# Resilient Multicast using Overlays

Suman Banerjee, Seungjoon Lee, Bobby Bhattacharjee, Aravind Srinivasan

*Abstract*— We introduce PRM (Probabilistic Resilient Multicast): a multicast data recovery scheme that improves data delivery ratios while maintaining low end-to-end latencies. PRM has both a proactive and a reactive components; in this paper we describe how PRM can be used to improve the performance of application-layer multicast protocols, especially when there are high packet losses and host failures. Through detailed analysis in this paper, we show that this loss recovery technique has efficient scaling properties — the overheads at each overlay node asymptotically decrease to zero with increasing group sizes.

As a detailed case study, we show how PRM can be applied to the NICE application-layer multicast protocol. We present detailed simulations of the PRM-enhanced NICE protocol for 10,000 node Internet-like topologies. Simulations show that PRM achieves a high delivery ratio ($> 97\%$) with a low latency bound (600 ms) for environments with high end-to-end network losses (1-5%) and high topology change rates (5 changes per second) while incurring very low overheads ($< 5\%$).

## I. INTRODUCTION

We present a fast multicast data recovery scheme that achieves high delivery ratios with low overheads. Our technique, called Probabilistic Resilient Multicast (PRM), is especially useful for applications that can benefit from low data losses without requiring perfect reliability. Examples of such applications are real-time audio and video streaming applications where the playback quality at the receivers improves if the delivery ratios can be increased within specific latency bounds. Using terminology defined in prior literature [26] we call this model of data delivery *resilient multicast*.

In this paper we describe PRM in the context of overlay-based multicast [8], [10], [1], [31], [7], [25],

S. Banerjee is with the Department of Computer Sciences, University of Wisconsin, Madison, WI 53706 USA. S. Lee, B. Bhattacharjee, and A. Srinivasan are with the Department of Computer Science, University of Maryland, College Park, MD 20742, USA. B. Bhattacharjee and A. Srinivasan are also with the Institute for Advanced Computer Studies, University of Maryland.

[16]. Unlike native multicast where data packets are replicated at routers inside the network, in application-layer multicast data packets are replicated at end hosts. Logically, the end-hosts form an overlay network, and the goal of application-layer multicast is to construct and maintain an efficient overlay for data transmission. The eventual data delivery path in application-layer multicast is an overlay tree. While network-layer multicast makes the most efficient use of network resources, its limited deployment in the Internet makes application-layer multicast a more viable choice for group communication over the wide-area Internet.

A key challenge in constructing a resilient application-layer multicast protocol is to provide fast data recovery when overlay node failures partition the data delivery paths. Overlay nodes are processes on regular end-hosts which are potentially more susceptible to failures than the routers. Each such failure of a non-leaf overlay node causes a data outage for nodes downstream until the time the data delivery tree is reconstructed. Losses due to overlay node failures are more significant than regular packet losses in the network and may cause data outage in the order of tens of seconds (e.g. the Narada application-layer multicast protocol [8] sets default time-outs between 30-60 seconds).

PRM uses two simple techniques:

- A proactive component called *Randomized forwarding* in which each overlay node chooses a constant number (e.g., 1 or 2) of other overlay nodes uniformly at random and forwards data to each of them with a low probability (e.g. 0.01-0.03). This randomized forwarding technique operates in conjunction with the usual data forwarding mechanisms along the tree edges, and may lead to a small number of duplicate packet deliveries. Such duplicates are detected and suppressed using sequence numbers. The randomized component incurs very low additional overheads and can guarantee high delivery ratios even under high rates of overlay node failures.

- A reactive mechanism called *Triggered NAKs* to handle data losses due to link errors and network congestion.

Through analysis and detailed simulations we show

that these relatively simple techniques provide high resilience guarantees, both in theory and practice. PRM can be used to significantly augment the data delivery ratios of *any* application-layer multicast protocol (e.g. Narada [8], Yoid [10], NICE [1], HMTP [31], Scribe [7], Delaunay Triangulation-based [16], CAN-multicast [25]) while maintaining low latency bounds.

The contributions of this paper are three-fold:

- We propose a simple, low-overhead scheme for resilient multicast. To the best of our knowledge, this work is the first proposed resilient multicast scheme that can be used to augment the performance of application-layer multicast protocols.
- We present a full analysis of PRM and derive the necessary and sufficient conditions for PRM which will allow the control overheads at the group members to asymptotically decrease to zero with increasing group sizes.
- We demonstrate how our proposed scheme can be used with an existing application-layer multicast protocol (NICE [1]) to provide a low overhead, low latency and high delivery ratio multicast technique for realistic applications and scenarios.

The rest of this paper is structured as follows. In the next section we present the details of the PRM scheme and analyze its performance in Section III. In Section IV we present detailed simulation studies of the the PRM-enhanced NICE protocol. In Section V we describe related work and conclude in Section VI. The Appendix gives a brief sketch of the proofs.

## II. Probabilistic Resilient Multicast (PRM)

The PRM scheme employs two mechanisms to provide resilience. We describe each of them in turn.

### A. Randomized Forwarding

In randomized forwarding, each overlay node, with a small probability, proactively sends a few extra transmissions along randomly chosen overlay edges. Such a construction interconnects the data delivery tree with some cross edges and is responsible for fast data recovery in PRM under high failure rates of overlay nodes. Existing approaches for resilient and reliable multicast use either reactive retransmissions (e.g. RMTP [24], STORM [26] Lorax [18]) or proactive error correction codes (e.g. Digital Fountain [4]) and can only recover from packet losses on the overlay links. Therefore the proactive randomized forwarding is a key difference between our approach and other well-known existing approaches.

We explain the specific details of proactive randomized forwarding using the example shown in Figure 1. In the original data delivery tree (Panel 0), each overlay node forwards data to its children along its tree edges. However, due to network losses on overlay links (e.g. $\langle A, D \rangle$ and $\langle B, F \rangle$) or failure of overlay nodes (e.g. $C$, $L$ and $Q$) a subset of existing overlay nodes do not receive the packet (e.g. $D, F, G, H, J, K$ and $M$). We remedy this as follows. When any overlay node receives the first copy of a data packet, it forwards the data along all other tree edges (Panel 1). It also chooses a small number ($r$) of other overlay nodes and forwards data to each of them with a small probability, $\beta$. For example node $E$ chooses to forward data to two other nodes using cross edges $F$ and $M$.

Note that as a consequence of these additional edges some nodes may receive multiple copies of the same packet (e.g. node $T$ in Panel 1 receives the data along the tree edge $\langle B, T \rangle$ and cross edge $\langle P, T \rangle$). Therefore each overlay node needs to detect and suppress such duplicate packets. Each overlay node maintains a small duplicate suppression cache, which temporarily stores the set of data packets received over a small time window. Data packets that miss the latency deadline are dropped. Hence the size of the cache is limited by the latency deadline desired by the application. In practice, the duplicate suppression cache can be implemented using the playback buffer already maintained by streaming media applications.

It is easy to see that each node on average sends or receives up to $1 + \beta r$ copies of the same packet. The overhead of this scheme is $\beta r$, where we choose $\beta$ to be a small value (e.g. 0.01) and $r$ to be between 1 and 3. In our analysis we show that if the destinations of these cross edges are chosen uniformly at random, it is possible to guarantee successful reception of packets at each overlay node with a high probability.

Each overlay node periodically discovers a set of random other nodes on the overlay and evaluates the number of losses that it shares with these random nodes. In an overlay construction protocol like Narada [8], each node maintains state information about all other nodes. Therefore, no additional discovery of nodes is necessary in this case. For some other protocols like Yoid [10] and NICE [1] overlay nodes maintain information of only a small subset of other nodes in the topology. Therefore we implement a node discovery mechanism, using a random-walk on the overlay tree. A similar technique has been used in Yoid [10] to discover random overlay group members. The discovering node transmits a *Discover* message with a *time-to-live* (TTL) field to its parent on the tree. The message is randomly forwarded from
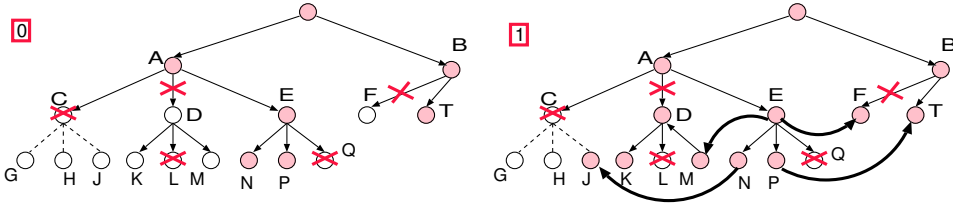
Fig. 1. The basic idea of the PRM scheme. The circles represent the overlay nodes. The crosses indicate link and node failures. The arrows indicate the direction of data flow. The curved edges indicate the chosen cross overlay links for randomized forwarding of data.
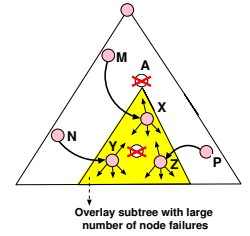


Fig. 2. Successful delivery with high probability even under high node failure rate.

neighbor to neighbor, without re-tracing its path along the tree and the TTL field is decremented at each hop. The node at which the TTL reaches zero is chosen as the random node.

*Why is Randomized Forwarding effective?* It is interesting to observe why such a simple, low-overhead randomized forwarding technique is able to increase packet delivery ratios with a high probability, especially when many overlay nodes fail. Consider the example shown in Figure 2, where a large fraction of the nodes have failed in the shaded region. In particular, the root of the sub-tree, node $A$, has also failed. So if no forwarding is performed along cross edges, the entire shaded sub-tree is partitioned from the data delivery tree. No overlay node in this entire sub-tree would get data packets till the partition is repaired. However, using randomized forwarding along cross edges, a number of nodes from the unshaded region will have random edges into the shaded region as shown ($\langle M, X \rangle, \langle N, Y \rangle$ and $\langle P, Z \rangle$). The overlay nodes that receive data along such randomly chosen cross edges will subsequently forward data along regular tree edges and any chosen random edges. Since the cross edges are chosen uniformly at random, a large subtree will have a higher probability of cross edges being incident on it. Thus as the size of a partition increases, so does its chance of repair using cross edges.

### B. Triggered NAKs

This is the reactive component of PRM. We assume that the application source identifies each data unit using monotonically increasing sequence numbers. An overlay node can detect missing data using gaps in the sequence numbers. This information is used to trigger NAK-based retransmissions. This technique has been applied for loss repair in RMTP [24].

In our implementation each overlay node, $x$, piggybacks a bit-mask with each forwarded data packet indicating which of the prior sequence numbers it has correctly received. The recipient of the data packet, $y$, detects missing packets using the gaps in the received
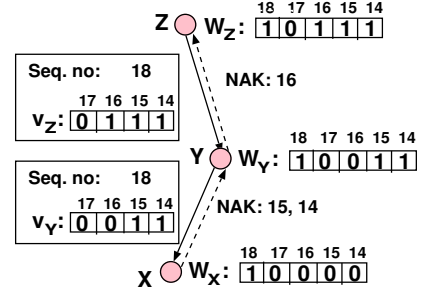


Fig. 3. Triggered NAKs to parent on the overlay tree when data unit with sequence number 18 is propagated along the overlay. The length of the bit-mask is 4.

sequence and sends NAKs to $x$ to request the appropriate retransmissions. Note that $x$ can either be a parent of $y$ in the data delivery tree, or a random node forwarding along a cross edge. We illustrate the use of triggered NAKs in Figure 3. Node $Y$ receives data with sequence number 18, which indicates that the parent $Z$ has data sequence numbers 14, 15, and 16. Since node $Y$ does not have sequence number 16, it sends a NAK for 16 to $Z$. Similarly, node $X$ sends a NAK to its parent $Y$ for 14 and 15. Note that $X$ does not have data with sequence number 16, but does not request retransmission from its parent. This is because the parent $Y$ does not currently have this data and has already made an appropriate retransmission request to its parent $Z$. On receiving this data, $Y$ will automatically forward it to $X$.

### C. Extensions

We describe two extensions to the PRM scheme that further improve the resilience of the data delivery.

*Loss Correlation:* This is a technique that can be used to improve the randomized forwarding component of PRM. As described in Section II-A each overlay node chooses a small number of cross edges completely at random for probabilistic data forwarding on the overlay. In practice, it is possible to increase the utility of these cross edges by choosing them more carefully. In
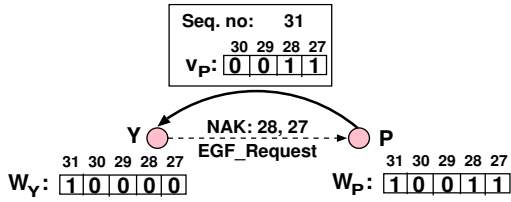
Fig. 4. Triggered NAKs to source of random forwarding for data with sequence number 31. The value of $\Delta$ for Ephemeral Guaranteed Forwarding is set to 3.

particular if $S$ is the root (and source) of the overlay tree, we want to choose a cross edge between two overlay nodes $X$ and $Y$, if and only if the correlation between the packets lost on the overlay paths $\langle S \longmapsto X \rangle$ and $\langle S \longmapsto Y \rangle$ is low. Clearly if these two overlay paths share no underlying physical links, then we expect the losses experienced by $X$ and $Y$ to be uncorrelated. However, such a condition is difficult to guarantee for any overlay protocol. Therefore under the loss correlation extension, we let each node $X$ to choose a random edge destination, $Y$, with which has the minimum number of common losses over a limited time window.

*Ephemeral Guaranteed Forwarding (EGF):* This is an extension to the triggered NAK component and is also useful in increasing the data delivery ratio. Consider the case when node $Y$ receives a data packet with sequence number $j$ along a random edge from node $P$. If on receiving the data packet with sequence number $j$, $Y$ detects a large gap (greater than a threshold $\Delta$) in the sequence number space it is likely that the parent of $Y$ has failed. In this case, $Y$ can request $P$ to increase the random forwarding probability, $\beta$, for the edge $\langle P, Y \rangle$ to one for a short duration of time. To do this $Y$ sends an *EGF_Request* message to $P$. Note that the EGF state is soft and expires within a time period $T_{EGF}$. This is shown by an example in Figure 4. Node $Y$ receives data with sequence number 31 along a random edge from $P$. $Y$ immediately requests retransmissions for data with sequence numbers 28 and 27. Since $\Delta = 3$, $Y$ also sends the *EGF_Request* message to $P$. If the tree path of $Y$ is repaired before the EGF period expires, $Y$ can also send an *EGF_Cancel* message to $P$ to terminate this state.

The EGF mechanism is useful for providing uninterrupted data service when the overlay construction protocol is detecting and repairing a partition in the data delivery tree. In fact, putting such a mechanism in place allows the overlay nodes to use larger timeouts to detect failure of overlay peers. This, in turn, reduces the control overheads of the application-layer multicast protocol. In practice, the EGF mechanism can sometimes

be overly aggressive and cause false positives leading to a higher amount of data duplication on the data delivery tree. Thus, the improvement in performance is at the cost of additional state, complexity and packet overhead at nodes. Depending on the reliability requirements, applications may choose to enable or disable EGF.

## III. EVALUATION OF PRM

A key component of the PRM scheme is the randomized forwarding technique which achieves high delivery ratios in spite of a large number of overlay node/link failures. In this section we present our analysis of this scheme.

Recall from Section II that the per-node overhead of PRM is $\beta r$. We will now primarily be concerned with the case of low overhead; in particular, the case where the overhead is (much) smaller than 1. Thus, up to Section III-A, we will consider the case where $r = 1$: here, each node does random forwarding to just one node, with a probability of $p = \beta$. However, all of our results will also hold for $r$ being arbitrary. In this paper, we prove that even if the probability of random forwarding $p$ is made an *arbitrarily small* positive constant (i.e., even if the data overhead $\beta r = pr = p$ is made negligible), the scheme can be designed so that almost all surviving overlay nodes get the data with high probability. In particular, the system scales: as the number $n$ of nodes increases, our probability of successful data delivery tends to 1.

We start with some notation and assumptions.

(A1) All nodes at the same level of the tree have the same number of children. The total number of nodes is denoted by $n$.

(A2) There are parameters $\epsilon$ and $\delta$ such that the probability of any given node failing is at most $\epsilon$, and the probability of any given link failing is at most $\delta$. We only require that $\epsilon$ and $\delta$ be bounded away from 1: e.g., we could have $\epsilon, \delta \leq 0.5$. (Indeed, a multicast tree composed of elements that may fail with more than 50% probability is in effect useless; in practice, we expect $\epsilon$ and $\delta$ to be close to zero.) The failure events are all independent.

We next present a theorem that deals with the asymptotic regime where $n$ is large. We then discuss a "tree augmentation" technique and general optimality of our results in Section III-A. We complement these in Section III-B with simulation results for the "non-asymptotic" regime of size 10,000.

*Theorem 3.1: Let the probability of random forwarding $p$ be an arbitrary positive constant (i.e., it can be*

5

*arbitrarily small).* Then, *there is a constant* $C > 0$ *such that the following holds. Suppose every non-leaf node has at least* $C \log n$ *children. Then, with probability tending to* 1 *as* $n$ *increases, the following two claims hold simultaneously:*

(i)  *All the non-leaf nodes that did not fail successfully get the data.*

(ii)  *At least an* $(1-\epsilon)\cdot(1-\delta)\cdot(1-g(n))$ *fraction of the leaf nodes that did not fail, successfully get the data; here* $g(n)$ *denotes a negligibly small quantity, which tends to* 0 *as* $n$ *increases.*

*Proof:*  See Appendix. ∎

Theorem 3.1 shows that as long as the node-degrees are at least logarithmic in $n$, the tree is highly resilient to node- and link-failures, even with arbitrarily low data overhead. To take a realistic example, we consider the case when the end-to-end losses on overlay links is 2%, and simultaneous number of overlay nodes failures is 1% of the group (for group of size 10,000 this translates to 100 simultaneous failures). Theorem 3.1 states that with a high probability all surviving non-leaf overlay nodes and at least 97% of the surviving leaf nodes continue to receive data packets using the existing overlay data paths and random edges.

It is, in fact, possible to increase the delivery to leaf-nodes to to increase the delivery to leaf-nodes to an arbitrarily high value using a "tree augmentation" scheme described next.

### A. Tree-augmentation extensions to PRM

If we desire, for an arbitrary given (small) constant $\phi$, that at least a $(1-\phi)$–fraction of the surviving leaves get the data with high probability (in addition to item (i) of Theorem 3.1), then it suffices to augment the tree as follows: each leaf connects to $1 + \lfloor \frac{\log(1/\phi)}{\log(1/\epsilon)} \rfloor$ randomly chosen non-leaf nodes, and gets the data from any one of them that has received the packet. For example, for the 1% node failure rate case, the tree augmentation scheme will guarantee that 99.98% of all nodes (including leaf nodes) will successfully get the data packets with an overhead bounded by the constant two.

In fact, if we require a $(1-\phi)$–fraction of the surviving leaves to get the data with high probability, this amount of overhead (i.e., $1 + \lfloor \frac{\log(1/\phi)}{\log(1/\epsilon)} \rfloor$) is necessary for *any* protocol. In Appendix we show that our protocol's lower bounds on the fraction of surviving leaves getting the data—both with and without the random augmentation for leaves—are optimal. Also in Appendix we show that the logarithmic degree-requirement of Theorem 3.1 is both necessary and sufficient if we desire a low overhead. Specifically, if the degrees, e.g., of the parents of the
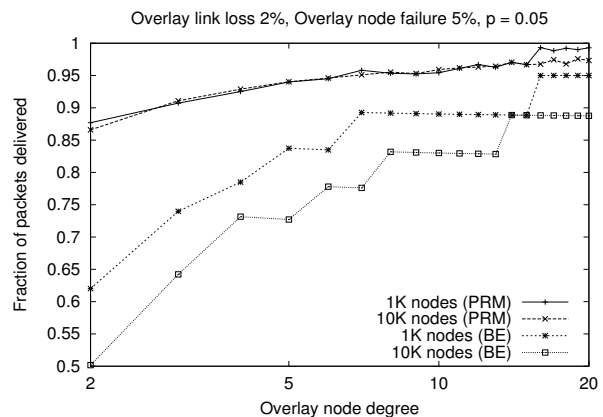


Fig. 5.  Variation of data delivery ratio with overlay node degree. BE stands for "Best Effort."

leaves are only some small constant times $\log n$, then in fact a large number of the leaves' parents will fail to get the data with probability tending to 1 as $n$ increases.

### B. Expected Scaling Behavior for Finite Groups

In the analysis of the PRM scheme (described above), we have explored its asymptotic behavior with corresponding increase in the group size. Through simulations we now show that PRM is expected to perform very well in practice even for groups of moderate size. In these idealized simulations, we assume that there exists some application-layer multicast protocol which constructs and maintains a data delivery tree. When data packets are sent on this tree, a random subset of the overlay nodes fail simultaneously. The failed subset is chosen independently for each data packet sent. Additionally, data packets also experience network layer losses on overlay links. Consider a regular tree, where all non-leaf nodes have the same degree. From the analysis we can intuitively expect that as the degree of the tree increases, so does the data delivery ratio.

In Figure 5 we illustrate how the data delivery ratio for the non-leaf nodes of an overlay tree improves with increase in degree. We considered two different tree sizes – 1000 nodes and 10,000 nodes. In this example, we assume that for each overlay link experiences a loss rate of 2% and the node failure rate is 5% (which implies 50 simultaneous failures for a 1000-node tree and 500 simultaneous failures on a 10,000-node tree). Such failure rates are very high by the usual Internet standards. The randomized forwarding probability $p$ is chosen to be 0.05 (i.e. the data overhead is 5%). We can see that even under such adverse conditions, the randomized forwarding technique achieves data delivery ratio of about 93% even with a tree degree of 5; the

delivery ratio exceeds $95\%$ when the degree is made 10, quickly approaching 1 as the degree is increased further. The results for the leaf nodes in practice, are close to those of the non-leaf nodes.

## IV. Simulation Experiments

The PRM scheme can be applied to any overlay-based multicast data delivery protocol. In our detailed simulation study we implemented PRM over the NICE application-layer multicast protocol [1]. We used NICE because of three reasons: (1) the authors in [1] show that the NICE application-layer multicast protocol achieves good delivery ratios for a best-effort scheme; (2) NICE is a scalable protocol and therefore we could perform detailed packet-level simulations for large overlay topologies; (3) the source-code for NICE is publicly available.

We have studied the performance of the PRM-enhanced NICE protocol using detailed simulations. We compare the following four schemes, all of which use NICE as the underlying multicast protocol:

**BE:** This is the best-effort multicast scheme and has no reliability mechanisms and therefore serves as a baseline for comparison.

**HHR:** This is a reliable multicast scheme, where each overlay link employs a hop-by-hop reliability mechanism using negative acknowledgments (NAKs). A downstream overlay node sends NAKs to its immediate upstream overlay node indicating packet losses. On receiving these NAKs the upstream overlay node retransmits the packet. This scheme, therefore, hides all packet losses on the overlay links.

**FEC-$(d, r)$:** This is another enhanced version of the NICE protocol in which the source uses a forward error correction mechanism to recover from packet losses. In this scheme, the source takes a set of $d$ data packets and encodes them into a set of $d + r$ packets and sends this encoded data stream to the multicast group. A receiving member can recover the $d$ data packets if it receives *any* $d$ of the $r + d$ encoded packets [1]. Additional data overheads of this scheme are $r/d$.

**PRM-$(r, \beta)$:** This is our proposed PRM enhancements implemented on the basic NICE application-layer multicast protocol. The meanings of $r$ and $\beta$ are the same as in Section II. For all these experiments we implemented the loss correlation extensions to PRM. We enable EGF for only one specific experiment (described later) due to its higher overheads. Our results will show that EGF is

---

[1]The Digital Fountain technique [4], uses Tornado codes that require the receiver to correctly receive $(1 + \epsilon)d$ packets to recover the $d$ data packets, where $\epsilon$ is a small constant.

useful for very dynamic scenarios, at the cost of higher data overheads.

*Choosing FEC parameters:* Since the FEC-based schemes need to send $d + r$ packets instead of $d$ packets we use a higher data rate at the source (i.e. a data rate of $(d+r)/d$ times the data rate used by the other schemes). The resilience of an FEC-based scheme can be increased by increasing the overheads parameter, $r$. Also for the same amount of additional data overheads, resilience against network losses of the FEC-based schemes improve if we choose higher values of $d$ and $r$. For example, FEC-(128,128) has better data recovery performance than FEC-(16,16) even though both have $100\%$ overhead. This improved reliability comes at the cost of increased delivery latencies. Therefore, the maximum value of $d$ and $r$ depends on the latency deadline. We have experimented with a range of such choices up to the maximum possible value that will allow correct reception at receivers within the latency deadline. However, we observed that in presence of failures of overlay nodes increasing $d$ and $r$ does not always improve the resilience properties. This is because choosing higher values of $d$ and $r$ leads to increased latencies in data recovery. However when the group change rate is high the data delivery paths break before the FEC-based recovery can complete, and the achieved data delivery ratio is low.

### A. Simulation Scenarios

In all these experiments we model the scenario of a source node multicasting streaming media to a group. The source sends CBR traffic at the rate of 16 packets per second. For all these experiments we chose a latency deadline of upto 8 seconds. As a consequence the size of the the packet buffer for NAK-based retransmissions is 128. The packet buffer will be larger for longer deadlines.

We performed detailed experiments with a wide-range of parameters to study different aspects of the PRM scheme. The network topologies were generated using the Transit-Stub graph model, using the GT-ITM topology generator [5]. All topologies in these simulations had $10,000$ routers with an average node degree between 3 and 4. End-hosts were attached to a set of routers, chosen uniformly at random, from among the stub-domain nodes. The number of such hosts in the multicast group were varied between 8 and 8192 for different experiments. Each physical link on the topology was modeled to have losses. Inter-domain links had 0.5-0.6% loss rates, while intra-domain links was about 0.1% loss rates. We also model bursty losses as follows: if a packet is lost on a physical link we increase the loss probability

for subsequent packets received within a short time window. The average propagation and queueing latency on each physical link was between 2-10 ms. In all our experiments we use a *HeartBeat* period of 5 seconds for NICE and its extensions as was described by the authors in [1].

We have simulated a wide-range of topologies, group sizes, member join-leave patterns, and protocol parameters. In the experiments, all departures of end-hosts from the multicast group were modeled as "ungraceful leaves." This is equivalent to a host failure, where the departing member is unable to send a *Leave* message to the group.

In the experiments reported in these section, we first let a set of end-hosts join the multicast group and stabilize into an appropriate multicast data delivery tree. Subsequently a traffic source end-host starts sending data group and end-hosts continuously join and leave the multicast group. The join and the leave rate for members are chosen to be equal so that the average size of the group remained nearly constant. The instants of group membership changes were drawn from an exponential distribution with a pre-defined mean, which varied between experiments. We studied the various data delivery properties of our proposed scheme over this dynamically changing phase of the experiment.

### B. Simulation Results

We have studied the three metrics of interest: data delivery ratio, delivery latency and data overheads. The data overheads in PRM are because of duplication due to randomized forwarding, and due to redundant encoding in FEC-based schemes. We also examine the additional control overheads due to NAKs, random member discovery etc.

*Delivery Ratio:* In Figure 6 we show the delivery ratio of the different schemes as the frequency of changes to group membership is varied. The average size of the group was 512. The average loss rate for physical links for this experiment was 0.5%, which corresponds to between 2-5% end-to-end losses on each overlay link.

We plot the data delivery ratios as the group change rate is varied between 0 and 10 changes per second. Note that even 5 changes per second implies that 512 (which is also the size of the group) membership changes happen in less than two minutes! While such a high change rate is drastic, it is not improbable for very large distribution groups in the Internet. The PRM scheme is able to recover from a vast majority of these losses through the use of randomized forwarding mechanism. The delivery ratio for PRM-(3,0.01) is > 97% for a group membership change rate of 5 per second and > 80% for a group
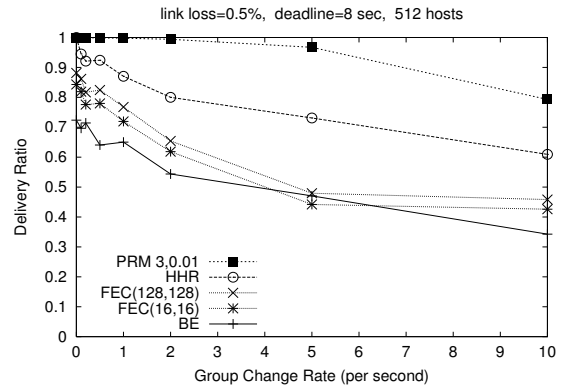


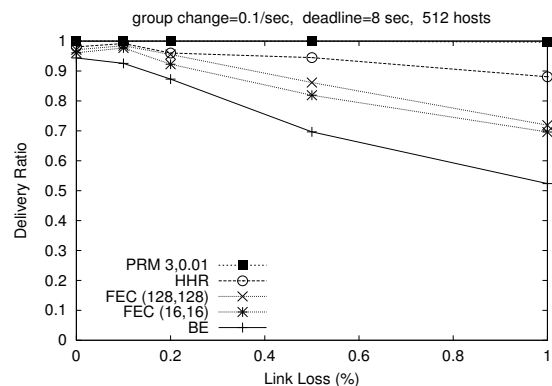Fig. 6. Delivery ratio with varying rate of changes to the group.



Fig. 7. Delivery ratio with varying network loss rates.

membership change rate of 10 per second. Additionally PRM incurs a very low (3%) additional data overhead.

The data delivery ratio for the best-effort protocol falls significantly (to about 0.35 for change rate of 10 group changes per second) with increase in the change rate to the overlay. In [1], the authors had shown that the NICE application-layer multicast protocol achieves good delivery ratios for a best-effort scheme, and is comparable to other best-effort application-layer multicast protocols, e.g. Narada [8]. Therefore, we believe that PRM-enhancements can significantly augment the data delivery ratios of all such protocols. An FEC-based scheme is typically able to recover from all network losses. However changes in the overlay data delivery path significantly impacts the performance of an FEC-based scheme. Note that the performance of the FEC-based scheme degrades with increasing frequency of group changes and even falls below the simple best-effort scheme for high group change rates.

In Figure 7 we compare the delivery ratio of the different schemes as the average packet loss rate on
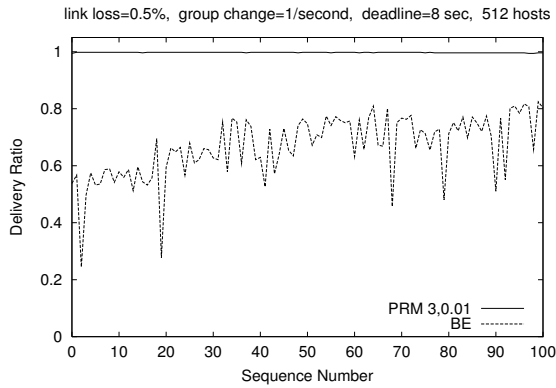
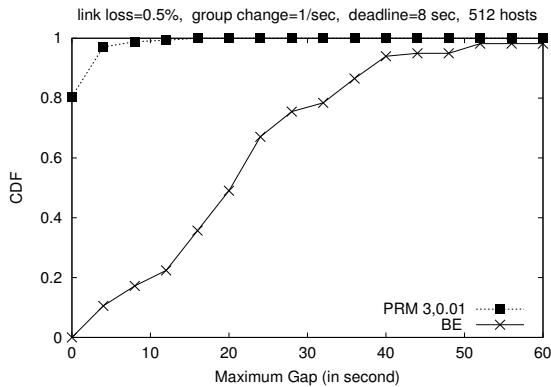Fig. 8. Time evolution of the delivery ratio for a sequence of data packets.



Fig. 10. Cumulative distribution of average latency for packets received at the different nodes.



Fig. 9. Cumulative distribution of the maximum time gap over which an overlay node lost all data packets.



Fig. 11. Delivery ratio achieved with varying deadline within which the data packets are to be successfully delivered.

the physical links of the topology are varied. In this experiment, we use a lower failure rate, and changes to the overlay topology (including both joins and leaves) occur with a mean of 0.1 change per second. In contrast to the other schemes, which suffer between 20% to 55% losses, the PRM-(3,0.01) scheme achieves near perfect data delivery under all data loss rates.

In Figure 8 we show how the delivery ratio achieved by the PRM scheme evolves over time in comparison to the best-effort protocol. In this experiment, the group change rate was one per second, i.e. the entire group can potentially change in less than 10 minutes. In the best-effort protocol, for any particular packet, 20–40% of the group members fail to receive it. In contrast, the losses experienced in the PRM scheme are minimal. For the same experiment, we plot the cumulative distribution of the maximum data outage period of the overlay nodes in Figure 9. Most overlay nodes had no significant data outage period in PRM and more than 98% of the overlay nodes had a maximum outage period less than 5 seconds.
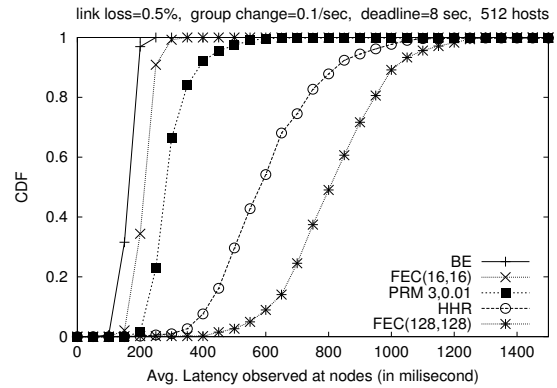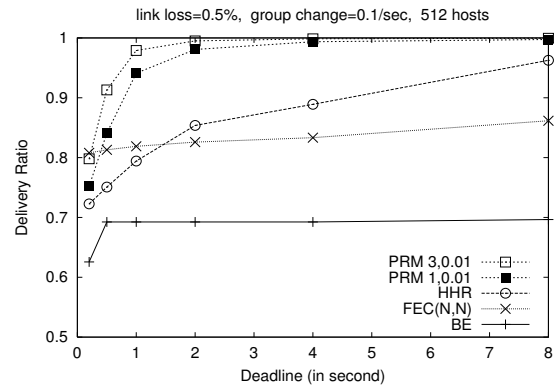
This is a significant improvement over the best-effort protocol where more than 20% of the overlay nodes experience a maximum data outage of more than 30 seconds.

*Delivery Latency:* In Figure 10 we show the distribution of latency experienced by the data packets at the different overlay nodes. In this experiment, the average group membership change rate was 0.1 per second and the average loss probability at the physical links was 0.5%. Note that the latency of data packets is the lowest for the best-effort NICE protocol. This is because the best-effort scheme incurs no additional delay due to timeout-based retransmissions or delivery using alternate longer overlay paths. FEC-(16,16) incurs a slightly higher latency. This is because, to recover a lost data packet in the FEC-based scheme has to wait for additional (encoded) packets to arrive. FEC-(128,128) can decode data only after sufficiently many packets are received, and as a result, it incurs highest latency. The HHR scheme also suffers from high latency since

it is purely reactive; in our simulations of HHR, packet losses due to node failure are often detected after the tree is recovered by underlying multicast protocol. The PRM scheme is a combination of proactive and reactive schemes and therefore incurs significantly lower latency than the HHR scheme. However data packets delivered using PRM still incur higher latency than the simple best-effort delivery. This is because many of the data packets that are lost on the shortest best-effort path are successfully delivered either using Triggered NAK-based retransmissions or randomized forwarding using a longer overlay path. More than 90% of the overlay nodes receive data packets within 500 ms, which is 2.5 times the worst case overlay latency on the topology.

In Figure 11 we show the effect of imposing a deadline for packet delivery. The deadline specifies a time upper bound within which a packet must reach an overlay node to be be useful to the application. For a deadline of $x$ seconds, we allow different overlay nodes to have slightly different additional slacks in the time upper bound within which packets must be delivered. This slack is to account for the minimum latency that data packets encounter on the shortest path from the source to that overlay node. The maximum slack for any overlay node was less than 200 ms. The best-effort NICE protocol makes a single attempt for packet delivery and therefore achieves almost identical delivery ratio for all deadlines. The performance of the HHR scheme increases gradually with increase in the deadline due to its dependence on timeouts. In contrast, for short deadlines, the PRM scheme achieves rapid improvements due to its proactive component and further gradual improvements for longer deadlines due to its reactive component. For the FEC-based scheme we used $d = r$ and chose the value of $d$ based on the deadline imposed. It achieved between 80-87% delivery ratios in these experiments.

*Additional Data Overheads:* In Table I, we compare the overheads of PRM and FEC-based schemes to achieve different delivery ratios. The table shows the additional data overheads for both schemes under different parameters (e.g. latency bounds, group change rate, etc.). The FEC-based schemes perform poorly when the frequency of changes on the overlay is high. Hence, we used an order of magnitude lower group change rates (0.1 changes/sec) for the FEC-based schemes than what we used for PRM (1 change/sec).

The table shows that PRM incurs very low additional data overheads to achieve relatively high delivery ratios within low latency bounds. For example for a group change rate of one per second and data latency bound of 0.5 seconds, the PRM scheme incurs 3-6% additional data overheads to achieve data delivery ratio of 90%. In

| scheme changes/sec, deadline (sec) | Delivery Ratio | | | | |
|---|---|---|---|---|---|
| | 80% | 85% | 90% | 95% | 99% |
| FEC, 0.1, 0.5 | 88-100 | - | - | - | - |
| FEC, 0.1, 2.0 | 62-75 | - | - | - | - |
| FEC, 0.1, 8.0 | 50-62 | 75-87 | - | - | - |
| FEC, 0.1, 64.0 | 37-50 | 50-62 | 75-87 | 87-100 | - |
| PRM, 1, 0.2 | 9-12 | 18-21 | 21-24 | 30-60 | - |
| PRM, 1, 0.5 | 0-1 | 1-3 | 3-6 | 9-15 | 30-60 |
| PRM, 1, 2.0 | 0-1 | 0-1 | 0-1 | 0-1 | 3-9 |
| PRM, 1, 8.0 | 0-1 | 0-1 | 0-1 | 0-1 | 1-3 |

TABLE I

COMPARISON OF ADDITIONAL DATA OVERHEADS (IN %) REQUIRED FOR PRM AND FEC-BASED SCHEMES TO MEET DIFFERENT DELIVERY RATIOS FOR SPECIFIC GROUP CHANGE RATES AND LATENCY BOUNDS. WE DO NOT REPORT RESULTS WHEN THE OVERHEADS EXCEED 100% (MARKED BY -).

| Group Size | Control Overheads (pkts/sec/node) | | Delivery ratio | |
|---|---|---|---|---|
| | BE | PRM | BE | PRM |
| 128 | 1.43 | 2.03 | 0.58 | 0.99 |
| 256 | 1.67 | 2.22 | 0.57 | 0.99 |
| 512 | 1.62 | 2.21 | 0.59 | 0.99 |
| 1024 | 1.73 | 2.32 | 0.49 | 0.97 |
| 2048 | 2.17 | 2.70 | 0.43 | 0.97 |
| 4096 | 2.50 | 3.19 | 0.43 | 0.96 |
| 8192 | 3.65 | 4.56 | 0.40 | 0.96 |

TABLE II

COMPARISON OF BEST-EFFORT AND PRM-ENHANCED NICE PROTOCOLS WITH VARYING GROUP SIZES. WE USE PRM-(3,0.01) IN THESE EXPERIMENTS.

fact for most of the scenarios shown in the table, PRM requires overheads less than 10% to meet the desired deadline and delivery ratios. PRM requires higher overheads for only the very stringent deadline of 200 ms and to achieve 99% delivery ratio for a 500 ms deadline. As is evident from the table, FEC-based schemes require far higher overheads even for much lower group change rates (0.1 per second).

*Scalability:* In Table II we show the effect of multicast group size on control overheads and delivery ratio. In this set of results, the group change rate was 0.2% of the group size. For example, for 512-node groups, the change rate is 1 node per second, while for the 8192-node case, it is 16 nodes per second. In the original best-effort NICE application-layer multicast, the control overheads at different overlay nodes increase logarithmically with the increase in group size. The control overheads for the PRM-enhanced NICE are higher, due to the additional messages such as random node *Discover* messages and NAKs. However the amount of increase at each overlay node is less than one control packet

per second, which is negligible in comparison to data rates that will be used by the applications (e.g., media streaming). We also observe that the data delivery ratio of the PRM-enhanced NICE protocol is high across various group sizes.

*Loss Correlation and EGF:* We briefly describe other experiments to demonstrate the benefits of the two proposed extensions to the basic PRM scheme. We simulated some specific pathological network conditions that led to highly correlated losses between large groups of members; here use of the loss correlation technique improved data delivery rates by up to 12%.

EGF is beneficial under high frequency of group changes. For the experiment with 512 overlay nodes and 10 group changes per second, EGF can improve the data delivery ratio from about 80% (see Figure 6) to 93%. Note that under these circumstances 512 changes (i.e. same as the size of the group) to the group happen in less than a minute. However, it also increases duplicate packets on the topology by nearly 10%.

## V. Related Work

A large number of research proposals have addressed reliable delivery for multicast data, most notably in the context of network-layer multicast. A comparative survey of these protocols is given in [17] and [28]. In SRM [9] receivers send NAKs to the source to indicate missing data packets. Each such NAK is multicast to the entire group and is used to suppress NAKs from other receivers that did not get the same packet. In this approach, however, a few receivers behind a lossy link can incur a high NAK overhead on the entire multicast group.

Tree-based protocols provide another alternative solution for reliable and resilient multicast. In this approach the receivers are organized into an acknowledgment tree structure with the source as the root. This structure is scalable because the acknowledgments are aggregated along the tree in a bottom-up fashion and also allows local recovery and repair of data losses. Protocols like RMTP [24], TMTP [30], STORM [26], LVMR [19] and Lorax [18] construct this structure using TTL-scoped network-layer multicast as a primitive. In contrast, LMS [23] uses an additional mechanism, called directed subcast, to construct its data recovery structure. Our work differs from of all these above approaches in two key aspects. First, unlike all these protocols that employ network-layer multicast service for data distribution our scheme is based upon an application-layer multicast delivery service. To the best of our knowledge the PRM scheme is the first application-layer multicast based scheme that addresses resilience.

Second, all the network-layer multicast based schemes described employ completely reactive mechanisms for providing data reliability and therefore incurs moderate or high delivery latencies. As we show in this paper, proactive mechanisms, e.g. randomized forwarding, can be used to significantly improve resilience for applications that require low latency data delivery.

PRM is not the only proactive approach to provide improved reliability performance for multicast data. There exists some well-known forward error correcting code based approaches that are also proactive in nature. For example, Huitema [13] had proposed the use of packet level FECs for reliable multicast. Nonnenmacher et. al. [22] studied and demonstrated that additional benefits can be achieved when an FEC-based technique is combined with automatic retransmission requests. APES uses a related approach for data recovery [27]. Digital Fountain [4] and RPB/RBS [20] are two other efficient FEC-based approaches that provide significantly improved performance. All these FEC based approaches can recover from network losses. However, they alone are not sufficient for resilient multicast data delivery when overlays are used. Overlay nodes are processes on regular end-hosts and are more prone to failures than network routers. FEC-based approaches are not sufficient to recover from losses due to temporary losses on the data path, especially when low-latency delivery is required. The PRM scheme differs from all these other schemes by providing a proactive component that allows the receivers to recover from losses due to overlay node failures. In Table III we summarize the characteristics of all these schemes.

A different randomized error recovery scheme called RRMP is proposed in [29]. RRMP differs from PRM in that nodes detecting losses send randomized repair request (reactively) while PRM uses proactive randomized forwarding. Gossiping [11] and PRM both use randomized forwarding. However, randomized forwarding in PRM only provides additional resilience to data delivery along multicast tree, while gossiping only uses communication with random peers. Also related is dispersity routing scheme [21], where additional messages are sent using different routes. The difference is that PRM forwards data packets through randomized routes instead of a fixed set of paths between source and destination as in dispersity routing.

Since PRM was first published [2], there have been two interesting pieces of related work. SplitStream [6]

[2]Although FEC-based schemes can be implemented over application-layer multicast, as this paper shows, it alone is not sufficient to achieve high delivery ratios even under moderate frequency of membership changes on the overlay.

| Scheme | Data delivery | Recovery mechanism | Overheads | Recovery latency |
|--------|---------------|--------------------|-----------|-------------------|
| SRM [9] | Network multicast | Reactive NAKs with global scope | High (for high network losses) | High |
| STORM [26] Lorax [18] | Network multicast | Reactive NAKs on ack tree | Low | Moderate |
| LMS [23] | Network multicast and directed subcast | Reactive NAKs on ack-tree | Low | Moderate |
| RMTP [24] LVMR [19] | Network multicast | Reactive/periodic ACKs with local scope | Low | Moderate |
| TMTP [30] | Network multicast | Reactive NAKs and periodic ACKs with local scope | Low | Moderate |
| Parity-based [22] (APES [27]) | Network multicast (and directed subcast) | Reactive NAKs and FEC-based repairs | Moderate | Moderate |
| FEC-based [13], [22], [4], [20] | Network multicast or App-layer multicast [2] | Proactive FECs | High | Low |
| Overcast [14] | App-layer multicast | Reactive ACKs (TCP) | Low | Moderate |
| Bullet [15] | App-layer multicast | Perpendicular streams | High | Low |
| PRM | App-layer multicast | Proactive randomized forwarding and reactive NAKs | Low | Low |

TABLE III

COMPARISON OF DIFFERENT RELIABILITY/RESILIENCE MECHANISMS FOR MULTICAST DATA.

tries to achieve a balance of forwarding load among various overlay nodes. SplitStream constructs multiple independent trees and partitions the data into multiple stripes, each of which is sent along each tree. Although we can use a special coding technique over multiple stripes to achieve higher data resilience, SplitStream itself has no redundancy or resilience mechanisms. In contrast, PRM uses the probabilistic forwarding approach to send redundant data packets to handle overlay node failures. In fact, PRM can be employed on each SplitStream tree to achieve high resilience. Bullet [15] achieves high bandwidth data dissemination through "perpendicular streams." These perpendicular streams are analogous to our (low bandwidth) probabilistic non-tree edges, but their non-tree edges are persistent. Also, the probabilistic edges in PRM serve recovery from losses while the perpendicular streams in Bullet are used to improve bandwidth. Although it is possible to consider Bullet as a resilient overlay multicast technique, the performance results presented in the Bullet paper show high data overhead (e.g., more than 50%). In contrast, we show the low overhead and resilience of PRM through analysis as well as simulations.

## VI. CONCLUSIONS

In this paper, we have shown how relatively simple mechanisms can be used to provide highly resilient data delivery over application-level multicast distribution trees. We have identified randomized forwarding as a key mechanism to mask data delivery failures due to failed overlay nodes, and have shown how even a very low overhead randomized forwarding is sufficient for handling rapid and massive changes to the distribution group. Our results are especially interesting since previously studied error-recovery techniques, such as FEC, alone do not provide adequate data recovery especially when overlay node failure rates are high. We have analytically shown why a randomized forwarding approach is able to achieve high data delivery ratios with low latencies. We have also derived necessary and sufficient conditions that enable PRM to have such asymptotic scaling properties.

Our detailed packet-level simulations show that the mechanisms described in this paper are immediately useful in realistic scenarios involving streaming media applications. The low bandwidth, storage and processing overheads of PRM makes it attractive for both low and high bandwidth streaming applications. Further, the very

high recovery ratios—often in excess of 97% under adverse conditions—will allow PRM to be used with more efficient media encodings (which are usually less tolerant of data loss). We have recently incorporated the idea of PRM into a system for resilient media streaming [3]. We have experimented the streaming system in an emulated network[3] and a wide-area testbed, and obtained similar results to those from simulations. We believe the techniques we have described will become standard and essential components of streaming media applications implemented over application-layer overlays.

### REFERENCES

[1] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proc. ACM Sigcomm*, Aug. 2002.

[2] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. Resilient multicast using overlays. In *Proc. ACM SIGMETRICS*, 2003.

[3] S. Banerjee, S. Lee, R. Braud, B. Bhattacharjee, and A. Srinivasan. Scalable resilient media streaming. In *Proceedings of NOSSDAV '04*, pages 4–9. ACM Press, 2004.

[4] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE Journal on Selected Areas in Communications*, 20(8), Oct. 2002.

[5] K. Calvert, E. Zegura, and S. Bhattacharjee. How to Model an Internetwork. In *Proc. of Infocom*, 1996.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: high-bandwidth multicast in cooperative environments. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 298–313. ACM Press, 2003.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications*, 20(8), Oct. 2002. To appear.

[8] Y.-H. Chu, S. G. Rao, and H. Zhang. A Case for End System Multicast. In *Proc. ACM SIGMETRICS*, 2000.

[9] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6), Dec. 1997.

[10] P. Francis. Yoid: Extending the Multicast Internet Architecture, 1999. White paper http://www.aciri.org/yoid/.

[11] S. M. Hedetniemi, T. Hedetniemi, and A. L. Liestman. A Survey of Gossiping and Broadcasting in Communication Networks. *NETWORKS*, 18, 1988.

[12] W. Hoeffding. Probability inequalities for sums of bounded random variables. *American Statistical Association Journal*, 58:13–30, 1963.

[13] C. Huitema. The case for packet level FEC. In *Proc. 5th International Workshop on Protocols for High Speed Networks*, Oct. 1996.

[14] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of the 4th Symposium on Operating Systems Design and Implementation*, Oct. 2000.

[15] D. Kostic;, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: high bandwidth data dissemination using an overlay mesh. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 282–297. ACM Press, 2003.

[16] J. Leibeherr and M. Nahas. Application-layer Multicast with Delaunay Triangulations. In *IEEE Globecom*, Nov. 2001.

[17] B. Levine and J. Garcia-Luna-Aceves. A comparison of reliable multicast protocols. *Multimedia Systems Journal*, 6(5), Aug. 1998.

[18] B. Levine, D. Lavo, and J. Garcia-Luna-Aceves. The case for concurrent reliable multicasting using shared ack trees. In *Proc. ACM Multimedia*, Nov. 1996.

[19] X. Li, S. Paul, P. Pancha, and M. Ammar. Layered video multicast with retransmissions (LVRM): Evaluation of error recovery schemes. In *Proc. NOSSDAV*, 1997.

[20] A. Mahanti, D. Eager, M. Vernon, and D. Sundaram-Stukel. Scalable on-demand media streaming with packet loss recovery. In *ACM Sigcomm*, Aug. 2001.

[21] N. F. Maxemchuk. Dispersity routing. In *Proc. ICC*, pages 41.10–41.13, 1975.

[22] J. Nonnenmacher, E. Biersack, and D. Towsley. Parity-based loss recovery for reliable multicast transmission. *IEEE/ACM Transactions on Networking*, 6(4), Aug. 1998.

[23] C. Papadopoulos, G. Parulkar, and G. Varghese. An error control scheme for large-scale multicast applications. In *Proc. Infocom*, 1998.

[24] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (rmtp). *IEEE Journal on Selected Areas in Communications*, 15(3), Apr. 1997.

[25] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *Proc. of 3rd International Workshop on Networked Group Communication*, Nov. 2001.

[26] X. Rex Xu, A. Myers, H. Zhang, and R. Yavatkar. Resilient multicast support for continuous-media applications. In *Proceedings of NOSSDAV*, 1997.

[27] D. Rubenstein, S. Kasera, D. Towsley, and J. Kurose. Improving reliable multicast using active parity encoding services (APES). In *Proc. Infocom*, 1999.

[28] D. Towsley, J. Kurose, and S. Pingali. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas on Communication*, 15(3), Apr. 1997.

[29] Z. Xiao and K. Birman. A randomized error recovery algorithm for reliable multicast. In *Proc. Infocom*, 2001.

[30] R. Yavatkar, J. Griffioen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia*, Nov. 1995.

[31] B. Zhang, S. Jamin, and L. Zhang. Host multicast: A framework for delivering multicast to end users. In *Proc. IEEE Infocom*, June 2002.

### APPENDIX

#### A. Proof sketch for Theorem 3.1

We now outline the main ideas of the proof. In several places below, we will employ the *union bound*: for any collection of events $E_1, E_2, \ldots, E_m$,

$$\Pr[E_1 \vee E_2 \vee \cdots \vee E_m] \leq \sum_{i=1}^{m} \Pr[E_i].$$

We will also use the Hoeffding bound [12] often. In particular, this bound shows the following. Let $\exp(x)$

---

[3] http://www.emulab.net

denote $e^x$. Suppose $X_1, X_2, \ldots, X_\ell$ are *independent* random variables, each taking values in the set $[0,1]$. Let $X = X_1 + X_2 + \cdots + X_\ell$, and let $\mu$ be the expected value $E[X]$ of $X$. Then, for any $\alpha \in [0,1]$, the Hoeffding bound shows that

$$
\begin{aligned}
\Pr[X \geq \mu(1+\alpha)] &\leq \exp(-\mu\alpha^2/3); \\
\Pr[X \leq \mu(1-\alpha)] &\leq \exp(-\mu\alpha^2/2).
\end{aligned}
$$

We will use the phrase "high probability" to refer to any probability value that tends to 1 as the number $n$ of nodes in the overlay tree increases; complementarily, "low probability" will refer to any probability value that tends to 0 as $n$ increases.

Let the overlay tree have some depth $d$; for $i = 0, 1, \ldots, d-1$, suppose all nodes at depth $i$ have some $D_i$ children. (Recall that the root is at depth 0, its children at depth 1, etc.) Recall the failure model (A2) of Section III. Suppose node $u$ does not fail. Then, define $\mathcal{C}(u)$ to be the connected component (subtree) containing $u$, after the node- and link-failures occur. For each connected component $\mathcal{C}$, we choose as a *leader*, the element of $\mathcal{C}$ of *smallest depth*. Suppose the probability of random forwarding $p$ has been chosen (as an arbitrarily small positive constant), and that each value $D_i$ is at least $C \log n$ where $C$ is a sufficiently large constant, as required by Theorem 3.1. We first show that the following conditions hold simultaneously with high probability:

(P1)   the number of surviving nodes at every depth $i$ is at least $(1 - 2\epsilon) \cdot D_0 D_1 \cdots D_{i-1}$; and

(P2)   for all $i \leq d - 1$ and for all surviving nodes $u$ at depth $i$, the number of descendants of $u$ that lie in $\mathcal{C}(u)$, is at least $((1-\epsilon)^2(1-\delta))^{d-i} \cdot D_i D_{i+1} \cdots D_{d-1}$.

Claim (P1) is proved using a simple application of the Hoeffding bound, followed by the union bound. Consider any $i$, and let $\lambda = 1 - (1-2\epsilon)/(1-\epsilon) \sim \epsilon$. The expected number $X$ of surviving nodes at depth $i$ is $\mu_i = (1-\epsilon)D_0 D_1 \cdots D_{i-1}$. The probability that $X$ is smaller than $(1-2\epsilon)D_0 D_1 \cdots D_{i-1}$ is at most $\exp(-\lambda^2 \mu_i/2)$ by the Hoeffding bound. Now, the reader may verify, using the fact that $D_j \geq C \log n$ for all $j$, that this bound is at most, say, $1/(2n^2)$ if the constant $C$ is large enough. Applying a union bound over all $i$, we get that (P1) holds with high probability. It is for such technical reasons that we require the degrees $D_j$ to be at least logarithmic in $n$ here, and below; we will also see briefly in Appendix B that such degree bounds are necessary.

Claim (P2) is proven using a similar iterative application of the Hoeffding bound; the intuition is as follows. Call a node *good* if it, as well as the link connecting it to its parent, survives. The expected number of good

children of $u$ is $D_i(1-\epsilon) \cdot (1-\delta)$; since $D_i$ grows at least logarithmically in $n$, one can use the Hoeffding bound to show that with high probability, at least $D_i(1-\epsilon)^2 \cdot (1-\delta)$ children of $u$ are good. Iterating this argument down the tree and applying a union bound over all $u$, we prove (P2).

A union bound that sums up the probabilities of (P1) and (P2) not holding (each of which is negligible, as seen in the above two paragraphs), shows that both (P1) and (P2) hold, with high probability. Thus, for the rest of this proof sketch for Theorem 3.1, we assume that (P1) and (P2) hold. Note that all the probabilities considered in this proof sketch so far are only w.r.t. the random failures in the overlay tree; the probabilities from now are on w.r.t. the random forwarding.

The heart of the rest of the proof is as follows; we first give a proof sketch for part (i) of Theorem 3.1. To show that every surviving non-leaf node gets the data with high probability, it suffices to show that for every non-leaf *leader* $u$, some node in $\mathcal{C}(u)$ gets the data with high probability. (Once this happens, the data gets reliably transmitted across the links of $\mathcal{C}(u)$ with probability 1.) For the sake of simplicity of our mathematical expressions, we assume here that all non-leaf nodes have the same number of children $D$ (this condition is only required for notational convenience here). We show the following claim by induction on $i$, where $a$ is a positive constant, and $\psi$ is a strictly positive quantity which depends only on $p$, $\epsilon$ and $\delta$:

> *Let $u$ be an arbitrary non-leaf leader at depth $i$. Then, conditional on (P1) and (P2), the probability of no node in $\mathcal{C}(u)$ getting the data is at most $\exp(-a(D\psi)^{d-i})$.*

The base case of the induction is for $i = 0$ which corresponds to the case where $u$ is the root. Since the root never fails, the claim holds trivially. Next suppose $i \geq 1$. Assuming the claim for all $i' < i$, we complete the induction for $i$ as follows. Since (P1) and (P2) are assumed to hold, it can be shown that the cardinality of the set $\mathcal{S}$ of surviving nodes in connected components whose leaders are at depth strictly smaller than $i$, is at least

$$
\begin{aligned}
n_1 &= \Omega(D^d((1-\epsilon)^2(1-\delta))^{d-i+1}) \\
&= \Omega(n \cdot ((1-\epsilon)^2(1-\delta))^{d-i+1}).
\end{aligned}
$$

By choosing $a$ appropriately, we can apply the induction hypothesis and a union bound to show that the probability of even one such connected component (whose leader is at depth smaller than $i$) not getting the data, is much smaller than $\exp(-a(D\psi)^{d-i})$. Next, assuming (P2), the

size of $\mathcal{C}(u)$ is at least

$$n_2 = ((1-\epsilon)^2(1-\delta))^{d-i} \cdot D^{d-i}.$$

Thus, the probability that no random forward from $\mathcal{S}$ arrived into $\mathcal{C}(u)$, is at most

$$(1-p/n)^{n_1 n_2} \leq \exp(-(p/n) \cdot n_1 n_2);$$

i.e., at most

$$\exp(-\Omega(p \cdot D^{d-i} \cdot ((1-\epsilon)^2(1-\delta))^{2(d-i)+1})),$$

which can be bounded by $\exp(-a(D\psi)^{d-i})$ for a suitable choice of $\psi$. These ideas help complete the induction proof; a union bound over all surviving non-leaf nodes (using the facts $d-i \geq 1$ and that $D$ grows at least logarithmically in $n$) then shows that all of them get the data with high probability.

Having shown the above, we can handle the surviving leaves, to prove part (ii) of Theorem 3.1. Consider the surviving leaves; the fraction of these such that their parent, as well as the link to their parent, survive, can be shown to be at least $(1-\epsilon) \cdot (1-\delta) \cdot (1-g(n))$ with high probability using the Hoeffding bound, where $g(n)$ tends to 0 as $n$ increases. Now, we have argued above that all surviving non-leaves get the data with high probability; thus, all surviving leaves that remain connected to their (surviving) parent, will receive the data with high probability. This completes the proof sketch for part (ii) of Theorem 3.1.

### B. Lower bounds on tree-degree

The degree lower bound of a suitable constant times $\log n$ can be shown to be asymptotically necessary, as follows. Suppose, for some small constant $b > 0$, that all parents of leaves have degree at most $b \log n$. Then, we give a proof sketch a few lines below that with probability tending to 1 as $n$ increases, a substantial number of *surviving* parents-of-leaves will have the following property: they get disconnected from all of their neighbors in the tree. Then, if the overhead needs to be kept small, most of these isolated parents-of-leaves will not get the data, with high probability. Here is a proof sketch. Suppose the number of parents-of-leaves that survive is some value $t$. For any one of them, the probability that it gets disconnected from all of its neighbors can be as high as

$$
\begin{aligned}
(\epsilon + \delta - \epsilon\delta)^{b \log n} &\geq \epsilon^{b \log n} \\
&= n^{-b \log(1/\epsilon)}.
\end{aligned}
$$

One can then show that with high probability, the number of these completely isolated nodes is $\Omega(t \cdot n^{-b \log(1/\epsilon)})$, which is large if $b$ is small enough (e.g., if $b \leq$ $1/(2 \log(1/\epsilon))$). Furthermore, if we require low overhead, the random forwards will with high probability not reach most of these nodes, thus leading to several surviving nodes not receiving the data.

### C. Tree augmentation

Next, consider the tree-augmentation scheme of Section III-A. From the above analysis, we basically see that all surviving non-leaf nodes receive the data with high probability. Thus, essentially the only way that augmentation does not help a particular surviving leaf is when all its $1 + \lfloor \frac{\log(1/\phi)}{\log(1/\epsilon)} \rfloor$ random connections in the tree augmentation were to failed nodes. But, this happens with probability

$$\epsilon^{1+\lfloor \log(1/\phi)/\log(1/\epsilon) \rfloor} < \epsilon^{\log(1/\phi)/\log(1/\epsilon)} = \phi.$$

This can be used to show that with high probability, the fraction of surviving leaves that get the data, is at least $1 - \phi$.

We now sketch why the overhead of $\Omega(\frac{\log(1/\phi)}{\log(1/\epsilon)})$ is necessary. Suppose the average overhead of a node is at most $\alpha \doteq c \cdot \frac{\log(1/\phi)}{\log(1/\epsilon)}$, for some small constant $c$. Then, since most of the nodes are at the leaf level, most leaves get connected to at most $3\alpha$ other nodes, including the random forwards. For a given such leaf, the probability that all of these $3\alpha$ interconnections go to failed nodes, is

$$\epsilon^{3\alpha} = \phi^{3c},$$

which is much larger than $\phi$ if $c \ll 1/3$ (recall that $\phi < 1$). Thus, the fraction of successful nodes that do not get the data, will with high probability be much more than $\phi$ in such a situation. Finally, a similar proof shows that The "$(1-\epsilon) \cdot (1-\delta) \cdot (1-g(n))$ fraction" bound of part (ii) of Theorem 3.1 is optimal if we desire low overhead. The idea is that under low overhead, with high probability the only connections (including the random forwards) for most leaves will be to their parents. Now, each surviving leaf has its connection to its parent in tact with probability $(1-\epsilon) \cdot (1-\delta)$: both the parent, and the link to the parent, must survive. Thus, with high probability, only about a $(1-\epsilon) \cdot (1-\delta)$–fraction of the successful leaves will receive the data with high probability, if we aim to keep the overhead low.