mDNS - A Proposal for Hierarchical Multicast Session Directory Architecture

Piyush Harsh and Richard Newman
(pharsh, nemo)@cise.ufl.edu
Telephone/Fax: +1.352.392.1200/1220
Computer and Information Science and Engineering
CSE Building Room E310, University of Florida P.O. Box 116120
Gainesville, FL 32611-6120 USA

TARGET CONFERENCE: ICOMP'08
Contact Author: Piyush Harsh

March 7, 2008

Abstract

Bandwidth in the Internet is constantly increasing. The last mile problem of the Internet has almost been solved. Multimedia has emerged as the favorite mode of information dispersal on the net. Multicast is increasingly being seen as the vehicle of choice for multimedia streams. What has been the one true stumbling roadblock in widespread deployment is the lack of DNS like structure for multicast session discovery. In this paper we look into some of the existing techniques that tries to address this issue, find out benefits and drawbacks of such schemes. We will propose our hierarchical and globally scalable sessions directory architecture. This will be followed by analysis of benefits and drawbacks in our scheme and why our scheme might be generally more suitable for global deployment which may allow end users to enjoy the true power and efficiency of IP multicast.

Keywords Multicast, Internet Architecture, DNS, Resource Discovery, Scoping

1 Introduction

IP Multicast [4] is seen as the popular vehicle of content delivery for rich multimedia applications in the coming future. Even though Internet average bandwidth is increasing and more homes are getting access to high speed internet, IP unicast simply can not handle huge subscriber base. The bandwidth requirements would scale linearly for each subscriber and no amount of bandwidth would be sufficient. Maintaining huge server farms can become a costly proposition for organizations.

Even though the time seems appropriate for widespread popularity and deployment for IP Multicast, it has not been the case. In our opinion, the lack of DNS [16] like structure may be the single most important stumbling block for this. Some people may argue using DNS itself for multicast sessions information dissemination but we disagree. Information stored in DNS is relatively stable over long period of time whereas multicast sessions can be very dynamic. Multicast session addresses itself are not long term entities generally (except in a few cases). There is a need for a similar directory structure which is hierarchical yet which can handle dynamic multicast environment at the same time being robust and globally scalable.

Until now, the only way users could use Multicast sessions was if they somehow knew the session details beforehand. This has been done via emails, IRC channels and blog postings. If multicast has to become truly usable, a way has to be devised for users to retrieve these critical session parameters on the fly. Session retrievals using URL like strings can go a step further towards this goal.

In the rest of this paper we will provide summary on current and past proposed solutions for the issues raised above. Next we will provide the top-level architectural layout of our proposed scheme. We will provide a general URL construction scheme for multicast sessions and would propose additional parsing and translation rules and query handling infrastructure geared towards making multicast more user friendly and easily accessible and practical. Towards the end we will discuss possible drawbacks and benefits of our approach. We will end this paper with possible future research directions and interesting questions that still remains to be answered.

2 Related Research

'sdr' [7] - Session Directory tool has been very popular among Mbone [2] enthusiasts. It has been used as a session management tool and is primarily based on LBL's Session Directory tool - 'sd'. It makes use of SDP - Session Description Protocol [8] to announce the critical characteristics of multicast sessions such as channel address, port numbers, timing and resource information for remote hosts to join the session. It uses a well known multicast channel address for propagating this information on the Internet. It also maintains a cache of other multicast sessions advertised elsewhere on the Internet through sdr. Based on the cache information maintained locally by each 'sdr' client, it tries to assign a new channel address to requesting multicast sessions in such a way as to reduce the address collisions among different sessions. It is the general consensus of the research community that even though 'sd' / 'sdr' was a great technology demonstrator; it is not suited to scale globally. All 'sdr' clients essentially maintain a flat hierarchy on the Internet which makes information dissemination among different 'sdr' clients a challenging prospect.

Andrew Swan and team [20] at Berkeley developed a completely decentralized sessions directory which they incorporated as part of their Light-Weight multimedia sessions framework. In this architecture they advertised the multimedia session's bindings at well known bootstrap address. They made use of Sessions Announcement Protocol [9] for this purpose. To overcome the latency issue that plagues the multicast session directory architecture based on LBL's 'sd' application and SAP announcement bandwidth limitations, they proposed a tired announcement rate approach. The announcement agents under local scope announces session advertisements at a much higher frequency than traditional SAP clients. They also proposed splitting the traditional SAP client into two parts, one persistent server that runs SAP and caches all the network SAP announcements heard over a long period of time, and another ephemeral client that contacts this persistent server for the cached list of available sessions.

In [18] Joaquim and team analyze the use / misuse of SDP as a session directory tool to advertise multimedia sessions. They argue that the session directory information that is embedded inside SDP fields is not standardized. Had it been standardized, these fields such as "media", "repeat time", "time active" etc. may be used for aggregating sessions which could then be used later to query for sessions. They propose that user should not be burdened with the task of browsing a flat structure; instead the task could be assigned to a server. This information could be presented to the user using either a well known multicast channel, or using several multicast channels or maybe using some specific server database. The article did not specify to what extent any of these goals were already implemented and what was the current status of their work.

Another attempt at making a distributed information discovery system in the Internet was Harvest [3]. The system was built using subsystems such as gatherers that were placed at the resource site, brokers which collected data from gatherers and incorporated this data in their resource index, brokers further consisted of index/search subsystems that were optimized for space and/or search time. The system also made use of replicators to replicate the data over multiple site in the Internet and object caches at critical sites to minimize the communication overload in the Internet. Instead of harvest being truly hierarchical, we believe it was a replicated, Internet wide cache and not suitable for multicast sessions discovery problem. It would be difficult to incorporate scoping and session lifetime requirements within their proposed framework. Also the dynamic nature of many sessions would result in cache instability.

Researchers at UCLA have proposed a scalable multicast information discovery graph (IDG) [19] based on the semantic description of stored as well as real time multimedia content over the Internet. This work is being done under Sematic Multicast project [1]. Even though their proposal provides for a hierarchical semantic directory, its not truly distributed. They have proposed making use of caching and soft state refreshes to make their system more scalable and robust. In their approach, new multicast user may have to start searching at a well known root directory server which may create a bottleneck as the number of sessions and users grow. This scheme may also be problematic in the face of stale caches and periodic cache refreshes. The semantic hierarchy in their proposal currently is coarsely defined and may require significant rework in order to account for variety of multimedia sources uploaded online these days. The architecture does allow for much better search time compared to LBL's 'sd' tool but the bandwidth requirement grows linearly with the number of data sources. This is clearly a source of concern. Also multicast scoping requirements have been overlooked in the published work.

In [12], the authors proposed building an anycast SDP Proxy in order to give end-users immediate access to session announcements. They proposed an architecture along with HTTP style protocol format to access session information as well as create session entry at the remote SDP Proxy. This approach still suffers from the traditional 'sdr' issues of non-scalability in the face of large number of sessions. Another issue with their approach is deploy-ability over SSM [11] only networks.

In [17] the authors deal specifically with multicast session announcements over SSM networks. They propose a 2 tier hierarchy of dedicated session announcement servers (SAS). They propose to reduce the SAP related delay by increasing the allowed bandwidth limit of 4kbps to 50kbps for intra-domain announcements. SAS servers located in the backbone network of various ISP networks (level 2 SAS servers) act as relays and no not cache any announcement. Just increasing the intra-domain bandwidth seems to be a nearsighted solution at best. They state that whenever a new SAS level 2 server is added in the ISP's backbone network, the ISP finds out the address of other level 2 SAS servers and this information is configured into the new server. They failed to mention how this is achieved and whether the configuration is manual or automatic?

3 mDNS: DNS-aware multicast session directory architecture

We have proposed a globally scalable multicast session directory architecture which has been designed on the similar principle of domain name server (DNS) hierarchy. Here are the list of terminology that we will use in the rest of this section -

- MSD_x^y - Multicast Session Directory (MSD) server 'y' in domain 'x'
- MSD_x^d - Designated MSD Server in domain 'x'
- DNS_x - Domain Name Server for domain 'x'
- URS_x - URL registration server in domain 'x'

We will assume that each domain knows its DNS server address and DNS servers know about their parent DNS server's address which is a reasonable assumption to make. Also for global discover-ability of multicast sessions, we will assume that at least one MSD server coexists with the DNS server at each domain level. Failure of doing so may result in disconnected islands of sessions discovery zone in the global Internet (which may be desired in some cases). We propose to make an additional entry into DNS server's record table. We call it MCAST record and it contains such details as 'anycast IP address' [13] for MSD servers in that domain, globally scoped multicast channel details for establishing multicast group with designated MSD server of a particular domain and designated MSD servers in the children subnets and the globally scoped multicast channel details for establishing multicast group with designated MSD server of that particular domain and designated MSD server in the parent's domain. Additionally it may contain address of URS server in its domain. An example DNS MCAST record entry may look like -

```
@MCAST{
    ANYCAST=a.b.c.d
    CMCAST=233.[ASN Byte1].[ASN Byte2].XXX
    PMCAST=233.[ASN Byte1].[ASN Byte2].???
    PORT=pqrs
    URS=x.y.z.w
}
```

Note that the above example is just for illustrative purpose and actual DNS entry must follow the correct DNS entry format. Notice that we have suggested the use of globally scoped addresses from the GLOP [14] address range. These address are assigned by the ISPs and the domain owners / administrators must apply for these addresses from their ISPs. Also ASN denotes the AS (Autonomous System) Number. The system has been proposed to co-exist with our HOMA [10] multicast address allocation and management scheme.

3.1 mDNS hierarchy construction

We will describe the hierarchy of our scheme through two example networks, namely .edu hierarchy network and a general hypothetical ISP network. Our scheme will work with any network organization scheme as long as our initial DNS server and MSD server assumptions remain valid. A typical example is shown in figure 1 above.

Now we try to explain the proposed mDNS hierarchical architecture in some detail now. Let us concentrate on just the .edu hierarchy. Under the

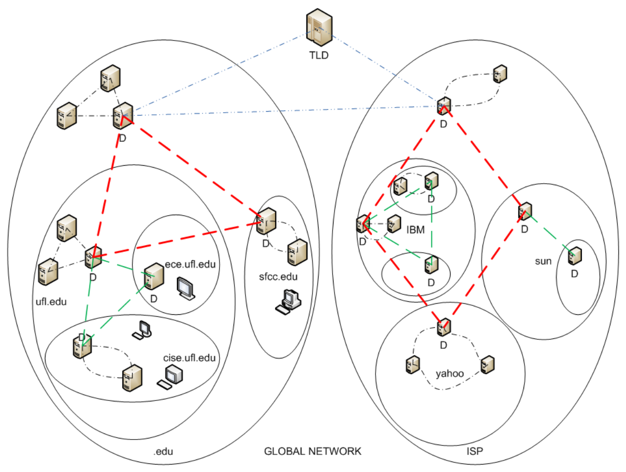

Figure 1: a typical MSD hierarchy

.edu domain we have two university networks. Under UF network we have 2 sub-domain namely CISE and ECE, both of these are in themselves independent administratively scoped multicast domains. Further UF is also administratively scoped domain. CISE and ECE maintain their own DNS server whose parent DNS server will be the DNS_{UF} server. DNS_{UF} server is a child node of the TLD $DNS_{.edu}$ Server. UF maintains multiple MSD_{UF} servers, all of which subscribe to a fixed (possibly IANA assigned) administratively scoped multicast channel. From here on we will refer to this channel as MSD-LOCAL-MCAST channel. CISE and ECE also maintain their own sets of MSD_{CISE} and MSD_{ECE} servers, again those subscribe to MSD-LOCAL-MCAST channel. Since this channel is administratively scoped channel, if the edge routers are properly configured then there should be no cross-talk among these channels.

If there are multiple MSD servers maintained under a domain, a designated MSD server is chosen based on some leader election algorithm [15] [5]. In figure 2, these are marked with the letter 'D' next to them. The designated MSD^d servers joins two globally scoped multicast channels namely those specified by CMCAST and PMCAST entries in the MCAST record of the DNS server in their domain. If any of these two entry is NULL that particular channel is

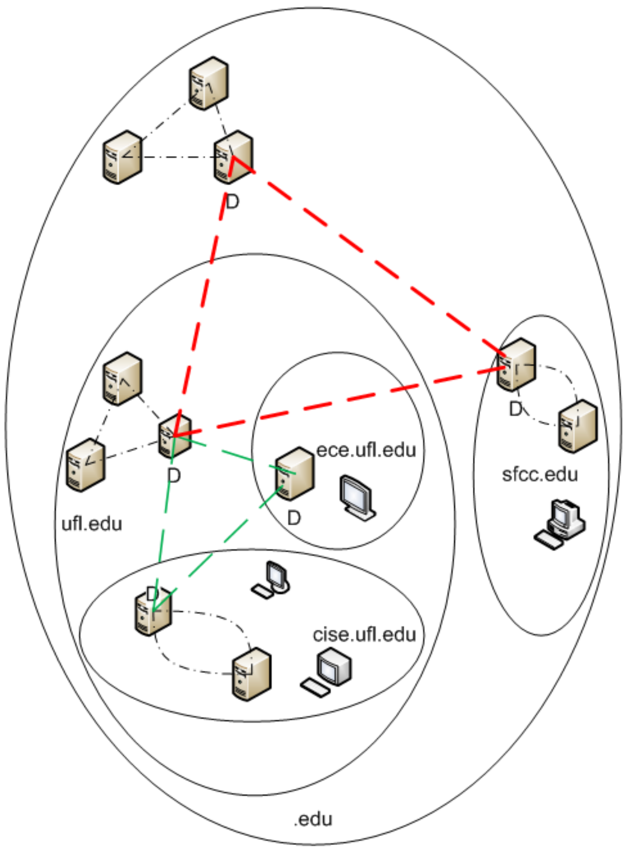

Figure 2: an example .edu hierarchy

not subscribed to. It is important to note that all the MSD servers in any particular domain (excluding children subdomains) are anycasted at the anycast IP address specified in the DNS server's MCAST record.

```
MSD Server's Base Algorithm
join MSD-LOCAL-MCAST channel
initiate leader election on this chanel
if elected leader then
    query local DNS server for PMCAST and CMCAST
    join PMCAST and CMCAST channels
end if
```

This is how MSD servers' hierarchical structure is established in mDNS architecture. It is quite easy to see that the hierarchy will exist as long as the initial two assumptions are satisfied in any network domain hierarchy. We chose to present our example for a .edu domain just for illustration purposes.

3.2 Session registration in mDNS

mDNS architecture has been designed to co-exists with our HOMA [10] proposal. We would assume that an application in any domain before transmitting multicast traffic (in any scope) has an appropriate multicast channel address allotted to it. In this paper we have not provided details on the internal database maintained at MSD servers but let us assume further that MSD servers are capable of registering channel keywords along with channel details on behalf of multicast applications. It is the responsibility of the session creator to provide up-to maximum of 10 keywords correctly describing the session. These keywords will aid in session discovery process described later.

An application under mDNS architecture, after it has acquired a valid channel address, will execute the following pseudocode -

```
Session registration pseudocode
contact local DNS Server to find URS address
if URS server exist then
    request URS server to register a channel descriptive 'keyword'
    if keyword registration at URS successful then
        done.
    else
        pick another keyword
        try again
    end if
end if
initiate channel registration request on the MSD-LOCAL-MCAST
register the channel details with MSD server
provide list of keywords (max upto 10)
provide session duration and operating times
provide URS registerd 'keyword' if any
```

.

The MSD server will correctly identify the scope of the multicast channel based on the channel address being registered with it.

3.3 mDNS search operation

Multicast sessions in mDNS can be searched using session keywords. Sessions can also be accessed directly if the session creator successfully registered a valid 'keyword' with the domain's URS server. We have proposed a simple URL scheme in order to facilitate multicast channel details access in order for the remote host to sunscribe to that particular channel on the Internet under mDNS architecture. We believe that having an URL scheme will greatly enhance the usability of IP-multicast and would alleviate it to its fullest potential rapidly.

mDNS URL is constructed using the following syntax -

<protocol>://<domain URL>/<URS Keyword>

In the above URL scheme, protocol could be the method the remote user will use to communicate with the MSD server defined in the DNS MCAST record. It could be HTTP or a similar protocol. The domain URL helps resolve the MSD server located in the multicast session creator domain. It must begin with 'mcast' to specify the MSD server. For example, let us assume that under cise.ufl.edu sub-domain, we have created a globally scoped multicast channel that multicasts information about gators. Further assume we have successfully registered the keyword 'gators' with the URS_{CISE} server located in CISE domain. Someone else can then directly access the multicast session details to subscribe to the gator channel using this URL -

http://mcast.cise.ufl.edu/gators

The search is done in a very similar fashion, under mDNS scheme, user can do domain specific search or general global search. If the user wishes to do domain specific search, he can specify the specific domain to search for a particular keyword in the same fashion as specified above but instead now using the qualifiers "search" and "keyword" in the URL. For example, if the user wishes to find sessions with keyword "gators" under cise.ufl.edu domain, they can do so by using the string -

mcast.cise.ufl.edu/search=all&keyword=gators

The end users' multicast search application would resolve the mcast.cise.ufl.edu anycast address and would connect to one of the MSD_CISE servers located in the cise.ufl.edu domain. MSD servers perform database search against the keyword "gators" and every content type. Content type could be audio, video, whiteboard, etc. to name a few possible types. The search is performed in a top-down hierarchy starting from the domain specified in the search URL and percolating down to all sub-domains (if any exists). In our present scheme, the search results are returned from MSD servers at each level directly to the requesting client. There are few other delivery schemes we are considering that we will describe in future research section. The MSD servers are smart in a sense that they recognize whether the query comes from a host inside its domain or outside the domain. If the querist is located outside the MSD domain, the MSD returns only those search results that are globally scoped channels. Administratively scoped sessions are only returned as part of query if the querist resides in the same domain.

Additionally any user can choose to perform a global keyword search. In order to do this, the client must contact the local MSD server co-located at the same zone as its DNS server. Global search is propagated by the MSD servers on both PMCAST and CMCAST channels and its own MSD-LOCAL-MCAST channel thereby spreading the search to both child domains as well as parent domain. The propagated search request contains uniquely identifying string that allows the originator MSD to kill the search if the same query id comes again to the same MSD server. Also the search query based on the identifying sting is never re-propagated on the same channel on which it was received. It is easy to see that in the proposed mDNS architecture, this uniquely identifying search id will be generated by the designated MSD server at each level and only when the search query was received on the MSD-LOCAL-MCAST channel.

Here is the pseudocode for search which is executed at each MSD server -

```
MSD pseudocode for search
received search query on subscribed channels
```

```
search sessions internal database for search match
if multicast sessions found then
    if querist resides in another domain then
        return only globally scoped session details (if any)
    else
        return every result found directly to the querist
    end if
end if
if MSD server is a designated MSD server then
    if search unique ID is missing then
        generate unique search ID
    end if
    if query request was not received on CMCAST channel then
        propagate search on CMCAST channel
    end if
    if search is global in nature then
        if query request was not received on PMCAST channel then
            propagate search on PMCAST channel
        end if
    end if
    if search query has self generated previous ID then
        drop search request
    end if
end if
```

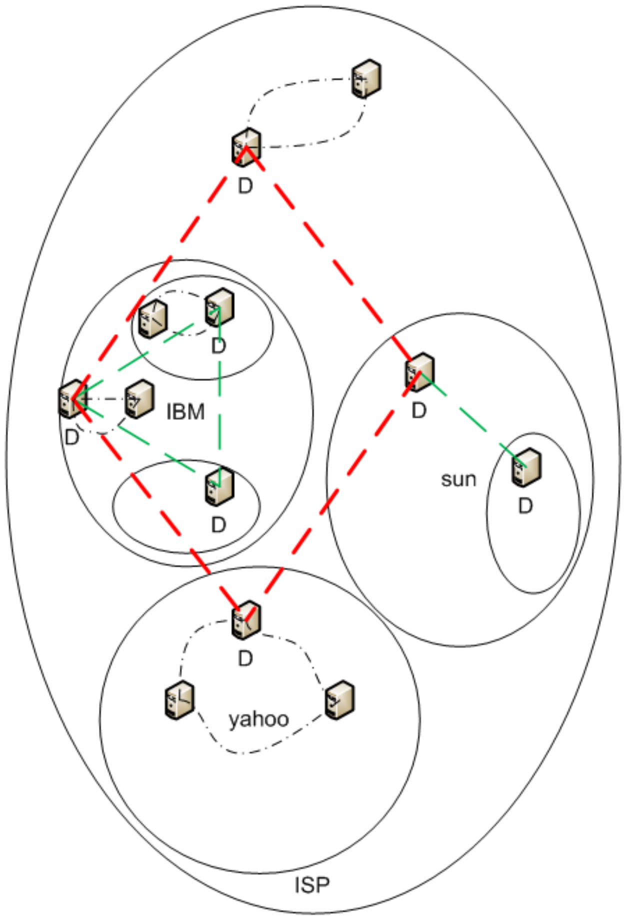

Figure 3: an general mDNS hierarchy

3.4 search example

Lets take an example hierarchy to see how search operation is performed in mDNS architecture.

Assume the search is initiated by an end host at the only subnet in the SUN network. Also assume that the search is global in nature. The search goes to the only $MSD_{SUN_{subnet}}$ server in the subnet which is also the designated MSD server. The $MSD^d_{SUN_{subnet}}$ server searches its local sessions database and returns all the hits to the requesting host's application.

The designated $MSD^d_{SUN_{subnet}}$ server generates a unique search query ID and propagates the search to both CMCAST and PMCAST channels. In this example network, CMCAST channel does not exist for the SUN internal subnet, so the search is propagated only on PMCAST channel (green dashed line). It reaches the SUN network wide MSD_{SUN} server, this outer MSD_{SUN} server searches its internal sessions database. Since the search came from a subnet host and not from a coexisting host, it only returns those results that are global in scope. MSD^d_{SUN} propagates the search on the PMCAST channel (red dashed line).

Now the search query reaches the MSD servers in the IBM i.e. MSD^d_{IBM}, Yahoo i.e. MSD^d_{Yahoo} and the ISP's TLD MSD server i.e. MSD^d_{ISP}. This is how the query eventually propagates to all MSD servers in the mDNS architecture.

4 High level analysis of mDNS

We believe that mDNS has features to take IP-Multicast to the next level of deployment in the general consumer network. Together with HOMA [10] architecture, mDNS has true potential to make multicast session discovery and deployment seamless and consumer friendly. mDNS URL scheme presented in section 3 would make bookmarking of popular multicast sessions feasible and equally userfriendly as webpage bookmarks in the current Internet.

But mDNS still being in early development stages has lots of drawbacks in its current form. It is specially vulnerable to DoS attacks. Also each global query has the potential to activate every MSD server deployed under mDNS scheme. Further more direct query results transmission to the querist from MSD servers creates the possibility of DDoS [6] attack on some remote host on the Internet using IP spoofing attacks.

There are several immediate benefits of the mDNS scheme. First among many is the database space savings. In LBL's sd and later sdr software implementation, possibly every sdr client may cache the sessions detail in the software's local cache. This makes the sdr scheme particularly not scalable. In mDNS scheme only the leaf subnet's / domain's MSD server in the hierarchy maintains the multicast session database. Also the receiver application does not have to wait 10s of minutes to discover a session because of SDP global bandwidth usage restriction. We conjecture that session search and discovery in mDNS scheme should be many orders of degree faster than current schemes. Benefits of URLs have already been discussed earlier.

5 Ongoing and future research goals

We are also considering subscription based approach where each MSD client in addition to maintaining locally residing session source database, also maintains keywords subscription level by hosts in its subnet / domain. We are trying to come up with a threshold based system to propagate the keyword subscription up the mDNS hierarchy.

One of the major concerns we have is activation of every MSD server in mDNS scheme for every global search. This could prove to be very taxing on MSD servers. We are trying to look into intelligent cache placements along with smart caching techniques in order to reduce the possible workload on MSD servers.

Security remains a major concern in any distributed deployment in the Internet today. mDNS in its current form is vulnerable to all kinds of attacks. We are also studying into the possible nature of threat to mDNS and possible solution to thwart those threats.

This paper provides a very high level description of mDNS architecture. Design of protocol messages and database records structure are not yet done. Implementing a prototype system would give the researchers first hand and real experience with mDNS. It would also allow us to analyze the real stress levels on MSD servers.

References

[1] Semantic multicast project. http://www.wins.hrl.com/projects/semcast/.

[2] K.C. Almeroth. The evolution of multicast: from the mbone to interdomain multicast to internet2 deployment. *Network, IEEE*, 14(1):10–20, Jan/Feb 2000.

[3] C. Mic Bowman, Peter B. Danzig, Darren R. Hardy, Udi Manber, and Michael F. Schwartz. The harvest information discovery and access system. *Computer Networks and ISDN Systems*, 28(1–2):119–125, December 1995.

[4] S. E. Deering. Multicast routing in internetworks and extended lans. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols*, pages 55–64, New York, NY, USA, 1988. ACM.

[5] Shlomi Dolev, Amos Israeli, and Shlomo Moran. Uniform dynamic self-stabilizing leader election.

IEEE Trans. Parallel Distrib. Syst., 8(4):424–440, 1997.

[6] L. Garber. Denial-of-service attacks rip the internet. *Computer*, 33(4):12–17, Apr 2000.

[7] M. Handley. The sdr session directory: An mbone conference scheduling and booking system. April 1996.

[8] M. Handley and V. Jacobson. Sdp: Session description protocol, 1997.

[9] M. Handley, C. Perkins, and E. Whelan. *Session Announcement Protocol*, 2000. Internet Engineering Task Force, RFC 2974.

[10] Piyush Harsh and Richard E. Newman. An overlay solution to ip-multicast address collision prevention. *IASTED EuroIMSA Conference Procedings*, Mar 17–19 2008.

[11] H. Holbrook and B. Cain. *Source-Specific Multicast for IP*, October 2003. Internet Engineering Task Force, Work in Progress.

[12] Pieter Liefooghe and Marnix Goosens. The next generation ip multicast session directory. *SCI, Orlando FL*, July 2003.

[13] C. Metz. Ip anycast point-to-(any) point communication. *Internet Computing, IEEE*, 6(2):94–98, Mar/Apr 2002.

[14] D. Meyer and P. Lothberg. *GLOP Addressing in 233/8*, 2001. Internet Engineering Task Force, RFC 3180.

[15] Mehdi Mirakhorli, Amir Azim Sharifloo, and Maghsoud Abbaspour. A novel method for leader election algorithm. In *CIT '07: Proceedings of the 7th IEEE International Conference on Computer and Information Technology*, pages 452–456, Washington, DC, USA, 2007. IEEE Computer Society.

[16] P. Mockapetris and K. J. Dunlap. Development of the domain name system. *SIGCOMM Comput. Commun. Rev.*, 18(4):123–133, 1988.

[17] P. Namburi and K. Sarac. Multicast session announcements on top of ssm. *Communications, 2004 IEEE International Conference on*, 3:1446–1450 Vol.3, 20-24 June 2004.

[18] Alexandre Santos, Joaquim Macedo, and Vasco Freitas. Towards multicast session directory services.

[19] N.R. Sturtevant, N. Tang, and L. Zhang. The information discovery graph: towards a scalable multimedia resource directory. *Internet Applications, 1999. IEEE Workshop on*, pages 72–79, Aug 1999.

[20] Andrew Swan, Steven McCanne, and Lawrence A. Rowe. Layered transmission and caching for the multicast session directory service. In *ACM Multimedia*, pages 119–128, 1998.