Path Diagnosis with IPMP

Matthew Luckie
mjl@wand.net.nz

Tony McGregor
tonym@wand.net.nz

WAND Group, University of Waikato, New Zealand
NLANR/MNA, University of California, San Diego [*]

ABSTRACT

The ability to measure and identify performance fault locations on an Internet path between two hosts is an important first step towards diagnosing and correcting a fault or avoiding fault locations entirely. The ability to identify fault locations on *both* the forward and reverse paths from a single point would be very powerful for both operators and users. Rather than describing a tool for path diagnosis *per se*, this paper describes how one could apply a simple measurement protocol to diagnose faults.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement techniques

General Terms

Measurement, performance

Keywords

Path diagnosis, measurement protocols

1. INTRODUCTION

This paper focuses on locating performance faults such as loss, reordering, and queueing at specific routers along Internet paths from an end user's perspective. These end-user techniques are as valuable to an operator as they are to an end user. Small amounts of loss, reordering, or jitter can have a large impact on TCP [13] and real-time applications [7]. Given the ability to monitor for and diagnose path faults in real time, it might be possible to compare paths and select an alternative route rather than to simply choose the shortest AS path in the case where there is no policy route taking precedence [17]. It will be useful to the readers' understanding of our paper to be familiar with prior path diagnosis work in [11].

This paper looks beyond path diagnosis towards identification of performance limiting path characteristics. A narrow link [3] is a link that limits the maximum capacity available from a path with TCP or any other transport protocol. This paper investigates an efficient method to identify the location and capacity of the narrow link without probing individual hops.

Current techniques and tools for measuring these limiting properties of a path are forward path bound and rely on IP protocol features that were not designed with measurement or path diagnosis in mind. Therefore, we wish to build operator support for a measurement protocol that forms the basis of this work, capable of measuring these properties and diagnosing paths.

We provide a limited review of the literature on path diagnosis and techniques for inferring path characteristics in Section 2. In Section 3 we provide a short description of a simple protocol developed for IP path measurement known as the IP Measurement Protocol (IPMP) [10] and describe operator motivations for supporting the protocol. In Section 4 we provide an overview of how use of the protocol might enhance user-level path diagnosis if the protocol were implemented and deployed throughout the Internet. Finally, we conclude in Section 5.

2. PREVIOUS WORK

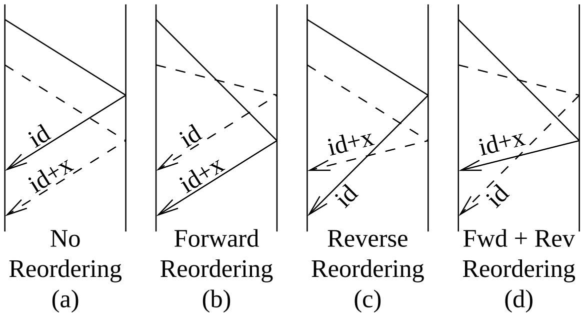
In this section, the packet probing techniques that form the basis of current path diagnosis and the basis of ours are briefly reviewed. The techniques used by the `tulip` tool in [11] infer loss, reordering, and queueing using existing IPv4 protocol features. `cing` focuses on measuring network-internal delays [1] using the same ICMP protocol features used by the `tulip` tool. The techniques used by the `pathrate` tool [3] provide the building blocks of a capacity measurement tool, and the techniques used by the `pathchar` tool [4] provide the building blocks of a tool to measure the capacity of each hop on a path.

2.1 Packet Reordering

`tulip` infers reordering by soliciting ICMP messages from an end host that provide sequential IP-ID values, as shown in Figure 1. Each time a router creates an ICMP response for an error condition, it should assign a sequential IP-ID value held centrally by the router into the response packet. The reordering metric is reported at the path level; i.e., reported as occurring on the forward path and/or reverse path.

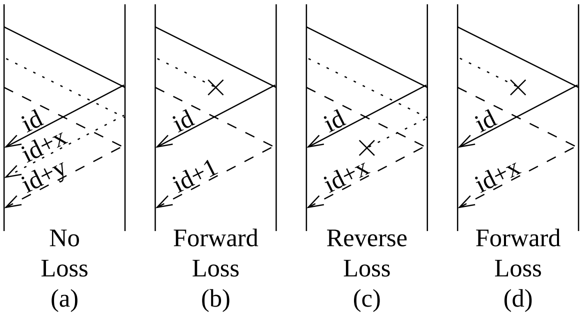
[*] The NLANR Measurement and Network Analysis Group (NLANR/MNA) is supported by the National Science Foundation (NSF) under cooperative agreement no. ANI-0129677.

SIGCOMM'04 Workshops, Aug. 30+Sept. 3, 2004, Portland, Oregon, USA.



No Reordering (a) Forward Reordering (b) Reverse Reordering (c) Fwd + Rev Reordering (d)

Figure 1: Detecting reordering with `tulip` using the IP-ID field.



No Loss (a) Forward Loss (b) Reverse Loss (c) Forward Loss (d)

Figure 2: Detecting loss with `tulip` using the IP-ID field.

If the responses arrive back in order from the destination with sequential IP-ID values, they are not reordered unless they were reordered an even number of times on the forward or reverse paths. If they arrive back out of order but with sequential IP-ID values, the reordering occurred on the forward path. If the responses arrive back out of order without sequential IP-ID values, the reordering occurred on the reverse path. Finally, if the responses arrive back in order but without sequential IP-ID values, there was reordering on both the forward and reverse paths.

2.2 Packet Loss

`tulip` measures loss at the path level by soliciting ICMP messages from the end host with incremental IP-ID values, as shown in Figure 2. The probing host sends three probe packets with the intent of inducing loss on the middle packet. The two outer packets are small *control* packets while the inner packet is a large *data* packet. The large packet is more likely to be dropped than the two small packets. If the middle packet is lost on the forward path and the two control packets arrive at the destination back-to-back, they will be assigned incremental IP-ID values, and `tulip` can infer forward loss as shown in Figure 2b.

`tulip` cannot definitively infer reverse path loss with this technique. The IP-ID values in the two control packets returned as a result of the data packet being lost on the reverse path (Figure 2c) are the same as if the data packet were lost on the forward path and an ICMP response were generated for another packet between the two control packets arriving at the destination (Figure 2d).

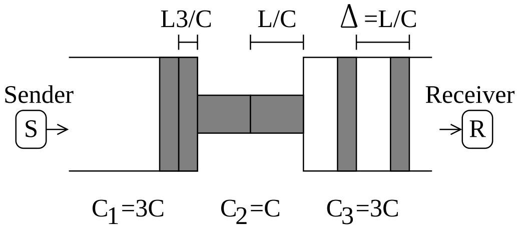


Figure 3: The packet pair technique as used by `pathrate`. Provided the pair do not encounter cross traffic or a tighter link, the packets will remain separated by the same distance until they arrive at the receiver.

2.3 Packet Queueing

`tulip` sends ICMP timestamp request probes to intermediate hops to measure one-way packet queueing delays to each hop. `tulip` assumes that if enough probes are sent to each hop, at least one probe will experience the minimum queueing delay to allow it to estimate queueing delays for the other probes with larger one-way delays.

ICMP timestamps have a maximum resolution of 1ms. Often, the timestamps returned in ICMP timestamp response packets are generated by free running clocks, or the host's clock is free running, so `tulip` assumes the clocks are not calibrated, and uses `fixclock` [16] to calibrate the clocks.

ICMP timestamp response packets can be rate limited like other ICMP response packets. Experience with `tulip` in [11] has shown that some routers are slow to generate ICMP timestamp responses if the CPU is busy with other tasks.

2.4 Capacity Estimation

`pathrate` uses the packet pair technique – where two probes are sent back-to-back – to measure the capacity of the forward path between a sender and a receiver. For the packet pair to measure the capacity of the narrow link, it must enter the capacity limiting link back-to-back and remain separated by the same amount through any remaining links to the receiver, as shown in Figure 3. Cross traffic means that the separation of the packets may change as they pass through the network. To remove the effect of cross traffic, it is not sufficient to conduct minimum sampling on the separation of the probes as they arrive at the receiver. If the pair compresses due to queueing behind cross traffic after the narrow link, minimum sampling will overestimate the capacity of the narrow link. `capprobe` [8] uses the same packet pair technique, but uses the minimum propagation delay through the network to indicate that the pair did not incur queueing delays anywhere in the path, including the narrow link.

These tools do not report *where* in the path the capacity limiting link occurs. `pathchar` [4] is the original variable packet size technique, where the capacity of each hop is estimated by measuring the extra delay incurred to each point in the network by a larger packet compared to the delay incurred by sending a smaller packet to the same point. More sophisticated techniques for estimating the capacity of each hop in the network exist, such as packet-tailgating [9] and packet-quartets [14]. In order to identify the capacity

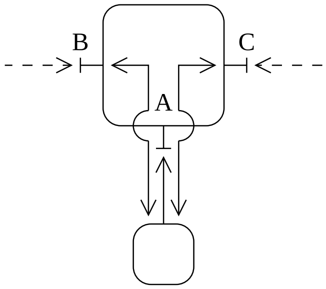


Figure 4: Matching Interfaces using UDP probes. Three interfaces (A, B, C) on the router are probed from a single location. The ICMP response for each probe is sent with a source address set to the egress interface. In this case, the egress interface used is A.

limiting link, a TTL-limited search of the forward path is required with these per-hop capacity estimation methods.

2.5 Interface Matching

ICMP TTL Expired responses may originate from different interfaces from the same router depending on the particular interface a TTL-limited packet arrives at. If `traceroute` is used to infer the forward path towards a common destination from several sources, a diagnosis tool may not have obtained the actual topology until it has matched interface addresses together to form router nodes. There are at least two probing techniques for matching these addresses.

The original technique used in the Mercator [5] and Skitter [6] projects is to send UDP probes addressed to high numbered ports to each IP discovered with the `traceroute` phase. When a router returns an ICMP error in response to these UDP probes, it may do so with the source address set to the egress interface leading back to the probe's source. Any other probes sent to the same router may be returned using the same source address and may be matched, as shown in Figure 4.

The `ally` tool described in [18] uses the IP-ID field to infer that two interfaces belong to the same router. `ally` sends a probe to each of two candidate interfaces. If the responses x and y to the probes have incremental IP-ID fields, then a third probe z is sent to the interface that returned the packet with the lowest IP-ID field. If $z > y > x$ then `ally` infers that these interfaces belong to the same router.

2.6 Problem Statement

The techniques described so far and other similar techniques have a series of weaknesses. They are forward path bound and can provide only limited diagnostic utility to the reverse path without specific end-host support. If the diagnosis is initiated by an end user, frequently he or she is most interested in diagnosing the reverse path to gain insight into why downward transfers do not meet his or her expectations.

The techniques often involve sending many packets into the network to infer a particular path characteristic, as they either have to sectionalise the path into hops (`traceroute`, `tulip`, `pathchar`) or send enough probes to gain confidence that a probe incurred minimum delay (`pathrate`, `capprobe`). If the route changes while the tool is collecting data, the tool must be able to detect this; many tools assume the route does not change during measurement due to difficulties in detecting route changes. In addition, it can be difficult to estimate the delay incurred by a probe after a point in the network when the path leading up to that point has a significant amount of jitter.

The techniques often require the probes to generate an ICMP response from an intermediate hop or the end station. Creating an ICMP message is more difficult than forwarding a packet, so responses may be rate-limited as recommended by RFC 1812 [2].

Some techniques require fields and packet types found in IPv4, but not in IPv6. For example, ICMP6 does not provide a timestamp request message, and the IPv6 header does not have an ID field.

3. IPMP

3.1 Motivation

The Internet needs a protocol designed for measurement and path diagnosis purposes. From an operator's point of view, the political motivation for enabling measurement of their network may involve network monitoring and analysis, validation of service level agreements, or a desire to allow a third party to accurately measure the performance of their network, rather than be inaccurately measured as may be the case currently. The technical motivation for an operator to support a measurement protocol might be to improve the accuracy of the measurements conducted or to reduce the impact any measurement activity has on their network.

3.2 Details

IPMP combines both path and delay measurements into a single packet exchange. A specially marked echo packet is sent to a destination IP address. An IPMP packet is easily recognised by intermediate nodes as an IPMP packet because it is encapsulated directly inside of an IP packet with a distinct IP Protocol Type. As an IPMP echo packet is routed towards a destination, each intermediate router inserts a *path record* that includes, among other things, the interface the packet was received on and a 48-bit timestamp that records when the interface received the packet, as shown in Figure 5. Full details can be found in the current IETF Internet Draft (`draft-mcgregor-ipmp-05.txt`).

The format of the timestamp inserted into a path record is not specified by the protocol. For low speed links, an NTP-formatted timestamp [12] without the first 16 bits might be inserted. For high speed links, the timestamp might be a free running clock operating at an eighth of the bit speed of the receiving interface. The measurement host discovers the relationship between the timestamp inserted and a 64-bit, NTP-formatted timestamp in a separate packet exchange, known as the IPMP information exchange. The timestamps do not have to be synchronised at any hop for the path diagnosis techniques described in this paper to succeed, because each timestamp is compared to other timestamps generated by the same clock.

If a single probe is sent towards a destination, it can collect a list of IP addresses between the two nodes using the data contained in the packet exchange. If a series of probes are sent towards a destination, the probes can collect data that enables the measurement of variations in queue lengths and can help determine which hops are primarily responsible for

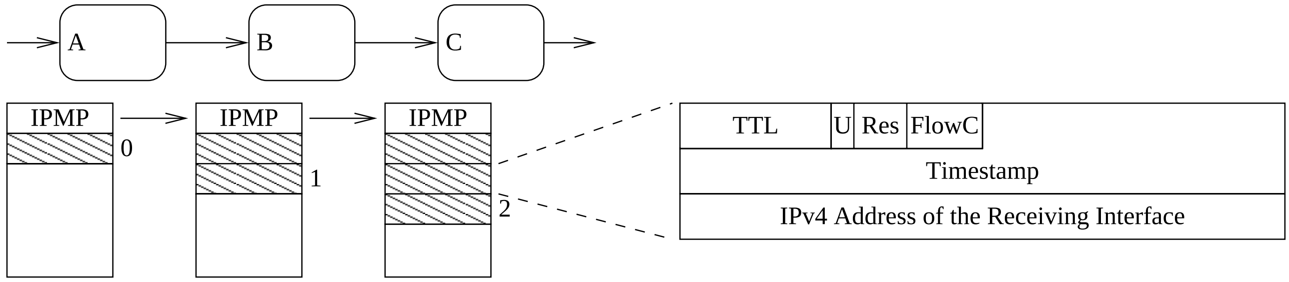


Figure 5: IPMP Echo Packet Dynamics. At interface A, path record 0 is inserted. The echo packet is a fixed size. There is space for a fourth path record to be inserted after the packet has been forwarded through interface C. The fields in the path record are: TTL – TTL copied from the IP header after decrement, U – bit to indicate this path record has been used, Res – currently reserved, FlowC – flow counter for an IPMP flow, Timestamp – when the packet was received, Address – the IP address of the interface that received this packet.

variations in available bandwidth. Simple protocol support at the end host is sufficient to probe and measure the reverse path.

The minimum packet size that must be supported by an IPv4 network, 576 bytes, allows for 45 path records to be stored in an echo packet. The IPv6 minimum MTU of 1280 bytes allows for 50 path records to be stored. These numbers increase to 122 IPv4 path records or 61 IPv6 path records when a 1500 byte packet is used. If measurement using smaller packet sizes is required, the sender can restrict path record insertion to specific hops in the network by presetting TTL values in each path-record to the required values in the echo packet.

There is a high burden to overcome in attempting to get router manufacturers to modify the fast path – the path in a router that a packet takes when forwarded from one interface to the next. The echo protocol is designed to make fast path implementations very straight forward. The echo protocol is designed to follow the same processing stream as any other packet being forwarded and can be processed as a bit stream without requiring storage of the packet. The most difficult modification to an echo packet is the IPMP checksum modification, although that can be optimised since most checksum modifications to an echo packet can be pre-computed and the checksum updated incrementally [15]. An IPMP checksum modification requires 16 bits of the packet to be buffered in the bit stream, which is also the case with a checksum modification to an IP header that occurs when an IP header's TTL value is decremented. If modifications to an IPMP packet are made in parallel with modifications to the IP header, the possibility of denial of service attacks is eliminated and the quality of measurements is improved.

4. PATH DIAGNOSIS WITH IPMP

This section describes how to use the measurement support in IPMP to diagnose loss, reordering, and queueing, and compare the approach with IPMP to the approach with `tulip`. This section also describes how to use the measurement support in IPMP to identify capacity limits.

For each performance fault, three cases of IPMP deployment are considered. The first case considered is the optimal case where all stations have IPMP support on the forward and reverse paths. In this case, each IPMP echo packet P on a path with H hops total has a path record inserted at each hop in the network. If the sender inserts the first path record into the echo packet when transmitting the packet, then $H + 1$ path records are inserted in total. The path records can be thought of as being held in an array, so that the timestamp inserted into the packet P at hop h is $P_{[h]}$. The second case considered is the likely case where some stations have IPMP support at strategic locations such as at AS boundaries. The last case considered is the minimum case where only the end hosts have IPMP enabled. For these last two cases, an IPMP echo packet will capture a subset of the packet's path through the network. Each IPMP echo packet P has N path records inserted in the packet, where $N <= H$.

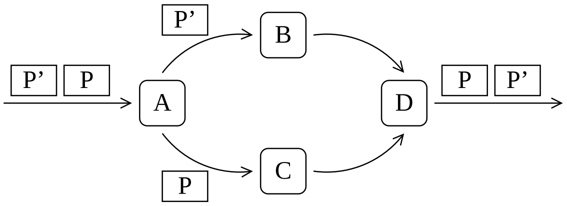


Figure 6: Detecting reordering with IPMP using timestamps. If the packets have successive timestamps in path records inserted at A but not at D, the packets were re-ordered.

4.1 Packet Reordering

If two packets P and P' are sent from a measurement host towards a target host, diagnostic tools can identify reordering by using the timestamps inserted by the stations that implement IPMP. If the the timestamps inserted into P and P' at hop h have timestamps inserted in-order, but at hop $h + x$ they are inserted in P' first and then in P, reordering between those two hops can be reported.

Using the TTL values in the relevant path records, diagnostic tools can report on the length of the path upon which the reordering occurred. Using the addresses inserted in the relevant path records, these tools can report the path segment or segments where reordering occurred. If reordering occurs due to the probe packets taking different paths, it might be possible to identify the two paths by the IP addresses that make up those paths.

In the case where IPMP is supported by all stations between the source and the destination, diagnostic tools can identify the stations between which reordering occurred and can identify reordering if it occurs between more than one segment on a path. In the case where IPMP is partially deployed at strategic locations through the path, these tools can narrow the search for the reordering segment to the applicable AS. In the case where IPMP is deployed only at the measurement host and the end host, these tools can identify forward path versus reverse path reordering as `tulip` does.

4.2 Packet Loss

In [11], it is suggested that IPMP could be used to identify where loss occurs if each IPMP flow has an associated packet counter at each router's interface. Rather than use a central IP-ID counter to diagnose loss at the path level, each IPMP flow has an optional flow counter. Each IPMP packet with the same source address, destination address, and IPMP-ID value has a flow counter that is incremented each time a router receives a packet with these values. The flow counter is copied into a path record before the router increments it.

Diagnostic tools do not need to send a triplet of packets formed in the `tulip` style (Section 2.2) where the outer two packets are small control packets used to infer the direction that loss occurred. Rather, the sender records the flow counters seen in the last packet that was not lost, and infers the position(s) in the network the lost packet(s) were lost upon receiving the next echo response.

Hop	Packet No.			
	0	1	2	3
0	0	1	2	3
1	0	1	2	3
2	0	0	1	2
3	0	0	0	1
4	0	0	0	1

Table 1: Detecting loss with IPMP using flow counters. The body of the table represents the flow counter at each hop. Four packets are sent towards a destination. Packets 0 and 3 are successfully returned from the end host, while Packets 1 and 2 are lost between hops 1+2 and 2+3 respectively.

Table 1 provides an example on detecting the hops responsible for packet loss. In this case, four packets are sent into the network but only two are received from the end host, packets 0 and 3, referred to as P0 and P3. Based on the IPMP flow counters inserted into the two packets returned – P0 = {0, 0, 0, 0, 0} and P3 = {3, 3, 2, 1, 1} – it can be determined that loss occurred once between hop 1 and hop 2, and once between hop 2 and hop 3. The order in which the two packets were lost cannot be determined from this information because the flow counters stored in P3 would be the same if P1 were lost between hops 2 and 3, and P2 were lost between hops 1 and 2.

In the case where IPMP is supported by all stations between the source and the destination, diagnostic tools can identify the link where the loss occurred. In the case where IPMP is partially deployed at strategic locations through the path, these tools can identify the AS or ASes responsible for causing the packet loss. In the case where IPMP is deployed only at the measurement host and the end host, these tools can determine with certainty whether the loss occurred on the forward path or the reverse path.

4.3 Packet Queueing

Our method to estimate packet queueing differs slightly from the one used in `tulip` by providing the possibility of estimating queueing behaviour on a per-hop basis. Instead of estimating the queueing on the forward path to a specific hop by calibrating and comparing clocks at the sender and the receiver, diagnostic tools calibrate and compare clocks between hops that have inserted a path record. Using the timestamps inserted into a probe packet confines the effect of jitter to the link or links that connect the two stations. In the case where more than one station contributes substantially to path jitter, it is possible to diagnose each of the jitter locations with IPMP. A diagnostic tool that uses IPMP to measure packet queueing does not have to trade off run-time with network load, as an IPMP echo packet can potentially collect timestamps from all stations on a path with a single packet, rather than collecting data for a single hop with each probe.

In the event where two or more clocks are established to be calibrated, IPMP enables absolute one-way delay measurements to be taken between hops. Suitable calibration sources might be a high quality real-time source such as a GPS or CDMA time receiver, or a recovered line clock from a SONET link provided the master clock is sufficiently accurate.

In the case where all stations between the source and the destination support IPMP, diagnostic tools can measure packet queueing between all routers and can report the links that are responsible for jitter due to relatively large queueing times. In the case where IPMP is partially deployed at strategic locations through the path, these tools can identify the AS or ASes responsible for the jitter. In the case where IPMP is deployed only at the measurement host and the end host, these tools can only determine the forward path and reverse path jitter.

4.4 Capacity Estimation

Given a packet pair that consists of P and P' of size S, if enough packet pairs are sent through the network, at some point each link will have at least one packet pair traverse it back-to-back. Preliminary capacity estimates c of each hop h in the forward path hops can be calculated as follows:

$$c = \min_{h=0 \ldots H} \left\{ \frac{S}{P_{[h]} - P'_{[h]}} \right\} \tag{1}$$

The capacity C of the path is the minimum c calculated for each of the hops. The capacity estimates c for links other than the capacity-limiting link are inaccurate.

The main advantage of this method is that the behaviour of each hop in the face of cross traffic is limited to just that hop, minimising the effect of that hop on the measurement of subsequent hops. Using IP addresses inserted into path records by stations connecting a hop, the location of the capacity-limiting link in the path can be identified.

To measure the capacity of the reverse path, path records inserted on the reverse path can be used to estimate the capacity of the reverse path, assuming the packet pair arrives back-to-back at the capacity-limiting link, using Equation 1. It may be possible to use a pace-setter packet as in [9, 14] to coerce the packet-pair into arriving back-to-back at the destination to diagnose the reverse path.

4.5 Interface Matching

An IPMP echo packet collects the IP address of each receiving interface as it passes through the network, so diagnostic tools still need a mechanism to allow interfaces to be matched to router units, as in Section 2.5. IPMP provides a separate packet exchange known as the Information Exchange. This packet exchange is not time critical since it is not used for measuring delay. An information response packet is required to be sent from a constant IP address regardless of the interface that was queried. If an information request packet is sent to each interface IP address included in a path record in an echo packet, a tool can match interfaces to routers by matching the address the information request was sent to and the address used to source the information response.

5. CONCLUSION

This paper has described how to use a simple measurement protocol (IPMP) to diagnose path faults. Since the protocol combines path and delay measurements, fewer packets are sent than with TTL or hop-limited probes to measure per-hop behaviours. The protocol allows each link to be measured independently of the behaviour seen by the probe on prior links. If the protocol were standardised and deployed, operators would be able to diagnose path faults in greater detail than at present, using fewer probes.

Two software implementations are publicly available for the current IPMP specification. If an operator wishes to experiment with a software implementation of IPMP for either a BSD system or a Linux 2.4 system, please visit `http://mna.nlanr.net/AMP/IPMP/`. A VHDL implementation is also available.

Acknowledgements

The authors are thankful to the anonymous reviewers for their helpful comments, and to Maureen C. Curran and Joe Groff for editorial assistance. Joanne Ball, Hans-Werner Braun, Matt Brown, Perry Lorier, Jörg Micheel, and James Spooner provided useful reviews in the process of writing this paper.

6. REFERENCES

[1] K. Anagnostakis, M. Greenwald, and R. Ryger. cing: Measuring network-internal delays using only existing infrastructure. In *Proceedings of IEEE Infocom*, San Francisco, CA, Apr. 2003.

[2] F. Baker (Editor). Requirements for IP Version 4 routers. RFC 1812, Cisco, June 1995.

[3] C. Dovrolis, P. Ramanathan, and D. Moore. What do packet dispersion techniques measure? In *Proceedings of IEEE Infocom*, Anchorage, Alaska, Apr. 2001.

[4] A. Downey. Using pathchar to estimate Internet link characteristics. In *Proceedings of SIGCOMM '99*, Cambridge, MA, Aug. 1999.

[5] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *Proceedings of IEEE INFOCOM*, pages 1371–1380, Tel-Aviv, Israel, Mar. 2000.

[6] B. Huffaker, D. Plummer, D. Moore, and k. claffy. Topology discovery by active probing. In *Symposium on Applications and the Internet (SAINT)*, pages 90–96, Nara, Japan, Jan. 2002.

[7] V. Jacobson. Compressing TCP/IP headers for low-speed serial links. RFC 1144, LBL, 1990.

[8] R. Kapoor, L.-J. Chen, A. Nandan, M. Gerla, and M. Sanadidi. CapProbe: a simple and accurate capacity estimation technique for wired and wireless environments. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, pages 390–391, New York, NY, June 2004.

[9] K. Lai and M. Baker. Measuring link bandwidths using a deterministic model of packet delay. In *Proceedings of SIGCOMM '00*, Stockholm, Sweden, Aug. 2000.

[10] M. Luckie, A. McGregor, and H.-W. Braun. Towards improving packet probing techniques. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pages 145–151, San Francisco, CA, Nov. 2001.

[11] R. Mahajan, N. Spring, D. Wetherall, and A. T. User-level Internet path diagnosis. In *Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP)*, pages 106–119, Bolton Landing, NY, Oct. 2003.

[12] D. Mills. Network Time Protocol (version 3): Specification, implementation and analysis. RFC 1305, University of Delaware, 1992.

[13] J. Padhye, V. Firoiu, D. Towlsey, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of SIGCOMM '98*, Vancouver, Canada, Aug. 1998.

[14] A. Pasztor and D. Veitch. Active probing using packet quartets. In *Proceedings of the ACM/SIGCOMM Internet Measurement Conference*, pages 293–305, Marseille, France, Nov. 2002.

[15] A. Rijsinghani (Editor). Computation of the Internet Checksum via incremental update. RFC 1624, Digital Equipment Corporation, 1994.

[16] R. Ryger. `fixclock`: removing clock artifacts from communication timestamps. DCS/TR 1243, Yale University, Mar. 2003.

[17] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of Internet path selection. In *Proceedings of ACM/SIGCOMM '99*, pages 289–299, Cambridge, MA, Aug. 1999.

[18] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *Proceedings of ACM/SIGCOMM '02*, pages 133–145, Pittsburgh, PA, 2002.