The Internet Protocol Journal

March 2003 Volume 6, Number 1

A Quarterly Technical Publication for Internet and Intranet Professionals

In This Issue

You can download IPJ back issues and find subscription information at: **www.cisco.com/ipj**

From The Editor

Even the most carefully designed and operated IP network is subject to any number of performance problems ranging from overloaded links and mis-configured routers to server failures. For these situations, the network manager has several diagnostic tools as options. Geoff Huston gives us an overview in an article entitled "Measuring IP Network Performance."

Voice over IP (VoIP) is an emerging application, as well as a rapidly growing market. Use of the corporate network or the Internet at large to carry telephone traffic has many advantages, not the least economic ones. A successful VoIP network must not only support IP-based telephones, but also provide a means of seamlessly integrating the IP-based network with traditional telephone networks. At the core of VoIP lies the *Session Initiation Protocol* (SIP) and a few related protocols. Bill Stallings describes SIP in our second article.

Book reviews published in *The Internet Protocol Journal* can rarely be characterized as "controversial." However, when the book in question deals with ICANN, it is perhaps not surprising that strong opinions emerge. Thus, following the review of *Ruling the Root* in our last issue, we received a letter from the author that is included in our "Letters to the Editor" section (along with a response from the book reviewer). I would like to take this opportunity to remind our readers that book reviews do represent the *opinion* of the reviewer and should be read in that light.

Our online subscription system has been up and running for a couple of months. Please give it a try at: **www.cisco.com/ipj**.

—Ole J. Jacobsen, Editor and Publisher
ole@cisco.com

Measuring IP Network Performance

by Geoff Huston, Telstra

If you are involved in the operation of an IP network, a question you may hear is: "How *good* is your network?" Or, to put it another way, how can you measure and monitor the quality of the service that you are offering to your customers? And how can your customers monitor the quality of the service you provide to them?

These questions have been lurking behind many public and enterprise IP networks for many years now. With the increasing levels of deployment of various forms of high-speed (or broadband) services within today's Internet there is new impetus to find some usable answers that allow both providers and users to place some objective benchmarks against the service offerings. With the lift in access speed with broadband services, there is an associated expectation on the part of the end user or service customer about the performance of the Internet service. It should be "better" in some fashion, where "better" relates to the performance of the network and the service profile that is offered to network applications. And not only is there an expectation of "better" performance, it should be measurable. This article looks at network performance and explores its definition and measurement.

A Functional Definition of Network Performance

An informal functional approach to a definition of network performance is measuring the speed of the network. How fast is the network? Or, what is the elapsed time for a particular network transaction? Or, how quickly can I download a data file? This measurement of time for a network transaction to complete certainly relates to the speed of the network, and speed is a good network performance benchmark, but is speed everything?

When looking at the broad spectrum of performance, the answer is that speed is not everything. The ability of a network to support transactions that include the transfer of large volumes of data, as well as supporting a large number of simultaneous transactions, is also part of the overall picture of network load and hence of network performance. But large data sets is not everything in performance. Consideration should also be given to the class of network applications where the data is implicitly clocked according to some external clock source. Such real-time applications include interactive voice and video, and their performance requirements include the total delay between the end points, or latency, as well as the small-scale variation of this latency, or *jitter*. Such performance measurements also include the ratio of discarded packets to the total number of packets sent, or loss rate, as well as the extent to which a sequence of packets is reordered within the network, or even duplicated by the network. Taken together, this set of performance factors can be considered as a form of the amount of distortion of the original real-time signal.

Accordingly, a functional description of network performance encompasses a description of speed, capacity, and distortion of transactions that are carried across the network. This informal description of what

constitutes network performance certainly feels to be on the correct path, given that if one knew the latency, available bandwidth, loss, and jitter rates and packet reorder probability as a profile of network performance between two network end points, as well as the characteristics of the network transaction, it is possible to make a reasonable prediction relating to the performance of the transaction.

Taking this informal definition, the next step is to create a more rigorous framework for measuring performance. For any single network path between an entry and egress point, it is possible to measure the path latency, available peak bandwidth, loss rates, jitter profile, and reorder probability. But there is a difference between a description of the performance of a particular path across a network and the performance of the network as an aggregate entity. Given a set of per-path performance measurements, how can you construct a view of the performance of the network? A common methodology is to take a relatively complete set of path measurements across a network and then combine them to create an average metric. Although this accomplishes a useful reduction in the size of the data, there is also a loss of information. The average network performance measurements have little relationship to the performance of any individual path.

There are various ways to improve this loss of information, including weighting the individual path measurements by the amount of traffic passed along the path. Such techniques are indeed to ensure that paths that use far-flung network outliers that carry relatively low volumes of traffic have a much lower impact on the overall network performance metric than the major network transit paths.

Measuring Network Performance

Given these performance indicators, the next step is to determine how these indicators may be measured, and how the resulting measurements can be meaningfully interpreted. At this point it is useful to look at numerous popular network management and measurement tools and examine their ability to provide useful measurements. There are two basic approaches to this task; one is to collect management information from the active elements of the network using a management protocol, and from this information make some inferences about network performance. This can be termed a *passive approach* to performance measurement, in that the approach attempts to measure the performance of the network without disturbing its operation. The second approach is to use an active approach and inject test traffic into the network and measure its performance in some fashion, and relate the performance of the test traffic to the performance of the network in carrying the normal payload.

Measuring Performance with SNMP

In IP networks the ubiquitous network management tool is the *Simple Network Management Protocol* (SNMP). There is no doubt that SNMP can provide a wealth of data about the operational status of each management network element, but can it tell you anything about the overall network performance?

The operation of SNMP is a *polling* operation, where a management station directs periodic polls to various managed elements and collects the responses. These responses are used to update a view of the operating status of the network.

The most basic tool for measuring network performance is the periodic measurement of the interface byte counters. Such measurements can provide a picture of the current traffic levels on the network link, and when related to the total capacity of the link, the relative link loading level can be provided. As a performance indicator this relative link loading level can provide some indication of link performance, in that a relatively lightly loaded link (such as a load of 5 to 10 percent of total available capacity) would normally indicate a link that has no significant performance implications, whereas a link operating at 100 percent of total available capacity would likely be experiencing high levels of packet drop, queuing delay, and potentially a high jitter level. (Figure 1) In between these two extremes there are performance implications of increasing the load. Of course it should be noted that the characteristics of the link have a bearing on the interpretation of the load levels, and a low-latency 10-Gbps link operating at 90-percent load will have very significantly lower levels of performance degradation than a 2-Mbps high-latency link under the same 90-percent load. (Figure 2)

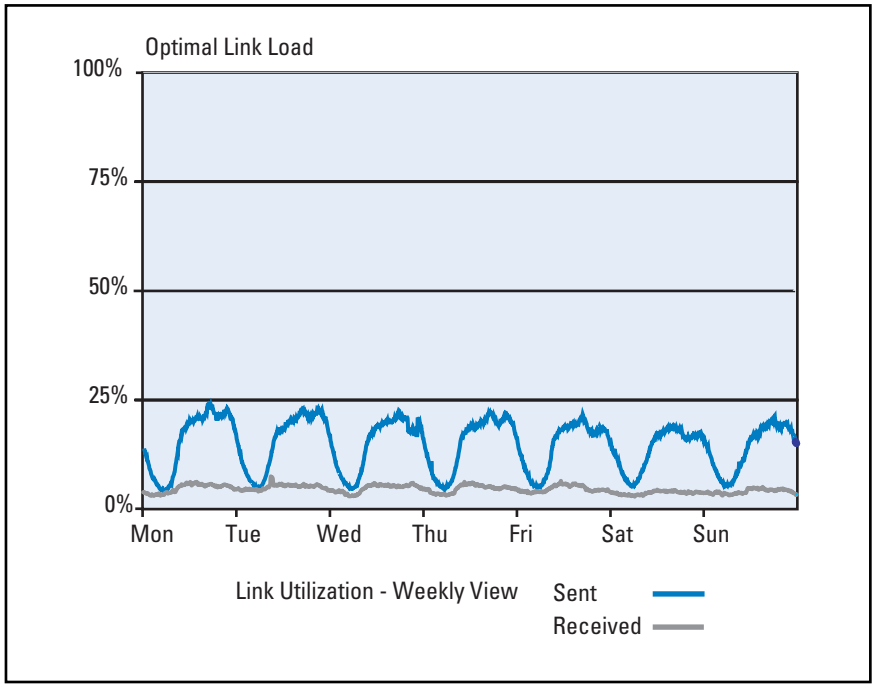


Figure 1a: Relative Link Loading – An Optimally Loaded Link

Figure 1b: Relative Link Loading – A Maximally Loaded Link

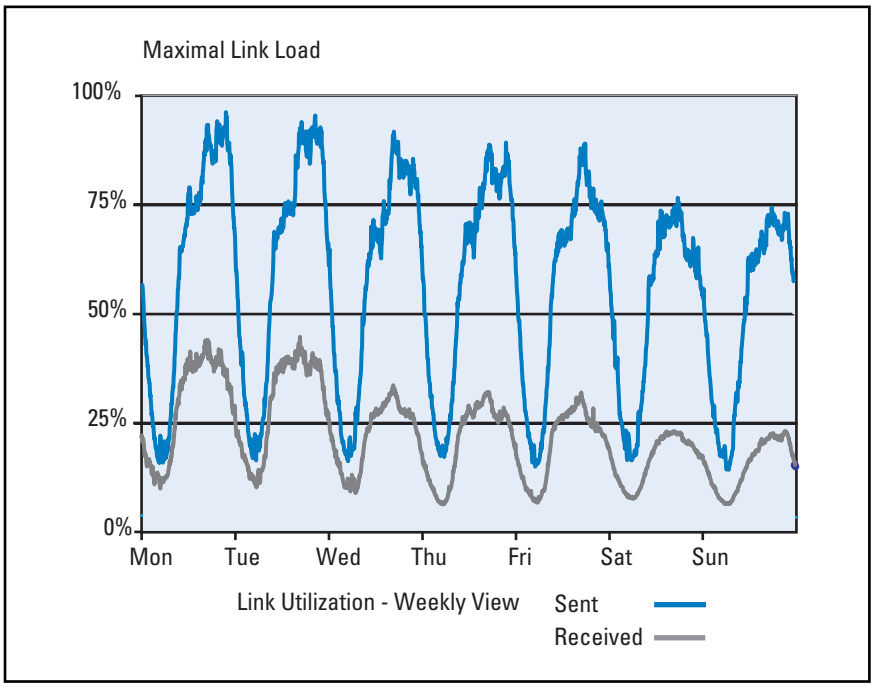


Figure 1c: Relative Link Loading – Highly Degraded Link

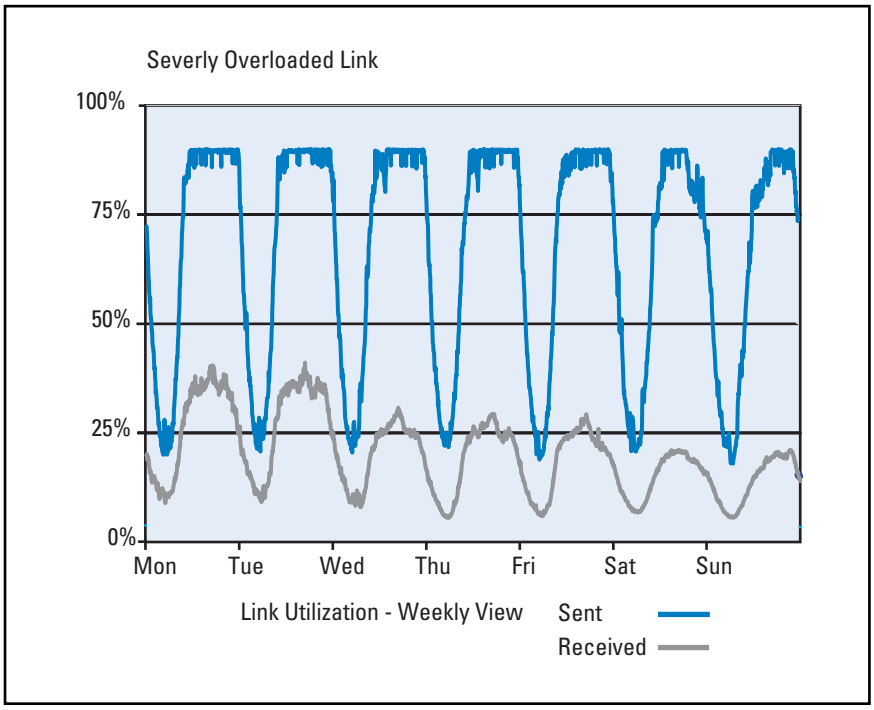
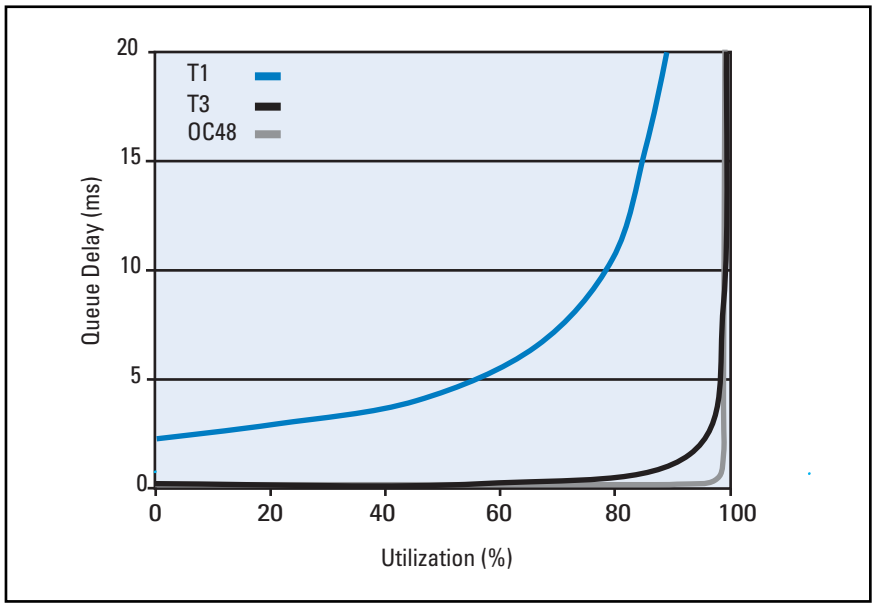


Figure 2: Queuing Delay Simulation (David Meyer, Sprint, November 2002)



Relative traffic load on each link can be complemented by measurement of performance-related SNMP counters. A management system can poll each active network element to retrieve the number of packets dropped for each interface, and the number of packets successfully forwarded. From these two data items, the relative drop proportion of packets can be calculated on an element-by-element and potentially a link-by-link basis, and a series of element measures can provide a per-path drop proportion by combining the individual packet-forwarding measurements for the interfaces on the path.

Because some count of relative packet drop rate can be gathered from each network element, with the additional input of the current forwarding state of the network it is possible to predict the path a packet will take through the network, and hence estimate the path probability of drop. However, this information is still well short of being a reliable measurement of service performance.

Queuing delay is somewhat more challenging to measure on an element-by-element basis using element polling with SNMP. In theory, the polling system could use a rapid sequence of polling the output queue length of a router and estimating the queuing delay based on an average packet size estimate, together with the knowledge of the available output capacity. Of course, such a measurement methodology assumes a simple *first-in, first-out* (FIFO) queuing discipline, a queue size that varies slowly over time, and slow link speeds. Such assumptions are rarely valid in today's IP networks. As the link speed increases, the queue size may oscillate with a relatively high frequency as a function of both the number and capacity of the input systems and of the capacity of the output system. In general, queuing delay is not easily measured using network element polling.

There is no ready way for a polling mechanism to detect and count the incidence of reordered packets. Packet reordering occurs in many situations, including the use of parallel switching fabrics within a single network element and the use of parallel links between routers.

IP routers are not typically designed to detect, let alone correct, packet reordering and because they do not detect this condition, they cannot report on the incidence of reordering via SNMP polling.

The generic approach of network management polling systems is that the polling agent, the network management station, is configured with an internal model of the network; status information, gathered through element polling, is integrated to the network model. The correlation of the status of the model to the status of the network itself is intended to be accurate enough to allow operational anomalies in the network to be recognized and flagged. The challenge is that a sequence of snapshots of element status values cannot readily be reconstructed into a comprehensive view of the performance of the network as an entire system, or even as a collection of edge-to-edge paths. Measurement techniques using polling and modeling can track the performance of the individual elements of the network, but they cannot track per-path service levels across the network. The network-element polling approach can indicate whether or not each network element is operating within the configured operational parameters, and alert the network operator when there are local anomalies to this condition. But such a view is best described as *network centric*, rather than service centric. An implicit assumption is that if the network is operating within the configured parameters, then all service-level commitments are being met. This assumption may not be well founded.

The complementary approach to performance instrumentation of network elements is active network probing. This requires the injection of marked packets into the data stream; collection of the packets at a later time; and correlation of the entry and exit packets to infer some information regarding delay, drop, and fragmentation conditions for the path traversed by the packet. The most common probe tools in the network today are *ping* and *traceroute*.

Measuring Performance with Ping

The best known, and most widely used active measurement tool is *ping*. Ping is a very simple tool: a sender generates an *Internet Control Message Protocol* (ICMP) echo request packet, and directs it to a target system. As the packet is sent, the sender starts a timer. The target system simply reverses the ICMP headers and sends the packet back to the sender as an ICMP echo reply. When the packet arrives at the original sender's system, the timer is halted and the elapsed time is reported. An example ping output is shown in Figure 3.

Figure 3: Example Ping Report

```
% ping www.iab.org
PING www.iab.org (132.151.6.25): 56 data bytes
64 bytes from 132.151.6.25: icmp_seq=0 ttl=44 time=254.409 ms
64 bytes from 132.151.6.25: icmp_seq=1 ttl=44 time=254.197 ms
64 bytes from 132.151.6.25: icmp_seq=2 ttl=44 time=255.238 ms
64 bytes from 132.151.6.25: icmp_seq=3 ttl=44 time=255.874 ms
--- www.iab.org ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 254.197/254.930/255.874/0.670 ms
```

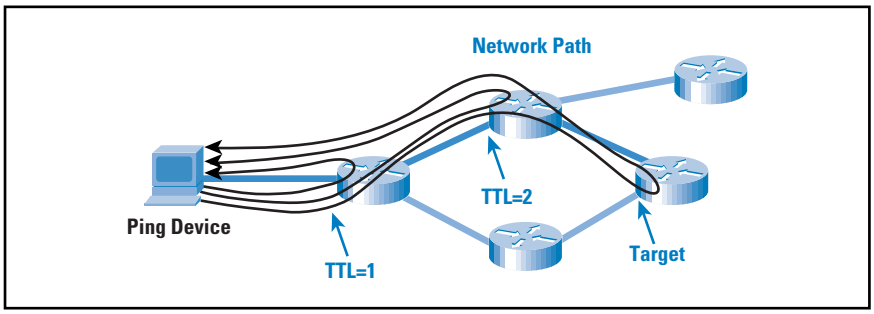
This simple active sampling technique can reveal a wealth of information. A ping response indicates that the target host is connected to the network, is reachable from the query agent, and is in a sufficiently functional state to respond to the ping packet. In itself, this response is useful information, indicating that a functional network path to the target host exists. Failure to respond is not so informative because it cannot be reliably inferred that the target host is not available. The ping packet, or perhaps its response, may have been discarded within the network because of transient congestion, or the network may not have a path to the target host, or the network may not have a path back to the ping sending host, or there may be some form of firewall in the end-to-end path that blocks the ICMP packet from being delivered.

However, if you can ping a remote IP address, then you can obtain numerous performance metrics. Beyond simple reachability, further information can be inferred by the ping approach with some basic extensions to our simple ping model. If a sequence of labeled ping packets is generated, the elapsed time for a response to be received for each packet can be recorded, along with the count of dropped packets, duplicated packets, and packets that have been reordered by the network. Careful interpretation of the response times and their variance can provide an indication of the load being experienced on the network path between the query agent and the target. Load will manifest a condition of increased delay and increased variance, due to the interaction of the router buffers with the traffic flows along the path elements as load increases. When a router buffer overflows, the router is forced to discard packets; and under such conditions, increased ping loss is observed. In addition to indications of network load, high erratic delay and loss within a sequence of ping packets may be symptomatic of routing instability with the network path oscillating between many path states.

A typical use of ping is to regularly test numerous paths to establish a baseline of path metrics. This enables a comparison of a specific ping result to these base metrics to give an indication of current path load within the network.

Of course, it is possible to interpret too much from ping results, particularly when pinging routers within a network. Many router architectures use fast switching paths for data packets, whereas the central processing unit of the router may be used to process ping requests. The ping response process may be given a low scheduling priority because router operations represent a more critical router function. It is possible that extended delays and loss, as reported by a ping test, may be related to the processor load or scheduling algorithm of the target router processor rather than to the condition of the network path. (Figure 4)

Figure 4: Ping Path



Ping sequences do not necessarily mimic packet flow behavior of applications. Typical TCP flow behavior is prone to cluster into bursts of packet transmissions on each epoch of the round-trip time. Routers may optimize their cache management, switching behavior, and queue management to take advantage of this behavior. Ping packets may not be clustered; instead, an evenly spaced pacing is used, meaning that the observed metrics of a sequence of ping packets may not exercise such router optimizations. Accordingly, the ping results may not necessarily reflect an anticipation of application performance along the same path. Also a ping test does not measure a simple path between two points. The ping test measures the time to send a packet to a target system and for the target to respond back to the sender. Ping is measuring a loop rather than a simple path.

With these caveats in mind, monitoring a network through regular ping tests along the major network paths can yield useful information regarding the status of the network service performance.

Many refinements to ping can extend its utility. Ping can use *loose source routing* to test the reachability of one host to another, directing the packet from the query host to the loose source routed host, then to the target host and back via the same path through the specified approach. However, many networks disable support for loose source routing, given that it can be exploited in some forms of security attacks. Consequently, the failure of a loose source routed ping may not be a conclusive indication of a network fault.

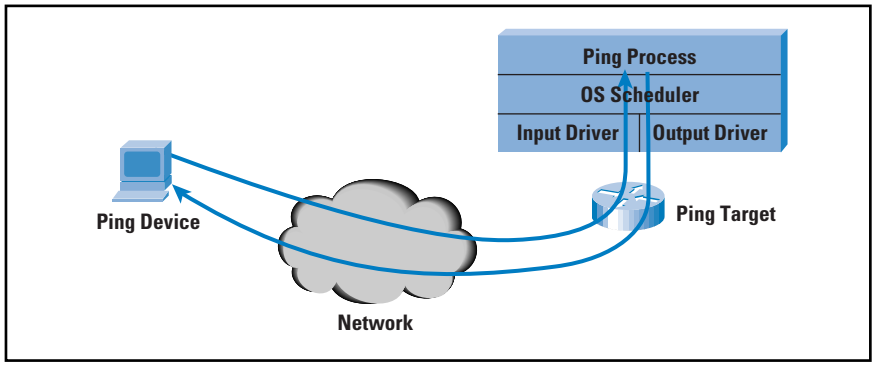
Ping also can be used in a rudimentary way to discover the provisioned capacity of network links. By varying the packet length and comparing the ping times of one router to the next-hop router on a path, the bandwidth of the link can be deduced with some degree of approximation required because of a background queue-induced level of network jitter.

A more sophisticated variation of ping is to pace the transmission of packets from the received packets, mimicking the behavior of the TCP flow control algorithms with *Slow Start* and subsequent congestion avoidance. *Treno* is such a tool. In Treno, the transmission of ping packets is managed by the TCP Reno flow-control algorithm, such that further ping packets are triggered by the reception of responses to earlier packets, and the triggering of further packets is managed by an implementation of the TCP control function. Such a tool can indicate available flow rate-managed capacity on a chosen path.

Path Discovery Using Traceroute

The second common ICMP-based network management tool, *traceroute*, devised by Van Jacobson, is based on the ICMP *Time Exceeded* message. Here, a sequence of *User Datagram Protocol* (UDP) packets are generated to the target host, each with an increased value of the *Time To Live* (TTL) field in the IP header. This generates a sequence of ICMP Time Exceeded messages sourced from the router where the TTL expired. These source addresses are those of the routers, in turn, on the path from the source to the destination. (Figure 5)

Figure 5: Traceroute Path



Like ping, traceroute measures the elapsed time between the packet transmission and the reception of the corresponding ICMP packet. In this way, the complete output of a traceroute execution exposes not only the elements of the path to the destination, but also the delay and loss characteristics of each partial path element. Traceroute also can be used with loose source route options to uncover the path between two remote hosts. The same caveats mentioned in the ping description relating to the relative paucity in deployment of support for loose source routing apply. An example of a traceroute report is shown in Figure 6.

Figure 6. Traceroute report

```
% traceroute www.cisco.com
traceroute to www.cisco.com (198.133.219.25), 64 hops max, 40 byte packets
1  dickson-gw1.Canberra.telstra.net (203.50.0.1)  0.272 ms  0.265 ms  0.270 ms
2  GigabitEthernet4-1.civ12.Canberra.telstra.net (203.50.8.1)  0.402 ms  0.272 ms  0.259 ms
3  GigabitEthernet3-1.civ-core2.Canberra.telstra.net (203.50.7.5)  0.214 ms  0.227 ms  0.193 ms
4  GigabitEthernet2-2.dkn-core1.Canberra.telstra.net (203.50.6.126)  0.459 ms  0.394 ms  0.385 ms
5  Pos4-0.ken-core4.Sydney.telstra.net (203.50.6.121)  3.806 ms  3.762  ms  3.770 ms
6  Pos2-0.pad-core4.Sydney.telstra.net (203.50.6.22) 3.907 ms  3.959 ms  3.913 ms
7  GigabitEthernet0-1.syd-core01.Sydney.net.reach.com (203.50.13.246)  3.898 ms  3.866 ms  3.977 ms
8  i-13-2.sjc-core01.net.reach.com (202.84.143.41)  191.361 ms  191.365 ms  191.341 ms
9  sl-st21-sj-6-1.sprintlink.net (144.223.242.1) 186.955 ms  186.851 ms  187.010 ms
10  sl-bb25-sj-5-1.sprintlink.net (144.232.20.73)  187.241 ms  187.337 ms  187.055 ms
11  sl-gw11-sj-10-0.sprintlink.net (144.232.3.134)  187.279 ms  186.898 ms  186.821 ms
12  sl-ciscopsn2-11-0-0.sprintlink.net (144.228.44.14)  187.572 ms  187.495 ms  187.620 ms
13  sjck-dirty-gw1.cisco.com (128.107.239.5)  184.533 ms  184.686 ms  184.694 ms
14  sjck-sdf-ciod-gw1.cisco.com (128.107.239.106)  184.676 ms  184.686 ms  184.644 ms

15  www.cisco.com (198.133.219.25)  185.017 ms 185.122 ms 185.019 ms
```

Notes:
1) There are interprovider handovers at hops 7, 9, and 13.
2) There is a sudden jump in response times at hop 8. The additional 182 ms of round-trip latency corresponds to a 36,000-km submarine cable path. This can be explained by the hop-7 to hop-8 segment, including a submarine cable path between Australia and the United States.

Traceroute is an excellent tool for reporting on the state of the routing system. It operates as an excellent "sanity check" of the match between the design intent of the routing system and the operational behavior of the network.

The caveat to keep in mind when interpreting traceroute output has to do with asymmetric routes within the network. Whereas the per-hop responses expose the routing path taken in the forward direction to the target host, the delay and loss metrics are measured across the forward and reverse paths for each step in the forward path. The reverse path is not explicitly visible to traceroute.

One-Way Measurements

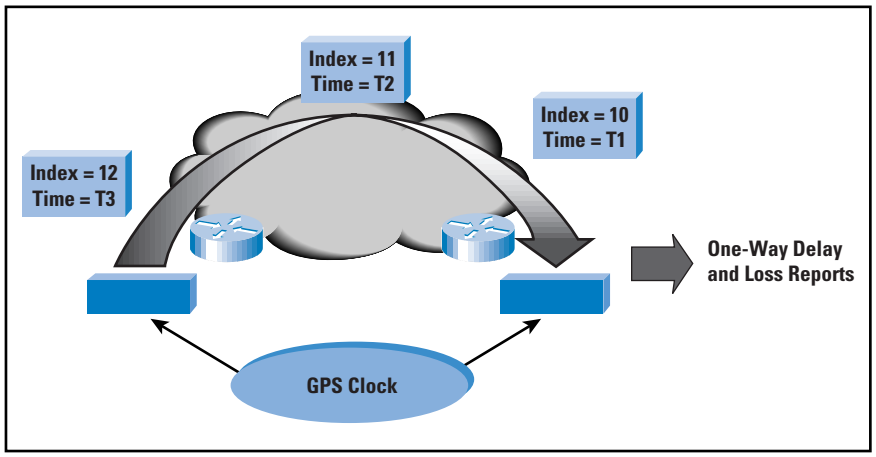
Round-trip probes, such as ping and traceroute, are suited to measuring the total network path between two ends of a transaction, but how can a network provider measure the characteristics of a component of the total end-to-end path? In such a case the network provider is interested in the performance of a set of unidirectional transit paths from an network ingress point to an egress point. There are now some techniques that perform a one-way delay and loss measurement, and they are suited to measuring the service parameters of individual transit paths across a network. A one-way approach does not use a single network management system, but relies on the deployment of probe senders and receivers using synchronized clocks.

The one-way methodology is relatively straightforward. The sender records the precise time a certain bit of the probe packet was transmitted into the network; the receiver records the precise time that same bit arrived at the receiver. Precisely synchronizing the clocks of the two systems is an interesting problem, and initial implementations of this approach have used *Global Positioning System* (GPS) satellite receivers as a synchronized clock source.

One of the noted problems with the use of GPS was that computers are generally located within machine rooms and a clear GPS signal is normally available only on a rooftop. Later implementations of this approach have used the clock associated with the *Code Division Multiple Access* (CDMA) mobile telephone network as a highly accurate, synchronized, distributed clock source, with the advantage that the time signal is usually available close to the measurement unit.

Consequent correlation of the sender's and receiver's data from repeated probes can reveal the one-way delay and loss patterns between sender and receiver. To correlate this to a service level requires the packets to travel along the same path as the service flow and with the same scheduling response from the network.

Figure 7: One-Way Measurements



Ping and traceroute are ubiquitous tools. Almost every device can support sending ping and traceroute probes, and, by default almost every device, including network routers, will respond to a ping or traceroute probe. One-way measurements are a different matter, and such measurements normally require the use of dedicated devices in order to undertake the clocking of the probes with the required level of precision (Figure 7).

Choosing the Right Time Base

Whether it is an active or passive measurement regime, the next basic decision is the time base to use for the measurements. Many applications are very sensitive to short-lived transient network conditions. This may take the form of a burst of packet loss, or a period of packet reordering, or a switch to a longer round trip time. TCP may react by halving its sending rate, or by entering an extended wait state while awaiting the retransmission timer to expire. In either case it will take numerous round trip time intervals for the transport session to recover, and this may impact the behavior of the application. On the other hand, a periodic network probe may miss the transient event altogether and report no abnormalities whatsoever.

IP networks have bursty traffic sources, and there is a marked self-similarity in the traffic patterns. This appears to be consistent over a wide range of networks, where large-capacity systems tend to observe large burst patterns and smaller systems also see bursts of a similar proportionate size. So the question is, what time interval for measurements can provide meaningful aggregation of information, while at the same time be sensitive enough to report on the outcomes of transient bursts within the network? Intuitively a measurement time base of hourly measurements is very insensitive to capturing transient bursts, whereas a time base of a millisecond would generate a massive amount of data, a scenario that would tend to smother the identification of abnormalities. Interestingly enough, the choice of a measurement base has little to do with the capacity of the links within a network, but it has a close relationship to the average routing trip time of the individual transport sessions that are active within the network.

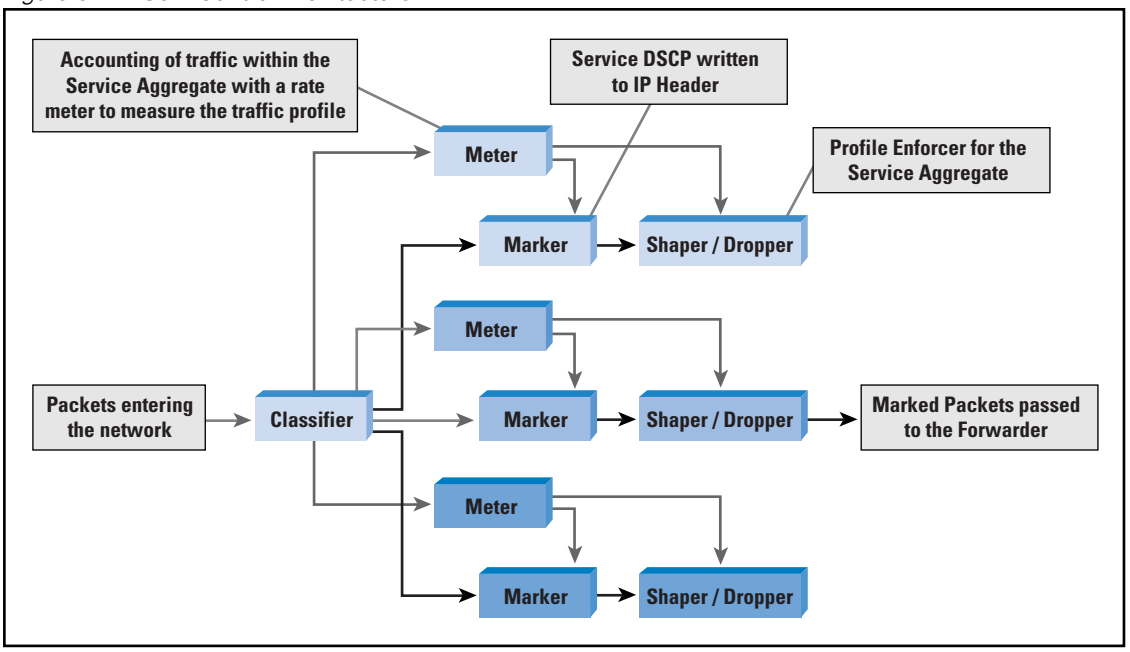
The profile of IP networks is one that is dominated by TCP traffic, and TCP traffic uses a transport control mechanism where the returning stream of *acknowledgement* (ACK) packets governs the actions of the sender. This implies that network-based distortion in the forward data path will not be signaled back to the sender for one complete round-trip time interval, and the consequent adaptation of the sender to the conditions of the network will take numerous additional round-trip times. The implication is that in order to capture a comprehensive view of network performance, a time base of 1 to 2 seconds is appropriate. However, for large networks, such a view generates a massive amount of data. It appears that many networks use a measurement time base of about 60 to 300 seconds, representing an acceptable compromise between sensitivity of the measurement system and the consequent volume of measurement data to analyze.

What About QoS Networks?

So far the assumption has been that the network operates with a single service level and that probes of the network operate at the same service level as the network payload. This is certainly a common situation, but the total picture is slightly broader. When the network provider attempts to create a premium response for certain classes of traffic, and where the customer is paying a premium tariff to use such a premium service, the question of performance becomes a matter of deep concern to both the provider and the customer. After all, the customer is now paying a premium for improved performance, so it would help all concerned if this could be clearly defined and measured.

Solutions exist in both the passive and active polling domains. In the case of SNMP there is a monitoring framework (or Management Information Base, MIB) relating to the *Differentiated Services* (DiffServ) model of *Quality of Service* (QoS), and also MIBs relating to the *Integrated Services* (IntServ) QoS model. For the DiffServ MIB, it is first necessary to define an abstract model of the operation of a DiffServ admission router, by looking at the major functional blocks of the router. The first of these blocks is the definition of the supported behavior aggregates provided by the network. Within the network path, the initial active path element is the traffic classification module, which can be modeled as a set of filters and an associated set of output streams. The output stream is passed to the traffic-conditioning elements, which are the traffic meters and the associated action elements. Many meter profiles can be used in the model: an average data rate, an exponential weighted moving average of one of numerous various traffic profiles that can be expressed by a set of token-bucket parameters using an average rate, a peak rate, and a burst size. More elaborate meter specifications can be constructed using a multilevel token-bucket specification. From the meter, the traffic is passed through an action filter, which may mark the packets and shape the traffic profile through queues or discard operations. Together, this sequence of components forms a *traffic conditioning block*. The traffic is then passed into a queue through the use of a queuing discipline that applies the desired service behavior. (Figure 8)

Figure 8: DIffServ Control Architecture



From this generic model it is possible to define instrumentation for SNMP polling, where each of these five components—the behavior aggregate, the classifier, the meter, profile actions, and the queuing discipline—correspond to a MIB table. With this structure it is possible to parameterize both the specific configuration of the DiffServ network element and its dynamic state. This MIB is intended to describe the configuration and operation of both edge and interior DiffServ network elements, the difference being that interior elements use just a behavior aggregate classifier and a queue manager within the management model, whereas the edge elements use all components of the model.

A comparable MIB is defined for the IntServ architecture and an additional MIB for the operation of guaranteed services. The IntServ MIB defines the per-element reservation table used to determine the current reservation state, an indication of whether or not the router can accept further flow reservations, and the reservation characteristics of each current flow. No performance polling parameters or accounting parameters are included in the MIB. The guaranteed services MIB adds to this definition with a per-interface definition of a backlog. This is a means of expressing *packet quantization delay*, a delay term, which is the packet propagation delay over the interface, and a slack term, which is the amount of slack in the reservation that can be used without redefining the reservation. Again, these are per-element status definitions, and they do not include performance or accounting data items.

The IntServ MIB is being further defined as a *Resource Reservation Protocol* (RSVP) MIB for the operation of IntServ network elements[14]. There are a larger number of objects within the MIB, including General Objects, Session Statistics Table, Session Sender Table, Reservation Requests Received Table, Reservation Requests Forwarded Table, RSVP Interface Attributes Table, and an RSVP Neighbor Table.

Interestingly, the MIB proposes a writeable RSVP reservation table to allow the network manager to manually create a reservation state that can be removed only through a comparable manual operation. The MIB enables a management system to poll the IntServ network element to retrieve the status of every active IntServ reserved flow and the operational characteristics of the flow, as seen by the network element.

In a QoS DiffServ environment, ping and traceroute pose some interesting engineering issues. Ping sends an ICMP packet. The network QoS admission filters may choose a different classification for these packets from that chosen for normal data-flow TCP or UDP protocol packets; as a result, the probe packet may be scheduled differently or even take a completely different path to the network. In an IntServ QoS network, the common classification condition for a flow is a combination of the IP header source and destination addresses and the TCP or UDP header source and destination port addresses. The ping probe packet cannot reproduce this complete flow description, and therefore cannot, by default, be inserted into the flow path that it is attempting to measure. With traceroute, the packet does have a UDP protocol address, but it uses a constant port address by default, causing a similar problem of attempting to be inserted to an IntServ flow. DiffServ encounters similar problems when attempting to pass the probe packet into the network via the DiffServ admission classification systems. Inside the network, it is possible to insert the probe packet into the network with the IP *Differentiated Services Code Point* (DSCP) field set to the DiffServ behavior aggregate that is being measured.

The measurement of delay and loss taken by ping and traceroute is a cumulative value of both the forward and return path delay and loss. When attempting to measure unidirectional flow-path behavior, such as an IntServ flow path, this measurement is of dubious value, given the level of uncertainty as to which part of the path, forward or reverse, contributed to the ping or traceroute delay and loss reports.

For one-way delay measurements, in DiffServ networks, this can be done within the network, setting the DSCP field to the value of the service aggregate being monitored. Of course, from the customer's perspective, the DiffServ network service profile includes the admission traffic-conditioning block, and the interior one-way measurements are only part of the delivered service. In the IntServ network, the packets have to be structured to take the same path as the elevated service flows; they are classified by each element as part of the collection of such elevated service flows for the purposes of scheduling.

Measuring Performance—The Client Perspective

From the client's perspective, the measurement choices are more limited. A client does not normally enjoy the ability to poll network elements within a provider's network. One way for a client to measure service quality is to instigate probing of the network path, whereby a sender can pass a probe packet into the network and measure the characteristics of the response. Of course, the problems of inserting probe packets into the service flow remain, as do the issues of unidirectional elevated service flows with bidirectional probes.

However, the client does have the advantage of being able to monitor and manipulate the characteristics of the service flow itself. For TCP sessions, the client can monitor the packet retransmission rate, the maximum burst capacity, the average throughput, the *round-trip time* (RTT), RTT variance, and misordered packets, by monitoring the state of the outbound data flow and relating it to the inbound ACK flow. For UDP sessions, there is no corresponding transport-level feedback information flow to the sender as a part of the transport protocol itself. The receiver can measure the service quality of the received datastream using information provided in the *Real-Time Protocol* (RTP) information feedback fields—if RTP is being used for real-time data or as an application-related tool for other application types. If sender and receiver work in concert, the receiver can generate periodic quality reports and pass these summaries back to the sender. Such applications can confirm whether an application is receiving a specified level of service. This approach treats the network like a black box; no attempt is made to identify the precise nature or source of events that disrupt the delivered service quality. There are no standardized approaches to this activity, but numerous analysis tools are available for host platforms that perform these measurements.

Though the client can measure and conform service quality on a per-application level of granularity, the second part of the client's motivation in measuring service quality is more difficult to address. The basic question is whether the service delivered in response to a premium service request is sufficiently differentiated from a best-effort service transaction. Without necessarily conducting the transaction a second time, the best approach is to use either one-way delay probes, for unidirectional traffic, or a bulk TCP capacity probe, to establish some indication of the relativity in performance. From a client perspective none of these are simple to set up, and the dilemma that the customer often faces is the basic question of whether the cost of operating the measurement setup is adequately offset by the value of the resulting answers.

Measuring Networks—Looking for Problems

So far we have been looking at the ways of measuring network performance as a general task. Of course degraded performance does not happen by accident (well, sometimes accidents do happen), and it makes the measurement task easier if you can identify precisely what it is that you are looking for. This approach requires identification of the various situations that can impact network performance and then set up network measurement and monitoring systems that are tuned to identify these situations.

Within this approach, the motives for network measurement are concerned with identification of traffic load patterns that cause uneven network load, monitoring, and verification of service-level agreements, detection of abnormal network load that may be a signature of an attack, forecasting and capacity planning, and routing stability.

The objective here is to create a stable and well-understood model of the operational characteristics of the network, and then analyze the situations that could disrupt this stable state and the implications in terms of delivered performance under such conditions.

Such an approach could be described in terms of opposites—instead of measuring network performance, the approach is measuring the network to identify the conditions that cause nonperformance at particular times within particular network paths. As a performance management technique, this approach has been very effective—rather than taking a larger amount of performance data and merging and averaging it into a relatively meaningless index, the approach is to isolate those circumstances where performance is compromised and report on these exceptions rather than on the remainder of the time.

Of course measuring what is "normal" may involve more than assembling a benchmark set of SNMP-derived polling data and a collection of latency, loss, and jitter profiles obtained from analysis of large volumes of ping data. One additional tool is the router itself. Because the router uses many IP packet header fields to switch each packet, one approach is to get the router to assemble and aggregate information about the characteristics of traffic that has been passed through the router, and send these aggregated reports to a network management station for further analysis. *NetFlow* is the most common tool to undertake this form of reporting. Like SNMP, NetFlow can report on the characteristics of traffic as it passes a point in the network. For measuring end-to-end performance of individual applications, NetFlow has the same limitations as SNMP. The analogy is one of standing on a street corner counting cars that go past and from that measurement attempting to derive the average time for a commuter to drive to or from work. However, the value of NetFlow is that in this context of performance measurement, it can be used to derive a picture of the baseline characteristics of the network, including identification of the endpoints of the traffic flows. Extending the car analogy further, NetFlow can provide an indication of the origins and ultimate destinations of the cars as they pass the monitoring point. This information is useful in terms of designing networks that are adequately configured to handle the transit traffic load. In addition, with careful analysis, NetFlow can be used to identify exceptional traffic conditions. The advantage here is that NetFlow data can be used to identify both the abnormal traffic load and also provide some indication of the endpoints of the abnormal flows. In this way, NetFlow can be deployed as both a baseline network traffic profile benchmarking tool and a performance exception diagnosis tool.

This approach of capturing the packet header information as the traffic passes a monitoring point in the network has been implemented in numerous ways, and NetFlow is not the only data-collection tool in this space. One interesting approach has been used by NeTraMet, an implementation of the *Internet Engineering Task Force's* (IETF's) *Realtime Traffic Flow Measurement* architecture for traffic flow measurement.

The feature here is a powerful ruleset within the tool that allows the flow collector to be configured to collect information about particular traffic flows and their characteristics. In the context of measuring performance, one of the abilities of the tool is to match the outbound data flow with the inbound acknowledgement stream, allowing an analyzer some ability to infer end-to-end performance of the application based on the collected information.

Where to Go from Here

It is clear that the picture is so far very incomplete. The active probe measurements require either some latitude of interpretation or dedicated instrumentation to take measurements with some necessary level of frequency and precision. The passive approach of probing the active switching elements of the network is constrained by a very basic model of the switching system, so that the collectable values provide only a very indirect relationship to the manner in which the switching element is generating queuing delays and traffic flow instability.

Perhaps what is also increasingly unclear is the relationship between performance and networks in any case. The last few years have seen a massive swing in public Internet platforms away from networks where some level of congestion and contention was anticipated to networks that are extensively overprovisioned, and there packet jitter and loss are simply not encountered. With the ever-decreasing cost of transmission bandwidth in many markets, this environment of abundant network capacity is now also finding its way into various enterprise network sectors. In such worlds of abundant supply and overengineering of networks, there is really little left to measure within the network. The entire question of performance then becomes a question phrased much closer to home: how well is your system tuned to make the most of its resources and those of the server? Often the entire issue with performance is a situation of abundant network resources, abundant local memory and processing resources, and poor tuning of the transport protocol stack. That is, of course, quite properly the subject of another article.

Further Reading

The Internet offers a wealth of material on the topic of network measurement, and the major exercise is undertaking some filtering to get a broad collection of material that encompasses a range of perspectives on this topic. The following sources were used to prepare this article, and are recommended as starting points for further exploration of this topic.

[1] *Internet Performance Survival Guide,* Geoff Huston, Wiley Computer Publishing, 2000.

[2] "IPPM Metrics for Measuring Connectivity," J. Mahdavi, V. Paxson, RFC 2678, September 1999.

[3] "A One-way Delay Metric for IPPM," G. Almes, S. Kalidinki, M. Zeukuaskas, RFC 2679, September 1999.

[4] "A One-way Packet Loss Metric for IPPM," G. Almes, S. Kalidinki, M. Zeukuaskas, RFC 2680, September 1999.

[5] The RIPE Test Traffic Measurement service at:
http://www.ripe.net/ripencc/mem-services/ttm/

[6] Treno, online at:
http://www.psc.edu/networking/treno_info.html

[7] "Trends in Measurement and Monitoring of Internet Backbones," session at the 26th North American Network Operators Group, hosted by D. Meyer,
http://www.nanog.org/mtg-0210/measurement.html,
October 2002.

[8] "Some thoughts on CoS and Backbone Networks," D. Meyer, presentation to the IEPREP Working Group, IETF-55,
http://www.maoz.com/~dmm/IETF55/ieprep/, November 2002.

[9] NetFlow resource page:
http://www.cisco.com/warp/public/732/Tech/nmp/netflow/netflow_techdoc.shtml

[10] Netramet, and many other interesting measurement tools are referenced in a resource page at: **http://www.caida.org/tools**

This area of research is active, and numerous activities are ongoing in the area of research group activities and workshops.

[11] The Internet Research Task Force has an Internet Measurement Research Group. Further details can be found at:
http://www.irtf.org/charters/imrg.html

[12] ACM SIGCOMM, the ACM Special Interest Group on Data Communications, sponsors an Internet Measurement Workshop. Proceeding of the November 2002 workshop can be found at:
http://www.acm.org/sigcomm/imw2002/

[13] The details of the 2003 Passive and Active Measurement Workshop can be found at: **http://www.pam2003.org**

[14] "RSVP Management Information Base using SMIv2," F. Baker, J. Krawczyk, A. Sastry, RFC 2206, September 1997.

GEOFF HUSTON holds a B.Sc. and a M.Sc. from the Australian National University. He has been closely involved with the development of the Internet for the past decade, particularly within Australia, where he was responsible for the initial build of the Internet within the Australian academic and research sector. Huston is currently the Chief Scientist in the Internet area for Telstra. He is also a member of the Internet Architecture Board, and is the Secretary of the APNIC Executive Committee. He is author of *The ISP Survival Guide,* ISBN 0-471-31499-4, *Internet Performance Survival Guide: QoS Strategies for Multiservice Networks,* ISBN 0471-378089, and coauthor of *Quality of Service: Delivering QoS on the Internet and in Corporate Networks,* ISBN 0-471-24358-2, a collaboration with Paul Ferguson. All three books are published by John Wiley & Sons. E-mail: **gih@telstra.net**

The Session Initiation Protocol

by William Stallings

The *Session Initiation Protocol* (SIP), defined in RFC 3261[6], is an application level signaling protocol for setting up, modifying, and terminating real-time sessions between participants over an IP data network. SIP can support any type of single-media or multimedia session, including teleconferencing.

SIP is just one component in the set of protocols and services needed to support multimedia exchanges over the Internet. SIP is the signaling protocol that enables one party to place a call to another party and to negotiate the parameters of a multimedia session. The actual audio, video, or other multimedia content is exchanged between session participants using an appropriate transport protocol. In many cases, the transport protocol to use is the *Real-Time Transport Protocol* (RTP). Directory access and lookup protocols are also needed.

The key driving force behind SIP is to enable Internet telephony, also referred to as *Voice over IP* (VoIP). There is wide industry acceptance that SIP will be the standard IP signaling mechanism for voice and multimedia calling services. Further, as older *Private Branch Exchanges* (PBXs) and network switches are phased out, industry is moving toward a voice networking model that is SIP signaled, IP based, and packet switched, not only in the wide area but also on the customer premises[2, 3].

SIP supports five facets of establishing and terminating multimedia communications:

- *User location:* Users can move to other locations and access their telephony or other application features from remote locations.
- *User availability:* This step involves determination of the willingness of the called party to engage in communications.
- *User capabilities:* In this step, the media and media parameters to be used are determined.
- *Session setup:* Point-to-point and multiparty calls are set up, with agreed session parameters.
- *Session management:* This step includes transfer and termination of sessions, modifying session parameters, and invoking services.

SIP employs design elements developed for earlier protocols. SIP is based on an HTTP-like request/response transaction model. Each transaction consists of a client request that invokes a particular method, or function, on the server and at least one response. SIP uses most of the header fields, encoding rules, and status codes of HTTP. This provides a readable text-based format for displaying information. SIP incorporates the use of a *Session Description Protocol* (SDP), which defines session content using a set of types similar to those used in *Multipurpose Internet Mail Extensions* (MIME).

SIP Components and Protocols

A system using SIP can be viewed as consisting of components defined on two dimensions: client/server and individual network elements. RFC 3261 defines client and server as follows:

- *Client:* A client is any network element that sends SIP requests and receives SIP responses. Clients may or may not interact directly with a human user. User agent clients and proxies are clients.
- *Server:* A server is a network element that receives requests in order to service them and sends back responses to those requests. Examples of servers are proxies, user agent servers, redirect servers, and registrars.

The individual elements of a standard SIP configuration include the following:

- *User Agent:* The user agent resides in every SIP end station. It acts in two roles:
  - User Agent Client (UAC): Issues SIP requests
  - User Agent Server (UAS): Receives SIP requests and generates a response that accepts, rejects, or redirects the request
- *Redirect Server:* The redirect server is used during session initiation to determine the address of the called device. The redirect server returns this information to the calling device, directing the UAC to contact an alternate *Universal Resource Identifier* (URI). A URI is a generic identifier used to name any resource on the Internet. The URL used for Web addresses is a type of URI. See RFC 2396[1] for more detail.
- *Proxy Server:* The proxy server is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, meaning that its job is to ensure that a request is sent to another entity closer to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it.
- *Registrar:* A registrar is a server that accepts REGISTER requests and places the information it receives (the SIP address and associated IP address of the registering device) in those requests into the location service for the domain it handles.
- *Location Service:* A location service is used by a SIP redirect or proxy server to obtain information about a callee's possible location(s). For this purpose, the location service maintains a database of SIP-address/IP-address mappings.

The various servers are defined in RFC 3261 as logical devices. They may be implemented as separate servers configured on the Internet or they may be combined into a single application that resides in a physical server.

Figure 1: SIP Components and Protocols

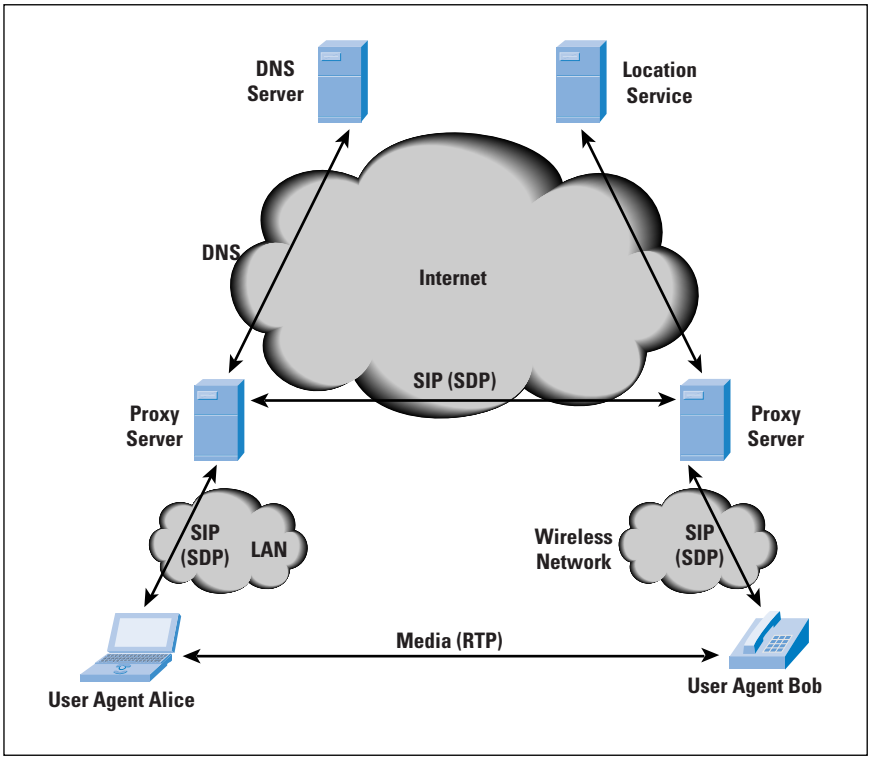


Figure 1 shows how some of the SIP components relate to one another and the protocols that are employed. A user agent acting as a client (in this case UAC Alice) uses SIP to set up a session with a user agent that acts as a server (in this case UAS Bob). The session initiation dialogue uses SIP and involves one or more proxy servers to forward requests and responses between the two user agents. The user agents also make use of the SDP, which is used to describe the media session.

The proxy servers may also act as redirect servers as needed. If redirection is done, a proxy server needs to consult the location service database, which may or may not be colocated with a proxy server. The communication between the proxy server and the location service is beyond the scope of the SIP standard. The *Domain Name System* (DNS) is also an important part of SIP operation. Typically, a UAC makes a request using the domain name of the UAS, rather than an IP address. A proxy server needs to consult a DNS server to find a proxy server for the target domain.

SIP often runs on top of the *User Datagram Protocol* (UDP) for performance reasons, and provides its own reliability mechanisms, but may also use TCP. If a secure, encrypted transport mechanism is desired, SIP messages may alternatively be carried over the *Transport Layer Security* (TLS) protocol.

Associated with SIP is the SDP, defined in RFC 2327[4]. SIP is used to invite one or more participants to a session, while the SDP-encoded body of the SIP message contains information about what media encodings (for example, voice, video) the parties can and will use. After this information is exchanged and acknowledged, all participants are aware of the participants' IP addresses, available transmission capacity, and media type. Then, data transmission begins, using an appropriate transport protocol. Typically, the RTP is used. Throughout the session, participants can make changes to session parameters, such as new media types or new parties to the session, using SIP messages.

SIP Universal Resource Indicators

A resource within a SIP configuration is identified by a URI. Examples of communications resources include the following:

- A user of an online service
- An appearance on a multiline phone
- A mailbox on a messaging system
- A telephone number at a gateway service
- A group (such as "sales" or "help desk") in an organization

SIP URIs have a format based on e-mail address formats, namely **`user@domain`**. There are two common schemes. An ordinary SIP URI is of the form:

```
sip:bob@biloxi.com
```

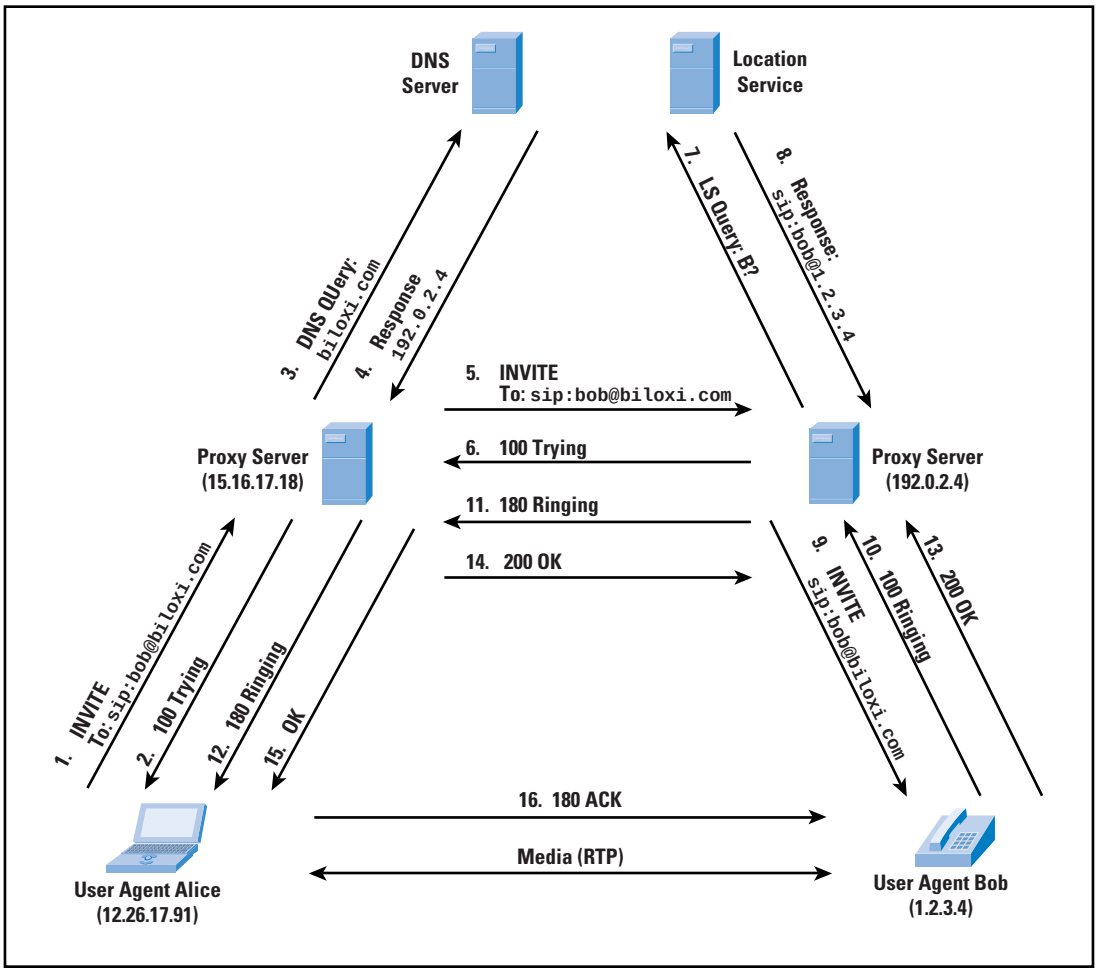
The URI may also include a password, port number, and related parameters. If secure transmission is required, "**`sip:`**" is replaced by "**`sips:.`**" In the latter case, SIP messages are transported over TLS.

Examples of Operation

The SIP specification is quite complex; the main document, RFC 3261, is 269 pages long. To give some feel for its operation, we present a few examples.

Figure 2 shows a successful attempt by user Alice to establish a session with user Bob, whose URI is **`bob@biloxi.com.`**[9] Alice's UAC is configured to communicate with a proxy server (the outbound server) in its domain and begins by sending an INVITE message to the proxy server that indicates its desire to invite Bob's UAS into a session (1); the server acknowledges the request (2). Although Bob's UAS is identified by its URI, the outbound proxy server needs to account for the possibility that Bob is not currently available or that Bob has moved. Accordingly, the outbound proxy server should forward the INVITE request to the proxy server that is responsible for the domain **`biloxi.com.`** The outbound proxy thus consults a local DNS server to obtain the IP address of the **`biloxi.com`** proxy server (3), by asking for the DNS SRV resource record that contains information on the proxy server for **`biloxi.com.`**

Figure 2: SIP Successful Call Setup

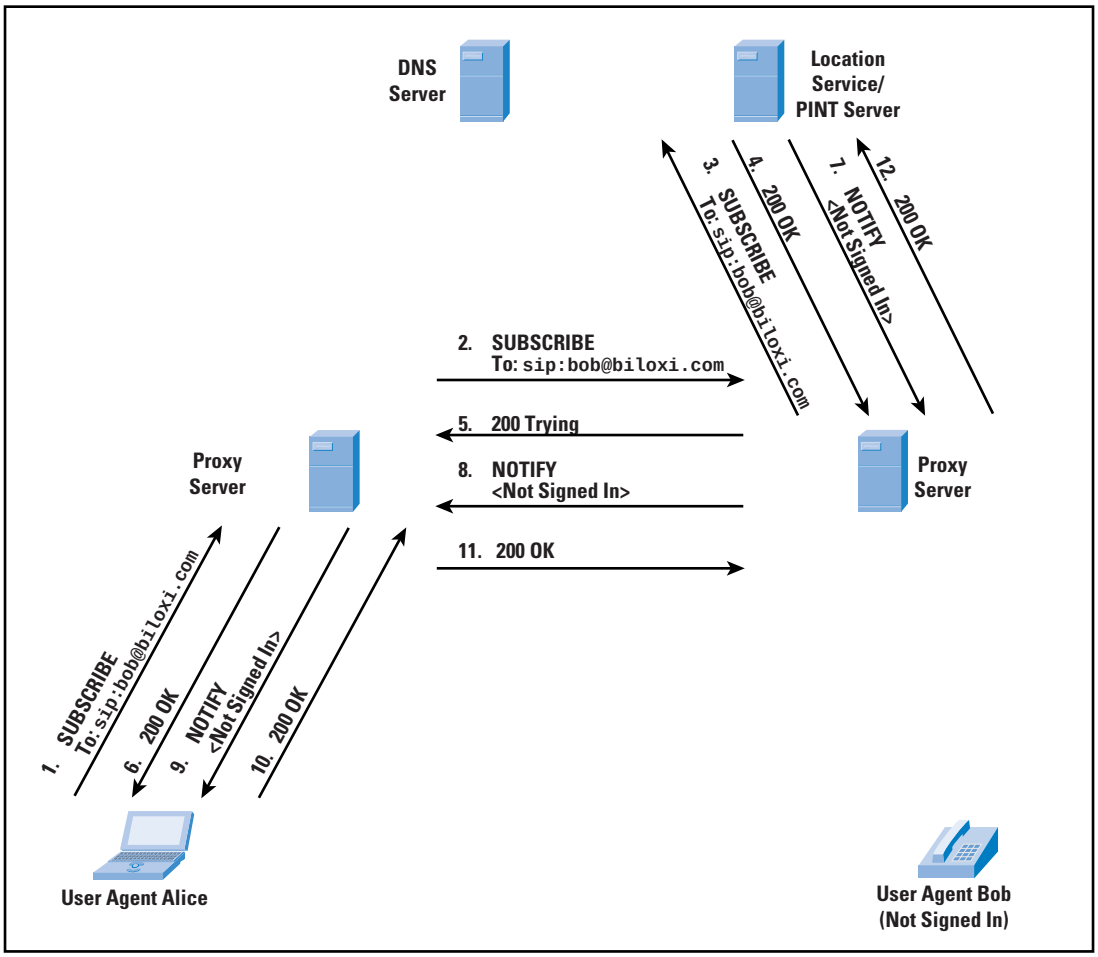


The DNS server responds (4) with the IP address of the **`biloxi.com`** proxy server (the inbound server). Alice's proxy server can now forward the INVITE message to the inbound proxy server (5), which acknowledges the message (6). The inbound proxy server now consults a location server to determine Bob's location Bob (7), and the location server responds with Bob's location, indicating that Bob is signed in, and therefore available for SIP messages (8).

The proxy server can now send the INVITE message on to Bob (9). A ringing response is sent from Bob back to Alice (10, 11, 12) while the UAS at Bob is alerting the local media application (for example, telephony). When the media application accepts the call, Bob's UAS sends back an OK response to Alice (13, 14, 15).

Finally, Alice's UAC sends an acknowledgement message to Bob's UAS to confirm the reception of the final response (16). In this example, the ACK is sent directly from Alice to Bob, bypassing the two proxies. This occurs because the endpoints have learned each other's address from the INVITE/200 (OK) exchange, which was not known when the initial INVITE was sent. The media session has now begun, and Alice and Bob can exchange data over one or more RTP connections.

Figure 3: SIP Presence Example

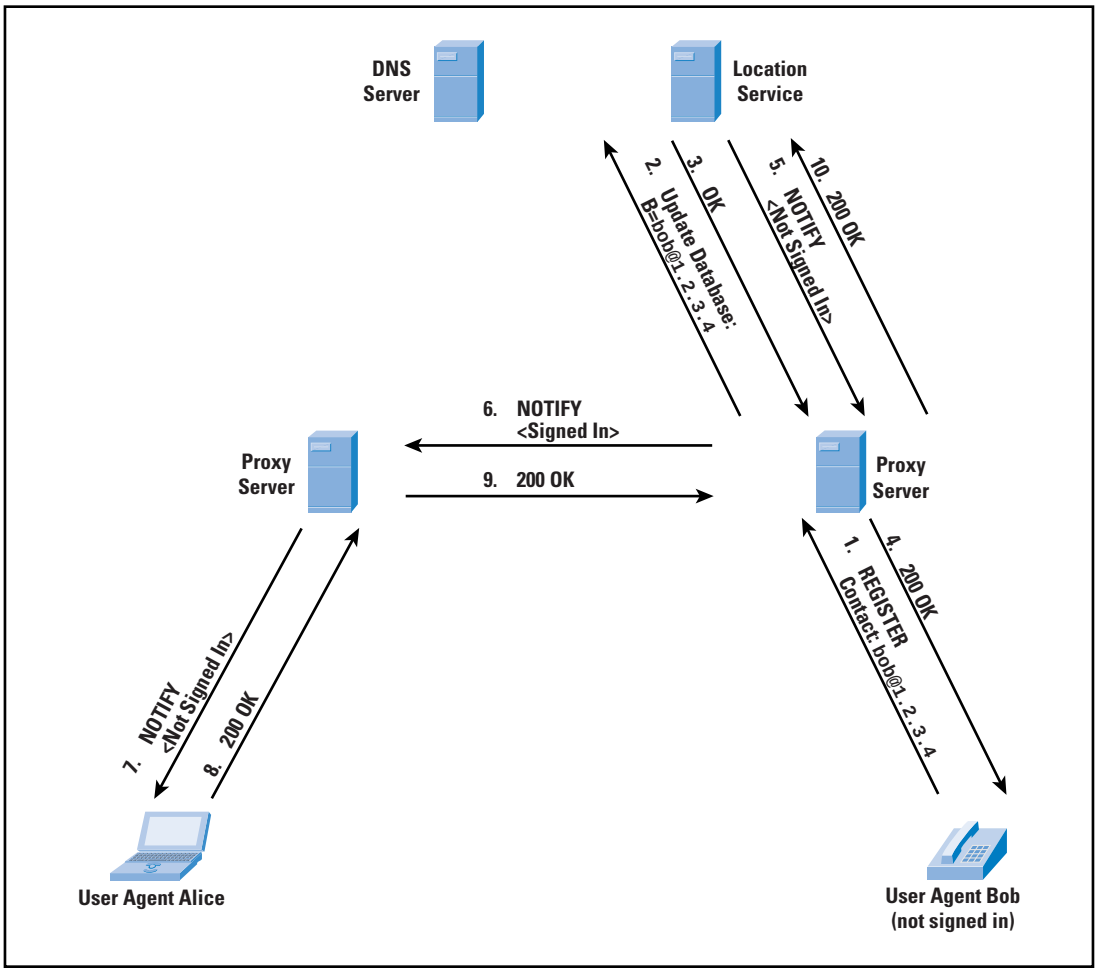


The next example (Figure 3) makes use of two message types that are not yet part of the SIP standard but that are documented in RFC 2848[5] and are likely to be incorporated in a later revision of SIP. These message types support telephony applications. Suppose that in the preceding example, Alice was informed that Bob was not available. Alice's UAC can then issue a SUBSCRIBE message (1), indicating that it wants to be informed when Bob is available.

This request is forwarded through the two proxies in our example to a PINT (*Public Switched Telephone Network [PSTN]-Internet Networking*) server (2, 3). A PINT server acts as a gateway between an IP network from which comes a request to place a telephone call and a telephone network that executes the call by connecting to the destination telephone. In this example, we assume that the PINT server logic is colocated with the location service. It could also be the case that Bob is attached to the Internet rather than a PSTN, in which case the equivalent of PINT logic is needed to handle SUBSCRIBE requests. In this example, we assume the latter and assume that the PINT functionality is implemented in the location service. In any case, the location service authorizes subscription by returning an OK message (4), which is passed back to Alice (5, 6). The location service then immediately sends a NOTIFY message with Bob's current status of not signed in (7, 8, 9), which Alice's UAC acknowledges (10, 11, 12).

Figure 4 continues the example of Figure 3. Bob signs on by sending a REGISTER message to the proxy in its domain (1). The proxy updates the database at the location service to reflect registration (2). The update is confirmed to the proxy (3), which confirms the registration to Bob (4). The PINT functionality learns of Bob's new status from the location server (here we assume that they are colocated) and sends a NOTIFY message containing Bob's new status (5), which is forwarded to Alice (6, 7). Alice's UAC acknowledges receipt of the notification (8, 9, 10).

Figure 4: SIP Registration and Notification Example



SIP Messages

As was mentioned, SIP is a text-based protocol with a syntax similar to that of HTTP. There are two different types of SIP messages, *requests* and *responses*. The format difference between the two types of messages is seen in the first line. The first line of a request has a method, defining the nature of the request and a Request-URI, indicating where the request should be sent. The first line of a response has a response code. All messages include a header, consisting of a number of lines, each line beginning with a header label. A message can also contain a body such as an SDP media description.

For SIP requests, RFC 3261 defines the following methods:

- *REGISTER:* Used by a user agent to notify a SIP configuration of its current IP address and the URLs for which it would like to receive calls
- *INVITE:* Used to establish a media session between user agents
- *ACK:* Confirms reliable message exchanges
- *CANCEL:* Terminates a pending request, but does not undo a completed call
- *BYE:* Terminates a session between two users in a conference
- *OPTIONS:* Solicits information about the capabilities of the callee, but does not set up a call

For example, the header of message (1) in Figure 2 might look like the following:

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP 12.26.17.91:5060
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com
From: Alice <sip:alice@atlanta.com;tag=1928301774
Call-ID: a84b4c76e66710@12.26.17.91
CSeq: 314159 INVITE
Contact: <sip:alice@atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```

The first line contains the method name (**INVITE**), a SIP URI, and the version number of SIP that is used. The lines that follow are a list of header fields. This example contains the minimum required set.

The **Via** headers show the path the request has taken in the SIP configuration (source and intervening proxies), and are used to route responses back along the same path. As the INVITE message leaves, there is only the header inserted by Alice. The line contains the IP address (**12.26.17.91**), port number (**5060**), and transport protocol (**UDP**) that Alice wants Bob to use in his response.

The **Max-Forwards** header limits the number of hops a request can make on the way to its destination. It consists of an integer that is decremented by one by each proxy that forwards the request. If the **Max-Forwards** value reaches 0 before the request reaches its destination, it is rejected with a 483 (**Too Many Hops**) error response.

The **To** header field contains a display name (Bob) and a SIP or SIPS URI (**sip:bob@biloxi.com**) toward which the request was originally directed. The **From** header field also contains a display name (Alice) and a SIP or SIPS URI (**sip:alice@atlanta.com**) that indicate the originator of the request. This header field also has a **tag** parameter that contains a random string (**1928301774**) that was added to the URI by the UAC. It is used to identify the session.

The **`Call-ID`** header field contains a globally unique identifier for this call, generated by the combination of a random string and the host name or IP address. The combination of the **`To`** tag, **`From`** tag, and **`Call-ID`** completely defines a peer-to-peer SIP relationship between Alice and Bob and is referred to as a dialog.

The **`CSeq`** or *Command Sequence* header field contains an integer and a method name. The CSeq number is initialized at the start of a call (**`314159`** in this example), incremented for each new request within a dialog, and is a traditional sequence number. The CSeq is used to distinguish a retransmission from a new request.

The **`Contact`** header field contains a SIP URI for direct communication between user agents. Whereas the **`Via`** header field tells other elements where to send the response, the **`Contact`** header field tells other elements where to send future requests for this dialog.

The **`Content-Type`** header field indicates the type of the message body. The **`Content-Length`** header field gives the length in octets of the message body.

The SIP response types defined in RFC 3261 are in the following categories:

- *Provisional* (1xx): The request was received and is being processed.
- *Success* (2xx): The action was successfully received, understood, and accepted.
- *Redirection* (3xx): Further action needs to be taken in order to complete the request.
- *Client Error* (4xx): The request contains bad syntax or cannot be fulfilled at this server.
- *Server Error* (5xx): The server failed to fulfill an apparently valid request.
- *Global Failure* (6xx): The request cannot be fulfilled at any server.

For example, the header of message (13) in Figure 2 might look like the following:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com
Via: SIP/2.0/UDP 12.26.17.91:5060
To: Bob <sip:bob@biloxi.com;tag=a6c85cf
From: Alice <sip:alice@atlanta.com;tag=1928301774
Call-ID: a84b4c76e66710@12.26.17.91
CSeq: 314159 INVITE
Contact: <sip:bob@biloxi.com>
Content-Type: application/sdp
Content-Length: 131
```

The first line contains the version number of SIP that is used and the response code and name. The lines that follow are a list of header fields. The **`Via`**, **`To`**, **`From`**, **`Call-ID`**, and **`CSeq`** header fields are copied from the INVITE request. (There are three **`Via`** header field values—one added by Alice's SIP UAC, one added by the **`atlanta.com`** proxy, and one added by the **`biloxi.com`** proxy.) Bob's SIP phone has added a **`tag`** parameter to the **`To`** header field. This tag is incorporated by both endpoints into the dialog and is included in all future requests and responses in this call.

Session Description Protocol

The *Session Description Protocol* (SDP), defined in RFC 2327, describes the content of sessions, including telephony, Internet radio, and multimedia applications. SDP includes information about[8]:

- *Media streams:* A session can include multiple streams of differing content. SDP currently defines audio, video, data, control, and application as stream types, similar to the MIME types used for Internet mail.
- *Addresses:* SDP indicates the destination addresses, which may be a multicast address, for a media stream.
- *Ports:* For each stream, the UDP port numbers for sending and receiving are specified.
- *Payload types:* For each media stream type in use (for example, telephony), the payload type indicates the media formats that can be used during the session.
- *Start and stop times:* These apply to broadcast sessions, for example, a television or radio program. The start, stop, and repeat times of the session are indicated.
- *Originator:* For broadcast sessions, the originator is specified, with contact information. This may be useful if a receiver encounters technical difficulties.

Although SDP provides the capability to describe multimedia content, it lacks the mechanisms by which two parties agree on the parameters to be used. RFC 3264[7] remedies this lack by defining a simple offer/answer model, by which two parties exchange SDP messages to reach agreement on the nature of the multimedia content to be transmitted.

References

[1] T. Berners-Lee, R. Fielding, and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, August 1998.

[2] S. Borthick, "SIP Services: Slowly Rolling Forward," *Business Communications Review,* June 2002.

[3] S. Borthick, "SIP for the Enterprise: Work in Progress," *Business Communications Review,* February 2003.

[4] M. Handley and V. Jacobson, "SDP: Session Description Protocol," RFC 2327, April 1998.

[5] S. Petrack and L. Conroy, "The PINT Service Protocol: Extensions to SIP and SDP for IP Access to Telephone Call Services," RFC 2848, June 2000.

[6] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261, June 2002.

[7] J. Rosenberg and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol," RFC 3264, June 2002.

[8] H. Schulzrinne and J. Rosenberg, "The Session Initiation Protocol: Providing Advanced Telephony Access Across the Internet," *Bell Labs Technical Journal*, October-December 1998.

[9] Figures 2 through 4 are adapted from ones developed by Professor H. Charles Baker of Southern Methodist University.

WILLIAM STALLINGS is a consultant, lecturer, and author of over a dozen books on data communications and computer networking. He also maintains a computer science resource site for CS students and professionals at **WilliamStallings.com/StudentSupport.html**. He has a PhD in computer science from M.I.T. His latest book is *Computer Networks, with Internet Protocols and Technology* (Prentice Hall, 2003). His home in cyberspace is **WilliamStallings.com** and he can be reached at **ws@shore.net**

Letters to the Editor

Ruling the Root

Ole,

As a matter of principle, I don't mind having my book *Ruling the Root* reviewed by David Crocker. Mr. Crocker was a significant figure in some of the key events covered in the book. His assessment and opinion of the book had the potential to be quite interesting.

One can only be disappointed with the results, however. The review reveals an inability to rise above partisan sniping and engage rationally with an different view. That, as a matter of policy, is why serious journals don't publish unsolicited reviews of books. Unsolicited reviewers tend to fall into one of two types: unabashed promoters with a personal interest in the success of the book, or people with an axe to grind trying to shoot down a perceived enemy.

I offer a rebuttal only because I think it is vital that the Internet technical community, the presumed readers of *The Internet Protocol Journal,* achieve a higher standard in their discussion of Internet-related policy issues.

Ruling the Root is a serious attempt to analyze the intersection of technology and policy. It offers a way of understanding that intersection based on theories of institutions and property rights. I know that this intersection irritates many engineers, who often harbor a wish that it would go away. By now we should know that it won't. Technical systems raise political issues. Technical people, economists, lawyers, and policy analysts, therefore, must be able to engage in rational dialogue about institutional issues, even when the discussion comes uncomfortably close to home. If we can't, the world is in big trouble.

The review completely misses this big picture. It begins with an attempt to belittle the policy significance of domain name management by inventing a mythical decree that all street names have to be in an obscure language. Crocker's attempt at humor falls flat, given today's headlines. Virtually the same day his review was published a German registrar was ordered to take a domain name away from a Web site with objectionable content. Not too long after, an ICANN Task Force published a WHOIS policy proposal that allows domain names to be shut down after 15 days if someone challenges the accuracy of the contact information, raising issues of privacy and harassment. ICANN regulates the prices of registries and entry into the market for domain name services. No one, not even ICANN itself these days, pretends that domain name administration is an exclusively technical matter.

Instead of engaging on those terms, the review concentrated on factual nitpicking. Take this one: "...the book does not consider NSI's role in ICANN-related political processes." This is an astoundingly inaccurate statement. The index of the book under "Network Solutions" contains 33 listings under 5 separate headings.

The book analyzes at length NSI's origins and ownership changes, its opposition to the IAHC and gTLD-MoU, its implied threat to establish a new root, and its policy conflicts with ICANN and the U.S. Department of Commerce.

Crocker claims that I "[characterize] the pre-ICANN *International Forum for the White Paper* (IFWP) as 'the real arena for arriving at a decision [about the details of the new organization].'" His use of a sentence fragment covers up what appears to be a deliberate distortion. I really wrote that some people viewed the IFWP in that way, while others, notably Joe Sims, Jon Postel, and the Information Technology Association of America, did not; see pages 176–178. I wrote at length about how that basic lack of agreement between adherents of IFWP and followers of IANA over legitimacy led to lasting conflict over ICANN's formation.

Crocker was one of Jon Postel's appointees to the *International Ad Hoc Committee* (IAHC). The review takes issue with my characterization of the IAHC, but unfortunately only to maintain Crocker's fictional self-conceptions. He denies that the IAHC ever claimed that "the root was theirs to dispose of." He also denies that IAHC was intended to be the seed of an alternative DNS governance structure. He's wrong on both counts. There is a voluminous record on this question, comprising contemporary news accounts, e-mail list archives, and my own recorded interviews with principal figures such as Don Heath.

Crocker's assertion that IAHC was "explicitly subordinate to IANA" is rather disingenuous, because IANA's U.S. government funding was ending and IAHC was explicitly perceived by Postel and ISOC as a mechanism for continuing its funding. So IAHC was intended to be the governance and support structure for IANA, just as ICANN now is. Indeed, today's ICANN has many features in common with the IAHC proposal, such as the shared registry concept, the slant toward intellectual property interests, the treatment of TLDs as "public resources," and a compulsory and uniform dispute resolution procedure.

What is really at issue here? It is this: Crocker cannot accept the simple fact that a political battle was under way for control of the root, and Postel/IAHC, as well as NSI and the U.S. government, were contenders for that control. Crocker's review challenges the claim in the book that Postel's root redirection exercise in January 1998 was "apparently" based on "concerns about the direction U.S. policy was taking." This judgment was based on interviews with people who were involved with Postel's effort. Of course I cannot read Postel's mind, but neither can David Crocker. My interpretation of why Postel acted is based on the timing and on evidence drawn from first-hand participants. Crocker offers an alternative interpretation, plausible but based on nothing but his own assertion. There is plenty of room for legitimate debate about historical interpretations. Such debate is useful, however, only if it is aimed at discovering the truth.

Regarding the status of IANA, I am sure we will never agree. I see it fundamentally as a DARPA contractor subject to U.S. governmental authority; Crocker views it in almost mystical terms as the embodiment of the Internet community. He says nothing about who paid the bills. Yet, we are not as far apart on the facts as he wants to make it seem. Contrary to the review, the book does document in great detail how a new community for Internet standards development grew up around the old DARPA-funded cadre of Postel, Cerf, and the IAB, and created its own standards of legitimacy and process. My book doesn't dispute Postel's tremendous respect and legitimacy among the technical community. But when it comes to institutionalizing control and ownership of the name and address roots of the Internet, whoever pays the piper calls the tune. And Postel's ability to perform the IANA functions was supported by U.S. government money from day one.

Hence, it was unrealistic to expect Postel to be exempt from governmental authority after domain names became resources of economic value and produced legal and political conflict over that value. Nor is it correct to imply, as Crocker does, that knowledge of the operational details of a technology automatically confers wisdom as to the correct public policies that should be adopted when that happens. Of course, policy decisions must respect technical facts and technical constraints. It is this relationship between technical system, technical community, and the worlds of business, law, and government that is cntral to the story told by *Ruling the Root.*

Crocker's final stab at discrediting the book involves some rather spurious charges of ethics problems. "In his criticism of dispute-resolution activities, he neglects to mention that he is a paid arbitration panelist," he writes. Crocker here refers to the fact that I was one of the few nonlawyers allowed by WIPO to serve as one of three judges in domain name—trademark disputes brought under ICANN's UDRP. The "pay" he refers to is a $500 or $750 honorarium for each case. I do about ten cases a year. I fail to see any conflict of interest or ethical problem here. Crocker implies that my meager remuneration for assuring that justice is done in UDRP cases somehow corrupts me, but he knows perfectly well that I am an opponent of the UDRP and would happily stop receiving those honoraria if the darn thing went away. Besides, no one is in a better position to understand what is right and what is wrong with UDRP than someone who is involved in the actual cases. I do not even understand what his concern is about the noncommercial DNSO constituency. I deal with it in one sentence in the book, and most of my activity in a "management capacity" (i.e., as an elective representative) came after the book manuscript was written.

—Milton Mueller, Syracuse University
Mueller@syr.edu

The author of the book review responds:

Professor Mueller's response discusses his goals of the book and his opinions of my review, to which he is, of course, entitled. He characterizes *Ruling the Root* as an academic consideration of the policy issues pertaining to the Domain Name Service, which he casts as global Internet administrative services. Note that the tag line to the title of his book, however, casts it more even more generally as "Internet governance." Academic and policy work need to be conducted carefully. Unfortunately, Professor Mueller confuses the issues, rather than elucidating them.

The opening, mythical decree of the review was carefully constructed to make the perspective of the book on communication system administrative policy clear: Professor Mueller confuses an administrative agency, such as ICANN or its telephonic equivalent, with a national government such as Germany. He also confuses control over administrative information, such as names and addresses associated with registrations, with primary content, such as a Web page.

Professor Mueller defends his writing about the IFWP as merely reporting the view of others, rather than being his own advocacy. However, his reporting is highly selective and results in his confusing the difference between tension that was *within* the IFWP process, versus *between* IFWP and IANA. His casting the issue as being with IANA is contrary to the formal documentation of IFWP, and contrary to the style and content of its process. IFWP was not designed, nor was it conducted, as a decision-making body.

Professor Mueller confuses the actions and intent of the IAHC with those of IANA (and ISOC). He claims to have extensive substantiation for his assessment of the IAHC. Yet none that is relevant to this confusion appears in his book or his letter. This omission is in spite of the fact that his view is at odds with the formal charter for the IAHC, the group's published report, and the direct record of the group's actions.

The review cites IANA's community-based authority. Professor Mueller confuses this with a rejection of the importance of funding, which it was not. He further confuses the IETF technical standards specification process with the operations administrative work of IANA. He continues to misunderstand the role of operational expertise in policy planning for critical infrastructure services, and he ignores the particular 15-year history of successful administrative policy activities provided by operations geeks, for DNS and IP addresses.

Lastly, given the minor points that Professor Mueller chose to address in his response, it is curious that he fails to respond to the primary ethics point raised in the review, namely his pattern of erroneous or absent citations that substantially undermine many of the assertions of his book.

—Dave Crocker, Brandenburg InternetWorking
dcrocker@brandenburg.com

Zero Configuration

IPJ,

I recently read Edgar Danielyan's article on Zero Configuration Networking in the December 2002 issue of IPJ. As is always the case, as a journalist Edgar is entitled to hold and express his own opinions, so as I began the article I didn't know whether to expect glowing praise of Zeroconf, or a savage attack. Thankfully I needn't have worried. I found an excellent and well-balanced article.

I have two brief comments to make.

1. Since Edgar wrote his article, the old expired Internet Drafts have been updated. The drafts Edgar worked from discussed names ending in local.arpa. The actual shipping version of Mac OS X 10.2 ("Jaguar") uses names ending in just local. to designate link-local names, (link-local names are locally assigned, unique only within the local link, not required to be globally unique).
2. Edgar expressed the opinion that Zeroconf is only useful on small networks, not large networks.

While Edgar is correct that Zeroconf per se is aimed at solving the "small network" problem, discovering your local peers is useful no matter how big the network. At the recent IETF meeting in San Francisco, there was a large network with full connectivity to the Internet, including IPv6, yet the printers were still advertised using Rendezvous, and for Mac users those printers showed up automatically in the "Printer" popup menu in the print dialogs, with zero configuration.

There is also the issue that Rendezvous (the Apple product) will go beyond just what is required for Zeroconf (the IETF Working Group). Service Discovery, on which Rendezvous is based, doesn't have to be used only with link-local multicast DNS. It can also be used with conventional unicast DNS. For a preview of what the future might hold, you can browse to find an example list of printers at my house. Type: **`nslookup -q=ptr _ipp._tcp.stuartcheshire.org`**

Thanks for publishing a great article.

—Stuart Cheshire, Apple Computer, Inc.
`cheshire@apple.com`

List of Acronyms

DARPA	*Defense Advanced Projects Agency*
DNS	*Domain Name System*
DNSO	*Domain Name Supporting Organization*
gTLD-MoU	*generic Top Level Domain-Memorandum of Understanding*
IAHC	*International Ad Hoc Committee*
IANA	*Internet Assigned Numbers Authority*
ICANN	*Internet Corporation for Assigned Names and Numbers*
IETF	*Internet Engineering Task Force*
IFWP	*International Forum for the White Paper*
ISOC	*Internet Society*
UDRP	*Uniform Domain Name Dispute Resolution Policy*
WIPO	*World Intellectual Property Organization*

Book Review

Troubleshooting Campus Networks

Troubleshooting Campus Networks: Practical Analysis of Cisco and LAN Protocols, by Priscilla Oppenheimer and Joseph Bardwell, Wiley, 2002

It is perhaps rare that a book review would encompass the acknowledgements. A break from tradition here is warranted, though, because both authors reveal up front what every prospective reader should know when faced with a purchase decision: Is this work drawn merely from professional circumstance on the part of the author or does it embody a passion held by the author? Judge for yourself. How often do the words "love," "wonderful," and "protocol analysis" congregate?

Coauthors Priscilla Oppenheimer and Joseph Bardwell consider the spectrum of protocols and technologies likely to be encountered in a campus environment. A campus network, it is said by the authors, is any one that spans buildings (whether or not in an educational setting). Of course, bricks and mortar are functionally transparent to most modern technologies, and thus the definition of campus could easily be narrowed to any collection of departments or perhaps even any collection of LANs. A contrast is simply being made against the larger metropolitan or wide-area arena.

Although this book does include substantial theory and background for context, it is not yet another rehash of how things *ought* to behave in the vacuum of a lab environment (indeed, the authors occasionally express surprise at their own observations). Neither is it a step-by-step troubleshooting checklist for novice network administrators. To generalize the format, a thorough decomposition of the whole into its many parts follows an introductory discussion of the subject protocol or technology. It is next released into the wild and is quietly observed. Some conclusions are then drawn (some by the authors, some by the reader) regarding appropriate and inappropriate behavior. Lastly, possible courses of action in response to poor or abnormal performance or behavior are considered. This, again, is merely a generalization. The authors take great care to keep the discussion interesting and relevant, often doing so by sharing real-world experiences.

Organization

The six pages that comprise chapter 1 seek to set a stage, define a scope, and target an audience. The reviewer would add only that those of us who trade in wide-area networks also stand to gain a great deal from the experience.

If chapter 2 were packaged for individual sale, it would find its way under the Christmas tree of every colleague, customer, and boss this reviewer has ever encountered. Those readers familiar with Ms. Oppenheimer's acclaimed *Top-Down Network Design*[1] may be surprised to find the expression "bottom-up" in any of her work. It is, however, cornerstone not only to the chapter, but also to the remainder of the book.

This seemingly obvious approach to trouble-shooting and analysis could not possibly be emphasized enough according to this reviewer's professional observation.

Chapters 3, 5, and 6 delve into campus datalink layer technologies, protocols, and architectures, including Ethernet, *Spanning-Tree Protocol* (STP), and *Virtual Local-Area Networks* (VLANs). Yawn? The reviewer challenges the reader to finish these three chapters without learning something of considerable value. The Ethernet discussion, for example, breaks from the traditional approach where a cursory review of frame types, cable types, and topologies is deemed sufficient. Where Ethernet came from, where it is going, how it is encoded and presented to the physical layer (and why), and how to interpret frame size distribution using *Remote Monitoring* (RMON) or a protocol analyzer are but a few of the topics considered. Extensive use of protocol analyzer capture files casts new light on STP and VLANs.

Chapter 4 additionally addresses a Layer 2 technology (IEEE 802.11 wireless LANs) but warrants honorable mention. Rare is the *radio frequency* (RF) engineer who possesses a full appreciation for the heretofore all-digital, all-wired campus realm. Perhaps less common would be the network administrator with a capacity to do much other than tune in an FM radio station on a digital set. The authors masterfully string together all the relevant RF concepts, at exactly the right level of detail, to allow for a solid fundamental comprehension of 802.11 networks, technologies, architectures, and deployment. This chapter also would do superbly for anyone with a generic interest in RF units of measurement.

Chapter 7 advances the discussion up to the network layer. Although this may seem common knowledge for readers of a publication such as the IPJ, it is written from the perspective of seasoned protocol analysts. It is worth your time.

Chapter 8 persists at Layer 3 with a thorough discussion of relevant routing protocols. It is again worth noting the emphasis on analysis versus simple textbook theory. It, too, is worthy of your investment.

Chapter 9 rounds out the protocol stack, beginning with an emphasis on Layer 4 protocols *Transmission Control Protocol* (TCP) and *User Datagram Protocol* (UDP). One of the highlights found here is a thorough lesson on TCP window size analysis. Could there perhaps be a little more to this seemingly intuitive concept than you at first thought? The chapter closes following an in-depth consideration of application layer protocols such as the *File Transfer Protocol* (FTP), *Hypertext Transfer Protocol* (HTTP), and the *Domain Name System* (DNS). The fundamental mechanics of these protocols and how they interact with their lower-layer counterparts make for a good page-turner.

Chapters 10, 11, and 12 are dedicated to troubleshooting and analysis of *Internetwork Packet Exchange* (IPX), AppleTalk, and Windows networking, respectively. The latter is arguably the more relevant. The other two are nonetheless interesting and left the reviewer longing for a decent AppleTalk trace file with which to recreate.

Chapter 13, WAN Troubleshooting for LAN Engineers, covers the obvious wide-area technologies and architectures, such as *Integrated Services Digital Network* (ISDN), Frame Relay, and *Synchronous Optical Network* (SONET) in about as much detail as the typical LAN engineer or administrator is likely to tolerate. The subject of WAN analysis warrants a volume or two on its own in any case and thus would have been out of place if explored in much greater detail.

Conclusion

The reading of Troubleshooting Campus Networks is not to be approached as a spectator sport. Although the protocol analyzer screen captures are aplenty, and they suitably complement the lessons, merely thumbing the pages would be an opportunity missed. This reviewer chose a free, open-source protocol analyzer (readily available on the Internet) as a reading companion. Although likely far less capable, particularly in terms of graphing, than the oft-referenced Wildpackets EtherPeek product, it nevertheless affords the reader a Layer 2 through 7 window into a living, breathing network.

It bears mentioning that although "Cisco" appears in the subtitle, vendor neutrality is, on the whole, maintained. The Cisco sanctioned troubleshooting methodology is given brief mention in chapter 2. Coverage of the Cisco proprietary *Interior Gateway Routing Protocol* (IGRP), the *Enhanced IGRP* (EIGRP), and the Cisco Discovery Protocol is included, as is coverage of Cisco's "enhancements" to STP. Lastly, where appropriate, Cisco IOS® "show" and "debug" output is included alongside protocol analyzer screen captures. None of this coverage appears to be included in the spirit of product promotion (bear in mind that this is not a Cisco Press title and that neither author is presently employed by Cisco Systems). Rather, it seems simply to be an acknowledgement that the target audience might very well include candidates for Cisco's professional and expert-level certification programs (and rightly so).

It is probably anticlimactic that the reviewer would offer a strong buy recommendation for those with an interest in the fundamental interworkings of campus protocols and technologies. The authors' enthusiasm for packet capture and analysis is infectious. Mr. Bardwell, in fact, is apparently so infatuated that he is at times moved to poetry. This could well be one for the ages.

—Scott Vermillion, IT Artisans Group
scott@itartisans-group.com

References

[1] *Top-Down Network Design,* Priscilla Oppenheimer, ISBN 1578700698, Cisco Press, 1998.

Call for Papers

The Internet Protocol Journal (IPJ) is published quarterly by Cisco Systems. The journal is not intended to promote any specific products or services, but rather is intended to serve as an informational and educational resource for engineering professionals involved in the design, development, and operation of public and private internets and intranets. The journal carries tutorial articles ("What is…?"), as well as implementation/operation articles ("How to…"). It provides readers with technology and standardization updates for all levels of the protocol stack and serves as a forum for discussion of all aspects of internetworking.

Topics include, but are not limited to:

- Access and infrastructure technologies such as: ISDN, Gigabit Ethernet, SONET, ATM, xDSL, cable fiber optics, satellite, wireless, and dial systems
- Transport and interconnection functions such as: switching, routing, tunneling, protocol transition, multicast, and performance
- Network management, administration, and security issues, including: authentication, privacy, encryption, monitoring, firewalls, trouble-shooting, and mapping
- Value-added systems and services such as: Virtual Private Networks, resource location, caching, client/server systems, distributed systems, network computing, and Quality of Service
- Application and end-user issues such as: e-mail, Web authoring, server technologies and systems, electronic commerce, and application management
- Legal, policy, and regulatory topics such as: copyright, content control, content liability, settlement charges, "modem tax," and trademark disputes in the context of internetworking

In addition to feature-length articles, IPJ will contain standardization updates, overviews of leading and bleeding-edge technologies, book reviews, announcements, opinion columns, and letters to the Editor.

Cisco will pay a stipend of US$1000 for published, feature-length articles. Author guidelines are available from Ole Jacobsen, the Editor and Publisher of IPJ, reachable via e-mail at **ole@cisco.com**

The Internet Protocol Journal

Ole J. Jacobsen, Editor and Publisher

The Internet Protocol Journal is published quarterly by the Chief Technology Office, Cisco Systems, Inc.
www.cisco.com
Tel: +1 408 526-4000
E-mail: ipj@cisco.com

The Internet Protocol Journal, Cisco Systems
170 West Tasman Drive, M/S SJ-7/3
San Jose, CA 95134-1706
USA

ADDRESS SERVICE REQUESTED

PRSRT STD
U.S. Postage
PAID
Cisco Systems, Inc.