Scalable IP Multicast Sender Access Control for Bi-directional Trees

Ning Wang & George Pavlou

Center for Communication Systems Research, University of Surrey, United Kingdom
{N.Wang, G.Pavlou}@eim.surrey.ac.uk

Abstract. Bi-directional shared tree is an efficient routing scheme for interactive multicast applications with multiple sources. Given the open-group IP multicast service model, it is important to perform sender access control so as to prevent group members from receiving irrelevant data, and also protect the multicast tree from various Denial-of-Service (DoS) attacks. Compared with source specific trees and uni-directional shared trees where information sources can be authorized or authenticated at the single root or Rendezvous Point (RP), in bi-directional trees this problem becomes challengeable since hosts can send data to the shared tree from any network point. In this paper we propose a scalable sender access policy mechanism for bi-directional shared trees so that irrelevant data is policed and discarded once it hits any on-tree router. We consider the scenario of both intra-domain and inter-domain routing in the deployment of the policy, so that the mechanism can adapt to situations in which large-scale multicast applications or many concurrent multicast sessions are involved, potentially across administrative domains.

1 Introduction

IP multicast [9] supports efficient communication services for applications in which an information source sends data to a group of receivers simultaneously. Although some IP multicast applications have been available on the experimental Multicast Backbone (*MBone*) for several years, large-scale deployment has not been achieved until now. IP multicast is also known as "Any Source Multicast (*ASM*)" in that an information source can send data to any group without any control mechanism. In the current service model, group management is not stringent enough to control both senders and receivers. *IGMPv2* [11] is used to manage group members when they join or leave the session but in this protocol there are no control mechanisms to avoid receiving data from particular information sources or prevent particular receivers from receiving sensitive information. It has been observed that the above characteristics of IP multicast have somehow prevented successful deployment of related applications at large scale on the Internet [10].

Realizing that many multicast applications are based on one-to-many communications, e.g. Internet TV/radio, pushed media, etc., H. W. Holbrook et al proposed the *EXPRESS* routing scheme [14], from which the Source Specific Multicast (*SSM*) [15] service model was subsequently evolved. In *SSM* each group is identified by an address tuple (*S*, *G*) where *S* is the unique address of the information

source and *G* is the destination channel address. A single multicast tree is built rooted at the well-known source for delivering data to all subscribers. Under such a scenario, centralized group authorization and authentication can be achieved at the root of the single source at the application level. Currently *IGMPv3* [7] is under the development to support source specific joins in *SSM*.

On the other hand, it should be noted that there exist many other applications based on many-to-many styled communication, such as multi-party videoconferencing system, Distributed Interactive Simulation (*DIS*) and Internet games etc. For this type of interactive applications, bi-directional multicast trees such as Core Based Tree (*CBT*) [2], Bi-directional *PIM* [13], and *RAMA* style Simple Multicast [19], are efficient routing schemes for natively delivering data between multiple hosts. However, since there is no single point for centralized group access control, sender authorization and authentication become new challenges. Typically, if a malicious host wants to perform Denial-of-Service (*DoS*) attack it can flood bogus data from any point of the bi-directional multicast tree. Sender access control for bi-directional trees based on IP multicast model is not provided in the specification of any corresponding routing protocols such as [2, 13]. One possible solution that has been proposed is to periodically "push" the entire sender access list down to all the on-tree routers, so that only data from authorized senders can be accepted and sent onto the bi-directional tree [6]. This simple access control mechanism has been adopted in the *RAMA*-style Simple Multicast [14]. However, this policy is not very scalable especially when many multicast groups or large group size with many senders are involved. A more sophisticated scheme named *Keyed-HIP* (*KHIP*) [21] works on the routing level to provide data access control on the bi-directional tree, and flooding attacks can be also detected and avoided by this network-level security routing scheme.

In this paper we will propose an efficient and scalable sender access control mechanism for bi-directional trees in the IP multicast service model. The basic idea is to deploy access policy for external senders on the tree routers where necessary, so that data packets from unauthorized senders will be policed and discarded once it hits the bi-directional tree. Our proposed scheme causes little impact on the current bi-directional routing protocols so that it can be directly implemented on the Internet without modifying the basic function of the current routing protocols. Moreover, the overhead introduced by the new control mechanism is much smaller than that proposed in [6] and [19].

The rest of the paper is organized as follows: Section 2 gives the overview of our proposed dynamic maintenance of the policy. Sections 3 and 4 introduce sender authorization and authentication in intra-domain and inter-domain routing. Operations on multi-access networks are specially discussed in section 5. We examine the scalability issues of our proposed scheme in section 6, and finally we present a summary in section 7.

2 Sender Authorization and Authentication Overview

Compared with source specific trees and even uni-directional shared trees such as *PIM-SM* [8], in which external source filtering can be performed at the single source or Rendezvous Point (*RP*) where the registrations of all the senders are processed and authorized, in bi-directional trees this is much more difficult since data from any source will be directly forwarded to the whole tree once it hits the first on-tree router. In fact, since there is no single point for centralized sender access control, information source authorization and authentication has to be deployed at the routing level. As we have already mentioned, the simplest solution for this is to periodically broadcast the entire access control list down to all the routers on the bi-directional tree for deciding whether or not to accept data (e.g., [19]). However, this method is only feasible when a few small-sized groups with limited number of senders are considered. For large scale multicast applications, if we don't send the whole policy down to all the on-tree routers so as to retain the scalability, three questions need to be answered as proposed in [6]: (1) How to efficiently distribute the list where necessary? (2) How to find edge routers that act as the trust boundary? (3) How to avoid constant lookups for new sources? In fact if we try to statically mount the access control policy to an *existing* bi-directional multicast tree, none of the above three questions can be easily answered.

It should be noted that most multicast applications are highly dynamic by nature, with frequent join/leaving of group members and even information senders. Hence the corresponding control policy should also be dynamically managed. Here we propose an efficient sender-initiated distribution mechanism of the access list during the phase of multicast tree construction. The key idea is that each on-tree router only adds its *downstream* senders to the local Sender Access Control List (*SACL*) during their join procedure, and the senders in the access list are activated by the notification from the core. In fact, only the core has the right to decide whether or not to accept the sources and it also maintains the entire *SACL* for all the authorized senders. Packets coming from any unauthorized host (even if it has already been in the tree) will be discarded at once when they reach any on-tree router. To achieve this, all senders must first register with the core before they can send any data to the group. When a registration packet hits an on-tree router, the unicast address of the sender is added into the *SACL* of each router on the way. Under this scenario, the access policy for a particular sender is deployed on the branch from the first on-tree router where the registration is received along to the core router. Here we define the interface from which this registration packet is received as the *downstream interface* and the one used to deliver unicast data to the core as the *upstream interface*. The format of each *SACL* entry is (*G, S, I*) where *G* indicates the group address, *S* identifies the sender and *I* is the downstream interface from which the corresponding registration packet was received. If the core has approved the join, it will send a type of "activating packet" back to the source, and once each on-tree router receives this packet, it will activate the source in its *SACL* so that it will be able to send data onto the bi-directional tree from then on. Under such a scenario, an activated source can only send group data to the tree via the

path where its *SACL* entry has been recorded, i.e., even if a sender has been authorized, it cannot send data to the group from other branches or elsewhere. Source authentication entries kept in each *SACL* are maintained in soft state for flexibility purpose, and this requires that information sources should periodically send refreshing packets to the core to keep their states alive in the upstream routers. This action is especially necessary when a source is temporarily not sending group data. Once data packets have been received from a particular registered sender, the on-tree router may assume that this source is still alive and will automatically refresh the state for it. If a particular link between the data source and the core fails, the corresponding state will time out and become obsolete. In this case the host has to seek alternative path to perform re-registration for continuing sending group data.

When a router receives a data packet from one of its downstream interfaces, it will first check if there exists such an entry for the data source in its local *SACL*. If the router cannot find a matching entry that contains the unicast address of the source, the data packet is discarded. Otherwise if the corresponding entry has been found, the router will verify if this packet comes from the same interface as the one recorded in the *SACL* entry. Only if the data packet has passed these two mechanisms of authentication, it will be forwarded to the upstream interface and the other interfaces with the group state, i.e., interfaces where receivers are attached. On the other hand, when a data packet comes from the upstream interface, the router will always forward it to all the other interfaces with group state and need not perform any authentication. Although the router cannot judge if this data packet is from a registered sender, since it comes from the upstream router, there exist only two possibilities: either the upstream router has the *SACL* entry for the data source or the upstream router has received the packet from its own parent router in the tree. The extreme case is that none of the intermediate ancestral routers have such an entry and then we have to backtrack to the core. Since the core has recorded entries for all the registered senders and it never forwards any unauthenticated packet on its downstream interfaces, we can safely conclude that each on-tree router can trust its parent, and hence packets received from the upstream interface are always from valid senders. However, this scenario precludes the case of routers attached on multi-access networks such as *LANs*, and we will discuss the corresponding operations in section 5.

Compared with source specific trees and even uni-directional shared trees such as *PIM-SM* [8], in which external source filtering can be performed at the single source or Rendezvous Point (*RP*) where the registrations of all the senders are processed and authorized, in bi-directional trees this is much more difficult since data from any source will be directly forwarded to the whole tree once it hits the first on-tree router. In fact, since there is no single point for centralized sender access control, information source authorization and authentication has to be deployed at the routing level. As we have already mentioned, the simplest solution for this is to periodically broadcast the entire access control list down to all the routers on the bi-directional tree for deciding whether or not to accept data (e.g., [19]). However, this method is only feasible when a few small-sized groups with limited number of senders are considered. For large scale multicast applications, if we don't send the whole policy down to all the on-tree routers so as to retain the scalability, three questions need to be answered as proposed in [6]: (1) How to efficiently distribute the list where necessary? (2) How to find edge routers that act as the trust boundary? (3) How to avoid constant lookups for new

sources? In fact if we try to statically mount the access control policy to an *existing* bi-directional multicast tree, none of the above three questions can be easily answered.

It should be noted that most multicast applications are highly dynamic by nature, with frequent join/leaving of group members and even information senders. Hence the corresponding control policy should also be dynamically managed. Here we propose an efficient sender-initiated distribution mechanism of the access list during the phase of multicast tree construction. The key idea is that each on-tree router only adds its *downstream* senders to the local Sender Access Control List (*SACL*) during their join procedure, and the senders in the access list are activated by the notification from the core. In fact, only the core has the right to decide whether or not to accept the sources and it also maintains the entire *SACL* for all the authorized senders. Packets coming from any unauthorized host (even if it has already been in the tree) will be discarded at once when they reach any on-tree router. To achieve this, all senders must first register with the core before they can send any data to the group. When a registration packet hits an on-tree router, the unicast address of the sender is added into the *SACL* of each router on the way. Under this scenario, the access policy for a particular sender is deployed on the branch from the first on-tree router where the registration is received along to the core router. Here we define the interface from which this registration packet is received as the *downstream interface* and the one used to deliver unicast data to the core as the *upstream interface*. The format of each *SACL* entry is (*G, S, I*) where *G* indicates the group address, *S* identifies the sender and *I* is the downstream interface from which the corresponding registration packet was received. If the core has approved the join, it will send a type of "activating packet" back to the source, and once each on-tree router receives this packet, it will activate the source in its *SACL* so that it will be able to send data onto the bi-directional tree from then on. Under such a scenario, an activated source can only send group data to the tree via the path where its *SACL* entry has been recorded, i.e., even if a sender has been authorized, it cannot send data to the group from other branches or elsewhere. Source authentication entries kept in each *SACL* are maintained in soft state for flexibility purpose, and this requires that information sources should periodically send refreshing packets to the core to keep their states alive in the upstream routers. This action is especially necessary when a source is temporarily not sending group data. Once data packets have been received from a particular registered sender, the on-tree router may assume that this source is still alive and will automatically refresh the state for it. If a particular link between the data source and the core fails, the corresponding state will time out and become obsolete. In this case the host has to seek alternative path to perform re-registration for continuing sending group data.

When a router receives a data packet from one of its downstream interfaces, it will first check if there exists such an entry for the data source in its local *SACL*. If the router cannot find a matching entry that contains the unicast address of the source, the data packet is discarded. Otherwise if the corresponding entry has been found, the router will verify if this packet comes from the same interface as the one recorded in the *SACL* entry. Only if the data packet has passed these two mechanisms of authentication, it will be forwarded to the upstream interface and the other interfaces with the group state, i.e., interfaces where receivers are attached. On the other hand, when a data packet comes from the upstream interface, the router will always forward

it to all the other interfaces with group state and need not perform any authentication. Although the router cannot judge if this data packet is from a registered sender, since it comes from the upstream router, there exist only two possibilities: either the upstream router has the *SACL* entry for the data source or the upstream router has received the packet from its own parent router in the tree. The extreme case is that none of the intermediate ancestral routers have such an entry and then we have to backtrack to the core. Since the core has recorded entries for all the registered senders and it never forwards any unauthenticated packet on its downstream interfaces, we can safely conclude that each on-tree router can trust its parent, and hence packets received from the upstream interface are always from valid senders. However, this scenario precludes the case of routers attached on multi-access networks such as *LANs*, and we will discuss the corresponding operations in section 5.

3 Intra-domain Access Control Policy

3.1 SACL Construction and Activation

As we have mentioned, all the information sources must register with the core before they can send any data to the bi-directional tree. For each on-tree router, its *SACL* is updated when the registration packet of a new sender is received, and the individual entry is activated when its corresponding activating notification is received from the core.

If a host wants to both send and receive group data, it must join the multicast group and become a Send-Receive capable member (SR-member, *SRM*). Otherwise if the host only wants to send messages to the group without receiving any data, it may choose to act as a Send-Only member (SO-member, *SOM*) or a Non-Member Sender (*NMS*). In the former case, the host must join the bi-directional tree to directly send the data, and its designated router will forward the packets on the upstream interface as well as other interfaces with the group state. In the IP multicast model information sources are allowed to send data to the group without becoming a member. Hence, if the host is not interested in the information from the group, it may also choose to act as a non-member sender. In this case, the host must encapsulate the data and unicast it towards the core. Once the data packet hits the first on-tree router and passes the corresponding source authentication, it is decapsulated and forwarded on all the other interfaces with the group state. The following description is based on the *CBT* routing protocol, but it can also apply to other bi-directional routing schemes such as Bidir-*PIM* and *RAMA*-style Simple Multicast.

(1) SR-member Join

When the Designated Router (*DR*) receives a group *G* membership report from a SR-member *S* on the *LAN*, it will send a join request towards the core. Here we note that the group membership report cannot be suppressed by the *DR* if it is submitted by a send-capable member. Once a router receives this join-request packet from one of its interfaces, say, *A*, then the (*G, S, A*) entry is added into its *SACL*. If the router is not been on the shared tree, a (**, G*) state is created with the interface leading to the core as the upstream interface and *A* is set to the downstream interface. At the same time,

interface *A* is also added to the interface list with group state so that data from other sources can be forwarded to *S* via *A*. If the router already has the (*, *G*) state, but *A* is not in the interface list with group state, then it is added to the list. Thereafter, the router just forwards the join-request to the core via its upstream interface. Once the router receives the activating notification from the core, the (*G, S, A*) entry is activated so that *S* is able to send data.

(2) SO-member join

Similar to SR-member joins, the *DR* of a SO-member also sends a join-request up to the core and when the router receives this request from its interface *A*, the (*G, S, A*) entry is added to the local *SACL*. If the router is not yet on the tree, (*, *G*) state will be generated but the interface *A* is not added to the interface list with group state. This is because *A* needs not to forward group data to a send-only member.

(3) Non-Member Sender (*NMS*) registration

Here we use the terminology "registration" instead of "join request", since this host is not a group member and need not be on the tree to send group data. The registration packet from the Non-Member Sender is unicast towards the core and when it hits the first router with (*, *G*) state, the (*G, S, A*) entry will be created in the local *SACL* of all the on tree routers on the way leading to the core. It should be noted, if a router is not on the tree, it does not maintain *SACL* for the group.

Finally, if a receive-only member (also known as the *group member* in conventional multicast model) wants to join the group, the join request only invokes a (*, *G*) state if the router is not on the tree, but no new *SACL* entries need to be created. Moreover, once the join request hits any on-tree router, a join-notification is immediately sent back without informing the core.

The forwarding behavior of an on-tree router under send access control mechanism is as follows. If group data comes from downstream interfaces, the router will authenticate the information source by looking up the local *SACL* and if the sender has its entry in the list and comes from the right interface, the data is forwarded on the upstream interface and other interfaces with group state. If the corresponding *SACL* check fails, the data is discarded at once. On the other hand, if the data comes from the upstream interface, it is forwarded to all the other interfaces with the group state because a router's parent is always trusted by its children.

3.2 An Example for Intra-domain Access Policy

A simple network model is given in Fig. 1 . We assume that node *A* is the core router and all the Designated Routers (*DR*) of potential members of group *G* should send join request to this node. Hosts *H1-H5* are attached to the individual routers as shown in the figure.

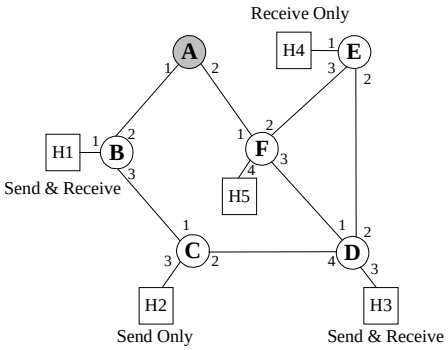

Fig. 1. Intra-domain network model

Initially suppose *H1* wants to join the group, its *DR* (router *B*) will create (*, *G*) state and send the join request to the core *A*. Since *H1* is a SR-member that can both send and receive data to/from the group, each of the routers that the join request has passed will add this sender into its local *SACL*. Hence both router *B* and *A* will have the *SACL* entry (*G, H1, 1*), since they both receive the join request from interface 1. Host *H2* only wants to send messages to group *G* but does not want to receive any data from this group, and so it may choose to join as a SO-member or just act as a *NMS*. In the first case, its *DR* (router *C*) will create (*, *G*) state indicating that this router is an on-tree node and then add *H2* to its *SACL*. Thereafter, router *C* will send a join request indicating *H2* is a SO-member towards the core; when *B* receives this request, it will also add *H2* to its local *SACL* and then forward the join-request packet to *A*. Since *H2* does not want to receive data from the group, link *BC* becomes a send-only branch. To achieve this, router *B* will not add *B3* to the interface list with group state. If *H2* chooses to act as the Non-Member Sender, router *C* will not create (*, *G*) state or *SACL* for the group but send a registration packet towards *A*. When this packet hits an on-tree router, say, *B* in our example, *H2* will be added to the local *SACL* of all the routers on the way. When sending group messages, router *C* just encapsulates the data destined to the core by setting the corresponding IP destination address to *A*. When the data reaches *B* and passes the *SACL* authentication, the IP destination address is changed to the group address originally contained in the option field of the data packet, and the message is forwarded to interfaces *B1* and *B2* to get to *H1* and the core respectively. After *H3* and *H4* join the group, the resulting shared tree is shown in Fig. 2, and *SACLs* of each on-tree router are also indicated in the figure. It should be noted that *H4* is a receive-only member, and hence Router *E*, *F* and *A* need not add it to their local *SACLs*. Suppose router *F* has received group data from *H3* on interface *F3*, it will check in its local *SACL* if *H3* is an authorized sender. When the data passes the address and interface authentications, it is forwarded to both interfaces *F1* and *F2*. When group data is received on the upstream interface *F1*, since its parent *A* is a trusted router (in fact the data source should be either *H1* or *H2*), the data is forwarded to *F2* and *F3* immediately without any authentication. However, if the non-registered host *H5* wants to send messages to the group, data won't be forwarded to the bi-directional tree due to the *SACL* authentication failure at router *F*.

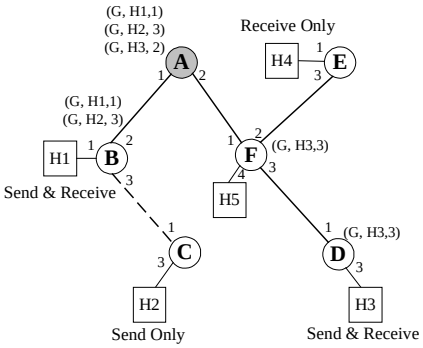

Fig. 2. Bi-directional tree with *SACL*

4 Inter-domain Access Control Policy

4.1 Basic Descriptions

As we have mentioned above, on-tree routers only maintain the access policy for all the downstream senders. However, if large-scale groups with many senders or many concurrent sessions are considered, the size of the *SACL* in the routers near the core will become a heavy burden for these on-tree routers. In this section we discuss how this situation can be improved with the aid of inter-domain IP multicast routing semantics.

Our key idea is based on hierarchical access control policy to achieve scalability. All routers only maintain *SACL* for the downstream senders in the *local* domain and need not add sources from downstream domains to their local *SACLs*. In other words, all the senders for the group are only authenticated in the local domain. In the root domain, the core needs to keep entries only for local senders; however in order to retain the function of authorizing and activating information sources from remote domains, on receiving their registrations the core router needs to contact a special access control server residing in the local domain, which decides whether or not to accept the sending requests.

For each domain, a unique border router (*BR*) is elected as the "policy agent" and keeps the entire *SACL* for all the senders in the local domain, and we name this *BR Designated Border Router* (*DBR*) for the domain. In fact the *DBR* can be regarded as core of the sub-tree in the local domain. In this sense, all the data from an upstream domain can only be injected into the local domain from the unique *DBR* and all the senders in this domain can only use this *DBR* to send data up towards the core. This mechanism abides to the "3rd party independence" policy in that data from any sender must be internally delivered to all the local receivers without flowing out of the domain. This requires that joins from different hosts (including both senders and receivers) merge at a common point inside the domain. In *BGP*-4, all the *BRs* of a stub domain know for which unicast prefix(es) each of them is acting as the egress router, this satisfies the above requirement of "path convergence" of internal joins.

Since individual sender authentication is performed within each domain and invalid data never gets any chance to flow out of the local domain, the on-tree *BR* of the upstream domain will always trust its downstream *DBR* and assumes that all the data packets coming from it are originated from authorized senders. Hence, when a packet leaves its local domain and enters remote domains, no further authentication is needed. This also avoids constant lookups when the authenticated data is traveling on the bi-directional tree.

4.2 Inter-domain *SACL* Construction and Activation

Since Border Gateway Multicast Routing (*BGMP [16]*) has been considered as the long-term solution to the Inter-domain multicast routing, in this section we will take *BGMP* as an example to illustrate how sender access control policy can be deployed in inter-domain applications.

First we will discuss how the *DR* for a group member sender submits its join request and how it is added to the *SACL* and activated. This applies to both SR-members and SO-members, the only difference between the two being whether or not to add the interface from which the join-request was received to the interface list with the group state. Only if an on-tree router receives a join request from a sender in the local domain, it will add this sender to its *SACL*, otherwise the router will just forward the join request towards the core without updating its local *SACL*.

In Fig. 3, when host *S* wants to become a SO-member to send data, its *DR* (router *A*) sends a join request towards the *DBR* router *B*, which has the best exit to the root domain. All the internal routers receiving this request will add *S* into their local *SACLs*. Since *B* is the core of the sub-tree for the local domain, it also needs to create a *SACL* entry for host *S* once it receives the join request from its Multicast Interior Gateway Protocol (*M-IGP*) component. Thereafter, *B* finds in its Group Routing Information Base (*G-RIB*) that the best route to the root domain is via its external peer *C* in the transit domain, so router *B* will send the *BGMP* join request towards *C* via its *BGMP* component. Once router *C* receives the join request, it creates (*, *G*) state (if it has not been on the tree), but will not create an entry for *S* in its local *SACL*. When *C* finds out that the best exit toward the root domain is *D*, it just forwards the join request to this internal *BGMP* peer, and hence router *D* becomes the *DBR* of the transit domain for group *G*. Suppose *Bidir-PIM* is the *MIGP*, the *RP* in this transit domain should be placed at *D*, and router C will use its *M-IGP* component to send the join request towards *D*. When this join request travels through the transit domain, none of the internal routers along the way in the domain will add *S* into their local *SACLs*. After the join request reaches the root domain and the core router *F* authorizes the new sender by contacting the access control server and sends back the activating-notification, all the on-tree routers (including internal on-tree routers and the *DBR*) in the transit domain just forward it back towards the local domain where the new sender *S* is located. When the packet enters the local domain, all the on-tree routers (namely *B* and *A* in Fig. 3) will activate *S* in their *SACLs*.

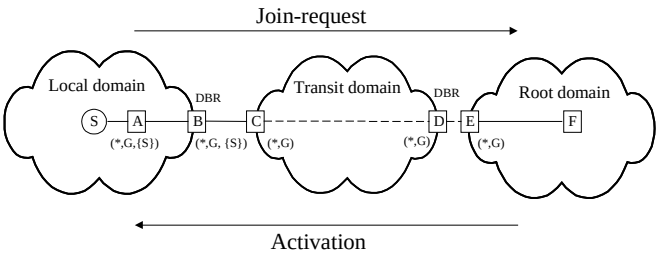

Fig. 3. Fig. 3 Inter-domain join-request

As we have also mentioned, a send-only host may also choose to act as a Non-Member Sender (*NMS*). However there are some restrictions when inter-domain multicast routing is involved. If a send-only host is located in the domain where there are no receivers (we call this domain a *send-only domain*), then the host should join the bi-directional tree as a SO-member other than a Non-Member Sender (*NMS*). Otherwise if the host acts as a *NMS*, its registration packet will not hit any on-tree router until it enters remote domains. This forces the on-tree router there to add the sender that is from another domain to its local *SACL*, which does not conform to the rule that on-tree routers only maintain access policy for senders in the local domain. On the other hand, if the host joins as a SO-member and since its *DR* will be on the bi-directional tree, the authentication can be achieved by the on-tree routers in the local domain.

4.3 An Example for Inter-domain Access Policy

An example for inter-domain sender access control is given in Fig. 4. *C* is the core router and domains *X*, *Y* and *Z* are remote domains regarding the core *C*. Hosts *a*, *b*, *c* and *d* are attached to the routers in different domains. Also suppose that host *a* only wants to receive data from the group, hosts *b* and *c* want to both send and receive, while host *d* only wants to send messages to the group without receiving any data from it. In this case, *X* is a receive-only domain and *Z* is a send-only domain. *X1*, *Y1* and *Z1* are border routers that have been selected as the *DBR* for each domain. According to our inter-domain access control scheme, on-tree routers have the *SACL* entry for downstream senders in the local domain, and each *DBR* has the policy for all the senders in the local domain. Hence, *Y1* has the entry for hosts *b* and *c* in its *SACL* while the *SACL* of *X1* contains no entries at all. Although *X* is the parent domain of *Y* and *Z* which both contain active senders, all the on-tree routers in *X* need not add these remote senders to their *SACL*. In fact data coming from *Y* and *Z* has already been authenticated by their own *DBRs* (namely *Y1* and *Z1*) before it flows out of the local domains. Since host *d* only wants to send data to the group and there are no other receivers in domain *Z*, as we have mentioned, host *d* should join as a send-only member. Otherwise if *d* acts as a non-member sender and submits its registration packet towards the core, this makes the first on-tree router (*X2*) add *d* to its *SACL*, however this is not scalable because on-tree routers are forced to add senders from remote domains. On the other hand, if host *d* joins as a send-only member, the shared

tree will span to its *DR*, namely *Z2*, and then the authentication can be performed at the routers in the local domain.

As we know, *BGMP* also provides the mechanism for building source-specific branches between border routers. In Fig. 4, we suppose that the current *M-IGP* is *PIM-SM*. At certain time the *DR* in domain *Y* such as *Y3* or *Y4* may wish to receive data from host *d* in domain *Z* via the shortest path tree. Hence (*S, G*) state is originated and passed to the border router *Y5*, which is not the current *DBR* of domain *Y*. When *Y5* receives the source specific join, it will create (*S, G*) state and then send the corresponding *BGMP* source specific join towards *Z1*. On the other hand, since *Z1* is the *DBR* of domain *Z*, intra-domain sender authentication has been performed before the traffic is sent to *Z1's BGMP* component for delivery to remote domains. In fact *Y5* will only receive and accept data originated from host *d* in domain *Z* due to its (*S, G*) state filtering. Once *Y5* receives the data from host *d*, it can directly forward it to all the receivers in the local domain, since *RPF* check can be passed. When the *DR* receives the data from *d* via the shortest path, it will send a source specific prune message up towards the root domain to avoid data duplication. It should be noted that (**, G*) state should only exist in the *DBR* for each domain/group, and internal nodes may only receive source specific traffic via alternative border routers. From this example, it is observed that source specific tree can also interoperate with the proposed sender access control in the receiver's domain (note that the *MIGP* in domain *Y* is not bi-directional routing protocol).

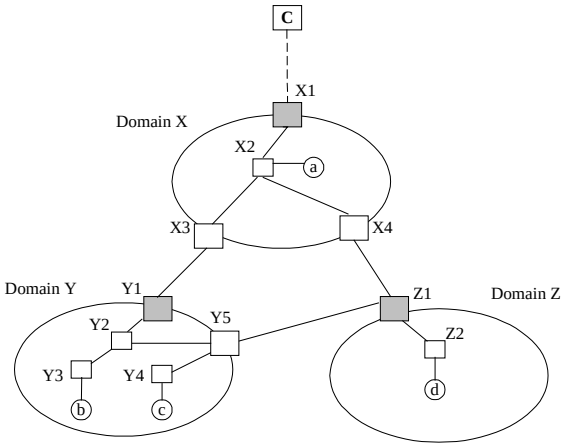

Fig. 4. Example for Inter-domain sender access control

5 Operations on Multi-access Networks

We need special consideration for protecting group members from unauthorized sources attached to multi-access networks such as *LANs*. As we have mentioned, if an on-tree router receives data packets from its upstream interface, it will always forward them to all the other interfaces with group state, since these packets have been

assumed to come from an authorized information source. However this may not be the case if the upstream interface of an on-tree router is attached to a broadcast network. When an unauthorized host wants to send data with group address to the multi-access *LAN*, a corresponding mechanism must be provided to prevent these packets from being delivered to all the downstream group members. To achieve this, once the Designated Router (*DR*) on the *LAN* receives such a packet from its downstream interface, if it cannot find a matching access entry for the data source in its *SACL*, it will discard the packet, and at the same time this *DR* will send a type of "forbidding" control packet containing the unicast address of the unauthorized host to the *LAN* from its downstream interface. Take the *CBT* routing protocol as an example, the IP destination address of this forbidding packet should be "all-cbt-router address (224.0.0.15)" and the value of *TTL* is set to 1. Once the downstream router receives this packet on its upstream interface, it will stop forwarding the data with this unicast address that originates from an unregistered host attached to the *LAN*. Hence all the downstream session members will only receive little amount of useless data for a short period of time. In terms of implementation, the downstream on-tree routers should maintain a "forbidding list" of unauthorized hosts recorded. Since all the possible unauthorized hosts can only come from the local *LAN*, this list will not introduce much overhead to the routers. In Fig. 5, suppose the unauthorized host *S* sends data to the group. When the *DR* (router *A*) cannot find the corresponding entry in its local *SACL*, it immediately discards the packet and then sends a "forbidding" packet containing the address of *S* onto the *LAN*. Once the downstream router *B* receives the forbidding packet, it will stop forwarding data coming from host *S*.

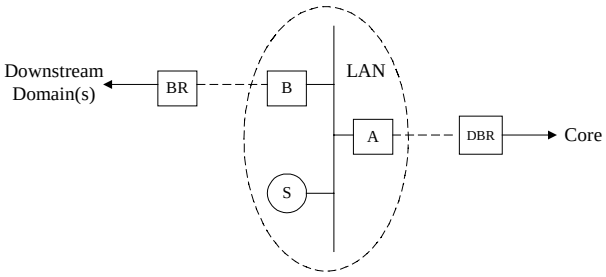

Fig. 5. Access control operation on *LANs*

In inter-domain routing, further consideration is necessary for data traffic traveling towards the core. This is because routers in transit domains do not have *SACL* entry for remote senders in their *SACLs*. Also take Fig. 5 as an example, suppose that the *LAN* is located in a transit domain where there are no local authorized senders, and hence router *A*'s *SACL* is empty. If there is data appearing on the *LAN* destined to the group address, there are only two possibilities: (1) the data came from a downstream domain and was forwarded to the *LAN* by router *B*; (2) a local unregistered host attached to the *LAN* (e.g., host *S*) sent the data. It is obvious that in the former case router *A* should pick up the packet and forward it towards the core, and for the latter, it should just discard the packet and send the corresponding "forbidding" packet onto the *LAN*. Hence this requires that the router be able to distinguish between packets

coming from remote domains and packets coming from directly attached hosts on the *LAN*. However, this is easy to achieve by simply checking the source address prefix.

6 *SACL* Scalability Analysis

In this section we discuss scalability issues regarding router memory consumption. For simplicity we only discuss the situation of intra-domain routing here. Nevertheless, when inter-domain hierarchical sender access control is involved, the situation can be improved still further. It is obvious that the *maximum* memory space needed in maintaining a *SACL* is *O(ks)* where *k* is the number of multicast groups and *s* is the number of senders in the group. Typically this is exactly the size of *SACL* in the core router. However, since on-tree routers need not keep the access policy for all sources but only for downstream senders, the average size of *SACL* in each on-tree router is significantly smaller.

We can regard the bi-directional shared tree as a hierarchical structure with the core at the top level, i.e., level 0. Since each of the on-tree routers adds its downstream senders to its local *SACL*, then the *SACL* size *S* of router *i* in the shared tree *T* can be expressed as follows:

$$S(i) = \sum_{(i,j)\in T} (S(j)) \tag{1}$$

and the average *SACL* size per on-tree router is:

$$\overline{S} = \frac{\sum_{i=0}^{H} \sum_{j=1}^{L_i} S(j)}{\sum_{i=1}^{n} Y_i} \tag{2}$$

where H is the number of hops from the farthest on-tree router (or maximum level) and L_i is the number of routers on level i, while

$$Y_i = \begin{cases} 1 & \text{if router } i \text{ is included in the shared tree} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

To ensure that the scalability issues are fairly evaluated throughout our simulation, random graphs with low average degrees, which represent the topologies of common point-to-point networks, e.g., *NSFNET*, are constructed. Here we adopt the commonly used Waxman's random graph generation algorithm [22] that has been implemented in *GT-ITM*, for constructing our network models. For simplicity, we only consider intra-domain routing scenarios in our simulation.

First we study the relationship between average *SACL* size and total number of senders. In the simulation we generate a random network with 100 routers with the core router also being randomly selected. The number of senders varies from 10 to 50 in steps of 10 while the group size is fixed at 50. Here we study three typical situations regarding the type of sending hosts:

(1) All senders are also receivers (*AM*);
(2) 50% senders are also receivers (*HM*);
(3) None of the senders are receivers (*NM*).

All send-only hosts choose to act as Non-Member Senders (*NMS*).

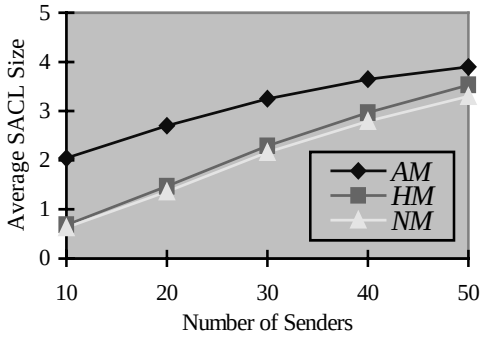

Fig. 6. *SACL* size *vs*. number of senders (I)

From Fig. 6 we can see that the average *SACL* size grows as the number of senders increases. However, it can be observed that even when the number of senders reaches a size as large as 50, the average *SACL* size is still very small (less than 4 in size on average). This is in significant contrast with the strategy of "full policy maintenance" (*FPM*) on each router [6, 19]. Further comparison between the two methods is presented in Table 1. From the figure we can also find that if all the senders are also receivers on the bi-directional tree (case *AM*), this results in a larger average *SACL* size. On the other side, if none of the senders is a receiver (case *NM*), the corresponding *SACL* size is smaller. This phenomenon is expected because given the fixed number of receivers on the bi-directional tree as well as the sender group, the larger the proportion of senders coming from receiver set, the larger the resulting average *SACL* size. However this gap decreases with larger sender group size.

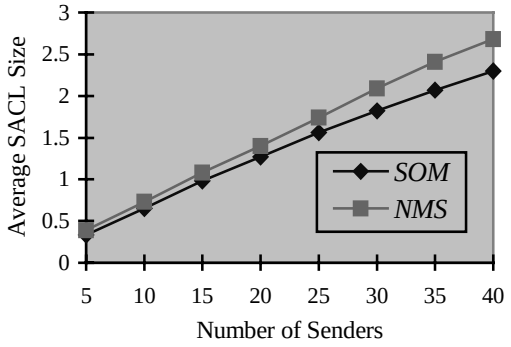

Fig. 7. *SACL* size *vs*. number of senders (II)

Next we study the effect on *SACL* size resulting from the senders' choice of acting as a Send-Only Member (*SOM*) or a Non-Member Sender (*NMS*). As we have mentioned, a host only wishing to send data to the group can decide to act as a *SOM* or *NMS*. Fig. 7 illustrates the relationship between the *SACL* size and total number of senders. The group size is fixed at 50 and the number of senders varies from 5 to 40 in steps of 5. It should be noted that in this simulation all group members are receive-only hosts and do not send any data to the group. From the figure we can see that the *SACL* size also grows with the increase of the number of senders. Moreover, if all the hosts join the bi-directional tree and act as Send-Only Members (*SOM*), the average *SACL* size is smaller. The reason for this is obvious: If the hosts choose to take the role of *SOM*, this will make the bi-directional tree expand for including the *DRs* of these senders. Since the number of on-tree routers grows while the total number of senders remains the same, the resulting average *SACL* size will become smaller. On the other hand, if all of the hosts just act as Non-Member-Senders, the figure of the shared tree will not change and no more on-tree routers are involved.

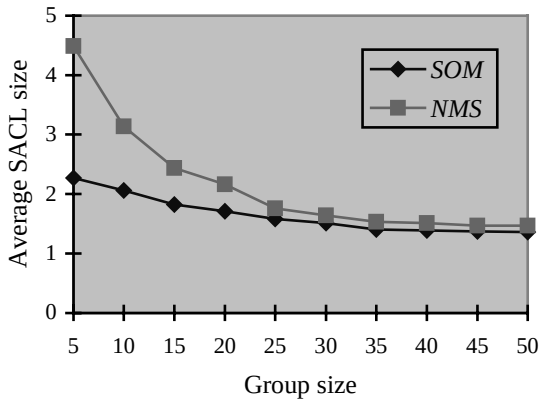

Fig. 8. Average *SACL* Size *vs*. Group Size

We continue to study the relationship between the average *SACL* size and the group size (number of receivers) with number of senders fixed at 20. We still let these senders choose to act as a *SOM* or *NMS* respectively. From Fig. 8 we can see that the *SACL* size decreases with the growth of the group size in both cases. On the other hand, *SOM* join results in smaller average *SACL* size compared with *NMS*. The gap is more significant when there are fewer receivers. This is because if senders choose to act as *SOM*, they have to join the tree and generate many send-only branches, i.e., more routers are involved in the bi-directional tree. If the hosts just send data without becoming group members, the shared tree won't span to any of these senders, so that the number of on-tree routers is independent of the number of senders. When the group size is small (e.g., 5 receivers), the size of the bi-directional tree will be increased significantly to include all the senders if they join as *SOMs*. This explains why the gap is more obvious when a small set of receivers is involved.

S	10	20	30	40
FPM	10	20	30	40
SOM	0.65	1.27	1.82	2.3
NMS	0.73	1.4	2.09	2.73

Table 1. Comparison with *FPM*

Finally we give the comparison between our method and the "full policy maintenance" (*FPM*) strategy regarding router's memory consumption. Table 1 gives the relationship of *SACL* size and total number of senders (*S*). From the table we can see that the length of the access list recorded in each on-tree router in *FPM* mechanism is exactly the number of active senders. This imposes very big overhead on routers compared with our proposed scheme. Although the core router also has to maintain the full access list in our method when intra-domain routing is considered, the situation could be improved in large-scale multicast applications by hierarchical control in inter-domain routing which we introduced in section 4.

7 Summary

In this paper we propose an efficient mechanism of sender access control for bi-directional multicast trees in the IP multicast service model. Each on-tree router dynamically maintains access policy for its downstream senders. Under such type of control, data packets from unauthorized hosts are discarded once they hit any on-tree router. In this sense, group members won't receive any irrelevant data, and network service availability is guaranteed since the multicast tree is protected from denial-of-service attacks such as data flooding from any malicious host. In order to achieve scalability for large-scale multicast applications with many information sources and to accommodate more concurrent multicast sessions, we also extend our control mechanism to inter-domain routing where hierarchical access policy is maintained on the bi-directional tree. Simulation results also show that the memory overhead of our scheme is quite light so that good scalability can be achieved.

Nevertheless, this paper only provides a general paradigm of sender access control, but does not present a solution to the restriction of sources based on the specific interest from individual receivers. Related works include [12], [17] and [18], and this will be one of our future research directions.

References

[1] K. C. Almeroth, "The Evolution of Multicast: From the MBone to Inter-domain Multicast to Internet2 Deployment", IEEE Network special issue on Multicasting, Jan., 2000

[2] T. Ballardie, P. Francis, J. Crowcroft, "Core Based Trees (CBT): An Architecture for Scalable Multicast routing", Proc. SIGCOMM'93, pp85-95

[3] A. Ballardie, "Scalable Multicast Key Distribution", RFC 1949, May 1996

[4] A. Ballardie, J. Crowcroft, "Multicast-Specific Security Threats and Counter-measures", Proc. NDSS'95, pp2-16

[5] S. Bhattacharyya *et al*, "*An Overview of Source-Specific Multicast (SSM) Deployment*", Internet Draft, draft-ietf-ssm-overview-*.txt, May 2001, work in progress

[6] B. Cain, "Source Access Control for Bidirectional trees", 43rd IETF meeting, December, 1998

[7] B. Cain *et al*, "*Internet Group Management Protocol, Version 3*", Internet draft, draft-ietf-idmr-igmp-v3-*.txt, Feb. 1999, work in progress

[8] S. Deering *et al*, "*The PIM Architecture for Wide-Area Multicast Routing*", *IEEE/ACM* Transactions on Networking, Vol. 4, No. 2, Apr. 1996, pp 153-162

[9] S. Deering, "Multicast Routing in Internetworks and Extended LANs", Proc. ACM SIGCOMM, 1988, pp55-64

[10] C. Diot et al, "Deployment Issues for the IP Multicast Service and Architecture", IEEE Network, Jan./Feb. 2000, pp 78-88

[11] W. Fenner, "Internet Group management Protocol, version 2", RFC 2236, Nov. 1997

[12] B. Fenner *et al*, "*Multicast Source Notification of Interest Protocol (MSNIP)*", Internet Draft, draft-ietf-idmr-msnip-*.txt, Feb. 2001

[13] M. Handley *et al*, "*Bi-directional Protocol Independent Multicast (BIDIR-PIM)*", Internet Draft, draft-ietf-pim-bidir-*.txt, Nov. 2000, work in progress

[14] H. W. Holbrook, D. R. Cheriton, "IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications", Proc. ACM SIGCOMM'99

[15] H. W. Holbrook, B. Cain, "*Source-Specific Multicast for IP*", Internet Draft, draft-holbrook-ssm-arch-*.txt, Mar. 2001, work in progress

[16] S. Kummar et al, "The MASC/BGMP Architecture for Inter-domain Multicast Routing", Proc. ACM SIGCOMM'99

[17] B. N. Levine et al, "Consideration of Receiver Interest for IP Multicast Delivery", Proc. IEEE INFOCOM 2000, vol. 2, pp470-479

[18] M. Oliveira et al, "Router Level Filtering for Receiver Interest Delivery", Proc. NGC' 2000

[19] R. Perlman *et al*, "*Simple Multicast: A Design for Simple, Low-overhead Multicast*" Internet Draft, draft-perlman-simple-mulitcast-*.txt, Oct. 1999, work in progress

[20] C. Rigney et al, "Remote Authentication Dial In User Service (RADIUS)", RFC 2138, Apr. 1997

[21] C. Shields et al, "KHIP-A Scalable Protocol for Secure Multicast Routing", Proc. ACM SIGCOMM'99

[22] B. M. Waxman, "*Routing of multipoint connections*", *IEEE JSAC* 6(9) 1988, pp1617-1622