Enabling End-User Network Monitoring via the Multicast Consolidated Proxy Monitor

Anshuman Kanwar
Elect. and Computer Engr.
Univ of California
Santa Barbara, CA
akanwar@cs.ucsb.edu

Kevin C. Almeroth
Computer Science
Univ of California
Santa Barbara, CA
almeroth@cs.ucsb.edu

Supratik Bhattacharyya
Sprint ATL
One Adrian Court
Burlingame, CA
supratik@sprintlabs.com

Matthew Davy
Univ Info Tech Services
Indiana University
Bloomington, IN
mpd@iu.edu

ABSTRACT

The debugging of problems in IP multicast networks relies heavily on an eclectic set of stand-alone tools. These tools traditionally neither provide a consistent interface nor do they generate readily interpretable results. We propose the "Multicast Consolidated Proxy Monitor" (MCPM), an integrated system for collecting, analyzing and presenting multicast monitoring results to both the end user and the network operator at the user's Internet Service Provider (ISP). The MCPM accesses network state information not normally visible to end users and acts as a proxy for disseminating this information. Functionally, through this architecture, we aim to a) provide a view of the multicast network at varying levels of granularity, b) provide end users with a limited ability to query the multicast infrastructure in real time, and c) protect the infrastructure from overwhelming amount of monitoring load through load control. Operationally, our scheme allows scaling to the ISPs dimensions, adaptability to new protocols (introduced as multicast evolves), threshold detection for crucial parameters and an access controlled, customizable interface design. Although the multicast scenario is used to illustrate the benefits of consolidated monitoring, the ultimate aim is to scale the scheme to unicast IP networks.

Keywords: multicast monitoring, end-user network view, consolidated monitoring

1. INTRODUCTION

The Internet does not provide for efficient monitoring and troubleshooting of multicast traffic. In fact, monitoring any aspect of a system as dynamic and distributed as the Internet requires considerable forethought and planning. Although many stand-alone tools are available for diagnosing multicast connectivity problems, they are not designed with the end user in mind. It is unfortunate that despite the availability of the required infrastructure and the obvious benefit of saving on bandwidth, there exists an inexplicable gap between the maturity of IP multicast protocols and their field deployment. It is hoped that making the task of multicast troubleshooting simple – to the point where even the end user can participate – will help in eliminating this gap.

Since its first deployment in 1992, IP multicast has transitioned from a virtual overlay, DVMRP-based, flat network into one driven by a mix of several fairly complicated protocols. The current Internet multicast infrastructure runs the following protocols: PIM-SM for both inter- and intra-domain multicast routing; MBGP for policy-based route exchange; MSDP for active source discovery, and IGMP for group management. DVMRP and PIM-DM, which should have been phased out, still persist in some networks for intra-domain routing [1]. In addition, ICMP and SNMP play important roles in conveying information about the status of the network [2].

A plethora of problems arise due to the interplay of the various multicast protocols and consequently there exists a wide variation in the kinds of problems faced by end-users and Internet Service Providers. One of the most frequently encountered problems is when a user tries to join a multicast group and sees no traffic. The first issue is that there may not be a problem at all. The source(s) might simply not be transmitting, but this fact is not easy to determine. Second, if there is a problem, the number of potential causes is large. For instance, the inability to receive traffic can be due to problems in joining a group, congestion at a proximal router or perhaps a Network Address Translator (NAT) on the upstream link. A more serious problem might be the presence of a non-pruning router in the multicast path. In the most extreme case, an ISP itself might face serious problems with congestion or security issues arising due to a newly deployed multicast protocol or malicious applications. Any of these problems might be caused by a bug in protocol design, router configuration error or incorrect deployment.

Though present debugging tools offer insights into many of the above problems, they provide information on very specific aspects of a multicast system. To the personnel in the Network Operations Center (NOC) investigating

multicast outages these tools yield pieces of a jigsaw puzzle. It then falls on the network engineer to use his experience with these pieces and assemble the complete picture of what happened or might still be happening. If a glitch is non-persistent in nature, it is possible that network conditions at the time of the discrepancy are not recorded anywhere. Hence the task of reconstruction and inference is both time consuming and difficult.

A consolidation mechanism is required therefore, to assemble the area-specific knowledge created by existing or future tools into an aggregate picture. The aim is to help expedite troubleshooting and error detection. Such a system must be able to identify a set of important parameters that represent a reasonable view of the network and be able to automatically capture this snapshot as and when an error is flagged.

In this paper we define the need for, and outline the architecture of a consolidated multicast monitoring system. Our system is called the Multicast Consolidated Proxy Monitor (MCPM). In its quiescent state, MCPM collects statistics and allows user queries about network status. In case an error is detected, it launches a scenario-specific sequence of "probe-agents" and consolidates the generated results. The dual purpose of this architecture is to help the end user troubleshoot insignificant problems, as well as, to allow the support staff to gather instance relevant data for easier troubleshooting. We aim for flexibility, robustness and adaptability by keeping protocol specific "probe-agents" distinct from the data aggregation and presentation aspects of the consolidated monitor. The proposed monitor is located at an ISP level and services customers and the Network Operations Center (NOC). Here MCPM has access to the routing tables and the state of various network layer elements. It can communicate with the infrastructure using Simple Network Management Protocol (SNMP) or using other problem-specific tools. The information generated in this way is a) provided to the end-user (or network operator) in response to specific queries, b) used by the system to watch for any error conditions, and c) used to monitor real time media streams and report the loss of adequate QoS.

Another issue we strive to address is providing a differentiated service scheme. By this we mean that users are provided with varying levels of access based on a user profile. We illustrate this by the following example. Access to resources is granted via group-based permissions. The *end-user* group for example has the privilege to interact with MCPM through a web interface. This group can issue limited queries to determine only the basic information related to network status. For example a query can be "I cannot watch session x. What might be wrong?". The system in response to this high level query might return that the network is operating correctly or might return the existence of a problem and notification that a network operator has been informed. The network operator on the other hand, has a more detailed view of the network and can, in the same situation, invoke another tool from within MCPM to gather more detailed information. MCPM would present a verbatim output of the tool along with an analysis of what could be wrong. Hence, different classes of users are presented with different levels of query capabilities and reporting detail even though the underlying mechanisms used remain the same.

Another feature of MCPM is that one instance of this system running on one ISP can be interconnected (peered) with other such systems to provide a distributed error detection and flagging system. We do no attempt to build a system which captures the global state of multicast but rather depend on the localized detection of faults and pass this information to adjacent peers. The distributed capabilities of MCPM are designed to be functionally similar to those of Globally Distributed Troubleshooting (GDT) [3]. Our approach helps with scalability, simplifies deployment, and maintains security across domain boundaries .

The rest of this paper is organized as follows. We justify the need for a system like MCPM in section 2 and review related work in section 3. In section 4 we discuss the targeted design goals. Section 5 is devoted to architecture issues. We conclude in section 6.

2. JUSTIFICATION

Troubleshooting network glitches even in the current infrastructure is largely a matter of following a pre-defined set of instructions, running a sequence of tools, and inferring from the generated results. A related Internet draft [4] outlines numerous routine sequences to be followed in the case of multicast problems. In this section we briefly describe a typical traffic debugging scenario.

Consider congestion and rate limiting problems that result in high packet loss with subsequent loss of session announcements and a decrease in the quality of audio and video. These problems may first be investigated by using *RTPmon* to check for receivers experiencing packet loss. Second, an *mtrace* from the source to the problematic might be performed in order to locate the problematic hops. This might be followed by an *mtrace* in the opposite

direction to help distinguish whether the problem is with the router or the link at that hop. If the reverse *mtrace* shows similar loss at the hop adjacent to the lossy hop in the forward *mtrace*, odds are that the intermediate router is overloaded. *mrinfo* can be used to check the fanout on the router. Similarly if the reverse *mtrace* shows no problems near that hop, (indicating that loss is one-way), then the router on the upstream end of the link can be checked with "*mstat -nv*". The results of this query will show if a rate-limit is set on the link, and if the link traffic is near that limit. If the reverse *mtrace* shows loss over the same link, the problem is likely to be link congestion. This process of gathering information – or trying to alternate tools in the case of failure of one – can continue almost indefinitely.

From the above example it can be seen that multicast troubleshooting is largely a matter of experience – both in using a number of different tools and interpreting the results. Once this experience has been distilled, it can be loosely used to form a set of rules that define a plan of action. In real life, network maintenance personnel then simply go through this plan of action every time a problem arises. The choice of which plan to follow is based on observed network behavior.

Clearly, much of this work can be automated. A system is needed that can execute a plan of action in case of a failure and build easily understandable statistics which are presented to the operations personnel. This system might have some expert-system like features to suggest solutions and fixes as well. In case the problem is first encountered by an end-user who is assumed to have no technical knowledge, the system should confirm (or alleviate) the user's suspicion of a problem (or the lack thereof). If there is a problem, the bulk of debugging activity should be left to the support personnel. During any of the tests performed by the end user, the level of detail presented should be palatable to the typical end-user. In some cases a typical user might be given additional levels of access. Having many users periodically inspecting and questioning a networks' robustness should be seen as a "good thing" if the monitoring load does not adversely affect the network operation.

3. PREVIOUS WORK

Thaler and Adoba [4] separate the expected problems in a multicast network into three classes: session announcement problems, reception problems (at the end host), or multicast router problems. While the last problem has been the focus of additional work [5], not much has been said about the former two. We believe these two classes of problems, both of which have to do with data delivery, have been frequently over looked.

Multicast traffic is routed on the basis of a dynamic distribution tree. The combination of multiple hosts into a single logical group makes multicast inherently more difficult to observe than unicast. IP multicast also offers interesting challenges because of its rapidly evolving nature: evolving in terms of group membership, protocol deployment, as well as the underlying infrastructure. Since multicast runs under the assumption that unicast communication is operating correctly in the network, healthy multicast, to a large extent, implies healthy unicast. Arguably scalable methods developed for monitoring mulicast traffic can be extended to the unicast model. On the other hand, the extension of existing unicast tools to multicast has been difficult [6,2].

Traditionally, a number of tools have been used for troubleshooting multicast. *mtrace* [7,8] the multicast equivalent of traceroute, is one of the the oldest monitoring tools. However *mtrace* provides only receiver-to-source traces and must be repeated for each group member. Hence it suffers from serious scalability problems. *MHealth* [9] provides a visual front-end for *mtrace*. Similarly, *MantaRay* [10] is a visualization tool for *mrwatch/mrinfo*[11] which can be used to query a multicast router for information about its interfaces and neighbors. Other tools such as *mstat, mrtree and mview* [12] collect data using SNMP [13,14]. Still other tools such as *mlisten*[15], *rtpmon*[16] and *sdr-monitor*[17], collect data at the application layer. *Netstat* provides information about the local machine configuration. Tools such as *mrinfo, map-mbone* are DVMRP specific and are not useful in the current multicast infrastructure. Taxonomies of useful multicast tools exist and serve as a useful aid in understanding the options for management of multicast networks [2,4,18,19]. As can be seen from these taxonomies, the resource requirements and functionality of these programs varies widely.

Many past monitoring efforts for the multicast infrastructure have been global in nature. For example, Mantra [20] addresses the need for global views taken from routers world-wide by capturing MBGP and MSDP state information within the routers. This tool has been designed to deal with topological complexity as well as diverse multicast protocols. Though Mantra provides a great deal of information about the state of multicast in the Internet, it cannot be queried by the end user; does not use SNMP; and cannot communicate with remote installations of itself. The problem that MCPM seeks to address is the dynamic gathering of information and its analysis and presentation on

a local scale. Further, MCPM shares this information using a peer to peer model. We stay away from trying to construct a centralized view of the whole network.

The NLANR multicast beacon [21] is also an important tool for the measuring multimedia stream qualities like load, delay, jitter, and order. A brief discussion of such measurements with respect to multimedia streams is given in the Real Time Protocol (RTP) standard [22]. An interesting application built around the beacon project is the multicast tester [23]. This tester is a Java applet that allows an end user to detect whether the network is multicast enabled or not by listening to the NLANR beacons. A few web-enabled multicast visualization tools exist as well. For example, the Abilene Multicast Route Viewer maps the shortest-path tree of a given mroute tree across the Abilene backbone [24]. The Abiline weather map [25] represents one aspect of what MCPM achieves in a more localized, problem-specific domain.

Our work with MCPM adds to the list of individual tools, an infrastructure for multicast monitoring using these pre-existing tools. Besides, MCPM is expandable to accommodate new generic tools and new protocols. Turning now to MCPM, specifically, we first speculate on the functionality that can be supported through MCPM and illustrate its benefits. Second, we propose an architecture for the system. Third, we describe what information is important to an end user and techniques for how that information can be gathered. Finally, we consider important issues on the level of detail an average user can handle, how to provide controlled access to critical infrastructure components, usage priorities and cacheability of retrieved information.

4. DESIGN GOALS

We aim to develop a generic architecture for collecting state information from multicast capable entities and for troubleshooting multicast problems. Figure 1 shows the outline of the design and how it connects to external entities. MCPM provides monitoring as well as management functionality. To better define the exact role which this system should play in the network environment, we first sub-divide management into the following broad areas [26]:

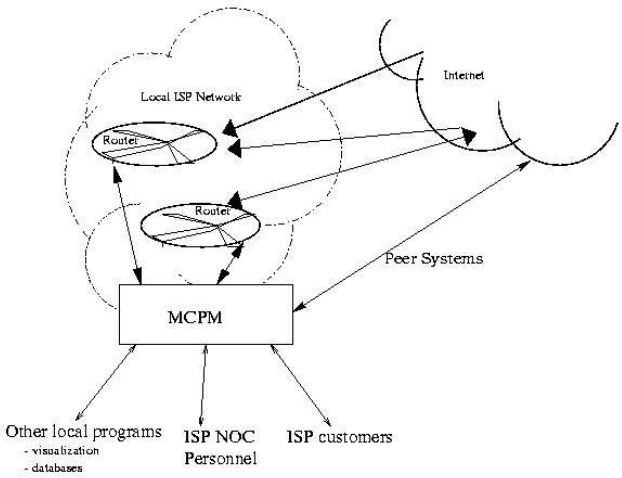

Figure 1. Information flow through MCPM.

- **Operational maintenance**: This area deals with the proper operation of various interacting layers and the maintenance of distributed state information. Management of this area requires network surveillance, error detection, diagnosis, and system control.
- **Resource management and statistics gathering**: This area is responsible for gathering and analyzing data from the network. Accounting and financial management is also a part of this segment.
- **Planning and provisioning**: This area includes evaluating and aggregating the data gathered from the previous two tasks and using it to provision future needs or to prevent future outages.

Typical network utilities or tools fall into at most one of these broad areas. The idea of consolidated monitoring is to provide a combined solution for more than one of these areas through a single system. In case all capabilities are not built into the system itself, it must provide a means for other applications to fulfill those roles. In this context, MCPM fits well in the resource management, statistics gathering and operational maintenance roles.

Refining these several areas into a more specific set of functional goals means that the proposed architecture should have the following capabilities:

- The system must be able to detect the various protocols in use and interact with the local agents responsible for those protocols.
- The system must be capable of performing connectivity tests and queries on specific <S,G> pairs.
- The syatem should be capable of enforcing policy and must distinguish between end-users, NOC operators and administrators and provide service according to the defined policy for these classes of users.
- The system must help in regulating network load created by active monitoring.
- On a macro level, the system must present hooks to connect with its peers so as to build a global monitoring system.
- Finally, the system must present a simple interface to the end user and at the same time must provide a useful level of detail to the network engineers.

On an architectural level, the design must be flexible and extensible. First, the software in its initial implementation serves as a wrapper around pre-existing tools and consolidates the results put out by individual tools into a consistent whole. For this reason and also because multicast is a rapidly evolving science, there must be a way to incorporate new data gathering modules or tools into the system. In the future, modules might be written directly as plug-ins for MCPM or could be stand-alone tools wrapped and plugged into the system. Second, it is envisioned that MCPM could communicate with Mantra-like tools and use the information gathered by them. Third, when new protocols are introduced in the mesh, monitoring problems might arise. For example, the almost complete dismantling of DVMRP rendered MBone-specific tools useless. Similarly, the use of Source Specific Multicast (SSM) [27] will eliminate the use of MSDP. Therefore tools that depend on MSDP peers for data collection will become obsolete. It will be viable to collect per group statistics but per source monitoring will become much more difficult. Because of all these factors, extensibility is a major requirement.

In addition, the solution must also scale well to fit the size of the ISP where it is deployed. The peer-peer communication should also be designed with scalability as a primary concern. Also robust error control must be built into the design such that if a sub-process faults the whole system must not stop. Also if a sub-process returns erroneous results recovery must be graceful, alternative routes for diagnosis must also be available.

The final requirement is to present information in an intuitive manner. In this regard, presentation tools such as RRD [28], Otter [29] and GeoPlot [30] provide mechanisms for presenting various types of results ranging from statistical, to topology views, to geo-mapping. These must be well integrated into the system or an interface must be provided to pass data to them.

5. ARCHITECTURE

Keeping in view the high level set of design goals, we now describe the architecture for MCPM. A pictorial representation of our scheme is shown in Figure 2. The system broadly consists of data collection, data archival, policy enforcement, and scheduler sub-systems. The flow of data and control in MCPM can be described as follows. Data is collected through two methods: 1) through constant monitoring, and 2) asynchronously as per specific user and NOC queries. Data collection activities are checked by the policy enforcement and control component. This allows us to apply unique policies to each data gathering element. These data gathering elements in turn, might be wrapper functions, peers, or any othe type of agents. The filtered data produced through the policy component is then archived and at the same time processed and displayed. We now discuss each of these components in greater detail.

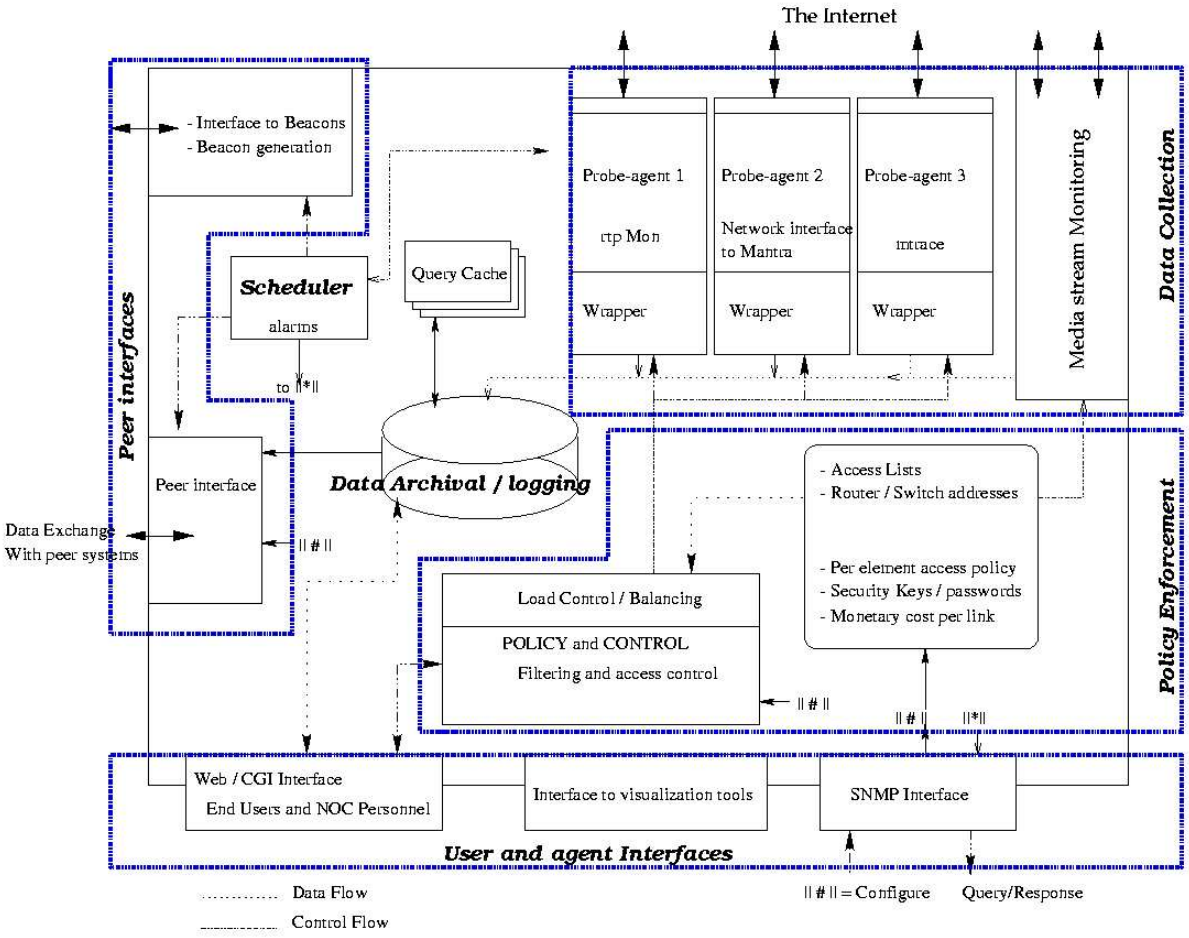

Figure 2. The MCPM architecture.

5.1. Data Collection

The architecture of MCPM is dictated by the type of data collected and dispersed by it. Data collection is sub-divided into collecting a) intra-domain global data b) inter-domain global data and c) local views of session data. Global data is collected by crawling the topology and by periodically probing the network. This data is available to all running threads and reporting components. Good candidates for this kind of continuous active monitoring are protocols and their parameters, version information, total number of multicast capable hosts online on the subnet, active alarms, other information about system state, and peering status. The further subdivision of this data on the basis of inter- or intra-domain is based on the premise that gathering data within a local domain is easier than accessing other domains. Thus the access policy for inter-domain data collection would be comparatively more restrictive. The gathering of both these global data types is dictated by a scheduler thread that periodically invokes a sequence of probe-agents. This sequence of agents is fully configurable by the administrator.

In contrast to this background crawling, a major component of MCPM works on processing real time user queries. We define session data as the data generated as a result of specific end user inputs which cause the launch of probes. This activity would include <S,G> reachability questions, connectivity questions, protocol specific queries, and tree visualization queries. It is likely that these queries are launched by network personnel investigating some aspect of multicast traffic. Of course, some queries are satisfied by supplying global data and no probing is required. Still other queries might be satisfied from the cache of recently supplied results.

This sub-division of the data collection activity helps during implementation because each task uses an independent threading structure. The actual task of data gathering is performed by agents called "probe-agents" which we now describe.

Probe Agents and Wrappers. The core of MCPM consists of 1) pre-existing tools and 2) interfaces to other systems (e.g. Mantra). The problem we face while trying to leverage pre-exsting tools, is the great diversity of input and output formats accepted by the present set of tools. To solve this problem we define *wrappers* around each tool. The combination of a *wrapper* and the wrapped tool is called a *probe-agent*. These *wrappers* convert tool-specific output into eXtensible Markup Language (XML) which is then archived on to the *data store* and also used to form the response to the querying entity. Each *wrapper* has an associated Data Type Definition (DTD) which defines the validity of the document structure. The use of XML allows us to leverage the already existing parsing and validation tools. Defining a propriety format does not offer us any tangible advantages. The wrappers provide abstraction from the specific input and output formats of the tools. This allows us to focus more on developing a meaningful user interface. The more interesting part of this abstraction is that in addition to launching local tools, MCPM can communicate with remote tools seamlessly. These tools, such as Mantra and the looking glass [*] sites all over the world can be plugged inside the wrappers as if they were local tools. Their output can then be manipulated and utilized as per normal policy. In Figure 2 notice the control dependency between the *Scheduler* component and *probe-agents*. This means that the same tools can be called upon during "crawling" (data collection) phases exactly as if they were invoked as a result of specific user queries. Another possibility is that SNMP agents can be wrapped in a similar fashion and can communicate with routers using SNMP. This would provide us an alternative path to heavier active probing.

Media Stream Monitoring. In addition to tool-based data gathering, there is also a a dedicated component of MCPM that joins various multimedia streams as a receiver and monitors the quality of the stream based on loss, delay, and jitter. Each join and leave event is preprogrammed by an operator based on the begin and end time of a media session. The measured parameters are compared with acceptable values and an alarm is generated for any aberrations. Along with an alarm, a plan of action to further diagnose the problem might be invoked. By the time support staff react to the alarm, the snapshot of the system at the point of failure is already available for troubleshooting. The advantage of our consolidated approach is that it allows us to use existing like *RTPmon* and offers the flexibility to add other probe-agents as other kinds of traffic arise in the network.

[*] http://nitrous.digex.net/

5.2. Data Storage and Caching

All data, whether session specific or global, is logged after compression to disk. The use of XML makes data storage and offline analysis easier and and more flexible. It is tempting to think of caching the results of user queries, and use the cache results in case of repeated queries. This would of course suggest that caching be made a function of load, i.e. if a particular tool is invoked repeatedly by different users, then the system should start caching its results. The age on the cache content could be made an inverse function of the load on that resource. The most important aspect though is not to return the cached copy to the same user twice, i.e. if different people ask for the same *mtrace* at approximately the same time, it is allowable to return to the second person the results of the first. But, if the request is from a single source then the tool must be run again. The key issue is a tradeoff between whether the network can tolerate additional query load while maintaining the required service levels. The viability of caching is still an open area and the subject of ongoing research.

5.3. Interfaces

The MCPM offers two groups of interfaces to the outside world: a) user/agent and b) peer interfaces.

User/Agent Interface. The user group is further sub-divided into three interfaces. The first of these is a web-based interface through which normal end-users and NOC personnel launch queries to gather information and try to troubleshoot any problems. This is the default interface for human interaction. Figure 3 shows the main pieces of this component. The level of detail that passes through to the end-user depends on the group to which the user belongs. This filtering is performed by the output component of the WEB/CGI interface.

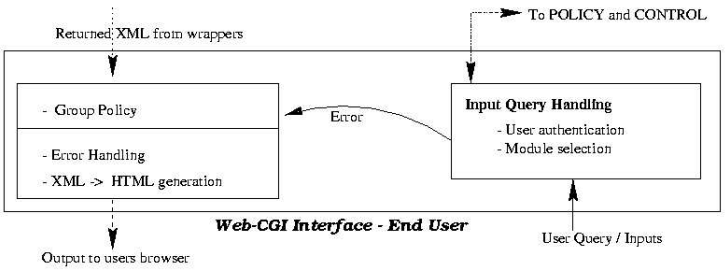

Figure 3. Details of the Web/CGI interface.

The second interface is an SNMP MIB that allows any outside SNMP agent to query MCPM about its state. It is through this interface that various parameters inside the system are set. For example, the rate of exchanging information with peers, the level of detail allowed, modifying access policy, setting up security keys, setting time outs on alarms are all remotely managed using SNMP. This is the preferred administrative interface.

The third interface allows customized data linkage for offline analysis and visualization. Existing tools such as Otter and Geoplot can be used for realtime display of results, long term trend analysis, and trend prediction. Potentially numerous applications can be linked to MCPM using this interface. For example:

- **Visualization** tools that can be used to plot collected data over various levels of granularity.
- **Database** and data mining applications can be used here to organize data or understand long term dynamics.

Peer Interface. The second group, that of peer interfaces, is responsible for listening to beacons, generating beacon data, and communicating with other similar systems. Thaler proposes a Globally Distributed Troubleshooting (GDT) protocol for interconnecting arbitrary troubleshooting systems [3]. The GDT protocol or a variant could be used by MCPM to communicate with other GDT-capable systems through this interface.

	Agents	End User	NOC personnel	Administrator
Input high level queries	No	Yes	Yes	Yes
Invoke probe-agents directly	No	No	Yes	Yes
Enforcement of group policies	Strict	Strict	Flexible	No
Access to raw data	Yes	No	No	Yes
Text output and suggestions	No	Yes	Yes	Yes
Topological/Map based output	No	Yes	Yes	Yes
Interpretations of data	No	Yes	Yes	Yes
Upgrade system	No	No	No	Yes
Shutdown system	No	No	No	Yes

Table 1. User groups and associated mcpm privileges.

5.4. Policy and Control

One of the major aims of this project is to discourage end-users from indiscriminately running monitoring tools and causing random load on the network. A centralized troubleshooting system would be able to access may more resources than an end-user normally has access to, and would be able to present the results in an understandable, coherent manner. This implies that MCPM would proxy the requests on behalf of the clients and perform the monitoring itself. This proxying also means that an ISP can hide sensitive parts of its network from the users.

Another implication is that the system must be aware of the sensitivity of information returned by each type of query. To this end, the system maintains user groups and access lists to specify what entities can be queried by which class of users. We implement four classes of users as shown in Table 1. If logging into a router requires the use of a special key or password, that information is maintained here as well. On the implementation level each resource has an access script associated with it and access to this script is controlled by the access policy. We create a separate set of policies dictating how much load each network element can handle without affecting its core functionality, thus specifying upper bounds on the rate at which the resource can be queried. For example, it may be determined that a router should only support a certain number of SNMP queries in a given time frame. This would be specified as an access policy and MCPM would prioritize and possibly reject queries based on load.

Policy is also used to bar some networks or resources from being accessed at all. Some policies might place a usage weight on a resource. Hypothetically this might indicate the price for monitoring that resource and access rights would depend on this parameter as well. Finally, since the MCPM allows interaction between the NOC and end-users therefore if an end user runs a few tests and the conclusion is that multicast is not being received, this information should be sent to the ISP's NOC immediately. Because the NOC cannot monitor all systems all of the time, utilizing the problems detected by end users is important data that should not be ignored. The decision on how to route this error report on the basis of severity is also made by the policy and control component.

6. FUTURE WORK AND CONCLUSION

At present there is a wide gap between the maturity of IP multicast protocols, their field deployment, and the ability to adequately manage multicast traffic. In this paper, we have described the problems caused by the lack of a uniform multicast management system. Because such a system does not exist, NOC personnel are left to using a variety of task-specific tools. As a result, NOC personnel need a vast amount of expertise and experience in debugging multicast networks. Furthermore, there is often a significant barrier to allowing end users access to even the most basic information about multicast traffic in their ISP's network.

Our solution to this problem is a system called the Multicast Consolidated Proxy Monitor. The basis idea of the system is to present a unified interface that provides varying levels of monitoring functionality depending on the audience. NOC personnel have full access to all types of information, while end users have access only to basic information. Another key to the MCPM system is that it takes advantage of the numerous existing tools and codified debugging strategies. Input and output wrappers are provided for existing tools, and various "plans of action" have been created to automatically handle simple debugging scenarios. In essence, The MCPM system is a centralized control and coordination for multicast monitoring and management.

While we have developed the basic system design and implemented some of the basic functionality, most of the development and evaluation of the system remains. Utilizing our experience in developing multicast monitoring systems, we hope to be able to build a prototype and deploy it in a production ISP's network. Evaluation will consist of qualitatively measuring the utility of the system, and also quantitatively measuring various properties including scalability, caching effectiveness, system performance, monitoring overload protection, etc.

Through this work we hope to make the task of multicast troubleshooting simpler–to the point where not only is multicast a robust service, but also that users are able to verify proper operation. Multicast traffic management is one of the steps necessary to further the goal of widespread multicast deployment.

And finally, although the multicast scenario is used to illustrate the benefit of consolidated proxy monitoring, the ultimate aim is to generalize the scheme to unicast IP networks.

REFERENCES

1. K. Almeroth, "The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment," *IEEE Network*, vol. 14, pp. 10–20, January/February 2000.
2. K. Sarac and K. Almeroth, "Supporting multicast deployment efforts: A survey of tools for multicast monitoring," *Journal of High Speed Networking–Special Issue on Management of Multimedia Networking*, March 2001.
3. D. Thaler, "Globally distributed troubleshooting (GDT): Protocol specification." Internet Engineering Task Force (IETF), draft-thaler-gdt-*.txt, January 1997.
4. D. Thaler and B. Aboba, "Multicast debugging handbook." Internet Engineering Task Force (IETF), draft-ietf-mboned-mdh-*.txt, March 1997. work in progress.
5. K. Almeroth, L. Wei, and D. Farinacci, "Multicast reachability monitor (MRM)." Internet Engineering Task Force (IETF), draft-ietf-mboned-mrm-*.txt, October 1999.
6. C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, pp. 10–20, January/February 2000.
7. W. Fenner and S. Casner, "A 'traceroute' facility for IP multicast." Internet Engineering Task Force (IETF), draft-ietf-idmr-traceroute-ipm-*.txt, August 1998.
8. B. Fenner, *Multicast Traceroute (mtrace) 5.2*, September 1998. ftp://ftp.parc.xerox.com/pub/net-research/ipmulti/.
9. D. Makofske and K. Almeroth, "MHealth: A real-time graphical multicast monitoring tool for the MBone," in *Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, (Basking Ridge, New Jersey, USA), June 1999.
10. B. Huffaker, K. Claffy, and E. Nemeth, "Tools to visualize the Internet multicast backbone," in *Proceedings of INET '99*, (San Jose, California, USA), June 1999.
11. B. Fenner, *mrouted 3.9-beta, mrinfo, and other tools*, March 1998. Available from ftp://ftp.parc.xerox.com/pub/net-research/ipmulti/.
12. D. Thaler, *Mview, Mstat, and Mrtree.* Merit Network and Univ of Michigan. Available from http://nic.merit.edu/~mbone/mviewdoc/Welcome.html and http://www.merit.edu/net-research/mbone/mstat.html.
13. 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.
14. J. Case, R. Mundy, D. Partain, and B. Stewart, "Introduction to version 3 of the internet-standard network management framework." Internet Engineering Task Force (IETF), RFC 2570, April 1999.
15. K. Almeroth, *Multicast Group Membership Collection Tool (mlisten).* Georgia Institute of Technology, September 1996. Available from http://www.cc.gatech.edu/computing/Telecomm/mbone/.
16. 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/.
17. K. Sarac and K. Almeroth, "Monitoring reachability in the global multicast infrastructure," in *International Conference on Network Protocols (ICNP)*, (Osaka, JAPAN), November 2000.
18. *Multicast Debuggung Handbook Home Page.* http://imj.ucsb.edu/mdh.
19. CAIDA, *Multicast Measurement Tools Taxonomy.* http://www.caida.org/tools/taxonomy/multicast.xml.
20. P. Rajvaidya, K. Almeroth, and K. Claffy, "A scalable architecture for monitoring and visualizing multicast statistics," in *IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM)*, (Austin, Texas, USA), June 2000.
21. *NLANR Multicast Beacon (v 0.63).* http://dast.nlanr.net/Projects/Beacon/.
22. H. Schulzrinne, S. Casner, R. Frederick, and J. V., "RTP: A transport protocol for real-time applications." Internet Engineering Task Force (IETF), RFC 1889, January 1996.
23. *Multicast Tester.* http://www.multicasttech.com/mt/.
24. *Abilene Multicast Router Viewer.* http://palpatine.ucs.indiana.edu/mrview-abilene/.
25. *Abilene NOC Weathermap.* http://hydra.uits.iu.edu/ abilene/traffic/.

26. M. Yoshida, M. Kobayashi, and H. Yamaguchi, "Customer control of network management from the service provider's prespective," *IEEE Communications Magazine*, March 1990.
27. H. Holbrook and B. Cain, "Source-specific multicast." Internet Engineering Task Force (IETF), draft-holbrook-ssm-arch-*.txt, September 2001. work in progress.
28. *RRD Tool.* http://ee-staff.ethz.ch/~oetiker/webtools/rrdtool/.
29. B. Huffaker, E. Nemeth, and K. Claffy, "Otter: A general-purpose network visualization tool.," in *INET*, (San Jose, California, USA), June 1999.
30. R. Periakaruppan, *GeoPlot - A general purpose geographical visualization tool.* Available from http://www.caida.org/Tools/GeoPlot/.