White paper

virtana

Best Practices for Evaluating Storage Using Workload Testing & Validation

When evaluating new or changed storage platforms to determine the optimal storage technology, product, or configuration for your organization, three factors are critical:

- The workload must accurately reflect the environment; it must be realistic
- The environment must be properly configured for each platform tested
- The tests must follow a consistent methodology

This paper offers a proven methodology, focused primarily on block-based systems, for framing these tests to provide a fair and accurate representation of the new or changing technologies, and how these technologies should be tested and validated using workload profiles representative of the environment into which they will be installed.

virtana

TABLE OF CONTENTS

INTRODUCTION

Virtana is the leading provider of storage performance testing and validation solutions, serving IT organizations as well as leading storage technology vendors. Storage performance testing and validation matters to IT storage teams because it's critical for improving their job performance – enabling them to more intelligently purchase, configure and deploy networked storage. For many years, other IT disciplines like networking and application management have enjoyed relatively modern preproduction testing and performance validation tools. The storage team just had freeware tools with no professional support. So, they were mostly condemned to explaining why I/O is slow... on the production floor.

Estimating capacity needs is pretty easy; rarely have we heard of IT organizations unexpectedly running out of storage. But it's a good bet that everyone reading this document has personally experienced the effects of a "performance outage" or "brownout". It's partially because it's really difficult to project performance needs unless you have the right performance tools and the right methodology. So, overprovisioning happens not just for capacity, but just as often, to mitigate performance risks.

Doing storage performance testing well matters because it helps to answer these common questions:

- Can I improve application performance by implementing new storage technologies or products, or changing configurations? If so, by how much?
- Can I afford the performance improvement? Will new techniques like compression / deduplication / snapshots reduce the effective $/GB without substantially impacting performance?
- How do I select the best technology, product, or configuration to match my application workloads?
- Which of my workloads will gain the most by moving to new architectures or products?
- Where are the performance limits of potential new configurations?
- How will storage behave when it reaches its performance limits?
- What are the performance limits?

The primary economic benefits of storage performance testing come from three areas: (1) improved application performance, (2) storage cost optimization, and (3) risk mitigation, which allows IT architects to innovate faster.

So why aren't more people doing this today? Simple: it has been hard and too time-consuming using DIY or shareware tools, and frankly, the results simply do not reflect reality. These tools cannot accurately simulate real-world production workloads. That's why Virtana sells Load DynamiX Enterprise and specifically, why we wrote this document, to expose storage engineers and architects to a Best Practices Methodology to accompany the use of Load DynamiX Enterprise.

OBJECTIVES

What are the specific objectives of this document? To go beyond the technology and to:

- Provide guidance for customers to ensure a fair evaluation of new technologies and vendor products
- Ensure architectural differences in new technologies, vendor platforms and vendor-specific configuration best practices are accurately represented in customer testing
- Ensure customers' needs are clearly communicated to vendors, and vendors provide a configuration to meet those specific needs for the testing process
- Allow technologies to be accurately evaluated based on testing results and evaluation criteria
- Provide customers the opportunity to include the initial array setup and configuration process in their evaluation criteria
- Discuss common pitfalls associated with testing and why these should be avoided

THE NEED TO TEST AND VALIDATE

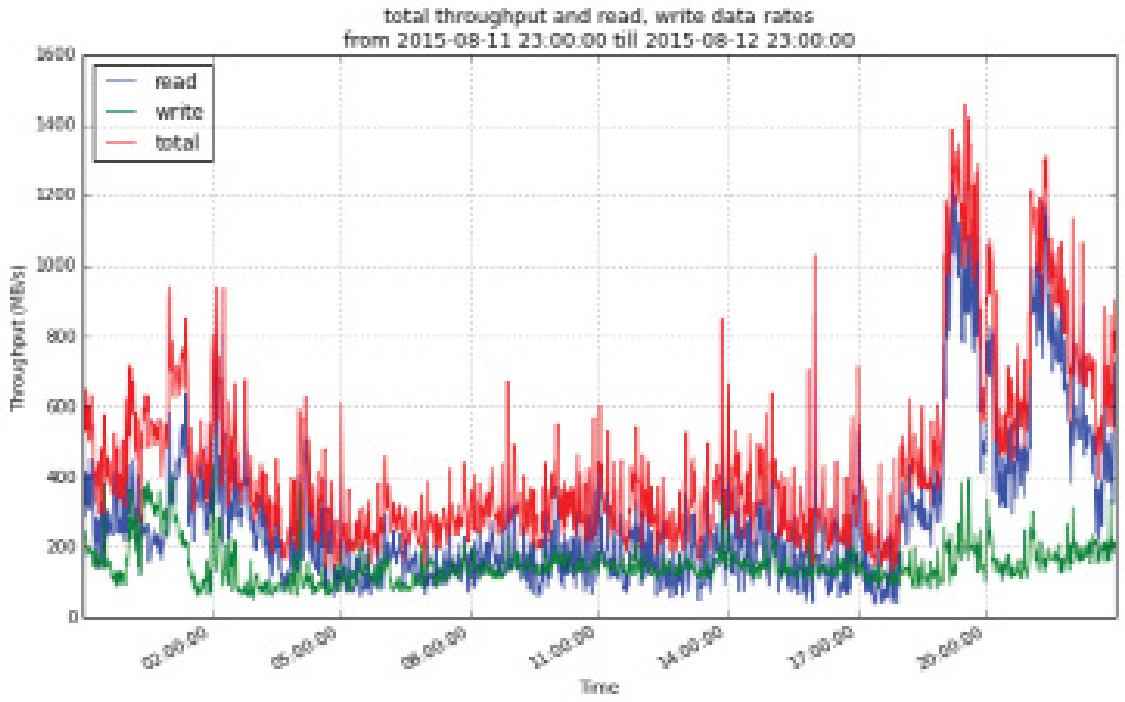
There are several benefits and use-cases for test and validation:

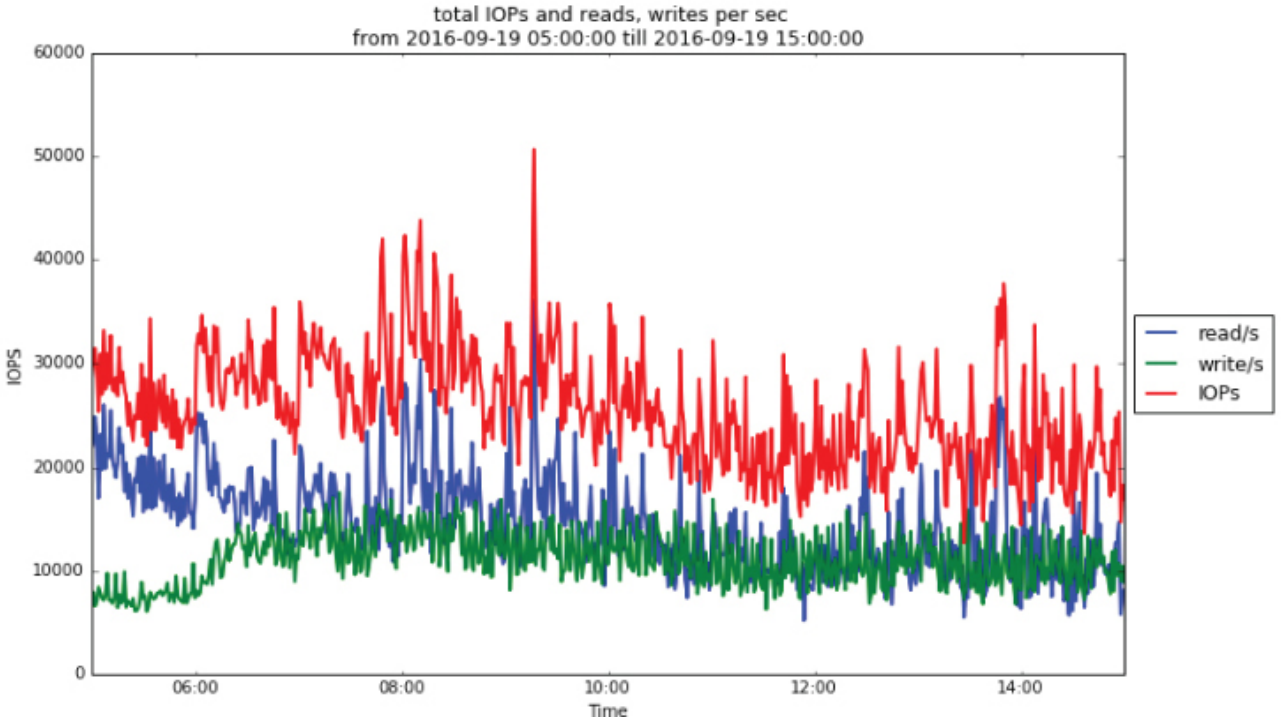
- Moving to new platforms, like from fibre channel to iSCSI, or to Software Defined Storage, can involve leaps of faith, and risk. Test and validation by far the best way to mitigate this risk.
- If storage vendors could custom-build products and configurations based on individual customer application profiles, there might not be such a need to test. But they do not, and so storage engineers and architects must go beyond the simple performance profiles that appear in spec sheets, and the guidelines provided by "industry standard" benchmarks. Customers must have data that's relevant to their environment.
- Storage workloads are different across different applications and platforms. No one would expect a streaming video application to stress a storage array in the same way as an email server.
- Applications have wildly different requirements for storage performance. Latency for some applications have to be measured in microseconds, while for some, multiple seconds.
- No one storage array, technology or configuration can universally provide the best price/performance for all application types. Different storage systems perform differently for different workloads: there is no one storage that fits all. In fact, we know that both storage vendors and component manufacturers optimize their arrays for particular workload types. But their sales teams generally are not aware of this level of detail, and often sell a "one size fits all" solution.
- It is important to know how a storage system performs in your environment, including expected growth.
- It can literally save millions of dollars in new storage procurements to understand the configuration best suited for your application. We have examples where customers halved the number of scale-out nodes that were originally proposed by their vendors.
- Industry standard benchmarks are known to be "gamed" by vendors who rightly want to present their products in the best possible light. Often, configurations are tested and reported on, which bear no resemblance to real customer configurations.
- Vendors are under pressure from their marketing departments so they routinely publish specs that are not realworld in nature. A huge IOPS number may be offered, but it may have been achieved by using block sizes that are not representative of most applications. Or they may claim "random access" numbers based on a small dataset or data read by cache. And there never is one block size for the composite workloads running on modern arrays. It's a distribution of sizes for different LUNs, reads and writes, and it changes over time. Running a test showing only an average block size isn't helpful to the customer.

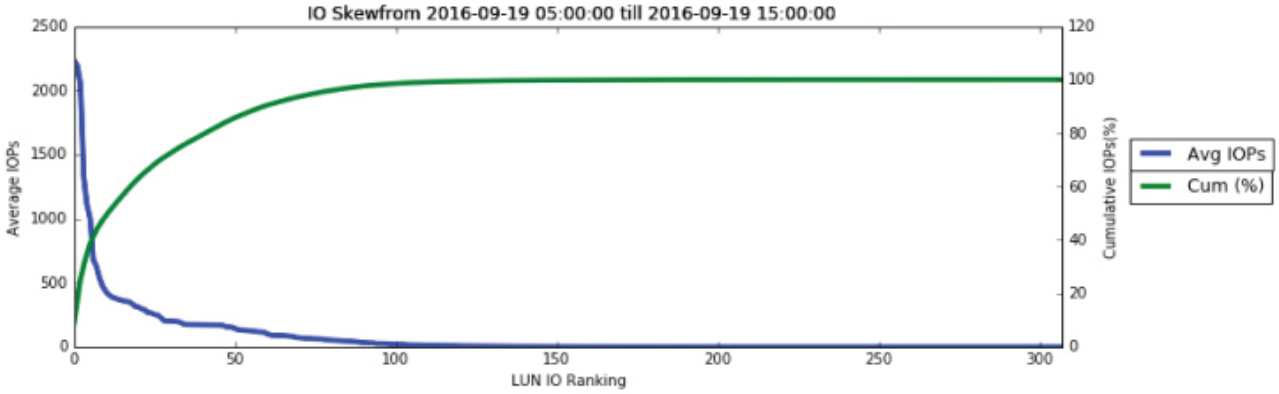
CHARACTERISTICS OF A PRODUCTION WORKLOAD

Production workloads have variations of IO sizes, rates, sequentiality and working sets over time on a small (milliseconds) and large (hours) scale across hundreds of LUNs, and affects tens of terabytes of data during a 24h work cycle. In the following graphs, you can see the workload I/O rates and size variations over time. You can see variations in throughput from 200 to over 1400 MB/s, and in IOPS from 15K to 50K. A more typical graph you might see from simple performance measurement tools would average these numbers, producing a flatter, and much less accurate portrayal of performance.

Past simple system level metrics, it's extremely useful to measure SAN components, such as Front-End-Ports and LUNs. In the graph below, you can see a real-world example of how production workloads can use LUNs unevenly. In this example, a very small number of LUNs are handling nearly all the real world I/O. It is important that these differences are maintained in the test. In one example, 8 workloads were created for different levels of LUN activity and types of access versus 1 workload that treated all LUNs equally. The workload where the ratios of LUNs that had work were maintained the array could run the workload with reasonable latency all day long. The single workload that treated all LUNs the same resulted in 130ms latencies.


total throughput and read, write data rates
from 2015-08-11 23:00:00 till 2015-08-12 23:00:00
read
write
total
Throughput (MB/s)
Time


total IOPs and reads, writes per sec
from 2016-09-19 05:00:00 till 2016-09-19 15:00:00
IOPS
Time
read/s
write/s
IOPs


IO Skewfrom 2016-09-19 05:00:00 till 2016-09-19 15:00:00
Average IOPs
Cumulative IOPs(%)
LUN IO Ranking
Avg IOPs
Cum (%)

DEFINING THE TEST PROJECT AND BUILDING YOUR OWN BENCHMARK

Document Subjective Factors

To fairly and accurately perform the evaluation, you should document any potential factors non-essential to the evaluation. Consider the influence (if any) these factors may have on your results. They are valid considerations, but should be factored out when doing a purely performance test. **Examples of subjective factors include:**

- Existing professional relationships
- Length of relationship with incumbent vendors
- Perceived risks and complexities of changing technologies or migration
- Unsubstantiated third party reviews
- Unsubstantiated "rules of thumb"
- Internal concerns over change and what this might mean to existing processes
- Benchmarks that profile other applications
- Current knowledge about managing and maintaining particular vendors or products
- Maturity and feature sets of the array and vendor management and monitoring tools

Define Testing Objectives

Common examples:

- Determine the best performing platform for your existing workload at 1x
- Determine best performing platform for same workload when scaled (1.25X, 1.5X, 1.75X, 2x, etc.) It's a good idea to scale in small, regular increments. For instance, if you think your workload might grow 4x over the life of the array, consider testing at 2x, and 3x as well.
- Determine platforms and configurations that satisfy application performance requirements
- Identify the point at which workload exceeds performance requirements of a platform At this point, it's good to ask yourself if your testing objectives influenced by subjective factors, and factor them out.

Define Evaluation Criteria for Proof-of-Concept Testing, aka Bake-offs

As with Objectives, ask yourself if your evaluation criteria are influenced by subjective factors.

Business

- Long-term viability of the vendor
- Existing relationships (quid pro quo)
- Industry reputation

Financial

- Costs (Acquisition, Maintenance, TCO)
- Price / Performance

Operational

- Reliability
- Support and supportability (consider both the local service and remote support teams, and their ability to coordinate amongst your local and world-wide teams

Performance

- Identify the highest acceptable average latency
- KPI: Latency, Throughput, IOPS

Scalability

- Define the highest IOPS and throughput at acceptable latency for each platform

Identify Test Environment / Protocol

Be specific, and commit these details in writing. Get specific recommendations from all interested parties, including your DB and apps teams, server and networking teams, your vendors, and of course, your testing vendor. Ask for examples of previous test environments similar to yours. **Then specify:**

- What tests will be run?
- How will test results be collected? Here's where the Load DynamiX Enterprise repository will be invaluable.
- How will results be evaluated, and by whom?
- What are the success criteria for each test? Again, get input from other teams.
- What results are you expecting, based on what the vendor has stated/published?

Define Growth Assumptions

What is the anticipated growth of the workload over the expected life of the array?

- Can you correlate your growth in workload to business growth? Barring other input, it's a good place to start. Ask for revenue or number-of-client-user projections.

- Are your workload growth estimates reasonable,

given past growth patterns and business growth?
> Are there any ongoing initiatives that can will impact these forecasts (i.e. cloud migrations or M&A activity?)

Consider the possibility the array remains in service longer than expected.
> If the array remains in service longer than expected, what is a reasonable timeline?
> What (if any) growth would be expected during this extended time period?
> What are your expectations for performance and stability?
> How does this compare to the vendors expected EOL date?
> At what point will you stop adding to the array? (servers or VM's, storage capacity, workload, etc.)

Is consolidation required at a future date?
> Are the storage array ports sized correctly to accommodate consolidation?
> How will LUN consolidations change the characteristics of the workloads?

Develop Testing Schedule

As careful as you are, the schedule can change for a number of reasons. Try to build in flexibility; there's nothing more frustrating than finding some question you did not anticipate and not having the time to fully explore your options.
> What is the deadline for testing completion? Soft deadline or will you lose access to the test target?
> What is the sequence of testing that will occur in the timeline? Are all the resources lined up?
> Will the vendor or component supplier be required to perform the setup activities? Reconfigurations?
> How long is each test? Include set-up time, time to build the models, time to execute, and to create custom reports, if needed. Ask your testing vendor for estimates if you are new at this. Virtana has years of relevant experience to draw from.
> What time buffer is available if tests run longer than expected?

Candidate Selection

> What were the selection criteria for defining the candidates to include in the test?
> Why were these criteria important?
> What products will be included in the test?

Notify Suppliers

Although storage vendors or component suppliers will likely not be involved in the actual testing, vendors will size and configure their recommended optimal configuration, and will be responsible for the setup and configuration. And you WANT them to do the setup and configuration. Also, consider that the selected platform may remain after testing and become the production system, so implementation of the array should follow all best practices. When engaging your supplier, be sure to clearly review these points. It will enable them to do a better job of proposing the right array and configuring it properly. So, review:
> Testing objectives
> Duration of the entire testing process
> Evaluation criteria
> Description of the testing environment
> Equipment they will be expected to provide
> Services they are expected to provide
> Expected schedule of testing for their components
> Services they are expected to provide
> Expected schedule of testing for their components

TESTING

Optimal Test Environment

To some, when a new box arrives on the loading dock, it's an event not unlike a birthday or Christmas morning. Without delay, and frequently without a proper inventory control process, cartons full of shiny new technology are unboxed, racked or not, and connected to whatever server happens to be available. Storage devices are configured and manuals consulted only in the unfortunate event that the user interfaces aren't properly intuitive. What happens next may be performance testing, but it may also be an exercise in building storage performance anecdotes.

Why do we say this? Absent of a defined test process and controlled testing environment, you will be able to declare only that "I did X and Y was the result". This is a subjective statement, and it may contain valuable, experiential data. It is not, however, an objective analysis. The accuracy and interpretation of such results are not definitive of the subject under test. That is not to say that this sort of

experimentation has no value; it should demonstrate out of the box functionality (or lack thereof) and may provide valuable insight into ease-of-use. What it does not do is provide evidence of a rigorous testing methodology, including controlled inputs and outputs, repeatability and error analysis.

Testing can, and does, imply a means by which the quality and function of a subject is evaluated. In the strictest sense the "kick the tires" test process described above fits that definition. However, truly objective testing requires a controlled test process and a controlled test environment, or test bed if, you prefer.

A test bed provides a means of isolating the system under test from external influences and protecting non-test resources from any fallout from the testing process. These are important factors for the generation of the reliable test data and for maintaining business continuity. A test bed must be defined and documented for repeatability and to simplify the analysis of results.

Storage performance testing is disruptive and should never be performed in compute environments that support any kind of production or revenue-generating computing. Nor should performance testing coincide with other testing, such as user validation or ease of use testing. Meaningful storage performance data can only be generated in a controlled testbed and the most rigorous testing will adhere to well recognized standards and practices. These requirements typically call for a dedicated test environment to be set aside for the purpose of validating storage performance.

A well-designed test bed will incorporate similar conditions as expected in the environments considered for eventual deployment of the devices under test. Conditions designed to maximize some metric in a way that is not reproducible outside the test lab creates results that may look good on a marketing brochure, but which hold little value to you, the customer.

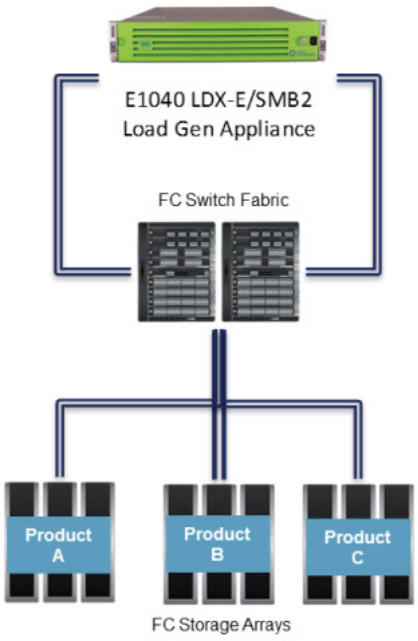
A comprehensive test environment will include the following elements and attributes:

- A space for the testing to occur, separate from conflicting activities
- A mechanism or engine to drive workloads (Load DynamiX appliance)
- Infrastructure to support applicable connection protocols
- A mechanism to record and analyze test data (Load DynamiX Enterprise)
- A library or toolbox of workloads to generate desired I/O patterns (Load DynamiX Enterprise)
- The number of storage ports involved in testing should be comparable to the target production environment
- Switches and network gear to emulate production
- Multipath access to the LUNs (Active/Active)
- The number of LUNs should be of the same order as in production. For instance, if the production environment is expected to have 2,000 LUNs, don't test with 2 LUNs, shoot for ~200. 10
- Replication or other internal processes (deduplication, compression, garbage collection, etc.) enabled in production should be enabled during the test runs

And, optionally

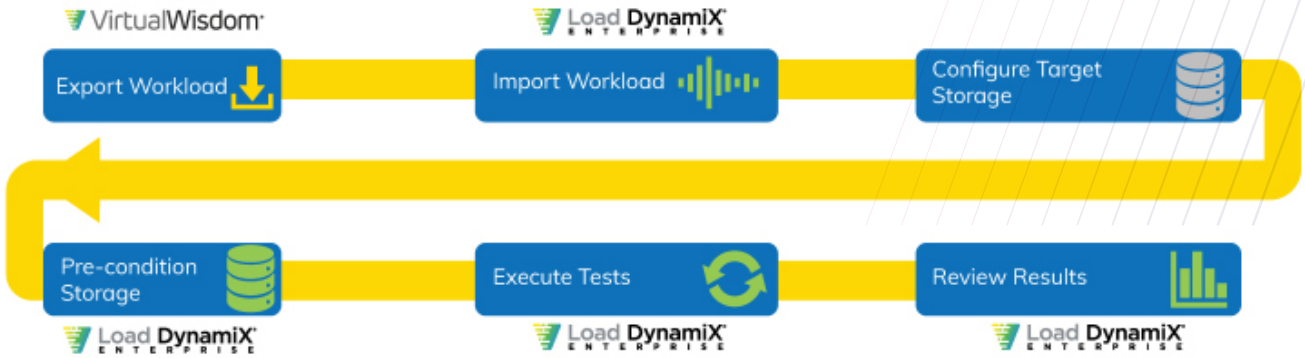
- A source for the modeling data (VirtualWisdom or existing production storage array)
- A controlled power source
- Power measuring devices
- A climate controlled environment

The test environment should be representative of the way the proposed storage will be used in production and should be centralized and easy to access and use. Here's a simple example of a test topology.



Storage Testing Workflow

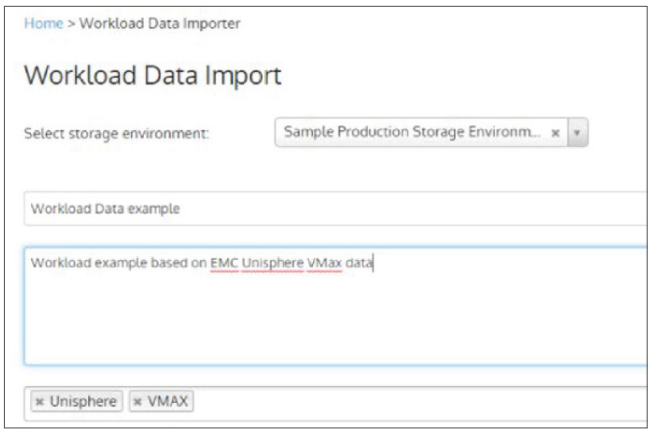
Here's a workflow illustration. We'll discuss each step in order, below.



Importing and Exporting Workloads

One of the first issues to understand is the workload sample you're starting with. with thus far. Often, testers start out thinking they should grab a week's worth of data and run that. For most use cases, that is a waste of time. It is best to pick a few key time periods, maximum IOPs, maximum throughput, and maximum latency and grab 1 to 4 hours of this data to run against the array. This will provide the most important results in the least amount of time.

The days of spending dozens of hours parsing log files, in the oftenfutile attempt to create realistic workload models are over. The Workload Data Importer module of Enterprise is designed to import and analyze workload data from storage infrastructures, automatically, to create extremely realistic workload models. The workload data can be exported from VirtualWisdom, storage vendor monitoring tools or server utilities such IOStat as Comma Separated Variable (CSV) text files. The data is analyzed to detect workload IO profiles to understand the workload characteristics in terms of IOPs, Throughput, Read/Write ratio etc. The WDI GUI is here (right): Storage Testing Workflow Here's a workflow illustration. We'll discuss each step in order, below. The ideal data source, to create the most hyper-realistic model, is VirtualWisdom. It can obtain up to 30 different metrics.



The ideal data source, to create the most hyper-realistic model, is VirtualWisdom. It can obtain up to 30 different metrics.

Prepare and Configure Storage Target – Precondition

Storage platforms under test should be configured according to the vendor's best practices.

SAN storage arrays should be "preconditioned" prior to the testing:

- Data must be written to an array prior to reading it
- AFA solid state storage should be broken in to bring it to the state it is in production
- Utilize the purpose-built pre-conditioning workload in Virtana SLT-E 5.3 and later.
- Depending on the configuration and the vendor, the storage array might be left for a period of time to process the data offline. The vendor will offer advice.

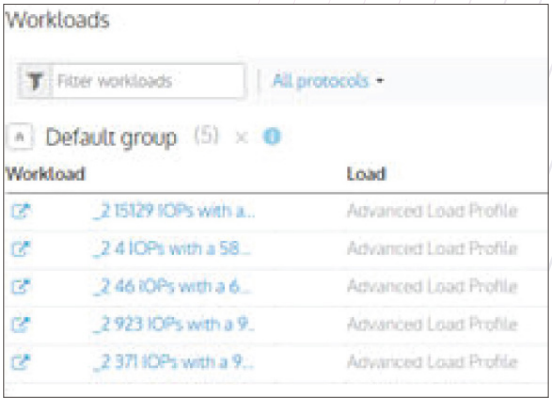
Execute Tests – Current Production Workload

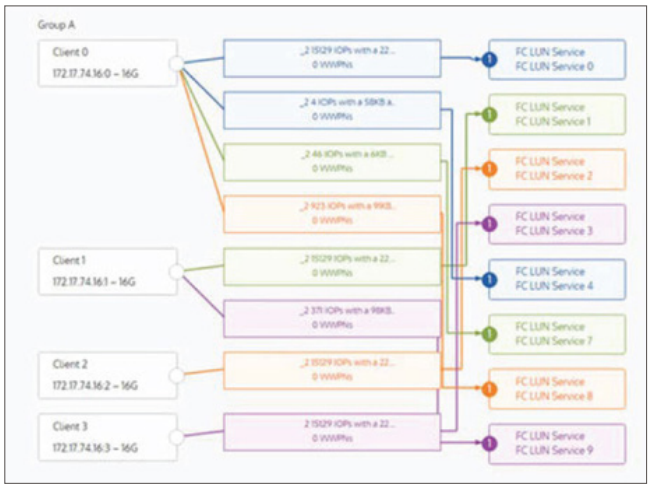
Perform a baseline test of workload across all systems using the current production workload at 100%. Precision should be to the highest possible extent to produce useful results. **This includes:**

- Read and write rates and their variations in time and across the LUNs
- Request sizes and their variations in time and across the LUNs
- The amount of capacity (logical address space) read from and written to during the test

Workload scaling tests the storage performance at higher loads using the characteristics of the current production workload.

- Consider iteratively performing the workload tests gradually increasing scale to reflect typical growth.
- Example: If planning to test up to 3x growth in workload, perform tests using increments of load scale of .5x
- The “sweet spot” of a storage array varies, and you may find the array that performs well at 1x or 1.5x, may degrade at 2x, while other arrays are just warming up. This granularity will help identify this, and define the optimal workload ranges for a product.

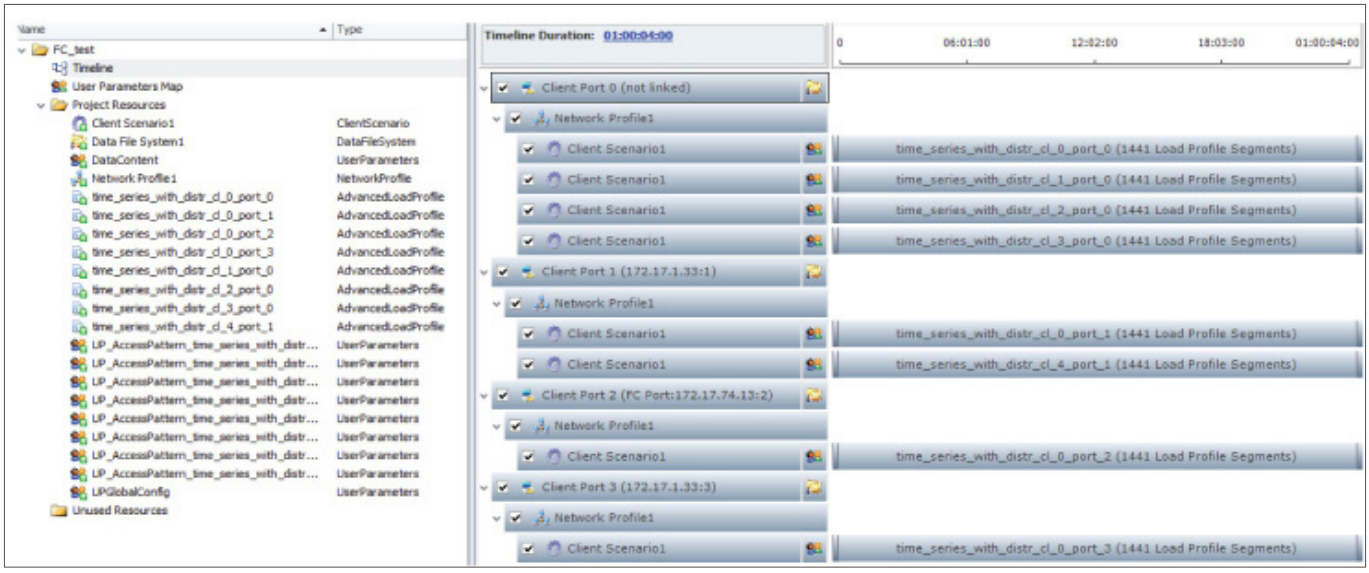




Test management environment should be centralized and easy to use. The screen shots below demonstrate how simply a complex, composite workload may be illustrated. The graph on the left lists multiple workloads, and the graph on the right shows the relationships between clients and targets, in a single test environment.

Execute Tests with TDE for models that exceed Virtana SLT-E functionality

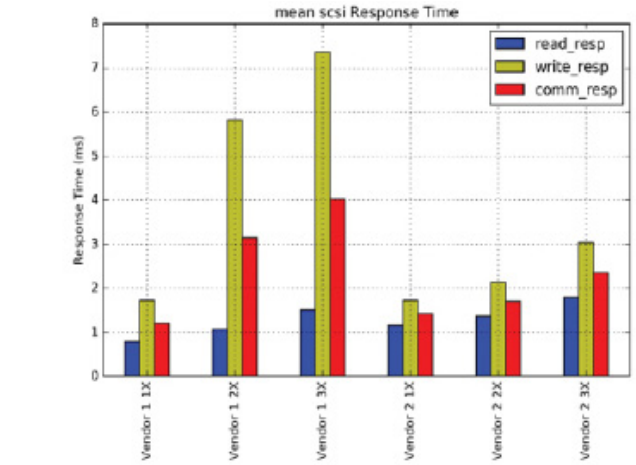
Here’s a TDE Project corresponding to the Virtana SLT-E case presented above, just as an example.

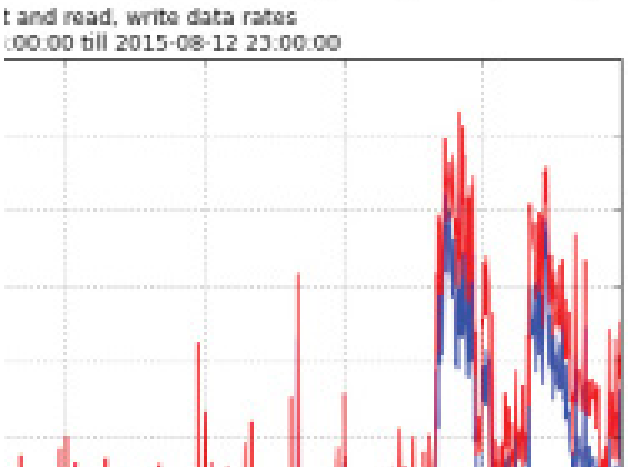


Review Results

Generate easily readable reports, typically involving latency, throughput and IOPs. Here, in a multi-step test, you may determine if some products simply shouldn't move on to the next round of testing because of their shortcomings.

Below, average response time and IOPS for a baseline and scaled-up test in a 2-vendor bake-off. In the first graph below, it's easy for the reader to see the effect on latency, of increasing the workload (by increasing IOPS). In this case, vendor 2 scales more effectively. At a 3x workload, vendor 1 combined response time is 4ms, while vendor 2 is approx 2.3ms (exact figures are available).





COMMON TESTING PITFALLS

#1 Skipping Conditioning

The preconditioning of the storage is a must pre-requisite of the testing for the following reasons:

- Storage arrays typically "know" whether a certain address has been previously written to or not. In case the test workload reads from the address that has now data, the array returns zeros without consuming internal resources. The response time is unrealistically low. The test results are skewed towards low response times
- Fresh arrays have empty metadata tables. Access to them consumes resources and might affect response times. Having them empty or containing few entries (by writing zeros, for example) will lead to unrealistically low response times in the test
- AFA to be tested as in working conditions need to have their solid-state storage be initialized and written to perform as it performs in your real-life production environment.

#2 Using Bad Workload Models

Misconception

- Production workload can be simulated using scripts and/or tools like Iometer or Vdbench in a test lab. We often talk about an early user of Load DynamiX, GoDaddy. GoDaddy tried to use Iometer and got artificially high performance results on the target array. Sadly, when deployed into production, actual user experience was poor. This was mostly due to Iometer's inability to recognize the huge impact metadata I/O had on the workload. The same tests run using Load DynamiX turned out to be extremely realistic and gave GoDaddy the results that resulting in good deployment decisions, and happy GoDaddy customers.

Reality

- There is no substitute for actual production workload
- Building a test environment is resource intensive
- True production conditions may not be accurately captured in the test environment
- Incorrect assumptions and deviations in testing steps in the simulation can lead to the wrong decision.

High-level steps of building the test environment (the hard way)

- Identify physical infrastructure to be used (server, san, storage)
- Consider make, model, age, OS version/patch levels, firmware levels, driver levels, etc.
- Build /update the test environment to reflect production
- Ensure components are at supported levels for the testing platform

- Configure SAN connectivity
- Configure matching number of ports, port speeds, cable types, patch panels, etc.
- Create scripts to simulate workload
- Distribute scripts to systems that will run the scripts and drive the workload

Setup (the hard way)

- Operating System Updates
- HBA Drivers & Firmware
- Load Balancing Configuration
- IP Addresses, DNS Records
- Testing Scripts
- Zoning & Masking

Execute (the hard way)

- Execute scripts on servers/VMs
- Monitor performance
- Collect & carefully aggregate test results from the various servers

Why this approach fails:

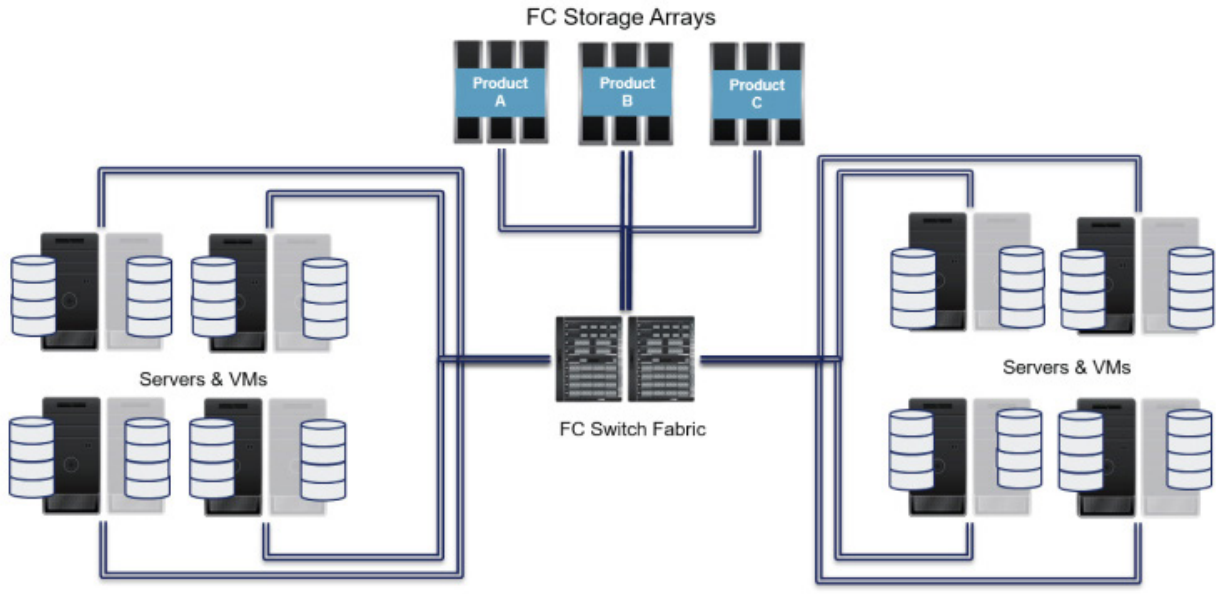
- Workload testing is partially an art and partially a science. With the DIY approach, you have only your own experience to depend on. And perhaps the kindness of forum participants. When, not if, questions come up; it's Best Practice to have a team with decades of experience to rely on.
- Conventional tools and scripts attempt to approximate and simulate workloads; and for all but the smallest, simplest workloads, they fail.
- Setup effort increases exponentially, the more accurately you try to simulate workloads.
- Testing is not centralized; results need to be collected and correlated to minimize mistakes and make your efforts repeatable. A test that's not repeatable is, by definition, not a good test.
- Due to the complexity of setup, execution, collection, and analysis, customers generally perform fewer testing iterations, resulting in fewer useful data points.
- Test are based on assumed characteristics (aka guesses), which are then scaled; the effect of multiple inaccurate assumptions is compounded and multiplied.
- You won't know if your assumptions are truly correct until you are in production and driving workload.

Why spend significantly more effort, time, and money to create an approximation, when you can use your production workload?

#3 Forcing Unnatural Array Configurations

"We are going to configure all arrays exactly the same, test the same workload, and compare the results" is not a Best Practice, though it seems fair. It's not. Imposing a like-for-like configuration across different products:

- Often deviates from the best practices for one or more products in the test group
- Would not match the solution / configuration you purchase or put into production
- May reflect a technically UNSUPPORTED configuration
- Produce sub-optimal performance metrics



The target testing environment should reflect the configuration you would put in production, even if the configuration is not a direct comparison to the other configurations in the test. By sharing your objectives with your vendor, you will get their years of experience in system configuration.

#4 Not Agreeing on Terminology

There are some common and not-so-common terms that get thrown around; and without a common definition, can cause barriers between vendors, managers, engineers, and other involved parties. Here are some terms we should all be familiar with:

All Flash Array (AFA)	An all-flash array is a solid-state storage disk system that contains multiple flash memory drives instead of spinning hard disk drives.
Array metadata table	A table, that has properties that store metadata such as variable names, row names, descriptions, and variable units.
Data compression	Encoding information using fewer bits than the original representation.
Deduplication	A specialized data compression technique for eliminating duplicate copies of repeating data
Garbage Collection	Solid state storage garbage collection (GC) is the process by which a solid-state drive (SSD) improves write performance. Garbage collection, like TRIM, pro-actively eliminates the need for whole block erasures prior to every write operation. When a file is deleted from a computer, most operating systems (OSs) delete the table of contents entry, but do not delete the actual data blocks from the storage media. Hard disk drives will simply overwrite the unneeded data blocks. Flash SSDs, however must erase the unneeded data blocks before new data can be written. Working in the background, garbage collection systematically identifies which memory cells contain unneeded data and clears the blocks of unneeded data during off-peak times to maintain optimal write speeds during normal operations.
Multi-pathing	Multipath I/O is a fault-tolerance and performance-enhancement technique that defines more than one physical path between the CPU in a computer system and its mass-storage devices through the buses, controllers, switches, and bridge devices connecting them.
Pre-conditioning	Preconditioning is the writing of data (random pattern) to the entire flash device to “break it in”. Data is written to the flash memory in units called pages (made up of multiple cells). However, the memory can only be erased in larger units called blocks (made up of multiple pages). If the data in some of the pages of the block are no longer needed (also called stale pages), only the pages with good data in that block are read and re-written into another previously erased empty block. Then the free pages left by not moving the stale data are available for new data.
Sequentiality	Degree to which data I/O stays in sequence
Working Sets	Defines the amount of memory that a process requires in a given time interval

CONCLUSIONS

By following the outlined Best Practices, you can realize the following benefits:

- Performance assurance: Ensure storage solutions will meet performance SLAs under specific workloads and confidently choose the optimal solution for those workloads.
- Reduced storage costs: Reduce over-provisioning by choosing the lowest cost systems for specific workloads; quantify the benefit and effects of every system.
- Increased uptime: Identify problems in the development lab prior to production deployment; validate all infrastructure changes against workload requirements and troubleshoot more effectively by re-creating failureinducing workload conditions in the test and development lab.
- Accelerate new application deployments: Accelerate time to market by validating new applications on new systems, making deployment decisions faster and more confidently.

CONTRIBUTORS:

Michael Bello
John Cryan
Carmine DellAquila
Craig Foster
Yang Gao
Peter Murray
Scott Newkirk
Elizaveta Tavastcherna
Jim Bahn