Estimating Router ICMP Generation Delays

Ramesh Govindan and *Vern Paxson*

International Computer Science Institute
Berkeley, CA USA
{ramesh, vern}@icsi.berkeley.edu

Abstract—**A number of Internet measurement techniques rely on eliciting ICMP replies from routers inside the network [7], [9]. One question regarding the accuracy of these techniques concerns whether routers introduce delays when generating the ICMPs. We present a technique for estimating these delays. The approach is based on sending packets with *spoofed* source addresses, some with sufficient TTL to travel all the way to the destination ("direct"), others with a limited TTL that will cause them to be transformed at a given hop into an ICMP Time Exceeded (TE) packet. By spoofing the source address of the probes to match the destination, these TEs are routed identically to how the direct probes are routed. The technique thus allows us to factor out effects due to asymmetric routing or conditions on the return path from a given router back to the source.**

We have implemented the technique in a tool, *fsd* (fast-path/slow-path discriminator). In this paper, we discuss an analysis of a set of measurements made using *fsd* on the NIMI measurement infrastructure [13]. Our analyses provide helpful insight into the question of "what proportion of routers in today's Internet are slow at generating TE replies?" Our results indicate that the answer to this question is "few" which implies that tools such as *pathchar* and *treno* will not in practice suffer greatly from slow-path/fast-path differences, as has been a concern.

I. Introduction

As a consequence of layering in the Internet protocol suite, ICMP responses provide the only available general mechanism for attaining visibility into the Internet's internal packet dynamics. While routers themselves can provide performance measurements directly, access to them is administratively controlled and split among multiple, non-cooperating providers; and the information that routers do propagate more publicly (*i.e.,* routing updates) is exceedingly coarse-grained.

Accordingly, a number of Internet measurement techniques rely on eliciting ICMP replies from routers inside the network. For example, the widely used *traceroute* utility sends packets with limited TTL hopcounts in their IP headers in order to trigger ICMP Time Exceeded replies from the routers along the path to the destination.

However, inferring anything useful about network performance, as opposed to simply topology, requires analyzing the *timing* associated with packets as they traverse the network. For example, *pathchar* [7] (and the related *clink* [5] and *pchar* [8]) and *treno* [9] both send large numbers of TTL-limited packets in order to analyze the delays between the transmissions of the packets and the receipt of the corresponding ICMP replies. (This is also sometimes done with much less precision for *traceroute* timings in order to locate the source of unexpected delays within the network.)

There are two significant difficulties with analyzing such timing, however. The first is that the ICMP reply might not be generated in a timely fashion. Router packet processing is assumed to often consist of a *fast path* and a *slow path.* Fast path processing is used for common cases such as forwarding in the absence of options, and is usually accomplished in hardware. Slow path represents infrequent processing, usually accomplished in software at a central processor: examples include IP option processing and ICMP generation. If the measured ICMP replies are generated using the slow path, then slow-path processing can inflate the apparent delays associated with receiving the replies, complicating the performance analysis. Furthermore, if the slow-path delays vary from router to router, then comparing delays computed from different routers is likewise susceptible to inaccuracies that have nothing to do with the network dynamics we wish to measure, but with differences in the router internals.

A second difficulty with analyzing timing based on ICMP replies arises when we want to compare timings measured at different routers. Suppose we are sending packets from A_0 and receiving replies from routers A_i and A_{i+1}. Due to the prevalence of asymmetric routing in the Internet [10], the route taken back to A_0 by the replies sent by A_i might well differ significantly from the route taken by A_{i+1}'s replies. For example, the return path from A_i might well be *longer* than that from A_{i+1}, or might be considerably shorter than the forward-path latency from A_i to A_{i+1}. More generally, a common method of estimating the latency of the link between A_i and A_{i+1} is as half of the difference of the measured RTT to A_{i+1} and to A_i. If, however, the return path from A_{i+1} to the sender differs in more than just the final hop compared to that from A_i to the sender, then this method can yield significantly erroneous results.

In this paper, we present a technique for measuring a router's ICMP Time Exceeded (TE) generation time using only end-system measurements. The technique can be used to address the first difficulty above, *i.e.,* taking into account varying slow-path delays when timing TE replies. The technique does not suffice to address the second difficulty of asymmetric return paths taken by TE replies from different routers, but at least the technique is not itself susceptible to errors due to such asymmetries.

We have implemented the technique in a tool, *fsd* (fast-path/slow-path discriminator), and below discuss an analysis of a preliminary set of measurements made using *fsd* on the NIMI measurement infrastructure [13]. In principle, *fsd* provides a way to help factor out the "noise" measurements incur if they rely on timing TE replies. While in its current form *fsd* is not practical to incorporate directly into tools like *pathchar* and *treno* (since *fsd* has to run on end systems on both sides of the router to be measured), it can still provide helpful insight into the question of "what proportion of routers in today's Internet are slow at generating TE replies, anyway?" If the answer to this question is "few," which is what our preliminary results indicate, then we have a useful positive result that tools such as *pathchar* and *treno* will not in practice suffer greatly from slow-path/fast-path differences, as has been a concern.

II. Experimental Methodology

In this section, we describe our experimental methodology for computing router overheads when generating TE messages. One obvious approach would have been to individually test a variety of routers using a black-box approach: compute the difference in the time taken by a router to process an IP packet and to generate a TE message in response to that packet, and the time taken by a router to forward a normal IP packet on the fast path. This approach presents logistical challenges in assembling routers from different vendors, and evaluating ICMP generation times on different software releases. In addition, this approach does not reveal the distribution of router overheads in the deployed infrastructure.

Instead, we aim to develop a technique that can be used to estimate TE generation times using active measurements made solely at the network's edges. In addition, we have begun to make "mesh" measurements between a number of measurement points in order to gain insight into the prevalence of different degrees of TE generation overhead within today's Internet, in a style similar to Paxson's NPD measurements [10].

To measure TE generation times, we use the following technique. Suppose we send a packet from host A to host B with a TTL limit such that it will expire at router R inside the network. R will then construct a TE message and send it back to A. However, if we *spoof* a source address of B in the packet sent to B (thus, the packet will have both a source and a destination address of B), then R will in fact send the TE message *forward to* B rather than in the reverse toward A. Thus, the effect of the TTL expiration is to *transform* the packet from an ordinary IP datagram into a ICMP TE message, but otherwise to not perturb the packet's routing through the network. This trick of exploiting spoofing to keep the TE message from returning to A addresses the second difficulty mentioned in Section I, that TE replies can take asymmetric return paths back to the original sender.

Now consider that host A sends a stream of packets to host B, some with a limited TTL such that they expire at R, others with sufficient TTL to reach B. Averaged over enough instances, we should find that the difference between the measured transit times for the packets that expired at R and those that made it all the way to B untransformed will correspond to the only difference between the two sets of packets, namely the additional overhead incurred at R with transforming the original packet into a TE message.

Thus, *providing we can spoof source addresses*, we can measure the TE generation overhead for any particular router in the path from A to B by selecting an appropriate TTL to target that router.

A. *fsd* : The Fast-Path/Slow-Path Discriminator

To implement the above measurement technique, we developed a tool that we call *fsd*. Logically, *fsd* contains functionality for acting as both a *sender* of probes as well as a *receiver*. The operation of *fsd* is best described by considering a single sender-receiver pair. *fsd* computes one-way delays, with the receiver serving as a passive entity. The sender, given the receiver's IP address and the hop count of the path to the receiver, generates two kinds of probes: *direct* probes (sufficient TTL to reach the receiver) and *hop-limited* probes (varying TTLs, insufficient to reach the receiver, and instead selecting different routers along the path). Each probe is uniquely identified by a sequentially increasing *probe identifier*; the probe identifier helps us correlate sent and received probes.

As discussed above, direct probes are intended to sample fast-path forwarding at all routers along the path to the receiver. Direct probes are specially crafted *ICMP Echo Reply* packets.[1] Their IP identifier field encodes the probe identifier. Both the sender and the receiver log the addresses, the identifiers, and the times of sending (or receiving) direct probes.

Hop-limited probes are intended to tickle ICMP Time-Exceeded processing at a specified hop on the path between the sender and the receiver. The hop-limited probe is also a specially crafted ICMP Echo Reply packet, and again its IP identifier field encodes the probe identifier. The *gateway* field of the ICMP header encodes the sender's IP addresses (recall that we can't use the IP header's source address, since that is spoofed to instead be the receiver's address). Again, both the sender and the receiver log the addresses, the identifiers, and the times of sending (or receiving) hop-limited probes or their TE equivalents.

fsd is careful to pad both direct and hop-limited probes so that their size exactly matches the size of the ICMP TE response, thereby avoiding any systematic propagation time differences that might arise due to packet size differences.

The sender periodically (with some jitter to avoid synchronization) sends probes to the receiver. It uses the following decision procedure to generate a probe at each step. It first determines whether to send a direct probe or a hop-limited probe; direct probes are chosen with a probability p (0.25 in our experiments). Thus, regardless of the length of the path between sender and receiver, on average 1 in 4 probes are direct probes. If the sender chooses to send a hop-limited probe, it picks the designated hop uniformly. Thus, if there are N hops between sender and receiver, it picks a given hop with probability $1/N$. This procedure allows us to randomly sample the fast path as well as the TE generation time. The direct path is sampled more frequently than each hop, giving *fsd* a better baseline against which to compare the TE generation overhead. Since direct probes do not cause any particular stress on routers, sending them at a higher rate should be acceptable.

There exists an alternative technique for measuring TE generation times.[2] At each step, the sender sends a hop-limited probe followed by a direct probe, *back-to-back*. As before, it picks the designated hop uniformly. With this procedure, every TE generation sample is associated with a sample of the fast path, when both probes are successfully received. In that event, we can compute TE generation overhead by taking the difference between the two samples. *fsd* has a mode for this measurement technique which we use to analyze the efficacy of this approach in Section III.

A single instance of *fsd* is designed to concurrently act as a receiver, as well as to send probes to multiple receivers. It takes as input a list of

[1]The choice of an ICMP Echo Reply packet was dictated by convenience. We could have chosen to use UDP datagrams, but that would have necessitated different processing paths for receiving direct probes and hop-limited probes. We could not have used an ICMP Echo packet because most stacks do not pass ICMP Echoes up to user-level software.

[2]Suggested by Stefan Savage.

pairs containing IP addresses and corresponding hopcounts, as well as an experiment duration. Each pair represents a receiver to which the sender sends probes for the specified duration. Thus, the following steps constitute a typical *fsd* experiment: select a set of probe machines; invoke *fsd* on each machine with all the other machines as receivers for a specified duration; collect all sender and receiver logs centrally; and correlate sent and received probes.

Some practical issues arise in the context of designing *fsd* and conducting wide-area measurements. First, many routers are believed to effect ICMP rate-limiting,[3] *i.e.*, these routers do not respond to ICMP Time-Exceeded messages at a rate greater than 1 per second. Accordingly, by default *fsd*'s inter-probe interval is set conservatively to 1 second.

Second, since *fsd*'s hop-limited probes spoof the source address of the receiver, *fsd* is less effective in the presence of ingress [6] and/or egress filtering. Specifically, if the first hop router attached to the sender implements egress filtering, all hop-limited probes are dropped at this router. If the sender is not egress filtered but the receiver is ingress filtered, hop-limited probes are not dropped except at those hops between the ingress filtering router and the receiver (this assumes that *fsd* does not spoof the source address on direct probes; we discuss this below). That is because the *source address on the ICMP TE messages is a valid address*; it is the address of the router that generates the TE message, rather than the spoofed address of the destination. Conveniently, some ingress/egress filtering routers generate ICMP Administratively Prohibited unreachable messages when dropping such spoofed packets. We find that these too are routed to the receiver, since the unreachable is sent to the spoofed source address! So we can actually use these to measure (Section III) the Administratively Prohibited generation time of that particular router; but we can not measure any routers beyond it.

Third, load balancing on multiple physical links can distort our results in a subtle way. Some routers in the Internet are connected to their next hops using multiple physical links. Usually, such routers employ load balancing; they attempt to distribute traffic to that next hop evenly across the multiple links. At least one vendor [2] implements load-balancing such that packets for a given source-destination pair are always consistently forwarded to the next hop on the same physical link, but packets from a different source to the same destination may be forwarded over a different link. Now, because the direct probes and the hop-limited probes have different source addresses, they may actually take *different physical paths* between sender and receiver!

We can avoid this by spoofing the source address on direct probes as well. *fsd* has a mode to do so, which we use to evaluate whether load balancing affects our conclusions in Section III. One drawback of spoofing source addresses on direct probes is that if either the sender or the receiver is ingress/egress filtered, we are unable to obtain any TE overhead samples along the path.

If load-balancing occurs on the path between the designated hop h and the receiver, then direct probes may travel on a different physical path than the TE response because the source address on the latter is the IP address of the router at hop h. This is harder to avoid[4] and *fsd* does not attempt to do so.

Finally, *fsd* is unaware of path changes during the course of an experiment; a future version of *fsd* will sample the path to the receiver and adjust the generation of probes to reflect the new path.

B. Estimators for TE Generation Times

Above we discussed an estimator for the TE generation time at a particular hop between two hosts A and B: the difference between the average transit time for hop-limited probes processed at the hop, and the average transit time for direct probes from A and B. While this estimator is appealing in its simplicity, it suffers from several problems. First, network delays not infrequently exhibit large "spikes" [14] that can significantly skew arithmetic averages. Second, it is possible that ICMP generation delays include similar sorts of spikes. Third, in the presence of *clock skew*, the computed transit times (re-

[3]ICMP rate-limiting is permitted by [4] and has also been observed in practice for TE messages[11]. However, the extent of its deployment is unclear. Furthermore, at least one document [1] indicates that for one widely used implementation, ICMP rate-limits only apply to ICMP Destination Unreachables.

[4]For example, one might attempt to spoof the source address on direct probes using the address of hop h. This strategy, however, fails if load balancing occurs on the path before h.

ceiver timestamp minus sender timestamp) can vary considerably over the course of a dataset, enough so to completely (and artificially) dominate the delay comparisons.

In light of these problems, we experimented with a number of potentially more robust estimators. First, we assessed and removed the linear relative clock skew present in the timings using robust linear regression, based on the techniques presented in [12]. We then tried applying "min filtering," *i.e.,* comparing the *minimum* transit time observed for a particular hop versus that observed for direct probes. We found, however, that our data included occasional delay "dips," which dominated the minima even though they clearly appeared to be some sort of measurement artifact (perhaps a lull in capturing a packet timestamp at the sender side).

To avoid being skewed by these dips, we then tried comparing the *10th percentile* of the transit times. However, this, too, proved noisy, we believe due to imperfections in removing clock skew.

Sophisticated techniques for removing clock skew have been published in the literature [3], but we hit upon a simpler scheme that minimizes the effects of clock skew and transient queueing delays. Suppose for a hop-limited probe sent at time t_i we compute a transit time of h_i. Let t_{i-1} denote the time of the nearest direct probe sent *before* the hop-limited probe, and t_{i+1} the time of the nearest direct probe sent *after* it. Let the corresponding transit times for these be d_{i-1} and d_{i+1}. We then form a *weighted average* of the estimated transit time we would have observed had we sent a direct probe at time t rather than a hop-limited probe:

$$\hat{d}_i = d_{i-1} + \frac{d_{i+1} - d_{i-1}}{t_{i+1} - t_{i-1}}(t_i - t_{i-1}).$$

That is, $\hat{d}_i$ corresponds to linear interpolation of the trend from the direct probe at t_{i-1} to the direct probe at t_{i+1}, stopping at t_i.

We then take as our estimate of the TE generation time for the hop-limited probe at time t_i the quantity $x_i = h_i - \hat{d}_i$. (For hop-limited probes which did not have a direct probe sent before or after them, we simply discarded the probe's measurement.) Clearly, the individual elements of x_i might exhibit a great deal of noise due to fine-grained delay fluctuations along the path being measured; but we use for our final estimate the *median* of all of the x_i, and that quantity should be quite robust, reflecting the overall excess seen for hop-limited probes versus direct probes.

The appeal of this approach is that it *(i)* removes clock skew, *(ii)* does so in a fashion that works even if the clock skew changes over the course of the trace, *(iii)* works in the presence of clock jumps such as those documented in [12], and *(iv)* smoothes delay fluctuations due to transient queueing.

In summary, our final methodology for computing TE generation times at a router R at hop h on the path from A to B was:

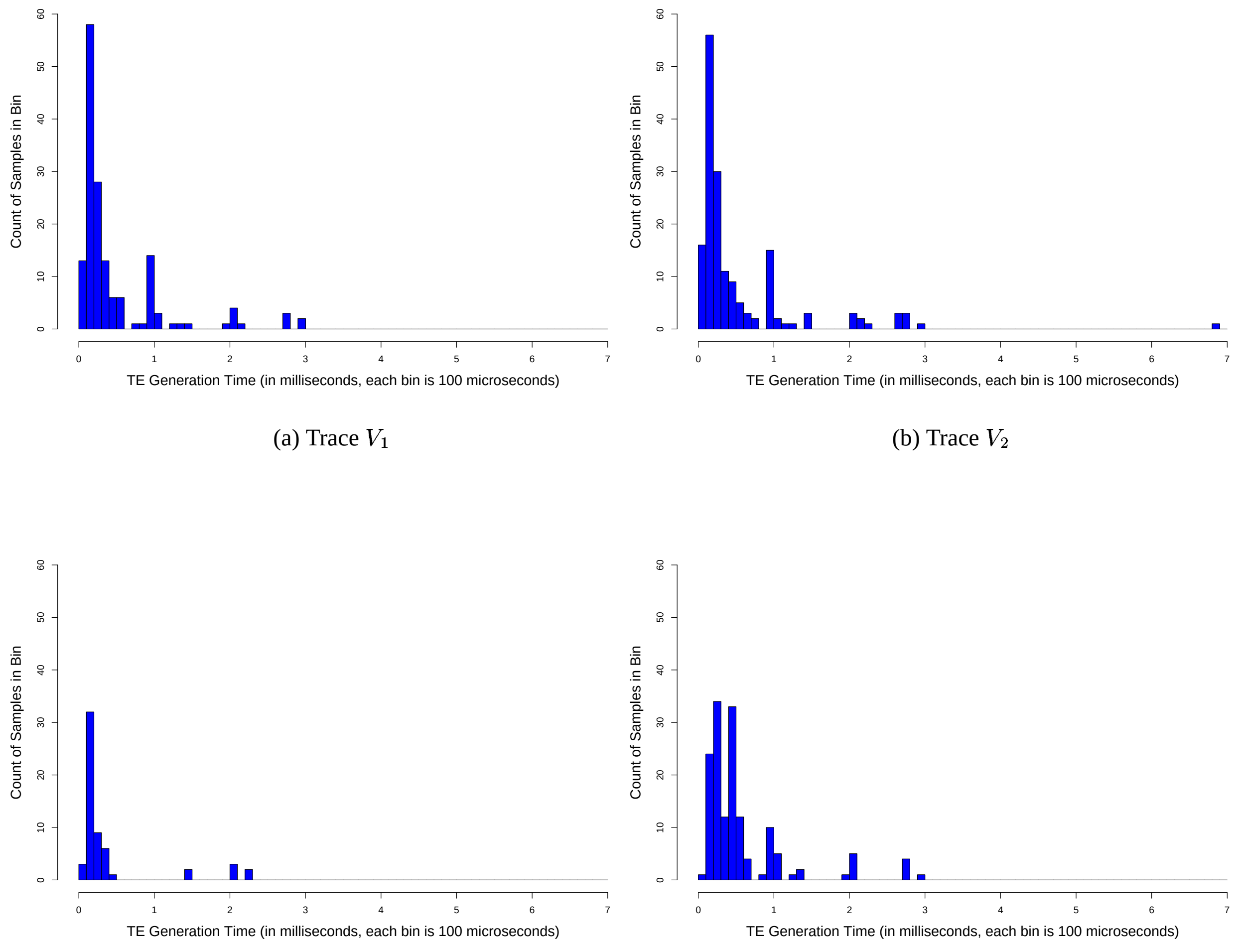
1. Extract from packet traces logged at A and B the time series of direct probes and of probes hop-limited at h.

- Discard time series from paths that had an insufficient number (150) of usable direct probes. A small number of paths had fewer than these many samples because the duration of our experiments was insufficient given the lengths of those paths.
- Discard transit times from paths tainted by a *path change*. A path change manifests itself in our traces as a change in the identity of the router at some hop h' at time T. We only use transit time samples before T.

2. For each hop-limited transit time, compute the corresponding $\hat{d}_i$ and x_i.
3. Form the estimate of the TE generation time as median(x_i).

III. Results

In this section, we present results from four different experiments, each two hours long (Figure 2 describes these experiments). Each of our experiments used the same set of ten geographically dispersed NIMI probe machines, of which six were not behind ingress/egress filtering routers. Two experiments, that we call V_1 and V_2, use the basic (or "vanilla") *fsd* design described in Section II-A: source addresses on direct probes are not spoofed, and direct probes are not sent back-to-back with hop-limited probes. A third experiment, labeled S, uses the *fsd* mode where the source address of each direct probe is spoofed. The fourth experiment, labeled B, uses the *fsd* mode where every hop-limited probe is associated with a direct probe sent immediately after it, back to back.



(a) Trace V_1 (b) Trace V_2

(c) Trace S (d) Trace B

Fig. 1. Histograms of TE generation times: V_1 and V_2 were obtained using the "vanilla" *fsd* design, S was obtained by spoofing the source address on the direct probes, and B by sending a direct probe back-to-back with every hop-limited probe.

In this section, we discuss the results obtained from the traces gathered during each of these experiments.

In our vanilla datasets, about 310,000 of the approximately 700,000 probes sent (about 44%) were received (for the B and S traces, the corresponding percentages were 60% and 26%). There exist several reasons why so few of the sent probes were received. In some cases, *fsd* was started at different times by the NIMI software on different machines; as a result, there was no functional receiving *fsd* along a path for some interval during an experiment. In other cases, no hop-limited probes were received at all for some hops along the path, apparently due to routers that simply do not generate TE messages. Where one or both ends of a path were behind ingress/egress filtering routers, hop-limited probes were only received for some, but not all, hops along the path. Finally, packet drops accounted for some (less than 5%) of the lost probes for those hops for which probes were received.

Figure 1 depicts the histogram (using 100 μsec bins) of TE generation estimates for traces V_1 and V_2, obtained by applying the estimator described in Section II-B. The first thing we see is that

Label	Description	Time
V_1	Vanilla *fsd*	January 19, 2002 at 1515 PST
V_2	Vanilla *fsd*	January 20, 2002 at 0830 PST
S	Spoofed source addresses	January 19, 2002 at 2130 PST
B	Back-to-back direct probes	January 19, 2002 at 2350 PST

Fig. 2. Table of Experiments

most routers have TE generation times less than 500 μsecs. However, we also see several modalities common to both plots: at 1 msec, around 2 msec, and between 2.5 and 3 msec. The joint presence of these secondary peaks in both datasets, and the absence of other peaks present in one dataset but not the other (which would be suggestive of possible coincidental alignment for the common peaks) gives us some confidence in the robustness of our estimator. We should point out that we were able to obtain estimates for approximately 70 distinct routers. It is certainly conceivable that more extensive experiments may reveal other modalities.

Thus, while the first conclusion is that, in general, Internet routers generate TE ICMPs with very little overhead, the second conclusion is that there do indeed appear to be some routers that exhibit markedly slower TE generation compared to their normal forwarding processing.

One puzzle regarding the secondary peaks, however, is that they are *intermittent*: measurements of the same router will sometimes *not* exhibit the peak. In addition, the peaks are particularly observed for routers on paths terminating at *one* of the probe machines. (In fact, all TE generation times greater than 2.5 msec in Figure 1 were obtained from such paths.) While some of this (in particular, the 1 msec peaks) can be explained as due to load-balancing (see below), for others we have so far been unable to identify a systematic error common to these paths to explain the peculiarity. Indeed, it was the possibility of these effects being due to clock anomalies common to the two hosts that led us to develop the more robust estimator outlined in Section II-B, but the phenomenon remained after switching to it. We are currently investigating this inconsistency further.

Figure 1(c) shows the TE generation times obtained from the S trace. These TE generation estimates were also computed using the estimator described in Section II-B. We see that the results from S are consistent with those in Figure 1, but with some important exceptions. Recall that in the S experiment, the source address of direct probes is spoofed. In S, therefore, if either end of a path was ingress/egress filtered we obtained no samples. For this reason, the peaks in Figure 1(c) are lower than those in Figure 1. Figure 1(c) is also missing TE generation samples around 1 msec, whereas both V_1 and V_2 have peaks around 1 msec. We verified that these peaks in V_1 and V_2 were due to load balancing – direct probes were using a different physical link than the hop-limited probes. In fact, all of these data points were from routers along paths that were affected by the *same* load balancing link. Predictably, these data points did not appear in the S trace. Furthermore, the S trace does not exhibit some of the modalities in the 2.5 to 3 msec range. In V_1 and V_2, these peaks were observed on some measurements taken on paths terminating at a particular probe machine. That probe machine was ingress-filtered so these peaks disappeared from the S trace.

Figure 1(d) describes the TE generation times obtained from the B trace. In the B experiment, every hop-limited probe was immediately followed by a direct probe. We measured TE generation times as follows. For each hop-limited probe sent to a particular hop, we computed the difference between the transit time for that probe, and the transit time for the associated direct probe. We then estimated the TE generation time for that hop as the *median* of all these samples. In a qualitative sense, Figure 1(d) is also consistent with Figure 1(a-b) and does not invalidate our previous conclusions. However, there are important differences. It appears that the back-to-back method in B leads to overestimates of the TE generation time. Both the V_1 and V_2 exhibit a dominant peak in the 100-200 μsec range, but in B there are two significant peaks, one in the 200-300 μsec range, and the other in the 400-500 μsec range. More generally, 75-80% of the TE generation estimates in B are higher than their “vanilla” counterparts. We also observe significant reordering of the hop-limited and direct probes sent in B. Of the pairs of hop-limited and back-to-back direct probes that were successfully received, 81% were reordered. Thus, it appears that when sending the probes back-to-back, the delay of converting the hop-limited probe into a TE allows the direct probe sent behind it to catch up and often pass it. If passed by the direct probe, the TE will then experience additional delay queueing behind it.

Finally, recall that four of our ten probe machines were behind ingress/egress filtering routers. All these routers sent ICMP Administratively Prohibited messages. For these four routers we can estimate the time to generate these ICMP messages. We found that three of these four routers had generation times less than 0.5 msec while the fourth exhibited a generation time of approximately 3.5 msec.

IV. Conclusions

This paper describes some preliminary results in estimating the time taken to generate ICMP Time-Exceeded messages at routers, using a tool we developed called *fsd*. The key technique exploited by *fsd* is the use of packets with *spoofed* source addresses in order to remove timing complications due to asymmetric return paths, as well as the robust estimation of the generation overhead, based on using linear interpolation from the direct probe timings, which also eliminates possible estimation errors due to any linear clock skew.

Understanding TE generation delays has applicability to the accuracy of several end-to-end measurement tools, such as *pathchar* and *treno*. Our preliminary results indicate that, contrary to current wisdom, the generation times are generally in the sub-millisecond range. We have been careful to validate this conclusion using a different estimator for TE generation that uses back-to-back direct probes. There are, however, several puzzles related to our measurements (such as sporadic timing spikes, apparently associated with a particular probe machine) that will require further investigation to resolve.

References

[1] *ICMP Unreachables Rate Limitation*. `http://www.cisco.com/warp/public/105/traceroute.shtml#unreach`.

[2] *Configuring Cisco Express Forwarding*. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/switch_c/xcprt2/xccefc.htm`

[3] S. Moon and P. Skelley and D. Towsley. Estimation and Removal of Clock Skew from Network Delay Measurements. In *Proc. of IEEE Infocom*, 1999.

[4] F. Baker. Requirements for IPv4 Routers. Request for Comments 1812, Internic Directory Services, June 1995.

[5] A. B. Downey. Using pathchar to Estimate Internet Link Characteristics. In *Proc. of ACM SIGCOMM*, September 1999.

[6] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which Employ Source Address Spoofing. Request for Comments 2267, Internic Directory Services, January 1998.

[7] V. Jacobson. pathchar - A Tool to Infer Characteristics of Internet Paths. available from `ftp://ftp.ee.lbl.gov/pathchar`, April 1997.

[8] Bruce A. Mah. pchar: A Tool for Measuring Internet Path Characteristics. available from `http://www.employees.org/ bmah/Software/pchar/`.

[9] M. Mathis and J. Mahdavi. Diagnosing Internet Congestion with a Transport-Layer Performance Tool. In *Proc. of INET*, Montreal, June 1996.

[10] V. Paxson. End-to-end Internet Packet Dynamics. In *Proceedings of the 1997 ACM SIGCOMM Conference on Communication Architectures and Protocols*, September 1997.

[11] V. Paxson. *Measurements and Analysis of Internet Packet Dynamics*. PhD thesis, University of California, Berkeley, 1997.

[12] V. Paxson. On Calibrating Measurements of Packet Transit Times. In *Proc. of ACM Sigmetrics*, June 1998.

[13] V. Paxson, A. Adams, and M. Mathis. Experiences with NIMI. In *Proceedings of the Passive and Active Measurements Workshop*, 2000.

[14] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the Constancy of Internet Path Properties. In *Proceedings of the ACM SIGCOMM Measurement Workshop*, San Francisco, CA, November 2001.