

White Paper: An Advanced Approach to Testing Unified Storage

Alan Newman
VP Marketing and Founder

ABSTRACT

Vendors and end-users of Unified Storage solutions need testing tools to confirm the performance of their unified storage products at *cloud scale*. Until very recently, commercial test tools have not been available and homegrown tools have been the only alternative.

Homegrown test tools, however, have difficulty keeping up with the challenges of increasingly complex and high-performance multi-protocol storage systems. Moreover, when individual departments create their own tools, multi-protocol system tests are difficult or even impossible to implement and maintain.

This paper explains these challenges and shows how they are addressed by the SwiftTest product family. These tools deliver multi-protocol testing at cloud scale at substantially less cost than homegrown solutions.

Target Readership

This paper is intended for technical and management personnel who design, implement, and/or maintain significant unified storage infrastructure.

Such personnel work for:

- Equipment vendors (both unified storage vendors and storage-aware networking product vendors)
- Storage service providers
- Enterprises
- Consultants

They hold a broad range of titles, and have job-specific interest in testing tools:

- Development engineers performing unit tests
- QA engineers running functional, stress and performance tests
- Manufacturing engineers running final tests prior to customer shipment
- Marketing departments doing competitive analysis
- Support engineers duplicating and troubleshooting problems
- Field systems engineers winning new customers or satisfying customer needs for environment-specific performance data
- Executive management approving budgets for testing equipment

They share these high-level goals:

- Contain cost
- Accelerate time to market
- Increase quality
- Consistent testing practices
- Gain competitive advantage

Many factors contribute to achieving these goals; this paper focuses on the contribution that testing tools can make.

Typical Unified Storage Testing Practices and Challenges

Unified storage vendors, storage service providers and enterprises have been forced to build their own test tools to meet their needs. These homegrown tools are used in conjunction with industry standard benchmarks, including Spec SFS, but benchmarks do not address the requirement for functional, stress, and performance testing.

Homegrown testbeds rely on racks of servers to generate the traffic required to fully test their unified storage systems. Users must purchase, install in racks and load operating system software onto each server. Next, the user needs to load and configure testing software. When real applications are used to generate traffic, they need to be surrounded by homegrown scripts to control them. In the extreme case, the QA team will develop all of the software in-house (except for the operating system).

Multi-protocol testing at cloud scale presents a set of technical as well as business challenges. Some apply to testing in general and others are specific to the unified storage environment. Below we elaborate on those challenges, in terms of five business imperatives.

Contain Costs

Testing at cloud scale via general-purpose servers, especially when burdened by the full code-path of actual customer applications (e.g., Microsoft Office), are inefficient. In testing environments it's typical for customers to compensate for this inefficiency by configuring large numbers of general purpose servers with Virtual Machines. Such proliferation magnifies their costliness.

Cost is not limited to purchase price. To budget properly, customers must calculate the *ongoing* costs of rack space, power, cooling, yearly maintenance and the system administration for (often) large numbers of general purpose servers and the associated software (applications and operating systems). These can be sizable, as compared to (and in addition to) the acquisition cost.

Finally, to the above readily identifiable hard costs must also be added the salary and benefits of engineering talent diverted to internal test tool development. These development costs are ongoing as protocols are updated and new protocols are added.

Accelerate Time to Market

In today's increasingly competitive markets, it is not enough to bring cost-effective products and services to market. One must introduce them *on time*. "First to market" often translates to market dominance. Delayed delivery of well-tested products and feature enhancements can have a permanent negative effect on product success.

A major contributor to delay stems from the inflexibility of homegrown tools, especially when testing configurations must be dynamic to accommodate a wide range of engineering needs. Homegrown tools lack the sophistication needed to rapidly execute a switch to a previously defined testing configuration. One customer reported that it took them *hours* to reconfigure their homegrown testbed each time they ran a different set of tests. In a typical QA cycle with multiple different tests within it, the setup overhead was adding up to days. Proliferating large numbers of general purpose servers can mitigate such wastes of time by turning a serial reuse model into a parallel-setup model, but cost-conscious customers are rarely prepared to make the additional investment in test tool hardware to support parallel testing.

Increase Quality

A shortcoming of homegrown testing approaches is that even if a customer believes they've achieved 100% comprehensiveness along *individual* testing axes (e.g., SMB/CIFS, NFS, iSCSI, HTTP), their homegrown tool lacks the sophistication necessary to test along multiple axes *concurrently*. To illustrate, a storage vendor had two distinct homegrown tools, one for NFS and one for SMB/CIFS. It was extremely difficult to measure the impact of running large volumes of NFS traffic on CIFS performance because the two tools generated different reporting data.

Consistent Testing Practices

Today's unified storage servers simultaneously support files, blocks and HTTP. Homegrown testbeds are not able to deliver what QA needs, namely, a *single* test tool capable of generating the workload of multiple file, block and HTTP clients concurrently, and measuring the interaction when both protocols are accessing the same content.

When a department (engineering, QA, manufacturing, sales) needs to test, it will, even if it must “roll its own” tools. However, not all departments can “roll” with equal ability or speed. Moreover, when they build their own tools, departments typically don’t invest the effort to coordinate their tool development with other departments. The resulting testing “tower of Babel” is not aligned for maximum corporate health. It is neither typical nor easy for independent departments to leverage each other’s knowledge and re-use each other’s test suites. Homegrown testbeds rarely unify all departments around a common testing language and process, lack of which could be thought of as a hidden—though difficult to measure—cost.

Gain Competitive Advantage

Most companies (correctly) assume that competitive advantage is largely gained by releasing compelling products. Too often the competitive advantage of using good tools is forgotten. Consider these facts:

- Resolve problems quickly: One QA engineer told us that he was unable to recreate a customer reported problem using their homegrown tools. Within hours, they were able to recreate it using SwiftTest.
- Respond to competitive sales situations quickly: A vendor increases its credibility if it can efficiently configure, run, and document results in customer-specified tests using an independent, commercial test tool.
- Gain competitive knowledge: Thoroughly testing the strengths and weaknesses of one’s own products and those of the competition leads to knowledge, this in turn translates to market power.

Introducing the SwiftTest Solution

Multi-Protocol Testing at Cloud Scale

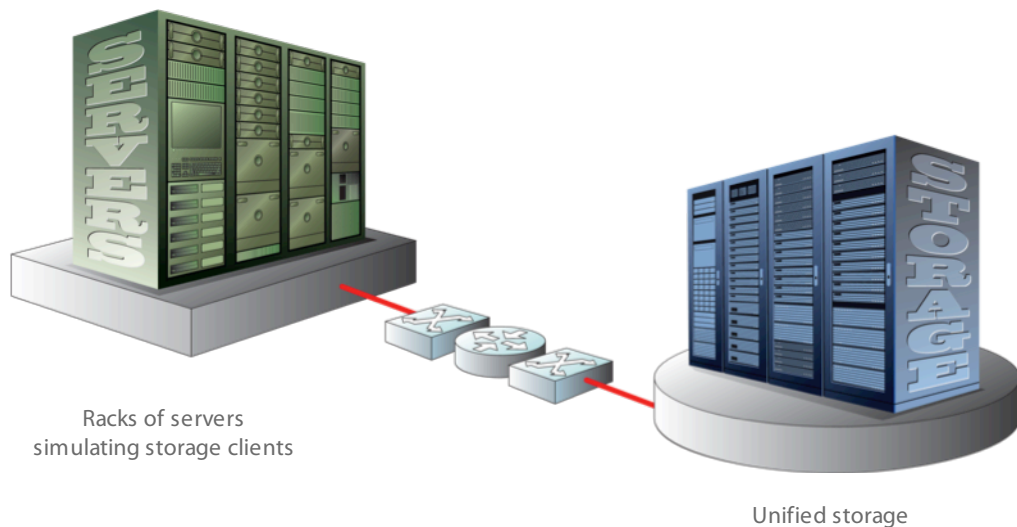
SwiftTest’s products are hardware and software solutions that include an appliance plus the SwiftTest Test Development Environment application for test configuration, test execution, and results analysis. For complete and up to date product specifications, refer to the SwiftTest website (www.swifttest.com). Below are the features which markedly distinguish the product from homegrown implementations:

- Unified, multi-protocol scripting and reporting
- Simultaneous support of file (NFS & SMB/CIFS), block (iSCSI) and cloud (HTTP) storage
- Ability to configure unique IP & MAC addresses for each simulated user
- During performance testing, correlated confirmation of data integrity (read after write)
- Repeatable test results
- Graphical test development environment
- Flexible automation

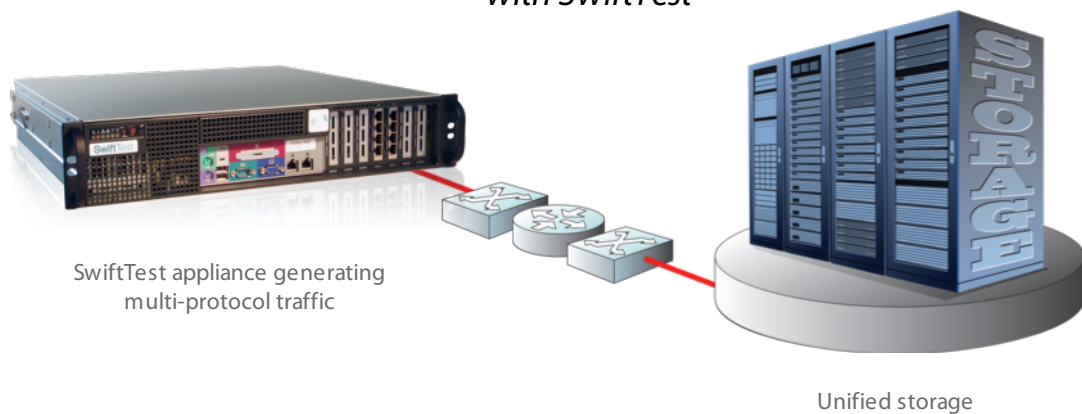
Eliminate multiple, disparate test systems using the SwiftTest appliance configured as a client.

In the following scenario, you're testing a storage device. In the past, you simulated client traffic using tens or even hundreds of servers. And the software you used to generate the load was difficult to maintain, inflexible, and required expensive licenses and hardware. With SwiftTest, you can simulate thousands of clients with multiple protocols.

Unified Storage Testing Today



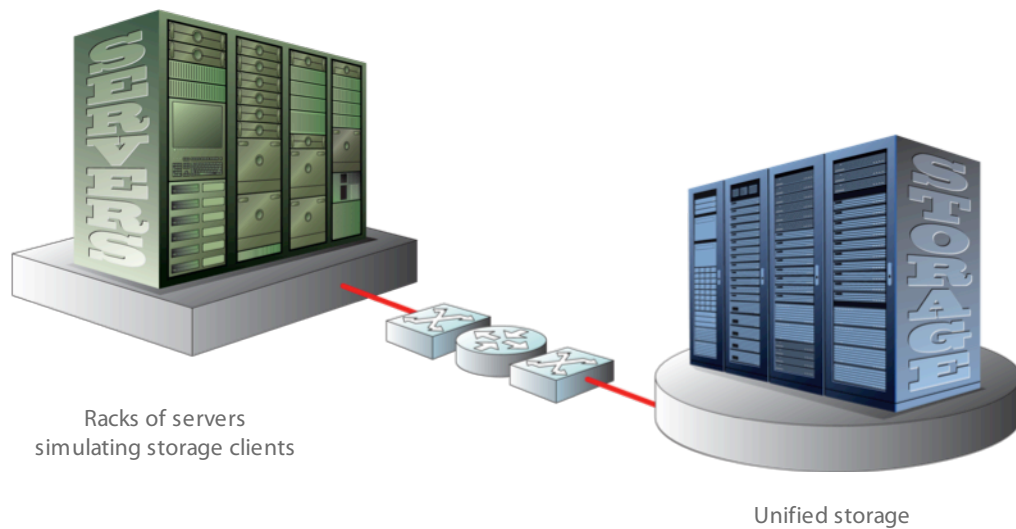
Unified Storage Testing with SwiftTest



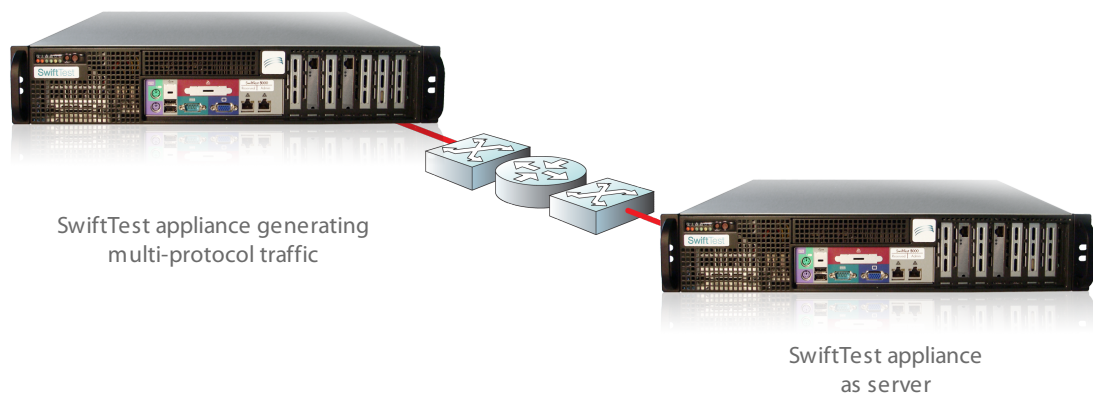
Test a storage-aware network device using the SwiftTest appliance configured as both a client and a server.

In the following scenario, you're testing a storage-aware network device. In the past, you needed a storage device plus tens or even hundreds of servers to generate the client traffic. With SwiftTest, you can replace both your client systems and storage device. Simply configure some SwiftTest ports as client and some as server, then use the SwiftTest Test Development Environment to configure and run your tests, and analyze the results.

Network Testing Today



Network Testing with SwiftTest



Appliance Design

The purpose-built, dedicated-design philosophy summarized in the word “appliance” has proven exceptionally effective in network protocol routing and switching as well as unified storage servers. Several pioneering high tech companies have enjoyed great commercial success by taking this approach, wherein specialized software runs atop commodity hardware to cost-effectively solve specific challenges.

SwiftTest products are likewise appliances, streamlined and purpose-built for the challenges they solve – comprehensive performance and functional testing of file servers and storage-aware network devices. The field-observed results have been:

- SwiftTest users report greater cost-effectiveness and ease of use than they’ve experienced with homegrown, general-purpose-OS-based implementations
- A modest amount of SwiftTest testing infrastructure can keep pace with the modern high-performance networks and storage servers being tested. To illustrate, a SwiftTest 3000, which is a 2U, 8-port system, can generate the load of dozens of white boxes.

SwiftTest is currently offering two high performance appliances:

SwiftTest 3000:

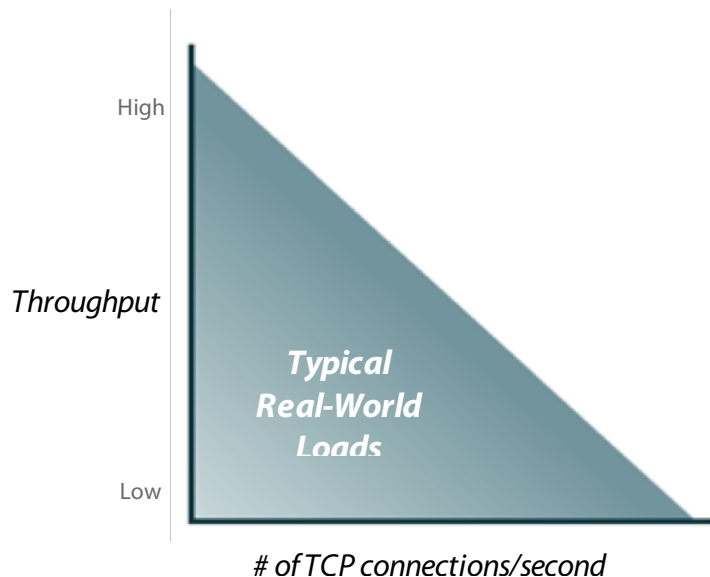
- 8 x 1GE ports
- Over 14Gbps of full-duplex traffic
- Millions of simultaneous simulated clients

SwiftTest 5000:

- 2 x 10GE ports
- Over 30Gbps of full-duplex traffic
- Millions of simultaneous simulated clients

Testing at Cloud Scale

Performance—and specifically price/performance—is clearly critical. As illustrated, a testbed can be configured for throughput (the vertical axis), or TCP connections per second (the horizontal axis), which is critical in high-user-count environments. No matter the environment, there is always going to be a tradeoff between the two; no server under test can deliver both simultaneously. The SwiftTest product line is able to saturate the server under test at both extremes of performance, and all points in between.



Advanced and Evolving

Customers profit from the cumulative years that SwiftTest's engineers have spent working in the testing, storage and networking disciplines. The products include features satisfying a broad spectrum of technically demanding and cost-conscious customers. By contrast, the corporation that "rolls its own" tooling cannot afford to dedicate an experienced engineering team full-time to the development, maintenance and functional evolution of such tools. Competitive companies drive to maximize the productivity of their engineers by focusing them on the most strategic initiatives possible.

As well, the corporation developing its own tools is limited to an insular view of the problem. Homegrown tools cannot mature by incorporating the testing lessons learned at other companies, some of whom may be competitors. As mentioned earlier, individual departments within a corporation may not compare notes to realize a unified testing practice. On the other hand, SwiftTest's engineers carry "Swiss passports." They are actively invited by customers and motivated by SwiftTest management to single-mindedly study and solve the challenge in all its variations.

SwiftTest Advantages

In summary, SwiftTest customers have gained the following advantages by deploying SwiftTest products:

- Unified testing at cloud scale: One test tool for all protocols with unified scripting and results enables shorter test cycles and effective use of engineering talent.
- Cost savings: SwiftTest costs less than the servers and software required to generate the same volume of traffic.
- Go green: SwiftTest uses less power, cooling, and rack space than comparable servers.
- Company expertise: Leverage SwiftTest testing, networking and storage protocol expertise.

ROI Analysis

SwiftTest customers have determined that SwiftTest products cost significantly less than equivalent homegrown solutions. Our customers have included the cost factors listed below to make their buying decision. In performing a ROI analysis, the reader should apply weighting factors, discounts, and lifecycle duration appropriate to their own circumstances. Recognize that some costs are not applicable in the SwiftTest case (e.g., test tool development engineer).

- Hardware acquisition cost (performance will dictate number of units required)
- Environmental costs (e.g., electricity, cooling, rack space)
- Application software acquisition cost
- Hardware maintenance fees
- Software maintenance fees
- Test tool development engineer(s), fully loaded including salary and benefits

Please contact SwiftTest (sales@swifttest.com) for a sample ROI spreadsheet.

Concluding Remarks

Unified storage is growing dramatically and is on an accelerating trajectory. The need to test multi-protocol storage systems at cloud scale is a given. But until now, no commercially available test tools have emerged to fill that need.

Grounded in the same appliance ethic espoused by major networking and storage vendors, SwiftTest technology satisfies this important need. It achieves low cost of ownership, while delivering capabilities that extend well beyond the capabilities of traditional homegrown solutions.

To learn more about product availability and pricing, please contact sales@swifttest.com.