



NORTHWESTERN UNIVERSITY

Computer Science Department

Technical Report
NWU-CS-04-50
October 11th, 2004

The Costs of Resilience in Overlay Multicast Protocols

Stefan Birrer and Fabián E. Bustamante

Abstract

One of the most important challenges of peer-to-peer multicast protocols lies in their ability to handle the high degree of transiency inherent to their environment. A number of techniques have been recently proposed aimed at improving the resilience of these application-layer approaches. However, achieving high delivery ratios without sacrificing end-to-end latencies or incurring additional costs has proven to be a challenging task. In this paper we evaluate the effectiveness of various representative protocols by contrasting their performance and associated cost through simulation and wide-area experimentation. Our results show that while it is possible to improve the resilience of an existing protocol, there are clear advantages to building resilience into a protocol from the outset. To the best of our knowledge, this is the first reported comparison of DHT-based and alternative end-system multicast schemes.

Keywords: Peer-to-peer, overlay, multicast, resilience, wide-area evaluation, comparison

The Costs of Resilience in Overlay Multicast Protocols

Stefan Birrer and Fabián E. Bustamante
Department of Computer Science
Northwestern University, Evanston, IL 60201, USA
Email: {sbirrer,fabianb}@cs.northwestern.edu

October 11th, 2004

Abstract

One of the most important challenges of peer-to-peer multicast protocols lies in their ability to handle the high degree of transiency inherent to their environment. A number of techniques have been recently proposed aimed at improving the resilience of these application-layer approaches. However, achieving high delivery ratios without sacrificing end-to-end latencies or incurring additional costs has proven to be a challenging task. In this paper we evaluate the effectiveness of various representative protocols by contrasting their performance and associated cost through simulation and wide-area experimentation. Our results show that while it is possible to improve the resilience of an existing protocol, there are clear advantages to building resilience into a protocol from the outset. To the best of our knowledge, this is the first reported comparison of DHT-based and alternative end-system multicast schemes.

1 Introduction

Deployment issues with IP Multicast [18, 19] have motivated recent work on alternate, peer-to-peer approaches for supporting group communication applications over the Internet [15, 24, 22, 30, 13, 4, 12, 32, 43, 29, 38]. In this application-layer approach, participating peers organize themselves into an overlay topology for data delivery. The topology is an overlay in that each edge corresponds to a unicast path between two end systems or peers in the underlying Internet. All multicast related functionality is implemented at the peers instead of at the routers, and the goal of the multicast protocol is to construct and maintain an efficient overlay for data transmission.

However, as multicast functionality is pushed to autonomous, unpredictable peers, significant performance loss can result from the end hosts' higher degree of transiency when compared to routers [6]. Consequently, a number of techniques [5, 10, 8] have been recently proposed that try to improve overlay resilience by exploiting path diversity [1, 36] and minimizing node dependencies [2]. The different techniques can be classified into three different categories – multiple trees, single-tree redundancy and cross-link redundancy. The **multiple tree** approach creates several overlapping trees over which stripes of the multicast stream are forwarded [10, 29, 39]. The **single-tree** and **cross-link redundancy** approaches instead, improve resilience by adding extra links to a single multicast tree. The **single-tree redundancy** approach creates alternative paths only around nodes with forwarding responsibility [8, 41], while the **cross-link redundancy** approach forwards an additional fraction of the stream over extra cross-cutting links connecting random peers in the tree [5].

Delivering high application performance, at relatively low costs and under high degree of transiency has proven to be a difficult task [33, 14, 27]. Each of the proposed resilient overlay multicast techniques comes with a different tradeoff in terms of delivery ratio, response time and additional network traffic. To help guide further research, this paper evaluates the effectiveness of representative techniques by contrasting their performance and associated cost through simulation and wide-area experimentation. We restrict our comparison to stream-based protocols, where timely data delivery is a key requirement. The evaluation of resilient protocols for bulk data dissemination [26, 16] lies outside the scope of this paper. To the best of our knowledge, this paper presents the first comparison of DHT-based and tree-based end-system multicast schemes.

Our results show that while it is possible to improve the resilience of an existing protocol, there are clear advantages to building resilience into a protocol from the outset. Nemo [8] achieves lower response time under high failure rates than the alternative resilient protocols (Nice-PRM and SplitStream), while maintaining a high delivery ratio at a comparatively low cost. Evaluation results also indicate the performance advantage of tree-based multicast over DHT-based multicast approaches, with the former achieving higher delivery ratios than their counterparts when exercised under continuous failures. Our results confirm previous findings which identified congestion collapse as a main problem in existing DHT protocols [33].

The remainder of the paper is organized as follows. Section 2 provides some useful background information and Section 3 briefly discusses the evaluated protocols. We outline our evaluation environment in Section 4 and report experimental results from simulations and wide-area experimentation in Section 5. We discuss our findings in Section 6. Section 7 describes related work and Section 8 concludes.

2 Background

Multicast is an efficient mechanism to support group communication applications such as audio and video conferencing, multi-party games and content distribution. It decouples the size of the receiver set from the amount of state kept at any single node and potentially avoids redundant communication in the network. Partially in response to the deployment issues of network-layer multicast [18, 19], a number of protocols recently proposed adopt an alternative peer-to-peer, or application-level approach, with all multicast related functionality implemented exclusively at end-systems [15, 24, 22, 13, 4, 12, 32, 29, 38].

All application-level multicast protocols organize the participating peers in two topologies: a control overlay for group membership related tasks, and a delivery tree for data forwarding. Extending a previous classification [3], we group the available protocols into four classes based on the sequence adopted in their construction and their structuring approach – tree-first, mesh-first, DHT-first and implicit. In a **tree-first** approach [22, 24, 30], peers directly build the data delivery tree by selecting their parents from among known peers. Additional links are later on added to define the control topology. With a **mesh-first** approach, peers construct a more densely connected graph (mesh) over which a (reverse) shortest path spanning trees, rooted at any peer, is built [15]. Under the **DHT-first** approach, peers organized themselves into a well-defined geometrical structure over which a data delivery topology is built [34, 32]. With an **implicit** approach, peers create only a control topology, while the data delivery tree is implicit defined by packet forwarding rules based on the initial control tree [4].

The migration of multicast functionality to end hosts may result in significant performance loss due to the peers’ relatively high failure rates [6]. A good indicator of peer transiency is the peers’ *median session time*, where *session time* is defined as the time between when a peer joins and leaves the network. Measurement studies of widely used P2P systems have reported median session times ranging from 90 to one minute [9, 14, 35, 23]. Although mostly collected from file-sharing applications, these measurements give us an idea of the high level of transiency that can be expected in large peer populations.

The following section briefly describes the set of representative protocols for resilient overlay multicast we employ for our study: SplitStream [10], a DHT-based, multiple-tree system; Nemo [8], an implicit, single-tree redundancy protocol; and Nice-PRM [5], an implicit, cross-link redundancy protocol.

3 Protocols

Most of the resilient protocols studied can be better described, and their effectiveness better evaluated, when put in the context of a baseline, i.e. non-resilient alternatives. Thus, in this section we outline the three resilient protocols studied as well as two other non-resilient ones: Nice [4] and Scribe [12].

For Nice-PRM [5] and Nemo [8], Nice [4] is an obvious choice as the baseline protocol. In **Nice**, participating peers are organized into clusters based on network proximity, with every peer being a member of a cluster at the lowest layer. Clusters vary in size between d and $3d - 1$, where d is a constant known as *degree*. Each of these clusters selects a *leader* that becomes a member of the immediately superior layer. The process is repeated, with all peers in a layer being grouped into clusters and new leaders elected and promoted to participate in the next higher layer. Hence peers can lead more than one cluster in successive layers of this logical hierarchy. **Nemo** adopts the same implicit approach to overlay multicast as Nice, organizing peers into a logical hierarchy over which the data delivery network is implicitly defined based on a set of forwarding rules. Nemo achieves resilience through the introduction of *co-leaders*, alternative leaders that share the forwarding load of clusters' leaders. Co-leaders improve resilience by avoiding dependencies on single nodes and providing alternative paths for data forwarding. Nemo improves recovery time for lost packets through a novel use of triggered negative acknowledgments and reduces overlay maintenance cost through the adoption of a probabilistic approach where operations are executed with some probability or, alternatively, deferred to the next interval. In the presence of high churn, many of these operations can not only be deferred, but completely avoided as follow-up changes may revert a previously triggering situation. The **Probabilistic Resilient Multicast (PRM)** is a general scheme to improve the resilience of overlay multicast [5]. PRM employs two key mechanisms to enhance resilience: triggered negative acknowledgments and randomized forwarding. When layered over Nice, PRM can be seen as creating cross-links, connecting random peers on the tree, over which an additional fraction of the stream is forwarded.

SplitStream [10] follows a multiple-tree approach to resilient overlay multicast. It relies on a set of multiple disjoint multicast trees over which different stripes of the data stream are disseminated. SplitStream could be potentially implemented using any of the tree-based application-level multicast systems. However, the only currently available implementation [10], and thus the one employed for our evaluation, is built using on **Scribe** [12] and Pastry [34]. Pastry is a structured (DHT) P2P overlay similar to Chord [37], CAN [32] and Tapestry [42]. Scribe is a group communication system built upon Pastry [34] - a rooted multicast tree is built by

the union of Pastry routes from each group member to the root. Messages are then multicast from the root using reverse path forwarding [17].

4 Evaluation

This section presents our evaluation setup, provides some details on the implementations of the compared protocols and discusses the metrics employed in our evaluation.

4.1 Evaluation Setup

The different protocols were evaluated through simulation, using SPANS a locally-written, packet-level, event-based simulator; and wide-area experimentation in the PlanetLab testbed [31].

We run our simulations using different topology generators, network and multicast group sizes. Here we present simulation results based on Inet [25] topologies with 8,192 nodes and a multicast group of 128, 256, 512 and 1024 members¹. Members are randomly attached to routers with a random delay (between 1 and 20 ms) and bandwidth capacity (between 1 and 100 MByte/sec) assigned to every link. We opted for relatively high bandwidths, when compared with streaming rate, to avoid it from becoming the primary limiting factor to throughput. Nevertheless, we employ a finite value in order to model the effect of queueing delays. The links use drop-tail queues with a buffer capacity of 0.5 sec.

Each simulation experiment lasts for 5 minutes (60 minutes for Scribe and SplitStream) of simulation time. All peers join the multicast group by contacting the rendezvous point at uniformly distributed, random times within the first 120 seconds of the simulation for FatNemo, Nice, Narada and Overast; the Pastry nodes join the network distributed over the first 30 minutes. While the tree-based protocols enable the multicast layer immediately after joining, Scribe and SplitStream are activated after Pastry runs for 20 minutes [10]. The warm-up time is 180 seconds for Nice, Nice-PRM and Nemo; Scribe and SplitStream are granted 57 minutes of warm-up time, during which the protocols establish and optimize their structures. Clearly, longer warm-up times allow a protocol to better adapt to the underlying network. The warm-up time is omitted from the figures. Following the warm-up period and lasting for 120 seconds, each simulation has a phase with rapid membership changes. During this time each protocol is exercised under two different failure

¹Comparable results were obtained with alternative topologies and different group sizes [7].

rates derived from Xu et al. [40] study on networked system failure and related research on resilient multicast [5, 40]. Under a *high-failure rate*, nodes fail independently at a time sampled from an exponential distribution with *mean time to failure (MTTF)* equal to 5 minutes, to rejoin shortly after (time sampled from an exponential distribution with *mean time to repair (MTTR)* equal to 2 minutes). The same set of simulations is also run with a *low-failure rate* defined by MTTF of 60 minutes and a MTTR of 10 minutes. The means for each failure rate are chosen asymmetrically to allow, on average, 5/7 (6/7) of all members to be up during this phase. We generate a failure event sequence based on the above MTTF and MTTR; the same sequence is used for all protocols and all runs.

For our wide-area experiments we employ 50 end hosts distributed across ~ 30 PlanetLab sites. The number of end hosts per site varies between 1 and 9, with most sites having a single node. We inject failures at the same rates as in our simulation. Each protocol is exercised under failures for 15 minutes; we present results measured during the last 10 minutes of the failure injection. The failure events are drawn from a exponential distributed random number generator based on the MTTF and MTTR of the corresponding failure scenario. The warm-up period is set to 10 minutes for Nemo, Nice and Nice-PRM. For SplitStream and Scribe, we use 30 minutes warm-up time. Since both of these two protocols run on top of a DHT, which periodically improves the routing tables at 15 minutes intervals, a comparatively long setup time was necessary to reach the best performance for both protocols.² The different protocols were evaluated multiple times (10) each, randomly alternating their order of execution in each instance.

To estimate the end-to-end delay, we make use of a global time server. Every peer estimates the difference of its local time to the time at the server. The algorithm is inspired by [28] and leads to sufficient accuracy for our application.

All experiments were run with a payload of 1000 Bytes. We employ a buffer size of 16 packets and a rate of 10 packets per second. This corresponds to the usage of a 1.6-second buffer and a data rate of 80 Kbps, a realistic scenario for applications such as multimedia streaming.

4.2 Details on Protocol Implementations

For each of the three alternative protocols, the values for the available parameters were obtained from the corresponding literature [4, 5, 8, 34, 12, 10].

We used our own implementation of Nice, Nice-PRM and Nemo (as described in [8]), which closely follows the descriptions from the literature. For Nice [4] and Nice-PRM [5], the cluster degree, k , was set to 3. We used PRM with three

²Longer warm-up periods do not offer a noticeable performance improvement to Scribe and SplitStream.

random peers chosen by each node, and with two percent forwarding probability.³ For Nemo, the cluster degree k and the crew size were set to 3. The grace period was set to 15 seconds for Nice, Nice-PRM and Nemo.

For Scribe and SplitStream, we used the implementations included as part of FreePastry 1.3.2 [20]. We layered FreePastry on top of SPANS for our simulation-based evaluation.

For the wide-area implementation, we employed UDP with TCP-friendly rate control [21] for Nice, Nice-PRM and Nemo. The loss rate is estimated with the Dynamic History Window method [21]. We limited the number of retransmissions to ten attempts for heartbeats and five for all other control traffic. Data communication did not employ retransmission. For Scribe and SplitStream we used instead the RMI communication layer included with FreePastry. We have opted for this approach, as it resulted in considerable better performance than our TCP-friendly rate control.⁴

4.3 Evaluation Criteria

We used several metrics to evaluate the different resilient overlay multicast protocols. The different metrics capture both delivered performance to the application and protocol's overhead. Application performance is captured by delivery ratio and end-to-end latency, while overhead is evaluated in terms of number of duplicate packets per sequence number and control-related traffic.

Delivery Ratio. Ratio of subscribers which have received a packet within a fixed time window. Disabled receivers are not accounted for.

Response Time. End-to-end delay (including retransmission time) from source to receivers, as seen by the application. This includes path latencies along the overlay hops, as well as queueing delay and processing overhead at peers along the path.

Duplicate Packets. Number of duplicate packets for all receivers counted per sequence number, reflecting an unnecessary burden on the network. Packets arrived outside of the delivery window are accounted for as duplicates, since the receiver already assumed them as lost.

Data-Related Traffic. Total data traffic in the system, in MB per second, during the failure interval. We measure the total data traffic during the observation interval by accounting packets at the router level.

³For evaluation results with different forwarding probabilities, see [8]

⁴We are currently investigating the reasons for the observed performance difference.

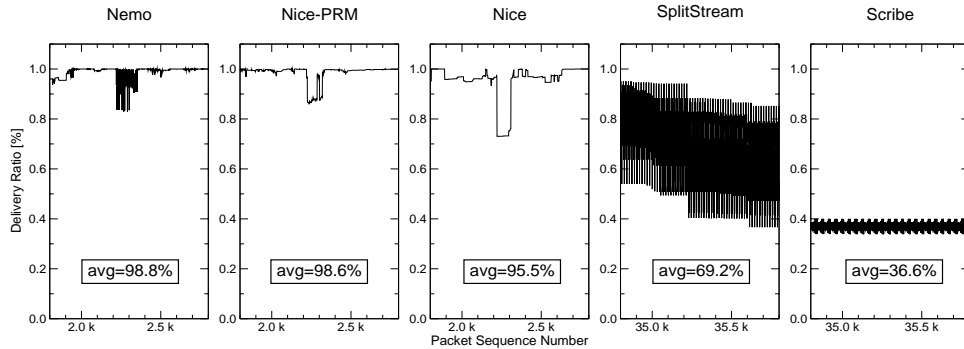


Figure 1: Delivery ratio (512 end hosts, low failure rate). Notice that Scribe and SplitStream were given a longer warm-up period compared to the other protocols, thus the sequence numbers start at a higher value.

Control-Related Traffic. Total control traffic in the system, in MB per second, during the failure interval. We measure the total control traffic during the observation interval by accounting packets at the router level.

In the following section we present results from our simulation and wide-area experimentation.

5 Evaluation Results

Unless otherwise noted, all reported results are based on ten independent runs per protocol and setup, for both the simulation and wide-area experiments.

Figures 1 and 2 illustrate the protocols ability to cope with transiency. In both cases, tree based protocols show a smoother and considerable higher delivery ratio than DHT-based protocols. The oscillating delivery ratio for Scribe and SplitStream seem to indicate that both protocols create hot spots, where some fraction of the packets are dropped, while others are still forwarded in the distribution tree. For the high failure rate scenario, SplitStream’s delivery ratio deteriorates over time, possible indicating the presence of congestion collapse also found in Pastry networks by previous evaluations [33].

The experienced performance under low and high failure rates are summarized in Table 1. Implicit protocols deliver a large fraction of the packets; their resilient variants Nemo and Nice-PRM perform best with a delivery ratio of more than 98%

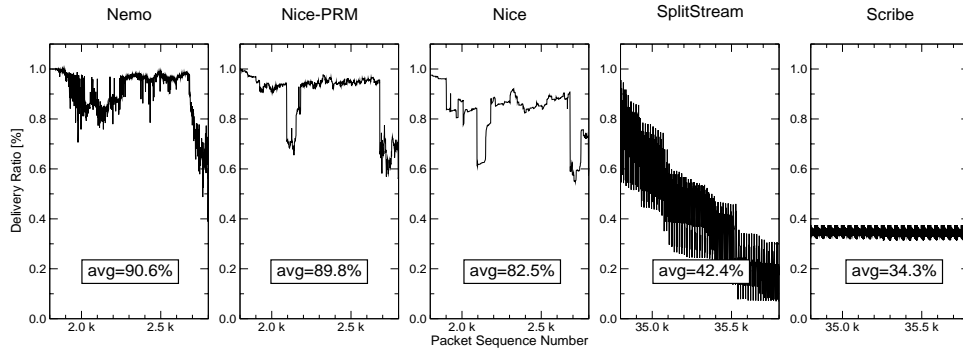


Figure 2: Delivery ratio (512 end hosts, high failure rate). Notice that Scribe and SplitStream were given a longer warm-up period compared to the other protocols, thus the sequence numbers start at a higher value.

under low failures and $\sim 90\%$ under high failures. Scribe, on the other hand, delivers only a small fraction of the forwarded data ($< 37\%$). Its resilient counterpart considerably improves its delivery ratio (by nearly 70%) under the low failure rate. Under high failure rates, SplitStream’s benefits are less clear (a $\sim 24\%$ improvement).

In terms of response time, tree based variants show clear performance advantages, especially when considering their higher delivery ratios. Packets traveling longer paths (in terms of overlay links), and thus resulting on possible higher response times, have a higher chance of getting lost.

Higher resilience may also result on an increased number of duplicate and late packets (counted as duplicates as they have become unusable). Although the evaluated resilient tree-based protocols exhibit similar delivery ratios under low failures, the number of duplicates per sequence number differ significantly. Nemo has the lowest number of duplicates under low failures (< 3 pkt/seqnr) (together with the non-resilient Nice), while Nice-PRM’s approach results on a considerable larger number of duplicate packets at the receivers (> 14 pkt/seqnr).

The evaluated DHT-based variants’ duplicate packets are dominated by late packets, which explains the unexpected high count of duplicates for Scribe and SplitStream in the low failure case. Under high failure, SplitStream and Scribe generate no duplicates. One of the reasons for this significant drop emerges from the effects of queuing delays, which are emulated in the simulator at the end hosts and at network links. The duplicates for the low failure rate are generated by packets traversing links with high queuing delays while subsequent packets take a

Table 1: Simulation (512 end hosts).

Protocol	Response time		Delivery ratio		Duplicate packets	
	Mean	Std	Mean	Std	Mean	Std
Low Failure Rate						
Nemo	0.134	0.096	0.988	22.4E-3	2.74	3.62
Nice-PRM	0.115	0.076	0.986	27.9E-3	14.46	13.58
Nice	0.125	0.087	0.955	70.8E-3	2.34	6.10
SplitStream	0.195	0.015	0.692	30.0E-3	5.56	11.00
Scribe	0.140	0.030	0.366	94.3E-3	16.98	24.61
High Failure Rate						
Nemo	0.157	0.225	0.906	79.4E-3	8.35	1.71
Nice-PRM	0.163	0.225	0.898	160.E-3	17.32	12.84
Nice	0.120	0.139	0.825	53.8E-3	4.30	4.32
SplitStream	0.152	0.011	0.424	16.6E-3	0.00	0.00
Scribe	0.147	0.029	0.343	89.9E-3	0.00	0.00

faster route⁵ moving the receiver window ahead, so that the slower packets arrive so much out-of order that they account as a duplicate. Under the high failure rate, more peers leave the session ungracefully and thus reducing the total bandwidth consumption which generally reduces queuing delays and therefore eliminates the out-of order effect the protocols experience under low failure rate.

Figures 3 and 4 present the CDF of the response time for the low and high failure rate, respectively. The cost for delivering packets without retransmission is shown on the left of the plot. As may be expected, protocols with similar approaches exhibit similar costs, i.e. the shape of the curve toward the left side. The implicit approach shows, in general, a lower cost when compared with DHT-based schemes.

The improvements to delivery ratio offered by the recovery schemes can be observed at the point where the plots slowly converge to the maximal delivery ratio at the right side of the graphs. The resilient tree-based variants show a good recovery rate under high failures. Based on this results and considering previous work on multiple trees [39], we believe it should be relatively straightforward to increase SplitStream’s delivery ratio with a simple packet recovery scheme.

Tables 2 and 3 present performance results for the evaluated protocols with different group sizes. ⁶ The results show the effectiveness of the multiple stream approach, as SplitStream yields a constantly higher delivery ratio than its baseline

⁵Potentially induced by a change in the distribution tree and/or the multiple-tree approach of SplitStream.

⁶Given the memory limitations on JVMs, we were unable to gather number for Scribe and SplitStream with over 1024 peers. For a comparison of tree-based protocols with larger group sizes see [7].

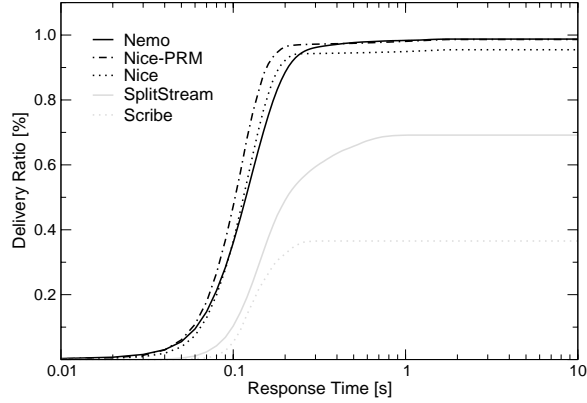


Figure 3: Response Time CDF (512 end hosts, low failure rate).

Table 2: Scalability (low failure rate).

Size	Delivery ratio					Duplicate packets				
	Nemo	Nice-PRM	Nice	SplitStream	Scribe	Nemo	Nice-PRM	Nice	SplitStream	Scribe
128	0.954	0.970	0.960	0.621	0.725	2.12	3.66	1.24	0.0	0.0
256	0.997	0.995	0.958	0.544	0.566	3.87	6.76	3.26	0.0	0.0
512	0.988	0.986	0.955	0.692	0.366	2.74	14.46	2.35	5.56	9.97
1024	0.993	0.989	0.883	0.443	0.157	6.78	23.57	19.63	0.0	0.0

protocol, Scribe, despite the latter’s constant drop in delivery ratio. The implicit multicast schemes are also able maintain the highest delivery ratio despite the increasing group sizes. While offering the highest delivery ratio, implicit multicast schemes exhibit the lowest total data traffic, with a reduction up to 70% when compared with the DHT-based approaches, making them the most efficient of the evaluated approaches.

Wide-area experimental results are summarized in Table 4. The delivery ratios are slightly lower than as reported in the simulation results, with packet losses and resource utilization accounting for the increased loss rate. While the high delivery ratio of tree-based protocols are comparable, Nemo offers the best response time when compared with Nice-PRM (and even in comparison with the non-resilient Nice). The larger number of intermediate nodes visited by a routed message under DHT-based protocols (with the correspondingly extra queuing and processing delays), explain the higher response time observed.

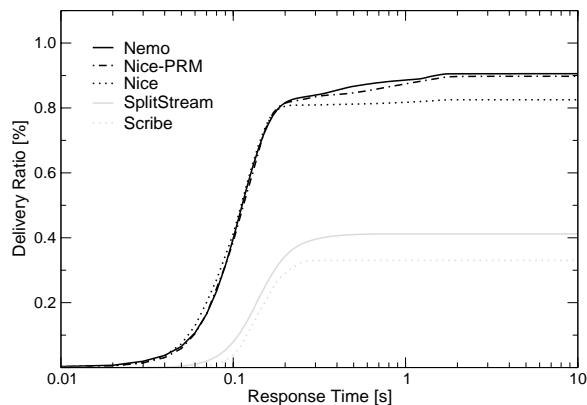


Figure 4: Response Time CDF (512 end hosts, high failure rate).

Table 3: Scalability (low failure rate).

Size	Data Traffic [MBps]					Control Traffic [MBps]				
	Nemo	Nice-PRM	Nice	SplitStream	Scribe	Nemo	Nice-PRM	Nice	SplitStream	Scribe
128	17.59	17.90	17.45	83.23	97.93	2.58	3.11	2.55	0.84	0.61
256	36.21	36.77	35.31	150.72	145.17	5.36	7.30	6.15	1.89	1.14
512	71.15	74.38	70.07	373.90	217.33	10.42	13.22	10.84	2.55	3.94
1024	146.99	150.19	131.71	477.04	170.60	37.12	7.77	119.76	5.30	4.84

In the evaluated DHT-based multicast systems, publishers route their message first to a well known source (with node id closest to the session id) for distribution. These extra hops, beyond adding extra queuing and processing delay, increase potential message losses. Also, in the presence of congestion collapse, the routing to the root of the distribution tree will likely fail, further reducing the actual throughput. This is observed for the high failure case where Scribe delivers no packets at all.

Figure 5 and 6 show the CDF of the response time for the wide-area evaluation under low and high failure rates. While the response time is higher in the wide-area experimentation than in the simulation, both sets of results show a similar trend. We see that Nemo has an advantage for earlier delivery of all packets when compared to the other implicit approaches. Through load balancing introduced by co-leaders, Nemo reduces the queuing and processing delay for each packet. Because these two costs become more visible in the wide-area environment, the

Table 4: Wide-Area Results (PlanetLab, 50 end hosts).

Protocol	Response time		Delivery ratio		Duplicate packets	
	Mean	Std	Mean	Std	Mean	Std
Low Failure Rate						
Nemo	0.734	2.044	0.900	0.062	2.73	5.79
Nice-PRM	1.104	2.577	0.884	0.060	3.84	5.45
Nice	1.112	2.047	0.889	0.079	1.67	4.26
SplitStream	33.481	51.003	0.203	0.087	10.12	13.15
Scribe	19.394	27.322	0.115	0.046	0.04	0.35
High Failure Rate						
Nemo	0.654	1.054	0.837	0.069	2.52	5.28
Nice-PRM	1.022	1.626	0.803	0.086	3.37	5.14
Nice	1.017	1.800	0.820	0.086	1.95	4.49
SplitStream	20.906	5.907	0.042	0.035	0.00	0.07
Scribe	N/A	N/A	0.000	0.000	0.00	0.00

benefit from load balancing is most visible here. DHT-based multicast protocols, on the other hand, have high response times for all packets indicating that congestion collapse takes place already at a low failure rate under real world conditions.

6 Discussion

We have evaluated three alternative techniques for higher resilience: Nemo’s co-leaders, Nice-PRM’s probabilistic forwarding and SplitStream’s multi-tree approach. While all of the evaluated techniques are able to substantially increase the resilience of their base protocol, tree-based protocols achieve the highest delivery ratio under the different failures scenarios in both simulation and wide-area environments. Exercising the protocols under continuous failures revealed a fundamental problem in the DHT-based protocols, which failed to repair the mesh quickly enough.

Similar findings were reported by Rhea et al. [33], where the authors found congestion collapse as a major performance limitation of Pastry when exercised under continuous peer failures. While routing through intermediate nodes improves resilience in the presence of network failures. However, the larger number of hops that are regularly part of DHT-based protocols’ routes, introduces a number of additional threats to the overall resilience of the system. Extra nodes are also extra points of failure. Also, routing through these additional nodes increases their load, lowering the total available system capacity and puts the system potentially closer to congestion collapse. While Scribe and SplitStream suffer in part from Pastry’s low resilience, alternate DHT protocols such as Bamboo [33] may fix some of these

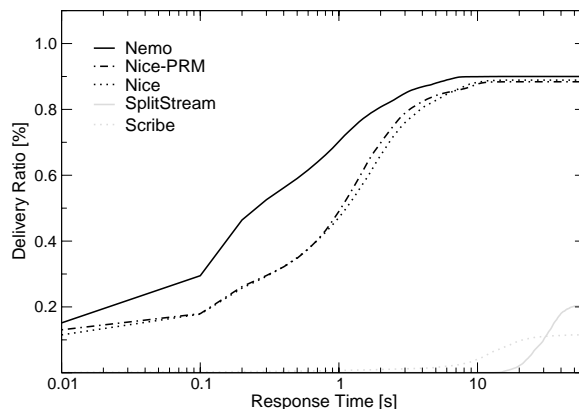


Figure 5: Response Time CDF (PlanetLab, 50 end hosts, low failure rate). This graph shows a representative run of each protocol normalized to the average delivery ratio achieved over all runs.

issues.

In sum, protocols built from the outset with resilience in mind, such as Nemo and Bamboo, seem to provide higher resilience with relatively low additional costs; while systems with “added” resilience (such as Nice-PRM) often result on significant additional costs to those of their base protocol. When resilience is built into a protocol from the ground up, there are no inherited costs.

7 Related Work

To the best of our knowledge, this is the first direct comparison of DHT-based and tree-based end-system multicast schemes using both simulation and wide-area experimentation.

Banerjee et al. [3] describe and analytically compare a set of proposed application layer protocols including DHT-based and tree-based techniques. Castro et al. [11] contrast CAN-style versus Pastry-style overlay networks using multicast communication workloads running on an identical simulation infrastructure. They conclude that the DHT-based, tree-building approach achieves lower delay and overhead than flooding regardless of the underlying DHT system, and that multicast trees built on Pastry provide higher performance than those using CAN [32]. Rhea et al. [33] show, through an emulation-based evaluation, the inability of current DHT implementation (specifically Pastry and Chord) to handle degrees of churn experienced in currently deployed systems. The authors propose Bamboo, a

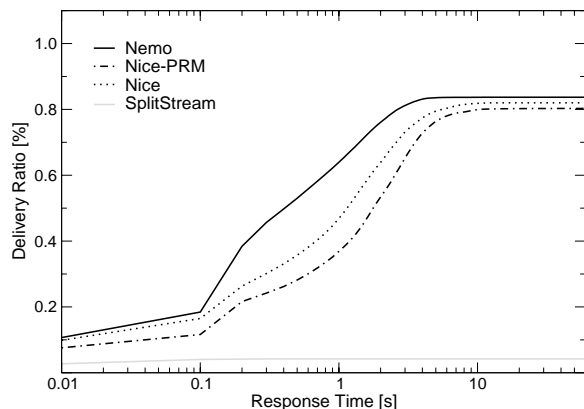


Figure 6: Response Time CDF (PlanetLab, 50 end hosts, high failure rate). This graph shows a representative run of each protocol normalized to the average delivery ratio achieved over all runs.

more resilient DHT-based system designed considering identified factors that determine DHT performance under transient populations.

Our experimental study compares the performance of five tree-based and DHT-based overlay multicast systems through simulation and wide-area experimentation in the PlanetLab testbed. The resilience and overhead of the different protocols is evaluated under a continuous stream of failures at two different rates obtained from the literature.

8 Conclusions

We have evaluated different techniques for resilient peer-to-peer multicast and analyzed their effectiveness in the context of their non-resilient alternatives. Protocols designed from the outset with resilience in mind feature comparatively high degrees of resilience under churn at significantly lower costs. While multiple trees achieve promising performance gains (up to 90% increase in delivery ratio), single-tree and cross-link redundancy exhibit the highest degree of resilience among the evaluated protocols. In wide-area evaluation, single-tree redundancy, e.g. Nemo, showed significant savings in response time (of up to 34%) when compared to cross-link redundancy and even in contrast to the non-resilient baseline system (Nice).

Implicit approaches, where the control tree is also used for data forwarding, have demonstrated scalable resilience. DHT-based implementations, on the other

hand, suffer congestion collapses already at the lowest of two the failures rates employed. Lessons from robust DHT protocols, such as Bamboo [33], that specifically address the resilience issue may prove useful in the design of other DHT, churn-resilient systems. We are in the process of porting Bamboo to our evaluation environment.

Acknowledgments

We would like to thank Karsten Schwan and Peter Dinda, who kindly loaned us their equipment for some of our experiments. We are also grateful to the the FreePastry group at Rice University for making their code available, especially to Alan Mislove for answering our many questions concerning FreePastry, and to Jeanine M. Casler, Yi Qiao, Bin Lin and Ashish Gupta for their helpful comments on early drafts of this paper.

References

- [1] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. of the 18th ACM SOSP*, October 2001.
- [2] T. Anderson, S. Shenker, I. Sotica, and D. Wetherall. Design guidelines for robust Internet protocols. In *Proc. of HotNets-I*, October 2002.
- [3] S. Banerjee and B. Bhattacharjee. A comparative study of application layer multicast protocols, 2002. Submitted for review.
- [4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proc. of ACM SIGCOMM*, August 2002.
- [5] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. Resilient multicast using overlays. In *Proc. of ACM SIGMETRICS*, June 2003.
- [6] M. Bawa, H. Deshpande, and H. Garcia-Molina. Transience of peers & streaming media. In *Proc. of HotNets-I*, October 2002.
- [7] S. Birrer and F. E. Bustamante. Nemo - resilient peer-to-peer multicast without the cost. Tech. Report NWU-CS-04-36, Northwestern U., May 2004.
- [8] S. Birrer and F. E. Bustamante. Resilient peer-to-peer multicast without the cost. In *Proc. of MMCN*, January 2005.

- [9] F. E. Bustamante and Y. Qiao. Friendships that last: Peer lifespan and its role in P2P protocols. In *Proc. of IWCW*, October 2003.
- [10] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: High-bandwidth multicast in cooperative environments. In *Proc. of the 19th ACM SOSP*, October 2003.
- [11] M. Castro, M. B. Jones, A.-M. Kermarrec, A. Rowstron, M. Theimer, H. Wang, and A. Wolman. An evaluation of scalable application-level multicast built using peer-to-peer overlays. In *Proc. of IEEE INFOCOM*, March 2003.
- [12] M. Castro, A. Rowstron, A.-M. Kermarrec, and P. Druschel. SCRIBE: A large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communication*, 20(8), 2002.
- [13] Y. Chawathe. *Scattercast: an architecture for Internet broadcast distribution as an infrastructure service*. Ph.D. Thesis, U. of California, Berkeley, CA, Fall 2000.
- [14] Y.-H. Chu, A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang. Early experience with an Internet broadcast system based on overlay multicast. In *Proc. of USENIX ATC*, June 2004.
- [15] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *IEEE Journal on Selected Areas in Communication*, 20(8), October 2002.
- [16] B. Cohen. BitTorrent. bitconjurer.org/BitTorrent/, 2001. File distribution.
- [17] Y. K. Dalal and R. M. Metcalfe. Reverse path forwarding of broadcast packets. *Communication of the ACM*, 21(12):1040–1048, 1978.
- [18] S. E. Deering. Multicast routing in internetworks and extended LANs. In *Proc. of ACM SIGCOMM*, August 1988.
- [19] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network*, 14(1), January/February 2000.
- [20] P. Druschel, E. Engineer, R. Gil, J. Hoye, Y. C. Hu, S. Iyer, A. Ladd, A. Misllove, A. Nandi, A. Post, C. Reis, A. Singh, and R. Zhang. FreePastry 1.3.2. freepastry.rice.edu/, February 2004. FreePastry is a modular, open source implementation of the Pastry p2p routing and location substrate.

- [21] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *Proc. of ACM SIGCOMM*, August 2000.
- [22] P. Francis. Yoid: Extending the Internet multicast architecture. <http://www.aciri.org/yoid>, April 2000.
- [23] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling and analysis of a peer-to-peer file-sharing workload. In *Proc. of ACM SOSP*, December 2003.
- [24] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O’Toole Jr. Overcast: Reliable multicasting with an overlay network. In *Proc. of the 4th USENIX OSDI*, October 2000.
- [25] C. Jin, Q. Chen, and S. Jamin. Inet: Internet topology generator. Technical Report CSE-TR-433-00, U. of Michigan, Ann Arbor, MI, 2000.
- [26] D. Kostić, A. R. adn Jeannie Albrecht, and A. Vahdat. Bullet: High bandwidth data dissemination using an overlay mesh. In *Proc. of the 19th ACM SOSP*, October 2003.
- [27] R. Mahajan, M. Castro, and A. Rowstron. Controlling the cost of reliability in peer-to-peer overlays. In *Proc. of IPTPS*, February 2003.
- [28] D. L. Mills. Improving algorithms for synchronizing computer network clocks. In *Proc. of ACM SIGCOMM*, August 1994.
- [29] V. N. Padmanabhan, H. J. Wang, and P. A. Chou. Resilient peer-to-peer streaming. In *Proc. of IEEE ICNP*, 2003.
- [30] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proc. of USENIX USITS*, March 2001.
- [31] PlanetLab Consortium. PlanetLab. <http://www.planet-lab.org>.
- [32] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *Proc. of NGC*, November 2001.
- [33] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *Proc. of USENIX ATC*, December 2004.
- [34] A. Rowstron and P. Drushel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM Middleware*, November 2001.

- [35] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. In *ACM SIGCOMM Internet Measurement Workshop*, November 2002.
- [36] A. C. Snoeren, K. Conley, and D. K. Gifford. Mesh-based content routing using xml. In *Proc. of the 18th ACM SOSp*, October 2001.
- [37] I. Stoica, R. Morris, D. Krager, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proc. of ACM SIGCOMM*, August 2001.
- [38] D. A. Tran, K. A. Hua, and T. Do. ZIGZAG: An efficient peer-to-peer scheme for media streaming. In *Proc. of IEEE INFOCOM*, April 2003.
- [39] K.-F. S. Wong, S. G. Chan, W.-C. Wong, Q. Zhang, W.-W. Zhu, and Y.-Q. Zhang. Lateral error recovery for application-level multicast. In *Proc. of IEEE INFOCOM*, March 2004.
- [40] J. Xu, Z. Kalbarczyk, and R. K. Iyer. Networked Windows NT system field failure data analysis. In *Proc. of PRDC*, December 1999.
- [41] M. Yang and Z. Fei. A proactive approach to reconstructing overlay multicast trees. In *Proc. of IEEE INFOCOM*, March 2004.
- [42] B. Y. Zhao, J. Kubiawicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Tech. Report UCP//CSD-01-1141, U.C. Berkeley, April 2001.
- [43] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiawicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proc. of NOSSDAV*, June 2001.