Paper Analysis

A comparative study of Application Layer Multicast Protocols

Multicast addressing is used for the delivery of data to a group of hosts simultaneously, using the most efficient strategy to send messages such that as less number of copies of the data as possible are sent over the same link or node. The first implementation of the addressing scheme was done using routers, that is in the IP or network layer. However, this IP Multicast scheme suffers from the drawback of poor scalability as each router will need to store state information of every multicast group and thus is not very workable. Therefore, IP multicast has not been deployed widely in the Internet. Several application layer multicast schemes have been proposed and in this paper, the authors have described and then compared three classes of such application layer schemes using some examples.

The organization of the paper is pretty straightforward and logical. First, the authors introduce us to the problems that IP multicast scheme has:

1. Routers need to store a per multicast group state. This causes severe scalability issues.
2. The multicast group addresses are not easily aggregatable which increases the complexity needed to incorporate at the routers.
3. Lack of understanding of reliability and congestion control has also prevented the large scale deployment of the IP multicast.

Thus, the authors provide us with the motivation to understand the alternatives, that is the application layer mutlicast protocols, more deeply. Application layer multicast protocols take out the implementation of the protocols from the routers and put them on the end hosts. The multicast data packets are now replicated at the hosts rather than at the routers as in the case of IP multicast. Basically, the nodes form a overlay network. Understandably, this leads to some performance implications which need to be analyzed. In their analysis, the authors used some metrics: Quality of data path and Control Overheads. The quality of the data path impacts the available bandwidth and the control overheads impact the transmission time of the data. Quality of data is further decided by two factors:
Stress (at a node): it refers to the number of identical packets sent over that node.
Stretch (at a member): it refers to the ratio of source-member path length over the overlay network to the direct unicast path between them.

Even though the other metrics are mentioned in the introduction (degree distribution of members of the network, bandwidth requirements), they are hardly talked about throughout the paper. No evaluation or survey has been reported in the paper. I think including more metrics in the survey would make the work more exhaustive and useful.

Next, for the purpose of the survey, the authors classified the current (as in year 2002) Application layer multicast schemes in to three classes and selected a few applications 'representative' to those classes. However, the authors do not explain anywhere as to why these applications were selected. Later in the paper, they also mention that Narada and Yoid, two of the chosen example applications, were the first of their kinds. It generally is the case that the first applications are not the best performing applications. Thus, I am not very sure if the results published at the conclusion actually represent the classes.

An application layer multicast scheme members are organized into two topologies: Data topology (tree topology) and Control topology (mesh topology). The three classes specified by the authors are based upon the order of constructing these topologies and are:

1. Mesh first approach
2. Tree first approach
3. Implicit approach

Mesh first approach refers to the application scheme where the members organize themselves into an overlay mesh topologies and then discover unique data paths between them. The tree first approach is the opposite where a shared tree is first created and then links are added between the non-neighbors in the tree to create a mesh. The implicit approach uses a packet forwarding rule which implicitly defines the control and data topology for the network.

To 'represent' the mesh first approach, the authors took the example of Narada where each member of the network keeps the state information about all other members. Thus there is a high control overhead (due to a high number of refresh messages between the members) of $O(N^2)$, but also high robustness which helps to repair faults in the network. I particularly liked the idea of 'Mesh Refinement' where the links in the mesh keep rearranging depending upon the quality of the link. It would be interesting to if it would increase the control overhead further. However, the authors haven't mentioned about it anywhere. Secondly, the authors should have mentioned that they were going to use Fig.2 to explain the working of Narada(page 3). After reading through a whole paragraph does the reader realize that he should refer to the figure.

Similary, the authors have used two examples to explain the Tree First approach: Yoid and Host Multicast Tree Protocol (HMTP). Both the protocols require a tree to be constructed first and a new node to find a parent for itself in the tree to join it. The Rendezvous point (RP) helps the incoming node to find its most appropriate parent in the tree. Next, the members find a few other non-neighbors and establish links with them to make the control topology. A very good point that the authors raised what that the control information incorporates loop avoidance and loop detection in the trees. It also helps in re-arranging the tree as members keep finding 'better' parents.

The authors then describe the Implicit approach by taking three examples: NICE, CAN multicast and Scribe. In NICE, the main operation or objective is to maintain a certain hierarchy which implicitly defines the data paths in the overlay. The whole network is divided into hierarchical layers and each member is assigned to a particular layer. Further, members of each layer are divided into clusters which have a leader each. This leader is also a member of the one level above layer in the hierarchy and connects them to the members of that layer. Thus, the whole structure is hierarchically laid out and all communication is done through it.

CAN multicast scheme is based on the CAN addressing scheme where a set of end hosts implement a distributed hash table on the internet. Here, the control topology is constructed by the adjoining members peering with each other and the data topology is formed by directed flooding on the control topology. This directed flooding also takes care of loop avoidance in the network. Scribe follows a similar structure and is built on top of Pastry, a peer to peer protocol.

After the description of all the three classes of the protocols, the authors give a description of the comparative study of them. I feel highly unsatisfied with the way results were presented in this section. The goal of the paper was to analyze these protocols and compare them but most of the paper content was about the presently available protocols and too little on the comparative study. Therefore I think the authors did not do justice to that goal. The authors mention that “Simulations have shown reasonable to good *stretch* performance of all the other different protocols”, however they did not specify the criteria for deciding the 'goodness'. Then in the same section they also mention that “the metrics are indirectly related to the stress and stretch metrics”. It would have been much better had they written a line explaining the rationale behind it. Another point that they mention is that the average overhead for members is constant for NICE and CAN, but again they do not provide any reason for the same. The conclusion of the section was that Mesh-first protocols are good for small multicast groups whereas the implicit protocols are good for larger ones as they scale well.

Overall, I believe that the foundation of the paper was a bit weak. There was a goal of 'comparing the three classes' of application layer multicast protocols, but the author ended up simply explaining the protocols (which could have been understood from their own published papers) rather than focusing on the comparison. Furthermore, the paper also lacked a reading flow and was not terribly easy to read. There were some organizational glitches too. For example, they started talking about Figure 2 without mentioning that they were going to talk about it. However, for the kind of comparison the authors intended (not very detailed), perhaps dividing the protocols into classes was a good idea.