Real-Time Public Group Collaboration using IP Multicast Label Filters

Jaipal Singh, Vidyasagar Potdar
Digital Ecosystems and Business Intelligence Institute
Curtin University of Technology
Perth, Australia
e-mail: {j.singh, vidyasagar.potdar}@cbs.curtin.edu.au

Prakash Veeraraghavan, Samar Singh
Dept. of Computer Science and Computer Engineering
La Trobe University
Melbourne, Australia
e-mail: {p.veera, s.singh}@latrobe.edu.au

***Abstract*—Internet based enterprise level collaboration tools enable organizations to make decisions faster and more accurately with less effort. However, these tools provide limited real-time group collaboration within and across organizations. Multicast protocols were developed to provide efficient group communication. This paper proposes a novel IP multicast network layer filter architecture that provides efficient and scalable real-time group collaboration between the required entities within an organization. This proposed network architecture uses a label filter mechanism to improve scalability and bandwidth for one-to-many and many-to-many real-time collaboration.**

Keywords-Group Colloboration; Multicast Routing; Network Filtering

I. INTRODUCTION

In recent years, businesses have started adopting web-based Enterprise level internet collaboration tools for providing better networked workplace collaboration and information sharing within an organization, across different branches as well as with partner organizations. These collaboration tools use a web-based interface to provide features like document/content management, file sharing, real-time data updates, text chat, application/document sharing and threaded discussions. While the information is provided in real-time, the recipient will access the information at different times. These asynchronous (not real-time) information sharing tools provide very limited real-time collaboration between the various users.

This paper described an innovative publish-subscribe network architecture that provides efficient and dynamic real-time group communication in an enterprise. The proposed architecture will reduce the end user computation overheads and optimize the use of network bandwidth for delivering data packets between all users collaborating through the organization's network infrastructure. This architecture will provide a novel filtering architecture for public and open group communication on a shared network path so that the users will have more flexibility when using the organization's network infrastructure for real-time collaboration. This paper is organized in 5 sections. Section 1 gives a brief introduction to the problem domain. Section 2 provides an overview of existing network layer routing protocols. Section 3 describes the proposed network layer group collaboration architecture. Section 4 provides simulation analysis of the proposed architecture. Section 5 concludes this paper.

II. OVERVIEW OF NETWORK LEVEL GROUP COMMUNICATION ARCHITECTURES

The regular internet communication (unicast) protocol is a point-to-point protocol for communication between one sender node and one receiver node. Multicast routing protocols were introduced as a more efficient method for group communication over an IP network. The multicast protocol creates a logical network tree represented by a multicast group address that anonymously connects all nodes that want to receive the same communication. The sender will address packets to the multicast address and every node that subscribes to this group (address) will receive the communication. Multicast is more efficient that unicast since only a single instance of the data packet will be transmitted on every link in the multicast tree [1, 2].

There are two categories of multicast network protocols, source-based trees (SBT) and shared trees (ShT). A SBT protocol is used for a multicast group that only has one sender and many receivers (one-to-many communication). This protocol is ideal for one way communication such as distributing software updates or content to all receivers in the network. If other nodes on the SBT want to send messages, N SBT multicast groups have to be formed where N is the number of senders. The IPv4 multicast specification allows only 2^{28} multicast addresses. This limited address space presents a huge scalability issue and limits the number of multicast groups that can be created [3]. To solve this issue, ShT protocols were introduced. ShT overcomes the scalability problem of SBT since all nodes on the ShT can transmit and receive packets using the same tree (many-to-many communication). The ShT protocol can be used for providing interactive collaboration such as interactive video conferencing among many users for a particular subject. However, ShT protocols might incur more delay than SBT since all traffic has to go through a central core router [4, 5].

In terms of applications, a multicast group is ideal for distributing subject-based communication through the publish-subscribe model. All publish-subscribe systems are used for providing more efficient group level communication on a network. They are categorized as subject-based or content-based, where the subscribers will receive information in a timely manner based on a particular subject or specific content respectively [6].

978-0-7695-3767-2/09 $25.00 © 2009 IEEE
DOI 10.1109/NBiS.2009.28

This paper will use a ShT multicast protocol as the network routing protocol since it is built to support one-to-one, one-to-many, and many-to-many communication [7]. While ShT allows for multiple senders in one group, new multicast trees must still be built for different groups. The ShT multicast protocol still has scalability issues due to limited IP address allocations. The ShT protocol is not suited for content-based delivery since the receivers join the multicast group based on the subject.

Although multicast was introduced in the 1990s, it was not widely adopted due to the limitation of network bandwidth, no suitable pricing model, limited address range, scalability and limited router support. With the recent increase in network bandwidth (both wired and wireless), device capacity and computation power, the always connected nature of networked devices, and improved multicast support on the network infrastructure has increased the demand for multicast technologies with regards to video and audio based transmissions in a group environment such as video conferencing and Internet Television (IPTV) [8].

Previous researchers have proposed using filters on a multicast tree to improve scalability, privacy and implementing content-based distribution, but these methods still have drawbacks like processing delays and high network computation costs [9]. The need to improve scalability due to limited availability of group addresses and reduction in router entries will further improve the wide-spread adoption of multicast technologies for group communication on the internet.

A. Multicast filters

Filtering multicast communication by subject or content is a method to reduce the number of receivers to only relevant members in the organization's network, thus improving network bandwidth usage.

The simplest group filter would be based on distance. A multicast packet could be restricted by either roundtrip delay or hop count so only members within the restricted range will receive the packet. This method is not exact as nodes outside the range will not receive the packets while nodes that do not want to receive these packets but are within the required range will receive them. This becomes even more difficult in a dynamic multicast tree topology that changes as new members join and old members leave the group.

The publish-subscribe model provides more accurate filtering for group communication. The subject-based subscription model requires that a multicast group is classified into subjects and the receivers will join the tree based on the subject it is interested in. A content based subscription model allows a node to filter information to only what it wants to receive within a multicast group. The publish-subscribe model approach not only requires a classification method for multicast groups but it requires one multicast group for one subject/content matter. The creation of a new multicast group for every subject or content delivered to receivers is not a feasible option due to the limited multicast addresses and increase in routing states in the network.

This type of filtering is usually done through brokers that filter the packets before it reaches the receivers. There are three methods to filter multicast packets, namely reduced precision, group clustering and multi-hop routing [10].

The group approximation algorithms and network flooding are methods that reduce precision but improve multicast scalability by generalizing members into large groups so that the number of groups can be reduced. This approach does not provide fine granularity of communication to group members.

In order to improve precision, many multicast groups with receivers interested in the same subject/content are clustered together under one or a collection of broker nodes. A copy of the same data packets will be sent to all the cluster brokers, which in turn will multicast the packets to all the multicast groups under them.

Multi-hop routing is a hierarchical approach to the clustered multicast group method. The data source will send packets to brokers close to it, who in turn will forward the packets to brokers near to them, and so forth until all brokers receiver the packets. These brokers will then multicast the packets to the groups located under them.

Filtering of packets within a multicast tree using a hierarchical label [11] method provides a more exact method of filtering. It provides a label to routers directly connected to a receiver node in a multicast tree. The label specifies the routers position on the multicast tree relative to a core router. Each router will build a hierarchical label tree by using its parents label as a prefix to its own label. A sender will address packets by the labels of the interested receivers and these packets will be routed along paths that connect these receivers to the sender. Such a scheme does not require applications to understand IP since a label is used to identify receivers. However, this method is limited to one-to-one communication and the labels will increase according to the depth and branches of the multicast tree. Levine and Garcia-Luna-Aceves [11] states that at least three bits are required for each integer that comprises a label.

In this paper, we propose a novel network-based filter on a ShT multicast tree. We do not look at SBT since the nature of such a tree only allows one sender in a group. In an enterprise, a truly dynamic and collaborative environment will have multiple senders with multiple receivers across branches or organizations. Since ShT multicast is an open communication protocol, our proposed network layer architecture will also allow public and open group membership to a subset of receivers that are interested in the communication and not to the rest of the nodes on the tree. Our proposed novel IP multicast group filter architecture can implement both subject-based and content-based publish-subscribe models.

III. SHARED TREE MULTICAST LABEL FILTER

The proposed architecture described below is an extension of work previously done by Singh et. al. [12, 13] for providing a label filter architecture for use by mobile nodes communicating through a ShT mobile multicast protocol. The label filters improve scalability, reduce network bandwidth usage and support fast transmissions

after handover for many corresponding nodes to communicate with a mobile node over a shared multicast tree. This work was then expanded to provide both restricted and public filters for improved admission control [14]. A private multicast label filter architecture was developed in [15]. Due to space constrains, interested readers can view these papers for details about the original mobile multicast label filter architecture and extended private label subgroup architecture.

This paper proposes using a static label filter to route packets along an existing multicast tree to a subsets of user in a subgroup. This label will be made public and open for nodes to easily join and leave the subgroup. Our architecture is different from the multiprotocol label switching (MPLS) protocol [16, 17].

The proposed architecture works only in an IP-based network and can easily be implemented in the existing routers through the "Router Policy Engine". Thus, the real-time implementation of the proposed algorithm does not require any router upgrade. However, the MPLS architecture requires all participating routers to implement the MPLS protocol.

MPLS is inherently implemented as a unicast protocol although a proposal for MPLS in a multicast environment can be found in [18]. MPLS routes packets by switching the labels in the packet as it is transmitted over the network. The proposed label filter architecture only uses one static label to identify the path leading to all the recipients. No label switching is involved and thus no extensive table lookups are required.

The label filter architecture is limited to a multicast tree and cannot be used to route packets beyond the multicast tree. This reduces the complexity of label management while improving scalability since the labels are localised to one multicast tree.

The proposed label filter architecture can effectively utilise the allocated QoS of the existing multicast tree. The proposed subgroup can be constructed in linear time whereas constructing a new multicast tree with the prescribed QoS either through the conventional multicast shared tree protocol or using MPLS [18] takes quadratic time in terms of the message and time complexity. This fact is explained in [13].

In this paper, the nodes will initially join the multicast group using the core-based tree (CBT) [19, 20] multicast routing protocol even though the filter architecture can be used with any shared tree multicast protocol. In addition to a core router, the shared tree will also need a label manager/broker to manage any new communication requests by group members. This label manager can be collocated with the core (or rendezvous point) router or any other router on the tree. How this label manager is elected will not be covered in this paper. The main purpose for the label manager is to provide admission control for the member nodes that want to communicate within a subgroup. Once a request is approved, the manager will provide a label for the path used in the communication. The manager will also set any limitations to the communication like session duration and QoS requirements.

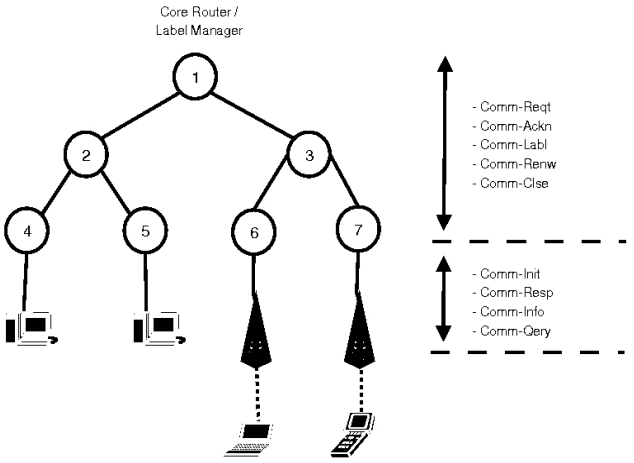

Figure 1. Control messages used to setup a label filter.

Fig. 1 shows the messages sent when setting up a label filter on the multicast tree. Like regular multicast control messages, the label filter control messages are segregated into end node (local membership) control messages and multicast on-tree routers (label management) control messages. The segregation of the messages allows the end-node messages to piggy-back with existing multicast control messages. Table 1 provides a brief description of the control messages.

Fig. 2 gives an example of a shared multicast tree with four nodes, i.e. nodes A, B, C and D where nodes A, B and C want to communicate between themselves on a different topic. The on-tree (OT) router OT1 is the multicast tree core and label manager. Node B is requesting the creation of a label where the label will be created from the label manager to all subgroup members. This public subgroup will enable member nodes to communicate only with each other

TABLE 1. LABEL FILTER CONTROL MESSAGES

Control Message	Membership	Description
Comm-Init	Local	Inform edge router to initiate new subgroup label
Comm-Info	Local	Edge router informs node of subgroup join requests, label group memberships, label creation status
Comm-Resp	Local	Node answers requests sent by subgroup label initiator or label manager
Comm-Qery	Local	Node queries edge router for every public subgroup label address in the multicast tree
Comm-Reqt	Management	Routers request label subgroup creation and inform nodes of the subgroup creation
Comm-Ackn	Management	Routers indicates permission to create or join label subgroup to recipient nodes
Comm-Labl	Management	Label Manager provides a token with this message to create a label on multicast tree
Comm-Renw	Management	Label Manager extends the duration of the label subgroup
Comm-Clse	Management	Label Manager tears down the label subgroup is the communication ends early

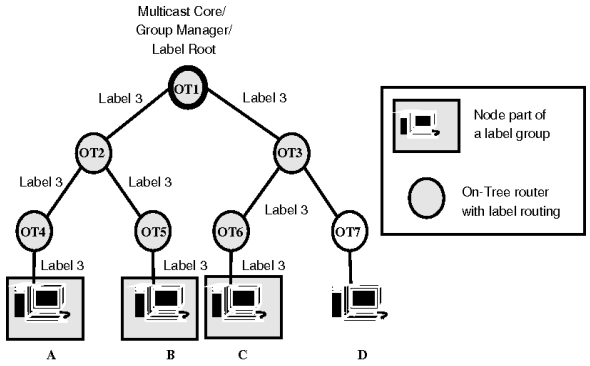

Figure 2. Multicast tree and label subgroups.

according to a specific subject/content, thus reducing information overload and network resource usage while increasing communication confidentiality.

A. Creating Public Subgroup Labels

The public subgroup label operation is explained in this section. For illustrative purposes, a public subgroup label between node B and node A will be created as shown in fig. 2.

While a node initiates the creation of a subgroup, the label manager will be the root of the requested subgroup. The label will first be created on this leaf router and travel outwards on every path connecting the root to all the other nodes interested in communicating using the label. The steps required for creating the label group are detailed below:

1. The initiating node will send a *Comm-Init* message to its leaf router. This message will contain the initiating node's address and a wildcard entry for recipient nodes along with other information regarding the creation of the label group. The leaf router will send a *Comm-Reqt* packet on the upstream path toward the multicast tree core router. If the core router is not the label manager, the core will forward the *Comm-Reqt* to the label manager. In the example network in fig. 2, Node B will send a *Comm-Init* message specifying the creation of a public label subgroup to its leaf router OT5 which in turn will send a *Comm-Reqt* message upstream to the label manager at OT1. This process is illustrated in fig. 3. The detailed algorithm to create a label filter is shown in fig. 4.

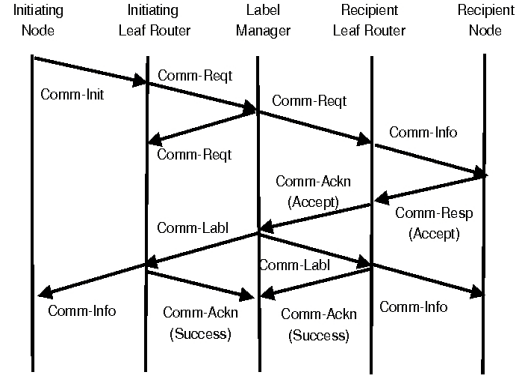

Figure 3. Public subgroup label creation process.

```
1  initiating node multicasts label creation request on multicast shared tree;
2  switch (the Node receiving label creation request) do
3      case (Label Manager)
4          if (Tree.Resources == available) and (Tree.Policy == Enable) then
5              Generate Comm-Ackn (Success) with label and token;
6          else
7              Generate Comm-Ackn (Failure);
8          end
9          Transmit Comm-Ackn on reverse path of Comm-Reqt;
10     end
11     case (Requested node)
12         if (Node wants to join the label sub-tree) then
13             Generate Comm-Ackn (Success);
14         else
15             Generate Comm-Ackn (Failure);
16         end
17         Transmit Comm-Ackn on reverse path of Comm-Reqt;
18     end
19     otherwise
20         Route packet to next router;
21     end
22 end
23 foreach (Router on identified label path) do
24     if (received packet is Comm-Ackn (Success)) then
25         Create label soft state;
26         Set soft state timer;
27     end
28     if (timer ≤ expiry time) and (receives label creation token) then
29         Change soft state label to permanent virtual label;
30     else
31         Flush soft state label;
32     end
33     Transmit packet to next router on label path;
34 end
35 if (timer ≈ label session expiry) then
36     initiating node sends renewal message (Comm-Renw) to label manager;
37     if (Label Manager extends label session) then
38         initiating node updates routers with new session time;
39     else
40         label path is torn down;
41     end
42 end
43 if (initiating node ends communication prematurely) then
44     initiating node tears down label path (Comm-Clse);
45 end
```

Figure 4. Algorithm to create a label filter path.

2. Once the label manager (OT1) receives the *Comm-Reqt* message, it will check the tree policy to allow a creation of a public label subgroup based on the request made by the initiating node. If the multicast group cannot allow the creation of a label subgroup, the manager will send a *Comm-Ackn* message with the failure flag set on the reverse path taken by the *Comm-Reqt* message otherwise it will multicast a *Comm-Ackn* message with the success flag set to all members in the shared tree. If the label manager approves the creation of this label subgroup, it will multicast a *Comm-Reqt* message to every node on the tree. The nodes will be informed about the creation of the label subgroup when the leaf router sends a *Comm-Info* message. If a node (node A) wants to join the label group, it will send a *Comm-Resp* message to its leaf router (OT4) which in turn will send a *Comm-Ackn* on the reverse path of the *Comm-Reqt* message. Any node that does not want to join the subgroup (node C, node D) will not do anything once it receives the *Comm-Info*. If the manager does not receive any *Comm-Ackn* from a node

besides the initiating node (node B), it will send a negative *Comm-Ackn* informing the initiating node that no nodes are interested in joining the label and release the label.

3. If the label manager (OT1) receives a *Comm-Ackn* message from nodes that want to join the label group, it will send a *Comm-Labl* message on the reverse path of the *Comm-Ackn* message to create a label (label 3) between itself and the joining nodes (node A). The label manager will also create a label path between itself and the originating node's leaf router (OT5) by following the reverse path of the *Comm-Reqt* message. Once the label has been successfully created, the leaf routers of all the joined nodes (OT4, OT5) will send a *Comm-Ackn* message using the label as acknowledgement. The routers that receive the *Comm-Labl* message will create an entry in their label routing table. Each table entry records four values: the label and port number for incoming packets and the label and port number for outgoing packets. For security purposes, the label will only be created if the *Comm-Labl* message contains a token created by the label manager.
4. Any of the nodes (node A, B) in the public label subgroup (label 3) can send a packet with the label so that only the nodes which are part of the sub-group will receive messages. A node will send an IP packet encapsulated with a link layer frame which uses the label for switching within the multicast tree. The end host will decapsulate this packet to receive the original IP packet. These nodes can still send a regular (not confidential) multicast packet to all members on the shared tree by not encapsulating the IP packet with a label.

B. Joining Public Subgroup Labels

The interaction within and between enterprises is dynamic, where departmental units and team members join and leave one or more online groups. Fig. 5 shows the operations for a new node joining an existing public label subgroup. We use the example from fig. 2 where node C joins the existing public label subgroup (label 3) to illustrate this process.

Since the public subgroup is controlled by the label manager, a label subgroup discovery method is required so that a node can quickly join the label subgroup without unduly increasing the load on the label manager. The steps below detail the operation for a node to join an existing public label subgroup.

1. The joining node C will send a *Comm-Qery* message to query the leaf router (OT6) for every public subgroup label address in the shared multicast tree. The leaf router will provide this information to node C in a *Comm-Info* packet.
2. Node C will select the subgroup label it wants to join (label 3) and will initiate the label join by sending a *Comm-Init* to its leaf router (OT6). The *Comm-Init* message will include the label address of the public subgroup the node wants to join and the address of the joining node. Router OT6 will send this request to the label manager using a *Comm-Reqt* message.
3. The label manager (OT1) will check its policy table for the joining node to become part of the label subgroup. There are two outcomes for this policy check at the label manager:
   (a) Approve – Most join requests for a public subgroup is given and the label manager will create a label path between itself and the joining node by sending a *Comm-Labl* message on the reverse path used by the *Comm-Reqt* message. As shown in fig. 2, the *Comm-Labl* will travel from OT1 to OT3 to OT6. Once the leaf router receives a *Comm-Labl* message, it will send an acknowledgement to the label manager (*Comm-Ackn*) and the joining node C (*Comm-Info*) regarding the successful joining and creation of a label path to the subgroup. The node is now successfully a member of the public label 3 subgroup.
   (b) Reject – There are situations where a node will not be allowed to join a public subgroup, such as when the policy limits the number of nodes in a public subgroup at any one time or if QoS is required for the communication and the path used by the new joining node fails to provide adequate QoS. The label manager will disallow the join request and it will send a negative *Comm-Ackn* to the joining node's leaf router (OT6). When the leaf router receives a negative *Comm-Ackn* from the label manager, it will send a *Comm-Info* stating the failure to join the subgroup to the joining node C.

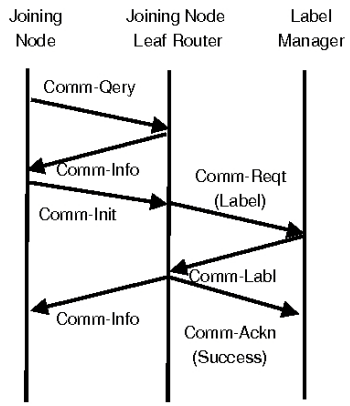

Figure 5. Public subgroup node joining process.

IV. ANALYSIS

This section analyses the savings of network resources on a shared multicast tree using the proposed label filter architecture. Let $G = (V, E)$ be the given network topology of the routers where V is the vertex set and E is the edge set. Only routers are modeled in the topology, denoted by $|V| = n$, with links denoted as $|E| = m$. For the purposes of this paper, the transmission medium will not affect the results of the analysis.

The label filter algorithm (fig. 4) will be compared with the time and message complexity used in creating a new CBT multicast tree [19, 20] for providing subgroup communication in a network. The time and message complexity to create a new CBT multicast tree is $O(n + m)$ while creating a new label filter subgroup is $m_1 + 2m_2 + p$ where m_1 is the number of messages sent on all paths in the multicast tree G from the label root, m_2 is the messages sent along the edges in the smallest sub-tree of G_1 containing the nodes $a_{i1}, a_{i2}, ..., a_{ir}$ for those nodes that want to join the label subgroup to the label root and p is the number of edges in the path between a_{i1} and the label manager. The details of this proof is provided in [13].

The level of network traffic is used as an indicator of network resource usage and scalability for concurrent group communication on a shared multicast tree. The proposed label filter architecture was simulated on two network topologies consisting of 15 to 150 nodes using the NS-2 network simulator [21]. A subgroup of nodes in these networks conducted audio communication among themselves using VoIP. Fig. 6 measures the packets sent on a multicast tree with and without implementing label filtering. It shows the average number of packets transmitted on a shared multicast tree when 3 nodes conduct VoIP communication among themselves. In the case of regular multicast, the packet count on the multicast tree is 60.98% higher than if the proposed label filter is used. The communication will also be received by the other 12 nodes in the network, thus exposing the communication to unwanted receivers. The label filter reduces the average packet duplication by 45.24% and maintains confidentiality among the participants on a public subgroup at a network level.

Our experiments, as illustrated in fig. 6 and fig. 7, show that the number of packets sent on the multicast tree will increase significantly on a shared multicast tree without using a label filter. This is because a multicast protocol will distribute the packets on all paths on the multicast tree until all nodes attached to the tree will receive the packets. The number of packets sent on the multicast tree will increase as the number of senders and the size of the multicast tree grows as shown in fig. 7.

With the label filter, the sender nodes will transmit packets only to the receiver nodes instead on all network

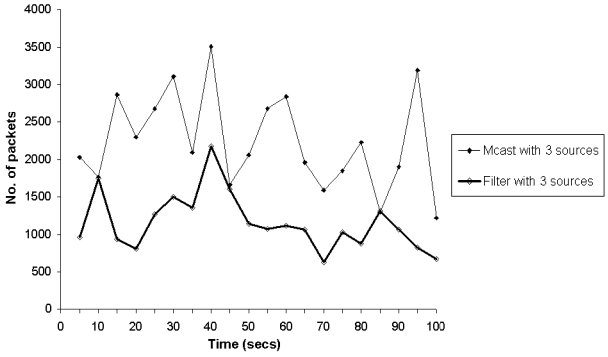

Figure 6. Average number of packets transmitted by 3 senders to members in a public subgroup.

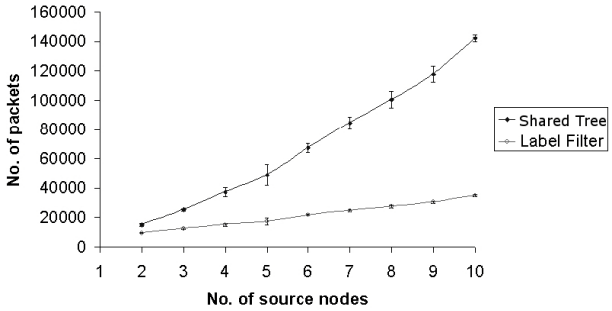

Figure 7. Average number of VoIP packets transmitted among many source nodes in a public subgroup.

paths on the multicast tree. In the case of multiple source nodes, the number of packets sent on the multicast tree will increase since the number of source nodes has increased. However, the total number of packets transmitted over the multicast tree when using a label filter is reduced by as much as 36.79% to 75.48%, with a 95% CI. This reduction in network traffic is important since the network resources can be utilized for other network traffic as well as providing more open access subgroup communication channels for nodes that require to communicate in specialized groups in an enterprise network.

The initial simulation results show that the proposed label filter architecture reduces network bandwidth usage and provides privacy during subgroup communication in an enterprise multicast tree. The label filter architecture can provide both subject based and content based filtering as a subgroup can be formed to cater for both instances. In future papers, this work will be extended to compare the bandwidth usage of the proposed label filter architecture against other publish-subscribe systems.

V. CONCLUSION

This paper presents a new networked group collaboration architecture using label filters on a shared multicast tree. The use of the proposed label filter architecture will increase the adoption of multicast technology as a more efficient and scalable networked group communication protocol. The proposed label filter architecture provides more efficient group communication by sharing an existing multicast tree for public subgroup communication among selected nodes, hence improving scalability by reducing the number of IP multicast addresses used for each sub-group based on subject/content. The label filter architecture can handle large number of node members as well as large number of subgroups. The on-tree routers do not need to create a new multicast entry for a sub-group but can use the 'router policy engine'. Other publish-subscribe architectures using multicast technology are not as scalable as they still aim to connect multiple multicast trees using a broker middleware. We have also shown in our previous work that the time and message complexity of setting up a labeled path is much lower than setting up a new multicast tree. This further proves the use of our proposed label filter improves scalability for group communication. The proposed label filter architecture also provides better use of available

bandwidth and network resources like QoS as shown by the simulated experiments. This will allow for more optimized group communication, leading to an increase in collaboration among branches and across organizations. The proposed architecture will simplify and enhance dynamic real-time collaboration among users in small ad-hoc groups within a large enterprise.

REFERENCES

[1] S. E. Deering and D. R. Cheriton, "Multicast routing in datagram internetworks and extended lans," *ACM Transactions on Computer Systems (TOCS)*, Vol. 8, No. 2, May 1990, pp. 85-110.

[2] M. Handley, I. Kouvelas, T. Speakman, and L. Vicisano, "Bidirectional protocol independent multicast (BIDIR-PIM)," IETF, RFC 5015, October 2007.

[3] D. Meyer and P. Lothberg, "GLOP addressing in 233/8," IETF, RFC 3180, September 2001.

[4] L. Wei and D. Estrin, "The trade-offs of multicast trees and algorithms," in *Proc. of the 4th Intl. Conf. on Computer Communications and Networks*, 1995, pp. 150-157.

[5] P. Paul and S. V. Raghavan, "Survey of multicast routing algorithms and protocols," in *Proc. of the 15th Intl. Conf. on Computer Communication*, Mumbai, India, 2002.

[6] G. Banavar, T. Chandra, B. Mukherjee, J. Nagarajarao, R. E. Strom, and D. C. Sturman, "An efficient multicast protocol for content-based publish-subscribe systems," in *Proc 19th IEEE Intl. Conf. on Distributed Computing Systems*, 1999, pp. 262-272.

[7] B. Quinn and K. Almeroth, "IP multicast applications: Challenges and solutions," IETF, RFC 3170, September 2001.

[8] Y. Xiao, X. Du, J. Zhang, F. Hu, and S. Guizani, "Internet protocol television (IPTV): The killer application for the next-generation internet," *IEEE Communications Magazine,* Vol. 45, No. 11, 2007, pp. 126-134.

[9] S. Langerman, S. Lodha, and R. Shah, "Algorithms for efficient filtering in content-based multicast," in *Proc. 9th Annual European Symposium Algorithms (ESA)*, Denmark, 2001, pp. 428-439.

[10] L. Opyrchal, M. Astley, J. Auerbach, G. Banavar, R. Strom, and D. Sturman, "Exploiting IP multicast in content-based publish-subscribe systems," in *Proc. of the IFIP/ACM Intl. Conf. on Distributed systems platforms*, New York, USA, 2000, pp. 185-207.

[11] B. N. Levine and J. J. G. Luna-Aceves, "Improving internet multicast with routing labels," in *Proc. of the Intl. Conf. on Network Protocols (ICNP)*, 1997, pp. 241-250.

[12] J. Singh, P. Veeraraghavan, and S. Singh, "Implementing label filters on a shared tree mobile multicast architecture," in *Proc. of the IEEE Intl. Conf. on Networks (ICON)*, Kuala Lumpur, Malaysia, 2005.

[13] J. Singh, P. Veeraraghavan, and S. Singh, "Performance of a shared tree multicast label filter architecture," in *Proc. of the IEEE Intl. Conf. on Networks (ICON)*, Kuala Lumpur, Malaysia, 2005.

[14] J. Singh, P. veeraraghavan, and S. Singh, "Network based filtering architecture on a shared multicast tree," in *Proc. of the IASTED Intl. Conf. on Networks and Communication Systems (NCS)*, Chiang Mai, Thailand, 2006, pp. 390-395.

[15] J. Singh, P. Veeraraghavan, and S. Singh, "Dynamic real-time IP-based publish-subscribe group collaboration using multicast label filters " in *Proc. of the Intl. Conf. on Digital Ecosystems and Technologies (DEST)*, Istanbul, Turkey, 2009.

[16] E. C. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," IETF, RFC 3031, January 2001.

[17] U. Black, *MPLS and label switching networks*, 2nd ed. New Jersey, USA: Prentice Hall, 2002.

[18] D. Ooms, B. Sales, W. Livens, A. Acharya, F. Griffoul, and F. Ansari, "Overview of IP multicast in a multi-protocol label switching (MPLS) environment," IETF, RFC 3353, August 2002.

[19] A. Ballardie, "Core based trees (CBT) multicast routing architecture," IETF, RFC 2201, September 1997.

[20] A. Ballardie, "Core based trees (CBT version 2) multicast routing: Protocol specification," IETF, RFC 2189, September 1997.

[21] The Vint Project, "The ns network simulator," Available from: http://nsnam.isi.edu/nsnam/index.php [Accessed 12 January 2009].