Supporting the Need for Inter-Domain Multicast Reachability

Kamil Saraç
Department of Computer Science
University of California
Santa Barbara, CA 93106-5110
ksarac@cs.ucsb.edu

Kevin C. Almeroth
Department of Computer Science
University of California
Santa Barbara, CA 93106-5110
almeroth@cs.ucsb.edu

Abstract

Internet multicast is transitioning from the flat, virtual topology known as the Multicast Backbone (MBone) to a hierarchical, globally deployed service. This transition introduces a number of important management issues. In particular, it is believed that the success of multicast on a large scale partly depends on the availability of good management tools. In this paper, we address the specific management problem of monitoring multicast reachability. We first define the semantics of multicast reachability and how they differ from unicast reachability. We then motivate the importance of being able to monitor multicast reachability. Based on this discussion, we have developed a system, called sdr-monitor, to monitor reachability on a global scale. Using sdr-monitor we have collected almost a year of reachability data. In analyzing the data, we first process it to remove artifacts caused by using sdr. We then analyze the data to calculate a percentage of reachability in the multicast infrastructure. While we find that the current infrastructure is significantly unstable, the main reasons are likely the newness of inter-domain multicast and the challenge of developing "in-the-network" services on top of IP.

1. Introduction

Within the last decade, the multicast service model[1], the one-to-many or many-to-many data delivery model, has been implemented and deployed as the research oriented Multicast Backbone (MBone)[2, 3]. In this model workstations running a multicast routing daemon (mrouted) were connected to each other via IP tunnels over the Internet. The overall topology of this service model was a virtual, flat network. Since then, there has been a continuous effort to make multicast a ubiquitous Internet service. Most router vendors now support native multicast routing and Internet Service Providers (ISPs) have started to deploy multicast in their networks. Today, the multicast service infrastructure is shifting from the flat network topology, the original MBone, to a hierarchical topology. In this new topology ISPs run potentially different intra-domain multicast routing protocols within their own networks and use a particular set of protocols to provide inter-domain multicast support[3].

Despite the recent advancement in multicast routing, deployment in commercial networks has been observed to be a slow process[4]. ISPs have been facing a number of difficulties in deploying native multicast in their networks. Some of the most important reasons include: 1) legacy hardware that does not support native multicast; 2) instability in the current set of multicast protocols; and 3) a lack of tools to aid in deployment, debugging, and management[4]. As multicast routing protocols continue to improve and as legacy hardware is replaced, it will become easier to deploy native multicast. On the other hand, work on management tools needs to continue in order to facilitate this process[5].

Availability of good management tools has become a crucial part of multicast deployment. Deploying multicast without such tools is likely to cause problems for network engineers and yield a less than satisfactory user-experience[4]. Many of the current multicast management tools are "freeware" and are developed, as needed, by MBone users[5]. These tools commonly lack the perceived effectiveness of unicast tools. Of critical importance in assessing effectiveness is to differentiate between the needs of network engineers and end-users. End-users typically have very few tools available. For example, tools like *ping* and *traceroute* are the most common network-based tools. Beyond these tools, end-users tend not to have access to tools that can access internal network state. Network engineers on the other hand use tools like *ping* and *traceroute* plus network-invasive diagnostic tools. For example, management platforms based on the Simple Network Management Protocol (SNMP)[6] are widely used.

For multicast, not only are generic traffic management tools needed, but also tools specific to the special delivery characteristics of multicast. One particular area of management, and the focus of this paper, is reachability. In the unicast world, network engineers and end-users alike use *ping* and *traceroute*. The belief is that these tools are effective for unicast reachability. While these tools do provide this functionality, the actual information returned is minimal. *Ping* returns a yes or no answer. *Traceroute* returns little information beyond what *ping* does. A network engineer would use these tools only to quickly determine if there was a problem. Then, other tools, available only to the network operator, would be used to further diagnose the problem. A user simply knows that a problem exists, and possibly knows how close or far away the problem is. The bottom line is that while reachability tools give users little information, this is exactly appropriate. A user with more information cannot do much to solve internal network problems. For multicast, similar functionality is needed. However, tools to support multicast reachability require additional semantics. We begin the paper by discussing the semantics of unicast and multicast reachability.

The primary motivation for studying reachability has been the transition of the MBone into a hierarchical, native multicast infrastructure. When the MBone was the only multicast-capable network, reachability either existed or it did not. Being flat, an end-system either had connectivity or it did not. Cases of only partial connectivity existed but were not the norm[7]. However, the problem of reachability has become more important in the inter-domain environment. As there are no end-user or system-wide tools to monitor reachability, we have no idea of the robustness and stability of the multicast infrastructure.

In this paper, we focus on assessing reachability in the inter-domain. We start by discussing reachability and then motivate the need to do a study of the multicast infrastructure. We then present a simple system that performs this monitoring task for at least some of the multicast infrastructure. Our system, called *sdr-monitor* is based on the ability to quickly and easily gather statistics about reachability. Our architecture is based on using multicast session announcements sent and received by the *sdr* tool. While *sdr*-based session announcements do not represent 100% of multicast sessions, their periodic, soft-state refresh messages are sufficient for our needs in sampling inter-domain reachability. Using this system, we have collected data for more than a year and have found the multicast infrastructure to have significant instabilities. However, as multicast is still evolving and is one of the first "in-the-network" style services to be deployed, we feel that a certain level of instability is to be expected. We attempt to explain some of the possible reasons for this instability.

The remainder of this paper is organized as follows. Section 2 defines reachability and motivates the need for monitoring the multicast infrastructure. Section 3 gives an overview of our reachability monitoring architecture. Section 4 describes the data collection process, data processing, and preliminary results. The paper concludes in Section 5.

2. Defining and Motivating Reachability

With the deployment of native multicast in commercial networks, multicast is getting closer to becoming a ubiquitous service in the Internet. However, before multicast can be used as a revenue-generating service, its robust and flawless operation needs to be established in the inter-domain. This requires availability of good management tools to help network administrators configure and maintain multicast functionality within and between multicast-enabled domains. While unicast management is well established and provides good support for robust network operation, there still exists a need for good multicast management. The management function needed in particular is reachability monitoring.

For multicast, the relative newness and complexity of the service model makes it more difficult to develop necessary management tools. The current multicast model is an open service model that supports sessions in which anyone can send data to a multicast group and/or can join and receive data. In this model, senders and receivers may not always be known to each other. Support for dynamic groups makes multicast management, in particular reachability, more difficult. Because there is no implicit group coordination or management, there can be no implicit way of knowing group members. This means an explicit mechanism for determining membership is necessary.

One explicit mechanism for determining who group members are and whether there exists reachability between source(s) and receiver(s) is the *ping* utility. In unicast, *ping* allows a source/receiver to test bi-directional reachability to a peer receiver/source. This relationship is shown at the top of Figure 1. A multicast version of ping has slightly more complicated semantics. Specifically, there is a difference between what sources and receivers hope to determine. Moreover, the operation of *mping* potentially involves responses from multiple hosts. In an *mping* request, a response is solicited and returned from each group member. However, because of the open service model and because *mping* requests are made to a *group* instead of to a *receiver*, the source does not know from whom to expect responses. As a consequence, receivers who do not have bi-directional connectivity with the source will not be heard. This scenario is shown at the bottom of Figure 1. In reality, it might be that the receivers responding to an *mping* request are a very small portion of the overall group membership. An *mping* with these semantics likely does not help in recognizing connectivity problems.

A multicast *mping* tool that is truly analogous to a unicast *ping* would first have different semantics for sources and receivers. A *source mping* would determine all group members and then determine reachability to each. Complicating this process is the fact that bi-directional connectivity is not required for multicast applications, and therefore, may not exist[1]. A *receiver mping* would attempt to determine the set of active sources and then measure their reachability status. The operation of *mping* for these two scenarios are shown in Figure 2. Not having a version of *mping* that first determines group sources/membership and then tests reachability is a severe limitation. Not only is this function needed for important management functions like accounting, but it is needed for network monitoring functions like fault detection and isolation.

With the transition of multicast into a hierarchical infrastructure, network operators have a more difficult time verifying the robust operation to each and every other multicast-enabled domain. The unicast analogy of simply counting and monitoring Border Gateway Protocol (BGP)[8] routes does not work because the multicast infrastructure is not as stable. Furthermore, the topology changes much

[1] In reality, bi-directional connectivity is not required for UDP-based unicast applications either, but the use of *ping* as a reachability check certainly requires it.

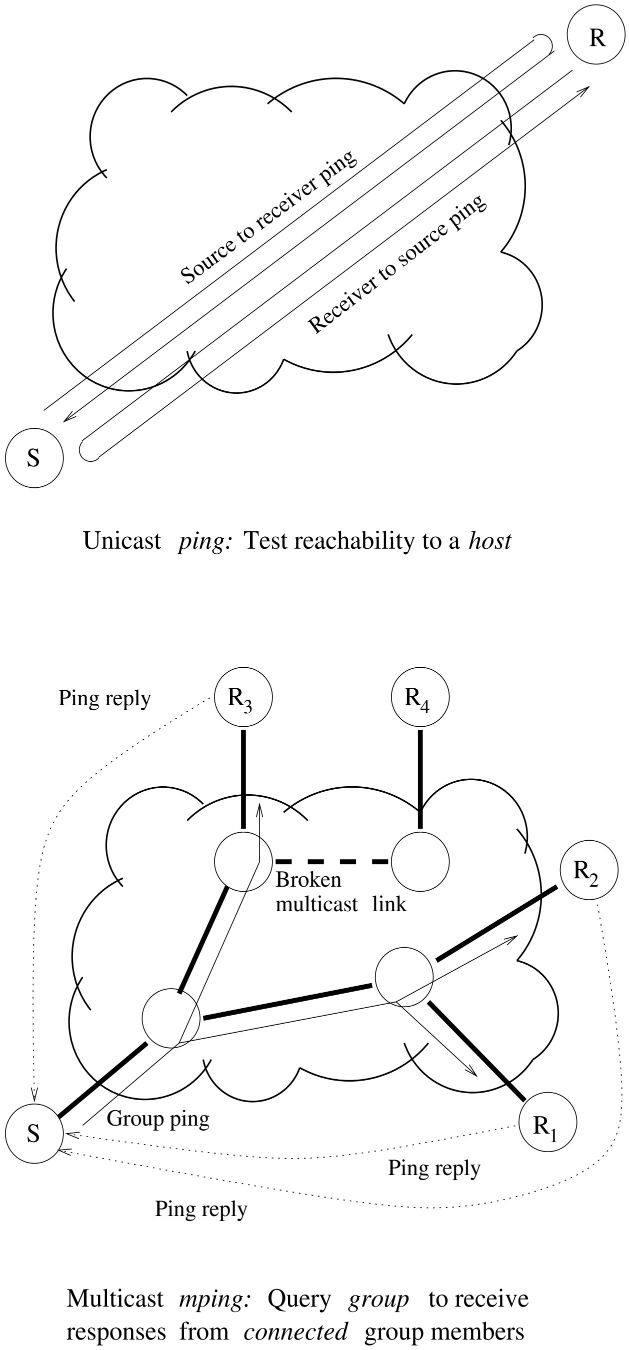

Figure 1. Semantics of the current unicast *ping* (top) and multicast *mping* (bottom).

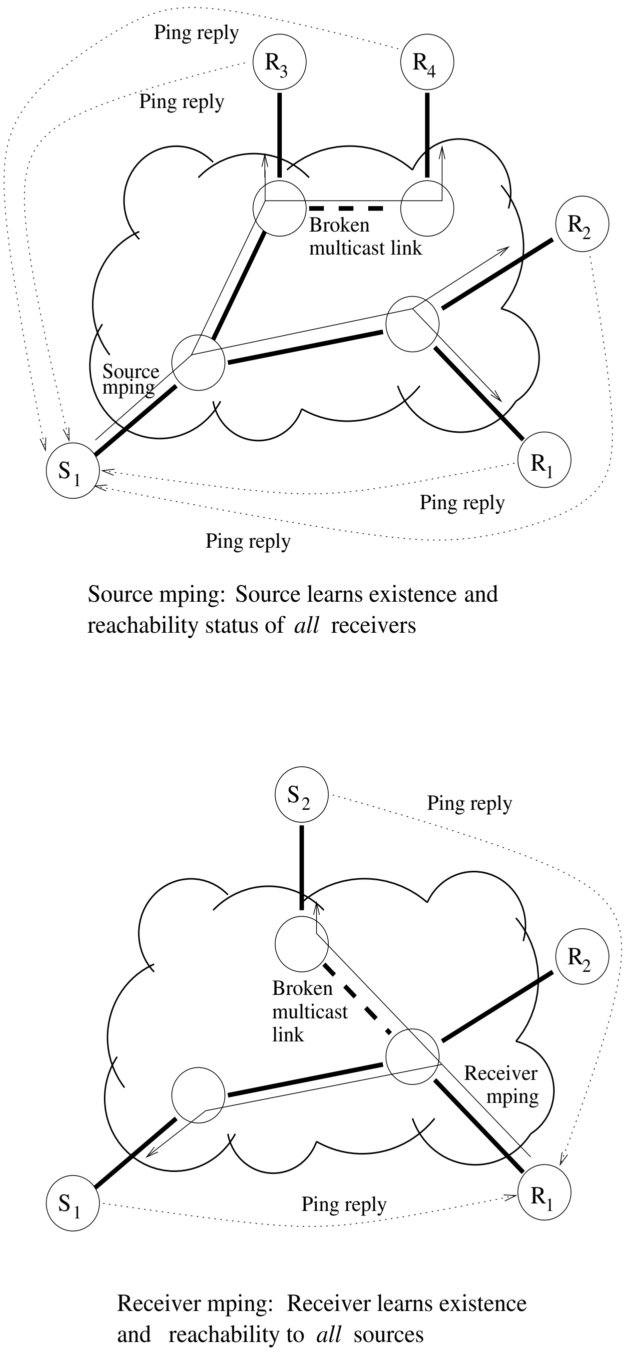

Figure 2. Semantics of a source (top) and a receiver (bottom) ***mping*****.**

more frequently as various operators deploy and remove test networks. In order to verify multicast reachability and help identify inter-domain multicast problems, we need tools to monitor and report reachability problems between multicast-enabled domains. In particular, there are two types of monitoring tasks that we believe are necessary to help support reachability monitoring. These are:

- **Testing during deployment/configuration**: As mentioned above, multicast deployment is a difficult service to deploy[4]. While intra-domain multicast routing is relatively straightforward to deploy, providing inter-domain multicast service requires knowledge of multiple protocols, cooperation between domains, and the active participation of several actual people. After deploying an inter-domain multicast service or performing configuration changes, network managers would need to test correct operation in both the intra-domain and the inter-domain. In these tests, network operators would like to learn whether their network can successfully send and receive multicast data to other networks in both incoming and outgoing directions.

  Network managers can test multicast connectivity in the incoming direction by joining several external sessions and checking if they receive data from external sources successfully. In cases where a network manager needs to test connectivity with a particular external domain, (s)he may need to find someone from this particular domain to create a test session and source data for testing purposes. To test multicast connectivity in the outgoing direction, a network manager can create a test session and source data to this session. In this case, the manager may need to contact people in other domains and ask them to join and test if they can receive session data successfully. The process of repeatedly setting up arbitrary sessions to test multicast connectivity is tedious at best and incomplete at worst. Since the multicast infrastructure has transitioned to a hierarchical topology, network administrators would now conceivably have to test reachability to every domain.

- **Monitoring stable multicast networks**: Continual monitoring can either consist of general reachability tracking or monitoring specific multicast sessions. During multicast transmission of an important event, network managers may want to monitor reception quality of the session data. This monitoring can help network managers to identify possible problems in data reception quality and take action to solve them if possible. If the problem is external or involves some other networks, then managers in respective domains can work together to fix problems.

Given this motivation and these scenarios, we now turn our attention to a system to perform this monitoring in the inter-domain. Our aim in developing this architecture is to create a quick and easily deployable system using existing mechanisms. This system will provide us with a basic measure of reachability status in the global multicast infrastructure and will help us better understand the requirements of reachability monitoring task in the inter-domain. Developing end-user tools with complete reachability monitoring functionality is a longer-term project left to future work.

3. An Architecture to Monitor Reachability

Our reachability monitoring system, called *sdr-monitor*, is based on multicast session announcements. Multicast session announcements are widely used to convey information about active groups to potential receivers. In our system, these announcements are also used as heartbeat messages and allow us to monitor reachability. The collection of global session announcements seen at a particular site can be used as a partial measure of reachability for that site.

We focus on global session announcements because they are intended for all receivers; there is no ambiguity about whether a site should be receiving a particular regional or local session announcement. Our belief is that by periodically collecting users' knowledge of active global sessions from many sites widely distributed around the world, we can generate a continuous visualization of the status of global multicast reachability. In the rest of this section, we review the session announcement mechanism used by *sdr* and then present details of our *sdr-monitor* system.

3.1. *Sdr* Session Announcements

As described above, we use multicast session announcements to monitor reachability. There are obviously many techniques available for a person who wants to create and advertise a multicast session. One technique which has been in use since the original MBone is to send session information using the Session Announcement Protocol(SAP)[9]. In SAP, announcements are periodically sent to a well-known multicast address (sap.mcast.net) with a certain scope. It is the responsibility of the underlying network to deliver these announcements to receivers joined to the SAP address. SAP is a soft-state based protocol so announcements are periodically transmitted but are not delivered reliably, i.e. there is no acknowledgement mechanism. From a reachability monitoring perspective, these periodic messages can be considered as one-way ping messages to a potentially large number of receivers. The challenge then is to determine whether receivers actually received the announcement.

Sdr is a tool for creating and communicating session announcements[10]. When a session is created using *sdr*, it asks for information about the session including session name, session description, media types, duration, etc. *Sdr* creates an announcement entry using the Session Description Protocol (SDP)[11]. *Sdr* then uses SAP to periodically send announcements. In addition, *sdr* listens to the SAP address for announcements by other users. When an announcement is received, *sdr* displays it in a session list. *Sdr* will periodically write the current set of announcements to an *sdr* cache directory. When a user starts *sdr*, the tool uses the cache to present the list of sessions stored when the tool was last running. Finally, users can also use *sdr* to launch the necessary decoding and display tools to receive session content.

The *sdr* tool has a feature that enables users to write customized code which is executed when certain conditions occur. Users put their code into a file called *sdr.tcl*. When the *sdr* tool is started, it automatically reads a user's *sdr.tcl* file and executes it along with the *sdr* tool. This enables users to add extra functionality into the *sdr* tool. As we will describe, we use this extensibility to collect information from *sdr-monitor* participants.

3.2. *Sdr-Monitor* Architecture

The basic mechanism of *sdr-monitor* is to collect currently available session announcement entries from topologically and geographically distributed sites and process them to build a representation of reachability status in the global multicast infrastructure. The components of this mechanism include:

- **Session Announcement Originators**: Any user that sends multicast session announcements to the SAP address (using *sdr* or any other tool) unknowingly becomes a source for *sdr-monitor* heartbeat messages.

- *Sdr-Monitor* **Participants**: Any user that listens to the SAP address can potentially be part of our project. Currently, *sdr-monitor* has almost 100 registered participants. *Sdr-monitor* participants use a *sender script* to send their *sdr* cache entries to the *sdr-monitor* site. The sender script is a small *Tcl* script that is appended to a participant's *sdr.tcl* file. Once *sdr* is started, the sender script is invoked once an hour. The sender script first forces *sdr* to write all currently available sessions to the cache directory. The sender script then reads the session entries from the cache directory and sends them to a collection site via email. This mechanism provides a technique to reliably collect what will later become reachability data.

- **Processing of Global Reachability Information**: At the collection site, the *sdr-monitor* manager receives e-mail messages and processes them. The manager uses these messages to generate, in real-time, a web page displaying global reachability between session announcement originators and *sdr-monitor* participants. The *sdr-monitor* tool periodically checks for incoming e-mail from participants and processes them; updating the web page with any changes in status. In addition, the manager takes a snapshot of the system every hour and archives it for long-term reachability analysis.

4. Reachability Analysis

In this section, we present an analysis of a nine month set of hourly snapshots taken of the *sdr-monitor* system. First, we describe the characteristics of the data set. Next, we describe the filtering and pre-processing we performed to remove non-representative participant reports from the data set. And finally, we present an analysis of the processed data set.

4.1. Data Collection

The data set used for this paper was collected between April 1, 1999, and December 31, 1999. During this time, the *sdr-monitor* site received over 141,000 e-mail messages from participants. As long as *sdr* was running at a participant site, our sender script (running in these sites) periodically packed the available session announcement entries into an e-mail message and sent it to us. The data set essentially reflects the perceived reachability status at participants sites. However, perceived reachability may not reflect the actual reachability status. In the next section, we list some of the causes and how we processed the data to eliminate these problems.

4.2. Data Processing

In this step, we filter potentially problematic participant reports from the data set. We identified three cases that necessitated some form of filtering. These include:

- **Filter all non-global session announcements**: Session announcements for any session with a time-to-live (TTL) less than 127 and administratively scoped session announcements are filtered.

- **Handling old** ***sdr*** **cache entries**: When a user starts *sdr*, the tool first reads the cached announcements and treats them as newly received announcements. *Sdr* then invokes the sender script code which sends e-mail messages to the *sdr-monitor* site. These announcement entries are potentially old entries and are not a good representative for the *current* set of active sessions. Therefore, we do not consider the first e-mail message sent by a user.

- **Handling old, problematic versions of** *sdr*: Session announcements are conveyed using a soft-state mechanism and should expire from receivers' caches after some time (one hour). But some old versions of *sdr* keep announcements as long as *sdr* runs and only expires them when a user quits. Therefore, in such cases, if a participant runs *sdr* for a long time, a number of entries may be old and not indicative of the current set of active sessions. Therefore, these sessions should not be used in reachability analysis. We identify such entries, based on a “last-heard timestamp”, and filter inactive ones.

4.3. Data Analysis

In this section, we analyze reachability using the processed data set. Our analysis is based on the identification of the set of global sessions being advertised around the world. We then assume that if a site does not report seeing an announcement then there is a reachability problem between the announcement source and the *sdr-monitor* participant site.

In our analysis, we use reachability of session-announcing-sites rather than reachability of individual announcements. While some sites are responsible for numerous session announcements, we are only interested in basic reachability. We do not want to skew our statistics by arbitrarily weighting sites, i.e. we do not want to favor the particular reachability characteristics of sites who advertise more than one session.

Our analysis is based on defining two types of reachability. Both types are a form of source-to-receiver reachability[2]. The two types of source-to-receiver reachability we consider are:

- **Source-Based Reachability**: For each source site, we compute the percentage of *sdr-monitor* participants who see announcements from the source. We take the number of *sdr-monitor* participants who see the session and divide by the total number of current *sdr-monitor* participants.
- **Receiver-Based Reachability**: For each *sdr-monitor* participant site, we compute the percentage of global sessions seen. We take the number of sessions seen by an *sdr-monitor* participant and divide by the total number of global sessions.

There is more of a semantic difference between these two types of reachability than there is a performance difference. Therefore, we only need to focus on computing one type of reachability. Our analysis shows results for source-based reachability.

Source-based reachability is computed by calculating the daily average visibility value for each site announcing one or more globally-scoped sessions. We then group these sites into four categories: sites with announcements that are seen by 0%-24%, 25%-50%, 51%-75%, and 76%-100% of *sdr-monitor* participant. Figure 3 shows our grouping and the normalized visibility ratios over time. In normalizing the results, we divide the number of sites at each group by the total number of sites per day. In the figure, the bottom area corresponds to session originator sites with up to 25% visibility, and the second area on top of it corresponds to session originator sites with visibility 25% to 50%, etc. Looking at Figure 3, the conclusion seems to be that multicast reachability is unstable, and overall reachability is very poor.

[2]Because we are taking advantage of session announcement messages from source sites, we are not able to monitor reachability in the reverse direction, i.e. receiver-to-source reachability.

In the last two parts of this section, we explain our understanding of why the results appear as they do. The first identifies the inherent limits of using *sdr* as our underlying reachability monitoring mechanism. The second identifies what we feel are true reachability problems.

4.3.1. Limitations Due to Using *Sdr*

We believe using *sdr* contributes to irregular reachability results in four ways. These are:

- **Participant Participation**: During the data collection period, not all *sdr-monitor* participants are running *sdr* continuously. This means that the number and identity of participants actively sending reports is not consistent over time. Since each participant has a potentially different picture of global reachability, their joining and leaving cause changes in overall reachability characteristics.
- **Session-Announcing-Site Behavior**: Similar to the above case, the number of sites sending session announcements is also dynamic. Sites can start and stop announcements at their own will. Different session-announcing-sites might have different reachability characteristics. When different sites start or stop, normalized reachability ratios can be significantly affected.
- **Visibility Changes at Announcement Start and End**: When a site starts sending a session announcement, it takes some time until the announcement reaches to all participants. During this startup time, the reachability of such session-announcing-sites will be relatively poor. Similarly at the end of a session announcement, we see a similar behavior. Only once a session has achieved stability is it worthwhile to use it for reachability analysis.
- **Short Lived Sessions**: Due to the reachability characteristics at announcement start and end, session announcements with a short life time, e.g. a couple of hours, will have poor reachability for almost their entire lifetime. In our analysis we encountered 2550 (out of 5640) announcements with a duration of less than 6 hours.

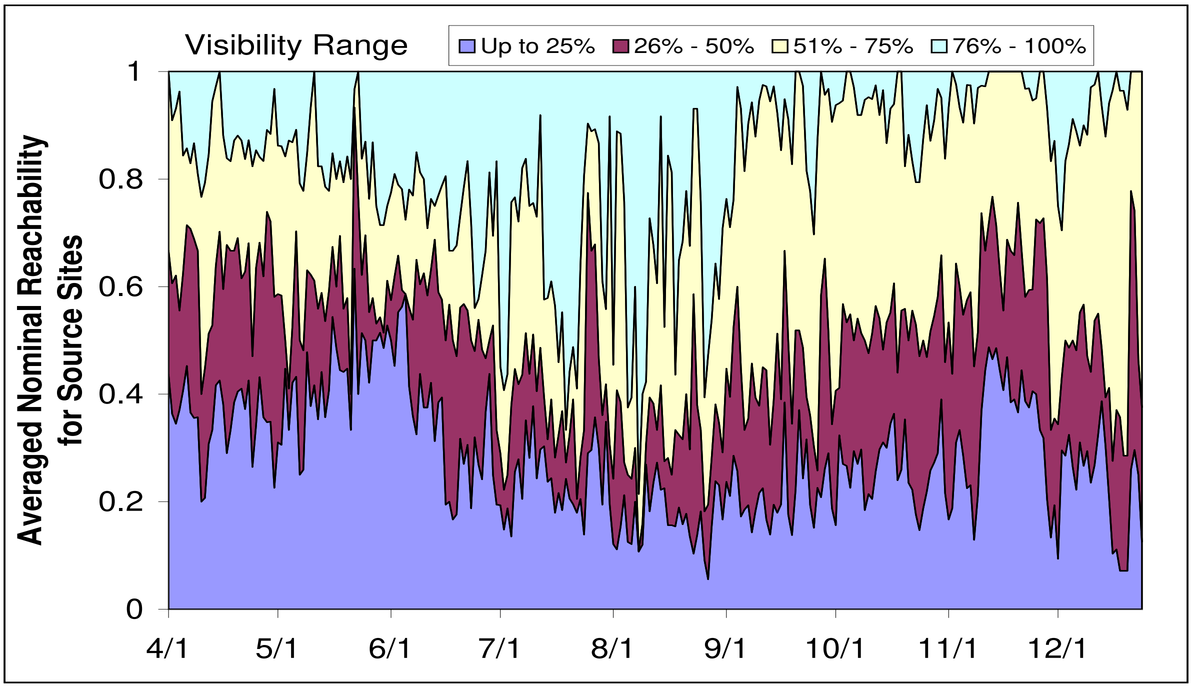

Figure 3. Visibility ratios for source-based reachability.

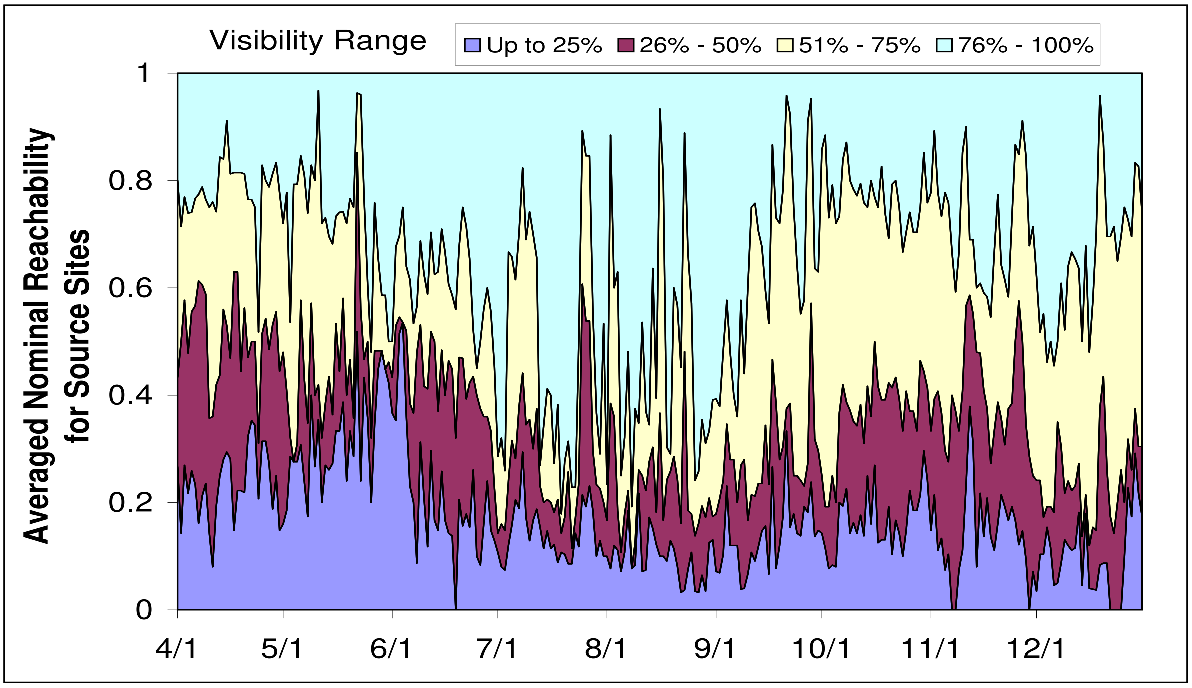

Figure 4. Visibility ratios for source-based reachability - after filtering sdr artifacts.

The reasons mentioned above clearly affect the reachability characteristics displayed in Figure 3. However, these effects are not due to true reachability problems and should be eliminated from consideration. In the next step of the analysis, we remove a number of data entries from our data set. Some of these entries belong to short-lived sessions (sessions living less than 6 hours) and some of them belong to session start and end intervals. By removing these entries, we filter out the negative effects of our data collection mechanism. Figure 4 shows the reachability status in a similar way as in Figure 3 but after this filtering process. According to this figure, the overall reachability status improves but still indicates some problems.

4.3.2. True Reachability Problems

We now describe a number of reasons that we have qualitatively identified as causing many of the reachability problems. We identify three types of problems. They are:

- **Site Inter-Domain Connectivity Problems**: During the data collection period, we observed cases in which some participants report only announcements sent by sites local to them. We can easily infer that such a reporting site has problems with at least receiving capability. If other *sdr-monitor* participant sites do not see session announcements from the local senders than we can conclude that there are also transmission problems. While local connectivity problems do occur, our intuition is that they are far less frequent than inter-domain problems.

- **Inter-domain Connectivity/Peering Problems** Another observation is that a number of announcements were only reported by one or a few number of non-local participants. Contrary to the above case, session-announcing-sites and participant sites were not local. However, they were topologically close. This situation is likely caused by inter-domain peering problems between multicast-enabled networks. Given the state of multicast protocols, the inter-domain protocols are newer and more unstable than intra-domain protocols. Still, the existence of *some* reachability is troubling. The multicast infrastructure seems to be exhibiting clouds of connectivity. In the unicast world, connectivity is robust and even if some links fail there is usually some sort of failover. Apparently multicast does not have this redundancy and is therefore less robust.

- **Transatlantic Connectivity Problems** The last reason is frequent problems with transatlantic links. In our data set, we observed numerous cases in which announcements originating from a site in Europe were only visible to participants in Europe, and announcements originating from a site in the United State were only visible to US participants. We strongly believe that these cases are related to network congestion and/or multicast connectivity problems between Europe and the US.

In the above discussion, we identified a number of problems contributing to poor global reachability. In considering the duration of the outage, we consider problems either to be short-lived or long-lived. Short-lived problems were quickly identified and corrected by network administrators. On the other hand, long-lived problems caused session announcements to experience poor reachability for extensive periods of time. Future work will need to examine these characteristics in more detail.

5. Conclusions

In this paper we address the issue of multicast reachability. Starting with the definition of multicast reachability and its fundamental differences to unicast reachability, we have motivated the need to monitor reachability in the growing global multicast infrastructure. We then presented a tool to monitor multicast reachability. This monitoring helps network administrators to identify potential problems related to multicast reachability within and between multicast-enabled domains. Using this tool, we have collected multicast reachability information for over a year. With this data, we have investigated the long term reachability behavior of the global multicast infrastructure. Our analysis has shown that the overall reachability is unstable and generally poor. We have identified a number of reasons for this behavior. We believe that the reasons are not indicative of a fundamental failure in multicast, but rather, are the result of the newness of multicast and the current difficulty in deploying it. With the deployment of native multicast, protocols will improve, network administrators will become more experienced; and most problems should diminish.

References

[1] S. Deering and D. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Transactions on Computer Systems*, pp. 85–111, May 1990.

[2] S. Casner and S. Deering, "First IETF Internet audiocast," *ACM Computer Communication Review*, pp. 92–97, July 1992.

[3] K. Almeroth, "The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment," *IEEE Network*, January/February 2000.

[4] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, January/February 2000.

[5] K. Almeroth, "Managing IP multicast traffic: A first look at the issues, tools, and challenges." IP Multicast Initiative White Paper, August 1999.

[6] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser, "Protocol operations for version 2 of the simple network management protocol (SNMPv2)." Internet Engineering Task Force (IETF), RFC 1905, January 1996.

[7] D. Massey and B. Fenner, "Fault detection in routing protocols," in *International Conference on Network Protocols (ICNP)*, (Toronto, CANADA), November 1999.

[8] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP-4)." Internet Engineering Task Force (IETF), RFC 1771, March 1995.

[9] M. Handley, "SAP: Session announcement protocol." Internet Engineering Task Force (IETF), draft-ietf-mmusic-sap-*.txt, March 2000.

[10] M. Handley, *SDR: Session Directory Tool*. University College London, November 1995. Available from ftp://cs.ucl.ac.uk/mice/sdr/.

[11] M. Handley and V. Jacobson, "SDP: Session description protocol." Internet Engineering Task Force (IETF), RFC 2327, April 1998.