Does Performance Through Failure Matter to You?

As we've built the Pure Storage FlashArray, one of the key features that customers have let us know is critical to them is performance through failure. Many traditional disk arrays have performance challenges to begin with – but the situation worsens deeply when a disk or controller failure occurs (whether unplanned, or much more frequently planned during maintenance/upgrade events). As we move into the the flash era, flash arrays are performance arrays, and "failure" has to be re-defined with a broader definition: availability loss *OR* performance loss. It's simply not OK to lose connectivity to storage *OR* performance of storage – ever.

The folks over at XtremIO have been busy this holiday season, penning a nearly 2,000-word blog to make the argument for their scale-out architecture vs. dual-controller architectures. Feel free to read their entire blog, but for the time-crunched reader I'll do my best to summarize both of our points-of-view, I hope fairly:

Performance level, 100 percent performance

XtremIO's Performance position:

- XtremIO is a scale-out architecture, scalable in 1, 2, or 4 dual-controller "bricks" (i.e. 2, 4, or 8 controllers)

- Since all controllers in all nodes are active, if a controller fails the performance impact is proportional to the number of controllers (i.e. 1/2, 1/4, or 1/8th)
- Scale-out enables XtremIO to use all controllers for performance all the time
- Failures almost never happen (0.001% of the time), so it is more important to leverage all performance of all controllers during the 99.999% of the time that failures don't happen than to design to eliminate the performance hit during the 0.001% of the time that failures do happen

Pure's Performance position:

- The Pure Storage FlashArray is an Active/Active dual-controller array, meaning that all volumes are available on all ports of both controllers at all times – our default multi-pathing policy is round-robin across both controllers. *But*, the performance of the FlashArray is limited to the performance that can be delivered by a single controller, so that the FlashArray can maintain full performance through failure and upgrades.
- Maintaining performance through failure is Job 1: all-flash arrays are deployed in some of the most mission-critical workloads in the world, and taking a 12.5%, 25%, or 50% performance hit during failures is simply not an option – it can result in significant application response issues or downtime.
- Beyond unplanned downtime (which we agree is 0.001% or less for any 5-9s array), users experience a much larger amount of planned downtime: array software updates, capacity upgrades, controller upgrades, etc.... and enabling all of these upgrades and maintenance events to happen without downtime *or* performance impact is critical for the storage team to do their job without impacting the applications team (i.e. never having to ask for an outage window). We call this non-disruptive operations.

So now that you understand both of our visions, I invite you to review and critique which of us is doing the best job on *delivering on our visions* for HA and non-disruptive operations, by reviewing our respective HA demonstrations.

XtremIO HA Demonstration:

This YouTube video (XtremIO High Availability Deep Dive) was released by XtremIO as part of the recent GA launch. If you just want to watch the demo part, it starts at about 15 minutes in. As you can see, the *actual* XtremIO HA experience differs quite a bit from the HA vision from by Robin Ren in the blog:

- A 2-brick (4-controller) XtremIO array is showcased
- According to the blog, this array should have a 1/4 or 25% performance impact with one controller failure
- Workload running at 200K IOPS and ~1ms average latency is started
- Power is pulled on controller at 5:07 (time in GUI)
- IO pauses completely for 10 seconds
- IO then restarts at well less than 50K IOPS at around 5:08 (>75% performance reduction)
- From 5:08 to roughly 5:18 The cluster then loads metadata on the remaining nodes, and the cluster ramps over 10 minutes time to a "degraded steady state" which for this workload is roughly 100K IOPS, or 50% performance impact from losing 1 of 4 controllers

Now let's look at the process of controller re-introduction:

- When the controller is re-introduced into the cluster IO pauses completely, again for 10 seconds (at roughly 5:37)

- Less than a minute later there is another 10 second IO pause (at roughly 5:38)
- After more than 5 minutes (from approximately 5:39 to 5:45), performance steadily recovers from the “degraded steady state” as the recovered controller is fully balanced into the system

So in summary:

- Performance impact is 50% when one controller of four is lost
- Performance impact initially is >75%, and only recovers to the “degraded steady state” of 50% after a 10-minute recovery period
- A similar ramp back to full performance happens when the controller is re-introduced

Now let’s compare with Pure Storage…

Pure Storage HA / Non-Disruptive Operations Demo:

The video details a similar HA/NDO scenario with Pure Storage:

- A two-controller FA-420 system is demonstrated
- A 200K IOPS workload with roughly .5 ms latency is started
- A controller is failed via a hard power pull
- The array experiences a very short (<5 second) IO pause while all IO is taken over by the remaining controller
- Performance continues immediately at the full 200K IOPS / .5 ms latency
- Zero performance loss, zero “warm up time”

And now let’s look at controller re-introduction:

- The controller is re-introduced into the array
- Performance resumes at the full 200K IOPS / .5 ms latency immediately

In Conclusion: Make HA Testing a Key Part of your PoC

As you can see, HA and non-disruptive operations experiences can vary widely. The best advice we can (and do) give customers is to do full and exhaustive HA/NDO and upgrade testing during EVERY PoC. Specifically:

- Generate a load as close to representative as possible to your production load
- Fail a controller and see what happens
- Fail at least two drives and see what happens
- Upgrade the array software between firmware revisions and see what happens
- Perform a hard, full power failure to the entire array and see what happens when you restore power

In each of these tests, watch:

- What happens to performance from the array's point-of-view, and your application's point-of-view
- How the array heals around and recovers from the failure
- How long a recovery to full performance takes, and how the application is impacted during that period
- How the vendor's support team handles the failure (remember, you are PoCing the support experience as well as product!). How long does it take them to call and see what's wrong?

Want to give this a whirl on a Pure Storage FlashArray in your environment? Drop us a line.