Replicated Data Queue Managers (RDQM) for High Availability (9.1.5)

Messaging Administrator

Summary: Replicated Data Queue Managers (RDQM) for High Availability

Introducing IBM MQ Replicated Data Queue Manager

IBM MQ High Availability Options

For many years there has been two well established options for IBM MQ single site high availability:

- **Multi-Instance Queue Managers**: An out of the box technology that requires a shared disk (such as NFS) for storing the Queue Manager data, and two severs in an active / standby configuration. If the active instance was to fail, the lock on the shared disk would be released, and the passive instance promoted to the active instance.
- **Operating System High Availability technologies**: IBM MQ explicitly supports a number of well-established operating system high availability technologies, such as Microsoft Cluster Service, PowerHA for AIX, Veritas Cluster Server, HP Serviceguard, or a Red Hat Enterprise Linux cluster with Red Hat Cluster Suite.

Each of the above have technical considerations and dependencies; for instance, the multi-instance queue manager is dependent on a shared disk, which must be redundant to provide effective high availability; while the operating system approach requires deep knowledge of the operating system, and in certain cases additional license entitlement for the operating system.

IBM MQ V9.0.4 Continuous Delivery Release provided a new option to complement the existing two, called MQ Replicated Data Queue Manager (RDQM). This technology is based on the MQ Appliance high availability capabilities, and is available on the Linux operating system. In the simplest of situations, it will include three standard RedHat Linux servers, one being active handling requests, with the other two replicating the data and waiting to become active in a similar logical manner to the standby instance within a Multi-Instance Queue Manager. When comparing to multi-instance, there are three key differences:

- **No shared disk**: Each individual server includes a complete replica of the Queue Manager data, removing the need for a shared disk, which simplifies the configuration, and potentially improves the performance of the overall solution.
- **Quorum based promotion**: Due to each server including a complete replica, in the case of a network failure between the servers, all could be promoted to be active, and a condition called “split brain” could occur. To avoid this from happening, three servers are part of the RDQM HA group, and a server will only continue to be active or promoted to the active instance if a quorum of the servers within the RDQM HA group are contactable.
- **Floating IP**: With Multi-Instance Queue Managers, two servers, with separate IP addresses could be hosting and running the active instance of the Queue Manager. This means that the client needs to be explicitly aware of the two IP addresses. With the new RDQM, a floating IP address is used, and associated with the server that is running the active queue manager. This means that clients only need to be aware of the floating IP address, simplifying the logic on the client.

Although the above are all drivers of why to consider the new capability, there are some points to consider when evaluating if this new HA model is suitable for a client:

- **Restricted to a single Data Center**: The replication between instances within a RDQM HA group is synchronous, and therefore if the servers are not co-located within the same data center, this can affect the performance of the solution. Also, since a floating IP address is used, this normally is limited to a single data centre.
- **Distributed Writes**: As mentioned above, the writing is completed using a synchronous write, and therefore involves writing to multiple disks across multiple servers. The performance characteristics of this approach, compared to the other options, need to be evaluated.
- **Dedicated Logical Volume Group**: Each RedHat Linux server requires a dedicated logical volume for the RDQM data. This normally means a separate disk associated with the VM or bare metal machine dedicated to the RDQM data. If the server is being specially provisioned for IBM MQ, this is unlikely to be a major issue.
- **Split Brain**: The quorum based approach to electing primaries greatly reduces the chances of a split brain situation occurring, but does not completely remove all edge cases.
- **RedHat Enterprise Linux 7.3, 7.4, 7.5, 7.6, or 7.7**: The RDQM capability is currently supported on RedHat Enterprise Linux 7.3, 7.4, 7.5, 7.6, and 7.7. Depending on the environment, this may or may not be a significant restriction.

To assist with the setup of RDQM, we’ve created a number of scripts and helper files. The following is based on the process required for a base RedHat Enterprise Linux 7.7 install.

Lab Introduction

This lab provides a demonstration of a new approach to High Availability in MQ on Linux, with the following key

features:

- Use of Distributed Replicated Block Device (DRBD) storage rather than network shared storage
- Use of a cluster resource manager (Pacemaker) to manage a cluster of three nodes
- A new kind of queue manager called a Replicated Data Queue Manager (RDQM):
  - an RDQM is active on only one node at any one time
  - each node can run different active RDQMs
  - each RDQM has a preferred location (node) in normal operation
  - a quorum prevents an RDQM from running on more than one node at the same time
  - a RDQM can have a floating IP address associated with it to simplify configuration of clients and other queue managers that need to communicate with the RDQM

Lab environment

- Three RHEL 7.7 x86_64 systems running in Skytap:
  - rdqm1 - This will be our primary node.
  - rdqm2 - This will be a secondary node.
  - rdqm3 - This will be another secondary node.
  - dr1 - DR Fail over primary node.
  - dr2 - DR Fail over secondary node.
  - dr3 - DR Fail over other secondary node.
- VMWare Workstation virtual networks:

Name	Type	SkyTap Network	Subnet	DHCP
ens34	Administration	ens34	10.0.1.0	no
ens35	HA Replication	ens35	10.0.1.0	no
ens36	DR Replication	ens36	10.0.2.0	no
ens37	Pacemaker primary	ens37	10.0.3.0	no
ens38	Pacemaker secondary	ens38	10.0.4.0	no

- Network interfaces:

Interface Purpose	Interface Name	rdqm1 (Primary node)	rdqm2 (Secondary node)	rdqm3 (Secondary node)
MQ Fixed IP	ens35	10.0.1.1	10.0.1.2	10.0.1.3
MQ Floating IP		10.0.1.10	10.0.1.10	10.0.1.10

HA interfaces are used as follows:

- HA Primary - to monitor the nodes in the cluster
- HA Alternate - backup for monitoring the cluster if the HA Primary network fails
- HA Replication - for synchronous data replication (the higher the bandwidth the better and the lower the latency the better)

Note: Hosts rdqm1, rdqm2, rdqm3 are tied to the Administration IP addresses above.

- Dedicated volume group “drbdpool” containing a single physical volume on each node for RDQM, but please note, you will not see any further reference to this in this document.
- The following groups configured:
  - **mqm** to allow user to run specific MQ commands
  - **haclient** to allow user to run HA-specific commands
- A normal user “ibmuser” has been defined for running applications and MQ commands.

Name	Password	Purpose	Group
root	IBMDem0s!	superuser	
ibmuser	engageibm	host vm user - MQ user	mqm

- Firewall (firewalld) enabled, and ports 1500 & 1501 will be defined during the lab.
- The following Pacemaker dependencies have already been installed. This list should be sufficient for a standard installation of MQ 9.1.5 on RHEL 7.7 Server or Workstation. For your own environment setup, or if you are using some other installation, additional packages may be needed:
  - cifs-utils
  - gnutls
  - libcgroup
  - libtool-ltdl

- lvm2
- net-snmp-libs
- nfs-utils
- perl-TimeDate
- psmisc
- PyYAML

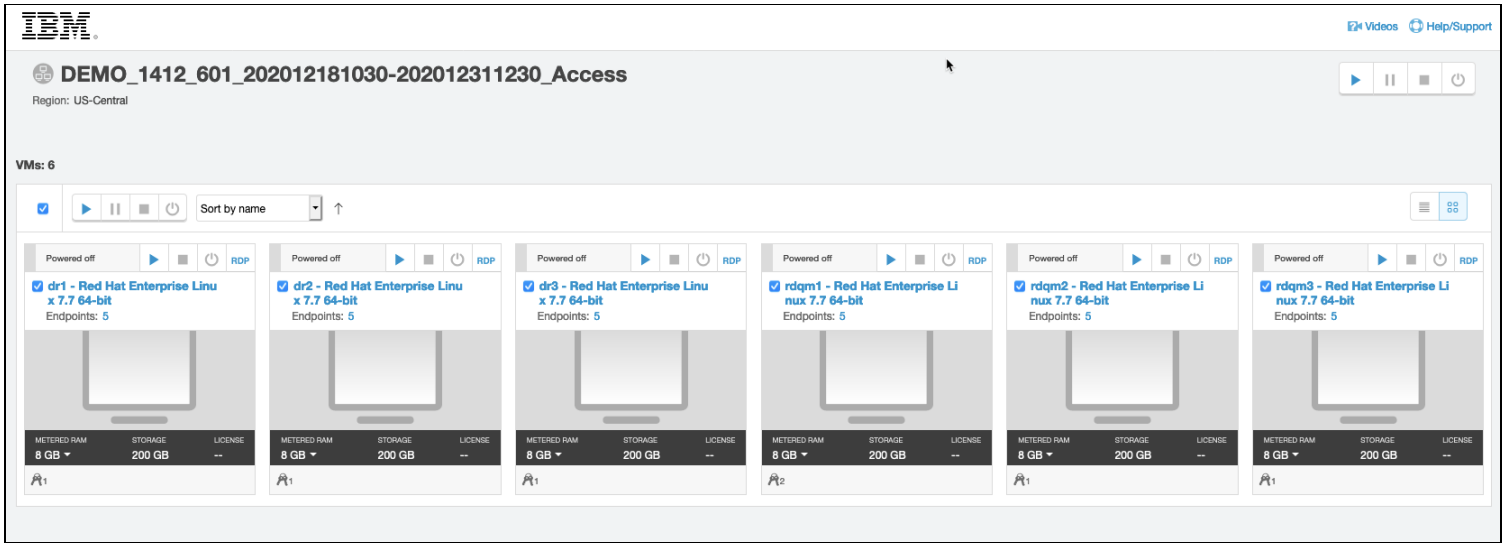
Depending on your security configuration, there are three different ways to configure the RDQM feature:

1. The simplest way is if the mqm user can ssh between the three nodes of the cluster without a password and can sudo to run the necessary commands.
2. The intermediate option is if the mqm user can sudo but not ssh. It is preferable if the actual users are also in the haclient group.
3. The default is that the mqm user cannot ssh or sudo.

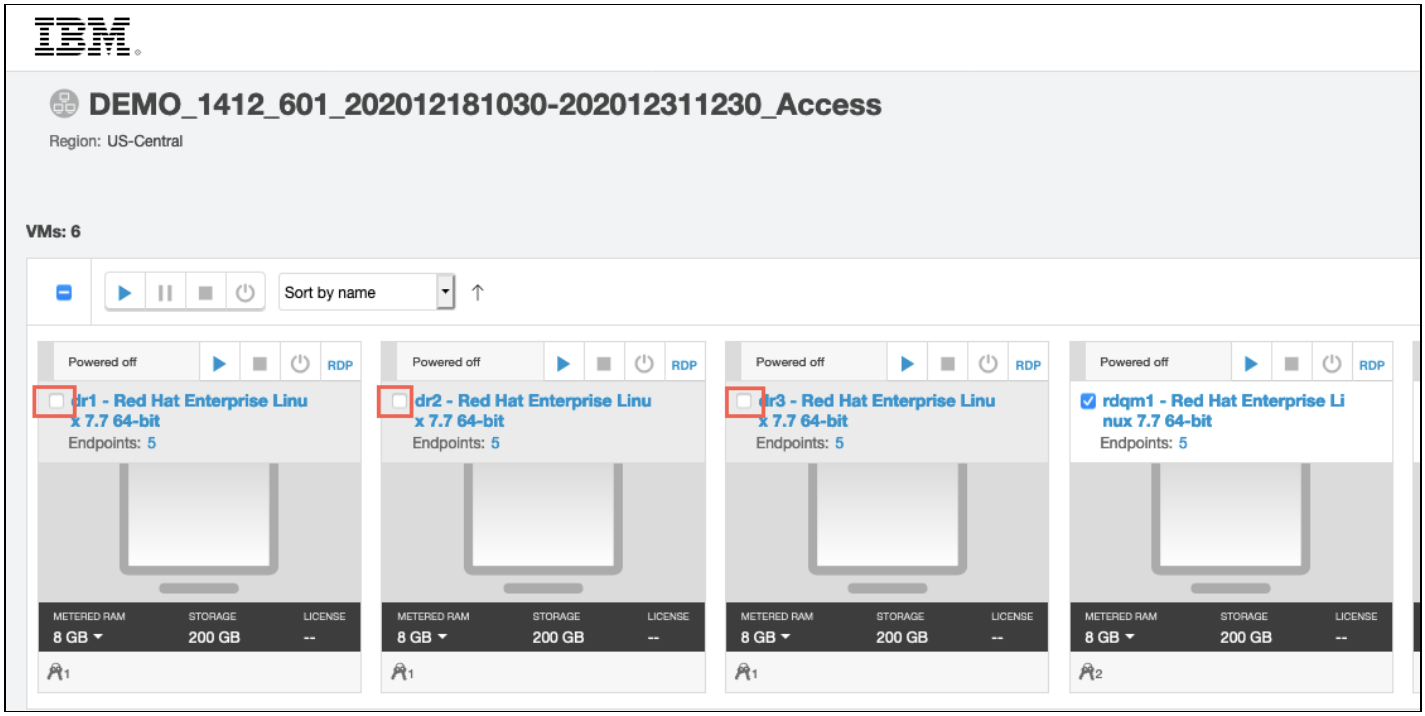
In this lab, instructions are provided to setup and test using the intermediate method.

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

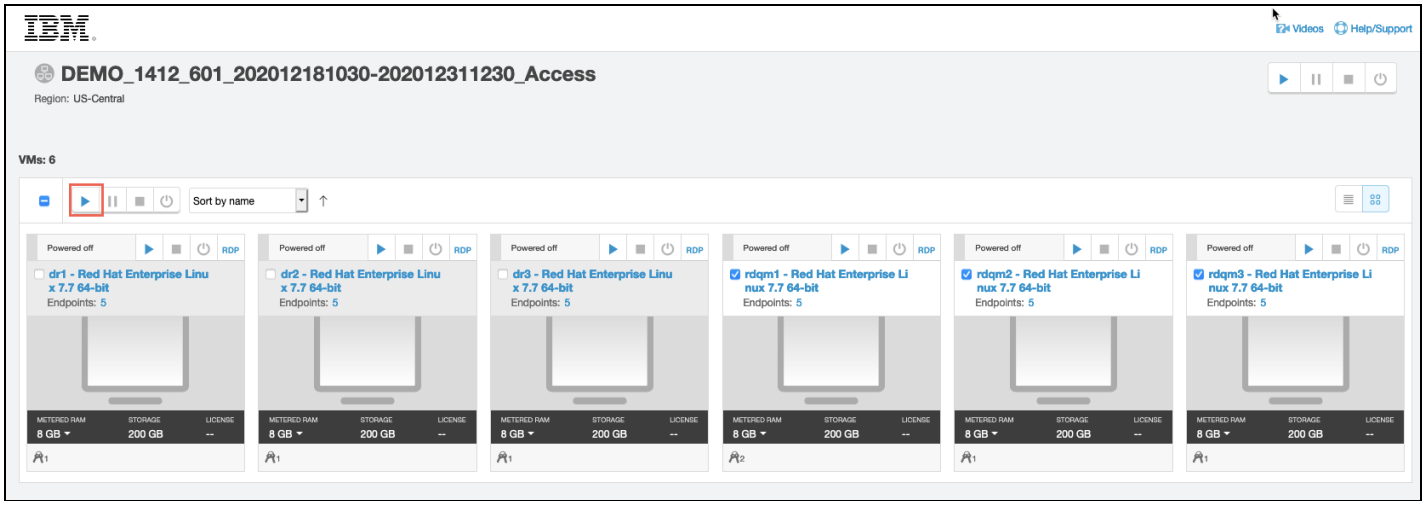
In the Skytap environment, there are six virtual machines: rdqm1, rdqm2, rdqm3, dr1, dr2, and dr3 which currently should be in a powered off or paused state.



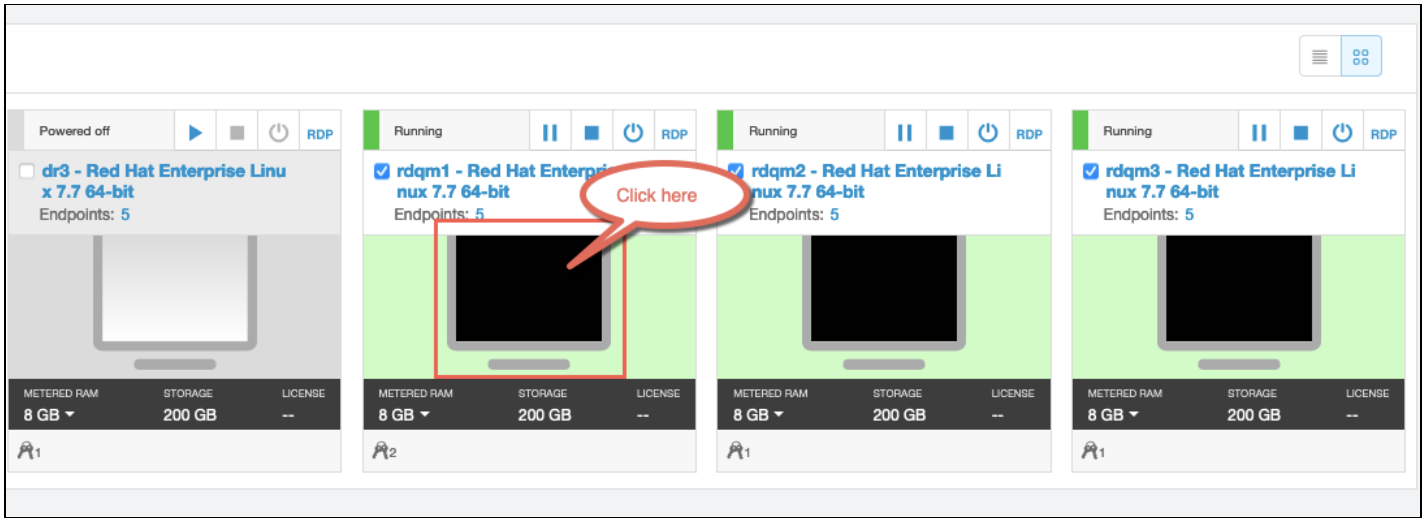
1. dr1, dr2, and dr3 will not be used in this lab, so you can leave them powered off by unchecking their labels.



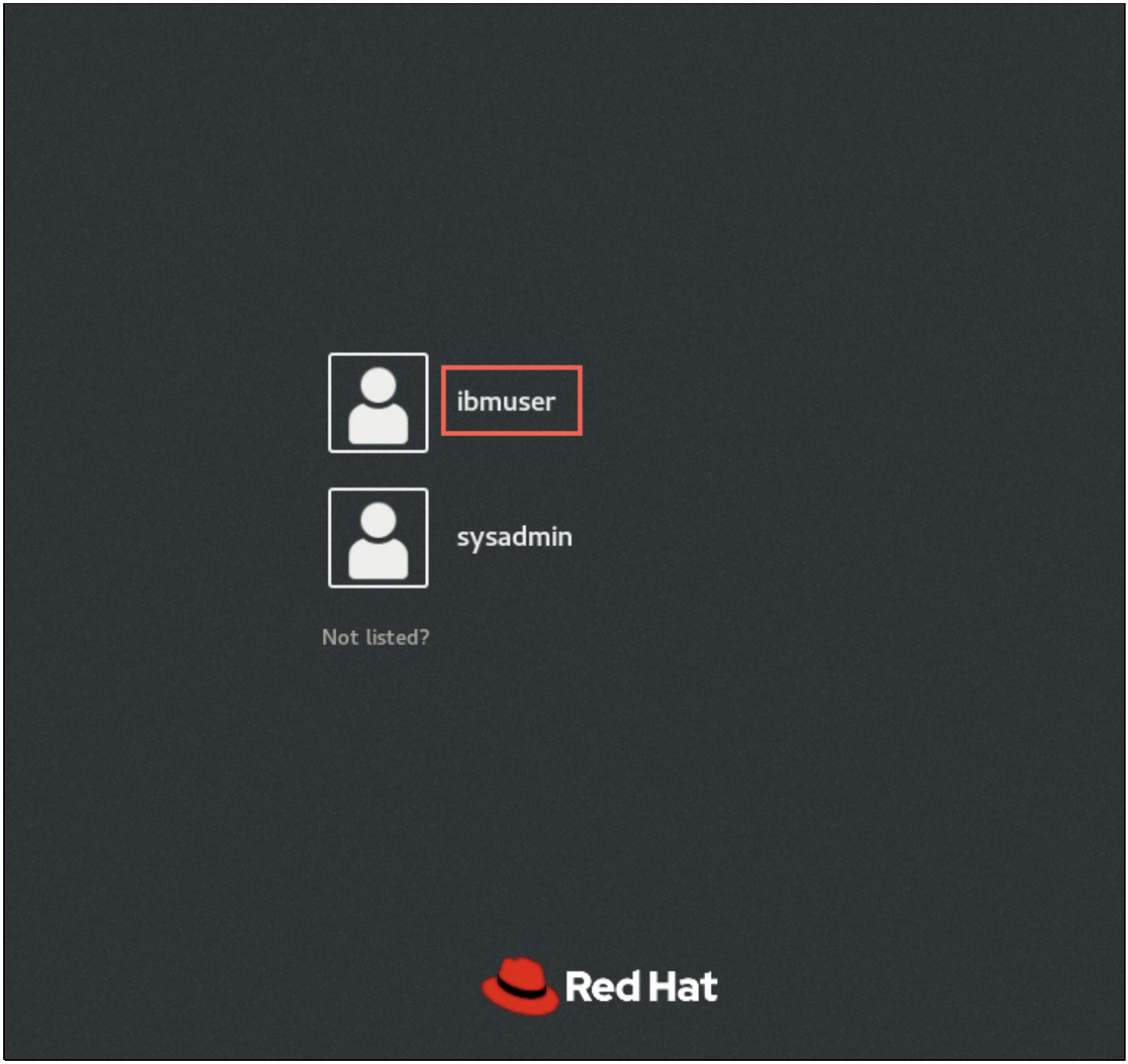
2. Click the **run** button to start or resume the VMs.



3. Wait for the monitor icons to turn green, approximately three minutes. Click the monitor icon for *rdqm1* which will launch the desktop in another browser tab.



4. Log on to VM *rdqm1* as user **ibmuser**, using password **engageibm**.



Pre-configuration steps

The following steps are necessary for configuring RDQM, and are shown for your reference. They have **already been completed** on the VMs.

- Extract and Install MQ 9.1.5

  The code is provided as a compressed tar file in the directory /home/ibmuser/.

- Install the MQ and RDQM code

RDQM is a single feature which now supports HA and DR and DR for the HA group. 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. Open another terminal, switch to user root, and run the sample file.

- 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

  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.

  If you want to allow a normal user in the mqm group to create RDQM instances etc., you need to grant the user access to certain commands via sudo. You will add the mqm user to the root and haclient group. Then add root and ibmuser to the mqm and haclient groups.

- Create the Logical Group for the QM data

  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).

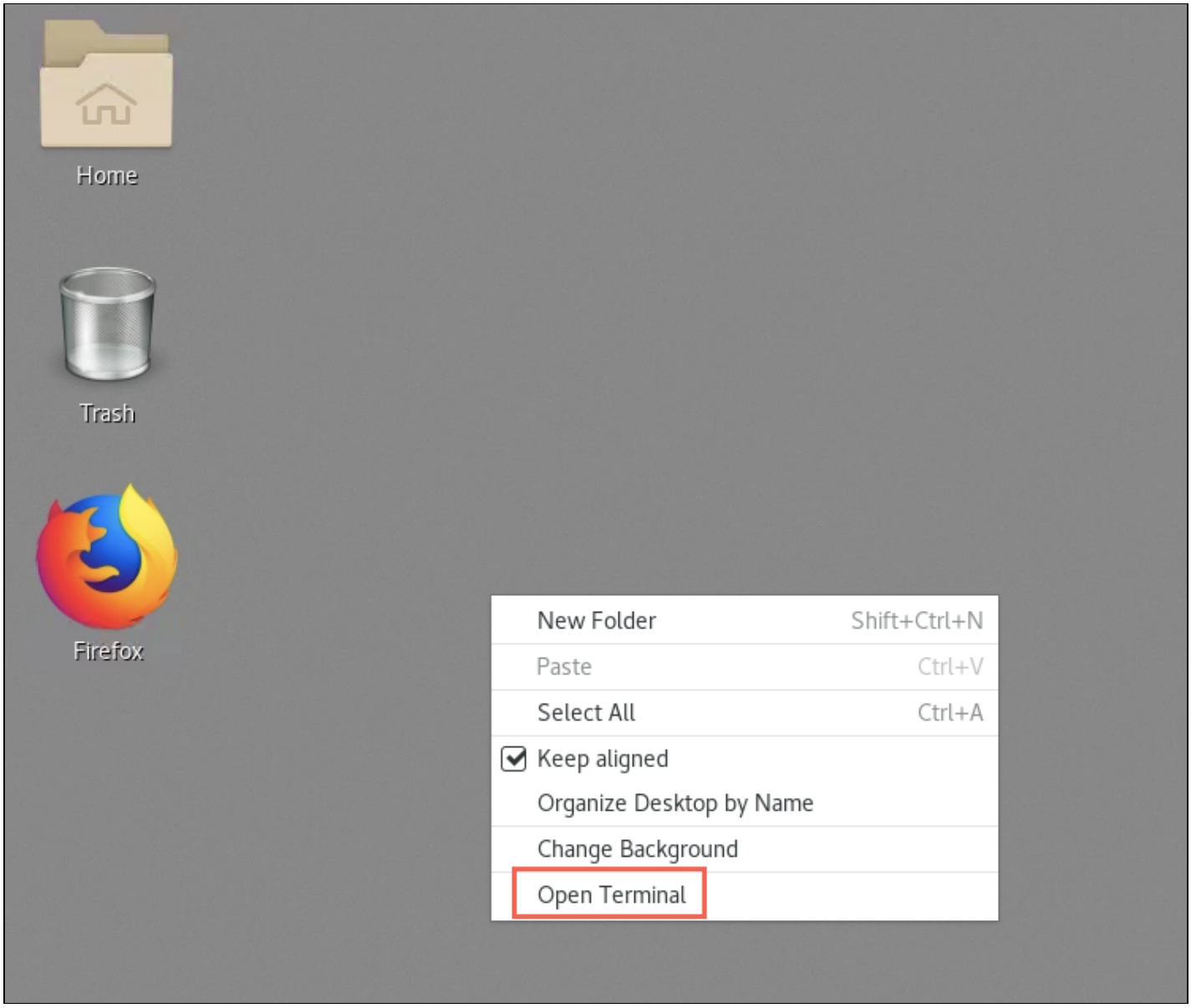
The above steps were completed on each node so at this point you are ready to begin RDQM configuration.

Configure RDQM

Install RDQM support

As previously stated, MQ 9.1.5 has already been installed on all VMs. The advanced feature RDQM support has also been installed on all the VMs *except* on **rdqm1**. You will need to install RDQM support on rdqm1 so you can see how easy it is to install. You will review the requirements for RDQM and scripts for installation and configuration.

1. On **rdqm1** open a new terminal window by right-clicking on the desktop and select Open Terminal.



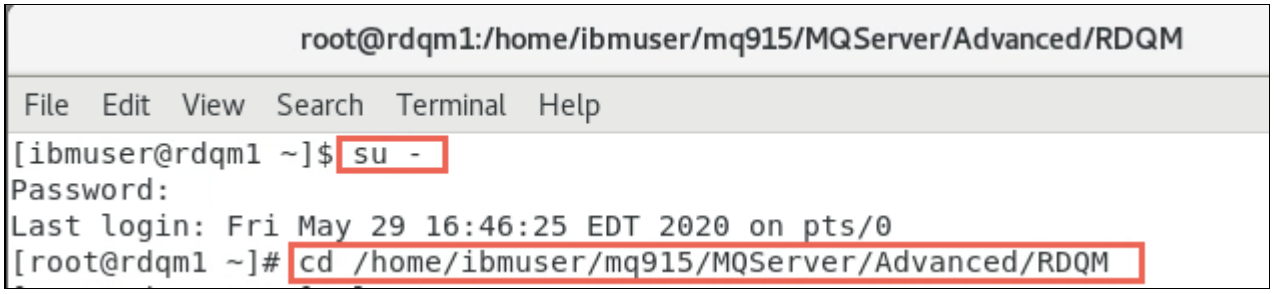
2. Switch user to root using the following command:

```
su -
```

When prompted, enter root's password *IBMDem0s!*.

3. Change to the /home/ibmuser/mq915/MQServer/Advanced/RDQM/.

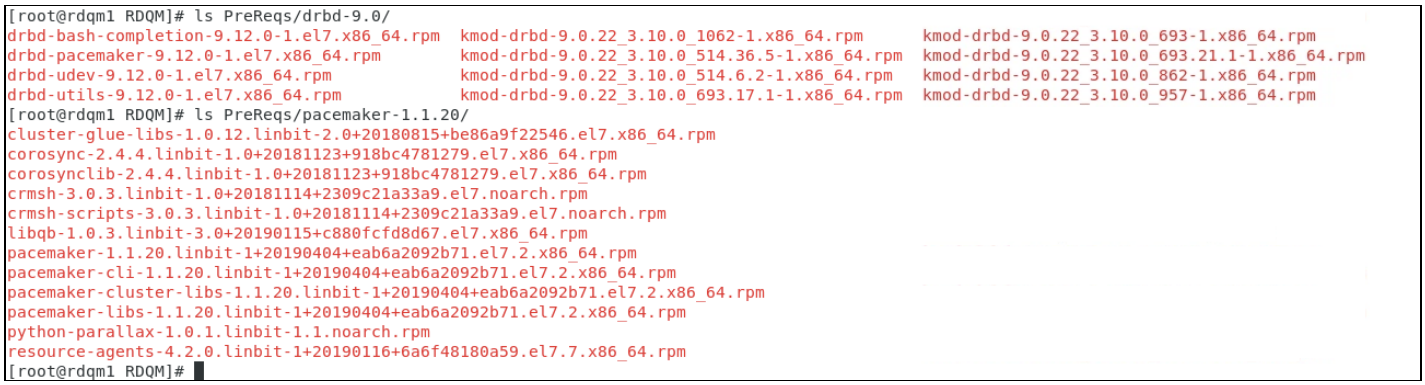
```
cd /home/ibmuser/mq915/MQServer/Advanced/RDQM
```



4. List the members of the directory. You will the see the RPM file for installing RDQM along with the scripts to install and uninstall the RDQM support. Also included is a subdirectory for the RDQM prerequisites.
5. List the members of directory *PreReqs* to see the important prerequisites of RDQM.

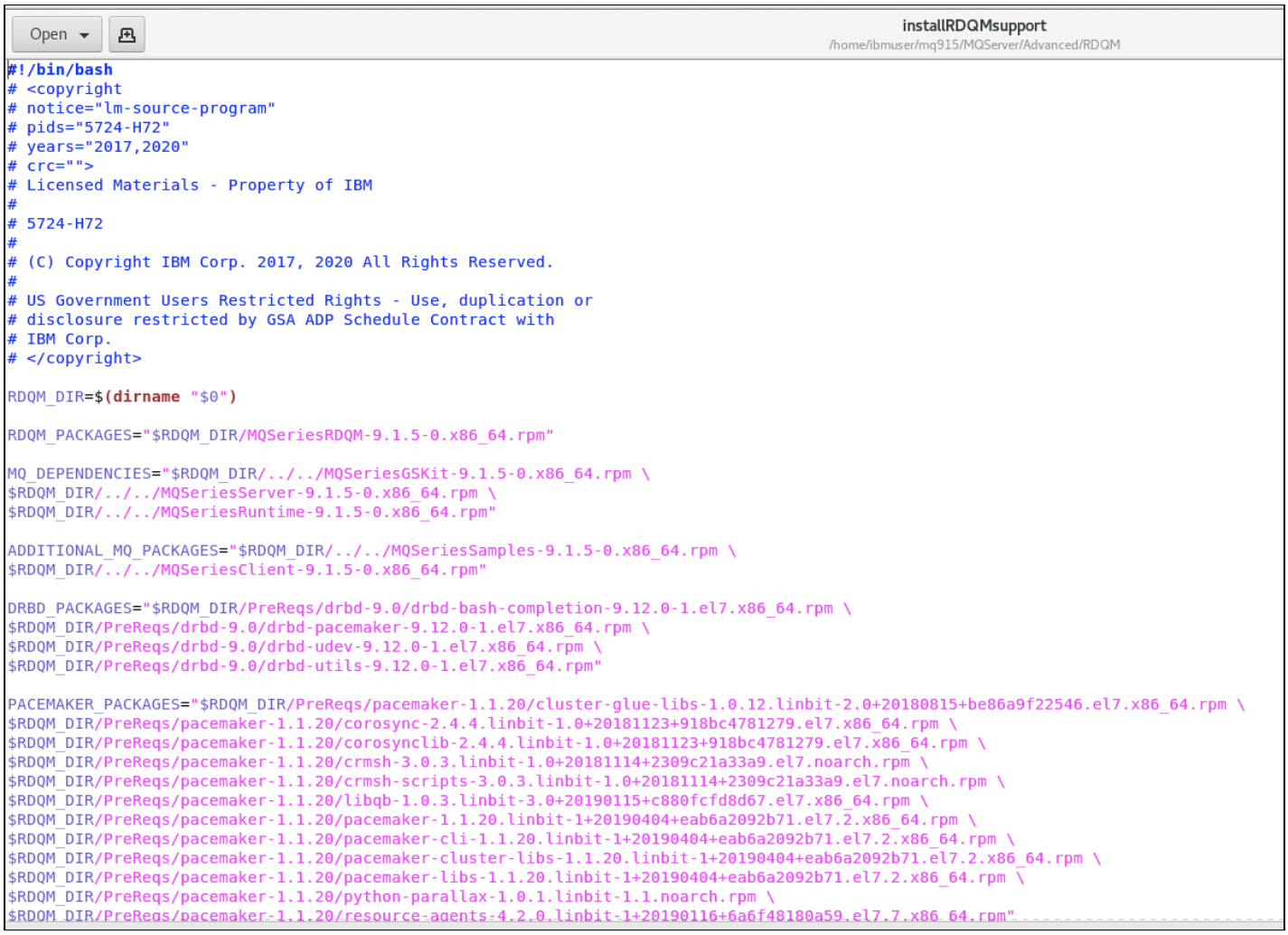
```
ls PreReqs
```

Here you see Pacemaker and DRBD. If you drill into those subdirectories, you will see the RPM packages for installing these prerequisites.



6. Enter the command to edit the *installRDQMsupport* shell script.

```
gedit installRDQMsupport
```



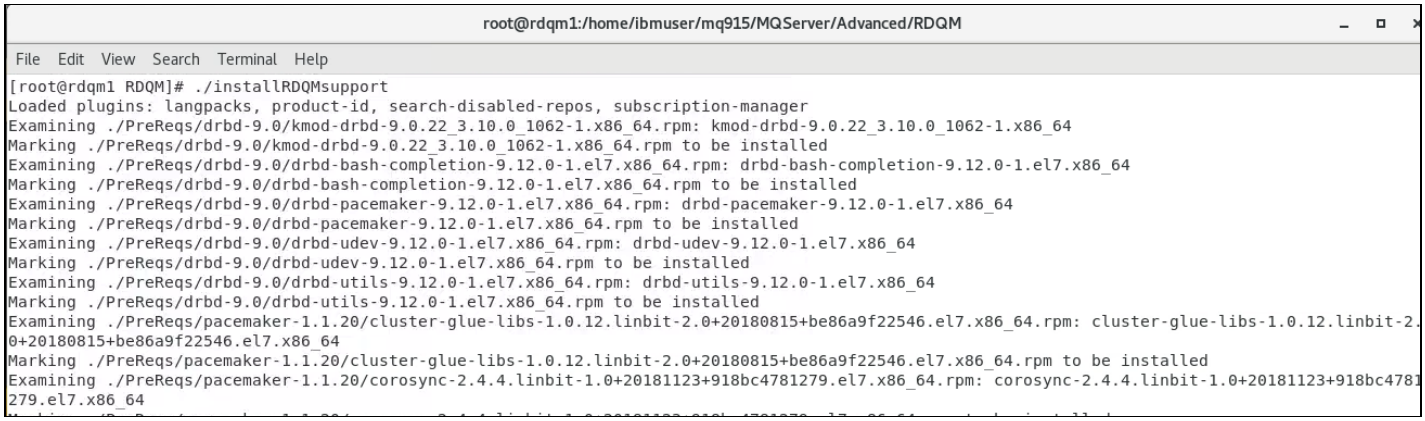
Review the script noting:

- current directory RDQM
- defining packages for RPMs
    - DRBD-KMOD, DRBD, PACEMAKER
    - MQ_DEPENDENCIES, RDQM_PACKAGES, ADDITIONAL_MQ_PACKAGES
- yum install for packages

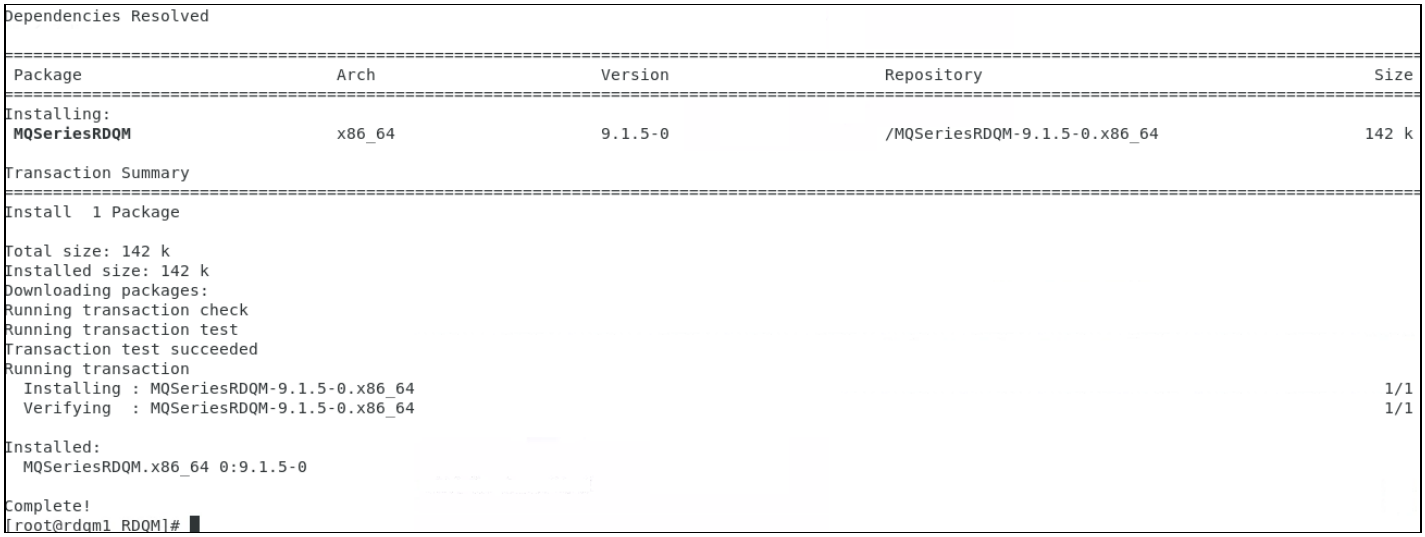
7. No changes are required, so close the editor by click the “X”.

8. Run the script to install RDQM with the following command.

```
./installRDQMsupport
```



Observe the script as it runs. It will take approximately three minutes. You will be notified of the results when complete.



9. RDQM is now ready as it has been installed with the prereqs on all of the VMs.

Configure the firewall

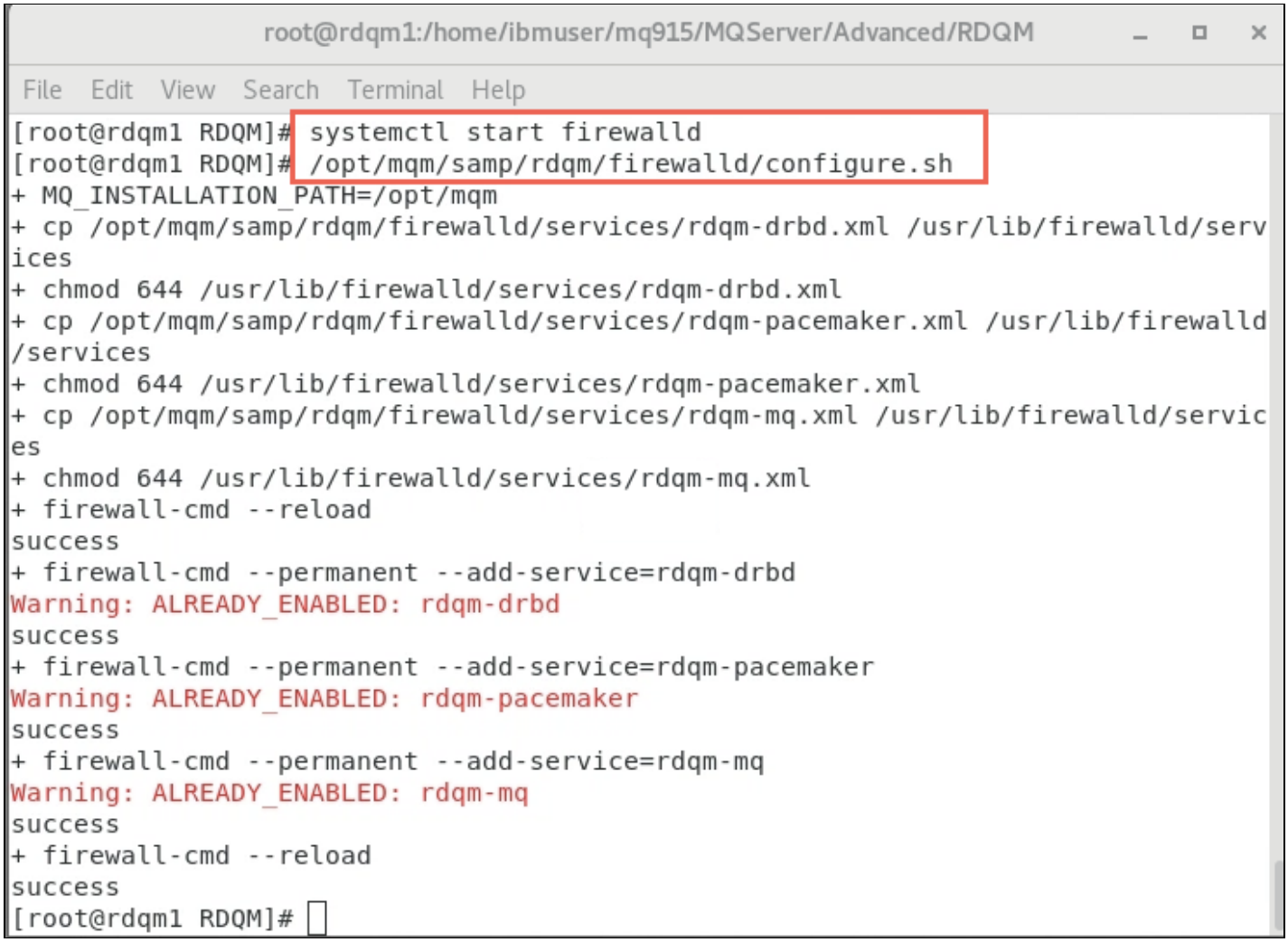
Normally, the firewall would have been configured as a pre-req. However during preparation of this environment, the default RHEL firewall was not configured. You need to do that now for the RDQM cluster.

1. While still logged in as root on **rdqm1** start the firewall with following command:

```
systemctl start firewalld
```

2. Run the following command to allow MQ, DRDB, and Pacemaker ports opened in the firewall:

```
/opt/mqm/samp/rdqm/firewalld/configure.sh
```



3. Exit from the root user.

```
exit
```

Be sure to exit out of root before continuing with the next section.

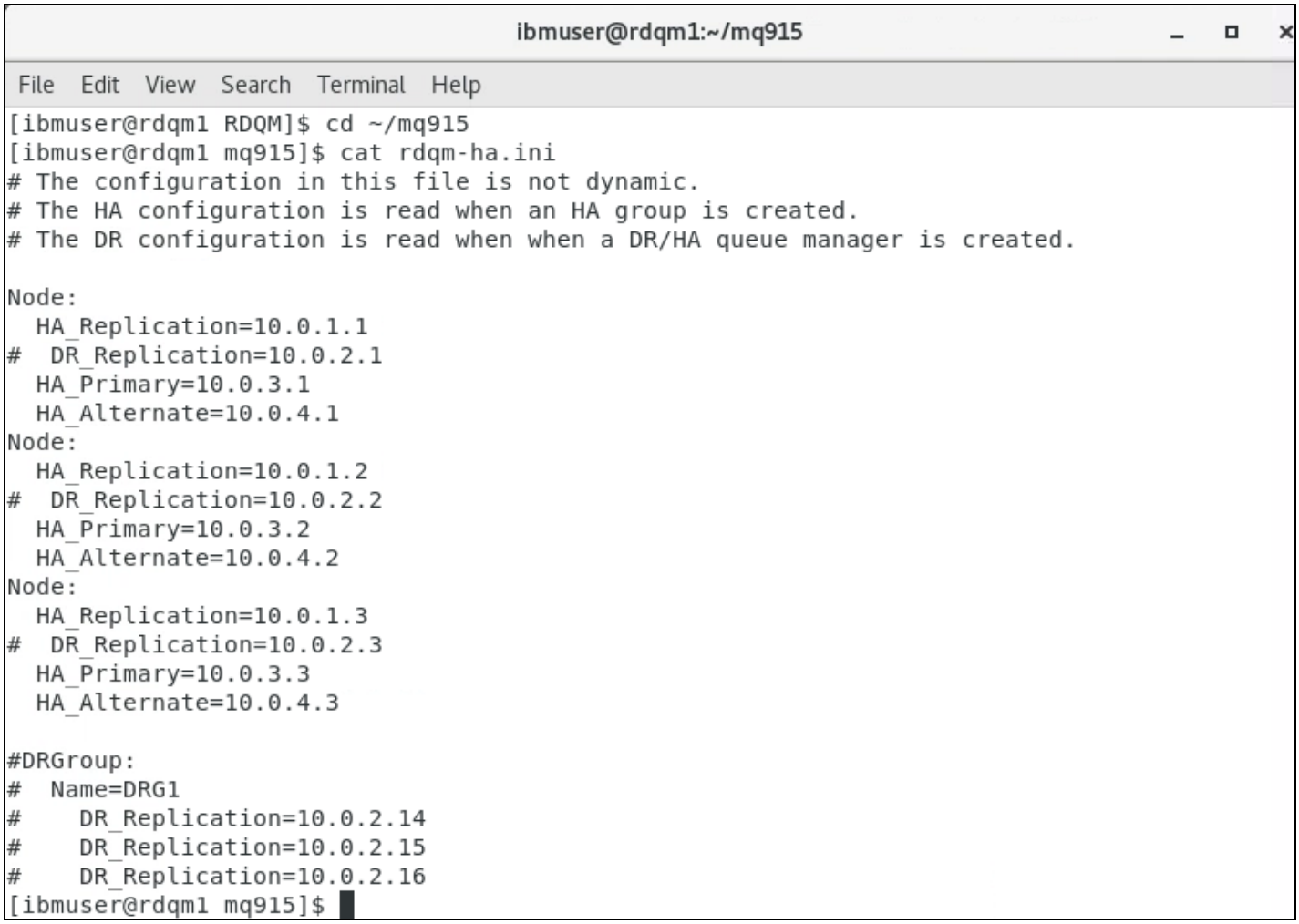
Configure the cluster

The cluster must first be created, and then an RDQM instance defined containing one or more queue managers. The RDQM code expects the rdqm.ini file to be in the /var/mqm directory.

The cluster is defined using the rdqm.ini file. The /home/ibmuser/ directory contains the rdqm.ini file we will use.

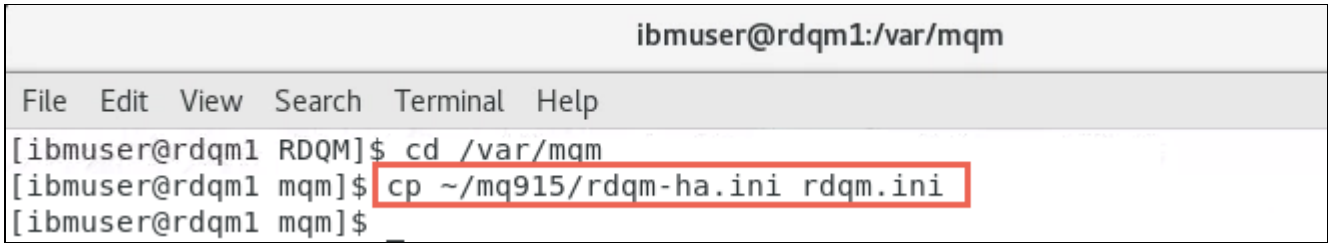
1. Return to rdqm1 and make sure you have exited your su session. Review this file with command:

```
cd ~/mq915
cat rdqm-ha.ini
```



2. On node **rdqm1**, in a terminal window (as **ibmuser**), navigate to */var/mqm*. Copy the provided **rdqm-ha.ini** file to the */var/mqm* directory with the following command:

```
cd /var/mqm
cp ~/mq915/rdqm-ha.ini rdqm.ini
```

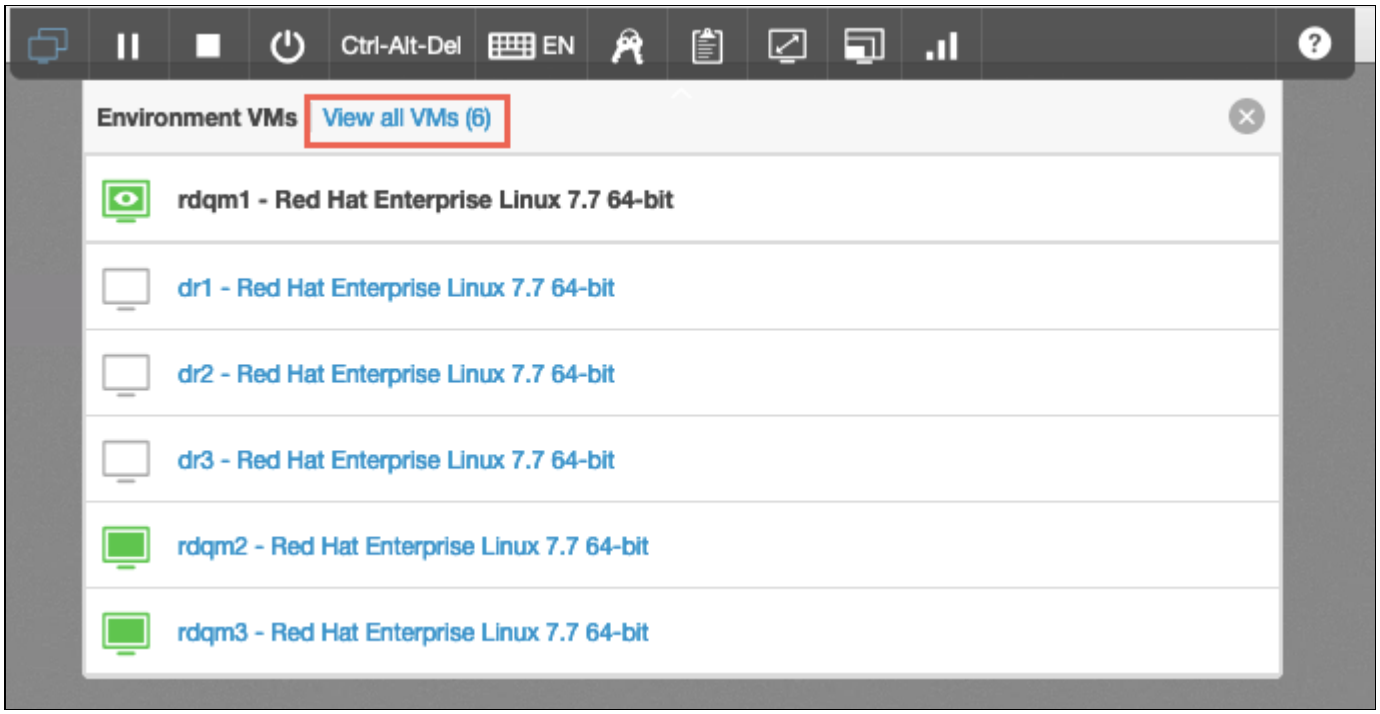


3. **IMPORTANT:** Repeat these commands on **rdqm2** and **rdqm3** before continuing.

   Hint: You can click the monitor icon for rdqm2 and rdqm3 which will launch the desktop for each in a new browser tab.

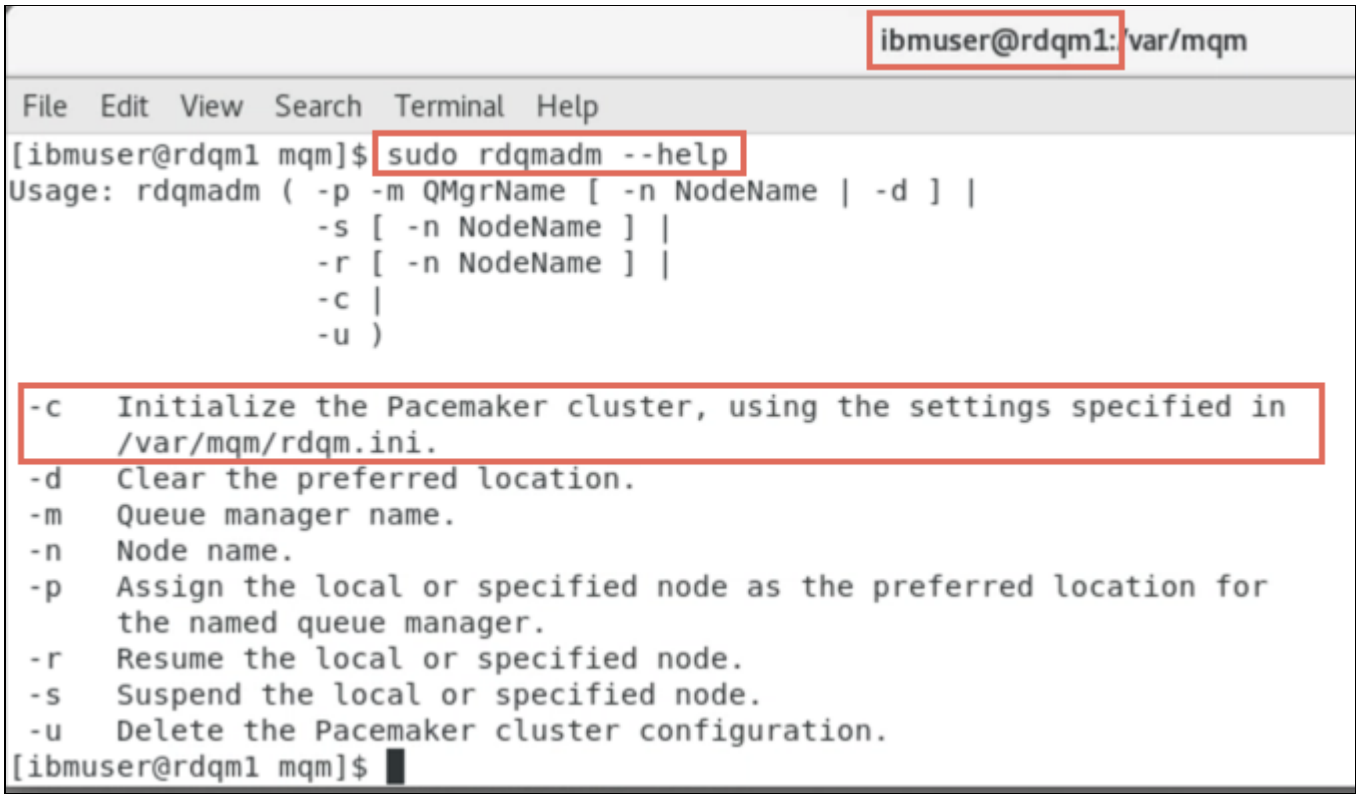
   Click the arrow in the black bar at top of screen to open the Skytap menu. Click the monitors icon on left end.

   Click *View all VMs (6)* to show the VMs. Then you can click the monitor icons. **Hint**



4. Return to the primary node **rdqm1**. Enter the following command to see the options for the *rdqmadm* command:

```
sudo rdqmadm --help
```



The “-c” parameter is used to configure the Pacemaker cluster

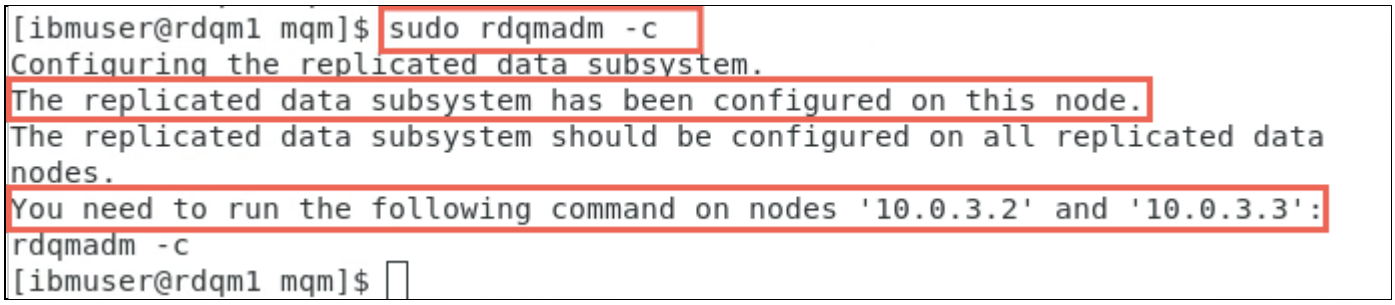
5. Enter the command to configure RDQM Pacemaker HA cluster:

```
sudo rdqmadm -c
```

❗ Warning:
If you are copying the command snippets from this lab guide and pasting them in the terminal windows: Beware that if the command does not work it may be that the copy assumes a long or double hyphen instead of a single hyphen. Just try overtyping the hyphen with the regular hyphen.

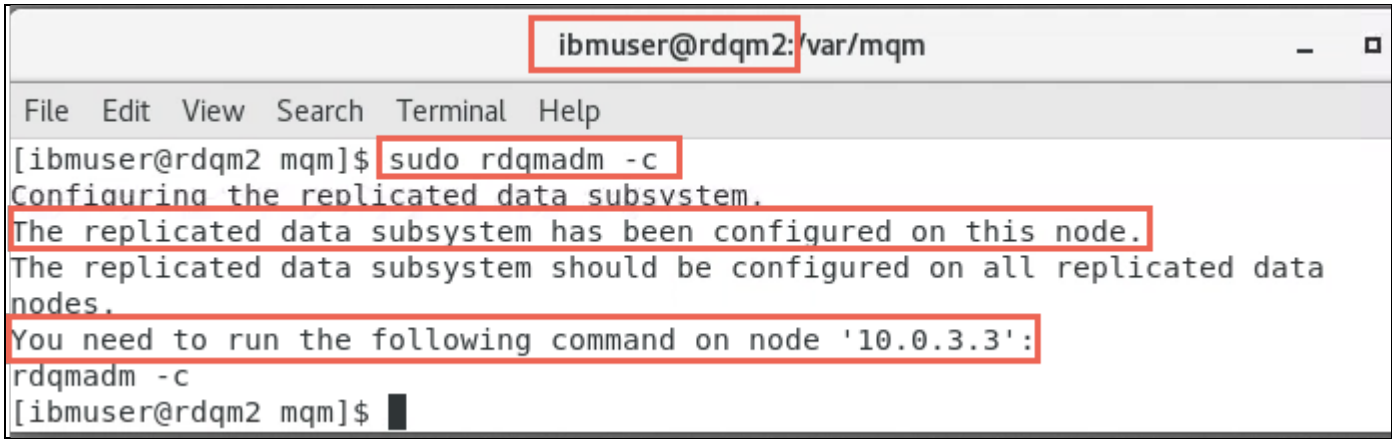
6. You have received the message that the replicated data system has been completed on this node

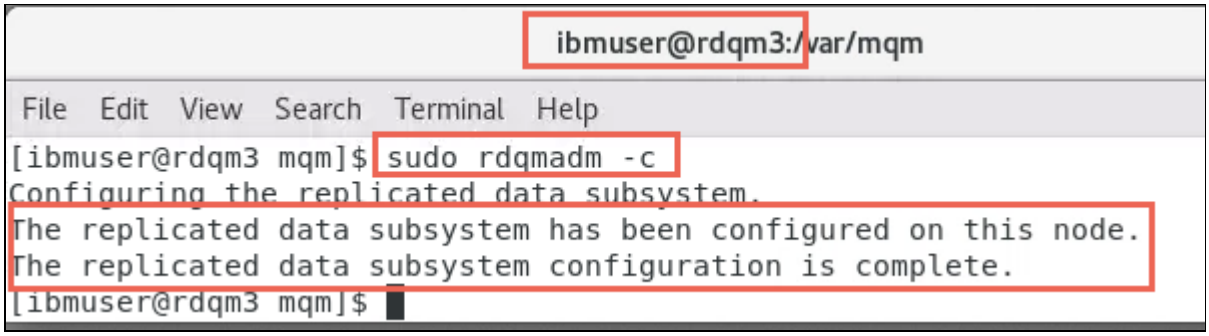
(10.0.3.1 - rdqm1). You are also told that you need to run the same command on **10.0.3.2** (rdqm2) and **10.0.3.3** (rdqm3).



7. Do as instructed and run the command on rdqm2 and rdqm3. Make sure to use root access (sudo). After running the command on **rdqm3**, you will see that the RDQM subsystem configuration has been completed.

```
sudo rdqmadm -c
```





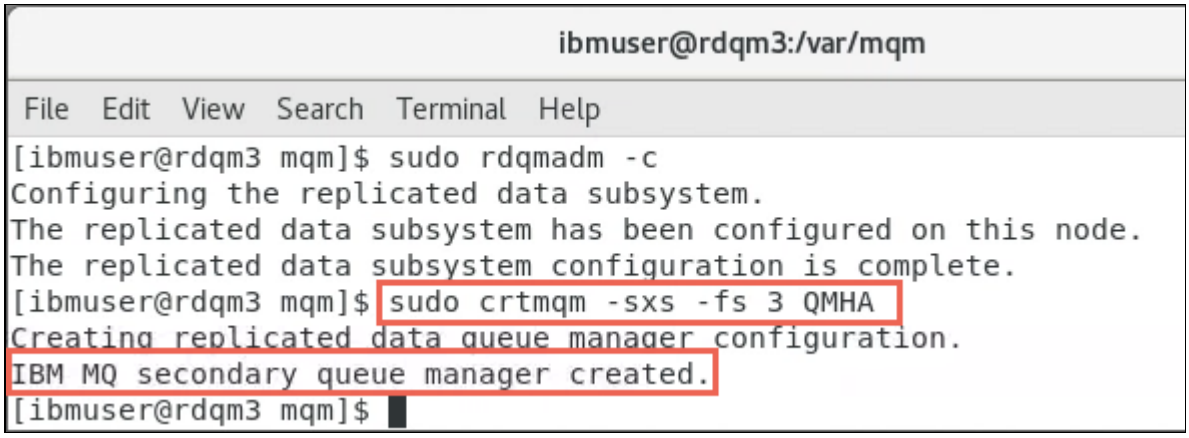
Configure the HA RDQM

The high availability replicated data queue manager (RDQM) now needs to be created. The *crtmqm* command includes a **-sx** parameter for a replicated data queue manager.

The secondary RDQMs need to be created first on two of the nodes with the **-sxs** parameter. This parameter indicates that the primary node's data will be replicated to this node and required disk space gets allocated.

1. Run the following command on **rdqm2** and **rdqm3**

```
sudo crtmqm -sxs -fs 3 QMHA
```

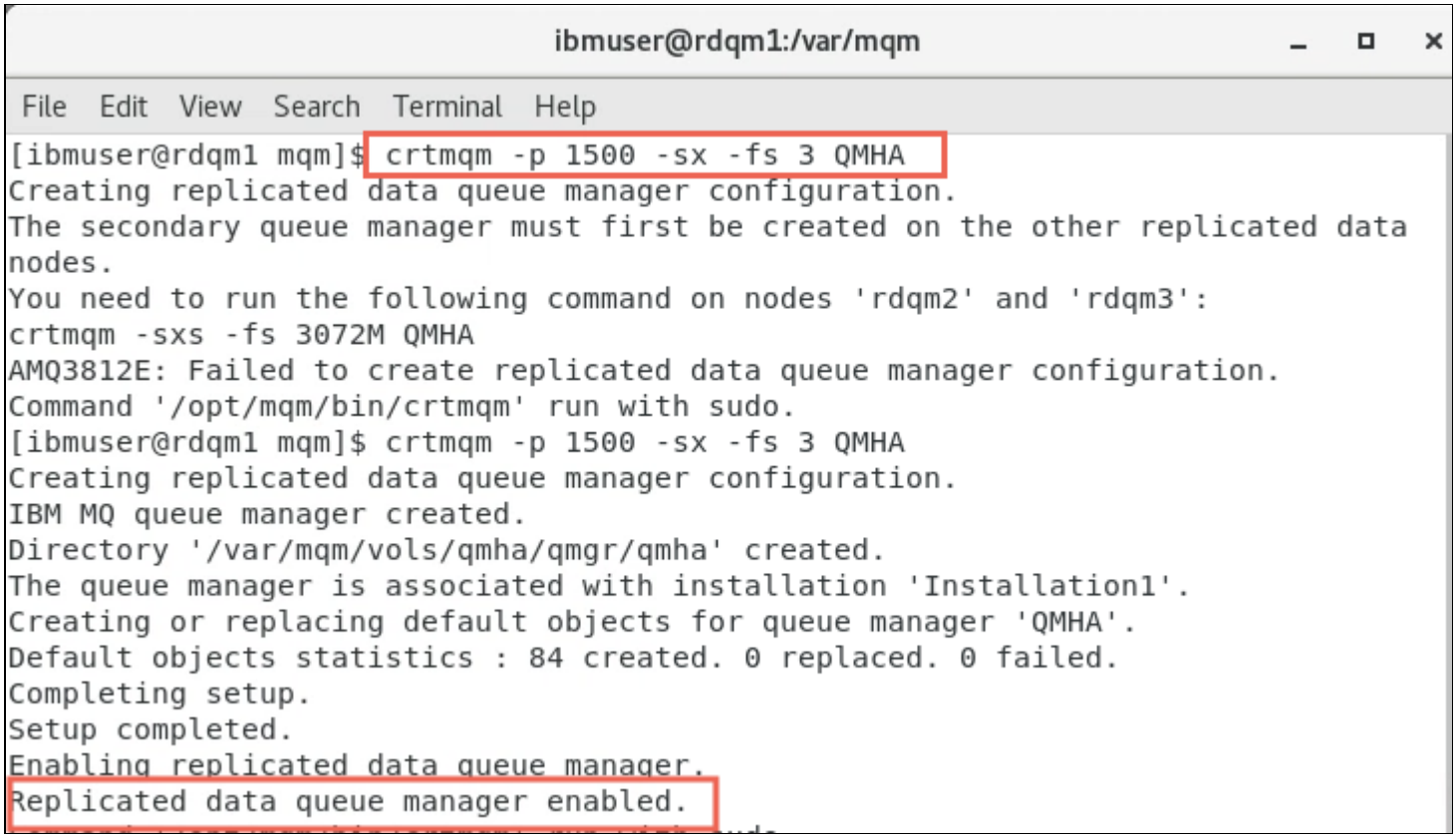


> **Note:**
> When running RDQM commands, you should run with root access. So precede the commands with sudo. You may notice that as a member of the mqm group ibmuser's commands complete even though not using root access. But when the command is complete you are reminded to run with sudo. You may notice this in the screen shots.

2. Now that the secondary queue managers have been created, the primary RDQM is created on the node where the command is run. This node becomes the RDQM's preferred location by default.

   You will now create the primary RDQM on rdqm1. Return to **rdqm1**. Create the primary RDQM, which will listen on port 1500 with the following command:

```
sudo crtmqm -p 1500 -sx -fs 3 QMHA
```

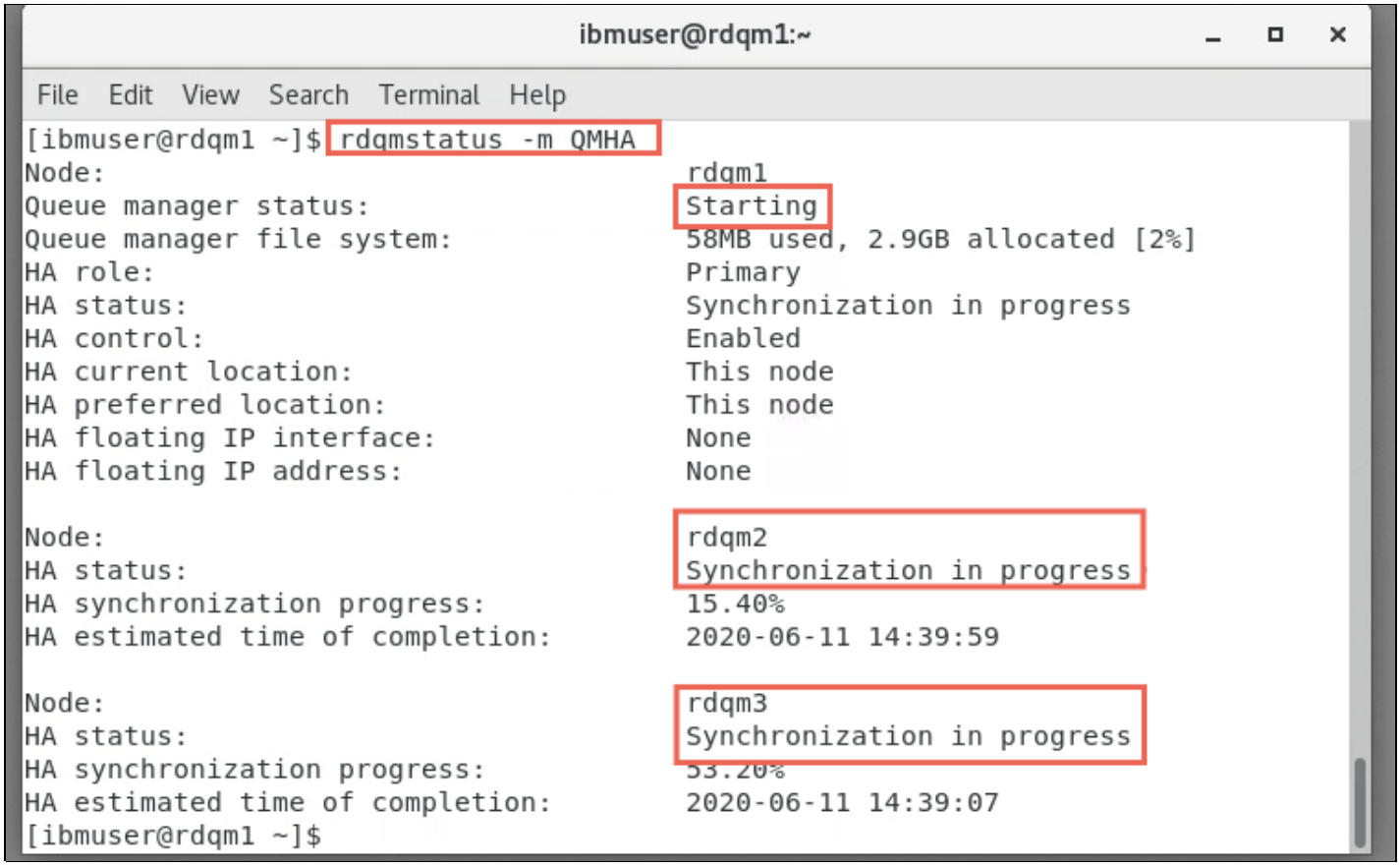


The primary queue manager is now created and running. ℹ **Note:**
In order to create RDQMs and issue commands, *ibmuser* must be in the mqm and haclient groups. *haclient* group is created as part of the RDQM installation. This was done as setup for this lab.

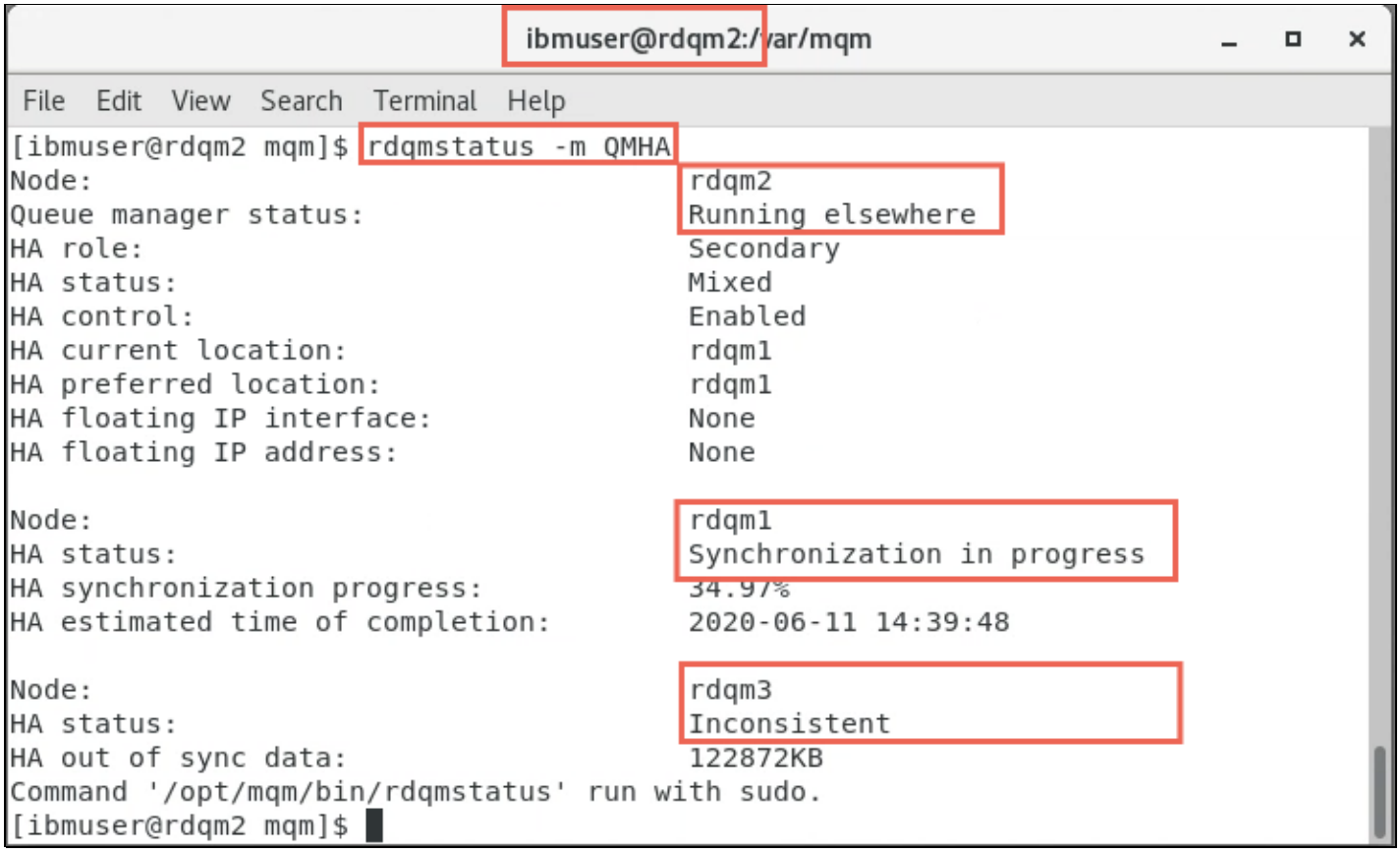
1. Check the status of the queue manager:

```
sudo rdqmstatus -m QMHA
```

Initially, the output should look similar to the following. Synchronization is in progress.

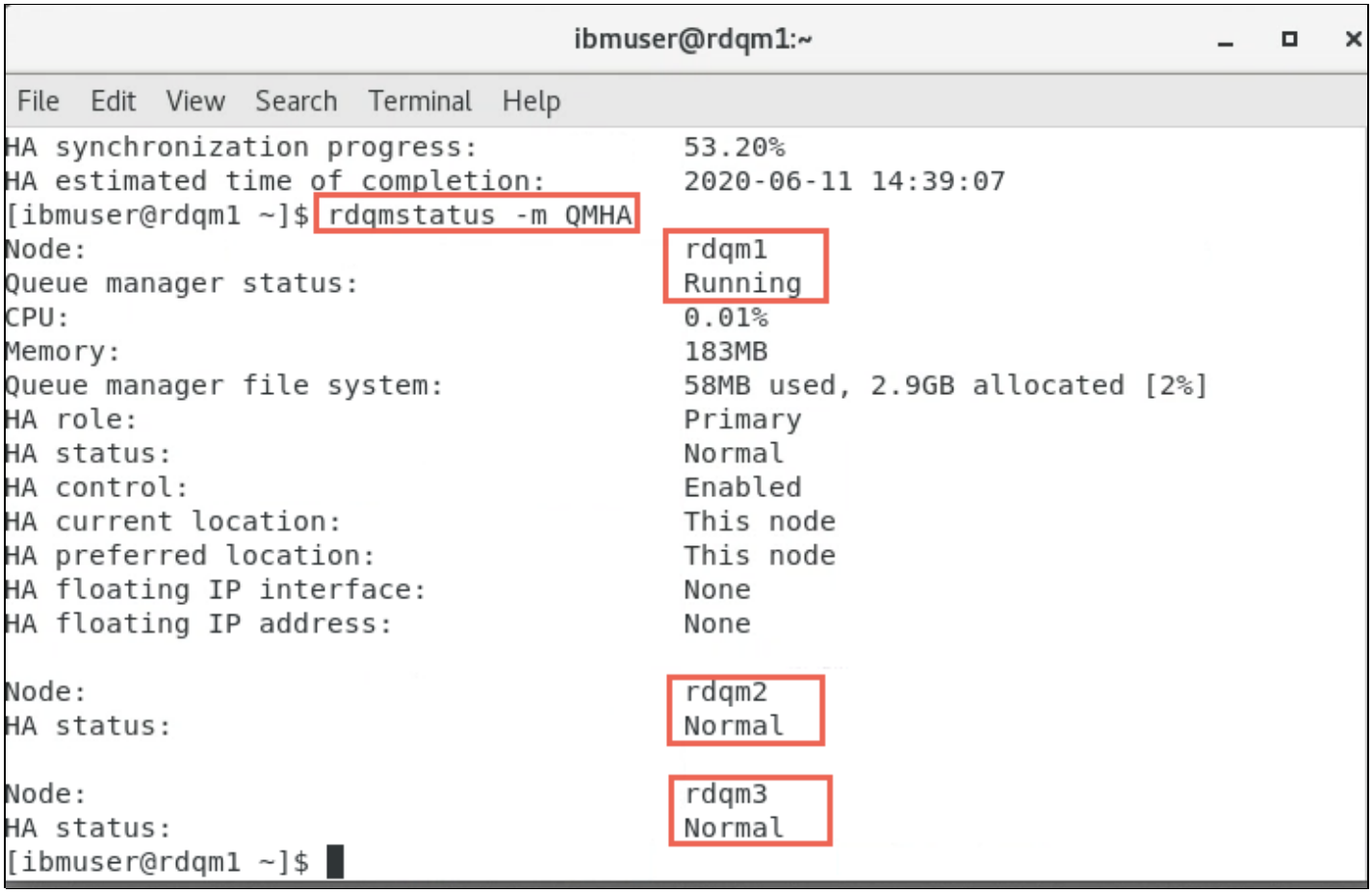


You can run the same command on the other nodes and get similar output.



2. When the nodes have completed synchronising, the HA Status field on all nodes should change to ‘Normal’. This may take a few minutes.

   Repeat the **sudo rdqmstatus** command again until you see the normal HA status (as shown on **rdqm1**).



Note:
RHEL default time before the screen locks is very short. If you need longer, you can turn off the screen lock in settings. Applications > System Tools > Settings > Power > Power Saving > Blank screen > Never.

Simple testing of RDQM

Once all nodes have an HA status of Normal, you can commence testing. You will perform some tests, which will show different use cases.

Failing over an RDQM instance to another node

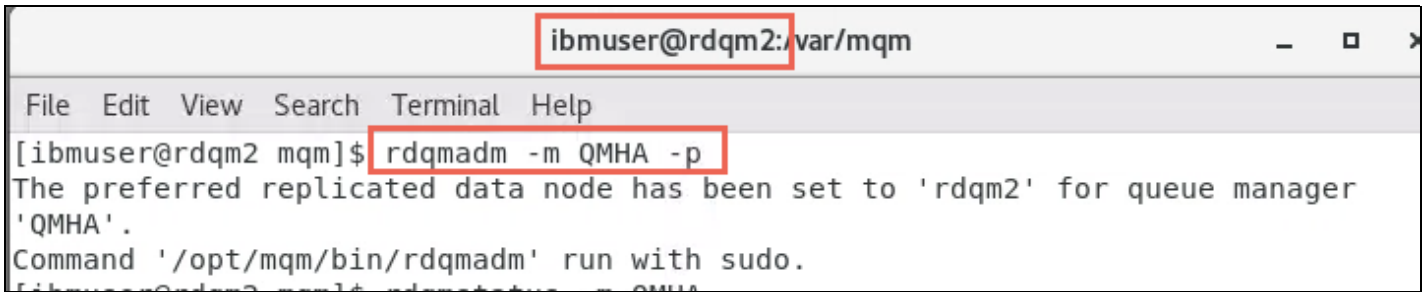
The easiest way to force an RDQM instance to fail over to another node is to change its preferred location.

The default location for RDQM is rdqm1. You will fail the RDQM instance to node 2, rdqm2.

1. Switch to **rdqm2**.
2. Open a new terminal window, and as the user ibmuser, issue the command to set the MQ environment.

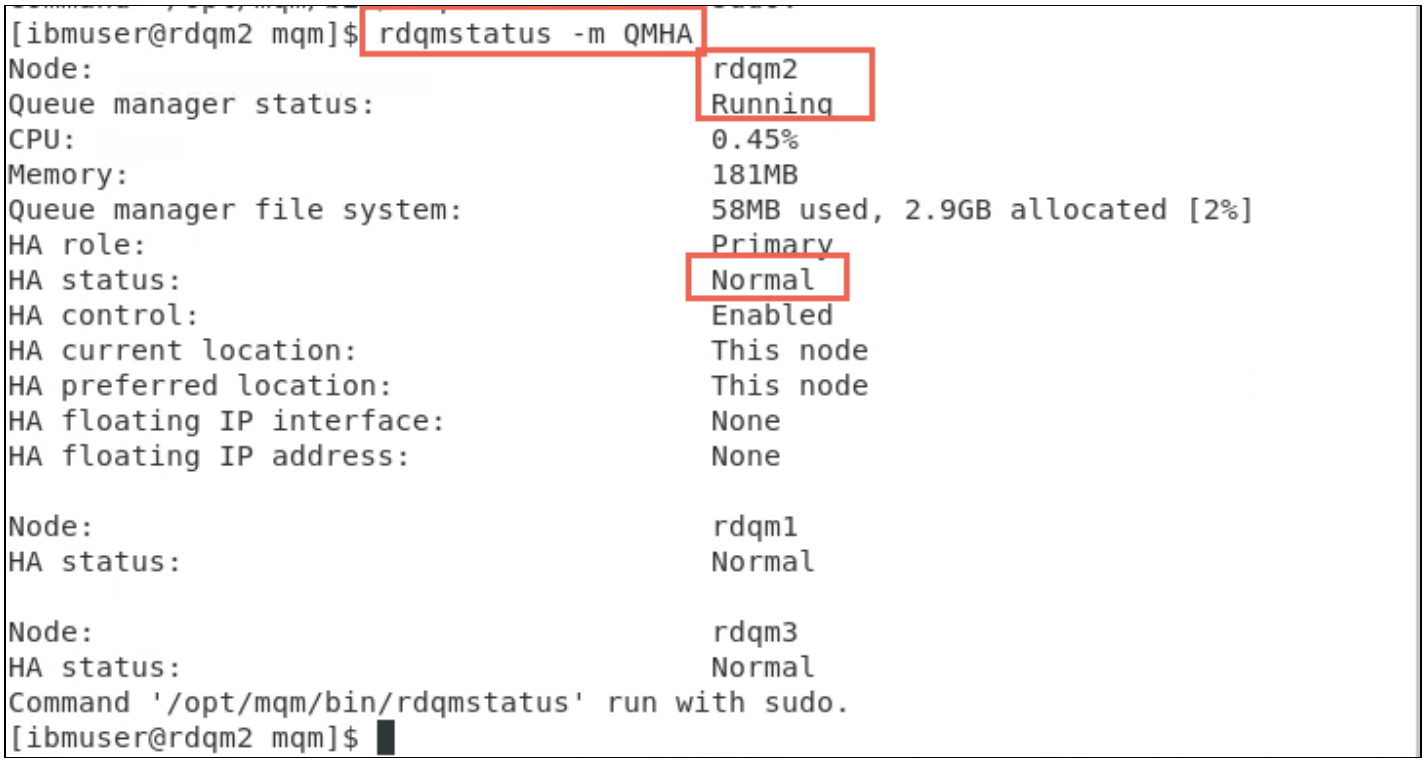
Make this node the primary instance with the following command:

```
sudo rdqmadm -m QMHA -p
```



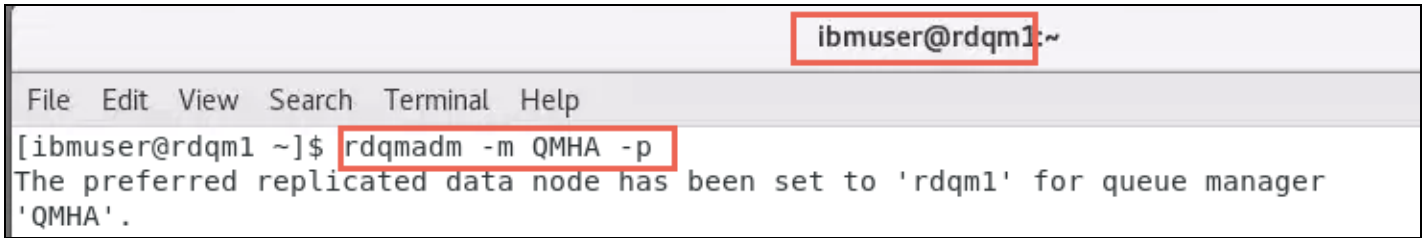
3. Confirm that **rdqm2** is now the primary node:

```
sudo rdqmstatus -m QMHA
```



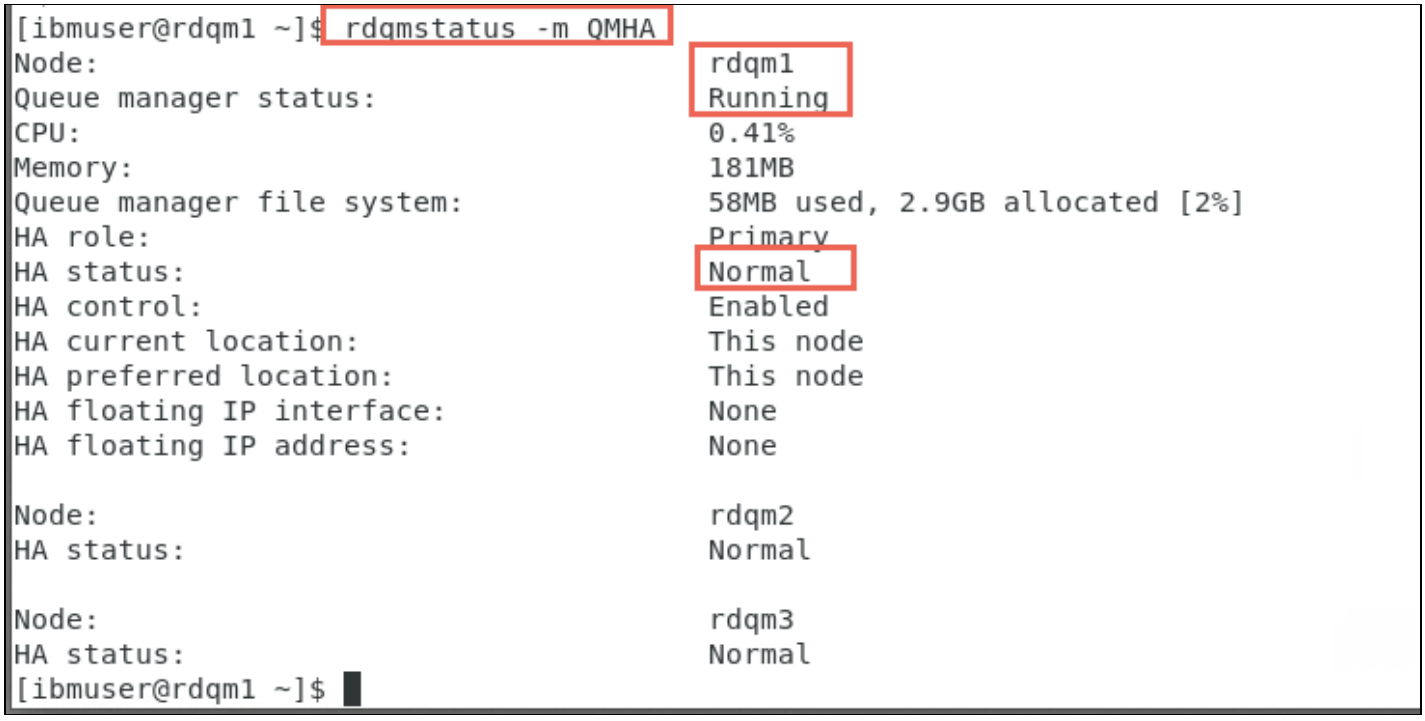
4. Now you can move the queue manager back to rdqm1. Return to **rdqm1** and run the **sudo rdqmadm** command again.

```
sudo rdqmadm -m QMHA -p
```



Check the status again to see it is now running on **rdqm1**.

```
sudo sudo rdqmstatus -m QMHA
```



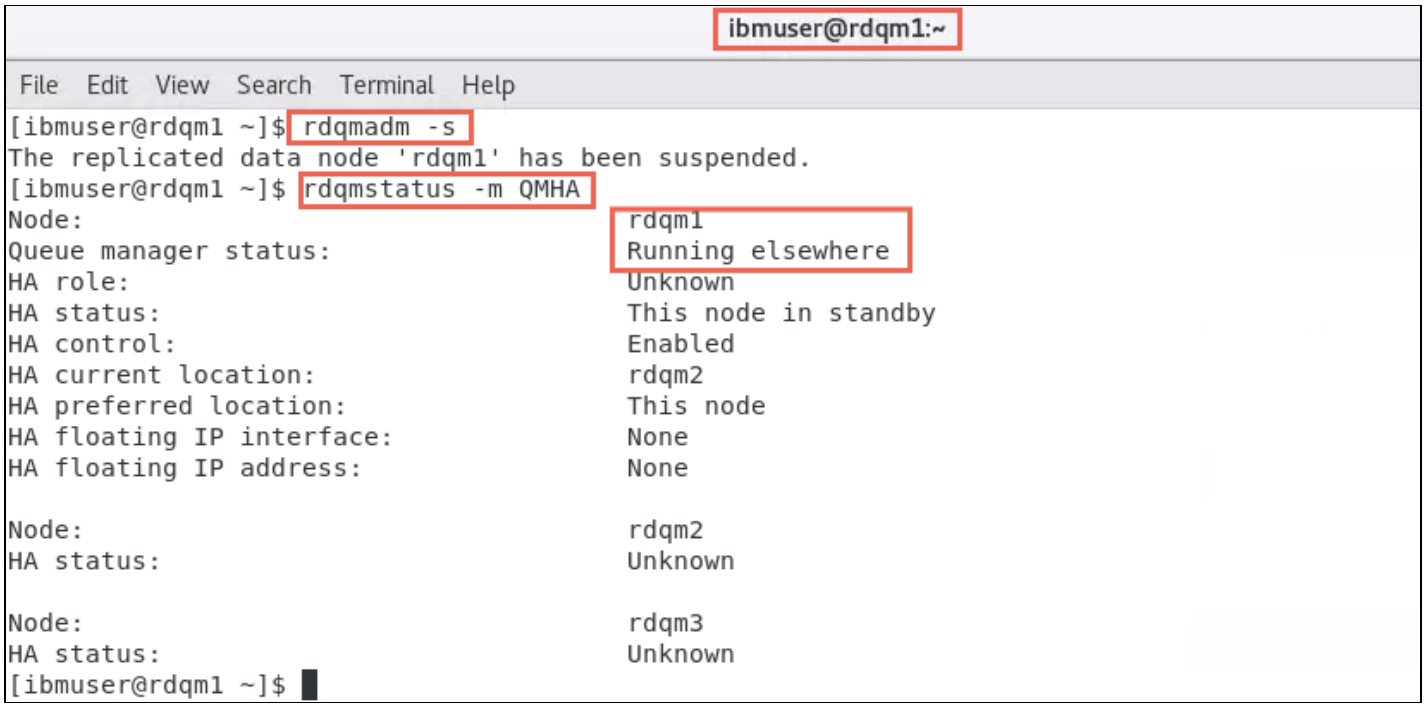
Move the RDQM by suspending a node

Another test is to move a RDQM by suspending the node on which it is running, as you may want to do when applying a Fix Pack.

1. On the node where QMHA is running (**rdqm1**), return to ibmuser's terminal window. As the user **ibmuser** (not as root), issue the command to suspend the queue manager:

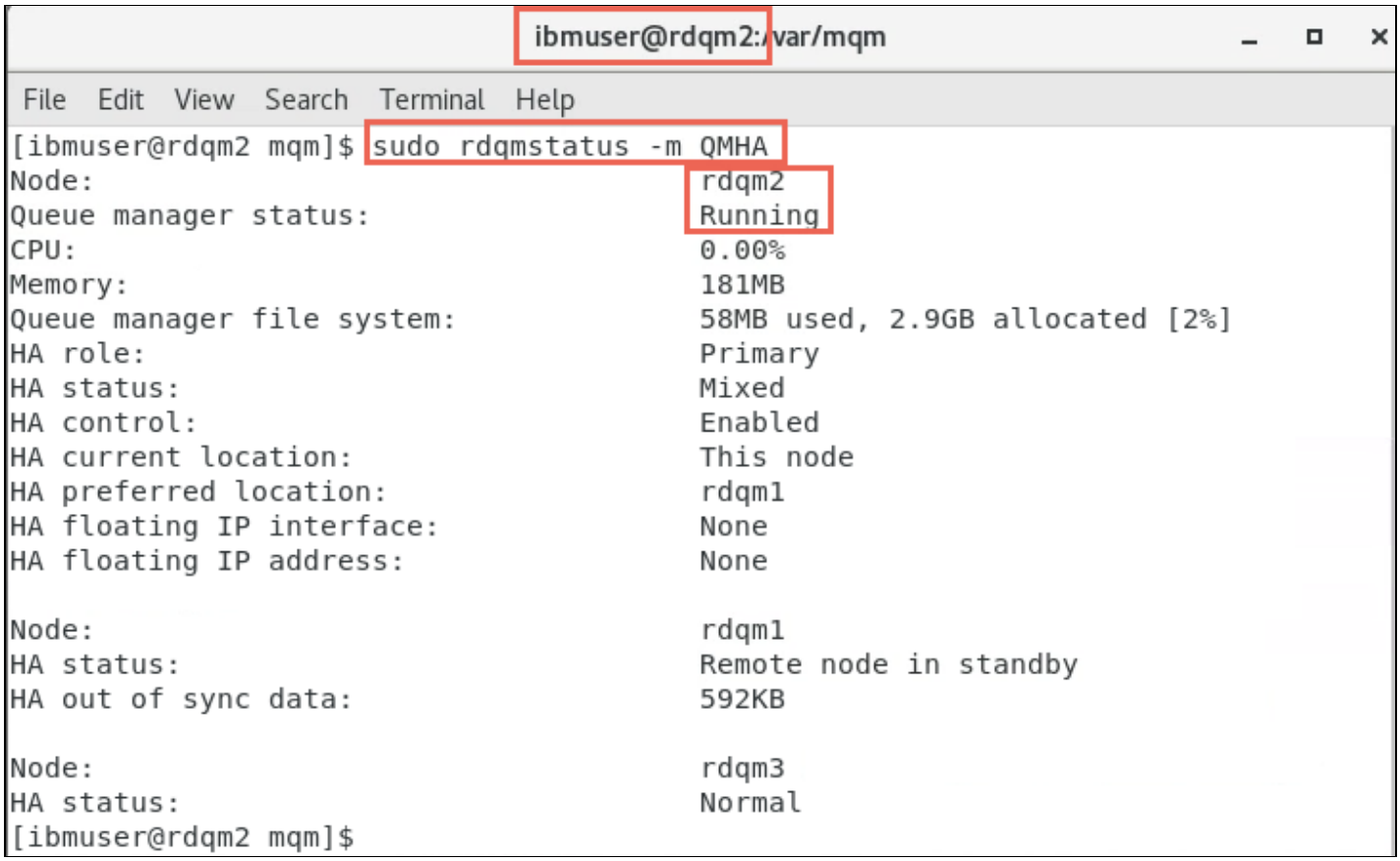
```
rdqmadm -s
```

2. As shown in the display, the replicated data node is suspended and goes into standby.



3. Switch to **rdqm2** and issue the command to display the status of QMHA and you see that it is now running on rdqm2.

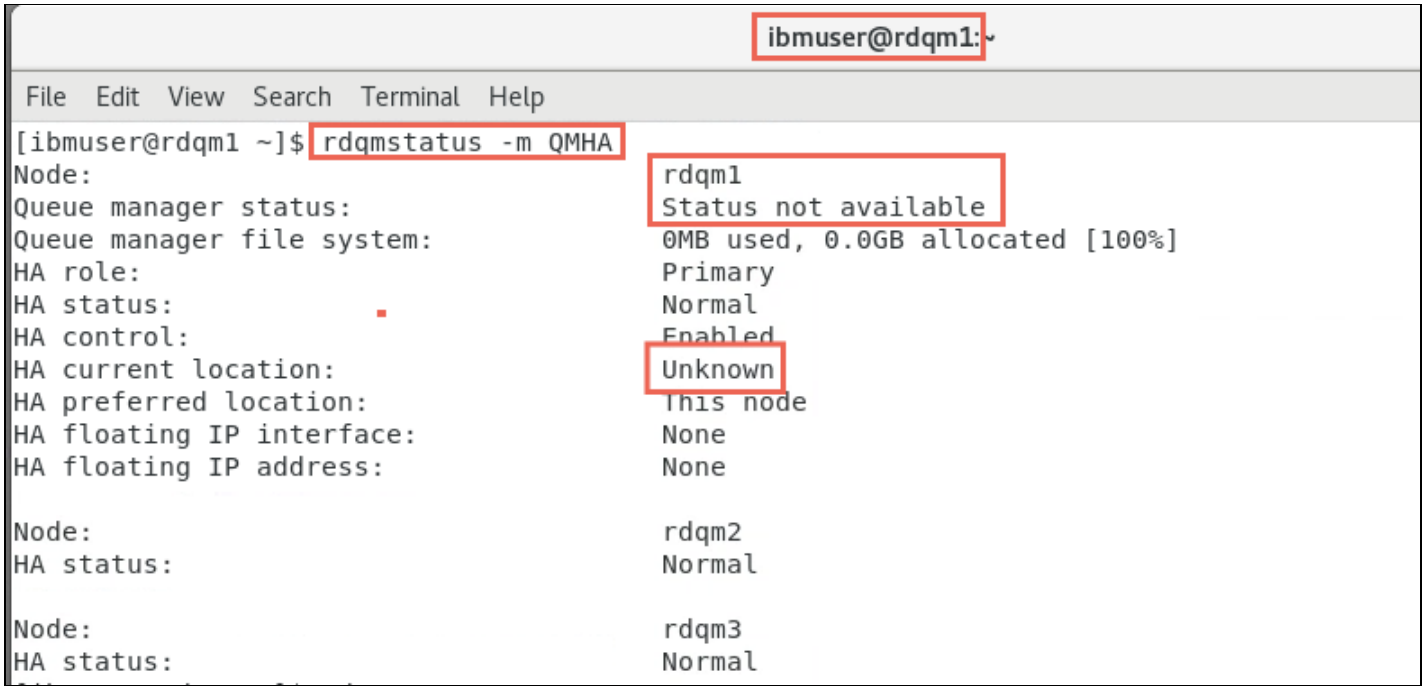
```
sudo rdqmstatus -m QMHA
```



4. Return to **rdqm1**, issue the command to resume the replicated data node in the cluster.

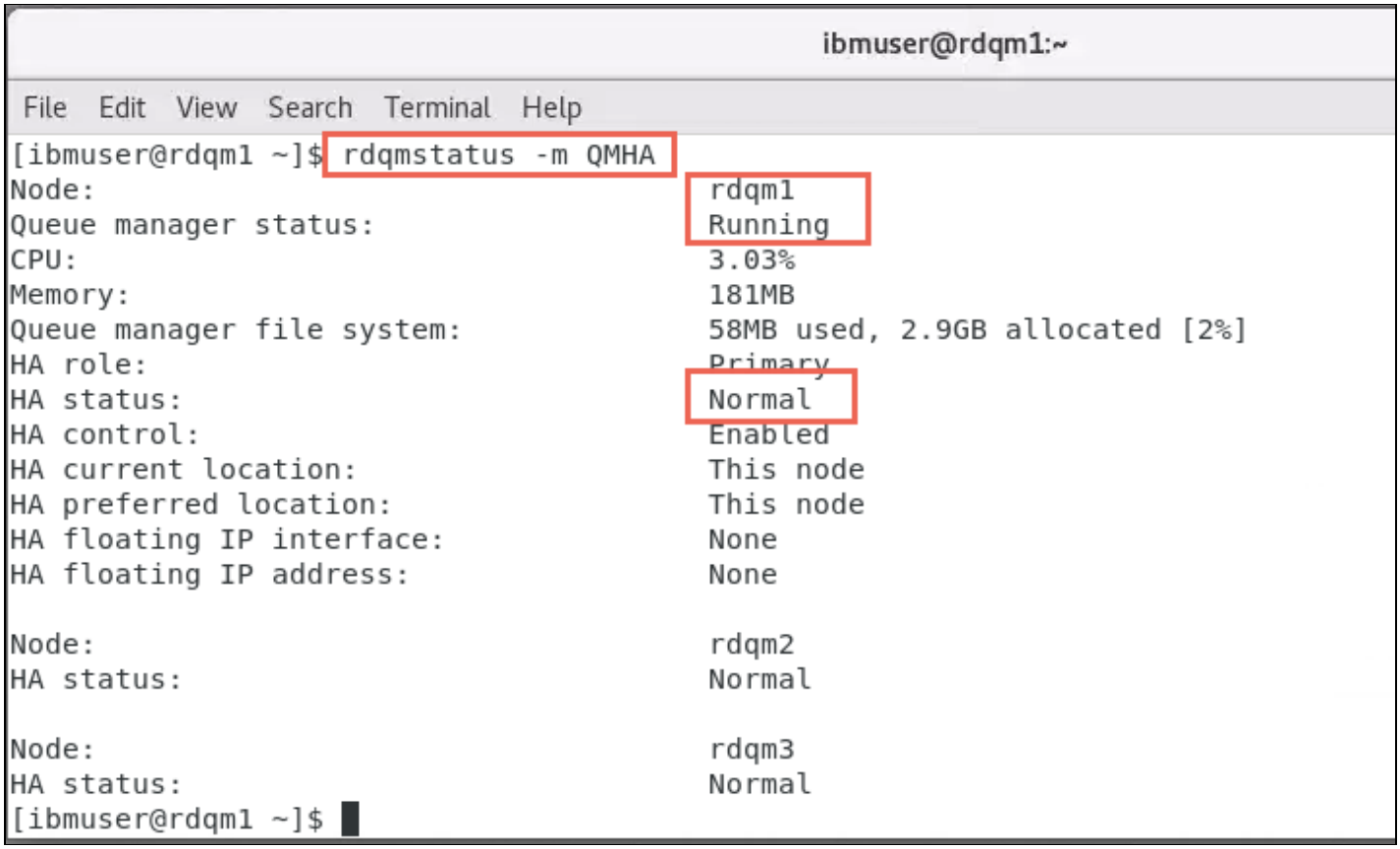
```
sudo rdqmadm -r
```

Quickly run the status command again. QMHA will initially run in a secondary role on this node. If you aren't quick enough, you may not catch this transitory state.



5. After the node has fully resumed, QMHA will run in a primary role on this node, as it was prior to being suspending. Issue the status command again to confirm that this has indeed happened.

```
sudo rdqmstatus -m QMHA
```



Testing RDQM using HA sample programs

Some High Availability sample programs are provided with MQ, which are a good visible demonstration for testing failovers. You will use these for testing:

- **amqsphac** - puts a sequence of messages to a queue with a two second delay between each message and displays events sent to its event handler. This will run on **rdqm3**.
- **amqsmhac** - copies messages from one queue to another with a default wait interval of 15 minutes after the last message that is received before the program finishes. This will run on **rdqm2**.
- **amqsghac** - gets messages from a queue and displays events sent to its event handler. This will run on **rdqm1**.

Create MQ resources

To run these samples, you will define two queues: for *SOURCE* and *TARGET*. You will also create a new channel using the 'MQ' IP address for each of the three nodes in our cluster (as the queue manager could run on any one of them) and the listener port for the queue manager. You will turn off CHLAUTH and CONNAUTH completely to keep things simple.

1. On **rdqm1** where the QMHA queue manager is running, in ibmuser's terminal window, run the command:

```
runmqsc QMHA
```

Run the following MQSC commands (remember that MQ objects are case sensitive):

```
ALTER QMGR CHLAUTH(DISABLED) CONNAUTH(' ')
```

```
REFRESH SECURITY TYPE(CONNAUTH)
```

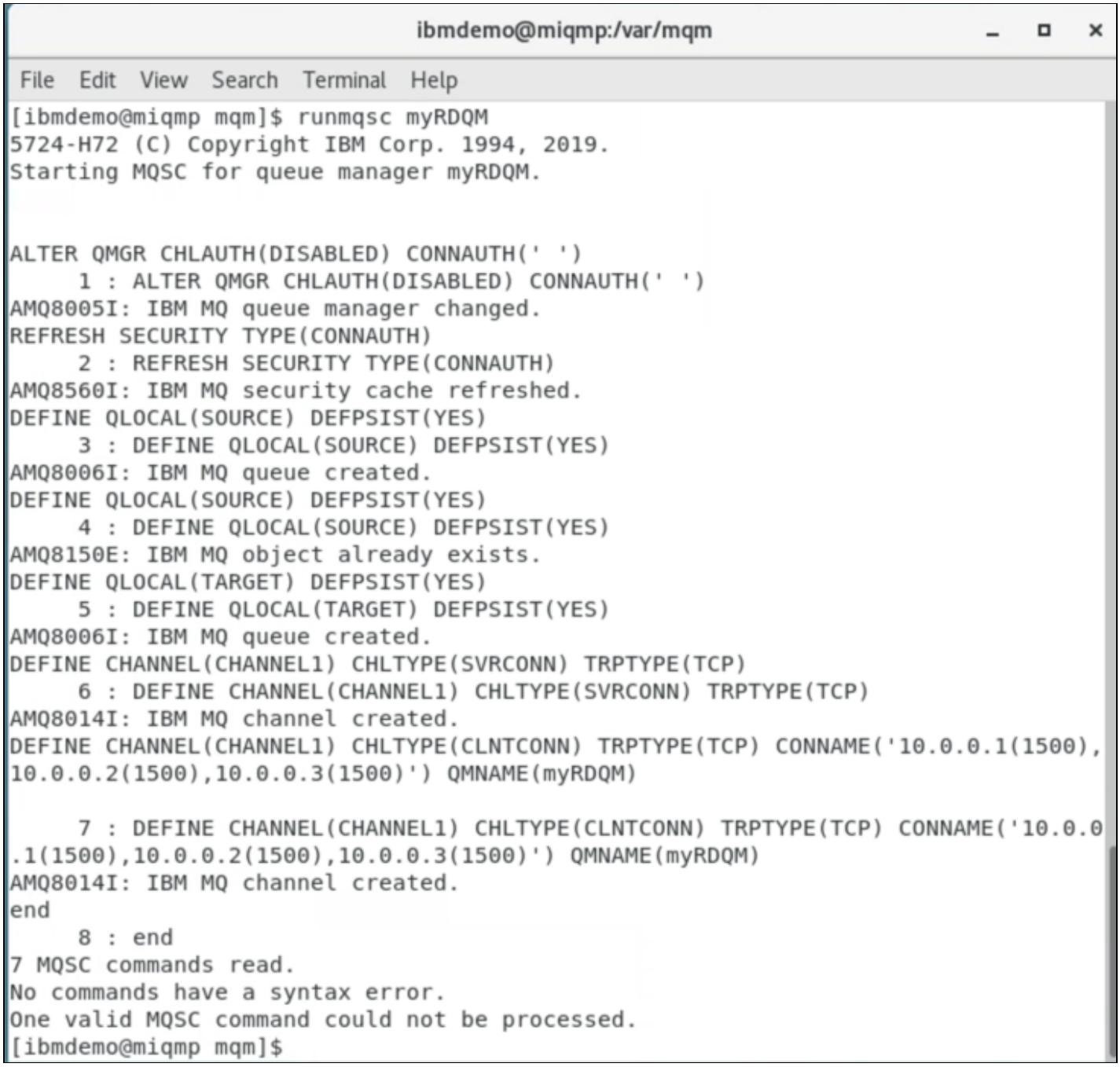
```
DEFINE QLOCAL(SOURCE) DEFPSIST(YES)
```

```
DEFINE QLOCAL(TARGET) DEFPSIST(YES)
```

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN) TRPTYPE(TCP)
```

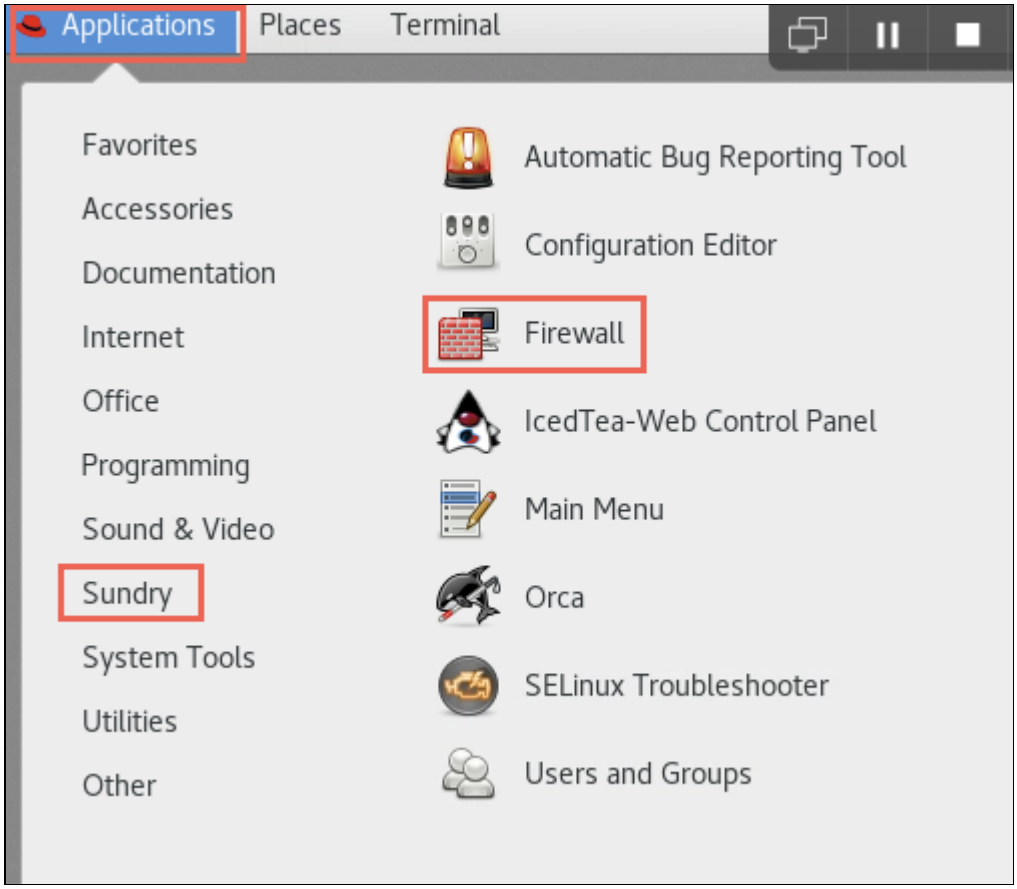
```
 DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) CONNAME('10.0.1.1(1500),1
0.0.1.2(1500),10.0.1.3(1500)') QMNAME(QMHA)
```

```
 END
```

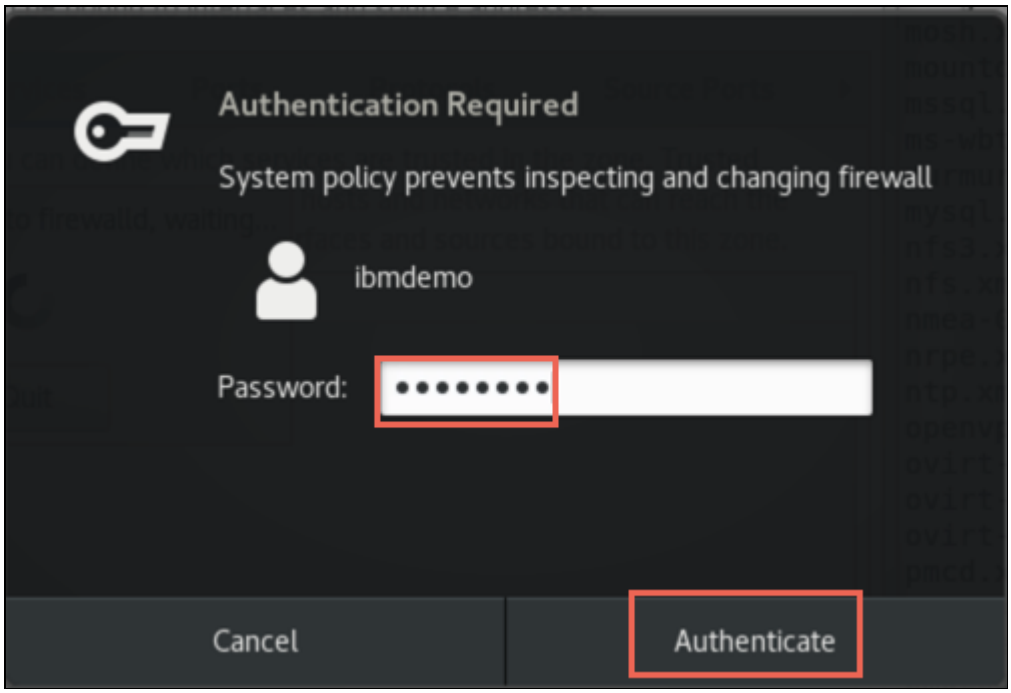


Update firewall rules

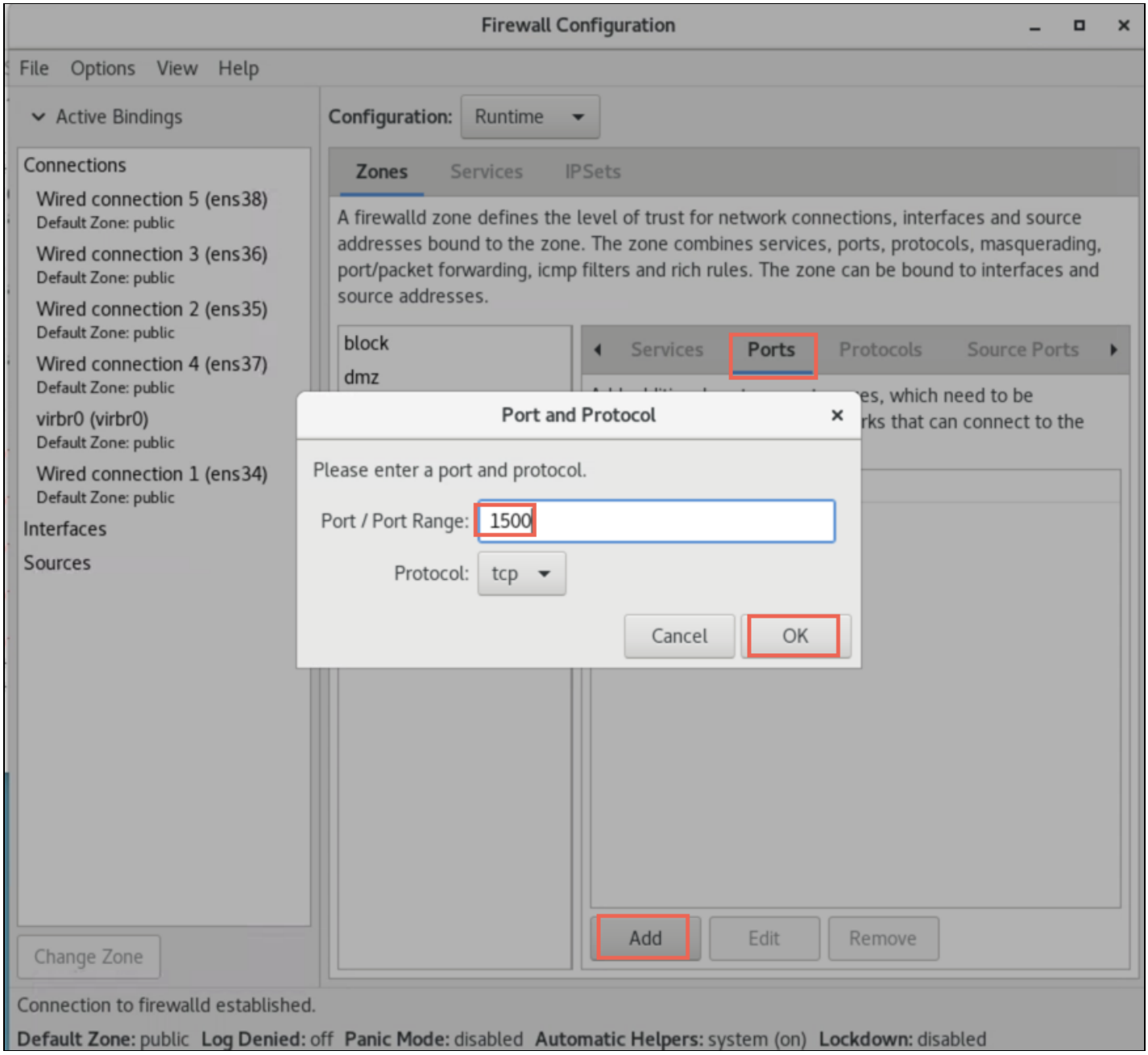
1. On each of the nodes, open the firewall port defined. Open the firewall from the top left of the screen, under *Applications -> Sundry -> Firewall*.



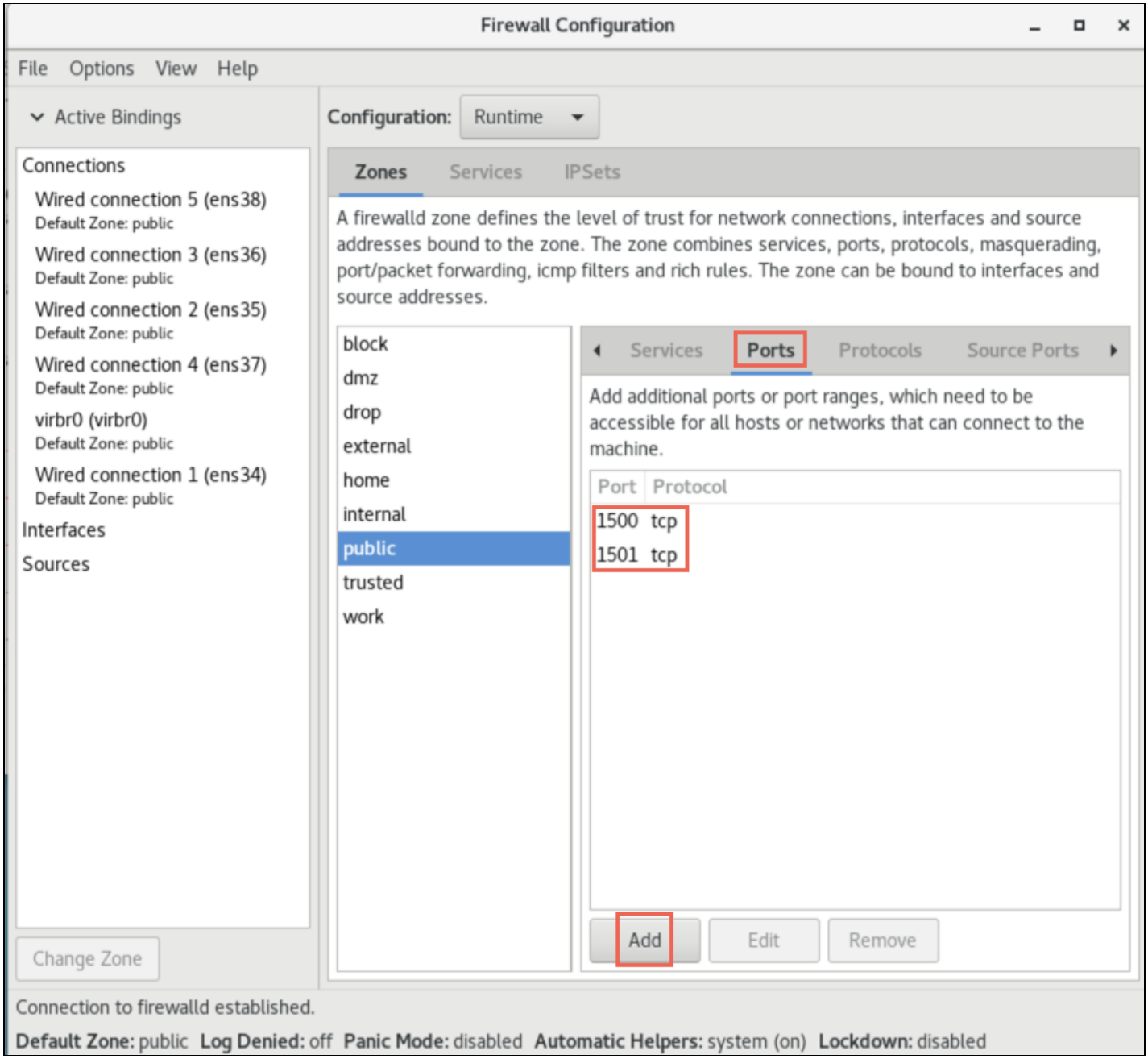
2. Enter the password for ibmuser, **engageibm**, then click *Authenticate*.



3. In the Ports pane, add TCP ports 1500 and 1501 (the latter will be used later).



Results should look like this:



Don’t forget, each node must have these ports opened in the firewall.

Start the HA sample programs

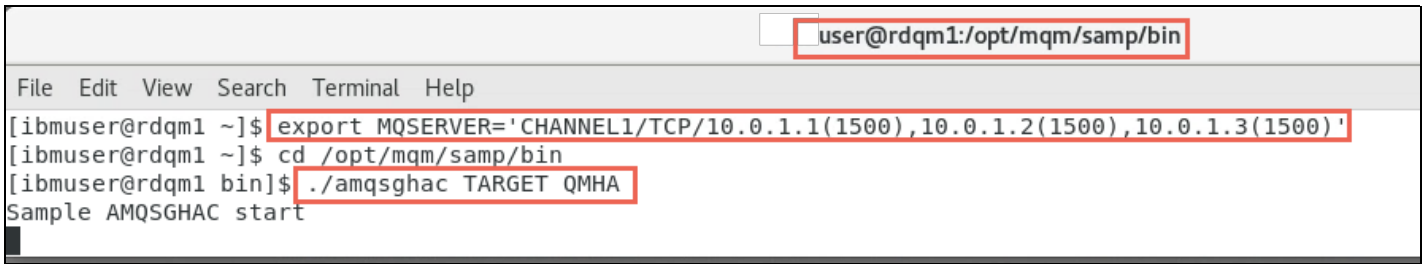
The easiest way to configure access to the queue manager from the sample programs is to use the MQSERVER environment variable. Again, as there are 3 possible nodes where our queue manager could run, each needs to be specified, along with the listener port for the queue manager.

1. On **rdqm1**, in the user ibmuser terminal window, enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.1(1500),10.0.1.2(1500),10.0.1.3(1500)'
```

2. Change to the **/opt/mqm/samp/bin** directory, and run the command: **amqsghac TARGET QMHA**

```
cd /opt/mqm/samp/bin
./amqsghac TARGET QMHA
```



Later, this will display the messages generated by amqsphac on rdqm3.

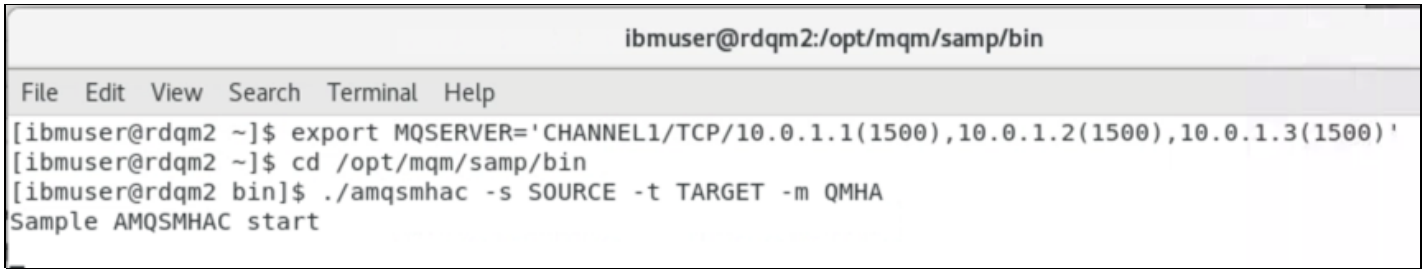
Leave this command to run!

3. Now switch to **rdqm2**. In the user ibmuser terminal window, enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.1(1500),10.0.1.2(1500),10.0.1.3(1500)'
```

4. Change directory to **/opt/mqm/samp/bin** and run the command: **amqsmhac -s SOURCE -t TARGET -m QMHA**

```
cd /opt/mqm/samp/bin
./amqsmhac -s SOURCE -t TARGET -m QMHA
```



Leave this command to run!

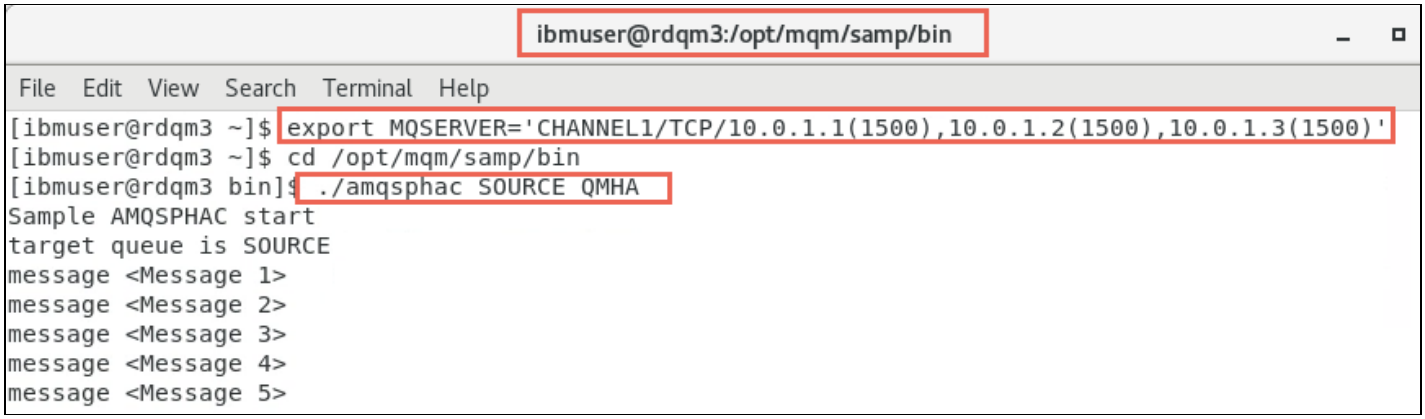
> **❗ Warning:**
> If you are copying the command snippets from this lab guide and pasting them in the terminal windows: Beware that if the command does not work it may be that the copy assumes a long or double hyphen instead of a single hyphen. Just try overtyping the hyphen with the regular hyphen.

5. Now switch to **rdqm3**. As before open a new terminal window. As the user ibmuser enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.1(1500),10.0.1.2(1500),10.0.1.3(1500)'
```

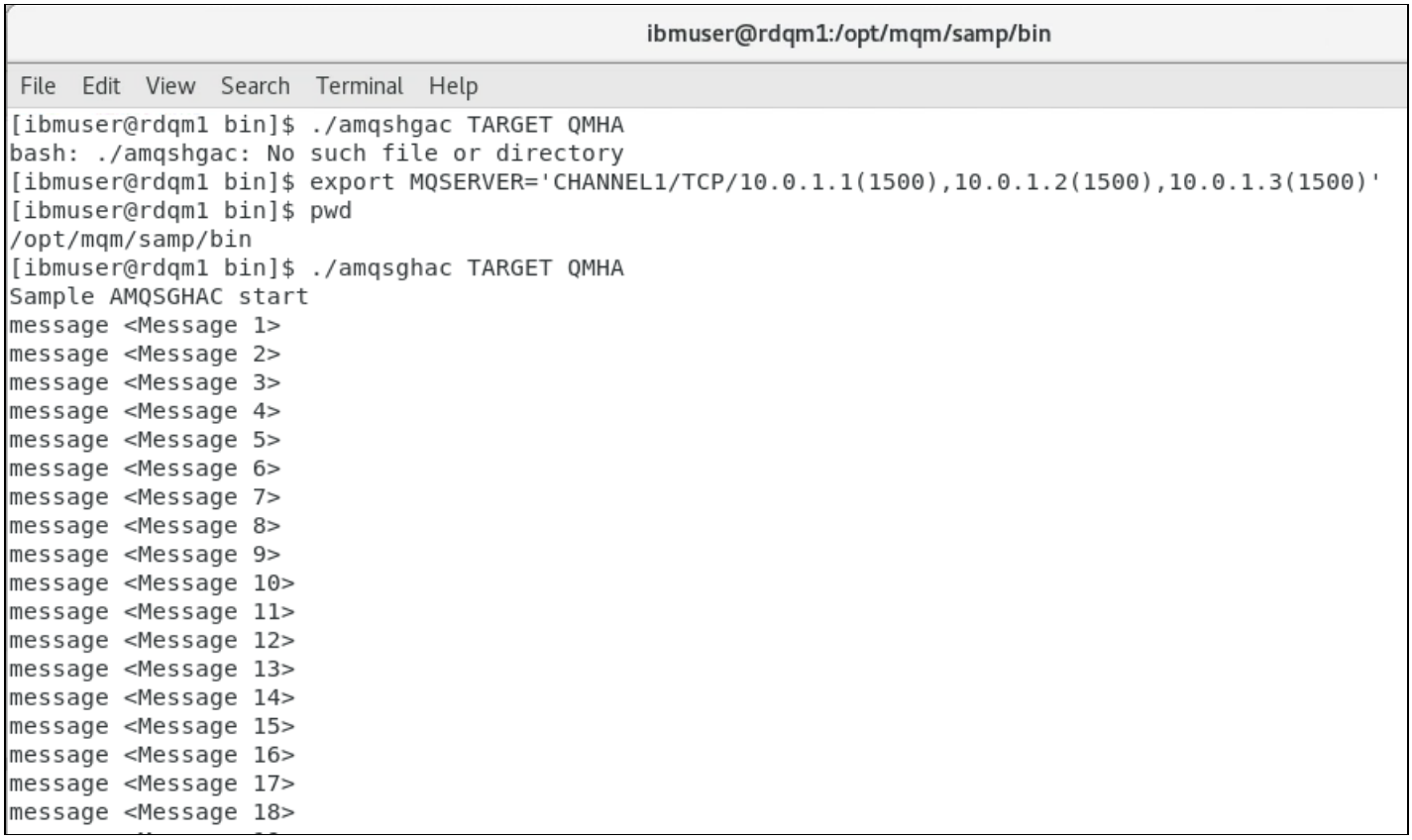
6. Change directory to **/opt/mqm/samp/bin** and run the command: **amqsphac SOURCE QMHA**

```
cd /opt/mqm/samp/bin
./amqsphac SOURCE QMHA
```



Leave this command to run!

7. Confirm that these messages are also being displayed on **rdqm1**.



Note: At this stage, the queue manager is running on the primary node (rdqm1) and each sample program is able to communicate with it, using the first location specified in the MQSERVER environment variable:

CHANNEL1/TCP/**10.0.1.1(1500)**,10.0.1.2(1500),10.0.1.3(1500)

Move the RDQM

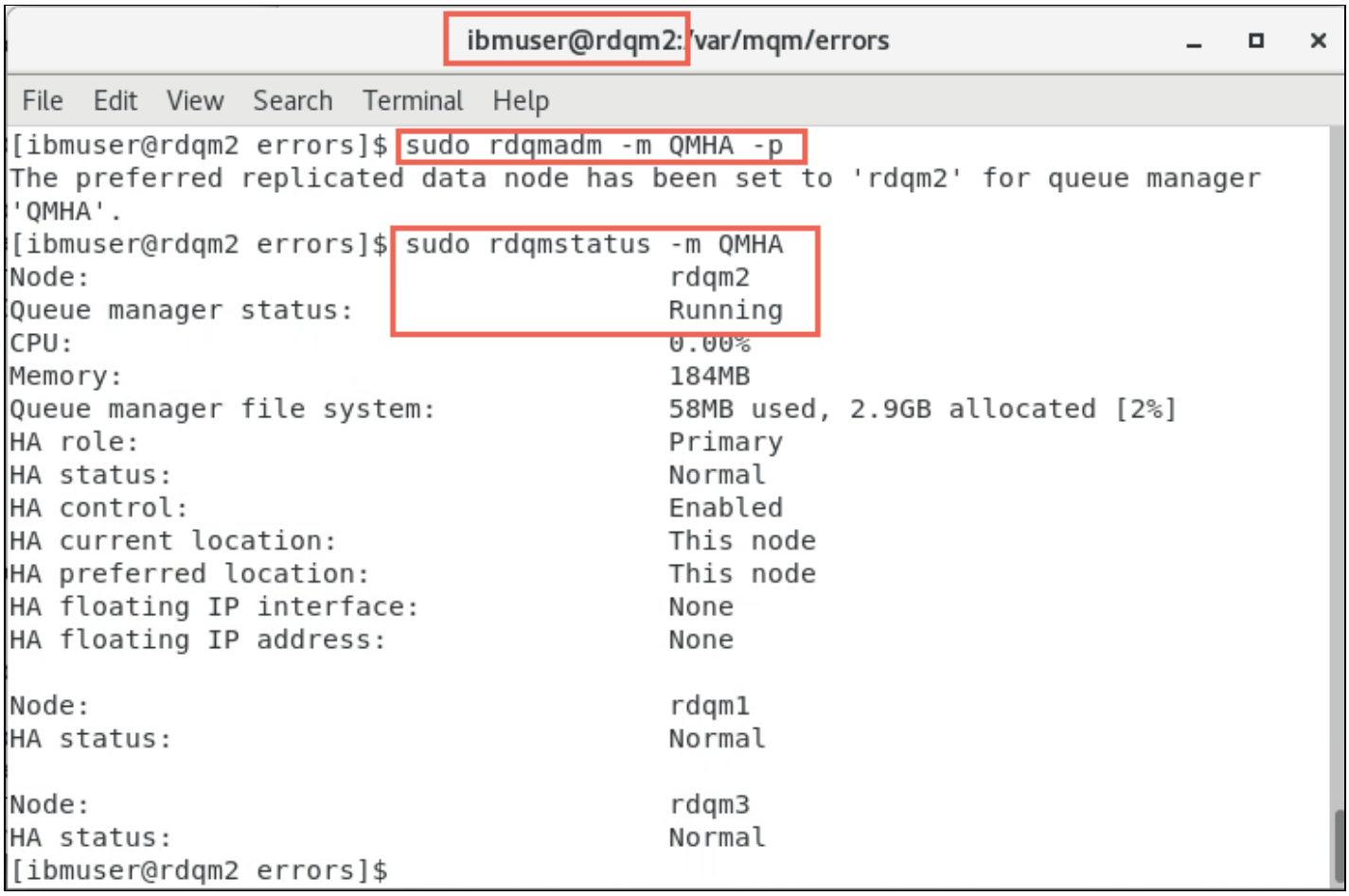
You will now use the approach of controlling where the RDQM runs by changing its preferred location, in this case to rdqm2.

1. Switch to **rdqm2**. In a new terminal window, run the following command as ibmuser:

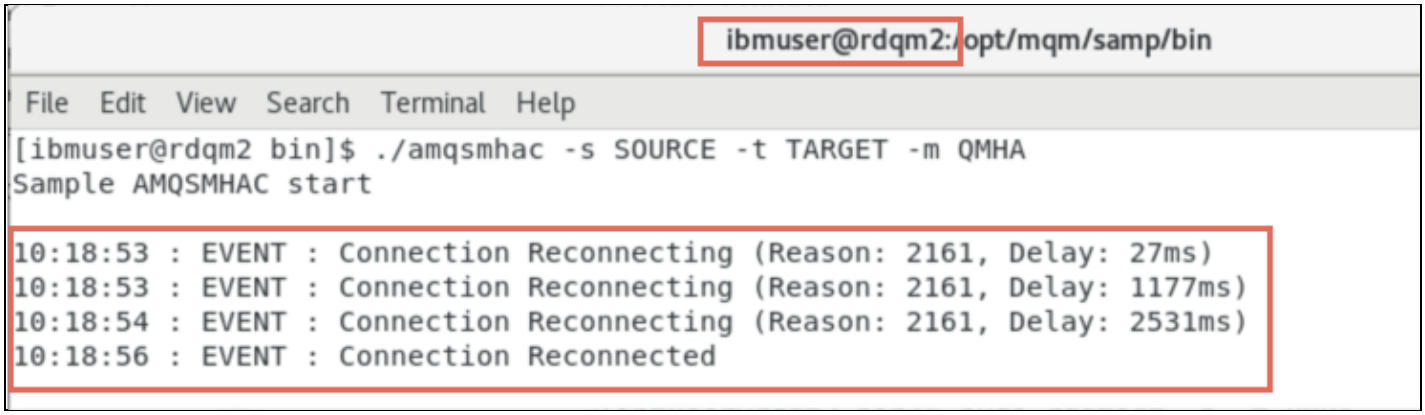
```
sudo rdqmadm -m QMHA -p
```

2. Check that the queue manager is indeed running on **rdqm2**, by running:

```
sudo rdqmstatus -m QMHA
```

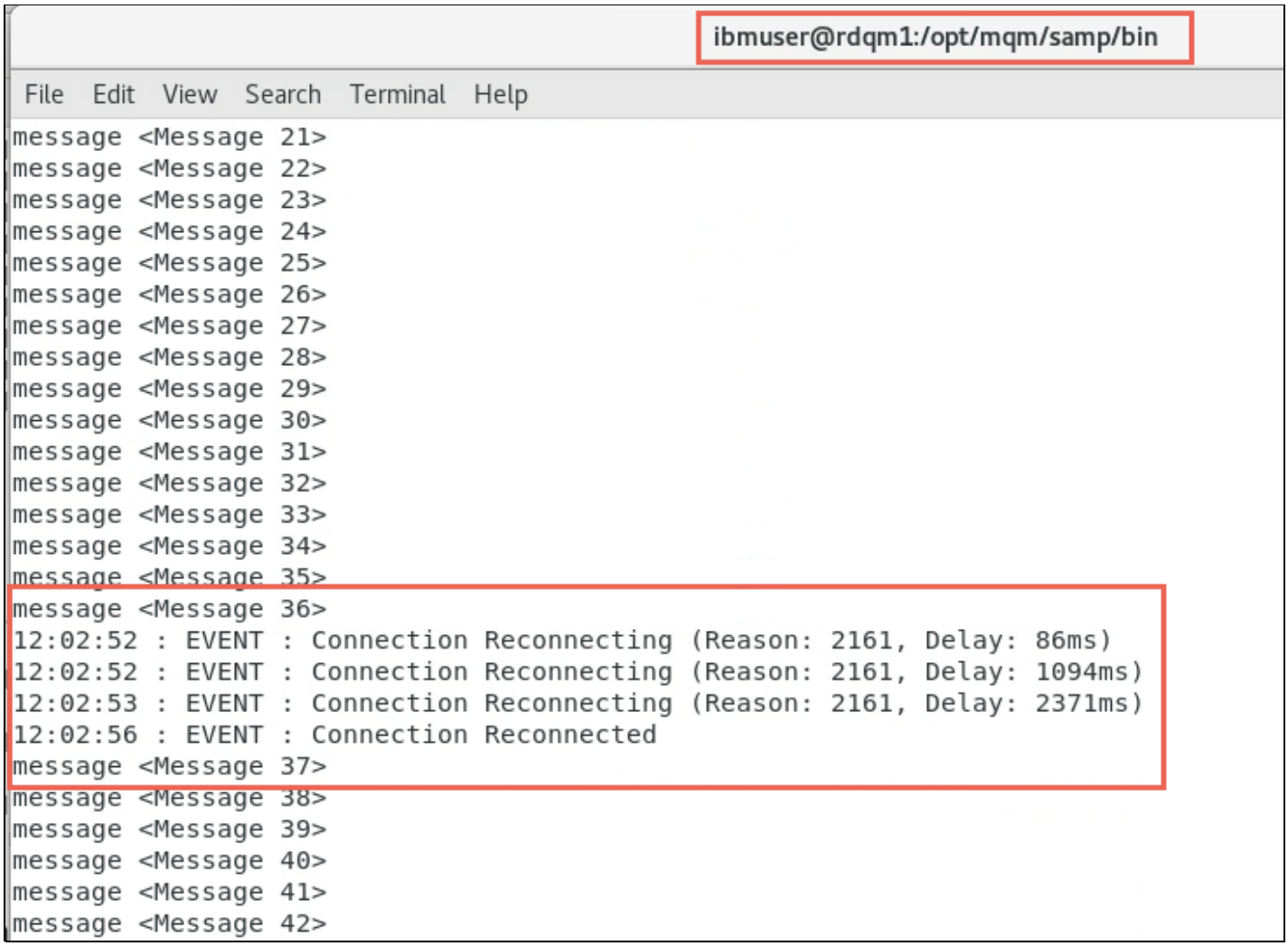


The output from the amqsmhac command, running in another window, should now be like this:



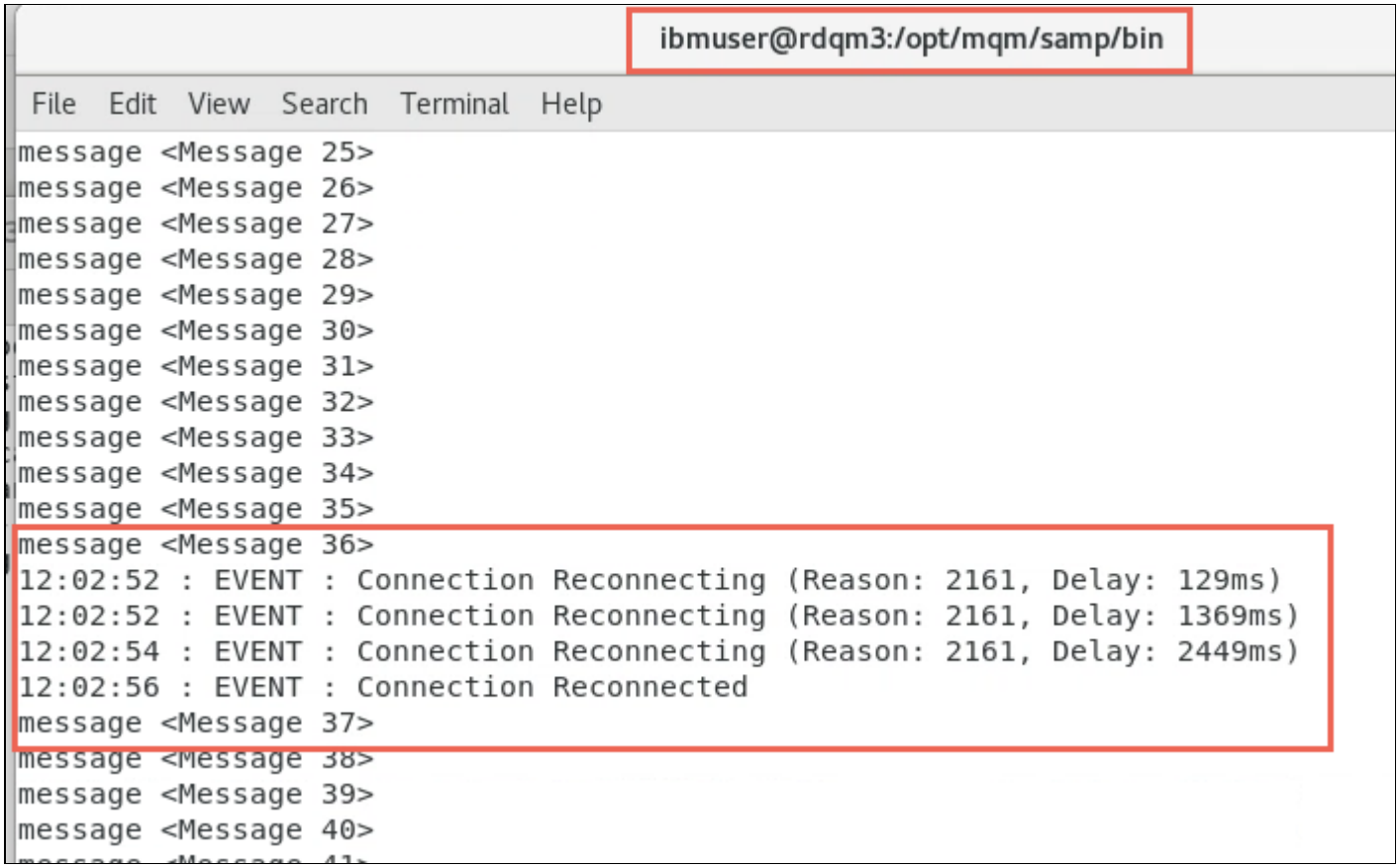
3. Now switch to **rdqm1**.

Messages should continue to be received without loss, by amqsghac, after it connects with the queue manager at the new location:



4. Now switch to **rdqm3**.

   The output from the amqsphac command, running on rdqm3, should similarly show messages continuing to be sent without loss:



Note: Now the queue manager is running on rdqm2, but each sample program is still able to communicate with it, this time using the second location specified in the MQSERVER environment variable:

CHANNEL1/TCP/10.0.1.1(1500),**10.0.1.2(1500)**,10.0.1.3(1500)

HA Sample programs with RDQM & Floating IP address

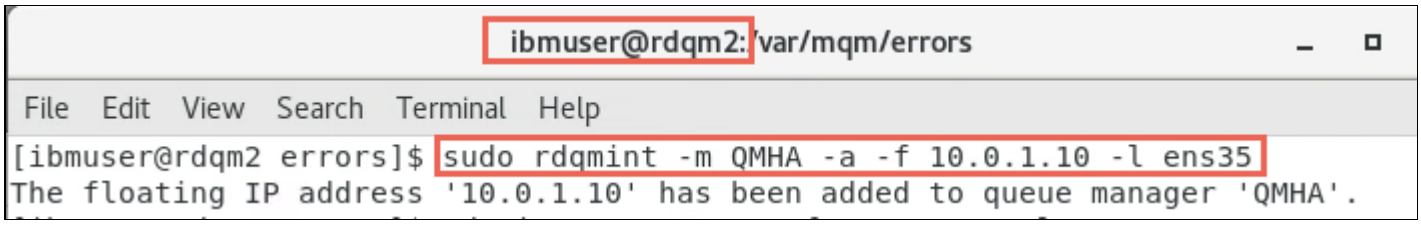
It is possible to associate a floating IP address with an RDQM so that it is not necessary to reconfigure clients, etc. with three IP addresses for the same queue manager. In this case, you will assign the floating address 10.0.1.10 to virtual adapter ens35, where currently you already have a fixed IP address configured on each virtual machine.

1. Stop (with **ctrl-C**) the HA sample programs that are currently running on each of the nodes.

   Do not close the terminal windows just yet as you will be running these programs again!

2. Switch to **rdqm2**. As this is currently the primary node for the QMHA queue manager, add the floating IP address, by running the command (as **ibmuser**):

```
sudo rdqmint -m QMHA -a -f 10.0.1.10 -l ens35
```

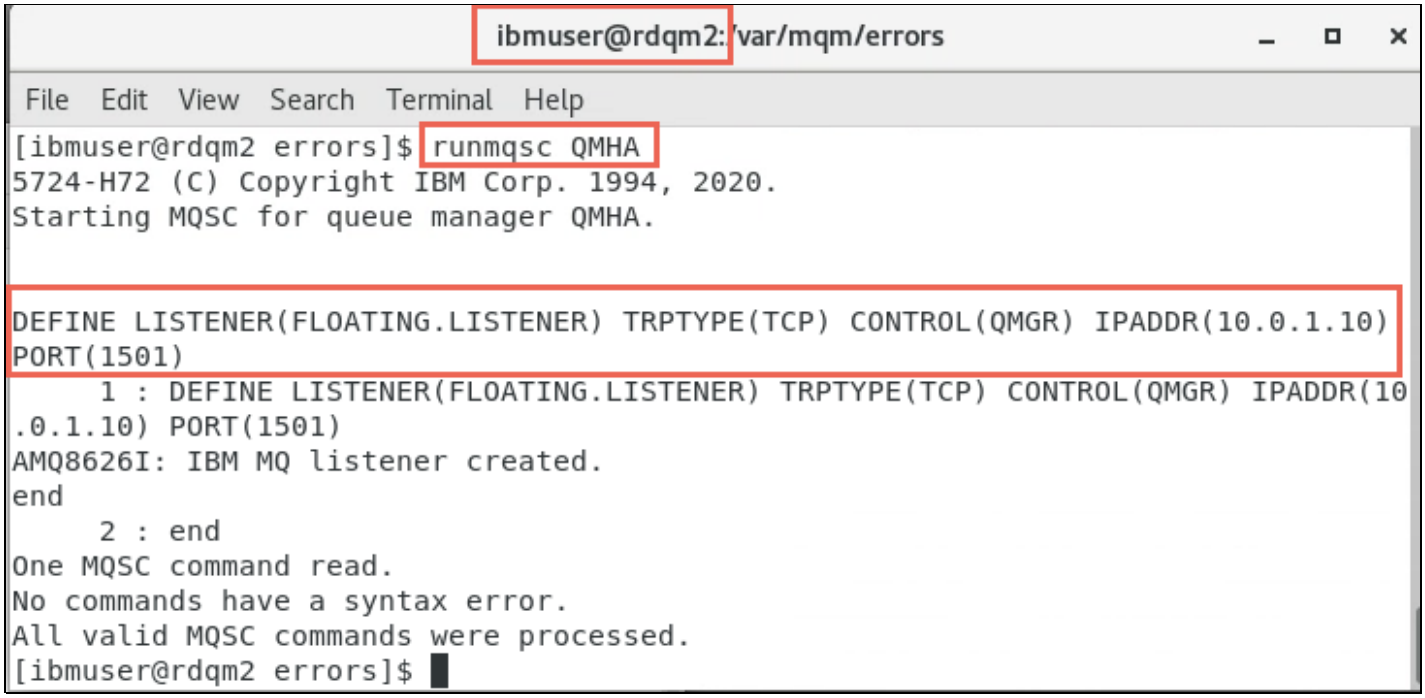


3. It is recommended to add another listener specifically for this floating IP address. By default, the standard listener will listen on the same port on every IP address, so a different port needs to be chosen for the additional listener. In a terminal window, create a listener by entering the runmqsc commands as follows:

```
runmqsc QMHA
```

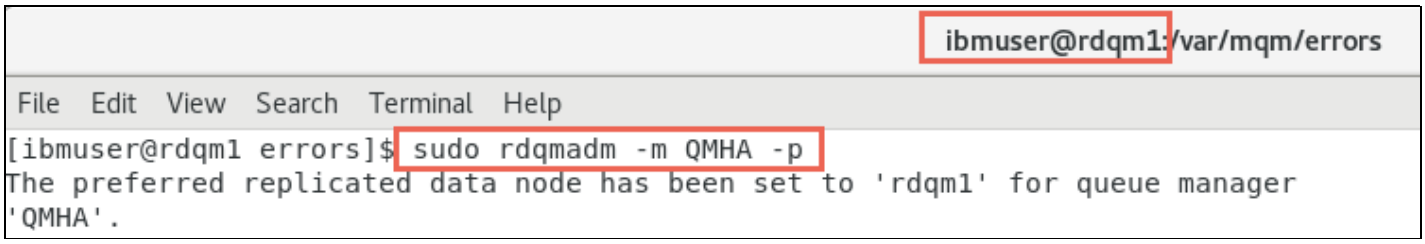
```
DEFINE LISTENER(FLOATING.LISTENER) TRPTYPE(TCP) CONTROL(QMGR) IPADDR(10.0.1.10) PO
RT(1501)
```

```
end
```



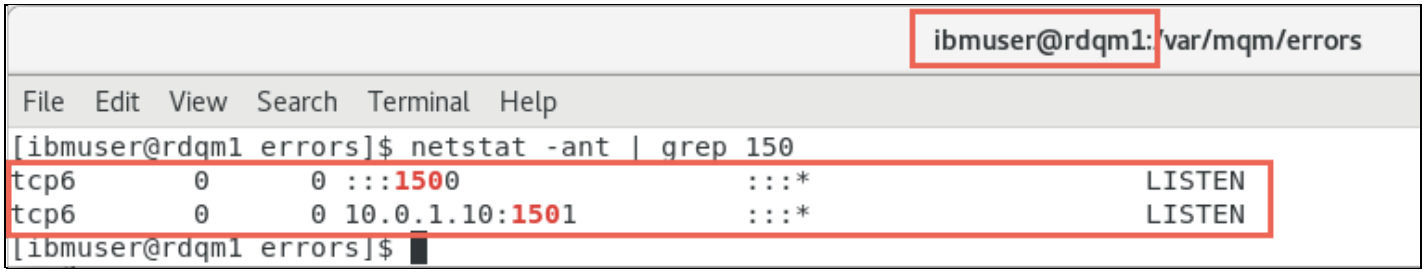
4. Switch to **rdqm1**. The new listener will not be started until the queue manager is restarted, so move the queue manager back to its original node by running the following command on node **rdqm1** in a terminal window, as user ibmuser:

```
sudo rdqmadm -m QMHA -p
```



You can check that both listeners are running by running the following command:

```
netstat -ant | grep 150
```



The first listener is the one created because -p **1500** was specified on the *crtmqm* command. This listener is listening on port 1500 on every IP address. The second listener, however, is listening on port **1501** on the floating IP address only.

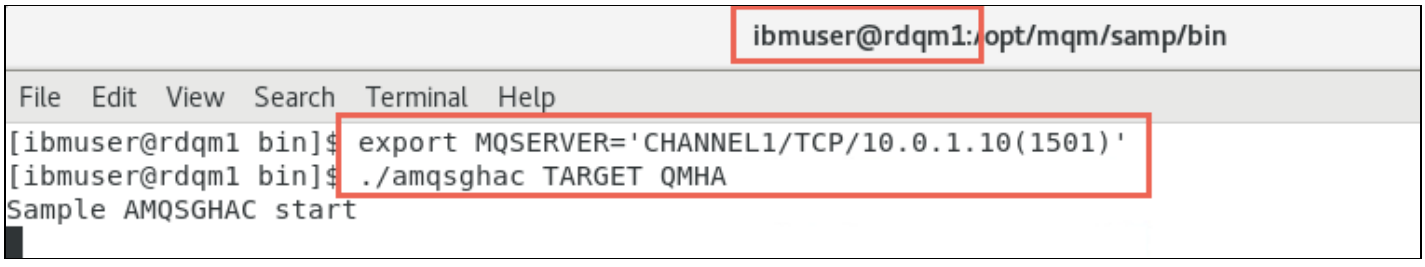
5. Now that you have a floating IP address associated with QMHA, you can change the MQSERVER environment variable to CHANNEL1/TCP/10.0.1.10(1501).

   Locate the window where the amqsghac program was running (on rdqm1) and enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.10(1501)'
```

6. Now re-run the command: **amqsghac TARGET QMHA**.

```
./amqsghac TARGET QMHA
```



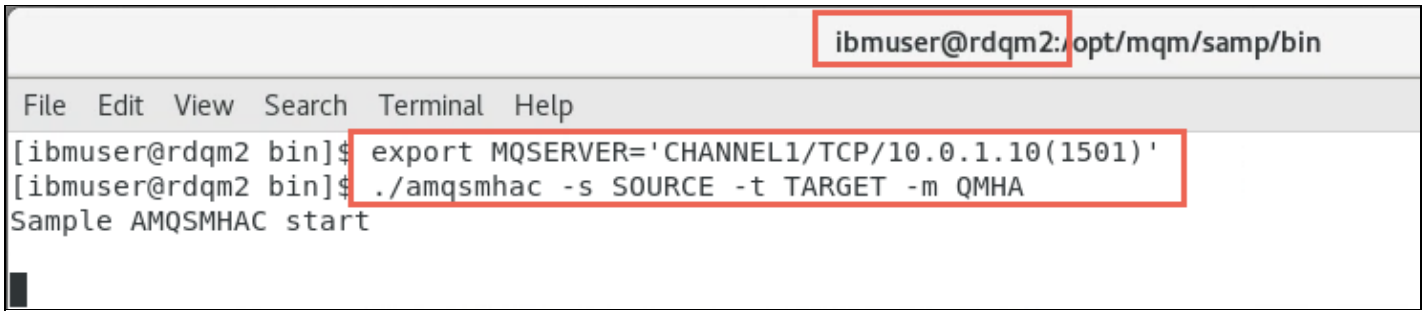
Leave this command to run!

7. Switch to **rdqm2**. Locate the window where the amqsmhac program was running and enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.10(1501)'
```

8. Now re-run the command: **amqsmhac -s SOURCE -t TARGET -m QMHA**

```
./amqsmhac -s SOURCE -t TARGET -m QMHA
```



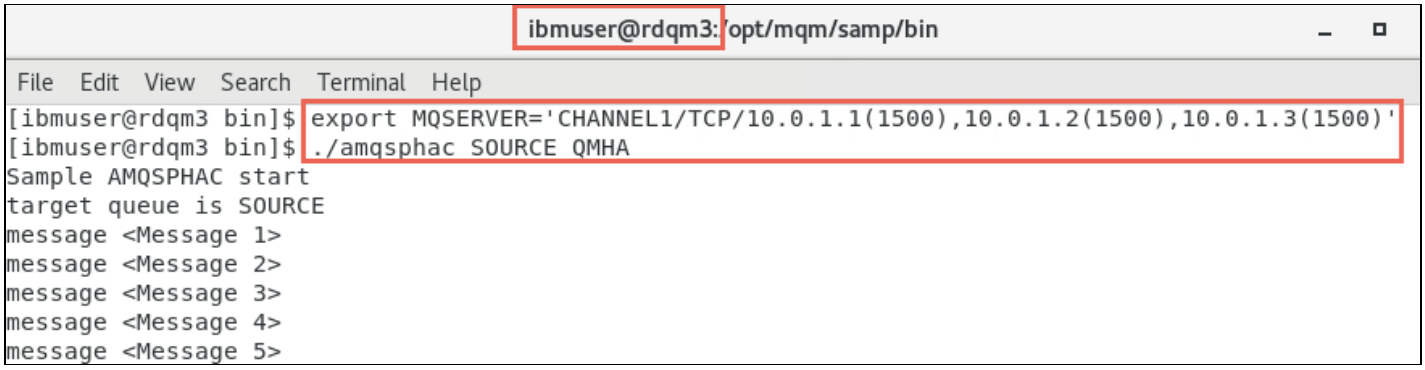
Leave this command to run!

9. Switch to **rdqm3**. Locate the window where the amqsphac program was running and enter:

```
export MQSERVER='CHANNEL1/TCP/10.0.1.10(1501)'
```

10. Now re-run the command: **amqsphac SOURCE QMHA**

```
./amqsphac SOURCE QMHA
```



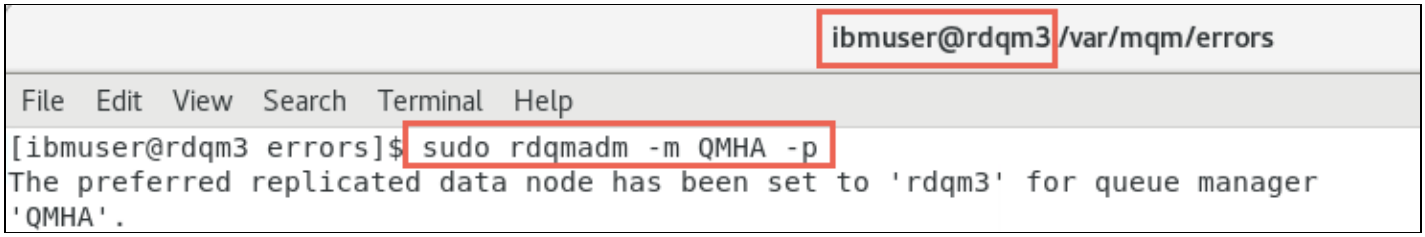
Leave this command to run!

11. Now repeat the test. This time move it to rdqm3.

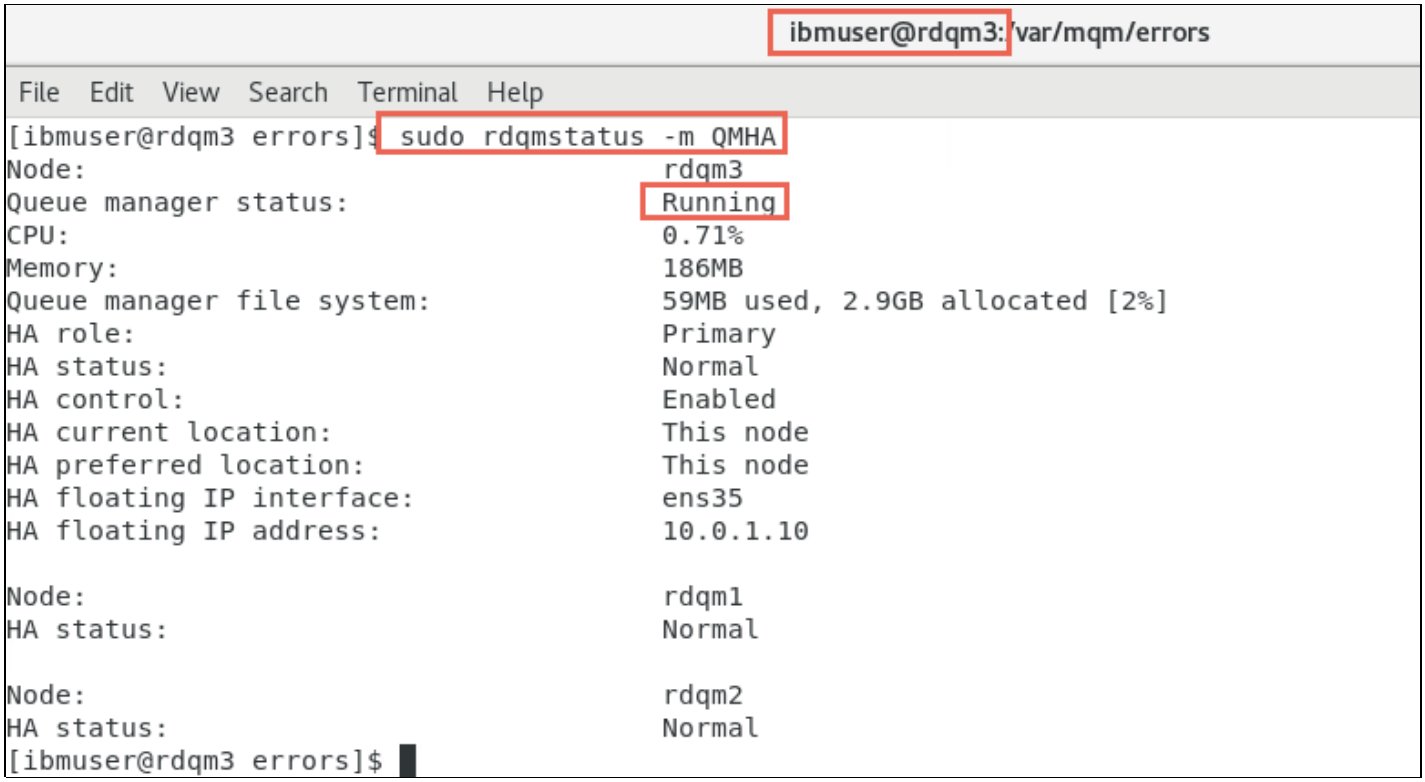
    a. On **rdqm3**, in the another terminal window as user ibmuser.

    b. Issue the command to move the RDQM to **rdqm3**.

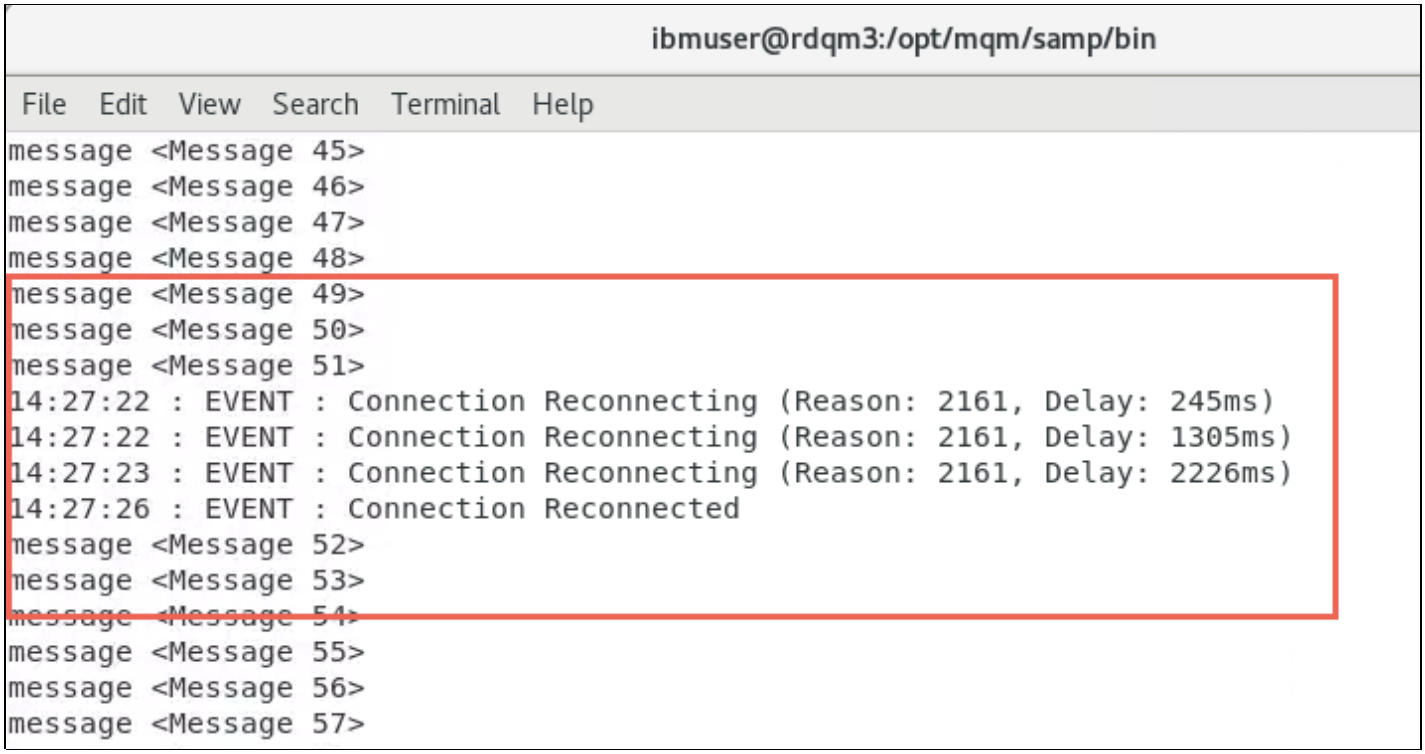
```
sudo rdqmadm -p -m QMHA
```

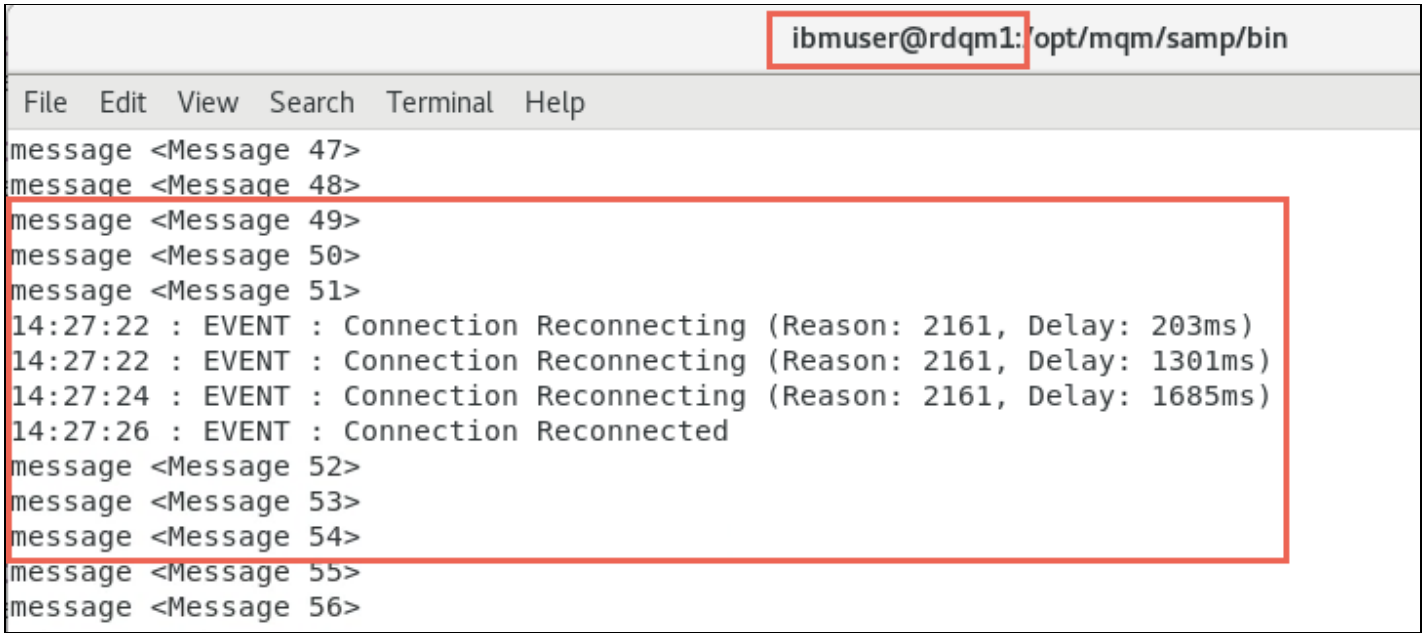


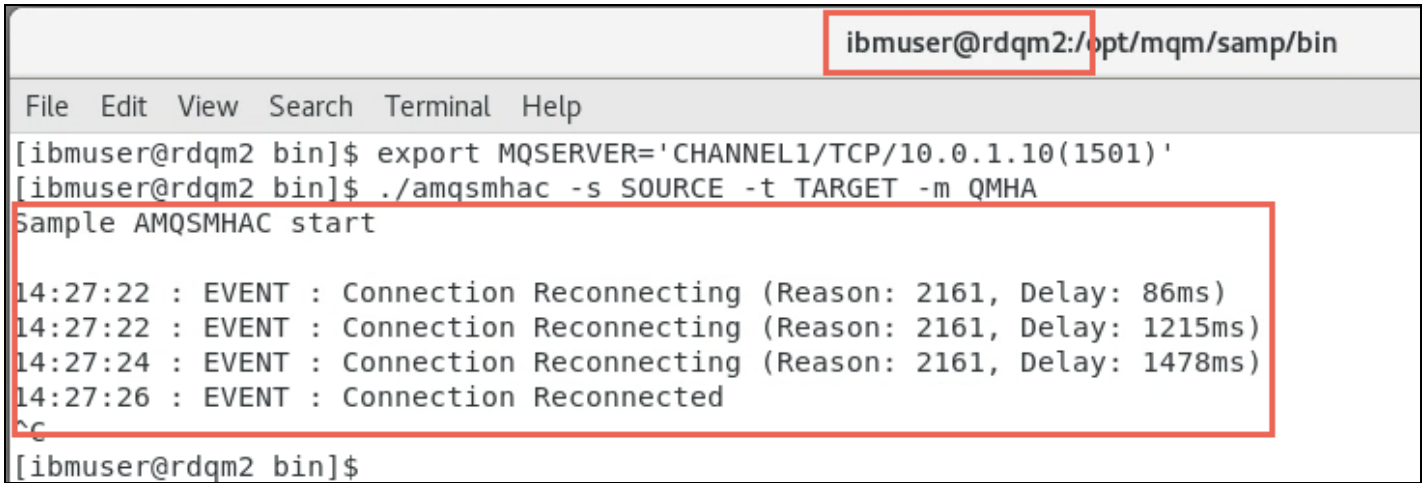
    c. Check the status of the queue manager to make sure it is running on **rdqm3**.



d. Confirm that messages continue to be sent and received without loss after the queue manager has been moved.





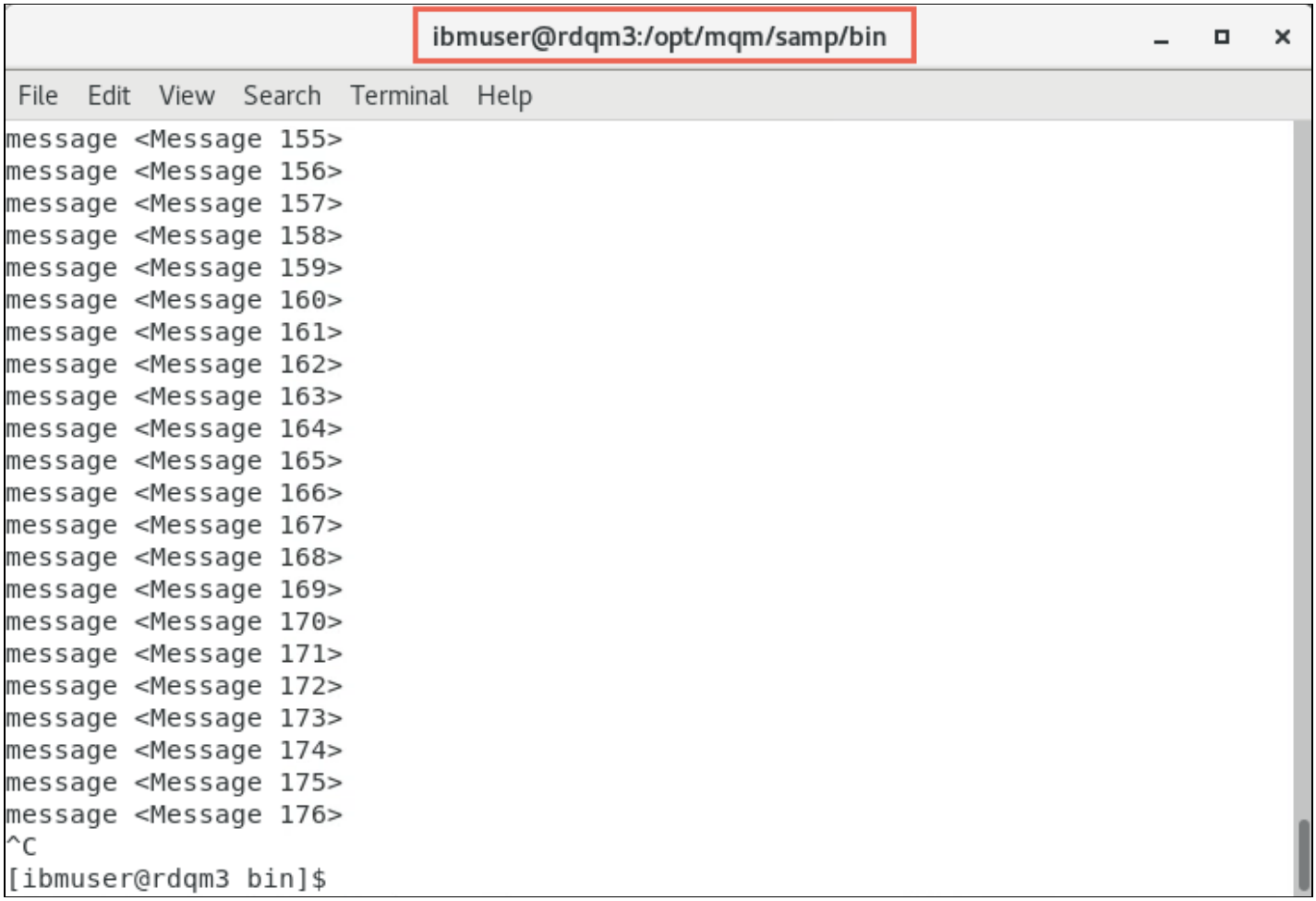


12. When completed testing, stop (with ctrl-C) the HA sample programs that are currently running on each of the nodes.

Cleanup environment

Stop and remove RDQMs

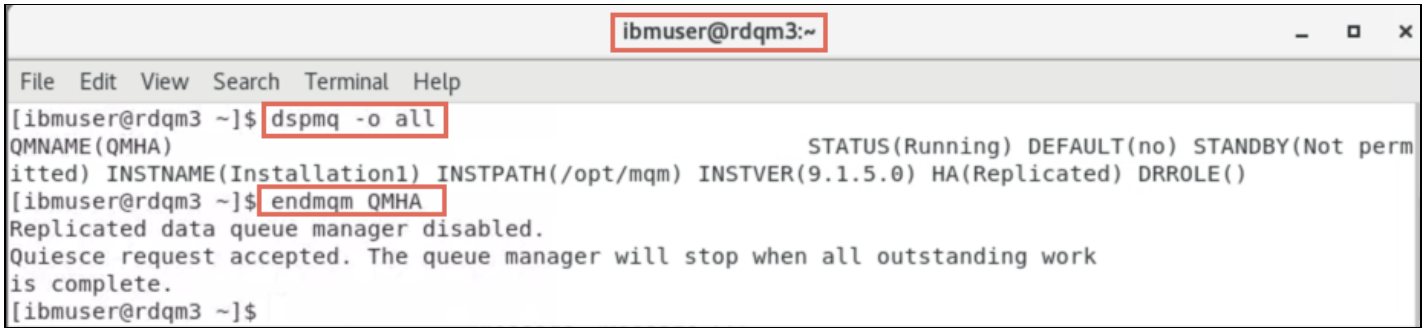
1. Stop the sample program on each node by entering *ctrl-c* in the terminal window.



2. On **rdqm3**, open a terminal window and stop running queue managers. Issue the following commands. Your displays and queue managers may not match the screenshots. Substitute your queue managers.

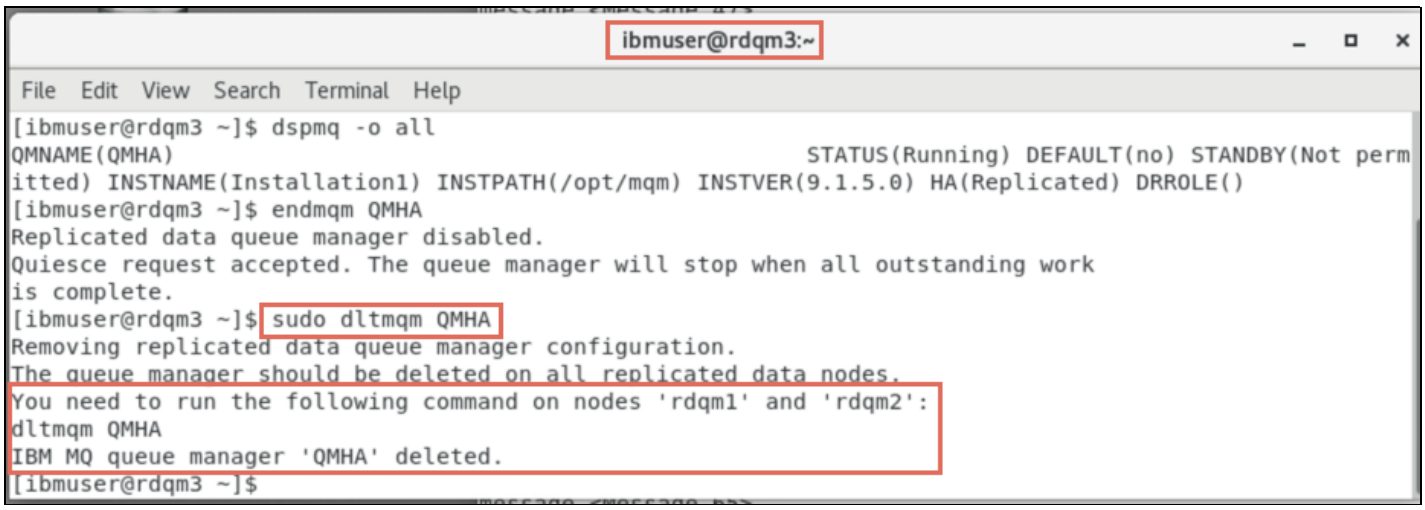
```
dsmpmq -o all
```

```
endmqm QMHA
```



3. Delete the queue manager *QMHA* on the primary node **rdqm3**.

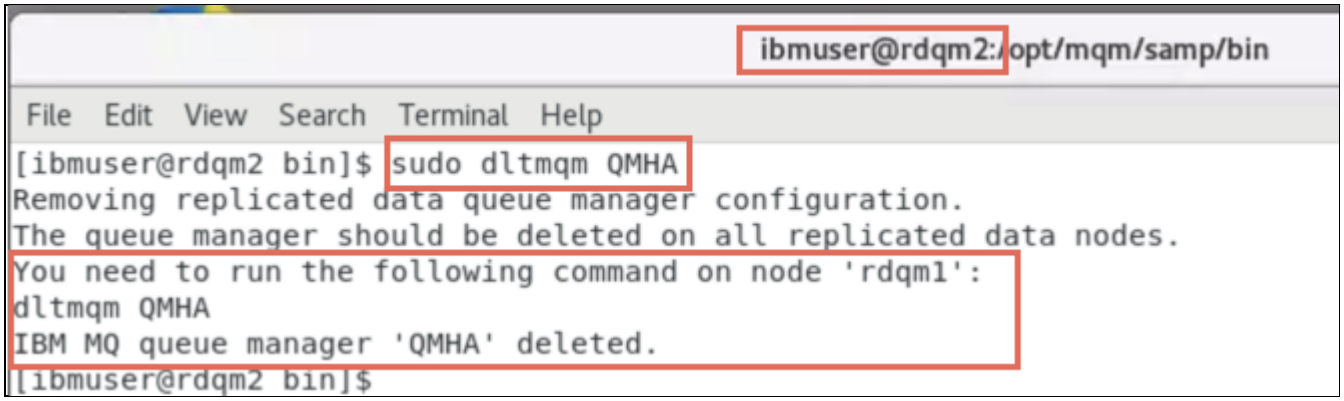
```
sudo dltmqm QMHA
```

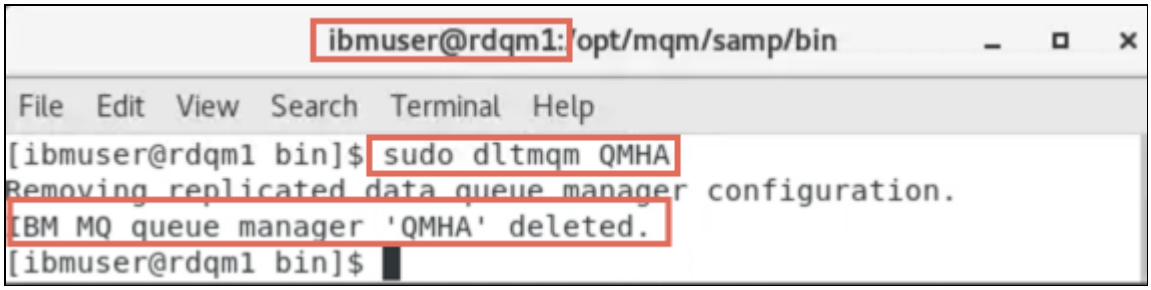


4. Delete the secondary queue managers on **rdqm2** and **rdqm1** per instructions in the output messages from previous commands.

   On **rdqm2** and **rdqm1**, delete the secondary queue manager *QMHA* with the following command:

```
sudo dltmqm QMHA
```

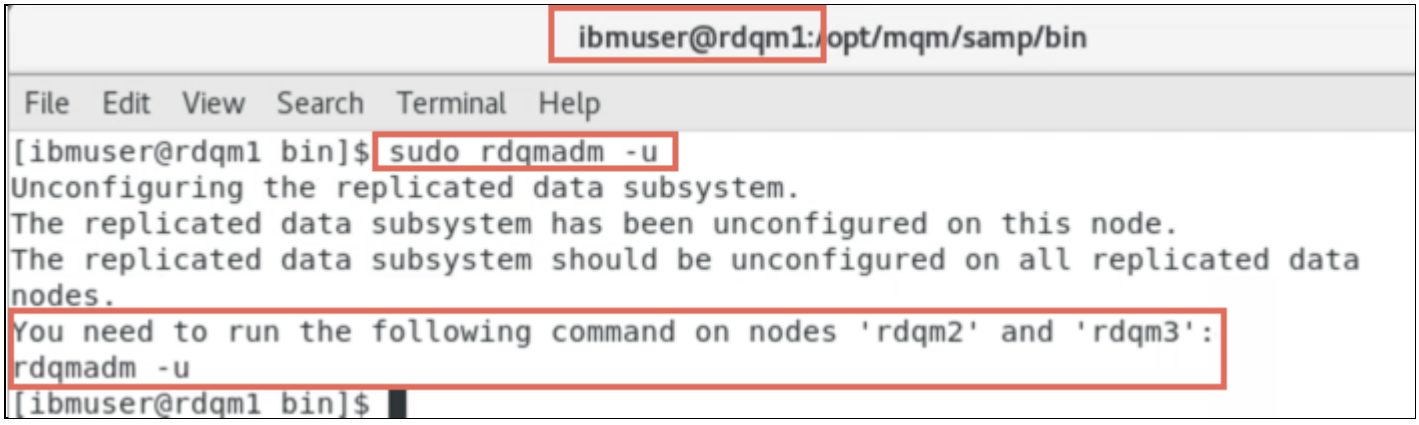




Delete HA Cluster

1. Still on **rdqm1** unconfigure (delete) the Pacemaker HA cluster.

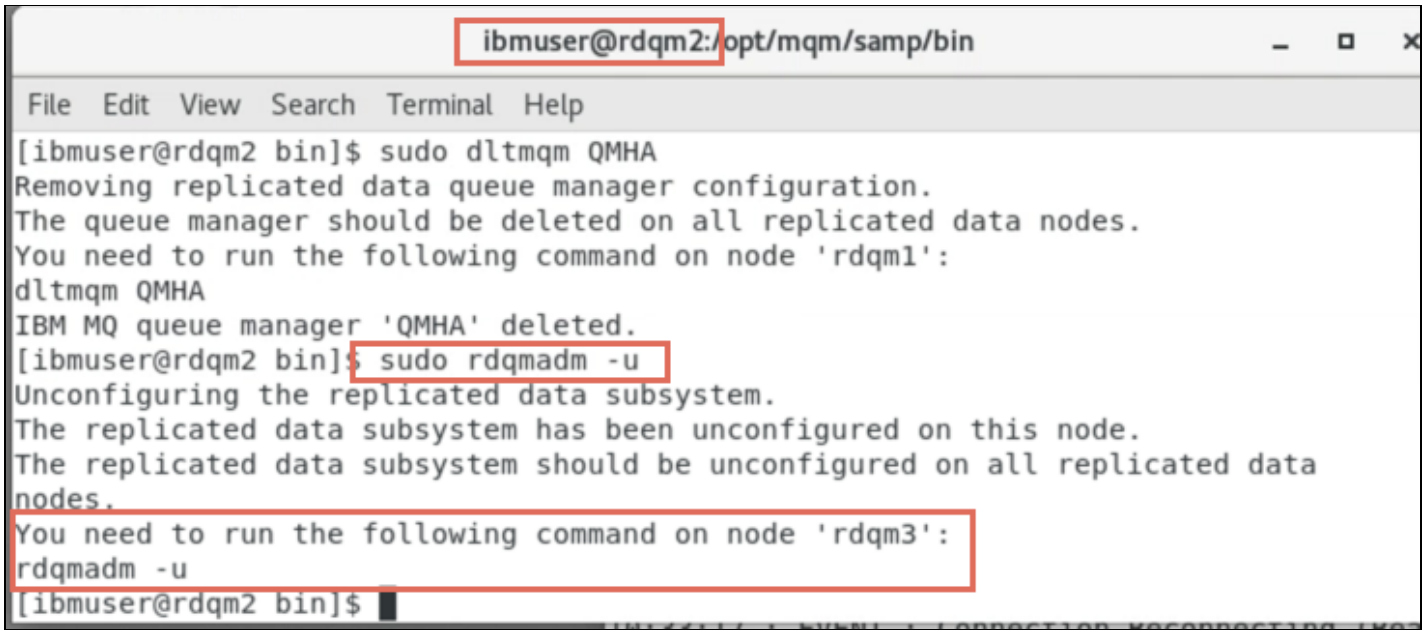
```
sudo rdqmadm -u
```

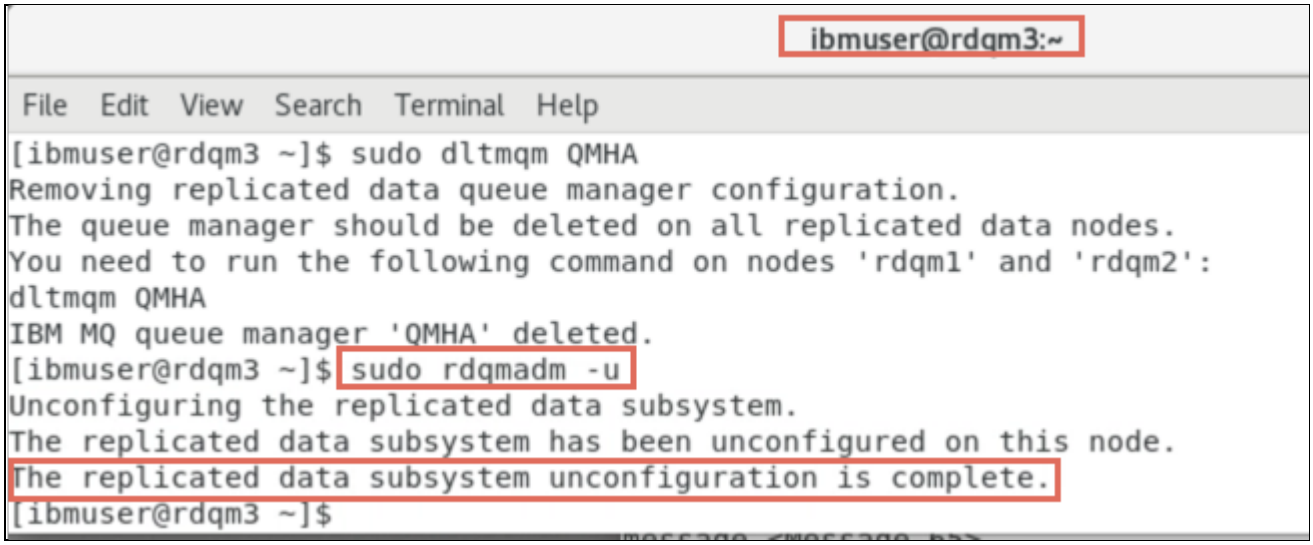


2. Delete the pacemaker HA cluster on the secondary nodes **rdqm2** and **rdqm3** per instructions in the output messages from previous commands.

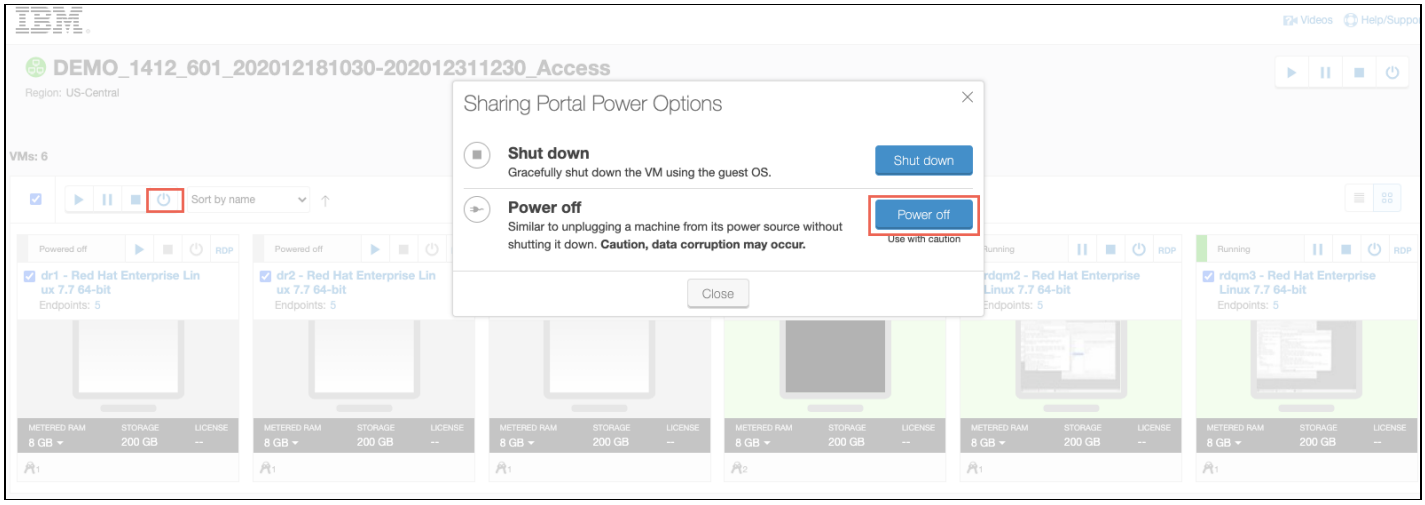
   On **rdqm2** and **rdqm3**, unconfigure the HA cluster.

```
sudo rdqmadm -u
```





3. Finally power off the VMs. Click the pull down menu and go to the *View VMs* display. The three VMs for this lab should still be running and their labels checked. Click the power icon and select *Power off*.



CONGRATULATIONS!

You have completed this hands-on lab.

You have created replicated data queue managers to provide high availability for IBM MQ, and you have tested failing over.

Continue to Lab 3 (mq_ha_pot_lab3.html)

Return MQ HA Menu (mq_ha_pot_overview.html)

Site last generated: Dec 28, 2020