QoS Enabled Multicast for Structured P2P Networks

Marc Brogle, Dragan Milic and Torsten Braun
Computer Networks and Distributed Systems
Institute of Computer Science and Applied Mathematics
University of Bern, Switzerland
Email: brogle|milic|braun@iam.unibe.ch
Telephone: +41 31 631 8668|5309|4994, Fax: +41 31 631 3261

Abstract— **In this paper we present a concept for providing QoS to multicast in structured P2P networks. We show on the example of Scribe / Pastry how to enforce QoS aware tree construction in a structured P2P network. We achieve this by modifying the ID assignment method of Pastry based on the QoS requirements of peers. As a result, the multicast tree holds the QoS (bandwidth) requirements on each of its end-to-end paths. We have evaluated the proposed concept by comparing default random Pastry ID assignment with our proposed method. The results of the evaluation show that using our method all end-to-end paths in the multicast tree fullfill the bandwidth QoS requirements, which is usually not the case for default Pastry.**

I. Introduction

In this paper we present a concept to enable Quality of Service (QoS) for multicast in the structured Peer-to-Peer (P2P) network Pastry [1] and the Application Level Multicast (ALM) infrastructure Scribe [2] running on-top of Pastry. This is achieved by replacing the random ID assignment of Pastry with a QoS requirements aware mechanism. We use this concept to enable QoS for IP Multicast over ALM in the European project called EuQoS [3], which aims to support end-to-end QoS over heterogeneous networks. EuQoS only supports unicast QoS between end-systems on the network level.

The remainder of the paper is structured as follows: In Section II we motivate the need for QoS enabled multicast Overlay Networks. In the same Section we present the use of such a network to provide QoS enabled IP Multicast services to end-systems in the EuQoS project. In Section III we present the ALM infrastructure Scribe and the P2P routing substrate Pastry. We present our proposal for modifying the Scribe / Pastry ID assignment to enforce the construction of QoS aware multicast trees in Section IV. We evaluate the QoS improvements by comparing default Scribe / Pastry ID assignment with our modifications in Section V. An outlook of future considerations and improvements can be found in Section VI. Finally, Section VII concludes this paper.

II. Motivation

Since IP Multicast is not widely deployed in the Internet, current solutions rely on ALM and Overlay Networks using unicast to provide multicast services. To make the use of ALM transparent to applications within the EuQoS [3] project, we provide an IP Multicast service to end-system applications by capturing the outgoing IP Multicast traffic from hosts and routing it using an ALM Overlay Network.

The goal of the EuQoS (End-to-end QoS support over heterogeneous networks) project is to resolve the required design issues presently associated with the delivery of end to end QoS service across heterogeneous networks. EuQoS only supports QoS for unicast connections between end-points. More details about the EuQoS architecture are found in [4].

The IP Multicast service, which is missing in the Internet today, is provided in the EuQoS project by the so called Multicast Middleware [5] feature. It also provides QoS guarantees for IP Multicast traffic by applying QoS mechanisms on the unicast links in the Overlay Network through which the multicast data is tunneled. The Multicast Middleware feature achieves this by capturing IP Multicast traffic on end-systems using a virtual network interface [6]. The captured IP Multicast packets are tunneled using a P2P Overlay Network based on Scribe / Pastry. The details of capturing and tunneling of IP Multicast data using the Multicast Middleware is shown in Fig. 1.

To satisfy the QoS requirements, our Multicast Middleware uses the EuQoS System to setup network level QoS for the unicast links of the Overlay Network. Since the QoS requirements of the end-systems within one IP Multicast group can be heterogeneous, it is necessary that the multicast tree is built in such way that the QoS requirements and capabilities of end-systems are considered.

In Section IV we describe the properties of a QoS aware multicast tree and how such trees can be constructed using Scribe / Pastry. The Multicast Middleware uses Pastry [1] as a

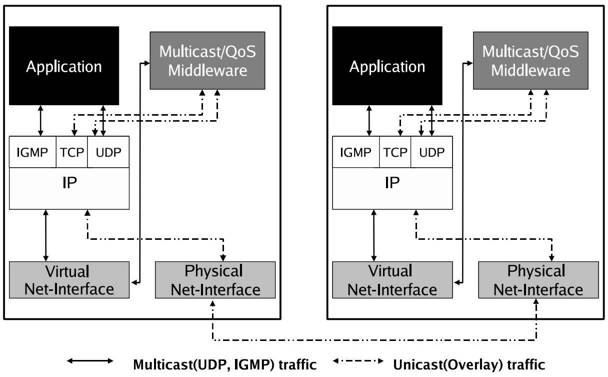

Fig. 1. EuQoS Multicast Middleware on end-systems

P2P routing substrate and Scribe [2] to handle multicast group management and data dissemination. Freepastry [7], an open source Java implementation of Scribe / Pastry, is used as the base implementation for the EuQoS Multicast Middleware. We use our modification presented in this paper to ensure that the construction of multicast trees in Scribe / Pastry is QoS aware. In the Multicast Middleware we create one dedicated Pastry network for each active IP Multicast group. This ensures that only end-systems, which are interested in receiving the multicast data of certain a group, are used to forward the traffic. This is also necessary to ensure fair charging.

III. Functional overview of Scribe / Pastry

Different P2P (P2P) architectures [1], [8]–[11] and ALM (ALM) systems [2], [12]–[17] have been presented over the past years. Also mechanisms on how to support QoS for specific P2P networks have been proposed in [18]–[21].

Pastry [1], [22] is a scalable distributed object location and routing substrate for P2P applications. In Pastry peers become a randomly chosen ID assigned when they join the P2P network. To route a message to a certain Peer, Pastry uses an efficient routing algorithm, which scales logarithmically with the number of peers in the P2P network. Pastry is self-organizing and completely decentralized. It also takes peer proximity information (in terms of end-to-end delay) into account to minimize the distance messages are traveling.

Scribe [2], [23] is an ALM infrastructure, which builds on top of Pastry. It supports large groups, is fault-tolerant and decentralized. Scribe only provides best-effort reliability guarantees. It balances the load on nodes to reduce delays and lower the link stress.

A. Pastry

Pastry is a P2P location and routing substrate. Each peer is identified by a 128 bit long ID. This ID is randomly chosen when joining a Pastry network. The choice of IDs is uniformly distributed. As a consequence, the choice of the ID neither takes locality nor QoS requirements into account.

Each Pastry peer has a routing table with the size of $(2^b - 1) * \lceil log_{2^b} N \rceil + l$ entries. The routing tables are organized into $\lceil log_{2^b} N \rceil$ rows with each $2^b - 1$ entries. The entries of row n of a peer's routing table point to other peers, which share the same first n digits of their ID with the peer itself, but the digit at position $n+1$ has one of the $2^b - 1$ possible values different from the digit at position $n + 1$ of the peer's ID. Each entry in the routing table consists of the destination's ID and its corresponding IP address. Additionally, each peer maintains a list of numerically closest peers (IDs and IP addresses) with $l/2$ entries for the larger and $l/2$ entries for the lower IDs. The locality is taken into account by choosing the closest peer (in terms the network latency) among candidates for an entry in the routing table. A message is routed to the closest neighbor found in the peer's routing table whose ID matches the messages destination ID prefix.

Routing uses less than $\lceil log_{2^b} N \rceil$ steps on average, where N is the amount of peers in the pastry Network and b is typically a parameter with the value 4. Pastry guarantees eventual delivery of a message unless $l/2$ or more peers with an adjacent ID fail at the same time, with l, an even number parameter, being typically 16. Fig. 2 shows a simplified example of how Pastry routing works. A message with the key *e8cd* is routed from a peer with ID *3d1f* to the peer *e8ca*, which is numerically closest to the message's key. On each hop from the source peer to the destination peer the message is sent to a peer whose ID matches more digits of the message key prefix as it did match on the hop before. For the first routing hop starting from peer *3d1f* the message is sent to peer *e2ce*, which shares the first digit *e* of the message key. On the second hop, the message is routed to the peer with ID *e831*, which shares the first two digits *e8*. Finally it is sent to peer *e8ca*, which is the peer closest to the message key and shares the first three digits *e8c* of the key.

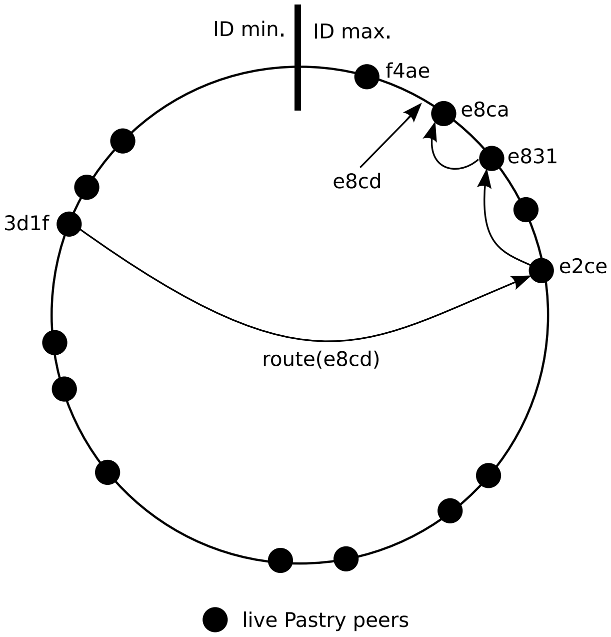

Fig. 2. Routing a message from peer *3d1f* with key *e8cd*

B. Scribe

Scribe [2] is a scalable ALM infrastructure, which runs on top of Pastry. Any Scribe node can join any multicast group (or topic in Scribe's terminology) at any time. For each topic, one node is designated to disseminate the topic's data in the Pastry network. This node, which is the root of the topic distribution tree, has the ID numerically closest to the topic's ID.

Scribe offers best-effort delivery of the multicast data without guaranteeing that the order of the packets is maintained. The multicast or topic tree is built using a scheme similar to reverse-path-forwarding. A Scribe node, subscribing to a certain topic, sends a join message for this topic-ID. This message is routed using Pastry's routing mechanism towards the topic's root. The next node to which the join message is routed to remembers that the node sending the join message is interested in data for this topic. If this intermediate node called a forwarder has not already joined this topic, it will itself send a join message to the same topic. This process is repeated until a node is reached that has already joined the topic or the root for the topic has been reached. The data dissemination within a topic is done from the root node of the topic towards the leave nodes by following all reverse-paths to the leaves. A side effect of this approach is that Scribe nodes forwarding messages for a certain topic are not necessarily subscribed to this topic.

IV. Making Scribe / Pastry QoS aware

In this paper, we only consider QoS classes, which have the following properties:

- There is a total order relation for all QoS classes.
- All parameters of the QoS classes are independent of link length and the number of hops in the network.
- The number of QoS classes is finite.

In other words, we require that the QoS classes can be ordered and that they are independent of path length. For example such QoS classes can contain parameters such as bandwidth, jitter and maximum packet loss, but all the possible QoS classes must be comparable. Also note that in general there is no total order for a combination of such parameters and that the QoS parameter for maximum delay is not supported by this proposal.

To provide QoS guarantees such as bandwidth or jitter in a multicast tree the structure of the tree has to be as follows:

- The root of the multicast tree must be the node with the highest QoS requirement. In the implementation of Pastry we used for the evaluation (see Section V) we can explicitly assign an ID to a new Pastry peer.
- Each child node can only have a smaller or equal QoS requirement than its parent node.

In other words, each end-to-end path from the root to a leaf node in the multicast tree has to have a monotonically decreasing QoS requirement. Fig. 3 shows an example of such a multicast tree holding the second property described above. The path indicated as well as all other end-to-end paths of this multicast tree hold this property: the QoS requirements (denoted by the thickness of the lines) are the same or decreasing when following the intermediate hops from the root node to a leaf node.

If we analyze Scribe's multicast tree construction, we can see that the constructed multicast tree does not necessarily hold this property. The reason for this is that the end-to-end path from a leaf to the root is more or less randomly chosen, due to random positioning of Pastry peers. Our evaluation in Section V shows that with Pastry's default peer ID assignment, less than 40% of all end-to-end paths hold the above described property for randomly assigned QoS requirements. Because Pastry's default ID assignment does not take QoS requirements of peers into account, the multicast trees constructed by Scribe are only by chance holding the described property. It is sufficient that only one link in an end-to-end path does not hold the described property to disable the holding of QoS guarantees for all nodes in the multicast tree below this link. To enforce the construction of a QoS aware multicast tree using Scribe we propose the following:

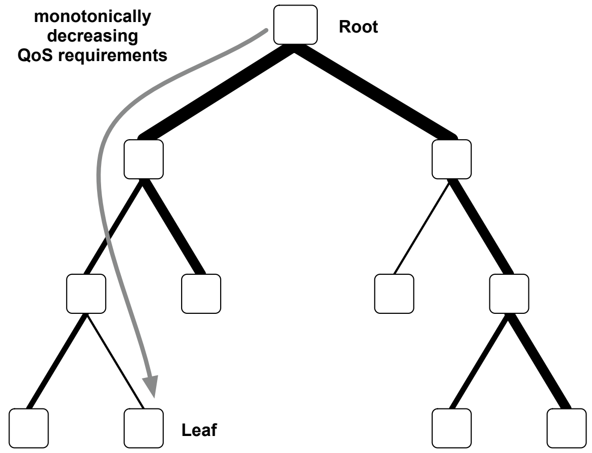

Fig. 3. Example of a multicast tree with monotonically decreasing QoS requirements from root to leaf nodes. Thickness of the lines represents the degree of the QoS requirement in terms of required bandwidth (thicker line = higher bandwidth requirement).

- For each multicast group exists a dedicated Pastry P2P network. The reason for this is to have only peers interested in receiving the multicast data as potential forwarders.
- In this Pastry network exists only one topic. This topic's ID is the highest possible topic ID.
- Since the the QoS requirements of a peer can be higher than its QoS capabilities, we choose the QoS class, which corresponds to the minimum of both.
- The ID space is partitioned into segments (see Fig. 4): one segment for each QoS class (we consider best-effort also to be a QoS class). The order of segments depends on the order of the QoS classes. The best-effort QoS class is located in the lowest segment and the highest QoS class is located in the highest segment. The assignment of IDs to joining peers depends on their QoS requirement / capabilities. The peer ID is randomly chosen within the corresponding segment of the ID space for the peer's QoS requirements / capabilities.

There are different possibilities on how large the segments should be, they do not necessarily have to be all of the same size and can for example decrease in size towards the root's ID. The partitioning strategy has an impact on the construction of the multicast trees and therefore on how well and evenly the overall traffic load will be distributed among the participating peers.

As shown in Section III the routing path from a peer with a lower ID to a peer with a higher ID always contains peers with increasing IDs. Because the root node of the multicast tree has the highest possible Pastry ID, we enforce that the routing always uses peers with increasing Pastry IDs for the hops on its path from leaf nodes towards the root node.

By assigning peer IDs proportional to the peer's QoS requirement, we enforce a construction of Scribe multicast trees, which hold the decreasing QoS requirement property for each end-to-end path from the root to the leaves. For each node that is passed through on the path from the root node to a leave node, the QoS requirement of the intermediate node is the same or lower than of its parent node.

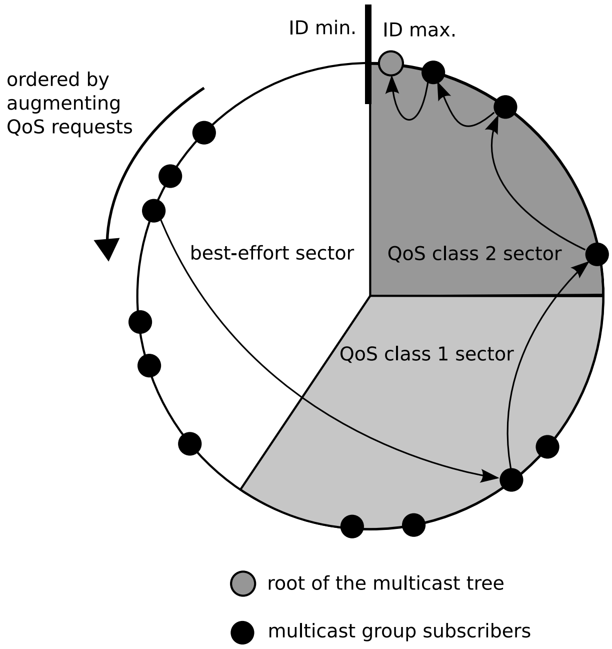
ID min.
ID max.
ordered by augmenting QoS requests
best-effort sector
QoS class 2 sector
QoS class 1 sector
root of the multicast tree
multicast group subscribers

Fig. 4. QoS aware distribution of peer IDs for Pastry

V. Evaluation

We evaluate the properties of the multicast trees created by Scribe / Pastry using our modification ID assignment by comparing it to the default (random) ID assignment of Pastry. For our evaluation we generated network topologies using the BRITE [24] network topology generator. We varied the number of hosts between 100 and 800 in steps of 100. For each number of hosts we generated 40 topologies. Each topology was generated using Waxman's model [25] with the following parameters: the surface of the simulated area is 5000×5000 meters, host placement is random, growth type is incremental, $\alpha = 0.15$, $\beta = 0.2$ and the number of neighboring hosts (m) is 2. Each such network topology is converted to a distance matrix by finding the optimal route (optimizing the number of hops) for each host pair and calculating the round trip time for that route. Additionally we assigned to each host a random QoS requirement (a value between 0 and 255). The generated distance matrix and the QoS requirement are then used to construct a multicast tree using Scribe / Pastry with our proposed ID assignment method and with the default (random) ID assignment method. The construction of the multicast tree is performed using the "BasicNetworkSimulator" provided by Freepastry [7].

For each generated multicast tree we evaluated whether the QoS requirement property holds for all end-to-end paths from the root to each node. If at least one hop on the path does not hold this property, we would not be able to guarantee the QoS requirement for that peer and all other nodes below this one in the multicast tree.

To verify our proposal, we compare for how many percent of the end-to-end paths the QoS property holds for the random and our ID assignment method. To see if our proposal produces longer end-to-end paths, we also compare the average and maximal path length for both ID assignment methods.

As we can see in Fig. 5 our modification of the ID assignment always performs better in respect of building QoS aware multicast trees than the random ID assignment of Pastry. As expected, the property described above holds for every end-to-end path in the multicast tree created by Scribe when using our method for assigning IDs.

Fig. 6 shows that the maximum and average path lengths (in terms of hop counts in the Pastry P2P network) for the two methods do not significantly differ, meaning that our modification of the ID assignment method is not worse than the random ID assignment of Pastry regarding average end-to-end path lengths in multicast trees.

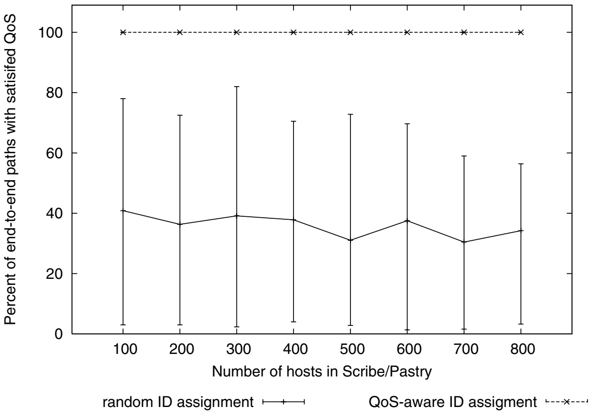

Fig. 5. End-to-end paths comparison regarding QoS satisfaction

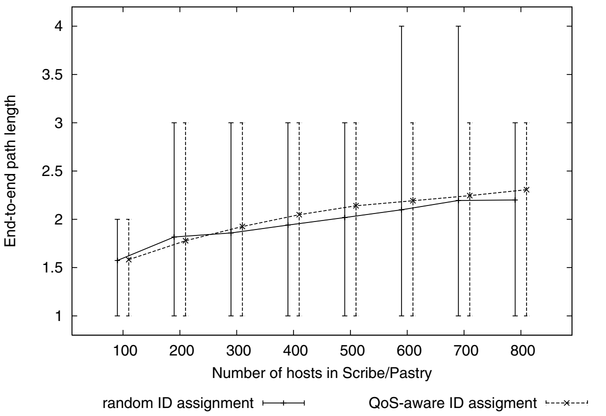

Fig. 6. Average and maximum path lengths (in hops) for all end-to-end paths

VI. Outlook

The ID-assignment presented in this paper is a part of the OM-QoS concept. OM-QoS aims to be a general method to enable QoS for multicast in structured P2P networks like CAN [9], Chord [10], Pastry, and similar P2P architectures. However OM-QoS is not necessarily applicable to any kind of structured P2P networks. Therefore we have to define appropriate requirements for P2P architectures.

Predictions of the peer life time and of the failure probability can be taken into account to build QoS aware multicast trees. Reputation concepts like Eigentrust [26], PACE [27] and others like [28], [29] have to be considered as an additional service for OM-QoS. Mechanisms for rewarding forwarding peers and charging issues have to be examined as well.

VII. Conclusion

In this paper we presented a concept to enable QoS multicast for structured P2P networks and applied it to Scribe / Pastry. Our evaluation showed that by using a QoS requirements aware ID assignment method, we can ensure that multicast trees built by Scribe hold the QoS requirements (for example bandwidth, jitter or maximum loss) on all end-to-end paths. The simulations showed that the default ID assignment method of Pastry resulted in an average of 30-40% of the end-to-end paths holding bandwidth QoS requirements. With our modification of the ID assignment we were able to construct multicast trees where every end-to-end path holds the bandwidth QoS requirement.

Acknowledgment

This work has been partly funded by the European project EuQoS (project number 004503).

References

[1] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, nov 2001, pp. 329–350.

[2] 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 Communication (JSAC)*, vol. 20, no. 8, oct 2002.

[3] [Online]. Available: http://www.euqos.org

[4] E. Angori and G. Martufi, "D1.1.2: System design: Functions, interfaces specification," Tech. Rep., May 2005.

[5] D. Milic, M. Brogle, and T. Braun, "Video broadcasting using overlay multicast," in *ISM '05: Proceedings of the Seventh IEEE International Symposium on Multimedia*. Irvine, CA, USA: IEEE Computer Society, 2005, pp. 515–522.

[6] [Online]. Available: http://vtun.sourceforge.net/tun/

[7] [Online]. Available: http://freepastry.org/

[8] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Berkeley, CA, USA, Tech. Rep., 2001.

[9] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2001, pp. 161–172.

[10] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2001, pp. 149–160.

[11] X. Y. Zhang, Q. Zhang, Z. Zhang, G. Song, and W. Zhu, "A construction of locality-aware overlay network: moverlay and its performance," *IEEE journal on selected areas in communications*, vol. 22, no. 1, pp. 18 – 28, January 2004.

[12] J. Zhang, L. Liu, C. Pu, and M. Ammar, "Reliable peer-to-peer end system multicasting through replication," in *P2P '04: Proceedings of the Fourth International Conference on Peer-to-Peer Computing (P2P'04)*. Washington, DC, USA: IEEE ComSoc, August 2004, pp. 235–242.

[13] R. Zhang and Y. C. Hu, "Borg: A hybrid protocol for scalable application-level multicast in peer-to-peer networks," in *NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, ACM. New York, NY, USA: ACM Press, June 2003, pp. 172–179.

[14] A. Sobeih, W. Yurcik, and J. C. Hou, "Vring: A case for building application-layer multicast rings (rather than trees)," in *MASCOTS '04: Proceedings of the The IEEE Computer Society?s 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'04)*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 437–446.

[15] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz, "Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination," in *NOSSDAV '01: Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*. NY, USA: ACM Press, 2001, pp. 11–20.

[16] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream: High-bandwidth multicast in a cooperative environment," in *19th ACM Symposium on Operating Systems Principles (SOSP'03)*, oct 2003.

[17] ——, "Splitstream: High-bandwidth content distribution in a cooperative environment," in *IPTPS'03*, feb 2003.

[18] Z. Li and P. Mohapatra, "Qron: Qos-aware routing in overlay networks," *Selected Areas in Communications, IEEE Journal on*, vol. 22, no. 1, pp. 29–40, 2004.

[19] Z. Li, "Resiliency and quality-of-service (qos) support in multicasting and overlay networks," Ph.D. dissertation, Davis, CA, USA, 2005, adviser-Prasant Mohaptra.

[20] J. Jannotti, D. K. Gifford, K. L. Johnson, and F. M. a. Kaashoek, "Overcast: Reliable multicasting with an overlay network," pp. 197–212. [Online]. Available: http://citeseer.ist.psu.edu/jannotti00overcast.html

[21] B. G. Rocha, V. Almeida, and D. Guedes, "Increasing qos in selfish overlay networks," *IEEE IC*, vol. 10, no. 3, pp. 24–31, 2006.

[22] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "One ring to rule them all: Service discover and binding in structured peer-to-peer overlay networks," in *SIGOPS European Workshop*, Sep 2002.

[23] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel, "Scribe: The design of a large-scale event notification infrastructure," in *Networked Group Communication, Third International COST264 Workshop (NGC'2001)*, ser. Lecture Notes in Computer Science, J. Crowcroft and M. Hofmann, Eds., vol. 2233, nov 2001, pp. 30–43.

[24] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite: An approach to universal topology generation," in *MASCOTS '01: Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01)*. Washington, DC, USA: IEEE Computer Society, 2001, p. 346.

[25] B. Waxman, "Routing of multipoint connections," pp. 347–352, 1991.

[26] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *Proceedings of the twelfth international conference on World Wide Web*, ACM. ACM Press, 2003, pp. 640 – 651.

[27] G. Suryanarayana, M. H. Diallo, J. R. Erenkrantz, and R. N. Taylor, "Architectural support for trust models in decentralized applications," in *ICSE '06: Proceeding of the 28th international conference on Software engineering*. New York, NY, USA: ACM Press, 2006, pp. 52–61.

[28] Z. Abrams, R. McGrew, and S. Plotkin, "A non-manipulable trust system based on eigentrust," *SIGecom Exch.*, vol. 5, no. 4, pp. 21–30, 2005.

[29] M. Khambatti, P. Dasgupta, and K. D. Ryu, "A role-based trust model for peer-to-peer communities and dynamic coalitions," in *IWIA '04: Proceedings of the Second IEEE International Information Assurance Workshop (IWIA'04)*. Washington, DC, USA: IEEE Computer Society, 2004, p. 141.