

Network Based Filtering Architecture on a Shared Multicast Tree

Jaipal Singh, Prakash Veeraraghavan and Samar Singh
Applied Computing Research Institute
Department of Computer Science and Computer Engineering
La Trobe University
Victoria 3086, Australia
email: {jaipal.singh, p.veera, s.singh}@latrobe.edu.au

ABSTRACT

This paper introduces a shared tree multicast filter that uses labels to create one-to-one, one-to-many and many-to-many communication on an existing multicast tree. All the receivers wishing to communicate privately will be connected virtually by a label to form a subtree within an existing multicast tree. The data packet is sent to the receivers on paths identified by the label. A label manager on the tree will create and manage these labels. This paper describes the benefits of using our filter architecture compared to creating a new multicast tree or using another proposed filter architecture.

KEY WORDS

Multicast Networks, Multicast Filter

1 Introduction

The next step forward in computer networking is ubiquitous computing where the user is surrounded by smart, intuitively operated devices that help the user organise, structure and master his or her everyday life. For this to become a reality, these ‘smart’ devices will require an efficient communication protocol for these devices to cooperate within an environment. We believe that a multicast routing protocol should be the dominant communication protocol used in ubiquitous computing.

Multicast routing protocols were introduced as a more effective method for one-to-many, many-to-one or many-to-many communication [1] over a shared network link. This shared link is a logical network tree which connects all nodes who want to receive the same communication and is represented by a multicast group address. The sender will address its packet to the multicast address and every node that subscribes to this group (address) will receive the packet. Unlike unicast, only a single instance of the data packet will be transmitted on every link in the multicast tree.

Network resources can be better utilised if a subset of member nodes of an existing larger multicast tree can reuse the existing multicast channel for private

communication between one or more member nodes without other member nodes receiving the data packets. Some researchers have proposed application layer solutions that work on top of the existing multicast routing protocol. These application layer protocols filter packets based on the intended recipient nodes.

The application layer protocol solution filters packets at the end routers or host nodes. The data packets will travel on every tree link and be dropped at either the end router or at the host for nodes that are not the recipients. If the multicast group is large, but only a small number of nodes want to communicate with each other, a lot of network resources will be wasted since packets will still be transmitted to nodes that are not part of this new subgroup.

Some other researchers have proposed a network layer solution to subgroup communication to minimise the overheads to network resources. The details of both application and network layer protocols currently used to filter or differentiate communication on a multicast tree are contained in section 2.

The prime motivation for this paper is to reduce network resource wastage for subgroup communication on a multicast tree. The network layer multicast filter protocol will only transmit packets on links that connect to nodes interested in communicating within the subtree. This will lead to a reduction of network resource wastage on the existing multicast tree. An added benefit is that the allocated quality of service (QoS) of the multicast group can be more efficiently utilised by this subgroup since it is built on top of the existing tree. We will describe our network layer filtering architecture in section 3. The advantages of our shared tree filter is presented in section 4 and we conclude this paper in section 5.

2 Related Work

A multicast group uses different network routing protocols depending on whether the group members are located densely or sparsely. A dense mode multicast group uses source-based tree (SBT) routing where a

multicast group will only have one sender and the receivers are densely grouped together in the network. If multiple nodes on the tree want to send messages, N multicast groups have to be formed where N is the number of senders. A sparsely populated multicast group can use either SBT or a shared multicast tree (ShT). Unlike SBT, ShT can have multiple senders on the same tree. ShT overcomes the scalability problem of SBT since all nodes whether sender or receiver use the same tree. However, ShT might incur more delay than SBT since all traffic has to go through a central core router.

To ease joining and leaving of multicast groups on a SBT group, the Internet Engineering Task Force (IETF) has come up with the Internet Group Management Protocol (IGMP) version 3 [2] which allows receivers to dynamically filter sources the receiver wants to listen to. A receiver will use an IGMPv3 message to instruct the end router which sources it wants to receive (INCLUDE mode) and which sources it does not want to receive (EXCLUDE mode) packets from. The router will join SBT groups specified in the INCLUDE list and leave those multicast groups the receiver is not interested in anymore based on the EXCLUDE list. The interaction between IGMPv3 and multicast routing protocols is detailed in [3]. The Multicast Listener Discovery (MLD) version 2 [4] functions similarly to IGMPv3 and is used in IPv6 networks.

Although IGMPv3 eases the joining and leaving of groups based on the source (sender), it can only be used on a SBT group and it still does not solve the problem of creating a new multicast tree for every new subgroup communication.

A multicast routing filter that routes packets to interested receivers on the same multicast tree would reduce this network wastage since only paths that contain members that want to receive the files will get the packet and no new multicast tree needs to be formed. The simplest filter would be based on distance. A multicast packet could be restricted by either round-trip 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 packet while nodes that do not want to receive these packets but are within the required range will receive them. This becomes even more difficult as the multicast tree topology will change as new group members are added and old members are removed from a tree.

For a more exact method of identifying members on a tree, a hierarchical label [5] is given 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, the drawback is that the labels will increase according to the depth and branches of the multicast tree. [5] states that at least three bits are required for each integer that comprises a label. This kind of filter is limited to one-to-one communication.

Multicast filters can also be implemented on the application layer. The publish-subscribe model uses subject-based subscription or content-based subscription [6] by applications to send and receive information it is interested on a multicast tree. 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 its interested in. A content-based subscription model allows a node to filter information to only what it wants to receive within a subject-based group. This publish-subscribe model approach not only requires a classification method for multicast groups but it still has the problem of creating one multicast group for one subject matter.

In this paper, we are interested in network-based filters on a ShT multicast tree. We do not look at SBT since the nature of such a tree only allows one sender for a group. However for ShT, every receiver on the tree will get the multicast packet. Our proposed network layer architecture will ensure that packets are only sent to the subset of receivers that are interested in the communication and not to the rest of the nodes on the tree.

3 Proposed Architecture

If a node on a multicast tree wants to communicate privately with one or more nodes on the same shared tree, we propose implementing a network based filter architecture that allows one-to-one, one-to-many and many-to-many communication only between a subset of members on a shared multicast group. The network filter can be used to create dynamic communication groups within an existing multicast shared tree. The multicast tree will use static labels to pass messages along the tree to members that are authorised to receive the private communication.

The proposed architecture is completely independent from the MPLS architecture [7]. Our proposed scheme 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 our 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 [8].

MPLS routes packets by switching the labels in the packet as it is transmitted over the network. Our scheme 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. Our filter architecture is limited to a multicast tree and cannot be used to route packets beyond the multicast tree.

Our proposed multicast filter architecture can effectively utilise the allocated QoS of the existing multicast tree. The proposed sub-tree 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 [8] takes quadratic time in terms of the message and time complexity. This fact is explained in [9].

In this paper, we assume that the nodes will join the multicast group using the core-based tree (CBT) [10] multicast routing protocol although any shared tree multicast protocol can be used with our network multicast filter. In addition to a core router, the shared tree will also need a label manager to manage any new communication requests by group members. This label manager can be 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 privately. 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 shows the messages sent when setting up a label filter on the multicast tree. These messages are segregated into messages sent to the end node and messages sent among the multicast on-tree routers. The segregation of the messages allows the end-node messages to piggy-back with existing multicast control messages.

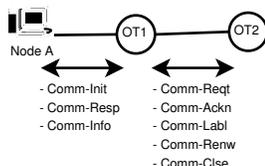


Figure 1. Messages used by member nodes and on-tree routers to setup a label filter

The Comm-Init message is used to inform the first hop on-tree router that a node wants to communicate privately with one or more nodes on the same tree. A member node indicates its approval or rejection to a communication request by sending a Comm-Resp

message to the first hop on-tree router. The Comm-Resp message is also used to send an acknowledgement once a label path is created. An on-tree router sends Comm-Info messages that inform the end nodes about any communication requests, the decision of the label manager regarding a communication request and whether the label has been created successfully.

The Comm-Req message is sent by the routers to the label manager and last hop (leaf) routers of the nodes to request the creation of a label group and inform nodes of the creation of the group. Source routing is used to send a Comm-Ackn message on the reverse path taken by the Comm-Req message.

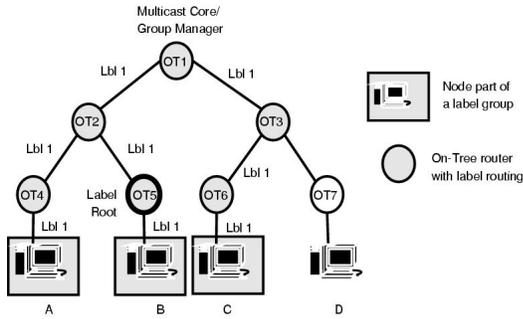
A Comm-Ackn message is sent by the label manager or the leaf routers connected to recipient nodes to indicate permission to create or join the label group. The Comm-Labl message is used to create a label along the routers connecting all the nodes interested in filtering their communication. The Comm-Renw message is used to extend the duration of a label group while the Comm-Clse message is used to tear down the label group if the communication ends earlier than the expiration time.

Figure 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 setup a new communication between themselves. 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 node B or the label manager depending on whether node B requests a public or restricted label. Figure 2(a) shows that the root will be on the leaf router (OT5) of the initiating node (node B) if node B restricts the nodes who can communicate on the label. If the nodes who will join the label group is unknown or any node can join the subtree, the label manager will be the root for the label as shown in figure 2(b).

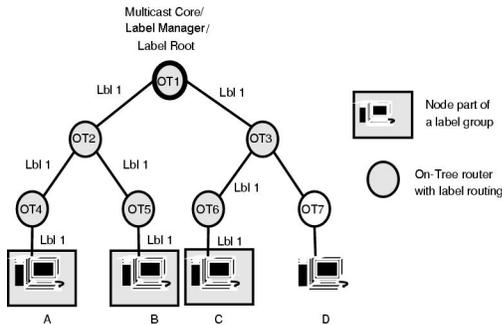
3.1 Restricted Label Group

When the initiating node has prior knowledge of the nodes joining the label group and wants to only allow selected nodes to join this group, the leaf router for the initiating node will be the central router or root for this label group. 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 inform its leaf router to request the creation of a label by sending a Comm-Init message to the leaf router. This message will contain the initiating node's address and the names of the intended recipients along with other information regarding the creation of the la-



(a) Label group root is the initiating node's (node B) leaf router (OT5)



(b) Label group root is the label manager (OT1)

Figure 2. Nodes linked by a label to form a subgroup within an existing multicast shared tree

bel group. The leaf router will multicast a Comm-Req message to all leaf routers and the label manager to request permission from the label manager to create the label and to find the path to the recipient nodes.

In the example network in figure 2(a), node B will send the Comm-Init message to router OT5 which in turn will multicast a Comm-Req message.

2. Once the label manager receives the Comm-Req message, it will check the tree policy to allow a creation of a label group based on the request made by the initiating node. If the multicast group cannot allow the creation of a label group, the manager will send a Comm-Ackn message with the failure flag set on the reverse path taken by the Comm-Req message otherwise it will send a Comm-Ackn message with the success flag set.

A successful Comm-Ackn message will contain the computer names of the approved parties for the subgroup communication, any limits on such a communication, the label to be used to forward the packets and a token which gives the originat-

ing leaf router authorisation to create the label in the multicast tree. Once the initiating node receives the Comm-Ackn from the label manager, it will wait until it receives a successful Comm-Ackn from the recipient nodes before creating the label.

While the label manager is processing the Comm-Req message, every leaf router in the multicast tree will also receive the same Comm-Req message. The leaf router will send a Comm-Info message to every node on the tree informing them about the creation of the label group. The nodes whose names are not listed in the Comm-Info message will drop the packet while each one that is listed will send a Comm-Resp message indicating whether it accepts or rejects the joining of this label group. The leaf router will send a Comm-Ackn indicating interest in joining back to the originating router based on the value of the Comm-Resp message.

If the originating node does not receive a Comm-Ackn from the manager or recipient nodes, it will multicast the Comm-Req message again. If the originating node gets a Comm-Ackn message indicating failure, it will do nothing further.

If we refer to the example, the label manager OT1 will receive the Comm-Req message from OT5. OT1 will then send a Comm-Ackn message on the reverse path taken by the Comm-Req message (OT1 → OT2 → OT5). All of the leaf routers OT4, OT6 and OT7 will receive the Comm-Req message and will send a Comm-Info to nodes A, C and D respectively. Node D will drop the packet since it's not the intended recipient while nodes A and C will send a Comm-Resp to their leaf routers indicating whether they accept or reject joining the label group request from node B. Routers OT4 and OT6 will send a Comm-Ackn message with the success flag set on the reverse path taken by the Comm-Req back to OT5.

3. The leaf router of the initiating node will create the label approved by the manager from itself to the leaf routers of all the recipients that accepted the label group request. The leaf router will send a Comm-Labl message on the reverse path used by the Comm-Ackn messages. 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.

Once the leaf router of a recipient node creates the label, it will send a Comm-Info message to the node informing it about the successful creation of the label path. The leaf router will also send a Comm-Ackn message using the label to inform

the originating leaf router that the path label has been successfully created.

Router OT5 will send a Comm-Labl message to OT4 for node A on $OT5 \rightarrow OT2 \rightarrow OT4$ and $OT5 \rightarrow OT2 \rightarrow OT1 \rightarrow OT3 \rightarrow OT6$ for node C. Both OT4 and OT6 will send Comm-Info message to nodes A and C respectively and will also send Comm-Ackn messages using the new label to indicate the successful creation of the label path.

Any of the nodes in the label group can send a packet with the label so that only the nodes which are part of the label group will receive this message. The 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 send a regular multicast packet to all members on the tree by not encapsulating the IP packet with a label.

3.2 Public Label Group

If an initiating node wants to create a label group but does not know who might be interested in joining this group or it wants this label to be public to all nodes on the tree, it will inform the label manager to create a label and be the root router for the label group. The label manager will multicast to every node in the tree if they are interested in joining this new label group. The label will be created from the label manager connecting every node that wants to join this label group. The steps required for creating this type of 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-Req message on the upstream path toward the multicast tree core router. If the core router is not the label manager, the core will send the Comm-Req to the label manager.
- In the example given in figure 2(b), node B will send the Comm-Init message to router OT5 which in turn will send a Comm-Req message upstream to the core ($OT5 \rightarrow OT2 \rightarrow OT1$).
2. The label manager will check the tree policy regarding creating the label based on the information from the Comm-Req message. If the manager will not allow the creation of this label group, it will send a negative Comm-Ackn message on the reverse path used by the Comm-Req message.

However, if the manager approves this label group, it will multicast a Comm-Req message to every node on the tree. If a node wants to join the label group, its leaf router will send a Comm-Ackn on the reverse path of the Comm-Req message. Any node that does not want to join the leaf router will not do anything once it receives the Comm-Req. If the manager does not receive any Comm-Ackn from a node besides the initiating node, it will send a negative Comm-Ackn informing the initiating node that no nodes are interested in joining the label and release the label.

As an example, the label manager OT1 will multicast a Comm-Req message to every leaf router OT4, OT5, OT6 and OT7. Nodes A, B and C will send a Comm-Resp message informing their respective leaf routers their intention of joining the label group. Their leaf routers will send a Comm-Ackn on the reverse path of the Comm-Req back to the label manager. Node D will not do anything further since it does not want to join the label group.

3. Once the label manager receives the 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 between itself and the joining node. Once the label has been successfully created, the leaf router will send a Comm-Ackn message using the label as acknowledgement.

In figure 2(b), the label manager OT1 will send Comm-Labl messages to create labels to node A on $OT1 \rightarrow OT2 \rightarrow OT4$, to node B on $OT1 \rightarrow OT2 \rightarrow OT5$ and to node C on $OT1 \rightarrow OT3 \rightarrow OT6$. These leaf routers will send an acknowledgement by transmitting a Comm-Ackn message using the newly created label.

4 Discussion

Based on the description of our proposed multicast shared tree filter in section 3, this architecture will provide benefits in terms of efficient usage of available network resources and faster packet delivery.

In a shared tree, the ability to filter packets to different receivers enables the nodes to reuse any existing provisioned resources like network path and quality of service (QoS). Rather than provision new resources for a new communication request, the members of an existing shared multicast tree can utilise the same path and available QoS on the multicast tree.

Besides better use of network resources, the filter will reduce the usage of IP multicast addresses since a new group does not have to be formed. The saving

on using new multicast group addresses is important in an IPv4 network as these addresses are very limited [11]. The filter architecture is scalable since it uses a numerical label which is only relevant within the multicast group. Thus, many multicast groups can use the same numerical label at the same time.

The label filter architecture retains the anonymous feature of the existing multicast protocols. The label is only used to identify paths on a multicast tree and not the individual nodes connected to the tree. The location of a node is anonymous even within a labelled group unlike in [5] where every node is identified by a label. This architecture can be easily extended for use in infrastructure-based mobile multicast networks where the nodes move around in the network. The node might change locations but the path label will still be the same. The implementation architecture of the label filter in a mobile multicast scheme like mobile core-based tree is shown in [12].

The use of label filters also improves the delay during packet delivery. The label routing table can be sorted and quickly found. If a binary tree is used to search for a label in the routing table, the time taken to search is $\log(n)$ where n is the number of labels in the routing table. The packet delivery will also be quicker than normal IP packet delivery since layer-two switching is used. Layer-two switching is quicker to process compared to layer-three routing.

Based on the advantages of using a shared tree filter, we are running simulations to show the time taken to form a label group compared to creating a new multicast tree. We will present our findings in a later paper although our theoretical proofs are presented in [9].

5 Conclusion

In this paper, we have presented a new scalable network layer filter mechanism using label routing on a multicast shared tree. A node on the tree can perform one-to-one, one-to-many or many-to-many communication with a subset of nodes within the same multicast group.

The label routing architecture described in section 3 brings benefits to an IP multicast architecture in terms of reduction of IP multicast addresses, faster packet delivery and better use of existing network resources like QoS. The network will not need to create new multicast groups for new communication between nodes already on the same multicast tree.

With this filtering architecture, we hope that multicast communication will be more widely deployed in the current Internet environment since one multicast tree can now support subtree communication without high network wastage. We believe this will simplify and more efficiently enable more devices to

communicate with each other and the user in an ubiquitous computing environment.

References

- [1] B. Quinn and K. Almeroth, "IP multicast applications: Challenges and solutions," IETF, RFC 3170, September 2001.
- [2] B. Cain, S. Deering, B. Fenner, I. Kouvelas, and A. Thyagarajan, "Internet group management protocol, version 3," IETF, RFC 3376, October 2002.
- [3] B. Haberman and J. Martin, "IGMPv3/MLDv2 and multicast routing protocol interaction," IETF, Internet Draft draft-ietf-magma-igmpv3-and-routing-04, January 2003.
- [4] R. Vida, L. H. M. K. Costa, S. Fdida, S. Deering, B. Fenner, I. Kouvelas, and B. Haberman, "Multicast listener discovery version 2 (MLDv2) for IPv6," IETF, RFC 3810, June 2004.
- [5] B. N. Levine and J. Garcia-Luna-Aceves, "Improving internet multicast with routing labels," in *Proc. of the Intl. Conf. on Network Protocols*, 28-31 October 1997, pp. 241-250.
- [6] 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, 3-7 April 2000, pp. 185-207.
- [7] U. Black, *MPLS and Label Switching Networks*, 2nd ed. Prentice Hall, 2002.
- [8] 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.
- [9] 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, 16-18 November 2005.
- [10] A. Ballardie, "Core based trees (CBT version 2) multicast routing: Protocol specification," IETF, RFC 2189, September 1997.
- [11] S. Deering, "Host extensions for IP multicasting," IETF, RFC 1112, August 1989.
- [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, 16-18 November 2005.