RED HAT MAGAZINE

Go
RSS OGG MP3
About
Editorial team
Letter to the editors
Call for writers
Rate this page del.icio.us Digg slashdot StumbleUpon

Enhancing cluster quorum with QDisk

by Rob Kenna

QDisk bolsters the quorum of small count clusters. This article outlines when and how to use QDisk.

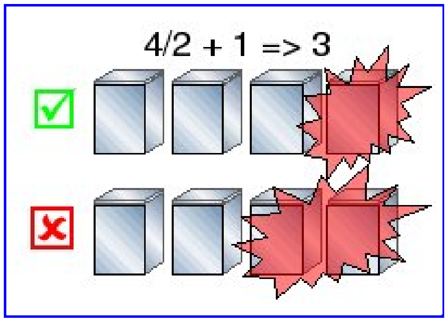
Note: QDisk was added to RHCS and GFS starting with the updates in Red Hat Enterprise Linux 4.4 and 5.0.

The problem: Multiple failures hammer your cluster

You have set up your web service environment. Red Hat Clustering has provided both the ability to scale performance across several machines *and* protect against a machine failure by automatically restarting the application on another node in the cluster. But sometimes things get really bad and a two-machine failure can still disable a four-node cluster. Well, that's inconvenient! But there is a solution. This article explores how to use the QDisk facility to take advantage of shared storage to keep your applications running.

Quorum, quorate, QDisk: What is this, a Latin lesson?

So clustering is all about "ganging up" on a problem–it's based on strength in numbers. It's a democratic process and at times requires voting to decide on future actions; like rebooting a hung machine. As with any good democracy, a simple plurality is required: A count of just over half the nodes in the cluster is needed to establish a quorum. Thus a three-node cluster needs two active nodes to function. A six-node cluster needs four nodes and so on. In general, you need (n/2 + 1) of n nodes to establish a quorate state.

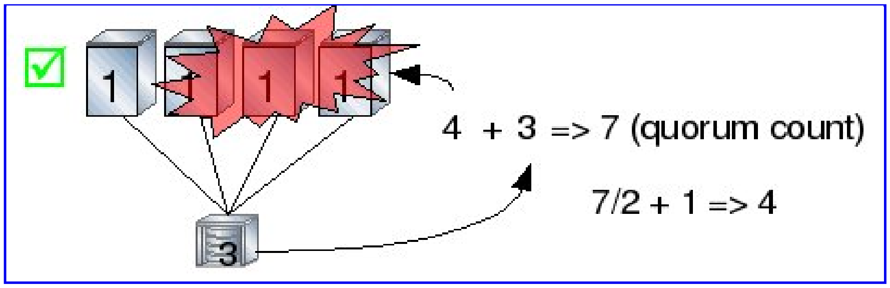


Red Hat Cluster Suite in Enterprise Linux 2.1 used a shared disk area to coordinate the state of the cluster with IP networking as a backup. Clustering in Enterprise Linux 3 switched this and used IP networking as primary with shared disk area as a backup. In Enterprise Linux 4 and 5, Cluster Suite joined GFS (Global File System) using a network (IP) based mechanism, replacing the need for a quorum partition. This was done to remove the requirement for SAN (Storage Area Network) storage. Plus, IP-based quorum systems scale especially well as you move to large node counts (greater than 16)

Even when the node count is reasonably modest, say 9 nodes, the chance of losing enough nodes to break quorum is small. For the 9 node case, up to 4 nodes can fail before quorum is lost. With a 20 node cluster, you'd need to lose 10 nodes; not a likely occurrence. But in the 3 or 4 node case, losing only 2 nodes becomes a problem. You could add spare machines to strengthen the quorum count, but that's likely overkill especially if you have SAN storage in your setup–an increasingly common case. Instead, use a small amount of shared storage to bolster quorum.

Quorum Disk (QDisk): An oldie but goodie

To solve the small cluster quorum problem it was decided to not reinvent the wheel, but rather reuse an old one with some mileage (and experience) when applied to the problem at hand. QDisk is based on the quorum mechanism found in the clustering of Red Hat Enterprise Linux 3. But for those in the know, there are some differences. Here's how it works.



QDisk uses a small 10MB disk partition shared across the cluster. Qdiskd runs on each node in the cluster, periodically evaluating its own health and then placing its state information into an assigned portion of the shared disk area. Each qdiskd then looks at the state of the other nodes in the cluster as posted in their area of the QDisk partition. When in a healthy state, the quorum of the cluster adds the count for each node plus the value of the QDisk partition. In the example above, the total quorum count is 7; one for each node and 3 for the QDisk partition.

If, on a particular node, QDisk is unable to access its shared disk area after several attempts, then the qdiskd running on another node in the cluster will request that the troubled node be fenced. This will reset that machine and get it into a operational state.

Heuristics for added assurance

As an option, one or more heuristics can be added to the qdisk configuration. Heuristics are tests run prior to accessing the qdisk partition. Typical tests include the ability to access network routers. These are sanity checks for the machine. If the heuristic tests fail, then QDisk will–by default–reboot the node in a attempt to place the machine in a better state. The value of minimum_score (see conga screen below) indicates the number of heuristics which are required to succeed.

Using DLM and QDisk with cluster databases (e.g. Oracle RAC)

Let’s move to a concrete use case for QDisk and show how to set things up. The Distributed Lock Manager (DLM) has recently been validated for use in conjunction with Oracle RAC (clustered database) 10gR2. Using DLM eliminates the need and expense of the three extra machines that were required to run the older GULM lock manager. QDisk ensures that all but one of the nodes in the cluster can fail while RAC continues to run.

Using DLM in place of GULM is actually quite simple. (Refer to Oracle Real Application Clusters GFS: Oracle RAC GFS, an instruction guide for Oracle RAC with GFS.) First, request that DLM be used in place of GULM. This is the default when creating a new cluster. Since DLM is an embedded lock manager, no further configuration is required for DLM.

Now, you need to configure the use of QDisk. Referencing the picture above, set up a four node cluster. This means that the total quorum count will be seven; one for each node (4), plus three for the quorum partition. The partition should be a raw partition and not managed by CLVM. Most sites will set up multipathing to the partition. For this, refer to the Using Device-Mapper Multipath. It is also recommended that either Fibre Channel or iSCSI array storage be used, like what would be used for the Oracle files.

Initializing the QDisk

Setting up a Quorum disk is fairly easy. First, create the shared quorum partition. Second, set the configuration of the cluster. This example uses an 11MB partition, /dev/sdi1, as can be seen by cat’ing /proc/partitions or viewing with parted. A raw partition of 10MB is the recommended size.

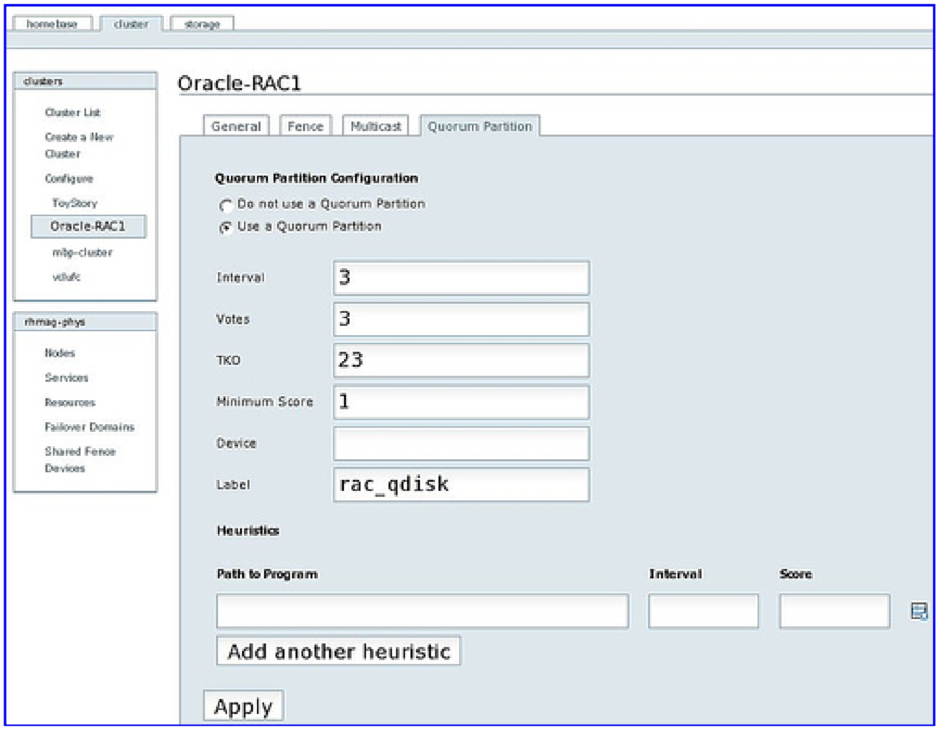
```
[root@et-virt09 ~]# cat /proc/partitions
major minor  #blocks  name
   8     0   71687325 sda
   8     1     104391 sda1
   8     2   71577607 sda2
        :       :
   8   128   55587840 sdi
   8   129      11248 sdi1
   8   130   55576576 sdi2
   8   144   53483520 sdj
   8   145   53483504 sdj1
[root@et-virt05 ~]# parted /dev/sdi
GNU Parted 1.8.1
Using /dev/sdi
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p

Model: EQLOGIC 100E-00 (scsi)
Disk /dev/sdi: 56.9GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number  Start   End     Size    Type     File system  Flags
 1      16.4kB  11.5MB  11.5MB  primary
 2      11.5MB  56.9GB  56.9GB  primary
```

Next the mkqdisk command is used to prepare the quorum partition. This will initialize 16 regions–the maximum size cluster allowed to use QDisk. Beyond that the IO can start to slow operations down. The IP quorum mechanism is more than adequate for that size of a cluster. Simply specify the device and a unique label. The label will then be referenced in the cluster.conf file. The result can the be checked with “mkqdisk -L”.

```
[root@et-virt08 ~]# mkqdisk -c /dev/sdi1 -l rac_qdisk
mkqdisk v0.5.1
Writing new quorum disk label 'rac_qdisk' to /dev/sdi1.
WARNING: About to destroy all data on /dev/sdi1; proceed [N/y] ? y
Initializing status block for node 1...
Initializing status block for node 2...
        :       :       :
Initializing status block for node 16...
[root@et-virt08 ~]# mkqdisk -L
mkqdisk v0.5.1
/dev/sdi1:
        Magic:   eb7a62c2
        Label:   rac_qdisk
        Created: Thu Dec  6 14:40:07 2007
        Host:    et-virt08.lab.boston.redhat.com
```

The cluster configuration

You can configure the use of QDisk via Conga as follows:



Oracle-RAC1

Quorum Partition Configuration

○ Do not use a Quorum Partition
● Use a Quorum Partition

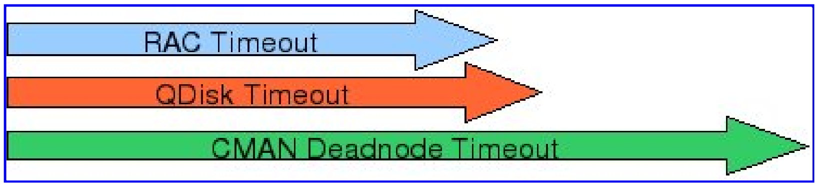
Interval: 3
Votes: 3
TKO: 23
Minimum Score: 1
Device:
Label: rac_qdisk

Heuristics

Path to Program	Interval	Score

A value of 3 votes is assigned to the quorum partition. This value is one less than the number of machines in the cluster. This means that only one node plus the quorum partition is required to hold quorum (4 out of 7.) Also note that the label on the quorum partition is referenced, not the device name. Device names can move on reboot, but the label will stay put. Note that in this example, no heuristic was specified and that the associated minimum score defaults to 1.

More importantly, note that QDisk will perform its evaluation every 3 seconds. There will be a TKO (Technical Knock Out) after 23 failures or a total of 69 seconds. This means that if a node cannot connect to its qdisk area, it will be reported as having failed on the quorum disk and will be fenced. This was done to allow time for Oracle RAC to take action first in the event of a node failure. In this setup, RAC timeout was set to 60 seconds. RAC uses “self fencing” which may not fire if the node is stuck. In this instance, Red Hat cluster fencing will subsequently fire and force the reboot of the failed machine.



Likewise, the cman_deadnode_timeout is set to 135 seconds. This is 1.5 times as long as the qdisk timeout to allow for the case of the failed node being the qdisk master. After 135 seconds, cman will then issue a fence operation to the failed node and reboot it. For non-RAC setups, the dead node timeout value need not be altered. But for this case you will need to directly change the cluster.conf file and propagate with ccs_tool. If a shorter timeout is desired, shorten the other parameters as well.

A third thing to note is that expected_nodes=”7″. This represents an augmented quorum count including the 3 provided by the qdisk partition.

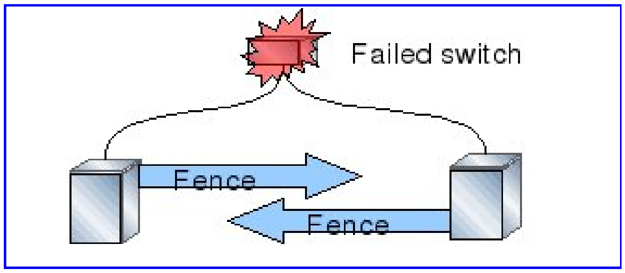
```
<?xml version="1.0"?>
<cluster alias="Oracle-RAC1" config_version="35" name="Oracle-RAC1">
        <quorumd interval="3" tko="23" label="rac_qdisk" votes="3"/>
        <cman deadnode_timeout="135"  expected_nodes="7"/>
        <clusternodes>
                <clusternode name="et-virt08.lab.boston.redhat.com" nodeid="1" votes="1">
                        <multicast addr="225.0.0.12" interface="eth1"/>
                        <fence>
                                <method name="1">
                                        <device name="wti_fence" port="12"/>
                                </method>
                        </fence>
                </clusternode>
                        :
                        :
        </clusternodes>
        <fencedevices>
                <fencedevice agent="fence_wti" ipaddr="192.168.77.128" name="wti_fence" passwd="password"/>
        </fencedevices>
         <rm log_facility="local4" log_level="6"/>
        <fence_daemon clean_start="0" post_fail_delay="0" post_join_delay="3"/>
</cluster>
```

Editing and updating the cluster.conf file directly

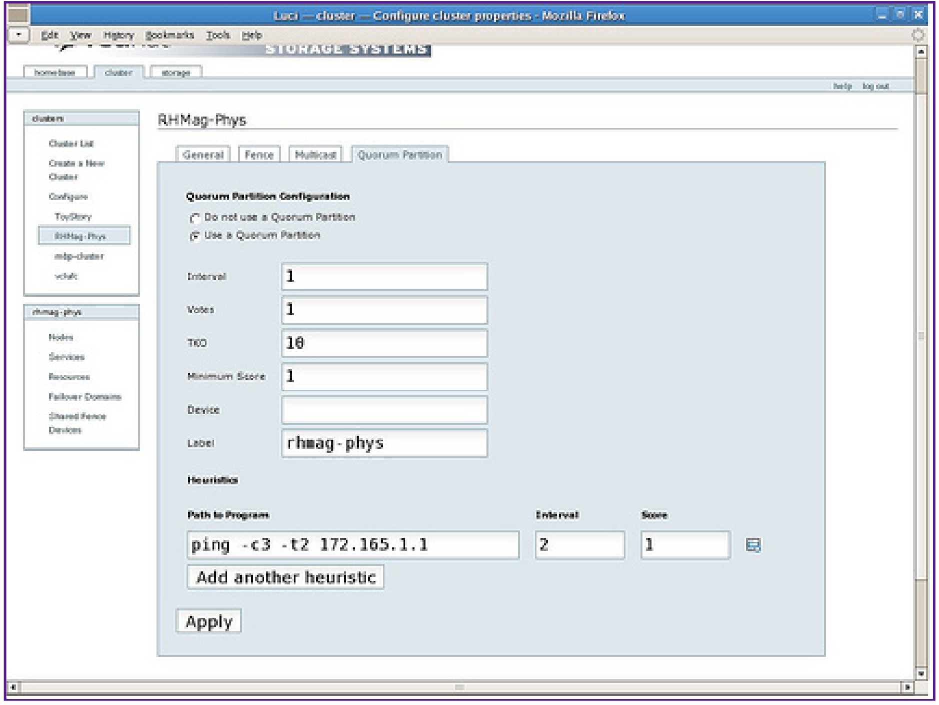
As in this example, it is sometimes required to directly edit and propagate the cluster.conf file across the cluster. This is done by creating a copy of the cluster.conf with the required changes on one of the nodes and then by running the css_tool with the update command. Remember to increment the version when editing the file. For example:

```
[root@et-virt05 cluster]# cp cluster.conf cluster.conf.new
[root@et-virt05 cluster]# vi cluster.conf.new
[root@et-virt05 cluster]# ccs_tool update cluster.conf.new
Config file updated from version 48 to 49
Update complete.
```

Shootout at the OK Corral



Two node clusters are common, but quorum is handled as a special case in that (n/2 + 1), where n equals 2, is still 2. Obviously quorum still needs to be maintained when one of the two nodes dies. As a special case, if one node fails the other simply takes over by itself. However, there is one edge condition to be aware of. When there is a network break between the two nodes, they both will believe the other has failed. This is called “split brain.” Each node assumes that they now own the cluster and will attempt to fence the other; typically via a reboot. While a rare case, it is still possible. QDisk can be used to bolster the quorum, thus avoiding this condition. Additionally, a heuristic can be added to evaluate a node’s network connectivity and remove itself from the cluster. The typical heuristic would be a ping to a network router. Here is an example using Conga to set this up.



RHMag-Phys

Quorum Partition Configuration

○ Do not use a Quorum Partition
● Use a Quorum Partition

Interval	1
Votes	1
TKO	10
Minimum Score	1
Device	
Label	rhmag-phys

Heuristics

Path to Program	Interval	Score
ping -c3 -t2 172.165.1.1	2	1

Note: To configure a two node cluster without QDisk, set the value of two_node=”1″ and expected_votes=”1″ in the cluster.conf. As in:

```
<cman expected_votes="1" two_node="1"/>
```

With QDisk, though, the the expected_count should be “3″ and two_node “0″ (or simply remove two_node.)

```
<cman expected_votes="3" />
```

Wrap up

Clustering is a powerful technology providing scale-out performance as well as high availability. The optional use of QDisk provides even more robust protection for smaller configurations with shared storage. The solutions outlined for Oracle RAC and two-node clusters are key examples of how this mechanism can keep your services running in the face of multiple failures.

Other articles in the Red Hat Magazine cluster series

The reader may also be interested in other articles on clustering in Red Hat Magazine. Expected future articles will expand on the use of clustering with virtual machines.

- Red Hat Magazine Article – Introduction to Conga
- Automated failover and recovery of virtualized guests in Advanced Platform

References

- Man pages for qdisk, mkqdisk
- Red Hat Enterprise Linux Advanced Platform
- Red Hat Enterprise Linux Global File System (GFS)
- Conga Project Page
- Red Hat Enterprise Linux Cluster Suite
- Oracle Real Application Clusters GFS: Oracle RAC GFS (html) or (pdf)
- Using Device-Mapper Multipath (html) or (pdf)

Acknowledgments

I'd like to tip my hat to the hard working teams at Red Hat and the open source community. Thanks especially to Lon Hohberger, Tom Tracy, Jeff Needham and others who have provided invaluable review comments.

About the author

Rob Kenna is Senior Product Manager for Red Hat's Storage and Clustering Products, including GFS (cluster file system), and RHCS (application failover). He brings a rich background as developer and manager for the creation of storage software.

This entry was posted by The editorial team on Wednesday, December 19th, 2007 at 1:17 pm and is filed under technical. You can follow any responses to this entry through the RSS 2.0 feed. Both comments and pings are currently closed.

3 responses to "Enhancing cluster quorum with QDisk"

1. *Tenyo Grozev* says:
   December 26th, 2007 at 3:11 pm

   Excellent article on quorum/qdisk! It's very hard to find information about it. We've been using it to resolve split-brain in two of our 2-node clusters since it came out in 4.4 and we're currently on 4.5. It worked around the split-brain perfectly, but introduced a new issue. Every time there's a delay on the SAN (caused, for example, by controller fail over or something else that doesn't affect any of the other servers using the SAN), qdisk cannot complete in time and the quorum of the entire cluster gets dissolved. The TKO parameter actually doesn't affect the qdisk write intervals but only the heuristics. As soon as qdisk cannot write to the quorum partition (2 seconds by default), it fails and there's no way to work around that.

2. *Michael Hagmann* says:

December 27th, 2007 at 4:07 pm

Hi Rob

Sounds great. You wrote:
"The Distributed Lock Manager (DLM) has recently been validated for use in conjunction with Oracle RAC (clustered database) 10gR2." Please can you post the Oracle Metalink Number where I can find the Certification. The Oracle Note:329530.1 is very confusing me, there is a Certification for GFS6.0 with DLM and also explicit they don't allow DLM "Embedded lock server configurations are NOT supported by Oracle for use with Oracle RAC"

What's true now?

thx mike

3. *Rob Kenna* says:
January 3rd, 2008 at 3:44 pm

Hi Mike -

The metalink note has 2 small mistakes which we will ask to be fixed. GFS 6.1 for Red Hat Enterprise Linux 4 should have been referenced (not GFS 6.0) and the statement that DLM "Embedded lock server configurations are NOT supported" currently only applies to 64 bit systems. This is a leftover from before the recent testing.

- Rob

- Truth Happens
- Red Hat People
- Red Hat Press
- redhat.com
- JBoss.com
- jboss.org

Links

- Creative Commons
- One Laptop Per Child
- The WordPress Project

Tags

- contests (2)
- culture (58)

- design (18)
- Dev Fu (83)
- documentation (18)
- education (15)
- events (68)
- Fedora (87)
- from the editors (56)
- JBoss (63)
- multimedia (67)
- music (11)
- One Laptop per Child (14)
- Red Hat Enterprise Linux (54)
- review (26)
- Shadowman (3)
- technical (113)
- tips and tricks (142)
- truth (29)
- Uncategorized (12)

Archives

- January 2010
- September 2009
- May 2009
- April 2009
- March 2009
- February 2009

Entries (RSS) and Comments (RSS).
Red Hat Magazine is powered by Blog at WordPress.com.

ISSN 1933-7973