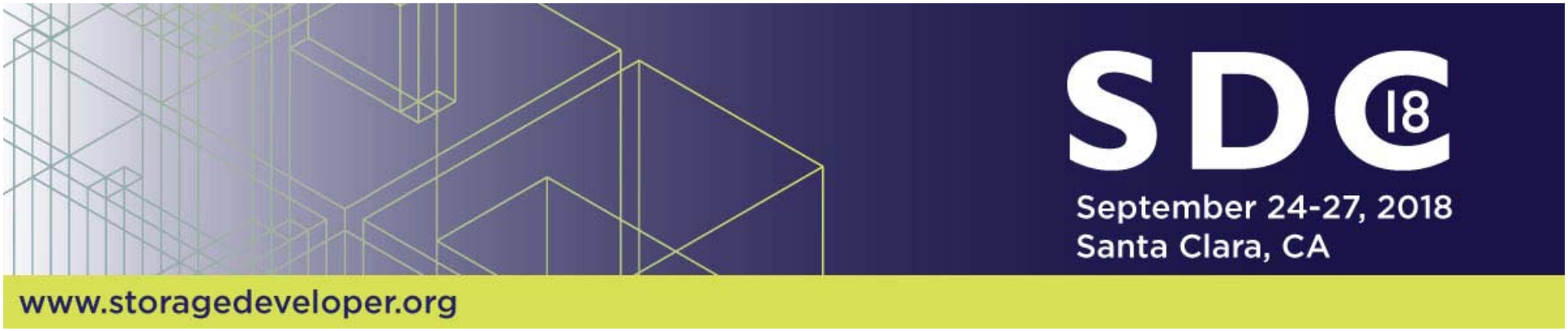


Solid State Storage System Technical Working Group Draft Methodology Status

Peter Murray, Virtual Instruments

Drew Tipton, Toshiba Memory America

Agenda

- S4 TWG history
- S4 TWG Methodology status
- Parallel work – Workload TWG
- Suggestions for future work
- Summary

S4 TWG History

- Solid State Storage System Technical Working Group
- Founded by a small group of people who realized that ‘normal’ spinning disk test paradigms didn’t properly characterize performance of solid state storage system technology
- The TWG was formed to address these issues and deliver a methodology to address the requirements for testing solid state storage
- This methodology is valid for testing any storage system
  - Features not included in a system can be noted to help others understand relative performance

The “What” of S4 testing… part 1

Big questions – first, what is a “Solid State Storage System?

It’s a “Solid State Array” (already defined by SNIA) with the following characteristics:

- Two or more redundant, networked storage controllers
- Solid State Storage Devices accessible by those controllers
- Two or more redundant data access paths.

The "What" of S4 testing… part 2

And, what is a "real world workload"

This one is a bit harder to answer…

- Reproduction of an application workload
  - Traffic patterns including temporal and spatial locality
  - Not just a random pattern—what does an application do?
- NOT simply a four corners testing methodology
- Can vary significantly over time and among customers
  - A database workload looks different than an email system workload or a file server workload or a virtual desktop workload
  - Or a different database workload, or… a combination workload…

The "Why" of S4 testing…

Now, why do we need a different methodology?

With Solid State Storage, there is much more that goes on within the array other than just data storage.

- Flash Translation Layers
- Flash Wear Leveling
- Compression (in-line or post-processing)
- Deduplication (also in-line or post-process)
- Snapshots (point-in-time copies, delta snapshots, read-only…)
- Garbage Collection (on both the array and the device level)

While some of these may occur in spinning disk arrays, the implementation of these features is part of what makes solid state arrays so different and efficient.

The “Who” of S4 testing…

There are two different ‘who’s’ involved here….

Who is doing the testing?

- Ideally, this is going to be either an impartial third-party or the customer themselves. One idea is to make this similar to SNIA Emerald’s testing and have results readily available.

Who is going to benefit from the testing?

- The target customer for all this testing is, ultimately, the consumer. The idea is to help consumers decide, through the use of impartial testing results, which system best suits their needs.

The “How” of S4 testing…

Now we get to the big question – how are we going to do what we’ve set out to do?

And it really is a big question… and lots of different people from lots of different companies have their own ideas about what is ‘right’ to do this testing.

Our goal was to come up with a unified, objective method for measuring real-world performance in Solid State Storage Systems. One that could be used by a customer to help them establish which systems were ‘right’ for them…

Capacity Optimization

Capacity Optimization (Data Reduction)

- Designed to reduce the amount of space needed to store data.
  - May work individually
  - May work collectively
- Combining techniques may not offer savings
  - E.g. compression may not help random data
  - Deduplication may not help highly random data

Space Consuming Considerations

- Device-based redundancy
  - Writing multiple copies of data
  - Parity calculations
- Overprovisioning
  - Provisioning enough space to accommodate post-ingest data reduction
- Provisioning sufficient space
  - Enough for the expected lifetime of an app

Capacity Optimization Techniques

- Thin provisioning
- Data Deduplication
  - Removing duplicate data patterns
  - On or off block boundaries
- Compression
  - Reducing redundant or unneeded characters
- Snapshots/Clones
  - Metadata-based snapshots/clones may minimally increase data requirements

Test Definition
And
Execution Rules

Test Definition and Execution Overview

- Workload creation
- System provisioning
- Preconditioning
- Baseline tests
- Feature tests
- Resilience tests

Workload Model Creation

- A new system is best measured running the application(s) that are to run on that system
  - However, doing so with multiple vendors is impractical
- Modeling the application(s) that are to run on that system is a safer and more realistic option
  - Model should reflect “normal” operation and significant periodic operation
  - Model should be as fine-grained as possible
  - Workload model should be collected either from storage array or network – and needs to be reproducible

Workload Model Procedure

- Obtain application sample:
  - Performance statistics from an array running same app(s)
  - Measure performance statistics from interface / software shim
  - Avoid introducing latency due to performance procedure / shim
- Create a model from the application sample:
  - 8 hours is proposed as collection time
    - Reflects a workday
    - May not be long enough for large systems or complex apps
    - May not be long enough to cover all periods of high activity
      - Example: backup windows occurring late at night or scripts to re-index and/or compress databases executing during off-hours. These also need to be included in planning.

Workload Model Sample Statistics

Application sample should contain:

- Read/Write ratio
- Random/Sequential access ratio
- Block sizes
- Transfer sizes
- IOPS
- Thread count
- Temporal locality
- Spatial locality

Workload Model Data Content

- Data content should represent the data transferred during the sampling period as closely as practicable.
- Data deduplication may be observed from:
  - Statistics gathered from an existing storage system
  - A network device capable of gathering and analyzing the information
- Data compressibility may be observed from:
  - Statistics gathered from an existing storage system
  - Derived by executing a compression algorithm against a LUN or directory to derive an average value

System Provisioning

- Full configuration required to run all tests:
  - Storage system
    - Remove metadata to ensure “clean” system state
    - Provision LUNs
  - Network
    - Provision zones, queue depths on switches and interfaces
  - Testbed and test tools
    - Install testbed
    - Configure tests

Provisioning Queue Depths

- Record fibre channel queue depths from storage interfaces and network
  - Take configuration information from all storage system HBAs and from the closest switch interfaces to the system
- Use these same queue depths to configure the testbed

Workload Independent Preconditioning

- Ensures system is in, as much as possible, an initialized, known state
  - As close to possible as it was received from the factory
- Prevents overstating performance during testing due to metadata or data advantages
- Ensure basic solid-state enterprise features are enabled if available (e.g. thin provisioning, deduplication, compression) or note if any are not enabled

Workload Independent Preconditioning Procedure

- Use storage system commands to:
  - Remove all LUNs / Volumes
  - Initialize metadata
- Configure testbed queue depths
- Sequentially write a non repeating, incompressible random data pattern to the entire advertised and reserve storage space, using a block size of 512 KiB
  - Typically involves writing twice to advertised storage space
- If cache is present, use only for intended purpose

Workload Dependent Preconditioning

- Ensures system reflects an application having been in operation for a “reasonable” period
  - Both data and metadata should be in “used” state
- This step is open to debate
    - JBOF systems may require this step
    - Larger systems may be difficult to precondition
- There is little to no benefit obtained from preconditioning systems that abstract physical locations by using metadata to determine where data resides over an entire system

Workload Dependent Preconditioning Procedure

- If modeling not possible, use an alternate
- Among alternate patterns being considered are the following:
  - Second IDC paper
  - Emerald EnergyStar test
  - Virtual Instruments workload
  - testmyworkload.com workload
  - Netmist
- In either case, run model against storage system

Workload Independent Preconditioning Procedure

- Run the model created for testing
- Resulting model is then run against the storage system
  - Time of 8 hours is proposed for now
    - May not be long enough for large systems
- Systems that abstract physical locations to metadata may not see any change to performance due to preconditioning
  - In this case, doing so does not aid testing

Baseline Test Procedure

- Collects active metrics to measure storage array performance
  - Is the precursor to all other testing
- Baseline test should reflect an application operating in a “normal” environment
  - I.e. it should not test additional array features/operations such as snapshots, clones, backup/restore, or data mining unless features are a part of normal operation
- Use model from the application sample:
  - Resulting model is then run against the storage system

Baseline Test Procedure

- Initial configuration should be well within the known limits of the system
- Run the first iteration
  - Note any significant transient performance deviations
    - Many flash systems experience brief spikes in latency, drop in performance
  - If performance remains flat, increase thread count and run another iteration
  - Continue iteration until a “knee” is recorded where performance dramatically drops or latency dramatically increases
- When maximum is reached, record parameters
- Note: A single workload may not load an array to maximum. If multiple applications are to run on the array, test with all applications. The “knee” should not be reached if the goal of testing is sizing for real-world use.

Baseline Test Procedure

- Some arrays do not implement data reduction
  - Some offer it as an extra feature
  - Others offer it as a basic feature
- To account for these variations, testing with or without data reduction is permitted
- The tester is free to choose either method; the chosen method must be disclosed
- The method chosen for baseline testing is also used for feature and resiliency testing

Feature Testing

- Additional tests based on the chosen baseline method
- Designed to show the performance impact of common features:
  - Snapshots
  - Clones
  - Backups
  - Deduplication
  - Compression
  - Thin provisioning

Feature Testing Procedure

- Feature tests follow baseline tests with no interruption
- Perform each test separately

Resiliency Tests

- Measure the effect of typical failures:
  - Controller and controller cable connection redundancy
  - Disconnecting a network cable from a storage array controller
  - Disconnecting a power cable from a storage array controller
  - Shutting down a storage array controller via the system console
  - Removing a power cable from a network switch
  - Removing one or more SSDs
  - Removing one or more connections bridging controller to system shelf
  - Removing a storage shelf from a redundant shelf array backplane

Resiliency Testing Procedure

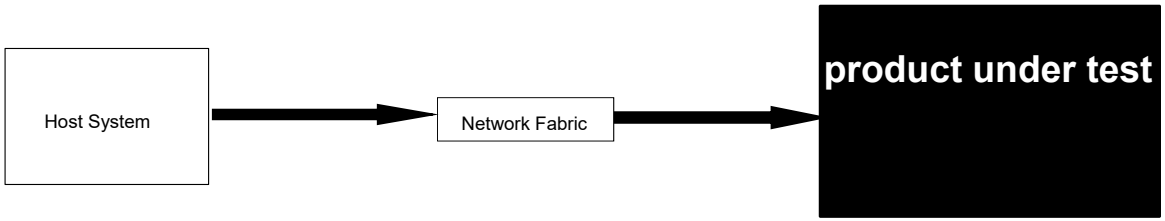
- Not required to follow feature tests without delay
- Perform each test separately
  - Ensure that system returns to steady state before continuing

Updates to Execution Rules

- Based on the collection of data using the aforementioned execution rules, the TWG will make updates to these rules as needed

Configuration Example

- This methodology does not constrain the precise configuration and interconnection of the hardware necessary to complete a SNIA Solid State Storage System Test Methodology
- Test sponsors are free to modify configuration to suit their needs and equipment, provided that the configuration is generally accessible to the public and that no other requirement of this methodology is violated. Multipathing between the host system and array is allowed.



Results Reporting

Results Reporting

- Results are to be reported following test:
  - Baseline tests
  - Feature tests
  - Resiliency tests
- The data in this section becomes the basis for comparison among storage arrays

Required Results Reporting

- The following results must be reported:
  - Throughput in MB/s (Amount Transferred)
  - Total IOPS
  - Response times:
    - Minimum
    - Average
    - Maximum

Optional Results Reporting

- Reporting for the following results are strongly recommended:
  - Per-LUN IOPs
  - Network fabric statistics
  - Data reduction efficiency
    - Size of written data vs. data at rest after all reduction processing
  - Bytes offered and transferred

Data Fest

- When preliminary methodology is approved, a data fest is proposed to test its validity
- Participation requires an NDA
- Testing will be performed remotely
- Results will be reviewed privately
- Anonymized results will be distributed to NDA signers
- Adjustment to the methodology will be made as required

Parallel Work – Workload TWG

As a partial result of the work done in this TWG, it was noted that a ‘standard’ workload needed to be developed in order to aid in testing and evaluation. An additional technical working group has been created in order to create guidelines for this ‘standard’ workload

Continuing Work

- Complete test procedures
- Review and refine results reporting
- Complete a second version of the specification

Want to help?

- S4 TWG meets Mondays at 1pm Pacific
- Workload TWG meets Wednesdays at 1pm Pacific
- We need your help!

Thank You