

Badlands Disk aka StorNext Q-Series Storage:

Preliminary Performance Evaluation

John H. Badger March 2012

# Background

Quantum has decided to OEM two RAID disk array controllers from NetApp/LSI. The first is a 4U high, 60 drive chassis to be sold by Quantum as the QD6000. The second is a 2U high, 12 drive chassis to be sold by Quantum as the QS1200 or QM1200. Expansion chassis are available for both. These RAID disk array controllers are known internally as “The Badlands Disks”.

The goal of this testing is to:

* Determine the performance limitations of the array controllers in the form of data throughput under ideal conditions.
* Evaluate some of the variables that can result in performance degradation.
* Be a learning exercise in how the StorNext file system performs using these disk arrays as data devices, how variations in configurations affect performance.
* Develop some guidelines for Quantum service personnel and customers on how to configure StorNext to run on these devices.

This testing exercise is not intended to be a complete performance evaluation of the Q-series disk arrays. The concentration is on the maximum data bandwidth one might achieve when reading and writing video frames. The performance of this type of application is fairly well understood, easy to emulate and is important to a significant number of StorNext customers. The StorNext test organization will run a more thorough test suite including transactional testing and StorNext metadata testing.

The number of variables in the I/O path from the application to the storage is truly staggering. This means there are nearly endless combinations of parameters to test. A lot of data was collected, and some of the more interesting results are shown here.

# Test configuration

* A single Dell R710 client with four QLogic ISP2532 based dual port 8 Gbit Fibre Channel cards. This is 8 x 8 Gbit ports for a maximum theoretical Fibre Channel bandwidth of 6.8 GBytes per second.
* No Fibre Channel switch was used. Additional evaluation with a switch may be done at some future date.
* A QD6000 Pikes Peak base array controller with 60 x 3 TByte 7.5K NL-SAS drives. A total of 8 x 8 Gbit FC host ports, four per controller.
* A QD1200 Snowmass base array controller with four expansion chassis for a total of 60 x 600 MByte 15K enterprise SAS drives. A total of 16 x 8 Gbit FC host ports, eight per controller. Only four host ports per controller were used.
* Arrays were configured with Quantum released Q-series standard platform RAID configurations. The only deviation was to disable write cache mirroring.
  + QD6000 - 6 x 8+2 RAID 6 LUNs.
  + QD1200 – 10 x 4+2 RAID 6 LUNs.
* StorNext 4.3 client and MDC running on Linux - SLES 11.1.
* Test was Mio. A multi-stream asynchronous I/O generator.
* Each data stream did 12 MByte sequential requests queued 8 deep.

# QD6000

## Test Configuration

* StorNext file was configured with a single stripe group with six LUNs (1 x 6).
* The default RAID segment size on the QD6000 is 128K over 8 data drives per LUN. That gives us a StorNext stripebreadth 1 MByte.
* StorNext trunk between 4.2 and 4.3.
* ASR was on and set to 1 GByte.
* Write testing was done with write cache mirroring both on and off.
* The mpp/RDAC multipath driver was used

Additional test data was gathered for StorNext stripegroup x LUN configurations of (6 x 1), (2 x 3), and (3 x 2). Previous experience has shown that more LUNs and fewer stripe groups prove better for this type of application and that was verified here. Supplemental data confirming this is available.

Additional testing was performed on the raw QD6000 block devices. That data is also available.

## QD6000 reads

NetApp claims 6 Gbytes/second read bandwidth for the Pikes Peak controller. I was able to get to around 5.5 Gbytes/second for big reads. See Figure 1.

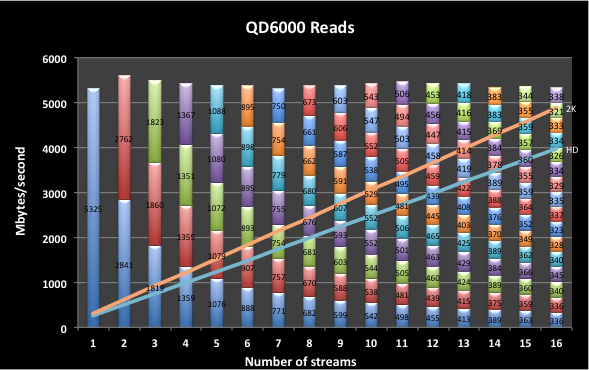


Figure 1 QD6000 maximum read rate

In my test configuration, QD6000 read performance was limited by the client. The Dell R710 client I was using has a maximum aggregate unidirectional path bandwidth from I/O to memory of about 5.7 Gbytes/second. I proved this by connecting two QD6000 arrays to my client. The combined rate did not change. I suspect with a more capable client, the QD6000 could achieve the 6 Gbytes/second rate claimed by NetApp/LSI.

The other item of note here is how well StorNext allocates and schedules I/O requests from the 16 streams across the LUNs in the stripegroup. Albeit, this is on a relatively new yet to be fragmented file system, the service delivered to the applications shows a nearly perfect distribution of capabilities.

The reference lines on these charts represent the data rate needed to stream HD video at 30 frames/second and full aperture 2K video at 24 frames/second.

## QD6000 writes

The QD6000 suffers from the same inadequacies that seem to plague nearly all dual A/B controller arrays; decreased write throughput when *write cache mirroring* is enabled. See Figure 2. Write cache mirroring makes a copy of all write data to the *other* controller via an inter-controller channel. In the event that a controller crashes, the data is available in the *other* controller and can be spooled to storage.

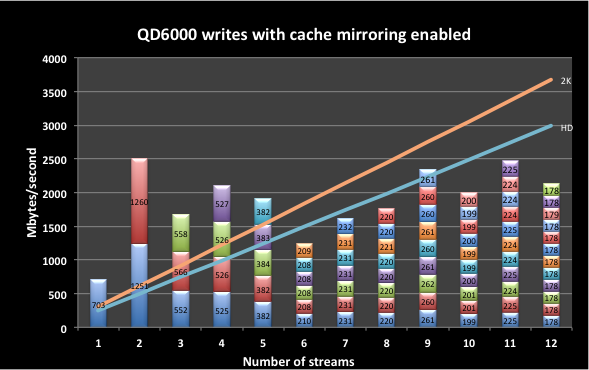


Figure 2 QD6000 writes with cache mirroring enabled

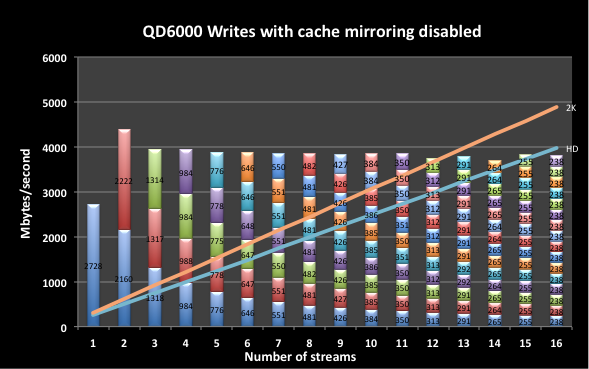


Figure 3 QD6000 writes with cache mirroring disabled.

NetApp/LSI claims the Pikes Peak controller has write bandwidth of 3.1 Gbytes/second with cache mirroring enabled. See Figure 2. I found that the latencies introduced when write cache mirroring is enabled, interferes with stream multiplexing enough to cause degraded and inconsistent write performance. These measurements were done using the mpp/RDAC driver. We’ll look at the effect that the multipath drivers have on performance later in this paper.

Figure 3 shows QD6000 writes with cache mirroring disabled. NetApp/LSI claims the Pikes Peak controller has a write bandwidth of 3.6 Gbytes/second with write cache mirroring disabled. I actually measured slightly better. The rates seem to be much more consistent across the number of streams with cache mirroring disabled.

Customers that do not have a requirement for *extreme* reliability should probably run with cache mirroring disabled for big data applications. Currently, cache mirroring is enabled by default on the QD6000 by the platform initialization software. We include Santricity (LSI array monitoring and maintenance software package) with each QD6000, allowing Quantum support or customers to disable write cache mirroring.

# QS1200

## Test Configuration

* StorNext file was configured with a single stripe group with eight LUNs (1 x 8).
* The default RAID segment size on the QS1200 is 128K over 4 data drives per LUN. That gives us a StorNext stripebreadth 512 KBytes.
* StorNext trunk between 4.2 and 4.3.
* ASR was on and set to 1 GByte.
* Write testing was done with write cache mirroring both on and off.
* The mpp/RDAC multipath driver was used.

The QS1200 units we received came with 4 x 12-drive expansion trays for a total of 60 enterprise quality SAS drives. Configured as 4+2 RAID 6 LUNs, this is a total of ten LUNs. My testing revealed that 48 drives configured as a total of 8 x 4+2 LUNs was enough SAS drives to max out the Snowmass controller. My tests saw no significant difference in performance between the eight and ten LUN configurations. The test results presented here are for a StorNext file system with one stripe group comprised of eight LUNs.

Additional testing was performed on the raw QS1200 block devices. That data is also available.

## QS1200 Reads

NetApp claims 5 Gbytes/second read bandwidth for the Pikes Peak controller. I was able to get between 4.5 – 4.8 Gbytes/second for big reads. See Figure 4 below. The QS1200 delivers very consistent read rates across a number of streams. Because of the lower bandwidth of the QS1200 compared to the QD6000, there was no limitation imposed by the I/O path to memory on the client.

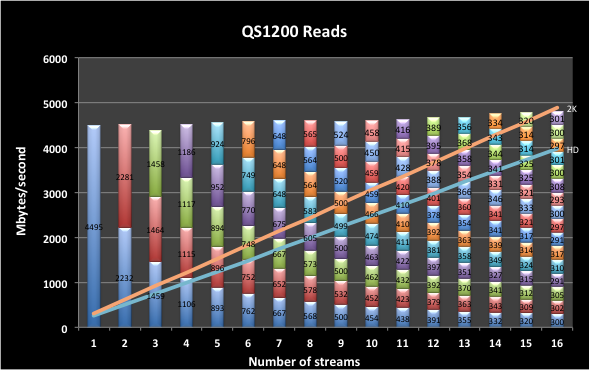


Figure QS1200 Reads

## QS1200 Writes

NetApp/LSI states that the expected maximum write rate for SAS drives via the Snowmass controller with cache mirroring enabled is 1350 Mbytes/second. While inconsistent, Figure 5 shows I was able to achieve better than this for all but a single stream.

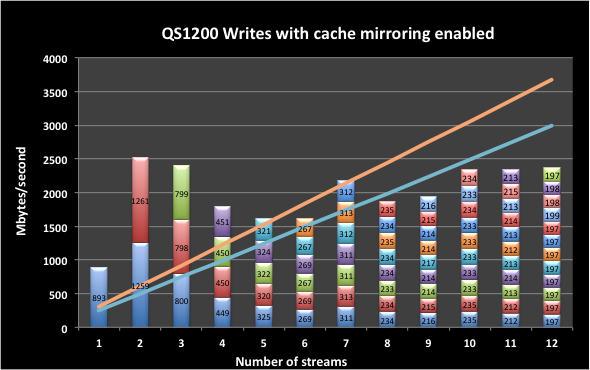


Figure QS1200 writes with cache mirroring enabled.

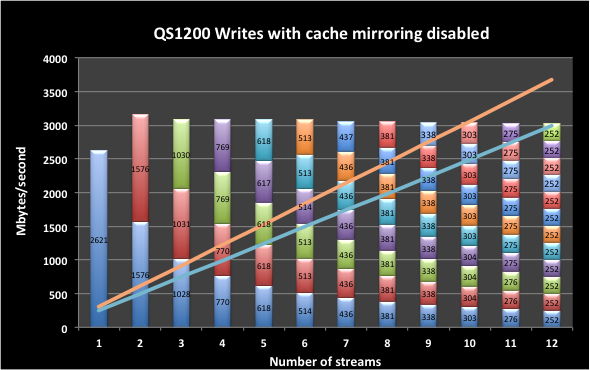


Figure QS1200 Writes with cache mirroring disabled.

As with the QD6000, much better and more consistent rates can be achieved by disabling write cache mirroring. See Figure 6. NetApp/LSI projected a rate with write cache mirroring disabled of 2.1 Gbytes/second. I was able to achieve slightly better than 3 Gbytes/second that was consistent over a range of streams.

# Multipath Driver Comparison

The purpose of a multipath driver is two fold:

1. Provide for path failover in the even that a path or controller failure.
2. To do I/O request scheduling for improved performance.

For Linux, there are two multipath drivers generally available:

The Device Mapper Multipath driver (DMM) is included in the major Linux distributions. It is a meta-device driver that manages the underlying physical block devices, scheduling I/O requests, monitoring path availability, and managing failover.

The mpp/RDAC driver (MPP) was originally written by LSI and is now maintained and distributed by NetApp/LSI. The MPP driver has an upper component that monitors paths and manages failover, and a lower level component that multiplexes a single block device across multiple paths.

The testing presented thus far was done using the MPP driver. This is because the MPP driver seems to have a significant performance advantage over the DMM driver where there are multiple active paths to a LUN; at least for *big* I/O’s. For comparison purposes, I did a couple of test runs on the QS1200 that shows the differences. See Figure 7.

For I/O’s, the MPP/RDAC driver has approximately a 50% bandwidth advantage over the DM/Multipath driver. For *big* writes, the advantage closes a bit as more streams are added, but is still significant.

While we still have to test the failover component of the multipath drivers, the recommended driver to use with a Linux client, is the MPP/RDAC driver.

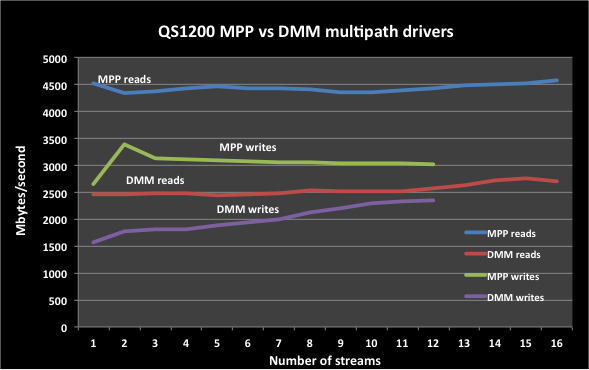


Figure MPP vs. DMM multipath driver performance comparison.

# Stripebreadth

The amount of data that StorNext writes contiguously to a LUN in a stripegroup before advancing to the next LUN is known in StorNext terminology as the *stripebreadth*. It is suggested that the stripebreadth be set to the *RAID segment size* x the number of data drives in the RAID volume (LUN).

The QS1200 has LUNs configured in 4+2 RAID 6. The platform initialization scripts set the RAID segment size to 128 KBytes. With four data devices, this gives us an *optimum* stripebreadth of 512 KBytes. Out of curiosity, I also tested a 1 MByte and 2 MByte stripebreadths on the QS1200. Figure 8 shows writes on 512K and 1M file system. The 2 MByte write is not shown, but is nearly identical. **Error! Reference source not found.**Figure 9 shows reads for 512K, 1M and 2M stripebreadth file system.

For *big* I/O’s, it seems that the stripebreadth in *nice* multiples of 512K doesn’t have much of an affect for the QS1200.

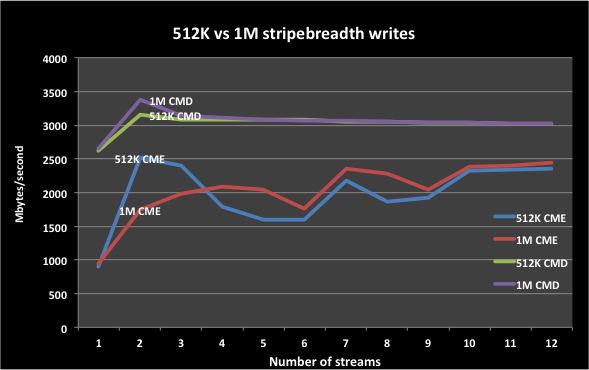


Figure Writes on a 512K and 1M stripebreadth file system

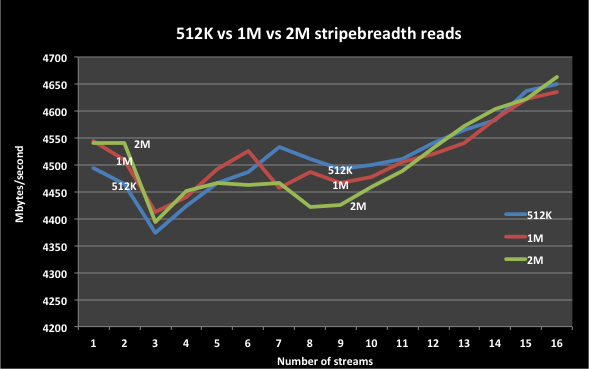


Figure Reads on 512K, 1M, and 2M stripebreadth file systems

# Block sizes

The following charts plot read and write operations over a range of block sizes. Each test was running 16 streams in power-of-two block sizes from 4 KBytes to 16 MBytes.

## QD6000

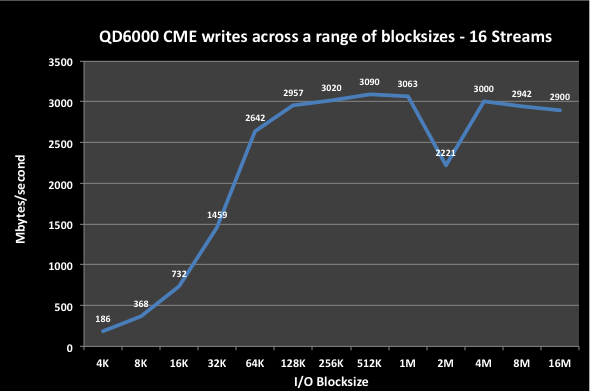


Figure QD6000 writes for various block sizes with cache mirroring enabled

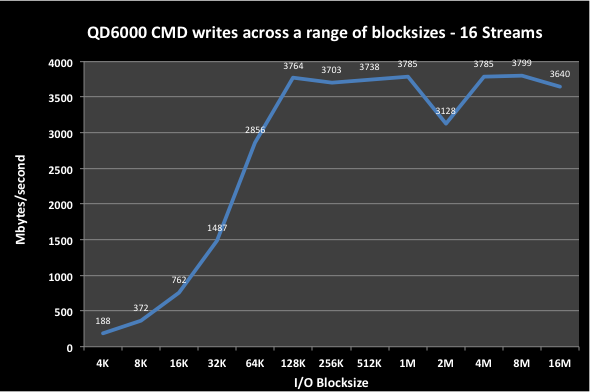


Figure QD6000 writes for various block sizes with cache mirroring disabled

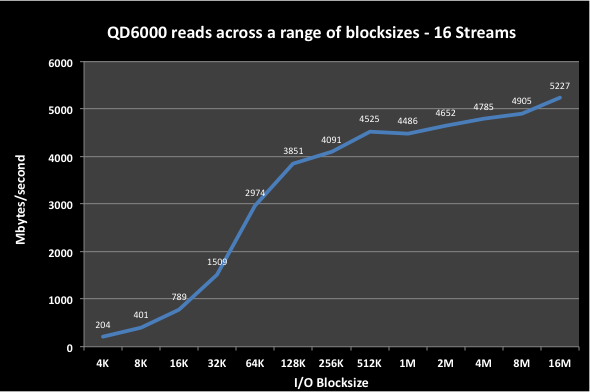


Figure QD6000 reads for various block sizes

## QS1200

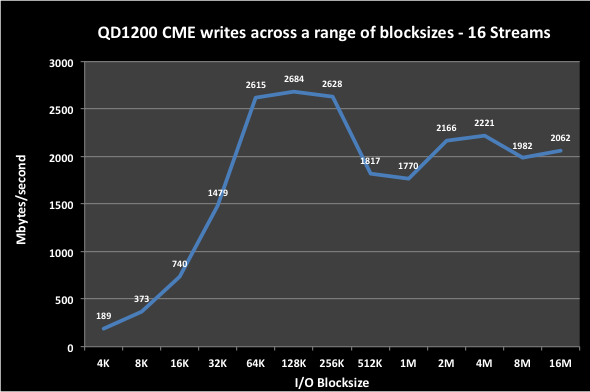


Figure QS1200 writes across a range of block sizes with cache mirroring enabled

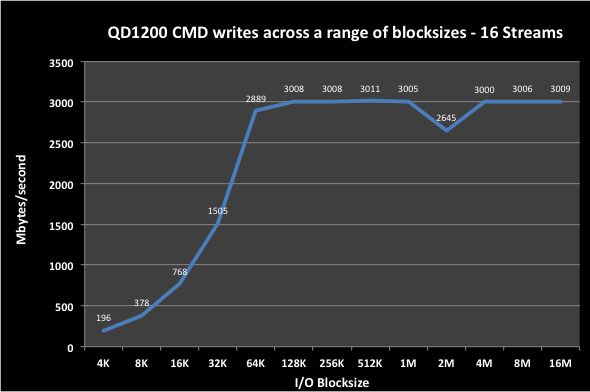


Figure QS1200 writes across a range of block sizes with cache mirroring disabled.

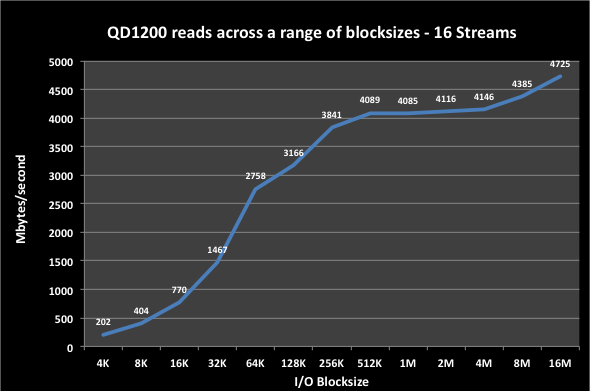


Figure QS1200 reads across a range of block sizes.