INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2005; **18**:935–960
Published online 12 July 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.741

Characterizing user-perceived impairment events using end-to-end measurements

Soshant Bali[1], Yasong Jin[2], Victor S. Frost[1,*,†] and Tyrone Duncan[2]

[1] *Information and Telecommunication Technology Center, Department of Electrical Engineering and Computer Science, University of Kansas, 2335 Irving Hill Road, Lawrence, KS 66045, U.S.A.*
[2] *Department of Mathematics, University of Kansas, 405 Snow Hall, 1460 Jayhawk Blvd, Lawrence, KS 66045, U.S.A.*

SUMMARY

Measures of quality of service (QoS) must correlate to end-user experience. For multimedia services, these metrics should focus on the phenomena that are observable by the end-user. Metrics such as delay and loss may have little direct meaning to the end-user because knowledge of specific coding and/or adaptive techniques is required to translate delay and loss to the user-perceived performance. Impairment events, as defined in this paper, are observable by the end-users independent of coding, adaptive playout or packet loss concealment techniques employed by their multimedia applications. Time between impairments and duration of impairments are metrics that are easily understandable by a network user. Methods to detect these impairment events using end-to-end measurements are developed here. In addition, techniques to identify Layer 2 route changes and congestion events using end-to-end measurements are also developed. These are useful in determining what caused the impairments. End-to-end measurements were conducted for about 26 days on 9 different node pairs to evaluate the developed techniques. Impairments occurred at a high rate on the two paths on which congestion events were detected. On these two paths, congestion occurred for 6–8 hours during the day on weekdays. Impairments caused by route changes were rare but lasted for several minutes. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: end-to-end Internet measurements; quality of service metrics; user-perceived impairments; congestion; route changes

1. INTRODUCTION

Effective quality of service (QoS) metrics must relate to end-user experience. For real-time multimedia (RTM) services these metrics should focus on phenomena that are observable by the end-user. In this paper methods are developed to predict network events that are observable by end-users independent of coding, adaptive playout or packet loss concealment (PLC) techniques that are often employed in RTM application. Long bursts of packet losses, high delays and a

*Correspondence to: Victor S. Frost, Information and Telecommunication Technology Center, Department of Electrical Engineering and Computer Science, University of Kansas, 2335 Irving Hill Road, Lawrence, KS 66045, U.S.A.
† E-mail: frost@ittc.ku.edu

Contract/grant sponsor: NSF; contract/grant number: ANI-0125410

Received 22 November 2004
Revised 28 February 2005
Accepted 1 April 2005

high random packet loss rate are all observable impairments. Metrics such as long term average delay, loss and jitter may have little direct meaning to end-users of rapidly changing multimedia applications because knowledge of the specific coding, adaptive playout and PLC techniques is required to translate delay and loss into the user-perceived performance. A user-perceived impairment event as defined here will impact the customer's QoS independent of the specific coding mechanism or of attempts to mask and/or adaptively compensate for its effects. The QoS metric, that is a rate of user-perceived impairment events, is easily understood by end-users and captures the network performance that is observable by network customers. Impairments arise from a variety of phenomena including long bursts of packet losses, random packet losses and high delays.

Long bursts of packet losses are known to be present in the Internet [1–3]. While loss concealment and channel coding techniques can improve overall performance in some cases, long sequences of packet losses causes a significant impairment. For example, when using the G.723.1 recommendation for compressed voice over IP networks (VoIP), only slight static and clipping result from one-to-four consecutive packet losses. However longer bursts of packet losses will significantly degrade the QoS delivered to the user. PLC and channel coding techniques attempt to hide the impact of a small number of losses. However, these techniques do not work when a large number of consecutive VoIP packets are lost. Thus, an impairment event occurs when a large number of consecutive packets are lost.

While long bursts of losses definitely cause user-perceived impairments, perceived quality also drops as random loss rate increases [4]. The minimum loss rate at which perceived quality becomes unacceptable for a majority of the users depends on the coding and loss concealment technique in use. For VoIP, mean opinion score (MOS) is a widely used metric to rate the quality of voice calls. MOS ranges from 0 to 5, with 5 being the best possible and 0 being the worst. A MOS smaller than 3.6 is considered unacceptable. Independent of the coding technique in use, MOS drops below 3.6 when random loss rate exceeds about 10% [5]. For RTM applications, channel coding is typically used in terms of block codes [6]. Specifically, for video streams a block code (e.g. Tornado code) is applied to a segment of k packets to generate an n packet block, where $n > k$. The channel encoder places k packets in a group and creates additional packets from them so that the total number of packets is n. This group of packets is transmitted to the receiver, which receives K packets ($n - K$ packets are lost). To perfectly recover a segment, a user must receive at least k packets (i.e. $K \geqslant k$) in the n packet block. If more than $n - k$ packets are lost then channel coding cannot recover any portion of the original segment. Some video coders adaptively increase $n - k$ when the packet loss rate is high. However, $n - k$ cannot be made arbitrarily large because coding delay and required capacity also increase with an increase in n. Moreover, many transport protocols decrease the rate when packet loss rate increases (to avoid congestion collapse) [7]. In our work, if the random packet loss rate is greater than some fixed threshold, then an impairment event is determined to have occurred.

High delays can also cause user-perceived impairments. A mouth-to-ear delay less than 150 ms is considered acceptable for most VoIP [8] applications. However, if the mouth-to-ear delay is greater than 400 ms, then most end-users are dissatisfied with the service. For multiplayer interactive network games, end-to-end delays greater than 200 ms are 'noticeable' and 'annoying' to end-users [9–11]. While end-users of sports and real-time strategy games are more tolerant to latency, even modest delays of 75–100 ms are noticeable in first person shooter and car racing games [9, 10]. RTM applications employ playout delay buffers at the receiver to

compensate for network jitter. When the jitter is very high, a large playout buffer is needed to avoid excessive packet losses due to late arrivals. Playout buffer delay however is added to the total delay (e.g. mouth-to-ear delay for VoIP). Thus, when the network jitter is high, playout delay buffer size is increased at the cost of increased total delay. In addition to the playout delay, source/channel coding/decoding delays also contribute to the total delay. In this work, when the sum of mean estimated one-way delay and playout buffer delay are greater than some threshold delay (such that interactivity is impacted), then an impairment event is inferred.

Impairment events can be caused by congestion or route changes. Congestion is a state of sustained network overload, where demand for resources exceeds the supply for an extended period of time. A congestion event may cause a number of consecutive packet losses. Congestion may also cause the random packet loss rate, mean delay and variation of delay to increase significantly, thus resulting in impairment events. However, congestion may not be sufficiently severe to cause an impairment. Congestion detection is needed to investigate the characteristics of impairment events that occur during congestion.

Route changes can also cause impairments (long bursts of lost packets). Route changes can be caused by router or link failures or when a failed component recovers from a failure. Failures are often followed by a service disruption that lasts from a few seconds to a few minutes while routing protocols converge to the new route [12–14]. Restoration at Layer 2 is usually faster than restoration at Layer 3 [15]. Layer 3 route changes can be detected at the end node from IP time to live (TTL) and traceroute changes (if intermediate network elements allow), whereas Layer 2 route changes are more difficult to detect. Thus, in some cases Layer 3 route changes can be explicitly detected while Layer 2 route changes must be indirectly inferred. A new algorithm is proposed here to detect Layer 2 route changes. Note that even though route changes do not always cause an impairment, route change detection is needed to investigate the correlation between route changes and impairments and to segregate appropriately the observations, e.g. round trip times (RTTs) into statistically homogenous regions (see Reference [16]).

In several measurement studies, end-to-end measurements have been used to detect failures or degradation events [5, 17–19]. In Reference [17], the authors discussed how non-intrusive active measurements were being used by Internet service providers (ISPs) to detect and identify faults from an operational perspective. Characteristics of degradations were studied in Reference [18] using end-to-end measurements. Methods to predict degradations were also discussed in Reference [18]. However, degradations in Reference [18] were simply defined to be significant deviations of RTTs. In Reference [5] a method was developed to assess quality of VoIP calls given end-to-end delays and loss data. End-to-end measurements were also collected over the Internet and used to estimate the VoIP call MOS. In Reference [19], the authors reported measurements of packet loss probability and outage characteristics. To our knowledge, there is no published work on the prediction of impairment events simultaneously for a wide class of end-users of RTM applications.

In this paper, end-to-end measurements are collected to identify *good* and *bad* regions (periods of time). A *bad* region is one in which it is predicted that a RTM application user would observe a noticeable impairment independent of the type of adaptive/coding technique or PLC technique in use. While subjective user tests were not performed, the inference of performance degradation is supported by other published subjective tests [5, 9, 10, 20]. In this work noticeable impairments (applicable to RTM services) occur when the end-to-end connection is in one or more of the following states: burst loss, high random loss, disconnected and delay impairment. In addition to these states, two other connection states are defined: congested and route change.

New methods are proposed to identify each of these states given packet losses, TTLs, measurements of RTTs and traceroutes. Conventional techniques are used for some measurements, but new techniques are also developed for detecting Layer 2 route changes (based on RTT measurements) and for detecting congestion (based on queuing theory and decision theory). The duration and rate of impairments from measurements over the Internet are also reported. End-to-end measurements were collected over 9 different node pairs for about 26 days using the Planet-Lab infrastructure [21]. The results indicate that while on some paths impairment events are rare, on others they occur at a high rate. The paths on which they occur at a high rate are determined to be congested. The measurements also reveal a correlation between Layer 3 route changes and impairments. Details of the measurement program are discussed in Section 2. In Section 3 algorithms are presented that determine connection state from packet losses, TTL, RTT and traceroute measurements. Procedures to detect congestion and route changes are developed in Sections 3.1 and 3.2. Results from Internet measurements are presented in Section 4.

2. MEASUREMENT METHODOLOGY

End-to-end active measurements were collected using the Planet-Lab [22] infrastructure. The measurement program used here is based on the client-server model. The client periodically sends user datagram protocol (UDP) packets to the server. On receiving a UDP packet from the client, the server records its sequence number and value stored in the TTL field of the IP header. It then generates a new UDP packet with the same sequence number as the sequence number of the incoming packet and sends it to the client. On receiving this packet, the client calculates the RTT and stores it along with the sequence number. It also records the value stored in the TTL field of the incoming packet. If the client does not receive a reply within t_l seconds (see Reference [21]) of sending a packet to the server, then that packet is considered lost. Optionally, the client can also be configured to send an Internet control message protocol (ICMP) packet to the server using ping every time it sends a UDP packet to the server. In that case, the RTTs returned by ping are stored in addition to RTTs calculated by the program using UDP packets. This option was enabled for the measurements because it was observed that RTT measurements using ping were less prone to end node operating system delay effects compared to RTT measurements using UDP. The probing rate changes are based on network conditions (for details see Reference [21]).

3. CONNECTION STATES

RTT measurements, packet losses, TTLs and traceroutes are used to determine the state of the connection. At any given time, the end-to-end connection is either in one or more of the states discussed in this section, or it is in the normal state. Congested and route change states are discussed first, followed by burst loss, disconnect, high random loss and delay impairment states. An observable impairment is inferred when the connection is in one or more of the states: burst loss, disconnect, high random loss and delay impairment. Since, not all route changes cause packet losses and since congestion may not be severe enough to cause an impairment, these two states are not considered impairment states. But congestion and route changes may cause one or more of the impairment states to occur.

3.1. Congested state

A queue is defined to be congested when the arrival rate of packets into the queue exceeds the service rate for an extended period of time. When one or more queues in the end-to-end connection are congested, the connection is in a congested state. It is well known that as the load increases, the mean and the variance of waiting times in queue increase. Specifically, for an $M/M/1$ queuing system [23], the mean and the variance of waiting times in queue are given by

$$M_W = \frac{\rho}{\mu - \lambda}, \quad \sigma_W^2 = \frac{\rho(2-\rho)}{(\mu-\lambda)^2}$$

where λ is the arrival rate, μ service rate and $\rho = \lambda/\mu$ is the load.

Let the ratio $\eta = \sigma_W/M_W$ (η is the coefficient of variation). Simplifying for η, it follows that

$$\eta = \sqrt{\frac{2-\rho}{\rho}}$$

Solving for ρ, there is the equality

$$\rho = \frac{2}{\eta^2 + 1} \tag{1}$$

Thus, given a set of waiting time samples from an $M/M/1$ queue, ρ can be estimated using the above equation. Clearly, real-world router queues are not accurately modelled as $M/M/1$ queues. Moreover, waiting time samples from each individual queue along an end-to-end connection are not observable at the end nodes. However, Equation (1) suggests a decision variable that should be correlated to the congestion along an end-to-end path. The pseudo-waiting times are extracted from the RTT samples and used to estimate the value of the decision variable $\tilde{\rho}$.

Let RTT_i be the ith RTT sample in ms, then the pseudo-waiting time w_i is given by

$$w_i = \mathrm{RTT}_i - \mathrm{MinRTT}$$

where MinRTT is the minimum of all RTT samples collected on the current route. Thus, if j and k are sequence numbers where route change events nearest to sequence number i occurred (route change detection is discussed later) and $j < i < k$, then

$$\mathrm{MinRTT} = \min\{\mathrm{RTT}_j, \mathrm{RTT}_{j+1}, \ldots, \mathrm{RTT}_{i-1}, \mathrm{RTT}_i, \mathrm{RTT}_{i+1}, \ldots, \mathrm{RTT}_k\}$$

However, the minimum RTT of the current route computed using the above procedure is not always accurate because of the timing issues on Planet-lab nodes [24]. Apparently, the minimum RTT drops to a very low value momentarily during network time protocol (NTP) resynchronization events. To remove these minimum RTT outliers caused by timing mechanisms, all RTT samples are removed that have a value less than a 1 percentile value of RTT samples from the current route. The minimum RTT is then computed using the remaining RTT samples.

The mean and the standard deviation of the waiting times are estimated over a window of cWindow samples.

$$\tilde{M}_i = \mathrm{mean}\{w_i, w_{i-1}, \ldots, w_{i-c\mathrm{Window}+1}\}$$

$$\tilde{\sigma}_i = \text{standard deviation}\{w_i, w_{i-1}, \ldots, w_{i-c\mathrm{Window}+1}\}$$

Then, the decision variable $\tilde{\rho}_i$ is formed by

$$\tilde{\rho}_i = \frac{2}{\tilde{\eta}_i^2 + 1} \quad \text{where } \tilde{\eta}_i = \frac{\tilde{\sigma}_i}{\tilde{M}_i}$$

RTTs that are collected over a period of one day are shown along with the decision variable $\tilde{\rho}$ in Figure 1(a). For a period of about 7–8 hours during the day, RTT is much longer than the mean RTT of 15 ms. Variation of RTTs and packet loss rate also increase substantially during

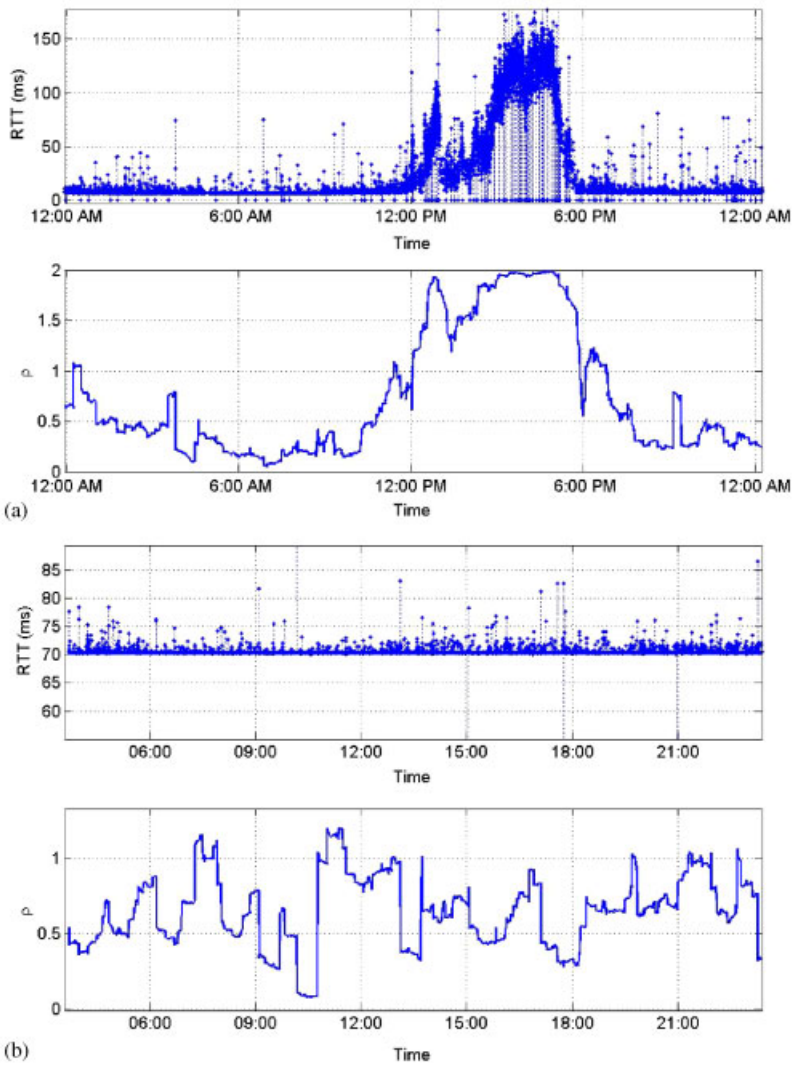


Figure 1. RTTs and decision variable $\tilde{\rho}$: (a) a case where $\tilde{\rho}$ is high when the load is high (planetlab2.ashburn.equinix.planetlab.org and planetlab1.comet.columbia.edu, February 2004); and (b) a case where $\tilde{\rho}$ is high when the load is very low (planet2.berkeley.intel-research.net and planet2.pittsburgh.intel-research.net, August 2004).

this period. Possibly, one of the router queues in the end-to-end connection is congested. It is clear from this figure that the decision variable $\tilde{\rho}$ is correlated with congestion as $\tilde{\rho}$ is higher during the congestion event.

At first, it might seem that choosing a constant threshold ρ_T and checking for the condition $\tilde{\rho}_i > \rho_T$ is sufficient to detect congestion. But this method results in many false positives as $\tilde{\rho}$ is high not only during congestion but also when queues in the end-to-end path are very lightly loaded. This is illustrated in Figure 1(b) where RTTs are almost the same. Mean and standard deviation of waiting times are close to zero. However, the decision variable $\tilde{\rho}$ has a value close to 1 for a significant portion of the trace. This happens because when the queues in the end-to-end path are very lightly loaded, waiting times w are close to 0. In that case, the variation in delay is small (e.g. from processing delays in routers, Ethernet contention delays, etc.). Thus, often $\tilde{\sigma}$ is less than $\tilde{M}$, i.e. the ratio $\tilde{\eta}$ is less than 1 and as a result $\tilde{\rho}$ is greater than 1.

Therefore, to remove the false positives two more conditions are checked. First, false positives occur when the mean waiting time is small. Thus, if $\tilde{M}_i < M_T$ (e.g. $M_T = 5$ ms) then the event is a false positive. Second, during congestion when arrival rate exceeds service rate for an extended period of time, packet losses are observed. Let $\tilde{L}_i$ be the observed percentage packet loss, i.e.

$$\tilde{L}_i = \frac{\text{number of losses from sequence numbers } (i - c\text{Window} + 1) \text{ to } i}{c\text{Window}}$$

If $\tilde{L}_i < L_T$ (e.g. $L_T = 0.016$) then the event is a false positive.

To summarize, if $(\tilde{\rho}_i > \rho_T)$ and $(\tilde{M}_i \geqslant M_T)$ and $(\tilde{L}_i \geqslant L_T)$ then a congestion event is detected. Figure 2 illustrates a congestion event detected using the above procedure with cWindow set to 160 samples and ρ_T set to 0.75.

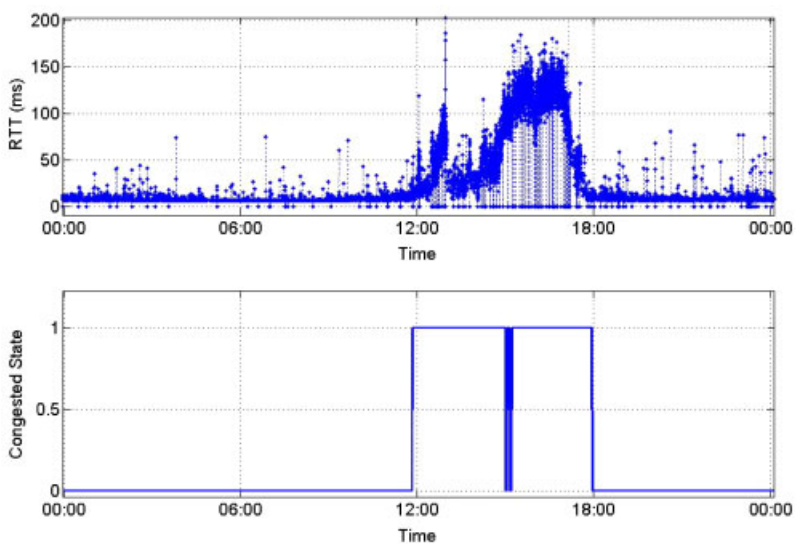


Figure 2. RTTs and a congestion event detected using the discussed procedure (planetlab2.ashburn.equinix.planetlab.org and planetlab1.comet.columbia.edu, February 2004).

3.2. Route change state

Route changes are usually caused by router or link failures or when the failed component recovers from an outage. Failures are followed by a service disruption that lasts from a few seconds to a few minutes while routing protocols converge to the new route. When the failed network component recovers from the failure, routers in the new route can instantly switch packet forwarding to the old route. Thus, there is not always a perceived impairment when network components recover from failures. In most cases, Layer 3 route changes can be detected at the end node using traceroutes and the TTL field in the IP header. However, not all route changes are visible at Layer 3. Route changes that occur at Layer 2 (from e.g. SONET, ATM) may be detected when minimum RTT of the new route is significantly different from minimum RTT of the old route. Also note that in some cases Layer 3 route changes are not directly visible at the end nodes because some routers are configured not to respond to traceroute messages.

3.2.1. Layer 3 route change. The measurement program periodically performs a traceroute, e.g. once every 15 min. If the route returned by current traceroute is not the same as the route returned by the previous traceroute then the route changed. Also, the program records the value in the TTL field of each arriving probe packet. If the TTL value of an arriving packet is different from the TTL value of a packet that arrived immediately before the current packet, then it is inferred that there has been a Layer 3 route change. Since probe packets are sent at a higher rate than traceroute measurements are performed, route changes are detected faster using the TTL change method. However, not all route changes cause a TTL change (e.g. when the new route has same number of hops as the old route).

3.2.2. Layer 2 route change. Figure 3 shows the graph of RTTs observed on 12 August, 2004 between planet2.berkeley.interl-research.net and planet2.pittsburgh.intel-research.net. The minimum RTT increases from 68 to 75 ms initially and then subsequently decreases to 68 ms. Neither the TTLs nor the routes returned by traceroute changed during this period. Thus, it can be inferred that there was a Layer 2 route change. The algorithm that follows detects such Layer 2 route changes from the RTT pattern under certain conditions.

First, RTTs are grouped into non-overlapping windows and minimum RTT of each window is calculated. The minimum RTT of the most recent window is then compared to minimum RTTs of previous windows to answer two questions. First, is the utilization of queues low enough for the Layer 2 route change detection algorithm to work (see Figure 4(a))? If the answer to the first question is yes, then second, has the minimum RTT changed? A change in the minimum RTT is either due to a route change or due to random delays possibly arising from queuing (see Figure 4(b)). When the minimum RTT changes, the algorithm tries to determine whether or not it was caused by a route change.

Minimum RTT is computed as follows. W is the route change window size, e.g. $W = 40$ samples. i is the sequence number of the most recent RTT sample. S_i is the set of W sequence numbers, $S_i = \{i, i-1, i-2, \ldots, i-W+1\}$. RTT^{S_i} is the set of W RTT samples: $\mathrm{RTT}^{S_i} = \{\mathrm{RTT}_i, \mathrm{RTT}_{i-1}, \mathrm{RTT}_{i-2}, \ldots, \mathrm{RTT}_{i-W+1}\}$. $\min[\mathrm{RTT}^{S_i}]$ is minimum of all RTTs in the set RTT^{S_i}. Then $\min[\mathrm{RTT}^{S_i}]$ is the minimum RTT of the current window.

Minimum RTT of the current window is compared to minimum RTTs of the previous windows. The set prevMins_i stores minimum RTTs of previous windows. numPrevWindows_i is the number of windows since the most recent route change. If the most recent route change

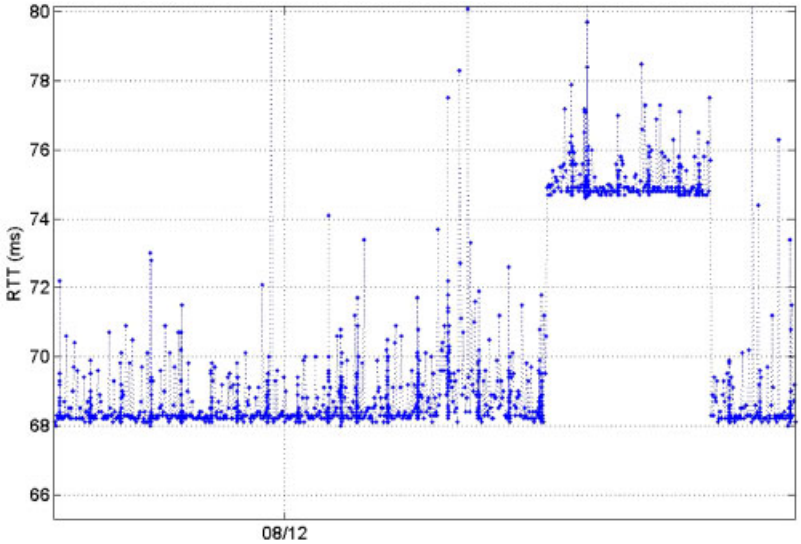


Figure 3. Layer 2 route change observed between planet2.berkeley.intel-research.net and planet2.pittsburgh.intel-research.net on 12 August, 2004.

occurred at sequence number j ($j<i$), then $\text{numPrevWindows}_i = \lfloor (i-j+1)/W \rfloor$. prevMins_i is the set of minimum RTTs of previous $\text{numPrevWindows}_i - 1$ windows, i.e. $\text{prevMins}_i = \{\min[\text{RTT}^{S_{i-W}}], \min[\text{RTT}^{S_{i-2W}}], \ldots, \min[\text{RTT}^{S_{i-(\text{numPrevWindows}-1)W}}]\}$.

The first step in the process is to compare the minimum RTT of the current window to minimum RTTs of previous windows to determine whether or not variation in RTTs is low enough for Layer 2 route changes to be detected. Figure 4(a) illustrates the case where delay variation is high and Layer 2 route changes cannot be reliably inferred from the observations. $\text{RTT}_k = \text{FixedDelay}_k + \text{VariableQueuingDelay}_k + \text{VariableOtherDelays}_k$, where FixedDelay_k is the sum of fixed propagation and transmission delays, $\text{VariableQueuingDelay}_k$ is the sum of all queuing delays experienced by packet k and $\text{VariableOtherDelays}_k$ is the sum of all other variable delays (e.g. link layer contention delay). It is assumed that $\text{VariableOtherDelays}_k$ are less than ε (e.g. $\varepsilon = 0.5$ ms) [25]. If all queues in the end-to-end path have low utilization, then it is likely that at least one of the W RTT probe packets experiences zero queuing ($\min[\text{RTT}^{S_k}] = \text{FixedDelay}_m + \text{VariableOtherDelays}_m, m \in S_k$). In that case, minimum RTTs in the set prevMins_i differ from each other by an amount less than ε, i.e. $\max[\text{prevMins}_i] - \min[\text{prevMins}_i] < \varepsilon$. If however, one or more queues in the end-to-end path have significant utilization then it is likely that none of the W RTT probe packets experience zero queuing ($\min[\text{RTT}^{S_k}] = \text{FixedDelay}_m + \text{VariableQueuingDelay}_m + \text{VariableOtherDelays}_m, m \in S_k$). In that case, minimum RTTs in the set prevMins_i usually have a range greater than ε ($\max[\text{prevMins}_i] - \min[\text{prevMins}_i] > \varepsilon$).

Thus, if $\text{std}[\text{prevMins}_i] < \varepsilon$ then the delay variation is not significant and Layer 2 route changes can be detected. Otherwise one or more queues in the end-to-end path may be congested and Layer 2 route changes cannot be accurately detected using this approach.

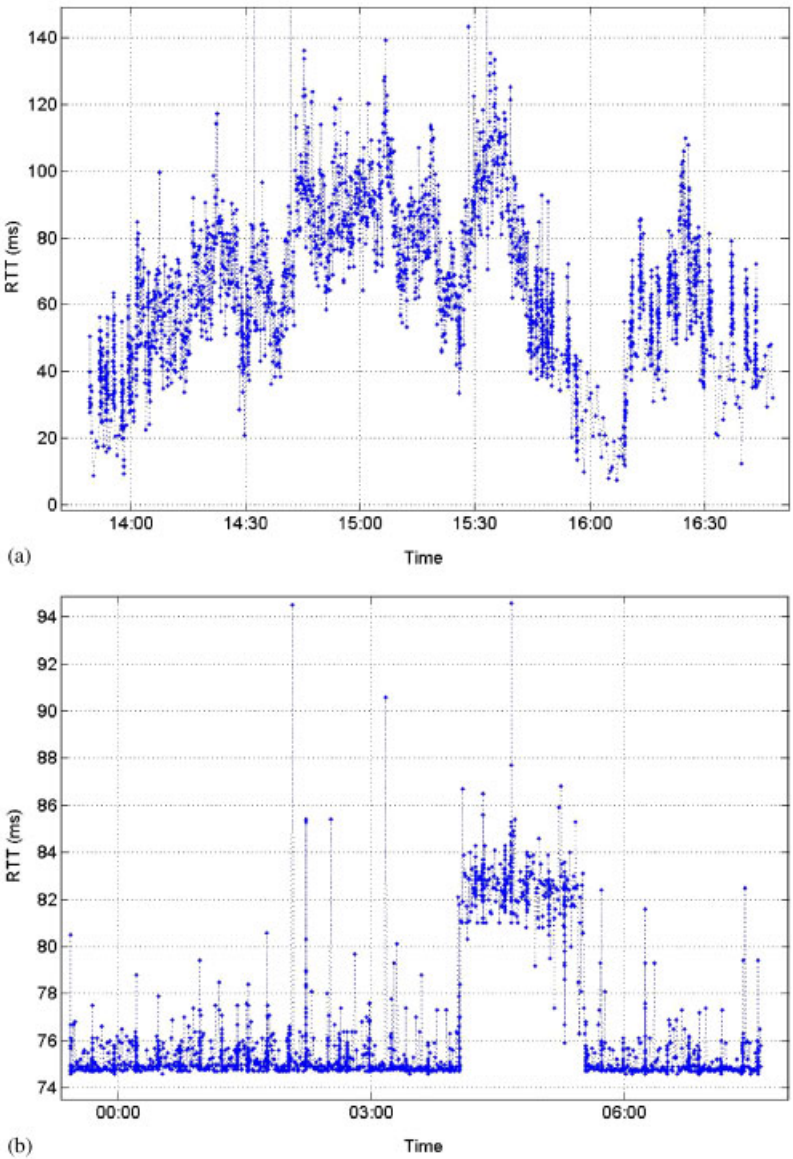


Figure 4. Detecting Layer 2 route changes: special cases: (a) an example case where Layer 2 route changes cannot be detected because queuing delays are very large (Ashburn-Columbia, February 2004); and (b) an example of minimum RTT change caused by a heavily loaded queue (Berkeley-Pittsburgh, August 2004).

Next, minimum RTT of the current window is compared to minimum RTTs of previous windows to determine whether or not the minimum RTT has changed. If either $\min[\mathrm{RTT}^{S_i}] < (\mathrm{median}[\mathrm{prevMins}_i] - \varepsilon)$ or if $\min[\mathrm{RTT}^{S_i}] > (\mathrm{median}[\mathrm{prevMins}_i] + \varepsilon)$ then it is inferred that the minimum RTT has changed.

This minimum RTT change is either caused by a route change or delay variation possibly caused by queuing in a heavily loaded router queue. Figure 4(b) illustrates minimum RTT change caused by a large delay variation. Minimum RTT changes from 74 ms to a little over 80 ms in this case. If $(\text{median}[\text{RTT}^{S_i}] - \min[\text{RTT}^{S_i}]) < \kappa$ (e.g. $\kappa = 1$ ms) then it is inferred that the minimum RTT change is caused by a route change. However, it is possible that there is a Layer 2 route change and one of the switches in the new route is heavily loaded. To detect that condition, examine $\gamma \times W$ (e.g. $\gamma = 3$) more RTT samples. If $(\text{median}[\text{RTT}^{S_i}] - \min[\text{RTT}^{S_i}]) > \kappa$ ms then $\gamma \times W$ more RTT samples are collected before it can be decided whether or not it was a route change that caused the minimum RTT change. Let $n = i + (\gamma \times W)$, where i is the sequence number of most recent RTT probe packet. $S_{i:n}$ is the set of sequence numbers from i to n, i.e. $S_{i:n} = \{i, i+1, \ldots, n-1, n\}$. $\text{RTT}^{S_{i:n}}$ is the set of RTT samples $\text{RTT}^{S_{i:n}} = \{\text{RTT}_i, \text{RTT}_{i+1}, \ldots, \text{RTT}_{n-1}, \text{RTT}_n\}$. $\min[\text{RTT}^{S_{i:n}}]$ is the minimum of all RTTs in the set $\text{RTT}^{S_{i:n}}$. If for each $p, p \in S^{i:n}, \min[\text{RTT}^{S_p}] \leqslant (\min[\text{RTT}^{S_{i:n}}] + \varepsilon)$, then it is inferred that the minimum RTT change is caused by a route change. Otherwise it is inferred that delay variation caused the minimum RTT change.

If a route change is detected, routes returned by traceroute and TTLs are compared. If neither traceroutes nor TTLs change, then it is inferred that there was a Layer 2 route change. Figure 5 illustrates Layer 2 route changes detected using the above procedure. The RTTs were collected in August 2004 between planet2.berkeley.intel-research.net and planet2.pittsburgh.intel-research.net. While the algorithm successfully detected most of the Layer 2 route changes, it failed to detect the three marked as missed events. Note that when the first missed event occurred, very few RTT samples (less than $(v+1) \times W$ samples) were collected since the most recent successfully detected route change. The set prevMins_i should have at least v (e.g. $v = 3$) elements before minimum RTT changes can be detected. Since, less than W RTT samples were collected in the new route before another route change occurred, and since the set prevMins was empty, this route change was not detected. For the second and third missed events, the

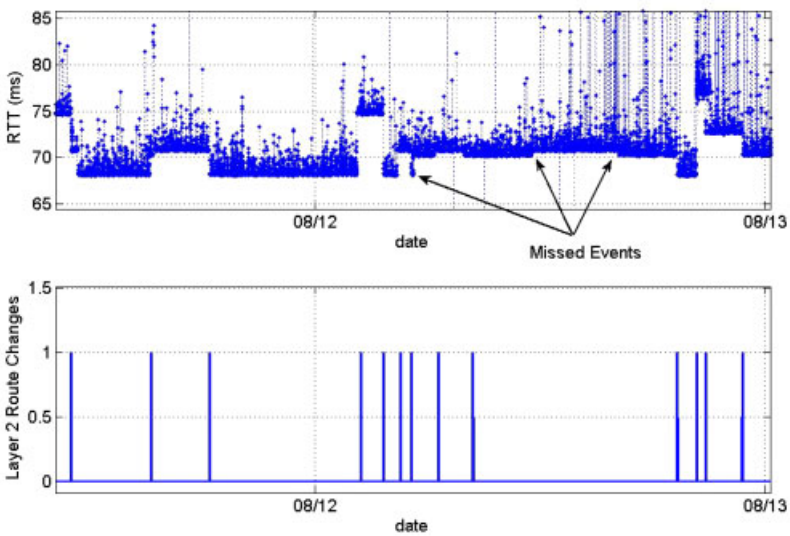


Figure 5. Layer 2 route change detected using the discussed procedure (planet2.berkeley.intel-research.net and planet2.pittsburgh.intel-research.net, August 2004).

minimum RTT changed by 0.4 ms and ε was set to 0.5 ms. Since the algorithm can detect route changes only when the minimum RTT changes by more than ε ms, these route changes were not detected. In the measurements collected over the Internet for about 26 days on 9 different node pairs, 96 events were manually found by visual analysis of RTT data that looked like route changes. The Layer 2 route change algorithm detected 71.8% of these 96 events. The other events were not detected, either because the minimum RTT changed by a value less than ε, or because after the first route change, very few RTT samples were collected in the new route before the route changed again. The algorithm also generated some false positives (detected a route change when visually we could not find any route changes). About 4% of the detected events were false positives.

The minimum RTT of a path can change not only when there is a Layer 2 route change but also due to a Layer 3 route change. Thus, this algorithm for detecting Layer 2 route changes can also be used to detect Layer 3 route changes. This is useful when administrators restrict ICMP response from routers and traceroute cannot be used to detect Layer 3 route changes. For the measurements that were conducted, Layer 3 route changes detected using the RTT-based algorithm were compared with the ones detected using traceroute or TTL changes. For 38.7% of all Layer 3 route changes detected using traceroute or TTL changes, minimum RTT change was not significant (less than ε ms) and hence the RTT-based algorithm did not detect these route changes. About 38.1% of the Layer 3 route changes detected using traceroute or TTL changes were either preceded by another Layer 3 route change (with less than $v \times W$ RTT samples between the two route changes) or followed by another Layer 3 route change (with less than W RTT samples between the two route changes). These route changes were also not detected by the RTT based algorithm. We expect the proposed RTT based algorithm to detect a Layer 3 route change when there are enough RTT samples and the change in RTT is significant; in those cases the technique detected 89.7% of the route changes. The algorithm missed 10.3% of those Layer 3 route changes. Future work could lead to improvements in the algorithm to increase the probability of detecting route changes.

3.3. Burst loss state

Packet loss concealment, coding and interleaving techniques do a good job of masking audio and video effects of packet losses. But when a large number of consecutive packets are lost over several seconds, these techniques do not help. Multiple consecutive losses typically cause a noticeable impairment and then the connection is in burst loss state. Formally, when all transmitted probe packets are lost for more than ξ (e.g. $\xi = 6$) seconds (but less than ψ seconds) then the connection is in the burst loss state. Congestion events [26] and route changes [14] can cause burst losses. Specifically, service disruption due to IS-IS route convergence has been reported to be at least 6.6 s [14].

3.4. Disconnected state

When all transmitted consecutive packets are lost for a very long period, then an event of a different nature (e.g. other than congestion) is directly responsible for the losses. If all transmitted probe packets are lost for ψ or more seconds (e.g. $\psi = 300$) then the connection is defined to be in the disconnected state. Such outages can be caused by failures at the edge or in the core of the network [27]. Failures can be caused by many events, e.g. scheduled maintenance, loss of power, fiber cut, hardware failure, malicious attack, software bugs, configuration errors,

etc. [13, 28, 29]. At the edge, where the end customer connects to its service provider, traffic cannot be routed around the failure and the outage persists until the problem is resolved. In the core, traffic can be routed around the failure but routing protocols take from several seconds to several minutes to converge [12]. In the meantime, routing errors occur, causing outages for the end-user.

3.5. High random loss state

While long bursts of losses definitely cause user-perceived impairments, a high random loss rate is also observable by users. PLC and interleaving techniques do a good job of masking effects of packet losses for audio and video streams as long as the random loss rate is below some threshold. Above this threshold rate, packet losses introduce audio and video impairments that cause perceived QoS to decrease to an unacceptable level. For random losses, i.e. non-consecutive losses, let the threshold packet loss probability be τ. Then, if loss probability is greater than τ it can be inferred that the connection is in high random loss state.

The procedure to detect high random loss state is based on the premise that at least M (e.g. $M = 10$) loss events are needed to obtain an acceptable estimate of loss probability [30], i.e. for the standard deviation of the estimate for the loss probability to be on the order of ($0.1 \times$ loss probability), approximately 10 loss events must be observed. In this algorithm, the trace is scanned for packet losses in an increasing order of sequence numbers until M loss events are found. Loss probability is then inferred from the distance between first and Mth lost packet's sequence numbers. If the first and Mth lost packets are very far apart then loss probability is low. If the losses are close to each other then loss probability is high. A threshold distance $\zeta = \lfloor M/\tau \rfloor$ corresponds to loss probability τ. If the difference between Mth lost packet's sequence number and first lost packet's sequence number is greater than ζ, then loss probability is less than τ; otherwise if this difference is less than ζ, then loss probability is greater than τ. When the loss probability is greater than ζ, connection is in high random loss state. The above procedure to detect high random loss state is then repeated for 2nd and $(M + 1)$th lost packets, 3rd and $(M + 2)$th lost packets and so on.

VoIP MOS is a function of loss probability and it decreases as random loss probability increases. It is evident from the discussion in Reference [5] that the shape of the MOS curve depends on a number of factors such as codec used, PLC technique used and whether packet losses are bursty or uniform. For most codecs and PLC techniques, MOS is below 3.6 when random loss probability is greater than 0.1 (see Reference [5]). MOS below 3.6 is considered unacceptable. Three values of τ are evaluated here: $\tau = 0.05, 0.1$ and 0.15.

3.6. Delay impairment state

Most real-time applications experience transparent interactivity when the latency is less than some threshold. High delays can be noticeable and annoying to end-users of RTM applications. In VoIP the mouth-to-ear delay is the sum of coding/decoding delay, network delay and delay in dejitter/playout buffer. Coding delay depends on the type of coding technique in use and dejitter buffer delay depends on playout technique in use and on the network jitter. When the jitter is very high, a large playout buffer is needed to avoid excessive packet losses due to late arrivals. Thus, when the network jitter is high, playout delay buffer size is increased at the cost of increased mouth-to-ear delay. When the sum of mean one-way delay and dejitter buffer delay is greater than some threshold delay, then the interactivity is severely impacted and the connection is defined to be in delay impairment state.

Adaptive playout delay techniques attempt to minimize the playout delay while avoiding excessive packet loss due to late arrival of packets at the receiver [31,32]. Given the observable RTT data, an estimate is made of the minimum playout delay buffer size that is needed to avoid excessive packet losses. Most adaptive playout schemes will likely have a playout buffer that is larger than this minimum. Since RTT measurements and not one-way delay measurements are collected, it is necessary to first form the one-way delays. Round trip propagation delay is simply the minimum RTT of the current route or MinRTT (more details in the discussion of congested state). A simplifying assumption is made that the forward and the reverse paths are symmetric and the one-way propagation delay is one half MinRTT. Subtracting one-way propagation delay from RTTs gives an approximation for the one-way delays. As is evident, simplifying assumptions are used to form one-way delays from RTTs, however if one-way delays are available the procedure discussed below to estimate minimum playout delay can be applied directly.

Let the one-way delay estimate for RTT_i be OWD_i and let jWindow be the window size (e.g. 160 samples). Then, M_i^{O} is the sample mean of all one-way delay samples in a window of jWindow samples and S_i^{O} is the sample standard deviation, i.e.

$$M_i^{\mathrm{O}} = \mathrm{mean}\{\mathrm{OWD}_{i-j\mathrm{Window}+1}, \mathrm{OWD}_{i-j\mathrm{Window}+2}, \ldots, \mathrm{OWD}_i\}$$

$$S_i^{\mathrm{O}} = \text{standard deviation}\{\mathrm{OWD}_{i-j\mathrm{Window}+1}, \mathrm{OWD}_{i-j\mathrm{Window}+2}, \ldots, \mathrm{OWD}_i\}$$

Then, $M_i^{\mathrm{O}} + S_i^{\mathrm{O}}$ is one possible estimate of the minimum playout delay that is needed to avoid excessive packet losses due to late arrivals. Most playout schemes will likely have a playout delay greater than this minimum. Estimated one-way delays and minimum playout delays are shown in Figure 6. When this minimum playout delay exceeds a threshold delay (i.e. $M_i^{\mathrm{O}} + S_i^{\mathrm{O}} > D^{\max}$) then it is inferred that interactivity for RTM applications is impacted

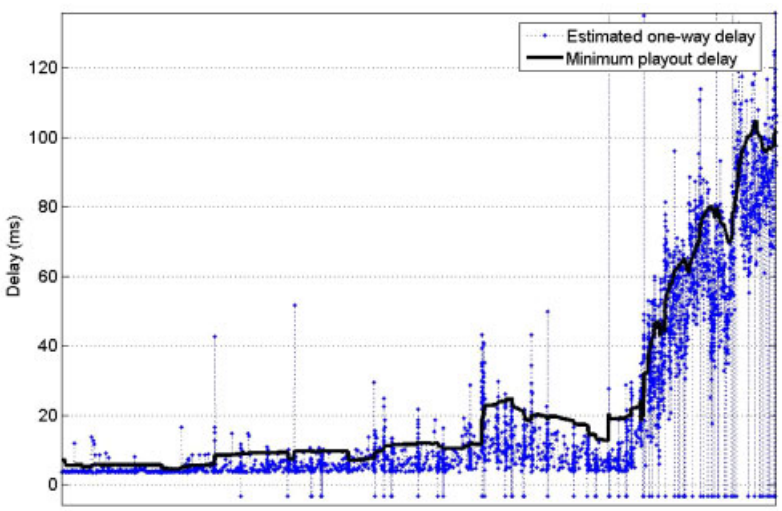


Figure 6. Estimated one-way delays and minimum playout delay (planetlab2.ashburn.equinix.planetlab.org and planetlab1.comet.columbia.edu in February 2004).

(regardless of the type of playout scheme really in use) and the connection is defined to be in delay impairment state. To evaluate this approach, three different thresholds for our measurements: 100, 150 and 200 ms are considered.

4. MEASUREMENT RESULTS

End-to-end measurements were conducted for 26 days on 9 different end-node pairs connected to the Planet-Lab infrastructure. This section discusses the characteristics of impairment events that were observed in these measurements. Paths on which congestion events were observed had a large number of high random loss and burst loss events. Congestion events were observed on two out of the nine pairs. Characteristics of these congestion events are discussed. Layer 3 route changes were observed on all the 9 pairs and Layer 2 route changes on 6 pairs.

Table I lists the location of end nodes, dates and number of days on which data were collected. Node pairs for which both end nodes are within the United States are labelled DC (for domestic-commercial) since at least one of the two end nodes is a non-Internet2 node. When both end nodes are Internet2 nodes (nodes in a university or a research center), then traffic is routed through Internet2 routers that have a very low load. For this reason, only pairs were

Table I. Measurement sites, dates and number of days on which data were collected.

Label	Node pair	Dates	Number of days
DC1	Ashburn (Virginia) [planetlab2.ashburn.equinix.planetlab.org]—Columbia Univ. (New York) [planetlab1.comet.columbia.edu]	6 Feb. to 27 Feb. 2004	21
DC2	Columbia Univ. (New York) [planetlab2.comet.columbia.edu]—Sanjose (California) [planetlab2.sanjose.equinix.planetlab.org]	18 Feb. to 27 Feb. 2004 5 March to 18 March 2004	10 14
DC3	Berkeley (California) [planet2.berkeley.intel-research.net]—Pittsburgh (Pennsylvania) [planet2.pittsburgh.intel-research.net]	8 Aug. to 2 Sep. 2004	26
DC4	Seattle (Washington) [planet1.seattle.intel-research.net] Santa Clara (California) [planetlab-2.scla.nodes.planetlab.org]	8 Aug. to 2 Sep. 2004	26
DC5	Seattle (Washington) [planet2.seattle.intel-research.net]—Sterling (Virginia) [planetlab-2.stva. nodes. planetlab.org]	8 Aug. to 2 Sep. 2004	26
I1	Cambridge (United Kingdom) [planetlab1.cambridge.intel-research.net]—Berkeley (California) [planet1.berkeley.intel-research.net]	8 Aug. to 2 Sep. 2004	26
I2	Athens (Greece) [planetlab1.cslab.ece.ntua.gr]—Cornell Univ. (New York) [planetlab1.cs.cornell.edu]	13 Aug. to 2 Sep. 2004	21
I3	Zurich (Switzerland) [planetlab02.ethz.ch—Copenhagen] (Denmark) [planetlab1.diku.dk]	8 Aug. to 2 Sep. 2004	26
I4	Durham (North Carolina) [planetlab1.cs.duke.edu]—Taipei (Taiwan) [planetlab1.iis.sinica.edu.tw]	8 Aug. to 2 Sep. 2004	26

used for which at least one node is a non-Internet2 node. Node pairs labelled *I* (International) have either one or both end nodes outside the United States. There are 21 days of data for node pairs DC1 and I2, 24 days for DC2 and 26 days for the remaining node pairs.

Statistics of all detected impairment events are listed in Table II. Rate of impairments is higher for DC1 and DC2 than for the rest of the pairs. For DC1 and DC2, impairment events occur at an average of once every five hours or less. For all other pairs except I1, mean time between events is more than 60 h. Mean duration of impairment events that occur in DC1 and DC2 is also longer than the mean duration of events in other paths. Mean duration of events for DC1 and DC2 ranges from 35 to 92.5 min, while the mean duration of impairments for the other sites ranges from 4 to 28 min. No impairments were observed on I4. Mean duration of impairment events for all sites combined is 40 min and mean time between impairment events is 9.8 h.

Table III lists the observed mean number of burst loss events, high random loss events (loss probability > 0.05), congestion events and delay impairment (> 100 ms) events per day for each data set. Congestion and high random loss events occurred only in data sets DC1 and DC2.

Table II. Statistics of user-perceived impairments.

Data set	Mean duration of impairments (minutes)	Mean time between impairments (hours)	Mean number of impairments per day
DC1	35.9	3.52	7
DC2 (Feb.)	92.5	5.22	4
DC2 (Mar)	37.3	4.95	4.78
DC3	9.7	62.97	0.23
DC4	4.4	89.44	0.115
DC5	15.61	268	0.004
I1	14.5	15.2	1.38
I2	28.4	121.17	0.19
I3	4.62	122.47	0.15
I4	—	—	0

Table III. Mean number of loss, congestion and delay impairment events per day.

Data set	Mean number of burst loss events per day	Mean number of high random loss events (loss rate > 5%) per day	Mean number of congestion events per day	Mean number of delay impairment (100 ms) events per day
DC1	7	6.7	2.5	2.28
DC2	1.125	4.08	1.66	2
DC3	0.11	0	0	0.07
DC4	0	0	0	0.07
DC5	0	0	0	0.07
I1	0.8	0	0	1.57
I2	0	0	0	4.76
I3	0.07	0	0	3.42
I4	0	0	0	0

Mean number of burst loss events per day in DC1 and DC2 are greater than number of burst loss events in other data sets.

An interesting trend is observed in both data sets DC1 and DC2. Connection state alternates between congested and normal states for 6–8 h during daytime on weekdays. This is illustrated using 1 week of data from DC1 in Figure 7 (day labels on time axes correspond to start of that day). Except for Sunday night, congestion events started at about 11 AM and lasted until 6 or 7 PM on weekdays. On this particular week, congestion events were also observed on late Sunday night and early Monday morning. Histograms of duration and time between congestion events for DC1 and DC2 are shown in Figures 8(a) and 8(b). Mean duration of congestion is 42.6 min and mean time between congestion events is 4.4 h.

For DC1 and DC2, burst loss, high random loss and high delay impairments occurred when the connection was in congested state. For DC1, delay and random loss impairments occurred for 59.2% of the time while connection was in the congested state. For DC2, impairments occurred for more than 90% of the time the connection was in the congested state. Table IV lists the impairment time during which the connection is not in the congested state. It is clear that most of the delay and high random loss impairments in DC1 and DC2 occur when the connection is in the congested state. From a total of 174 burst loss events that occurred in DC1 and DC2, 61% occurred when the connection was in the congested state. Mean time between burst loss events that occur when the connection is in the congested state is 14 min and the mean duration is 22.64 s. About 75% of all burst loss events that occur when the connection is in the congested state are less than 8 min apart and 50% are less than 2.5 min apart. Therefore impairments are more likely to occur when the connection is in the congested state as defined here.

Mean and standard deviation of duration and time between high random loss, delay impairment and congestion events are listed in Table V. These statistics were computed using data from all the 9 node pairs. But since congestion and high loss events occur only in data sets

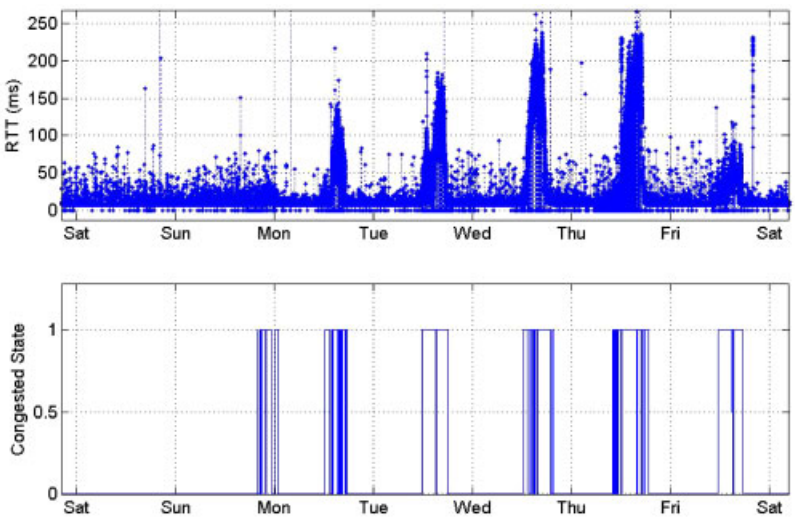


Figure 7. Congestion events observed over a period of 1 week (DC1).

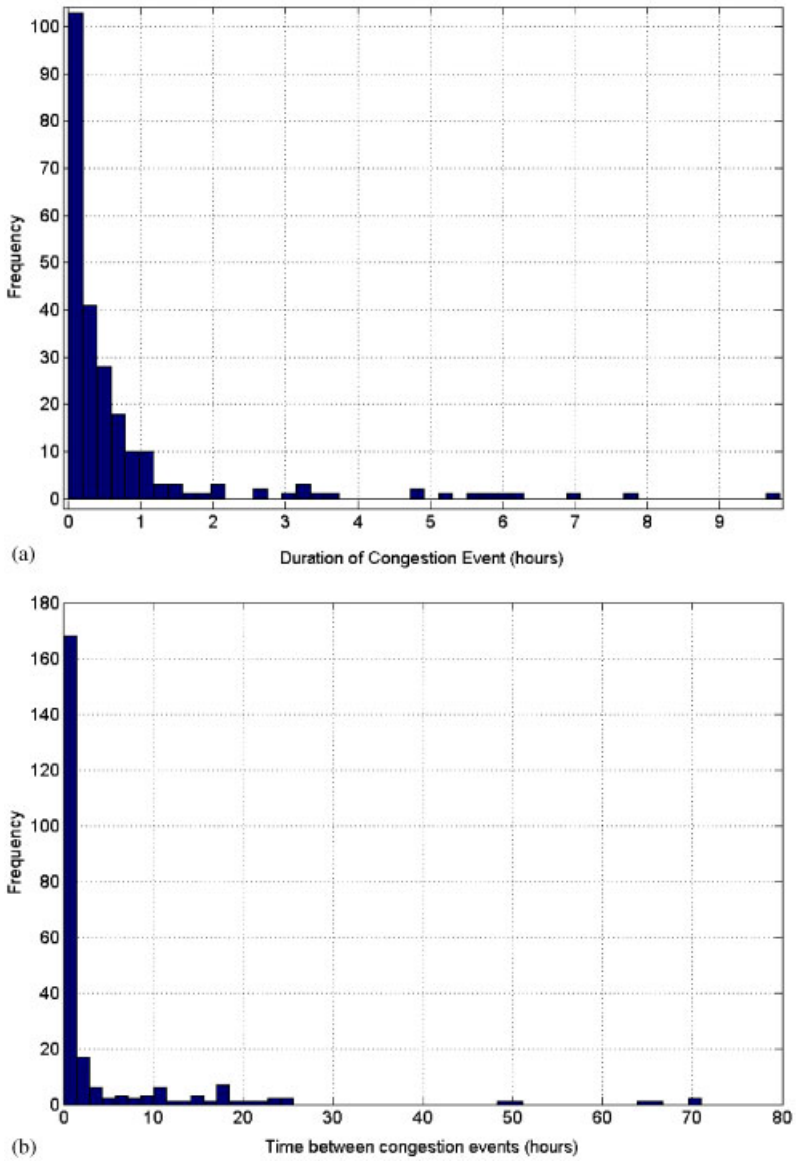


Figure 8. Duration and time between congestion events on DC1 and DC2: (a) histogram of duration of congestion events; and (b) histogram of time between congestion events.

DC1 and DC2, mean and standard deviation for these events were computed using data only from data sets DC1 and DC2.

Table VI lists the mean number of Layer 2 and Layer 3 route changes per day for all the data sets. The CDF of time (in hours) between Layer 3 route changes is shown in Figure 9. About 80% of all Layer 3 route changes are less than 45 min apart and mean time between Layer 3 route changes is 7.23 h. Figure 10 shows the histogram of time (in seconds) between Layer 3

Table IV. Percentage of impairment state time during which connection was not in congested state.

Impairment	DC1	DC2 (Feb)	DC2 (March)
High random loss	46.4	23.4	48.1
High delay	19.9	11.2	17.7

Table V. Mean and standard deviation of duration and time between events.

Event	Mean duration (minutes)	Std. dev. of duration (minutes)	Mean time between events (minutes)	Std. dev. of time between events (minutes)
Congestion	42.63	82.3	264	648
High random loss (5%)	119.8	180.8	666	822
High random loss (10%)	45.16	58.32	522	1206
High random loss (15%)	13.9	38.1	355.2	1128
Delay impairment (100 ms)	37.07	76.9	685.8	2063.4
Delay impairment (150 ms)	39.06	120.85	2295.6	4434
Delay impairment (200 ms)	24.99	16.42	4222.2	5987.4

Table VI. Observed number of route changes per day.

Data set	Mean number of Layer 2 route changes per day	Mean number of Layer 3 route changes per day
DC1	0	0.428
DC2	0.041	4.08
DC3	0.69	2.76
DC4	0	1
DC5	0.34	0.115
I1	0.23	5.84
I2	0.095	1.14
I3	0	3.8
I4	0.146	3.76

route changes. About 18% of all Layer 3 route changes are 1 s apart and about 15% are 2 s apart. These are caused by frequent route changes that occur when routers are trying to converge to a new route. Since the measurement client sent probe packets once every second on detecting a TTL change, frequency of 1 and 2 s times is high in the histogram. A histogram of time (in hours) between Layer 2 route changes is shown in Figure 11. Time between route changes for about 40% of Layer 2 route changes is less than 3 h. Mean time between Layer 2 route changes is 58.22 h.

Since route changes occur when a network component fails, they are often preceded by packet losses. About 8% of all Layer 3 route changes were preceded by burst or disconnect loss events. Mean duration of loss events that precede Layer 3 route changes is 113.5 s. This is about five

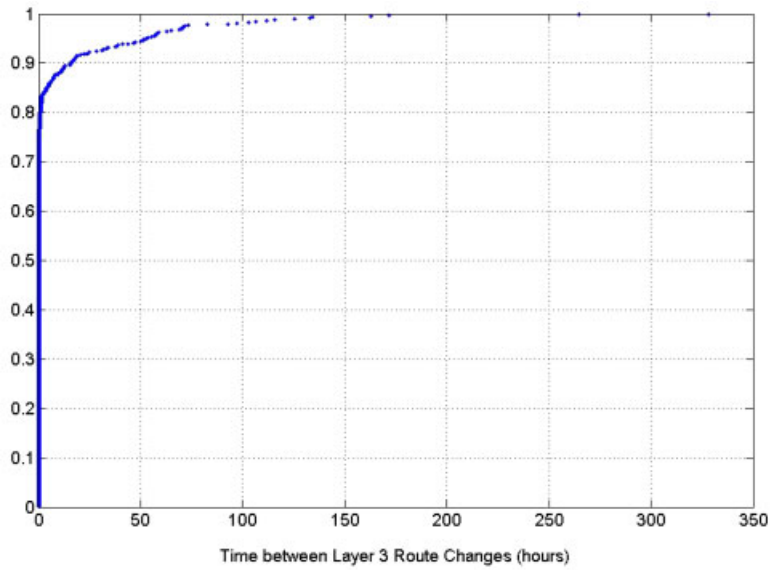


Figure 9. CDF of time between Layer 3 route changes.

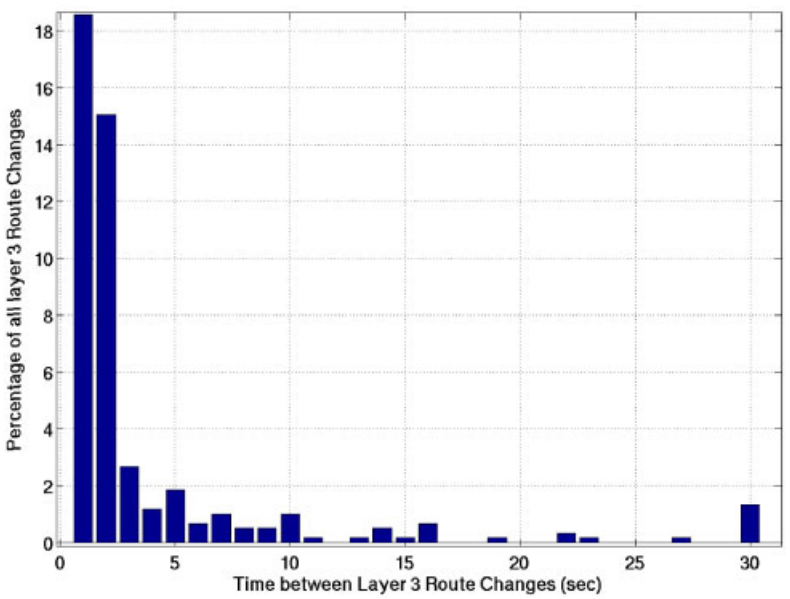


Figure 10. Histogram of time between Layer 3 route changes.

times the mean duration of burst loss impairment events that occur during the congestion state (22.64 s). Mean time between burst loss events that precede Layer 3 route changes is more than 7.23 h while the mean time between burst loss events during congestion was only 14 min. No correlation between Layer 2 route changes and packet losses was observed. The fact that there

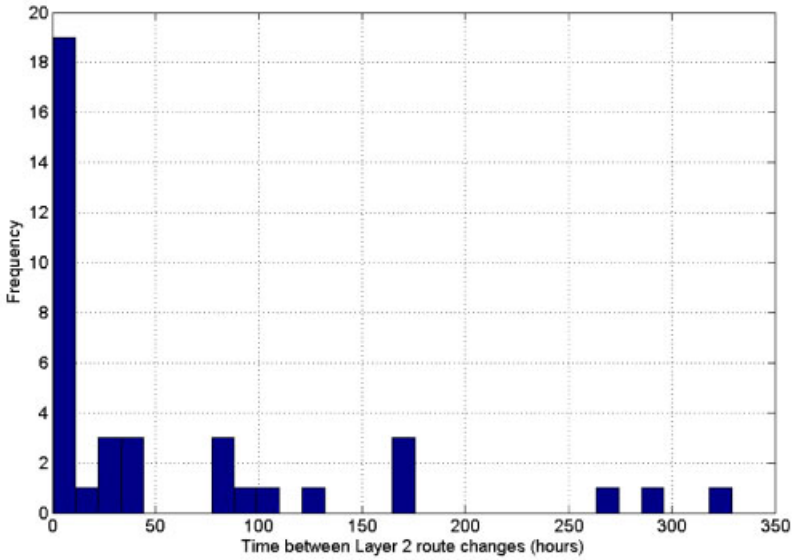


Figure 11. Histogram of time between Layer 2 route changes.

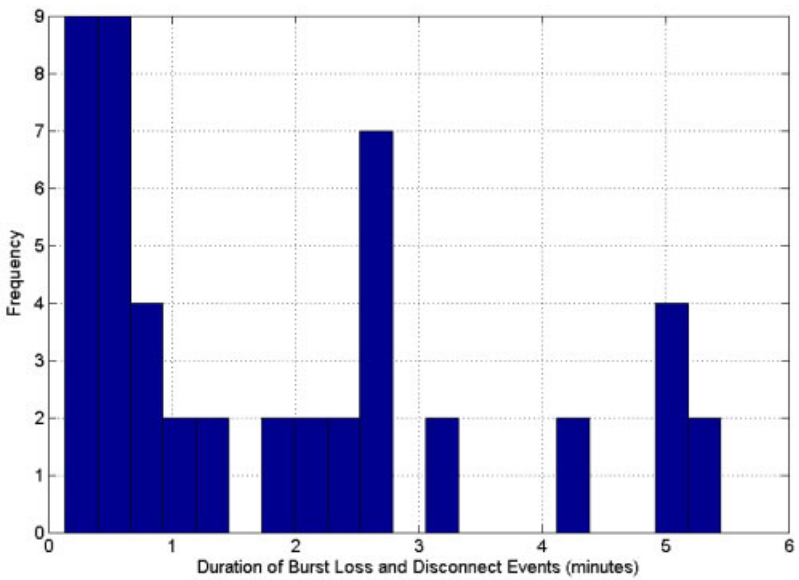


Figure 12. Histogram of duration of burst loss and disconnect events that precede Layer 3 route changes.

were no burst or disconnect loss events preceding Layer 2 route changes reinforces the idea that Layer 2 restoration is faster than restoration at Layer 3. A histogram of duration of the loss events preceding a Layer 3 route change is shown in Figure 12. Maximum duration for which there were packet losses before a route change event is 5.45 min.

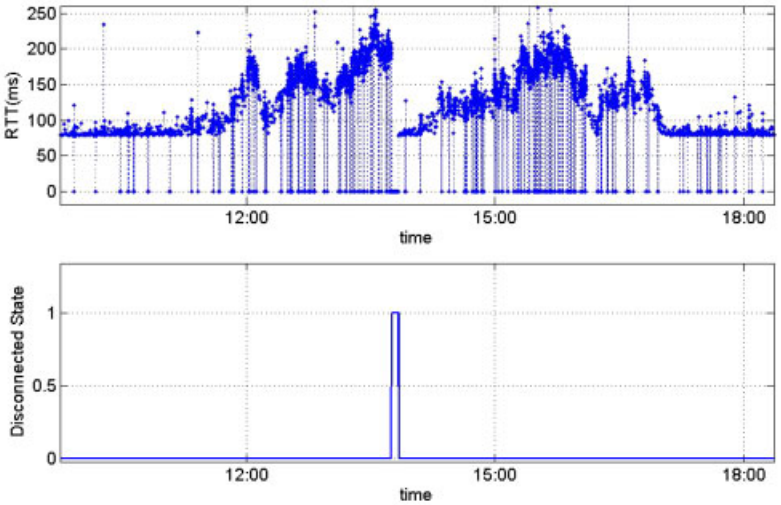


Figure 13. Disconnect event due to problem in congested router.

A total of seven disconnect events were observed. For five out of the seven disconnect events at least one traceroute was performed during the disconnect event. Examination of these traceroutes reveals that in all five cases the problem was in a router 3 or 4 hops from one of the end nodes. Either there was a routing loop or traceroute packets did not return from the problem router. The two disconnect events for which traceroute was not performed during the disconnect event occurred during congestion. Figure 13 illustrates one of the two events. The other disconnect event looks similar to this one. At approximately time 14:00, a disconnect event occurred. Since mean delay drops substantially after the disconnect event (and slowly starts increasing again), it seems that the disconnect event occurred due to a problem in the congested router. Although no traceroutes were performed during the disconnect event, the TTL changed for the first two packets that were sent immediately after the disconnect event.

5. CONCLUSION

End-users of RTM applications are likely to experience a noticeable performance degradation when the connection is in burst loss, high random loss, disconnected or delay impairment state. The contribution of this paper includes procedures to detect these impairment states using end-to-end measurements. Techniques to detect Layer 2 route changes and congestion were also developed. A contribution of this work is a set of techniques that simultaneously consider multiple metrics to detect the network state. End-to-end measurements were conducted on 9 different node pairs for about 26 days to evaluate the developed techniques.

Congestion was observed on two out of the nine paths. Random loss, delay and burst loss impairments occurred for most of the 6–8 hour period during the day on weekdays when the two connections were congested. Combining observations from all nine paths, burst loss events preceded 8% of the Layer 3 route changes and lasted for a mean time of about 2 min. In contrast, burst loss events that occurred during congestion had a mean duration of only 22 s. Mean time between impairments caused by route changes was 7.23 h. To summarize, when

congestion occurred, it persisted for several hours and impairments occurred at a high rate during that period. Also, impairments caused by route changes were rare but lasted for several minutes.

The primary application of this work is the definition and measurement procedure for new QoS metrics that can be used in service level agreements (SLAs). Delay and loss are commonly used QoS metrics reported as a part of the SLAs. These metrics may have little direct meaning to the end-user because knowledge of specific coding and/or adaptive techniques is required to translate delay and loss to the user-perceived performance. The impairment events defined here provide a new set of QoS metrics for real-time multimedia applications. Impairment events, as defined in this paper, are observable by the end-users independent of coding, adaptive playout or packet loss concealment techniques employed by their multimedia applications. Internet service providers can use the procedures developed in this paper to detect the impairment events within their networks. The time between impairments and duration of impairments are easy to understand QoS metrics that can be reported to the customers as a part of the SLAs. Customers can use the same procedures to verify the statistics reported in the SLAs.

The procedures developed in this paper have several other applications. Techniques to detect congestion and route changes can be used at end nodes and by the service providers to determine what caused the impairments. Overlay [33] and content delivery networks [34] may use these procedures to predict duration and recurrence of impairments. Traceroute, which is a popular tool used to detect route changes may, not always work because it requires ICMP response from routers (sometimes administrators restrict the ICMP response to prevent probing from external sources [35]). In such cases, the algorithm developed in this paper to detect Layer 2 route changes could be used to detect Layer 3 route changes. However, this algorithm does not detect route changes when delay difference between the old and the new routes is small, when the delay variation is high or when route changes occur very closely together in time. Some applications like service mirroring, distributed games and peer-to-peer applications require an estimate of the minimum path RTT between hosts [36]. These applications can use the developed algorithm to detect route changes from RTT measurements. The minimum path RTT estimates can then be updated on detecting the route changes. Also, the tool RTTometer [36] that measures minimum RTT of the path, can incorporate RTT based route change detection to form better minimum path RTT estimates. The route change detection algorithm can also be used to segregate a sequence of RTT measurements into statistically homogenous regions [16].

Future work will focus on developing and validating a theoretical framework for the state detection process and determining the probability of false alarms and missed events. The route change detection algorithm will also be studied in more detail and detection results validated using either simulation or emulation.

ACKNOWLEDGEMENTS

This work was supported in part by NSF Grant ANI-0125410.

REFERENCES

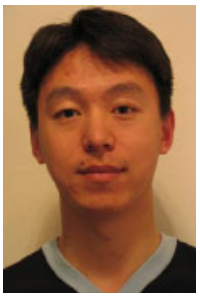
1. Borella M, Swider D, Uludag S, Brewster G. Internet packet loss: measurement and implications for end-to-end QoS. In *ICPPW '98*: *Proceedings of the 1998 International Conference on Parallel Processing Workshops*, Minneapolis, MN, U.S.A., 1998; 3–15.

2. Markovski V, Trajkovic L. Analysis of loss episodes for video transfer over UDP. In *Proceedings of the SCS Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'2K)*, Vancouver, British Columbia, Canada, 2000; 278–285.
3. Jiang W, Schulzrinne H. QoS measurement of Internet real-time multimedia services. In *Proceedings of NOSSDAV*, Chapel Hill, NC, June 2000.
4. James JH, Bing C, Garrison L. Implementing VoIP: a voice transmission performance progress report. *IEEE Communications Magazine* 2004; **42**(7):36–41.
5. Markopoulou AP, Tobagi FA, Karam MJ. Assessing the quality of voice communications over Internet backbones. *IEEE/ACM Transactions on Networking* 2003; **11**(5):747–760.
6. Wu D, Hou YT, Zhang Y-Q. Transporting real-time video over the Internet: challenges and approaches. *Proceedings of the IEEE* 2000; **88**(12):1855–1877.
7. Floyd S, Handley M, Padhye J, Widmer J. Equation-based congestion control for unicast applications. In *SIGCOMM 2000*, Stockholm, Sweden, August 2000; 43–56.
8. ITU-T Recommendation G.114, May 2003.
9. Beigbeder T, Coughlan R, Lusher C, Plunkett J, Agu E, Claypool M. The effects of loss and latency on user performance in unreal tournament 2003. In *Proceedings of ACM SIGCOMM 2004 Workshops on NetGames '04,* Portland, OR, U.S.A., 2004; 144–151.
10. Nichols J, Claypool M. The effects of latency on online Madden NFL football. In *NOSSDAV '04: Proceedings of the 14th International Workshop on Network and Operating Systems Support for Digital Audio and Video,* Kinsale, County Cork, Ireland, 2004; 146–151.
11. Pantel L, Wolf LC. On the impact of delay on real-time multiplayer games. In *NOSSDAV '02: Proceedings of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video,* Miami, FL, U.S.A., 2002; 23–29.
12. Labovitz C, Ahuja A, Bose A, Jahanian F. Delayed Internet routing convergence. In *Proceedings of ACM SIGCOMM,* Stockholm, Sweden, 2000; 175–187.
13. Labovitz C, Ahuja A, Jahanian F. Experimental study of Internet stability and wide-area backbone failures. *Technical Report CSE-TR-382-98*, University of Michigan, 1998.
14. Iannaccone G, Chuah C, Mortier R, Bhattacharyya S, Diot C. Analysis of link failures in an IP backbone. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop,* Marseille, France, November 2002.
15. Iannaccone G, Chuah C-N, Bhattacharyya S, Diot C. Feasibility of IP restoration in a tier-1 backbone. *IEEE Networks Magazine* (Special Issue on Protection, Restoration and Disaster Recovery) 2004; **18**(2):13–19.
16. Zhang Y, Du N, Paxson V, Shenker S. On the constancy of Internet path properties. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, USA, November 2001.
17. Ciavattone L, Morton A, Ramachandran G. Standardized active measurements on a tier 1 IP backbone. *IEEE Communications Magazine* 2003; **41**(6):90–97.
18. Bremler-Barr A, Cohen E, Kaplan H, Mansour Y. Predicting and bypassing end-to-end internet service degradations. *IEEE Journal on Selected Areas in Communications* 2003; **21**(6):961–978.
19. Jiang W, Schulzrinne H. Assessment of VoIP service availability in the current Internet. In *Proceedings of the Passive and Active Measurement Workshop*, La Jolla, CA, U.S.A., April 2003.
20. Claypool M, Zhu Y. Using interleaving to ameliorate the effects of packet loss in a video stream. In *23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03)*, Providence, RI, U.S.A., May 2003.
21. Cook M, Lu J, Frost V. End-to-end monitoring of network traffic and congestion events on the Planet-Lab network with the pmclient and pmserver tools. *Technical Report ITTC-FY2005-TR-30540-01*, Information and Telecommunication Technology Center, University of Kansas, Lawrence, KS, September 2004.
22. Planet-lab website. http://www.planet-lab.org.
23. Kleinrock L. *Queueing Systems*, *Volume 1*: *Theory*. Wiley: New York, 1975.
24. Timing problem on planetlab (bad NTP), March 2004, http://lists.planet-lab.org/pipermail/users/2004-March/000050.html.
25. Papagiannaki K, Moon S, Fraleigh C, Thiran P, Diot C. Measurement and analysis of single-hop delay on an IP backbone network. *IEEE Journal on Selected Areas in Communications* 2003; **21**(6):908–921.
26. Frost VS. Quantifying the temporal characteristics of network congestion events for multimedia services. *IEEE Transactions on Multimedia* 2003; **5**(4):458–463.
27. Yin S, Twist K. The coming era of absolute availability. *RHK White Paper*, May 2003, http://www.chiaro.com/pdf/rhk.pdf.
28. Donelan S. Internet outage trends. *NANOG Meeting*, February 2001, http://www.nanog.org/mtg-0102/donelan.html.
29. Gilmer GH. The real-time IP network: moving IP networks beyond best effort to deliver real-time applications. *AVICI Systems White Paper*, 2003.
30. Shanmugan KS, Breipohl AM. *Random Signals: Detection, Estimation and Data Analysis*. Wiley: New York, 1988.

31. Moon SB, Kurose J, Towsley D. Packet audio playout delay adjustment: performance bounds and algorithms. *Multimedia Systems* 1998; **6**(1):17–28.
32. Ramjee R, Kurose JF, Towsley DF, Schulzrinne H. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *Proceedings of IEEE INFOCOM,* Toronto, Canada, 1994; 680–688.
33. Andersen DG, Balakrishnan H, Kaashoek F, Morris R. The case for resilient overlay networks. In *8th Workshop on Hot Topics in Operating Systems*, Elmau/Oberbayern, Germany, May 2001.
34. Akamai website. http://www.akamai.com.
35. Duffield N. Simple network performance tomography. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement,* Miami Beach, FL, U.S.A., 2003; 210–215.
36. Zeitoun A, Wang Z, Jamin S. Rttometer: measuring path minimum rtt with confidence. In *3rd IEEE Workshop on IP Operations and Management (IPOM 2003),* Kansas City, MO, U.S.A., 2003.

AUTHORS' BIOGRAPHIES

Soshant Bali is currently pursuing a PhD degree in Electrical Engineering at the University of Kansas. His advisor is Dr Victor S. Frost. He received a BE degree in Electronics Engineering from Mumbai University (India) in July 2000 and MS degree in Electrical Engineering from Virginia Tech in December 2002. His research interests include Internet measurements and quality of service.



Yasong Jin is a PhD candidate in the Mathematics Department of the University of Kansas. He received the BS degree in Automation from Beijing Polytechnic University, China, in 1998 and the MA degree in Mathematics from the University of Kansas in 2003. His research interests include stochastic analysis, network modelling and large deviation theory. He has been working on NSF granted projects on stochastic systems and quality of service in communication networks since August 2001. He is a student member of AMS, SIAM and IEEE.

Dr Victor S. Frost is currently the Dan F. Servey Distinguished Professor of Electrical Engineering and Computer Science and Director of the University of Kansas Information and Telecommunications Technology Center (ITTC). He is a Fellow of the IEEE and received a Presidential Young Investigator Award from the National Science Foundation in 1984. His current research interest is in the areas of Internet quality of service, traffic management, and integrated broadband communication networks. He has been involved in research on several national scale high speed wide area testbeds. Some of his publications have focused on reporting the measured performance of these wide area broadband networks. Government agencies, including, NSF, DARPA, Rome Labs, and NASA have sponsored his research. Dr Professor Frost has been involved in research for numerous corporations, including Harris, Sprint, NCR, BNR, Telesat Canada, AT&T, McDonnell Douglas, DEC, and COMDISCO Systems. He has published over 100 journal articles and conference papers. Dr Frost received the BS, MS, and PhD degrees from the University of Kansas, Lawrence in 1977, 1978, and 1982, respectively. In 1982 he joined the faculty of the University of Kansas.

Tyrone E. Duncan (M'92-SM'96-F'99) received the BEE degree from Rensselaer Polytechnic Institute in 1963 and the MS and PhD degrees from Stanford University in 1964 and 1967, respectively. He has held regular positions at the University of Michigan (1967–1971), the State University of New York, Stony Brook (1971–1974) and the University of Kansas (1974–present), where he is Professor of Mathematics. He has held visiting positions at the University of California, Berkeley (1969–1970), the University of Bonn, Germany (1978–1979), and Harvard University (1979–1980) and shorter visiting positions at numerous other institutions. Dr Duncan is a Corresponding Editor of SIAM Journal on Control and Optimization and is a member of AMS, MAA, and SIAM.