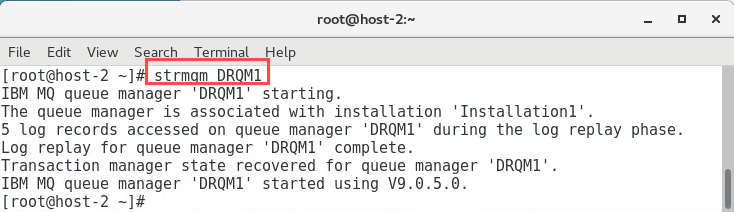
Designate the Secondary node as the Primary instance.

1. On the recovery node miqms, designate it as the primary instance using the rdqmdr command:

* *rdqmdr -m DRQM1 -p*
* 

1. Still on miqms, start the queue manager:

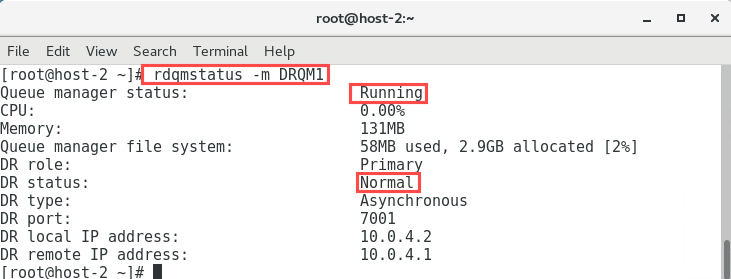
*strmqm DRQM1*

* 

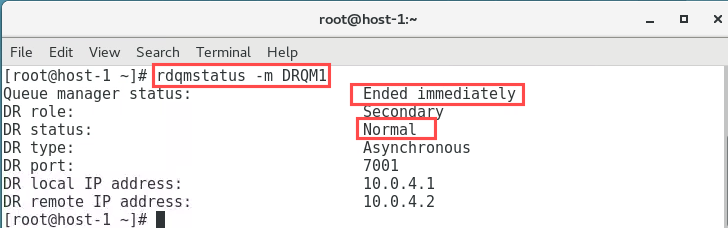
1. Confirm the status of both nodes:

* *rdqmstatus -m DRQM1*

On node miqms:



On node miqmp:



1. Provided that channels were defined with a list of alternative connection names specifying the primary and secondary queue managers, then applications will automatically connect to the new primary queue manager.

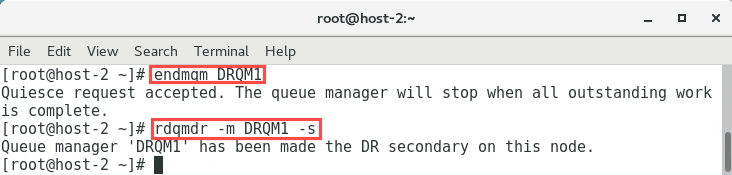
### Make the Primary instance the Primary again

If the loss of the Primary was only temporary, you would want to designate it as the Primary again. This would be achieved as described below.

1. On node miqms, stop the queue manager:

* *endmqm DRQM1*

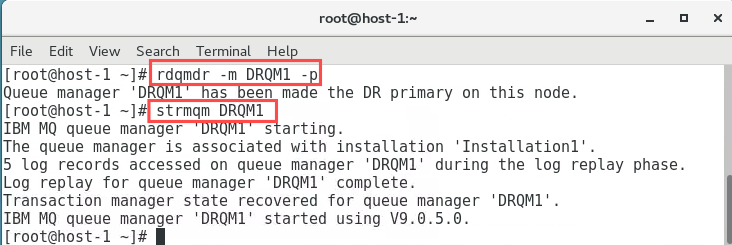
1. Designate node miqms, as the secondary:

* *rdqmdr -m DRQM1 -s*
* 

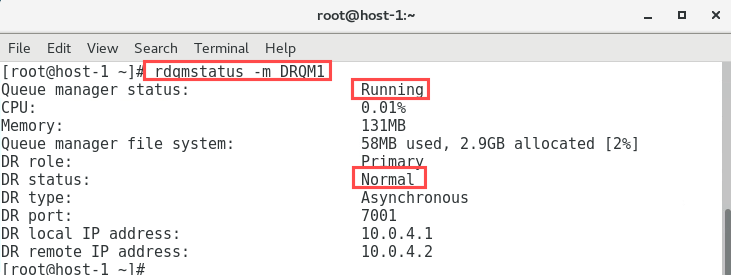
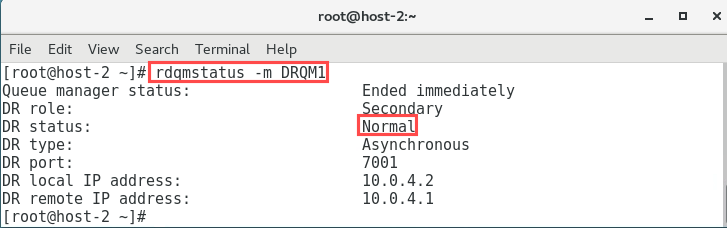
1. On the primary node miqmp, designate it as the primary instance again:

* *rdqmdr -m DRQM1 -p*

1. On the primary node miqmp, restart the queue manager:

* *strmqm DRQM1*
* 

1. Confirm the status of both nodes:

* *rdqmstatus -m DRQM1*
* On miqmp:
* 
* On miqms:
* 

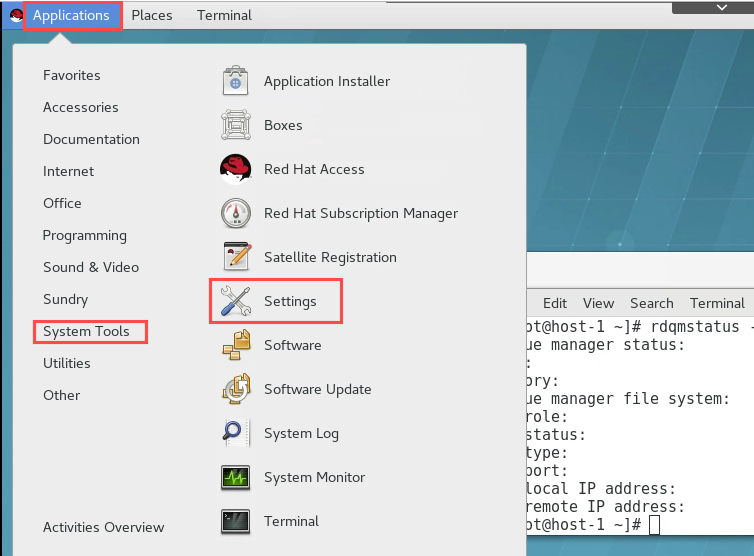
## Replace node that was running a DR Primary

Suppose the loss of the primary node was due to a failure, which resulted in the node having to be replaced. You would want to replace the primary node while the queue manager runs on the secondary node. Then restore the original disaster recovery configuration.

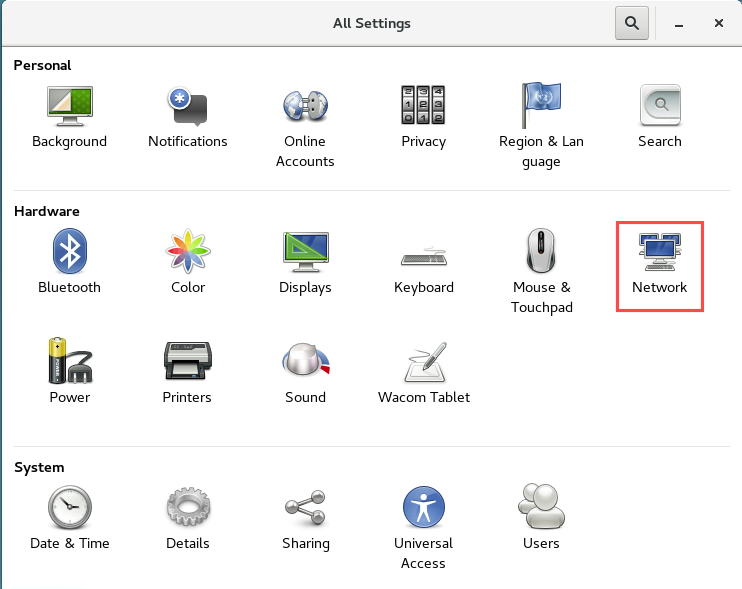
### Simulate the loss of the Primary node

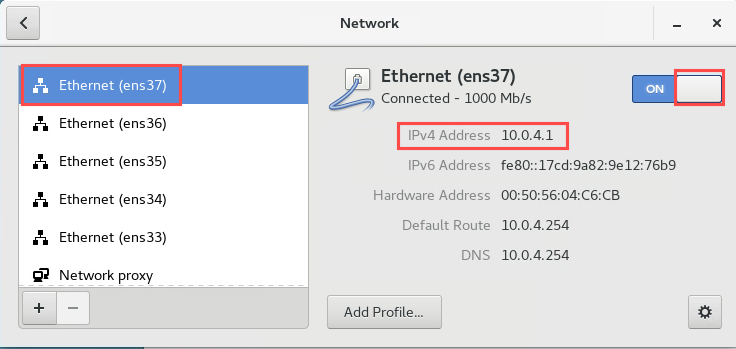
Although the node has not been lost, you will simulate it by disabling the DR Replication Network adapter and deleting the queue manager.

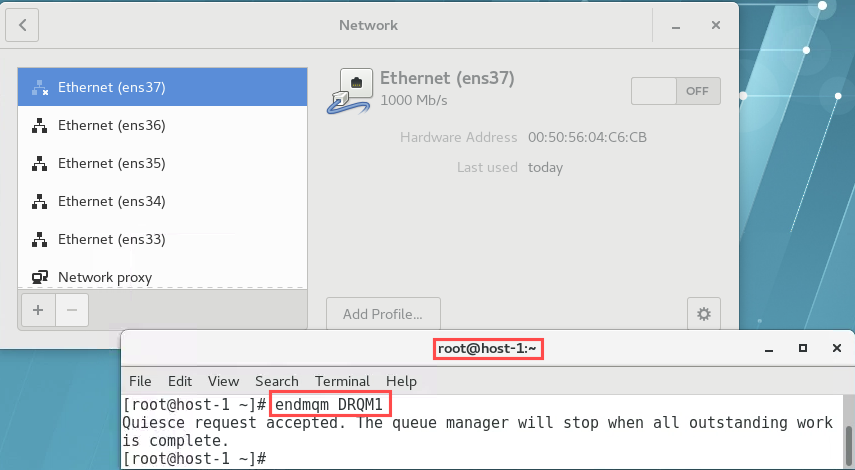
1. On node miqmp, go to Applications -> System Tools -> Settings:



1. Go to Network:



1. For the DR Replication adapter (IP address 10.0.4.1).
2. Click the button to switch it off.
3. 
4. On node miqmp, stop the queue manager:

* *endmqm DRQM1*
* 

1. Remove the queue manager:

* *dltmqm DRQM1*
* 

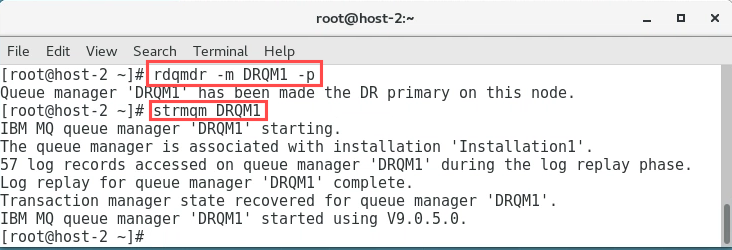
### Make the Secondary instance the Primary

Designate the Secondary node as the Primary instance.

1. On the recovery node miqms, designate it as the primary instance:

* *rdqmdr -m DRQM1 -p*

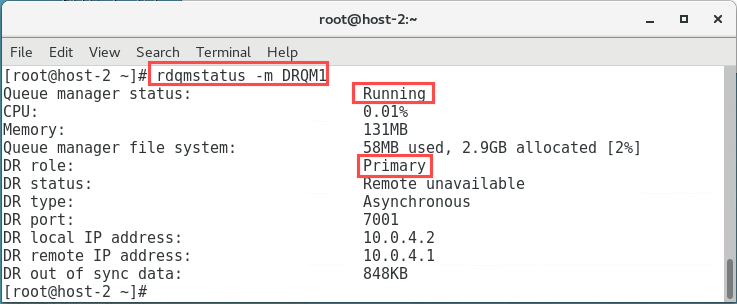
1. Start the queue manager:

* *strmqm DRQM1*
* 

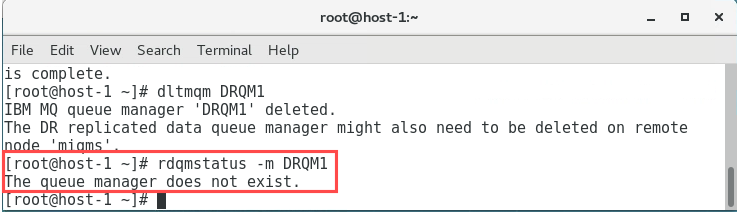
1. Confirm the status of both nodes:

* *rdqmstatus -m DRQM1*

1. On miqms:



1. As there is no longer a queue manager defined on node miqmp, the output should look similar to the following:



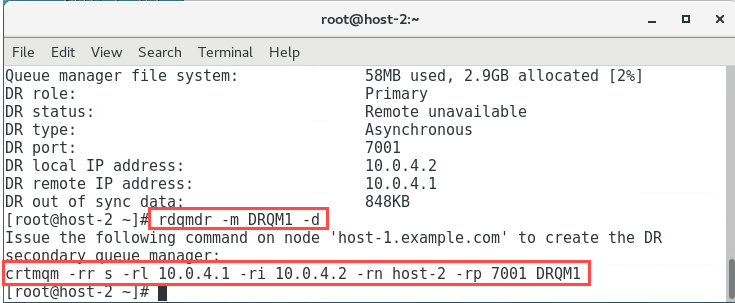
### Add the new Primary node into the DR configuration

For the replacement node to be brought back into the DR configuration, it must assume the identity of the failed node: the name and IP address must therefore be the same.

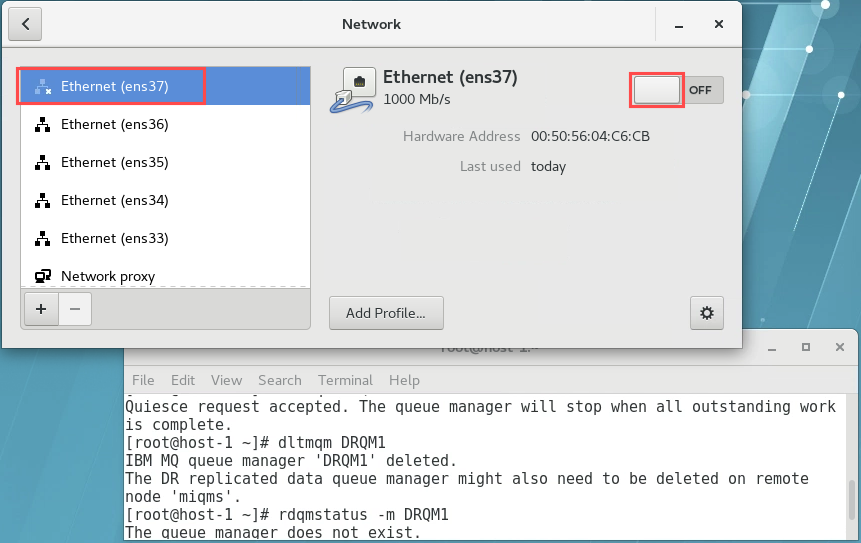
1. You will determine the command that needs to be run on the new Primary node. On node miqms, run the command :

* *rdqmdr -m DRQM1 -d*

1. The output should look similar to the following:



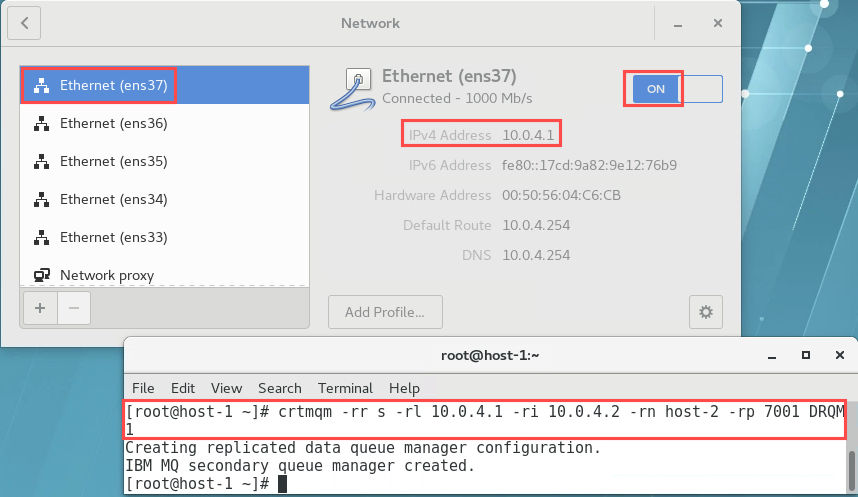
1. On node miqmp, restart the DR Replication network interface. Go to Applications -> System Tools -> Settings -> Network.



1. Copy the command (as highlighted above) into the command line of the new Primary node, miqmp, to run it:

* *crtmqm -rr s -rl 10.0.4.1 -ri 10.0.4.2 -rn host-2 -rp 7001 DRQM1*

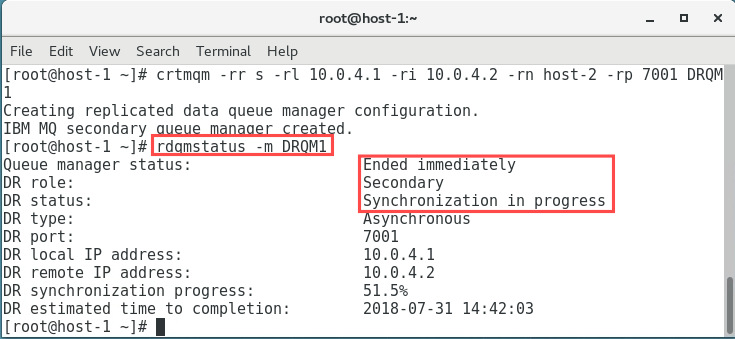
1. The output should look similar to the following:



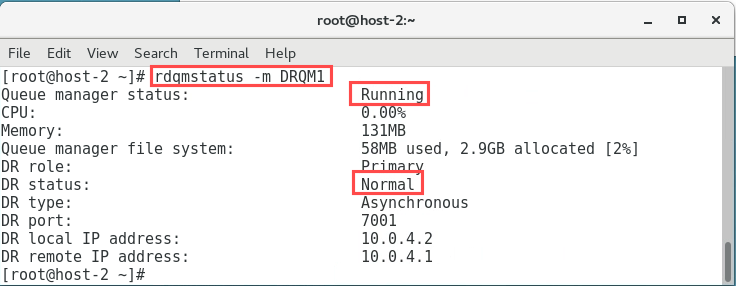
1. Check the status of the synchronisation on both nodes:

* *rdqmstatus -m DRQM1*

On miqmp:



On miqms:



### Restore the original DR configuration

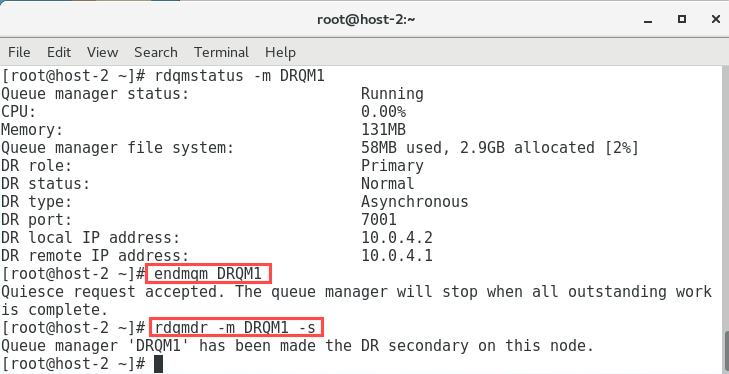
To restore the original DR configuration, you would want to designate the Primary node as the Primary instance again.

1. When the initial synchronisation is complete on the primary node miqmp, you can designate the secondary node as the secondary instance again. End the queue manager on node miqms:

*endmqm DRQM1*

1. Designate node miqms as the secondary instance again:

*rdqmdr -m DRQM1 -s*

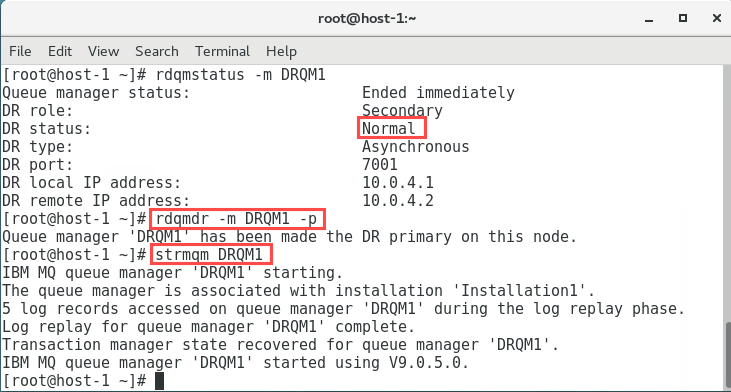


1. On node miqmp, designate it as the primary instance again:

*rdqmdr -m DRQM1 -p*

1. Start the queue manager on the primary node:

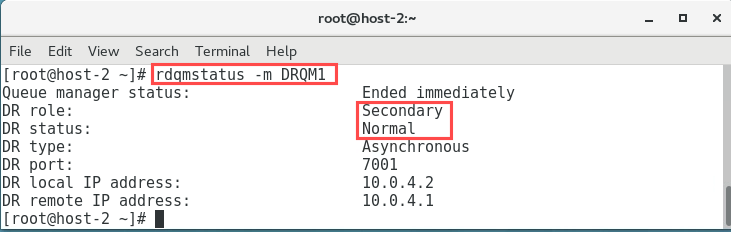
*strmqm DRQM1*



1. Confirm the status of both nodes:

*rdqmstatus -m DRQM1*

1. On miqmp:



1. On miqmp:

## Replace node that was running a DR Secondary

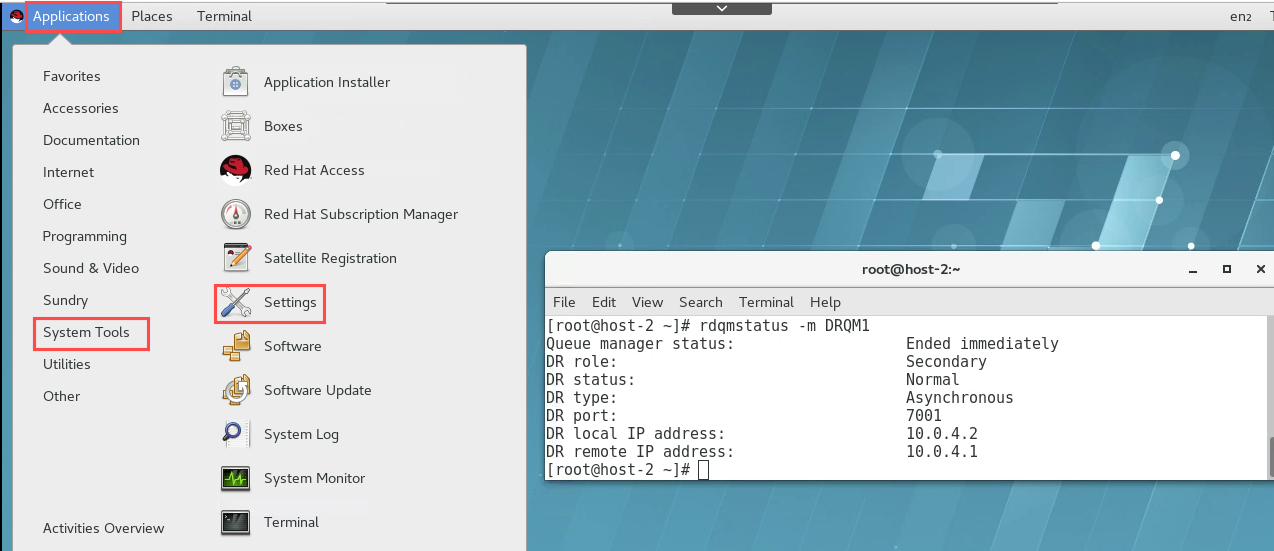
If it is a secondary node that needs to be replaced, you would just replace it and restore it to the original disaster recovery configuration.

### Add the new Secondary node into the DR configuration

To simulate this, there is no need to change the DR designations, prior to the Secondary node being replaced. You will simply disable the DR Replication Network adapter.

For the replacement node to be brought back into the DR configuration, again it must assume the identity of the failed node: the name and IP address must be the same.

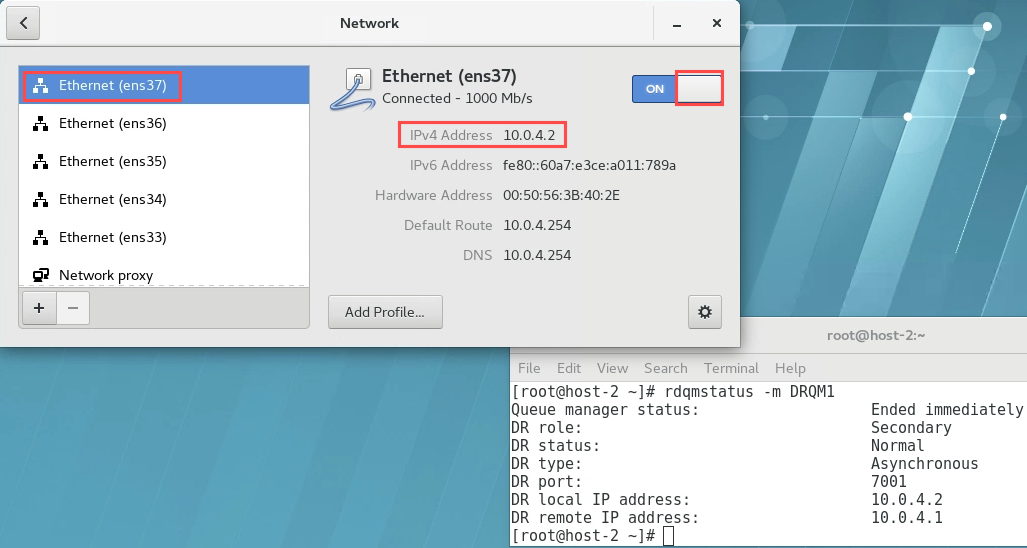
1. On node miqms, go to Applications -> System Tools -> Settings.



1. Select Network.

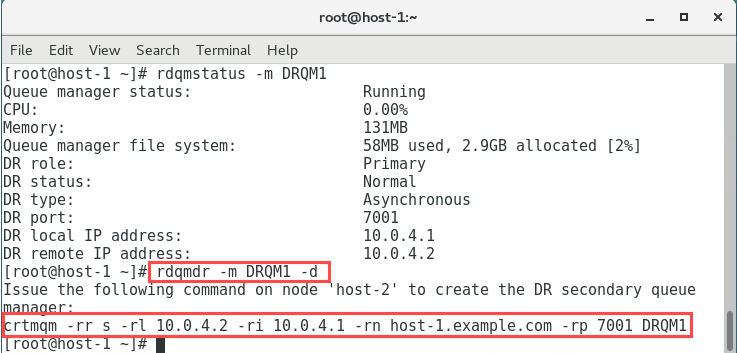


1. For the DR Replication adapter (IP address 10.0.4.2). Click the button to switch it off.

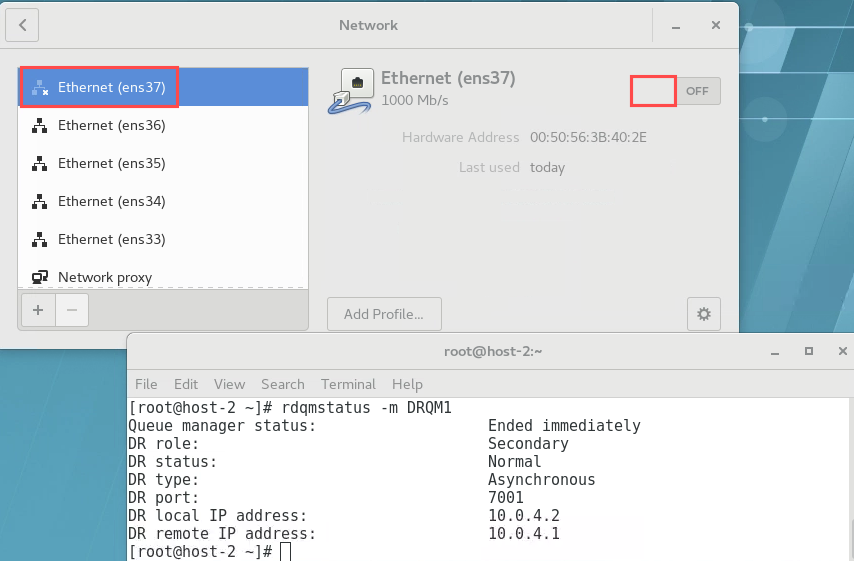


1. You will assume the secondary node has been replaced. Determine the command that needs to be run on the new Secondary node. On node miqmp, run the command :

*rdqmdr -m DRQM1 -d*

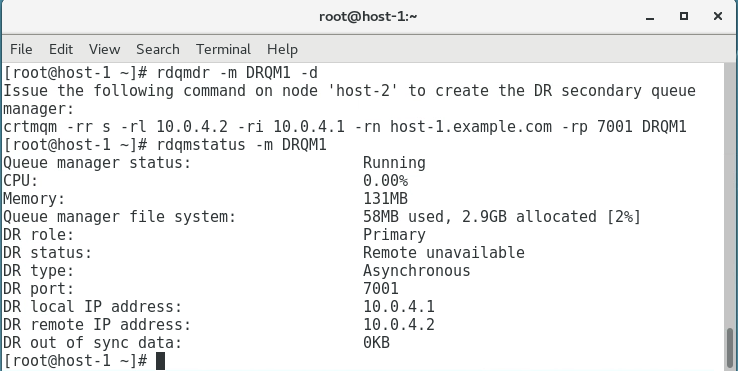


1. On node miqms, go to Applicatiions -> System Tools -> Settings -> Network. Click the button for the DR Replication adapter (IP address 10.0.4.2), to switch it back on.



1. Copy this command into the command line of the new Secondary node, miqms, then run it:

* *crtmqm -rr s -rl 10.0.4.2 -ri 10.0.4.1 -rn host-1.example.com -rp 7001 DRQM1*

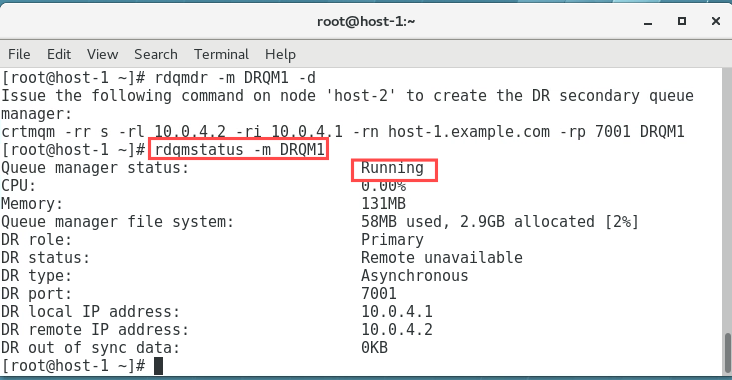


1. Check the status of the synchronisation on both nodes:

*rdqmstatus -m DRQM1*

1. When the initial synchronisation has completed, on the secondary node miqms, designate it as the secondary instance:

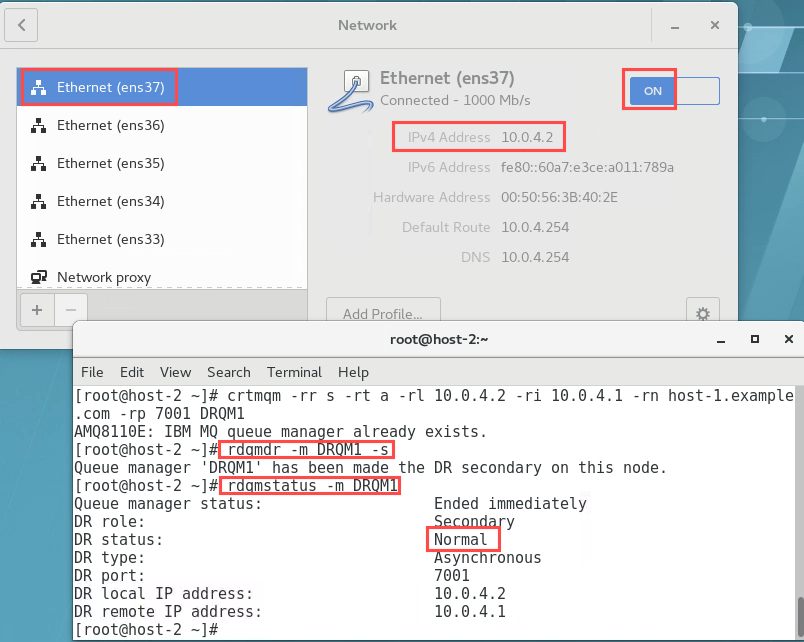
*rdqmdr -m DRQM1 -s*



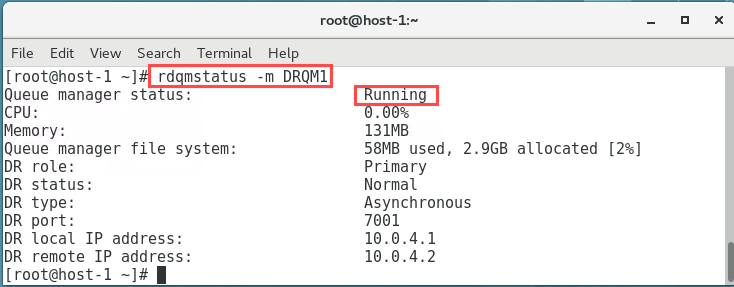
1. Confirm the status of the DR configuration on both nodes:

rdqmstatus -m DRQM1

On miqms:



On miqmp:



## Reverting to a snapshot

Suppose a network connection between the nodes is lost, the changes to the persistent data for the primary instance of a queue manager are tracked. When the network connection is restored, a synchronisation process is used to get the secondary instance up to speed as quickly as possible.

While synchronization is in progress, the data on the secondary instance is in an inconsistent state. A snapshot of the state of the secondary queue manager data is taken.If a failure of the main node or the network connection occurs during synchronization, it would be necessary to revert the secondary instance back to this snapshot.

### Create a snapshot

You will first create an Inconsistent state on your DR nodes.

1. On miqmp, open a new terminal and switch user to ibmdemo. Enter *MQpot2018* for ibmdemo's password when prompted.

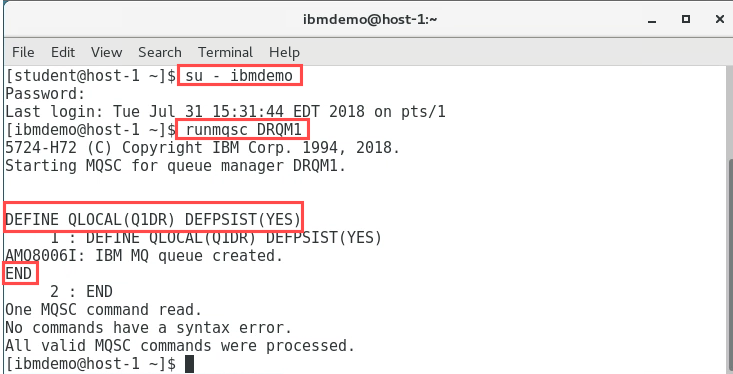
*su - ibmdemo*

1. Create a local queue for placing messages to provide some data for later synchronisation. Use the runmqsc command to create a local, persistent queue, called Q1DR.

*runmqsc DRQM1*

*DEFINE QLOCAL(Q1DR) DEFPSIST(YES)*

*end*



1. Check the status on both nodes is normal.

*rdqmstatus -m DRQM1*

1. Get ready to simulate a failure of the DR replication network adapter.On node miqmp, open a new window, as user *root / Passw0rd!*. Use the iptables rules to remove all output packets from the network adapter interface.

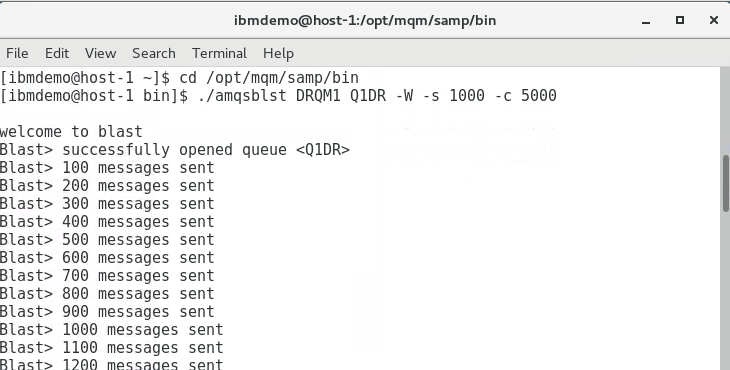
*su –*

*iptables -I OUTPUT -o ens35 -j DROP*

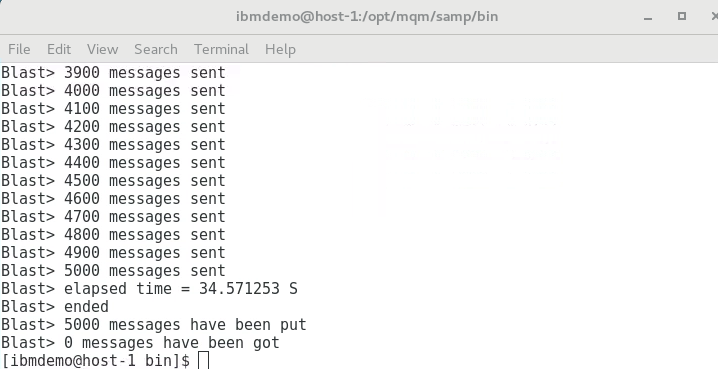
1. DO NOT PRESS ENTER YET - the rule will be applied BEFORE all the messages have been put on the queue.
2. In the user ibmdemo window, start putting some messages onto the queue. Use the amqsblst sample to do this:

*cd /opt/mqm/samp/bin*

./amqsblst DRQM1 Q1DR -W -s 1000 -c 5000

The output should look similar to the following:  
  


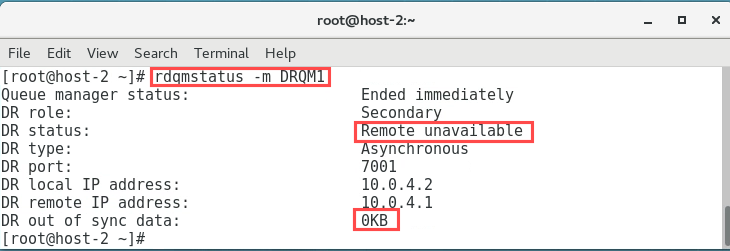
1. Before all the messages have been put onto the queue, in the user root window, press <enter> to add the iptables rule. This will simulate a network outage on node miqmp. There will be a short pause, then the placing of messages will resume.



1. In another window as user ibmdemo, check the status on both nodes

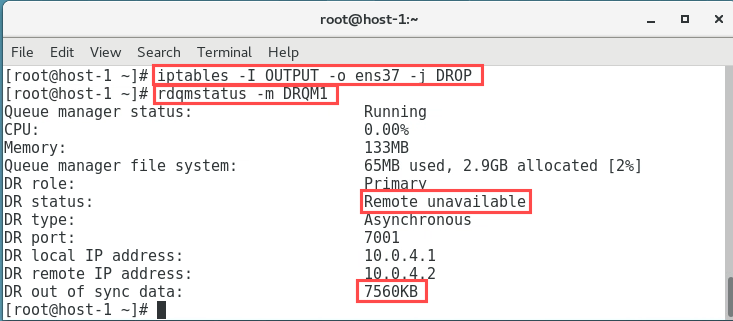
*rdqmstatus -m DRQM1*

Notice on node miqmp, the DR status is showing as ‘Remote unavailable’.



Similarly on node miqms:

1. Issuing the command again on node miqmp, when all the messages have been placed on the queue, you will notice the ‘DR out of sync data’ has changed.



1. Simulate the restoration of the network outage on node miqmp by issuing an iptables remove previous rule command. In the user root window:

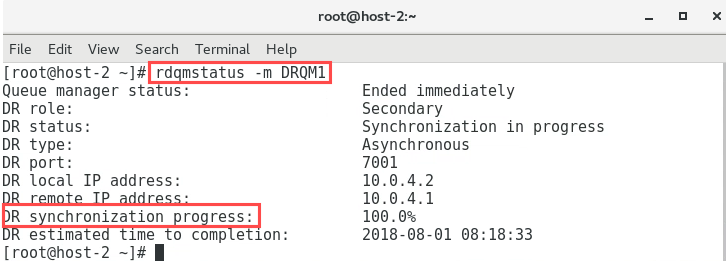
*iptables -D OUTPUT -o ens35 -j DROP*

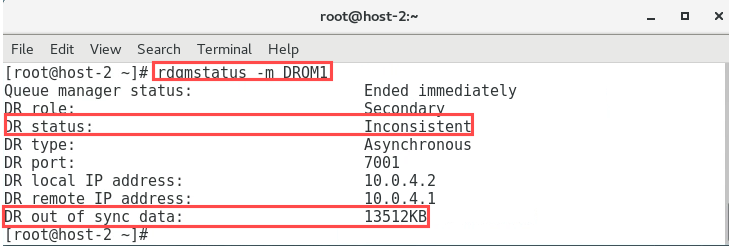
1. The nodes will start synchronising as soon as this happens. Immediately (before synchronisation is complete), simulate a network outage on node miqmp again.

*iptables -I OUTPUT -o ens35 -j DROP*

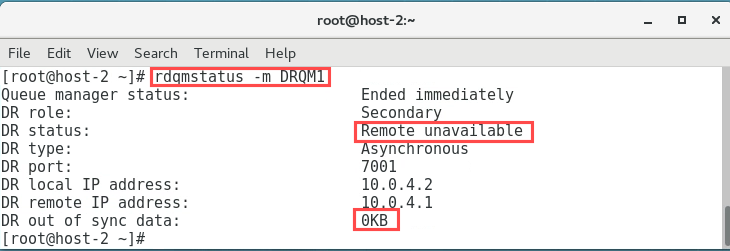
1. Checking the status on both nodes immediately before switching off the network will look similar to the following:

*rdqmstatus -m DRQM1*

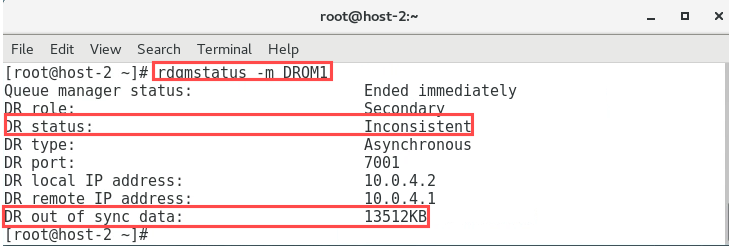




1. When the network is detected to have failed again, the status on the primary node, miqmp goes back to ‘Remote unavailable’.



1. The status on the secondary node, miqms is ‘Inconsistent’.



1. There is also an indication of the amount of data that is out of synchronisation on both nodes.

### Revert to a snapshot

You will now see how the secondary instance reverts to its snapshot and the queue manager data. Note any updates that have happened since the original network failure will however be lost.

1. The assumption now is that the Primary node is no longer usable, so the replication node must be made the new Primary instance. Designate miqms as the primary instance:

*rdqmdr -m DRQM1 -p*

1. Due to its former ‘Inconsistent’ state, miqms will revert to a snapshot. Check the status to confirm this.

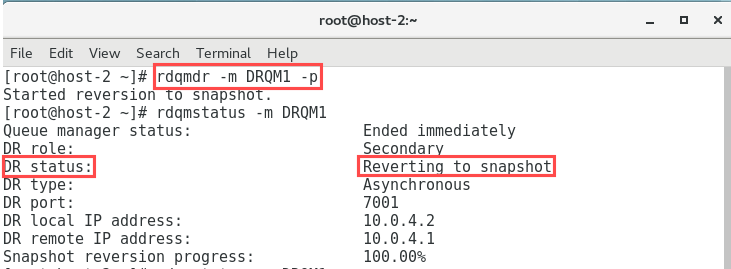
*rdqmstatus -m DRQM1*

The output should look like the following:

1. When node miqms has completed reverting to the snapshot. Checking the status again.

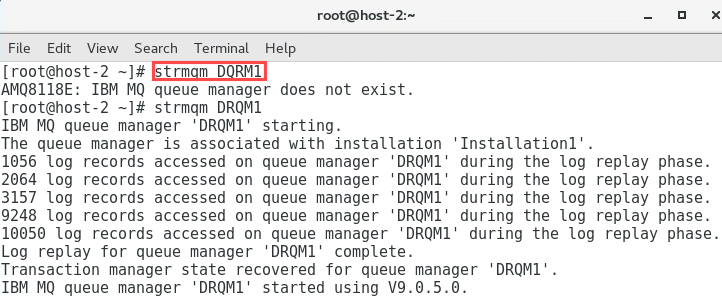
*rdqmstatus -m DRQM1*

1. should look similar to the following:



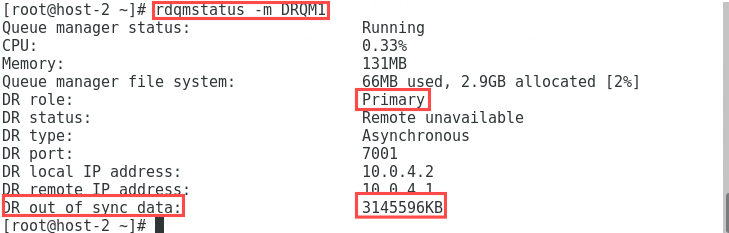
1. Notice the status indicates the queue manager ‘Ended unexpectedly", and there is data that is out of synchronisation.
2. As this situation would have occurred as a result of a possible failure of the Primary node, in reality you would go through the process described earlier to ‘Add the new Primary node into the DR configuration’.
3. Once the new Primary node was part of the DR configuration again, you would follow the steps to ‘Restore the original DR configuration’ of the Primary node being the primary instance of the queue manager.
4. Here you will simulate something similar. You will start the queue manager on node miqms. You will delete the queue manager on node miqmp, and go through the latter of the above again, to show some additional screens not seen previously.
5. Start the queue manager on node miqm2, which is now the primary instance.

*strmqm DRQM1*



1. Confirm the queue manager on node miqms is running as the primary instance by checking the status.

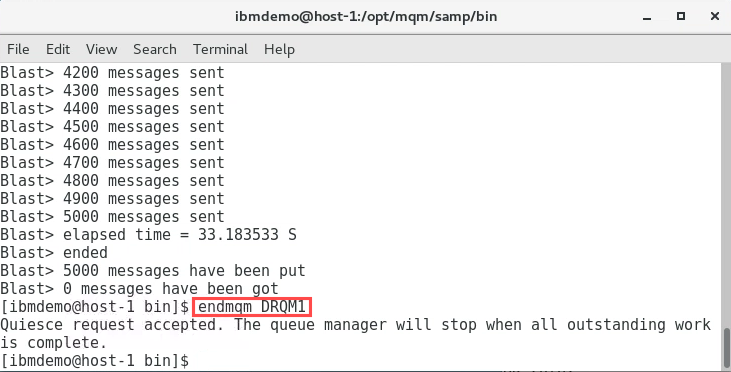
*rdqmstatus -m DRQM1*



Notice that as a result of reverting to a snapshot there is data out of synchronisation.

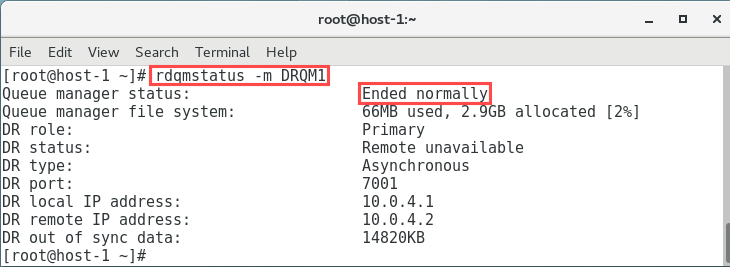
1. ‘Simulate’ the replacement of node miqmp. On node miqmp, in the user ibmdemo window stop the queue manager:

*endmqm DRQM1*



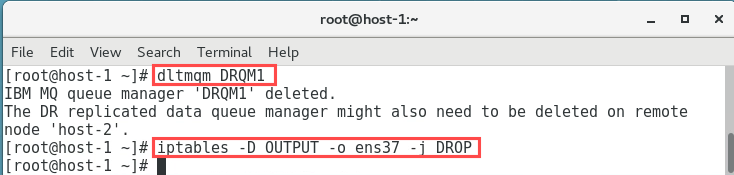
1. Confirm the queue manager on node miqmp ended normally by checking the status

*rdqmstatus -m DRQM1*



1. Delete the queue manager on node miqmp.

*dltmqm DRQM1*

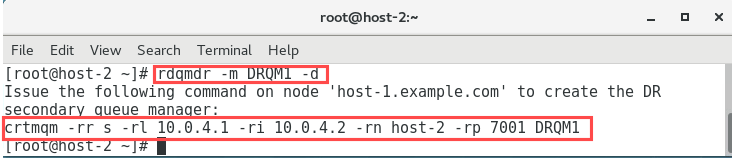


1. On node miqmp, as user root restart the network interface.

*iptables -D OUTPUT -o ens35 -j DROP*

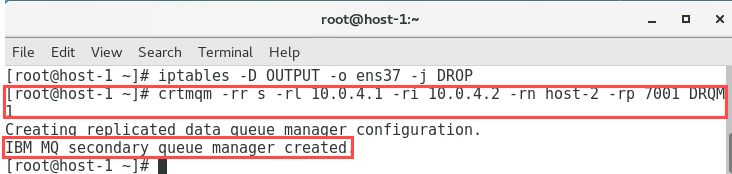
1. On node miqms, enter the command to determine the command needed to recreate the queue manager on node miqmp.

*rdqmdr -m DRQM1 -d*



1. On node miqmp, as user root issue the command to recreate the queue manager:

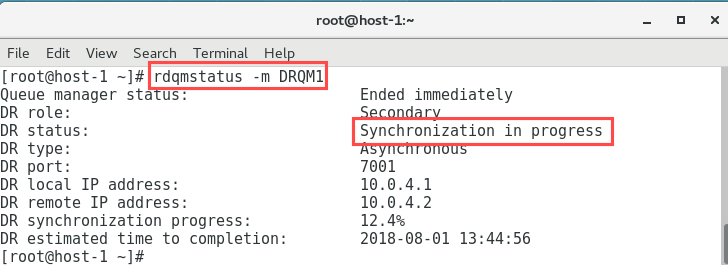
*crtmqm -rr s -rl 10.0.4.1 -ri 10.0.4.2 -rn host-2 -rp 7001 DRQM1*



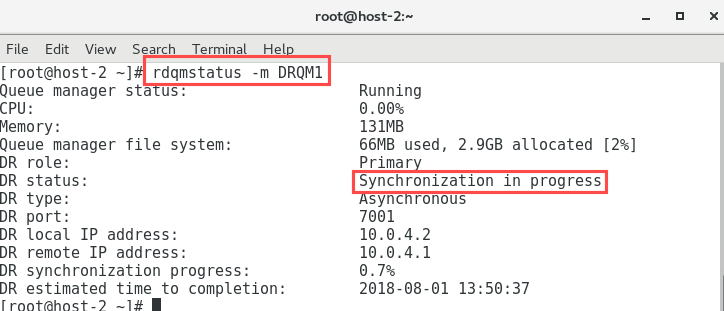
1. Check the status on both nodes:

*rdqmstatus -m DRQM1*

On miqmp:

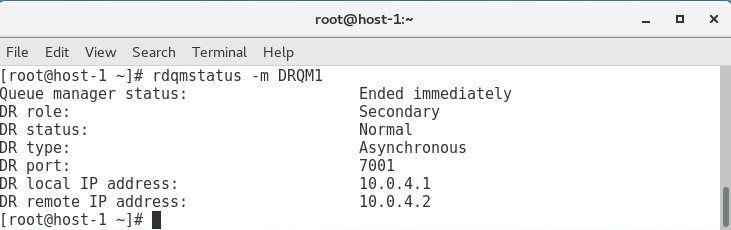


On miqms:

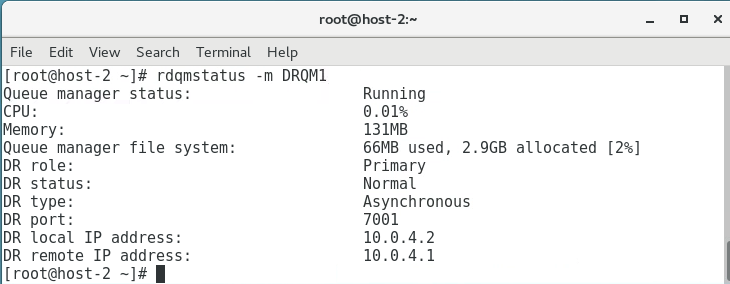


1. Notice that synchronisation of data is taking place between the nodes. Indications are given on its progress and estimated completion time.
2. When data synchronisation has completed the status of the nodes will look similar to the following:

On miqmp:

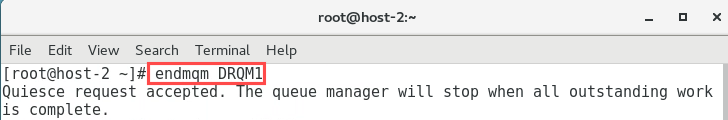


On miqms:



1. Restore the DR configuration. On node miqms, as user root, stop the queue manager

*endmqm DRQM1*



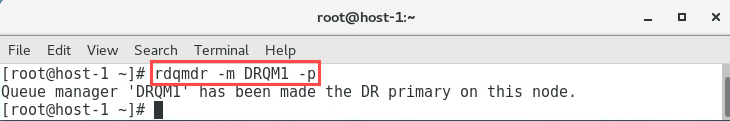
1. Make node miqms the secondary instance:

*rdqmdr -m DRQM1 -s*



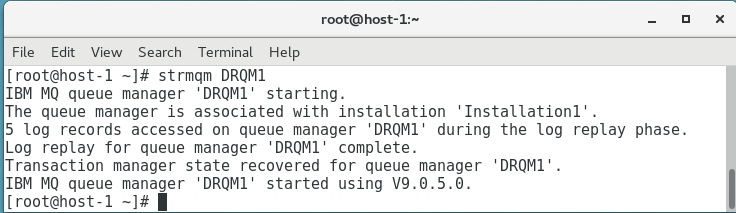
1. On node miqmp, as user ibmdemo, make it the primary instance of the queue manager:

*rdqmdr -m DRQM1 -p*



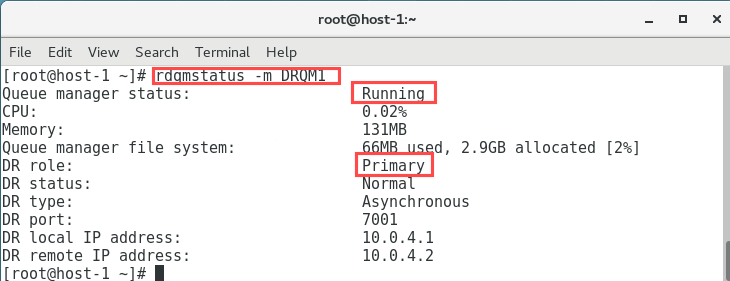
1. Start the queue manager on node miqmp:

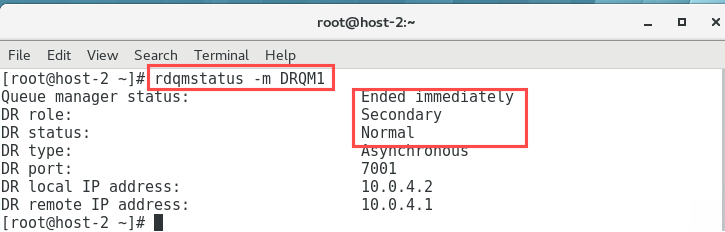
*strmqm DRQM1*



1. Confirm the DR configuration by checking the status on both nodes:

*rdqmstatus -m DRQM1*





## Delete a DR RDQM

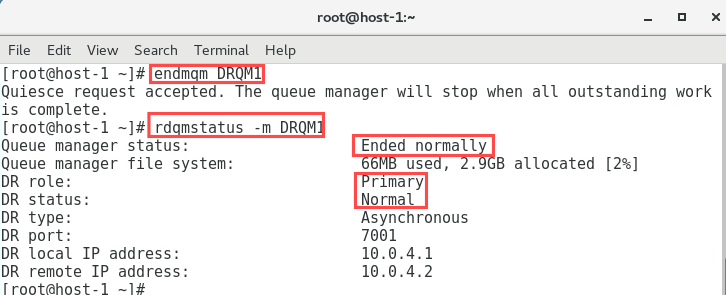
If Disaster Recovery is no longer required for a queue manager, the queue manager needs to be deleted to be removed from the DR configuration. This is achieved as follows:

1. On miqmp stop the queue manager:

*endmqm DRQM1*

1. View the status to confirm that has happened :

*rdqmstatus -m DRQM1*



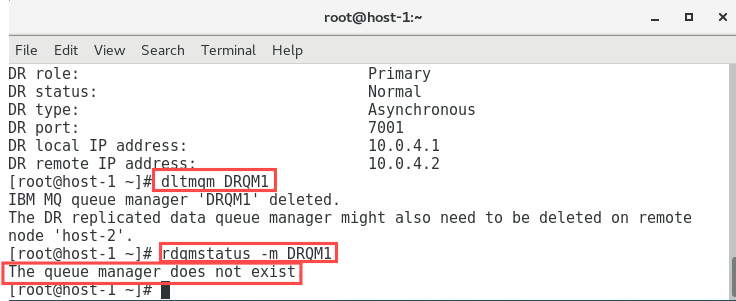
1. On miqmp remove the queue manager:

*dltmqm DRQM1*

1. Viewing the status:

*rdqmstatus -m DRQM1*

1. will confirm the queue manager no longer exists on the primary node.



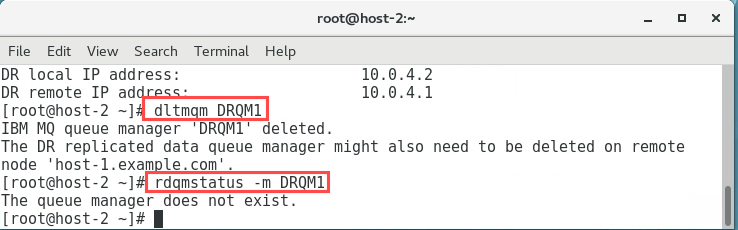
1. Also remove the queue manager on miqms:

*dltmqm DRQM1*

1. Again, viewing the status:

*rdqmstatus -m DRQM1*

1. will confirm the queue manager no longer exists on the secondary node either.



## CONGRATULATIONS!

### You have completed this hands-on lab.

You have created replicated data queue managers to provide high availability for IBM MQ.