Implementation of Narada protocol for End System Multicast

Ajit Kulkarni
Masters Student
Department of Computer Science
Virginia Tech
ajitkulk@cs.vt.edu

Logambigai Venkatachalam
Masters Student
Department of Computer Science
Virginia Tech
lokeya@cs.vt.edu

Tilottama Gaat
Masters Student
Department of Computer Science
Virginia Tech
tgaat@cs.vt.edu

ABSTRACT
The IP layer has been the preferred protocol layer for implementing Multicast related functionality. End System Multicast. was introduced as an alternate way which pushed the functionality of Multicast to the higher level layers (application layer). The Narada protocol is one of those used for implementing End System Multicast and we have implemented that in our project. We have implemented most of the features of the protocol on the Java Network Simulator (JNS1.7) which was considered as our network topology simulator. End system multicast has an advantage over IP Multicast that it resolves many of the latter's problems but it has reduced performance. We evaluate the performance of our implementation of the Narada protocol by conducting some experiments with the simulator by measuring network parameters like latency and bandwidth. Our results are consistent with that of the original authors and hence we conclude that End system Multicast is a viable alternative to IP multicast due to its advantages inspite of the performance penalty associated with it.

Categories and Subject Descriptors
D.3.3 [**Operating Systems**]:Communications Management

General Terms
Algorithms, Management, Measurement, Performance, Design ,Reliability, Experimentation, Languages, Verification.

Keywords
Multicast ,End System Multicast, Graph, Network, Random numbers, Routers, Links, Bandwidth, Latency, Minimum Cost Spanning Tree, Java network simulator, Unicast ,datagram, IP multicast, Narada, performance, DVMRP

1. INTRODUCTION
Network Architectures:

Network architectures mainly have been divided into two areas end systems (hosts) and network (routers and switches).The network layer of the internet is implemented using the IP protocol. The IP protocol provides the basic functionality of unicast datagram message transfer while all the other functionality is pushed to the end systems. A new functionality of multicast was added to IP and the resulting feature was called IP multicast. However IP Multicast was plagued by many problems. Following are the advantages and disadvantages of IP multicast.

Advantages of IP Multicast:

- No duplicate packets send across any physical link
- Efficient bandwidth usage

Disadvantages of IP Multicast:

The first problem is that it requires every router to maintain the group state. This violates the initially envisioned "stateless" principle and it is also introduces a lot of complexity and has scalability constraints.

The second problem is that IP multicast tries to conform to the traditional separation of network and transport layers. This worked well in the unicast context but other features like reliability, congestion control, flow control and security are difficult to implement.

The third and final problem was that it requires changes at the infrastructure level and is hence is not easy to deploy.

An alternative to this is End System Multicast (1)(ESM) in which all the functionality of Multicast is pushed to the end systems and the application level considering only unicast IP. According to end to end arguments where a functionality should be pushed up to the higher layers is possible unless implementing the functionality at the lower layers gives performance benefits that outweigh the cost of adding the additional complexity at the lower layer. ESM can implement many complex features of multicast functionality. It basically constructs an overlay structure among all hosts in the network and then sends messages to the other end hosts in the overlay structure. This approach is stateless. Implementing all other features of multicast is easier at application level rather than at network level. Since the underlying network layer still implements naïve

IP unicast this is easier to deploy. Following are the advantages and disadvantages of ESM multicast:

Advantages of End System Multicast:

All packets are sent as unicast packets so deployment is accelerated.

No more routers need to maintain the per group state information. And the end systems take up this responsibility. Since these end systems are part of very few groups it becomes easy to scale the systems

Supporting higher layer features such as error, flow, and congestion control can be significantly simplified by leveraging well understood unicast solutions for these problems, and by exploiting application specific intelligence

Disadvantages of End System Multicast:

The overlay structure is built on existing physical links so you may have multiple overlays over a single physical link hence there will be redundant traffic across the links.

Any message traverses across other end systems before reaching its destination hence the latency will also increase.

Performance is reduced with this model.

2. Multicast Systems:

2.1 Naïve Unicast:

When Naïve unicast is used to multicast the source node send distinct messages to every other member in the group. This leads to significant redundancy near the source node and many duplicate messages over the other links in the network.

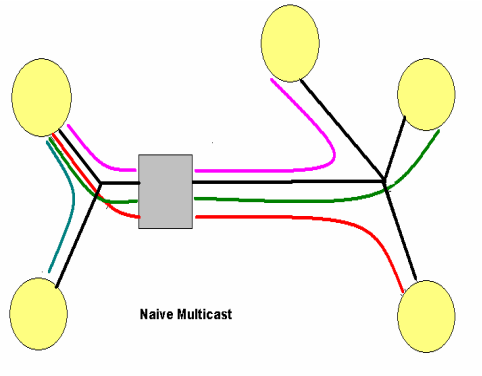

Fig1. Naïve Unicast

In the above diagram different colors signify the different unicast paths from the source to the destination nodes.

2.2 IP Multicast

It is generally implemented using DVMRP (Distance Vector Multicast Routing Protocol). In this data is delivered from source to recipients using an IP multicast tree from source to each recipient with the help of the intermediate routers. There is only one copy of a packet over a physical link and the delay is same for all recipients, each recipient would feel that the data is unicast to it.

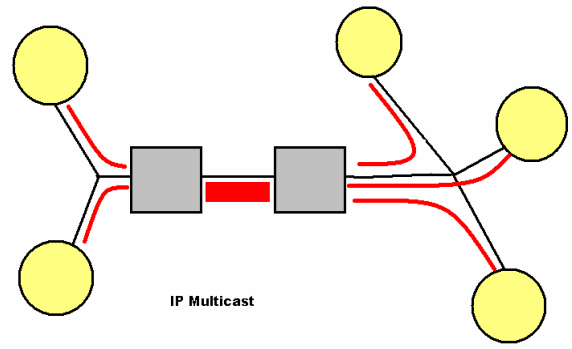

Fig2. IP Multicast

In the above diagram we can see that through every physical link there is only one copy of data which is transmitted. But the amount of data at the source node is more.

2.3 End System Multicast

It is implemented without changing the network layer. An intelligent multicast tree is created among hosts only and then messages are sent over that tree. This reduces the number of redundant copies as compared to naïve unicast but has more delay as compared to IP multicast. However in this implementation no change is required to the routers and all the functionality is implemented at end systems.

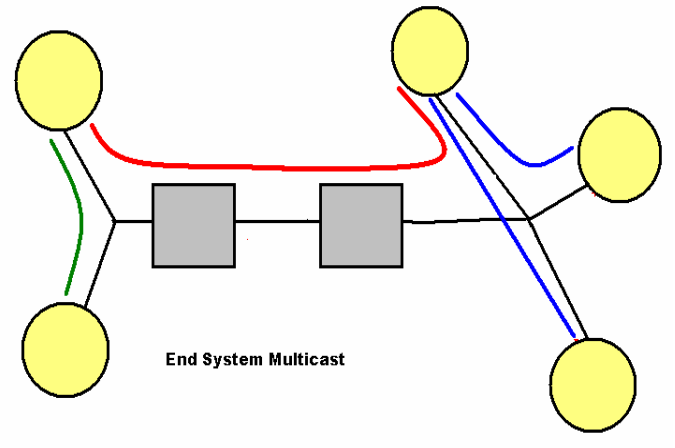

Fig3. End System Multicast

In the above diagram we can see the intelligent overlay tree being constructed at the end systems.

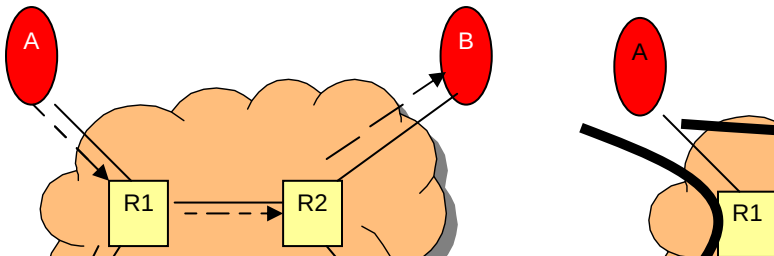

Fig4. Comparison of IP and End System Multicast

3 Narada Protocol Features:

Narada is the protocol to implement End System Multicasting. It has many features like:

Self organizing

The construction of the end system overlay in fully distributed fashion and is adaptive to dynamic changes in group membership

Overlay efficiency

The tree constructed is efficient both from application and network perspective and the number of redundant packets transmission is kept minimal. However the definition of efficiency differs for every application.

Self Improving

The end systems gather network information in a scalable fashion. So the overlay structure improves as more information becomes available.

Adaptive to network dynamics

The overlay created adapts to long term variations in internet path characteristics and it is resilient to the inaccuracies in the measurement of these quantities.

4 Narada Protocol Design

4.1 Tree and Mesh Creation

Narada creates a mesh, a highly connected graph between all the nodes in the group. It then creates a minimum cost spanning tree among all the hosts using the mesh. A mesh based approach is used for multi source applications .Also a single shared tree is susceptible to a central point of failure. They are not optimized for a single source. It is important to

create a good mesh for creating good trees. A good mesh has the following properties. Firstly quality (hence weight) of a path between any two members is comparable to the unicast path between the two members. Secondly each member is connected to a limited number of neighbors in the mesh. Narada runs a variant of standard distance vector routing algorithms and it creates reverse shortest path spanning trees for each source.

4.2 Group Management

Member Join

The joining member randomly selects a few group members from the list available to it and sends them messages requesting to be added as a neighbor. It repeats the process until it gets a response from some member, when it has successfully joined the group. Having joined, the member then starts exchanging refresh messages with its neighbors.

Member Leave and failure

When a member leaves a group, it notifies its neighbors, and this information is propagated to the rest of the group members along the mesh. We also need to consider the difficult case of abrupt failure. In such a case, failure should be detected locally and propagated to the rest of the group. In this paper, we assume a fail-stop failure model, which means that once a member dies, it remains in that state, and

the fact that the member is dead is detectable by other members.

Repairing Mesh partitions

Member failure can cause the mesh to be partitioned. In such a case, members must first detect the existence of a partition, and then repair it by adding at least one overlay link to reconnect the mesh. Members

on each side of the partition stop receiving sequence number updates from members on the other side . This condition is detected by a timeout of duration Tm.

4.3 Mesh Performance

The constructed mesh can be quite sub-optimal.

Data Delivery

On the top of the mesh Narada runs the distance vector protocol. Each member maintains a routing cost to the destination and also the path that leads to that node. A member M that receives a packet from source S through a neighbor N forwards the packet only if N is the next hop on the shortest path from M to S.Further, M forwards the packet to all its neighbors who use M as the next hop to reach S.

Application Specific Customizations

A key feature of End System Multicast is that it enables application customizable solutions.

5 Our Implementation of Narada

5.1 Tools Used

JNS 1.7 – Java Network Simulator

We used the Java implementation of the ns_2 network simulator originally from Berkeley. Using JNS we can simulate a network. We create entities like nodes, links, interfaces, routers etc and then send messages among them. The messages are in the form of standard packet entities which have similar contents as a standard packet. JNS, like every modern piece of software, is not one huge piece of code but partitioned into subsystems.

The packages in JNS are divided by functionality:

jns.element - contains the static elements of the network, nodes, links, etc.

jns.agent - contains the interfaces for agents.

jns.trace - contains everything that is necessary to snoop at what is going on in the network. A class is provided to turn events in the network into output suitable for use with *Javis* or *NAM*.

jns.command : Contains the Command abstract class which you will almost certainly extend if you write a new protocol. It can be used to schedule calls to your functions with the simulator.

jns.util - utilities, tools, etc. Contains an implementation of a Queue data structure (not a networking queue!), a priority queue, the IPAddr class, the Preferences class, etc

All the network entities are created in the first phase. Then it creates routes among the different nodes and routers depending on the delays of links given to them. Finally the simulator is run which schedules all events and messages to be sent at specified times during its execution. It creates a trace file as output which is used by the Javis animator program and shows the data routing which happens.

Spanning Tree Creation Algorithm

Kruskal's algorithm

This is an algorithm in graph theory that finds a minimum spanning tree for a connected weighted graph. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. If the graph is not connected, then it finds a *minimum spanning forest* (a minimum spanning tree for each connected component). Kruskal's algorithm is an example of a greedy algorithm.

function Kruskal(G)

for each vertex v in G do

Define an elementary cluster C(v) ← {v}.

Initialize a priority queue Q to contain all edges in G, using the weights as keys.

Define a tree T ← Ø //T will ultimately contain the edges of the MST

while T has fewer than n-1 edges **do**

(u,v) ← Q.removeMin()

Let C(v) be the cluster containing v, and let C(u) be the cluster containing u.

if C(v) ≠ C(u) **then**

Add edge (v,u) to T.

Merge C(v) and C(u) into one cluster, that is, union C(v) and u).

return tree T

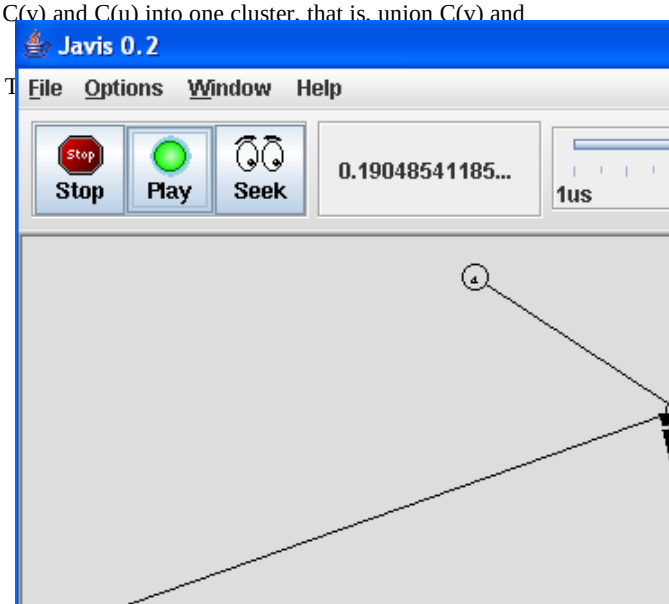

Fig5 Javis Network Animator

Network Animator : Javis 2.0

Javis is a software which acts as a packet flow and network animator. It takes as input simulation results from various network simulators which are generally trace files. It can show network topology, node locations, links between nodes,show the flow of messages sent across links and also topology evolution over time. It presents a simulation based on the trace files. It is a Java applet program.

5.2 Narada Features Implemented

Mesh Creation

We use the network entities given by JNS to create a network. We create entities like nodes, links, routers etc. We used a random number generator to assign weights to the links. The nodes have names 1,2 …etc. the number of edges in the network for a number of nodes is also generated by random numbers. We try to have a highly connected graph. All those nodes which are not connected have a weight of a constant high valued number representing an unreachable link.

Group Creation

A group member is defined as a separate entity in a class. In Narada every member of the group contains a list of all members in the group to which it is connected. So a GroupMember object has a Node object and an array of nodes and costs to reach them in it. If a member is not connected to a node it has the constant value representing an unreachable node in it. A group is defined as a list of GroupMember objects.

Member Join

When a new node wants to join a group it brings along with it some information about its distance to any existing group member with it. The group join algorithm works as follows. In the first step the list of the joining node is updated. All those elements to which it's not connected are added with unreachable weight to its list. Then it is added to the lists of all existing group members with corresponding weights. Finally it is added to the list of members of a group. When data routing has to be done a new spanning tree will be created with this node.

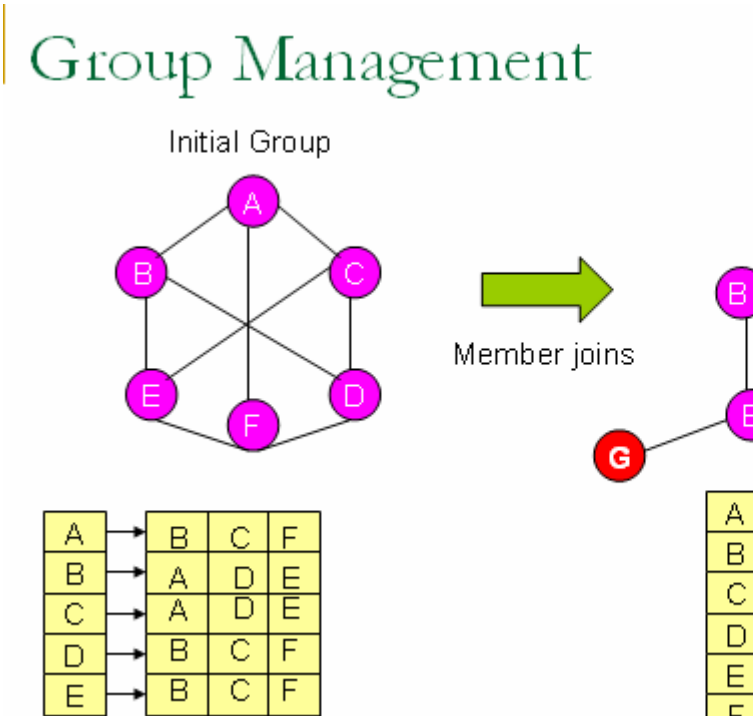

Fig6 Member joining a group

Member Leave –Graceful

When a member leaves the group gracefully it informs other group members that it is leaving. Accordingly when he leaves his list is deleted and his record is deleted from the its of all other existing group members. When data routing has to be done a new spanning tree will be created without this node.

Member Leave –Abrupt

When a member leaves a group abruptly or fails then the algorithm for abrupt leave has to be used. In our implementation in such cases all group members update their lists and give the unreachable weight to this node.But the record for that node isn't immediately deleted from the list. This is because they are not sure if he has temporarily disconnected or has permanently left the group. After a fixed interval of time the record is deleted from their lists.

Member Failure

The member failure is handled in the same manner as abrupt group leave.

Tree Creation

The entire structure of network consisting of all nodes and weighted edges is given to the spanning tree algorithm. We then use the Kruskal's algorithm to find the minimum cost spanning tree among these nodes. We also calculate the start and end times for each message of the spanning tree and also the hop number in the tree. If A->B has delay 0.04 and B->C has delay 0.05 and both these edges are the in spanning tree then the start an dend times for both of them are (0.01 ,0.05) and (0.05,0.10) .This information is then given to the simulator.

Data Delivery

The use enters a source and we consider the last node as the destination. We then extract a path from the spanning tree from the source to the destination. We then give the edges in the path to the simulator which sends the messages along those paths at the specified start times. A trace file is generated which can be given to Javis to show the animation.

5.3 Narada Features Not Implemented

There were some features of the protocol which we could not implement. As we were working with the simulator we were dealing with several limitations. The simulator runs for a specified time and we have to schedule events at specific instances within that time frame. Also there is a limit to the number of messages that can be sent in the simulator.

Refresh Messages

In the Narada protocol group members randomly exchange their knowledge of the group by refresh messages. This is how members come to know of new members joining the group or other members leaving the group. In our linear simulator we cannot schedule random refresh messages among group members. The limit of the simulator also creates a problem.

Repairing Mesh Partitions

In the Narada protocol we can detect and repair mesh partitions by adding links. A partition is detected when refresh messages are not received from certain nodes and this structure can break the network into two partitions. Since we didn't implement refresh messages we could not implement this. In the linear structure of the simulator it is not possible to randomly check the structure of the network for partitions.

Mesh Performance

To keep a optimal mesh and to maintain good performance the Narada protocol adds and removes links randomly when they satisfy some heuristic functions. These functions consider the state of the network at a particular time and then add or remove links if they can make the mesh more optimal. It is not possible for us to randomly probe the state of the network with the simulator. The heuristic functions consider many factors in their calculation and these factors are not available in the simulator which runs for a small time. Not many network metrics are available in the simulator. It is designed to implement new algorithms and test them and not to measure performance.

6. Evaluation

6.1 Experiment 1 – Latency vs. Group Size

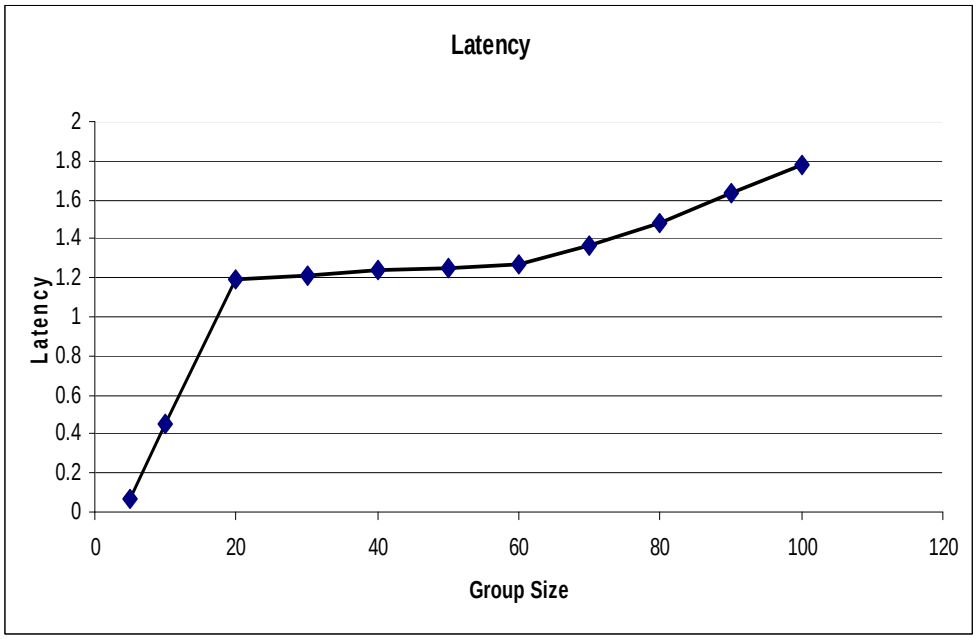

Fig 7: Latency v/s Group Size for seed value 8

In the following graph we used to grow the tree in a breadth wise manner and calculate latency. We started with a group size of 20 nodes and had grown the tree up to 120 nodes and have got the following graph.

We selected a seed for random number generation. Using the random numbers so generated, we created a tree structure of 5 nodes, with randomly assigned weights and edges.

For group size 10, we loaded the previous tree structure and appended nodes to it, in a breadth-wise fashion, to retain some consistency. In case of 20 nodes, we load the structure obtained for 10 nodes and so on. The tests were carried out till group size 100.

The graph below is for seed 8. At this particular seed, there is a large edge weight generated for the vertices 10 to 11, hence the jump in the graph from group size 10 to 20.

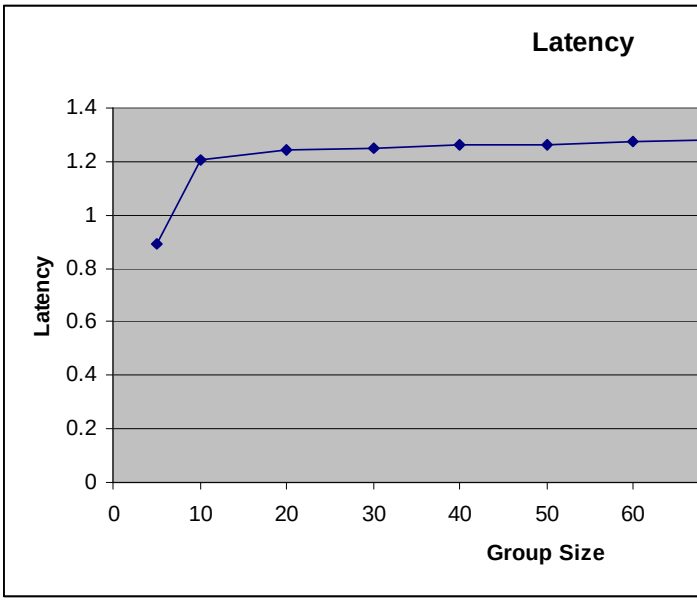

Fig 8: Latency v/s Group Size for various seeds

This is the graph that was obtained with various seeds. We repeated the runs for each group size with seeds 3,5,6,7,8,9. Thus for each group size we obtained 6 values for latency. The average for each was taken and the graph was plotted. These results in a smoother curve, than the one obtained for just 1 seed.

Results:

We carried out experimentation successfully for the parameter 'latency.' Due to the limitations imposed by the network simulator, we could not carry out benchmarking for parameters such as bandwidth and stress.

The X axis represents the group size and the Y axis represents the latency for the packet to reach destination from source. As the group size increases, the overlay becomes larger and hence the hop count also increases. As a consequence, the latency increases.

Applications

Our evaluation of End System Multicast targets a wide range of group communication applications such as audio and video conferencing, virtual classroom and multi-party network games. Such applications typically involve small (tens of members) and medium sized (hundreds of members) groups. While End System Multicast may be relevant even for applications which involve much larger group sizes such as broadcasting and content distribution - particularly in the context of proxy-based architectures

7. Related Work

Comparison of some well known ESM protocols

Protocols	Routing	Multicast Construction	Group Size
Narada	Tree	Mesh First	Small
NICE	Tree	Implicit	Large
Bullet	Mesh	Any	Large

Architecturally, proposals for overlay based multicast have primarily differed on whether they assume a peer-to-peer architecture, or a proxy (infrastructure) based architecture.

Yoid emphasis on peer to peer architectures less on performance.It Uses a shared tree among participating members.It is more susceptible to a central point of failure. Distributed heuristics for managing and optimizing a tree are more complicated as cycles must be avoided

Scattercast and Overcast emphasise on infrastructural support and proxy-based multicast. This uses a mesh but differences in protocol details

8. Conclusion

For small and medium sized multicast groups, it is feasible to use an end system overlay approach to efficiently support all multicast related functionality including membership management and packet replication. The potential benefits of transferring multicast functionality from end systems to routers significantly outweigh the performance penalty incurred due to end system multicast

One of the first self-organizing and self-improving protocols that constructs an overlay network on top of a dynamic, unpredictable and heterogeneous Internet environment without relying on a native multicast medium. Further, we believe that the techniques and insights developed in this paper are applicable to overlay networks in contexts other than multicast.

From our experiments , we observe that the performance penalty incurred by Narada protocol is negligible for small group sizes, in case of latency. But Narada, can address many unresolved issues in IP Multicast. Therefore, End System Multicast can prove to be a feasible alternative to IP Multicast.

Future Directions:

Keeping pieces of data at intermediate nodes near the receiver. Enabling fast access to this data from the receiver

Ensuring the co-operation of intermediate nodes

9.References

[1] Yang-Hua Chu, Sanjay Rao, and Hui Zhang. *A case for end system multicast*. In Proceedings of ACM Sigmetrics, Santa Clara, CA, 2000.

[2] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. ACM Transactions on Computer Systems, pages 277288, 1984.

[3] Java Network Simulator http://jns.sourceforge.net/

[4] http://homepages.cs.ncl.ac.uk/einar.vollset/home.formal/jns.html

[5] Javis – Java based n/w animator. ftp://cs.ucl.ac.uk/nets/src/jns/javis-0.2/.