**RDQM Build**

Contents

[1. Overview 3](#_Toc121218241)

[2. Update the repo file 3](#_Toc121218242)

[3. Install the PreReq and MQ binaries for RDQM clustering 4](#_Toc121218243)

[4. Dependency packages for Resource-Agents will need to be installed, perform 5](#_Toc121218244)

[5. Preparing hosts for SSH Key authentication 6](#_Toc121218245)

[6. Add Firewall/SELInux/ for Replication 6](#_Toc121218246)

[7. Building the RDQM Cluster 8](#_Toc121218247)

[8. Building your first RDQM Qmgr 8](#_Toc121218248)

[9. RDQM Qmgr Failover Steps 11](#_Toc121218249)

[10. Checking DRBD and Pacemaker status and logs 12](#_Toc121218250)

[11. IBM Questions 13](#_Toc121218251)

[12. Troubleshooting section 13](#_Toc121218252)

[13. Patching Considerations 13](#_Toc121218253)

[14. BACKUP/RESTORE policy 14](#_Toc121218254)

## Overview

The following document proceeds through the steps to install, build and run RDQM on a Linux server. It provides guidance on PreReqs, configuration and installation packages for MQ on Linux to support RDQM. This document has been used on a valid, running installation.

In this section, we will install the PreReqs for IBM MQ RDQM, including: DRBD, Pacemaker, and MQ binaries.

 NOTE: For v9.2, here is the IBM article for the task:

<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=multiplatforms-installing-rdqm-replicated-data-queue-managers>

<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=availability-requirements-rdqm-ha-solution>

Software Source: XXX-RHEL7-NonProd (Artifactory)

|  |  |
| --- | --- |
| Dev |  |
| NonProd |  |
| Prod |  |

## Update the repo file

Using the following information update the following file:  
/etc/yum.repos.d/rhel8\_artifactory\_new.repo

Before updating the file please take a backup as sudo

cd /etc/yum.repos.d

sudo cp -p rhel8\_artifactory\_new.repo rhel8\_artifactory\_new.repo\_<DATE>

sudo vi rhel8\_artifactory\_new.repo

Verify the Repo List

sudo yum repolist

|  |  |
| --- | --- |
| **Env** | **Yum repo source** |
| Dev/QC/BT | [XXX-RHEL7-NonProd]  name=XXX RHEL 8.x Software  baseurl=https://corp3artifactory01.XXX.org:443/artifactory/XXX-RHEL7-NonProd/x86/8.x/  sslverify = 0  enable=1  gpgcheck=0 |
| Prod | [XXX-RHEL7-Prod]  name=XXX RHEL 8.x Software  baseurl=https://corp3artifactory01.XXX.org:443/artifactory/XXX-RHEL7-Prod/x86/8.x/  sslverify = 0  enable=1  gpgcheck=0 |

MQ License deferred acceptance:

<https://www.ibm.com/support/pages/system/files/inline-files/MQ%209.2%20-%20deferring%20the%20acceptance%20of%20the%20license%20after%20the%20installation_1.pdf>

Prerequisites:

 Configure Artifactory repo (URL is above)



 Mqm account with sudo rights

## Install the PreReq and MQ binaries for RDQM clustering

* 1. Run "uname -r"

[mqm@<server> yum.repos.d]$ uname -r

4.18.0-193.el8.x86\_64

[mqm@<server> yum.repos.d]$

**Note :** Make sure the value of the above command output should be same in all environments otherwise installed binaries will be different.

* 1. If, output is "4.18.0-193.\*", perform: **sudo yum install kmod-drbd-9.0.28\_4.18.0\_193-1**  
     OR  
       
     If, output is "4.18.0-240.\*", perform: **sudo yum install kmod-drbd-9.0.28\_4.18.0\_240\***

Note : REFER to the IBM link for correct kernel modules listed :

<https://www.ibm.com/support/pages/ibm-mq-replicated-data-queue-manager-kernel-modules>

* 1. Install all the drbd packages, perform: **sudo yum install drbd-\***
  2. NOTE: Installs the following:
* drbd-bash-completion
* drbd-pacemaker
* drbd-udev
* drbd-utils

## Dependency packages for Resource-Agents will need to be installed, perform

**sudo yum -y install cifs-utils libtool-ltdl libxslt net-snmp-libs nfs-utils perl-TimeDate psmisc python36 python3-lxml  
  
sudo yum install cifs-utils nfs-utils**

* Install all the pacemaker packages, perform: **Note :** Please verify and make sure all "linbit" packages are installed .

**sudo yum install cluster-glue-libs-1.0.12.linbit-3.2+20210204+240ad6b5.el8 corosync-3.1.0.linbit-2.1.4.4a2f48b1.el8 corosynclib-3.1.0.linbit-2.1.4.4a2f48b1.el8 crmsh-4.3.0+git.1614332721.c866b885.linbit-1.0.el8 crmsh-scripts-4.3.0+git.1614332721.c866b885.linbit-1.0.el8 libknet1-1.20.linbit-1.0.el8 libnozzle1-1.20.linbit-1.0.el8 libqb-1.0.6.linbit-1.1.el8 pacemaker-2.0.5.linbit-1.0.el8 pacemaker-cli-2.0.5.linbit-1.0.el8 pacemaker-cluster-libs-2.0.5.linbit-1.0.el8 pacemaker-libs-2.0.5.linbit-1.0.el8 pacemaker-schemas-2.0.5.linbit-1.0.el8 python3-parallax-1.0.5.linbit-1.0 resource-agents-4.7.0.linbit-0.1.el8**

1. Accept the license for MQ prior to install, perform:

**sudo mkdir -p /tmp/mq\_license\_9.2.0/license**

**sudo touch /tmp/mq\_license\_9.2.0/license/status.dat**

1. Install the MQ Binaries (including RDQM), perform:

**sudo yum install MQSeriesGSKit-9.2.0 MQSeriesServer-9.2.0 MQSeriesRuntime-9.2.0 MQSeriesWeb-9.2.0 MQSeriesMan-9.2.0 MQSeriesSamples-9.2.0 MQSeriesRDQM-9.2.0**

1. Set the MQ installation to be primary:  
   **sudo /opt/mqm/bin/setmqinst -i -n Installation1**
2. Accept the MQ License with:  
   **sudo /opt/mqm/bin/mqlicense -accept**
3. Congratulations! You have installed the PreReq and MQ binaries, and are ready to setup SSH public key authentication.
4. Modify NOFILE kernel parameter for root on all nodes:
   1. Switch to the root user.
   2. mkdir /etc/systemd/system/pacemaker.service.d
   3. vi /etc/systemd/system/pacemaker.service.d/pacemaker.conf
   4. Update the new pacemaker.conf file with the [Service] stanza below and the Environment setting we want:**[Service]Environment="MQS\_GETGROUPLIST\_API=1" LimitNOFILE=65536LimitNPROC=32768**
   5. systemctl daemon-reload
   6. systemctl restart pacemaker.service

## Preparing hosts for SSH Key authentication

This section will cover creating a public/private key pair for SSH authentication between hosts, and deploying the key across the 3 RDQM nodes.

cd /home/mqm

sudo mkdir .ssh

sudo chmod 700 .ssh

sudo chown mqm:mqm .ssh

cd /home/mqm/.ssh

1. Generate an ED22519 key pair to use: **ssh-keygen -C mqm@RDQM -t ed25519 -a 100**
2. Copy the public key to both remote hosts: **ssh-copy-id mqm@<hostname>**
3. Perform this sequence on the other 2 nodes.
4. Modify visudo to allow mqm to run sudo commands passwordless, adding this line (**/etc/sudoers**):  
   **%mqm ALL=(ALL) NOPASSWD: ALL**
5. Congratulations! You should now be able to use passwordless-ssh to any MQ node from any MQ node.

## Add Firewall/SELInux/ for Replication

In this section, we will add the firewall rules for DRBD/pacemaker, and configure SELinux to allow DRBD execution.

1. Update SELinux to allow DRBD traffic: **sudo semanage permissive -a drbd\_t**

<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security-enhanced_linux/sect-security-enhanced_linux-fixing_problems-allowing_access_audit2allow>

It updates drbd related rules in : /etc/selinux/targeted/contexts/files/file\_contexts

---------------------------------

Verification for drbd :

[mqm@<server> etc]$ grep drbd /etc/selinux/targeted/contexts/files/file\_contexts

/dev/drbd[^/]\* -b system\_u:object\_r:fixed\_disk\_device\_t:s0

/var/lib/drbd(/.\*)? system\_u:object\_r:drbd\_var\_lib\_t:s0

/var/run/drbd(/.\*)? system\_u:object\_r:drbd\_var\_run\_t:s0

/sbin/drbdadm -- system\_u:object\_r:drbd\_exec\_t:s0

/sbin/drbdsetup -- system\_u:object\_r:drbd\_exec\_t:s0

/usr/sbin/drbdadm -- system\_u:object\_r:drbd\_exec\_t:s0

/usr/sbin/drbdsetup -- system\_u:object\_r:drbd\_exec\_t:s0

/etc/rc\.d/init\.d/drbd -- system\_u:object\_r:drbd\_initrc\_exec\_t:s0

/var/lock/subsys/drbd -- system\_u:object\_r:drbd\_lock\_t:s0

/usr/lib/ocf/resource\.d/linbit/drbd -- system\_u:object\_r:drbd\_exec\_t:s0

[mqm@bt1chpisomq01 etc]$

-----------------------------------------------------------

1. Update FirewallD for RDQM ports (DRBD, Pacemaker, and default MQ port 1414) **/opt/mqm/samp/rdqm/firewalld/configure.sh**  
     
   **Console Output :**[mqm@<server> etc]$ sudo /opt/mqm/samp/rdqm/firewalld/configure.sh

+ MQ\_INSTALLATION\_PATH=/opt/mqm

+ cp /opt/mqm/samp/rdqm/firewalld/services/rdqm-drbd.xml /usr/lib/firewalld/services

+ chmod 644 /usr/lib/firewalld/services/rdqm-drbd.xml

+ cp /opt/mqm/samp/rdqm/firewalld/services/rdqm-pacemaker.xml /usr/lib/firewalld/services

+ chmod 644 /usr/lib/firewalld/services/rdqm-pacemaker.xml

+ cp /opt/mqm/samp/rdqm/firewalld/services/rdqm-mq.xml /usr/lib/firewalld/services

+ chmod 644 /usr/lib/firewalld/services/rdqm-mq.xml

+ firewall-cmd --reload

success

+ firewall-cmd --permanent --add-service=rdqm-drbd

success

+ firewall-cmd --permanent --add-service=rdqm-pacemaker

success

+ firewall-cmd --permanent --add-service=rdqm-mq

success

+ firewall-cmd --reload

Success

Check if 2nd NIC has been configured with:  
  
**sudo nmcli con show**

1. If 2nd NIC does not exist, then add and configure with:  
   **sudo nmcli con add type ethernet con-name ens224 ifname ens224 ip4 xxx.254.XXX.20/24 ipv4.method "manual"**
2. If 2nd NIC does exist, then mod it with:  
   **sudo nmcli con mod ens224 ipv4.addresses "xxx.xxx.136.xx/24"**
3. Add a secondary IP (Only when 2nd NIC isn't created by Unix)  
   **sudo nmcli con mod ens192 +ipv4.addresses "xxx.xxxx.136.xx/24"  
   sudo nmcli con reload  
   sudo nmcli con up ens192**
4. Add mqm to haclient and svc\_bmcuser to mqm group  
   **sudo usermod -aG haclient mqm  
   sudo usermod -aG mqm svc\_bmcuser**

## Building the RDQM Cluster

In this section, we will build an rdqm.ini file that contains the node information.

NOTE: IBM Article for this section:  
<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=availability-defining-pacemaker-cluster-ha-group>

PreReqs for this section:

 DNS records are required for each node, so FQDN can resolve to an IP. (This can be local entries, if needed.)

 SSH authentication keys are needed between hosts.

 Plan IP's for the cluster: Primary, Alternate (Optional), Replication (Optional, but recommended)  
NOTE: DRBD traffic is unencrypted, recommended to isolated VLAN for traffic.

We can request Unix SA to assign an IP for second NIC for DRBD Traffic or use below commands to

sudo nmcli con add type ethernet con-name ens224 ifname ens224 ip4 xxx.254.3.22/24 ipv4.method "manual"

1. Gather the IP and DNS names for each node.
2. Open the /var/mqm/rdqm.ini on a node.
3. Copy the RDQM node configuration for all 3 nodes.
4. Save, exit, and copy the file to the other 2 nodes.

Run the following command to create RDQM Cluster as root or sudo on one node.

sudo rdqmadm -c

<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=linux-configuring-tuning-operating-system>

## Building your first RDQM Qmgr

In this section, we will create a Qmgr on the RDQM cluster, and see that it is registered on the other nodes.

Note: For `crtmqm` command, the -fs takes a simple number in GB, or it takes "100M" or "20G" style syntax.

1. Log onto the node that you would prefer run the qmgr (NODE 01)
2. Confirm the pacemaker service is running (either rdqmstatus or systemctl status)
3. Create the qmgr:

**crtmqm -sx -fs 3G -lf 16384 -lp 12 -ls 24 <qmgrname>**

(This will build the qmgr across the nodes, and a 3GB filesystem for data and log files.)

(Default state of the qmgr will be "Running" on the local node, and it will \*prefer\* that node)

1. To add a floating IP, first identify the VIP address, then:

**rdqmint -m <qmgrname> -a -f <VIP Address> -l <InterfaceName>**

1. (Optional) Remove preferred node on qmgr, preventing failback, but not failover:

**rdqmadm -p -m <QmgrName> -d**

1. Congratulations, you have created your first qmgr, prepared it for failover, and prevented failback (double outage).
2. After QMGRs build is completed, as a validation step :

* STOP the qmgrs
* GET the node(s) rebooted by UNIX team
* START the qmgrs
* df -h (to see the drdb volume groups)
* Run the following commands : **drbdadm status**, **crm status** and **rdqmstatus -n**

**Console output – VERIFICATION process**

df -h (console output : For e.g)

[mqm@<server> ~]$ df -h

Filesystem Size Used Avail Use% Mounted on

devtmpfs 3.8G 0 3.8G 0% /dev

tmpfs 3.9G 47M 3.8G 2% /dev/shm

tmpfs 3.9G 808K 3.9G 1% /run

tmpfs 3.9G 0 3.9G 0% /sys/fs/cgroup

/dev/mapper/rhel-root 9.8G 3.0G 6.4G 32% /

/dev/sda1 2.0G 148M 1.7G 9% /boot

/dev/mapper/rhel-var 9.8G 536M 8.8G 6% /var

/dev/mapper/rhel-opt 9.8G 1.2G 8.1G 13% /opt

/dev/mapper/rhel-tmp 7.9G 207M 7.2G 3% /tmp

/dev/mapper/rhel-home 9.8G 7.5G 1.8G 81% /home

/dev/mapper/DATA00-opt\_mqm 32G 1.2G 29G 4% /opt/mqm

/dev/mapper/DATA00-var\_mqm 32G 61M 30G 1% /var/mqm

/dev/mapper/rhel-var\_log 9.8G 3.0G 6.3G 33% /var/log

/dev/mapper/rhel-var\_log\_audit 9.8G 760M 8.6G 9% /var/log/audit

/dev/drbd100 20G 925M 18G 5% /var/mqm/vols/vol00

/dev/drbd101 9.8G 826M 8.5G 9% /var/mqm/vols/volim00

/dev/drbd102 20G 845M 18G 5% /var/mqm/vols/vol01

/dev/drbd103 9.8G 847M 8.5G 9% /var/mqm/vols/volim01

tmpfs 782M 0 782M 0% /run/user/1011

drbdadm status (console output : For e.g)

[mqm@<server>01 ~]$ drbdadm status

dchips00 role:Primary

disk:UpToDate

<server>02.xxxx.org role:Secondary

peer-disk:UpToDate

<server>03.xxxx.org role:Secondary

peer-disk:UpToDate

dchips01 role:Primary

disk:UpToDate

<server>02.xxxx.org role:Secondary

peer-disk:UpToDate

<server>03.xxxx.org role:Secondary

peer-disk:UpToDate

dchipsim00 role:Primary

disk:UpToDate

<server>02.xxxx.org role:Secondary

peer-disk:UpToDate

<server>03.xxxx.org role:Secondary

peer-disk:UpToDate

dchipsim01 role:Primary

disk:UpToDate

<server>02.xxxx.org role:Secondary

peer-disk:UpToDate

<server>03.xxxx.org role:Secondary

peer-disk:UpToDate

crm status (console output : For e.g)

[mqm@<server>01 ~]$ crm status

Cluster Summary:

\* Stack: corosync

\* Current DC: <server>01.xxxx.org (version 2.0.5.linbit-1.0.el8-ba59be712) - partition with quorum

\* Last updated: Thu Sep 8 16:03:59 2022

\* Last change: Thu Sep 1 12:39:36 2022 by mqm via cibadmin on <server>01.xxxx.org

\* 3 nodes configured

\* 28 resource instances configured

Node List:

\* Online: [ <server>01.xxxx.org <server>02.xxxx.org <server>03.xxxx.org ]

Full List of Resources:

\* p\_fs\_dchips00 (ocf::heartbeat:Filesystem): Started <server>01.xxxx.org

\* p\_rdqmx\_dchips00 (ocf::ibm:rdqmx): Started <server>01.xxxx.org

\* dchips00 (ocf::ibm:rdqm): Started <server>01.xxxx.org

\* Clone Set: ms\_drbd\_dchips00 [p\_drbd\_dchips00] (promotable):

\* Masters: [ <server>01.xxxx.org ]

\* Slaves: [ <server>02.xxxx.org <server>03.xxxx.org ]

\* p\_fs\_dchipsim00 (ocf::heartbeat:Filesystem): Started <server>01.xxxx.org

\* p\_rdqmx\_dchipsim00 (ocf::ibm:rdqmx): Started <server>01.xxxx.org

\* dchipsim00 (ocf::ibm:rdqm): Started <server>01.xxxx.org

\* Clone Set: ms\_drbd\_dchipsim00 [p\_drbd\_dchipsim00] (promotable):

\* Masters: [ <server>01.xxxx.org ]

\* Slaves: [ <server>02.xxxx.org <server>03.xxxx.org ]

\* p\_fs\_dchips01 (ocf::heartbeat:Filesystem): Started <server>01.xxxx.org

\* p\_rdqmx\_dchips01 (ocf::ibm:rdqmx): Started <server>01.xxxx.org

\* dchips01 (ocf::ibm:rdqm): Started dev3chpisomq01.xxxx.org

\* Clone Set: ms\_drbd\_dchips01 [p\_drbd\_dchips01] (promotable):

\* Masters: [ <server>01.xxxx.org ]

\* Slaves: [ <server>02.xxxx.org <server>03.xxxx.org ]

\* p\_fs\_dchipsim01 (ocf::heartbeat:Filesystem): Started <server>01.xxxx.org

\* p\_rdqmx\_dchipsim01 (ocf::ibm:rdqmx): Started <server>01.xxxx.org

\* dchipsim01 (ocf::ibm:rdqm): Started <server>01.xxxx.org

\* Clone Set: ms\_drbd\_dchipsim01 [p\_drbd\_dchipsim01] (promotable):

\* Masters: [ <server>01.xxxx.org ]

\* Slaves: [ <server>02.xxxx.org <server>03.xxxx.org ]

\* p\_ip\_dchips00 (ocf::heartbeat:IPaddr2): Started <server>01.xxxx.org

dspmq (console output : For e.g)

mqm@<server>01 ~]$ dspmq

QMNAME(DCHIPS00) STATUS(Running)

QMNAME(DCHIPSIM00) STATUS(Running)

QMNAME(DCHIPS01) STATUS(Running)

QMNAME(DCHIPSIM01) STATUS(Running)

rdqmstatus -n (console output : For e.g)

[mqm@<server>01 ~]$ rdqmstatus -n

Node <server>01.xxxx.org is online

Node <server>02.xxxx.org is online

Node <server>03.xxxx.org is online

rdqmstatus (console output : For e.g)

[mqm@<server>01 ~]$ rdqmstatus

Node: <server>01.xxxx.org

Queue manager name: DCHIPS00

Queue manager status: Running

HA current location: This node

Queue manager name: DCHIPSIM00

Queue manager status: Running

HA current location: This node

Queue manager name: DCHIPS01

Queue manager status: Running

HA current location: This node

Queue manager name: DCHIPSIM01

Queue manager status: Running

HA current location: This node

[mqm@dev3chpisomq01 ~]$

## RDQM Qmgr Failover Steps

This section has been moved to here: [RDQM Qmgr Failover Steps](onenote:#RDQM%20-%20Cluster%20Admin%20Notes&section-id={6B1CF37D-DC72-4D1D-A3DB-3DCD31AF59F9}&page-id={B1A9FB78-F9A1-414A-9AD9-C4B50748A589}&object-id={DC8ABF37-5C2A-49F5-8002-67606FC76096}&2C&base-path=https://tchsp.tchcorp.org/infrastructure/middleware/ApplicationDocuments/Middleware%20Systems/IBM%20MQ/RHEL.one)

SUSPEND a Node and RECONNECT :

rdqmadm -s <NODE-NAME> Suspend node

Reconnect a Node :

rdamadm -r <NODE-NAME> Resume node

## Checking DRBD and Pacemaker status and logs

In this section, we are diving under RDQM and looking at DRBD and Pacemaker services and logs.

Each node runs a SystemD service called Pacemaker, this runs binaries for:

* + Pacemaker
  + DRBD
  + Multiple Qmgr processes

Check the service with:

**systemctl status pacemaker.service**

 Console output :

[mqm@<server>01 ~]$ systemctl status pacemaker.service

â— pacemaker.service - Pacemaker High Availability Cluster Manager

Loaded: loaded (/usr/lib/systemd/system/pacemaker.service; enabled; vendor preset: disabled)

Drop-In: /etc/systemd/system/pacemaker.service.d

â””â”€pacemaker.conf

Active: active (running) since Thu 2022-09-01 12:34:59 EDT; 1 weeks 0 days ago

Docs: man:pacemakerd

https://clusterlabs.org/pacemaker/doc/en-US/Pacemaker/2.0/html-single/Pacemaker\_Explained/index.html

Main PID: 2027 (pacemakerd)

Tasks: 7

Memory: 106.1M

CGroup: /system.slice/pacemaker.service

â”œâ”€2027 /usr/sbin/pacemakerd -f

â”œâ”€2060 /usr/libexec/pacemaker/pacemaker-based

â”œâ”€2061 /usr/libexec/pacemaker/pacemaker-fenced

â”œâ”€2062 /usr/libexec/pacemaker/pacemaker-execd

â”œâ”€2063 /usr/libexec/pacemaker/pacemaker-attrd

â”œâ”€2064 /usr/libexec/pacemaker/pacemaker-schedulerd

â””â”€2065 /usr/libexec/pacemaker/pacemaker-controld

Sep 08 15:24:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: Transition 703 (Complete=0, Pending=0, Fired=0, Skipped=0, Incomplete=0, S>

Sep 08 15:24:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: State transition S\_TRANSITION\_ENGINE -> S\_IDLE

Sep 08 15:39:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: State transition S\_IDLE -> S\_POLICY\_ENGINE

Sep 08 15:39:58 <server>01.xxxx.org pacemaker-schedulerd[2064]: notice: Calculated transition 704, saving inputs in /var/lib/pacemaker/pengine/p>

Sep 08 15:39:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: Transition 704 (Complete=0, Pending=0, Fired=0, Skipped=0, Incomplete=0, S>

Sep 08 15:39:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: State transition S\_TRANSITION\_ENGINE -> S\_IDLE

Sep 08 15:54:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: State transition S\_IDLE -> S\_POLICY\_ENGINE

Sep 08 15:54:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: Transition 705 (Complete=0, Pending=0, Fired=0, Skipped=0, Incomplete=0, S>

Sep 08 15:54:58 <server>01.xxxx.org pacemaker-controld[2065]: notice: State transition S\_TRANSITION\_ENGINE -> S\_IDLE

Sep 08 15:54:58 <server>01.xxxx.org pacemaker-schedulerd[2064]: notice: Calculated transition 705, saving inputs in /var/lib/pacemaker/pengine/p>

lines 1-29/29 (END)

See logs for pacemaker with:

**journalctl -u pacemaker**

Pacemaker output of current status:  
crm\_mon -r -1

## IBM Questions

How can you stick a qmgr to a node? (No failback when node is healthy again.) (**rdqmadm -p -m <QmgrName> -d**)

## Troubleshooting section

IBM article on common errors:

<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=troubleshooting-example-rdqm-ha-configurations-errors>

## Patching Considerations

When OS Patching is performed, if new kernels are installed, a new kmod (Kernel modification) may need to be installed for DRBD.

Instructions can be found here to install before new kernel reboot:  
<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=aour-update-drbd-kernel-module-before-nodes-are-rebooted-into-new-kernel>

Instructions can be found here to install "AFTER" new kernel reboot:

<https://www.ibm.com/docs/en/ibm-mq/9.2?topic=aour-update-drbd-kernel-module-after-node-has-rebooted-into-new-kernel>

For mapping kmod-drbd to Kernel releases, here is a helpful table:

|  |  |  |  |
| --- | --- | --- | --- |
| RHEL Release | RHEL Kernel Release | RHEL Kernel Subrelease | RDQM Kernel Module |
| 7.6 | 3.10.0-957 |  | kmod-drbd-9.0.28\_3.10.0\_957-1 |
| 7.7 | 3.10.0-1062 |  | kmod-drbd-9.0.28\_3.10.0\_1062-1 |
| 7.8 | 3.10.0-1127 |  | kmod-drbd-9.0.28\_3.10.0\_1127-1 |
| 7.9 | 3.10.0-1160 |  | kmod-drbd-9.0.28\_3.10.0\_1160-1 |
| 8.2 | 4.18.0-193 |  | kmod-drbd-9.0.28\_4.18.0\_193-1 |
| 8.3 | 4.18.0-240 |  | kmod-drbd-9.0.28\_4.18.0\_240.1.1.el8\_3.x86\_64-1 |
| 8.4 | 4.18.0-305 |  | kmod-drbd-9.0.28\_4.18.0\_305-1 |

*From <*[*https://www.ibm.com/support/pages/ibm-mq-replicated-data-queue-manager-kernel-modules*](https://www.ibm.com/support/pages/ibm-mq-replicated-data-queue-manager-kernel-modules)*>*

## 14. BACKUP/RESTORE policy

Note: As per IBM’s recommendations XXX will setup ENTIRE FILE SYSTEM BACKUP/RESTORE on all the RDQM servers (not the current policy which has only the IMAGE LEVEL backup)

List of directories which needs to be backed up (during backup process)

**REBOOT strategy / steps :**

**Stop the Qmgr (on Primary RDQM node) to plan with UNIX team on reboot after below steps are completed :**

**Run the following command(s) to take entire backup of Qmgrs. For example (DEV env)**

sudo runmqras -qmlist DCHIPS00 -section defs,trace -caseno T08312022

sudo runmqras -qmlist DCHIPS01 -section defs,trace -caseno T08312022

sudo runmqras -qmlist DCHIPSIM00 -section defs,trace -caseno T08312022

sudo runmqras -qmlist DCHIPSIM01 -section defs,trace -caseno T08312022

**Also take an entire Qmgr config backup (including MQ objects). For example (DEV env)**

dmpmqcfg -m DCHIPS00 -x all > /home/mqm/backup/<server>01\_DCHIPS00\_09012022.mqsc

dmpmqcfg -m DCHIPSIM00 -x all > /home/mqm/backup/<server>01\_DCHIPSIM00\_09012022.mqsc

dmpmqcfg -m DCHIPS01 -x all > /home/mqm/backup/<server>01\_DCHIPS01\_09012022.mqsc

dmpmqcfg -m DCHIPSIM01 -x all > /home/mqm/backup/<server>01\_DCHIPSIM01\_09012022.mqsc

**Make a backup copy of :**

cd /home/mqm/backup

sudo tar -cvf pacemaker.tar  /var/lib/pacemaker  (Entire folder)

sudo tar -cvf drbd  /var/lib/drbd         (Entire folder)

sudo tar -cvf drbd.d.tar  /etc/drbd.d/         (Entire folder)

sudo tar -cvf usr\_bin.tar  /usr/bin             (Entire folder)

sudo tar -cvf vols.tar  /var/mqm/vols          (VIRTUAL folder for Qmgr(s))

sudo tar -cvf var\_mqm1.tar /var/mqm

sudo tar -cvf opt\_mqm.tar /opt/mqm

sudo tar -cvf var\_mqm2.tar /var/mqm Run this only after stopping QM

**\*Script to do a health-check on RDQM key files info during every midnight\***

(Venkat will explore if **Control-M** can be used here)

cib.xml

rdqm.ini

drbd process is up

crm status

pacemaker status

ping status between each nodes (connectivity check)

openssl

netstat check for port listening or not

sudo nmcli con show

**PING STATUS check on the 3 NIC Ips** (For e.g DEV) :

ping -I ensXXX 169.254.1.20

ping -I ensXXX 169.254.1.21

ping -I ensXXX 169.254.1.22

**MAINTENANCE Steps :**

**Once all the 3 nodes are rebooted and Unix team has done their server health check, we can start Qmgr on Primary node :**

strmqm <QMGR\_NAME>

Check the status :

rdqmstatus -m <QMGR\_NAME>

drbdadm status

crm status