# Practical Utilities for Monitoring Multicast Service Availability

Pavan Namburi
Department of Computer Science
University of Texas at Dallas
Richardson, Texas - 75083

Kamil Sarac
Department of Computer Science
University of Texas at Dallas
Richardson, Texas - 75083

Kevin Almeroth
Department of Computer Science
University of California Santa Barbara
Santa Barbara, California - 93106

***Abstract*— Monitoring has become one of the key issues for the successful deployment of IP multicast in the Internet. During the last decade, several tools and systems have been developed to monitor several different characteristics of IP multicast. In this paper, we focus on one specific monitoring task: monitoring end-to-end multicast service availability in the inter-domain. This task is important to maintain service robustness between sources and receivers. Without this assurance, the multicast infrastructure may become disconnected and essentially unusable. In this paper, we first study existing multicast diagnostic tools (e.g. *mping* and *mtrace*) and present their shortcomings in verifying end-to-end multicast availability. Then, we propose new multicast diagnostic utilities (*mcping* and *mcroute*) that can be used to perform various monitoring and measurement functions including verification of end-to-end service availability. We present a sample case study demonstrating the utility of these primitives in detecting and classifying multicast reachability problems in the inter-domain. The proposed utilities introduce only a few modifications to the service architecture and, in exchange, provide the multicast community with effective means to monitor and measure multicast service characteristics.**

**Keywords: Multicast monitoring, reachability monitoring, multicast availability.**

## I. Introduction

IP multicast [1] provides scalable and efficient mechanisms to support multi-receiver network applications in the Internet. Most of the work in IP multicast has been on developing the necessary protocols [2]; deploying them in the Internet [3]; and providing a number of additional services on top of the infrastructure including reliability [4], security [5], [6], and congestion control [7]. In addition to these efforts, the successful global deployment of an IP multicast service needs or strongly benefits from the availability of monitoring and measurement tools/systems.

Multicast service is currently widely used in intra-domain environments by enterprise networks [8]. Several multimedia players (Microsoft Media Player, Real Player, Helix Player, etc.) support the service. There are also companies (e.g., Digital Fountain Inc. and Multicast Technologies Inc.) that provide one-to-many bulk data transfer services using multicast [9]. On the other hand, the use of multicast in the inter-domain has been facing challenges [3]. One important challenge has been the complexity of the protocol architecture necessary to support the Any Source Multicast (ASM) service model. The introduction of the Source Specific Multicast (SSM) [10] service model removes most of these challenges and makes multicast more deployable and usable in the inter-domain. Multicast is now at a critical juncture and the success of inter-domain deployment and usage depends on the availability of monitoring tools to verify the availability and robustness of the service.

Multicast is realized through the creation and maintenance of forwarding trees connecting sources and receivers in a multicast group. These trees are dynamically created and maintained by the routers, yet there is no feedback information built into the process. If a group join fails because there is no path to the source, the receiver will never know. Local connectivity problems, inter-domain connectivity problems, link failures, node failures, configuration errors, policy incompatibilities, and congestion-related persistent data loss are possible reasons for multicast failure. Consequently, the ability to monitor service availability becomes very important to maintaining the robustness of the multicast service between sources and receivers. Without this assurance, the multicast infrastructure may become disconnected and essentially unusable.

During the past several years, a number of monitoring and measurement systems have been developed for IP multicast. As we discuss in Section II, these tools and systems were developed for performing specific monitoring and/or management functions. And while they have been successful in achieving their design goals, they fall short of performing all the functions necessary for monitoring multicast connectivity end-to-end. In addition, compared to unicast, we argue that IP multicast still suffers from a lack of practical and effective tools and primitives to monitor and measure the availability and performance of the service in the inter-domain.

In this paper, we use the term *availability* to indicate that an end system can successfully join a multicast group and receive multicast data from a remote end system sending to the group. Availability implies *connectivity* and *reachability* between the source site and the receiver site. Connectivity indicates the existence of a multicast join path between the receiver site and the source site. Connectivity also implies that multicast service is deployed in the source domain, in the receiver domain, and in all the other domains between the two. On the other hand, reachability indicates that a multicast forwarding path from the receiver site to the source site can be established and source data can successfully propagate toward the receiver site on the

established forwarding path. In this paper, we use the terms *availability* and *reachability* interchangeably.

In unicast, *ping* and *traceroute* are frequently used to monitor and measure the availability and performance of unicast in the Internet. In unicast, *ping* provides a convenient way of discovering reachability to a given remote system. The *ping* utility can also be used for multicast but provides different functionality. Multicast *ping* (*mping*) requests are sent to a multicast group address and these requests trigger group receivers to send *ping* responses to the pinging host via unicast. This essentially informs a pinging host that there are a number of receivers that received the request and sent their responses. The received information has only very limited value and the mechanism is vulnerable to feedback implosion. As a result, compared to the unicast *ping*, *mping* does not really help verify multicast reachability to a given remote system in the network.

*Mtrace* [11] is a multicast version of the *traceroute* utility. It is used to discover the multicast route between a given receiver and a source in a multicast group. A successful *mtrace* verifies the existence of a multicast route (i.e., the existence of a join path) from the receiver site to the source site. On the other hand, since connectivity does not necessarily mean reachability, the information obtained by *mtrace* does not always indicate reachability from the source to the receiver via multicast. In other words, even if the receiver-to-source join path exists, source data may not reach all the way to the receiver on this path. Problems due to forwarding errors, inappropriate TTL thresholds, or other configuration and interoperability issues may prevent multicast data from reaching the receiver. But, since *mtrace* checks connectivity (i.e., the existence of a receiver-to-source join path) only, it may fail to capture and locate potential reachability problems on the reverse data forwarding path. As a result, it fails to detect and locate a number of important multicast reachability problems in the network.

Based on the above observations, we argue that we do not have the multicast equivalents of the unicast *ping* and *traceroute* tools to perform basic monitoring for IP multicast. Therefore, in this paper, we develop a multicast equivalent of these tools which we call *mcping* and *mctrace* respectively. We also show how these tools can be used to monitor multicast in the inter-domain to detect and classify existing multicast problems in the global infrastructure.

The rest of the paper is organized as follows. The next section presents the related work in the area. In Section III, we introduce the *mcping* tool and demonstrate its utility through a case study. In Section IV we present the *mcroute* tool and present its usefulness in classifying multicast reachability problems in the inter-domain. Section V addresses several deployment and security issues. Finally, the paper concludes in Section VI.

## II. Related Work

The related work in multicast monitoring can be divided into two groups. In the first part, the main focus is to develop monitoring tools and systems to conduct inter-domain multicast monitoring. The goal in this direction is to provide monitoring services to multicast researchers and protocol designers to understand the overall operation of the service in the inter-domain. In the second direction, the main focus is developing necessary monitoring tools and systems for the operational management of multicast in the intra-domain. The goal in this direction is to develop necessary support primitives and services for network operators to effectively monitor and manage multicast in their networks. We briefly discuss the related work below.

### A. Inter-Domain Level Monitoring Tools

Inter-domain monitoring has been useful in understanding the multicast protocol interaction in the Internet and in measuring the robustness of the service in the inter-domain. The results of these monitoring efforts have been used to identify potential architectural problems and/or shortcomings of the protocols. These efforts lead to further research, standardization, and deployment. Inter-domain multicast monitoring tools can be divided into two groups: (1) application layer monitoring tools and (2) network layer protocol monitoring tools.

The early monitoring and measurement tools developed for IP multicast were mainly application specific monitoring tools and included *rtpmon* [12], *mhealth* [13], *sdr-monitor* [14], and the *multicast beacon* [15]. Due to their dependency on application-layer information, the tools in this group do not require any changes or additions to the infrastructure to perform their task. This makes them relatively easy to deploy and use. On the other hand, their scope is generally limited to the application that they leverage for their monitoring.

*Rtpmon* was designed to monitor quality of service characteristics as observed at the receiver sites in a multicast application. *Rtpmon* joins a particular multicast group address and receives feedback reports from all receivers. The feedback reports are generated by group members using the Real-time Transport Control Protocol (RTCP) which is part of the Real-time Transport Protocol (RTP) [16]. Due to its dependence on RTCP, *rtpmon* can only be used for monitoring applications that use RTP as their transport mechanism.

*Mhealth* combines *rtpmon* and *mtrace* to display a real-time, graphical representation of a particular group's multicast tree including packet loss characteristics of each link in the tree [13]. Similar to *rtpmon*, *mhealth* depends on RTP to perform its task and is not particularly scalable.

*Sdr-monitor* uses periodic session announcements as a monitoring heartbeat message. The announcements are created and exchanged between a large number of multicast users to inform each other about future multicast events. *Sdr* [17] is a well-known tool that is used to generate session announcements on the Internet. By collecting the available announcement messages from a number of different *sdr* user sites, *sdr-monitor* builds a real time reachability matrix presenting the multicast reachability characteristics among a number of multicast end points that use *sdr*. More recently, the *multicast beacon* [15] was developed as a follow-on project. It uses active monitoring

probes to monitor multicast reachability characteristics among a number of multicast end points. The end points are multicast users that volunteer in the monitoring effort.

The common characteristics of the above approaches are that they all depend on an existing application or application layer protocol to perform their specific monitoring task. They all lack the flexibility to extend their functionality to be a more general monitoring and measurement tool.

Among the network layer protocol monitoring tools, we present *mantra* [8] as the main example of a system developed to monitor the multicast routing infrastructure. *Mantra* collects multicast routing table information from a number of Internet backbone routers and processes this information to create a global view of the infrastructure. The information collected by *mantra* has helped researchers and network administrators understand the functioning and interaction of the various multicast routing protocols. *Mantra* uses approximately a dozen vantage points to collect its data and therefore can only present a partial picture of the global multicast routing infrastructure. Its ability to identify and isolate specific problems is also rather limited.

### B. Intra-Domain Level Monitoring Tools

The second group of related work includes protocols (*MRM* [18]) and systems (*SMRM* [19], *Mmon* [20], and *MRMON* [21]) that have been developed to monitor and manage IP multicast services in the intra-domain. These systems are powerful as they provide network administrators with the necessary primitives to monitor multicast availability along with a set of other multicast monitoring and management functions. On the other hand, they are limited to use in intra-domain environments.

The Multicast Reachability Monitor (*MRM*) [18] is a protocol used to create active and passive multicast monitoring and measurement scenarios. *MRM*-capable network devices can be configured to run an active multicast test session and collect performance information. Or, they can be configured to measure multicast quality for an ongoing application. *SMRM* [19] is a follow-up effort that incorporates *MRM* functionality into a Simple Network Management Protocol (SNMP)-based framework so as to provide a standard approach to perform multicast monitoring tests in the network.

*Mmon* [20] is a multicast network management suite developed by HP Labs and then included into HP's OpenView network management system. *Mmon* uses a number of multicast related routing protocol tables and Management Info Base (MIB) tables to provide a complete suite of multicast network management solutions. The Remote Multicast Monitoring (*MRMON*) [21] project is a more recent attempt that uses an SNMP-based framework to collect various types of multicast performance metrics from multicast end systems. It defines several multicast MIB groups to collect a comprehensive set of information about ongoing sessions. MRMON is a passive monitoring system and does not consume a large amount of network resources as in the case of active monitoring systems.

### C. New Approaches to Monitor Multicast Service Availability

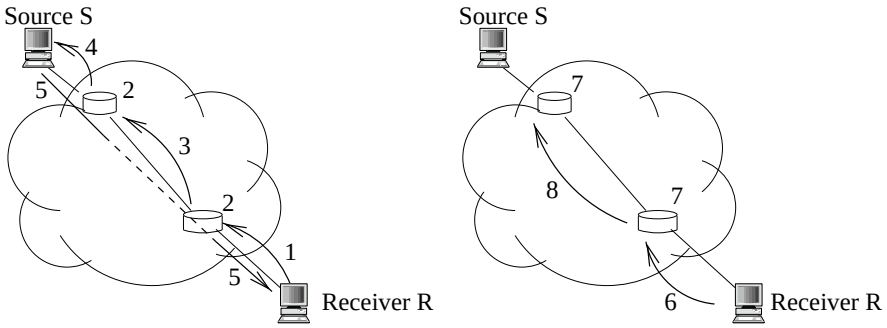
In this paper, we present *mcping* and *mcroute* as two multicast diagnostic tools analogous to unicast *ping* and *traceroute* respectively. As we discussed in Section I, *mping* and *mtrace* are not very effective in verifying end-to-end service availability for multicast. In addition, our discussion in this section shows that the existing multicast monitoring tools are limited in terms of their scope and functionality. Our goal, therefore, is to develop relatively simple primitives (*mcping* and *mcroute*) to verify multicast service availability and/or locate problem spots between two given remote end points. Considering the importance of *ping* and *traceroute* in unicast monitoring and measurement studies, we expect our tools (*mcping* and *mcroute*) to be equally important for multicast. In order to demonstrate the utility of our tools, we will present measurement studies that we have conducted using these tools.

## III. *Mcping*: A Multicast Ping Utility

In this section, we propose the *mcping* utility for verifying multicast availability between a local host site and a remote site. In this context the local host is assumed to be a potential multicast receiver and the remote host is a potential multicast sender. A positive response to *mcping* indicates that the local receiver can successfully join and receive multicast data from the remote host. By using a dedicated multicast group address, say PING.MCAST.NET, in the source specific multicast address range (232/8), an end system, R, sends an *mcping* request to a remote host, S, and expects to receive a *mcping* reply on the (S, PING.MCAST.NET) multicast channel. Since the overall mechanism uses the existing multicast service architecture between the two end points, the result of the test gives a definitive answer on the availability of multicast service to the remote system, S.

The proposed *mcping* mechanism works as follows. *Mcping* first sends an Internet Group Management Protocol (IGMP) [22] join request on the multicast channel (S, PING.MCAST.NET). Upon receiving this message, the Designated Router (DR) at the pinging site creates a Protocol Independent Multicast (PIM) [23]-based join message for (S, PING.MCAST.NET) and forwards it toward the pinged system, S. Each router on the R-to-S reverse shortest path creates a forwarding entry for the multicast channel (S, PING.MCAST.NET) and forwards the join message towards S. When the join request reaches the DR at S's subnet, this router forwards a message to S informing it about the *mcping* request. On receiving the *mcping* request, S creates a reply message and sends it to the (S, PING.MCAST.NET) multicast channel. This message propagates on the established multicast forwarding path between S and R and reaches the pinging host, R. The procedure is visually presented in Figure 1-a. During this operation, any problem that prevents the PIM-Join message from reaching S's site or the ping response from reaching R's site indicates the lack of multicast.

On receiving the *mcping* response, the pinging host R sends an IGMP Leave Group message to leave the multicast channel (S, PING.MCAST.NET). Consequently, the DR at R's




1. IGMP Receiver Report for (S, PING.MCAST.NET).
2. Router creates forwarding state for (S, PING.MCAST.NET).
3. Router forwards PIM-Join message for (S, PING.MCAST.NET).
4. Designated router on S's site sends an IGMP message to S to inform it about the mcping request.

(a)

5. S sends mcping reply on (S, PING.MCAST.NET).
6. IGMP message to leave the (S, PING.MCAST.NET) channel.
7. Router removes (S, PING.MCAST.NET) entry from its forwarding table.
8. Router sends a Prune message to upstream neighbor toward S.

(b)

Fig. 1. Operation of the *mcping* utility.

site sends a PIM-Prune message to its upstream neighbor on the tree to start flushing the forwarding state for (S, PING.MCAST.NET) in the network. Figure 1-b presents this operation. As a result, *mcping* provides R with the ability to test the availability of multicast to a remote end system, S. The mechanism does not depend on any other application and it does not require any user intervention or interaction.

### *A.* Mcping *Requirements*

*Mcping* requires a few modifications and additions to the existing multicast architecture. The basic mechanism uses the existing PIM-Join procedure to send the *mcping* request all the way to the DR of the pinged end system (S in the above discussion). At this point, it requires a new message between the DR and S (step 4 shown in Figure 1-a) to inform S about the incoming ping request. In order to achieve this, we introduce a new IGMP message. Using this message, the DR will inform S about the incoming ping request. This new message type will only be used when the DR receives a ping request. Normal join requests will terminate at the DR as usual.

An alternate approach would be to have the DR create a ping response and send it to (S, PING.MCAST.NET) on behalf of S. This approach would avoid modifications to IGMP but would fail to capture potential problems between the DR and S. As a matter of fact, during our implementation efforts we experienced one such problem.

In order to understand the development and operational issues, we built a test setup between an end host, $H_{UTD}$, at our site at the University of Texas at Dallas (UTD) and another end host, $H_{UO}$, at the University of Oregon (UO). Figure 2 shows the network layout, with the part of UTD magnified to show additional detail. As seen in the figure, $H_{UTD}$ is connected through a series of switches to the DR which is in turn connected to the border router of the UTD domain. In our experiments, we used $H_{UO}$ as the pinging host and $H_{UTD}$ as the pinged host.

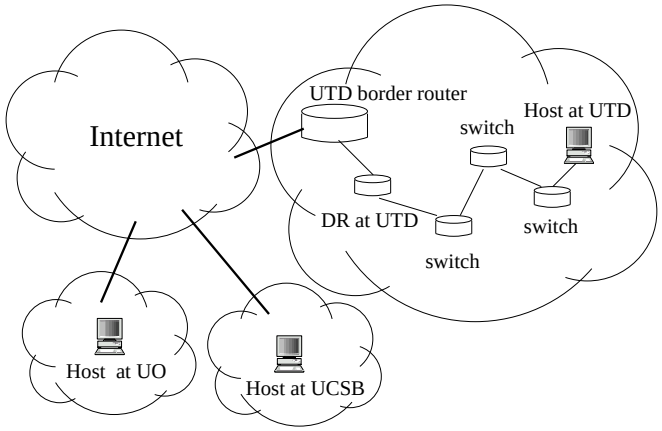
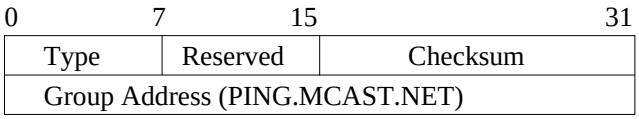



Fig. 2. Setup for the experiments.

During our experiments we observed that the ping responses sent by $H_{UTD}$ were not visible outside the UTD domain. After some investigation, we realized that this was due to a misconfigured switch between $H_{UTD}$ and its DR router at UTD. In fact, our *mcping* utility helped us debug and correct an existing multicast reachability problem in our own network. As a result, this experience demonstrates the need for a new IGMP message to convey incoming ping requests to a pinged system. Figure 3 gives the packet format for this new IGMP message type.

A second issue arises when there exists several simultaneous *mcping* requests sent towards an end system, S. The problem in this case is that not all *mcping* requests may reach all the way to S's subnet. When an *mcping* request (i.e., PIM-Join message) reaches a router which is already on the corresponding multicast tree, the router does not forward the join request but grafts the incoming path onto the existing multicast forwarding tree. Figure 4 presents an example scenario that visually explains this situation. In Figure 4-a, R1 sends an *mcping* request




Type: An 8-bit value to identify the payload as an mcping request (to be assigned by IANA)
Reserved: Not used
Checksum: 16-bit Internet checksum
Group Address: Includes an IP address from the SSM address range - (to be assigned by IANA)

Fig. 3. IGMP message format for *mcping* notification to the pinged source.

to a remote system, S. Routers on the path create forwarding state for the multicast channel (S, PING.MCAST.NET). In Figure 4-b, upon receiving the *mcping* request, S sends a ping response via multicast to (S, PING.MCAST.NET).

The problem occurs when another end system, R2, sends an *mcping* request to the remote system, S, as shown in Figure 4-c. On receiving this *mcping* request the router, X, will graft this new branch on the corresponding forwarding tree and will not forward the *mcping* request (i.e., PIM-Join message) further. This means that the *mcping* request of R2 will not reach S. In addition, since the ping reply originated from S (in response to R1's *mcping* request) already passed X, R2 will not receive a reply to its *mcping* request. Hence, it will mistakenly interpret this as a lack of multicast availability to the remote system, S.

We consider two solutions for this problem. In the first approach, we modify the PIM-Join mechanism such that routers will always forward *mcping* requests (i.e., PIM-Join messages for PING.MCAST.NET) toward their destination site. This would enable the pinged host to receive and respond to each *mcping* request. On the other hand, the main disadvantage of this approach is that the proposed modification to the PIM protocol requires updates to all the routers. In the second solution, we require the pinged end system to periodically send *mcping* responses on the PING.MCAST.NET multicast group address for some time. In this approach, as long as the DR router maintains forwarding state for the PING.MCAST.NET group, it will continue to forward the responses on the tree. The DR will continue to have forwarding state for the group as long as there are active remote pinging hosts. When the last pinging host (of several simultaneous pinging hosts) receives its response, it will leave the group and all forwarding state along the path will be removed. At that point, the DR at the pinged end system will also remove its forwarding state and will inform the host to stop generating responses. The advantage of this second solution is that it handles the problem without introducing any changes to the network.

### *B. Experiments with* Mcping

In order to demonstrate its utility and practicality, we have implemented *mcping* and conducted wide area measurements between three end systems on three different campuses: one at UC Santa Barbara (UCSB), one at the University of Oregon (UO), and one at UT Dallas (UTD). For the sake of presentation, we call these hosts $H_{UCSB}$, $H_{UO}$, and $H_{UTD}$. $H_{UCSB}$ and $H_{UO}$ are used as the pinging hosts and $H_{UTD}$ is used as the pinged host. From a reachability point-of-view, these experiments test multicast reachability between $H_{UCSB}$ and $H_{UO}$ as potential multicast receivers and $H_{UTD}$ as a potential multicast source.

**Experiment Setup.** In our setup, $H_{UCSB}$ and $H_{UTD}$ needed a few modifications to work correctly in our experiments. In the case of $H_{UCSB}$, the required functionality was to enable the system to initiate source specific join requests. In the case of $H_{UTD}$, the needed functionality was the ability to inform the host about an incoming *mcping* request.

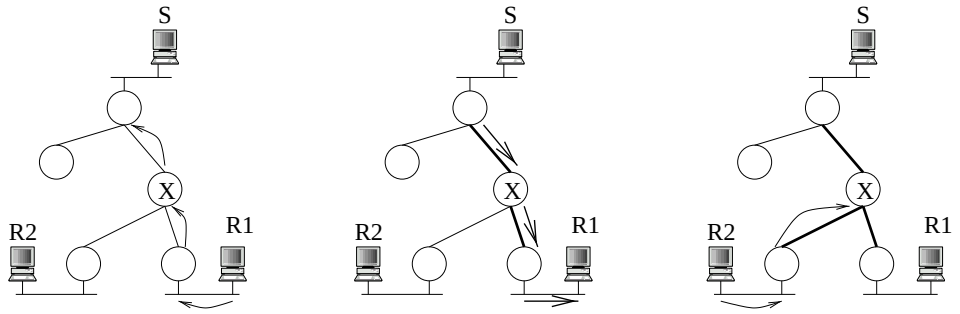
In order to implement the above modifications, we used an open source software router provided by the eXtensible Open Router Platform (XORP) [24] project. The aim of the XORP project is to develop an open source software router, flexible and extensible enough for research use. The XORP router provides support for PIM-SM and IGMP protocols. We extended XORP to implement the missing functionality both at $H_{UCSB}$ and $H_{UTD}$. At UCSB, we used the XORP software router to generate and send source specific PIM-Join messages. At UTD, we used the XORP software router as the DR for $H_{UTD}$.

***Mcping* Experiments.** After configuring the systems in our setup, we ran *mcping* queries between the pinging hosts ($H_{UCSB}$ and $H_{UO}$) and the pinged host ($H_{UTD}$) and verified multicast reachability between the sites. In addition, we conducted several different Round Trip Time (RTT) measurements among the sites. These are described below.

In the unicast world, in addition to verifying reachability, *ping* is used to measure RTTs to remote sites. A similar notion can now exist in the multicast world. In multicast, the delay between sending the join request and receiving the first packet can be called the *multicast* RTT (*mRTT*). The expectation is that *mRTT* will be higher than the corresponding unicast RTT (*uRTT*). As *mcping* requests propagate towards the queried host, routers on the direct path create multicast forwarding state for the *mcping* request. Since state construction requires additional local processing at the routers, this operation will introduce delay contributing to a larger *mRTT* value.

In addition to *mRTT*, one can also consider another RTT measurement between two end nodes. We call this second type of multicast RTT *mRTT2*. *mRTT2* is calculated as the round trip time on the path *after* the tree has been joined and the forwarding path established. As a result, *mRTT2* values are expected to be closer to *uRTT* values and are expected to be significantly smaller than *mRTT* values.

We ran experiments between our sites and recorded *uRTT*, *mRTT*, and mRTT2. The *uRTT* was measured using the standard unicast *ping* tool. The *mRTT* was measured from the time the receiver sent the *mcping* request to the time the *mcping* response was received. And finally, *mRTT2* was measured by




1. R1 sends an mcping query to S. (a)

2. S sends mcping response on PING.MCAST.NET. (b)

3. R2 sends an mcping query to S which terminates at router X. (c)

Fig. 4. An operational issue with *mcping*.

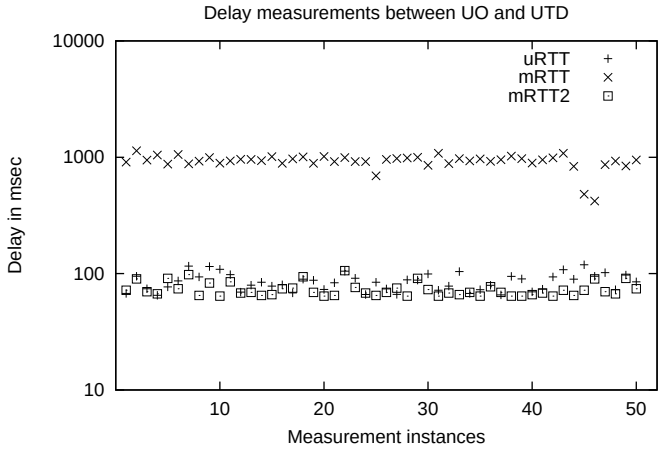



Fig. 5. Unicast and multicast RTT delays between UOregon and UTD.

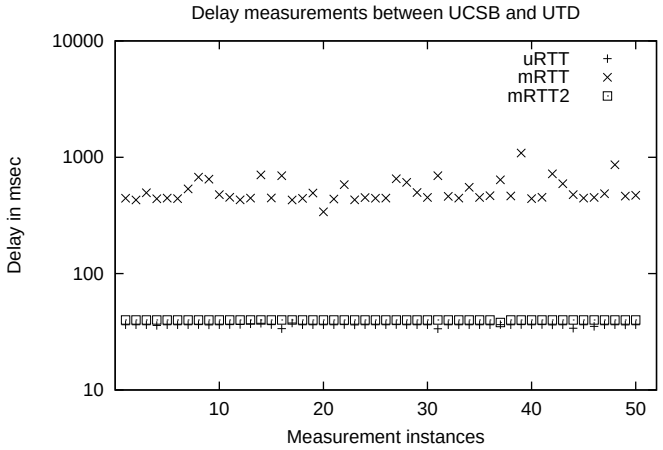



Fig. 6. Unicast and multicast RTT delays between UCSB and UTD.

first starting multicast traffic in both directions. We then sent a special multicast packet from $H_{UTD}$ representing a ping request and started a timer. On receiving this special multicast packet, $H_{UO}$ (or $H_{UCSB}$) immediately sent a response representing a ping reply. On receiving the reply, $H_{UTD}$ stopped the timer and returned the value as *mRTT2*.

We ran 50 measurements at regular intervals of 10 seconds between the three test sites. Figures 5 and 6 show the comparison between the *uRTT*, *mRTT*, and *mRTT2* values. The results support our earlier reasoning that *mRTT* is higher than *uRTT* and *mRTT2* is close to *uRTT*.

Based on the experiments, the average *uRTT* is 84ms, the *mRTT* is 953ms and *mRTT2* is 73ms between $H_{UTD}$ and $H_{UO}$. The average results for the experiments between $H_{UTD}$ and $H_{UCSB}$ are 40ms, 516ms and 40ms for *uRTT*, *mRTT*, *mRTT2* respectively. According to the path traces, both unicast and multicast paths are almost the same with an end-to-end hop count of 9 for the $H_{UTD}$ to $H_{UO}$ path and 10 for the $H_{UTD}$ to $H_{UCSB}$ path. If we assume that the propagation delay on the links to be same for both *uRTT* and *mRTT*, the difference between the two delays (i.e. 953 - 84 = 869ms for UO and 516 - 40 = 476ms for UCSB) gives us a good approximation for the total delay incurred by the routers for processing the join message. Hence, on an average 96.5ms (869ms/9) and 47.6ms (476ms/10) of processing time is introduced by each router for the paths to UO and UCSB hosts respectively.

## IV. *Mcroute*: A Multicast Route Discovery Tool

In this section, we propose a multicast route discovery tool that we call *mcroute*. *Mcroute* is analogous to the unicast *traceroute* utility. It differs from *mtrace* in a semantically small but operationally important way. Consider an *mtrace* query from R to S where both R and S are end systems in multicast-enabled networks. A successful *mtrace* from R to S has so far been interpreted as the proof of multicast connectivity between R and S. However, since *mtrace* does not use the PIM-Join mechanism but uses an explicit *mtrace* processing module, the success of an *mtrace* run does not necessarily indicates connectivity. In addition, even if there exists a multicast join path from R to S, it does not necessarily imply that multicast data from S can successfully reach R. In fact, during our

*sdr-monitor* project, we encountered numerous cases where *mtrace* was returning a successful path between a multicast receiver and a multicast source site while our application layer monitoring information was indicating that reachability not existed [14].

Our goal in this section is to develop a route discovery mechanism (*mcroute*) that uses the underlying PIM-Join mechanism for query request propagation and uses the underlying multicast packet forwarding mechanism for query response propagation. This way we can safely use the result of a successful *mcroute* query as the proof of multicast reachability between the end points.

*Mcroute* uses a dedicated multicast group address, MCROUTE.MCAST.NET in the source specific multicast address range (232/8). *Mcroute* is run from a receiver, R, toward a source, S, to collect the multicast path information in between. The receiver, R, causes its DR to issue a PIM-Join(S, MCROUTE.MCAST.NET) sent towards S. This is visually represented in Figure 7-a.

Next, *mcroute* uses a TTL-based approach to cause each router on the R-to-S multicast path to send *mcroute* responses on the (S, MCROUTE.MCAST.NET) multicast channel. The first *mcroute* request is sent with an initial TTL value of 1 and the TTL is incremented each round thereafter (see Figure 7-b). When the TTL reaches 0 at an on-tree router, the router sends an *mcroute* response to (S, MCROUTE.MCAST.NET), spoofing the IP address of the source, S. The router includes its own IP address in a protocol field in the response message. Source spoofing allows the *mcroute* response to propagate on the existing multicast forwarding path toward the receiver. When the receiver receives an *mcroute* response originating from an on-tree router, X, this is interpreted as there being reachability between the router X and the receiver. If the receiver stops receiving *mcroute* responses before reaching the source, S, but after an on-tree router, Y, it indicates that there is a potential reachability problem at or after Y. As a result, we argue that *mcroute* can be used to accurately detect and locate multicast reachability problems between a source and a receiver.

### *A. Required Modifications for* Mcroute

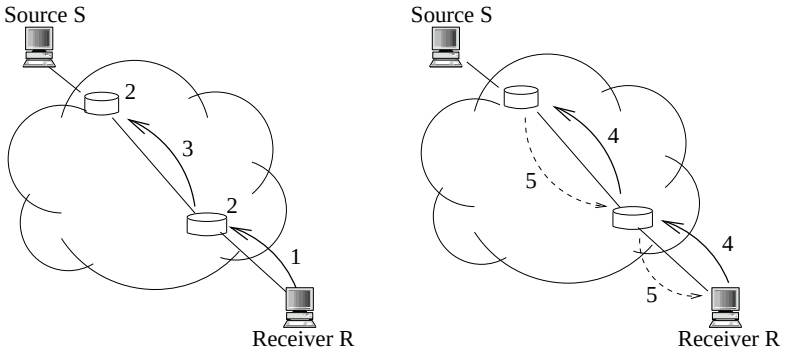
Compared to unicast *traceroute*, tracing a path in multicast requires additional support from the routers. In unicast, routers use ICMP error messages to inform the source of the packet causing the error condition. This is leveraged to provide *traceroute* functionality without requiring additional support from the routers.

In multicast, in order to avoid implosion at the source site, routers do not send ICMP error messages for error conditions caused by multicast packets. This prevents us from using an approach similar to unicast for multicast route tracing. Currently, *mtrace* functionality is specified in an Internet Engineering Task Force (IETF) Internet Draft and is implemented by several router vendors to support multicast path discovery. Similar to *mtrace*, our *mcroute* utility requires additional functionality in the routers. Since *mtrace* is not yet an IETF standard, *mcroute* functionality can be combined with or can replace the to-be-standardized *mtrace* functionality. In this case, routers will need to support processing of two new IGMP message types: *Mcroute-Request* and *Mcroute-Response* as shown in Figure 8 and discussed below.

When an end user on a host, R, wants to discover multicast route to a remote host, S, he will use the *mcroute* utility to create and send an *Mcroute-Request* message to its edge router. In this message, the tool will set InitTTL = TTL = 1 and will put its own IP address and the IP address of the remote source in the corresponding fields as shown in Figure 8. Then, the tool will assign a sequence number to the Query-Sequence-Number (QSN) field and send the request packet to the edge router. This request will cause the edge router to create and send an *Mcroute-Response* message to the (S,MCROUTE.MCAST.NET) address. When the router responds, it will copy the QSN, InitTTL and Querier IP Address values from the request packet. It will also include its own IP address in the Router-IP-Address field of the IGMP header.

On receiving the response, R will create and forward a new *Mcroute-Request* by incrementing InitTTL, TTL, and QSN by one and will send a new message to the edge router. The edge router, upon receiving the packet, will decrement the TTL value by one and forward it to its upstream router on the RPF path toward the queried remote host, S. Because a PIM-Join(S,MCROUTE.MCAST.NET) message was already sent, the router already has PIM forwarding state for (S,MCROUTE.MCAST.NET) and it will use the RPF interface information from this entry to forward the *Mcroute-Request* toward S. When the second hop router receives this request message, it will decrement the TTL value, which will now be 0, and will generate a response back to the querier, R, on (S,MCROUTE.MCAST.NET). The *Mcroute-Request* packets are forwarded hop-by-hop. In other words, during the *Mcroute* route discovery, we are not using IP TTL values to cause routers to create and send a response. Instead we use the TTL value of the IGMP *Mcroute-Request* header to cause the routers to send responses. By forwarding the request packets hop-by-hop, we enable each on-tree router to treat the query packet as a control packet and act on it. This is also consistent with the way PIM-Join messages propagate in the network from a joining receiver site to a remote source site.

Finally, from the end hosts point-of-view, *mcroute* does not require any modifications to the operating system of the end systems. In order to discover a multicast path, an end system will use an *mcroute* query tool. The tool will implement all the required functionality including creating and sending *Mcroute-Request* messages and receiving and processing *Mcroute-Response* messages. Since these messages will be created by the end system, the tool will need root privileges to construct and communicate the messages over raw sockets. On the other hand, the *mcroute* query can be stopped by the first hop router at the queried remote source site. This way, the queried end system will be relieved from the burden of implementing *Mcroute-Request* and *Mcroute-Response* functionality. As a




1. IGMP Receiver Report for (S, MCROUTE.MCAST.NET).
2. Router creates forwarding state for (S, MCROUTE.MCAST.NET).
3. Router forwards PIM-Join for (S, MCROUTE.MCAST.NET).
4. Receiver sends a mcroute probe.
5. Router sends response on (S, MCROUTE.MCAST.NET) when TTL = 0.

Fig. 7. Operation of the *mcroute* tool.

IGMP Mcroute Request Message

| 0 | 7 | 15 | 31 |
|---|---|---|---|
| Type | Reserved | Checksum | |
| IP addr of queried remote host (S) | | | |
| IP addr of querier host (R) | | | |
| Init. TTL | TTL | Query Seq No (QSN) | |

Type: An 8-bit value to identify the payload as an mcroute request/response (to be assigned by IANA)

Reserved: Not used
Checksum: 16-bit Internet checksum
S: 32-bit IP address of queried remote host

IGMP Mcroute Response Message

| 0 | 7 | 15 | 31 |
|---|---|---|---|
| Type | Reserved | Checksum | |
| IP addr of the router sending the response | | | |
| IP addr of querier host (R) | | | |
| Init. TTL | Reserved | Query Seq No (QSN) | |

R: 32-bit IP address of querier host
Init TTL: The initial TTL value set by R
TTL: TTL value to be decremented by each router
QSN: Query seq. no. assigned by querier R

Fig. 8. IGMP Mcroute-Request and Mcroute-Response messages.

result, the proposed *mcroute*-based multicast route discovery can be included in the network without needing modifications to end systems.

### *B. Using* Mcroute *to Classify Multicast Reachability Problems*

In this section we present a three-step procedure to classify multicast reachability problems and show how *mcroute* could support this procedure. In a typical multicast application scenario, a receiver joins the multicast group address and expects to start receiving packets from active sources. Assume that a receiver, R, wants to join a multicast group, G. For this, R uses IGMP to inform its DR about its request to join the group, G. If there are no problems, R will soon start receiving multicast packets from the network. On the other hand, if R does not receive any data, there are two possibilities: (1) there are no active sources sending to the multicast group, G, or (2) there is a source, S, who is sending to (S,G) but due to reachability problems, R cannot receive these packets. In our work, we are mainly interested in the second scenario. At this point, we enumerate the potential problems as being one of the following:

1) **Multicast connectivity problems:** One possibility is that the routers in the receiver's local domain do not have a multicast route to the domain in which the source, S, resides. Since multicast route availability is communicated using Multi-Protocol BGP (MBGP) [25], not having a route would be the result of a problem with MBGP and route advertisement.
2) **Source discovery problems:** Another possibility is that the RP in the receiver's domain does not have any information about S being an active source for the group, G. Since the Multicast Source Discovery Protocol (MSDP) [26] is the protocol used to communicate local source availability with remote domains, a lack of source information would indicate a problem with MSDP.
3) **Multicast join and/or data forwarding problems:** A third possibility is that the local domain has a multicast route to S, but R still does not receive any packets. This situation may be related to problems in forwarding PIM-Join messages toward the source, S, or it may be related to forwarding errors for the actual (S,G) data packets enroute to R's site. In order to distinguish this case from the previous two cases, we call these errors "multicast

join" and "forwarding errors" and identify PIM-SM as the source of the problem.

We are interested in classifying multicast reachability problems into one of the three groups above. Our aim is to identify the multicast routing protocols that are responsible. The procedure that we use to classify multicast problems into different groups is as follows. The typical scenario is that there is an active source, S, that is currently sending to a multicast group, G. A remote receiver, R, wants to join the group, G, and receive (S,G) data. We follow the steps describe below in order to classify potential problems:

1) First R sends a PIM-Join(*,G) toward the domain RP. After this, if R starts receiving (S,G) data, it means that there are no reachability problems and S's multicast data can reach R's site.
2) If the receiver, R, cannot receive any data, it sends a PIM-Join(S,G) toward the source site, S. If the receiver, R, starts receiving (S,G) data, this means that there exists a multicast route to S's site and the new join process was successful. Given the fact that the (*,G) join failed and the (S,G) join succeeded, we conclude that there is a potential MSDP problem between the two domains. In this scenario, the failure to receive (S,G) data after the initial (*,G) join indicates that the RP at R's site did not know about S as being an active source for the multicast group, G.
3) On the other hand, if the receiver, R, cannot receive (S,G) data even after joining the source specific group of S, the first and third options still remain as the likely cause of the problem. At this point, *mcroute* can be used to detect as well as locate the problem.

### C. A Case Study on Reachability Problem Classification

In this section, we present a case study on the usefulness of *mcroute* for classifying multicast reachability problems. For this, we use the existing *multicast beacon* [15] system. At the time of our experiments there were approximately 40 participants in the *multicast beacon group*. These participants were actively sending test multicast messages to the group address 233.4.200.21:10002. The multicast beacon web site continuously presented reachability information for these sites. In our case, we used the web site to identify the active participants and used our problem classification procedure to identify reachability between these sites as multicast senders and three other sites (UCSB, UTD, and UO) as multicast receivers.

As a summary, our approach includes three steps: issuing a (*,G) join, issuing an (S,G) join, and using our new *mcroute* tool. Due to the fact that *mcroute* is being introduced in this paper and is not deployed, our experiments have mostly depended on the first two steps. However, for the case of the UTD receiver, we have used support from the local RP in the form of MBGP and MSDP routing table information for the *multicast beacon* sources. We have used the MSDP information to verify the correctness of our two-step approach and have used MBGP information as a placeholder for *mcroute* in the third step of the procedure.

Table I lists the IP addresses of the *multicast beacon* sources that we used in our experiments. From each of our three receiver sites, we first issued (*,G) joins for the *multicast beacon* group and then issued (S,G) joins for each of the active sources listed in the table. In the table, a "X" indicates that the receiver was able to receive multicast data from the corresponding source after a (*,G) join as well as after an (S,G) join. As a result, these cases corresponds to properly working multicast between the source and the receivers. A "-" mark indicates that the receiver received no data from the corresponding source after (*,G) and (S,G) joins. At this point, without any further information, we cannot really know if this is a connectivity problem or other problem. Finally, a "z" indicates that the receiver did not receive multicast data from the corresponding source after a (*,G) join but did receive it after an (S,G) join. This case suggests that there exists a reachability problem between the source and the receiver, and this problem is most likely an MSDP problem. We now elaborate on this case.

When a (*,G) join does not result in data reception but an (S,G) join does, there are several possible reasons: (1) a PIM-SM problem between the DR at the receiver site and its RP caused the (*,G) join to fail before reaching the RP, (2) a PIM-SM problem between the RP and the source, S, caused the (S,G) join issued by the RP to fail before it reached S, or (3) an MSDP problem caused by the local RP not knowing about the source and therefore not knowing about (S,G).

Consider the topology in Figure 9 where we explicitly mark important segments of the paths from Path1 to Path4. The first possible reason given above can be shown not to be possible as follows. During our experiments, we used over 40 sources. For the majority of these sources, the (*,G) joins returned multicast data. This suggests that there does not exist any PIM-SM problem on Path1. Similarly, the second reason above can be shown not to be possible as follows. Following from the first case, the fact that (*,G) joins returned multicast data from the majority of the sources suggests that there does not exist a PIM-SM problem on Path2. In addition, the fact that an (S,G) join returned multicast data from the source, S, suggests that there does not exist a PIM-SM problem on Path3. Hence, we conclude that the problem is an MSDP problem.

We verified the correctness of our reasoning by using the only case in the UTD experiments in which the source 216.239.127.230 was not reachable via a (*,G) join but was reachable via an (S,G) join. We verified that the local RP at UTD did not have an MSDP cache entry for this source as an active source for the *multicast beacon* group address.

Finally, since *mcroute* is not currently available, we consulted the MBGP routing table at the RP at UTD to categorize problems as either MBGP or PIM-SM related problems. Table II shows the final classification for the UTD receiver. The listed sources are from Table I which failed on (*,G) joins for the UTD receiver. We see five instances of failures due to MBGP and eight instances related to incorrect operation of

| Source | UTD | UCSB | UO | Source | UTD | UCSB | UO | Source | UTD | UCSB | UO |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 129.78.157.172 | - | - | - | 129.128.125.62 | X | X | X | 206.167.204.18 | - | - | - |
| 132.246.2.20 | X | X | X | 204.174.103.32 | - | - | - | 132.246.130.26 | - | - | - |
| 142.55.1.205 | - | - | z | 129.128.25.72 | X | X | X | 129.128.25.181 | X | X | X |
| 216.239.127.230 | z | z | z | 63.105.122.14 | - | - | - | 192.108.35.16 | X | - | - |
| 128.123.3.74 | X | X | X | 128.118.146.51 | X | X | X | 128.118.146.52 | X | X | X |
| 128.118.57.33 | X | X | X | 130.160.4.113 | - | - | - | 128.111.252.50 | X | X | X |
| 128.111.2.2 | X | X | X | 137.110.147.70 | X | X | X | 132.239.253.141 | X | X | - |
| 128.227.212.96 | X | X | X | 131.193.77.102 | X | X | X | 141.142.2.168 | X | X | X |
| 141.142.64.5 | X | X | X | 128.223.157.25 | X | X | X | 192.236.37.104 | X | X | X |
| 155.101.3.111 | X | X | X | 128.83.6.240 | X | X | X | 160.36.188.124 | - | - | - |
| 198.82.169.70 | X | X | X | 198.82.169.72 | X | X | X | 130.215.32.94 | X | X | X |
| 130.215.5.21 | - | - | - | 130.215.201.81 | X | X | X | 193.166.3.92 | X | - | - |
| 192.31.96.42 | X | X | X | 203.181.248.186 | - | - | - | 203.181.249.74 | - | - | - |
| 205.189.33.130 | - | - | - | 138.18.250.6 | X | X | z | 194.80.35.36 | X | - | - |
| 195.194.24.19 | - | X | X | | | | | | | | |

TABLE I

RECEPTION OF (*,G) JOINS AT DIFFERENT MULTICAST BEACON SITES.

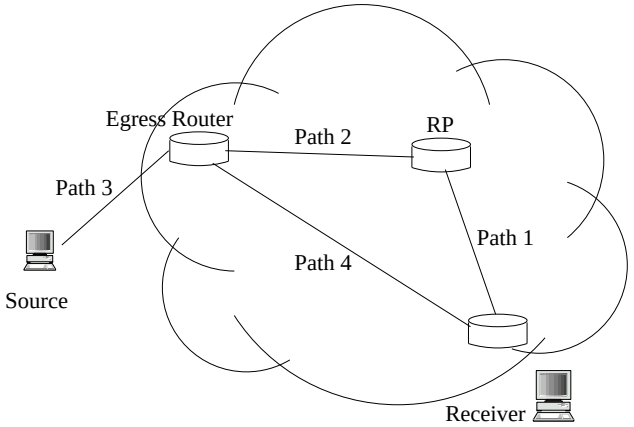



Fig. 9. (*,G) and (S,G) join paths.

| Source | MSDP | MBGP | PIM-SM |
|---|---|---|---|
| 129.78.157.172 | | X | |
| 206.167.204.18 | | X | |
| 204.174.103.32 | | X | |
| 132.246.130.26 | | | X |
| 142.55.1.205 | | | X |
| 63.105.122.14 | | X | |
| 130.160.4.113 | | X | |
| 160.36.188.124 | | | X |
| 130.215.5.21 | | | X |
| 203.181.248.186 | | | X |
| 203.181.249.74 | | | X |
| 205.189.33.130 | | | X |
| 194.80.35.36 | | | X |
| 216.239.127.230 | X | | |

TABLE II

CLASSIFICATION OF PROBLEMS AT UTD

PIM-SM. The latter cases are either due to protocol problems or configuration problems as categorized in the previous subsection. After cross checking the sites having PIM-SM problems from the *multicast beacon* web site [15], we see that two of these sites (130.215.5.21 and 194.80.35.36) have local connectivity problems and others seem to have reachability to only some *multicast beacon* participants beyond their own network.

## V. DEPLOYMENT AND SECURITY ISSUES

From a deployment point-of-view, both *mcping* and *mcroute* require some modifications to existing multicast protocols. *Mcping* requires both DRs as well as end systems to support the proposed IGMP extensions. In addition, *mcroute* requires routers to include explicit support for it to operation.

From an incremental deployment point-of-view, *mcping* requires edge routers and end hosts to be modified to generate *Mcping-Responses*. The required modification to edge routers necessitates an upgrade to the operating system of the router to a new version that supports *mcping*. No hardware updates are required. In addition, during the initial deployment, edge routers can create and send *Mcping-Responses* on behalf of end systems until the end system operating systems are updated to support the service. On the other hand, *mcroute* requires software updates on the routers, but no modifications to end systems. The service requires support from all the routers on the path. Finally, both tools resemble their unicast counterparts and both can send queries multiple times to account for potential packet losses in the network. Persistent packet loss indicates the existence of either multicast problems or at least significant congestion.

From a security point of view, the proposed utilities do not introduce any new security weaknesses for their users or the multicast infrastructure. In unicast, adversaries can use *ping* to launch reflector-based denial of service attacks [27] on third party sites. Due to the multicast forwarding mechanism, *mcping* requests and responses follow the same path. Hence, *mcping* cannot be used for third party denial of service attacks.

Another possibility for attacking a source, S, may be in the form of causing the source, S, to send redundant packets to arbitrary multicast channels. That is, an adversary may attempt to send join messages to a number of different multicast channels for a source, S, expecting the designated router at

S's site to deliver these joins to S (using the proposed new message in IGMP) . However, the designated routers will only send *mcping* request packets to the source, S, for the join messages coming to the PING.MCAST.NET multicast address. Therefore, adversaries cannot use such an approach to cause remote sources to misbehave.

Finally, a malicious end user can send a large volume of *mcroute* requests to keep routers busy processing these requests. Similar attacks are possible for *mtrace* and *mcroute* does not aggravate the attack. These attacks are prevented by limiting the rate of such queries coming from the same end system and/or the subnet by the routers.

## VI. Conclusions

In this paper, we have examined the availability and quality of monitoring and measurement support for IP multicast. First, we have considered *mping* and *mtrace* and shown that they cannot really help with basic diagnostic tasks such as verifying the availability of IP multicast between two remote systems (i.e., the multicast equivalent of the unicast *ping* service) and locating problem spots on an end-to-end multicast path (i.e., the multicast equivalent of the unicast *traceroute* service). We have therefore proposed the multicast equivalents of *ping* and *traceroute*, called *mcping* and *mcroute*. *Mcping* and *mcroute* are developed as two basic network diagnostic utilities for conducting various types of multicast monitoring and measurement studies. In order to demonstrate their utility, we have presented two different usage scenarios. In the first scenario, we used *mcping* to verify end-to-end multicast service availability among remote end systems. In the second scenario, we presented a methodology to classify IP multicast problems into three groups and showed the utility of *mcroute* in this context.

In addition to their utility, we have also shown that the tools require fairly small updates to the IGMP protocol and require a moderate modification to the *mtrace* processing modules of multicast-enabled routers. In return, they enable multicast service providers and their users to increase their effectiveness in monitoring and measuring multicast service characteristics in the Internet.

## Acknowledgment

We thank Joel Jaggli and Hans Kuhn of the University of Oregon for granting us access to their equipment for our experiments. We thank Paul Conally and Steve Hodo of the University of Texas at Dallas for helping us debug and configure the network. Finally, we thank Pavlin Radoslavov for his support with XORP modifications.

## 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] K. Almeroth, "The evolution of multicast: From the MBone to interdomain multicast to Internet2 deployment," *IEEE Network*, vol. 14, no. 1, pp. 10–20, January/February 2000.

[3] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, no. 1, pp. 78–88, January/February 2000.

[4] S. Paul, K.K. Sabnani, J.C. Lin, and S. Bhattacharyya, "Reliable multicast transport protocol (RMTP)," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 407–421, April 1997.

[5] P. Judge and M. Ammar, "Gothic: A group access control architecture for secure multicast and anycast," in *Proceedigns of IEEE INFOCOM*, New York City, NY, USA, June 2002, pp. 1547–1556.

[6] C. Shields and J.J. Garcia-Luna-Aveces, "KHIP - a scalable protocol for secure multicast routing," in *Proceedings of ACM SIGCOMM*, Cambridge, MA, USA, August 1999, pp. 53–64.

[7] S. Jagannathan, K. Almeroth, and A. Acharya, "Topology sesitive congestion control for real-time multicast," in *Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Chapel Hill, NC, USA, June 2000, pp. 41–50.

[8] P. Rajvaidya and K. Almeroth, "Analysis of routing characteristics in the multicast infrastructure," in *IEEE Infocom*, San Francisco, CA, USA, April 2003, pp. 1532–1542.

[9] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proceedings of ACM SIGCOMM*, Vancouver, BC, CANADA, September 1998, pp. 56–67.

[10] S. Bhattacharyya, "An overview of source-specifi c multicast (SSM)," Internet Engineering Task Force (IETF), RFC 3569, July 2003.

[11] W. Fenner, "A 'traceroute' facility for IP multicast," Internet Engineering Task Force (IETF), draft-fenner-traceroute-ipm-*.txt, December 2003, Work in progress.

[12] A. Swan, D. Bacher, and L. Rowe, *rtpmon 1.0a7*, University of California at Berkeley, January 1997, Available from ftp://mm-ftp.cs.berkeley.edu/pub/rtpmon/.

[13] D. Makofske and K. Almeroth, "Real-time multicast tree visualization and monitoring," *Software–Practice & Experience*, vol. 30, no. 9, pp. 1047–1065, July 2000.

[14] K. Sarac and K. Almeroth, "Monitoring reachability in the global multicast infrastructure," in *International Conference on Network Protocols (ICNP)*, Osaka, JAPAN, November 2000, pp. 141–150.

[15] NLANR, *Multicast Beacon*, National Laboratory for Applied Network Research, June 2000, Available from http://dast.nlanr.net/Projects/Beacon/.

[16] H. Schulzrinne, S. Casner, R. Frederick, and Jacobson V., "RTP: A transport protocol for real-time applications," Internet Engineering Task Force (IETF), RFC 1889, January 1996.

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

[18] K. Almeroth, K. Sarac, and L. Wei, "The multicast reachability monitor (mrm) protocol: An instantiation of a multicast management architecture," Tech. Rep., University of California–Santa Barbara, September 1998.

[19] E. Al-Shaer and Y. Tang, "SMRM: SNMP-based multicast reachability monitoring," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Florence, ITALY, April 2002, pp. 467–482.

[20] P. Sharma, E. Perry, and R. Malpani, "IP multicast operational network management: design, challenges, and experiences," *IEEE Network*, vol. 17, no. 2, pp. 49–55, March/April 2003.

[21] E. Al-Shaer and Y Tang, "Mrmon: Multicast remote monitoring," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Seoul, KOREA, April 2004, pp. 585–598.

[22] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan, "Internet group management protocol, version 3," Internet Engineering Task Force (IETF), RFC 3376, October 2002.

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

[24] *eXtensible Open Router Platform (XORP) Project*, http://www.xorp.org/.

[25] T. Bates, R. Chandra, D. Katz, and Y. Rekhter, "Multiprotocol extensions for BGP-4," Internet Engineering Task Force (IETF), RFC 2283, February 1998.

[26] D. Meyer and B. Fenner, "Multicast source discovery protocol (MSDP)," Internet Engineering Task Force (IETF), RFC 3618, October 2003.

[27] J. Mirkovic and P. Reiher, "A taxonomy of ddos attacks and defense mechanisms," *ACM SIGCOMM Computer Communications Review*, vol. 34, no. 2, pp. 39–54, April 2004.