Performance Of A Shared Tree Multicast Label Filter 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 defines a new multicast filter algorithm. This algorithm is used to filter packets on a mobile multicast architecture using a multicast shared tree. In a mobile multicast architecture, communications between a corresponding node (source) and the mobile node (receiver) should be private and not be sent to every node (receivers) on the multicast tree. We propose using an algorithm that sets up a label sub-tree on an existing mobile multicast shared tree to filter packets based on these labels. Our proposed label filter architecture is implemented differently to the current MPLS architecture. In this paper, we validate the effectiveness of the label filter in mobile communication compared to the traditional method of creating a new multicast tree by analysing the message and time complexity of the algorithm against the setting up time of a new multicast shared tree.**

I. Introduction

The next generation protocols for mobile Internet is an active area of research. Future mobile devices will migrate from circuit-switched networks to packet-switched networks using IP as the future routing protocol. However, traditional IP does not support mobile devices since routing is based on the network prefix of an IP address. Due to the way IP addressing was developed, any transport layer connection to the mobile node will have to be recreated as the mobile node changes its IP address as it moves from one domain subnet to another domain subnet. This is because the protocol as well as the routers are not designed to accommodate dynamism.

Mobile IP [1] and Mobile IPv6 [2] were designed to provide seamless mobility support on an IPv4 and IPv6 network respectively. Mobile IP works by using two IP addresses, a permanent home address for identifying the node and maintaining transport layer connections and another address called a care-of-address for routing the packet to the actual location of the mobile node. Corresponding nodes will communicate with the mobile node using it's home address. Once the packet reaches the mobile's home network, the packet is tunnelled to the mobile node's care-of-address by a home agent (HA). A foreign agent (FA) in the foreign network will receive and strip the encapsulated packet and forward the original packet to the mobile node.

Although Mobile IP provides seamless mobility support, its performance is greatly dependant on the frequency of mobile node hand off and the distance between the mobile node and the HA. Mobile IP incurs high latency when the mobile node is far away from the HA and the mobile node is frequently handed over from one access point to another. Mobile IPv4 also suffers from triangular routing which contribute to higher resource usage and increased packet latency. Triangular routing introduces asymmetrical routing for two-way communication between the corresponding node and the mobile node. Although Mobile IPv6 uses route optimisation to overcome the triangular routing problem, it still suffers from binding update delays since the mobile node has to directly inform every corresponding node its care-of-address.

Mobile IP is a unicast protocol and is mainly used for one-to-one communication. Although Mobile IP supports multicast protocols for group communication, it implements it inefficiently [3]. A mobile node can use remote subscription or bi-directional tunnels to join a multicast group.

A mobile node using remote subscription will join the multicast group with a different care-of-address every time is performs a handoff. This is not a good method for highly mobile devices since the setup latency after handing over is high. This method should only be used for mobile nodes that require quality of service (QoS) and are stationary for long periods of time.

In bi-directional tunnelling, the mobile node is connected to the multicast tree via the HA. The HA will tunnel any multicast traffic to the mobile node. Tunnelling multicast packets defeats the purpose and benefits of multicast. A serious drawback to bi-directional tunnelling 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.

Some multicast based mobility schemes have been proposed to overcome the shortcomings of Mobile IP. Multicast based mobility schemes have shown better handling of mobile hand-offs compared to Mobile IP [4], [5], [6], [7] and they natively support group communication in mobile devices. The use of a source-based tree (SBT) multicast mobility architecture was proposed by Helmy in [5] and a shared-tree (ShT) multicast mobility architecture was proposed by Castelluccia in [6] and Jaipal et al. in [7], [8].

1-4244-0000-7/05/$20.00 ©2005 IEEE.

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

Since Helmy's SBT architecture can only connect one CN to one mobile node, the SBT approach is not scalable as the number of CNs wishing to communicate with the mobile node increases. To overcome the scalability problem of SBT, the ShT mobile multicast architecture proposed by Castelluccia was developed where multiple senders (CN) can communicate with a mobile node on the multicast tree. Castelluccia's ShT architecture uses IPv6 packets where 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.

The mobile multicast architecture proposed by Jaipal et al. improved on Castellucia's work by implementing two-way multicast communication on the shared tree and decoupling the mobile node from the fixed network architecture. The performance of all these mobile multicast architectures is presented in [7].

All three mobile multicast architectures provide best-effort routing. If QoS traffic is required on the multicast tree, a multicast QoS mechanism like YAM [9], QMRP [10], QoSMIC [11] or EBPM [12] has to be used to setup a QoS path on the multicast tree. Jaipal et al. proposed implementing the M-CBT mobile multicast architecture with EBPM for setting up a quick QoS multicast tree for mobile networks in [12].

Although the ShT architecture enables multiple CNs to communicate with a mobile node on one multicast tree, the communication will be received by every other node on the tree. Not only is this a waste of network resources, but it is also a serious security problem that needs to be solved. This problem becomes more pronounced when the same multicast shared tree is used for two-way communication between the CN and mobile node as proposed in the M-CBT architecture.

The easiest method of implementing private communication on a multicast tree is to create new multicast trees for a subgroup of nodes. The is a time and resource intensive process and should not be undertaken for short communication sessions.

We propose instead the use of a network layer label filter to create a virtual sub-tree within an existing multicast shared tree. Our algorithm is implemented at the router level and does not require any changes to the existing multicast routing protocol for it to work. The proposed architecture is completely independent from the Multi Protocol Label Switching (MPLS) architecture [13]. Our proposed scheme works only in an IP multicast tree and can easily be implemented in the existing multicast 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. Currently, MPLS is inherently implemented as a unicast protocol although a proposal for MPLS in a multicast environment has been suggested in [14]. This proposal uses MPLS labels to create a logical multicast tree and to route packets to every node connected to the tree by a label. This architecture does not provide a filtering mechanism for communication between selected nodes on the tree.

Our proposed multicast architecture can effectively utilise the allocated bandwidth, in general the quality of service (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 [15], [16] or using MPLS [14] takes quadratic time in terms of the message and time complexity. This fact is explained in section III.

In this paper, we analyse the time and message complexity of creating a virtual subgroup using our proposed label filter algorithm and compare it against creating a new M-CBT multicast shared tree. We will provide a brief overview of the label filter algorithm in section II. We then present our theoretical proofs regarding the performance of the filter algorithm in section III before we conclude this paper in section IV.

II. Label Filter Algorithm

A brief description of our label filter algorithm is provided in fig. 1 based on the description of the label filter architecture presented in [17]. A detailed description of other proposed multicast filter architectures is also presented in the same paper.

III. Theoretical Proofs and Analysis of Multicast Label Filter

In this section, we theoretically analyse the time and message complexity of our proposed multicast label filter architecture against creating a new multicast tree like Mobile Core-Based Tree (M-CBT) as presented in [7], [8].

Let $G = (V, E)$ be the given network topology of the routers where V is the vertex set and E is the edge set. We denote $\mid V \mid = n$ and $\mid E \mid = m$. When modelling the topology, we only consider the routers and not the end nodes (hosts) associated with the routers. The routers are fixed nodes connected by wireless (airwaves) or wired links. However this is not going to affect the results presented in this paper. For standard Graph-theoretic terms not defined in here, one may refer to Bondy and Murty [18].

We present below an algorithm to construct a Mobile Core Based Tree (M-CBT) [7], [8] multicast shared tree for the nodes attached to the routers $a_1, a_2, \cdots, a_k$ where $a_i \in V(G)$, $1 \leq i \leq k$, with the assumption that the complete topology is known to every node (like in the case of any link state routing protocol). Before this, we give the definition of the *predecessor-successor*.

The **i-th neighbourhood** of a vertex u is defined as $N_i(u) = \{x : d(u, x) = i\}$ and we denote $N_1(u)$ by simply

```
initiating node multicasts label creation request on
multicast shared tree;
switch (the Node receiving label creation request) do
    case (Label Manager)
        if (Tree.Resources == available) and
        (Tree.Policy == Enable) then
            Generate Comm-Ackn (Success) with label
            and token;
        else
            Generate Comm-Ackn (Failure);
        end
        Transmit Comm-Ackn on reverse path of
        Comm-Reqt;
    end
    case (Requested node)
        if (Node wants to join the label sub-tree) then
            Generate Comm-Ackn (Success);
        else
            Generate Comm-Ackn (Failure);
        end
        Transmit Comm-Ackn on reverse path of
        Comm-Reqt;
    end
    otherwise
        Route packet to next router;
    end
end

foreach (Router on identified label path) do
    if (received packet is Comm-Ackn (Success)) then
        Create label soft state;
        Set soft state timer;
    end
    if (timer ≤ expiry time) and (receives label
    creation token) then
        Change soft state label to permanent virtual
        label;
    else
        Flush soft state label;
    end
    Transmit packet to next router on label path;
end

if (timer ≈ label session expiry) then
    initiating node sends renewal message
    (Comm-Renw) to label manager;
    if (Label Manager extends label session) then
        initiating node updates routers with new
        session time;
    else
        label path is torn down;
    end
end

if (initiating node ends communication prematurely)
then
    initiating node tears down label path
    (Comm-Clse);
end
```

Fig. 1: Algorithm to setup label path sub-tree on a multicast shared tree

```
graph MCBT-1(graph G; vertices a_1, a_2, ..., a_k);
begin
    graph G_1 = (V_1, E_1) with V_1 = E_1 = ∅;
    Execute BFS(a_1) to get the
    predecessor-successor relation;
    for (i=1; i ≤ k; i++) do
        flag processed(a_i) = 0;
    end
    for (i=2; i ≤ k; i++) do
        repeat
            V_1 = V_1 ∪ {a_i};
            if (predecessor of a_i > 1) then
                arbitrarily choose a predecessor ;
            end
            x = predecessor of a_i ;
            if (processed(x) == 1) then
                break ;
            else
                V_1 = V_1 ∪ {x};
                E_1 = E_1 ∪ {x a_i};
                a_i = x;
            end
        until (x == a_1);
    end
    return(G_1);
end
```

Fig. 2: Algorithm to setup M-CBT multicast tree

$N(u)$.

Definition 1: Let $u \in V(G)$ and $y \in N_i(u)$, for some $i \geq 1$. Then y is a **successor** (or a **child**) of x with respect to u if $x \in N_{i-1}(u)$ and $xy \in E$. x is also called a **parent** or a **predecessor** of y.

Now consider the successor relation with respect to u on G. Let $y \in N_j(u)$. We say y is a **descendant** of $x \in N_i(u)$ if $i < j$ and there exists an $x - y$ geodesic path of length $j - i$. In this case x is also called an **ancestor** of y.

By a neighbourhood structure of G with respect to u, we mean the predecessor-successor relation on $V(G)$ defined with respect to u.

Basically the pseudocode in fig. 2 is used to construct a M-CBT multicast delivery tree rooted at a_1, in which one or more vertices (a_i) are leaf-nodes. The above tree may not be the optimal tree (i.e. tree with minimum number of edges). However the algorithm can be slightly modified to output an optimal tree. But this will incur more processing time. We now analyse the complexity of the pseudocode in fig. 2.

Complexity Analysis Executing the breath first search (BFS) takes $O(n + m)$ in time and message complexity. For $a_i (i \geq 2)$, which is at level r (i.e. $a_i \in N_r(a_1)$) with respect to the BFS(a_1), finding a shortest path to a_1 takes r-time. Also r messages are exchanged in G to establish

a communication path. But at some level, if a predecessor of a_i is already processed, the "repeat" loop will be broken and the next a_i will be considered. Thus every vertex and edge in the outputted graph are processed once and hence the message and time complexity is $O(n_1 + m_1)$, where n_1 is the number of vertices in the outputted graph and m_1 is the number of edges. Since G_1 is a tree, $O(n_1 + m_1) = O(n_1)$. Hence the total time taken to construct the multicast tree is $O(n + m + n_1 + m_1) = O(n + m)$.

The M-CBT algorithm is modified slightly to construct a multicast tree with the required link QoS (like bandwidth). From the original network topology G, we remove all the links which do not provide the link QoS. This takes $O(m)$ time, as we need to scan through all the edges of G exactly once to construct the resultant graph G'. If G' is disconnected and a_is are in different components of G', then it will not be possible to construct a multicast tree providing the adequate QoS. However if all a_is are in the same component of G', we can construct a multicast tree providing the required QoS.

To detect this, the algorithm in fig. 2 needs to be modified as shown in fig. 3. To test whether G is disconnected or not and whether a_is belong to different components of G takes $O(n+m)$ time and message. Thus the total time and message complexity to construct the multicast tree with the required link QoS remains the same as implemented using the algorithm shown in fig. 2.

The algorithm in fig. 3 cannot be applied to the case where we want the end-to-end QoS (like end-to-end latency). For constructing a multicast tree with the required end-to-end QoS, we implement a more sophisticated algorithm like the Explore Best Path Multicast (EBPM) algorithm, which is presented in [12].

In several practical scenarios (like ad hoc networks), a node may not have the complete knowledge about the network topology. Any node will know only about its local neighbourhood, i.e. nodes which are directly connected to it (or in the same broadcast domain). Also in a typical multicast environment, the server will form the multicast tree first, then the clients will join one after another.

Theorem 1: The message and time complexity of $k(k \geq 2)$ or more nodes associated with the routers $a_1, a_2, \cdots, a_k$ to form a M-CBT multicast tree, where every router has only the local knowledge is $O(k(n + m))$.

Proof The M-CBT [7], [8] algorithm construct the multicast delivery tree in stages. Initially the tree will consists of a single node a_1 (in reality, it is the mobile node connected to this router which also happens to be the multicast tree core router). Node a_2 joins the tree through the shortest path connecting a_1 and a_2, then a_3 joins the existing tree through a shortest path connecting a_3 and some on-tree router x such that the distance $d(x, a_3)$ is minimum among all on tree nodes. If there are more than one such node x, we can randomly choose one from the available on-tree routers. Note that while selecting a path, we are not concerned about the QoS offered at every router along the path.

```
graph MCBT-1(graph G; vertices a_1, a_2, ..., a_k);
begin
    Construct G' from G by removing all the links
    which does not provide link QoS;
    graph G_1 = (V_1, E_1) with V_1 = E_1 = ∅;
    Execute BFS(a_1) to get the
    predecessor-successor relation;
    if (G' is disconnected) and (a_is belong to
    different component of G') then
        return(void);
    end
    for (i=1; i ≤ k; i++) do
        flag processed(a_i) = 0;
    end
    for (i=2; i ≤ k; i++) do
        repeat
            V_1 = V_1 ∪ {a_i};
            if predecessor of a_i > 1 then
                arbitrarily choose a predecessor ;
            end
            x = predecessor of a_i ;
            if processed(x) == 1 then
                break ;
            else
                V_1 = V_1 ∪ {x} ;
                E_1 = E_1 ∪ {xa_i} ;
                a_i = x ;
            end
        until (x == a_1);
    end
    return(G_1);
end
```

Fig. 3: Algorithm to setup M-CBT multicast tree with link QoS

In general if r-nodes $a_1, a_2, \cdots, a_r$ already formed a tree, and a_{r+1} wish to join the tree, a_{r+1} will connect to an on-tree router x with the property that $d(x, a_{r+1})$ is minimum among all on-tree routers x. Thus the worst case message and time complexity of a_{r+1} joining the tree is $O(n + m)$. The worst case complexity for k-nodes joining one after another is $O(k(n + m))$.

Remark If G is dense, then the M-CBT formed by these nodes will be sparse, even if the tree has the same number of vertices as that of G. However the complexity of constructing the M-CBT will be increased and is bounded by $O(n^3)$.

If we require a multicast tree with a required link QoS, as before, we execute the algorithm presented in fig. 3. Before that, we need to determine whether all a_is belong to a single component of G'. Thus in this case too the complexity (message and time) remain the same.

We now analyse the time and message complexity of the

proposed filtering algorithm and compare it with the MCBT setup time.

Let the nodes $b_1, b_2, \cdots, b_r$ wish to communicate with each other, initiated by b_1. If these are the entire nodes involved in the multicast tree G_1, there is no need to create a separate tree. They still can commnicate using the existing tree G_1. If $r = 2$, and b_2 is the mobile node that initiated the multicast, b_1 can use the default label to communicate with the mobile node (b_2). Under any other case, b_1 needs to contact the label manager to create a label path.

Theorem 2: Let the nodes $b_1, b_2, \cdots, b_r$ wish to communicate with each other, initiated by b_1. $a_{i_1}, a_{i_2}, \cdots, a_{i_r}$ are the routers in which the nodes $b_1, b_2, \cdots, b_r$ are connected to. The number of messages sent along the network to get the approval/rejection from the label manager is $m_1 + m_2 + p$, where m_2 is the number of edges in the smallest subtree of G_1 containing the nodes $a_{i_1}, a_{i_2}, \cdots, a_{i_r}$ and p is the number of edges in the path between a_{i_1} and the label manager.

Proof According the the proposed algorithm, the end host (here it is b_1) will initiate the label creation process by sending a "Comm-Init" packet. This will be received by a_{i_1}, which will multicast along the tree. The complexity of this is m_1. This message will be received by $b_2, b_3, \cdots, b_r$ and the label manager. The label manager will look at the tree policy and available resources. Based on this, either the session will be approved or rejected. This will go along the reverse path from the label manager to a_{i_1}, which has the message complexity p. The nodes $b_2, b_3, \cdots, b_r$ transmit their acknowledgement to join the private communication through the reverse path. All the routers which receive this acknowledgement will create a soft-state until it get the piggy-backed connection request consisting of the label and other connection related parameters together with the initial data. The message complexity is m_2. Thus the total time and message complexity is $m_1 + m_2 + p$.

IV. Conclusion

In this paper, we prove that our proposed filtering algorithm has the following advantages over setting up a new M-CBT multicast tree for new multicast or unicast communication between end hosts that are members of the same multicast group:

1) The proposed filtering algorithm is used for creating a short life private communication as a side-track to the main multicast communication. Here the private communication may not even last longer than the time to create a new tree.
2) If the nodes $b_1, b_2, \cdots, b_r$ create a new muticast tree, it takes $O(r(n + m))$ time and messages, whereas the proposed algorithm takes only $m_1 + m_2 + p$ time and messages to create the virtual sub-tree. The reduction in time and messages is significant if the initial topology is dense.
3) If we consider QoS traffic, the proposed filtering mechanism will better utilize the existing QoS, rather than creating a new QoS multicast tree. In a congested network, it may not even be possible to create a new QoS multicast tree.

In this paper, we have not considered implementing a security mechanism for communication between nodes in our proposed multicast shared tree filter architecture. We will present a filter architecture with secured communication in a future paper.

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 using IPv6," Dept. of Comp Sci. and Comp Eng., La Trobe University, TechnicalReportNo. 1/05, March 2005.

[8] ——, "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.

[9] K. Carlberg and J. Crowcroft, "Building shared trees using a one-to-many joining mechanism," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 1, pp. 5–11, January 1997.

[10] S. Chen, K. Nahrstedt, and Y. Shavitt, "A QoS-aware multicast routing protocol," in *Proceedings of INFOCOM 2000, 19th Annual Joint Conference of the IEEE Computer and Communications Societies,*, vol. 3, 26-30 March 2000, pp. 1594–1603.

[11] M. Faloutsos, A. Banerjea, and R. Pankaj, "Qosmic: quality of service sensitive multicast internet protocol," in *Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, 31 August - 4 September 1998, pp. 144–153.

[12] J. Singh, P. Veeraraghavan, and S. Singh, "Qos multicast routing using explore best path," *Computer Communications*, (submitted).

[13] U. Black, *MPLS and Label Switching Networks*, 2nd ed. Prentice Hall, 2002.

[14] 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.

[15] 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.

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

[17] J. Singh, P. Veeraraghavan, and S. Singh, "Implementing label filters on a shared tree mobile multicast architecture," in *Proceedings of the IEEE International Conference on Networks (ICON)*, Kuala Lumpur, Malaysia, 16-18 November 2005.

[18] J. A. Bondy and U. Murty, *Graph Theory with Applications*. New York, USA: Elsevier Science Publishing Co., Inc., 1976.