White Paper

Running on Faith—The Risk of Inadequate Network Testing

Table of Contents

Executive Summary

In many organizations, testing the network infrastructure is at best neglected and at worst completely overlooked. This situation persists despite the serious implications to productivity, profitability, liability, and even reputation.

The roster of enterprises that have suffered costly and damaging lawsuits as a result of outages or security breaches would be shorter had those organizations invested in testing a vital, mission-critical foundation of the modern enterprise, the network infrastructure.

Best-practices enterprise network testing includes discovering performance limitations and validating vendor claims during acquisition, performing the due diligence of proof-of-concept for network designs or upgrades, and planning for headroom and growth as requirements change.

However, inadequate testing is worse than no testing at all, as it creates a blind spot, a false sense of security. There are three elements of effective testing, the kind of testing that protects an organization from unpleasant and costly surprises: Proper tools, methodologies, and expertise. These three elements create test realism, which is essential for meaningful test results.

Take it for a Test Drive

Did you test drive the last car you bought? Of course you did. How did you test it? Drive it through traffic? Get it out on the highway, see how it accelerates on the on ramp? See how the suspension feels on a bumpy street, how tight the steering is, how it handles curves? Count the cup holders?

Maybe you even took it to a mechanic and had it checked out. They probably checked the drive train, fluids, air conditioner, brakes, and tires.

But did you or your mechanic verify against published standards which airbags deploy based on the speed and angle of impact? Or how the anti-lock brakes handle hitting a patch of black ice on a curve at 60 mph? Or whether fuel performance matches the mpg claims of the manufacturer?

Probably not. After all, you're just the end consumer, not Consumer Reports or Car and Driver magazine. Plus, the National Highway Traffic Safety Administration enforces laws and regulations to make sure the manufacturers test all that anyway.

What about your network? Did you test the data center core switch before you bought it, or did you just rely on the manufacturer's claims about performance?

When it comes to networking equipment, there is no national body that regulates what happens under the hood that consumers can rely on to verify performance, availability, security or scalability. Additionally, network equipment vendors run differing regression test cases depending on the type of software build being released, feature vs. maintenance release. Maintenance releases are by far the most installed releases. Unfortunately, to save time and money for maintenance releases vendors tend to test those features with code modifications. This filters out defects within a feature, but often fails to find defects dependent on the interaction of multiple features.

Trust but Verify

If you don't test your network device, you're running on faith. Not to suggest that network equipment vendors intentionally falsify or overpromise on the performance of their products. But a vendor will understandably present their products in the best light, and cannot possibly test all combinations of features under all scenarios their customers use.

To know the true performance of a device, you have to know what testing is behind the numbers in the brochure.

What They Don't Say is as Important as What They Do Say

Yes, a switch may have a throughput of a billion frames per second with ten-microsecond latency as advertised on the data sheet, but is that for all frame sizes? For a realistic mix that reflects real-world traffic? Or just for a specific frame size that the device forwards efficiently? Can it sustain those numbers at line rate? Or as it approaches congestion? During topology changes or card failures?

The same uncertainties apply to network protocol scaling. If the collateral says the device can scale to 10,000 OSPF routes, is that limit still accurate when the ARP cache has several thousand host addresses? What about when an additional 2,000 IGMP multicast source groups are learned through PIM-SM?

You can be fairly certain that the reported specifications are real in the context of an isolated protocol running on the device, but unless you verify them under real-world conditions, you don't know the limitations and weaknesses that aren't listed on the data sheet.

Your Network is Unique

The vendor may have done rigorous testing and reported the performance of the device in their literature, but one thing you can be certain of: They didn't test their device on your network.

Your network was designed specifically to address the needs of your organization, its geographic reach, its mix of users, applications, and mission-critical processes, its requirements for response time, throughput, redundancy, and failover.

A data sheet won't tell you how the device will work in your network. With the right test platform, your network can be modeled in the test lab, something your networking vendor didn't do. Placing a new device in your network could expose breaking points in the device, or breaking points in your network design and implementation. That is critical information you should learn through testing before purchase and certainly before deployment, not from users complaining after you go live.

When Networks Go Bad

You see it in the news, another outage, another security breach. Even a short outage can rack up significant costs. Gartner estimates the low end of the range of hourly cost of downtime for computer networks at $42,000. For a financial services company that trades on Wall Street, the cost could be ten times that or more. The loss of HIPAA data due to a breach has cost some companies as much as $1,000 per record in the resulting lawsuits.

In April, 2011, the Sony Playstation Network was compromised, exposing the credit card numbers and other personal information of 77 million users, creating a 24-day outage, and costing Sony over $24 billion.

In October, 2011, an outage that left Blackberry users without services for four days crippled RIM's competitive position against other smartphone platforms and resulted in a class-action lawsuit.

And now individual consumers are getting in on the act. In April, 2012, a consumer in California sued Verizon for breach of contract and violation of consumer-protection laws when she was talked into upgrading to a faster account, but got only half the speed the plan promised.

Some organizations balk at the cost of testing, but the cost of failing to test can be much higher. The cost of a test platform that delivers test realism is a fraction of the amounts awarded in many lawsuits resulting from outages or security breaches.

When Testing Goes Bad (and Why)

Clearly testing is important, but what you test and how you test it is just as important. Inadequate testing can be worse than no testing at all when it results in a false sense of security.

The point of testing is to validate the suitability of a device or design for a specific application and to discover and correct problems before deployment. To accomplish the test goal, the test must stress the device or design in the same manner that the target environment will.

Production Network as Test Bed

To recreate the target environment, some organizations choose to test on the production network during off hours. After all, what could be more like the production network than the production network itself?

The reality is that the production network without normal traffic is not the target environment. In fact, it's nothing like the target environment. Tests you run on the production network during off hours will not produce meaningful results. You still won't know how the device will perform under stress in all the diversity and complexity of production traffic.

In fact, even when the target environment is live, it changes from moment to moment in a nondeterministic and irreproducible fashion, which makes it particularly unsuitable for testing. For a test to be effective, it must be repeatable. When problems are revealed during a test and a solution is proposed, the only way to demonstrate the effectiveness of the solution is to repeat the test, to recreate the exact environment that revealed the problem, which is very difficult if not impossible in a live environment.

Even more importantly, a live network is not suitable for testing unless you want to create the very downtime you're attempting to prevent by testing.

Effective testing that not only exposes problems but allows you to troubleshoot the cause, find a solution, and verify the solution, requires a test bed. The right test bed.

Inadequate Tools

You can't use a sledgehammer to fix a pocket watch, or jeweler's tools to work on an oil rig. Oftentimes when, despite prior testing, problems arise in a device or design after deployment, the failure can be traced to inappropriate tools.

Homegrown

Many organizations have a scripting guru on the team who is either asked, or volunteers, to create a script to test a solution. Why not? He or she is already there, an in-house resource who understands the project and the requirements.

The first drawback to creating a homegrown test tool is the loss of return on investment from the resource who was hired to do something other than create test tools, a time-consuming activity. As long as this person is contributing to test tools, their actual job is going undone or receives insufficient attention.

Second is a matter of expertise. This person may be an expert on networking, but the skill set to build and manage a network is not the same skill set required to test a device beyond simple connectivity and throughput. It could even be that the person simply has scripting expertise and very little networking knowledge and consults with another party with greater knowledge, taking up yet another resource's time.

Third is the matter of objectivity. The person who designed a system should not be the person who writes the test plan. Inherent assumptions made during design will be blind spots during test planning, and flaws in the design or implementation will remain undiscovered.

Fourth is the matter of sustainability. What happens when the network changes again, when new applications or protocols are introduced, and the tool must be updated? Will this person still be available or will other high-visibility projects take priority? When the person is promoted or is no longer employed with the company, who will have the skill set and free time to maintain and use the homegrown tool?

Fifth is the matter of test realism. A script-based tool running on a CPU won't have the power or sophistication to recreate the diversity and complexity of the production network. Even after extensive testing with a homegrown tool, you still won't know how the device will perform when it goes live.

Freeware

There are several open-source software-based test tools available. They are useful for troubleshooting problems, fine tuning protocol settings, and basic functional and throughput tests.

But they were not designed to predict performance, availability, security, and scalability of a device in a live network. They have the same limitations as homegrown tools, lack of power and sophistication to support test realism.

For test realism, you need a hardware-based tool.

Packet Blasters

A basic traffic generator, also known as a packet blaster, is the faster, hardware-based version of freeware test tools. It allows you to configure traffic and send it out at line rate to test device performance, usually in terms of throughput, latency, and packet loss. It may also support basic functional testing, as well.

A packet blaster is a step closer to test realism in that it will truly stress the limits of the device or network, but only in one narrow category, data transport. The basic ability to forward frames at line rate with no loss and minimal delay is important, but it is also basic functionality. For a networking device, data transport is like table stakes in poker. It will get you in the game, but it's not enough to win.

Even a simple Layer 2 switch has much to contend with, such as spanning tree protocols, redundancy and failover schemes, and quality-of-service queuing. Add Layer 3 functions and protocols and the complexity grows exponentially.

Packet blasters are not designed or intended for testing the full functionality of a network device and are consequently insufficient as a basis for a test bed that supports test realism.

Inadequate Methodologies

The problem with the previously mentioned approaches to testing is that they don't go far enough. If you want to know how a device will perform in a live environment, it is not enough to drop it into the environment, because you don't know the traffic conditions at any given time in the environment and, more importantly, those specific conditions cannot be recreated on demand.

Repeatability is one key to meaningful testing. It enables you to reproduce the precise conditions that caused failure in a device, troubleshoot the root cause, and then test the solution to verify the problem was actually solved. This is the whole point of testing.

Beyond repeatability is test realism.

Test Realism: Emulation Versus Simulation

The ability to accurately evaluate a device is based on the simple but essential element of test realism. Realistic testing means re-creating the environment that the device lives in, from the provider to the customer, in all its dynamic and daunting complexity.

Test realism comes down to the difference between simulation and emulation. Simulation attempts to imitate the behavior of a system over time by creating an approximate mathematical model of the system based on a set of assumptions. In contrast, emulation replaces a part of the system and performs in nearly the same way as the element it replaces. It adjusts dynamically to a changing environment and responds to actual stimuli from the system it interacts with.

- Simulation is appropriate for analyzing a system to infer predictions about how it might behave
- Emulation is required to see what will actually happen in a given situation, to know how a device will respond under real-world conditions

There are three essential elements of test realism.

Real User Behavior

Users are as unique as their fingerprints. They vary significantly in how long and in what manner they navigate through an application and how they respond to sluggish performance, picture and voice dropouts, dropped calls, and other problems. They violate usage and security policies in different ways. They find unique ways to break your system.

> *Realistic testing means the flexibility and sophistication to emulate a wide range of user behavior, both benign and malicious.*

Hitting a device with a mix of traffic types (say, mixing HTTP requests/responses with multicast joins and leaves, and P2P file sharing or gaming traffic) isn't emulating user behavior. It's a simulation, just hitting the system with a mix of traffic, static packets that don't respond statefully to incoming responses from the device under test.

Emulating real user behavior means supporting stateful traffic that emulates how a user operates, including think time, click-through, abandon, channel surfing, etc. It means good users and malicious users simultaneously attempting to achieve their good and bad goals.

User-centric traffic reveals the performance of the device in a real environment. Queues, buffering, and other mechanisms behave differently depending on the order and the nature of the transactions. An arbitrary (and artificial) static mix of messages, or even a dynamic mix of messages that doesn't account for the stateful nature of user connections, will not stress the system the way real users do. As a result, failure points can remain undetected until real users start using it, exactly the wrong time for that to happen.

Real Converged Traffic

Real networks are characterized by diversity. Mobile and fixed-line voice and data, residential video, and MPLS-based VPNs sharing the same network. The different types of traffic carried on a network have different characteristics and requirements, but they all travel on the same path, dependent on class-of-service (CoS) and differentiated traffic rules to keep everyone happy.

> *Realistic testing means the power to create line-rate, fully-emulated, stateful traffic across hundreds of ports.*

Real converged traffic not only means a realistic mix of traffic types, but also realistic traffic encapsulation. For example, if the deployed system tunnels user PPP sessions over MPLS, then testing PPP setup-teardown rate and throughput performance without MPLS is not real converged traffic. It's a dangerous shortcut that will mask problems your users will discover after deployment. Real converged traffic means emulating the actual deployed topology, regardless of how complex or simple, including all encapsulations.

Real Network Conditions

The network creates time-varying conditions that are linked to a complex set of conditions, influenced by dynamic table updates, signaling protocols, queuing algorithms, buffering, traffic management, device management and policing policies, malicious attacks, EMI and other environmental factors.

> *Realistic testing means the power and complexity to create the dynamic, time-varying conditions found on deployed production networks.*

Real network conditions can't be emulated through static rates of delay/loss or distribution-based mathematical models of impairment. Real networks don't introduce impairments at fixed rates or follow neat curves. They behave in seemingly non-deterministic ways due to the number of factors affecting them. Testing under real network conditions means emulating this complexity to discover issues before the real network finds them.

Test Scope

Using test realism as the foundation, effective testing expands the scope of what is tested to address all the elements that affect the behavior of a device in a live network. These elements are performance, availability, security, and scalability, known collectively as PASS. The PASS test methodology takes all of these elements into consideration to completely characterize a device or system for a given application in the context of a production network in terms of user behavior, traffic mix, and network conditions.

For example, consider the four PASS components in the context of a data center.

- **Performance:** Optimize services and infrastructure to maximize user experience
- **Availability:** Ensure high availability in daily operation and under disaster conditions
- **Security:** Eliminate vulnerability and exposure between users and applications
- **Scalability:** Validate responsiveness as demand varies according to user needs

The methodology incorporates a wide range of context-aware use cases to assess the application components and network elements of a data center. For example, at the service level, PASS assesses user quality of experience under realistic and peak loads, scaling to multi-user loads, under the stress of failover scenarios, and while subjected to threats mixed with encrypted traffic.

At the infrastructure level, PASS assesses quality of service performance for the network elements (high-density 10/40/100 Gb Ethernet, converged FC and Ethernet), the application elements (firewalls, intrusion prevention systems, WAN accelerators, proxy servers, and others), and the virtual elements (virtual switches and virtual appliances), end to end.

Service Level Agreement Verification

A service level agreement (SLA) is a basic, quantifiable definition of what the subscriber can expect and the penalties he can collect if the metrics are not met. As such, it represents hard costs at risk if a violation occurs and is a prime candidate for test realism. Basic packet blasting can characterize a circuit in the terms that appear in many SLAs, but it can't recreate the environment that will exist on the connection once it goes live with a mix of data, voice, and video traffic, control and data plane traffic, and stateful traffic setting up, maintaining, and tearing down connections.

Going Beyond QoS to QoE

While real-time services are built on a foundation of a quality of service (QoS) enabled network, the real deliverable is not a set of priority rules and metrics. It's a high quality of experience (QoE) for the user. Deterministic legacy test methodologies deliver valuable information about individual components or functions, but good QoS numbers alone are no guarantee of QoE, no more than the performance of a file transfer indicates the transaction response time an interactive user will experience. A test methodology built on test realism goes beyond validation of discrete components and functions and recreates the complete environment in which new services must perform.

WAN Optimization

Test realism is particularly important to determine the best configuration of a WAN optimization device and verify that it meets or exceeds the expected performance, availability, and scalability goals. Combining a test methodology that incorporates realistic user behavior, traffic mixes, and network conditions with a test bed that has the power to emulate a real-world WAN produces a controlled, precise, and repeatable environment that makes meaningful test results possible.

Consider, for example, the experience of one North American organization that partnered with a team in Asia. They discovered sending documents over the WAN took 160 times longer than over the LAN, as transfer time increased from 30 seconds to 80 minutes. By testing a solution that employed TCP optimization and compression, they were able to reduce the overseas transfer time to 2 minutes, a 98 percent reduction that improved productivity dramatically.

Routing and MPLS

Nowhere is a methodology that validates performance, availability, security, and scalability more important than in routing. Demand for content from the always-on, always-connected generation places increased pressure on the edge and core. Today's network is a mix of very high performance, QoS-enabled IPv4 and IPv6 routing. Scale and performance is denser and more complex, with protocols stacked and interlinked. Older methodologies for testing routing, such as treating each protocol as an independent stack, no longer apply.

An effective test bed and methodology allow for real-world, object-based modeling of the network infrastructure and the ability to simulate actions over time in a deterministic, reproducible manner.

Security

Traditionally, security is placed in strategic physical locations, such as at the WAN edge where requests and traffic from the Internet can be filtered and decrypted. However, geographic locations of physical servers have less meaning in today's world of virtualization, as users might be tapping resources from VMs located on one of any number of servers or even data centers. When it comes to virtual security, test realism not only means delivering the maximum number of new connections per second and firewall bandwidth throughput while blocking threats and malicious traffic, but also being cloud-aware.

Inadequate Expertise

Just because you trade your minivan for a Maserati doesn't mean you'll be able to drive it like Mario Andretti. The best tool in the world will be ineffective in inexperienced hands.

While your IT technicians and network engineers are experts on designing, deploying, and maintaining a network, they are not likely to be experts in network test methodology. Relying on in-house engineering resources, which may lack the required up-to-date expertise and hands-on experience, can produce unreliable test results. An experienced test engineer is critical, especially when deadlines are short, budgets tight, and margin for error is zero. The cutting edge of networking is a moving target and requires the attitude of a lifelong learner. The world of network test methodologies is no different.

Some test platforms have built-in applications and wizards that incorporate industry best practices derived from standards-based test methodologies and decades of experience. Using such tools can produce more reliable and relevant results in a fraction of the time of manual testing.

Automation

Scripting and automation expertise is arguably the preeminent skill in reducing customer-reported bugs, expanding test coverage, increasing productivity and lab usage, reducing operating costs, and collapsing test cycles. Consider a few real-world examples:

- One lab testing their 10 GbE MPLS architecture with 20 different test cases freed up 3.5 engineers to add more test cases and expand test coverage through automation.
- Another lab testing RP switchover convergence took 35-45 minutes to run a single test case. With more than 20 iterations for each test, one test could take an entire work day with an engineer babysitting it to keep it going. Test automation eliminated the need for user interaction, a single change that produced a 40 percent productively gain–nearly half a person's daily activity recovered by automating a single test suite.

It would be difficult to find any other single factor that increases productivity by 40 percent. Some test engineers fear scripting themselves out of a job. The reality is that automating test cases doesn't make an engineer expendable, it increases that engineer's value. How? By escaping repetitive (but important) tasks to take on other high-value work that is being neglected, work that expands skill sets and helps the company bring products to market faster, improve product quality and keep or gain a competitive edge.

Previously we wrote about common shortcomings of test planning: Inexperienced staff, inadequate tools, and a lack of objectivity when developers test their products. Testing, like accounting, requires the objectivity and accuracy that comes with separation of duties. But many organizations can't afford a team of unbiased, dedicated test engineers.

Best practices recommend an experienced external organization for testing to provide a level of expertise not likely to be duplicated in-house or with a consultant. For example, in a recent project a major financial trading site flipped the switch on a new datacenter with zero problems in the first 48 hours, a first for them. They were one of only two trading sites in the US that were able to keep up with the pre-April 15th trading peak, avoiding the delays and outages experienced by other sites. They accomplished this impressive feat through the use of an experienced, independent test team conversant in current best-practices test methodologies.

About Spirent Communications

Spirent Communications (LSE: SPT) is a global leader with deep expertise and decades of experience in testing, assurance, analytics and security, serving developers, service providers, and enterprise networks.

We help bring clarity to increasingly complex technological and business challenges.

Spirent's customers have made a promise to their customers to deliver superior performance. Spirent assures that those promises are fulfilled.

For more information, visit: www.spirent.com

Choosing a Test Platform

When it comes time to select a test platform, with or without professional services support, there are several things to consider.

Testing should be a core competency of the vendor, not an open-source freeware or an ad hoc solution offered on request. This means the partner is an established global name in the test and measurement industry with verifiable experience and expertise.

The platform must have the power and sophistication to support all the elements of test realism: Real user behavior, real converged traffic, and real network conditions.

The platform must support robust, standards-based test methodologies that stress the performance, availability, security, and scalability of the device or system under test.

The system should support the latest automation tools and built-in GUI-based tools for simplifying and automating standards-based and customizable test cases without extensive scripting knowledge as a requirement.

Choosing a Test Partner

In the absence of a dedicated in house test lab and objective test engineers, you need a neutral party without a motivation to influence the test results, as may be the case with a system integrator testing their own solution. Some systems integrators who do their testing in house have a vested interest in delivering positive results. The testing team should have a holistic understanding of networks, be able to articulate testing benefits and ROI vs. risk, and have an established delivery process. A test partner's qualifications are further strengthened if they have extensive expertise in lab and test automation and can provide references of successful engagements. The most advantageous test partners that provide test rental equipment are those who supply devices manufactured by the same company.

spirent™
Promise. Assured.

Contact Us

For more information, call your Spirent sales representative or visit us on the web at www.spirent.com/ContactSpirent.

www.spirent.com

Americas 1-800-SPIRENT
+1-800-774-7368 | sales@spirent.com

US Government & Defense
info@spirentfederal.com | spirentfederal.com

Europe and the Middle East
+44 (0) 1293 767979 | emeainfo@spirent.com

Asia and the Pacific
+86-10-8518-2539 | salesasia@spirent.com

Rev C | 01/19