PAPER

Policy and Scope Management for Multicast Channel Announcement

Hitoshi ASAEDA[†a)], ***Member and*** **Vincent ROCA**[††b)], ***Nonmember***

SUMMARY A scalable multicast session announcement system is a key component of a group communication framework over the Internet. It enables the announcement of session parameters (like the {source address; group address} pair) to a potentially large number of users, according to each site administrator's policy. This system should accommodate any flavor of group communication system, like the Any-Source Multicast (ASM) and Source-Specific Multicast (SSM) schemes. In this paper we first highlight the limitations of the current Session Announcement Protocol (SAP) and study several other information distribution protocols. This critical analysis leads us to formulate the requirements of an ideal multicast session announcement system. We then introduce a new session announcement system called "Channel Reflector". It appears as a hierarchical directory system and offers an effective policy and scope control technique. We finally mention some design aspects, like the protocol messages and configuration structures the Channel Reflector uses.

key words: *multicast, session announcement, channel scope, directory system*

1. Introduction

Multicast communication is highly advantageous for contents distribution to a large number of receivers. Due to the multicast addressing architecture [1],[2], transient multicast addresses are dynamically assigned to each session for their entire duration, and released afterward. This is different from unicast addresses that are assigned to individual hosts for a long span of time. The direct consequence is that an end user, who is either a real person or an application running on a host, and who plans to join a multicast session must first resolve the transient multicast address used by the session.

There are two session discovery approaches on the Internet: the *"invitation model"* and the *"announcement model"*. In the *invitation model*, a user is explicitly invited by another user to join an on-going session. For instance the Session Initiation Protocol (SIP) [3] enables the user location discovery and the negotiation of session parameters. Although this approach works well within a small network, it is not suited for large multicast sessions since the inviter must know the unicast addresses of all possible participants beforehand.

The *announcement model* relies on a session directory system. *Sdr* is a well-known session directory system which has been intensively used in the Multicast Backbone (MBone) [4]. It can announce information for all available sessions to other directory systems, and assists end users to select the data flows they want to receive.

In both models, the Session Description Protocol (SDP) [5] is used to describe the session information, like the session name, the session time, the sender and multicast addresses, and the media format. This information is essential to enable each user to join the session. SDP does not specify how the information is distributed. This is traditionally the role of the Session Announcement Protocol (SAP) [6]. When a multicast application starts sending the data, it announces its session information to prospective participants with SAP. Therefore SAP is currently one of the necessary components of a session announcement system that follows the announcement model. Yet SAP has several major limitations as will be explained later on.

The goals of our work are therefore (1) to clarify the requirements that should fulfill an ideal session announcement system and (2) to propose a concrete architecture that can handle current and future needs, in particular when considering the scalability in terms of session announcements and the number of users, the need for policy and scope control mechanisms, and the support of any flavor of group communication system, like the Any-Source Multicast (ASM) and Source-Specific Multicast (SSM) schemes [7]. This paper only focuses on the session announcement architecture and leaves other aspects, e.g. how actual multicast data can be transfered effectively, to future work. Besides this paper considers neither multicast routing protocols nor its deployment aspects that are totally independent of our session announcement architecture since this latter only requires an inter-domain unicast routing service.

The remainder of this paper is organized as follows: Sect.2 examines the current multicast scoping techniques and analyzes SAP and scoping architecture. In the light of this critical analysis, Sect.3 formulates the requirements that an ideal multicast session announcement system should fulfill, and Sect.4 introduces

Manuscript received February 12, 2004.
Manuscript revised August 3, 2004.

[†]The author is with INRIA, PLANETE Research Team, 2004, Route des Lucioles, BP 93, 06902 Sophia Antipolis, France

[††]The author is with INRIA, PLANETE Research Team, 655, Avenue de l'Europe, Montbonnot - Saint Martin, 38334 Saint Ismier, France

a) E-mail: Hitoshi.Asaeda@sophia.inria.fr

b) E-mail: Vincent.Roca@inrialpes.fr

the architecture of a new session announcement system, called "Channel Reflector". In particular it focuses on the policy and scope management aspects. Sect.5 details some protocol and configuration aspects, and finally we conclude in Sect.6.

2. Analysis of Existing Scoping and Session Announcement Techniques

2.1 Scoping at the Multicast Routing Level

Multicast data senders or network administrators may want to define an area where data packets sent within a session will be confined. This area is called *"scope area"*, and *"scoping"* is the action of defining the scope area. In this scheme, only an end user who belongs to the scope area can receive the session data. This scoping mechanism has two major benefits: (1) it preserves bandwidth resources outside of the distribution area, and (2) it offers a certain level of confidentiality (since end users located outside of the scope will not by definition receive the session packets).

When IP multicast was initially deployed in the MBone, the Time-To-Live (TTL) field of the IP header was used to control the distribution of multicast traffic. A multicast router configured with a TTL threshold drops any multicast packet in which the TTL falls below the threshold. For instance, a router at the boundary of an organization configures the threshold to 32 which denotes an "organization" scope boundary. The drawbacks of this "TTL scoping" are: (1) the senders must be sufficiently aware of the network topology to determine the TTL value to use, and (2) complex scope areas cannot be defined (e.g. between overlapped areas). Especially the first point becomes big obstacles for general end users to precisely set up the data distribution area. *TTL scoping, which only defines a rough granularity, is definitively not an appropriate solution.*

On the other hand, the "administratively scoped IP multicast" approach [8] provides clear and simple semantics. Here scope boundaries are associated to multicast addresses. With IPv4, packets addressed to the administratively scoped multicast address range 239/8 (i.e. from 239.0.0.0 to 239.255.255.255) can not cross the configured administrative boundaries. Since scoped addresses are defined locally, the same multicast address can be used in different non-overlapping areas. Oppositely, an administrator can define multiple areas overlap by dividing the administratively scoped address range, which is not possible with TTL scoping.

Unfortunately, administrative scoping has several major limitations. An administrator may want to partition the scope area to disjoint areas on a per receiver basis, or he may want to limit data distribution according to the transmission rate or the content category of each session, or he may want to use the data sender's address as a keyword to set up the scope. Note that the latter aspect is nowadays feasible since Source-Specific Multicast (SSM) [7], which has recently been recognized as the most feasible multicast communication model in the Internet, requires that a join request specifies both the multicast and source addresses. SSM highlights another contradiction in the administrative scoping approach: the address range dedicated to SSM, 232/8 with IPv4 [9], cannot cover the address range dedicated to administrative scoping, 239/8. Although the problem can be solved by defining yet another SSM specific administrative scoping address range, such a patchwork defines a new addressing architecture which requires modifying application, end host and router implementations or configurations. *In our opinion, using multicast addresses to define a scope is not an appropriate solution either.*

2.2 SAP Protocol Analysis

The Session Announcement Protocol is a necessary component of a current multicast session announcement system. In a SAP announcement procedure, the entire session information must be periodically transmitted and all active session descriptions must be continuously refreshed. If ever a session is no longer announced, its description eventually times out and is deleted from the available session list. This is a major property of a *soft-state* protocol [10]. In contrast, a *hard-state* approach to flow state establishment would involve a specific setup and teardown mechanism, as with an FTP application.

The soft-state model has recently become popular because it enables to build robust and fault-tolerant systems and protocols on top of the best-effort UDP/IP semantics. However the periodic message transmission may cause major overheads. Additionally, improving SAP robustness and data consistency in front of packet losses requires transmitting each message several times. Although this strategy keeps the implementation simple, it arises important costs and further reduces its scalability.

The SAP specification addresses these issues by specifying that the bandwidth used by SAP announcement transmissions is limited by default to 4 kbps [6]. As a side effect, this solution largely increases the latency experienced by end users when the number of sessions increases. More formally, given [6]:

N = number of announcements
S = size of the announcement (bytes)
bw = defined bandwidth (bps)
I = interval (sec.)
O = offset (sec.)

the next announcement will be sent after T seconds:

$$T = I + O \tag{1}$$

where:

$$I = max(300, (8 * N * S)/bw)$$
$$O = rand(I * 2/3) - (I/3) \quad (2)$$

where $max(i, j)$ returns i if i is bigger than j, otherwise it returns j, and $rand(i)$ computes a sequence of pseudo-random integers in a $\{0; i\}$ range. Since T is at least 200, end users experience a minimum waiting time of 200 seconds to obtain the entire session list, irrespective of N, S and bw. We measured on the MBone that the average size of an announcement message is about 300 bytes. Therefore, when N reaches 500, I becomes greater than 300 seconds and the average value of T increases accordingly. Of course increasing the permitted bandwidth bw reduces the latency, but does not solve the fundamental problem.

Another key requirement is that SAP relies on the ASM model, since every SAP instance can send announcements in the SAP announcement group. For instance, to receive SAP announcement messages for the global scope IPv4 multicast sessions, all clients must join session 224.2.127.254 [6] (without specifying any source address). This is another major limitation of SAP since some Internet Service Providers (ISPs) may want to provide only SSM multicast routing. We believe that a versatile announcement protocol must not rely on any specific routing architecture.

2.3 Session Announcement Scoping

The *session announcement scoping* is a complementary scoping mechanism that operates at the session announcement level rather than at the data distribution level. It enables an announcement message to be confined to the smallest set of end users containing all potential legitimate receivers (and ideally only to them). As in section 2.1, we define the same *"scope area"* and *"scoping"* notions, but now restricted to the announcement messages.

The same motivations as those for scoping at the routing level apply here: (1) preserve bandwidth resources (this is less critical though, since the bandwidth required for announcements is usually lower than the one required for most sessions), and (2) offer a certain level of confidentiality (since local session announcements will be kept local).

A first idea could be to reuse the existing multicast routing scoping mechanisms to provide announcement scoping. This is the case with SAP since the announcement is multicast with the same scope as the session it is announcing. Both the TTL-based or administrative scoping mechanisms defined in section 2.1 are possible, with the same limitations as previously discussed.

Therefore *using multicast routing scoping techniques to offer an announcement scoping scheme is not sufficient in our opinion.* We will see in the following sections how to build an announcement scoping, *independently* of the underlying multicast routing scoping, in order to better control the scope area.

3. Multicast Session Announcement Scheme

3.1 Requirements

According to our analysis, an ideal session announcement scheme should fulfill the following requirements:
Scalability. A session announcement system can be used by a large number of end users spread throughout the Internet, and can manage a very large number of sessions.
Policy control. Administrators must be able to select what sessions announced from their internal network are announced outside, and vice versa. This policy control can be motivated by several criteria, like the transmission rate, the content, or duration of each session. This policy should be inherited along the hierarchy of the internal network, if any.
Scope control. As discussed in Sect.2.3, session announcement scoping is required to preserve bandwidth resources and offer a certain level of confidentiality.
High availability. The scheme must be robust in front of host/link failures and packet losses. This can be fulfilled either by transmitting messages periodically, as a soft-state approach does, or by keeping track of failures and recovering them if a hard-state approach is used.
Deployment on the existing infrastructure. The scheme must minimize changes to the current networking environment and protocols. In addition, it must accommodate (or be independent of) any kind of multicast routing protocol.

Additionally, the ideal session announcement scheme should optimize the following performance criteria:
Information consistency. Information consistency, which warrants that most (ideally all) end users have a consistant view of the announcements, is obviously of major importance.
Low information update latency. Multicast session information can be fully dynamic. The list of sessions should be updated rapidly after the creation, modification, or removal of a session announcement.
Low bandwidth consumption. A session announcement system should effectively consume the network bandwidth so that the system does not affect other communications or services.

3.2 Soft-State versus Hard-State Systems

We now discuss the soft-state versus hard-state approaches with respect to the three performance criteria identified in previous section. Let us consider a soft-state approach first. A trade-off clearly exists between the bandwidth consumption and the latency. Likewise, a certain level of information consistency can be

provided by frequent message transmissions over non-reliable connections, at the price of higher bandwidth consumption. No such problems exist with hard-state protocols, even if the overall complexity increases and includes explicit setup and teardown phases. These considerations suggest that a soft-state approach is incompatible with our design goal.

[11] offers an interesting complementary discussion, based on analytic Markov models, about soft-state, hard-state and protocols in between. One of the examples the authors show is that, although the data consistency decreases approximately linearly with the hop count of the data distribution tree, a hard-state protocol keeps a slightly higher consistency than a soft-state protocol. Their analysis supports the idea that a hard-state approach is preferable in front of our requirements.

3.3 Other Information Distribution Systems

Let us now discuss other potential multicast session announcement systems.

Domain Name System (DNS) is undoubtedly a successful information distribution system of the current Internet. Here a hierarchy of DNS servers maintains the information, and each prospective client can consult the database whenever required to obtain the desired information. DNS is a potential candidate to multicast session announcements, but two reasons prevent its use since they do not match our requirements: (1) precisely because DNS is already largely deployed, it is difficult to change all DNS systems including the client resolver to implement new record types and services supporting all (or most) of our requirements; and (2) because a DNS server does not necessarily access the original database upon each client request (instead it looks in its cache), the system cannot manage the information of highly dynamic sessions that are launched and stopped more frequently than the DNS cache refresh period.

Emails and the Web are two alternative ways of conveying session descriptions. Both applications are of wide use and are flexible enough to carry many kinds of information. To provide a multicast announcement service, however, either approach would have to rely on a central server. Defining and applying session scopes would be impossible, which contradicts our requirements.

[12] introduces session announcement architecture. It provides administrative scoping through protocol proxies called "agents" to admit layered multicast data transmission and reduces the start-up latency of end users. Although the split architecture proposed is useful to reduce the SAP latency problem, more frequent SAP message transmissions do not fulfill the scalability requirement we mentioned before. Besides the agent discovery process uses an "expanding ring search" approach which can greatly limit its feasibility and scalability depending on the network size and environment. Finally they only support TTL and administrative scoping, which we already identified as not being sufficient.

The Information Discovery Graph (IDG) [13] is a directory system that helps end users to discover sources of multimedia contents. It is structured as a semantic hierarchy: top-level nodes represent broad semantic categories (e.g. "sports" or "entertainment") that are progressively refined when going down in the hierarchy. A "hierarchical category-based directory system" is a reasonable choice for end users to discover interesting contents. However the IDG does not define any scope area. Besides, in order to avoid the SAP latency problem, an IDG manager must flood session information periodically to other managers. According to the simulations, the IDG finds session information faster than SAP, but the total multicast bandwidth is also increased.

A Content Discovery System (CDS) based approach [14] proposes a search engine for dynamic contents discovery. It uses several Rendez-vous Points (RPs) connected in a peer-to-peer (P2P) manner. The RP set is created by a system-wide hash function on the content name. In this architecture, a data sender first registers his content data to one of the RP set. An end user discovers the RP by hashing the content name, and then retrieves content information from this RP. This system has no single point of failure and offers a good scalability. However, since this approach leads to a logical flat overlay network, it cannot define per-content nested or hierarchical scope areas and cannot configure any administrator's policy, and therefore it fails to fulfill our requirements. In addition, each RP must synchronize content information among the RP set in a soft-state fashion. Although this synchronization procedure is a very important factor for the protocol measurement, this paper does not discuss about it with sufficient details. We also think that discovering the available contents only by searching their name is not powerful enough in practice.

4. Channel Reflector Architecture

In order to comply with the requirements stated above, we designed a multicast session announcement system called "Channel Reflector" (CR). This CR system appears as a directory system to announce multicast channel information[†] to the end users, who can retrieve the available or scheduled channel lists.

The CR introduces a new scoping mechanism that relies neither on a multicast address prefix nor on the

[†]Since the CR system focuses (but is not limited to) on SSM communications and services, we use the term "channel" instead of "session" hereafter.

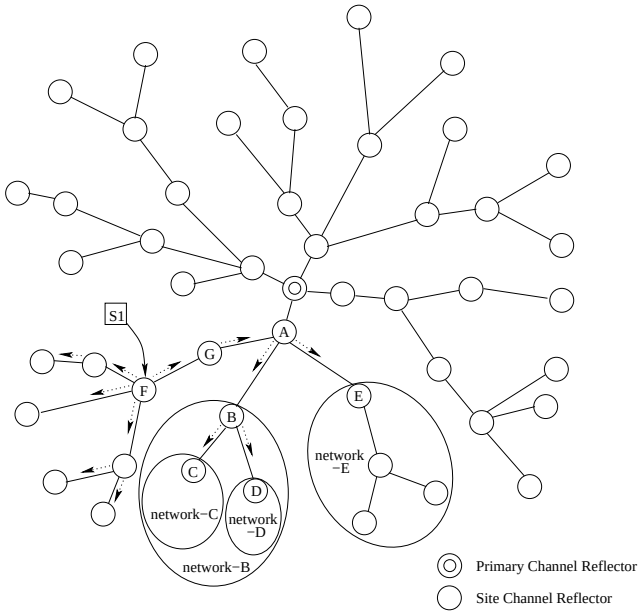

Fig. 1 Channel Reflector policy tree.

packet's TTL value. The CR architecture consists of a combination of a "primary CR" and one or more "site CRs". A primary CR is logically a single node in the Internet (its exact location is meaningless), while a site CR is located in each administrative network. The administrative network associated to a site CR forms a scope area. Each scope area is labeled with the site CR's Fully Qualified Domain Name (FQDN)[††], and is managed by an administrator.

A scope area covered by a primary CR is the entire Internet; therefore a primary CR retains information about globally available multicast channels, and its world-wide scope area is called "global scope". A scope area covered by a site CR is an administrative network; therefore a site CR retains information about multicast channels available in its own scope area. These available channels consist of global scope channels plus limited scope channels that are not announced to every CR.

Each CR has a "parent-and-child" relation with some other CRs and establishes a "policy tree" as shown in Fig.1. The root of the policy tree is necessarily the primary CR, while each site CR has one parent CR (either the primary CR or another site CR) and either several child CRs or no child at all (leaf). Although it is not compulsory, this policy tree usually adheres to the Autonomous System used by the BGP routing protocol [15] or other network topology hierarchies (e.g. routing tree). In other words, the hierarchy of the policy tree can be in accordance with the underlying network hierarchy (e.g. ISP – customer – customer's organization); this is what is assumed in this paper. Yet the policy tree is in general totally decoupled from the possible CR FQDN structure (e.g. in Fig.1 CR-C is not necessarily a sub-domain, in the DNS meaning, of CR-B). In addition, *all parent-and-child relations are configured statically* (this is similar to the static DNS primary and secondary server configuration).

Multicast channel information is transferred along this policy tree in a *hop-by-hop* manner. Thanks to this policy tree, the CR system provides the **scope control feature**; a multicast channel belonging to the defined scope area only appears on the CRs located inside the scope area, the *"scoped CRs"*.

In this scope control feature, a *"scope label"* is used to define the scope area of each channel. The scope label is simply one of the site CRs' FQDNs on the policy tree. Each site CR maintains available scope labels as the *"Scope List"* which consists of the FQDNs of all upstream site CRs on the same branch of the policy tree.

When a data sender or the site administrator (hereafter, both are referred to as the "registrant") registers a channel entry to his site CR, one scope label can be specified for the channel. If a registrant selects one of the available scope labels from the Scope List, his channel information is announced to the specified CR (upward) and to his site CR and its child CRs (downward). If he does not specify anything about the scope label, his channel information is only registered on the site CR and not transferred to other CRs. This condition sets up a site-local scope channel. If he wants to announce the data to the entire Internet, he must specify "global" (reserved word) as the scope label instead of a CR's FQDN.

After the channel registration, the channel entry is announced toward the CR having the corresponding FQDN, and downward to all child CRs up to the leaves of the tree. The CR having the corresponding FQDN becomes the *"scope boundary"* for the channel and this scope boundary does not forward the channel information to its parent CR.

For instance, in Fig.1, a scope area defined by CR-B is network-B. CR-C and CR-D define network-C and network-D that are the subnetworks under network-B. Host S1 registers a channel entry to its site CR (CR-F) and specifies CR-A's FQDN as the scope label. So the channel entry is transferred to the "neighbor CRs" (i.e. its parent CR, CR-G, and child CRs) and announced hop-by-hop toward the scope boundary (CR-A) and downward to each leaf CR.

The CR's policy tree also provides the **policy control feature**; the decision regarding which channel entries are imported and forwarded to the neighbor CRs depends on each CR's policy configuration.

Each site CR has its own policy configuration. Whenever the channel information is transferred from the neighbor CR, the site CR checks the properties of the channel and decides to import it or filter it out.

[††]An FQDN is a terminal name which resolves to a unique host on DNS, like "cr.example.com".

Criteria to take this decision can be for instance the planned transmission rate. This policy configuration is also inherited from the CRs upward in the policy tree, since each CR only forwards the channel information that it accepts.

In Fig.1, both CR-C and CR-D import S1's channel information, but CR-E filters out S1's channel information (by discarding the entry) due to CR-E's policy (e.g. if the stream plans to consume a bandwidth larger than the maximum threshold permitted by CR-E's administrator), and this entry does not appear on CR-E and its child CRs.

In order to fulfill the scope and policy control features, parent-and-child relations in this policy tree follow a *hard-state approach*. In the absence of any major event requiring a tree update (e.g. a CR failure), the parent-and-child state remains unchanged for an unbounded time. Reliable TCP connections are used between parent and child CRs in order to exchange announcement and control information. These choices are in line with our analysis of the hard-state versus soft-state models in Sect.3.2. With a hard-state protocol, once a connection is successfully created between a parent CR and a child CR, announcements can be transferred reliably, using TCP, and the sender knows that the remote CR is fully synchronized.

Yet if a parent CR fails, its child CRs cannot send or receive information to, or from, other CRs. This is one of the negative points of a hop-by-hop data transfer model and a hard-state protocol with static configuration (note that the same problem exists with the DNS infrastructure). While there are several possible solutions for ensuring a high availability in the CR system, a simple yet efficient solution consists in having a master CR (the official CR) and one (or more) slave CR (mirror): if the master CR fails, the slave CR takes over seamlessly. And once the master CR is recovered, it can synchronize all appropriate channel information from the slave CR.

Since the policy tree is statically established in the Internet, site administrators inform beforehand their end users of the appropriate site CR address (this is similar to the DNS client notification), which enables end users to retrieve the available or scheduled channel information. A user authentication/authorization mechanism is needed at each CR in order to avoid that illegitimate end users access the CR and retrieve channel information that do not belong to their respective scope areas. One of the simplest solutions is to set up an Access Control List (ACL) for legitimate nodes at the CR, but additional stronger security mechanisms like the authentication of the node [16] are encouraged to avoid that an illegitimate user spoofs his address.

5. Protocol Design

5.1 Scope Label Distribution

In the CR architecture, a primary CR must configure its child CRs statically, and a site CR must configure its parent and child CRs statically. While there is no other configuration required on a primary CR, a site CR additionally needs to set up its own "Scope List" to let a registrant select an appropriate scope area for his channel. To that purpose, the Scope List is exchanged among the CRs of each branch of the policy tree, thanks to two messages:

- SCOPE_NOTIFICATION
  This message has a type field to specify a JOIN or LEAVE operation. This message is sent to a parent CR.
- SCOPE_ANNOUNCEMENT
  This message has a LABELS type field to notify that a list of labels is included. This message is sent to child CRs, when a site CR receives SCOPE_NOTIFICATION (JOIN) message from a child CR, or when a site CR changes its own Scope List.

Since the messages are transmitted over a reliable TCP connection, they are sent only once.

When a site CR is initially attached to the policy tree or wants to refresh its Scope List, it sends a SCOPE_NOTIFICATION (JOIN) message to its parent CR. Once the parent CR has verified that the message originator is one of its child CRs, it sends back a SCOPE_ANNOUNCEMENT (LABELS) message with its Scope List. The site CR then registers the Scope List and appends its own FQDN. Fig.2 [A] shows such communications in a simple configuration.

When a site CR is removed from the policy tree or needs to change of parent CR, it sends a SCOPE_NOTIFICATION (LEAVE) message to its parent CR. Once the parent CR has verified the message originator, it disables the child CR (i.e. stops forwarding any information to the child CR). Since the site CR's Scope List is changed (i.e. its parent CR is removed), the site CR simultaneously sends a SCOPE_ANNOUNCEMENT (LABELS) message to its child CRs in order to eliminate its parent CR from each Scope List. This message is forwarded toward all leaf site CRs, and each CR can refresh the Scope List.

Note that a SCOPE_NOTIFICATION (JOIN) message is only used to obtain a Scope List from a parent CR, and SCOPE_NOTIFICATION (LEAVE) message is only used to disable the site CR from a child CR. In order to complete to shape or reform the policy tree, a site administrator must also change the static parent and child CR's configuration.

5.2 Channel Information Distribution

Channel information distribution is controlled by the following messages:

- CHANNEL_ANNOUNCEMENT
  This message contains a session description, and is used to announce channel information to the scoped CRs.
- CHANNEL_CANCEL
  This message cancels a channel information already announced.
- CHANNEL_RETRIEVE
  This message is used to obtain partial channel information. A site CR can specify various keywords (e.g. scope label(s) and bandwidth) to retrieve the appropriate information.

All messages are transmitted over a reliable TCP connection, like the scope labels, and they are sent only once.

A CHANNEL_ANNOUNCEMENT message is forwarded hop-by-hop toward the scope boundary and downward to each leaf CR to all the scoped CRs. Upon receiving this message, a CR first checks the scope label of the channel information. When the scope boundary receives this message, it stops forwarding the message to his parent CR. But if its channel scope is global, the message is forwarded to all CRs, including the primary CR.

Fig.2 [B] shows three channel announcement examples, assuming that there is no policy control that leads to discard the channel information.

Example 1: when CR-B registers a channel whose scope label is CR-A, CR-B sends a CHANNEL_ANNOUNCEMENT message to its parent CR (CR-A) and child CRs (CR-C and CR-D). CR-A stops forwarding the message upward, while CR-C and CR-D will forward it if they have child CRs.
Example 2: when CR-B receives a channel announcement from its parent CR, it forwards the message to all child CRs whenever the scope label is valid, i.e. is listed in its own Scope List.
Example 3: when CR-C sends a CHANNEL_ANNOUNCEMENT message whose scope label is CR-B, CR-B forwards the message to CR-D, but stops forwarding it to CR-A, because CR-B is the scope boundary.

These semantics can also avoid announcing invalid channels. For instance, since a legitimate channel must indicate either "global" or one of the scope labels kept in the site CR's Scope List, if a site CR receives channel information specifying an invalid scope label, it can just discard it.

The CR system, that uses a hard-state approach, needs an explicit message to cancel previously announced channel information. The CHANNEL_CANCEL message is used to that purpose when a registrant or the associated site CR's administrator wants to cancel an announcement. Upon receiving this message, and if the channel information is in its Scope List, the CR deletes the entry and forwards the message to the neighbor site CRs toward the scope boundary and each leaf CR. The message handling is the same as that of a CHANNEL_ANNOUNCEMENT message.

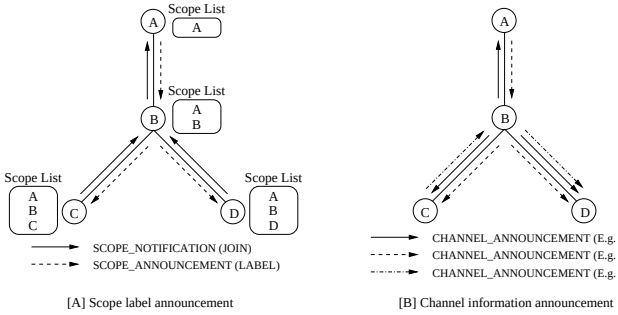

Fig. 2 Scope label announcement and channel information announcement.

A CHANNEL_RETRIEVE message enables a site CR to obtain partial channel information rather than all the channel information kept in the neighbor CRs. For instance, this is used to retrieve channels which were previously filtered out because of the previous policy (e.g. if an administrator increases the bandwidth threshold).

6. Conclusion and Future Work

In this paper we first analyzed the existing multicast session announcement and information distribution systems. This analysis led us to define requirements that an ideal multicast session announcement system should follow. We then introduced a new architectural approach called Channel Reflector (CR). A CR is a concrete system that provides effective policy and scope control mechanisms. In particular, a policy tree is created, using a hard-state model, that is more appropriate in this case than the commonly used soft-state model. Thanks to this policy tree, session (also called "channel" in this work) entry announcements are confined to their scope area. Here, the scoping mechanism relies neither on a multicast address prefix nor on the packet's TTL value, but on a scope label that is associated to each session when this latter is registered. This scope label is then used to make sure that the session information is only announced within the appropriate scope area along the policy tree.

We have already identified future work. Knowing the session entry distribution delays over a realistic topology and measuring the announcement traffic are two important performance criteria. To that purpose we have designed a simulator that will help us to ana-

lyze the behavior of the CR approach in case of a large scale deployment over the Internet.

Another future work is related to scalability. The current system may not work well if the number of site CRs largely increases, because the current session announcement tree is rooted at a single primary CR, and all global announcements must go through it. A solution to improve the scalability of the system would be to have multiple primary CRs. The most challenging aspect is to determine what policy topology would be the most effective in practice.

Finally, the CR architecture may help multicast routers in several tasks: if a multicast router accesses the local site CR, located in the same network, he can confirm that the source and group addresses of each join request triggered by their downstream hosts are legitimate. This is an asset since multicast routers traditionally need to perform complex source address discovery or validation procedures for establishing the routing trees whenever they receive multicast join requests. Moreover, data transfers on a session could easily be confined to the associated scope area, which encompasses all possible legitimate receivers, if multicast routers cooperate with the site CR and drop packets that leave the associated session scope. This approach would enforce that the CR's scoping mechanism, that operates at the session announcement level, and the multicast routing scoping mechanism, that operates at the data distribution level, are in line with one another. These two extensions will be considered in future work.

7. Acknowledgments

The authors would like to thank Kevin Almeroth, Torsten Braun, and Walid Dabbous who enabled us to improve the overall paper quality.

References

[1] S. Deering, "Host Extensions for IP Multicasting", RFC1112, August 1989.
[2] R. Hinden and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC3513, April 2003.
[3] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, "SIP: Session Initiation Protocol", RFC3261, June 2002.
[4] S. Casner and S. Deering, "First IETF Internet Audiocast", ACM SIGCOMM Computer Communication Review, vol.22, no.3, pp.92-97, July 1992.
[5] M. Handley and V. Jacobson, "SDP: Session Description Protocol", RFC2327, April 1998.
[6] M. Handley, C. Perkins and E. Whelan, "Session Announcement Protocol", RFC2974, October 2000.
[7] S. Bhattacharyya, "An Overview of Source-Specific Multicast (SSM)", RFC3569, July 2003.
[8] D. Mayer, "Administratively scoped IP multicast", RFC2365, July 1998.
[9] Z. Albanna, K. Almeroth, D. Meyer and M. Schipper, "IANA Guidelines for IPv4 Multicast Address Assignments", RFC3171, August 2001.
[10] S. Raman and S. McCanne, "A Model, Analysis, and Protocol Framework for Soft State-based Communication", Proceedings of ACM SIGCOMM, September 1999.
[11] P. Ji, Z. Ge, J. Kurose and D. Towsley, "A Comparison of Hard-state and Soft-state Signaling Protocols", Proceedings of ACM SIGCOMM, August 2003.
[12] A. Swan, S. McCanne and L. A. Rowe, "Layered Transmission and Caching for the Multicast Session Directory Service", Proceedings of ACM Multimedia '98, September 1998.
[13] N. R. Sturtevant, N. Tang and L. Zhang, "The Information Discovery Graph: Towards a Scalable Multimedia Resource Directory", Proceedings of IEEE Workshop on Internet Applications (WIAPP), July 1999.
[14] J. Gao and P. Steenkiste, "Rendezvous Points-Based Scalable Content Discovery with Load Balancing", Proceedings of International Workshop on Networked Group Communication (NGC), October 2002.
[15] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)", RFC1771, March 1995.
[16] S. Kent and R. Atkinson, "IP Authentication Header", RFC2402, November 1998.