



A Quick Check of Network Performance

by Jeffrey T. Hicks and John Q. Walker

Have you ever tried to measure the response time across a network? Do you sometimes wonder what throughput rate you're getting over a particular link? Are you concerned about the impact of adding streaming multimedia traffic to a network? Would you like to know the exact route your data is taking? Individual tools are available to measure the throughput and response time of your applications, trace a network route, or test a network's capacity for handling multimedia traffic. But a free software tool called *Qcheck* can perform all of these tasks and more, while providing a quick check of network performance.

This is an update of an article printed in the *International Journal of Network Management*, volume 11, number 1, January-February 2001, pages 65-72, ISSN 1055-7148.

Contents

What Is Qcheck?	2
Measuring Network Performance with Qcheck.....	4
Conclusion	8

What Is Qcheck?

Qcheck is a free software utility that provides network performance measurements. It can be downloaded at www.qcheck.net. It consists of two software components: a console with a graphical user interface, and distributed software agents called Performance Endpoints (or simply, "endpoints"). The Qcheck console is where you configure and run tests, and then view test results. You can select among several kinds of tests, together with the addresses of the endpoints and the protocol to use between them (TCP, UDP, SPX, or IPX). The test results are summarized in a window, while a more detailed report of test results can be viewed in a Web browser.

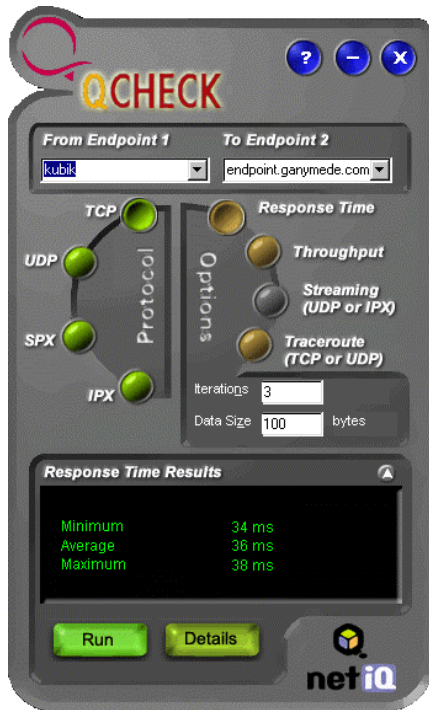


Figure 1. The Qcheck console runs as an application program on Microsoft's Windows 95, 98, Me, NT, and 2000. Endpoint software runs on more than 20 operating systems.

Qcheck uses endpoints to actively test response time, throughput, streaming, and traceroute performance. The Qcheck console sends test-setup instructions to a pair of endpoints which, in turn, execute the test and return the results.

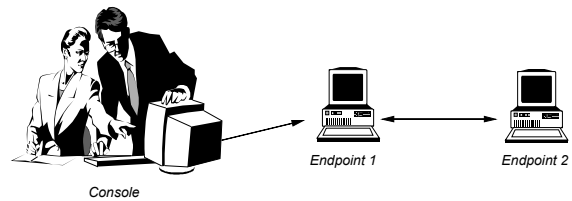


Figure 2. Performance tests are set up at the Qcheck console and run between two computers, labeled Endpoint 1 and Endpoint 2.

Network professionals use endpoints for sophisticated measuring and monitoring with products like Chariot and Pegasus from NetIQ Corporation. Qcheck was created so end users could also exploit the endpoint functions. Endpoint software for more than 20 different operating systems can be downloaded at no charge from www.qcheck.net/endpoint.html. Endpoint software needs to be installed on the computers that are running the performance tests.

How Does Endpoint Software Work?

Endpoints are software agents that run on a wide set of operating systems. These agents can be installed on mainframes, workstations, and personal computers. An endpoint runs as a system service or background process. While idle, an endpoint consumes few system resources as it awaits instructions.

Tests are run between a pair of endpoints, called *Endpoint 1* and *Endpoint 2*. Endpoints 1 and 2 behave as peer computers, with Endpoint 1 always starting the network connection. For client/server applications, Endpoint 1 typically emulates the client while Endpoint 2 emulates the server. In streaming tests, Endpoint 1 typically emulates a streaming server or sender and Endpoint 2 emulates the client or receiver of the data. A single endpoint can act in either role, as Endpoint 1 or Endpoint 2; there's no need to install separate agents.

To run a test, first enter the network addresses for Endpoint 1 and Endpoint 2 in the appropriate fields at the Qcheck console. Next, choose a network protocol and the kind of test to run. When you press the Run button, the console sends the test-setup instructions (addresses, protocol, etc.) to Endpoint 1.

Endpoint 1 extracts the instructions specific to it and forwards the rest to Endpoint 2. Endpoint 1 then starts the data exchange and returns the results to the console. In a streaming test, Endpoint 2 collects results and sends them back through Endpoint 1 to the console.

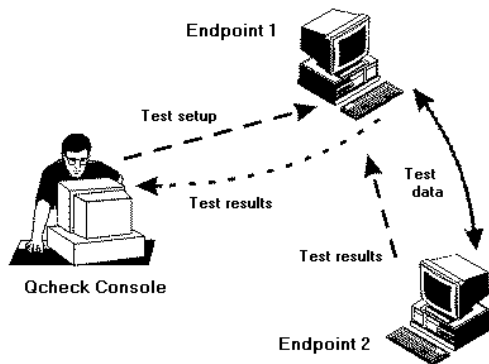


Figure 3. The Qcheck console sends test setup instructions to Endpoint 1. Tests are then run between Endpoints 1 and 2, which return their measurements to the console.

Qcheck measures network performance at the application level. Endpoints make calls using application programming interfaces (APIs) for the four supported network protocol stacks. For example, a test using the TCP or UDP protocol causes the endpoints to make calls to their Sockets APIs.

Calculations of throughput and response time are based on the number of bytes sent and received from the application's perspective. Qcheck's throughput measurement is lower than the throughput that you would see with a network protocol analyzer. Qcheck measurements reflect only the data payload – what an application user would see – without the protocol overhead, such as headers, trailers, flow control, connection setup, and retransmissions.

Reliable Delivery of Datagrams

Endpoints employ reliable datagram support for UDP and IPX tests of response time and throughput. Reliable delivery means that if all the data sent by one endpoint is not eventually received by the other endpoint, the test fails. For tests specifying UDP or IPX, the endpoints use a simple datagram protocol that emulates applications that cannot deal with loss of data. Here's how it works:

1. A windowing scheme is used as the flow control mechanism: the sender sends at most a certain amount of data before waiting for an acknowledgment from its receiver.
2. The sender waits for a period of time (the retransmission timeout period) to receive an acknowledgment from its recipient. If the acknowledgment does not arrive in the specified amount of time, the sender retransmits the window of unacknowledged data.
3. The receiver sends an acknowledgment when the following conditions apply:
 - A window is filled, or
 - All the data for a Receive call is received.

An acknowledgment is not sent immediately upon receipt of all the data for a Receive call. Instead, the acknowledgment is sent when the next API call is issued (unless the next call is Receive, in which case the endpoint keeps receiving until the window is full). This is a common way for peer “reliable datagram” applications to reduce the traffic flows they generate. It also means that the endpoint sends one less datagram (an acknowledgment) when a Receive is followed by a Send.

4. If the receiver detects lost datagrams, it sends an acknowledgment indicating how much of the window was received in sequence, thereby letting the sender retransmit what wasn't received.

Comparing Qcheck with Other Utilities

You might ask, “Why not use Ping or Traceroute to gather performance information?” Ping uses the Internet Control Management Protocol (ICMP) to measure response time between computers on IP networks. It's useful because most network devices and computers can respond to a Ping request. However, most TCP/IP applications do not use ICMP for communication; they use TCP or UDP. In addition, intermediate network devices, such as routers, may treat ICMP traffic differently. The Ping packet size is typically small and may be able to pass through routers faster (or slower), depending

on network configuration. And prioritization or packet-shaping devices typically do not prioritize Ping traffic.

By contrast, Qcheck uses the same protocols and APIs that real-world applications use: TCP, UDP, SPX, or IPX. Routers process the endpoint traffic just like real application traffic. And unlike Ping, Qcheck can test network throughput and see how a network supports streaming multimedia traffic.

Traceroute is another utility often used to gather network performance information. It traces the route that data is taking from source to destination through a network. Many operating systems provide a built-in traceroute command to provide traceroute information. Typically, a traceroute command returns information such as average round-trip time to each hop, hop addresses, and resolved hop names. Qcheck provides the same information, while also letting you trace routes from a remote location. For example, the Qcheck console located on a computer in a corporate network may instruct an endpoint located at another site to run a traceroute to a destination on the Internet. All of the results, one hop at a time, are returned to the Qcheck console.

Measuring Network Performance with Qcheck

What are the key network performance indicators, and why are they important? Qcheck gives a quick check of the response time, throughput, streaming capability, and routes between a pair of computers in a network. Depending on the application and user, different measurements are more revealing of overall network performance.

Key measurements for most business and Web transactions are response time and throughput [1]. When an answer or reply is needed quickly or is frequently repeated, users worry about response time. When large files must be transferred quickly, throughput is the primary concern.

Users of streaming applications need to know that a network can support a fixed level of

throughput. They also have an additional concern: how many datagrams are lost as the data is sent from the sender to the receiver. The lost data information Qcheck provides can help determine a network's readiness for streaming multimedia traffic.

Finally, traceroute information can show performance bottlenecks and slowdowns at intermediate nodes. When latency is important, either as a component of response time or for real-time streaming applications like Voice over IP [2], traceroute can show the paths in each direction and the round-trip time to each hop on the paths.

Response Time

The response-time measurement indicates how long it takes to send a request and receive a reply over a network. This is important for many network transactions. A browser sends a request and then waits for a reply to load a Web page. The longer the operation takes, the more impatient the user gets. Response time is a measurement that reflects the user's experience with a network. Response time for a transaction is usually described in milliseconds or seconds.

Qcheck measures response time by directing Endpoint 1 to send a block of data (from 1 to 32,000 bytes) to Endpoint 2. After the data is sent, Endpoint 1 waits to receive a reply from the other endpoint in the test. The response time is how long it takes to complete this round-trip transaction. The response time is averaged over a user-selected number of iterations, which can range from 1 to 10. Choosing a higher number of iterations, or the number of times the transaction is repeated and measured, gives a more representative average measurement

```
Endpoint 1                               Endpoint 2
-----
Connect -----> Accept
Loop N times                               Loop N times
  Start stopwatch
  Send S bytes -----> Receive S bytes
  Receive 100 bytes <----- Send 100 bytes
  Stop stopwatch
End Loop                                   End Loop
Disconnect                                 Disconnect
```

Figure 4. The transaction between the endpoints in a Qcheck response-time test. S is the size of the data block sent, which can be 1 to 32,000 bytes.

The equation used for calculating average response time is:

$$RT = (m1 + \dots + mN) / N$$

where:

RT = average response time, in sec
 N = number of iterations
 m = measured time, which is
 end time - start time
 m1 = m for the first iteration
 mN = m for the Nth iteration

The response time test summary shows the minimum, maximum, and average response times of all iterations. The average response time provides a good measure of the end-to-end round-trip time through a network.

Qcheck calculates round-trip response time. However, in some cases, you might like to know the time it takes for traffic to go end to end in just one direction. One-way response time is sometimes referred to as latency and can be an important measurement for real-time applications. Typically, you can calculate one-way response time by halving the response time results. However, this calculation is sometimes inaccurate, because the routes to and from the endpoints may be different. Running a Qcheck traceroute test can show whether the routes are identical.

Throughput

Throughput numbers tell the rate at which traffic can flow through a network. A typical network transaction causes blocks of data to be exchanged over a network. A network with higher throughput lets data be delivered in a shorter amount of time. Throughput is a measurement that reflects the capacity of a network and is usually measured in bytes or bits per second.

Qcheck measures throughput by testing how fast a block of data can be sent and received. You configure the size of the data block to send (from 1,000 to 1,000,000 bytes). Then Qcheck instructs Endpoint 1 to send the specified amount of data and wait for a 1-byte response. The throughput rate is calculated as the amount of data sent and received divided by the measured transaction time.

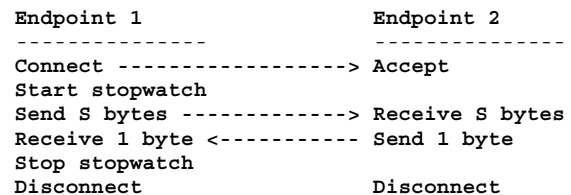


Figure 5. The transaction between the endpoints in a Qcheck throughput test.

The equation used for calculating throughput is:

$$T = (S + R) / m$$

where:

T = throughput rate, in bytes/sec
 S = bytes sent by Endpoint 1
 R = bytes received by Endpoint 1
 (always 1 byte)
 m = measured time, which is
 end time - start time

The throughput test summary shows the average throughput. The average throughput provides a good indication of the achievable network throughput. However, on higher speed networks, the throughput results may be much lower than the total available bandwidth. Why does this occur? So as not to swamp your network, Qcheck is designed to generate small, brief data flows; it's limited to a single connection and sends no more than 1,000,000 bytes of data.

Endpoint computer operating systems and protocol stacks can also limit throughput; today's personal computers can rarely generate throughput greater than 100 Mbps with a single connection. Therefore, the throughput results you see might be substantially lower than bandwidth capacity would indicate.

Qcheck can guide you in improving throughput by running the same tests after you make configuration changes to the operating system or protocol stack [3]. Tailoring the protocol stack configuration at each of the endpoints is an advanced technique; we don't recommend it for anyone but the most technically adventurous. For example, changing the value of the "TCP Receive Window" parameter at an endpoint can affect the results you see when testing for maximum throughput. The TCP Receive Window is a TCP/IP stack configuration parameter. Many TCP/IP stacks ship with a default value of 8K bytes. Changing to a larger value changes performance, increasing

throughput on some stacks and reducing it on others. Careful experimentation with the values you use for this parameter might help you increase throughput.

Streaming Performance

Another important measurement for some applications is the amount of lost data. Lost data is something that applications using connection-oriented protocols like TCP or SPX do not experience. If a TCP application sends data, the underlying protocol stack ensures that the data reaches the receiver. Connectionless protocols, such as UDP and IPX, can have data reliability schemes implemented by the application above the protocol. For UDP and IPX response time and throughput tests, reliable datagram support ensures that data is dependably transferred.

However, there is a class of multimedia applications, such as voice and video, which can tolerate the loss of some datagrams. These applications typically ride on top of UDP. UDP itself does not ensure that the data makes it to the receiver, so the applications must deal with lost data.

Why is measuring the amount of lost data important? Have you ever tried to watch a movie and seen it freeze before jumping to a different scene? Lost data can mean lost scenes in a video application. Likewise for telephone calls: lost data can cause the speaker's voice to sound unintelligible. How much is too much when it comes to lost data? This varies by application; some real-time applications can tolerate certain amounts of lost data because they buffer data as it is received. Other applications don't tolerate lost data. Typically, lost data is described as bytes lost or the percentage of bytes that were lost.

Qcheck measures lost data by instructing Endpoint 1 to stream data at a fixed rate for a short time. Lost data is data that is not received by the Endpoint 2. The data could be discarded in the network at various routers or it may be discarded by the receiving TCP/IP stack. If the data arrives out of order, it is also counted as lost. (Chariot shows many more details than Qcheck, including the number of duplicate packets and the number of packets that are considered out of order.)

Endpoint 2 calculates how much data was lost. The rate at which data is sent can be set to a value from 1,000 to 1,000,000 bits per second. The amount of time that the data is sent can be set to a value from 5 to 30 seconds.

```

Endpoint 1                               Endpoint 2
-----
Loop for duration                         Start stopwatch

    Send 64 bytes -----> Receive 64 bytes
    Sleep to control rate

End Loop                                  Stop stopwatch

```

Figure 6. The transaction between the endpoints during a Qcheck streaming test.

Qcheck uses the following equation for calculating the percentage of lost data:

$$L = ((S - R) / S) * 100$$

where:

L = percentage of lost data
 S = total bytes sent by Endpoint 1
 R = total bytes rcvd by Endpoint 2

The streaming test summary shows the amount of data lost in bytes, the percentage of data lost, and the actual throughput at the receiver. In Qcheck streaming tests, Endpoint 2 keeps track of the data that is lost. Endpoint 1 streams data at a fixed rate with no acknowledgment, telling Endpoint 2 how much data has been sent. Endpoint 2 calculates the lost data total and returns it to the Qcheck console, via Endpoint 1. Lost data is shown in increments of 0.1%.

A considerable percentage of data may be lost in streaming tests. Data loss has several causes. For example, the data rate may be higher than the maximum throughput potential of a network. Packets can be lost due to interference during transmission, or the receiver may not be capable of keeping up with the sender and may drop packets. A network may be congested: if intermediate routers are congested, datagram packets may be discarded. In these cases, check to make sure you've selected the correct units for your test. Try decreasing the streaming rate or changing the units to see if the results improve. Your network may be configured to give non-streaming traffic priority over streaming traffic [4] and may discard datagram packets when the two compete for bandwidth. Try running a throughput test in the corresponding connection-oriented protocol for comparison. If your

throughput is unexpectedly low, network congestion is the likely cause.

Traceroute

Traceroute information details the path that a packet takes in a network. Intermediate nodes or routers are shown along with the round-trip time between the source and each intermediate node. A traceroute test typically uses the ICMP protocol to send an echo packet with increasing Time-To-Live (TTL) values, although some traceroute implementations send a UDP echo packet instead. The endpoints used by Qcheck use ICMP for traceroute testing.

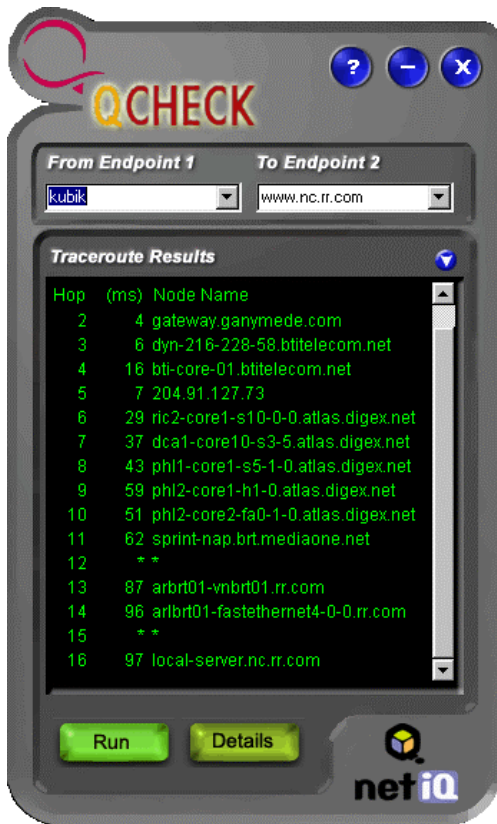


Figure 7. This traceroute test shows 16 hops from Endpoint 1 to the indicated Web site. The middle column shows the round-trip time, in milliseconds, to each hop. Asterisks indicate no reply was received at this hop within the timeout – this can occur when a device is configured to not reply or when replies are blocked by a firewall.

Qcheck gathers traceroute information by instructing Endpoint 1 to send ICMP Echo packets and to examine the responses. The address configured in Qcheck’s Endpoint 1 field must be a computer with active endpoint software. However, the Endpoint 2 address in a traceroute test can be any valid IP address; it doesn’t need to have endpoint software installed. Traceroute information shows the hop numbers, round-trip time to each hop, and node names.

Endpoint 1 sends multiple ICMP echo packets, with a series of TTL values. As a packet goes through a network, each router decrements the packet’s TTL value. When the TTL reaches 0, the router sends back an ICMP Error packet, indicating that the TTL expired. By sending out packets with increasing TTL values, Endpoint 1 builds up the route as the next router in the path sends back the TTL-expired message. For traceroute tests, Qcheck uses default values of 30 maximum hops and a 3-second timeout per hop.

```

Endpoint 1
-----
Intermediate node,
any IP address,
or Endpoint 2
-----
Loop until ICMP_ECHO_REPLY or 30
Start stopwatch
Send ICMP_ECHO ----> Receive ICMP_ECHO
<--- Send ICMP_ERROR
      (intermediate node)
or
Send ICMP_ECHO_REPLY
      (IP address or
      Endpoint 2)

Receive ICMP_ERROR
or
Receive ICMP_ECHO_REPLY
Increment TTL
Stop stopwatch
End Loop

```

Figure 8. The processing for a Qcheck traceroute test.

The traceroute test summary shows the hop numbers, round-trip time to each hop, and hop names. (The traceroute summary window can be expanded to show more hops, if necessary.) We’ve seen cases where a distant hop has a smaller round-trip time than a nearer hop. This is usually due to transient congestion at one of the intermediate nodes.

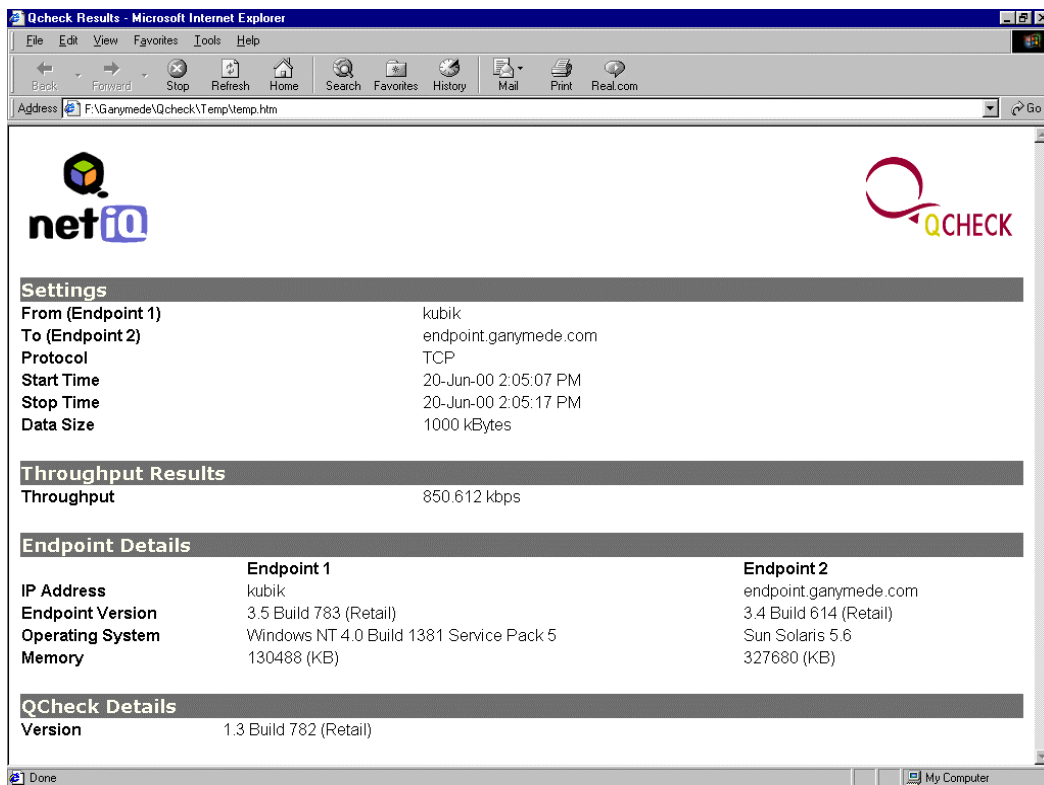


Figure 9. An example of Qcheck's HTML output, created by pressing the Details button after running a throughput test.

Result Details

After you've run a test, the Qcheck console shows a summary of its test results. Each kind of test provides different results in the summary area. In addition, the "Details" button can be used to launch the formatted test results in a Web browser. This browser view makes several additions to the summary information. First, the test settings are shown, including the Endpoint 1 address, Endpoint 2 address, protocol, start time, stop time, and any user configurable settings. The test results that are shown in the console summary window are also shown in the settings. And finally, endpoint details are provided: network address, endpoint version, operating system, memory, and, for streaming tests, CPU utilization. The browser makes it easy to print, save, or send the results to someone else via e-mail.

Conclusion

The simple Qcheck user interface lets a wide variety of users make complex network performance measurements. It shows the

response time and the throughput between pairs of computers anywhere in a network. It also demonstrates a network's ability to support real-time multimedia traffic and details the paths traversed between computers. Home users have verified the throughput of new cable and DSL Internet connections with Qcheck. Large corporations have used Qcheck to investigate user reports of poor response time, by helping to answer the question of whether a problem exists in the network or in the users' applications. Qcheck offers network testers a reliable way to generate streaming traffic and see how well it is supported.

Qcheck has proven very successful in giving both users and network administrators an easy way to see how their network is performing. As of spring 2001, around 1,000 copies of Qcheck are downloaded each day.

Qcheck provides a 'quick check' of network performance. We've significantly enhanced the same technology found in Qcheck to do more in-depth network performance testing, advanced troubleshooting, and 24-by-7 performance monitoring. These functions are available in our Chariot and End2End software tools – and they use the same endpoints.

Chariot runs performance tests by driving traffic that looks like real application traffic (such as, VoIP, streaming media ERP, file transfers, and e-mail). Chariot can generate a single connection like Qcheck or the traffic of hundreds or thousands of users.

End2End Performance Monitor reports on the performance of networked applications and

Web sites over time. End2End can run tests similar to Qcheck, but on a scheduled basis. It also can use the endpoints to passively monitor the network traffic generated in their computers.

For more information on these tools, see www.netiq.com/Solutions/Network_Performance_and_Testing/

About the Authors

Jeffrey T. Hicks is a senior developer with NetIQ Corporation, leading the Chariot development team. He can be reached at jeff.hicks@netiq.com. John Q. Walker II, Ph.D., is the director of network development and a co-founder of Ganymede Software, which joined NetIQ Corporation in 2000. He can be reached at johnq@netiq.com.

Acknowledgments

This paper was suggested by Katherine Demacopoulos. Several people improved this paper by contributing excellent reviews: Joana Bacon, Joan Bellinghausen, Sharon Bidaure, Marya DeVoto, Susan Pearsall, Jeff Puckett, Scott Smith, Kim Shorb, Chris Selvaggi, Carl Sommer, Ken Treimann, and John Wood

References

- 1) Wood, John L., Christopher D. Selvaggi, and John Q. Walker II. "Testing the Performance of Multiple TCP/IP Stacks," *Proceedings of CMG97*, December 7-12, 1997, volume 1, pages 626-638.
- 2) McCullough, Daniel J. and John Q. Walker II. "Interested in VoIP? How to Proceed," *Voice 2000*, a supplement to Business Communications Review, April 1999, pages 16-22.
- 3) *Optimizing Bandwidth*, Michele Jo Petrovsky, McGraw-Hill, May 1998, ISBN: 007049889X.
- 4) *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*, Paul Ferguson and Geoff Huston, John Wiley & Sons, Jan. 1998, ISBN: 00471243582.