Using Epidemic Techniques for Building Ultra-Scalable Reliable Communication Systems.

Werner Vogels, Robbert van Renesse, Ken Birman
Dept. of Computer Science, Cornell University†

Abstract

Building very large computing systems is extremely challenging, given the lack of scalable communication technologies. This threatens a new generation of mission-critical but very large computing systems. Fortunately, a new generation of "gossip-based" or epidemic protocols can overcome scalability problems, offering security and reliability even in the most demanding settings. Epidemic protocols emulate the spread of an infection in a crowded population, and are both reliable and stable under forms of stress that will disable most traditional protocols. Cornell University's Spinglass project is developing a new generation of epidemic-based technology for secure, reliable large-scale collaboration and soft real-time communications – even over global networks.

1 Introduction

Distributed computing will be central to advances in a broad range of critical applications, including intelligence information systems, military command and control, air traffic control, electric power grid management, telecommunications, and a vast array of web-based commercial and government applications. Indeed, a massive rollout of such systems is already underway. Yet while impressive capabilities have been easy to develop and demonstrate in small-scale settings, once deployed these systems often stumble badly.

Software that runs securely and reliably in small-scale mockups may lose those properties as numbers of users, the size of the network and transaction processing rates all increase. Whereas small networks are well behaved, any sufficiently large network behaves like the public Internet, exhibiting disruptive overloads and routing changes, periods of poor connectivity and throughput instability. Failures rise in frequency simply because the numbers of participating components are larger. A scalable technology must ride out such forms of infrastructure instability.

Our studies reveal that very few existing technologies have the necessary properties. Most, including the most prevalent commercial software, exhibit scalability problems when subjected to even modest stress. This finding reveals an imminent (and growing) threat to the full spectrum of emergent mission-critical computing systems. If we can't solve the scalability problem, and develop a methodology yielding applications that remain secure and robust even when failures occur – indeed, even under attack, or during denial-of-service episodes – the very technologies that hold the greatest promise for major advances will prove to be the Achilles Heel of a future generation of mission-critical military and public-sector enterprises.

The Spinglass project is working to overcome scalability barriers, starting with an idea that was first proposed in the context of replicated database systems. These systems employed what were called "epidemic-style" or "gossip" update algorithms, whereby sites periodically compare their states and reconcile inconsistencies, using a randomized mechanism for deciding when and with whom each participant will gossip. Traditionally, database systems used gossip protocols at low speeds. Our work employs gossip at very high speeds, yielding a new generation of protocols that have an unusual style of probabilistic reliability guarantees –guarantees of scalability, performance, stability of throughput even under stress, and remarkable scalability. These properties hold even on Internet-like platforms. Gossip protocols lend themselves to theoretical analysis, making it possible to predict their behavior with high confidence. However, the focus of our work at Cornell is mostly practical: we are using gossip, together with other more traditional mechanisms, to develop new generations of scalable communications software and middleware for a wide variety of settings.

Spinglass treats security and authentication as primary considerations, and our software is designed to coexist with modern firewalls and intrusion-detection solutions. Indeed, one potential application is to support improved intrusion detection mechanisms: a gossip based intrusion detection system has the potential to overcome scalability and fragility problems seen in with more standard client-server solutions, and avoids the single-point-of-failure concerns associated with such solutions. Our objective in the present paper is to review the scalability problem, and to summarize our approach to solving it. The need for brevity limits the technical detail here, but other publications are available for the interested reader who wishes to learn more.

† This research is supported by DARPA/ONR under contract N0014-96-1-10014, by the National Science Foundation under Grant No. EIA 97-03470 and by grants from Microsoft Research.

The authors can be reached at 4105A Upson Hall, Cornell University, Ithaca, NY 14853, fax 607-2554428, E-mail: *vogels@cs.cornell.edu.*

2 Scalability Problems in Current Communication Systems

The scalability of distributed protocols and systems is a major determinant of success in demanding systems. For example, consider the recent field-test of the Navy's Cooperative Engagement Capability (CEC). During the period Sept. 13-27, this system (which offers an over-the-horizon cooperative targeting capability for naval battleships) was subjected to a very modest stress test. The value of this system depends upon timely identification of threats and rapid decisions about which ship will respond to which threat: threats may be incoming missiles moving at several times the speed of sound. This translates to internal deadlines of about a second for the communication subsystem, which had been demonstrated easily capable of meeting the requirements under laboratory conditions with small numbers of participating computing systems. Yet under load, when even small numbers of battleships were added to the system, the underlying Data Distribution System (DDS) became unstable, failing outright, or delivering data after much more than the one-second threshold. (Defense News, October 16, 2000). The result was that the CEC basically failed – and this under rather benign conditions in which the only variable that changed was the number of participants.

This paper focuses on scalability of distributed protocols providing some form of guaranteed reliability when communication among multiple participants is required. Use of these reliable group communication (multicast) protocols is very popular in distributed systems, as there is a natural match between the group paradigm and the way large distributed systems are structured. These protocols allow systems to be built in pure peer-to-peer manner, removing the need for centralized servers, removing one of the bottlenecks in system scalability.

Traditional reliable multicast protocols all exhibit severe scaling problems, when applied in large Internet style settings. Even though some of these protocols appeared to be very promising in terms of scalability they all failed to operate as soon as the network conditions because the ideal. In general the performance analyses of the protocols focuses on two extreme cases: performance of the protocol under ideal conditions, when nothing goes wrong, and the disruptive impact of a failure. Reality forces use to take a look at these protocols from a different perspective: what happens to these protocols under mundane transient problems, such as network or processor scheduling delays and brief periods of packet loss. One would expect that reliable protocols would ride out such events, but we find that this is rarely the case, particularly if we look at the impact of a disruptive event as a function of scale. On the contrary, reliable protocols degrade dramatically under this type of mundane stress, a phenomenon attributable to low-probability events that become both more likely and more costly as the scale of the system grows.

Because of the need for brevity, we'll limit ourselves to a summary of our finding with respect to the growth rate of disruptive overheads for a number of widely used multicast protocols. Elsewhere [BGR01], we present a more detailed analysis of a variety of scenarios, modeled after the work of Gray *et. al.* [GHOS96], where a similar conclusion is reached with respect to database scalability. It is clear that scalability represents a widespread problem affecting a broad range of technologies and systems.

2.1 Common Problems

When subjected to transient problem that are related to scaling the environment, there are important categories of problems that appear:

- *Throughput Instability*. All of the protocols that implement reliable group communication are subject a breakdown of message throughput as soon as one or more members experience perturbation. A single slow receiver, which can be cause by CPU overhead or some localized message loss, will eventually have an impact on the throughput to the overall group. The results in [BGR01] show that this impact is even more dramatic and happens more rapidly if we scale up the system, making the protocol stability extremely volatile under even moderate scaling conditions.
- *Micropartitions.* In reaction to the throughput instability problem, designers often go for the approach to as soon as possible to remove the trouble-causing member from the group by using more aggressive failure detection mechanisms. However when scaling the system to moderate Internet environments, one quickly discovers that this has the adverse effect that transient network problems, which occur frequently, frequently trigger incorrect failure-suspicions. Erroneous failure decisions involve a particularly costly "leave/rejoin" events, where the overall system constantly needs to reconfigure itself. We will term this a *micropartitioning* of the group, because a non-crashed member effectively becomes partitioned away from the group and later the partition (of size one) must remerge. In effect, by setting failure detection parameters more and more aggressively while scaling the system up, we approach a state in which the group may continuously experience micropartitions, a phenomenon akin to thrashing.

  Costs associated with micropartitions rise in frequency with the square of the size of the group. This is because the frequency of mistakes is at least linear in the size of the group, and the cost of a membership change is also linear in the group size: a quadratic effect.
- *Convoys*. An obvious response to the scalability problems just presented is to structure large systems hierarchically, as trees of participant groups. Unfortunately this option is also limited by disruptive random events, albeit in a different way. Experimentation has shown that such a tree structure, when confronted with moderate network instability, exhibits an amplification of the burstiness of the message traffic. Even though messages enter the

system at a steady rate, the reliability and buffer strategies at each of the intermediate nodes in the trees have compressing effects on messages rates, especially when confronted with message loss. The term "convoy" has been used by the database community to describe this phenomenon, which is also well known to the packet routing community.

- *Request and Retransmission storms*. Frequently protocols resort to using scoped IP multicast techniques for finding other local participants to request retransmission of messages. Although this avoids sending requests back to the send, the message do read *all* local group members, often triggering storms of retransmission, or even storms of requests, if multiple receivers are experiencing similar message loss patterns.

2.2 Unsuccessful Solutions

Some reliable multicast protocols have been successful at larger scale, but only by very much limiting the functionality of the protocols. Techniques the developers have resorted to achieve scalability are:

- *Anonymous Membership*. The protocol basically streams out information to whoever wants to receive messages, without any notion of admission control, failure detection or session state management.
- *Single Sender Groups*. These protocols build a single dissemination tree per sender where each node is a potential receiver who cooperates in localized retransmission schemes. Groups with multiple senders, which are very common in distributed systems, are treated as multiple groups with single sender, requiring an explosion of state managed at each participant, and making it impossible to correlate messages from different senders.
- *Infinite Retransmission Buffers*. One of the more complex problems in multi-sender protocols is the management of the messages stored for retransmission. The messages can be released once the system is certain that no retransmission requests can arrive any more. Given that many protocols have no knowledge about the receivers, this certainty can never be achieved and they resort to a method called application level framing to require the application to reconstruct message for retransmission. This was applicable to some multi-user collaboration tools, but was unusable for the majority of distributed systems.
- *Complete Lack of Security*. In most protocols that combine application level framing with localized retransmission (others than the sender can retransmit lost messages), it is impossible guarantee secure communication, as nodes that retransmit can not sign the message with senders key. The original messages are not kept in lack of a garbage collection protocol, and the retransmission node cannot sign it with its own key as they are anonymous, and as such the receiver has no mechanism for checking the signature.

These techniques are unacceptable if one wants to build robust, reliable, secure distributed systems that can be the basis for the mission critical enterprise systems.

But the picture is not entirely bleak. After presenting these arguments, we shift attention to an new class of protocols based on an idea from NNTP, the gossip-based algorithm used to propagate "news" in the Internet, and Clearinghouse, the database replication technology developed at Xerox Parc in the 1980's. These turn out to be scalable under the same style of analysis that predicts poor scalability for their non-gossip counterparts.

3 Epidemic Techniques for Scalable Protocols

Not all protocols suffer the behavior seen in these reliable mechanisms. In the class of reliable multicast protocols, *Bimodal Multicast*, a protocol reported in [BHO99], scales quite well and easily rides out the same phenomena that cause problems with these other approaches to reliability and scalability.

Bimodal multicast is a *gossip-based* protocol that somewhat resembles the old NNTP protocol (employed by network news servers), but running at much higher speeds. The protocol has two sub-protocols. One of them is an unreliable data distribution protocol similar to IP multicast, or based on IP multicast when available. Upon arrival, a message enters the receiver's *message buffer*. Messages are delivered to the application layer in FIFO order, and are garbage collected out of the message buffer after some period of time.

The second sub-protocol is used to repair gaps in the message delivery record, and operates as follows. Each process in the system maintains a list containing some random subset of the full system membership. In practice, we weight this list to contain primarily processes from close by – processes accessible over low-latency links – but these details go beyond the scope of the current paper.

At some rate (but not synchronized across the system) each participant selects one of the processes in its membership list at random and sends it a *digest* of its current message buffer contents. This digest would normally just list messages available in the buffer: "messages 5-11 and 13 from sender *s*, …" for example. Upon receipt of a gossip message, a process compares the list of messages in the digest with its own message buffer contents. Depending upon the configuration of the protocol, a process may *pull* missing messages from the sender of the gossip by sending a retransmission *solicitiation*, or may *push* messages to the sender by sending unsolicited retransmissions of messages apparently missing from that process.

This simplified description omits a number of important optimizations to the protocol. In practice, we use gossip not just for multicast reliability, but also to track system membership and perform failure detection based on it [RMH98][GT92]. We sometimes use unreliable multicast with a regional TTL value instead of unicast, in situations

where it is likely that multiple processes are missing copies of the message. A weighting scheme is employed to balance loads on links: gossip is done primarily to nearby processes over low-latency links and rarely to remote processes, over costly links that may share individual routers [XB00]. The protocol switches between gossip pull and gossip push, using the former for "young" messages and the latter for "old" ones. Finally, we don't actually buffer every message at every process; a hashing scheme is used to spread the buffering load around the system, with the effect that the average message is buffered at enough processes to guarantee reliability, but the average buffering load on a participant decreases with increasing system size.

Bimodal Multicast has a number of important properties: the protocol imposes constant loads on participants, is extremely simple to implement and rather inexpensive to run. More important from the perspective of this paper, however, the protocol overcomes the problems cited earlier for other scalable protocols. Bimodal Multicast has tunable reliability that can be matched to the needs of the application (reliability is increased by increasing the length of time before a message is garbage collected, but this also causes buffering and I/O costs to rise). The protocol gives very steady data delivery rates with predictable, low, variability in throughput. For real-time applications, this can be extremely useful. And the protocol imposes constant loads on links and routers (if configured correctly), which avoids network overload as a system scales up. All of these characteristics are preserved as the size of the system increases.

The reliability guarantees of the protocol are midway between the very strong guarantees of virtual synchrony and the much weaker best-effort guarantees of traditional reliable mutlicast protocols. We won't digress into a detailed discussion of the nature of these guarantees, which are probabilistic, but it is interesting to note that the behavior of Bimodal Multicast is predictable from certain simple properties of the network on which it runs. Moreover, the network information needed is robust in networks like the Internet, where many statistics have heavy-tailed distributions with infinite variance. This is because gossip protocols tend to be driven by successful message exchanges and hence by the "good" quartile or perhaps half of network statistics. In contrast, traditional protocols often include round-trip estimates of the mean latency or mean throughput between nodes: estimates that are problematic in the Internet where many statistical distributions are heavy-tailed and hence have ill-defined means and very large variances.

4 Overcoming Limitation to Scale

We can generalize from the phenomena enumerated above. Distilling these down to their simplest form, and elaborating slightly:

- With the exception of the epidemic protocols, each of reliability model involves a costly, but infrequent fault-recovery mechanism:
  - Virtual synchrony based protocols employ flow control, failure detection and membership-change protocols; when incorrectly triggered, the cost is proportional to the size of the group.
  - Local repair based protocols have a solicitation and retransmission mechanism that involves multicasts; when a duplicate solicitation or retransmission occurs, all participants process and transmit extra messages.
  - FEC-based reliability mechanisms try to reduce retransmission requests to the sender by encoding redundancy in the data stream. As the group size grows, however, either the average multicast path length increases, hence so too the risk of a multi-packet loss. The sender will see increasingly many retransmission requests (consuming a scarce resource), or the redundancy of the stream itself must be increased (resulting in a bandwidth degradation and a system-wide impact).
- Again with the exception of the epidemic protocols, the mechanisms we've reviewed are potentially at risk from convoy-like behaviors. Even if data is injected into a network at a constant rate, as it spreads through the network, router scheduling delays and link congestion can make the communication load bursty. Under extreme condition this behavior can even trigger message loss at the end nodes. To smoothen burstiness of messages from multiple senders one needs a global view of the system, which most reliable protocols have found impossible to implement. Epidemic techniques however are ideal to implement this global state sharing and allow the overall system to gracefully adapt to changes in the network.
- Many protocols depend upon configuration mechanisms that are sensitive to network routing and topology. Over time, network routing can change in ways that take the protocol increasingly far from optimal, in which case the probabilistic mechanisms used to recover from failures can seem increasingly expensive. Periodic reconfigurations, the obvious remedy, introduce a disruptive system-wide cost.

In contrast, the epidemic mechanisms used in NNTP, the Xerox Clearinghouse system, and the Bimodal Multicast protocol appear to scale without these kinds of problems. Throughput is stable (at least, if measured over sufficiently long periods of time – gossip protocols can be rather *unstable* if metered on a short time scale). Overheads are flat and predictable, and can be balanced with information about network topology, so that links and routers won't become overloaded. And, the levels of reliability achieved are very high – indeed, potentially as high as those of the protocols purporting to offer stronger guarantees, if one considers the possibility that such protocols sometimes reconfigure themselves, incorrectly excluding a process as "faulty" when it may actually merely be the victim of bad luck.

Probabilistic guarantees may sound like a contradiction in terms, because one's intuition suggests that anything but an

absolute reliability guarantee would be the equivalent of no reliability at all. Our work suggests that this is not at all the case. First, it is possible to design mechanisms that have stronger guarantees, such as virtual synchrony, and yet reside in an end-to-end manner over the basic network architecture afforded by our gossip infrastructure.

An important observation, contributing to the overall success of this approach, is also that the epidemic tools exploit scalability, and as such turn scale into an advantage instead of a problem that must be overcome.

We are also finding ways of embedding probabilistic guarantees directly into useful tools that applications might find valuable in their own terms, without trying to superimpose some stronger (arguably, less natural) reliability abstraction over the basic properties of the protocol.

For example, our Astrolabe technology offers probabilistic guarantees for data managed in a scalable table. We believe that there are a tremendous number of ways this software can be used, as is, without resorting to end-to-end mechanisms that would try to strengthen these basic guarantees. Similarly, many data dissemination systems can operate directly over Bimodal Multicast: the properties of the protocol are well matched to the needs of systems that do require a degree of reliability, but can overcome *bounded* rates of error. The term bounded is the key here: unlike a traditional unreliable network mechanism that can behave arbitrarily badly if luck turns against the user, Bimodal Multicast overcomes all but the most severe outages and behaves in a predictable manner at all times.

5 Four Probabilistic Tools

Earlier we noted that the focus of the Spinglass project is on practical software tools that can really be used. In this section and the one that follows, we first review the tools we are currently developing, and then describe some applications which we view as especially promising early targets for the technology.

5.1 Bimodal Multicast

As described above, Bimodal Multicast is a one-to-many communications mechanism that achieves tunable, probabilistically reliable data delivery. The probability of successful outcomes, where all operational processes receive every multicast, can be made arbitrarily high simply by adjusting the parameters governing the frequency with which participants gossip and the length of time that they buffer copies of received multicasts. In ongoing work at Cornell University, we are exploring issues such as operation in wide-area networks with firewalls surrounding secured enclaves, using formal tools to characterize the conditions that a network must have in order to run this protocol, and understanding the kinds of network problems that Bimodal Multicast is capable of overcoming.

5.2 Astrolabe

Astrolabe [R00] is a system built using the same gossip mechanisms employed in Bimodal Multicast, but in support of a completely different data model. The basic idea of Astrolabe is to support a distributed shared memory in the form of a hierarchical table. The leaves of the hierarchy are regional tables in which there is one row per participating computer or application program, and where the columns contain application-defined data (this could be something small, like an indication of the security level of the machine or the version number of a program running on it or a load, or something large, like an XML encoding of an image). Within a region, all participants can see the entire regional table but each can only update its own row.

Higher levels of the hierarchy are formed using what we call condensation function. A condensation function is a computation on a column of a regional table that reduces the contents of that column to a single value. For example, *minimum* could be used as a condensation function for a column reporting load. The contents of a higher-level table will be one row for each of its child regions, with columns defined according to the condensation function. While a region only has direct access to its own regional table, all regions can access all of the higher-level tables in the hierarchy.

Astrolabe offers a powerful technology for scalable system management and resource discovery. In modern computing systems, simply knowing where data can be found or knowing versions of software and configuration information for other machines is a critical and yet poorly supported functionality. Astrolabe automates this job and does so in a manner that is scalable, has predictable delays (they grow slowly with system size), constant overheads, and remarkable stability.

5.3 Gravitational Gossip

This variation on the Bimodal Multicast protocol is designed to support large numbers of subgroups within a single network [JHB00]. Subgroups are a traditionally important problem in group communication systems, since one often uses such systems to support publish-subscribe styles of communication, where large numbers of communication groups can arise. With this new protocol, we are able to superimpose large numbers of subgroups on a large Bimodal Multicast group, and can arrange that each group member will receive just the data it desires plus a constant overhead. Moreover (this is the "gravitational" aspect) members of a subgroup can specify a quality rating. A member that wants to receive 100% of the data in a group can do so, paying the full cost for all multicasts in the group, but if a member only needs part of the data – say, 50% of the sensor readings or 20% of them – it can adjust its rating to correspond to its need, and the load associated with the protocol is reduced accordingly. Notice that we are deliberately reducing the reliability of the protocol to cut the load seen by processes with small rating values.

We like to visualize this protocol as emulating a gravitational well. In practice ratings can have any values desired, but these include values that give behavior like that of a gravity field. The idea, though, is that messages multicast within the base of the well and flow at full speed to other processes in the base. With some probability, these messages also ride up the walls of the well, but the steeper the wall, the less likely this is to occur, and a message that does ride up the wall is very likely to fall back towards the base. Processes residing on the wall thus see less load and receive just a percentage of the data items.

Obviously, gravitational gossip is only useful in settings where it is meaningful to take actions based on a randomly selected subset of sensor values. However, as discussed below, we know of a number of such applications.

5.4 Anonymous Gossip

This direction within our project applies gossip communication to nomadic wireless devices. We have a number of applications in mind, but started by looking at wireless multicast in so-called ad-hoc networks, which are common in military applications. Anonymous gossip is a technique for using gossip communication to improve the quality of existing ad-hoc multicast protocols. The idea is to take a multicast protocol (we've considered several, but worked most closely with AODV and the multicast layered over it, MAODV) and then superimpose a gossip repair mechanism similar to the one in Bimodal Multicast.

Where Anonymous Gossip departs from Bimodal Multicast is that in a wireless nomadic setting, little information is available about the identity of peers, since these can change rapidly. For example, a platoon of soldiers may fan out on a hillside, so that the network is always "fully connected" and yet the connectivity of any particular soldier's computer varies widely. The challenge is that in such a network, one has no idea with whom to gossip. Anonymous Gossip solves this problem by sending gossip messages that travel some distance over a randomly chosen path in the ad-hoc network. The eventual receiver replies to the sender.

The technique works extremely well, and we are now extending the basic protocol by looking at other metrics and at the use of gossip to maintain the basic routing infrastructure itself. At the same time, we are starting to look at application-level issues that arise in developing nomadic software to run over a gossip infrastructure.

References

[BGR01] Birman, K., Gupta, I, Van Renesse, R. Fighting Fire with Fire: Using Randomized Gossip to Overcome Probabilistic Limits to Scalability. In preparation, expected completion March 2001.

[BHO98] Birman, K., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M., Minsky, Y. Bimodal Multicast. Cornell University Dept. of Computer Science Technical Report, (Feb. 1998). Submitted to ACM TOCS.

[Bir97] Birman, K.P. *Building Secure and Reliable Network Applications.* Manning Publications and Prentice Hall, January 1997.

[Dem88] Demers, A. *et. al.* Epidemic Algorithms for Replicated Data Management. *Proceedings of the 6th Symposium on Principles of Distributed Computing (PODC),* Vancouver, Aug. 1987, 1-12.

[FJM95] Floyd, S., Jacobson, V., McCanne, S. Liu, C., and Zhang, L.. A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. In *Proc SIGCOMM `95,* Aug. 1995, Cambridge MA.

[GHOS96] J. Gray, P. Helland, P. O'Neil and D. Shasha. The dangers of replication and a solution. *Proceedings 1996 SIGMOD Conference*, June 1996.

[GT92] Richard Golding and Kim Taylor. Group Membership in the Epidemic Style. Technical report UCSC-CRL-92-13, University of California at Santa Cruz, May 1992.

[Guo98] Guo, K. *Scalable Message Stability Detection Protocols.* Ph.D. dissertation, Cornell University, 1998.

[JHB00] Jenkins, K., Hopkinson, K., and Birman, K. A Gossip Protocol for Subgroup Multicast. *Submitted to ICDCS Workshop on Reliable Group Communication.* Nov. 2000

[KB99] Kalantar, M and Birman, K. Causally Ordered Multicast: the Conservative Approach. *Proc. ICDCS 1999.* Austin, June 1999.

[Liu97] Liu, C. *Error Recovery in Scalable Reliable Multicast* (Ph.D. dissertation), University of Southern California, Dec 1997

[Luc98] Lucas, M.. *Efficient Data Distribution in Large-Scale Multicast Networks* (Ph.D. dissertation), Dept. of Computer Science, University of Virginia, May 1998

[R00] van Renesse, R. Astrolabe: A Scalable Resource Location Service. *Submitted to DISCEX-01*. December 2000.

[RMH98] van Renesse, R., Minsky Y, and Hayden, M. Gossip-Based Failure Detection Service. In Proc. of Middleware '98. England.

[SSL97] Sanjoy Paul, Sabnani, K., Lin, K. and Bhattacharyya, S. "Reliable Multicast Transport Protocol (RMTP)", IEEE Journal on Selected Areas in Communications, special issue on Network Support for Multipoint Communication, April 97, Vol 15, No. 3

[XB01] Xiao, Zhen and Birman, Ken. A Randomized Error Recovery Algorithm for Reliable Multicast. *Submitted to FTCS 2001*. December 2000.