

# Effective Diagnostic Strategies for Wide Area Networks

A white paper assembled from NSF proposal ANI-0334061

This document contains unpublished research plans.  
Please do not circulate without explicit permission from the authors.

This paper is significantly altered from the submitted proposal.

Matt Mathis <mathis@psc.edu>  
Peter O'Neil <poneil@ucar.edu>

November 5, 2003

# DRAFT

## Project Summary

We believe that a critical missing piece of the end-to-end network performance puzzle stems from the fact that current network diagnostic strategies do not adequately take into account the effects of path delay. We propose to develop extensions to existing diagnostic tools which will take path delay into consideration effectively, compensate for a variety of delay times, and test the effects of these new diagnostic tools with network users and operators using actual high performance applications.

We now recognize why end-to-end Internet diagnosis is such a difficult problem; the symptoms of nearly all flaws scale with path delay and the symptoms that scale with delay cause classical diagnostic strategies to yield misleading results. Classical diagnostic tools and strategies yield confusing results because merely changing the path delay changes the symptoms of the flaws, even though the flaws are not in the additional part of the path.

Our new diagnostic methodology uses several different techniques to compensate for the effects of path delay on diagnostic results: Emulated Delay, which uses specialized software to buffer and delay packets as though they traverse a long path, Parametric Model Scaling, a method of verifying that a short path can support TCP bulk transport over a long path by verifying that the short path implements the required properties for TCP to meet the end-to-end performance objective, and "Scenic" Virtual Private Networks, where packets can be delayed by using various IP tunnels or virtual private networks to send them on long "scenic" alternate routes. These techniques will be used in conjunction with existing diagnostic tools and strategies.

We will develop a suite of enhanced diagnostic tools that compensate for the effects of path delay. This suite will be targeted at several different audiences: true end users, application developers, network administrators and diagnostic experts or tool developers. The suite will be built from a small kit of core components that implement the previously described techniques that compensate for path delay, in conjunction with some simple diagnostic applications.

In summary, we believe that there are a vast number of performance problems in the greater Internet that remain hidden because they do not exhibit any symptoms when tested on short, low delay, paths. Our goal is to revolutionize Internet performance diagnosis by properly compensating for delay and eliminating false positive diagnostic results for short paths, resulting in effective diagnostic strategies for Wide Area Networks.

## **Introduction and Background**

We believe a key missing piece of the end-to-end performance puzzle is that the current set of diagnostic strategies do not adequately account for the effects of path delay. We propose to develop extensions to existing diagnostic tools which will effectively take path delay into consideration, compensate for a variety of delay times, and test the effects of

these new diagnostic tools with network users and operators, using actual high performance applications.

If you ask moderately savvy network users how quickly they can copy data across campus, you are likely to hear data rates above 10 MBytes/s. If you ask about the expected performance across the continent, you are likely to hear expectations for data rates well below 1 MB/s. These numbers do not necessarily strike us as strange until you realize that for High Performance Network (HPN) attached campuses; the slowest link is likely to be common to both paths - the first few yards, from the office to a wiring closet. Furthermore if you ask those people responsible for various components of the application and network path, e.g. the application programmer, system administrator, the campus network administrator, the local interconnection or GigaPoP administrator, and the transcontinental backbone operator, each one can claim to be able to demonstrate why their individual component and link path is not the problem, and how some other part of the system must be responsible for the perceived problem.

Due to recent insights learned from the Web100 and Net100 projects, we can show that the "missing piece" of the performance diagnostic puzzle is that the symptoms of most application and network defects scale with increasing path delay. For example, a minor defect in a campus LAN might have an insignificant or negligible effect on an application running on a 1 ms path across campus. However, that same defect has a greater impact on performance when running on a path across the continent. To reiterate, suppose I have an application that works fine on a 1 ms path, but performs unusably poorly at some remote location with a 100 ms path (See Figure 1). Given the current state-of-the-art in diagnostic tools, it is likely that those investigating the problem will conclude that the flaw lies somewhere along the additional 99ms wide-area path. Given the high degree of information uncertainty and lack of diagnostic tools, which incorporate the effects of delay, such assumptions, flawed as they may be, are reasonable given the limited information circumstances. In this project, we propose to develop new diagnostic techniques, which includes; a suite of diagnostic tools and techniques to test applications locally with a 100 ms virtual path, and then test each successive segment of the actual path extended with a virtual path to a total path delay of 100ms (See Figure 2). Such testing is expected to expose otherwise hidden flaws and impediments which reduce performance, since each component of the path and application can be tested in a context that is equivalent to an ideal end-to-end path, while ruling out other potential flaws.

These new diagnostic techniques entail adding several different techniques to classical diagnostic and debugging tools which will compensate for the effects of delay. Some techniques are as straightforward as adding a "virtual path" to a real path. This "virtual delay" can then be used in combination with conventional debugging techniques to predict performance over paths with longer delays. In most cases, we will be developing extensions to existing tools and techniques with methods which incorporate consideration for the effects of the variable delay, rather than initiating creation of completely new diagnostic tools. Although we believe such extensions can be applied broadly to nearly all tools, we propose to focus our efforts on a carefully selected set of tools.

We propose to develop a suite of enhanced diagnostic tools built on this new diagnostic technique. Most of the tools will use a web browser to connect through the network to a sophisticated diagnostic server. We propose to design, implement, and test the diagnostic software suite and deploy a prototype diagnostic infrastructure. Note that debugging local problems requires that these tools be made available to users and site network administrators while the ability to effectively debug wide area problems requires a deployed diagnostic infrastructure.

The proposed tools will be designed to work in concert with the different views and levels of concern of our various user communities. For example, tools designed to support end users will provide appropriate initial starting information for network administrators. The specific audiences we will address are: end users, application developers, network administrators, and network experts or tool developers.

The proposed plan will follow several stages. First, complete the core diagnostic elements. Second, package the elements into diagnostic suites which are specific to each audience. Third, validate with early adopters at PSC and NCAR. Lastly, deploy the diagnostic audience elements at selected affiliate locations. Note that for different audiences, deployment and feedback phases will be staggered: network savvy staff tools will be released first, followed by the tools which will be designed for application developers and end user researchers.

The core diagnostic components will include: Dummynet diagnostic kit, scenic Virtual Private Network (VPN) diagnostic kit, high performance discard servers, and path diagnostic servers. We will extend and enhance these core components and then integrate them together with a simple graphical user interface, which will serve as an end-to-end diagnostic suite aimed at several different audiences. In most cases, part of the diagnostics will be anchored in the network, as in the ANL diagnostic server, which has grown out of the Web100 instrumentation.

## **Results of Prior Work**

Pittsburgh Supercomputing Center (PSC) and the National Center for Atmospheric Research (NCAR) have long and successful track records, both individually and in collaboration, in high performance networking. For more than six years, PSC and NCAR collaborated on the Engineering Services component of the NLANR project (NLANR ES) ([www.ncne.nlanr.net](http://www.ncne.nlanr.net)). Gwendolyn Huntoon and Matt Mathis from PSC were co-principal investigators on this project, which was funded through NSF Cooperative Agreement ANI-9720674. Through this project, PSC and NCAR provided the engineering and technical support to enable more than 170 NSF-funded High-Performance Connection (HPC) sites to connect to and use the high-performance research network backbones, services and technologies. NLANR ES staff at PSC and NCAR also provided expert consulting to HPC campus network engineers, developed documentation on new and emerging HPN technologies, and provided tools for diagnosing and understanding network performance. One of the most successful components of the NLANR ES project was a series of technical workshops, known as the

"Joint Techs" Workshops, that NLANR ES staff organized and sponsored in conjunction with Internet2 ([www.ncne.nlanr.net/training/techs/](http://www.ncne.nlanr.net/training/techs/)). These workshops have become the primary forum for disseminating and exchanging information in the high-performance network community.

The Web100 Project ([www.web100.org](http://www.web100.org)) is a collaboration between PSC, NCAR, and the National Center for Supercomputing Applications (NCSA). Initial development of Web100 is funded by a three-year grant from the National Science Foundation (NSF) through grant (ANI-0083285). Work under this grant includes the initial development, deployment and documentation of core Web100 instruments and library functions for the Linux Operating System.

The Net100 Project ([www.net100.org](http://www.net100.org)) is a collaboration between the PSC, NCAR, Oak Ridge National Laboratory (ORNL) and Lawrence Berkeley Laboratory (LBL). Net100 is a DoE sponsored project that extends and enhances Web100 functions for the DoE environment.

PSC also developed the base model for TCP bulk performance [MSMO97] and has made substantial contributions to the IETF IP Performance Metrics working group [RFC2330, RFC3148, MHRRS03, MHR03]. Our ongoing research on TCP congestion control has resulted in the development of several technologies that can improve TCP performance in high-performance environments [RFC2018, RFC2525, RFC2760, RFC2883, MHRRS03]. PSC has also done groundbreaking work in TCP autotuning [SMM98]. In addition to its formal research contributions to understanding Internet performance, the PSC also maintains a number of online resources about Internet performance (<http://www.ncne.nlanr.net/TCP/index.html>). PSC's performance tuning web page; [http://www.psc.edu/networking/perf\\_tune.html](http://www.psc.edu/networking/perf_tune.html) is an authoritative source for information on how to hand-tune a wide variety of operating systems for use over the national HPN backbones.

NCAR has been the lead participant in a number of projects that have attempted to fully utilize high-performance networks. These projects include distributed climate modeling over the vBNS (DCSL: <http://www.scd.ucar.edu/vg/DCSL/DCSL.html>) and distributed mesoscale modeling over a high-speed satellite link (COOP-3D: <http://www.scd.ucar.edu/zine/96/spring/articles/SC95/coop3d.html>). NCAR is currently participating in the DOE Earth System Grid Project that is attempting to provide a high-performance networking environment for distributed modeling, data storage, and visualization (<http://www.scd.ucar.edu/css/esg/>). NCAR also has extensive experience in installing, maintaining, and optimizing high-performance LAN and WAN technologies including ATM LANE, CLIP and Gigabit Ethernet. NCAR also hosted an NSF workshop to determine atmospheric science high-performance networking needs now and into the future. (<http://www.scd.ucar.edu/nets/projects/NETSprojectplans/1999.complete.projects/nlanr/>)

## **Technical Section**

## Problem statement and approach

Due to our work on the Net100 and Web100 projects, we now recognize why end-to-end Internet diagnosis is such a difficult problem; the symptoms of all flaws scale with path delay, and those symptoms that scale with path delay cause classical diagnostic strategies to yield misleading results.

### The intrinsic sensitivity to delay

The following examples illustrate three different flaws in network and applications that have symptoms that scale with delay.<sup>1</sup>

- Consider an application that uses a simple request-response protocol. Suppose that typical user action causes 50 request-response transactions. On a 1 ms path, this adds 50 ms to the user response time, which is below human perception. On a 100 ms path the same 50 transactions add 5 seconds to the user response time, which might not be acceptable. A better application design might use fewer transactions, or perhaps overlap them so they can proceed in parallel.
- Consider a non-autotuned TCP connection with fixed 32 kByte socket buffers. Since reliably delivering data to the application requires sufficient buffer space to hold one round trip of data, this TCP would be limited to 32 kByte of data in flight in the network at one time. This is enough data to support more than 200 Mb/s on a 1 ms path, but only about 2 Mb/s on a 100 ms path.
- Consider the situation in Figure 1, where the flaw introduces 1% packet loss on a 100 Mb/s network (using 9kJumbograms). The data rate calculated for the 1ms path from the macroscopic performance model [MSMo97, PFTK98] would be about 500 Mb/s, so the calculated data rate will be limited by the link speed and not the background loss rate. However on a 100 ms path, the calculated rate drops to about 5 Mb/s. Furthermore, if you invert the TCP macroscopic model to estimate the maximum loss rate, you find you need approximately 0.0003 % loss. This is because the required loss rate is proportional to the square of round trip time (which is proportional to the square of the window size in packets).

An alternate way to think about the effects of path delay is that when the path delay is small, TCP can quickly compensate for flaws in the network or application. With longer path delays TCP's ability to compensate for flaws is diminished, and the flaws show correspondingly more serious symptoms.

All three of these examples have the potential to cause confusing results with classical diagnostic tools and strategies because merely changing the path delay changes the symptoms of the flaws, even though the flaw is not in the additional part of the path.

---

<sup>1</sup> For clarity of illustration we do not distinguish between one-way delay and round-trip time. In an actual calculation you would have to be sure to have the right factors of 2 in the proper places.

## Intrinsic sensitivity to delay confuses classical diagnostics

This intrinsic sensitivity to delay creates a confusing situation when trying to diagnose a network problem. Consider Figure 1: When an application is run from the local client (LC) to the server (S), it may get acceptable performance even though the path has a flaw. However, a remote client (RC) on a 100 times longer path is likely to experience unacceptable performance because the symptoms of the flaw are 100 times larger. In this situation it is only natural to assume that the flaw is in the backbone, because introducing the backbone to the path seemed to cause the failure. However, it was really the delay of the backbone magnifying the symptoms of an existing flaw.

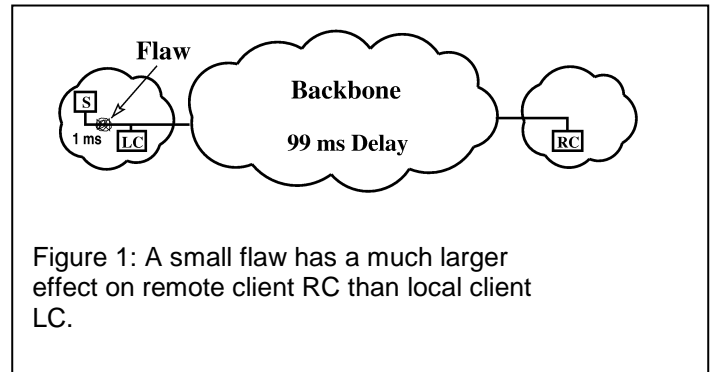


Figure 1: A small flaw has a much larger effect on remote client RC than local client LC.

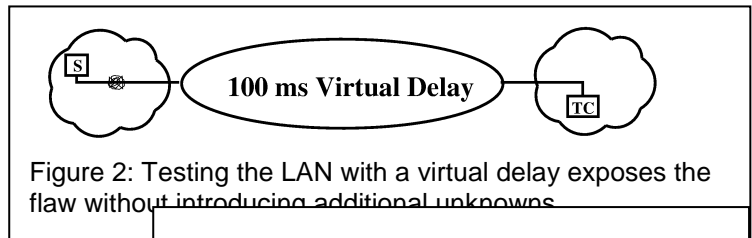
The goal of this project is to develop diagnostic tools and techniques to test applications and path segments in simple local short delay environments, and use these measurements to predict how they will perform on a real path with high delay. Diagnostic tools with the ability to compensate for the effects of delay will enable conventional debugging strategies such as testing successively smaller path segments or replacing incremental components to accurately find flaws, even when the flaw shows no symptoms on a short path.

Through our extensive background on TCP tuning and instrumentation [MMR03, SMM98, Web100, MHR03, MHR03a, DMT02, Net100], building tools [Mat94, Mat96, Mat00, RFC3148, SM01], educating communities on proper tool use (NLANR ES), and our collaborations with other organizations working on related problems (NLANR Distributed Application Support Team (<http://dast.nlanr.net>), NLANR Measurement and Network Analysis Group (<http://moat.nlanr.net>), and Internet2 Engineering, Performance and Measurement teams) we have a pretty thorough understanding of "TCP performance tuning" (as network diagnosis is frequently misnamed). We are using our expertise to design a suite of tools that will provide complete coverage of all known types of network flaws. We believe that there are a vast number of performance problems in the greater Internet (Internet 1, Internet 2, and the attached campuses) that remain hidden because they do not exhibit any symptoms at all when tested on short, low delay paths. Our goal is to revolutionize internet performance diagnosis by properly compensating for delay and eliminating false positive diagnostic results for short paths, resulting in effective diagnostic strategies for Wide Area Networks (WAN's).

## The General Method

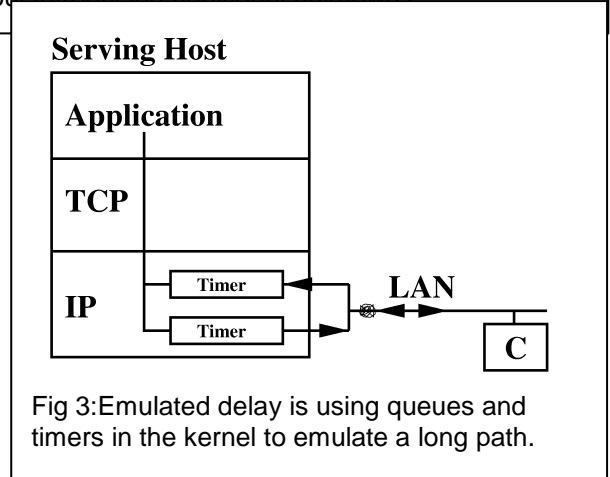
Our proposed diagnostic technique entails using several different techniques to compensate for the effects of path delay on diagnostic results. Some techniques are as straight forward as adding a long "virtual path" into a local diagnostic test as shown in

Figure 2. A test client (TC) connects to the server (S) through a virtual delay that emulates an ideal long path. This test should reveal the true performance impact of any hidden flaws without introducing additional unknowns. This approach is particularly powerful when investigating flaws that are local to the server, such as in the application itself, the server's TCP stack and nearby campus network infrastructure. Other techniques may be more appropriate for detecting flaws elsewhere in the network.



We will investigate at least three techniques for compensating for delay: emulated delay, parametric model scaling and scenic Virtual Private Networks (VPNs).

1. Emulated delay uses specialized software to buffer and delay packets as though they traverse a long path, as shown in Figure 3. The packets are queued in the kernel and scheduled to be processed or transmitted on the basis of timers in the operating system.



Although emulated delay is conceptually the simplest method to compensate for delay, it has two practical problems: first, the required code has to be installed in the operating system, which typically requires root privileges to enable and configure it. This makes it unavailable to most end users. Also the smallest time steps available to the operating system timers are much coarser than typical inter-packet intervals, so the emulated delay has to send bursts of packets at the approximate correct average spacing. While this presents a major accuracy problem for the research community when using emulated delays for protocol experiments, it actually makes diagnostic tests more stringent since the network has to tolerate burstiness that may not be present over the real path.

2. Parametric model scaling is a method of verifying that a short path can support TCP bulk transport over a long path by verifying that the short path implements the required properties for TCP to meet the end-to-end performance objective, as calculated by the TCP performance model [MSM097, PFTK98]:  

$$\text{rate} = (\text{MSS}/\text{RTT}) 0.7/\sqrt{p}$$

Figure 4 illustrates this approach. A simple diagnostic application is run on

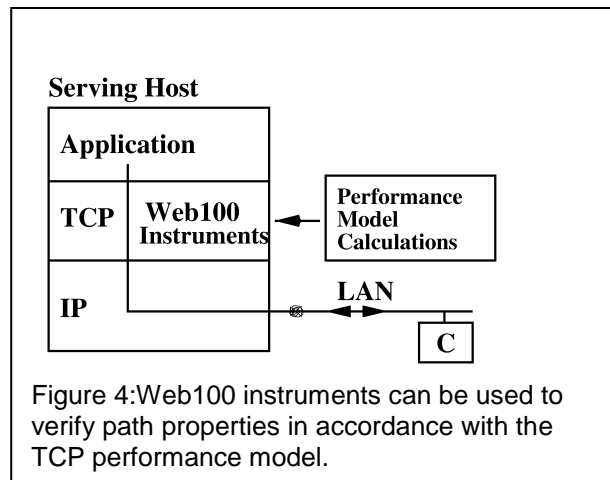


Figure 4: Web100 instruments can be used to verify path properties in accordance with the TCP performance model.

the short path. Web100 instruments are used to observe the Round Trip Time (RTT or twice the path delay), loss rate ( $p$ ), and Maximum Segment size (MSS) of the path and the model calculates if it meets the performance objective of the long path. For example if you want to attain 200 Mb/s using 1500 byte packets over a 50 ms path, you will need to have a total end-to-end loss rate which is slightly below 1 part per million. If the local network is losing more than 1 part per million, it can not support the higher data rate on the longer path, even if it can easily support more than 200 Mb/s for local TCP connections.

The Web100 instrumentation required for this approach is not yet standard (But see the TCP ESTATS MIB [MHRRS03]), so for the time being this requires that the Web100 modifications be installed in the kernel. Therefore, model scaling tools can not be readily installed by non-expert users.

We can deploy diagnostic servers in the network and make them available through a simple web based user interface. These diagnostic servers would use parametric model scaling to perform segment by segment testing of long paths. This approach will be described in much greater detail below.

3. Packets can also be delayed by using various IP tunnels or VPNs to send them on long "scenic" alternate routes as shown in Figures 5a-d. This approach is most useful for bench testing applications that are going to be deployed in some specific remote location, such as on a show network (See 5a). The application developer, working with both the client and server in a lab can configure tunnels and IP routing such that either the forward or reverse paths traverse the backbone (to a tunnel endpoint T near the future remote location, Figures 5b and 5c respectively). It is also possible to make both the forward and reverse paths traverse the backbone, for an effective path length which is twice as long as the path in Figure 5a.

All of these "scenic" routing approaches permit the application to be tested in a local lab, while using portions of the actual long path to a remote location. Note that these approaches have a number of potential problems associated with security and

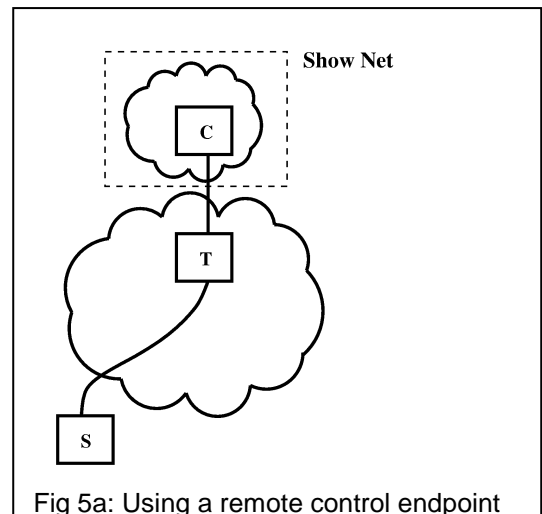


Fig 5a: Using a remote control endpoint

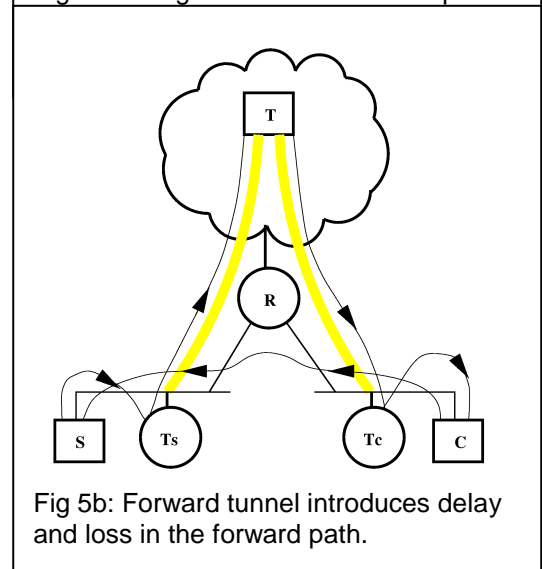
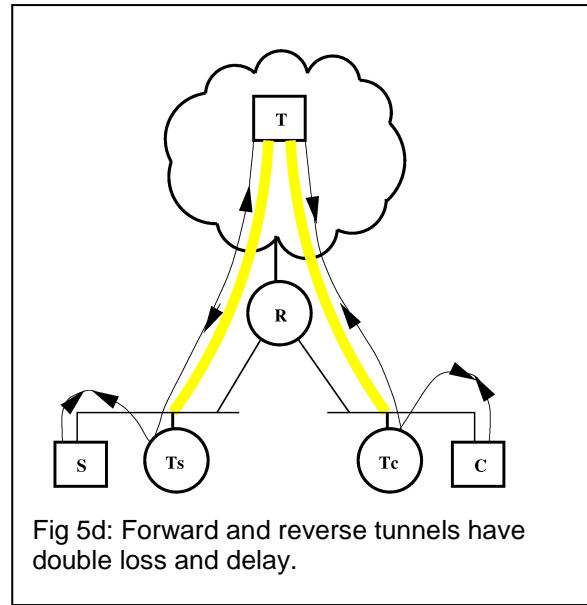
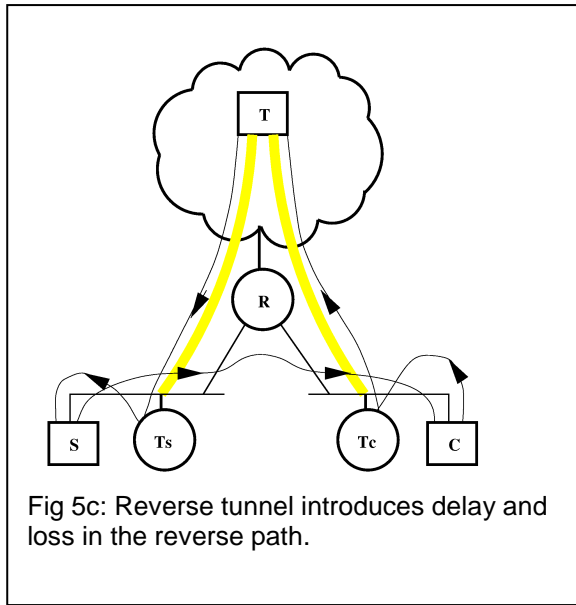


Fig 5b: Forward tunnel introduces delay and loss in the forward path.



possible misuse. The crucial design issue will be managing remote privileged access to the far tunnel endpoint, such that it cannot be used for nefarious purposes. We will start by relying on NCAR and other collaborators to manually configure remote tunnel endpoints. Once we understand the performance and diagnostic capabilities of this approach, we will look into better methods for controlling access, such that people do not have to have a privileged associate at the remote location to safely configure the tunnel.

The previous three techniques to compensate for the effects of path delay are not in and of themselves network or application diagnostics. They are used in conjunction with existing diagnostic tools and strategies. Also note that nearly any classical diagnostic tools and strategies could be adapted to use one of these techniques for compensating for delay. Since this would take a potentially unbound amount of effort, we are going to focus on a specific set of tools, as described below.

Note that short paths can obscure flaws in all parts of the stack: application+TCP+IP+link. We can further separate the problem space into two domains: the upper stack (application+TCP+IP) and the lower stack (TCP+IP+link). Although there is some overlap, different strategies are optimal for the two domains. The developer would much prefer to "bench test" applications over long virtual paths by inserting emulated delays or scenic VPNs within the development environment. The lower stack is most easily diagnosed with a light weight, portable or easily deployed synthetic application using parameter scaling because you potentially need to test from a large number of geographically separated vantage points to test all of the segments of the actual path.

In principle, all three delay compensation techniques can be applied to both upper and lower stack in conjunction with either the real application or various diagnostic applications. Furthermore all of these techniques can be combined in multiple ways. For example; a real application can be tested over a path that contains both scenic routing and emulated delays, or diagnostic tools can use both emulated delays and model scaling.

## **The tool development plan**

We propose to develop a suite of enhanced diagnostic tools that compensate for the effects of path delay. This suite will be targeted to specific groups: True end users, application developers, network administrators and other diagnostic experts or tool developers. The suite will be built from a small kit of core components that implements the previously described techniques that compensate for path delay, in conjunction with some simple diagnostic applications. The core components of this suite will be built on top of classical diagnostic tools.

Several years ago, we built a "testrig" (<http://www.ncne.nlanr.net/research/tcp/testrig>) package, which included several expert level TCP trace analysis tools packaged into a single easy-to-install and use utility. Testrig enabled moderately experienced programmers or network administrators to collect and analyze TCP traces using expert grade tools.

### **Tools suites for application developers**

For the application developer and other people looking for flaws in the upper stack, including in TCP itself, we will build a suite of tools to implement long virtual paths using scenic VPNs and emulated delays. These will enable application developers to "bench test" their code in the lab using long virtual paths. In this diagnostic setting the developer has full local access to all components of the application, and can observe how it functions over a hypothetical long path. The application developer still needs to use conventional distributed diagnosis tools such as Netlogger (<http://www-didc.lbl.gov/NetLogger/>) instrumentation for distributed applications.

These diagnostic suites will mostly be built on top of existing VPN and delay emulation tools and techniques. Since the existing tools were not designed for diagnostic use, we will need to adapt them as appropriate.

The VPN and tunneling tools are generally designed to be configured by a network administrator to address specific security or network connectivity problems. For production use they are most often implemented in routers or other expensive networking gear. However several techniques, such as IP in IP tunnels [RFC1853] and GRE tunnels [RFC2784] are implemented in common operating systems. Since VPNs and tunnels are generally used by a totally different audience, their documentation does not lend itself to diagnostic use. We will evaluate several different VPN and tunnel techniques for suitability as a diagnostic tool and repackage and document them specifically to aid application developers.

Emulated delay is also well known, but not as a diagnostic tool. Dummynet [Riz97] is heavily used in the networking research community and is embedded in the emulab [W+02] and netbed (<http://www.emulab.net/index.php3>) network emulation testbeds. We will adapt and document Dummynet for diagnostic use, as well as build a new delay

emulation tool that will be designed to be used in conjunction with some of our other path diagnostic tools described below.

### Tool suites for path diagnosis

Most of the path diagnosis tools that we will develop will be based on the remote client server model shown in Figure 6. The user interface will use a simple control protocol to invoke tests between the diagnostic server and the test target (TT). The test target is just a simple high performance TCP discard server.

Most of the complexity will be in the diagnostic server. It requires Web100 support in the kernel and diagnostic software to implement model scaling to compensate for the effects of path delay. Initial efforts will focus on the experimental "pathdiag" tool developed under the Net100 and Web100 projects (<http://www.psc.edu/~web100/pathprobe>).

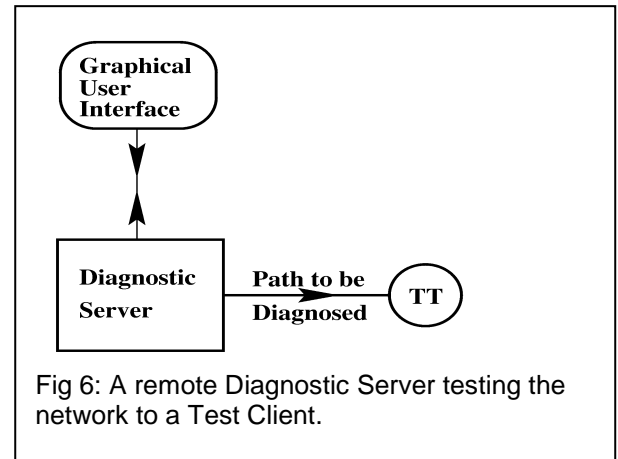


Fig 6: A remote Diagnostic Server testing the network to a Test Client.

Pathdiag uses two diagnostic modes to emulate different application and network contexts in addition to model scaling to compensate for path delay. In the laminar mode, it mimics a well behaved smooth bulk transport application, such as FTP on a quiet network. In burst mode it mimics the behavior of TCP with a bursty application (e.g. sending large messages separated by short idle intervals) or TCP under the effects of ACK compression [ZSC91] or other bursty cross traffic. Although laminar mode works very well, it is insufficient to detect flaws associated with inadequate queuing in the network. Burst mode should be able to test the queuing, but we do not yet understand how to calibrate the results. In particular we have examples of infrastructure that is known to successfully support high performance applications on long paths, yet fail the current experimental burst mode test in pathdiag. We are in the unusual position of having a diagnostic tool that is too sensitive. Under this project we propose to refine pathdiag burst mode. This is a potentially substantial research question, because there has been so little work on characterizing the burst capacity effects on network performance.

Pathdiag's current output is not suitable for non-experts. We will redesign its user interface such that non-expert users can see an indication of how well the tested path segment might support a high performance application, while more expert users can get some insights into the nature of the failure.

All three components illustrated in Figure 6 can be viewed as diagnostic building blocks that can be configured in several different ways, depending on the needs of the target audience. A big part of the difference between end user tools and expert tools will be the extent to which the user interface simplifies operation by limiting the complexity of the available tests and results.

For the least sophisticated users, the tools will present a web based graphical user interface (GUI). This interface will be implemented in Java and include a built in test target. The only supported test will be from a diagnostic server back to the test target built into the Java applet. A good example of this approach is the ANL diagnostic server at <http://miranda.ctd.anl.gov:7123>. This particular server does an extremely good job of detecting some specific network flaws, such as Ethernet duplex miss-match) however, it does not have a mechanism for compensating for path delay. We will build a similar tool that includes path delay compensation.

Experience with the ANL tester has shown that the trivial invocation of the web based user interface is likely to make it the first choice diagnostic tool for most users, including the experts. Our proposed version will include built-in reporting tools to support effective problem escalation for novice users. For all common flaws the non-expert end user tools will be able to easily detect and provide good starting data for trouble reporting to network administrators.

Network administrators will want to be able to do third party diagnosis, where the user interface is in the network operation center, and the diagnostic servers and test targets straddle successive sections of a long path, as shown in Figure 7. This requires a good authentication and authorization infrastructure and a means to locate successive diagnostic servers along a long network path. Both of these problems are already under investigation in other contests, such as the Grid, NIMI, and the Internet2 diagnostic adviser. We will actively incorporate other solutions in a flexible, modular way, so we can upgrade them in the future.

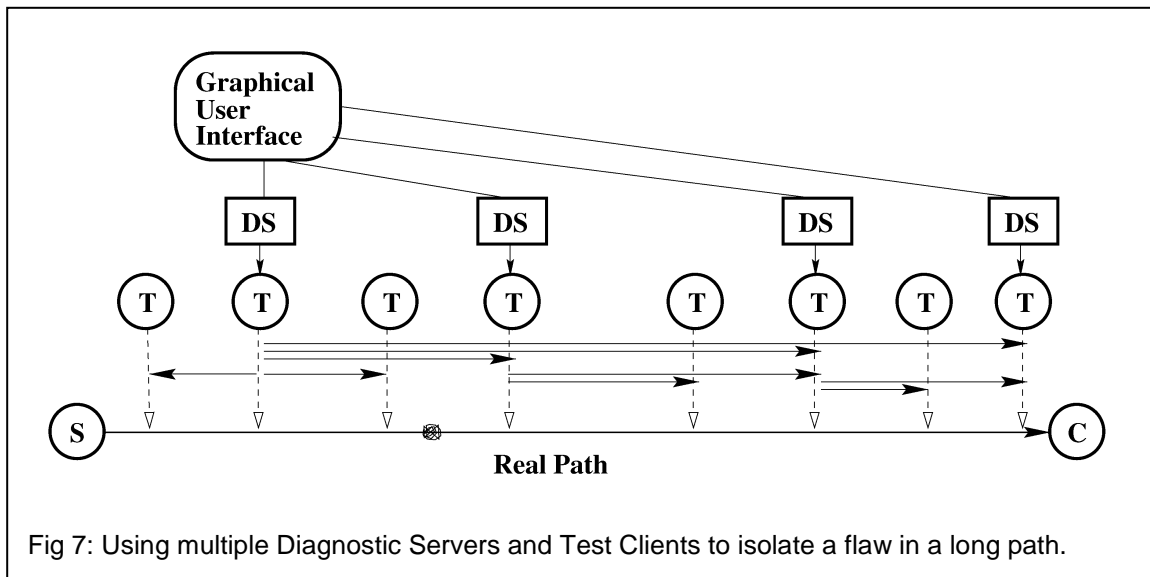


Fig 7: Using multiple Diagnostic Servers and Test Clients to isolate a flaw in a long path.

In a few cases, such as when there is a defective network connector in an office, network administrators may want to run diagnostic software in laptops that can be connected in the network in exactly the same locations as end user systems. Therefore, even the most sophisticated tools need to be able to be installed by moderately experienced network administrators. We consider this to be an important requirement.

### **Hybrid tools**

There are two hybrid diagnostic techniques that are worth mentioning.

1. Some end user applications can be observed directly with Web100 instrumentation and analyzed using model scaling techniques to compensate for delay. This is important if the application exhibits certain pathological behaviors that cause problems in the network. For example an application that is extremely bursty can cause queue overflows and excessive packet loss in an otherwise lossless network. Although the diagnostic server will include burst tolerance testing, this sort of "second order" problem can be very hard to reproduce. This passive model scaling tool will consist of the analysis code extracted from pathdiag, such that it can be applied to a connection that is part of a real application.
2. The other particularly useful hybrid technique will be adding emulated delay to the diagnostic server, such that it has an alternate method to compensate for the effects of path delay. This will also permit additional techniques to introduce burstiness into the traffic and enable us to better calibrate the burst tests for effects on burst intolerant equipment along the network paths.

**(Detailed Plans Omitted)**

### **References**

[WFG02] Wetzel, A.W., Feineigle, P.A., Gilbertson, J.R. "Design of a High-Speed Slide Imaging System for Pathology". Proceedings of IEEE International Symposium on Biomedical Imaging: Macro to Nano, July 7-10, 2002 (In Press)

[MSMO97] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. "The macroscopic behavior of the TCP Congestion Avoidance algorithm", Computer Communications Review, 27(3), July 1997.

[RFC2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis. "Framework for IP Performance Metrics", RFC2330, May 1998.

[RFC2678] J. Mahdavi, V. Paxson. "IPPM Metrics for Measuring Connectivity" RFC2678, September 1999.

[RFC3148] M. Mathis, M. Allman. "A Framework for Defining Empirical Bulk Transfer Capacity Metrics", RFC3148, July 2001.

[MHRRS03] M. Mathis, J Heffner, R Reddy, R Raghunarayan, J. Saperia. "TCP Extended Statistics MIB", IETF work in progress, draft-ietf-tsvwg-tcp-mib-extension-03.txt, March 2003.

[MHR03] M. Mathis, J. Heffner, R. Reddy. "Web100: Extended TCP Instrumentation", Submitted to ACM Computer Communications Review, 2003.

[RFC2018] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow. "TCP Selective Acknowledgement Options" RFC2018 October 1996.

[RFC2525] V. Paxson, M Allman, S. Dawson, W. Fenner, J. Griner, I. Heavens, K. Lahey, J. Semke, B. Volz. "Known TCP Implementation Problems" RFC2525 March 1999.

[RFC2760] M. Allman, Ed., S. Dawkins, D. Glover, J. Griner, D. Tran, T. Henderson, J. Heidemann, J. Touch, H. Kruse, S. Ostermann, K. Scott, J. Semke. "Ongoing TCP Research Related to Satellites" RFC2760 February 2000.

[RFC2883] S. Floyd, J. Mahdavi, M. Mathis, M. Podolsky. "An Extension to the Selective Acknowledgement (SACK) Option for TCP". RFC2883, July 2000.

[SMM98] J. Semke, J. Mahdavi, M. Mathis. "Automatic TCP Buffer Tuning", ACM SIGCOMM98, Computer Communication Review 28(4), October 1998.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. "Modeling TCP throughput: a simple model and its empirical validation". ACM SIGCOMM, September 1998.

[MMR03] J. Mahdavi, M. Mathis. and R. Reddy. "Enabling High Performance Data Transfers", [http://www.psc.edu/networking/perf\\_tune.html](http://www.psc.edu/networking/perf_tune.html), Revised January 2003.

[MHR03a] M. Mathis, J. Heffner, R. Reddy. "Web100 Tools", Presentation to Miami Joint Techs, Feb 2003.

[DMT02] T. Dunigan, M. Mathis, and B Tierney. "A TCP Tuning Daemon", Supercomputing 2002, Oct 2002.

[Mat94] M. Mathis. "Windowed Ping: An IP Level Performance Diagnostic", Proceedings of INET'94, June 1994. See: <http://www.psc.edu/~mathis/wping/>

[Mat96] M. Mathis. "Diagnosing Internet Congestion with a Transport Layer Performance Tool", Proceedings of INET'96, June, 1996, Montreal, Quebec.

[Mat00] M. Mathis. "TReno Bulk Transfer Capacity", draft-ietf-ippm-treno-btc-03.txt, Work in Progress Internet-Draft, expired June 2000.

- [SM01] J. Semke, M. Mathis. "A preconfigured TCP Test Rig".  
<http://www.ncne.nlanr.net/research/tcp/testrig/>, Updated Feb 2001.
- [RFC1853] W. Simpson. "IP in IP Tunneling". RFC1853, October 1995.
- [RFC2784] T. Li, S. Hanks, D. Meyer, P. Traina. "Generic Routing Encapsulation".  
(GRE). RFC2784, March 2000
- [Riz97] Rizzo, L. "Dummynet: a simple approach to the evaluation of network protocols". ACM Computer Communication Review 27(1), Jan 1997. See:  
[http://info.iet.unipi.it/~luigi/ip\\_dummynet/](http://info.iet.unipi.it/~luigi/ip_dummynet/)
- [W+02] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. "An Integrated Experimental Environment for Distributed Systems and Networks". In Proc. of the 5th Symposium on Operating Systems Design and Implementation, Boston, MA, December 2002.
- [ZSC91] L. Zhang, S. Shenker and D. D. Clark. "Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic". Proc. ACM SigComm1991, September 1991.