1-800-OVERLAYS: Using Overlay Networks to Improve VoIP Quality*

Yair Amir
Johns Hopkins University
Department of Computer
Science

Claudiu Danilov
Johns Hopkins University
Department of Computer
Science

Stuart Goose
Siemens Corporate Research
Multimedia Department

David Hedqvist
Siemens Corporate Research
Multimedia Department

Andreas Terzis
Johns Hopkins University
Department of Computer
Science

ABSTRACT

The cost savings and novel features associated with Voice over IP (VoIP) are driving its adoption by service providers. Such a transition however can successfully happen only if the quality and reliability offered is comparable to the existing PSTN. Unfortunately, the Internet's best effort service model provides no inherent quality of service guarantees. Because low latency and jitter is the key requirement for supporting high quality interactive conversations, VoIP applications use UDP to transfer data, thereby subjecting themselves to performance degradations caused by packet loss and network failures.

In this paper we describe two algorithms to improve the performance of such VoIP applications. These mechanisms are used for localized packet loss recovery and rapid rerouting in the event of network failures. The algorithms are deployed on the routers of an application-level overlay network and require no changes to the underlying infrastructure. Initial experimental results indicate that these two approaches can be composed to yield voice quality on par with the PSTN.

Categories and Subject Descriptors: C.2.4 [Computer Communication Networks]: Distributed Systems

General Terms: Design, Measurement

Keywords: Overlay Networks, VoIP

1. INTRODUCTION

It is non-trivial to engineer a system that meets the stringent constraints expected by humans for high quality, reliable, and real-time voice communications. Delays of 100-150 msec and above are detectable by humans and can impair the interactivity of con-

---

*This work was partially funded by NSF grant CNS0430271.

versations. By comparison, humans are far less tolerant of audio degradation than of video degradation. Hence, to meet these requirements it is crucial to minimize primarily the network latency and secondarily packet loss. To minimize latency, contemporary VoIP solutions rely upon UDP as the transport protocol. However doing so has the potential to expose VoIP packets to packet loss and network failures. Although the Internet can offer reasonable quality (relatively low loss and good stability) most of the time, it has been shown [1, 2, 3] that it remains vulnerable to occasional bursts of high loss and link failures. This prohibits the Internet from delivering the constant, high quality service demanded for telephony.

In this paper we describe an overlay architecture that improves the performance of VoIP applications during the intervals when Internet service suffers. It maintains a high packet delivery ratio even under high loss, while adding minimal overhead when no losses occur. In this architecture, application endpoints communicate through a series of overlay routers rather than directly to each other. Dividing the end-to-end path to a number of overlay hops has a number of benefits. First, it is often possible to recover packets even given the tight delay budget of VoIP. Even when the time budgets prohibit timely end-to-end recovery, it is possible to perform local recovery over the overly link where a packet is lost. Because these overlay links have small RTT compared to the end-to-end path, the majority of lost packets can be recovered while satisfying the delay budget. Second, the routing algorithm used by the overlay network can be tuned to avoid overlay links that are congested, experience high loss, or become unavailable. Due to the specialized nature of the algorithm such path adaptation can happen at time scales that are an order of magnitude smaller than the Internet, minimizing the impact on voice streams.

The main contributions of this paper are two complimentary algorithms that can be implemented in an overlay network tailored to VoIP: First, a real-time[1] packet recovery protocol that immediately delivers newly received packets, similarly to UDP, but this protocol attempts to recover missing packets. Recovery is attempted only once, and only if a packet is likely to arrive at the destination within the VoIP delay constraint. This protocol is deployed on every overlay link. Second, an adaptive overlay routing protocol tailored to

---

[1]Our definition of real-time refers to timely recovery of packets on short overlay links. Protocols such as RTP, RTCP, that do not recover packets, work independently of our protocols and benefit from our higher packet delivery ratio.
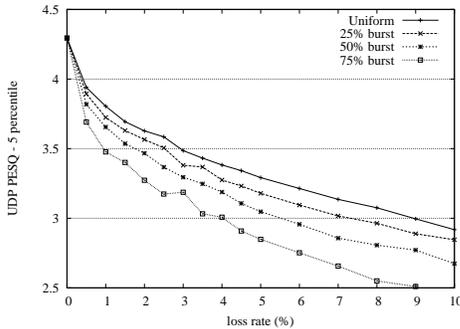
Figure 1: Network loss - 5 percentile PESQ

VoIP, that optimizes path selection based on an approximation metric that combines the measured latency and loss of a link.

The rest of the paper is organized as follows: In Section 2 we present the motivation and background of our work. Section 3 introduces our overlay architecture. We present and evaluate our protocols in Section 4. Section 5 discusses related work, and we conclude in Section 6.

2. BACKGROUND

2.1 Voice over IP

Unlike media streaming, VoIP communication is *interactive*, i.e. participants are both speakers and listeners at the same time. In this respect, delays higher than 100-150 msec can greatly impair the interactivity of conversations, and therefore delayed packets are usually dropped by the receiver codec.

Voice quality can be adversely affected by a number of factors including high latency, jitter, and node or link failures. The combined impact, as perceived by the end-users, is that voice quality is unpredictable—usually good but with periods of bad quality. Current VoIP codecs use buffering at the receiver side to compensate for slightly delayed packets, and use forward error correction (FEC) or packet loss concealment (PLC) mechanisms to ameliorate the effect of packet loss or excessive delay. The error correction mechanisms usually add redundancy overhead to the network traffic and have limited ability to recover from bursty or sudden loss increase in the network.

In our experiments we use a well-understood, widely deployed and good quality codec, the standard ITU-T G.711 [4], combined with its packet loss concealment mechanism [5]. The G.711 codec we used samples the voice signal at a rate of 8kHz and partitions the data stream into 20 msec frames, thus sending 160 byte packets at a rate of 50 packets/sec. VoIP quality is evaluated using an objective method described in ITU-T recommendation P.862 [6], known as Perceptual Evaluation of Speech Quality (PESQ). The PESQ score is estimated by processing both the input reference and the degraded output speech signal, similarly to the human auditory system. The PESQ score ranks speech signals on a scale from -0.5 (worst) to 4.5 (best), where 4.0 is the desired PSTN quality.

2.2 Internet loss characteristics

Packets are lost in the Internet due to congestion, routing anomalies and physical errors, although the percentage of physical errors is very small at the core of the network. Paxson in [2] studied the loss rate for a number of Internet paths and found that it ranged from 0.6% to 5.2%. Furthermore in that study and a follow-up [7], Paxson discovered that loss processes can be modeled as spikes

where loss occurs according to a two-state process, where the states are either "packets not lost" or "packets lost". According to the same studies, most loss spikes are very short-lived (95% are 220 msec or shorter) but outage duration spans several orders of magnitude and in some cases the duration can be modeled by a Pareto distribution. In a recent study, Andersen et al. confirmed Paxson's earlier results but showed that the average loss rate for their measurements in 2003 was a low 0.42% [3]. Most of the time, the 20-minute average loss rates were close to zero; over 95% of the samples had a 0% loss rate. On the other hand, during the worst one-hour period monitored, the average loss rate was over 13%. An important finding in [3] is that the conditional probability that a second packet is lost given that the first packet was lost was 72% for packets sent back-to-back and 66% for packets sent with a 10-msec delay, confirming the results in [7].

In addition to link errors and equipment failures, the other major factor contributing to packet losses in Internet is *delayed convergence* of BGP. Labovitz et al. found that 10% of all considered routes were available less than 95% of the time and that less than 35% of all routes were available more than 99.99% of the time [8]. In a followup study [9], Chandra et al. showed that 5% of all failures last more than 2 hours and that failure durations are heavy-tailed and can last as long as 20 hours before being repaired. All these statistics indicate the Internet today is not ready to support high quality voice service as we are going to show in the following section.

2.3 Voice quality degradation with loss

We evaluated the effect of loss patterns similar to those reported on the Internet on the VoIP quality, using the standardized PESQ measure. To do so, we instantiated a network with various levels of loss and burstiness (we define burstiness as the conditional probability of losing a packet when the previous packet was lost) in the Emulab [10] testbed, and measured the quality degradation when sending a VoIP stream.

We used the G.711 codec with PLC to transfer a 5 minute voice audio file using UDP over the lossy network, repeating each experiment for 20 times. The network had a 50 msec delay and 10 Mbps capacity, enough to emulate a trans-continental long-distance call over a wide area network. We finally decoded the audio file at the destination, dividing it into 12 second intervals corresponding to normal conversation sentences, and compared each interval with the original to generate its PESQ score.

On average, the G.711 codec could handle up to 1% loss rate, while keeping a PESQ score higher than 4.0 (the expected PSTN quality level). However, given the regular expectancy of high quality phone calls, we analyzed the most affected voice streams in this experiment. Figure 1 presents the lower 5 percentile of the PESQ score of all the sentence intervals, as a function of loss rate and burstiness on the link. We can see that for the most affected streams, burstiness has a significant impact, and even at 0.5% loss rate the G.711 codec cannot provide PSTN standard voice quality, as for 75% burstiness the PESQ score dropped to 3.69.

Since current loss rate measurements in the Internet average at about 0.42% with an average burstiness of 72%, and that occasionally loss can be even much higher, we believe that new solutions are required to improve the quality of VoIP traffic if it is to compete with the existing PSTN.

3. AN OVERLAY ARCHITECTURE

Overlay networks allow easy deployment of new services, as they allow full control over the protocols running between participating nodes. As opposed to the Internet provides generic commu-

nication solutions, an overlay network usually has a limited scope and therefore can deploy application aware protocols.

Spines [11, 12] is an open source overlay network that offers a two-level hierarchy, in which applications (clients) connect to an overlay node, usually the closest. This node is responsible for forwarding and delivering data to the final destination through the overlay network. The destination overlay node delivers the packet to the receiver application. The benefit of this hierarchy is that it limits the size of the overlay network, thus reducing the amount of control traffic exchanged between the nodes.

Spines nodes connect to each other using *virtual links* forming the overlay network. Spines offers a number of protocols on each virtual link, including a best effort service, a TCP-fair reliable protocol [12] and a real time recovery protocol that we describe below in section 4.1. Each overlay node pings its direct neighbors periodically to check the link status and latency. Spines nodes add a link specific sequence number on every data packet sent between two neighboring overlay nodes. The receiving overlay node uses this sequence number to detect missed packets and estimate loss rate of the link. Based on link loss and latency, a cost for each link is computed as described in Section 4.2 and propagated through the network by an incremental link-state mechanism that uses reliable control links created among neighboring Spines nodes.

4. IMPROVING VOIP QUALITY

Current VoIP systems use the UDP best effort delivery service to transfer data. One of the main reasons for not using packet retransmission protocols is that lost packets, even when recovered end-to-end from the source, are not likely to arrive in time for the receiver to play them. Moreover, reliable protocols such as TCP temporarily block in case of retransmission failures or timeouts. Overlay networks break end-to-end streams into several hops, and even though an overlay path may be longer than the direct Internet path between the two end-nodes, each individual overlay hop usually has smaller latency, thus allowing localized recovery on lossy overlay links.

4.1 Real-time recovery protocol

Our overlay links run a real-time protocol that recovers packets only if there is a possibility to deliver them in time, and forward packets even out of order to the next hop. We describe our real time recovery protocol next:

- Each node in the overlay keeps a circular packet buffer per outgoing link, maintaining packets sent within a time equal to the maximum delay supported by the voice codec. Old packets are dropped out of the buffer when they expire, or when the circular buffer is full.

- Intermediate nodes forward packets as they arrive, even out of order.

- Upon detecting a loss on one of its overlay links, a node asks the upstream node for the missed packet. A retransmission request for a packet is sent only once. By using negative acknowledgments only, we limit the amount of control traffic when no packets are lost.

- When an overlay node receives a retransmission request it checks in its circular buffer, and if it has the packet it resends it, otherwise it does nothing. A token bucket mechanism regulates the maximum ratio between the number of retransmissions and the number of data packets sent. This way we limit the number of retransmissions on highly lossy links.

- If a node receives the same packet twice (say because it was requested as a loss, but then both the original and the retransmission arrive), only the first instance of the packet will be forwarded towards the destination.

The protocol does not involve any positive acnowledgements, so unless packets are lost there is no aditional traffic sent on the network except for the mamagement of the overlay topology itself. In addition, there are no timeouts, and the protocol never blocks for recovering of a packet. The downside is that this is not a fully reliable protocol and some of the packets will be lost. Such events can appear when a packet is lost, the next packet arrives (this is how the loss is detected) and triggers a retransmission request, but the retransmission request is also lost. For a link with independent loss rate p in both directions[2], this happens with probability $p \cdot (1-p) \cdot p = p^2 - p^3$. Another significant case is when the retransmission request does arrive, but the retransmission itself is lost, which can happen with probability $p \cdot (1-p) \cdot (1-p) \cdot p = p^2 - 2p^3 + p^4$. Other types of events, that involve multiple data packets lost can happen, but their probability of occurrence is negligible. We approximate the loss rate of our real-time protocol by $2p^2 - 3p^3$, since p^4 is negligible for small values of p.

The delay distribution of packets follows a step function, such that for a link with delay T and loss rate p, $(1 - p)$ fraction of packets arrive in time T, $(p - 2p^2 + 3p^3)$ are retransmitted and arrive in time $3T + \Delta$, where Δ is the time it takes the receiver to detect a loss, and $(2p^2 - 3p^3)$ of the packets will be lost by the real time recovery protocol. For a path that includes multiple links, the delay of the packets will have a compound distribution given by the combination of delay distributions of each link of the path. The time Δ it takes the receiver to trigger a retransmission request depends on the inter-arrival time of the packets (the receiver of a link needs to receive a packet to know that it lost the previous one) and on the number of out of order packets that the protocol can tolerate. For a single VoIP stream, packets usually carry 20 msec of voice, so they arrive at relatively large intervals. However, multiple voice streams are aggregated over the same overlay link. Therefore, the inter-packet delay seen at an overlay link is much lower than that of a single VoIP stream. Packet losses are identified by gaps in the sequence numbers of packets arriving at the receiving end of each overlay links. While TCP uses three out of order packets as indication of loss, we issue a retransmission request after receiving the first out of order packet. This may generate some false positives, however, we do so because latency is crucial for VoIP applications, and recent evidence show that packet reordering in the network happens relatively rarely [13].

We implemented the real time protocol in the Spines overlay network platform and evaluated its behavior by running Spines on Emulab. Figure 2 shows the loss rate of the real time recovery protocol on a symmetric 10 msec link with various levels of loss and burstiness, and Figure 3 shows the combined loss for two concatenated 10 msec links that experience the same amount of loss and burstiness, in both directions, running Spines with the real-time protocol on each link. For each experiment, an application sent traffic representing the aggregate of 10 VoIP streams for a total of two million packets, and then average loss rate was computed. As is evident from the graphs, the level of burstiness on the link does not affect the loss rate of the real-time protocol. The real-time loss rate follows a quadratic curve that matches our $2p^2 - 3p^3$ estimate. For example, for a single link with 5% loss rate, applying the real-time

---

[2]In many cases, the loss rate probability may not be uniform. Later in the paper, we investigate the impact of burstiness on our protocols.
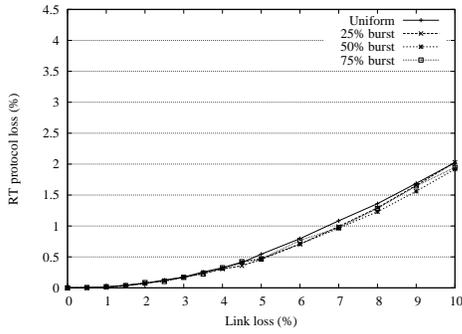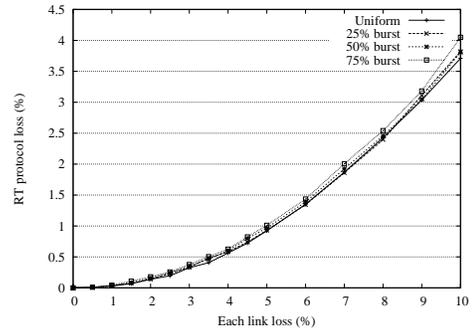
Figure 2: Real-time recovery loss - 1 link

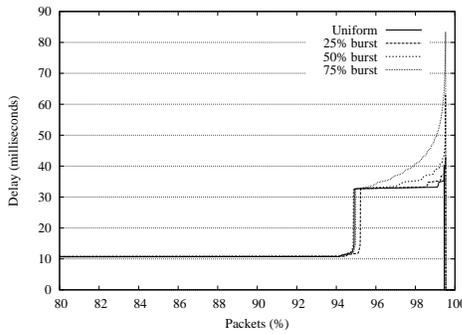**Figure 3: Real-time loss recovery - 2 concatenated links**

Figure 4: Delay distribution - 1 link, 5% loss
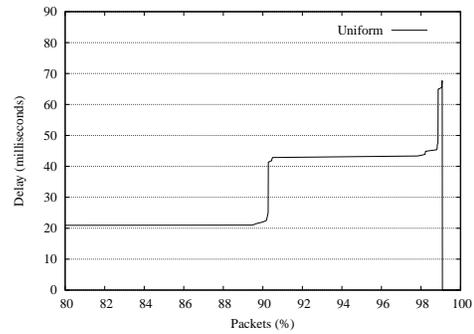

Figure 5: Delay distribution - 2 concatenated links, 5% loss each

protocol reduces the loss rate by a factor of 10, to about 0.5%, which yields an acceptable PESQ score.

For the single 10 msec link experiment with 5% loss rate, the packet delay distribution is presented in Figure 4. As expected, 95% of the packets arrive at the destination in about 10 milliseconds. Most of the losses are recovered, showing a total latency of 30 msec plus an additional delay due to the inter-arrival time of the packets required for the receiver to detect the loss, and about 0.5% of the packets are not recovered. In the case of uniform loss probability the delay of the recovered packets is almost constant. However, when the link experiences loss bursts, multiple packets are likely to be lost in a row, and therefore it takes longer for the receiver to detect the loss. The increase of the interval Δ results in a higher delay for the recovered packets. Obviously, the higher the burstiness, the higher the probability for consecutive losses. Figure 5 shows the delay distribution for the two-link network, where both links experience 5% uniform distribution loss rate. As in the single link experiment, most of the losses are recovered, with the exception of 1% of the packets. We notice, however, a small fraction of packets (slightly less than 0.25%) that are lost and recovered on both links, and that arrive with a latency of about 66 msec. This was expected to happen with the compound probability of loss on each link, $p_c = 0.05 \cdot 0.05$. Burstiness results for the two-link network, not shown in the figure, follow the same pattern as shown in Figure 4.

In order to evaluate the effect of local recovery on voice quality we ran the same experiment depicted in Figure 1 on top of a Spines overlay network. We divided the 50 msec network into 5 concatenated 10 msec links as shown in Figure 6, ran Spines with the real-time protocol on each link, and sent 10 VoIP streams in parallel from node A to node F. We generated losses with different
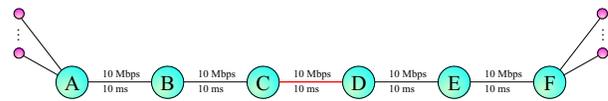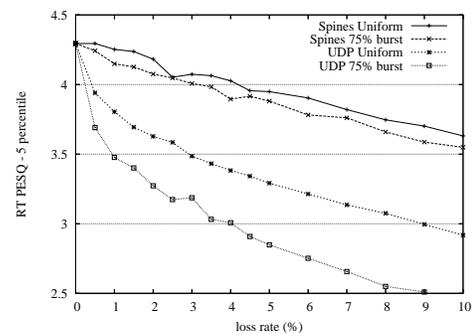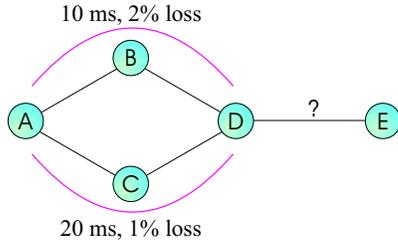

Figure 6: Spines network - 5 links

Figure 7: Real-Time protocol - 5 percentile PESQ

10 ms, 2% loss

A — B — D — ? — E

20 ms, 1% loss

Figure 8: Two-metric routing decision

levels of burstiness on the middle link $C - D$ and set the threshold network latency for the G.711 codec to be 100 msec. Figure 7 presents the lower 5 percentile PESQ scores of the G.711 streams using Spines, and contrasts them with the results obtained when sending over UDP directly. Since most of the packets are received in time to be decoded at the receiver, we can see that when using Spines, regardless of burstiness, the G.711 codec can handle up to 3.5% losses with PSTN quality.

4.2 Real time routing for voice

Our real time protocol recovers most of the missed packets in case of occasional, or even sustained periods of high loss, but if the problem persists, we would like to adjust the overlay routing to avoid problematic network paths.

Given the packet delay distribution and the loss rate of the soft real-time protocol on each overlay link, the problem is how to find which overlay path delivers the maximimium number of packets within the delay budget, so that the voice codec can play them. The problem is not trivial, and deals with a two metric routing optimizer. For example, in Figure 8, assuming a maximum delay threshold for the voice codec to be 100 msec, if we try to find the best path from node A to node E using an incremental algorithm, even in the simple case where we do not recover packets, we cannot determine which partial path from node A to node D is better (maximizes the number of packets arriving at E within 100 msec) without knowing the latency of the link D-E. On the other hand, computing all possible paths with their delay distribution and choosing the best one is prohibitively expensive.

However, if we can approximate the cost of each link by a metric dependent on the link's latency and loss rate, taking into account the characteristics of our real-time protocol and the requirements of VoIP, we can use this metric in a regular shortest path algorithm with reasonable performance results. Our approach is to consider that packets lost on a link actually arrive, but with a delay T_{max} bigger than the threshold of the voice codec, so that they will be discarded at the receiver. Then, the packet delay distribution of a link will be a three step function defined by the percentage of packets that are not lost (arriving in time T), the percentage of packets that are lost and recovered (arriving in $3T + \Delta$), and the percentage of packets missed by the real-time protocol (considered to arrive after T_{max}). The area below the distribution curve represents the expected delay of the packets on that link, given by the formula: $T_{exp} = (1-p) \cdot T + (p - 2p^2 + 3p^3) \cdot (3T + \Delta) + (2p^2 - 3p^3) \cdot T_{max}$. Since latency is additive, for a path consisting of several links, our approximation for the total expected delay will then be the sum of the expected delay of each individual link. We call this metric *expected latency* cost function.

We evaluated the performance of the *expected latency* based routing and compared it with other cost metrics. We used the BRITE [14] topology generator to create random topologies using the Waxman

model, where the probability to create a link depends on the distance between the nodes. We chose this model because it generates mostly short links that fit our goal for localized recovery. We assigned random loss from 0% to 5% on half of the links of each topology, selected randomly. We considered every node generated by BRITE to be an overlay node, and every link to be an overlay edge. For each topology we determined the nodes defining the diameter of the network (the two nodes for which the shortest latency path is longest), and determined the routing path between them given by different cost metrics.

We experimented with networks of different size, generating 1000 different topologies for each size. In each case, we selected nodes that are furthest apart in the network and evaluated the percentage of packets delivered when running the real-time protocol on the links of the network, using different routing metrics and then choosing the shortest path in that metric. Figure 9 shows the average delivery ratio for network topologies with 15 nodes and 30 links, and Figure 10 shows the delivery ratio for network topologies with 100 nodes and 200 links. For a link with direct latency T and loss rate p, considering an voice codec threshold $T_{max} = 100$ msec and the packet inter-arrival time $\Delta = 2$ msec, the cost metrics used are computed as follows:

- Expected latency: $Cost = (1 - p) \cdot T + (p - 2p^2 + 3p^3) \cdot (3T + \Delta) + (2p^2 - 3p^3) \cdot T_{max}$

- Hop distance: $Cost = 1$

- Link latency: $Cost = T$

- Loss rate: $Cost = -log(1 - p)$

- Greedy optimizer: We used a modified Dijkstra algorithm that, at each iteration, computes the delay distribution of the selected partial paths and chooses the one with the maximum delivery ratio.

- Best route: All the possible paths and their delay distributions were computed, and out of these the best one was selected. Obviously, this operation is very expensive, mainly because of the memory limitation of storing all combinations of delay distributions . Using a computer with 2GB memory we could not compute the best route for networks with more than 16 nodes.

As expected, for small diameter networks the loss-based routing achieves very good results, as the delay of the links is less relevant. With the increase in the network diameter, the latency-based routing achieves better results. At high latencies, the packet recovery becomes less important than the risk of choosing a highly delayed path, with latency more than the codec threshold. Interestingly, the greedy optimizer fails at high latency networks, mainly due to wrong routing decisions taken early in the incremental algorithm. The expected latency routing achieves slightly lower delivery ratio than the loss-based routing for small diameter networks, but behaves consistently better than the latency-based routing, even in high latency networks. The slight drop in delivery ratio for low diameter networks is causing just a small change in VoIP quality, while the robustness at high network delays delivers improved performance exactly where we need it the most.

5. RELATED WORK

Multi Protocol Label Switching (MPLS) [15] has been recently proposed as a way to improve the performance of underlying networks. This is done by pre-allocating resources across Internet
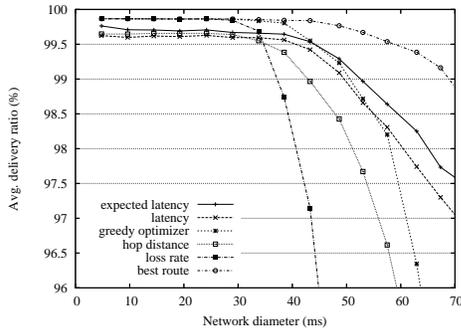
Figure 9: Comparing routing metrics - 15 node networks

Figure 10: Comparing routing metrics - 100 node networks

paths (LSPs in MPLS parlance) and forwarding packets across these paths. Our system is network agnostic and therefore does not depend on MPLS, but it can leverage any reduction in loss rate offered by MPLS. At the same time, MPLS will not eliminate route and link failures or packet loss. Since it runs at a higher level, our overlay network can continue to forward packets avoiding failed network paths. Forward Error Correction (FEC) schemes [16] have also been proposed as a method of reducing the effective loss rate of lossy links. These schemes work by adding redundant information and sending it along with the original data, based on the feedback estimate of loss rate given by RTCP, such that in case of a loss, the original information (or part of it) can be recreated. Most of the VoIP solutions today (including the G.711 codec we use in this paper) use some form of FEC to ameliorate the effect of loss. Given the occasional bursty loss pattern of the Internet, many times the FEC mechanisms are slow in estimating the current loss rate, and therefore we believe that localized retransmissions are required for maintaining voice quality.

Overlay networks have emerged as an increasingly growing field over the last few years, motivated mainly by the need to implement new services not supported by the current Internet infrastructure. Some of the pioneers of overlay network systems are X-Bone [17] and RON [18], which provides robust routing around Internet path failures. Other overlay networks focus on multicast and multimedia conferencing [19],[20]. Our work uses the same basic architecture of an overlay network but it is optimized to meet the specific requirements of VoIP traffic. Finally, OverQoS [21] is probably closest to our work, as it proposes an overlay link protocol that uses both retransmissions and FEC to provide loss and throughput guarantees. OverQoS does not provide a routing metric or path selection, and it depends on the existence of an overlay system like Spines (the authors suggest RON as an option) to provide overlay forwarding.

6. CONCLUSION

In this paper we have shown that current conditions inhibit the deployment of PSTN quality VoIP, and we proposed a deployable solution that can overcome the bursty loss pattern of the Internet. Our solution uses an overlay network to segment end-to-end paths into shorter overlay hops and attempts to recover lost packets using limited hop-by-hop retransmissions. We presented an adaptive overlay routing algorithm that avoids chronically lossy paths in favor of paths that will deliver the maximum number of voice packets within the predefined time budget. Our results show that the proposed mechanisms combined can be very effective in masking the effects of packet loss, thus offering high quality VoIP even at loss rates higher than those measured in the Internet today.
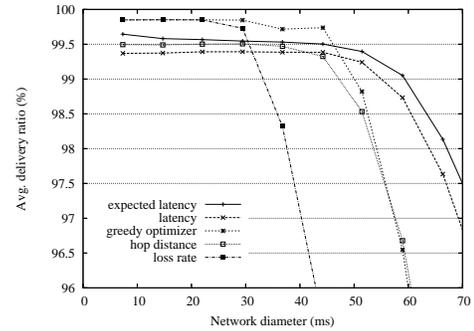
7. REFERENCES

[1] Athina Markopoulou, Fouad A. Tobagi, and Mansour J. Karam, "Assessing the quality of voice communication over internet backbones," *IEEE/ACM Transactions On Networking*, vol. 11, no. 5, pp. 747–760, October 2003.

[2] Vern Paxson, "End-to-End Packet Dynamics," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 277–292, 1999.

[3] David G. Andersen, Alex C. Snoeren, and Hari Balakrishnan, "Best-Path vs. Multi-Path Overlay Routing," in *Proceedings of IMC 2003*, Oct. 2003.

[4] "ITU-T Recommendation G.711: Pulse code modulation (PCM) of voice frequencies," http://www.itu.int/rec/recommendation. asp?type=items&lang=E&parent=T-REC-G.711-198811-I.

[5] "ITU-T Recommendation G.711 appendix I: A high quality low-complexity algorithm for packet loss concealment with G.711," http://www.itu.int/rec/recommendation. asp?type=items&lang=E&parent=T-REC-G.711-199909-I!AppI.

[6] "ITU-T Recommendation P.862: Perceptual evaluation of speech quality (PESQ)," http://www.itu.int/rec/recommendation. asp?type=items&lang=e&parent=T-REC-P.862-200102-I.

[7] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker, "On the Constancy of Internet Path Properties," in *Proceedings ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.

[8] C. Labovitz, C. Malan, and F. Jahanian, "Internet Routing Instability," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 515–526, 1998.

[9] B. Chandra, M. Dahlin, L. Gao, and A. Nayate, "End-to-End WAN Service Availability," in *Proceedings of 3rd USISTS*, Mar. 2001.

[10] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*, Boston, MA, Dec. 2002, USENIX Association, pp. 255–270.

[11] "The Spines Overlay Network," http://www.spines.org.

[12] Yair Amir and Claudiu Danilov, "Reliable communication in overlay networks," in *Proceedings of the IEEE DSN 2003*, June 2003, pp. 511–520.

[13] Gianluca Iannaccone, Sharad Jaiswal, and Christophe Diot, "Packet reordering inside the Sprint backbone," Tech. Rep. TR01-ATL-062917, Sprintlab, June 2001.

[14] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers, "BRITE: An approach to universal topology generation," in *International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems - MASCOTS '01*, August 2001.

[15] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," *RFC 3031*, Jan 2001.

[16] Jean-Chrysostome Bolot, Sacha Fosse-Parisis, and Donald F. Towsley, "Adaptive FEC-based error control for internet telephony," in *INFOCOM (3)*, 1999, pp. 1453–1460.

[17] J. Touch and S. Hotz, "The x-bone," in *Third Global Internet Mini-Conference at Globecom '98*, Nov. 1998.

[18] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. of the 18th Symposium on Operating Systems Principles*, Oct. 2001, pp. 131–145.

[19] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. of ACM SIGCOMM*, 2002.

[20] Yang hua Chu, Sanjay G. Rao, Srinivasan Seshan, and Hui Zhang, "Enabling conferencing applications on the internet using an overlay multicast architecture," in *ACM SIGCOMM 2001*. ACM, Aug. 2001.

[21] Lakshminarayanan Subramanian, Ion Stoica, Hari Balakrishnan, and Randy Katz, "OverQoS: An Overlay Based Architecture for Enhancing Internet QoS," in *USENIX NSDI '04*, Mar. 2004.