Implementing Label Filters On A Shared Tree Mobile Multicast Architecture

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 describes an architecture that filters packets within a subset of nodes on an existing shared multicast tree. The path connecting the group of nodes that want to communicate privately on the existing tree will be given a label. These labels are used to route one-to-one and group communication traffic for selected nodes on a multicast tree. Nodes connected to the tree but are not on this label path will not receive any filtered packets. This filter architecture reduces network resource waste by utilising the existing network resources on the multicast tree like quality of service (QoS). In this paper, we also describe how this architecture can be used for mobile communication when implemented in a shared tree mobile multicast architecture.**

I. Introduction

The recent uptake of wireless devices has increased the demand for mobile computing. Users want their wireless devices to provide seamless connection to the network when stationary or when mobile at any time and anywhere. IP has been suggested as the network protocol to be used for all mobile communication, including voice communication.

Mobile IP [1] and Mobile IPv6 [2] are two such protocols proposed to provide seamless mobility support on an IPv4 and IPv6 network respectively. A mobile node will have two IP addresses, a permanent home address and a care-of-address. The permanent home address is used for identifying the node and for maintaining transport and higher layer connections. The care-of-address is used to route the packet to the current location of the mobile node in a foreign network. Mobile IP introduces two new network agents, a home agent (HA) in the mobile node's home network and a foreign agent (FA) in the foreign network.

Any data packets sent to the mobile node will be addressed to the node's home address. The packet will reach the HA which will encapsulate it in another IP packet addressed to the care-of-address. This encapsulated packet will be tunnelled to the care-of-address (the FA) in the foreign network, where the IP encapsulation will be stripped and the original packet sent to the mobile node.

Although mobile IP provides seamless mobility in a mobile network, it performs poorly when a mobile node is frequently handed over from one access point to another. Mobile IPv4 suffers from triangular routing which contributes to higher resource usage and increased packet latency for every packet sent from a corresponding node (CN) to the mobile node. Mobile IPv6 uses route optimisation where the CN bypasses the HA to communicate directly with the mobile node. However, route optimisation suffers from binding update latency when the mobile node informs its current location after handoff to every CN. Both versions of mobile IP incur delay when registering the location of the mobile node with the HA. These latencies will become more pronounced if the mobile node is far away from the HA.

In addition to latency during handoff, mobile IP does not support group communication well. Multicast was developed for the fixed network and mobile IP implements it inefficiently in a mobile network [3]. The mobile node can join a multicast group in two ways: using remote subscription or via a bi-directional tunnel to the HA.

If the first method is adopted, the mobile node needs to rejoin the multicast tree with a different care-of-address every time it performs a handoff. This is not the best method for highly mobile devices since the setup latency during handoff and in joining the group will be high. Remote subscription should be used if quality of service (QoS) is important and the mobile node is stationary for long periods of time.

In bi-directional tunnelling, all multicast packets are tunnelled from the HA to the mobile node. A tunnel has to be built from the HA to the FA every time a mobile node performs a handoff. Tunnelling multicast packets defeats the purpose and benefits of multicast. A serious drawback is the tunnel convergence problem where multiple HAs create a tunnel between themselves to one FA where all their mobile nodes are visiting. If all of these mobile nodes are part of the same group, having multiple tunnels sending the same packet to one location is a waste of resources.

There have been proposals to use a multicast mobility architecture to provide faster handoffs than mobile IP [4], [5], [6]. If such an architecture is used, it will have the added benefit of providing multicast support for group communication in mobile devices. The use of a source-based tree (SBT) multicast mobility architecture was proposed by Helmy [5] and a shared-tree (ShT) multicast mobility architecture was proposed by Castelluccia [6] and Jaipal et al. [7], [8].

Helmy's SBT approach creates a multicast group based on the source (corresponding) node. The CN is connected to the

1-4244-0000-7/05/$20.00 ©2005 IEEE.

mobile node on a multicast tree. Each multicast tree will only consist of one CN and one mobile node. The CN will create a unicast packet which will be encapsulated in a multicast packet and send on the tree to the mobile node where it will decapsulate the multicast packet to receive the original unicast packet.

The SBT multicast architecture is not scalable as the number of source nodes in the multicast group increases. A SBT multicast group can only have one sender (CN) in a multicast group. If another node wants to send packets in the group, a new delivery tree has to be created. The total number of routing entries for the SBT protocol is S X G where S is the number of source nodes and G is the number of multicast groups.

To overcome the shortcoming of SBT, the ShT approach creates only one multicast tree that is shared by all nodes in the group. Many CNs can now join the multicast group and send packets to the mobile node on the tree. Castelluccia proposed in [6] a mobile ShT architecture using IPv6 where multiple CNs communicate with the mobile node on a multicast tree. A CN will multicast a data packet with the mobile node's unicast address in an IPv6 destination option header. The mobile node will replace the multicast address with the unicast address from the option header once it receives the packet.

Although ShT allows multiple nodes to send on the same multicast tree, every other node on the tree will also receive the packet. A ShT multicast group supports one-to-many and many-to-many communication. In the case of mobile devices, a packet sent by one CN to the mobile node will also be received by every other CN on the tree. Security and network utilization is a big issue on a shared tree since the packets will be sent to every node on the tree, including those nodes that do not want the packet.

Another issue with all of the multicast mobility approaches stated earlier is the use of asymmetrical network paths for packet delivery. The CN will use multicast to send packets to the mobile node but the mobile node will unicast packets to the CN. Although bi-directional communication on the same multicast tree is more efficient, it cannot be done using SBT which required one group for one sender. For bi-directional multicast, two SBTs have to be created. A sparse mode ShT multicast protocol can provide two-way communication although the packets will be received by every CN on the tree even if it is meant for the mobile node or a particular CN.

Jaipal et al. in [7], [8] proposed a refinement to the ShT mobile multicast architecture called Mobile Core-Based Tree (M-CBT). The M-CBT architecture uses bi-directional multicast communication only on the fixed wired network components. The mobile node is connected to the tree via the access point but is not part of the multicast group. This decoupling of the wireless element from the fixed network gives more flexibility to the mobile devices where they can use any network layer protocols to communicate with the access point and by extension the rest of the network. The access point will transform the data packet into an IP packet before sending it through the wired multicast tree. The access point will also transform any IP packets received from the tree to a format readable by the mobile node. The M-CBT architecture also uses multicast for two-way communication between the CN and mobile node. However, the problem of every node on the ShT receiving a packet meant for just one particular node on the tree makes ShT protocols unsuitable for wide implementation in mobile networks.

In this paper, we present a shared tree network-based filter that will only send packets to the intended recipients on the tree. The prime motivation for this paper is to reduce network resource wastage for subgroup (including one-to-one) communication on a multicast tree. We detailed any related work in multicast filtering in section II. Our own network layer filtering architecture is described in section III and the time and message complexity of our filtering algorithm is presented in [9]. We conclude this paper in section IV.

II. Related Work

Most of the proposed multicast mobility architectures implement sparse mode multicast routing protocols like PIM-SM [10] or CBT [11] multicast protocols. The PIM-SM protocol can be used for both source-based (SBT) and shared tree (ShT) routing. A SBT multicast group can only have one sender in the whole group while the ShT group can have multiple senders on the same multicast tree. In a SBT multicast group, the sender is connected on the shortest path to the receiver (mobile node) whereas in a ShT group all nodes are connected to a central core router for the whole group.

The main problem with using a mobile multicast protocol is keeping the communication between the mobile node (MN) and corresponding node (CN) private from the other members of the same multicast group. A SBT architecture like Helmy's PIM-SM protocol [5] does not have this problem since only one node on the tree can be the sender and the MN will be the sole receiver on the tree. A new multicast group and tree has to be created for a every node that wants to send packets using SBT multicast. The number of multicast groups created is N groups where N is the number of CNs communicating with the MN. The MN will join and leave a tree depending on which CN it wants to receive packets from. To do so, the MN will need to know the multicast group address for every CN it wants to communicate with.

To simplify this task, the Internet Engineering Task Force (IETF) has come up with the Internet Group Management Protocol (IGMP) version 3 [12] which allows receivers (MN) on a SBT to filter sources the receiver wants to listen to. IGMPv3 allows a receiver to dynamically join or leave multicast groups based on the sources it is interested in. A receiver will send an IGMPv3 message to a multicast router (m-router) specifying which sources it wants to receive (INCLUDE mode) and which sources it does not want to receive packets (EXCLUDE mode) from. The m-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 [13]. The Multicast Listener Discovery (MLD)

version 2 [14] functions similarly to IGMPv3 and is used in IPv6 networks.

The source filter feature of IGMPv3 can only be used in SBT multicast trees. IGMPv3 still does not overcome the scalability problem of creating a new multicast group every time a new node wants to send packets on the tree. In a ShT, no new groups have to be created as the number of CNs increase although a packet sent by one CN will be received by the MN and all the other CNs on the tree. In this situation, the delivery of packets to other members of the multicast group is wasteful. ShT multicast protocols will not be widely implemented for mobile communication until the multicast tree can limit delivery of packets from a CN to the MN without sending the packet to other nodes on the tree.

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 packets 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 filtering traffic on a tree, a hierarchical label [15] 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, this architecture does not scale well as the labels will increase according to the depth and branches of the multicast tree. Three bits of storage is required for each digit of the label. Another drawback is that this kind of filter can only work for one-to-one communication where the sender node knows the label used by the receiving node.

The added benefit of filtering on a ShT is the use of existing paths and any quality of service (QoS) already provisioned for that path. Members of a group might not fully utilise the reserved QoS on a multicast tree. A network layer filtering architecture will enable members of a subgroup which are part of the same multicast tree to better utilise the available resources. This type of filtering approach can only be used if the subgroup uses best-effort traffic or QoS that can be allocated to the subgroup by the multicast tree.

Multicast filters can also be implemented on the application layer. The publish-subscribe model uses subject-based subscription or content-based subscription [16] by an application to send and receive information it is interested in on a multicast tree. In this paper, we are interested in network-based filters on a ShT multicast tree. We do not consider SBT multicast since it is not scalable. 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 ShT tree. Our architecture can be used for one-to-one, one-to-many and many-to-many communication on a multicast tree.

III. Proposed Architecture

We propose a network based filter architecture that allows one-to-one, one-to-many and many-to-many communication between a subset of member nodes on an existing shared multicast tree for use in a mobile multicast architecture. We are proposing the use of labels to identify paths connecting nodes engaged in private communication on the multicast tree. The multicast routing protocol will route messages along the tree based on these labels to members that are authorised to receive the private communication.

In this paper, we assume that the nodes will join the multicast group using the Mobile Core-Based Tree (M-CBT) [7], [8] 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 the labels used in the multicast tree. This label manager can be the core router or any other router on the tree. How the label manager is selected will not be covered in this paper.

The main purpose of the label manager is to provide admission control for the creation of new labels on the existing multicast tree. 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.

Fig. 1 shows the control messages used in setting up a label filter on the multicast tree. These control messages are segregated into messages sent by host nodes to the on-tree router and messages sent between on-tree routers. The communication initiation (*Comm-Init*), communication response (*Comm-Resp*) and communication information (*Comm-Info*) messages are messages sent between a host node and the on-tree router it is directly connected to. The communication request (*Comm-Reqt*), communication acknowledgement (*Comm-Ackn*), communication label (*Comm-Labl*), communication renew (*Comm-Renw*) and communication close (*Comm-Clse*) messages are sent between on-tree routers (including the label manager). The segregation of the messages allows the label control messages to piggy-back with existing multicast control messages.

The following steps show how a label is created in a shared multicast tree.

1) The initiating node sends a *Comm-Init* message to instruct the on-tree router directly connected to it to request the label manager to create a label for subgroup

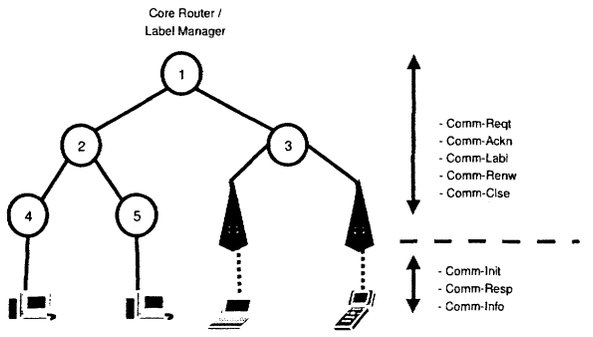

Fig. 1. Control messages used by member nodes and on-tree routers to setup a label filter

communication. This message carries the IP address or network name of the node initiating the communication request, the name or address of the node or nodes it wants to communicate with, a flag indicating uni-directional or bi-directional communication, a flag for closed or open communication, the duration of the communication and any QoS requirements.

The node address stored in the *Comm-Init* does not have to be an IP address. Most nodes are known at the application layer by their computer name rather than their IP address. Any on-tree router connected to a node will accept the packet using this identification. The *Comm-Init* message direction flag can be set to request that the label be used to route packets in only one direction (sender to receiver) or in both directions. The message has a 'label type' flag to indicate whether the label is open to the public or is closed to only the requested nodes. If the label group is open to the public, any node can join the label group by requesting the label manager to grant the node a path using the label. However if it is a close communication, the initiating node will have to inform the label manager which nodes can join the label group before the manager can grant access to the new requesting node.

An initiating node can specify the duration of the communication. If no duration is specified, the label manager will specify the duration for the communication. The *Comm-Init* is also used to specify if the communication requires QoS better than best-effort.

2) Once an on-tree router receives the *Comm-Init* message, it will send a *Comm-Reqt* message to the label manager. The *Comm-Reqt* message will contain all the information in the *Comm-Init* message and will be multicast on the tree until it reaches the label manager and the on-tree routers directly connected to the requested nodes. This message will keep a record of all on-tree routers on the path to the label manager and the path to the requested nodes.
3) The label manager checks the multicast tree policy (including resource availability if QoS is required) regarding creating labels for the initiating node after it receives the *Comm-Reqt* message. The manager will send a *Comm-Ackn* message to the initiating node with the flag set to success if the requested communication is allowed by the tree policy otherwise the flag is set to fail. If the *Comm-Ackn* flag is set to success, the *Comm-Ackn* carries a token which allows the initiating node to instruct on-tree routers to create the label. The details of the token are outside the scope of this paper.
4) An on-tree router that is directly connected to a recipient node listed in the *Comm-Reqt* message will send a *Comm-Info* message to the node. The *Comm-Info* is used to inform the node that the initiating node wants to communicate privately with this node using a label filter on the multicast tree. The *Comm-Info* message is also used to inform a node about the decision of the label manager regarding a communication request and whether the label has been created successfully.
5) After receiving the *Comm-Info* request, the recipient node will decide to join the new label sub-group. The node will send a *Comm-Resp* message to the on-tree router with this decision. The *Comm-Resp* message is used as an acknowledgement for any control messages received by a *Comm-Info* message.
6) The recipient node's on-tree router will send a *Comm-Ackn* message informing the initiating node whether the recipient node will join the label sub-group or not. This message will be sent on the reverse path of the *Comm-Reqt* message to the initiating node.

   Since member nodes on a multicast tree are anonymous, a method is required to find a path back to the node that sent a message on the tree. The *Comm-Reqt* and *Comm-Ackn* messages will keep a record of the path used to reach the label router or destination node. This way, the label router or destination node will be able to find the path to the originating node by using source based routing [17].
7) The initiating node's on-tree router will receive the *Comm-Ackn* from the label manager and recipient nodes. If the label creation is approved, the on-tree router will send a *Comm-Info* to the node with the token given by the label manager. The node will instruct the on-tree router using a *Comm-Resp* message to create a label from itself to the recipient nodes. This *Comm-Resp* message will be accompanied by the token.
8) The on-tree router will send a *Comm-Labl* message with the token provided by the label manager to create a label path between itself and the on-tree routers leading to the recipient nodes. The token provides authorisation for the *Comm-Labl* to create labels along on-tree routers. The path to the recipient nodes is the reverse path of the *Comm-Ackn* message from the recipient node.

When a node requests a label from the label manager, the label manager might put a time limit on the label. If a node wants to continue using the label after the duration expires, it

needs to send a *Comm-Renw* message to the label manager. If the label manager agrees to extend the duration of the label, it will send a *Comm-Ackn* with a new token to the originating node.

The label architecture also caters for early termination of the label. If the originating node wants to terminate the label earlier, it will send a *Comm-Clse* message to all on-tree routers that route the label.

A. Label Filters in a Mobile Multicast Architecture

The label filter architecture described in section III can be used to create a new multicast subgroup connected by labels within an existing multicast tree or to communicate privately (one-to-one) between two nodes on an existing tree. In the case of a mobile multicast architecture, the assumption is every node on the tree would want to have one-to-one communication with the mobile node.

When a mobile node initially registers with an access point, the mobile node will send a label token along with the registration message. The token used by the MN can only be used to create a default label in the multicast tree. This default label is a uni-directional label from the label manager to the mobile node catering for best-effort traffic only. This label is an open label where any node in the multicast tree can send packets to this label. Only one mobile node can create a default label in the multicast tree.

Since this is a new connection and the multicast tree has not been set up, the network will elect a core router which will be the label manager. The token is used to create a default label between the access point and core router. The core router can delegate the label manager function to another router but will always have a connection between itself and the label manager. For simplicity, we will assume the multicast shared tree core router and label manager are the same in this paper.

Fig. 2 gives an example of a shared mobile multicast tree with four nodes, ie. CN A, CN B, CN C and MN. The MN creates the multicast group and joins the core router. The path between MN and the core router will have a default label (label 1). The other nodes (including mobile CNs) will join the multicast tree if they want to communicate with MN. If a CN wants to communicate with every node on the tree, it will use regular multicast. However, if the CN only wants to communicate with MN, it will route the packet using the default label.

If a CN wants to send a packet to the MN, it will encapsulate the IP packet with a data link layer wrapper that uses the default label as the destination address. If the on-tree router does not have routing information for a label, it will send the label upstream towards the core (label manager). The label manager is the central depository that has full knowledge of all the labels used in the multicast tree. If CN C sends a packet to MN, OT7 will send the packet upstream to OT3 which in turn will forward it upstream to the core router. The core router has an entry in its routing table for the default label and will forward the packet through that port. The packet will go to OT2 followed by OT5 before reaching MN.

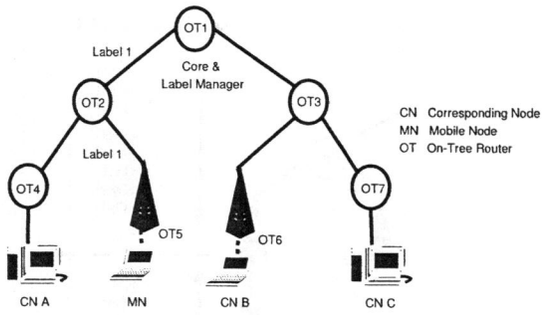

Fig. 2. M-CBT tree with 3 corresponding nodes and a mobile node

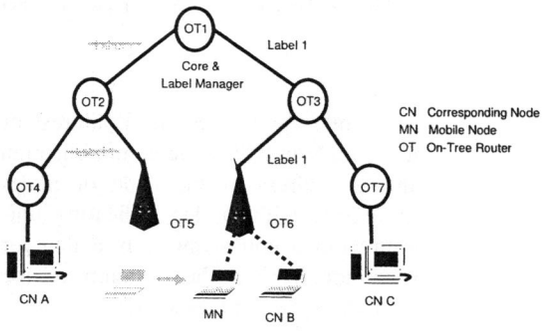

Fig. 3. Mobile node handoffs to a new access point

There might be cases where the packet might encounter the label path before reaching the core router. If CN A sends a packet to the default label, it will send the packet upstream to OT4. However, when the packet reaches OT2, the router has a routing entry for the label. The router will route the packet based on its routing table entry to OT5 instead of sending it upward to the core router.

With the use of a default label, the mobile node can be contacted using one-to-one communication by the other multicast tree nodes. However, if the mobile node wants to communicate with a corresponding node using a label, it will have to request a label from the label manager as described in section III.

B. Mobile Handoff

Unlike traditional wired network architectures, a wireless node can also be a mobile node. Our label filter architecture can be easily integrated with a mobile multicast architecture to ensure that communication between the corresponding node and mobile node are not interrupted even after the mobile node has been handed over to a new access point.

Fig. 3 shows the location of the nodes in the network after MN is handed over from OT5 to OT6. The M-CBT architecture uses advance registration to ensure that every time the mobile node performs a handoff, the new access point is already a part of the multicast tree. This make before break approach ensures packets are not lost during the handoff process.

When MN detects a stronger signal from OT6, it will register with this access point as described in [7], [8]. Since OT6 is already a part of the multicast tree, it does not have to join the multicast group. However, it still needs to create a new label path from the access point (OT6) to the label manager (OT1). OT6 will send a *Comm-Labl* message with the default token to create a default label from itself to the label manager. A flag in the *Comm-Labl* is set to indicate this label creation is done due to a handoff operation. Once the label manager receives the new *Comm-Labl*, it will block the port to the old default label path. If the new default label path intersects the old label path before reaching the label manager, the old label path will be blocked from the intersecting router instead of from the label manager.

Once the label manager receives the new default label path and blocks the old label path, it will send a *Comm-Ackn* to the new access point. At this time, the mobile node is connected to the tree at two points, OT5 and OT6. Any filtered packets from CN A will be received from OT5 while filtered packets from CN B and C will be received from OT6. If the connection at OT6 is successful, MN will send a leave message with a *Comm-Clse* to OT5 (old access point). The default label will be destroyed from OT5 to the label manager and the routers will leave the tree if no other nodes are interested in the multicast group.

In the case where MN is an initiating node for another label group, it will perform the same process as migrating a default label during handoff. However, if the MN is a regular member of the label group, how it sets up the label path after handoff depends on whether the label group is an open or closed group. In the case of an open group, any node can join this group and the MN sends a *Comm-Reqt* to the label manager. The label manager will create a label path from itself to the new location of the mobile node. However, if the group is a closed group, the mobile node informs the initiating node of its pending handoff. The initiating node will loan the mobile node its token which it will use to rebuild a label path from the new access point to an intersecting router. Once the new label path has been created the token will expire. However, if the new path does not provide the required QoS the label path operation will fail.

IV. Conclusion

In this paper, we propose a network label path filter that can be used to filter communication between a subgroup of nodes within a shared multicast tree. Only nodes on the label path will receive packets sent with the corresponding label. Any nodes not on the path will not receive the packets and network resource wastage will be reduced. The network filter also improves security as the packets are only routed from the source node to the destination node or nodes. Any other nodes that are not authorised to receive the packets will not get the packet.

We show the performance of using our label filter algorithm against creating a new multicast tree in [9]. The label filter algorithm as described in section III takes linear time to form a subgroup whereas creating a new M-CBT tree takes quadratic time in terms of time and message complexity. The M-CBT protocol used with the label filter algorithm makes mobile multicast a viable alternative to mobile IP for supporting mobility.

References

[1] C. E. Perkins, Editor., "IP mobility support for IPv4," IETF, RFC 3344, August 2002.

[2] D. Johnson, C. Perkins, and J. Arkko, "Mobility support in IPv6," IETF, Internet Draft draft-ietf-mobileip-ipv6-24, July 2003.

[3] V. Chikarmane, C. L. Williamson, R. B. Bunt, and W. Mackrell, "Multicast support for mobile hosts using mobile IP: design issues and proposed architecture," *Mobile Networks and Applications*, vol. 3, no. 4, pp. 365–379, 1998.

[4] J. Mysore and V. Bharghavan, "A new multicasting-based architecture for internet host mobility," in *Proceedings of the 3rd annual ACM/IEEE International Conference on Mobile Computing and Networking*, September 1997, pp. 161–172.

[5] A. Helmy, "A multicast-based protocol for IP mobility support," in *Proceedings of the Networked Group Communication NGC*, November 2000, pp. 49–58.

[6] C. Castelluccia, "A hierarchical mobility management scheme for IPv6," in *Proceedings of 3rd IEEE Symposium on Computers and Communications (ISCC).*, Greece, 30 June - 2 July 1998, pp. 305–309.

[7] J. Singh, P. Veeraraghavan, and S. Singh, "Core based tree multicast (M-CBT) approach in supporting mobility," in *Proceedings of the IASTED International Conference Parallel and Distributed Computing and Networks (PDCN)*, Innsbruck, Austria, 17-19 February 2004, pp. 147–152.

[8] ——, "Core-based tree multicast M-CBT approach in supporting mobility using IPv6," Dept. of Comp Sci. and Comp Eng., La Trobe University, TechnicalReportNo. 1/05, March 2005.

[9] ——, "Performance of a shared tree multicast label filter architecture," in *Proceedings of the IEEE International Conference on Networks (ICON)*, Kuala Lumpur, Malaysia, 16-18 November 2005.

[10] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C.-G. Liu, P. Sharma, and L. Wei, "Protocol independent multicast-sparse mode (PIM-SM): Protocol specification," IETF, RFC 2362, June 1998.

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

[12] B. Cain, S. Deering, B. Fenner, I. Kouvelas, and A. Thyagarajan, "Internet group management protocol, version 3," IETF, RFC 3376, October 2002.

[13] B. Haberman and J. Martin, "IGMPv3/MLDv2 and multicast routing protocol interaction," IETF, Internet Draft draft-ietf-magma-igmpv3-and-routing-04, January 2003.

[14] 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.

[15] B. N. Levine and J. Garcia-Luna-Aceves, "Improving internet multicast with routing labels," in *Proceedings of the International Conference on Network Protocols*, 28-31 October 1997, pp. 241–250.

[16] L. Opyrchal, M. Astley, J. Auerbach, G. Banavar, R. Strom, and D. Sturman, "Exploiting IP multicast in content-based publish-subscribe systems," in *Proceedings of the IFIP/ACM International Conference on Distributed systems platforms*, New York, USA, 3-7 April 2000, pp. 185–207.

[17] J. Postel, "Internet protocol," IETF, RFC 791, September 1981.