Inter-Domain Multicast: Edge Based Trees

Thesis submitted in partial fulfillment of the requirements for the degree of
Master in Computer Architecture, Networks and Systems

by

Florin-Tudorel Coras

Advisors:
Dr. Albert Cabellos-Aparicio
Dr. Jordi Domingo-Pascual

September 2011

Universitat Politècnica de Catalunya

Abstract

The rapid increase of Internet access speeds has lead to the proliferation of multimedia services which now constitute a significant portion of global inter-domain traffic. One such service is concurent content distribution to a large set of users. Due to the low uptake of inter-domain multicast, hampered by a plethora of reasons, content providers provision their services by means of application-layer overlays. However, as recent studies have shown, their streaming quality and scalability are not deterministic.

LISP is one of the incrementally deployable solutions proposed for the Internet's scalability problems. It aims to relax the pressure exerted by the edge- on core transit- networks through a semantic decoupling of *identity* and *location* at network level.

We propose a set of enhancements to the LISP protocol that aim to create a configurable *identity-layer* framework with native support for single sourced inter-domain multicast. Architecturally, our proposed overlay's characteristics fall between those of application- and network- layer multicasting schemes. As a result, the solution presents rapid reconfigurability and sensitivity to changing network performance metrics. But, at the same time, due to the strategic importance of the network equipment involved, it has intrinsic robustness. However, such benefits come at the cost of more stringent hardware constraints. We circumvent these restrictions through the centralization of multicast group management functions. By means of large scale simulations we compare several centralized algorithms and asses their deployability. Results show that the architecture can scale to thousands of participating *domains*.

Contents

List of Figures

Chapter 1

Introduction

1.1 Context

The sustained increase of network access speeds Internet users have experienced in recent years has lead to a proliferation of multimedia services that are now reportedly [1] among the largest and the fastest growing global bandwidth consumers. It is the advent of these technologies and their flexibility in interacting with content that has enabled and encouraged users to start transitioning from traditional broadcasting systems to Internet based IPTV or similar streaming services. Now, experts speculate [2] that IPTV revenues are to rise from less than USD 1B in 2006 and USD 6B in 2010 to USD 17B in 2016.

One such streaming service is the concurent delivery of content from a single source to a large group of receivers, or *source specific multicast* [3]. Typically, such dissemination technique is employed for real-time video streaming of sports events, news or popular broadcast channels and is most efficiently implemented by means of IP-multicast [4]. However, though seen as a panacea, multicast has never risen to the heights of its expectations for a multitude of reasons [5]. Initial work on MBone [6], an experimental global multicast infrastructure, failed to materialize in a clear service and protocol architecture that would have been easy to deploy, control and scale with the ever growing Internet. As a result, most ISPs had resisted multicast adoption by when a simplified, source-specific [3] solution had emerged. Nevertheless, by then, application layer solutions that obviate the need for network infrastructure changes had taken off and diminshed interest in the more efficient but harder and more expensive to deploy alternative. Yet, albeit the still to be solved lack of an inter-domain multicast infrastructure, source specific multicast has seen encouraging intra-domain adoption.

In light of IP-multicast's slow deployment, both industry and the research community have switched research focus to application layer multicasting and complex overlay management algorithms. Consequently, there are many academic [7, 8, 9, 10, 11] and commercial [12, 13, 14, 15, 16] products available. Some of them have enjoyed recent commercial success, and most notably, PPLive [13] is reported [17] to be used daily by millions of users. This obviously has led to an intense scrutiny of their performances and limitations [17, 18, 19].

Unfortunately, recent results [20] have brought forth limitations of the overlay architectures in scaling user quality of experience with the increase of client population. Reasons for such behavior have to do with unavailability of inter-peer bandwidth, inefficient supply of server (source) capacity or in some cases insufficient client upload capacity. The reactions to such observation can be twofold. On one hand, one would expect that such inconveniences can be alleviated through a better overlay architecture design. However, on the other, it should be acknowledged that hosts are not always permanent or at least stable members of the streaming architecture and their churn, their upload capacity heterogeneity and their position in edge networks combine as distinct limitations in a hard to come by optimization problem. If such concerns are not worrisome they should at least leave room for the possibility that better performing architectures may exist.

Independent of the multicast hindrance, the Internet has its own set of challenges and most recently, due to unexpectedly rapid and sizable growth, started facing increased scaling costs. Recently [21], industry and academia seem to have agreed that overloading of IP semantics, its identification of both *location* and *identity*, is to blame for the situation. Hence, improper usage of IP has left the core routing *vulnerable* to edge networks dynamics and as a result, has contributed to increased Default Free Zone routing tables. Currently, *Moore's Law* continues to assure a low priced operational cost, but to avoid a relatively near future Internet collapse, several improved architectures have been proposed [22]. Among them, LISP [23] aims to relax the pressure exerted by the edge- on core transit- networks through a semantic decoupling of *identity* and *location* at network level. It is one of the incrementally deployable solutions that, besides counting with support from both academia and industry [24], also relies on an experimental testbed [25] dedicated to its development. Its introduction of new routing mechanisms through the support of an additional indirection level, experimental status and possible future deployment makes it a suitable candidate for the implementation of improved multicast functionality.

1.2 Problem Formulation

In light of the IP-multicast deployment predicament, the application layer overlays' ostensible inability to perform deterministically and due to the recent development of LISP we have found it appropriate to propose a new inter-domain multicast architecture rooted in the still developing protocol.

Our goal is to semantically and syntactically support the already existing host implemented source specific multicast interface [3], yet we aim to create an easily deployable inter-domain packet distribution architecture with configurable performance. As a result, we limit the scope of our proposed changes to just the routers LISP's deployment is to upgrade and herein lies the solution to avoiding the prohibitive costs of deployment that affected previous architectures. Clearly, our use of LISP, as a native layer for inter-domain multicast, constrains the obtainable topological information and inherently the efficiency of the architecture relative to that of traditional IP-multicast. However, through a good design and with knowledge levereged from existing overlay routing research, we aspire to achieve an architecture that possesses:

- low deployment cost
- scalability
- configurability
- efficiency

The main obstacles to be circumvented are the limited *packet replication factor* and *usable computational power* present in equipment that was mainly developed for high throughput datagram routing, but part of whose packet processing operations now have to be altered. Moreover, for an easy adoption of the proposed extensions, we intend to minimize the number of changes to the current LISP specification.

Chapter 2

Background

2.1 Locator/Identifier Separation principle

2.1.1 Endpoints and Endpoint Names

In his paper [26], Noel Chiappa introduces the concept of Endpoint to solve what he belives to be an overloading of the name functionality in the context of networking. There are few fundamental objects in networking, also few names and, among these, good examples are, *host names* and *addresses*. He has reached the conclusion that the reasons for this situation are twofold: first, negligence. This goes back to the earliest of papers in networking when the authors were not careful to distinguish between the concepts of an object and their associated names. This has caused widespread confusion between the properties of the object and those of the name. The second reason would be the lack of a rich set of fundamental objects. When dealing with new problems, difficulties were encountered in finding/acknowledging the status of separate entities for previously existing, but *masked* objects.

In the days of the ARPANET the *address* had a straightforward meaning and was build by concatenating the number of the host with a port number. But as the scale of the internet has grown, the *tight* association between the functions of the term *address* and this sole *instantiation* of the name has stemed confusion. Chiappa and Saltzer [27] explain this by the small number of well defined concepts at hand.

To set asside the confusion it was proposed that the term *address* should be redefined and used with one of the current implied uses and be limited to this particular one. Further, the fundamental object, the *endpoint* is defined. In fact, it is acknowledged as previously present in the network but unnamed.

For a better understanding of the technical aspects, the author also definies bindings and namespaces. The binding are the associations between

a name and an object, but they may also map from one *kind* of name to another. Furthermore, instances of the same object may have more than one name and these names may be from the same class of names , or otherwise *namespace*. Depending on the namespaces we may have many-to-one bindings or alternatively many one-to-one bindings.

The relation between the structure of the names in a namespace and the function is also observed. This may be explained by the need for ease of use and a good example would be the structure implied by the IP addresses in order to ease routing. It is also obvious that names may have multiple ways of representation (eg. decimal printed notations and the bit representation for IPs).

The namespaces, togheter with the possibility to represent names in multiple ways, imply the existance of contexts which may help make the distinction between attachment of names to objects and mappings from one namespace to another.

All the above theoretical introduction can be used now to analyze the TCP/IP Architecture which did not make a clear distinction between the objects and the names of those objects. In fact it can be observed that the namespaces are as old as the NCP prococol architecture, namely the addresses and the host-names.

IP addresses are in fact the only *names* used throughout the TCP/IP architecture but they have multiple uses:

- Used by the routers when forwarding the user data packets
- Used to name a place in the network, the destination of a packet. Otherwise known as the *network attachment point* or *interface*
- Used in transport layer identifiers for both of the end-to-end communicating hosts.

The overloading of this single name and *field* with all these functionalities is not for the best and a soulution would be to split these three functions up and have them be performed by separate fields.

The other TCP/IP architecture namespace is that of host-names. They are human readable strings and contained in a hierarchical structure and can be looked up in the distributed and replicated database we all know as the DNS (Domain Name System). But their goal is, in fact, to map between the human readable characters and one or several IP addresses which will mediate the TCP *conversation* between the hosts, once determined.

As expected, the double function of the IP addresses, that of identifying the interfaces and the hosts has downsides and an important one is the

limitations of host mobility. This is because a TCP connection, as already mentioned, is always identified by the pair IP address and TCP port. It only follows that a change of the IP address , requested by the change of the position in the internetwork, will brake the TCP connection.

Chiappa tried to solve this problem by proposing a better definition for the term *address*, in fact new definitions, or better bounded ones, for all three possible meanings. He suggests using *address* when referring to an interface, selector when talking about the field used by routers when forwarding packets and introduces a new fundamental object, the endpoint, to identify an end-to-end communicating host. He was unable, though, to give the endpoint namespace/namespaces because their uses are multiple, and as stated in the begining of this section, the forms of names within a namespace are highly dependent on how they will be used. But he did give a rich list of characteristics which he deemed as useful.

In the current times, his ideas are somehow "woken up from slumber" to help solve the routing scalability problem by splitting the different functionalities of the term *address*: that of locator in the core routing system (selector) and endpoint identifier (the host interface).

2.1.2 Initial Proposals

Address rewriting

The ideea was originally proposed by Dave Clark and later by Mike O'Dell in his 8+8/GSE [28] specification. The aim was to take advantage of the 16-byte IPv6 address and use the lower 8 bytes as End System Designator(ESD), the top 6 bytes as a routing locator (*Routing Goop* or RG) and the ones left in between, 2 bytes, as Site Topology Partition (STP).

The model draws a strong distinction between the transit structure of the Internet and a Site that may contain a rich but private topology which may not *leak* into the global routing domain. Also the Site is defined as the fundamental unit of attachment to the global routing system, being in fact a leaf even if it is multihomed. But the above mentioned structure of the address brings also the desired distinction between the identity of end system and its point of attachment to the *Public Topology*. O'Dell also observed the overloading of the IP semantics and the consquences it has on address assignment when topology changes are done.

The most important part of the proposal, and which in fact insulates Sites from the global routing system, is the rewriting of the RG by the Site Border Routers. In this sense, when generating a packet, the source host fills in the destination address with the complete 16-byte IPv6 destination

address, RG included, that it receives through DNS resolution, and fills the source address's RG with a *site local prefix*. Now, when a packet destined for a remote host arrives at the local site egress router, it has its source RG filled in to form a 16-byte address. On the other hand, when a packet reaches the destination site's ingress router the RG is stripped off and replaced with a *site local prefix* to keep the local hosts and routers from knowing the domain's RG. The obvious result of this decision is that upper-layer protocols must use only the ESD for end point identification, pseudo-header checksums and the like.

The above mentioned insulation provides a site with flexibility of re-homing and multihoming. And this is because a site's interior hosts and routers are unaware of the RG and thus if a change in the RG, due to administrative decisions, does occur, the interior components need not know it. Moreover, this brings forth the possibility of topological aggregation, with the goal of routing scalability, by partitioning the Internet into what O'Dell named as "set of tree-shaped regions anchored by 'Large Structures'". The Routing Goop, in an address, would have the purpose of specifying the path from the root of the tree, or otherwise the *Large Structure*, to any point in the topology. In the terminal case that point would be a Site. Thess *Large structures*, thus have the goal of aggregating the topology by relational subdivision of the space under them and delegation. It also follows that in the case when no information about next hop is known, the Large Structures could be used as forwarding agents. This will significantly limit the minimally-sufficient information required for a router when doing forwarding. It was also envisaged that additional route information kept is the result of path optimizations from cut-throughs. This has been proven as a wrong observation and will be detailed when treating the limitations of this system.

For further details related to the structures within the IPv6 address and also possible solutions to re-homing and multihoming, reading the draft RCF [28] is highly recommended.

Given the age (this solution has been suggested in 1997) and also the lack of solutions, at the time, for some of the component proposals, it is only natural that today we see limitations with this design and in what follows some of them will be detailed. Good overviews of this system and its limitations are made by [29, 30].

The main flaw in GSE's design seems to be the use of DNS when learning about destination hosts. Even if one assumes that root servers will stay relatively stable, it must also accept that the ones under will not. And if considering that a site is multihomed, which and how many of its RG should be returned as reply to a DNS server lookup for that site? Furthermore, given its role, a DNS server must at all time know the RG of the site it

currently resides in such that a proper answer can be given for DNS queries. This comes to contradiction with the above stated *insulation principle*. In addition, the support for 2-faced DNS server is brought up, that is, the server must know if the query is remote or local site in nature in order to know if the RG should be included or not in the reply message.

Another issue, is handling border link failures. It is possible for the source site to be aware of the status of its border links and choose the one of those which are up, but at this point in the path followed by a packet from source to destination, it is imposible to determine if one of the border routers at the destination has lost connectivity to the site. Thus, as a solution, GSE propsed that the border routers for a site be manually configured to form a group and when one loses connectivity to the client site, it should forward the packet to others still connected. It should be noted that this is not an issue specific to GSE but with all solutions which propose a split between the edges and transit routing domains.

It was anticipated above that the Large Structure anchorage of the tree shaped regions, with little or none interconnection between the lower regions was a wrong hunch. And indeed it is, as the trend in the last ten or so years, has shown that the interconnections below the top level are the norm rather than controlled circumventions, thus the proposed RG structure my need revisiting.

Also, though it has scalable support for multihoming, GSE lacks support for traffic engineering. It may be possible for it to solve this goal but the existing proposal does not solve this problem. Same holds true for IP tunneling across RG boundaries and, given the extensive use of Virtual Private Networks (VPN), a thourough examination of tunneling operations is needed in the GSE context.

Map-and-Encap

The idea, as originally proposed by Robert Hinden in his ENCAP scheme [31], speaks about splitting the current single address space in two separate ones: the EID space, which covers the hosts, and the one used for transit between the domains, the RLOC space. It is belived that, through the decoupling of the EID, non topological aggregatable space, from the RLOC, provider owned space, aggregation can be achieved in the core routing domain and eventually routing scalability.

Whenever a source initiates communication with a destination host outside of its local domain, it generates a packet that first traverses the domain's network infrastructure and reaches a border router. The datagram has as source address the EID of the initiatior and as destination address the identi-

fier of the peer host, that could have been obtained by means of DNS. Next, the border router *maps* the destination EID to a RLOC, or an entry point in the destination network, by means of a mapping system. Once the mapping is obtained, the border router *encapsulates* the packet by prepending it with an outer header that carries as destination the obtained locator, and then proceeds to injecting the resulting packet in global routing system.

Once the encapsulated packet reaches the destination site, the border router proceeds to its decapsulation. The resulting datagram, identical to the one generated by the source, is then forwarded to the destination host part of the local domain. It should be observed that within both source and destination domains the EIDs must be routable (but their scope is local).

Besides the obvious architectural improvements that may help solve the current routing scalability problems other advantages of the map-and-encap solution are its lack of host stack and core routing infrastructure changes. Furthermore, this scheme works with both IPv4 and IPv6 addresses and retains the original source identifier, a feature useful in various filtering scenarious [32]

As downsides, this model, as the address rewrinting one, has problems in handling border link failures and the overhead implied by the encapsulation is stemming controversy.

Both of the above presented models, that have inspired new solutions in our current context, seem to arguably provide, in a sketched manner, ways to solving the routing scalability problem. But in doing so, given their reliance on the addition of a new level of indirection to the routing architecture, practically their intrinsec need to translate EIDs into RLOCs, they have created a new problem and the solution to it is the mapping system. The scalability problem has now shifted from the routing system to the mapping system and the success or failure of future solution will heavily depend uppon the careful design of the mapping architecture.

Finally, besides healing old but relevant routing problems the new indirection level allows us to be more creative when improving or developing new network services. In this sense, we feel that it is the semantically correct layer and it has the appropriate syntax to deal with multicast routing.

2.2 LISP

In October 2006 the Internet Advisory Board (IAB) held a Routing and Addressing Workshop in Amsterdam, Netherlands with the goal of developing a shared understanding of the problems that the large backbone operators are facing regarding the scalability of the Internet routing system. The outcome

of the meeting, their findings and suggestions, have been summed up in a Request For Comments (RFC 4984)[21] and forms the input to the Internet Engineering Task Force (IETF) community which aims to identify the next steps towards effective solutions.

While many aspects of a routing and addressing system were discussed, the participants deemed two as most important and subsequently formulated two problem statements:

- Problem 1: The scalability of the Routing System
- Problem 2: The Overloading of the IP Address Semantics

An in depth analysis of the problems (including the two above) affecting the routing system was also provided.

The Locator/Identifier Separation Protocol [23] came as a response to the Workshop's problem statement and aimed, as previously mentioned solutions and their derivatives, to solve the scalability of the routing system. One of the conclusions of the workshop was that any solution to the routing scalability is necessarily a cost/benefit tradeoff thus, given the high potential for gains of the indirection approach (a locator/identifier split), the map-and-encap ideea was regarded as one of the most promising solution spaces.

The LISP draft focuses on a router based solution and proposes an incremental deployment of the technology. As with map-and-encap there will be no changes to the hosts stacks and to the majority of the routers within an AS but it will be necessary to deploy some new equipment namely Ingress Tunnel Routers (ITRs) and Egress Tunnel Routers(ETRs) and possibly to develop and deploy a new Mapping System. The role of a site ITR is to find by means of the Mapping System the destination locator for a local site outgoing packet, construct and prepend a LISP header to the original IP datagram, and finally direct and sent the resulting packet to the just discovered ETR. The ETR's goal is to strip down the LISP header, if the received packet has as destination address one of the ETR's locators, and forward the resulting packet to the destination EID, whithin its local site (see figure 2.1). This approach also allows the possibility of implementing traffic engineering and multihoming in a scalable fashion.

Through its architecture, LISP seems to solve the main problems affecting the current routing system and offers itself as a medium term solution, until new, innovative architectures will be developed and finaly deployed. But before we can talk about a LISP *Default Free Zone* more research is needed, at least for the Mapping System part of the LISP architecture because, as in the case of past proposals, the solution itself, though elegant, introduces

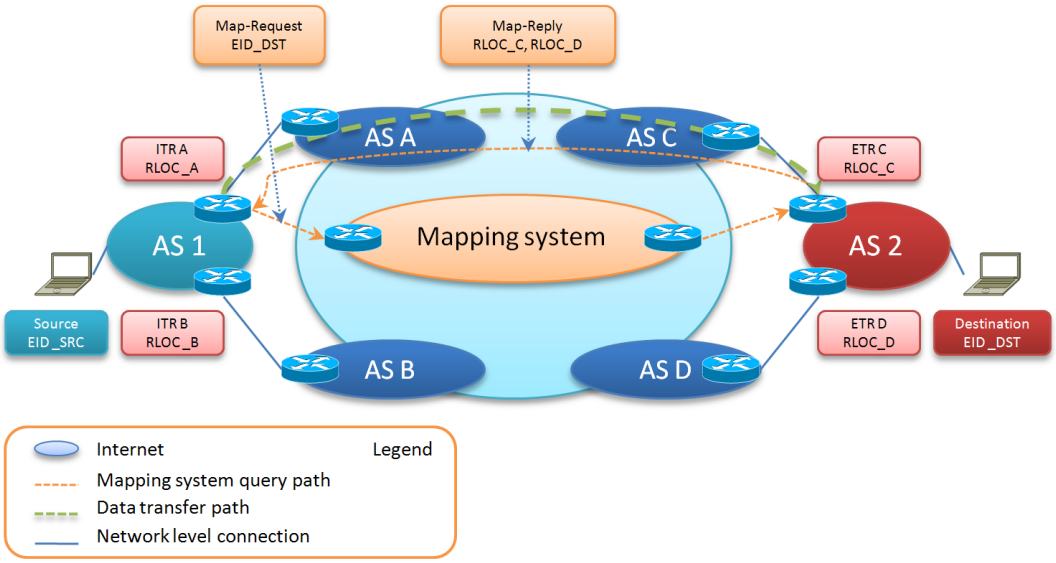

Figure 2.1: LISP architecture

new scalability problems. Considerable effort is being invested into finding what would be the best suited Mapping System both in terms of lookup latency and scalability. Among the currently proposed system are: ALT[33], CONS[34], NERD[35], DHT[36] and TREE[37].

LISP packet formats can be checked in [23]. For an in-depth understanding of the data and control plane implementation details or how deployment and interworking with legacy solutions are to be done, checking the the IETF working group charter page [24] is recommended. In what follows we briefly detail protocol and architectural aspects that are related with our multicast extensions.

2.2.1 Mapping System Interface

In order to support fast and easy deployment of new mapping systems architectures, the interactions with the LISP control plane are done through specifically designed interfaces [38]. Logically, they instantiate into two functions, one used for querrying about- and another for registering EID addressed. The former is known as the *Map-Resolver* function and the latter as *Map-Register* function. Independent of the algorithm implemented by the mapping system for destination discovery/reachability, network devices called *Map-Servers* are expected to receive on their client-facing interface *Map-Register* messages from ETRs. Whereas, *Map-Resolvers* are expected

to receive *Map-Request* messages from ITRs. Map-Register messages carry LISP reachability information for a set of EID prefixes and Map-Request messages query for the locations of EIDs. Such requests are forwarded to the mapping system where devices part of the architecture conspire to deliver the Map-Request to the Map-Server and, subsequently to the ETR announcing reachability of a prefix encompassing the requested EID. A peculiarity of the messages ingressing and egressing the mapping system is that they are *encapsulated* in a supplementary LISP header such that they may be routed in RLOC space. To be noted that noting precludes the implementation of both functions, Map-Request and Map-Register, by one single device which then can act as both Map-Server and Map-Resolver.

Additionally, a Map-Resolver may be configured to work as a caching resolver when it must save state for all ongoing EID resolutions and initiate Map-Requests in the name of its clients. All mapping results are cached, before being forwarded to the ITRs, for future reuse, which can aid in reducing mapping latency for clients but such practice interferes with inbound traffic engineering. The destination ETR can not see the address of the requester, only that of the Map-Resolver, and thus it can not tailor its responses based on its peer's identity.

Also, a Map-Server may be configured to perform *proxy map replying*, when, instead of forwarding a Map-Reply to the authoritative ETR, it generates an non-authoritative reply and forwards it to the requester. This simplifies the ETR's job in LISP's control plane and allows it to reach all LISP destinations without running BGP. Further, this leaves open the possiblity that the Map-Server performs additional traffic-engineering tasks and the like for the ETR.

2.2.2 Locator Reachability

One of the most important problems affecting all Loc/ID adopting architectures is the locator reachability problem[39]. It consists in the impossibility of knowing apriori if a path to a destination would be functional. Specifically, for LISP, an ITR may end up choosing to encapsulate towards an RLOC that has either lost Internet connectivity, if BGP has yet to propagate this loss of connectivity, or one that lost connectivity to the site it serves. To diminish the probability of encountering such situations, LISP makes use of several active and passive mechanism for determining locator reachability [23]. Among them, the *locator-status-bits*, present in the LISP header, are used by an ITR to indicate to its peer ETR the up/down status of all locators in the site of the sending host. This, together with BGP reachability information, if present, allows the ETR, if it also acts like the

site's ITR, to make an informed choice among the peer's possible locators. Furthermore, for bidirectional flows, an ITR can actively probe the forward and return paths to its peer ETR through a data-path algorithm known as *echo-noncing*, when an ETR is requested to *echo* back a 24-bit *nonce*. A non-echoed nonce is an indicator that the path is not usable.

At the cost of increased control traffic, an ITR may periodically use directed Map-Requests with the probe bit set in the packet's header to assess the reachability of certain locators. The procedure, named *RLOC probing*, besides enabling the ITR to discard the unreachable destination locators, also provides RTT estimates for the active ones.

2.3 IP Multicast

Multicast, as a general concept, is a transmission technique that delivers a message to a group of receivers in a single transmission with packet replication being done by on path elements. IP multicast [4] is an instantiation of the concept where the delivered contents are IP datagrams. It may take the shape of a *one-to-many* delivery, when the multicast group members are arranged in a source routed tree, or *many-to-many* delivery, when the members form a graph where all nodes can be both sources and listeners. Data plane scalability with multicast host group size is ensured by design as a source sends a packet just once, replication duties falling to the network. Control plane scalability is the result of no requirement for prior knowledge of the multicast's group size or member addresses.

For an efficient implementation of the IP multicast service, new routing protocols and management mechanisms were needed. The latter request has resulted in the development of a new protocol, the Internet Group Management Protocol (IGMP) [40], that can be used by IPv4 hosts to express their multicast group memberships to any neighboring multicast routers. The IPv6 equivalent of IGMP is Multicast Listener Discovery (MLD) [41]. Regarding the first request, since multicast's inception, several protocols were developed both for intra- and inter- domain multicast routing [42, 43, 44, 45, 46, 3, 47]. Many of them are extensions to established unicast routing protocols while others, like those pertaining to the Protocol Independent Multicast (PIM) family, rely on the topology discovery mechanism of their underlying unicast routing protocols. Independently of how multicast paths are discovered or computed, PIM protocols specifications center on tree management specifics and data path forwarding details. Still, a trait common to all multicast routing protocols is their employing of *reverse path forwarding* for routing loops avoidance. In spite of coordinated efforts to create an IP multicast backbone,

[6] none of the protocols got to see wide scale, inter-domain, deployment for both technical and economical reasons [5]. Nevertheless, reports seem to suggest that PIM Sparse Mode (PIM-SM) has seen the largest of deployments up to date. It is also the protocol that LISP's multicast extension builds upon [48].

A third implementation requirement for IP multicast was the necessity to distinguish unicast from multicast addressing. Consequently, the IPv4 (224/4) and IPv6 (FF00::/8) ranges have been set asside by IANA for multicast addressing. It is therefore required that all devices recognize the syntax of the two IP address ranges. Even so, functional compliance requires that such devices also recognize the address semantics and implement the necessary protocols.

Reviewing all IP multicast routing protocols is out of the scope of this document. However, due to functional similarities between the inter-domain multicast solution discussed in section 3.1 and PIM Source Specific Multicast, a particular case of PIM-SM, we will present in what follows a brief review of PIM-SSM's characteristics.

2.3.1 SSM

Source-specific multicast, as defined in [49] and [3], is a particularization and simplification of the more general Any-Source Multicast (ASM) service model [4] obtained by limiting the number of multicast sources to one. This requires the constraining of the SSM service interface to a *channel* identified by a source S and a host group destination G pair. The service is well suited to dissemination style applications where the listeners are aware of the sender's identity prior to the data stream's start. It also aims, as explained lower, to simplify multicast delivery architecture by requiring a much simpler management mechanism.

To distinguish SSM from other multicast destinations, IANA has set aside IPv4 232/8 and IPv6 FF3x::/96 ranges from their respective multicast allocated address sets. Further, the policy for allocating non reserved SSM addresses to sending applications is strictly determined by the local host. As a result, G can be thought of acting like a discriminator among the multiple channels host S might source. This is a reduction of scope if comparing source specific multicast group addresses to globally unique multicast identifiers. The semantics of SSM addresses are also slightly changed from those of ASM addresses and suppose the delivery of datagrams destined to channel address G and sourced by S to all processes or *sockets* and only the ones that have specifically requested (subscribed to) such content. This is to be contrasted with the more complex ASM service which in the simplest of sce-

narios, when no source filters are used, delivers packets from all sources to all the G address listeners. Moreover, the ASM architecture complicates further when providing an inter-domain multicast service by means of PIM-SM as it requires additional protocols for source and randezvous points (RP) discovery, like Multicast Source Discovery Protocol (MSDP) [50]. However, to provide the same service, SSM only requires a subset of ASM's router mechanism for the source routed shortest-path tree content distribution.

As in the case of ASM, the receiver set is unknown to the SSM source and dynamical. The channel's identity, the pair (S,G), would be typically discovered by the application protocol that could subsequently request its underlying network layer to subscribe to the newfound channel. The host's network stack may use IGMP to inform the upstream router of its multicast interest. This designated router, in the eventuality that it is not yet subscribed to the channel of interest, should proceed by sending a source specific PIM join towards the channel's source. The act would typically result in a valid reverse path and a multicast packet flow from S to the requesting device which can now replicate the packet stream towards the requesting host.

2.4 Aplication Layer Multicast

In light of slow inter-domain IP multicast deployments [5] and with the advent of streaming and conferencing applications, research efforts have switched focus from network equipment to end-hosts and application layer overlays. Though more inefficient in the use of network resources, overlays require no network infrastructure changes and implement all multicast functionality in the end-hosts. Therefore, member multicast forwarding functions consist of local replication of content and its exchange with peers by means of unicast. However, lack of topological information and client dynamics or *churn* complicate the control plane operations and scaling. Extensive academic research [7, 8, 9, 10, 11] was carried to design an efficient group management protocol with good scaling properties, efficient network usage and resiliance to client churn. Moreover, these efforts were supplemented by those of the industry with a set of commercial applications [12, 13, 15, 14, 16] that are now enjoying world spread usage. Yet, some measurements [20] have revealed shortcomings in the ability of overlays to maintain the user quality of experience with the increase of client population. Explanations for such behavior have to do with unavailability of inter-peer bandwidth, inefficient supply of server (source) capacity or, in some cases, insufficient client upload capacity. These can be circumvented only through an improved control plane design and thus leave open, on one hand, the search for better architectures and,

on the other, some worrying questions about the costs and complexity of improved solutions.

We shall not detail all the proposed architectures but limit ourselves to inviting the courious reader to start by checking some of the surveys [51, 52] dealing with application layer multicast systems and their approach to overlay routing. Still, due to the dependence of our solution on Scalable Application Layer Multicast [7], we provide a short review of it in what follows.

2.4.1 Scalable Application Layer Multicast

The NICE application-layer multicast protocol, proposed in [7], works by arranging the set of participating hosts in a logical hierarchy. The creation and maintenance of the hierarchy are in fact the main functions of the protocol. Reasons for its importance have to do with control plane scalability and the establishment of data plane paths. The members, besides being organized in layers, are also clustered based on a proximity metric which for NICE is latency. For each cluster, the node with the minimum latency to all other, or the topological center, is chosen as head and representant in a cluster part of the logical layer above. Accepted values for cluster size are in the range k and $3k - 1$, where k is a design constant. Growth beyond the upper limit results in a cluster split and the shrinkage under the lower one in a cluster merge. The first event supposes the split of the oversized cluster and thus creation of two new ones. Elections for cluster head should be held in both. The second event results in the dissolution of a cluster through the merger of its members with another group at the same layer and the removal of its head from the superior layer. Also, another design requirement is that all participating hosts are at least members of the lowest level layer.

Both the control and data plane paths are defined over the resulting topology. Control traffic is exchanged between nodes in the same cluster and as a result a node member of multiple clusters peers on the control plane with all the members of clusters it belongs to. Further, all nodes keep state about the members of the cluster their elected head belongs to. These exchange are enough to determine cluster membership changes and enable fast restoration in case of node failure. Moreover, they act as a member and topology discovery mechanism. In order to avoid loops, data packets follow a tree topology where, starting at the highest level, each node that receives the packet has to replicate it to all the members of the clusters it heads. This does result in high fanout for nodes members of the higher levels but they also benefit of a lower latency to the source.

All overlay operations have to maintain the hiearchical organization and

the clustering rules. Therefore, a host join requires the discovery of the appropriate low level cluster the host can become a member of. The procedure starts at the root of the hierarchy and recursively descends on the path formed by nodes towards which the newcomer has the lowest latency. Thus, given the possiblity to choose between multiple nodes, into whose clusters to descend, the algorithm will aways favor that closest regarding latency. The departure of a host not head of a cluster does not require any action. However, the departure of one that is head of one or more clusters requires head elections to be performed in all of the affected ones.

The defined mechanism results in a low overhead hierarchical control structure that can accommodate different data path distribution trees.

Chapter 3

Design

3.1 Proposed Solution

In this section we present basic architectural details of our inter-domain multicast solution and expand on its high level management functions. But before we dive into the details, we shortly present our design assumptions.

A fundamental constraint, that our architecture shares with those of LISP, is that host stacks are not to be changed. As a result, any interaction of a host with the content streaming architecture should be done by means of already deployed protocols with a high acceptance rate in mainstream operating systems. Consequently, given the nature of our solution, we expect participating domains, besides implementing our LISP extensions, to be providing their clients with a network layer multicast service interface. Alternatively, an application layer multicast proxying service may be implemented but with the obvious scalability and inefficient network usage drawbacks when used in conjunction with a large intra-domain client population. Intra domain multicast implementation allows the nodes to use IGMP for signalling their multicast interests. However, due to our reliance on source specific multicast address space, we have as an additional constraint that hosts must implement the third version [40] of the group management protocol.

Regarding the inter-domain multicast architecture, we have opted for the centralization of group management functions as a means to avoid increasing domain border routers computational overhead. Hence, all data-paths are computed and disseminated to all overlay members by a central node. Due to the strategical importance of border routers for a domain's Internet connectivity, we assert that a LISP tunnel router is much less probable to suffer from network connectivity disruptions than an end-host and thus, its departure from the overlay for such reasons can be considered a seldom event.

Further, by virtue of a router's persistance in the network after leaving a multicast overlay we require all members to announce their leaving intentions by means of a *leave message* and cease performing their data path duties only once an *acknowledgement* was received. We shall refer to this procedure as a *graceful leave*.

3.1.1 Over the Core Overlay

The introduction of LISP presents us with the rare opportunity of enhancing the *networks's* abilities by absorbing functionality that was prior encompassed, from architectural standpoint at least, by the application layer. In this sense, we adopt ideas from p2p overlay literature but adapt them to both the information exposed and the set of requirements posed by network-layer equipment. By design, LISP requires upgrading only a small percentage of deployed routers and the commonly held opinion [53] is that they should be those pertaining to the edge networks. As a consequence, in a LISP enabled Internet future, it should be commonplace to encounter LISP aware domain border routers. Naturally, these routers are the equipment whose functionality we plan to enhance. However, as previously mentioned in section 3.1, in order to support fast integration of our ideas into the LISP protocol specification we took on an additional constraint, that of limiting the changes to what LISP is today. As a result, we avoided introducing unwarranted functionality in the routers and tried to simplify their participation in the overlay's control plane. This lead to their almost complete exclusion from the multicast group management functions which, in our design, are performed by the Mapping Server (MS). The MS has a *privileged* position as, generally, any communication with the autonomous system it serves starts through it. Hence, for multicast traffic requests, it can perform authentication and authorization functions and subsequently control the joining node's behavior and location in the overlay. In a sense, its position resembles that of a p2p tracker but it can be much more involved in optimizing the delivery tree and implicitly in establishing inter-peer relationships.

Despite being in complete control of the overlay's hierarchical organization, the MS plays no role on the data path. The streaming's source border router heads this delivery tree and is the first to replicate packets. All other members, except the leafs, are obliged to replicate their received stream up to a certain preset limit.

The architecture is designed as a multipurpose framework that, on the one hand, follows the principles and constraints presented in in section 1.2 and, on the other, is configurable for specific operational needs. As a consequence, invariants like the high level functions used in the overlay hierarchy

construction, maintanance and dissolution are split and presented separately from protocol specification or overlay specific management functions. In what follows we decribe the former. The remaing two are to be dealth with in sections 3.3 and 3.2.

3.1.2 Member Subscription

In accordance with [3], a source specific multicast (SSM) stream or *channel* is identified by a pair consisting of a source address S and a multicast destination address G. S is the address of the channel's source, the content streamer, whereas G is the multicast host group address pertaining to the preallocated IPv4 and IPv6 SSM ranges [54, 55]. Despite bearing the syntax of IP addresses, G does not overload *location* and *identity* since it semantically identifies a group of hosts without conveying any topological information. Still, *location* information is maintained by S and as a result the pair *(S,G)* needs to be mapped, when in LISP context, to a *multicast location*. To reiterate, G serves as selector among the multiple channels sourced by the host at address S and it calls, when using LISP, for the use of a *multicast mapping function*. Indeed, S and *(S,G)* will generally map differently. For ease of reference, for the remainder of the paper, we shall refer to the pair *(S,G)* as multicast channel identidier or *MCID*.

Whenver a client, part of a domain not yet member of the overlay, sends a request for the streamed content, it triggers a Map-Request for the MCID on it's domain tunnel border router (ITR). The request propagates through the mapping system up to the entity authorized to provide answers for a prefix which encompasses S, the source's EID. This device may be the Map Server or the border tunnel router (ETR) for the stream's source domain. However, because overlay group management functions may be computationally intensive, we have decided against adding them to the ETR, which is typically a CPE router. Instead, we found them appropriate for the MS, that should be quite similar hardware wise to a general purpose PC. After the MS receives the Map-Request it performs a set of LISP security checks [56] to ensure that the request is genuine. If the tests are successfully passed, a search for an overlay parent with spare capacity is started. The search for the parent may be done randomly or, if additional topological information exists, in accordance to a predefined heuristic. Once found, it is sent a message requiring it to forward traffic to the joining domain's xTR. The process ends with an acknowledgement message being sent to the joing xTR.

Note that, if additional application layer security checks need to be performed, they should be done prior to adding the domain to the overlay. Such concerns are out of the scope of the current document.

3.1.3 Tree Optimizations

Often the overlay will be constrained to the minimization of a metric meant to improve content distribution efficiency. For instance, it may be required to minimize the latency between the source and all the clients for delay sensitive applications. Or, in order to efficiently use the network, it may be necessary to minimize the maximum number of times a link is used. Unfortunately, such constraints, combined with the restricted replication factor imposed for the tree member routers, usually give rise the NP-complete problems. Moreover, the problem gains in difficulty if even more constraints to how the hierarchy should be built are added. As a result, the joining process, even when complex heuristics are employed, will frequently lead to suboptimal distribution trees. Consequently, the tree, or if its size is a concern, only part of its branches, should be reshaped periodically or whenever the tree is supposed to be unballanced. Nontheless, such a procedure might result in a completely different distribution overlay with many of the previous data path links broken. Not done properly this could give rise to sustained packet loss. Therefore, in order to assure a seamless transition, a 'make before break' approach is used. Before breaking the data path links the Map Server configures the nodes to form a virtual topology that mimics the final one. However up to this point in time packets still follow the unoptimized paths. Once the virtual topology is finished the Map Server asks the tree's parent, whose position in the overlay has not changed, to switch to the new configuration. To mark the change, and subsequently force all downstream clients to use the new distribution tree, a bit in the LISP data packet header is set.

3.1.4 Member Unsubscription

In order to handle a node's departure we make use of the *graceful leave* procedure described in section 3.1. Any node wanting to leave its duties in the multicast tree needs to first inform the MS and once a confirmation is received effectively stop replicating traffic for the tree branch it servers. This message exchange offers the Map Server the chance to efficiently reorganize the nodes on the affected branch prior to acknowledging the node's departure. As in the case of tree optimizations, not to generate packet loss, a virtual topology is first created. Only after its initialization the departure is confirmed.

In the event that a node loses network connectivity, its data-path children will sense the failure either as a lack of data packets or by means of probing, should they know the locators of their parent. Hence, after a relatively short time period, they can inform the MS about the occurrence who will act as in

the event of a graceful leave. Still, such circumstances will result in packet loss for all members of the subtree headed by the affected node and out of band mechanisms would be required for remedying the failure. However, sudden loss of network connectivity for a domain's border router is a seldom occurance.

3.2 Overlay Management

Section 3.1 gave an overview of the group management mechanisms, or the LISP streaming interface, that needs to be implemented by compliant devices but avoided discussing about the exact topology discovery and tree building protocols. Both of them are to be detailed in this section. Nevertheless, they are presented in terms of abstract functions, without making use of LISP specific protocol messages, which are to be discussed in section 3.3.

The topological position, in Internet context, of the overlay members and the layer at which the protocol is being run makes it such that any member process sees any other peer as directly connected. In other words, the aggregated *view* of the member processes in an N node overlay, is that of a connected graph with $N(N-1)/2$ links. Obviously, this information is insufficient when optimizing the delivery tree and thus trying to decide a node's overlay position relative to its peers and the source. The problem is not specific to our design but one also encountered in application layer overlays (see section 2.4) and a common source of inneficient network usage for the two architectures when compared to IP multicast delivery. The solution is the employment of passive or active measurements between the members in order to quantify inter-peer distances with respect to a chosen metric or multiple ones.

As explained in the previous section 3.1, the group management functions are performed by the source's MS and thus it is also the device that needs to obtain member topological information. Therefore, depending on the type of information sought, both the Map Server and the overlay members will need to implement an interface capable of conveying measurement requests as well as their results. Consequently, the MS will be able to require coordinated measurements among a subset of peers, obtain the information and subsequently use it in the algorithms it needs to run. More details regarding the matter can be found in section 3.3.

It will often be required to optimize the multicast content delivery to a set of users with respect to a certain metric (or metric set) of importance either for the quality of experience of the streamed content or for the total delivery cost. Hence, it could be required to minimize the largest overlay delay to

the source, for real time content streaming, or the maximum number of AS hops passed in the delivery of one datagram to all clients, when efficient use of network resources is a concern. Therefore, optimization requirements are application specific and thus the number of possible optimization functions can be quite large. We limit our experiments to just two, those given as example above, but we note that many more could be implemented. Still, a common constraint for all optimization functions is the one limiting the node degree. It is a direct consequence of the restrictions imposed to AS border routers which, for performance reasons, are not to be exposed to large packet replication factors. To elaborate, for IP-multicast, a border router may be required to replicate a packet out all of its interfaces, but only once, however, in our solution or application layer overlays a router may be required to replicate a packet, multiple times, out the same interface. This, besides limiting the router's interface throughput and incurring larger processing times, as packets get queued, it also increases the latency for part of the multicast packets.

The act of combining multiple constraints, dealing with compound metrics and degree constraints optimizations, results in complex, NP-complete, optimization problems. Accordingly, the time needed to provide an optimal solution grows very fast with the size of the problem as there are no polynomial time algorithms for solving them. Therefore, we limit our search to approximate solutions that can be found in polynomial time. Nevertheless, even obtaining such a solution may still require relatively *large* amounts of time if the problem is large/complex. Ideally, the computation should take less than the average inter-member joining time. Yet this depends on the computational power of the machine running the algorithm or the algorithm's implementation efficiency. As both parameters are out of our control, such concerns are out of the scope of this document. Nevertheless, this should be a subject of future research. We detail aspects regarding the algorithm's implementation in the following subsection.

3.2.1 Optimization Algorithm

As explained above, the functions we minimize in our experiments are the maximum latency or number of AS hops from a host to the stream's source. To be noticed that the reference is the host and not the domain border router. Thus, what matters in deciding a member's position in the overlay tree is not solely its distance to the source but also the number of clients it serves. Then, a router close to the source but serving few clients might find itself lower in the hierarchy than another with a slightly higher metric to the source but with a larger client set. We optimize the quality of experience for

host clients not for overlay members. In what follows, we will be using the term *distance* when referring to a relative length or amplitude of a metric, observed on a path connecting two points, but when the exact nature of the metric is of no interest. To be noted that generally, the metrics encountered in the current document are lantecy and AS hops.

The problem described above, henceforth named *minimum average distance, degree bounded spanning tree (MADDBST)* may be formally stated the following way:

Definition 1. *Given an undirected complete graph G=(V,E), a designated vertex* $r \in V$*, a degree bound* $d(v) \leq d_{max}$*,* $\forall$ *vertex* $v \in V$*,* $d_{max} \in \mathbb{N}$*, a vertex weight function* $c(v) \in \mathbb{N}$*,* $\forall v \in V$ *and an edge weight function* $w(e) \in \mathbb{R}^+$*,* $\forall$ *edge* $e \in E$*. Let* $P_{r,v}^T$ *be the set of edges e on the path from vertex r to v in the graph's spanning tree T. Also, let* $W_{r,v}^T = \sum_{e \in P_{r,v}} w(e)$ *represent the cost of the path linking r and v in the spanning tree T. Find the spanning tree T of G, routed at r, satisfying* $d_T(v) \leq d_{max}$*,* $\forall v \in V$*, such that* $\sum_{v \in V, v \neq r} c(v) W_{r,v}^T$ *is minimized.*

We note that [57, 9] have previously defined and solved similar optimization problems. Shi et al. [57] also prove that a particular instance of the problem, where all vertices have weight 1, is NP-complete for degree constraints $2 \leq d_{max} \leq |V| - 1$. Similarly to our approach, they were interested in a centralized solution whereas Banerjee et al. [9] have successfully manged to distribute the algorithm.

The heuristic we use to solve the MADDBST problem is similar to the one used by Banerjee and it is a variant of the one proposed by Shi. The algorithm's pseudocode can be seen in 1. It works by incrementally growing a tree started at the root node r until it becomes a spanning tree. For each node v, not yet a tree member, we select a potential parent node u in the tree T, such that the metric $\delta(v) = (W_{r,u}^T + w(u,v))/c(v)$, or the distance to the source per client, is minimized. At each step, the node with the smallest metric value is added to the tree and the parent selection is redone.

In what follows we detail the topology discovery algorithms we have implemented and tested and their influence on how tree optimization are performed. Figures regarding their performances can be found in section 4.2.

3.2.2 Random Overlay

Actually, as the name indicates, this algorithm does not measure the topology, in fact it ignores it. We use it as a reference to gauge the performance of the other algorithms and also to evaluate the downsides of running an overlay with no underlying topological information. It has a very simple MS

Algorithm 1 Heuristic used to solve the MADDBST problem

Input: $G = (V, E)$; r; $w(u, v)$, $\forall u, v \in V$; c(v), $\forall v \in V$; d_{max}
Output: T

```
foreach v ∈ V do
  δ(v) = w(r, v)/c(v);
  p(v) = r;
end for
T ← (U = {r}, D = {});
while U ≠ V do
  let u ∈ V − U be the vertex with the smalles δ(u);
  U = U ∪ {u}; L = L ∪ {(p(u), u)};
  foreach v ∈ V − U do
    δ(v) = ∞;
    foreach u ∈ U do
      if d_T(u) < d_max and (W^T_{r,u} + w(u, v))/c(v) < δ(v) then
        δ(v) = (W^T_{r,u} + w(u, v))/c(v);
        p(v) = u;
      end if
    end for
  end for
end while
```

implementation, due to not requiring the added complexity of obtaining and constantly improving a topological database. Further, no tree optimizations are performed when this algorithm is ran. Not even for member departures which require the affected children to reperform the join procedure.

3.2.3 BGP Based Overlay

One of the best sources of topological information that is not or can not be usually used by application layer overlays is the BGP routing table. It does not provide a highly detailed map of the router infrastructure but it does provide a coarse, high level look at the intricate connection patterns between autonomous systems. The BGP information an AS router holds attempts to present an Internet wide interconnection map but due to the algorithm's distributed nature and its use of policy, both inaccuracies and incomplete data may exist. As a consequence, two autonomous systems, besides having two egocentric, non-overlapping or complementary views of the global infrastructure, they may also contain contradictory information. Still, such situations should be viewed as exceptions and not norm.

The Map-Server has two options for obtaining BGP topological information. On one hand, it may aggregate partial BGP feeds from multiple overlay members or, on the other hand, it may itself connect to BGP. The former could ensure a more detailed view of the topology, and thus grounds for better decisions, while the latter a more restricted, partial view of the interconnection map and seemingly worse performance. However, we carried several experiments to assess the benefits of a *global* vs a *local* BGP view of the topology for the overlay management. The results, though always indicating the first option as a better peformer, showed small relative differences in performance. If obtaining a *global* BGP view by aggregating BGP feeds and Internet wide measurements results is deemed accurate enough the architecture can be implemented relatively easily. However, aggregating multiple BGP tables, to obtain the *global* BGP topology view, may be a technically challenging task as it requires the transfer of potentially thousands of partial BGP tables pertaining to the overlay members. Not to mention that some domains may be reluctant to provide such information which they deem as sensitive. By contrast, the *local* topology gathering mechanism requires nothing more than the BGP feeds from the routers upstream of the source domain's overlay participating border router. Furthermore, there is no need for a communication protocol between the MS and the overlay members for the conveying of BGP reachability information.

We have opted in our experiments, due to its simplicity and relative good performance, for the BGP topology discovery mechanism based on *local* information. The metric it provides, inter AS hops, in conjunction with our tree optimization algorithm should result in an overlay ogranization of nodes that approximates that of a degree constrained shortest path tree. Or otherwise, a tree where the number of hops between the source and the clients, the overlay AS path length, is minimized. A byproduct of AS path length minimization is the decrease of the tree's AS hop cost and thus a more efficient use of the network infrastructure. Notably, the performance of this design relative to that of the latency based overlay should also give some insigths about how good AS hops are at estimating latency. Should there be interest in obtaining a minimum AS hop cost tree, at the expense of larger member latencies to the source, a degree constrained minimum spanning tree heuristic should be employed.

3.2.4 Latency Based Overlay

Inter-member latency is a metric commonly employed by application layer overlays in topology optimizations. Yet, the obtaining of a full inter-member latency map scales poorly with the population size and may require unac-

ceptably high number of measurements. For instance, N members would require the latency estimation of $N(N-1)/2$ links, which for an average value of N=1000 would amount to almost 500k measurements, or 500 measurements per member. Hence, a more intelligent approach for the selection of link latencies worth estimating is needed.

We obviate such large number of measurement by exploiting a mechanism similar to the one used by Banerjee et al. in [7] for group management. The architecture was previously described in section 2.4.1 but we will briefly review it in what follows. The NICE control plane protocol works by clustering nodes that are close to one another in terms of latency and thus by limiting the majority of inter-member measurements to just the pairs of nodes finding themselves in close proximity. It achieves and maintains the clustering by creating a hierarchy of topological representants that is headed by a set of overlay members and finally by the stream's source. The elected cluster representant or *head* is the node with the lowest aggregate latency to all other cluster members. Therefore, the highest layer cluster members should be the *centers* of the set of nodes they represent. Any joining node, starting at the highest layer, is required to iteratively measure its latency to the cluster's members, select the one it is closer to and repeat the process with the cluster headed by the just selected node until it finally reaches a lowest layer cluster which it finally joins as a member. The process ensures a low number of latency measurements and, considering how the hierarchy is built, that the joining node reaches a cluster with members to which it is topologically close to. Refinements require local measurements and ensure the correct cluster memberships and position in the overlay. All nodes must be members of cluster in the lowest layer and through election can incrementally join clusters in higher layers. The amortized cost analysis in the paper shows that for NICE, the control overhead at an average member is constant $O(k)$ whereas in the worst case it can reach $O(k \log(N))$. Where k is a constant limiting the size of the cluster and N the number of overlay members. These are relatively low values if we compare them to the naive approch we first discussed where the measurements overhead per node was $O(N)$.

Our implementation reduces the control overhead further as all the group management decisions are centralized and taken by the MS. There is no communication between overlay members. Consequently, the creation of a new communication protocol between members and MS is required. More details can be found in section 3.3. We have the MS run the control plane protocol of NICE just for the gradual discovery of latencies between topologically close members. Or in other words, we use it just as a topology discovery protocol. Otherwise, the overlay management is done as previously described in this section. We are interested just in the measurement of latency between nodes

in relative proximity because we generally expect, and encourage through design, that data paths follow such links for a minimal tree cost.

The combination of the latency discovery protocol and the optimization algorithm previously presented have resulted in an overlay architecture that approximates a latency shortest path tree. As a consequence, the paths connecting the source with the member nodes and their clients will tend to have very low latency but may follow longer AS paths. Such design would be suitable for multicast streaming of content with very stringent latency constraints.

A *peculiarity* of BGP routing is that it gives rise to latency triangle inequalities violations [58, 59] that some speculate are due to routing policies [59]. Their effect is that the BGP selected paths possess higher latency than others that, despite looking like detours to BGP's decision process, due to increased number of hops, have low aggregated latency. In other words, it may happen that a one hop AS path has higher latency than a two hop one, or that a triangle edge is larger than the sum of the remaining two. As a result, it may happen that overlays might offer lower inter-member latencies when compared to their underlying, unicast, ones. Both [58, 59] have identified lower latency paths than the BGP selected ones for more than 20% of the pairs in their datasets. Our results 4.2 exhibit similar traits.

3.3 Protocol Specification

After presenting the details of our proposed streaming architecture and multiple solution for how ovelay management could be done we now proceed to presenting the low level, LISP packet, changes that are required for the actual protocol implementation. As is the case for LISP, we are not proposing any host stack updates but we do require an interface between the domain local and inter-domain multicast which most probably will be implemented in the domain tunnel routers. And thus we require intra-domain multicast support. Furthermore, in what follows we assume that PIM-SM [46] is the deployed intra-domain multicast protocol. In this light, the *source specific multicast packet delivery to a host* problem can be seen as compososed of three sub-problems:

- intra domain multicast packet delivery
- domain local to inter-domain multicast conversion
- inter-domain multicast packet delivery

The first item requires no protocol changes in our architecture. A host reports its multicast group membership, or interest for a multicast channel (S, G), by means of IGMP. Due to our support just for SSM, G must be an IPv4 or IPv6 address from the IANA reserved SSM ranges [54, 55]. A local area network designated router should receive the client's report and proceed to sending a PIM Join, for channel (S, G), to one of the domain's border routers. The join is sent towards the IP address of the router that annouces reachability of a prefix ecompassing G through a domain specific IGP. Consequently, multiple joins for the same host group address should end up at the same border router if only one announces such reachability. The join creates the necessary intra domain multicast delivery state for the MCID (S, G). Finally, all intra-domain packets having as destination address G and source S will be received by the requesting host. A host unsubscribes from a multicast channel in a similar fashion.

The second item is an interface in the tunnel border router which needs to *translate* domain-local PIM messages to LISP multicast ones. How such an interface is to be implemented is out of the scope of the current document.

The third item is the sub-problem where our solution fits. It does require changes but only to the LISP protocol. By means of a LISP-cast (Lcast) subscribe message, sent to the source's Map Server, the router is added to the overlay and served the streamed content. If it is to replicate traffic to other overlay members a Lcast Map-Reply message will provide their locator sets. Also, depending on the employed topology discovery protocol, if any, it may be required that peers perform inter-member measurements which they need to convey back to the MS through another message exchange mechanism. Details about what they are and how such messages may be implemented are presented in the next subsections. They may be split in two categories, depending on their intended goal:

- overlay management
- topology discovery

Their implementation may be done by either allocating new LISP message types for the Lcast protocol or by reusing current LISP messages but in association with Lcast flags that change the scope of the carried information. We think the latter option is the most appropiate as **Map-Request** and **Map-Reply** messages are already designed to carry information of similar scope as the one we wish to convey. It also has the advantage of not requiring the consumption of additional LISP control message types. We explore this idea in what follows.

The first required change would be the allocation of a Lcast flag in the Map-Request and Map-Reply [23] messages headers for the identification of Lcast messages. Further, in order to carry the host group identifier, G, which does not have a direct LISP equivalent, we require that the Map-Request and Map-Reply records should stop transporting an EID-prefix, which in Lcast context is not defined, and convey G instead. Consequently, Lcast control packets based on LISP Map-Request messages will traverse the mapping system having S, an EID, as destination and carry G as a record. Similarly, Lcast control packets based on LISP Map-Reply messages will traverse the Internet having one of the requesters RLOCs as destination address and G in at least one of the EID-prefix fields. However, this information is not enough to distinguish between multiple Lcast overlays coordinated by a single MS for multiple sources that may be using the same host group identifier G. In other words, the MS may serve as overlay coordinator for both $(S1, G)$ and $(S2, G)$, with $S1$ and $S2$ part of the same EID prefix. Because the Map-Reply has no field for transporting information about the source EID we propose the use of a locator record, part of G's record, and the allocation of a flag/field that indicates the presence of S. As a result, together with the locator records that identify overlay members to which the receiver of the message is to replicate traffic to, another locator record will carry the source's EID. Thus, ideally, the locator header, besides needing a *Type* field that distinguishes between locator and S-EID records and an *AFI* field, that encodes the address family of the object carried by the record, would require two more fields, *Peer* and *State*, used for grouping of locators. Specifically, *Peer* would indicate the index of the overlay peer this locator belongs to, in no specific order, and *State* would indicate if the locator is the one to send traffic to or just a backup one. For a *Peer* just one of the locators can be active at a given time, the rest are sent just for resilience purposes. The resulting header could be similar to the one in figure 3.1.

All the required fields can be mapped to the current locator header specification because part of the information it carries bears no or slightly changed significance in Lcast context. Hence, one of the unicast *Priority* and *Weight* fields, could become type *Type* indicator, while the multicast *Priority* and *Weight* could be used as *Peer* and *State* fields respectively. The latter two are only slightly, semantically, different from the ones used in LISP-Multicast owning to the lack of priority between locators in Lcast.

Alternatively, not to overload the semantics of the Map-Reply record used in unicast, a new one could be defined for Lcast purposes only. We do not explore this possibility in the current document.

Figure 3.1: Redesigned locator record header

3.3.1 Lcast Overlay Management Messages

These are the messages used between the MS and the member nodes for joining or leaving the overlay and for more complex tasks like overlay reshaping. In the first category we have:

- **MSubscribe**: the multicast subscribe message is sent by an xTR to a MS when requesting to join an Lcast overlay serving content for a channel (S, G).
- **MUnsubscribe**: the multicast unsubscribe message is sent by an xTR to a MS to announce its intention of leaving an Lcast overlay that serves content for channel (S, G).
- **NoAction**: it may be used as an acknowledgement of reception for all the other messagages.
- **ReportClusterSize**: it is used by an xTR to report to an MS the intra-domain number of clients it serves for MCID (S, G).

The second category contains the messages that may be used to perform an overlay reshape with no packet loss. The messages are used to first create a virtual topology that does not forward packets, but once finished, it is initialized by the parent node in an organized way such that no data packets are dropped. The messages needed are:

- **StoreVirtualChildren**: this is a message sent by the MS to an xTR requesting it to store the nodes conveyed with the message as virtual, future, children.
- **SwitchTree**: the message is sent by the MS to the head of a virtual tree, once the tree is completed, to require that all the virtual tree members switch to the alternative, virtual topology.

- **SwitchTree data path bit**: this is not a control plane message but a bit in the data path header. It is used by the head of the virtual tree to indicate to all members that they may start using the virtual tree as the data path tree.

All messages must carry the multicast channel identifier, (S, G) in order to help the MS distinguish between the multiple overlays it controls. Considering their syntax, *MSubscribe*, *MUnsubscribe*, *StoreVirtualChildren* and *SwitchTree* are Map-Request type of messages whereas *NoAction* and *ReportClusterSize* are Map-Reply type of messages.

3.3.2 Lcast Topology Discovery Messages

The communication mechanism is to be used in situations where the MS requires member dependent topology discovery. Out of the tree solutions proposed in section 3.2 only the one using latency as a metric requires the use of these messages.

- **MeasurePeersRequest**: sent by the MS to an xTR to request that it measures its distance (the metric is also specified) to the set of nodes conveyed in the message.
- **MeasurePeersReply**: sent by the xTR as reply to the MSs request for measurements. It carries the results of the measurements.

The first message is of Map-Request type and the second is of Map-Reply type. If the metric to be estimated is latency, a simple measurement algorithm can be implemented with the help of LISP's protocol mechanics. To expand, the peers in the measurements request can be stored as locators of the host group identifier G with priorities of 255 which indicate that they are not to be used for forwarding. However, they will be RLOC probed and as a result a round trip time estimate will be found.

Chapter 4

Evaluation

4.1 Evaluation Methodology

In order to compare the performance of the overlay management algorithms proposed in section 3.3 we have created an event-based simulator. The datasets used in building a network topology were obtained from Internet wide measurements carried out by iPlane [60], RouteViews [61], CAIDA [62] and RIPE [63]. The participating domains, and their respective number of clients were obtained from a worldwide distributed capture of sopcast [12] p2p Internet TV channels streaming an UEFA Champions League semifinal football match.

In what follows we start by describing the datasets and methodology used to build a realistic Internet inter-domain topology. Then, we present the traces we generated by using the captured p2p traffic. We continue by introducing the metrics we considered for the evaluation of the simulated topologies and we conclude the section with a brief presentation of our simulator.

4.1.1 Internet Inter-Domain Topology

For the purpose of obtaining a realistic global inter-domain topology we have aggregated datasets that estimate how autonomous systems interconnect from multiple sources: iPlane, RouteViews, CAIDA and RIPE. All the used data is from April 2011. Figure 4.1 shows, for the resulting topology, the log-log plot for complementary cumulative distribution function (CCDF) of the AS-node degree. It can be observed that it closely follows a straight line, property found in power law distributions. As previously shown in [64] and [65] the Internet is a scale-free network with power law AS-node degree distribution. Further, the average path length in the obtained topology

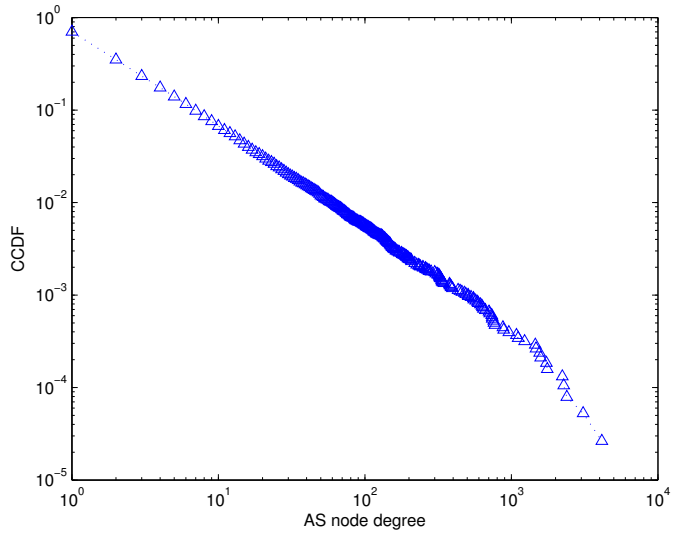

Figure 4.1: CCDF of the AS-node degree in the aggregate topology

is 3.5, just 5.4% lower than the one currently observed [66] in the Internet. These two results corroborate our claim that the aggregate topology has properties similar to those of Interet's AS graph. Nevertheless, it is worth noting that we have knowingly disconsidered in our design link specific BGP policy information that could transform part of the AS graph's edges in arcs (directed links). However, most affected by this assumption are the links between customers and their upstream providers. Such links can not be used by the upstream providers for transiting traffic to destinations other than those found in the client's network. Yet, the use of client routers for transiting/replicating traffic is one of the main features of our proposed inter-domain multicast solution. We therefore think that the discounting of BGP policy information should be of limited influence to our results.

Distance, in terms of AS hops, between any two ASes, was computed by applying Johnson's all pair shortest path algorithm to the obtained connected graph. For latency we made use of iPlane's [60] proven [67] latency prediction abilities for IP pairs. Because we are interested in estimating the latency between domain border routers, that in our design represent the clients, we had to elect for all participating ASes a representant. We did so by using iPlane's estimations for points of presence (PoP) to AS mapping and their inter-connection map. For any domain, the PoP with the largest degree was elected as the representant. In about 30% of the cases, when iPlane failed to provide an answer, we used an estimator based on geographical distance

Figure 4.2: P2P TV Traces Capture Points

described in [37].

Of course, all these efforts to approximate the Internet infrastructure, in spite of providing some of the most accurate datasets, are bound to have errors or to be incomplete due to practical limitations of the tools used in the measurements. Also, our assumptions are another source of innacuracies. However, we do not aim to have a *perfect* network topology, but one of similar properties to that of the current Internet with good approximations for topological and temporal distances between nodes.

4.1.2 The Generated Traces

We generated several meta-random traces that we used to evaluate the ability of the proposed solutions to deal with distinct types of client behavior. The participating domains and their respective number of clients were obtained from a passive distributed capture of several p2p TV channels whereas the client churn was modelled in accordance to recent results in the field. We detail both efforts in what follows.

The fast paced increase of Internet access speeds have popularized Internet TV services and among them sopcast [12] is frequently used for the streaming of live sports events. Wanting to model client distribution for large events of global, or at least spread interest, we decided to capture p2p streaming overlay characteristics for an UEFA Champions League semifinal. Though interest for such football matches is highest in Europe, the teams in-

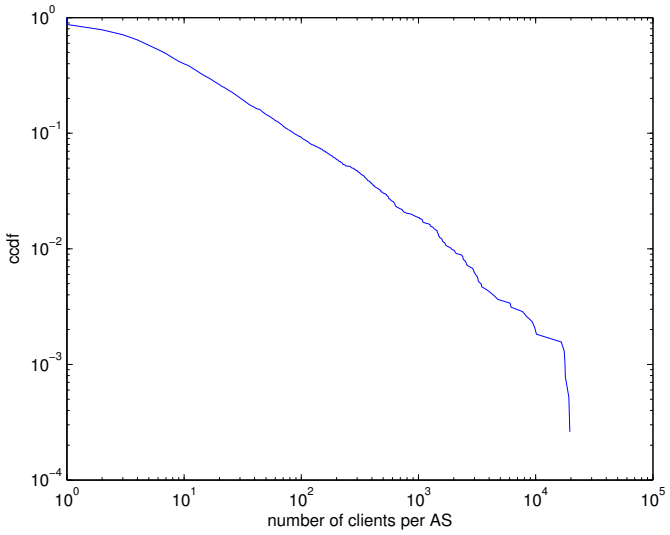

Figure 4.3: CCDF of the clients per AS distribution

volved are both highly appreciated worldwide and amount players spanning many nationalities. Also, interest was increased as this is the penultimate phase of the prestigious competition.

For the capturing process we used 2 vantage in USA, 5 in Europe and 2 in Asia. They span a total of 6 countries (see fig 4.2). We were interested in understanding how clients cluster in autonomous systems, not in the specific performance of a channel's overlay. Thus, depending on the upload capabilities of each vantage point, we joined a number of p2p channels, streaming the same event, at each node. The CCDF for the aggregate distribution of clients in ASes is presented in figure 4.3. Though common for such data to be accurately described by a power law, we couldn't find an appropriate fit. We thus concluded that it obeys a more complex law, despite it being similar to a Pareto distribution. The traces contain more than 146k distinct IPs spread in over 3.8k ASes. We had no means of detecting multiple clients behind NAT performing devices so the above value is a lower bound for the number of participating clients.

Despite the size of our captured dataset, lack of logs from the overlay's bootstrapping server made it impossible to approximate client lifetime in the overlay. We thus resorted to synthetic modelling of client churn. It is generally accepted [68, 18, 17, 69, 19] that client arrival process, at least for periods spanning dozens of minutes, can be modelled by a Poisson process. Furthermore, Sripanidkulchai et al. in [68], after analyzing 3 months worth

of Akamai logs, observe that *short duration* events, which last a couple of hours, present flash crowds whereas non-stop streams have a time of day behavior. These findings were confirmed by Veloso et al. in [69] who noticed that for *long* streams client inter-arrivals cand be modelled through a Pareto or a piecewise stationary Poisson.

The above consensus does not hold anymore for client session lengths which, depending on stream length or the type of system being analysed by either paper, follow different distributions. Still, with the exeption of [18], there seems to be an agreement that sessions should have lengths distributed according to a power law. Nevertheless, opinions diverge when assessing the *weight* of the tail.

Considering the works discussed above, in order to perform an evaluation of our proposed architecture that acknowledges the wide range of client behaviors, we have generated 3 traces with distinct properties. The goal was to model a short event, of 9000 seconds, with a piece-wise Poisson arrival process but with different shapes for the distribution mirroring session lengths. In order to capture the flash crowd effect we required that 80% of the clients join during the first 1800 seconds, and the rest spread over the time left. We decided to use a Pareto distribution for the session lengths and we varied its shape and scale parameters the following way: $(\alpha = 1.9; x_m = 30), (\alpha = 5; x_m = 2400), (\alpha = 10; x_m = 6000)$ in order to emulate low, medium and respectively high client interest in the streamed content.

4.1.3 Metrics

We evaluate the performance of the propsed schemes along the following dimensions:

- **latency stretch**: This metric measures a node's relative gain in latency to the stream's source when compared to the unicast one way delay between the two. It is defined as $lat_{overlay}/lat_{unicast}$ and it may be used for overlay members or application clients.

- **hop stretch**: It measures a node's relative gain in number of AS hops to the stream's source when compared to the number of hops crossed by the unicast path linking the two. It is defined as $hops_{overlay}/hops_{unicast}$ and it may also be used for overlay members or application clients.

- **tree cost**: Wanting to quantify the property of efficient network usage, we defined a cost that measures the number of AS hops crossed in order to have one piece of information reach all nodes. It is computed

as $\sum_{v \in V} hops(v; parent_v)$. Where V is the set of all members, *hops* is a function that returns the number of hops between two nodes and $parent_v$ is the overlay parent for v.

- **control traffic overhead**:It measures the number of messages exchanged by the source with the tree members for the purpose of creating a tree, maintaining tree integrity and optimizations.

These metrics are a subset of those usually used in the literature[7, 8]. The first provides performance figures of interest for applications, the next two evaluate the efficiency in using the network and the last quantifies the complexity due to a centralized architecture.

4.1.4 Simulator

With the aim of evaluating the overlay management algorithms presented in section 3.2 we have implemented an event-based simulator that builds on the the above described AS topology and client traces. The result was the partial implementation of an Lcast compliant Map-Server. We were just interested in testing the feasibility of our idea and in quantifying the relative performances of the proposed overlay management algorithms. Consequently, there was no need to implement protocols like UDP, LISP or our Lcast extensions whose communication functionality was implemented by means of simplified software interfaces.

Operationally, members are joined or leave the overlay in accordance with the client traces. Typically, a join will be done at the first randomly found position, however, periodically and depending on the considered algorithm, the MS proceeds to optimizing the data path for the minimization of the selected metric. Once a tree is computed, connectivity information is conveyed to the nodes that rearrange accordingly. Similarly, an announced departure requires the MS to reshape the subtree served by the leaving node. As for global optimizations, once the best tree is computed, nodes are signalled to reorganize. This continously changing overlay state is sampled periodically, $100s$ in our simulations, and saved for the performance evaluations that we perform offline.

The obtained results can be checked in the following section.

4.2 Results

In this section we will be presenting the results obtained after simulating the overlay architectures described in 3.2 with the help of the traces generated in 4.1.2. For ease of reference, we will refer to the three generated client traces, with parameters $(\alpha = 1.9; x_m = 30), (\alpha = 5; x_m = 2400), (\alpha = 10; x_m = 6000)$ as *trace 1*, *trace 2* and *trace 3* respectively. The performance evaluation is done considering the metrics defined in the previous section. Generally we present a comparison of the average performance but for completeness and, in some situations, a better understanding we also provide the 95% confidence interval. We expect that routers will be using very low replication factors 3, maybe 4, but for an improved understanding of the algorithms and the overall architecture performance we present results for fanout values up to 10.

4.2.1 Latency Stretch

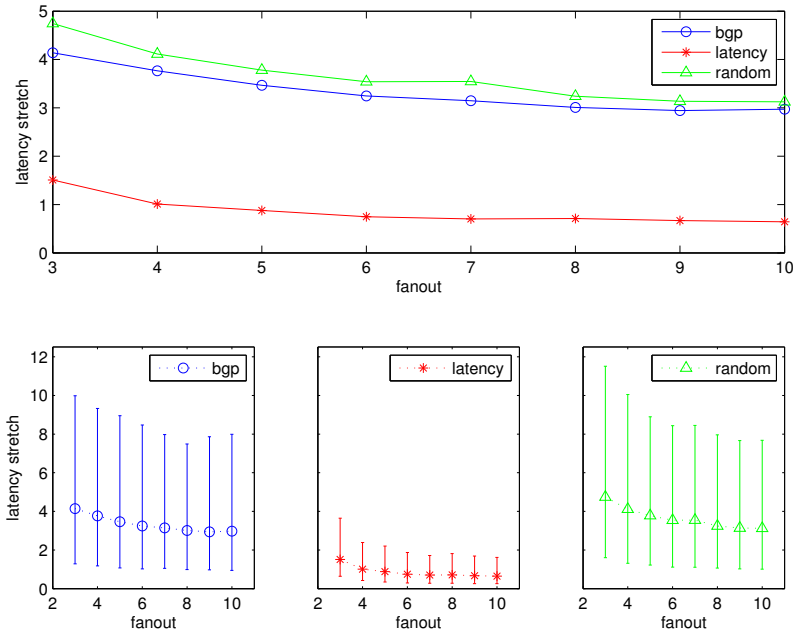

Figure 4.4: Latency stretch for trace 1

The latency stretch, as previously explained, measures the performance of the overlay path relative to that of the underlying, unicast one. It is computed as a fraction and thus a value higher than 1 would mean that, on

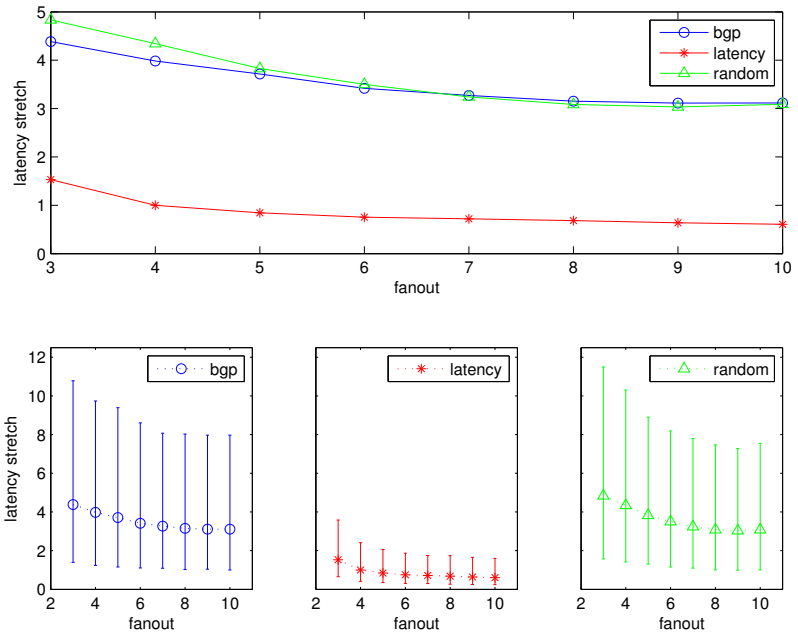

Figure 4.5: Latency stretch for trace 2

average, unicast delivery has lower latency than the overlay routing whereas a value lower than 1 means that the paths in the overlay outperform (have lower latency) than unicast ones.

Figures 4.4, 4.5 and 4.6 present the results per client for the three traces. As expected, the latency based overlay outperforms the other two architectures but what surprises is the very good performance of the random overlay relative to that of the BGP based one. We see that for very dynamic client behavior (figure 4.4) the BGP information is relevant enough to give its overlay a slight edge over random but fails to do so for traces with more stable clients. Nevertheless, it is worth observing that low fanout values favor BGP independent of the client churn and for fanout 3 BGP has a significant advantage.

The *NICE* control protocol, that we have used for latency discovery, seems to be providing adequate topological information as the results show a very good performance of the overlay for this specific metric. Actually, for larger fanout values it continuously outperforms unicast delivery. This can be explained by what are known as *triangle inequality violations*. They are caused by the existance of BGP discarded long AS paths with very low latency. For a more detailed discussion see section 3.2.4.

As a general trend, it worth observing that for all overlays peformance

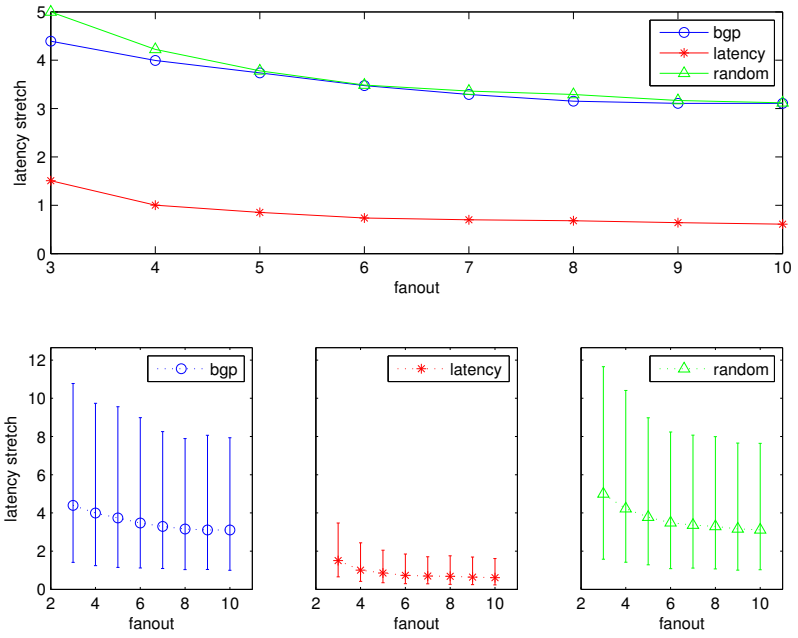

Figure 4.6: Latency stretch for trace 3

improves very little for fanout values higher than 6.

4.2.2 AS Hop Stretch

The hop stretch measures the increase of the number of hops to the stream's source relative to the BGP AS path length. Similarly to latency stretch, it is computed as a fraction and thus values higher than 1 indicate an elongated overlay path whereas values lower than 1 would be indicators of improper BGP behavior. Because we compute both the unicast and multicast paths we do not encounter such situations however this does not preclude such situation from happening in Internet due to BGP instability.

Figures 4.7, 4.8 and 4.9 present the AS hop stretch per client results. This situation is somewhat reversed when comparing the results with those of the previous metric. The worst performer is the overlay based on latency, and this is to be expected, as its goal is to minimize latency stretch, but again, random showed an unexpectedly good result. Yet, in contrast to the latency stretch results, the BGP based overlay manages to maintain the advantage for the whole range of fanout values. Despite apparently small, a a 23% decrease in number of hops to the root on average for a client can't be ignored. We speculate that the random overlay's good performance can

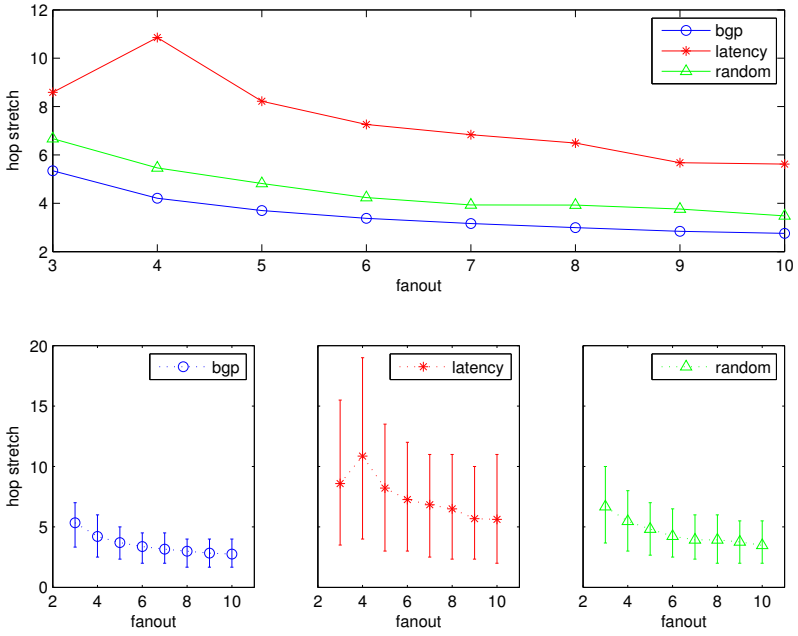

Figure 4.7: Hop stretch for trace 1

be explained through the very low average AS path length in our topology, 3.5 hops. However, this also holds true for current day Internet average AS path length which is just 5.4% higher, at about 3.7 hops.

As with the previous metric, improvement increments are minimal after fanout 6.

4.2.3 Tree Cost

We use tree cost to assess the advantage of using a source specific multicast delivery system instead of unicast. Besides the obvious benefit of avoiding the very large unicast fanout, which for our experiment would be of around 140k, the overlay should make better use of the network's resources. Namely, due to on-path replication, the overlay should use much fewer links for the delivery of a packet to all clients. Figures 4.10, 4.11 and 4.12 provide some unicast normalized performance figures. Expectedly, the BGP based overlay requires the smallest number of AS hops to deliver a packet to all clients. However, as results show, all architectures outperform unicast by almost two orders of magnitude.

The metric also reveals that the random overlays have a greater tree cost than the latency based overlays, despite outperforming them in terms

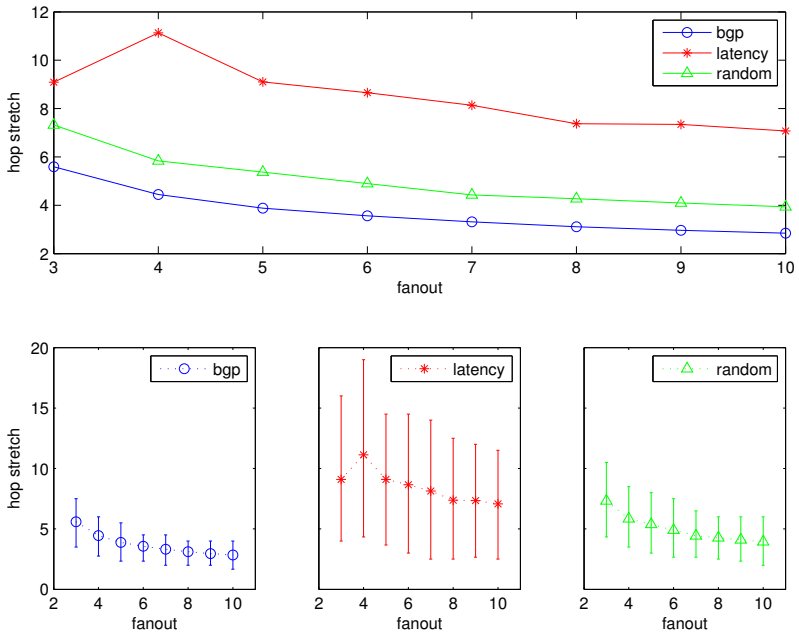

Figure 4.8: Hop stretch for trace 2

of hop stretch. This shows that although many of the nodes in the latency based overlay have long in-tree AS paths they also seem to efficiently choose overlay parents as the total tree cost is diminished. Indeed, the optimization algorithm increases tree diameter with the aim of minimizing overall latency to the root. However, the random topology tends to build a complete tree by using all of a node's fanout, but due to the random nature of the in-tree links the average inter child-parent number of AS hops seems to be larger.

Notably, there appears to be almost no correlation between the fanout and the tree's cost.

4.2.4 Control Traffic Overhead

We use this metric to quantify the cost of maintaining an optimized overlay both for members and the coordinator. All messages have the MS as either source or destination and thus it is incurred a much higher load than any other overlay node. However, this design decision transforms into an advantage when one acknowledges that a distributed protocol would probably require, on behalf of all overlay members, more measurements (or control traffic overhead) and more computing power.

Figures 4.13, 4.14 and 4.15 present the results per overlay member. To

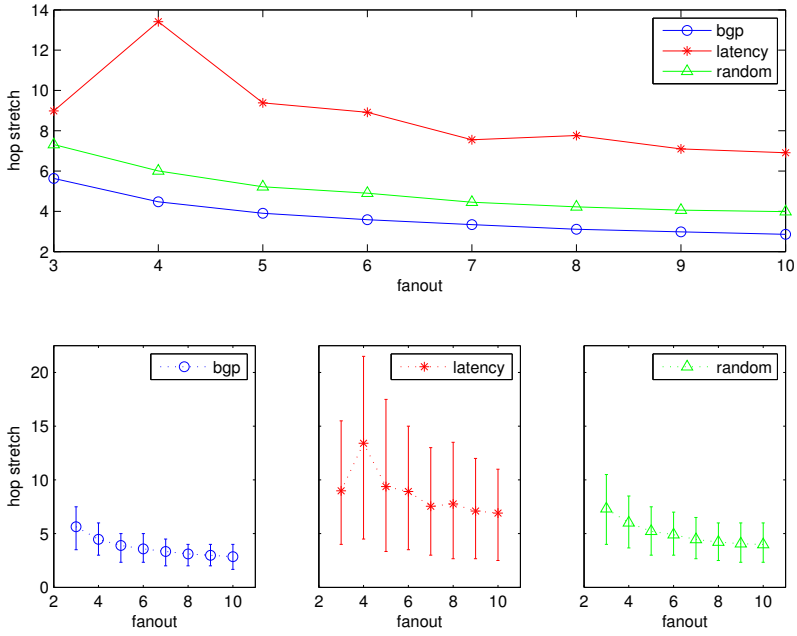

Figure 4.9: Hop stretch for trace 3

be noted that for the latency based overlay the control traffic overhead does not consider the messages exchanged for topology discovery purposes which are to be discussed separately. The plots indicate that both the BGP and latency based overlays require on average significantly more messages than the random overlay if we look at the values in relative terms. Nevertheless, an average of 10 to 15 message exchanges with the root, spanning a period of $9000s$, or a maximum of 1 message every 10 minutes, is, in our opinion, an acceptably low value. This been said, it can be observed that despite a very low average, there are nodes that exchange up to 40 messages with the MS. Still, albeit the increase, an overhead of one message at 3 minutes remains, in our opinion, acceptable.

Large fanouts do diminish member control traffic overhead. Especially for the BGP based overlay.

Figures 4.16 presents the total number of messages exchanged by the MS with the Lcast clients for the purpose of overlay management. Again, the results for the latency based overlay do not include the messages exchanged done for topology discovery purposes. In the worst case the MS is required to exchange short of $44k$ messages which, on average, equates to less than 5 messages per second. By contrast, the lowest possible overhead is that of random and it consists in 0.7 messages per second. The BGP based overlay

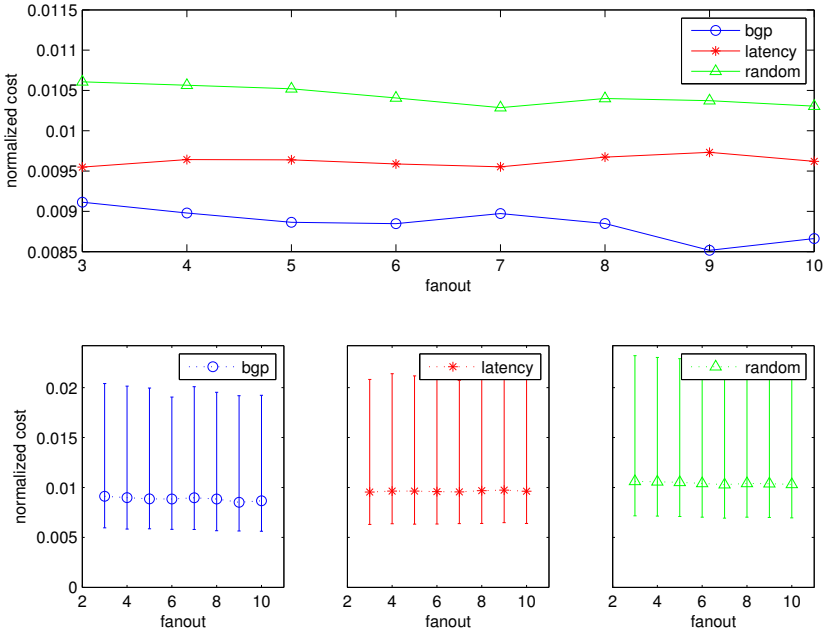

Figure 4.10: Normalized tree cost for trace 1

continously requires less control traffic overhead and it deals better with larger fanout values than the latency based one. Also worth noting that, with the exception of random, the MS is required to exchange more messages when the clients have lower churn, and thus are less susceptible to leaving the overlay.

Finally, figures 4.17 and 4.18 show the communication requirements of the topology discovery mechanism. The values should be taken as a higher bound due to our use of a maximum cluster size of 30 for the SALM overlay. Figure 4.17 shows that on average 500 to 700, or around 3.3 to 10.5 per minute, link latencies are discovered per overlay member. Note however that this is not the number of active measurements performed by the node, which is expected to be lower as other overlay peers may instead initiate the probing procedure. The exact number of links measured depends on the algorithm employed by the MS for measurements load spreading which, alas, was not considered in our simulations. The *busiest* of nodes, those in the higher layers of the SALM overlay, have their latency measured to almost 2000 overlay peers, or one active or passive measurement every 4.5 seconds. There appears to be no correlation between fanout or client dynamics and number of measurements. We would need more tests to confirm that the number of measurements depends just on the number of participants and

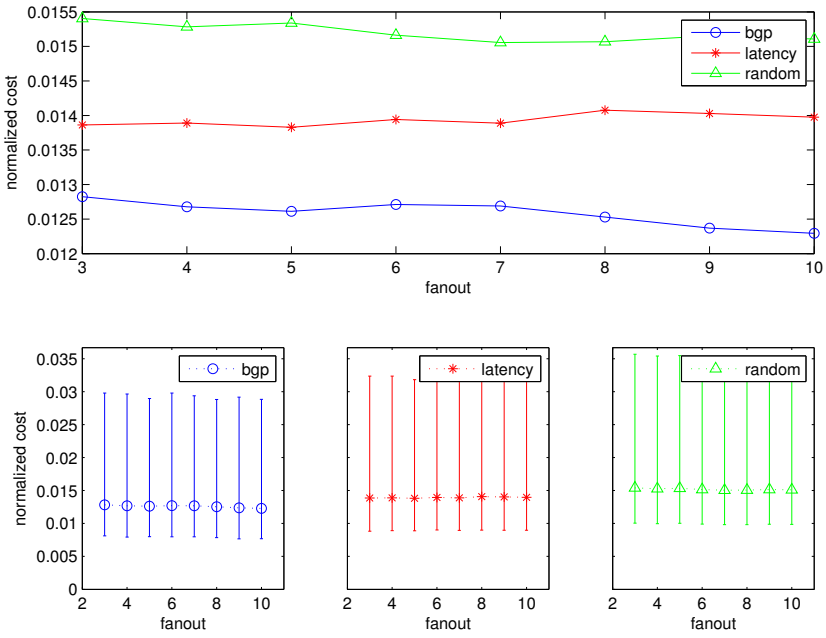

Figure 4.11: Normalized tree cost for trace 2

the cluster size.

Figure 4.18 shows the total number of pairs that were measured for the whole fanout range and all traces. Although the values appear to be very large, ranging from around $700k$ to $1.05M$ pairs, they represent about only one fifth of the total number of pairs that could be measured. Furthermore, the SALM architecture assures that these measurements are performed in an organized manner. Also, as previously mentioned, these values should be taken as upper bounds due to our choice of cluster size. Regarding the load they may incur on the MS, due to our implementation's lack of a measurements load balancing functions, we just note that the MS may, with a single message, require a node to measure a large set of peers and the same holds true for the reply that conveys the latency estimates.

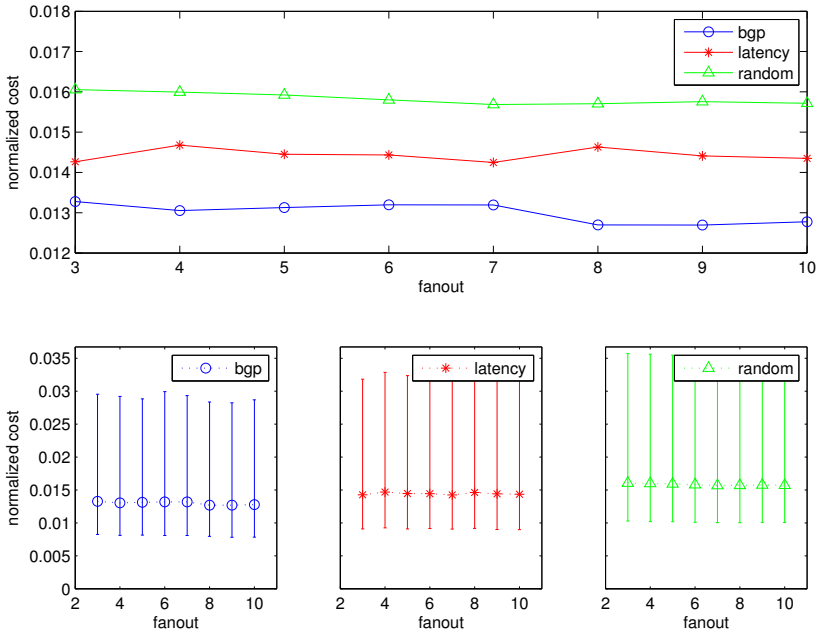

Figure 4.12: Normalized tree cost for trace 3

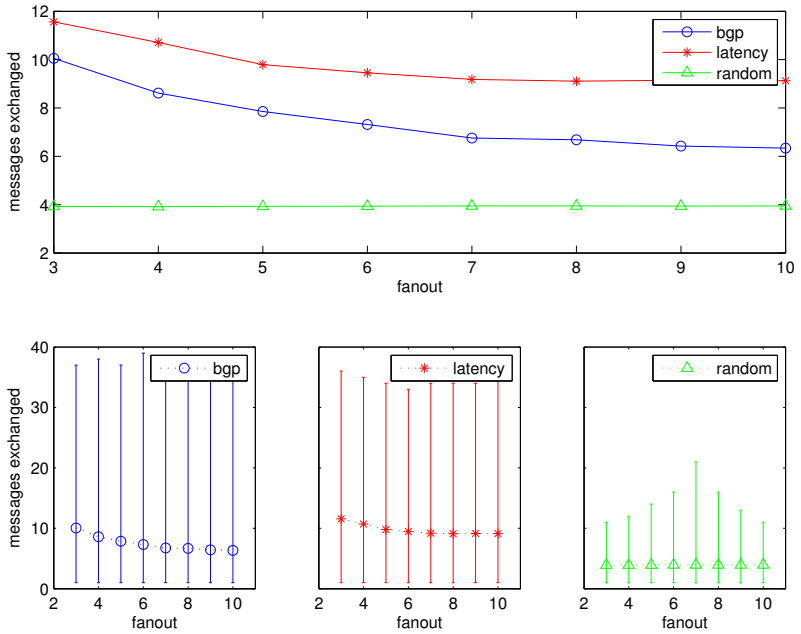

Figure 4.13: Control traffic overhead per member for trace 1

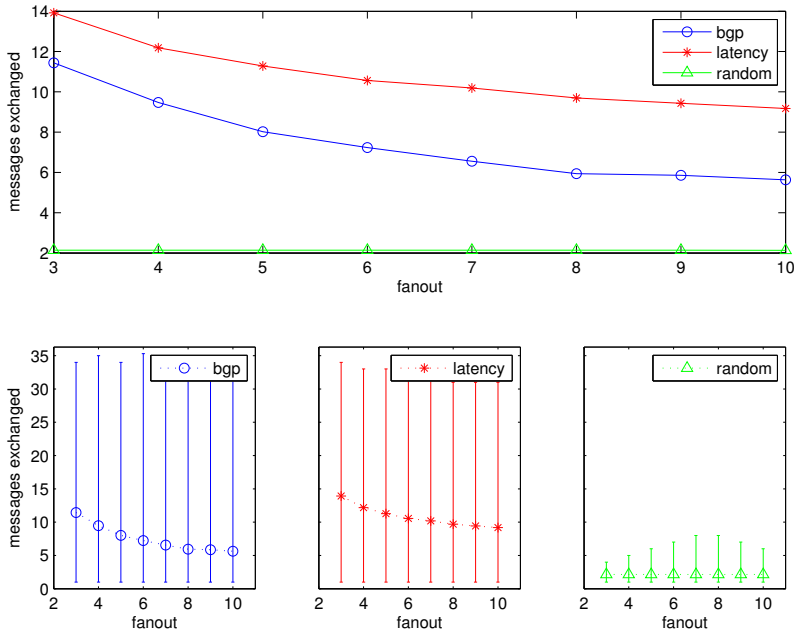

Figure 4.14: Control traffic overhead per member for trace 2

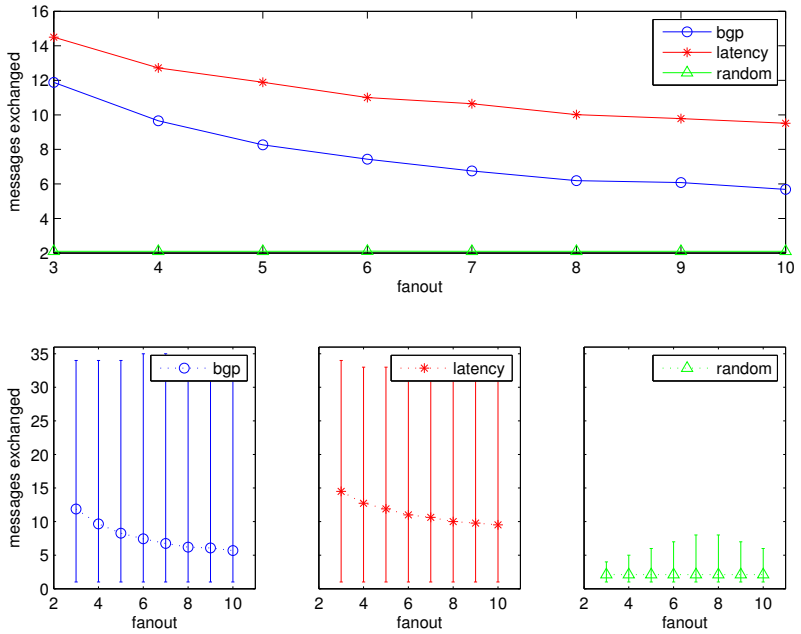

Figure 4.15: Control traffic overhead per member for trace 3

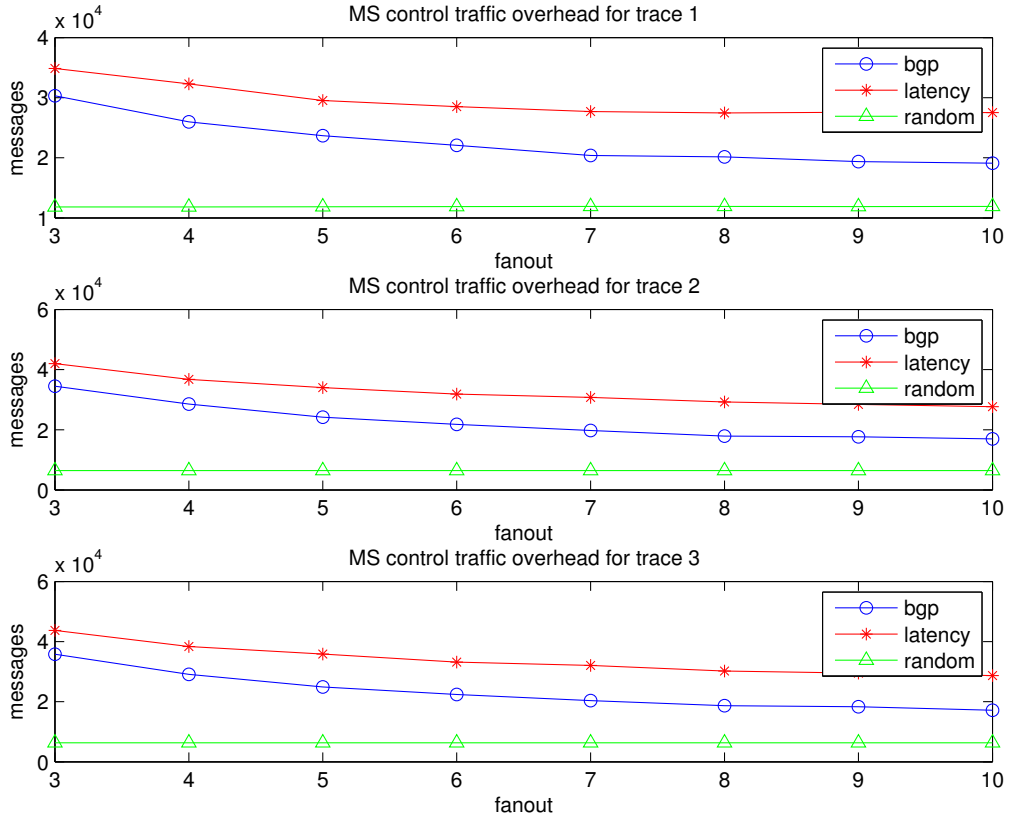

Figure 4.16: MS control traffic overhead

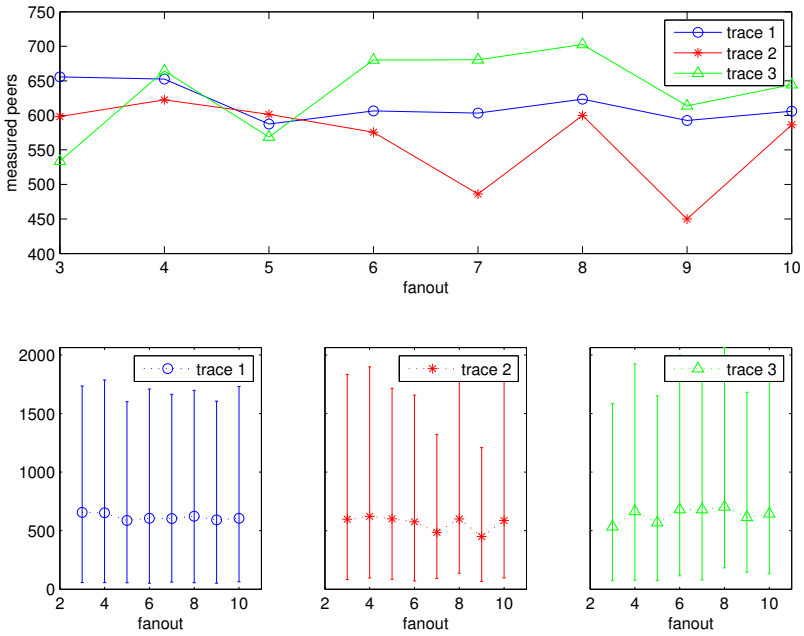

Figure 4.17: Number of measured peers per member for the latency based overlay

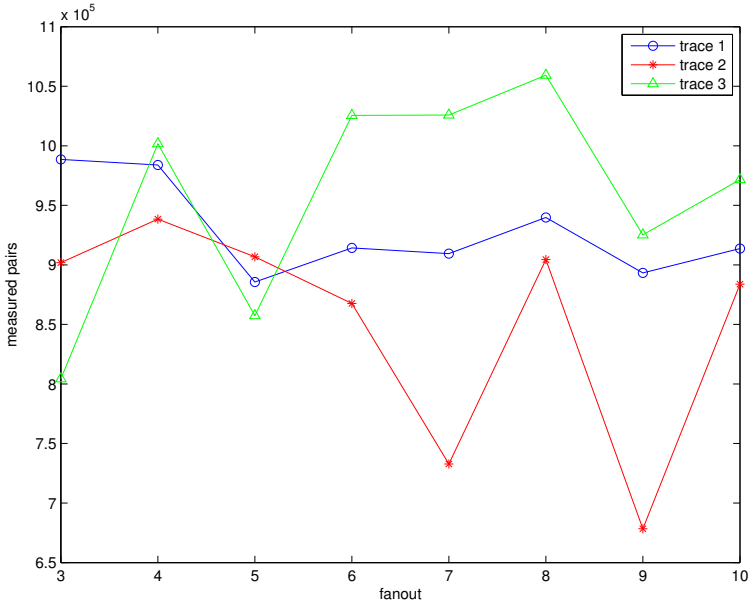

Figure 4.18: Number of measured pairs for the latency based overlay

Chapter 5

Conclusions

5.1 Summary of Results

The results presented in the previous section quantified and compared metric specific overlay performances. In this section we look at the overall results and try to understand the strengths and weaknesses of each architecture in part.

Clearly, the latency based overlay has the best overall performance but also the highest maintenance cost. The SALM topology discovery overlay manages to provide latency estimates for the right member pairs that, combined with the optimization algorithm, make this overlay management architecture the best suited for latency critical multicast packet distribution. Despite very large member hop stretch, the overlay shows low normalized tree cost and thus good network usage efficiency. It is however the scheme that requires the largest amount of message exchanges for management purposes. Should this be a constraint, or if more efficient network usage is sought, a look at the BGP based overlay should be given.

Wanting an overlay with less control traffic overhead we decided to test the ability of BGP AS hops to estimate inter-AS latency. As explained in section 3.2.3, for simplicity of deployment we opted for a *local* BGP topology view as opposed to a supposedly harder to obtain, *global* view. Results, rather surprisingly, show that AS Paths, despite doing a relatively good job at estimating latency for low fanouts, perform identically to random for large replication factors. However, this drawback is compensated by the lowest tree cost and much lower than the latency based overlay control traffic overhead. Altogether, an average stretch of around 4.3 for an overlay of routers with replication factor of 3, very low tree cost and no overhead due to topology discovery mechanism seems to be, in our view, a good result.

If the control traffic overhead is a concern, or if latency and tree cost are not necessary to be minimized, then, the random overlay offers a decent performance. We explain its not so bad results by the highly interconnected nature of ASes in current day Internet which leads to both low AS Path length and low inter-AS latency. Notably, if the processing overhead of the MS becomes prohibitive, or should the topology discovery mechanism fail to function properly, a random overlay management algorithm could be used as a fallback solution.

5.2 Concluding Thoughts

Our goal with the current work was to devise a LISP based inter-domain multicast architecture that, besides possessing a low deployment cost, is also characterized by ease of configurability and scalability. The former requirement was fulfilled by our use in the multicast overlay of just LISP enabled domain border routers, without requiring any further support or changes in the Internet's core. But, equally important, by exposing the service to the host clients by means of an already implemented protocol. Member participation in the overlay tree is conditioned by the implementation of a set of invariant functions dealing with overlay membership and responsabilities. Yet, the overlay management functions are independent and centralized in the overlay coordinator, the MS. As a result, change or switch of tree optimization algorithms can be done easily, even on-line, assuring fast (re)configuration of the overlay's performance. The isolation through design between local-domain and inter-domain multicast insures the separation of the overlay's core router members from the churn specific to client hosts and thus relieves the architecture's control plane from the inherent overhead. This ensures the scaling of the architecture with the number of clients however, the scaling with the number of member domains is attained through proper control plane design. We have tested three possible overlay management algorithms with very different control traffic requirements and concluded that they are all fit for optimizing overlays with thousands of members. Although, as expected, the complexity of the problem makes it such that better performing algorithms also require much more management overhead.

Acknowledgements

My thanks goes to my two advisors, Albert Cabellos and Jordi Domingo-Pascual, for their guidance and pacience whilst allowing me the room to work in my own way.

I would like to thank my parents for their unconditioned support and understanding of my continuous unavailability.

A big thanks goes to my girlfriend for bearing with my chaotic and nightly work schedule and foremost for being the support that I most needed.

Abbreviations

AFI	Address Family Identifier
ALM	Application Layer Multicast
AS	Autonomous System
ASM	Any-Source Multicast
BGP	Border Gateway Protocol
CCDF	Complementary Cumulative Distribution Function
DFZ	Default Free Zone
DHT	Distributed Hash Table
DNS	Domain Name System
EID	Endpoint Identifier
ESD	End System Designator
ETR	Egress Tunnel Router
IAB	Internet Advisory Board
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IGP	Interior Gateway Protocol
IP	Internet Protocol
ITR	Ingress Tunnel Router

LISP	Locator/Identifier Separation Protocol
MADDBST	Minimum Average Distance Degree Bounded Spanning Tree
MCID	Multicast Channel Identifier
MLD	Multicast Listener Discovery
MS	Map-Server
NAT	Network Address Translation
NP	Nondeterministic Polynomial time
PIM	Protocol Independent Multicast
PIM-DM	Protocol Independent Multicast Dense Mode
PIM-SM	Protocol Independent Multicast Sparse Mode
RFC	Request For Comments
RG	Routing Goop
RLOC	Routing Locator
RTT	Round Trip Time
SALM	Scalable Application Layer Multicast
SSM	Source-Specific Multicast
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
xTR	LISP Tunnel Router

Publications

List of the author's publications:

- Dimitri Papadimitriou, Florin Coras and Albert Cabellos, *Path-vector Routing Stability Analysis*, Workshop on Mathematical Performance Modeling and Analysis, June 2011
- Loránd Jakab, Albert Cabellos-Aparicio, Florin Coras, Damien Saucez and Olivier Bonaventure, *LISP-TREE: A DNS Hierarchy to Support the LISP Mapping System*, IEEE Journal on Selected Areas in Communications, vol. 28, no. 8, pp. 1332-1343, October 2010
- Loránd Jakab, Albert Cabellos-Aparicio, Thomas Silverston, Marc Sol, Florin Coras and Jordi Domingo-Pascual, *CoreCast: How Core/Edge Separation Can Help Improving Inter-Domain Live Streaming*, Computer Networks, vol. 54, no. 18, pp. 3388-3401, December 2010
- Loránd Jakab, Albert Cabellos-Aparicio, Florin Coras, Jordi Domingo-Pascual and Darrel Lewis, *LISP Network Element Deployment Considerations*, draft-ietf-lisp-deployment-01 (Work in progress)
- Florin Coras, Loránd Jakab, Albert Cabellos-Aparicio, Jordi Domingo-Pascual and Virgil Dobrota, *CoreSim: A Simulator for Evaluating Locator/ID Separation Protocol Mapping Systems*, Poster at Trilogy Future Internet Summer School 2009

Bibliography

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 75–86, 2010.

[2] "Digital TV World Revenue Forecasts," Jun 2011. [Online]. Available: http://www.digitaltvresearch.com/products/product?id=21

[3] H. Holbrook and B. Cain, "Source-Specific Multicast for IP," RFC 4607 (Proposed Standard), Internet Engineering Task Force, Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4607.txt

[4] S. Deering, "Host extensions for IP multicasting," RFC 1112 (Standard), Internet Engineering Task Force, Aug. 1989, updated by RFC 2236. [Online]. Available: http://www.ietf.org/rfc/rfc1112.txt

[5] C. Diot, B. N. 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, Jan. 2000.

[6] H. Eriksson, "Mbone: the multicast backbone," *Commun. ACM*, vol. 37, pp. 54–60, August 1994. [Online]. Available: http://doi.acm.org/10.1145/179606.179627

[7] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, 2002, pp. 205–217.

[8] Y. Chu, S. Rao, and H. Zhang, "A case for end system multicast," *Performance Evaluation Review*, vol. 28, no. 1; SPI, pp. 1–12, 2001.

[9] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, "Construction of an efficient overlay multicast infrastructure for real-time applications," in *INFOCOM 2003*, vol. 2. IEEE, 2003, pp. 1521–1531.

[10] D. Tran, K. Hua, and T. Do, "Zigzag: An efficient peer-to-peer scheme for media streaming," in *INFOCOM 2003*, vol. 2. IEEE, 2003, pp. 1283–1292.

[11] P. Francis, "Yoid: Extending the internet multicast architecture," *Unpublished paper, available at http://www.aciri.org/yoid/docs/index.html*, 2000.

[12] "Sopcast P2P Internet TV." [Online]. Available: http://www.sopcast.com

[13] "PPLive." [Online]. Available: http://www.pplive.com/

[14] "UUSee." [Online]. Available: http://www.uusee.com/

[15] "TVAnts." [Online]. Available: http://www.tvants.com/

[16] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," in *Proceedings of the 24th Conference on Computer Communications (IEEE INFOCOM '05)*, Mar. 2005.

[17] X. Hei, C. Lian, J. Lian, Y. Liu, and K. W. Ross, "A Measurement Study of a Large-Scale P2P IPTV System," *TOM*, vol. 9, no. 8, pp. 1672–1687, Dec. 2007.

[18] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt, "Measurement and modeling of a large-scale overlay for multimedia streaming," *QSHINE*, no. 1, p. 1, 2007.

[19] T. Silverston and O. Fourmaux, "Measuring P2P IPTV Systems," *NOSSDAV*, 2007.

[20] C. Wu, B. Li, and S. Zhao, "Diagnosing network-wide p2p live streaming inefficiencies," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 2731–2735.

[21] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984 (Informational), Internet Engineering Task Force, Sep. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4984.txt

[22] T. Li, "Recommendation for a Routing Architecture," RFC 6115 (Informational), Internet Engineering Task Force, Feb. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6115.txt

[23] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "Locator/ID Separation Protocol (LISP)," draft-ietf-lisp-15, Internet Engineering Task Force, Jul. 2011, work in progress.

[24] "IETF Locator/ID Separation Protocol WG ." [Online]. Available: https://datatracker.ietf.org/wg/lisp/charter/

[25] "LISP Testbed." [Online]. Available: http://www.lisp4.net/

[26] N. Chiappa, "Endpoints and endpoint names: A proposed enhancement to the internet architecture," 1999. [Online]. Available: http://www.chiappa.net/~jnc/tech/endpoints.txt

[27] J. Saltzer, "On the Naming and Binding of Network Destinations," RFC 1498 (Informational), Internet Engineering Task Force, Aug. 1993. [Online]. Available: http://www.ietf.org/rfc/rfc1498.txt

[28] M. O'Dell, "GSE - An Alternate Addressing Architecture for IPv6," draft-ietf-ipngwg-gseaddr-00.txt, 1997. [Online]. Available: http://www.watersprings.org/pub/id/draft-ietf-ipngwg-gseaddr-00.txt

[29] L. Zhang, "An overview of multihoming and open issues in GSE," *IETF Journal*, vol. 2, no. 2, 2006.

[30] D. Meyer, "Update on routing and addressing at ietf 69," *IETF Journal*, vol. 3, no. 2, October 2007.

[31] R. Hinden, "New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG," RFC 1955 (Informational), Internet Engineering Task Force, Jun. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1955.txt

[32] D. Meyer, "The locator identifier separation protocol (LISP)," *The Internet Protocol Journal*, vol. 11, no. 1, pp. 23–36, 2008.

[33] V. Fuller, D. Farinacci, D. Meyer, and D. Lewis, "LISP Alternative Topology (LISP+ALT)," draft-ietf-lisp-alt-07, Internet Engineering Task Force, Jun. 2011, work in progress.

[34] S. Brim, N. Chiappa, D. Farinacci, V. Fuller, D. Lewis, and D. Meyer, "LISP-CONS: A Content distribution Overlay Network Service for LISP," draft-meyer-lisp-cons-04, Internet Engineering Task Force, Apr. 2008, work in progress. [Online]. Available: http://tools.ietf.org/html/draft-meyer-lisp-cons-04

[35] E. Lear, "NERD: A Not-so-novel EID to RLOC Database," draft-lear-lisp-nerd-04, Internet Engineering Task Force, Apr. 2008, work in progress. [Online]. Available: http://tools.ietf.org/html/draft-lear-lisp-nerd-04

[36] L. Mathy, L. Iannone, and O. Bonaventure, "LISP-DHT: Towards a DHT to map identifiers onto locators," draft-mathy-lisp-dht-00, Internet Engineering Task Force, Feb. 2008, work in progress. [Online]. Available: http://inl.info.ucl.ac.be/system/files/draft-mathy-lisp-dht-00.txt

[37] L. Jakab, A. Cabellos-Aparicio, F. Coras, D. Saucez, and O. Bonaventure, "Lisp-tree: A dns hierarchy to support the lisp mapping system," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 8, pp. 1332 –1343, october 2010.

[38] D. Farinacci and V. Fuller, "LISP Map Server," draft-fuller-lisp-ms-10, Internet Engineering Task Force, Jul. 2011, work in progress.

[39] D. Meyer and D. Lewis, "Architectural implications of locator/id separation," draft-meyer-loc-id-implications-01, Internet Engineering Task Force, Jan 2009, work in progress. [Online]. Available: http://http://tools.ietf.org/html/draft-meyer-loc-id-implications-01

[40] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan, "Internet Group Management Protocol, Version 3," RFC 3376 (Proposed Standard), Internet Engineering Task Force, Oct. 2002, updated by RFC 4604. [Online]. Available: http://www.ietf.org/rfc/rfc3376.txt

[41] R. Vida and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6," RFC 3810 (Proposed Standard), Internet Engineering Task Force, Jun. 2004, updated by RFC 4604. [Online]. Available: http://www.ietf.org/rfc/rfc3810.txt

[42] D. Waitzman, C. Partridge, and S. Deering, "Distance Vector Multicast Routing Protocol," RFC 1075 (Experimental), Internet Engineering Task Force, Nov. 1988. [Online]. Available: http://www.ietf.org/rfc/rfc1075.txt

[43] J. Moy, "Multicast Extensions to OSPF," RFC 1584 (Historic), Internet Engineering Task Force, Mar. 1994. [Online]. Available: http://www.ietf.org/rfc/rfc1584.txt

[44] T. Bates, R. Chandra, D. Katz, and Y. Rekhter, "Multiprotocol Extensions for BGP-4," RFC 4760 (Draft Standard), Internet Engineering Task Force, Jan. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4760.txt

[45] A. Adams, J. Nicholas, and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)," RFC 3973 (Experimental), Internet Engineering Task Force, Jan. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc3973.txt

[46] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)," RFC 4601 (Proposed Standard), Internet Engineering Task Force, Aug. 2006, updated by RFCs 5059, 5796, 6226. [Online]. Available: http://www.ietf.org/rfc/rfc4601.txt

[47] M. Handley, I. Kouvelas, T. Speakman, and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)," RFC 5015 (Proposed Standard), Internet Engineering Task Force, Oct. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5015.txt

[48] D. Farinacci, D. Meyer, J. Zwiebel, and S. Venaas, "LISP for Multicast Environments," draft-ietf-lisp-multicast-07, Internet Engineering Task Force, Jul. 2011, work in progress.

[49] S. Bhattacharyya, "An Overview of Source-Specific Multicast (SSM)," RFC 3569 (Informational), Internet Engineering Task Force, Jul. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3569.txt

[50] B. Fenner and D. Meyer, "Multicast Source Discovery Protocol (MSDP)," RFC 3618 (Experimental), Internet Engineering Task Force, Oct. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3618.txt

[51] M. Hosseini, D. Ahmed, S. Shirmohammadi, and N. Georganas, "A survey of application-layer multicast protocols," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 58–74, 2007.

[52] S. Banerjee and B. Bhattacharjee, "A comparative study of application layer multicast protocols," *Network*, vol. 4, p. 3.

[53] L. Jakab, A. Cabellos-Aparicio, F. Coras, J. Domingo-Pascual, and D. Lewis, "LISP Network Element Deployment Considerations," draft-ietf-lisp-deployment-01.txt, Internet Engineering Task Force, Jul. 2011, work in progress.

[54] "IPv4 Multicast Address Space Registry ." [Online]. Available: http://www.iana.org/assignments/multicast-addresses/multicast-addresses.xml

[55] B. Haberman, "Allocation Guidelines for IPv6 Multicast Addresses," RFC 3307 (Proposed Standard), Internet Engineering Task Force, Aug. 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3307.txt

[56] F. Maino, V. Ermagan, A. Cabellos, D. Saucez, and O. Bonaventure, "LISP-Security (LISP-SEC)," draft-ietf-lisp-sec-00.txt, Internet Engineering Task Force, Jul. 2011, work in progress.

[57] S. Shi, J. Turner, and M. Waldvogel, "Dimensioning server access bandwidth and multicast routing in overlay networks," in *Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*. ACM, 2001, pp. 83–91.

[58] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of internet path selection," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4, pp. 289–299, 1999.

[59] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee, "Triangle inequality and routing policy violations in the internet," *Passive and Active Network Measurement*, pp. 45–54, 2009.

[60] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: An information plane for distributed services," in *USENIX OSDI*, Nov. 2006.

[61] University of Oregon, "Routeviews project." [Online]. Available: http://www.routeviews.org

[62] Y. Hyun, B. Huffaker, D. Andersen, E. Aben, M. Luckie, kc claffy, and C. Shannon, "The IPv4 Routed /24 AS Links Dataset - 2011-04." [Online]. Available: http://www.caida.org/data/active/ipv4_routed_topology_aslinks_dataset.xml

[63] RIPE, "Routing Information Service (RIS)." [Online]. Available: https://labs.ripe.net/datarepository/data-sets/routing-information-service-ris-raw-data-set

[64] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *SIGCOMM*. New York, NY, USA: ACM, 1999, pp. 251–262.

[65] X. A. Dimitropoulos, D. V. Krioukov, and G. F. Riley, "Revisiting Internet AS-level Topology Discovery," in *PAM*, 2005, pp. 1–13.

[66] G. Huston, "AS6447 BGP routing table analysis report." [Online]. Available: http://bgp.potaroo.net/as6447

[67] H. V. Madhyastha, E. Katz-Bassett, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane Nano: path prediction for peer-to-peer applications," in *NSDI*. Berkeley, CA, USA: USENIX Association, 2009, pp. 137–152.

[68] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An Analysis of Live Streaming Workloads on the Internet," in *IMC*, 2004.

[69] E. Veloso, V. Almeida, W. Meira, and A. Bestavros, "A hierarchical characterization of a live streaming media workload," *TON*, vol. 14, no. 1, pp. 133–146, Feb. 2006.