On the Scalability of LISP and Advanced Overlaid Services

Florin Coras

Advisor: Albert Cabellos-Aparicio, PhD
Co-Advisor: Prof. Jordi Domingo-Pascual, PhD

Department of Computer Architecture

Universitat Politécnica de Catalunya

This dissertation is submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

2015

To my family

Acknowledgements

While challenging if not impossible for me to acknowledge all those who have helped me throughout this journey, there are a few I would like to thank for making my work possible and the overall experience worthwhile.

First and foremost, I am deeply indebted to my advisor, Albert Cabellos-Aparicio, for his tireless help and pragmatic guidance that have contributed greatly to shaping this thesis. He is a great researcher of seemingly unlimited energy and determination, always ready to take ideas to the next level and capable of providing the right input and motivation needed to finish a task. Thank you for believing in my work even when I had my doubts. I would also like to thank my co-advisor Jordi Domingo-Pascual for his continuous support and patient guidance throughout the years. I consider myself fortunate to have worked with both and to have benefited both their perspectives.

I owe a special thanks to Damien Saucez. Damien is not only a direct contributor to this thesis but also an indirect one by being the smart blackboard off of which I often bounced my crazy ideas or dumb questions. Despite the bias towards the latter, he was always supportive of my research and a never-ending source of insightful advice. Damien was also instrumental in organizing my visit to Université catholique de Louvain, were Olivier Bonaventure was gracious enough to host me. A big thank you to Olivier for giving me the chance to work with and learn from him and his team; an eye opening experience, of which I have fond memories. In particular, I would like to thank Pierre Francois for his very informative class on BGP.

I have to thank Loránd Jakab, my office mate during the first two and a half years, for a couple of things: First, for introducing me together with Albert to LISP. Those days of crazy but creative working hours shared with Lori, Albert, Damien and Olivier working on LISP-TREE have been my inspiration and motivation to starting a Ph.D. And second, for being the friend I needed in a town and country whose languages, at the time, I did not even speak.

I was fortunate enough to start working early on with Dino Farinacci and Darrel

Lewis, two of the original four Cisco engineers that designed LISP. Apart from being extremely engaging and fun to work with on anything LISP related, Dino was also fundamental to the design of Lcast and LISP-RE. It is always easier to work on multicast when you known you can ask questions to a multicast expert. Darrel has always been supportive and provided great feedback to both the cache modeling and multicast works. He is also one of the most fun guys to hang out with.

Noel Chiappa and John Day are two Internet veterans who let me steal enormous amount of their time and wisdom while I was struggling to better understand the fundamental concepts in networking and the reasoning behind them. I benefited greatly from these interactions. To Noel I also owe a thank you for having provided lots of constructive feedback on an early version of LISP-MPS.

Luigi Iannone was kind enough to host me during my visit to Telecom ParisTech and his hospitality made Paris feel like home for three months. During this time I also had the pleasure of working with him, Damien and Benoit Donnet (rock on!) on measuring the LISP beta infrastructure, an entertaining project with a group of great researchers. I am also thankful to Luigi (and Damien) for showing me the ropes at the IETF meetings and for always making them enjoyable. Out of the group at TPT I would also like to thank Xavier Misseri for all the interesting chats on LISP, BGP and multicast.

Fabio Maino I must thank for his unlimited generosity and confidence. Apart from being a supporter of the multicast work, he also gave me the unique chance of working with **the** LISP team by graciously organizing and sponsoring my visit to Cisco. In particular, in Cisco I had a great time learning and working with Vina Ermagan on a slew of very interesting LISP-related projects which she always managed with amazing skill. I am grateful to both Vina and Fabio for their hospitality and for giving me such an exposure within Cisco; I am quite certain they are the best managers one could want. I also have to thank Marc Portoles-Comeras for our chats on everything network related (or unrelated).

I was lucky enough to be surrounded by a lot of people who made my stay at UPC a pleasant experience. Foremost, I would like to thank the LISP mob: Albert López Brescó and Alberto Rodríguez-Natal with whom I had the great pleasure of collaborating on all things LISP-related in general and the code of LISPmob in particular. They are proper LISP experts and great friends to have. Albert I must also thank for being a flawless sysadmin and a patient teacher while I was learning the inner workings of LISPmob. Ignacio Llatser is one of the main reasons why I managed to complete my Ph.D. His skill at swimming through the pool of paperwork saved me multiple times. To Josep Solé-Pareta, Pere Barlet-Ros and Josep Sanjuàs-Cuxart I am grateful for the helpful conversation and advice. Pere and Josep, thanks for all the traces! Thanks to the N3Cat team: Sergi Abadal-Cavallé, Raul Gomez Cid-Fuentes and Albert Mestres-Sugrañes, I will miss our daily interaction and the often not-that-nano-conversations we had over lunch, and to all past and present office mates: Valentin Carela-Español, Jakub Mikians, Ignasi Paredes-Oliva, Sergio Ricciardi, Victor López-Ferrando, Oscar Pedrola-Escribà and Pedro Pedroso: you have have made D6-008 an excellent - and at times loud - place to work. I am also thankful to all administrative staff of the DAC department for their help and to the Spanish Ministry of Education for its generous support of this thesis through the AP2009-3790 grant.

To January Tabaka, my flat mate, I owe in good part my sanity. Thanks for showing me that outside the confinements of our flat there lies the sunny and amazingly beautiful Catalan outdoors.

I will always be grateful to my parents and sister (you too, Ionuţ) for their unconditional support and encouragement. Last but not least, I would like to express my deepest gratitude to Roxi. I owe you an apology for the chaotic lifestyle, weird working hours and constant unavailability: Thanks for being there when I needed it most.

Abstract

In just four decades the Internet has gone from a lab experiment to a worldwide, business critical infrastructure that caters to the communication needs of almost a half of the Earth's population. With these figures on its side, arguing against the Internet's scalability would seem rather unwise. However, the Internet's organic growth is far from finished and, as billions of new devices are expected to be joined in the not so distant future, scalability, or lack thereof, is commonly believed to be the Internet's biggest problem.

While consensus on the exact form of the solution is yet to be found, the need for a semantic decoupling of a node's location and identity, often called a *location/identity* separation, is generally accepted as a promising way forward. Typically, this requires the introduction of new network elements that provide the binding of the two namespaces and caches that avoid hampering router packet forwarding speeds. But due to this increased complexity the solution's scalability is itself questioned.

This dissertation evaluates the suitability of using the Locator/ID Separation Protocol (LISP), one of the most successful proposals to follow the location/identity separation guideline, as a solution to the Internet's scalability problem. However, because the deployment of any new architecture depends not only on solving the incumbent's technical problems but also on the added value that it brings, our approach follows two lines. In the first part of the thesis, we develop the analytical tools to evaluate LISP's control plane scalability while in the second we show that the required control/data plane separation provides important benefits that could drive LISP's adoption.

As a first step to evaluating LISP's scalability, we propose a methodology for an analytical analysis of cache performance that relies on the working-set theory to estimate traffic locality of reference. One of our main contribution is that we identify the conditions network traffic must comply with for the theory to be applicable and then use the result to develop a model that predicts average cache miss rates. Furthermore, we study the model's suitability for long term cache provisioning and assess the cache's vulnerability in front of malicious users through an extension that accounts for cache polluting traffic. As a last step, we investigate the main sources of locality and their impact on the asymptotic scalability of the LISP cache. An important finding here is that destination popularity distribution can accurately describe cache performance, independent of the much harder to model short term correlations. Under a small set of assumptions, this result finally enables us to characterize asymptotic scalability with respect to the amount of prefixes (Internet growth) and users (growth of the LISP site). We validate the models and discuss the accuracy of our assumptions using several one-day-long packet traces collected at the egress points of a campus and an academic network.

To show the added benefits that could drive LISP's adoption, in the second part of the thesis we investigate the possibilities of performing inter-domain multicast and improving intra-domain routing. Although the idea of using overlaid services to improve underlay performance is not new, this dissertation argues that LISP offers the right tools to reliably and easily implement such services due to its reliance on network instead of application layer support. In particular, we present and extensively evaluate Lcast, a network-layer single-source multicast framework designed to merge the robustness and efficiency of IP multicast with the configurability and low deployment cost of application-layer overlays. Additionally, we describe and evaluate LISP-MPS, an architecture capable of exploiting LISP to minimize intra-domain routing tables and ensure, among other, support for multi protocol switching and virtual networks.

Resumen

En menos de cuatro décadas Internet ha evolucionado desde un experimento de laboratorio hasta una infraestructura de alcance mundial, de importancia crítica para negocios y que atiende a las necesidades de casi un tercio de los habitantes del planeta. Con estos números, es difícil tratar de negar la necesidad de escalabilidad de Internet. Sin embargo, el crecimiento orgánico de Internet está aún lejos de finalizar ya que se espera que mil millones de dispositivos nuevos se conecten en el futuro cercano. Así pues, la falta de escalabilidad es el mayor problema al que se enfrenta Internet hoy en día.

Aunque la solución definitiva al problema está aún por definir, la necesidad de desacoplar semánticamente la localización e identidad de un nodo, a menudo llamada locator/identifier separation, es generalmente aceptada como un camino prometedor a seguir. Sin embargo, esto requiere la introducción de nuevos dispositivos en la red que unan los dos espacios de nombres disjuntos resultantes y de cachés que almacenen los enlaces temporales entre ellos con el fin de aumentar la velocidad de transmisión de los enrutadores. A raíz de esta complejidad añadida, la escalabilidad de la solución en si misma es también cuestionada.

Este trabajo evalúa la idoneidad de utilizar Locator/ID Separation Protocol (LISP), una de las propuestas más exitosas que siguen la pauta locator/identity separation, como una solución para la escalabilidad de la Internet. Con tal fin, desarrollamos las herramientas analíticas para evaluar la escalabilidad del plano de control de LISP pero también para mostrar que la separación de los planos de control y datos proporciona un importante valor añadido que podría impulsar la adopción de LISP.

Como primer paso para evaluar la escalabilidad de LISP, proponemos una metodología para un estudio analítico del rendimiento de la caché que se basa en la teoría del working-set para estimar la localidad de referencias. Identificamos las condiciones que el tráfico de red debe cumplir para que la teoría sea aplicable y luego desarrollamos un modelo que predice las tasas medias de fallos de caché con

respecto a parámetros de tráfico fácilmente medibles. Por otra parte, para demostrar su versatilidad y para evaluar la vulnerabilidad de la caché frente a usuarios malintencionados, extendemos el modelo para considerar el rendimiento frente a tráfico generado por usuarios maliciosos. Como último paso, investigamos como usar la popularidad de los destinos para estimar el rendimiento de la caché, independientemente de las correlaciones a corto plazo. Bajo un pequeño conjunto de hipótesis conseguimos caracterizar la escalabilidad con respecto a la cantidad de prefijos (el crecimiento de Internet) y los usuarios (crecimiento del sitio LISP). Validamos los modelos y discutimos la exactitud de nuestras suposiciones utilizando varias trazas de paquetes reales.

Para mostrar los beneficios adicionales que podrían impulsar la adopción de LISP, también investigamos las posibilidades de realizar multidifusión inter-dominio y la mejora del enrutamiento dentro del dominio. Aunque la idea de utilizar servicios superpuestos para mejorar el rendimiento de la capa subyacente no es nueva, esta tesis sostiene que LISP ofrece las herramientas adecuadas para poner en práctica de forma fiable y fácilmente este tipo de servicios debido a que LISP actúa en la capa de red y no en la capa de aplicación. En particular, presentamos y evaluamos extensamente Lcast, un marco de multidifusión con una sola fuente diseñado para combinar la robustez y eficiencia de la multidifusión IP con la capacidad de configuración y bajo coste de implementación de una capa superpuesta a nivel de aplicación. Además, describimos y evaluamos LISP-MPS, una arquitectura capaz de explotar LISP para minimizar las tablas de enrutamiento intra-dominio y garantizar, entre otras, soporte para conmutación multi-protocolo y redes virtuales.

Contents

List of Figures

List of Tables

Nomenclature

Roman Symbols

AFI	Address Family Identifier
ALM	Application Layer Multicast
AS	Autonomous System
ASBR	AS Border Router
ASM	Any-Source Multicast
BGP	Border Gateway Protocol
CCDF	Complementary Cumulative Distribution Function
CDF	Cummulative 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
GZipf	Generalized Zipf (see [105])
IAB	Internet Advisory Board
IANA	Internet Assigned Numbers Authority

ICN	Information-Centric Networking
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IGP	Interior Gateway Protocol
ILNP	Identifier-Locator Network Protocol
IP	Internet Protocol
ISP	Internet Service Provider
ITR	Ingress Tunnel Router
LCAF	LISP Canonical Address Format
LFU	Least frequently used
LISP	Locator/ID Separation Protocol
LRU	Least Recently Used
MADDBST	Minimum Average Distance Degree Bounded Spanning Tree
MCID	Multicast Channel Identifier
MLD	Multicast Listener Discovery
MPLS	Multiprotocol Label Switching
MS	Map-Server
MTU	Maximum Transmission Unit
NAT	Network Address Translation
NICE	NICE is the Internet Cooperative Environment
NP	Nondeterministic Polynomial time
PA	Provider Aggregatable
PI	Provider Independent
PIM	Protocol Independent Multicast

PIM-SM	Protocol Independent Multicast Sparse Mode
PxTR	Proxy Tunnel Router
RFC	Request For Comments
RG	Routing Goop
RINA	Recursive InterNetwork Architecture
RLOC	Routing Locator
RTR	Re-encapsulating Tunnel Router
RTT	Round Trip Time
SALM	Scalable Application Layer Multicast
SDN	Software-defined Networking
SMR	Solicit Map-Request
SSM	Source-Specific Multicast
TCP	Transmission Control Protocol
TTL	Time to Live
UDP	User Datagram Protocol
VPN	Virtual Private Network
xTR	LISP Tunnel Router

Part I

Introduction and Background

Chapter 1

Introduction

The Internet is an ever evolving system; constantly having new hosts and networks attached to it in what resembles a continuous organic growth. This ongoing evolution becomes clear by tracing its history back to the Internet's humble beginnings, but also by paying close attention to what people expect of it in the near future. Even though it would be hard to predict how the Internet is to look like or how we are to use it in the not so distant future, it seems to be a safe bet to assume that it will be considerably different from the one we know today.

The Internet started roughly four decades ago, and initially only a few research networks and their gateways (i.e., what we call today routers) where attached to it. Inter-gateway packet exchanges were ensured by a routing protocol but, although it connected more than one institution, it viewed and helped exchange reachability information as if the whole topology was a single routing domain. In the face of imminent growth, a more scalable distributed routing system was developed. The solution was to abstract a domain's network complexity to a single point, an Autonomous System (AS), within an inter-domain map. The most recent, and in use at the time of this writing, implementation of this routing protocol is the Border Gateway Protocol (BGP).

Over the course of its evolution, similar crises driven by growth or change in use patterns have led to a better understanding and improvement of congestion control, queue management and addressing among other. It would thus appear as if the Internet always follows a minimalist design that it gradually refines, in accordance to new contexts defined by the needs of its users and the economical constraints of the operators providing the service. Many seem to agree that this simplicity and low adoption barriers that walk the thin line at the intersection of complexity and scalability have been the seeds of its success and thus, those that ultimately made it the most complex distributed system ever developed by humanity. In the future, this growth is expected to continue and, in light of the technological innovation of recent years, its rate will all but diminish, as it is envisaged that various mobile accessories and even smaller devices are to be attached to it.

Unfortunately though, this exponential growth imposes a huge stress onto the routing system, which has to manage an always-increasing and more variable Internet default-free zone (DFZ) routing table. In fact, the research community agrees that it is once again out-growing its original design. Evidence for this is widespread and the problem is affecting all the stakeholders in various ways. As stated in a presentation done to the North American Network Operator's Group: *"For service providers, the Internet is about to become a lot more expensive to deploy and operate; for users, the Internet is about to become a lot less reliable and a lot more expensive (and balkanized)"* [102].

Fixing the Internet however is no easy task. First, the solution must be highly scalable and able to cope with the size and dynamic nature of the routing infrastructure. Second, the Internet has grown organically, pushed by economical and social forces, therefore not always following a technical criterion. As a result, these non-technical aspects that sum up to a considerable amount of inertia favoring architectural stagnation must also be considered when developing a solution. One good example of how the interplay between these two points influence the evolution of the Internet is IPv6 deployment. Although the research community agreed that IPv6 can technically solve IPv4's address exhaustion problem, more than 15 years were required, since IPv6's inception, and the complete IPv4 address space depletion, for IPv6 adoption rate to pick up momentum.

Opinions on how to solve the growth of the routing tables, and in general, how to improve the Internet's architecture, are split and at odds. On one side, there are those in favor of a clean-slate, complete architecture redesign, in the context of a quest for a better understanding of the fundamentals of networking. On the other, there are those arguing for an evolutionary (or incremental) approach that also satisfies backwards compatibility [114].

Although clean-slate solutions have their advantages, most notable being their ability to produce long-term aiming results without the burden of the past, it is also a fact that fundamentally changing the established practice is hard. Moreover, clean-slate designs would need to pass through an evolutionary process, from incipient form to global scale, a refining process very much like the one experienced by the current Internet architecture. To a certain extent then, following a clean slate path is prone to repeating some of the past errors and quite probable to initially result in a least reliable architecture. In contrast, in spite of its limitations, the incumbent routing infrastructure is already in a mature state and under the impulse of a well pondered update could evolve toward an improved and stable form.

This dissertation argues that performing a semantic separation of the two roles held by IP, namely, those of *locator* and *identifier* is a sufficient incremental step towards solving the routing scalability problem. In particular, we show that implementing the Locator/ID Separation Protocol (LISP), one of the architectures implementing a location/identity split, has two important benefits that make it an attractive short-term architectural enhancement:

- It ensures the scalability of the routing system for today's type of network traffic.
- It enables the (re)implementation of network functions in a scalable fashion. Specifically, we show how both multicast and intra-domain routing can be implemented as overlays.

In the remainder of the chapter we survey some of the causes for the routing scalability problem, pinpoint the problems with upgrading the Internet and summarize our contribution.

1.1 Motivation and challenges

Beyond the growth attributable to organic development, by itself not necessarily unmanageable, there are several other causes supported by established operator best practices that break topological aggregation and drive routing table growth [99]. In this sense, to avoid network *renumbering* operators prefer using Provider Independent (PI) address space (non topological prefix allocations), as opposed to Provider Aggregatable one, since it offers transparency to upstream provider changes. *Multihoming* is another common practice, typically employed to ensure failure resilience for business critical applications, when operators buy transit services from multiple upstreams and advertise a protected prefix of choice through all. Hence, independent of the type of addresses used (PI or PA), multihoming always leads to an increase of the routing table size; the damage being only marginally lower when the protected prefix is PA since the provider still manages to advertise an aggregate.

Finally, another important growth driver is *traffic-engineering.* Looking to spread incoming traffic across multiple links, network operators often de-aggregate prefixes in order to steer traffic of a particular address range over specific paths. There are also known situations when de-aggregation was used to prevent prefix hijacking.

Apart from driving routing table growth, these practices further contribute to the worrisome increase of number of routing table updates (BGP churn) [51], which in turn translates to increased router CPU usage and convergence times [109] exceeding several minutes. Given that prefixes suffering such outages often lose global reachability, this can result in important service downtime.

It could be argued that all of the previous problems are the result of operators abusing existing routing mechanisms and it is somewhat fair that they pay the due price. However, it has been shown that there exists a clear misalignment of benefits and costs as such practices are often exploited by few but the costs are borne by the whole infrastructure [73]. And, as a counter point, it is remarkable that despite the reliance of many of our critical services on the Internet, the existing architecture cannot scalably support these simple network functions. Then, because of critical implications concerning future resilience, maintenance and attachment costs for operators and quality of service experienced by clients, solving the routing table growth should become paramount for the short term stability of the Internet.

One approach to lowering the number of routing table entries, in particular the size of the Forwarding Information Base (FIB), is to perform in router prefix aggregation. For instance, *FIB Aggregation* is an opportunistic technique that offers per router FIB size reductions by algorithmically removing specific forwarding (child) entries which share the same next hop with their trie ancestors. The procedure ensures forwarding correctness however, depending on the employed algorithms, it may introduce previously non-routable address space in the FIB. There are several proposals [27, 50, 88, 143] that recommend the use of these techniques for reducing routing table sizes. Notably, [143] presents a systematic analysis of costs and benefits for FIB aggregation and concludes that it is a viable short-term solution.

Similarly, Virtual Aggregation [17] tries to diminish the routing tables of routers within an autonomous system (AS) by having legacy routers forward their traffic to several *aggregation point routers* (APRs) instead of the best egress points. The forwarding on this second section (from the APR to the AS border router) is done by using MPLS tunnels in order to avoid routing loops. As a result, the number of FIB entries in legacy routers is limited to the number of APRs. As downside, it introduces additional path-stretch within the AS. Another drawback of all FIB aggregation based solutions is their high CPU resource consumption on routing table updates.

Despite being efficient at treating the effect (the growth), these solutions do not remove the source of the problem which is the architecture's inability to topologically aggregate prefixes. A statement that eloquently sums up the situation, and is colloquially referred to as "Rekhter's Law", pertains to Yakov Rekhter: *"Addressing can follow topology or topology can follow addressing. Choose one."* Or in other words, for the Internet's routing system to scale, addressing (i.e, IP allocations) must be congruent with the topology describing inter-operator connections. Unfortunately, experience has shown that address allocations for endpoints follow an organizational, not topological, structure and, due to IP's dual semantics of both *identifier* and *locator* of an endhost, *"a natural incongruence arises"*. In fact, Quoitin *et al.* have experimentally shown [112] that separating these functions (by implementing LISP) could greatly reduce core router FIBs as it would once again allow hierarchical aggregation.

Although consensus on how this indirection level could be implemented is yet to be found, it is generally accepted that separating the identity and location namespaces is an important step towards improving the routing architecture, even if challenging [89, 95, 99, 145]. Nevertheless, what precludes the Internet's passing to one of the many proposed architectures built around this paradigm (for a review see [88] or Section 2.2) is the answering of at least two fundamental questions:

1. There exists the question about the system's scalability. Separating location and identity solves the problems of the current inter-domain routing system but in doing so it introduces new network elements, notably a global distributed mapping database and mechanisms for scalable querying of locators to identifiers bindings. There are not few those who have questioned the scalability of these *control plane* mechanisms and thus see the split as an attempt at shifting the problem from the routing architecture to edge devices that must interact with the mapping system. It is therefore fundamental to provide a better understanding of the control plane's performance and eventually a characterization of its scaling properties.

2. The deployment of any new architecture is not only dependent on its abilities to solve the problems of the widely deployed incumbent but also on the added value that it brings. This exact point is made quite clearly by Dovrolis in [114]. He argues that the relevant question to ask is not if a solution is su-

> perior technologically to current practices but whether if the new technology offers new services that the current one cannot. Industrial economics generally dictates that inability to provide such benefits impedes deployment, whereby it is also critical to understand the extended set of benefits provided by a new architecture to ensure its adoption.

Out of the plethora of solutions to implement a locator/identifier separation, we think Locator/ID Separation Protocol (LISP) is the one showing the greatest promise and thereby the one on which we further focus our studies and contributions. Notably, LISP enjoys both academic and industry support, the result of which is considerable joint effort in protocol development and implementation, but equally important, it has a complete implementation for the Cisco NX-OS and IOS platforms, an interoperable open source implementation for Linux/OS, Android [10], FreeBSD [111] and an open source LISP Mobile Node implementation [10].

1.2 Thesis overview and contributions

The objective of this thesis is to study the suitability of using LISP, as a particular instantiation of a semantic decoupling of location and identity at network layer, with the aim of improving the current inter-domain routing architecture. To do so, we answer the two questions posed in the previous section.

In LISP control plane, routers retrieve mappings on user demand, as opposed to proactively fetching them. This is done such that the amount of memory a router requires to participate in the system does not grow with identifier space, as is the case today, but is instead dependent on the packet level traffic the router processes. As a result, to diminish retrieval times, increase packet forwarding speed and to protect the mapping system from floods of resolution requests, routers are provisioned with mappings caches (*map-caches*) that temporarily store in use bindings. Thereby, as a first step to understanding the control plane's scalability we must first understand map-cache performance.

Although caches placed between processor and main memory, in operating systems or in web proxies are well studied [11, 23, 116], route and mappings caches have yet to be thoroughly analyzed. A considerable number of experiments have empirically evaluated map-cache performance, however they are mainly focused on providing a circumstantial description of cache behavior, that is, for particular cache configurations and network traffic traces, as opposed to a general one [75, 78, 83, 84, 140].

Typically, these results yield accurate estimates for cache performance but unfortunately cannot be extrapolated to provide rough projections or bounds on cache behavior for workloads with different characteristics; nor can they provide insight into what traffic properties influence cache performance and to what degree. Answering such questions would not only be a first important step towards understanding the overall performance of the mapping-system, but would also provide a quick way of gauging the expected map-cache performance of any network domain.

In this thesis, we present an analytical model that, to the best of our knowledge, constitutes the first theoretical framework for map-cache performance analysis and provisioning. The model relies on coarse traffic parameters and aims to be applicable to a wide range of scenarios. In particular, we first show how the working-set theory [45] may be used to estimate simple parameters that characterize the intrinsic locality of network traffic and thereafter explain how they can be leveraged to link cache size and miss rate. The underlying assumption that enables the analysis is that traffic can be approximated as having a stationary generating process. We find stationarity to hold for real network traffic, and, to facilitate the use of the model, we also devise a simple methodology that tests for it in network traces. We validate the result by emulation, using packet traces collected at the edges of a campus and an academic network.

To understand if the model can be used for long-term cache provisioning and the cache's vulnerability to attacks, we exploit our result to (*i*) perform an in-depth, over time analysis of cache performance for our datasets and (*ii*) study the security of the map-cache by evaluating the effects of scanning attacks.

For the sake of clarity, we focus our analysis on the performance of LISP map-cache. Nevertheless, the results are relevant for other architectures inspired by the location/identity split paradigm, including those like ILNP [15] that use DNS as their mapping system, since the equations could be used to approximate DNS resolver caching performance. Moreover, the cache models could be applied to route-caching and scalability techniques that focus on shrinking routing tables to extend router lifetimes [17].

As a second step to understanding the scalability of LISP's control plane, we investigate the main sources of temporal locality of reference in network traffic. By leveraging previous results [23, 81] we show that for our datasets it is feasible to entirely characterize the average working-set solely from the destination popularity distribution without having to factor in much harder to model short term correlations. Then, by using this result to extend our initial cache model, we finally obtain a model that allows us to reason about cache performance scalability with respect to Internet growth (number of address prefixes) and LISP site growth (number of local clients) on cache size. Surprisingly, under a set of simple constraints, we find control plane performance to be virtually independent of both types of growth, that is, map-cache performance should scale constantly, $O(1)$, regardless of the number of destination prefixes and the number of local clients. However, if the constraints are not met, then cache size grows linearly, $O(N)$, in the worst case.

The second question that needs to be answered when discussing deployability, concerns LISP ability to support new network functions. However, given the open ended nature of this query, we do not focus on providing a general answer but instead, thinking in more practical terms, offer a specialized one. For instance, it has been previously shown that LISP can be efficiently leveraged to improve routing diversity and traffic engineering for stub networks [104, 122, 123]. Thereby, another aim of this thesis is to make good use of this rare window of opportunity that LISP deployment could present and (re)design protocols or network services that are well-known but subject to unsolved problems. Perhaps the most important features LISP introduces are the control/data plane separation and the use of automatic tunneling to cross the core of the Internet. The former allows the implementation of complex policies within simple data plane devices, conceptually similar to some of the ideas being used in Software Defined Networking (SDN) [97], whereas the latter enables the transiting of new address families over legacy ones and opens the possibility to performing complex re-routing based on source or control plane policy. Thereby, together, they may be easily used to deploy complex overlaid services.

The idea of using overlays to improve underlay network performance is not new, in fact, it has been exploited, among others, for improving resilience [14] or for implementing application layer multicast (see Section 7.6). This thesis however argues that LISP offers the tools to deploy such services with little added complexity and improved reliability due to the reliance on network instead of application layer support. To prove this point we show how LISP may be used to implement *i*) Lcast, a scalable architecture for inter-domain multicast, not hindered by the deployment issues of IP-multicast, with better client performance and improved operator configurability than application layer multicast (ALM) *ii*) LISP-MPS, a BGP-free operator core with improved support for intra-domain routing, multi-protocol switching and virtual networks. To evaluate Lcast we make use of extensive simulation supported by an Internet-like AS level topology and large client traces that emulate realistic user behavior, obtained through a globally distributed capture of SopCast [5] overlays.

In the case of LISP-MPS, we use BGP traces obtained from RouteViews [134] and Rocketfuel [7] to show that traffic engineering possibilities of an AS are drastically improved.

In summary, the main contributions of this thesis are as follows:

- We devise a practical, simple to use map-cache model that approximates cache miss rates from traffic working-sets. We use the model to understand how operational provisioning should be performed but also extend it to account for and evaluate cache polluting attacks.
- We show that working-sets can be efficiently approximated from destination popularity distribution. This enables us to extend our map-cache model and evaluate LISP control plane scalability.
- We design and evaluate a LISP-based multicast framework meant to merge the benefits of IP multicast and those of ALM.
- We design and evaluate a LISP-based BGP-free operator core architecture.

1.3 Thesis outline

The remainder of this thesis is structured as follows. The next chapter presents the necessary background for the routing scalability problem, including a description of the location/identity separation paradigm and the solutions derived thereof. Chapter 3 presents in greater detail the LISP architecture and associated protocol mechanisms. The following chapters, divided in two parts, present the four main contributions of this thesis. Chapter 4 introduces our cache model, which constitutes the first fundamental block of our LISP control plane scalability analysis, together with a validation of our result using real packet races. This Chapter is based on our work published in [35, 37]. Chapter 5 shows the model's extension for polluting attacks and it is based on work published in [37]. Chapter 6 presents the relationship between temporal locality and cache performance and discusses some scalability aspects of LISP's control plane. The results are still under review. In Chapter 7 we present Lcast, our scalable inter-domain multicast architecture along with an extensive performance evaluation and the description of the experimental setup. Most of this chapter has been presented in [38] while the datasets have been presented in [33]. In Chapter 8 we present LISP-MPS, discuss the architecture's benefits and evaluate its potential. All this Chapter is based on work published in [36]. Finally, Chapter 9 concludes the thesis and presents ideas for future work.

Chapter 2

Background

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 were facing regarding the scalability of the Internet routing system. The outcome of the meeting, their findings and suggestions, were summed up in RFC 4984 [99]. 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

Although tempting to consider them as complementary, their relation is in fact causal and, as it will be later seen, the latter is among the chief causes for the former. Having this said, we start by elaborating the first problem, expanding on the issues currently faced by the global routing infrastructure and their apparent motives, and thereafter descend into the more subtle reasons that triggered the predicament. Finally, we dedicate the rest of the chapter to revising some of the most important incremental and clean-slate solutions to the problem.

2.1 Routing System Scalability Problem

When analyzing the scalability of the routing system one may use two dimensions over which to characterize its performance: *i*) the size of the inter-domain routing table and *ii*) the amount of updates (or churn) that it is exposed to. While tracing the time evolution of the first provides a rough estimate of the system's ability to absorb topological growth and thus to limit router memory use, the joint analysis of both parameters furnishes insights into the processing burden of routers and the expected convergence time of BGP.

Historically, the Internet has had several periods of super-linear growth (see Figure 2.1) and most certainly a super-linear growth over the past two and a half decades. However this fast paced growth raises concerns about the future management costs to be incurred to operators and even about the ability to develop routers at an acceptable price point. As shown in [107], there are several aspects that need to be considered when discussing the problem. First, technically, it is becoming harder for routers to maintain an ever increasing and more dynamic state information all while forwarding at line rate and ensuring acceptable BGP convergence time frames. Second, from business perspective, it would be preferable to limit the self-induced cost of scaling however the converse is observed today. Namely, operators see their infrastructure costs driven up due to factors outside their control. Finally, several operator practices that involve routing table growth lack a natural negative feedback loop needed to balance costs and benefits.

To illustrate these points: at the time of this writing, the Internet is composed of approximately $47k$ ASes however the FIB of routers participating in the DFZ has already passed half a million entries. Considering that that vast majority of ASes (about $40k$) are stubs, i.e., they should only advertise reachability information for the address space they own, this order of magnitude difference indicates that the majority of the routes advertised serve other operational purposes.

In theory, aggressive aggregation [62] should minimize the number of prefixes in the DFZ however, in practice, strict adherence to CIDR has proven to be difficult [107] for some of the reasons we discuss next.

Stub (or edge) networks can obtain their address space either leased from a transit operator, these prefixes are called Provider Aggregatable (PA), or directly from a Regional Internet Registry (RIR), in which case, the addresses pertain to operators and are referred to as Provider Independent (PI). PA prefixes are generally allocated out of a larger, transit provider owned prefix and thereby follow a topological allocation, i.e, client advertisements are aggregated by the provider and result in only one prefix being announced into the DFZ. Conversely, PI, does not follow topological allocations, and for each newly attached edge site, the DFZ routing table size is incremented. For reasons having to do with the complexity of performing renumbering, customers generally prefer to use PI address space and therefore foster the

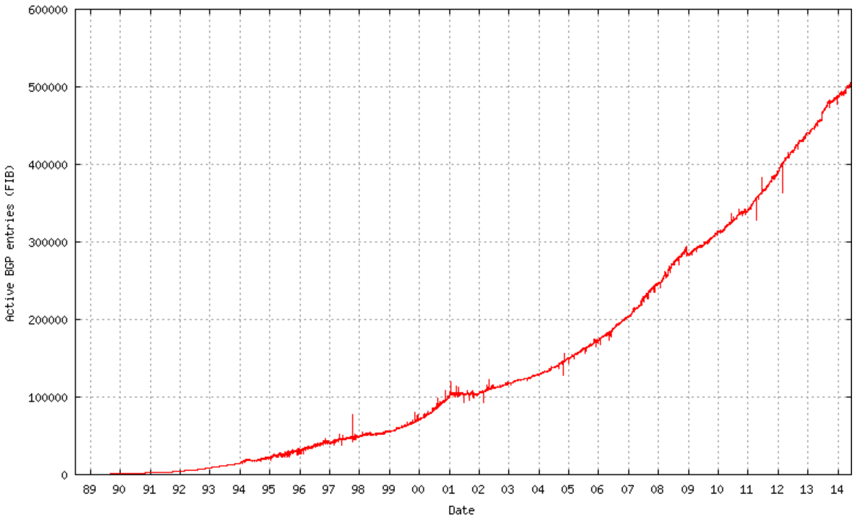


Fig. 2.1 Routing table (FIB) growth from 1989 to 2014. *source:* potaroo.net

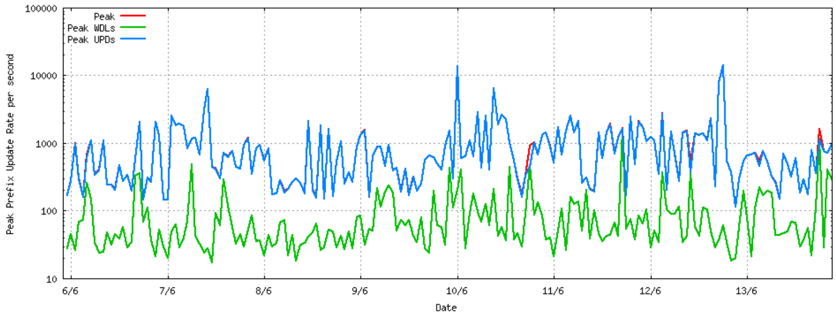


Fig. 2.2 Peak routing table updates for one week June 2014. *source:* potaroo.net

break-up of topological addressing.

Multihoming is generally described as the practice of using multiple providers, with the aim of improving failure resilience of mission and business-critical applications or to enable load-sharing and more complex performance objectives. Multihoming can be accomplished with both PA and PI address space and consists in advertising the site owned address space through all providers. This address space cannot typically be aggregated, save when it is PA and then only by the provider but, interestingly, at the cost of losing the customer's traffic. That is, due to forwarding being longest prefix match, the provider's aggregate prefix will always be discarded in favor of more specific (longer) prefixes advertised through the remaining transit providers. In such situations, the PA provider may at times prefer to advertise both aggregate and specific prefixes not to lose revenues. Hence, irrespective of the address space being PA or PI, multihoming results in more prefixes being advertised into the DFZ.

Traffic engineering (TE) consists in driving traffic over certain policy (e.g. capacity) selected paths typically for an economical end. In particular, inbound TE is performed by operators in order to control how packets ingress their networks and it is generally meant for: *i*) spreading, or load balancing, traffic over multiple links or *ii*) in order to ensure that certain policies are met. Given that routing is done on a longest prefix match, this is achieved by advertising de-aggregated prefixes (more-specifics) along the path of choice, to steer traffic away from the path it would otherwise choose. This then, again results in more prefixes being advertised into the DFZ.

An obvious implication of the DFZ routing table size growth is the increase of the number of routing updates that need to be processed in routers per unit of time. An interesting aspect to be considered here is that, in spite of the obvious implication linking routing table size increments to larger churn, the routing table size is actually physically bounded whereas the churn is unbounded. In fact, absolute values were already noteworthy at $500k$ updates per day with a peak arrival of $1k$ per second in 2006 and can reach peaks of $10k$ per second today (see Figure 2.2). However, of equal concern is the effect of the widespread use of de-aggregation that results in the presence of very specific prefixes (/24s) in the core routing tables. Therefore, updates affecting a very limited number of users (at most 253) are propagated by BGP to all of the DFZ participants. In other words, the core routers are exposed to the dynamics of the edge networks and in consequence the convergence times of BGP are detrimentally affected. Worse, these updates can, and measurements have shown them to be, generated by only a subset of ASes [73].

2.2 Location/Identity Separation

In response to the publishing of IAB's workshop conclusions the Routing Research Group (RRG) of the Internet Research Task Force (IRTF) was re-charted to elaborate a new inter-domain routing architecture that could solve all of identified problems but also better withstand the test of time. Unfortunately, the group never reached consensus on the technical details of an ideal solution so, instead, a slew of architectures were scrutinized and presented as possible steps forward in the group's report [88].

Still, a "rough consensus" existed that a split between locator and identifier roles of IP was much needed. Moreover, a shared conclusion of the two research groups was that a solution to the routing scalability is necessarily a cost/benefit trade-off. That is, no clear solution exists, and choosing one should involve a clear understanding of the inherent drawbacks. Following such a detailed analysis of existing approaches, the possibility of using a semantic separation of locator and identifier functions of IP, in essence by introducing another level of indirection, was deemed worthy of special attention as it showed a high potential for gains. Nevertheless, implementing such split is far from trivial as the paradigm only outlines a set of guidelines, whereas in practice several far reaching design decisions must be taken. Apart from the obvious need to design and deploy a distributed database for location to identity bindings lookups, most important points to be clarified are:

- The establishment of the topological boundary, where translation from one namespace to the other must be performed. Despite being intuitively understood that locators should be used within the core routing infrastructure, it is debatable if identifiers should bear scope at autonomous system level, within a smaller administrative domain or only at end-host level. Note that this choice also poses constraints on identifier syntax, since scope implies the array of devices that must be updated to understand the syntax of choice.
- Decide how forwarding between namespaces is to be performed. Although syntactically, identifiers and locators could be identical, their semantics should be different. Thereby, routing on identifiers in locator space, or vice versa, should not be possible.
- Incremental deployment mechanisms.

In this section we first provide an overview of naming and addressing issues of IP and then revise a set of proposals that try to solve these problems by implementing a semantic separation of location and identity.

2.2.1 Endpoints and Endpoint Names

Chiappa introduces in [31] the concept of *endpoint* to solve what he believes, and what was previously identified by Soch [127] and Saltzer [118] in their respective works, 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.* In his opinion, 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 instantiation of the name has stemmed confusion. Both Chiappa and Saltzer [118] explain this by the small number of well defined concepts at hand, which finally led to IP addresses being the only *names* used throughout the TCP/IP architecture, despite pointing to multiple objects. Namely, IP addresses are:

- 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.

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 mobility. Considering the case of end-host mobility, this happens because a TCP connection actually results in a static pathname [41] being formed across the layers, such that the connection endpoint 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 result in the breakup of the communication channel.

To set aside the confusion, Chiappa proposes better bounded definitions for all three possible meanings of *address*. 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*, defined as a participant of an en end-to-end communication.

Despite some limitations we discuss in Section 3, these ideas have since been incorporated by location/identity separation advocates in a slew of architectures meant to improve the Internet's scalability. We briefly discuss two classes of implementations in the next section.

2.2.2 Address rewriting

The idea was originally proposed by Dave Clark and later by Mike O'Dell in his 8+8/GSE [108] 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 consequences 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*. 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. Conversely, 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.

A first clear benefit is that this type of *insulation* provides a site with flexibility of re-homing and multihoming. Moreover, it 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'" (LS). The Routing Goop, in an address, would have the purpose of specifying the path from the root of the tree, or otherwise an LS, to any point in the topology. In the terminal case that point would be a Site. These LSs, 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, significantly decreasing the minimally-sufficient information required for a router when doing forwarding. For further details related to the structures within the IPv6 address and also possible solutions to re-homing and multihoming, reading the internet-draft [108] is highly recommended.

Given the age (this solution has been suggested in 1997) and also the lack of technical solutions, at the time, for some of the proposed mechanisms, 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 [100, 141].

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 impossible to determine if one of the border routers at the destination has lost connectivity to the site. Thus, as a solution, it was proposed that a site's border routers be manually configured to form a group and when one loses connectivity to the client site, it should forward packets to others still connected. Note that this is not an issue specific to GSE but to all solutions that propose a split between the edges and transit routing domains.

It was originally envisaged that the Internet's topology should resemble a tree anchored by LSs, with "cut-through" links between any two LS bellow the top level being seldom. However, 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.

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 original 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 thorough examination of tunneling operations is needed in GSE context.

Some of the design principles employed by GSE have later been reused by other proposals. Notably, Identifier-Locator Network Protocol (ILNP) [16], leverages address rewriting and the DNS as a mapping system implementation, but at end-host level. The main drawback to this approach is that adopting sites cannot stop announcing their PI prefixes into the DFZ up to when the entire site, that is all the end-hosts, have transitioned their software stacks to ILNP. A more detailed critique can be found here [88].

2.2.3 Map-and-Encap

The idea, as originally proposed by Robert Hinden in his ENCAP scheme [66], speaks about splitting the current single IP address space in two separate ones: the end-host identifier, and the one used for transit between the domains, the routing locator space. The goal is to achieve aggregation in the core routing infrastructure and, eventually, routing scalability, through the decoupling of identifier, i.e., non topological aggregatable space, from locator, provider owned and aggregatable space.

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 identifier of the initiator and as destination address the identifier of the peer host, that could have been obtained by means of DNS. Next, the border router *maps* the destination identifier to a locator, 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 identifiers must be routable.

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 scenarios [101]. As downsides, this model, as the address rewriting one, has problems in handling border link failures and the overhead implied by the encapsulation is stemming controversy.

2.3 Clean-Slate Architectures

Another approach to improving the Internet's architecture, often advocated in the research community, is the complete, "clean slate" redesign with the aim of building a new architecture, significantly better - in terms of performance, security, resilience - free of the considerable minutiae of today's protocols, operational practices and the challenges of incremental deployment [114]. Since such effort will most probably entail more scalable designs, we briefly introduce in what follows a couple of proposals, currently under consideration within academia.

2.3.1 RINA

The principles behind the Recursive InterNetwork Architecture were first set in John Day's book *Patterns in Network Architecture: A Return to Fundamentals* [42] and have as point of departure Robert Metcalfe's idea that *Networking is inter-process communication (IPC)*. Starting from this simple premise, RINA views networking not as a layered set of different functions (as per the OSI or TCP/IP models) but rather a single layer of distributed IPC that repeats over different scopes [43]. All IPC instances implement the same functions but are configured to operate with different performance constraints (e.g., capacity, delay, loss).

Following Operating Systems principles, where an IPC facility allows local inter-process communication, in RINA a distributed IPC facility (DIF) allows application processes on different hosts to communicate and share state. Then, although both TCP/IP and RINA use a layered design, they differ in what functions they expect layers to provide. For instance, if with TCP/IP the network layer provides the transport layer with packet delivery across the Internet, with RINA it is argued that the two should be part of the same DIF that offers services to application processes. That is, a DIF should provide a protocol that implements an IPC mechanism but also a protocol for managing distributed IPC (i.e., routing, security and other management tasks).

An important property stemming from the repeating IPC structure assumed by RINA, is that IPC processes within an IPC facility are in fact application processes requesting service from the IPC layer bellow. As such, IPC facilities repeat, starting from application layer until mapped to physical medium, a variable number of times that ultimately depends on the type of services required and the underlying physical network. Intra-layer communication is done between application processes using unique names however, since there is no name leak from upper to lower layers (as is the case today with sockets), application names must be unique only within their respective DIF. Consequently, the only global namespace required is the uppermost application layer while the lower ones may use private addresses, without any of the drawbacks of Network Address Translation (NAT).

Forwarding is done in a two-step logical process. First, routes, i.e., sequences of nodes computed over graphs abstracting network topology at a given layer, are obtained. Second, once a next-hop is known (from the route), the node address to point of attachment address(es) mapping of the neighbor can be computed, so a path in the underlying layer to the next-hop can be selected. This distinction is very similar to the one made by location/identity separation solutions whereby identity of a node is used to find its attachment point (or location) within the network by means of a mapping system. However, the important difference resides in how this mechanisms is used. If in the latter case, the mapping is computed only once, at the boundary of the two namespaces (i.e, core and edge networks), for RINA the process repeats in all the nodes on path between source and destination and part of a given DIF.

From this perspective, location/identity separation solutions, as discussed in the previous sections, could be seen as a particular instantiation of a two DIF architecture where all routes within the identity space are sequences of only two nodes, so where the next-hop is always the destination. One could then view the core of the Internet as the underlying DIF for the one composed of all edge networks.

This clearly shows the benefits due to RINA's clean design and therefore recommends it as a future Internet architecture. Nevertheless, its intrinsic dependence on computing inter-DIF bindings also means it raises the same type of concerns as the rest of the location/identity solutions. Notably, as one ascends in the layered architecture, scope tends to increase; so RINA will most probably have to design mapping systems for large (almost global) scope namespace within the higher layer DIFs. Alternatively, it may try to limit scope by introducing more nodes within higher DIFs but with potential downsides to routing scalability or node mobility. That is, if not well designed, routing systems may end up exposed to granular node information or incapable of supporting nodes changing their topological position without incurring too much routing churn. Unfortunately these are questions still to be answered before RINA can be considered for wide adoption, but on the upside, we believe that research on location/identity separation could at least clarify some of them.

2.3.2 Information-Centric Networking

The increasing demand for scalable and efficient content distribution has led to the development of several Internet architectures [13, 76, 85] centered on the information to be delivered, such as web pages or videos, called named data objects (NDO), instead of the communication channel. This approach, known as Information Centric Networking (ICN), can be contrasted with current networks which focus on naming the hosts in a communication due to their reliance on a client-server content delivery model.

ICN architectures leverage in-network storage for caching, multiparty communication through replication and interaction models that decouple senders and receivers [12]. Ultimately, the goal is to provide a general platform for communication services that is today provided only by peer-to-peer overlays or systems like content delivery networks. NDOs are self-contained and topology independent objects with verifiable authenticity ensured through a close biding between object name and data content. Naming schemes are either hierarchical, to ensure a better routing scalability, or flat. Therefore, to establish object authenticity a Public Key Infrastructure (PKI) must be employed or, respectively, NDOs must be self-certifying.

Communication follows a model whereby content providers publish or register their content with the network while receivers ask for NDOs using their names. There are two approaches to handling routing, dependent on the nature of the properties of the namespace. First, content names could be solved to topological locations of known sources using name routing and subsequently content be retrieved using topological based routing. This is similar to a location/identity separation, where the identifier is a the content name. Second, name based routing could be used at network level, by means of routing protocols, to find content sources and content could then be delivered to a client by backtracking the discovered path. For both approaches, any node on path from source to destination could employ application-transparent caching and therefore satisfy future requests for cached NDOs.

Apart from the obvious improvements to the scalability and cost-efficiency of content distribution, ICN could potentially improve the current network security model, support end-host mobility and multihoming, and improved reliability if applications are disruption tolerant. Nevertheless, ICN must also address a series of serious challenges prior to reaching a wider adoption. Foremost, it remains to be shown that routing in systems with vastly more NDOs than hosts, and thereby than the number of routes in today's DFZ, can be performed in a scalable fashion. Concerns dealing with privacy, as requests are visible to the whole ICN network, and legal issues surrounding ubiquitous caching must also be dealt with. Finally, since ICN affects the existing network layer economical relations between producers and consumers, a widespread deployment requires a better understanding of the incentives for all parties involved [12].

Chapter 3

Locator/ID Separation Protocol

The Locator/ID Separation Protocol (LISP) [9, 55, 124], is one of the solutions [88] offered as a response to the IAB workshop problem statement concerning routing scalability. However, in contrast to its competition, it gives high priority to interoperability with legacy Internet, ease of deployment and transparency for end-hosts.

To achieve these goals, two important design decision had to me made. First, LISP uses autonomous system administrative boundaries, like border routers, to separate end-host identifier (EID) and routing locator (RLOC) namespaces but does not mandate the change of address syntax in either of them. As a result, end-hosts and core infrastructure require no protocol stack updates, making for a lean transition. Note however that this does not preclude the use of LISP on end-hosts, it is only done in the interest of simplifying global adoption. Second, LISP extends *map-and-encap*, instead of *address rewriting* both because of its focus on not modifying end-hosts but also because it allows a more flexible management of identifier and locator namespaces. For instance, encapsulation can be exploited to transit IPv6 EIDs over an IPv4 RLOC space or the other way around.

For map-and-encap operation, LISP introduces two new network functions to support packet encapsulation/decapsulation, i.e., tunneling, and a distributed database, commonly referred to as a *mapping system*, to support the lookup of bindings between the two namespaces. Prior to forwarding an end-host generated packet, a LISP router maps the destination address, the EID, to a corresponding destination RLOC by querying a LISP specific mapping system. Once a mapping is obtained, the border router tunnels the packet from source edge to corresponding destination edge network by encapsulating the packet with a LISP-UDP-IP header. The outer IP addresses are RLOCs pertaining to the corresponding routers (see Figure 3.1).

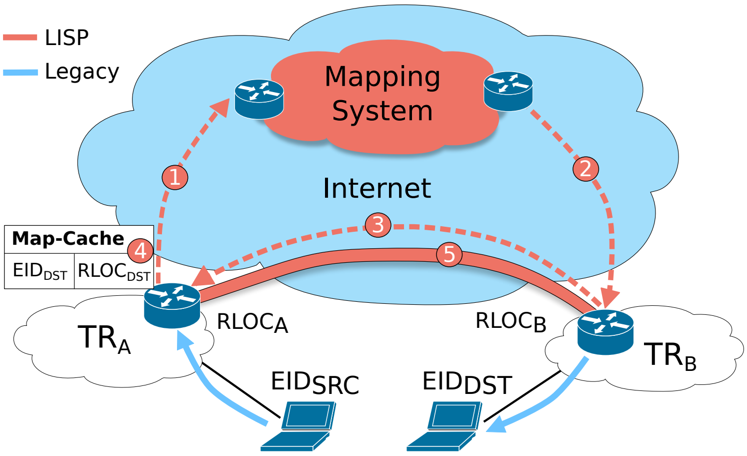


Fig. 3.1 Example packet exchange between EID_{SRC} and EID_{DST} with LISP. Following intra-domain routing, packets reach TR_A which, not having a prior map-cache entry, obtains a mapping binding EID_{DST} to $RLOC_B$ from the mapping-system (steps 1-3). TR_A stores the mapping in the map-cache (step 4) and then encapsulates and forwards the packet to $RLOC_B$ over the Internet's core (step 5). TR_B decapsulates the packets and forwards them to their intended destination.

At the receiving router, the packet is decapsulated and forwarded to its intended local-site, destination end-host.

In LISP parlance, the source router, that performs the encapsulation, is called an Ingress Tunnel Router (ITR) whereas the one performing the decapsulation is named the Egress Tunnel Router (ETR). One that performs both unidirectional tunnel terminating functions is referred to as a Tunnel Router (xTR). Additionally, LISP makes use of Re-encapsulating Tunnel Routers (RTRs), that perform re-encapsulation, i.e., decapsulation followed by encapsulation, to enable packet re-routing based on EID.

Generally, we shall refer to the aspects concerning forwarding and encapsulation as *data plane* operations in contrast to those relating to the interaction with the mapping system or any other LISP specific messaging, which we shall refer to as *control plane* operations.

An immediate benefit inherited from map-and-encap is that LISP natively supports multihoming since mappings may carry multiple RLOCs associated to an EID. Moreover, tunnel routers can control their ingress traffic, that is, how incoming packets are distributed over their interfaces. Specifically, when an ITR obtains the mapping from an ETR it also receives a priority ordering of how the ETR expects its locators to be used, and for the cases when locators are equally preferable, a list of weights, describing how traffic is to be load balanced over them. These *priorities* and *weights* are defined as attributes of locators in mapping. It is also interesting to observe that ETRs may employ more complex traffic engineering policies by providing custom, per ITR, mappings, since they possess information about their peer ITRs from `Map-Request` messages.

Since the packet throughput of an ITR is highly dependent on the time needed to obtain a mapping, but also to avoid over-loading the mapping-system, ITRs are provisioned with mappings caches (map-caches) that store recently used EID-prefix-to-RLOC bindings. Stale entries are avoided with the help of timeouts, called time to live (TTL), that mappings carry as attributes and whose expiry triggers the removal of unused mappings from the cache. Consistency is ensured by a proactive LISP mechanisms, Solicit Map Request (SMR), through which the xTR, owner of an updated mapping, informs its peer that it should request updated mapping information. The information exchange is done using `Map-Request` messages and the SMR fields in the LISP header.

Intuitively, the map-cache is most efficient in situations when destination EIDs present high temporal and/or spatial locality and its size depends on the diversity of destinations visited by a site's clients. As a result, map-cache performance depends entirely on provisioned size, traffic characteristics and the eviction policy set in place. We elaborate these points and discuss suitable cache eviction policies and mechanisms to protect the cache from polluting attacks in the next part of the thesis.

3.1 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 [61]. Logically, they instantiate two functions, one used for querying about and another for registering EID addressed. The former is known as the *Map-Resolver* function and the latter as *Map-Server* function. Independent of the algorithm implemented by the mapping system for identifier resolution, network devices implementing the Map-Server function receive `Map-Register` messages from ETRs on their client-facing interface, while *Map-Resolvers* 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 request to the Map-Server and, subsequently to the ETR advertising reachability of a prefix encompassing the requested EID. Finally, the ETR answers the requester's query by sending it a `Map-Reply`. 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. Despite being treated separately, noting precludes the implementation of both functions, Map-Resolver and Map-Server, by a single device.

A Map-Server may be configured to perform *proxy map-replying*, when, instead of forwarding a `Map-Request` to the authoritative ETR, it generates an non-authoritative reply and forwards it to the requesting ITR. This simplifies the ETR's job in LISP's control plane and allows it to reach all LISP destinations even when unable to natively forward traffic, like for instance, when behind a Network Address Translation (NAT) box. Further, this leaves open the possibility that the Map-Servers performs additional traffic-engineering optimization and the like on behalf of ETRs.

A possible optimization could be to configure a Map-Resolver to work as a caching resolver. In this case, the resolver must save state for all ongoing EID resolutions and initiate Map-Requests in the name of its clients. On the up side, all mapping results are cached for future reuse, before being forwarded to the ITRs, which can aid in reducing mapping latency for clients. But as a downside, in practice this could interferes with inbound traffic engineering policies. 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.

Considerable effort has been invested into finding the best suited mapping system both in terms of lookup latency and scalability. Among the currently proposed system, approaches vary from using a BGP overlay (ALT [63]), a DNS like architecture (DDT [64, 78]), distributed hash tables (DHT [96]), hybrid push-pull mechanisms (CONS [24]) to using a simple push architecture (NERD [87]). A detailed comparison can be found in [67]. Currently, the generally held opinion is that pull architectures, i.e., those where mappings are retrieved on demand, as opposed to being pushed to ITRs, are more desirable due to the possibility that the EID address space may outgrow feasible storage space within ITRs. In this sense, DDT is seen as striking the best balance between, scalability with EID address space growth, resolution latency and state ITRs must store. As a result, it is the "de facto" LISP mapping system and the one currently deployed in the LISP-Beta network [8].

It should be noted that although the choice between a pull and a push design fundamentally affects the size of mapping caches stored in tunnel routers, within the subclass of pull architecture, the resolution algorithm employed by the mapping system does not influence cache size. In fact, the independence with respect to the size of the identifier namespace is entirely dependent on customer traffic and caching algorithms employed by ITRs. If a network's egress traffic has a uniform popularity distribution of destinations, i.e., all destinations are equally probable, map-cache size will most probably need to be as large as the EID namespace, to achieve good performance. As a consequence, we shall generally not be interested over the course of this thesis in aspects concerning EID resolution algorithms but instead, will only focus on a push/pull distinction when discussing mapping systems' designs.

3.2 Locator Reachability

One of the most important problems affecting all locator/identifier separation adopting architectures is the locator reachability problem [103]. It consists in the impossibility of knowing apriori if a path to a destination is 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 [55]. 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 message's LISP 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, thus opening the possibility of optimizing locator selection.

3.3 Incremental Deployment

As previously discussed, the success of any new architecture depends on its ability to provide added value without disrupting normal operations. Two aspects had to be considered when designing LISP: *i*) inter-domain delivery of encapsulated packets and *ii*) interworking with sites not upgraded to LISP (we further refer to them as legacy Internet).

The first aspect, despite apparent simplicity, is rendered rather complex because of the existence of so called *middle-boxes* (e.g., firewall, NAT). These devices build and use dynamic state based on IP or transport header fields, always presumed to be present in packets. Thereby, packets not carrying a UDP or TCP header are at times quietly discarded. Moreover, operators often employ load balancing techniques, like Equal Cost Multipath (ECMP), to distribute flows over core router interfaces by hashing source and destination IP addresses but also transport level port numbers [70, 132]. Because these limitations preclude the use of a simple IP-in-IP encapsulation, LISP uses an additional UDP and LISP extended header at the cost of decreasing the maximum transmission unit (MTU). The former solves the above limitations while the latter is meant to convey LISP specific information. The default LISP data-plane UDP port is 4341. Similarly, control plane messaging is carried over UDP using port 4342.

To ensure seamless interworking with legacy Internet sites, LISP introduces two proxy devices: Proxy ITR (PITR) and Proxy ETRs (PETR). The former, allows non-LISP sites to send packets to LISP enabled sites without any infrastructure support within the legacy domains. Functionally, a PITR is in charge of encapsulating and forwarding legacy traffic to its intended LISP domain. To ensure inter-domain routability of EIDs, a PITR advertises EID-prefixes, or if it serves more than one domain, aggregate EID-prefixes, on behalf of the LISP sites.

PETRs, perform the conceptually complementary function. Namely, a PETR acts as ETR for all destinations in legacy Internet. Given that such destinations do not have a mapping (i.e., they have a negative mapping), ITRs not BGP enabled or unable to forward traffic natively, send their LISP encapsulated packets to PETRs that decapsulate and forward the packets natively in the legacy Internet.

These types of devices have been in the last years leveraged in practice for interworking the legacy Internet and LISP-Beta, the experimental LISP testbed [8]. The members of this joint effort are academics, research laboratories, small enterprises that offer LISP services but also large companies (e.g., Microsoft, Verisign and Face-book) and operators (e.g., Level3). LISP-Beta users are distributed in 27 countries, but the largest concentrations are found in Europe and North America.

Initial measurements [34, 120] have shown that reliance on BGP and little adoption do affect interworking performance in terms of latency stretch. So more effort is needed to improve the performance as the userbase grows. But, on the up side, the system's performance seems to be unaffected by the steady year-over-year growth of the served EID space. For a detailed discussion on how LISP network devices could be deployed and their potential impact on the DFZ routing table see [79].

Part II

LISP Mapping System Scalability

Chapter 4

Cache Model

This chapter introduces an analytical cache model that, to the best of our knowledge, constitutes the first theoretical framework for map-cache performance analysis and provisioning. Our approach relies on the working-set as an estimator of traffic locality of reference and on the associated theory, to understand the influence of locality on cache performance. Therefore, as first step, we identify the conditions that network traffic must comply with for the theory to be applicable and thereafter develop a model that links miss rate and cache size. The key assumption that enables the analysis is that the stocastic process underlying the generation of traffic is stationary. So, to facilitate the use of the model, we also propose a methodology to test for this property in network traces. Finally, we validate the model and perform an extensive, over time analysis of cache performance using packet traces collected at the egress points of a campus and an academic network.

These results are the first step in our analysis of map-cache performance. In the next chapter we extend the model to evaluate the impact of cache polluting traffic while in Chapter 6 we further investigate the sources of temporal locality to understand the asymptotic scaling of the map-cache's performance.

4.1 Working-set model for traffic locality

For operating systems, a general resource allocation treatment was possible after it was observed that programs often obey the so called *principle of locality.* The property arises from the empirical observation that programs favor only a subset of their information at a given time, so they may be efficiently run only with a fraction of their total data and instruction code. It was shown that if the subset, called the

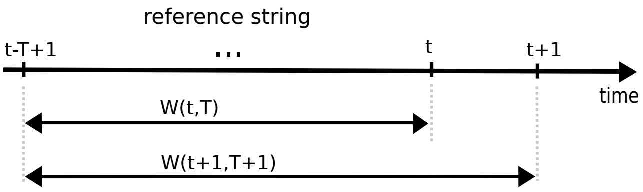


Fig. 4.1 The Working-Set Model. At time t, $W(t,T)$ is the set of units recently referenced in the time window T, i.e., during the interval $[t-T+1,t]$.

program's *locality*, can be computed, the policy of keeping the locality in memory is an optimal or near optimal memory management policy [130].

Based on previous results [75, 83, 84], we argue that prefix-level network traffic roughly obeys the same principle because of, among others, skewed destination popularity distributions and flow burstiness in time, as they gives rise to temporal locality, but also due to aggregation, i.e., multiplexing of a large number of user flows, which leads to a form of geographical locality. We therefore evaluate the feasibility of in-router *prefix/mappings caching* by analyzing the locality of network traffic. Since we want to avoid any assumptions regarding the structure of the process generating the network traffic, we opt in our evaluation for the working-set model of locality. For a list of other locality models see [130].

Next we provide a summary of the working-set terminology. For brevity, we use the term *unit of reference*, or simply *unit*, as a substitute for the referenced object (e.g., prefixes); and *reference set* to represent the set of all referenced units. Then, considering a *reference set* N, we define a *reference string* as the sequence $\rho = r_1 r_2 \dots r_i \dots$ where each unit $r_i \in N$. If t is a measure of time in *units*, then we can state:

Definition 1. *Given a reference string, the working-set $W(t,T)$ is the set of distinct* units *that have been referenced among the T most recent references, or in the interval $[t-T+1,t]$.*

A graphical depiction can be found in Figure 4.1. In accordance with [46] we refer to T as the *window size* and denote the number of distinct pages in $W(t,T)$, the *working-set size*, as $w(t,T)$. The *average working-set size*, *s(T)*, measures the growth of the working-set with respect to the size of the window T, extending in the past, but independent of absolute time t. It is defined as:

$$s\left(T\right) = \lim_{k\to\infty} \frac{1}{k} \sum_{t=1}^{k} w(t,T) \tag{4.1}$$

It can be proved that the the *miss rate, m(T)*, which measures the number of units of reference per unit time returning to the working-set, is the derivative of the previous function and that the sign inverted slope of the miss rate function, the second slope of $s(T)$, represents the *average interreference distance* density function, $f(T)$. For a broader scope discussion of these properties and complete proofs, the interested reader is referred to [46].

It is important to note that $s(T)$ and $m(T)$, if computable, provide estimates on the minimum average size of a cache able to hold the working-set, i.e., the prefixes in the active locality, and its corresponding miss rate with respect to the number of references processed. Our goal is to determine if $m(s)$ exists for real network traffic and if it can be modeled as simple function without a considerable loss of precision.

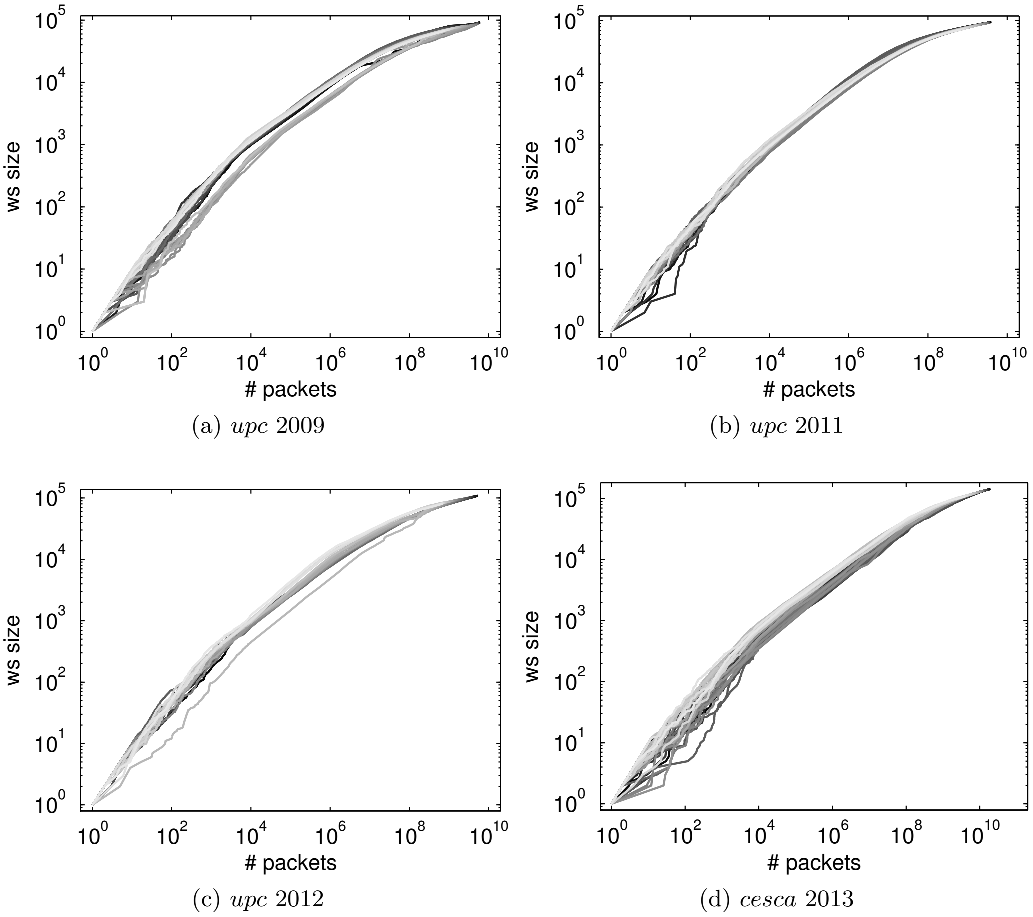
4.2 Network Traffic Locality

As explained by Denning in [46], a working-set analysis of reference strings may be performed only if based on three constraints that provide for a more rigorous definition for *locality of reference*:

1. Reference strings are unending
2. The stochastic mechanism underlying the generation of a reference string is stationary, i.e. independent of absolute time origin.
3. For $t > 0$, r_t and r_{t+x} become uncorrelated as $x \to \infty$

The first, though obviously not fulfillable, introduces an insignificant error because the reference strings generated by practical programs or traces are long from a statistical standpoint. The third requires that references become uncorrelated as the distance goes to infinity. This can usually be asserted as being true in practice. The most restrictive of the three is the second, which limits the analysis to a locality where all three constraints, including stationarity are satisfied.

In practice however, network traffic reference strings may consists of sequences of localities when either each is characterized by a distinct stationary processes or, alternatively, when some present nonstationarities. In such scenarios the results, like the average working-set size, are only valid within a locality, and not to be extrapolated for the whole trace. To test for this possibility and to identify the reference string segments having different generating processes in network traffic traces, we devise the simple experimental methodology that follows.



(a) *upc* 2009

(b) *upc* 2011

(c) *upc* 2012

(d) *cesca* 2013

Fig. 4.2 Empirical working-set curves with starting times spaced by 30 min and evenly distributed over a day. Closeness of color nuance reflects closeness of start time. Notice their tendency to cluster.

To enable our analysis, since computing the $w(t,T)$ for all acceptable integer combination of t and T is intractable, we define the *working-set curve* to be $w(t,T)$ as a function of T, when the past time reference of the working-set is held fixed, i.e., $t - T = cst$. For instance, considering Figure 4.1, $w(t,T)$ and $w(t+1,T+1)$ are consecutive points on a working-set curve with start time $t - T$. Then, for a given trace, we compute multiple empirical destination prefix working-set curves with start times spanning one day and spaced by a fixed interval. Intuitively, one would expect that the clustering patterns of the curves should distinguish between the multitude of generating process. That is, curves with close start times should have a similar growth shape (cluster), because they follow a similar sequence of localities, whereas those separated by larger time lags should behave differently. More formally stated:

Proposition 1. *The clustering of the working-set curves, under which $\forall T, w(t,T)$ is normally distributed, is equivalent to the stationarity of the process generating the reference string.*

Proof. If $\forall T$ the working-set size $w(t,T)$ is normally distributed, and therefore independent of t, it follows that the process generating $w(t,T)$ is stationary. This in turn implies the stationarity of the process generating the reference string and, as a result, necessity is proven. It remains to be proved that if the working-set curves are generated by the same stationary process then they will tend to cluster. In [46] it is shown that for a certain window size T the distribution of $w(t,T)$ converges to a normal distribution if the locality conditions hold. This proves sufficiency. □

In addition, we empirically confirm the result by using it to determine the stationary processes embedded in four real network traffic traces. Details regarding the traffic captures can be found in Section 4.4.1.

For each network trace we computed working-set curves spaced by half an hour. Figure 4.2 presents the working-set curves of the four datasets in log-log plots. It can be noticed that all traces exhibit a strong clustering and sublinear growth, due to temporal locality. Although, the number of samples does not allow for an accurate enough testing, we could also confirm that $\forall T, w(t,T)$ is close to normally distributed by manual inspection and graphically. Therefore, in light of the previous proposition, we find that each trace can be considered as generated by a single stationary process.

To validate the result, we also independently tested the stationarity of the process generating the reference string by applying the augmented Dickey-Fuller unit root test [18] to the interreference distance time series. Due to the large size of the dataset we first aggregated the time series by computing for each window of $10k$ data points the mean and subsequently applied the test to the resulting mean series. For all traces, the null hypothesis that the series has a unit root, with the alternative that the series is stationary, was rejected ($p < 0.01$).

Undoubtedly, the most surprising result of the analysis is the stationarity of the processes generating the four traces. It follows that the average working-set and the other metrics derived from it characterize a trace in its entirety. This might seem to be somewhat counter-intuitive if one considers the nonstationarities of network traffic when analyzed in the time domain. However, we believe stationarity to be the result of flow multiplexing, whereby the effect of short term correlations is canceled out by the influence of destination popularity, as shown in the case of Web traffic in [81].

Similarly, Kim et al. observed in [83] that the working-set size for prefixes tends to be highly stable with time for traffic pertaining to a large ISP. We would like to stress that we do not assume all traffic in the Internet is generated by stationary processes, i.e., possesses an approximate time translation invariance of the working-set curves like the one observed in Figure 4.2. And in fact, we require that the model we develop further be applied only to traces that have this property.

4.3 Analytical Cache Model

This section presents the theoretical background and analytical methodology used to model map-cache performance. After briefly introducing the working-set theory we formalize the cache modeling problem and show that the working-set is suitable for the analysis of our network traces. The results enable us to derive a cache model and by extension one that accounts for cache polluting attacks.

Because the exact form of the average working-set size, as obtained using (4.1), is rather cumbersome to work with, a shorter but approximate representation would be desirable. With hindsight, one can recognize that the empirical working-set curves from Figure 4.2 are piecewise linear when depicted in log-log scale. This observation enables us to approximate the average working-set size, $s(u)$, for each trace with respect to the number of packets u, by means of a piecewise linear fit of the log-log scale plot. We therefore obtain estimates of both the slope, α, and the y-intercept, β, for all segments. In our results, we limited the number of segments to just four, however if better fits are desirable, more segments may be used. Through conversion to linear scale the average working-set equation becomes piecewise power law of the type:

$$s(u) = e^{\beta(u)} u^{\alpha(u)} \tag{4.2}$$

where, u represents the number of referenced destination prefixes, or the window size, $s(u)$ is the fitted working-set size function and $\alpha(u)$, $0 < \alpha(u) \leq 1$, and $\beta(u) \geq 0$ are piecewise constant, decreasing and respectively increasing functions obtained through fitting. Defined as such, the pair $(\alpha(u), \beta(u))$ provides a compressed characterization of the temporal locality present within a trace with respect to time, i.e., number of packets.

We can estimate the miss rate for a trace by computing the derivative of $s(u)$ like:

$$m(u) = e^{\beta(u)}\,\alpha(u)\,u^{\alpha(u)-1} \tag{4.3}$$

Taking the inverse of (4.2) and replacing it in (4.3) we obtain an analytical relation that links cache size and the estimated miss rate:

$$m(s) = e^{\beta^*(s)/\alpha^*(s)}\,\alpha^*(s)\;s^{1-1/\alpha^*(s)} \tag{4.4}$$

where, s represents the cache size in number of entries and $\alpha^*(s)$ and $\beta^*(s)$ are piecewise constant functions with knees dependent on s. This equation accurately predicts cache performance over longer spans of time, if $s(u)$ is relatively stable in the considered time frame (more on this in Section 4.5).

Regarding the type of cache modeled, it is useful to note that the working-set, $W(t,u)$, generally models a cache that always contains the $w(t,u)$ most recently referenced units. Then, given that $\forall u, w(t,u) \sim N(\sigma,\mu^2)$, $W(t,u)$ actually models a cache of size normally distributed and dependent on u. In particular, when σ is small, or goes to zero, it behaves like the size were fixed. Finally, because the implicit eviction policy requires that entries not referenced in a window of length u are discarded, for low σ, like in our traces, the working-set simulates a cache of fixed size with a LRU eviction policy.

To summarize, because we find that $\forall u$, the working-set size is normally distributed and that traffic may be seen as generated by a stationary process, (4.4) actually models a cache with LRU eviction and size dependent on u. We validate the model and the efficiency of LRU in Section 4.4.3.

4.4 Model Validation

In this section, we present the evaluation methodology and the results that validate our models. We start by describing our datasets and then pass on to present the ITR emulator employed in the empirical evaluation of the cache. Last, we compare the empirical results to those predicted by the analytical models.

4.4.1 Datasets

We use four one-day packet traces that only consist of egress traffic for our experiments. Three were captured at the 2Gbps link that connects our University's campus network to the Catalan Research Network (CESCA) and span a period of 3.5 years, from 2009 to 2012. The fourth was captured at the 10Gbps link connecting CESCA to the Spanish academic network (RedIris) in 2013. UPC campus has about 36k users consisting generally of students, academic staff and auxiliary personnel while CESCA provides transit services for 89 institutions that include the public Catalan schools, hospitals and universities.

Table 4.1 summarizes some of the important properties of our datasets. First of all, it can be seen that *cesca* 2013, being an aggregate trace, is about 3.6 times larger than the most recent UPC trace in terms of number of packets and packet rate. However, it only contains 1.3 times more prefixes. This shows that although the number of users and packets exchanged is considerably higher, the diversity of the destinations is only slightly incremented. Out of the UPC traces, *upc* 2009 exhibits a surprisingly high number of packets but this is explained by the very large packet rates seen during the active hours of the day. In fact, the average packet rate during the peak hours was 4.7 times higher that for the rest of the day. Again, this difference did not reflect in the number of unique prefixes observed in a one second window as, on average, we observed just 1.3 times more prefixes in the peak hours than during the remainder of the day. These two observations suggest that higher packet rates, either resulting from larger user sets or from higher throughput flows, do not increase destination diversity (as illustrated in Figure 4.2) but instead reinforce temporal locality. In addition, these properties also explain why the working-set curves for *upc* 2009 present a time-of-day behavior (see Figure 4.2a).

A close inspection of *upc* 2012 uncovered a 23 minute time window when approximately 60M packet egressed towards a Chinese destination and 44M packet towards a Saudi Arabian one. Just these two transfers averaged a higher packet rate than the overall average of the trace. This had a perceptible effect on the working-set, as depicted in Figure 4.2d.

4.4.2 Map-Cache Emulator

To evaluate the two models and the effectiveness of the working-set as a tool for cache performance prediction, we implemented a packet trace based emulator that mimics basic ITR functionality.

Table 4.1 Datasets Statistics

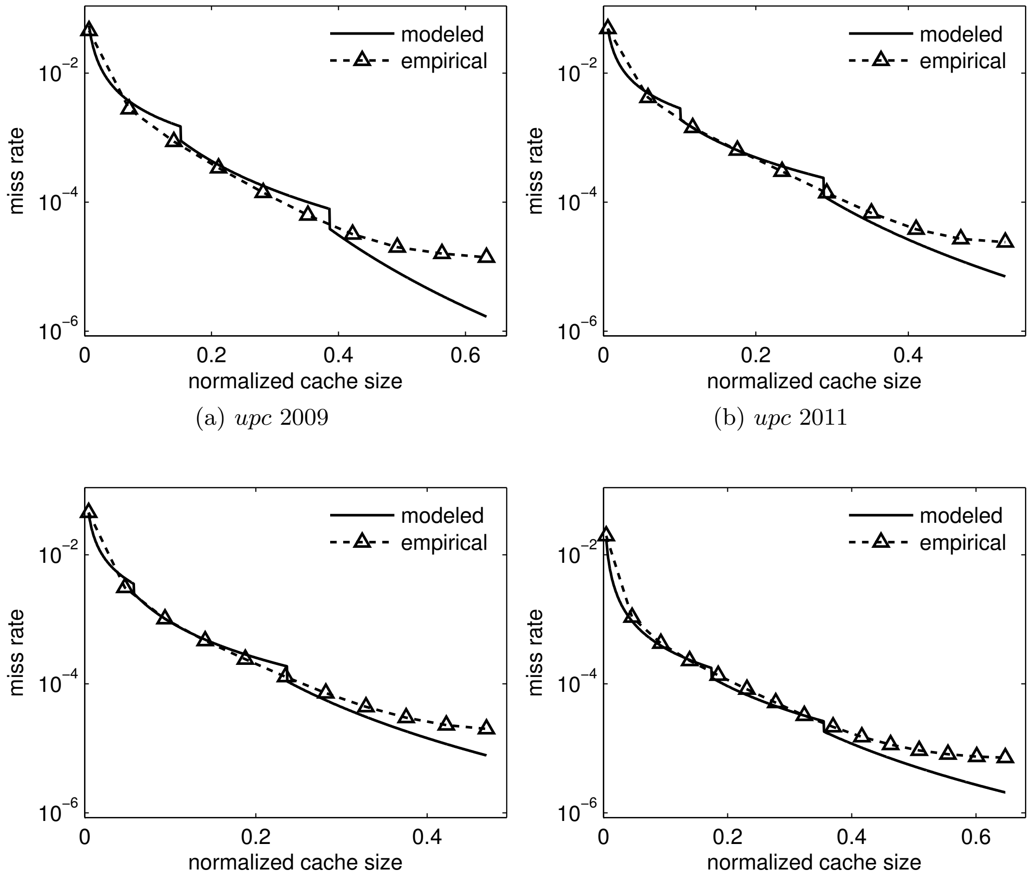
	upc 2009	**upc 2011**	**upc 2012**	**cesca 2013**
Date	2009-05-26	2011-10-19	2012-11-21	2013-01-24
Packets	6.5B	4.05B	5.57B	20B
Av. pkt/s	75312	46936	64484	232063
Ψ	92801	94964	109451	143775
Av. pref/s	2323	1952	2123	2560

Table 4.2 Routing Tables Statistics

	upc 2009	**upc 2011**	**upc 2012**	**cesca 2013**
$\mathtt{BGP}_{RT}$	288167	400655	450796	455647
$\mathtt{BPG}_{\phi}$	142154	170638	213070	216272
$\Psi/\mathtt{BGP}_{\phi}$	0.65	0.55	0.51	0.66

Both for computing the working-sets in Section 4.2 and for the cache performance evaluation, destination IP addresses had to be mapped to their corresponding prefixes. We considered EID-prefixes to be of BGP-prefix granularity. For each traffic trace, we linked IP addresses to prefixes using BGP routing tables ($\mathtt{BGP}_{RT}$) downloaded from the RouteViews archive [134] that matched the trace's capture date. In particular, we used collector *route-views4* situated at University of Oregon. The only preprocessing we performed was to filter out more specific prefixes. Generally, they are used for traffic engineering purposes but LISP provides mechanisms for a more efficient management of these operational needs that do not require EID-prefix de-aggregation. We refer to the resulting list as $\mathtt{BGP}_{\phi}$. Table 4.2 shows the size of the routing tables used for each trace and provides the proportion of prefixes seen within each trace out of the total registered in the filtered routing table, $\Psi/\mathtt{BGP}_{\phi}$. It may be seen that, as the ratio is always higher that 0.5, more than half of the possible destination prefixes are visited in one day for all traces.

For each packet processed, the emulator maps the destination IP address to a prefix in $\mathtt{BGP}_{\phi}$. If this prefix is already stored in the ITR's cache, its cache entry is updated and the emulator continues with the next packet. Should the prefix not yet be stored in the cache, two possibilities arise. First, if the cache is not full, the destination prefix is stored in and the processing proceeds to the next packet. Second, if the cache is full, an entry is evicted, the new prefix is stored



(a) *upc* 2009

(b) *upc* 2011

(c) *upc* 2012

(d) *cesca* 2013

Fig. 4.3 Comparison between empirical and modeled miss rates, as estimated by (4.4), for normal cache operation.

in and then the emulator moves to the next packet. The entry to be evicted is chosen according to the LRU eviction policy. We use LRU because, as mentioned in Section 4.3, its performance should be close to optimal due to the stationarity of the trace generating process. Accordingly, the performance of the cache should be appropriately described by (4.4).

4.4.3 Comparison of Analytical and Empirical Results

To validate the models we use the emulator to estimate empirical cache miss rate for several cache sizes. Figure 4.3 presents a comparison of the empirical and predicted results for normal traffic, where cache size is normalized with corresponding $\mathtt{BGP}_\phi$

routing table size (see Table 4.2). It may be seen that, typically, the absolute error is negligible despite the discontinuities of the model, which are due to the piecewise fitting. Further, the equation appropriately predicts that performance stays acceptable, even for small cache sizes, fact also observed by [75, 78, 84]. The result is even more remarkable as we never remove stale entries in our emulator, i.e., we consider TTL to be infinite.

The figures present the cache miss rate only up to a fraction of $\texttt{BGP}_\phi$, the one associated to $|\Psi|$, because the growth of $s(u)$ cannot be extrapolated after this point. Since the working-set grows slower for larger u, as $\alpha(u)$ is a strictly decreasing function, potentially much longer traces would be needed to enable inference about larger cache sizes. In fact, given that only part of the whole prefix space may be visited by the clients of a stub network, even for longer traces the analysis may be limited to a cache size lower than $|\texttt{BGP}_\phi|$.

There are two limitations to the precision of our analysis for large u values. First, as cache size increases and approaches $|\Psi|$, the accuracy of the prediction diminishes. This is explained by the 24-hour length of the traces, whereby there are few working-set curves that span close to a whole day and thus grow to reach the maximum number of destination prefixes. Recall that the start times for the working-set curves span the whole trace and are spaced by 30 minutes, so the last curves consist of few packets. As a consequence, $s(u)$ is estimated using a reduced number of points, i.e., with a lower precision, at the higher end. To counter this effect, we compute average working-sets of slightly diminished length. The second limitation is the bias of our emulation results for large cache sizes. Caches whose sizes are close to $|\Psi|$ fill only once the traces are processed almost in their entirety. Due to this cold-start effect, cache are exposed to a low number of hits up to the end of the traces. As these hits do not manage to outweigh the misses generated during the cache fill up, the miss rate of the emulator for large cache sizes is slightly overestimated. Nevertheless, despite these limitations, it may be seen that the results still yield a good fit for large u.

4.5 Cache Model Results and Predictions

In this section we discuss the results and predictions of our models regarding map-cache performance for both normal and malicious traffic. We also discuss possible avenues to diminishing the effect of cache attacks.

The results we obtained are relevant only when reasoned about jointly with

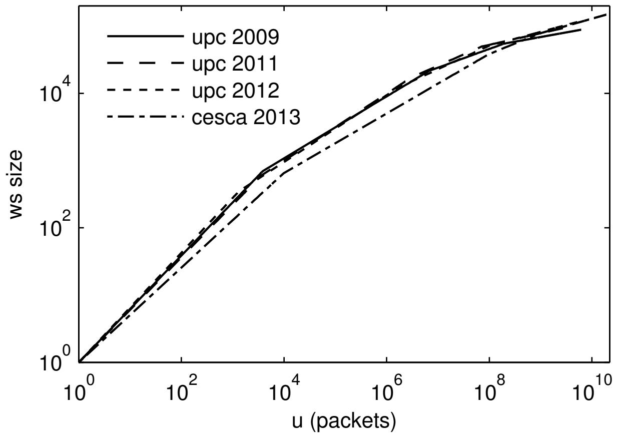


Fig. 4.4 Evolution of $s(u)$ over the years. The shape appears to be influenced by the number of users and the destination popularity distribution.

the traffic traces used in the analysis. However, the diversity of our data sets and previous results from Kim [83], Iannone [75] and Kim [84] suggest that the properties uncovered are not the expression of isolated user and network behavior.

Under the condition of a stationary generating process or, equivalently, the approximate time translation invariance of the working-set curves, our methodology enables the estimation of the time invariant piecewise functions $\alpha(u)$ and $\beta(u)$ that characterize the locality of a network traffic trace from the average working-set size $s(u)$. This further facilitates the following two findings. First, due to the low variance of $s(u)$ and experimentally proven good performance, we can now recommend the use of the LRU eviction policy for LISP caches. Second, in such situation, (4.4) may be used to dimension the cache sizes in operational environments, according to the desired miss rate. The prediction of its mathematical expression, considering that $\alpha^*(c) \to 0$ when c increases, is that miss rate decreases at an accelerated pace with cache size and finally settles to a power-law decrease. This may also be observed in Figure 4.3 where at each discontinuity point the function switches to a faster decreasing curve. Of course, the speed of the decrease depends on the degree of locality present in the trace. Overall the equation indicates that cache sizes need not be very large for obtaining good performance. For instance, having a cache of size 10% of BGP_ϕ, about $14k - 21k$ entries for UPC traces and $21k$ for the CESCA one, would result in a miss rate of approximately $0.09\% - 0.2\%$ and respectively 0.03%.

In this context, an important point would be to determine the extent of time over which the results and predictions hold. Figure 4.4 provides a coarse answer for the particular case of the traffic used in our analysis. First, considering the UPC traces, it may be observed that over a span of tree and a half years, the average working-set is rather stable when size is less than about $50k$ prefixes. In fact, the $s(u)$ curves of *upc* 2011 and *upc* 2012 are very similar, independent of u value, while the one for *upc* 2009 exhibits a lower slope for $u > 100M$ packets. This might appear to be in agreement with the relative, year-over-year, increase of BGP_ϕ. But the relative differences between the values of $s(u)$ and the increase of BGP_ϕ are not directly related since the growth from 2009 to 2011 was smaller than the one from 2011 to 2012. This is to be expected because many of the new prefixes, resulting from the sustained growth of the Internet's edge [32], may never become destinations for users of other edge networks. So, a direct relation between the increment of the routing table size and that of $s(u)$ should not necessarily be expected. Instead, given the good overlap for lower values of u, prefix popularity distribution should be more relevant to the shape of the working-set than the absolute number of destination. Second, the comparison between *cesca* 2013 and the UPC traces reveals that larger user sets, and implicitly higher traffic rates, result in a slightly slower growing $s(u)$. In fact, the only noticeable difference is at short time scales, where the larger trace has a smaller slope. This could be explained by a change in the destination popularity distribution, as *cesca*2013 aggregates more types of user profiles, but also by a shift in the short-term temporal correlation. However, considering the large number of users, their synchronization at short time scales seems rather unlikely.

Then, although apparently stable over relatively long time spans, the shape of $s(u)$ seems to be influenced by non trivial interactions between the number of clients the prefix popularity distribution and possibly other unexplored factors. We further explore these interactions in Chapter 6. Thus, our cautious inference here is that the average working-set should be stable over time, if the number of clients and the popularity distribution are relatively stable.

Despite not being indicated by our measurements, it may be finally proven that the variability with time of $s(u)$ is highly dependent on properties of the network being measured, themselves time dependent. Should this be the case, the methodology we develop is still valuable for the analysis of cache performance if not for long term provisioning of caches.

4.6 Related work

Feldmeier [59] and Jain [77] were among the first to evaluate the possibility of performing destination address caching by exploiting the locality of traffic in network environments. Feldmeier analyzed traffic on a gateway router and showed that caching of both flat and prefix addresses can significantly reduce routing table lookup times. Jain however performed his analysis in an Ethernet environment. He was forced to concede that despite the locality observed in the traffic, small cache performance was struggling due to traffic generated by protocols with deterministic behavior. Both works were fundamental to the field however their results were empirical and date back to the beginning of the 1990s, years when the Internet was still in its infancy.

Recently, Kim et al. [83] showed the feasibility of route caching after performing a measurement study within the operational confinement of an ISP's network. They show by means of an experimental evaluation that Least Recently Used (LRU) cache eviction policy performs close to optimal and that working-set size is generally stable with time. We also observe the stability of the working-set for our data sets but we further leverage it to build a LRU model instead of just empirically evaluating its performance.

Several works have previously looked at cache performance in location/identity split scenarios considering LISP as a reference implementation. Iannone et al. [75] performed an initial trace driven study of the LISP map-cache performance. Instead of limiting the cache size by using an eviction policy, their cache implementation evicted stale entries after a configurable timeout value. Further, Kim et al. [84] have both extended and confirmed the previous results with the help of a larger, ISP trace and by considering LISP security aspects in their evaluation. Ignoring control plane security concerns, which we did not consider, and despite differences regarding the cache eviction policy, the results of these last two works seem to be in agreement with ours. Zhang et al. [140] performed a trace based mappings cache performance analysis assuming a LRU eviction policy. They used two 24-hour traffic traces captured at two egressing links of the China Education and Research Network backbone network. They concluded that a small cache can offer good results. Finally, Jakab et al. [78] analyzed the performance of several LISP mapping systems and, without focusing on a cache analysis, also observed very low miss rates for a cache model similar to that used in [75].

Our work confirms previous LISP cache analysis results however, it also tries to provide a better understanding of the reasons behind the relatively good performance of map-caches. In this sense it introduces an analytical model that could be used to theoretically evaluate or dimension for operational needs the caching performance. Moreover, to the best of our knowledge, it is also the first work to perform an analysis and propose an analytical model for the map-cache performance when under scanning data-plane attacks.

4.7 Chapter Summary

In this chapter, we propose a methodology to evaluate map-cache performance. Our model is built by exploring the link between cache performance and parameters that approximate the intrinsic locality of packet level user traffic. To this end, we advance the use of the working-set model as a tool to capture said properties but also as a performance predictor. Accordingly, we define a framework wherein to perform the analysis and find that the clustering of the working-set curves is the only condition needed to ensure the accuracy of the model. We empirically validate our result by emulation, using traffic traces collected at the edges of a campus and an academic network. The piecewise power-law dependence between cache size and miss rate finally explains previous empirical observations according to which increasing cache size quickly diminishes miss rates.

Besides the possibility of using the model for cache dimensioning in operational environments, we believe the equation may also be used as part of more complex models that evaluate the scalability of loc/id architectures. To stress this point, we show the versatility of our methodology by characterizing map-cache performance for our datasets and by extending the model in the next chapter to account for cache pollution attacks.

Chapter 5

Cache Polluting Attacks

In situations when intra-domain users perform EID address space scanning attacks, the working-set curves are significantly altered and as a result, the map-cache hit rate significantly drops. In order to better understand the effect of such attacks, we extend the model developed in Section 4.3. It is worth noting that we focus on assessing the damage users can inflict through data-plane attacks and do not consider control-plane attacks like those described in [121].

5.1 Cache Model for Cache Pollution Attacks

We define a scanning attack as the situation when one or multiple users, acting jointly, send packets over a large period of time (e.g., hours), to destinations having a high probability of not being found in the cache. The goal would be to either generate cache misses, resulting in control plane overload or, if the cache is not large enough, to generate cache evictions, which would affect ongoing flows. For instance, having a list of EID prefixes, an attack would consist in sending packets with destinations enumerating all prefixes in the set in a random order, at a certain packet rate. Once all destinations are exhausted the enumeration would start over.

In what follows we formally define the parameters of the attack. Let, Ω be the EID-prefix set used in the attack and Ψ the network's visited EID-prefix set. We define the relative attack intensity ρ as the ratio between the attack packet rate and the legitimate traffic packet rate, additionally, let the attack overlap δ be the ratio between the number of prefixes common to Ω and Ψ and the cardinality of Ψ thus, $\delta = |\Omega \cap \Psi|/|\Psi|$.

If a network trace with average working-set $s(u)$ is augmented by a scanning attack of relative attack intensity ρ and overlap δ, the resulting average working-set becomes:

$$s_a(u+\rho u) = \begin{cases} s(u) + \rho u - \dfrac{\delta s(u_k)}{u_k} u, & u < u_k \\ s(u) + |\Omega| - \delta s(u) \end{cases} \tag{5.1}$$

where $u_k = |\Omega|/\rho$ and it represents the number of legitimate packets after which the attack exhausts all $|\Omega|$ destinations and the scan restarts. The aggregate working-set has three components. The first is due to legitimate traffic, $s(u)$, and the second, due to the attack packets, ρu. However, because the two may overlap, a third component subtracts the number of shared prefixes. For simplicity, we approximate the probability of having destinations repeat to be uniform. Thereby, the growth of the overlap is linear with u up to u_k, where it reaches a maximum of $\delta s(u_k)$ and afterwards linear with $s(u)$.

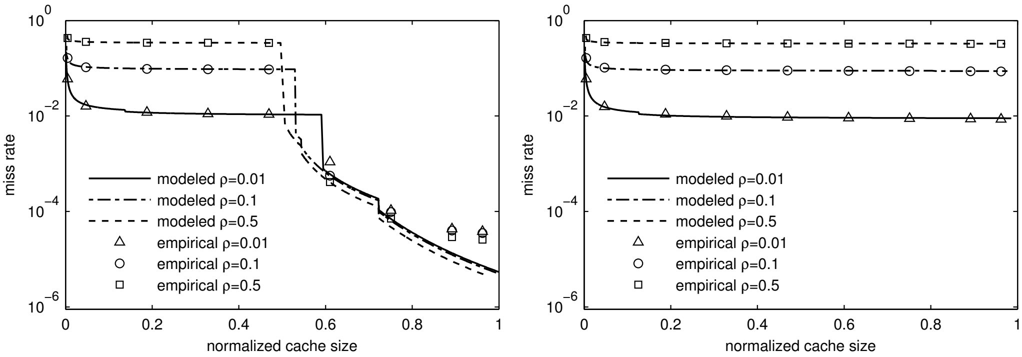
After a change of variable and denoting $\tau = 1/(1+\rho)$, or the ratio of legitimate traffic in the trace, $u_k = \dfrac{|\Omega|}{1-\tau}$ and the equation becomes:

$$s_a(u) = \begin{cases} s(\tau u) + \left(1 - \tau - \dfrac{\tau \delta s(u_k)}{u_k}\right) u, & u < u_k \\ (1-\delta) s(\tau u) + |\Omega| \end{cases} \tag{5.2}$$

Then, the miss as a function of the number of processed packets is:

$$m_a(u) = \begin{cases} \tau\, m(\tau u) + \left(1 - \tau - \dfrac{\tau \delta s(u_k)}{u_k}\right), & u < u_k \\ \tau(1-\delta)\, m(\tau u) \end{cases} \tag{5.3}$$

However, in this case the miss rate cannot be represented analytically as a function of the cache size since $s_a^{-1}(u)$ is not expressible in terms of standard mathematical functions. It can though be computed numerically as a function u, when $s(u)$ is known. Then, given that both $s_a(u)$ and $m_a(u)$ are known, they suffice to understand the cache's miss rate as a function of the cache size. The resulting model predicts overall cache misses, not only those due to legitimate traffic. Therefore, it provides an estimate of the control plane overload, not an estimate of the data plane miss rate for legitimate traffic.



(a) No overlap ($\delta = 0$), $\Omega = \texttt{BGP}_\phi - \Psi$ (b) Complete overalp ($\delta = 1$), $\Omega = \texttt{BGP}_\phi$

Fig. 5.1 Comparison between empirical and modeled miss rates, as estimated by (5.2) and (5.3), under scanning attacks with attack intensity $\rho \in \{0.01, 0.1, 0.5\}$ for *upc 2012*.

5.2 Model Validation

To validate the model, we reuse the emulator presented in Section 4.4.2. We simulate scanning attacks by uniform insertion of attack packets in-between those pertaining to the processed traffic trace, according to the relative attack intensity, ρ. The number of attack addresses generated depends on δ, the overlap between Ω and Ψ, and the number of destinations in $\texttt{BGP}_\phi$. Note that $\Omega \subseteq \texttt{BGP}_\phi$ and $\Psi \subseteq \texttt{BGP}_\phi$. Therefore, assuming the attack maximizes number of addresses used, to increase effectiveness, $|\Omega| = |\texttt{BGP}_\phi - \Psi| + \delta|\Psi|$. In particular, when no overlap exists, we generate $|\Omega| = |\texttt{BGP}_\phi - \Psi|$ new destination addresses while for a full overlap, the attack consists of $|\Omega| = |\texttt{BGP}_\phi|$ addresses. If $\delta \neq 0$, the addresses used in the attack and part of Ψ are uniformly distributed among those part of $\texttt{BGP}_\phi - \Psi$.

We validate the model considering two scenarios: an attack when the overlap is complete and one when the overlap is zero. Thereby, the attackers may use as attack prefix set either the whole EID-prefix space or just the part not visited by the attacked network's clients. In the latter case, if the cache is not large enough to hold all prefixes, all attack packets would generate a cache miss. However, note that building such a prefix set would require full knowledge about the network's traffic. In the former case, some packet destinations may generate cache hits but $|\Omega|$ may potentially be much larger and this could prove beneficial to the attacker. In light of these properties we consider the two attacks as *worst case* scenarios, from the attacked network's perspective, for the situations when attackers respectively have or do not have knowledge about the attacked network's traffic.

Figure 5.1 compares the analytical and empirical results for the cache miss rate, when $\delta \in \{0, 1\}$ and $\rho \in \{0.01, 0.1, 0.5\}$. We present the results just for *upc 2012* since those for the other three traces are similar. It may be observed that for both no overlap and complete overlap the results exhibit little absolute error. In particular, for $\delta = 0$ when cache size is larger and miss rates are less than 10^{-4} the errors are more significant. As in the case of the simple cache model discussed in the previous chapter, this error is explained by the trace length and the reduced number of points used in estimating the higher end of $s(u)$. The effect is not noticeable for $\delta = 1$ because $s_a(u)$ reaches its maximum (saturates) for low values of u due to the larger attack set. Consequently, the fit is very good along the whole spectrum of cache sizes. Once the cache size reaches $|\mathtt{BGP}_\phi|$ the miss rate becomes 0 since there are no more destination prefixes outside those already present in the cache to generate a miss.

5.3 Cache Poisoning and Management Strategies

Looking at Figure 5.1, we see that if cache sizes are small, both attacks results in very high cache miss rate, including for $\rho = 0.01$, when the attack has a rate of only 644 packets per second. In this range, miss rate is almost independent of attack overlap, only slightly higher for $\delta = 0$ due to the informed selection of attack address space. However, when $\delta = 0$ the cache performance is much improved after a certain threshold is passed whereas for $\delta = 1$ it barely changes up to when cache size becomes $\mathtt{BGP}_\phi$ and miss rate drops to 0. In other words, the non overlapping attack may be absorbed with larger cache sizes while the overlapping one cannot. Perhaps counter-intuitively, overlapping attacks are more damaging against a map-cache. They are easier to generate, as they do not require prior knowledge about the attacked network, but they are also harder to defend against since, after a certain point and for a wide ranges of values, increasing the cache sizes does not yield much improved performance, if less than $\mathtt{BGP}_\phi$.

Arguably, the most worrisome result we observe is the rather high miss rate which barely drops under 0.01, even for $\rho = 0.01$ and only when the normalized cache size is higher than 0.2. As a comparison, under normal operation this miss rate would be obtained with a normalized cache size of about 0.02, an order of magnitude less. Therefore, considering the high packet throughput of border routers, some more complex cache management strategies should be set in place to avoid hundreds to thousands of packet drops per second.

One possible first step to circumventing the effect of cache polluting attacks would be to detect them prior to taking action. This may be achieved with $s(u)$, if a ground truth estimation of its shape exists. So, if $s(u)$ is known beforehand, an estimate of the average miss rate for the cache size used can be computed. Then, if the instantaneous miss rate surpasses this estimate by more than configured threshold, a cache protecting action could be taken. For instance, the top, or most recently used, part of the cache could be "protected" against eviction. Other measures could include the implementation of a cache hierarchy or the limiting of user request rate for new destinations. In the former case, evicted entries should be stored in a larger but higher access time cache while in the latter some of the network elements should monitor per user traffic and filter out attack attempts. Since it is not within the scope of our work we do not explore other aspects related to the implementation of such tools.

5.4 Chapter Summary

To prove the versatility of our cache model but also to better understand the vulnerability of the map-cache in the face of malicious users, in this chapter we develop an extension capable to account for polluting attacks. If previously we observed that, in normal conditions, increasing cache size quickly diminishes miss rates we now found that it has little to no effect under simple cache pollution attacks. As a result, we advise that more complex management strategies be devised and set in place.

Chapter 6

LISP Control Plane Scalability

In this chapter we investigate the main source of temporal locality of reference in network traffic. We show that destination popularity is enough to entirely characterize the average working-set, independent of the much harder to model short term correlations. Leveraging the result from Chapter 4, we obtain a cache model that together with a small set of assumptions allows us to reason about asymptotic cache performance scalability.

6.1 Sources of Temporal Locality in Network Traffic

Consider the following formalization of traffic, either at Web page or packet level, we previously presented in Section 4.1 and which we repeat here for clarity. Let D be a set of objects (Web pages, destination IP-prefix, program page etc.). Then, we define traffic to be a strings of references $r_1, r_2, \ldots, r_i \ldots$ where $r_i = o \in D$ is a reference at the ith unit of time that has as destination, or requests, object o. Generally, we consider the length of the reference string to be N. Also, note that we use object and destination interchangeably.

Two of the defining properties of reference strings, important in characterizing cache performance, are the heavy tailed *popularity distribution* of destinations and the *temporal locality* exhibited by the requests pattern. We discuss both in what follows.

6.1.1 Popularity Distribution

Copious amounts of studies in fields varied as linguistics [105, 146], Web traffic [23, 94], video-on-demand [29], p2p overlays [40] and flow level traffic [119] found the probability distribution of objects to have a positive skew. Generally, such distributions are coined Zipf-like, i.e., they follow a power law; whereby the probability of reference is inversely proportional to the rank of an object. Typically, the relation is surmised as:

$$\nu(k) = \frac{\Omega}{k^{\alpha}} \tag{6.1}$$

where ν is the frequency, or number of requests observed for an object, k is the rank, $\Omega = 1/H(n, \alpha)$ is a normalizing constant and $H(n, \alpha)$ is the n^{th} generalized harmonic number.

It is interesting to note that although Zipf's law has its origins in linguistics, it was found to be a poor fit for the statistical behavior of words frequencies with low or mid-to-high values of the rank variable. That is, it does not fit the head and tail of the distribution. Furthermore, it's extension due to Mandelbrot (often called the Zipf-Mandelbrot law) only improves the fitting for the head of the distribution. Such discrepancies were also observed for Web based and p2p reference strings. Often the head of the distribution is flattened, i.e., frequency is less than the one predicted by the law, or the tail has an exponential cutoff or a faster power law decay [40, 105]. But these differences are usually dismissed on the basis of poor statistics in the high ranks region corresponding to objects with a very low frequency.

Nevertheless, Montemurro solved recently the problem in linguistics by extending the Zipf-Mandelbrot law such that for high ranks the tail undergoes a crossover to an exponential or larger exponent power-law decay. Surprisingly, he found this features, i.e. deviations from the Zipf-like behavior, to hold especially well when very large corpora [105] are considered. We further refer to this model as the Generalized Zipf law or GZipf and, in light of these observations, we assume the following:

Assumption 1. *The popularity distribution of destination IP-prefix reference strings can be approximated by a GZipf distribution.*

6.1.2 Temporal locality

Temporal locality can be informally defined as the property that a recently referenced object has an increased probability of being re-referenced. One of the well established ways of measuring the degree of locality of reference strings is the inter-reference distance distribution.

Breslau et al. found in [23] that strings generated according to the Independent Reference Model (IRM), that is, assuming that references are independent and identically distributed random variables, from a popularity distribution have an inter-reference distribution similar to that of the original string. Additionally, they inferred that the probability of an object being re-referenced after t units of time is proportional to $1/t$. Later, Jin and Bestavros proved that in fact temporal locality emerges from both long-term popularity and short-term correlations. However, they found that the inter-reference distance distribution is mainly induced through long-term popularity and therefore is insensitive to the latter. Additionally, they showed that by ignoring temporal correlations and assuming a Zipf-like popularity distribution then an object's re-reference probability after t units of time is proportional to $1/t^{(2-1/\alpha)}$. These observations then lead to our second assumption:

Assumption 2. *Temporal locality in destination IP-prefix reference strings is mainly due to the prefix popularity distribution.*

We contrast the two assumptions with the properties of several packet-level traces in 4.4. In what follows we are interested in characterizing the inter-reference distribution of a GZipf distribution and further on the cache miss rate using the two statements as support.

6.1.3 GZipf generated inter-reference distribution

In this section we compute the inter-reference distance distribution for a GZipf popularity. The result is an extension of the one due to Jin and Bestavros for a Zipf-like popularity. As a first step we compute the inter-reference distribution for a single object and then by integration obtain the average for the whole reference string, which we denote by $f(t)$.

If ν is the normalized frequency, namely, the number of reference to an object divided by the length of the reference string N, then, as shown in [105] the probability of observing objects with frequency ν in the reference string is:

$$p_\nu(\nu) \propto \frac{1}{\mu\nu^r + (\lambda - \mu)\nu^q} \tag{6.2}$$

where $1 \leq r < q$ are the exponents that control the slope of the power laws in the two regimes and μ and λ are two constants that control the frequency for which the tail undergoes the crossover.

From Assumption 2 it follows that references to an object are independent whereby the normalized inter-reference distance t is distributed exponentially with expected value of $1/(N\nu)$ and $0 < t < 1$. Then, if we denote by $d(t,\nu)$ the number of times the inter-reference distance for an object with frequency ν is t, we can write:

$$d(t,\nu) \sim (\nu N - 1)\nu N e^{-\nu N t} \tag{6.3}$$

If ν_{min} and ν_{max} are the minimum and respectively the maximum normalized frequency observed for the reference string, we can compute the inter-reference distance for the whole string as:

$$\begin{aligned} f(t) &\sim \int_{\nu_{min}}^{\nu_{max}} p_\nu(\nu)\, d(t,\nu)\mathrm{d}\nu \\ &\sim \int_0^1 \frac{(\nu N - 1)\nu N e^{-\nu N t}}{\mu\nu^r + (\lambda-\mu)\nu^q}\mathrm{d}\nu \end{aligned} \tag{6.4}$$

Unfortunately, the integral is unsolvable, nevertheless, we can still characterize the properties of $f(t)$ in the two regimes of the GZipf distribution. In the high frequency region, where term having q as exponent dominates the denominator we can write:

$$\begin{aligned} f_q(t) &\sim \int_{\nu_k}^1 \frac{\nu^2\ e^{-\nu t}}{\nu^q}\mathrm{d}\nu \\ &= \frac{\Gamma(3-q,\nu_k t) - \Gamma(3-q,t)}{t^{3-q}} \end{aligned} \tag{6.5}$$

where, $\Gamma(n,z) = \int_z^\infty x^{n-1}e^{-x}\mathrm{d}x$ is the incomplete Gamma function. $\nu_k = (\mu/(\lambda-\mu))^{1/(q-r)}$ is the frequency for which the two terms that make up the denominator are equal. It is useful to note that for low t values that correspond to high frequencies the nominator presents a constant plateau that quickly decreases, or bends, at the edges as $t \to 0$ and $t \to 1/\nu_k$. Therefore, we can approximate:

$$f_q(t) \sim \frac{1}{t^{3-q}} \tag{6.6}$$

Similarly, it may be shown that for low frequencies, that is, in the region where term with r as exponent dominates:

$$f_r(t) \sim \frac{1}{t^{3-r}} \tag{6.7}$$

Finally, we conclude that the inter-reference distance distribution can be approximated by a piece-wise power-law. Our result is similar to the single sloped power-law obtained by Jin under the assumption of Zipf distributed popularity or the empirical observations by Breslau et. al in [23] for Web reference strings. However, due to its general form it should be able to capture the properties of more varied workloads.

6.2 Updated Cache Model

Reusing the methodology we presented in Section 4.3, we deduce the miss rate of an LRU cache when fed by a reference string obtained using IRM and a GZipf popularity distribution. The miss rate for the upper part of $f(t)$ is:

$$m_q(t) = -\int \frac{C}{t^{3-q}}\mathrm{d}t = -C\frac{t^{q-2}}{q-2} \tag{6.8}$$

where, $t < 1/\nu_k$, $1 < q < 2$ and C is a normalizing constant which ensures that $\sum_{t=1}^{N-1} Cf(t) = 1$. We can further compute the average working-set size as:

$$s_q(t) = \int C\frac{t^{q-2}}{q-2}\mathrm{d}t = -C\frac{t^{q-1}}{(q-1)(q-2)} \tag{6.9}$$

To obtain the miss rate as a function of the cache size, not of the inter-reference distance, we take the inverse of s_q and replace it in (6.8). For $s < s_q(1/\nu_k)$ we get:

$$\begin{aligned} m_q(s) &= C^{\frac{1}{q-1}}(2-q)^{-\frac{1}{q-1}}(q-1)^{\frac{q-2}{q-1}}s^{\frac{q-2}{q-1}} \\ &\propto s^{1-\frac{1}{q-1}} \end{aligned} \tag{6.10}$$

This suggests that the asymptotic miss rate as a function of cache size is a power law of the cache size with an exponent dependent on the slope of the popularity distribution. Similarly, for large inter-reference distances, when $s > s_r(1/\nu_k)$:

$$m_r(s) \propto s^{1-\frac{1}{r-1}} \tag{6.11}$$

Then, for a reference string whose destinations have a GZipf popularity distribution and where the references to objects are independent, we find that the miss rate presents two power-law regimes with exponents only dependent on the exponents of the popularity distribution and the cache size. We test the ability of the equations to fit empirical observations in 6.3.3.

6.2.1 Cache Performance Analysis

We now investigate how cache size varies with respect to the parameters of the model if the miss rate is held constant. By inverting (6.10) and (6.11) we obtain the cache size as a function of the miss rate:

$$s(m) = \begin{cases} g(q)\, m^{1-\dfrac{1}{2-q}}, & m \leq m_k \\ g(r)\, m^{1-\dfrac{1}{2-r}}, & m > m_k \end{cases} \tag{6.12}$$

with $g(x) = -C^{\frac{1}{2-x}} \dfrac{(2-x)^{\frac{x-1}{x-2}}}{2-3x+x^2}$, $m_k = \dfrac{C}{\nu_k^{r-2}(2-r)}$, $\nu_k = \left(\dfrac{\mu}{\lambda-\mu}\right)^{q-r}$ and $0 < m < 1$.

We see that $s(m)$ is *independent* of both the number of packets N and the number of destinations D and is sensible only to changes of the slopes of the popularity distribution q, r and the frequency at which the two slopes intersect, ν_k. We do note that C does depend analytically on N as it can be seen by considering C's defining expression (see discussion of (6.8)): $1/C = H(1/\nu_k, 3-q) - \zeta(3-r, N) + \zeta(3-r, 1/\nu_k)$ where $H(n,m) = \sum_{k=1}^{n} 1/k^m$ is the generalized harmonic number of order n of m and $\zeta(s,a) = \sum_{k=0}^{\infty} 1/(k+a)^s$ the Hurwitz Zeta function. However, the first and last terms of the expression depend only on popularity parameters while the middle one quickly converges to a constant as N grows. Whereby it is safe to assume C constant with respect to N and consequently that the number of packets does not influence $s(m)$.

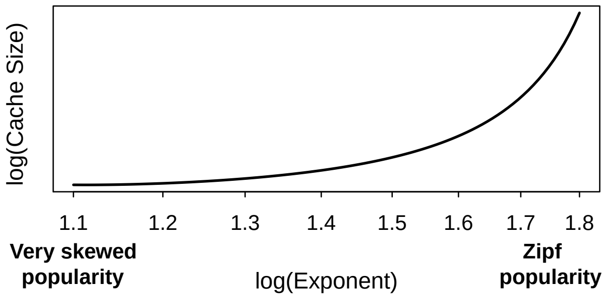


Fig. 6.1 Cache size as a function of a GZipf exponent for a fixed miss rate

On the other hand, if the parameters of the popularity distribution are modified, some interesting dependencies can be uncovered. For brevity, we explore only the case when q and r vary but still respect the constraint that $1 < r < q < 2$. When both exponents jointly change, the cache size required to maintain the miss rate will qualitatively vary as depicted in Fig. 6.1. Specifically, as their value approach 1, that is, when the popularity distribution is strongly skewed, cache size asymptotically goes to a low value constant, whereas when the exponent approaches 2, the required cache size grows very fast (notice the superlinear growth in the log-log scale). Despite not being indicated by (6.12), $s(m)$ is defined when q or r are 2, that is, it does not grow unbounded. The expression can be obtained if we replace q by 2 in (6.8) and recompute all equations:

$$s(m) = (C + m)\, e^{-\dfrac{m}{C}} \tag{6.13}$$

6.2.2 Discussion of Asymptotic Cache Performance and Impact

Using the results of the analysis performed in the previous section we are now interested to characterize the asymptotic scalability of the LISP cache size with respect to (i) the number of users in a LISP site (ii) the size of the EID space and (iii) the parameters of the popularity distribution. To simplify the discussion, we assume there are no interactions between the first two and the third:

Assumption 3. *The destination prefix popularity distribution is independent of the number of users in a LISP site and the size of the EID space.*

Whereby (i) contemplates the variation of the number of packets, N (ii) the variation of the number of destinations D and (iii) the variation of the GZipf parameters q, r, μ and λ, independently. We acknowledge that the popularity distribution may be influenced by a multitude of factors, and in particular by the growth of the users generating the reference string. Nonetheless, we argue that our assumption does make practical sense. For instance, a typical LISP router is expected to serve hundreds to thousands of clients so fluctuations proportional to the size of the user set should not affect overall homogeneity and popularity distribution. Additionally, although user interest in content quickly changes, the same is not necessarily true for the content sources, i.e., prefixes from where the content is served, which the user cannot typically select. This split between content and its location can result in relatively stable popularity distribution of the prefixes despite the dynamic popularity of actual content. We show an example network where this assumption holds in Section 6.3.1.

In the previous section we found that when the parameters of the popularity distribution are held constant, the cache size is independent of both the number of packets and destinations. As a result, cache size scales constantly, O(1), with the number of users within a LISP site and the size of EID-prefix space for a fixed miss rate. This observation has several fundamental implications for LISP's deployment. First, caches for LISP networks can be designed and deployed for a desired performance level which subsequently does not degrade with the growth of the site and the growth of the Internet address space. Second, splitting traffic between multiple caches (i.e., routers) for operational purposes, within a large LISP site, does not affect cache performance. Finally, signaling, i.e., the number of Map-Request exchanges, grows linearly with the number of users if no hierarchies or cascades of caches are used. This because the number of resolution requests is $m(s)\,N$.

If the previous assumption does not hold, then cache size scales linearly with the $|D|$. This follows if we consider that, as the growth of N and D flatten the distribution, thus leading to a uniform popularity, the cache size for a desired miss rate becomes proportional to the $|D|$.

6.3 Empirical Evidence of Temporal Locality

In this section we verify the accuracy of our assumptions regarding the popularity distribution of destination prefixes and the sources of locality in network traffic. We also verify the accuracy of the predictions regarding the performance of the LISP cache empirically. But first, we present our datasets and experimental methodology.

6.3.1 Popularity Distribution

Figure 6.2 presents the frequency-rank distributions of our datasets for both absolute and normalized frequency. A few observations are in place. First, although clearly not accurately described by Zipf's law, they also slightly deviate from a GZipf. Namely, the head of the distribution presents two power-law regimes followed by a third that describes the tail as it can be seen in Fig. 6.2 (down). This may be either because a one day sample is not enough to obtain accurate statistics in the Zipf-Mandelbrot head region, or because popularity for low ranks follows a more complex law. Still, we find that for all traces the frequencies of higher ranks (above 2000) are accurately characterized by two power-law regimes (see Fig. 6.4).

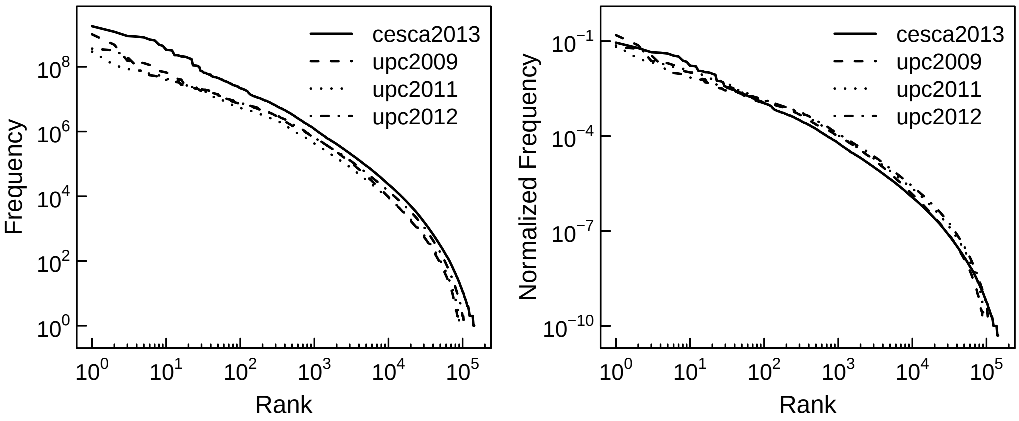


Fig. 6.2 Destination Prefix Popularity

Secondly, the frequency-rank curves for the UPC datasets are remarkably similar. Despite the 50% increase of $\mathtt{BGP}_\phi$ (i.e., D), changes in the Internet content provider infrastructure over a 3.5 years period, and perhaps even changes in the local user set, the popularity distributions are roughly the same.

Finally, the normalized frequency plots for all traces are similar, in spite of the large difference in number of packets between CESCA and UPC datasets. These observations confirm our assumption that growth of the number of users within the site or of the destination space do not necessarily result in a change of the popularity distribution.

To confirm that these results are not due to a bias of popularity for larger prefixes sizes, that is, larger prefixes are more probable to receive larger volumes of traffic because they contain more hosts, we checked the correlation between prefix length and frequency. But (not shown here) we didn't find any evidence in support of this.

6.3.2 Prefix Inter-Reference Distance Distribution

We now check if knowledge about the popularity distribution suffices to accurately characterize the inter-reference distance distribution or if short-term correlations must also be taken into account. To achieve this, we use a methodology similar to the one used in [81] for Web page traffic. We first generate random versions of our traces according to the IRM model, i.e., by considering only the popularity distribution and geometric inter-reference times, and then compare the resulting inter-reference distance distributions to the originals. Results are shown in Fig. 6.3.

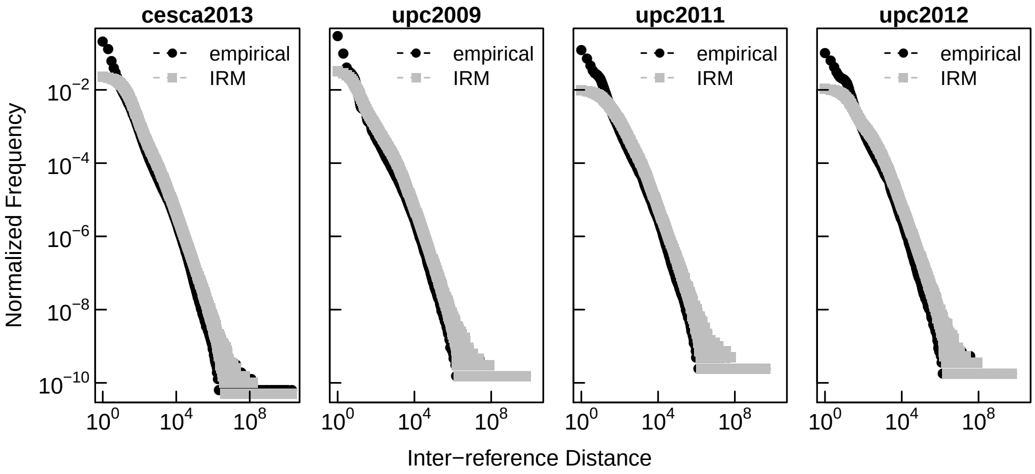


Fig. 6.3 Empirical and IRM generated inter-reference for the four traces

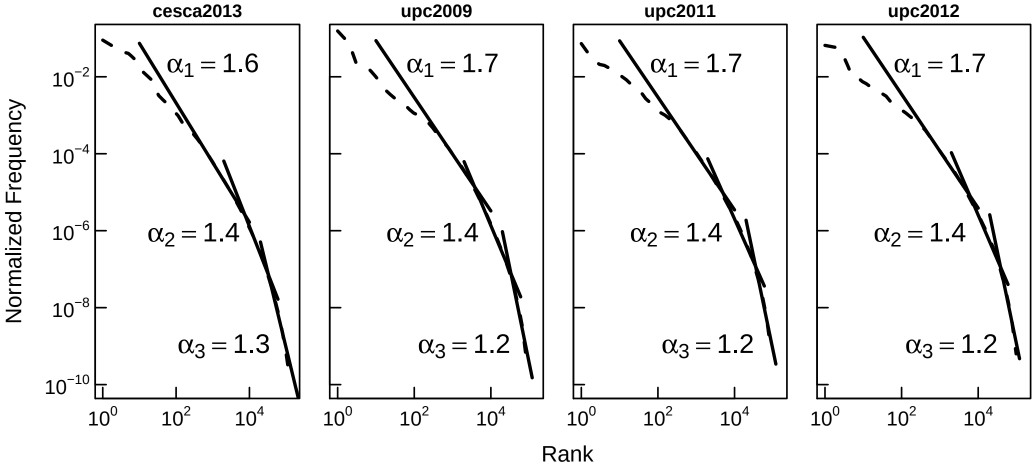


Fig. 6.4 Frequency-rank distribution of destination prefixes and a linear least squares fit of the three power-law regimes. $\alpha_i = 1 + 1/s_i$, where s_i is the slope of the ith segment.

We find that for all traces, popularity alone is able to account for the greater part of the inter-reference distance distribution, like in the case of Web requests. The only disagreement is in the region with distances lower than 100 where short-term correlations are important and IRM traces underestimate the probability by a significant margin.

A rather interesting finding is that the short-term correlations in all traces are such that the power-law behavior observed for higher distances ($t > 100$) is extended up to distance 1. In this region, the exact inter-reference distance equation (6.5) is a poor fit to reality as it follows the IRM curve. However, the empirical results are appropriately described by our approximate inter-reference model (6.6) which avoids IRM's bent by assuming (6.5)'s numerator constant.

6.3.3 Cache Performance

Having found that our assumptions regarding network traffic properties hold in our datasests we now investigate if the cache model (see (6.10) and (6.11)) is able predict real world LRU cache performance.

As mentioned in Section 6.3.1 and as it may be seen in Fig. 6.4, the head of the popularity distribution exhibits two power-law regimes instead of one. Then, two options arise, we can either use the model disregarding the discrepancies or adapt it to consider the low rank region behavior. For completeness, we choose the latter in our evaluation. This only consists in approximating $p_\nu(\nu)$ (see (6.2)) as having three regions, each dominated by an exponent α_i. Recomputing (6.11) we get that the miss rate has three regions, each characterized by an α_i. However, choosing the first option would only result in an overestimation of cache miss rates for low cache sizes.

To contrast the model with the empirical observations, we performed a linear least squares fit of the three regions of the popularity distribution. This allowed us to determine the exponents α_i, computed as $1 + 1/s_i$ where s_i is the slope of the ith segment, and to roughly approximate the frequencies ν_{k1} and ν_{k2} at which the segments intersect. Using them as input to (6.10) we get a cache miss rate estimate as shown in Fig. 6.6. Generally we see that the model is a remarkably good fit for the large cache sizes but constantly underestimates the miss rate for sizes lower than 1000. This may be due to the poor fit of the popularity for low ranks. Nevertheless a more elaborate fitting of ν_{k1} and ν_{k2} should provide better results as it may be seen in Fig. 6.5 where we performed a linear least squares fit of the three power law regions of the cache miss rate. Knowing that the slope of the cache miss rate is $s_i = 1 - 1/(\alpha_i - 1)$ (see (6.8)), we computed the exponents as depicted in the figure. Comparison with those computed in Fig. 6.4 shows they are very similar. Overall, we can conclude that the model accurately predicts cache performance.

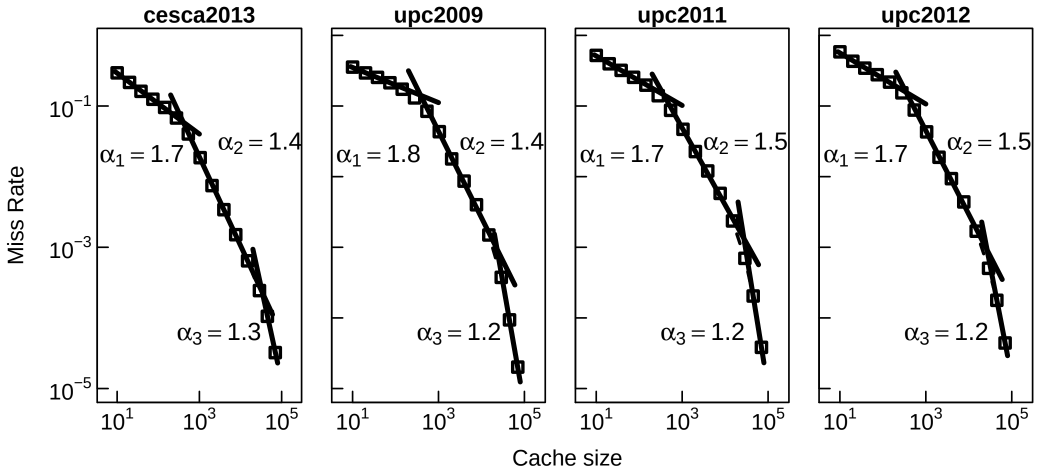


Fig. 6.5 Empirical miss rate with cache size and a linear least-squares fit of the exponent for the three power-law regions. Notice the similarity with the exponents of the three regions of the popularity distribution in Fig 6.4.

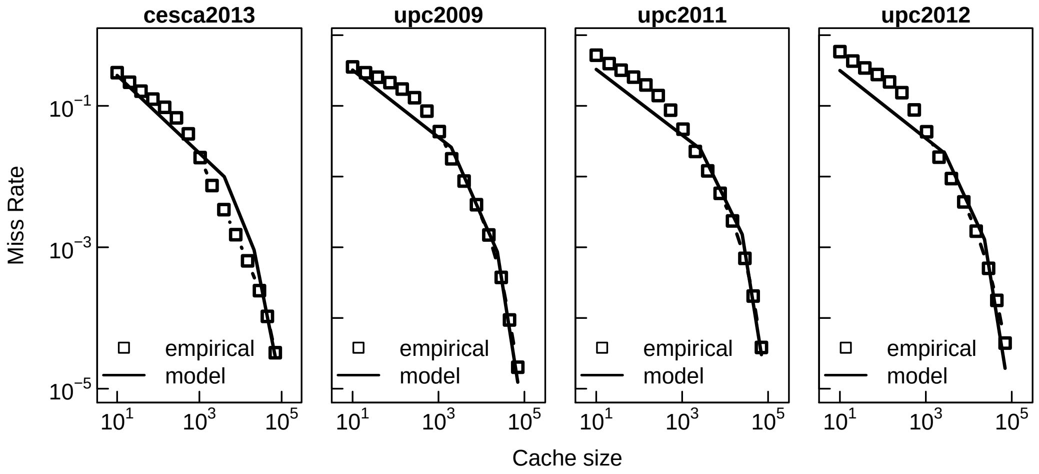


Fig. 6.6 Empirical miss rate with cache size together with a fit by (6.10) and (6.11)

6.4 Related work

As explained in Section 4.1, Denning was first to recognize the phenomenon of temporal locality in his definition of the working-set [45] and together with Schwartz established the fundamental properties that characterize it [46]. Although initially designed for the analysis of page caching in operating systems, the ideas were later reused in other fields including Web page and route caching.

In [23] Breslau et al. argued that empirical evidence indicates that Web requests popularity distribution is Zipf-like of exponent $\alpha < 1$. Using this finding and the assumption that temporal locality is mainly induced through long-term popularity, they showed that the asymptotic miss rates of an LFU cache, as a function of the cache size, is a power law of exponent $1 - \alpha$. In this chapter we argue that GZipf with exponents greater than 1 is a closer fit to real popularity distributions and obtain a more general LRU cache model. We further use the model to determine the scaling properties of the cache.

Jin and Bestavros showed in [81] that the inter-reference distribution is mainly determined by the the long-term popularity and only marginally by short-term correlations. They also proved that the inter-reference distribution of a reference string with Zipf-like popularity distribution is proportional to $1/t^{2-1/\alpha}$. We build upon their work but also extend their results by both considering a GZipf popularity distribution and by using them to deduce an LRU cache model.

6.5 Chapter Summary

In this chapter we answer the following question: does the newly introduced LISP edge cache scale?

Our findings show that the miss rate scales constantly O(1) with the number of users as well as with the number of destinations. For this, we start from two assumptions: (i) the popularity of destination prefixes is described by a GZipf distribution and (ii) temporal locality is predominantly determined by long-term popularity. Fundamentally, these assumptions are often observed to hold in the Internet [83, 119] but also in other fields such as web traffic [23], on-demand video [29] or even linguistics [146]. Arguably, they are inherent to human nature and, as such, are expected to hold in the foreseeable future. Nevertheless, we also show that if the converse holds, then cache size scales linearly O(N) with the number of destinations.

At the time of this writing there is an open debate on how the Internet should look like in the near future and in this context, it is important to analyze the scalability of the various future Internet architecture proposals. This chapter fills this gap, particularly for the Locator/ID split architecture. Furthermore, our results show that edge networks willing to deploy LISP will not face scalability issues -as long as both assumptions hold- in the size of their map-cache, even if the edge network itself becomes larger (i.e., more users) or the Internet grows (i.e., more prefixes).

Part III

Advanced Overlaid Services

Chapter 7

LISP-based inter-domain multicast

7.1 Motivation

The Internet is gradually becoming the preferred infrastructure for delivering live content such as sports events or news to large user sets. According to recent reports, video streaming is among the largest and the fastest growing bandwidth consumers [86] and IPTV driven revenues are to rise from less less than USD 9.7B in 2011 to USD 21.3B in 2017 [106].

For such scenarios, where one-to-many content delivery to large number of receivers is required, IP multicast [44, 68] is perhaps the most efficient solution in terms of bandwidth consumption. However, although often supported within the confinements of campus, enterprise or service provider networks, IP multicast deployments have been generally done disregarding inter-domain connectivity, thereby resulting in disconnected multicast *islands* [82, 139, 144]. One fundamental cause for the slowly advancing deployment is the requirement that *all* routers be upgraded to support the protocol. But other, frequently cited, reasons regard management complexity and the lack of a clear commercial service [49]. While the former incurs high operational expenditure the latter leads to situations when multicast implementation over links with unicast economical agreements result in loss of revenue.

The low uptake of IP multicast has led over the last decade to the development of many application-layer multicast (ALM) solutions that build end-host overlays to ensure Internet wide content dissemination. They are designed to be very flexible in accommodating globally spread users and adaptable to changing network conditions.

However, the incongruence they introduce between overlay and underlying network topology diminishes their delivery efficiency with respect to that of IP multicast (see Appendix A). Furthermore, performance analysis has uncovered significant limitations of these architectures in scaling user quality of experience with the increase of client population [137]. Reasons often reported are unavailability of inter-peer bandwidth, churn or, in some cases, insufficient upload capacity. Since they are related to either end-host behavior or upload abilities, these limitations are intrinsic to the overlay's design and, therefore, not avoidable through optimized overlay management.

In this chapter, we propose a network-layer single-source multicast framework designed to merge the benefits of IP multicast and ALM while avoiding their respective deployment and scaling issues. Our goal is to enable large scale single-source streaming by interconnecting existing multicast capable domains devoid of end-host software upgrades and transparently for the greater part of existing multicast routers. This approach is complementary to existing ALM solutions as it aims to offer overlay management control to network operators in exchange for improved reliability and more efficient network resource use but at the cost of minimal infrastructure support. To achieve our goal, we exploit the unique window of opportunity offered by the development and deployment of LISP, which we use as support for our proposal and in light of the intrinsic dependency, we refer to our solution as *LISP-casting* or shortly, *Lcast*.

Lcast creates and optimizes a LISP router overlay and transparently interfaces with end-hosts and legacy multicast routers by means of existing IP multicast protocols [26, 44, 68]. Group management functions are *logically centralized* and performed by an overlay coordinator whereby members require no prior configuration nor need to be manually managed. We stress this as a crucial property since it circumvents the management complexity issue that plagues traditional IP multicast deployment and further opens the possibility to dynamically optimize delivery with respect to overlay topology maps. This ability could be exploited by content providers to define their own overlay coordinating algorithms or perform on-line switching between multiple ones, according to specific operational requirements or economical agreements. In this sense, Lcast is akin to Software-Defined Networking (SDN) [97] and, in fact, the two share many of the properties that derive from the implementation of a programmable control plane.

From an architectural standpoint, Lcast's ability to accommodate large number of clients is ensured through design, by decoupling the control and forwarding functions within the overlay. As a result, each can evolve, be optimized and scale according to specific needs. In particular, data plane scalability can be achieved by constraining router replication factors, to avoid performance penalties due to unicast replication inefficiency, while control plane scalability may be ensured by limiting communication overhead. In both cases the trade-off is overlay efficiency. To assess control plane scalability, the impact of replication factors on efficiency and overlay configurability, we evaluate Lcast's ability to deliver low latency content in three distinct operational setups. Our simulations make use of (*i*) an Internet-like autonomous system (AS) level topology and (*ii*) large client traces that emulate realistic client behavior, consisting of $3k$ ASes and approximately $140k$ unique IPs obtained through a globally distributed capture of SopCast [5] overlays.

First of all, the results show the control plane's ability to scale. Even when active topology discovery mechanisms are used and client churn is high, the load is manageable by a single server acting as overlay coordinator. Second, replication factors need not be large for efficient content delivery. Finally, the overlay can be easily optimized considering various operational constraints. Notably, if inter-member latency can be estimated, Lcast can deliver content at close to unicast latencies, independent of the overlay's size.

The rest of the chapter is structured as follows. Section 7.2 describes the Lcast framework and Section 7.3 introduces an optimization algorithm and two ways to obtain topology maps that may be used to optimize the overlay. Section 7.4 presents our evaluation methodology and in Section 7.5 we discuss the results. We discuss the related work in Section 7.6 and finally conclude the chapter in Section 7.7.

7.2 Lcast Architecture

This section presents our proposal for supporting inter-domain multicast streaming. We start by providing an overview of the architecture and then describe in greater detail group management procedures and signaling.

7.2.1 Architecture Overview

Lcast is a LISP extension that provides a single-source multicast service to clients in disjoint multicast islands by means of a router overlay. It compensates for the lack of an inter-domain multicast infrastructure by performing unicast encapsulated, and if possible also multicast encapsulated, replication of multicast traffic across the Inter-

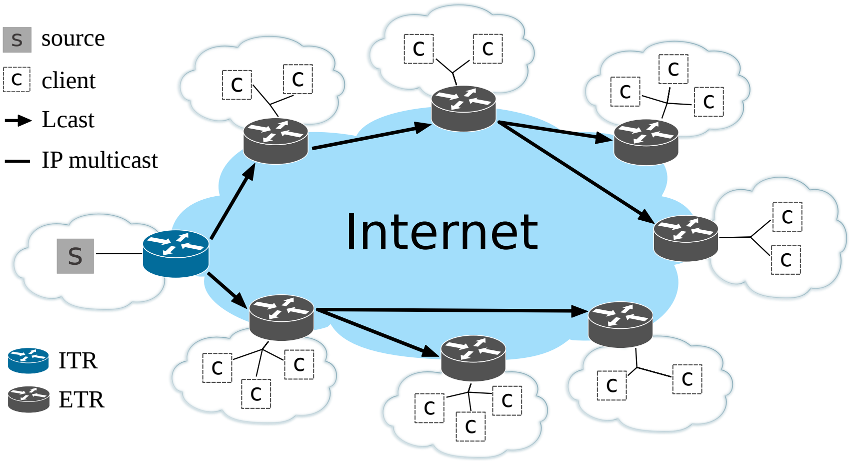


Fig. 7.1 Example Lcast data-plane architecture. The ITR is the first to replicate the content and all downstream ETRs may replicate up to a fixed fan-out value. In the example, fan-out is constrained to 2.

net's core. The resulting overlay interfaces with existing intra-domain IP multicast protocols so it does not require any end-host software upgrades. All member domains must be LISP enabled and may participate in the overlay with at least one of their border routers (ETRs). On the data path, the source domain's border router, an ITR, heads the distribution tree and is the first to perform encapsulated replication. Subsequently, all downstream overlay members, save for the leaves, replicate the received packets up to a certain fan-out. Note that since traffic is unidirectional, from multicast source to clients, all routers but the source ITR perform either only decapsulation or decapsulation and re-encapsulation. For brevity we refer to all of them as ETRs, although those that perform both functions also implement RTR functionality. See Figure 7.1 for a depiction of an example Lcast data plane.

An important drawback to unicast encapsulated replication is that it reduces throughput proportionally to the replication factor, if performed multiple times out the same interface. As a result, increasing fan-out can quickly saturate router interfaces and therefore not only deteriorate overlay performance but also congest other flows sharing the same links. Additionally, since packet replication is performed sequentially, the time difference between the instance the first and last replicas are forwarded may be considerable. So, besides increasing the delay to obtaining the multicast packets for directly connected downstream members, the resulting latency may accumulate and distribute unevenly across the hierarchy, randomly leading to branches with low performance. Finally, apart from the limitations concerning performance, unbounded fan-out can also lead to unfair and/or economically unfeasible situations. Generally, Lcast and other island multicast solutions substantially reduce inter-domain traffic exchange, if compared to simple unicast delivery or unoptimized P2P overlays (see Section 7.5). However, if the distribution tree is not carefully constructed, member routers serving few clients might be requested to replicate a disproportionate number of times, against their interest and to the advantage of other economically benefited peers. To avoid these inefficiencies and in the interest of fairness, we request that Lcast members have a *constrained fan-out* whereby the overlay must be organized as a degree-constrained tree, despite the potential to reduce distribution efficiency. Fan-out values could be fixed for the whole overlay or reported at subscription by each member.

It should be noted that, similarly to end-host overlays, purely replicating routers, RTRs, could be provisioned in transit domains to ensure improved overlay stability, performance and also considerably reduce or remove altogether the replication overhead of ETRs. However, such a solution also implies a business model, different to the one discussed here, where third party entities manage the RTRs and engage in economical agreements with the source and client edge-domains. Since we are not ready to model business relationships, or speculate how such RTRs could be deployed, we limit the analysis to overlays where replication is performed exclusively by ETRs. For a technical discussion on how RTRs could be configured to participate in the overlay, we refer the interested reader to an Internet-Draft [39] we published on the subject.

Another limitation to having routers participate in an overlay, is that they are generally inefficient at handling complex computation tasks since they are designed to perform fast forwarding of packets as opposed to general purpose computing. To circumvent this drawback, Lcast leverages LISP's native separation between data and control plane to ensure the logical centralization of group management functions in an overlay coordinator. Thereby, at data plane level routers only perform encapsulated replication while at control plane level, the coordinator must compute a distribution tree and ensure members are organized according to it. Besides supporting our original goal of having no management costs for routers, this design also opens the possibility for enhanced overlay configurability. In this sense, if the coordinator obtains or is configured with a map of the locator underlay, it may proceed to optimize the distribution tree with respect to a given metric. Moreover, the architecture allows the switching between multiple optimization functions or metrics, even on-line, to meet changing operational requirements. Note that we are not the first to propose such split. This ideas has been previously recommended to aid routing scalability [58] and is nowadays central to current SDN research [138].

Possible implementers of the overlay coordinator may be the source ITR or the MS. Reusing the previous argument, since the ITR is a router, we consider the MS better fit for the function. In fact, due to its position in the LISP control plane, the MS is required to process and provide answers to all `Map-Requests` originated by routers willing to initiate communication with the multicast source. Then, as it can recognize and keep track of all overlay members, the MS should also be the one to decide the attachment point for a joining member or the one to optimize the overlay. Although the design allows for the overlay state and/or management functions to be distributed, in this article we are interested in evaluating if the control plane overhead is sustainable by one, off-the-shelf, server acting as MS.

Lcast is compatible with the current LISP specification, but apart from the canonical LISP messages we introduced in Section 3, it additionally requires the signaling messages defined in [54] for conveying joining (`Join-Request` message) and leaving (`Leave-Request` message) multicast information. They are not Lcast specific and have been designed to simplify the connecting of multicast capable sites with LISP-Multicast.

7.2.2 Member Subscription

In Lcast enabled domains, end-hosts request single-sourced multicast content, the way they do with traditional IP multicast, namely, by subscribing to a multicast stream using the Internet Group Management Protocol (IGMP) [68, 69]. They learn the *channel identifier* (S-EID,G), consisting of the multicast source address S-EID and a multicast group address G, used to distinguish between the multiple channels originated by a source, either through configuration or with the help of application-layer protocols. Except in the particular case when S-EID is part of the same domain, and therefore the content may be delivered locally without Lcast, the subscription request propagates intra-domain up to one of the domain's border routers, an ETR. If the ETR is already a member of the multicast channel, it starts replicating the multicast content towards the requesting end-host and no further action is taken. If not, the ETR initiates a two step overlay join procedure whereby it first attaches to the Lcast overlay serving (S-EID,G) and secondly it advertises its ability to replicate multicast content.

To complete the first step, the ETR must initially obtain the locator, of at least one of the already connected routers, to be used as overlay attachment point. It achieves this by requesting that the channel identifier be mapped to the locators of potential overlay parents, in essence, by sending a `Map-Request` for (S-EID,G). The request propagates through the mapping system up to the coordinating MS, which ensuing the request's receipt, starts a search for overlay members with spare capacity. The search may be done randomly or, if additional topological information exists, in accordance to a predefined heuristic that could ensure that an optimal attachment point is chosen. Once obtained, the result, consisting of a list of one or multiple RLOCs pertaining to the overlay members able to accommodate new children, is conveyed to the joining ETR in a `Map-Reply`. Typically, the MS will offer an ETR the possibility of choosing its upstream either when not optimizing the overlay or when all the choices have an equal cost in the distribution tree. Using local policy (e.g., shortest AS path) and the priority and weight values associated to the list entries, the ETR chooses the best RLOC and sends it a `Join-Request` message to request the setting up of an overlay branch between the two. The parent router appends the RLOC of the joining ETR to the list towards which it performs unicast encapsulated replication, therefore concluding the ETR's attachment. Alternatively, if the two routers can be connected by inter-domain multicast, the joining ETR first performs a protocol dependent multicast join to the parent in the underlying inter-domain network. Afterwards, it indicates the multicast channel identifier, to be used as destination for the multicast encapsulated packets (as opposed to unicast encapsulated), in the `Join-Request`.

A special case arises when an ETR is first to join the overlay. In this situation, the ETR requests the multicast content from the ITR, but it may happen that the ITR is not yet subscribed to (S-EID,G). Therefore, on receipt of the `Join-Request`, the ITR must first subscribe to the multicast source, using IGMP or a PIM Join [60] message, to obtain the streamed content to be replicated towards the joining ETR.

The second step in a member's subscription procedure is to signal that it can perform replication within the overlay. To this end, once the ETR is attached, it starts registering (S-EID,G) with the MS. The `Map-Register` message conveys the ETR's RLOC, that of the chosen parent and the number of intra-domain clients it serves at a certain time instant. To be noted that the estimating of the membership in a multicast session, although traditionally a difficult task, can be achieved within a domain using the explicit tracking capabilities of both IGPM and PIM. Then, having for a channel identifier the registration messages of all the overlay members, allows the MS to build an aggregated (S-EID,G) mapping that provides a complete view of the overlay's topology, i.e., of how the members organize in a distribution tree. This mapping, together with underlay topological information, if any exists, is used by the MS to answer `Map-Requests` of joining ETRs and to optimize the distribution tree. Two additional benefits of the procedure, since registrations are refreshed periodically, are that the MS implicitly detects the failure of an ETR and also becomes aware of the changes in client population within a domain.

The way the distribution tree is built has two advantages. First, it ensures that multicast packets in the source domain do not reach the ITR if no ETR is joined and the ITR does not participate in a local multicast group. Second, packets are forwarded from ITR to all ETRs without mapping database lookups thus, with minimum overhead.

7.2.3 Member Failure and Unsubscription

If a member loses network connectivity, its data path children will sense the failure either as a lack of multicast packets or by means of a LISP specific mechanism, called *RLOC-Probing*. This procedure, used by xTRs, consists in the use of `Map-Request` messages to determine reachability of peer xTRs and to estimate round-trip times (RTT). Once the children detect the failure they look for new overlay parents by either sending `Join-Requests` to other RLOCs in the mapping associated to (S-EID,G) or, if no other exists, by redoing the whole subscription procedure. Still, such circumstances will result in packet loss for all members of the subtree headed by the affected router and out of band mechanisms would be required for remedying the failure. Such mechanisms are out of the scope of this dissertation. However, sudden loss of network connectivity for a domain's border router should be a seldom occurrence.

An ETR initiates its unsubscription from the Lcast overlay when the last of its clients leaves the intra-domain multicast group. First, if the ETR replicates content to other overlay members, it increments the priority of the RLOC registered with the MS to the least preferable value and replies to RLOC-Probing messages indicating that its RLOC is unreachable. The update forces the MS to avoid assigning the ETR new children and forces the existing ones to find new overlay parents. The lookup procedure is identical to the one followed in the event of a parent failure however, in this case there are no packet losses. Alternatively, when the MS senses the departure of an ETR, it could proceed to optimizing the whole affected subtree to avoid performance degradation. As a second step, the ETR sends a `Leave-Request` to its overlay parent and stops registering (S-EID,G) with the MS, concluding the unsubscription.

7.2.4 Distribution Tree (Re)Configuration

The position of an overlay member on the data-path is established at subscription time, however the MS could be configured to further optimize the distribution tree, if provided with information about the overlay's topology. In this case, distribution efficiency is controlled by the MS through optimal placement of joining ETRs and/or through periodic or enforced tree reshaping.

When reorganizing the distribution tree, the MS informs members of their new positions through updated mappings. To avoid packet loss and to assure a seamless transition, members use a *make before break* procedure when switching parents. Hence, prior to sending `Leave-Request` to their overlay parents, members first attach to those indicated in the updated mappings. If any duplicate packets arise, they should be discarded by end-hosts at application layer.

This type of centralized management enables the easy customization of the distribution tree as routers are oblivious to optimization algorithm changes. In fact, a *key feature* of Lcast, is that an operator in control of the MS can perform on-line switching between multiple optimization algorithms or topology discovery protocols, if more are supported, to better adapt the overlay to operational constraints. In the next section, we provide as example a possible tree optimizing algorithm and ways of obtaining topological information that could be implemented in Lcast.

7.3 Overlay Optimization

The configuration of the Lcast overlay controller is application and operator specific. To illustrate this point, in this section we consider the classical problem of delivering latency constrained content (e.g., live streaming of news and sports events) and show how Lcast could be used to solve it. We first propose an algorithm to compute the distribution tree and afterwards point out how ITR-local BGP routing tables and organized inter-domain latency measurements could be used to approximate overlay topology. For brevity, we further refer to the combination of an optimization algorithm and a topology discovery mechanism as an *optimization strategy.*

7.3.1 Distribution Tree Optimization Algorithm

In what follows, we use 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. Considering our goal of delivering content for delay sensitive applications, the function we minimize in our experiments is the maximum distance (e.g. latency or number of AS hops) from a client to the multicast source. Notice that the reference is the end-host and not the domain border router (ETR). Thus, what matters in deciding an ETR's position in the overlay tree is not solely its distance to the ITR 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 distance but with a larger client set. In other words, the algorithm tries to improve average end-host quality of experience by optimizing the router overlay considering two dimensions, inter-router distance and the size of the client set served by a router. This also ensures the algorithm is fair to members. Domains with fewer clients are more likely to become leaves while those with larger user sets, the ones that benefit most from Lcast, are required to contribute by replicating.

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

Definition 2. *Given an undirected complete graph G=(V,E), a designated vertex* $r \in V$*, a degree bound* $d(v) \leq d_{max}$*,* $\forall\ v \in V$*,* $d_{max} \in \mathbb{N}$*, a vertex weight function* $c(v) \in \mathbb{N}$ *and an edge weight function* $w(e) \in \mathbb{R}^{+}$*,* $\forall$ *edge* $e \in E$*. Let* $P^{T}_{r,v}$ *be the set of edges e on the path from vertex r to v in the graph's spanning tree T. Also, let* $W^{T}_{r,v} = \sum_{e \in P^{T}_{r,v}} w(e)$ *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* $d_T(v) \leq d_{max}$*, such that* $\sum_{v \in V, v \neq r} c(v) W^{T}_{r,v}$ *is minimized.*

We note that [126] and [19] have previously defined and solved similar optimization problems. Shi et al. [126] also proved 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. [19] have successfully managed to distribute the algorithm.

The heuristic we used to solve the MADDBST problem is similar to the one used by Banerjee and it is a variant of the one proposed by Shi. In short, the algorithm

Algorithm 1 Heuristic used to solve the MADDBST problem

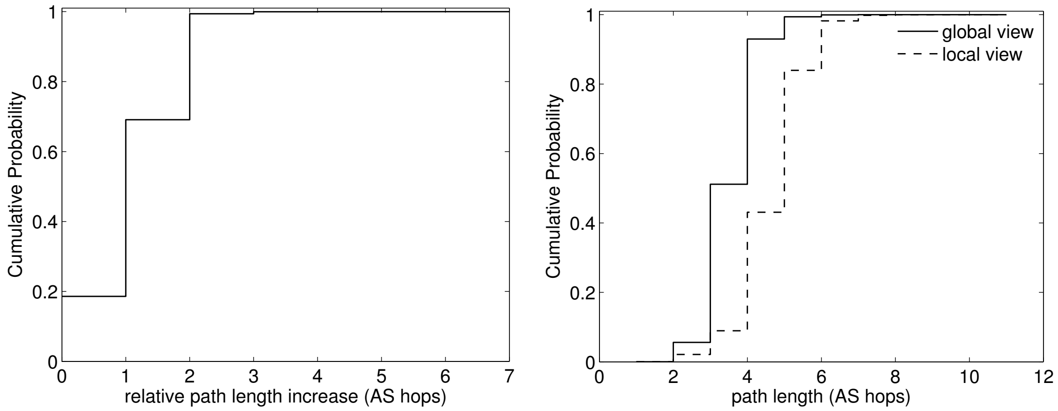
```
Input: $G = (V, E)$; $r$; $w(u, v)$, $\forall u, v \in V$; c(v), $\forall v \in V$; $d_{max}$
Output: T
  foreach $v \in V$ do
    $\delta(v) = w(r, v)/c(v)$;
    $p(v) = r$;
  end for
  $T \leftarrow (U = \{r\}, D = \{\})$;
  while $U \neq V$ do
    let $u \in V - U$ be the vertex with the smallest $\delta(u)$;
    $U = U \cup \{u\}$; $L = L \cup \{(p(u), u)\}$;
    foreach $v \in V - U$ do
      $\delta(v) = \infty$;
      foreach $u \in U$ do
        if $d_T(u) < d_{max}$ and $(W^T_{r,u} + w(u, v))/c(v) < \delta(v)$ then
          $\delta(v) = (W^T_{r,u} + w(u, v))/c(v)$;
          $p(v) = u$;
        end if
      end for
    end for
  end while
```

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, it selects a potential parent node u in the tree T, such that the metric $\delta(v) = (W^T_{r,u} + w(u, v))/c(v)$, i.e., 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.

7.3.2 BGP-based Topology Map

One of the best sources of topological information that is not or can not be commonly used by application layer overlays is the BGP routing table. 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 information may exist.

The ITR has two options for obtaining BGP topological information. First, it may aggregate partial BGP feeds from multiple overlay members (*global view*) or second, it may itself connect to BGP (*local view*). The former could ensure a more detailed description of the topology, and thus grounds for better decisions, while



(a) Cumulative probability of the relative path length increase due to local view

(b) Cumulative probability of the AS path lengths

Fig. 7.2 Comparison of BGP local and global views

the latter a more restricted, partial view of the interconnection map and seemingly worse performance. Another aspect to be considered regarding the global view is that an off-line obtained BGP map may be rendered inadequate due to churn whereas one obtained through on-line aggregation of multiple BGP tables may be a technically challenging task. Even more so as some domains may be reluctant to provide such information which they often deem as sensitive. By contrast, the local, on-line topology gathering mechanism requires nothing more than BGP feeds from the ITR. Additionally, there is no need for a communication protocol between the MS and the overlay members for the conveying of BGP reachability information.

To compare the two alternatives, we take as global view the Internet-like topology we use in our evaluation and as local view the routing table of the ITR. More details on how we obtained the dataset can be found in Section 7.4.2. Using these two topologies we computed the relative AS path length increase of the local view and the distribution of the path lengths for both. Results are depicted in Figure 7.2. If we focus on Figure 7.2a, we see that 99% of the local view paths are at most 2 hops longer than in the global view and about 20% have an identical length. On average, path length in the local view increases only about 1.1 hops. This is also illustrated in Figure 7.2b where we can also note that, save for the average 1 hop increase, the distributions of hop lengths are similar. Given the relatively small difference, we are lead to conclude that the local view presents a reasonably accurate description of the topology.

Due to its relatively good accuracy and, more importantly, due to the implementation simplicity, we opted in our experiments for the BGP topology discovery mechanism based on local information. The metric it provides, inter AS hops, in combination with the optimization algorithm results in a degree-constrained shortest AS path tree optimization strategy. For brevity, we shall refer to it as *bgp*. Should there be interest in obtaining a minimum AS hop cost tree, at the expense of larger number of hops to the source, a degree-constrained minimum spanning tree heuristic should be used.

7.3.3 Latency-based Topology Map

Inter-member latency is a metric commonly employed by application layer overlays in topology optimizations. Yet, obtaining an inter-member latency map may scale poorly with the population size and therefore its implementation may be both expensive and technically challenging. For instance, in a topology consisting of N members, a naive approach, whereby each member measures all possible peers, would require N-1 measurements per member. To prevent scaling the number of measurements with the size of potentially large overlays, a more intelligent approach for the selection of link latencies worth estimating is needed.

We avoid performing a large number of measurements and assure they are carried out in an optimized order by exploiting a mechanism similar to the one used by Banerjee et al. for the group management of NICE [20]. The solution consists in clustering nodes that are close to one another in terms of latency and limiting the inter-member measurements to just the pairs finding themselves in close proximity. The amortized cost analysis shows that the number of control plane peers (i.e., the number of peers measured) at an average member is constant $O(k)$ while in the worst case it can reach $O(k \log(N))$. Where, k is a constant limiting the node degree and the size of the cluster and N is the number of overlay members. Even in the worst case, given that N may be in the range of thousands to tens of thousands, this is a considerable decrease from $O(N)$.

Another advantage of the centralized group management is that the latency discovery mechanism, when implemented in Lcast, has a lower per member communication overhead than in NICE, as members do not participate in a separate control plane protocol. However, LISP's extension to provide for a simple mechanism to convey latency measurements between ETRs and the MS is required. Since ETRs check the liveness of the locators associated to cached mappings with RLOC-probing, the extension requires just the implementation of a message, similar to a `Map-Reply` for reporting the RTT estimates.

The combination of the latency topology discovery protocol and the optimization algorithm results in an optimization strategy we further refer to as *lat*.

7.4 Evaluation Methodology

To compare the performance of the overlay optimization strategies proposed in the previous section, we implemented an event-based simulator. In what follows we describe the simulator's components and our evaluation methodology. We start by describing the datasets and the procedure followed to build an Internet-like inter-domain topology. Subsequently, we present the methodology used to generate traces that emulate realistic client behavior and explain our simulation setup. We conclude the section with a brief presentation of the metrics used to evaluate overlay performance.

7.4.1 Simulation Methodology

Our experimental evaluation simulates a set of $140k$ end-hosts spread in $3k$ autonomous systems, watching a live stream over an Internet-like topology with the help of Lcast. For this purpose, we developed a event-base simulator capable of handling large scale Lcast overlays and several optimization strategies. This resulted in the partial implementation of Map-Server and ETR functionality.

In all the experiments we employ as content source an arbitrary autonomous system as we observed that the choice does not influence the results. Client ASes participate in the overlay with one ETR and their decision to subscribe or unsubscribe is triggered by the activity of intra-domain users they serve. To simulate various types of user behavior, the latter is provided as input to the simulator in the form of trace files that log end-host join and leave events. ETR subscriptions are not optimized, but done at the first randomly found free position in the distribution tree not to bias the effect of the optimization strategies and are always based on unicast connections. The distribution tree is optimized by the MS periodically (10 min) or if more than a third of the members sustain an increase of the served client set above 10 or drop to 1, join or leave the overlay. These values were chosen to balance the computation costs and the overlay's content delivery efficiency. Additionally, to evaluate the influence of tree optimizations on communication overhead, we require that all member departures trigger the optimization of the affected sub-trees instead of only having the affected children reperform the subscription procedure. We detail the optimization algorithm, the Internet-like topology and the three traces that describe user behavior in the next sections.

The performance of each optimization strategy is evaluated by running simulations with respect to the client trace and fan-out values, which we vary between 2 and 10 to understand how replication factors influence performance. For each such simulation run, we sample and store for analysis overlay state once per minute and control traffic overhead once per second.

Finally, to better gauge the performance of *bgp* and *lat*, we also define and evaluate a very simple overlay management strategy that does not perform topology discovery or tree optimizations. In this scenario, we further refer to as *rnd*, members join at random positions in the distribution tree and member departures always require the affected children to repeat the subscription procedure.

7.4.2 Internet Inter-Domain Topology

To obtain a realistic global inter-domain topology we aggregated datasets that estimate how autonomous systems interconnect from multiple sources: iPlane [92], RouteViews [134], CAIDA [74] and RIPE [115]. All the data used is from April 2011. The dataset lacks link specific BGP policy information that could transform part of the AS graph's edges in arcs (directed links). Most affected by this assumption are the links between customers and their upstream providers and peering links between stub ASes. The first type may not be used by upstream providers for transiting traffic to destinations other than those found in their clients' network. The second type may not be used to transit traffic to destinations outside a peer's network. Since, Lcast only replicates traffic between stub domain border routers, the two types of links may be misused only when a non-member stub domain transits traffic to or from an Lcast member. However, stub domains generally have a much less diverse connectivity than transit domains thereby, such situations should be a seldom occurrence.

For the resulting inter-AS topology, we observed that the log-log plot for the complementary cumulative distribution function (CCDF) of the AS-node degree follows a straight line, a property found in power law distributions. Accordingly, as previously shown in [53] and [48], the Internet AS topology is a scale-free network with power law node degree distribution. Further, the average path length in our topology is 3.5 or 5.4% lower than the 3.7 observed [72] in the Internet. These two results corroborate our claim that the aggregate topology has properties similar to those of Internet's AS graph.

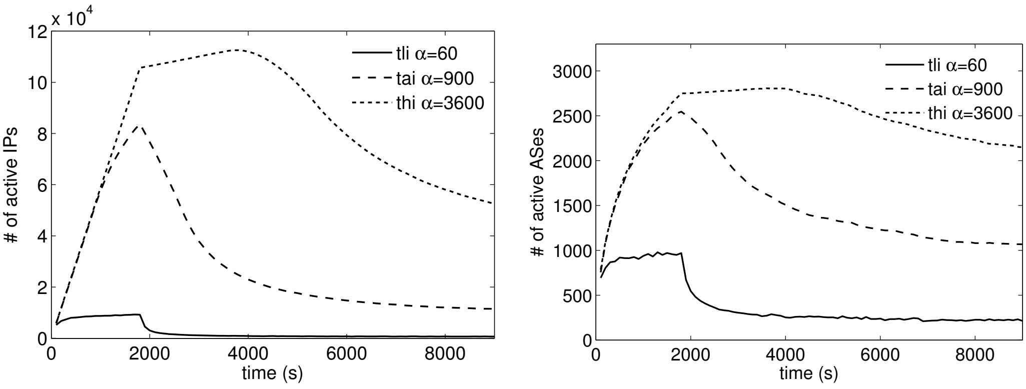
For estimating inter-AS latency, we made use of iPlane's [92] proven latency prediction abilities for IP pairs [93]. Because we needed to estimate the latency between domain border routers we had to elect for all participating ASes a representant. We did so by using iPlane's estimations that associate points of presence (PoP) to ASes 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 a latency estimator based on geographical distance described in [78].

7.4.3 The Client Traces

To ensure a thorough evaluation of the optimization strategies, we make use of client traces that emulate complementary types of user behavior. The domains that participate in the overlay and their respective number of clients were obtained from a passive distributed capture of several P2P TV channels whereas the client churn was modeled in accordance to recent results in the field. We detail both efforts in what follows.

SopCast [5] is one of the P2P TV applications frequently used for streaming of live sports events. Wanting to model client distribution for large events of global, or at least wide-spread interest, we captured the traffic pertaining to several SopCast overlays during an 2011 UEFA Champions League semifinal. To this end, we used 2 vantage points in USA, 5 in Europe and 2 in Asia, spanning a total of 6 countries. 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. As a result, the traces finally contained more than 145k unique IPs spread in over 3.8k ASes. Out of them, in our simulations we used 3k ASes for which we could compute pairwise latency estimates. More information about the traces we captured and their properties can be found in [33] and in Appendix A.

In spite of the large 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 modeling of client churn. As shown by several studies [65, 129, 131, 135, 136], it is generally accepted that client arrival process, at least



(a) Clients (b) ASes

Fig. 7.3 Number of active clients and ASes for the generated traces with time.

for periods spanning dozens of minutes, can be modeled by a Poisson process. Furthermore, Sripanidkulchai et al. observed in [131], after analyzing 3 months worth of Akamai logs, 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 in [135] who also noted that for *long* streams client inter-arrivals can be modeled through a Pareto or a piecewise stationary Poisson process.

For client session lengths however, consensus could not be found. Thus, depending on stream length or the type of system being analyzed by either paper, they may follow different distributions. Still, with the exception of [136], there seems to be an agreement that sessions should have lengths distributed according to a power law but 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 behavior, we generated 3 traces with complementary properties. The goal was to model a short event, spanning 2h 30min, with a piece-wise Poisson arrival process but with different shapes for the session length distributions. In order to capture the flash crowd effect we required that 80% of the clients join during the first 30min, and the rest spread over the time left. For the session lengths we used a Pareto distribution with a shape parameter of 1.5 and a scale parameter, denoted α, that took the values 1min, 15min and 1h in order to emulate low, average and respectively high client interest in the streamed content. Figure 7.3 depicts the evolution of the number of active clients and ASes with time for the tree traces. For brevity, we shall refer to them as *tli*, *tai* and *thi*, respectively, as a shorthand of the client interest modeled in each case.

The first trace represents the worst case scenario from overlay stability perspective, the reason being that clients leave the stream soon after joining, that is, they perform what's known as *channel surfing*. As the total number of active clients both in the overlay and within each AS is the lowest, the trace is also good for evaluating the low bound of the overlay's efficiency. Conversely, with *thi* we can assess Lcast's efficiency and ability to optimize overlays with large number of clients in low churn conditions. In light of the generation procedure, each of the three traces should be a good approximation of specific but realistic client behavior when part of a multicast group. However, if considered together, they should provide a good coverage of all practically encountered behavior. The client traces along with the SopCast ones can be found at `http://www.cba.upc.edu/lcast`.

7.4.4 Metrics

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

- **latency stretch**: this metric measures a client's relative gain in latency to the stream's source when compared to the unicast one way delay between the source and the client. While a value lower than 1 indicates that Lcast delivers packets faster than unicast, a value larger than 1 does not necessarily imply a large absolute delay.

- **hop stretch**: it measures a client's relative gain in number of AS hops to the stream's source when compared to the number of hops on the unicast path linking the two.

- **tree cost**: is a metric we define to quantify Lcast's efficiency in using underlying network resources. It is computed as the ratio of the number of AS hops crossed for the delivery of one packet to all end-host clients to the number of AS hops crossed when using unicast for the same purpose:

$$\frac{\sum\limits_{v \in V} hop(v, p_v)}{\sum\limits_{v \in V} c(v)\, hop(v, root)} \tag{7.1}$$

  where V is the set of all member routers, $hop(\cdot)$ is a function that returns the number of AS hops between two routers, p_v is the overlay parent for v and $c(\cdot)$

is a function that returns the number of end-hosts served by a member router. Tree cost is lower than 1 if the overlay is more efficient than unicast delivery.

- **control traffic overhead**: to evaluate the scalability of Lcast's control plane we use a set of metrics that measure the number of messages exchanged by the MS with the tree members for the purpose of creating a tree, maintaining tree integrity, tree optimizations and topology discovery.

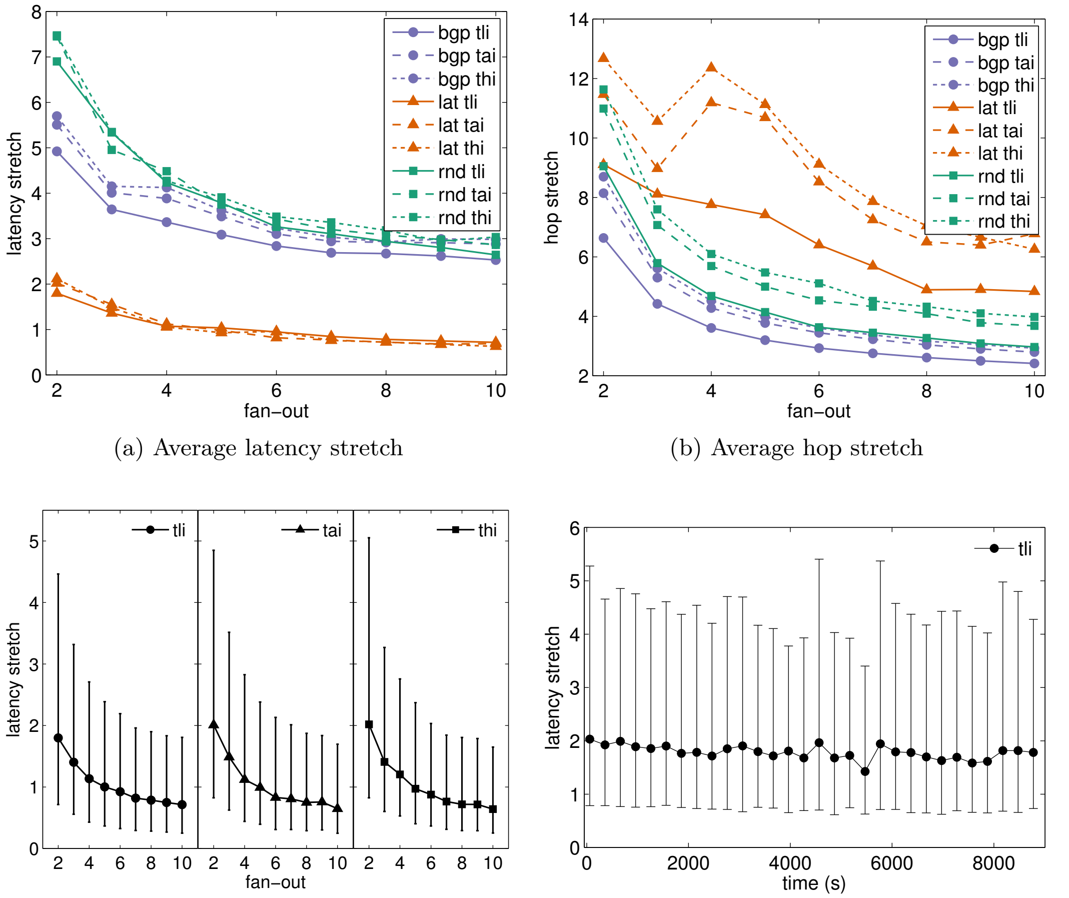
7.5 Results and Discussion

In this section we discuss the experimental evaluation results of the tree optimization strategies previously presented and afterwards look at how Lcast compares to Island Multicast, a P2P architecture that also exploits intra-domain multicast deployments.

7.5.1 Latency and Hop Stretch

Figure 7.4a presents the average latency stretch for the three optimization strategies versus member fan-out. One of the first things to be noticed is the clustering of the results based on optimization strategy. On the one hand, this suggests their independence from client churn and thereby also from the size of the overlay. On the other, it indicates that the choice of the topological information to be used greatly influences latency stretch. In fact, if we split results with respect to optimization strategies, we see that *lat* outperforms *bgp* and *rnd* by a significant margin and it is generally able to ensures an average latency stretch lower than 2. Moreover, Figure 7.4c shows that not only the average is small but also the bounds are tight. That is, the $95th$ percentile of the latency stretch is smaller than 5 and if we focus on high fan-outs, 95% of the overlay members receive multicast content with a delay only 2 times larger than that of unicast. Since for traces like *thi* the overlay can reach up to 3000 active members, these results confirm the efficiency of the optimization algorithm and therefore Lcast's ability to deliver latency constrained multicast content. Additionally, due to the independence from churn, noticeable at least for *lat* and *rnd*, the results also indicate Lcast's adaptability to dynamic overlay conditions. This point is further supported by the observation that latency stretch is rather stable with time, even when client interest is low, as depicted in Figure 7.4d.

As a general trend, the latency stretch values decrease with fan-out, but more importantly, increasing fan-out above 6 yields little benefit. Then, even if left un-



(a) Average latency stretch

(b) Average hop stretch

(c) Average latency stretch of *lat* for the three traces with 95% confidence intervals

(d) Latency stretch for *lat* with fan-out 2 and high client churn with 95% confidence intervals

Fig. 7.4 Latency stretch and hop stretch results

constrained, fan-out should only seldom exceed 10 for a subset of the members. Nevertheless, such a high value might prove unacceptable in practice, therefore our decision to constrain fan-out is warranted.

A rather surprising result is the effectiveness of *rnd* relative to *bgp*. Not only are AS hops a bad estimate for latency, but using them when optimizing a distribution tree with high member fan-out yields only marginally better results than building a random tree. This, of course, questions the practicality of *bgp*. But, despite not being appropriate for minimizing latency, it will be seen later that *bgp* should be used when the aim is to minimize underlay network resource use. On the contrary, given its reasonable performance, *rnd* could be used as a backup, no overhead optimization strategy for *lat*.

Average hop stretch results are shown in Figure 7.4b. The clustering with respect to optimization strategy is still noticeable however, in this case we see a clearer influence of overlay size. All results improve again with fan-out, but given the dependency of hop stretch on tree depth, fan-out here has a more important contribution. Both observations can be explained by Lcast's inability to use for replication the much better connected routers pertaining to transit domains, a typical problem for overlays. In addition, the situation is further worsened in Lcast's case by low router out degrees which prevent the optimization algorithm from taking advantage of the better connected edge routers.

Nevertheless, we see that *bgp* manages to keep hop stretch relatively low, but only once the fan-out exceeds 6. Notably, if replication factors are left unconstrained, hop stretch can drop under 3, even for large member sets and despite the imperfect BGP map used. Then, given that *bgp*'s results are a lower bound for hop stretch, it follows that Lcast will inevitably build high hop stretch paths when large number of members are joined and fan-out is kept low. The biggest disadvantage to such paths is their higher chance of being unstable as length increases, even if the inter-domain links that make them up are generally more stable than links in edge networks. So, if overlay stability is a concern, one the one hand, it could be achieved by relaxing the fan-out constraint for routers high in the distribution tree (i.e., close to the ITR), as a compensation for their lower latency stretch. On the other, it could be ensured solely through Lcast mechanisms at the cost of higher communication overhead.

Like for latency stretch, *rnd* has hop stretch close to that of *bgp* and actually performs better than *lat*. We explain the result by the fact that *rnd* generally tries to build k-ary complete (i.e. low depth) trees in a topology with a low average AS path length. In fact, since there are few inter-AS paths that could penalize overlay efficiency, *bgp*'s margins over *rnd* are not very large. In contrast, *lat* builds trees with low latency paths at the cost of higher tree depth and therefore higher overlay path lengths.

It can be noticed that for high fan-out values the latency stretch of *lat* (see Figure 7.4c) is lower than 1. So, the use of Lcast with *lat* as optimization strategy should result in lower average latency stretch than IP-multicast implemented with existing BGP policies. This is due to BGP's limited decision process whereby the best path is usually computed based only on AS hop distance, independent of latency. Such artifacts have been termed *latency triangle inequality violations* [91, 125] and their effect is that a subset of the BGP selected paths possess higher latency than others which, despite looking like detours due to increased number of hops, have low

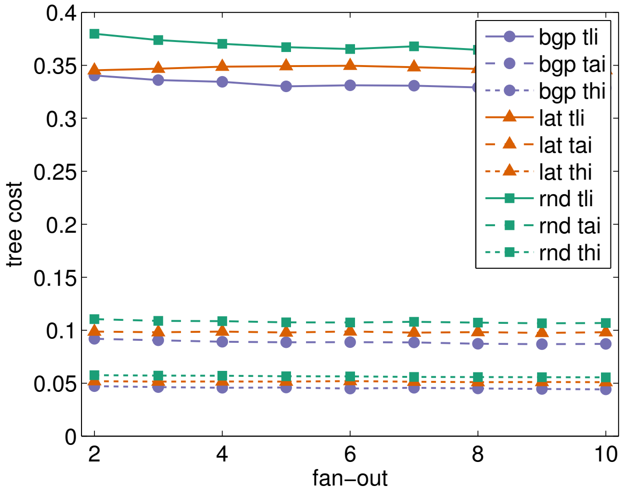


Fig. 7.5 Tree cost with respect to member fan-out.

aggregated latency. As a result, it may happen that overlays offer at times lower inter-member latencies than the underlying unicast topology. Both [91, 125] have identified lower latency paths than the BGP selected ones for more than 20% of the pairs in their datasets. Then,

7.5.2 Tree Cost

Figure 7.5 depicts the results for tree cost. Independent of churn, results show *rnd* and *bgp* as the the worst and respectively best performer, although differences are quite small. This corroborates our assumption that *rnd*'s previous results benefit from the low length AS paths and not from efficient opportunistic distribution trees. In contrast, we see that *lat*'s high hop stretch does not result in a high tree cost. Therefore, it follows from (7.1) that the long overlay paths it builds are reused by large client sets. So, in unstable network conditions, using BGP information to constrain hop stretch, i.e., building a hybrid *lat* and *bgp* optimization strategy, may be advisable. Notice however that this is not characteristic to *lat* but to all overlays that have as objective the minimization of latency alone.

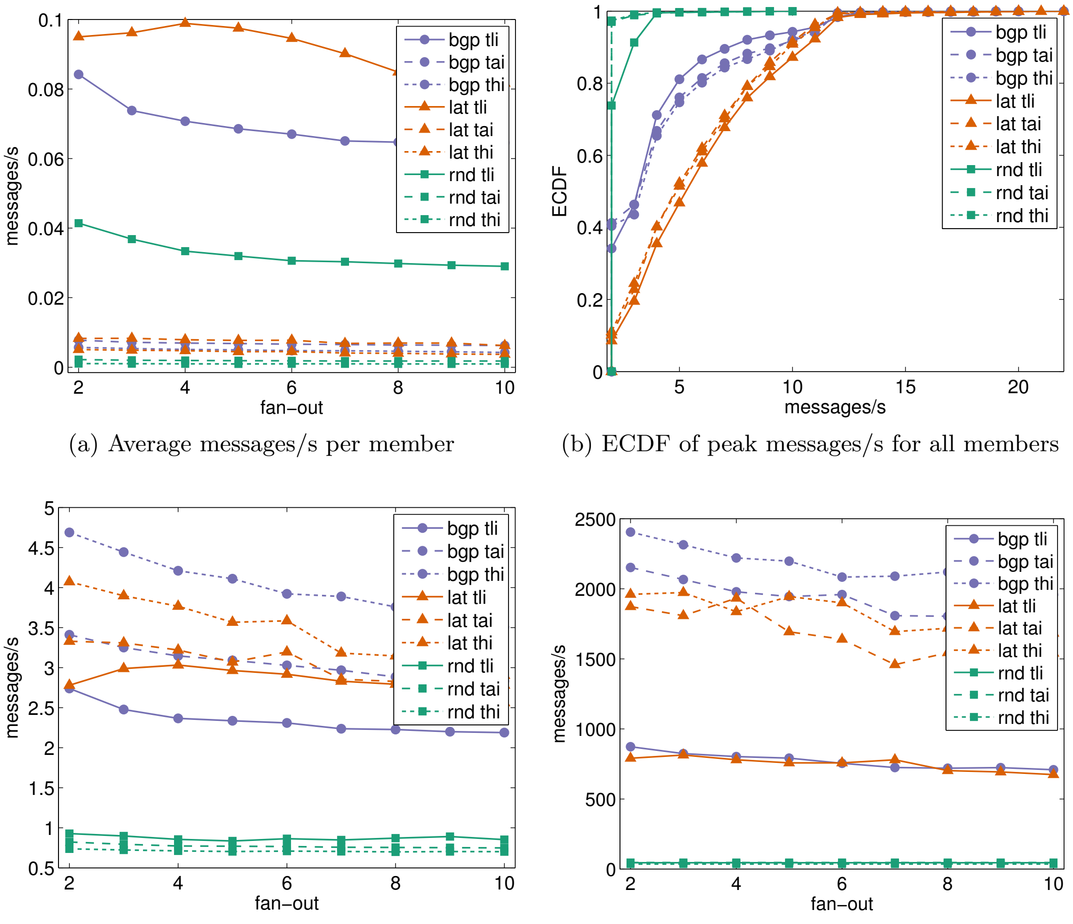
The highest tree cost, registered by *rnd*, is under 0.4 for high client churn. Hence, even in the worst case, when clients are sparsely distributed within domains, Lcast requires less than half of the unicast case AS hops to deliver the content to all clients. But more importantly, for average and low client churn our solution is more than one order of magnitude more efficient than unicast.

It is also interesting to observe in Figure 7.5 that tree cost is independent of fan-out and barely affected by optimization strategy. Or otherwise stated, the metric is independent of the shape of the distribution tree. This is in agreement with the findings of Chalmers and Almeroth, who have previously shown that multicast efficiency, if defined similarly to our tree cost, only depends on the number of clients in the system [30]. Note though that, here, the ratio between the average tree costs for different traces is not in direct relation with their ratio of the number of clients in the overlay. This because, as it may be seen in Figure 7.3, the ratio changes over a simulation run. We explain the slight dependence on optimization strategy by the inefficient branching done by both *rnd* and *lat*, which does not follow underlay topology, i.e., an AS path may be crossed several times. For *bgp* this topological incongruence is minimized and considering the result above, its tree cost should be close to optimal, despite its use of an algorithm that minimizes average client AS hops to the source, not overall bandwidth usage. Moreover, this also implies that Lcast with *bgp* should in general have a slightly smaller tree cost than other architectures focused on minimizing latency stretch.

7.5.3 Control Traffic Overhead

We split control traffic overhead in management overhead, needed for group management due to peer churn, and active topology discovery overhead needed to perform and convey topology measurements. The only optimization strategy to employ active topology discovery is *lat*.

Figure 7.6a show the results for average management overhead from member perspective. The highest average rate is less that 0.11 messages/s and indicates that members seldom exchange messages with their peers or the MS. It would appear that higher churn results in higher rates but this can be attributed to the low number of overlay members. That is, for average and high client interest the overlay contains many members that seldom exchange messages so the average is kept very low. Figure 7.6b illustrates for each optimization strategy the Empirical Cumulative Distribution Function (ECDF) of the peak messages/s per member. For each member, the peak is computed as the maximum over all fan-outs. It may be seen that, independent of churn, members have the highest instantaneous overhead for *lat* while for *rnd* the lowest. In particular, for the former 99% of the members have peaks under 13 messages/s while for the latter the peaks are under 4 messages/s. Even in the worst recorded case, a member does not exceed 22 messages/s. We



(a) Average messages/s per member

(b) ECDF of peak messages/s for all members

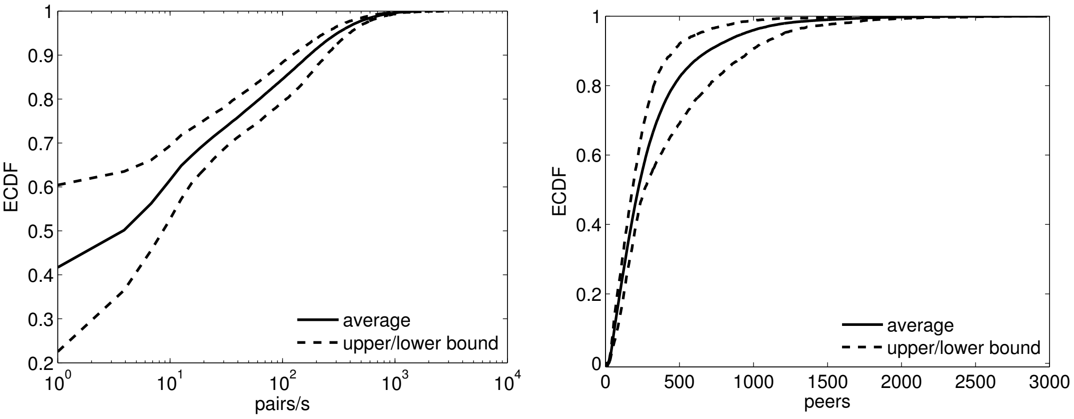
(c) Average messages/s per MS

(d) Peak messages/s for MS

Fig. 7.6 Control traffic overhead

can therefore conclude that both average and instantaneous member management overhead in Lcast is negligible. An explanation for the higher number of messages exchanged when using *lat* versus *bgp* is that the overlay has a higher chance of being optimized once a new member joins or when new inter-member latencies are measured.

Looking at Figure 7.6c and Figure 7.6d, we see that the MS is also exposed to low average and instantaneous group management overhead. In fact, the highest instantaneous rate registered is 2500 messages/s and the average never goes above 5 messages/s. These message rates are easily manageable by off-the-shelf hardware. As expected, overhead is considerably higher for *bgp* and *lat* than for *rnd*, and in



(a) ECDF for peer pairs measured/s

(b) ECDF for peers measured/member

Fig. 7.7 Control traffic overhead for *lat* due to active topology measurements considering all simulation runs.

the case of the former two, increases with overlay size. Surprisingly, we also see that *bgp* requires slightly higher message rates than *lat*. So, although members have higher instantaneous message rates for *lat*, the MS has higher peak rates for *bgp*. This means that in general, *bgp* is more likely to update the whole topology while *lat* often performs local optimizations. Nevertheless, Figure 7.6c shows that both are very stable due to the small average rates.

Compared to management overhead, the active topology discovery employed by *lat* requires more involvement from the MS and members. Alas, not having implemented an optimized communication protocol between MS and members, we can not provide the exact number of messages that are exchanged. However, in the worst case, the MS would need one message exchange with a member to request that it measures one of its peers and to receive the measurement result. Although, we stress that in an optimized situation it may batch multiple measurement requests in one packet. Then, considering this approximation, we can use the number of member pairs measured per second as worst case estimate of the MS's message rate. In the experiments we set k, the constant that limits the size of the cluster for our latency discovery protocol, to 5.

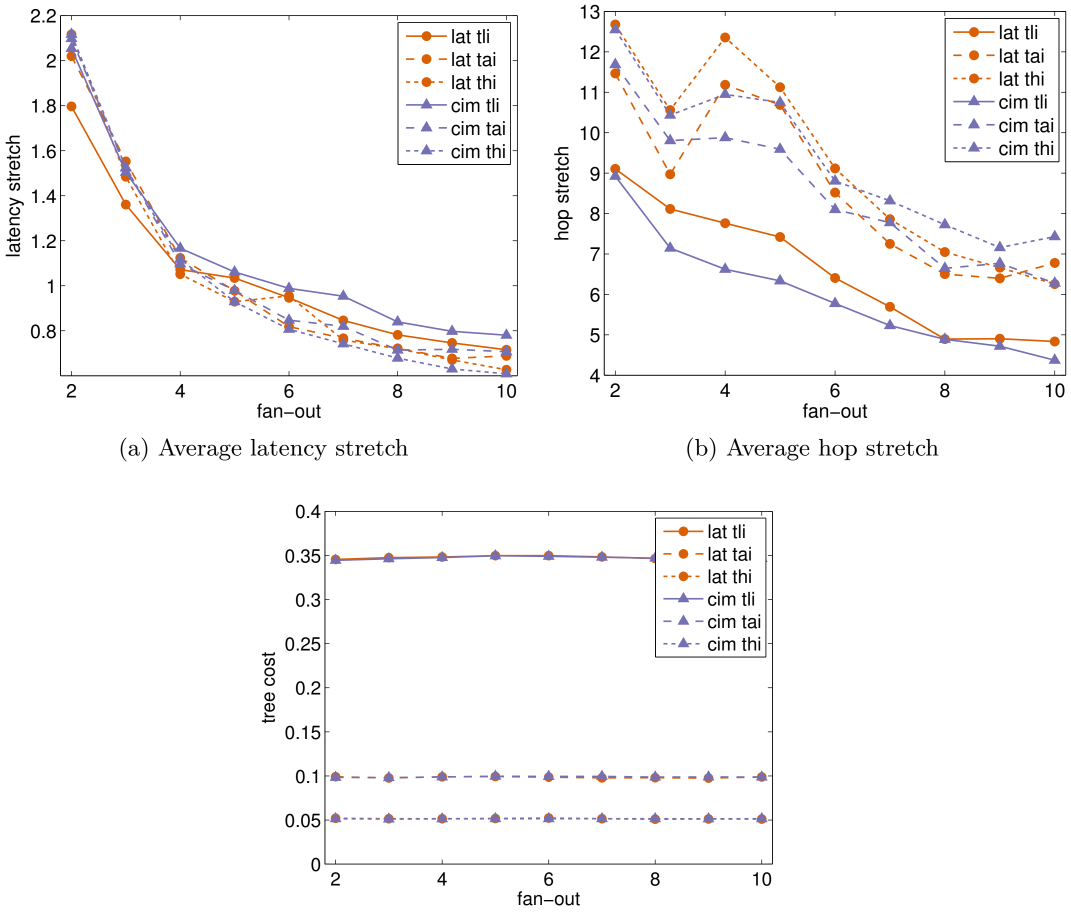
Figure 7.7a depicts the average ECDF for the number of member pair latencies measured per second, with lower and upper bounds, computed over all the *lat* simulation runs. We see that on average, during more than 41% of the time spent in a simulation, the MS does not request any measurements while in 79% to 89% of the time, less than 100 pairs are measured per second. In the worst situation, just 0.8% (or 72s) of the simulation time is spent performing more than $1k$ measurements/s. Then, typically, topology discovery overhead is negligible and in the worst case, the message rate is the same order of magnitude as peak management overhead. So, even when the two are considered together, they should still be manageable by one server. Moreover, because the MS coordinates the measurements, it is the only one affected by requests peaks. Hence, in overload situations, it may queue requests for later processing with limited or no effect for the quality of the distribution tree.

Finally, Figure 7.7b shows that in general half of the overlay members participate in less than 220 measurements and only 4% of the members participate in more than $1k$ pairwise latency estimations over the entire length of the simulation. Thereby, the topology discovery overhead from member perspective is low, despite the large number of ASes participating in the overlay.

7.5.4 Comparison with Island Multicast

Island Multicast (IM) [82] is a P2P architecture that optimizes content delivery efficiency by exploiting intra-domain multicast deployments. Similarly to Lcast, it uses unicast to connect multicast islands however unlike Lcast, the overlay is constructed with end-hosts. IM can operate either with a centralized controller (CIM) when it optimizes the distribution tree using a variant of Shi's algorithm [126] or fully distributed (DIM) when it relies on a Delaunay triangulations (DT) overlay protocol to connect the multicast islands. Although less scalable, CIM is comparable to DIM in terms of latency stretch for large sessions, and, as shown in [90], DT is actually less efficient than overlays that take into consideration network layer latency. We therefore focus in what follows on the comparison between *lat* and CIM.

Both Lcast and CIM can control member fan-outs to limit processing and bandwidth overhead. Even though CIM can individually constrain end-host fan-outs, we perform the comparison considering fixed domain, or multicast island, out degrees and suppose that the replication load is distributed optimally over a domain's end-host population. Arguably, if fan-outs are very large, this gives a scaling advantage to CIM since all processing associated to packet replication is supported by border routers in Lcast, despite the bandwidth restrictions that must be met at border routers being the same for both solutions. Nonetheless, a disadvantage to end-host replication is that it increases tree latencies as packets need to travel intra-domain and accumulate additional processing delays. Still, for simplicity, in our simulations



(a) Average latency stretch

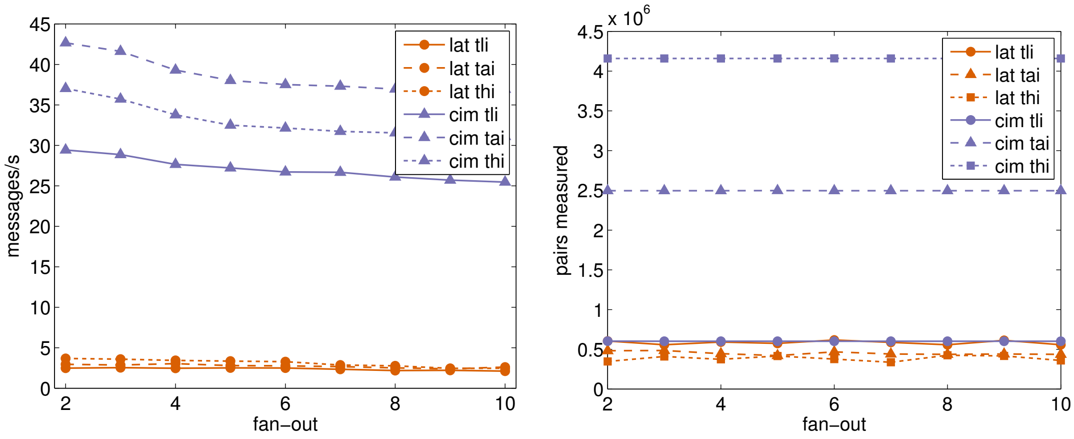
(b) Average hop stretch

(c) Tree cost with respect to member fan-out.

Fig. 7.8 Latency stretch, hop stretch and tree cost comparison for *lat* and CIM.

we consider intra-domain propagation and end-host processing times as negligible to inter border router latencies.

Figure 7.8 depicts the latency stretch, hop stretch and tree cost for *lat* and CIM. Since both use the same optimization algorithm the three metrics have similar values. Particularly, in the case of latency stretch, for low and average client interest, Lcast offers slightly better performance. For *thi* however, CIM's random latency discovery algorithm benefits from end-host stability and discovers sufficient link latencies to ensure a more efficient distribution tree for high fan-outs. The fact that *lat* and CIM generate similar but not identical trees, due to their distinct approaches to latency discovery, is also supported by the average hop stretch results depicted in Figure 7.8b. As expected, tree cost results are identical, given their dependence on



(a) Average messages/s for MS

(b) Number of AS pairs measured

Fig. 7.9 Control traffic and topology discovery overhead comparison between *lat* and CIM.

the number of clients in the system.

An important difference between the two solution is that CIM's controller must communicate with all end-hosts participating in the overlay. As a consequence, the good performance seen in the previous comparisons comes at cost of much higher communication overhead due to peer churn. Figure 7.9a illustrates this point. We see that in all cases CIM requires almost one order of magnitude more messages per second than *lat* for overlay management.

In addition, one of CIM's least efficient or scalable mechanisms is the random topology discovery algorithm. This procedure is overseen by the overlay coordinator and consists of end-hosts measuring their latency to 5 random peers to discover link latencies or failed hosts. However, a clear downside to it is that if good quality distribution trees are required, measurements must be performed at intervals inversely proportional to the overlay size, that is, more often as the overlay size grows. In our simulations we set a lower bound of 20s and optimize the peer selection to ensure that inter-domain latencies are measured only once and intra-domain ones never. Figure 7.9b shows the total number of pairs measured and contrasts it with that of *lat*. For small overlays and high churn, the two architectures measure approximately the same number of peers however, despite our optimizations, as client churn diminishes and the overlay sizes grow, CIM requires more measurements whereas *lat* requires less. In the extreme case, for *thi*, CIM measures about an order of magnitude more peers. Considering that latency stretch is generally on par, this confirms again the ability of *lat* to efficiently manage link measurements.

Overall, the comparison shows that, if evaluated under the same constraints, *lat* and CIM optimized overlays have similar properties, save for communication overhead, which is considerably lower in *lat*'s case. This confirms *lat*'s efficient design and ability to accommodate large overlay sizes. It is also worth noting that CIM's impracticality for large overlays makes Lcast, to the best of our knowledge, the only solution capable of ensuring on-line swapping of overlay optimization algorithms. However we would like to stress that, due to its need for infrastructure support and provided feature set, Lcast is of greater interest for network operators and thereby complementary to existing P2P streaming solutions.

7.6 Related Work

As a long standing academic and commercial research challenge, single-source live media streaming benefits from copious amounts of related literature. Consequently, we restrict the discussion to a limited set of solutions and mainly focus on those that bear similarities to Lcast.

Multicast functionality that enables live media streaming was originally offered as a network-layer service. But in light of IP multicast's lack of inter-domain deployment, network layer solutions have turned into architectures that leverage isolated multicast deployments. One notable example is MBone [52], a virtual network designed to connect multicast islands by means of static unicast tunnels. Although it was first to support distribution of content to users spread in multiple domains, it proved hard to extend since setting up tunnels involved manual configuration. AMT [25] circumvents this limitation by providing mechanisms for automatizing the tunnel setup process with the help of dedicated servers (relays and gateways) placed in source and destination domains. However, AMT does not support dynamic reconfiguration of the tunnels, i.e., of the inter-domain distribution topology. Therefore, unlike Lcast, it is not able to adapt to changing network conditions, client churn or to limit replication overhead.

A large set of application layer solutions, including NICE [20], Narada [71], OMNI [19], ZIGZAG [133] and Scribe [28], have been proposed by academia in the last decade. Out of them, OMNI [19] is the closest in spirit to our proposal. It requires service providers to deploy a set of proxying nodes that self organize in an overlay and forward traffic to subscribed clients. The optimizing algorithm employed is a distributed instance of the one we use and the metric considered is latency. In contrast, Lcast works at network-layer and is supposed to be deployed, at no additional cost, together with LISP. Moreover, Lcast's logically centralized control plane allows easy deployment of new optimization algorithms without requiring router changes.

Apart from the academic solutions, a large array of commercial ALM architectures like SopCast [5], PPLive [4], CoolStreaming [142] or UUSee [6] are widely used for Internet content streaming. Being closed source their architectures are not completely understood, nevertheless their performance has often been the subject [65, 129, 135, 136] of academic scrutiny. The results have shown significant limitations of these architectures in scaling user quality of experience with the increase of client population. Lcast, being an extension of LISP, operates on domain border routers and thus builds an overlay topology that is not directly exposed to client churn. Furthermore, through design it avoids imposing bandwidth strain on overlay members and could assure certain performance bounds.

Another approach to delivering inter-domain multicast is to connect islands of multicast enabled end-hosts by means of application layer overlays. Two of the solutions to follow this design guideline are Universal Multicast [139] and Island Multicast [82]. They are similar to Lcast in their use of existing multicast deployments for intra-island content delivery and of tunnels to connect multicast islands. However, they ensure inter-island multicast delivery by building and optimizing overlays consisting of end-hosts. This gives rise to three fundamental differences. First, Lcast does not require changes to end-hosts as the inter-domain router overlay seamlessly interfaces with local multicast. Second, Lcast should use more efficiently the underlying intra-domain network since packet replication is always performed in domain border routers and therefore packets avoid traveling intra-domain prior to being replicated and forwarded to hosts in foreign domains. As a downside to this, when fan-out values are large, the routers have a higher processing overhead however, as shown by our experiments, they need not be large for good performance. Finally, all solutions offer the option to constrain fan-out values but Lcast offers control to operators who have a vested interest in the efficiency of the overlay. That is, router out degrees should be generally limited to protect routers from saturating their interfaces and to ensure fairness in distributing the replication responsibilities. We then believe that providing the ability to configure fan-out actually makes Lcast better suited for operational deployments than its P2P counterparts.

We previously proposed CoreCast [80], a LISP inspired inter-domain streaming architecture where source and client routers operate according to a client-server model. Both CoreCast and Lcast are based on LISP protocol mechanisms but they have quite different approaches to delivering inter-domain traffic. In this sense, CoreCast aims to diminish inter-domain bandwidth use when compared to P2P live streaming systems while Lcast aims to provide a scalable and easily reconfigurable offloading mechanism for the source LISP router. Finally, it is worth mentioning that LISP-Multicast [56] integrates IP multicast functionality into LISP. Naturally, it inherits all the properties of traditional network layer multicast however it also requires core router support for inter-domain use, thus making it unfeasible for a wide-scale deployment.

7.7 Chapter Summary

Our goal with Lcast was to devise an inter-domain multicast framework that, besides possessing a low deployment cost, is also easily configurable and scalable. The former requirement was fulfilled by using just LISP enabled domain border routers to form an inter-domain overlay, without requiring any further support or changes in the Internet's core. But, equally important, by exposing the service to the clients by means of existing intra-domain multicast protocols, and by limiting the router overlay fan-out (replication factor) to low values.

Configurability was ensured by two design decisions: first, the separation of the control and data-planes and second, the centralization of the control-plane functions in the MS. Member participation in the data-plane is conditioned only by the implementation of LISP functionality. However, member presence in the control plane is not required since all optimization functions are centralized in the MS. As a result, operators may switch between tree optimization algorithms easily, even on-line, assuring fast (re)configuration of the overlay's topology to meet operational performance requirements.

The isolation through design between local-domain and inter-domain multicast allows the separation of the overlay's router members from the churn specific to client end-hosts and thus relieves the architecture's control plane from the inherent overhead. This ensures the scaling of the architecture with the number of end-hosts however, the scaling with the number of member domains is attained through proper data and control plane design.

We evaluated three possible overlay management strategies for low latency content delivery and inferred that they are all fit for optimizing large overlays. Several conclusions can be drawn from the analysis. We saw control overhead is manageable by a single server, independent of client churn and even when active topology discovery is employed. Client churn, generally, slightly influences performance but it does increase management overhead. Another very encouraging result is that Lcast's performance does not depend on large fan-outs and in fact, fan-outs larger that 6 offer limited benefits. Finally, we saw that Lcast can be used to minimize various metrics and its performance is comparable to other ALM solutions. Notably, when used with *lat*, it can deliver content with a very low, unicast like, latency in exchange for increased but still manageable control overhead.

Chapter 8

LISP Multi Protocol Switching

8.1 Motivation

As exposed in [99] and [107], the growth of the DFZ routing table has detrimental effects on the operational costs of Internet Service Providers (ISPs) in the current operating environment. Driving costs up is the increased technical complexity required for the management of large tables. For instance, if scaling a router's Routing Information Base (RIB) is assured by the commonly accepted *Moore's Law*, the same can not be said about the Forwarding Information Base (FIB) table. The former is generally stored in cheap, mass produced, control plane memory whereas the latter is stored in much faster but also more expensive and difficult to scale line card memory. More technological limitations are discussed in [99]. Overall, they translate to increased router prices and, in the long run, due to accelerated table growth, to shorter router life spans.

In this chapter we propose to complement the traditional LISP's inter-domain use with a new deployment case restricted to the scope of an AS. Similar to the inter-domain location-identity dichotomy, in intra-domain context there's a distinction to be made between an IGP-external destination prefix and the location of the points whereby it could be reached. For instance, all IP routers in an AS's backbone are required to carry BGP routes although no BGP decision is taken within the domain. This needlessly exposes routers to external routes when information about egress points would suffice.

The goal of our work is to devise a mechanism that reduces the size of the routing tables in IGP backbone routers and enables advanced intra-domain traffic engineering. To this end, we propose the use of LISP's tunneling ability to obtain a BGP-free core but also as a mechanism to control the points through which packets egress a domain. In our solution, border routers select the local egress points for transiting packets towards which they tunnel the datagrams by means of encapsulation.

Thinking in a swift deployment, we propose to reuse existing iBGP infrastructure as a mapping system and require just a mild upgrade to enable LISP functionality. The resulting mapping-system *pushes* bindings to tunneling routers and therefore ensures no mapping misses and update propagation times no worse than those in current networks. Additionally, for traffic engineering and resilience purposes, a router and router interface addressing scheme is proposed.

The architecture we propose bears similarities with networks that jointly deploy MPLS and BGP [113, 117]. However, following the lead of [98] we advance our architecture as an IP-routing based alternative. In [98] Metz et al. express concerns that MPLS might possess a control-plane complexity factor and argue that IP mechanisms might be equally suited at performing MPLS functions. Furthermore, MPLS has a constrained footprint, and cannot be natively forwarded between disjoint networks, whereas IP is ubiquitous and easily supports coordination of disjoint sub-domains. In homage to MPLS and because of the vague similarity between MPLS and our architecture, we named our proposal *LISP Multi Protocol Switching*, or LISP-MPS.

The remainder of this chapter is organized as follow. First, we provide necessary background current routing practices in Internet Service Providers (ISP) in Section 8.2. Then, we present the details of our LISP-MPS architecture that relies on LISP encapsulation and an iBGP control plane in Section 8.3. Further, we discuss the added value of LISP-MPS in Section 8.4 and evaluate its benefits in Section 8.5. We finally contrast LISP-MPS to the related work in Section 8.6 and conclude this chapter in Section 8.7.

8.2 ISP Routing

All along this chapter, we use the following taxonomy that splits a domain's routers in three categories: *i*) AS Border Routers (ASBRs), routers found at the border with other ASes, *ii*) Customer Border Routers (CBR), routers that connect local customer networks to the backbone and *iii*) Backbone Routers (BBR), all AS core routers not ASBRs or CBRs. We may refer to the first two simply as edge or Border Routers (BR).

In what follows we shortly review some of the mechanisms related to intra-domain routing and expose several of their limitations. The presentation is based on the assumption that the intra-domain and inter-domain packet routing for an AS are assured by an Interior Gateway Protocol (IGP) and the Border Gateway Protocol (BGP), respectively.

If not otherwise stated, we consider BBRs as BGP enabled. Further, we expect that iBGP is used for the intra-domain advertisement of BGP reachability information between BRs and to BBRs. Also, we suppose that Route Reflectors (RR) [21] are are used for scaling the iBGP route redistribution.

One of the main drawbacks of such deployments is the need for BGP enabled BBRs. Normally, prior to forwarding a datagram, a router needs to determine a next-hop for the packet's destination address and subsequently an interface out on which this next-hop may be reached. Thus, all routers within a domain must be able to determine a next-hop for any globally or intra-domain routable destination address a packet may hold. Typically, this results in the routers, besides participating in the IGP, being provided DFZ reachability information by means of iBGP (see Fig. 8.1). Consequently, they all need to store two different scope routing tables and deal with their associated protocol instabilities.

Additionally, due to iBGP's design, the two routing tables are coupled in the resolution chain of an outgoing interface for non-IGP destinations. In such a scenario, the next-hop of the iBGP learned route will typically be an address not adjacent to the resolver. Instead, it could either pertain to the router advertising the route in iBGP (a local BR) or to the foreign BGP peer from which the local BR learned the route. Therefore, a second resolution is needed, of the next-hop against the IGP table, for the discovery of an interface out on which the packet can be forwarded to the next-hop. Such resolution process can be intuitively interpreted as a double mapping. First, an address is mapped to a gateway, the BGP route's next-hop, which at its turn is mapped to an IGP route learned over a local interface. From the perspective of on path backbone routers the procedure is obviously redundant as they all perform identically the first mapping, if iBGP is converged.

To avoid storing BGP routing tables in BBRs, ISPs may use MPLS for tunneling traffic between BRs. Additionally, this results in several traffic engineering benefits. First, the ability to speed up the the forwarding of traffic over a domain's backbone, optionally under QoS constraints. Second, due to MPLS's fast reroute capabilities good resilience to failures. Finally, in combination with Multipath BGP, MPLS tunneling could be used for load balancing traffic between multiple egress points (BRs), instead of just one. However, as explained in [98], MPLS is quite complex to

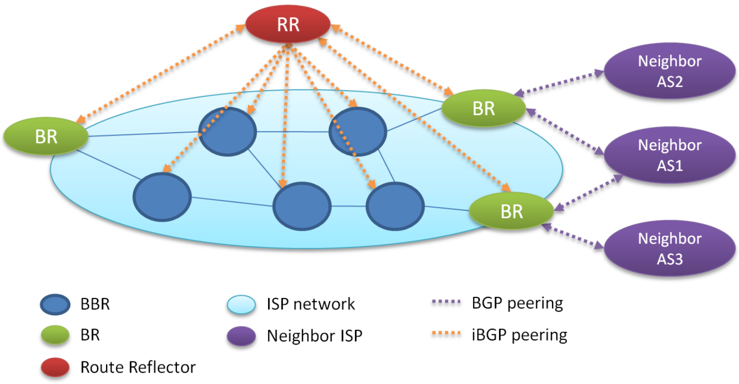


Fig. 8.1 ISP network example

manage and requires support in all backbone routers. Furthermore, its deployment is typically limited to a domain so disjoint networks are hard to interconnect.

8.3 Proposed Architecture

The driving goals of our proposal, LISP-MPS, are to *i*) devise a solution for ISPs wishing to diminish the size of the routing tables in the routers part of their backbone networks and *ii*) enable more complex intra-domain traffic engineering policies. This section presents how these could be achieved with LISP. However, the proposed architecture boasts a much larger feature set which we will expand on in Section 8.4.

8.3.1 Overview

As explained in Section 8.2, within an autonomous system, backbone routers must store BGP routes, although they can not influence the intra-domain routing of transiting packets. Furthermore, configuration of these intra-domain paths is not possible with a simple BGP-enabled core. As a result, network operators seeking a BGP-free core and intra-domain traffic engineering capabilities employ MPLS tunneling over the network's backbone.

Following the lead of Metz [98] we propose the use of LISP as a more flexible alternative to MPLS. Thus, with LISP-MPS, for a packet transiting a domain, the egress BR is chosen at the ingress BR and stored in the datagram by means of LISP encapsulation. All further intra-domain routing of the packet will be done only based on IGP information. This obviates the need for iBGP route redistribution to

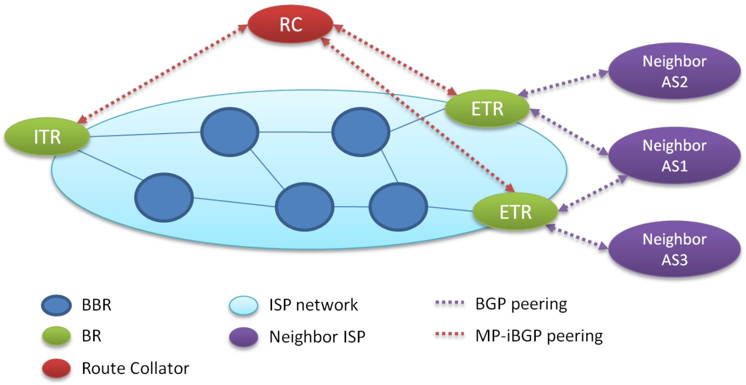


Fig. 8.2 Proposed LISP-MPS Architecture

BBRs and therefore limits the scope of the DFZ routing information to the points of interaction with neighboring domains, the ASBRs, and local customers, CBRs. Encapsulating BRs learn the mappings between external prefixes and the addresses of the BRs announcing their reachability by means of a mapping system (cf. Fig. 8.2). From traffic-engineering perspective, apart from the ability to precisely choose the traffic egress points at any ingress BR, LISP-MPS allows an operator flexibility in updating its running configuration in a timely fashion.

Henceforth, given a BGP-learned prefix, we shall refer to the IGP addresses of its iBGP originators, as Prefix Attachment Points (PAPs). By virtue of the previous definition, a PAP may be a synonym of the router itself or one of its interfaces. We further refer to the former as Router Name (RN) and to the latter as Router Interface Name (RIN). An addressing scheme for the two is suggested in Section 8.3.4.

We detail in what follows the functioning of LISP-MPS's control and data plane.

8.3.2 Control Plane

To avoid the introduction of new network equipment, we exploit the iBGP implementation in edge routers and Route Reflectors (RRs) for the distribution of mapping information. However, we do require an upgrade of the RRs, or their pairing with an additional device, in order to support LISP functionality. To avoid confusion, we call the new route reflecting network element a Route Collator (RC).

So, similarly to an RR, an RC (see Fig. 8.2) is fed by BRs all their external BGP learned routes. As added constraints, all routes must have as next-hop attribute the RN of their advertising BR and must carry information about all the PAPs of the BR. This is achieved with the help of MP-BGP [22]. On the resulting RIB the RC runs the BGP decision process and selects the best router for each external prefix. If multiple border router advertisements tie in the selection process run by the RC, instead of breaking the tie by means of IGP metric, all the PAPs of the tied BRs should be saved as viable attachment points for the considered prefix. The resulting egress point diversity enables a fine grained tuning of the traffic engineering policies for a domain. Each PAP is associated a *priority* to mark the preference of using it out of possible candidate set. Load balancing among equally preferred attachment points is performed according to another associated value, the *weight*. The Route Collator pushes, via iBGP, the selected routes to the border routers, but with neither priority nor weight information.

Apart from the iBGP updates, Route Collators also build prefix-to-PAP mappings and push them to border routers that use them to populate their map cache. For prefixes with multiple PAPs the messages also convey priority and weight information. Considering that the LISP upgrade is the only disruptive change when moving from RR to RC functionality, a more cost-effective upgrade to LISP-MPS would be to pair a LISP capable device with an RR. Therefore, iBGP responsibilities would be fulfilled by the RR whereas LISP related ones by the new server with the help of iBGP feeds shared by the RR.

Note that BGP syntax could be enhanced to carry all LISP required information. However, we avoided this solution not to correlate iBGP and LISP updates and to avoid triggering the BGP decision process on LISP updates. Still, this alternative might be worth more consideration in the future.

8.3.3 Forwarding Changes

The simplification of the forwarding in backbone routers is counter-balanced by a slight complication of data-plane operations in border routers. On receiving a packet, a BR performs a longest prefix match of the destination address in the LISP map-cache. Besides the prefix encompassing the destination address, the router learns the PAP(s) of the BR(s) announcing reachability of the matched prefix and their associated priorities and weights. Having these, the BR selects one of the attachment points and then proceeds to LISP encapsulating the datagram. The resulting datagram is forwarded across the backbone network solely by the IGP. Once the packet reaches the destination edge router, it gets decapsulated and forwarded natively to the neighboring AS.

8.3.4 Border Router Addressing

Aiming to improve intra-domain traffic engineering, we seek to provide the means to an RC to establish ITR-to-ETR paths distinct from those computed by IGP. As a result, we propose an intra-domain router and router interface naming scheme that makes use of IP prefix aggregation properties for enhanced router addressing. Each border router is allocated a local domain prefix whose reachability it must announce out all its interfaces. By convention, we consider the first address in the prefix to be the RN and attribute it to the router's loopback interface. The rest of the prefix is split in smaller blocks, each advertised out on and used to address one of the router's interfaces. Overall, a border router announces reachability for $N+1$ prefixes, where N is the number of its IGP facing interfaces. The fact that an interface can be *selected* out of those pertaining to a router and the way the router addressing is performed are beneficial for traffic engineering and failure recovery. Both are discussed in more depth in Section 8.4.

The number of additional entries to add in the FIB of each BBR is then given by:

$$\Omega = \sum_{r \in \mathbf{B}} |\mathbf{I}_r| + 1, \tag{8.1}$$

where $\mathbf{B}$ is the set of border routers and $\mathbf{I}_r$ is the set of IGP facing interfaces of a router r. Similarly, the number of entries necessary to add at a BR, for any $r \in \mathbf{B}$, is given by:

$$\Omega - (|\mathbf{I}_r| + 1) \,. \tag{8.2}$$

Note that Ω is independent of the global routing table (i.e., BGP) and only depends on the network topology (i.e., number of BRs and that of their IGP facing interfaces).

8.4 Discussion

This section presents an analysis of the benefits and drawbacks of LISP-MPS. A comparison of the routing protocols ran by routers in domains with BGP, LISP-MPS and BGP/MPLS enabled backbones is shown in Table 8.1.

8.4.1 Routing Table Reduction

One of the most important benefits of LISP-MPS is that it reduces the size of the routing tables on the backbone routers of an ISP. It does so by isolating the intra-domain routing from the inter-domain routing and by pushing all the inter-domain reachability state to the edges of the AS's topology. The result is that in our solution the BBR table sizes are bounded by the IGP size. Comparatively, in a BGP enabled backbone they grow proportionally to the number of prefixes in the DFZ. With BGP/MPLS they are also limited to the size of the IGP routing table but BBRs need to store an additional table for label switching.

8.4.2 Virtual Networks

LISP supports network virtualization with the help of an address-space extending field called *Instance-ID* (IID). Per organization Instance-IDs are used such that they can tag their packet with their IID. Consequently, several organizations can interconnect their own site-networks using the same private address space as the Instance-ID will be used to distinguish them. Obviously, all sites pertaining to an organization have the same IID and their extended address space is unique. We call such multi-site networks, where one organization controls all sites but not the network interconnecting them, *virtual networks*. To distinguish between all clients, routers at the transit-client border must install per virtual network forwarding tables and in-transit packets must carry the IID.

With LISP-MPS, at transit ingress ASBR, packets are matched (e.g., based on interface, VLAN tag) to a virtual network and thus to an established IID. The packets are subsequently encapsulated and forwarded based on the virtual network' map cache. Finally, at the egress ASBR, packets are decapsulated and forwarded to the client network that matches the conveyed IID. BRs continue to populate their RIB by means of iBGP. Additionally, all ETRs pertaining to the transit provider tag the virtual networks route advertisements (i.e., mappings) with a *RouteTarget* [117] equal to the IID. As a result, the ITRs may build per RouteTarget map-caches. In particular, this feature could be used by a service provider to offer virtual private network (VPN) services to its clients. An advantage over MPLS based VPNs is that this solution does not require the use of double encapsulation.

8.4.3 Multi Protocol Switching

LISP supports the encapsulation of a large set of protocols (e.g., IPv4, IPv6, or Ethernet Frames). Furthermore, by means of [57], can be extended to support virtually any protocol. As a result, LISP-MPS can be used to setup layer-2 VPNs or IPv6 networks independent of the underlying IGP routing protocol. Regarding IPv6 transit, besides requiring no backbone network upgrade, the solution avoids running two separate forwarding tables and thus worsening the FIB growth.

8.4.4 Flexible Routing Control

Access to IGP information should allow the collator to compute for all destination prefixes all the possible intra-domain paths. Depending on the network's complexity, an efficient distribution of traffic that minimizes metrics like link stress, bandwidth usage or latency could be implemented by configuration or with the help of an heuristic. The results may be imposed with the help of PAP priorities and weights. In this sense, traffic may be distributed among multiple PAPs with the same priority and for a specific PAP, traffic should ingress according to the weights associated to its interfaces.

8.4.5 Resilience to IGP Link and Router Failures

In the event of a BB router, or one of its interfaces failing, the IGP should generally deal with the re-routing of in-flight packets around the affected patch of network. Nevertheless, there are two IGP failure scenarios where a more complex network reaction is needed. First, if the disruption disconnects a border router's interface from the backbone network, its associated IGP route (as described in Section 8.3.2) disappears. Consequently, all packets destined to the affected interface will match the associated aggregate prefix and will be delivered to the border router on one of its still active interfaces. The failure only affects intra-domain traffic engineering policies, if any were in place, but results in no packet loss. The second failure scenario is the result of a complete isolation from the backbone network or halting of an edge router. To avoid packet black-holing we propose the use of *re-encapsulators*. These devices attract with routes covering the whole PAP address space all packets whose egress points have failed. They then re-encapsulate this traffic towards alternative border routers. If no such router exists, the packets are dropped.

To be noted that both types of failures are detected by LISP after a time threshold and subsequently the PAP used in the encapsulation is changed with a valid

Table 8.1 Comparison of Solutions (Differences in Gray)

Router	BGP Backbone	LISP-MPS	BGP/MPLS
ASBR	IGP + eBGP + iBGP		
		LISP	MPLS
CBR	IGP + iBGP		
		LISP	MPLS
BBR	IGP		
	iBGP		MPLS
RR(RC)	iBGP	MP-iBGP	MP-iBGP

one.

8.4.6 Resilience to eBGP Adjacency Failure

In this case, reachability of the prefixes advertised only through the affected adjacency will be, independent of LISP-MPS, lost. Still, the prefixes with multiple potential egress points will have their best path recomputed once the failure is advertised. Therefore, once the new routes are distributed to the BRs, the transit paths of affected prefixes switch to valid egress points. However, all in-flight packets are dropped if they reach the affected border router before it updates its forwarding table with new egress points for destinations it lost connectivity to. Alternatively, re-encapsulators could be used to avoid all packet loss for prefixes with multiple egress routers.

8.4.7 Deployment

Because of a limited number of upgrades, the proposal presents a low overall deployment cost. The architecture's data plane requires just the upgrading of a domain's BRs. Furthermore, the mapping system reuses the iBGP protocol and only requires the upgrading of RRs to LISP functionality. Alternatively, RRs could be coupled with devices that perform LISP mapping-system specific functions.

If the scalability of the RC is a concern due to associated operational complexity, solutions like [110] could be implemented for distributing the collator.

8.5 Evaluation

LISP-MPS offers operators flexibility in controlling their transit traffic over the different egress points. In this section, we evaluate two aspects of LISP-MPS. On the one hand, we estimate the gain in term of path diversity that an operator can expect if it deploys LISP-MPS. On the other hand, we determine the cost of using the technology to leverage the diversity by estimating the overhead in the routing table caused by the injection of interface related prefixes.

8.5.1 Path diversity incentives

BGP is such that only one route can be used to reach a destination. However, it is frequent that an AS receives several routes for each prefix, and this diversity is lost because of the BGP decision process. To quantify the potential path diversity that an operator can use by using LISP-MPS, we studied the diversity of BGP routes. For that purpose, we analyzed the BGP feeds of the four routers belonging to the University of Oregon available at Routeviews [134]. For each router, we took the Routing Information Based snapshot at midnight on March 15^{th}, 2012. Fig. 8.3 shows distribution of route diversity for three different filtering rules. More precisely, the figure shows the cumulative distribution of the number of prefixes (among the 424,833 prefixes) grouped by the number of routes that remain to reach them after being filtered. The curve label `no filter` gives the number of routes received for each prefix. As we can see, 95.5% of the prefixes have at least 2 routes. In other words, in general prefixes have path diversity. However, some routes should not be used because they are too long and would impact the performance. The curve labeled `shortest AS path` takes the length of the path into account and filters the RIB to only keep the routes that minimize the path length. In this situation, the proportion of prefixes with at least two routes is still 70% which means that the traffic for almost two thirds of the prefixes could be load balanced between paths of same length. Finally, the curve labeled `same AS path` determines all the routes that have the same AS path as the route that would have been chosen by BGP's decision process. We observe that we have still 50.6% of the prefixes with at least two routes. In this particular case, the routes can be used in parallel, without disrupting BGP, as the AS path is preserved.

As a summary, an operator can see benefits in using LISP-MPS as it enables the use of several routes in parallel. This increases its traffic engineering capabilities and potentially reduces the traffic's transit cost [47].

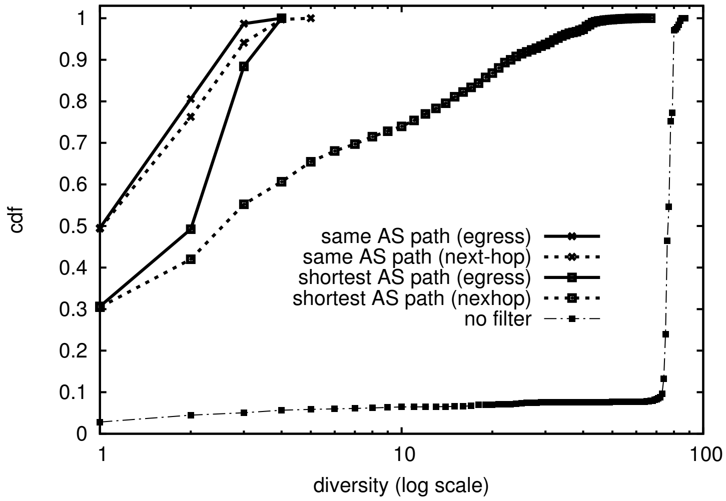


Fig. 8.3 Distribution of the BGP path diversity (i.e., the number of routes for a prefix) under different filtering rules

8.5.2 Routing overhead

In the previous section, we saw that operators could gain in terms of diversity when using LISP-MPS. In this section we put this gain in perspective by estimating the routing overhead caused by the router addressing scheme required for enhanced traffic control. The proposed scheme consists in advertising all IGP facing interfaces of border routers in the IGP as well as an aggregate to protect against failure.

We have estimated Ω, see eq. (8.1), for 8 different topologies. Among them, there is the topology provided by Internet2 [3], the topology of Géant [2] and the last 6 are taken from Rocketfuel [7]. For Géant and Internet2, all the details are provided so we can determine exactly the BRs and the BBRs. Alas, this is not possible for the Rocketfuel topologies. Therefore, we assigned the role of BR to two routers in each city. For this, we assumed that every city is a Point-of-Presence (PoP) and that a PoP must be protected against the failure of one router, hence two BR per city. For the considered topologies, we observe the value of Ω to be 8, 21, 128, 151, 166, 200, 294, and 513. In addition, we found that the number of IGP facing interfaces at the BRs is 4 ± 0.48. What is interesting in these results is that even in the case of large networks, the number of additional routing entries remains small in comparison to those necessary to operate BGP. The network with the largest Ω (of 513) is the one reported by Rocketfuel for Sprint. It has no less than 1944 links, 315 routers, and for which we accounted no less than 83 BRs.

8.6 Related Work

The section reviews the ideas of some similarly aimed works carried in the field.

FIB Aggregation is an opportunistic technique that offers per router FIB size reductions by algorithmically removing specific forwarding (child) entries which share the same next hop with their trie ancestors. The procedure ensures forwarding correctness however, depending on the employed algorithms, it may introduce previously non-routable address space in the FIB. There are several proposals [27, 50, 88, 143] that recommend the use of these techniques for reducing routing table sizes. Notably, [143] presents a systematic analysis of costs and benefits for FIB aggregation and it concludes that it is a viable short-term solution.

Several works propose, like us, the use of *tunneling* for relieving the pressure exerted by the size of forwarding tables on routers. Virtual Aggregation [17] tries to diminish the routing tables of routers within an AS by having the legacy routers forward their traffic to several *aggregation point routers* (APRs) instead of the best egress points. The forwarding on this second section (from the APR to the ASBR) is done by using MPLS tunnels in order to avoid routing loops. As a result, the number of FIB entries in legacy routers is limited to the number of APRs. A downsides of this solution is that it introduces additional path-stretch within the AS. Many Loc/ID split proposals [88] make use of encapsulation to decouple core from edge routing. Depending on how their deployment is to be done, they could reduce the size of the DFZ routing table.

These solutions manage to decrease the intra-domain routing tables, either through aggregation or by exclusion of edge-networks (EID) address space. Even so, there is still a direct relation between the size of the RLOC space and the size of the routing tables in a domain's backbone network. Our solution however, isolates intra-domain from inter-domain routing and directly relates the backbone routing table size to the number of BRs.

RCF 3107 [113] suggests the distribution of BGP routes with MPLS label mappings piggybacked onto them. Should border routers be using this mechanism together with a intra-domain label distribution protocol, then there is no need for BB routers to run iBGP if they support MPLS. At the edge of the domain a packet would get encapsulated with the label mapped on its matched route and subsequently MPLS forwarded over the backbone to its intra-domain next hop. We make use of BGP label mapped routes in our proposal however, instead of using MPLS, we use LISP encapsulation. This saves the need to support MPLS in the network's backbone and the deployment of a label distribution protocol.

8.7 Chapter Summary

In this chapter we have devised and analyzed LISP-MPS, a LISP based solution that increases the lifespan of ISP backbone network routers by making the size of their routing tables proportional to the number of border routers. Our idea is to use LISP encapsulation as a simple and efficient mechanism to reduce routing table growth driven by inefficient intra-domain DFZ route redistribution. Although we propose the use of the existing iBGP infrastructure to implement a domain constraint mapping system, we explain how it could be enhanced with a Route Collator, a domain's routing controller, capable of implementing high-level routing policies. Finally, we show with BGP traces obtained from RouterViews and topologies from Rocketfuel that the traffic engineering opportunities of an AS are drastically increased when using LISP-MPS. Furthermore, we show that the offered feature set reduces the capital costs but maintains strong resiliency capabilities. More work is needed to understand how to implement the Route Collator high level routing policies in a distributed way.

Chapter 9

Conclusions and Future Work

Despite its tremendous success, the Internet's routing infrastructure is facing increasing scalability problems due to growing routing tables and routing information churn. After careful analysis, this is now generally accepted as a consequence of IP's semantics being overloaded with both location and identity information. However, while their separation appears to be an obvious solution to the problem, such decision has non-trivial architectural implications and not necessarily a clear cut implementation as shown by the various, and slightly different, proposals in favor of the split.

In light of its development and community support, LISP is arguably the most advanced of these solutions. But although its deployment entails only mild infrastructure upgrades they are of fundamental consequence to how forwarding is performed chiefly due to the introduction of a mapping system and dependence on caching. Naturally, this raises the question: *Is LISP a good step forward, towards improving the overall routing infrastructure?* This dissertation shows that with regard to two of the most difficult needs that it must cater for, the answer is positive: on the one hand, the architecture scales constantly with today's traffic patterns and should do so in the foreseeable future, and on the other, LISP packs additional benefits, the system currently lacks, that make it attractive to early adopters and thereby encourage its global deployment.

While LISP and the set of requirements to be satisfied by new Internet architectures will undoubtedly evolve, we hope that the analysis proposed here can prove to be, at the very least, a useful reference point. LISP tunnel routers employ a pull approach to obtaining loc/id mappings and therefore render their performance dependent on allotted cache size. Nevertheless, we saw that aggregate traffic properties enable cache provisioning from a simple to obtain set of parameters and quite importantly, ensure that cache performance is independent of Internet and user growth. In a context where the identity namespace could become vastly larger than the locator namespace and extremely expensive to hold in memory, these results show that a pull instead of a push design approach could be easily warranted, despite the inferior performance.

Even if LISP is not adopted due to its scalability properties, it should still be considered for the benefits derived from separating control and data plane. As services requested of the networks are becoming increasingly more complex and scale out, they are also becoming harder to deploy and more expensive to manage. This complexity is often managed by simplifying forwarding and centralizing control with the help of specialized overlays. As we showed, LISP offers the right tools to automate overlay setup and due to its logically centralized control, i.e, the mapping system, can be easily programmed to solve specific pain points in operator networks. We next summarize our contributions.

In the first part of the thesis, we first showed that the working-set can be efficiently used to approximate traffic locality and further on to build a practical analytical map-cache model. Thanks to this model, operators can now easily approximate tunnel router performance and dimension their network in accordance to operational needs. Using one day long real traffic network traces we found that miss rate decreases at an accelerated pace with cache size and finally settles to a power-law decrease. And as a result, cache sizes need not be very large with respect to the size of the identifier namespace for obtaining good performance. Our main result was to show that there exists a closed form equation that estimates cache performance starting from parameters that estimate intrinsic locality of packet level user traffic which can be easily captured using the average working-set size. We also proved that the clustering of the working-set curves is the only condition necessary to be satisfied when applying the model.

To prove the versatility of our model but also to investigate the vulnerability of unprotected LISP deployments, we design an extension capable of evaluating the impact of malicious users performing cache polluting attacks. Our first rather surprising result was that attacks carried in a random fashion are more damaging than those aimed at maximizing attack efficiency by focusing on address space apriori known not to be present in the victim's map-cache. Second, although in normal conditions increasing cache size quickly diminishes the miss rate, it has close to no effect under polluting attacks, including when the attacks are of low intensity. Given that even a single compromised hosts would be enough to induce significant damage, we briefly sketched how more complex management strategies could be developed and set in place.

As a last step in our analysis we evaluated the map-cache's performance scalability. Starting from a set of empirical observations, which we also confirmed using real traffic traces, we formulated a set of assumptions regarding the properties of network traffic that help in defining network locality: (i) long-term popularity can be modeled as a constant Generalized Zipf distribution and (ii) temporal locality is predominantly determined by long-term popularity. Then, under these assumptions we deduced a generalized model capable of predicting map-cache miss rates with respect to cache size using as input only the parameters defining the popularity distribution. Apart from the obvious benefit of allowing network provisioning based on expected or theoretical popularity distributions, the result also allows us to reason about the cache's scalability.

In light of the observation that (iii) popularity distribution is independent of the number of users in a LISP site and the size of the identity namespace, we find that cache size scales constantly with the number of users and destinations. This has several important implications for LISP's deployment. First, caches can be provisioned for desired performance with subsequently should not degrade as the site and the Internet grow. Second, load balancing of traffic within a large site does not affect performance. And finally, if no cache hierarchies are used, the number of resolution requests scales linearly with the number of users, so mapping systems should be designed to gracefully cope with higher loads. Nevertheless, if the assumptions do not hold, we also show that cache sizes should at worst grow linearly with the number of destinations.

In the second part of the thesis we focused on mechanisms that can be easily leveraged to build overlays that solve specialized problems. The first problem we tackled was that of designing an inter-domain multicast framework. Traditionally, efficient inter-domain data delivery may be implemented either as a network or application layer multicast service. However, while the former has seen little uptake due to prohibitive deployment costs the latter is widely used today, but often without a minimum guaranteed performance. In this chapter we presented Lcast, a network-layer single-source multicast framework designed to merge the robustness and efficiency of IP multicast with the configurability and low deployment cost of application-layer overlays. The architecture involves no end-host changes and only requires the upgrading of a small set of routers to support the Locator/ID Separation Protocol (LISP), an incrementally deployable enhancement to the current global routing infrastructure. Content distribution over the Internet's core is done by means of a router overlay while within domains, end-hosts interface with Lcast using conventional multicast protocols. The overlay's scalability and topological configurability is sustained by logically centralizing group management. We illustrated Lcast's versatility by designing and assessing the scalability and performance of three management strategies for low latency content distribution. Our analysis was based on large scale simulations supported by realistic user behavior and Internet-like network topologies. The results showed Lcast's low management overhead and ability to optimize delivery to meet various operational constraints. Notably, we found that it can deliver traffic with latencies close to unicast ones, independent of overlay size and that randomly built distribution trees offer surprisingly good performance.

Finally, in the last chapter we devised and analyzed LISP-MPS, an architecture that isolates an AS' the intra-domain routing from its inter-domain routing. The resulting separation implies the decrease of backbone routing table sizes and enables the AS to control the forwarding of traffic inside its network. We explained that for a seamless, cost effective, and incremental deployment, LISP-MPS should leverage iBGP to implement LISP mapping system functionality with minimal modification to a small subset of deployed equipment. Using realistic topologies we showed that, despite changing packet forwarding within a network, the architecture does not lose resilience to failures. Moreover, we also argued why it can be a viable alternative to BGP/MPLS deployments due to its low implementation cost.

Throughout the document we already pointed out some of the limitations of our work that open interesting opportunities for future research. We briefly summarize them in the following paragraphs.

The cache models deduced in the first part of thesis always produce average performance numbers. It would be an interesting exercise to try and evaluate the possibility of designing a cache model capable of capturing instantaneous cache behavior. This would open the possibility of provisioning caches to minimize miss bursts.

Throughout the dissertation, LRU has been the only eviction policy considered for use in map-caches. In fact, in Chapter 4 we show that its performance should be close to ideal. Nevertheless, it would be worth investigating other eviction policies, especially those capable of withstanding pollution attacks, given the poor performance of map-caches in such situations. Moreover, it would be interesting to see if coupling the eviction policy with a more complex cache attack detection mechanism, based for instance on the average working-set, like we recommended in Chapter 5, would help improve performance.

An important aspect not consider for Lcast is ETR multihoming. Designing the mechanisms needed for inter-ETR load balancing could result not only in a more efficient use of router resources but also a simple mechanism that could avoid packet loss when random failures affect distribution tree links.

Since our work on LISP-MPS has been mainly conceptual, it would be interesting to implement the architecture and evaluate the interaction between BGP and LISP. It would also be worth comparing the operational overhead of LISP-MPS with that of MPLS.

References

[1] Cymru, 2012. URL `http://www.team-cymru.org`.

[2] Geant, 2012. URL `http://www.geant.net/About_GEANT/pages/home.aspx`.

[3] Internet2, 2012. URL `http://www.internet2.edu/`.

[4] PPLive P2P Internet TV, 2012. URL `http://www.pplive.com/`.

[5] Sopcast P2P Internet TV, 2012. URL `http://www.sopcast.com`.

[6] UUSee P2P Internet TV, 2012. URL `http://www.uusee.com/`.

[7] Rocketfuel, 2012. URL `http://www.cs.washington.edu/research/networking/rocketfuel/`.

[8] LISP Beta Network, 2015. URL `http://www.lisp4.net/`.

[9] IETF Locator/ID Separation Protocol WG, 2015. URL `https://datatracker.ietf.org/wg/lisp/charter/`.

[10] LISPmob, 2015. URL `http://www.lispmob.org`.

[11] A. Agarwal, J. Hennessy, and M. Horowitz. An analytical cache model. *ACM Trans. Comput. Syst.*, 7(2):184–215, May 1989.

[12] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *Communications Magazine, IEEE*, 50(7):26–36, 2012.

[13] M. Ain, D. Trossen, P. Nikander, S. Tarkoma, K. Visala, K. Rimey, T. Burbridge, J. Rajahalme, J. Tuononen, P. Jokela, et al. D2. 3–architecture definition, component descriptions, and requirements. *Deliverable, PSIRP 7th FP EU-funded project*, 2009.

[14] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles*, SOSP, pages 131–145, New York, NY, USA, 2001. ACM.

[15] R. Atkinson and S. Bhatti. Identifier-Locator Network Protocol (ILNP) Architectural Description. RFC 6740 (Experimental), November 2012. URL `http://www.ietf.org/rfc/rfc6740.txt`.

[16] R. Atkinson and S. Bhatti. Identifier-Locator Network Protocol (ILNP) Engineering Considerations. RFC 6741 (Experimental), November 2012. URL http://www.ietf.org/rfc/rfc6741.txt.

[17] H. Ballani, P. Francis, T. Cao, and J. Wang. Making routers last longer with viaggre. In *NSDI*, volume 9, pages 453–466, 2009.

[18] A. Banerjee, J. Dolado, J. Galbraith, and D. F. Hendry. Cointegration, error correction and the econometric analysis of non stationary data, 1993.

[19] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller. Construction of an efficient overlay multicast infrastructure for real-time applications. In *Proceedings of IEEE INFOCOM*, pages 1521–1531, 2003.

[20] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proceedings ACM SIGCOMM*, 2002.

[21] T. Bates, E. Chen, and R. Chandra. BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP). RFC 4456 (Draft Standard), April 2006. URL http://www.ietf.org/rfc/rfc4456.txt.

[22] T. Bates, R. Chandra, D. Katz, and Y. Rekhter. Multiprotocol Extensions for BGP-4. RFC 4760 (Draft Standard), January 2007. URL http://www.ietf.org/rfc/rfc4760.txt.

[23] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. In *INFOCOM, In Proceedings. IEEE*, volume 1, pages 126–134 vol.1, 1999.

[24] 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, April 2008. URL http://tools.ietf.org/html/draft-meyer-lisp-cons-04. Work in progress.

[25] G. Bumgardner. Automatic Multicast Tunneling. draft-ietf-mboned-auto-multicast-14, June 2012. Work in progress.

[26] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan. Internet Group Management Protocol, Version 3. RFC 3376 (Proposed Standard), October 2002. URL http://www.ietf.org/rfc/rfc3376.txt. Updated by RFC 4604.

[27] B. Cain. Auto aggregation method for ip prefix/length pairs, June 4 2002. US Patent 6,401,130.

[28] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. Scribe: a large-scale and decentralized application-level multicast infrastructure. *Selected Areas in Communications, IEEE Journal on*, 20(8):1489 – 1499, oct 2002. ISSN 0733-8716.

[29] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2007.

[30] R. Chalmers and K. Almeroth. On the topology of multicast trees. *Networking, IEEE/ACM Transactions on*, 11(1):153–165, 2003. ISSN 1063-6692.

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

[32] L. Cittadini, W. Muhlbauer, S. Uhlig, R. Bush, P. Francois, and O. Maennel. Evolution of internet address space deaggregation: myths and reality. *Selected Areas in Communications, IEEE Journal on*, 28(8):1238–1249, 2010.

[33] F. Coras, T. Silverston, J. Domingo-Pascual, and A. Cabellos-Aparicio. A Measurement Study of SOPCast. Technical Report UPC-DAC-RR-CBA-2012-2, UPC, 2012. URL `http://personals.ac.upc.edu/fcoras/publications/2012-fcoras-sopcast-study.pdf`.

[34] F. Coras, D. Saucez, L. Iannone, and B. Donnet. On the performance of the lisp beta network. In *Proc. IFIP Networking*. Springer, June 2015.

[35] F. Coras, A. Cabellos-Aparicio, and J. Domingo-Pascual. An analytical model for the LISP cache size. In *Proc. IFIP Networking*, pages 409–420. Springer, 2012.

[36] F. Coras, D. Saucez, L. Jakab, A. Cabellos-Aparicio, and J. Domingo-Pascual. Implementing a bgp-free isp core with lisp. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 2772–2778. IEEE, 2012.

[37] F. Coras, J. Domingo-Pascual, D. Lewis, and A. Cabellos-Aparicio. An analytical model for loc/id mappings caches. *arXiv preprint arXiv:1312.1378*, 2013.

[38] F. Coras, J. Domingo-Pascual, F. Maino, D. Farinacci, and A. Cabellos-Aparicio. Lcast: Software-defined inter-domain multicast. *Computer Networks*, 2013.

[39] F. Coras, A. Cabellos-Aparicio, J. Domingo-Pascual, F. Maino, and D. Farinacci. Locator/ID Separation Protocol (LISP). draft-coras-lisp-re-06, October 2014. Work in progress.

[40] G. Dán and N. Carlsson. Power-law revisited: large scale measurement study of p2p content popularity. In *IPTPS*, page 12, 2010.

[41] J. Day. *Patterns in network architecture: A return to fundamentals.* Pearson Education, 2007.

[42] J. Day. *Patterns in network architecture: a return to fundamentals.* Pearson Education, 2007.

[43] J. Day, I. Matta, and K. Mattar. Networking is ipc: a guiding principle to a better internet. In *Proceedings of the 2008 ACM CoNEXT Conference*, page 67. ACM, 2008.

[44] S. Deering. Host extensions for IP multicasting. RFC 1112 (INTERNET STANDARD), August 1989. URL `http://www.ietf.org/rfc/rfc1112.txt`. Updated by RFC 2236.

[45] P. J. Denning. The working set model for program behavior. *Commun. ACM*, 11(5):323–333, 1968.

[46] P. J. Denning and S. C. Schwartz. Properties of the working-set model. *Commun. ACM*, 15(3):191–198, 1972.

[47] A. Dhamdhere and C. Dovrolis. ISP and Egress Path Selection for Multihomed Networks. In *INFOCOM*, pages 1 –12, April 2006.

[48] X. Dimitropoulos, D. Krioukov, and G. Riley. Revisiting Internet AS-Level Topology Discovery. In *Passive and Active Network Measurement*, volume 3431 of *Lecture Notes in Computer Science*, pages 177–188. Springer Berlin Heidelberg, 2005.

[49] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network*, 14(1): 78–88, January 2000.

[50] R. Draves, C. King, S. Venkatachary, and B. Zill. Constructing optimal IP routing tables. In *INFOCOM*, pages 88–97, 1999.

[51] A. Elmokashfi, A. Kvalbein, and C. Dovrolis. BGP churn evolution: a perspective from the core. *Networking, IEEE/ACM Transactions on*, 20(2):571–584, 2012.

[52] H. Eriksson. MBONE: the multicast backbone. *Communications of the ACM*, 37(8):54–60, August 1994.

[53] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, 1999. ISBN 1-58113-135-6.

[54] D. Farinacci and M. Napierala. LISP Control-Plane Multicast Signaling. draft-farinacci-lisp-mr-signaling-01, January 2013. Work in progress.

[55] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis. The Locator/ID Separation Protocol (LISP). RFC 6830 (Experimental), January 2013. URL `http://www.ietf.org/rfc/rfc6830.txt`.

[56] D. Farinacci, D. Meyer, J. Zwiebel, and S. Venaas. The Locator/ID Separation Protocol (LISP) for Multicast Environments. RFC 6831 (Experimental), January 2013. URL `http://www.ietf.org/rfc/rfc6831.txt`.

[57] D. Farinacci, D. Meyer, and J. Snijders. LISP Canonical Address Format (LCAF). draft-ietf-lisp-lcaf-06, October 2014. Work in progress.

[58] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. Van Der Merwe. The case for separating routing from routers. In *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, pages 5–12. ACM, 2004.

[59] D. Feldmeier. Improving gateway performance with a routing-table cache. In *INFOCOM'88. Networks: Evolution or Revolution, Proceedings. Seventh Annual Joint Conference of the IEEE Computer and Communcations Societies, IEEE*, pages 298–307. IEEE, 1988.

[60] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas. Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised). RFC 4601 (Proposed Standard), August 2006. URL `http://www.ietf.org/rfc/rfc4601.txt`. Updated by RFCs 5059, 5796, 6226.

[61] V. Fuller and D. Farinacci. Locator/ID Separation Protocol (LISP) Map-Server Interface. RFC 6833 (Experimental), January 2013. URL `http://www.ietf.org/rfc/rfc6833.txt`.

[62] V. Fuller and T. Li. Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan. RFC 4632 (Best Current Practice), August 2006. URL `http://www.ietf.org/rfc/rfc4632.txt`.

[63] V. Fuller, D. Farinacci, D. Meyer, and D. Lewis. Locator/ID Separation Protocol Alternative Logical Topology (LISP+ALT). RFC 6836 (Experimental), January 2013. URL `http://www.ietf.org/rfc/rfc6836.txt`.

[64] V. Fuller, D. Lewis, V. Ermagan, and A. Jain. LISP Delegated Database Tree. draft-ietf-lisp-ddt-03.txt, April 2015. Work in progress.

[65] X. Hei, C. Lian, J. Lian, Y. Liu, and K. W. Ross. A Measurement Study of a Large-Scale P2P IPTV System. *TOM*, 9(8), December 2007.

[66] R. Hinden. New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG. RFC 1955 (Informational), June 1996. URL `http://www.ietf.org/rfc/rfc1955.txt`.

[67] M. Hoefling, M. Menth, and M. Hartmann. A survey of mapping systems for locator/identifier split internet routing. *Communications Surveys Tutorials*, 2013.

[68] H. Holbrook and B. Cain. Source-Specific Multicast for IP. RFC 4607 (Proposed Standard), August 2006. URL `http://www.ietf.org/rfc/rfc4607.txt`.

[69] H. Holbrook, B. Cain, and B. Haberman. Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast. RFC 4604 (Proposed Standard), August 2006. URL `http://www.ietf.org/rfc/rfc4604.txt`.

[70] C. Hopps. Analysis of an Equal-Cost Multi-Path Algorithm. RFC 2992 (Informational), November 2000. URL `http://www.ietf.org/rfc/rfc2992.txt`.

[71] Y. hua Chu, S. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *Selected Areas in Communications, IEEE Journal on*, 20(8):1456 – 1471, oct 2002. ISSN 0733-8716.

[72] G. Huston. BGP Report, 2011. URL `http://bgp.potaroo.net/`.

[73] G. Huston and G. Armitage. Projecting future ipv4 router requirements from trends in dynamic bgp behaviour. In *Proc. of ATNAC*, 2006.

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

[75] L. Iannone and O. Bonaventure. On the Cost of Caching Locator/ID Mappings. In *Proceedings of the 3rd International Conference on emerging Networking EXperiments and Technologies (CoNEXT'07)*, pages 1–12. ACM, December 2007. ISBN 978-1-59593-770-4.

[76] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM, 2009.

[77] R. Jain. Characteristics of destination address locality in computer networks: A comparison of caching schemes. *Computer networks and ISDN systems*, 18 (4):243–254, 1990.

[78] 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*, 28(8):1332 –1343, october 2010.

[79] L. Jakab, A. Cabellos-Aparicio, F. Coras, J. Domingo-Pascual, and D. Lewis. Locator/Identifier Separation Protocol (LISP) Network Element Deployment Considerations. RFC 7215 (Experimental), April 2014. URL `http://www.ietf.org/rfc/rfc7215.txt`.

[80] L. Jakab, A. Cabellos-Aparicio, T. Silverston, M. Sole, F. Coras, and J. Domingo-Pascual. Corecast: How core/edge separation can help improving inter-domain live streaming. *Computer Networks*, 54(18):3388 – 3401, 2010.

[81] S. Jin and A. Bestavros. Sources and characteristics of web temporal locality. In *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, In Proceedings. International Symposium on*, pages 28–35. IEEE, 2000.

[82] X. Jin, K.-L. Cheng, and S.-H. Chan. Island multicast: combining IP multicast with overlay data distribution. *Multimedia, IEEE Transactions on*, 11(5): 1024–1036, 2009.

[83] C. Kim, M. Caesar, A. Gerber, and J. Rexford. Revisiting Route Caching: The World Should Be Flat. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 3–12, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00974-7.

[84] J. Kim, L. Iannone, and A. Feldmann. A deep dive into the LISP cache and what ISPs should know about it. In *Proceedings of the 10th international IFIP TC 6 conference on Networking - Volume Part I*, NETWORKING'11, pages 367–378, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-20756-3. URL `http://dl.acm.org/citation.cfm?id=2008780.2008817`.

[85] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. *ACM SIGCOMM Computer Communication Review*, 37(4):181–192, 2007.

[86] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. Internet inter-domain traffic. *ACM SIGCOMM Computer Communication Review*, 40(4):75–86, 2010.

[87] E. Lear. NERD: A Not-so-novel Endpoint ID (EID) to Routing Locator (RLOC) Database. RFC 6837 (Experimental), January 2013. URL `http://www.ietf.org/rfc/rfc6837.txt`.

[88] T. Li. Recommendation for a Routing Architecture. RFC 6115 (Informational), February 2011. URL `http://www.ietf.org/rfc/rfc6115.txt`.

[89] T. Li. Design Goals for Scalable Internet Routing. RFC 6227 (Informational), May 2011. URL `http://www.ietf.org/rfc/rfc6227.txt`.

[90] J. Liebeherr, M. Nahas, and W. Si. Application-layer multicasting with delaunay triangulation overlays. *Selected Areas in Communications, IEEE Journal on*, 20(8):1472–1488, 2002.

[91] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee. Triangle inequality and routing policy violations in the internet. In *PAM*, pages 45–54. Springer, 2009.

[92] 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*, November 2006.

[93] H. V. Madhyastha, E. Katz-Bassett, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane Nano: path prediction for peer-to-peer applications. In *NSDI, Proceedings USENIX*, pages 137–152, 2009.

[94] A. Mahanti, C. Williamson, and D. Eager. Traffic analysis of a web proxy caching hierarchy. *Network, IEEE*, 14(3):16–23, 2000.

[95] D. Massey, L. Wang, B. Zhang, and L. Zhang. A scalable routing system design for future internet. In *Proc. of ACM SIGCOMM Workshop on IPv6*, 2007.

[96] L. Mathy, L. Iannone, and O. Bonaventure. LISP-DHT: Towards a DHT to map identifiers onto locators. draft-mathy-lisp-dht-00, February 2008. Work in progress.

[97] N. McKeown. Software-defined Networking. Keynote Talk at IEEE INFOCOM 2009, 2009. URL http://tiny-tera.stanford.edu/~nickm/talks/infocom_brazil_2009_v1-1.pdf.

[98] C. Metz, C. Barth, and C. Filsfils. Beyond MPLS... Less Is More. *Internet Computing, IEEE*, 11(5):72–76, 2007.

[99] D. Meyer, L. Zhang, and K. Fall. Report from the IAB Workshop on Routing and Addressing. RFC 4984 (Informational), September 2007. URL http://www.ietf.org/rfc/rfc4984.txt.

[100] D. Meyer. Update on routing and addressing at ietf 69. *IETF Journal*, 3(2), October 2007.

[101] D. Meyer. The locator identifier separation protocol (LISP). *The Internet Protocol Journal*, 11(1):23–36, 2008.

[102] D. Meyer. It's the end of the Internet as we know it. NANOG 45 Presentation, January 2009. URL https://www.nanog.org/meetings/nanog45/presentations/Monday/Meyer_iteotwawki_N45.pdf.

[103] D. Meyer and D. Lewis. Architectural implications of locator/id separation. draft-meyer-loc-id-implications-01, Jan 2009. URL http://http://tools.ietf.org/html/draft-meyer-loc-id-implications-01. Work in progress.

[104] X. Misseri, J. Rougier, and D. Saucez. Internet routing diversity for stub networks with a map-and-encap scheme. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2861–2866. IEEE, 2012.

[105] M. A. Montemurro. Beyond the zipf–mandelbrot law in quantitative linguistics. *Physica A: Statistical Mechanics and its Applications*, 300(3):567–578, 2001.

[106] S. Murray. Digital TV World Revenue Forecasts, July 2012. URL http://www.digitaltvresearch.com/ugc/press/37.pdf.

[107] T. Narten. On the Scalability of Internet Routing. draft-narten-radir-problem-statement-05, February 2010. Work in progress.

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

[109] R. Oliveira, B. Zhang, D. Pei, and L. Zhang. Quantifying path exploration in the internet. *Networking, IEEE/ACM Transactions on*, 17(2):445–458, 2009.

[110] I. Oprescu, M. Meulle, S. Uhlig, C. Pelsser, O. Maennel, and P. Owezarski. oBGP: an Overlay for a Scalable iBGP Control Plane. *NETWORKING 2011*, pages 420–431, 2011.

[111] D. Phung, S. Secci, D. Saucez, and L. Iannone. The openlisp control plane architecture. *Network, IEEE*, 28(2):34–40, March 2014. ISSN 0890-8044.

[112] B. Quoitin, L. Iannone, C. De Launois, and O. Bonaventure. Evaluating the benefits of the locator/identifier separation. In *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, page 5. ACM, 2007.

[113] Y. Rekhter and E. Rosen. Carrying Label Information in BGP-4. RFC 3107 (Proposed Standard), May 2001. URL `http://www.ietf.org/rfc/rfc3107.txt`. Updated by RFC 6790.

[114] J. Rexford and C. Dovrolis. Future internet architecture: clean-slate versus evolutionary research. *Communications of the ACM*, 53(9):36–40, 2010.

[115] RIPE. Routing Information Service (RIS), 2011. URL `https://labs.ripe.net/datarepository/data-sets/routing-information-service-ris-raw-data-set`.

[116] L. Rizzo and L. Vicisano. Replacement policies for a proxy cache. *IEEE/ACM Trans. Netw.*, 8(2):158–170, April 2000.

[117] E. Rosen and Y. Rekhter. BGP/MPLS IP Virtual Private Networks (VPNs). RFC 4364 (Proposed Standard), February 2006. URL `http://www.ietf.org/rfc/rfc4364.txt`. Updated by RFCs 4577, 4684, 5462.

[118] J. Saltzer. On the Naming and Binding of Network Destinations. RFC 1498 (Informational), August 1993. URL `http://www.ietf.org/rfc/rfc1498.txt`.

[119] N. Sarrar, S. Uhlig, A. Feldmann, R. Sherwood, and X. Huang. Leveraging zipf's law for traffic offloading. *ACM SIGCOMM Computer Communication Review*, 42(1):16–22, 2012.

[120] D. Saucez, L. Iannone, and B. Donnet. A first measurement look at the deployment and evolution of the locator/id separation protocol. *ACM SIGCOMM Computer Communication Review*, 43(1):37–43, April 2013.

[121] D. Saucez, L. Iannone, and O. Bonaventure. LISP Threats Analysis. draft-ietf-lisp-threats, April 2014. Work in progress.

[122] D. Saucez. *Mechanisms for interdomain Traffic Engineering with LISP*. PhD thesis, Université catholique de Louvain, 2011.

[123] D. Saucez, B. Donnet, L. Iannone, and O. Bonaventure. Interdomain traffic engineering in a locator/identifier separation context. In *Internet Network Management Workshop, 2008. INM 2008. IEEE*, pages 1–6. IEEE, 2008.

[124] D. Saucez, L. Iannone, O. Bonaventure, and D. Farinacci. Designing a Deployable Internet: The Locator/Identifier Separation Protocol. *IEEE Internet Computing*, 16:14–21, 2012.

[125] 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*, 29(4):289–299, 1999.

[126] S. Shi, J. Turner, and M. Waldvogel. Dimensioning server access bandwidth and multicast routing in overlay networks. In *NOSSDAV, Proceedings ACM*, pages 83–91, 2001.

[127] J. F. Shoch. Inter-network naming, addressing, and routing. In *COMPCON*, pages 72–79. IEEE, 1978.

[128] T. Silverston, L. Jakab, A. Cabellos-Aparicio, O. Fourmaux, K. Salamatian, and K. Cho. Large-scale measurement experiments of p2p-tv systems insights on fairness and locality. *Signal Processing: Image Communication*, 2011.

[129] T. Silverston, L. Jakab, A. Cabellos-Aparicio, O. Fourmaux, K. Salamatian, and K. Cho. Large-scale measurement experiments of P2P-TV systems insights on fairness and locality. *Signal Processing: Image Communication*, 26(7):327 – 338, 2011.

[130] J. R. Spirn and P. J. Denning. Experiments with program locality. In *Proceedings of the December 5-7, 1972, fall joint computer conference, part I*, pages 611–621. ACM, 1972.

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

[132] D. Thaler and C. Hopps. Multipath Issues in Unicast and Multicast Next-Hop Selection. RFC 2991 (Informational), November 2000. URL `http://www.ietf.org/rfc/rfc2991.txt`.

[133] D. Tran, K. Hua, and T. Do. A peer-to-peer architecture for media streaming. *Selected Areas in Communications, IEEE Journal on*, 22(1):121 – 133, January 2004. ISSN 0733-8716.

[134] University of Oregon. RouteViews Project, 2011. URL `http://www.routeviews.org`.

[135] E. Veloso, V. Almeida, W. Meira, and A. Bestavros. A hierarchical characterization of a live streaming media workload. *Networking, IEEE/ACM Transactions on*, 14(1):133–146, 2006.

[136] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt. Measurement and modeling of a large-scale overlay for multimedia streaming. *QSHINE*, 2007.

[137] C. Wu, B. Li, and S. Zhao. Diagnosing network-wide p2p live streaming inefficiencies. In *INFOCOM, Proceedings IEEE*, pages 2731–2735, 2009.

[138] S. Yeganeh, A. Tootoonchian, and Y. Ganjali. On scalability of software-defined networking. *Communications Magazine, IEEE*, 51(2):136–141, 2013.

[139] B. Zhang, W. Wang, S. Jamin, D. Massey, and L. Zhang. Universal IP multicast delivery. *Computer Networks*, 50(6):781–806, 2006.

[140] H. Zhang, M. Chen, and Y. Zhu. Evaluating the performance on ID/Loc mapping. In *Global Telecommunications Conference (GLOBECOM 2008)*, pages 1–5. IEEE, 2008.

[141] L. Zhang. An overview of multihoming and open issues in GSE. *IETF Journal*, 2(2), 2006.

[142] X. Zhang, J. Liu, B. Li, and Y.-S. Yum. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In *INFOCOM, Proceedings IEEE*, volume 3, pages 2102 – 2111, march 2005.

[143] X. Zhao, Y. Liu, L. Wang, and B. Zhang. On the Aggregatability of Router Forwarding Tables. *INFOCOM*, pages 1–9, March 2010.

[144] L. Zheng, Z. Zhang, and R. Parekh. Survey Report on PIM-SM Implementations and Deployments. draft-zzp-pim-rfc4601-update-survey-report, December 2012. Work in progress.

[145] L. Zhu, P. Leach, and S. Hartman. Anonymity Support for Kerberos. RFC 6112 (Proposed Standard), April 2011. URL `http://www.ietf.org/rfc/rfc6112.txt`.

[146] G. K. Zipf. *Human behavior and the principle of least effort.* Addison-Wesley Press, 1949.

Appendix A

A Measurement Study of SOPCast

In this appendix we present and analyze a set of experimental datasets that show some of the characteristics of SOPCast, a P2P live content streaming application.

A.1 Datasets

The aim, when building our testbed, was to create an infrastructure capable of performing a world-wide distributed passive capture of large P2P live content streaming overlays. The choice of the streaming content was driven by the subjectively perceived importance of the ongoing events at the time the experiment took place. As a result, all traces in our dataset consist of traffic pertaining to popular sports events. Reasons for our choice were threefold. First, users are generally interested in consuming such traffic live (as it gets produced). Secondly, interest for such events tends to be high world-wide. Finally, the usefulness of such content has been proven by previous [128] works that also aimed to characterize P2P live streaming overlays. In particular, we captured the content streamed by several SOPCast channels during the closing stages of the 2011 UEFA Champions League. Although we obtained datasets from multiple encounters, in this chapter we focus solely on a 2011 UEFA Champions League semifinal match. In spite of the fact that interest for such football matches is highest in Europe, the teams involved are both highly appreciated worldwide and amount players spanning many nationalities. Additionally, interest was increased as this was the penultimate phase of the prestigious competition.

For the capturing process we used 2 vantage points in USA, 3 in Europe and 2 in

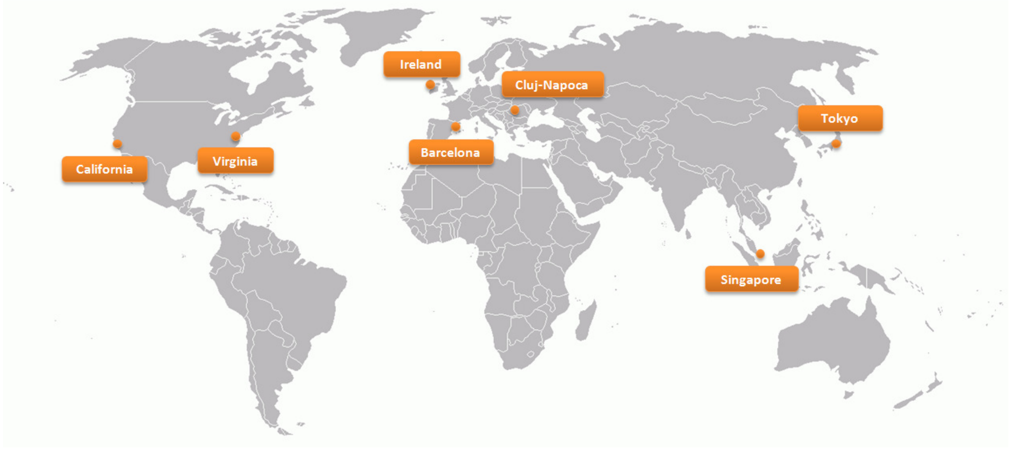


Fig. A.1 Vantage Points

Asia, spanning a total of 6 countries. A map depicting their geographic position is presented in Fig.A.1. In order to better capture the client interest for the streamed content, on each machine involved in the experiment we joined a number of SOPCast channels streaming the same event. Statistical properties of the traces are given in Table A.1. All machines involved ran Ubuntu Linux and had a 100Mbps Ethernet connectivity to the Internet. The packet capturing was done with *tcpdump*. Because multiple SOPCast instances ran in parallel on each PC, filtering of packets per channel was done based on UDP port number. The capture duration was always higher than the time span of the match in order to observe transitory peer behavior.

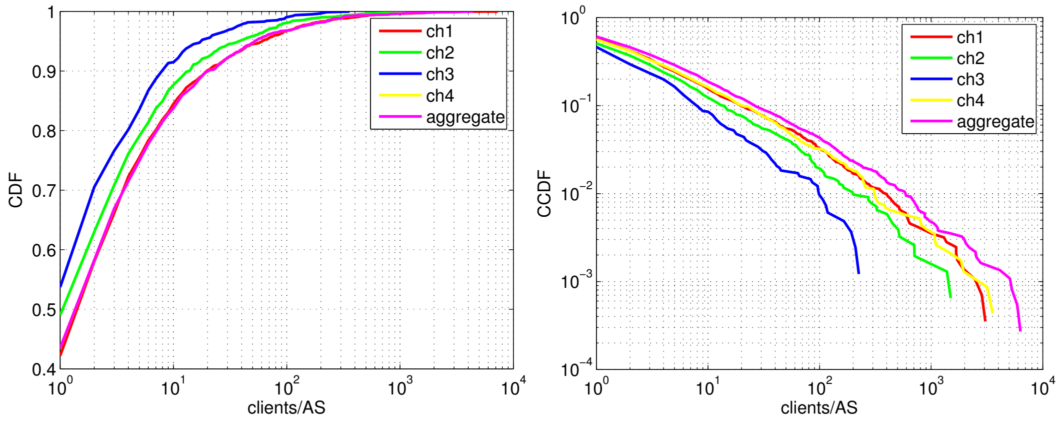
Peer IPs were mapped to their autonomous systems (ASes) with the help of an origin AS database obtained from RouteViews [134]. For brevity, in what follows we shall be referring to the traces by associating a vantage point with one of the four captured channels (e.g., *ch1-barcelona*).

A.1.1 Observations

Some of the statistical properties of the traces captured are presented in Table A.1. Among them, we have the count of peering IPs encountered in each trace, which may be used as an estimate for the number of connected end-hosts. However, because we did not identify hosts behind NAT boxes, this value should be held as a lower bound estimate. From the peer IPs, the number of Autonomous Systems (AS) exchanging traffic with our nodes is inferred. A similarity metric (described lower) is also computed for both IPs and ASes and the ratio of uploaded/downloaded traffic

Table A.1 Trace properties

	Download (GB)	**Number of IPs**	**Number of ASes**	**Up** (%)	**Down** (%)	**Similarity IP** (%)	**Similarity AS** (%)
ch1@850Kbps	9.63	64586	2839	89	11	12	68
california	1.45	19250	1469	76	24	93	98
cluj	1.50	32229	1980	90	10	92	96
ireland	0.99	13522	1294	66	34	92	98
barcelona	1.31	34320	1940	91	9	78	94
singapore	1.39	37164	2039	89	11	89	96
tokyo	1.18	37822	2028	95	5	91	96
virginia	1.33	21864	1745	88	12	92	96
ch2@345Kbps	2.48	19987	1539	82	18	24	87
california	0.42	9213	1060	74	26	92	98
cluj	0.47	11432	1212	91	9	85	93
ireland	0.19	6164	878	69	31	92	97
barcelona	0.38	7475	962	78	22	88	96
singapore	0.30	6025	709	73	27	82	96
tokyo	0.29	7014	865	77	23	87	96
virginia	0.43	7044	937	78	22	90	97
ch3@525Kbps	6.77	4815	820	93	7	26	90
california	0.93	3439	656	92	8	97	99
cluj	1.24	3515	659	94	6	98	98
ireland	0.87	3269	645	95	5	96	98
barcelona	1.06	3333	640	91	9	94	97
singapore	0.76	2715	568	79	21	98	98
tokyo	0.80	2853	590	69	31	98	99
virginia	1.11	3756	702	94	6	95	97
ch4@800Kbps	7.37	49814	2303	89	11	19	82
california	1.05	23307	1592	89	11	91	97
cluj	1.12	16734	1403	86	14	92	97
ireland	0.86	23983	1593	91	9	92	96
barcelona	1.36	23040	1614	90	10	92	96
singapore	0.79	16240	1185	85	15	88	98
tokyo	1.00	22926	1539	89	11	92	96
virginia	1.19	25254	1636	91	9	91	96



(a) CDF (b) CCDF

Fig. A.2 Clients per AS distribution

from each vantage point.

The balance between the upload and download ratios provides an insight into the peers altruistic nature. Content to be consumed is downloaded only once however depending on the peer's upload capacity it may be replicated several times. Channels with higher bitrate require an even larger effort from the peers. The second column in Table A.1 shows the volume of content downloaded by each of our vantage points. The variations between the peers serving content in the same channel can be accounted for by congestion and differences in signaling traffic. In fact, an altruistic peer offering to replicate traffic receives in return a large volume of signaling traffic.

As in [128], we defined a similarity metric in order to evaluate the breadth of the peer and AS population that we measured. For a vantage point in a channel, the metric was defined as the ratio of IPs/ASNs that overlap with those encountered in traces from other vantage points. For instance, in the case of the trace *ch1-california*, we observed a total of around 19k IPs and 1.4k ASNs which possess a similarity of 93% and respectively 98%. Overall, the similarity for IPs seldom drops under 85% and for ASNs never drops under 93%. The high IP similarity values indicate that in each channel our vantage points exchanged traffic with a large fraction of the peer population, leading to an accurate aggregate view of the whole overlay. Furthermore, the AS similarity values suggest that we have a precise estimate of the ASes exchanging traffic in all the channel overlays. This is also confirmed by the similarity of the curves describing the distribution of the clients in ASes (see Fig. A.2).

If we aggregate the traces pertaining to each channel and perform the same analysis we observe that there is little overlap between the peer IPs. However, we do observe high values for the ASN overlap with the outlier being $ch1$ due to its much larger client population. Overall, we can infer that channels generally have non-overlapping clients (as expected) however, their clients pertain to autonomous systems that have a larger overlap.

Differences between channel client population sizes are explained by differences in popularity between the channels. From our dataset we deduced that the streaming bitrate and the language are two important factors to influence a channel's popularity. For instance, $ch1$ and $ch4$ are fairly popular due to their better streaming quality whereas $ch2$ and $ch3$ raise a lower interest. Moreover, the fact that Romanian was the language used in $ch3$ explains the lower number of clients.

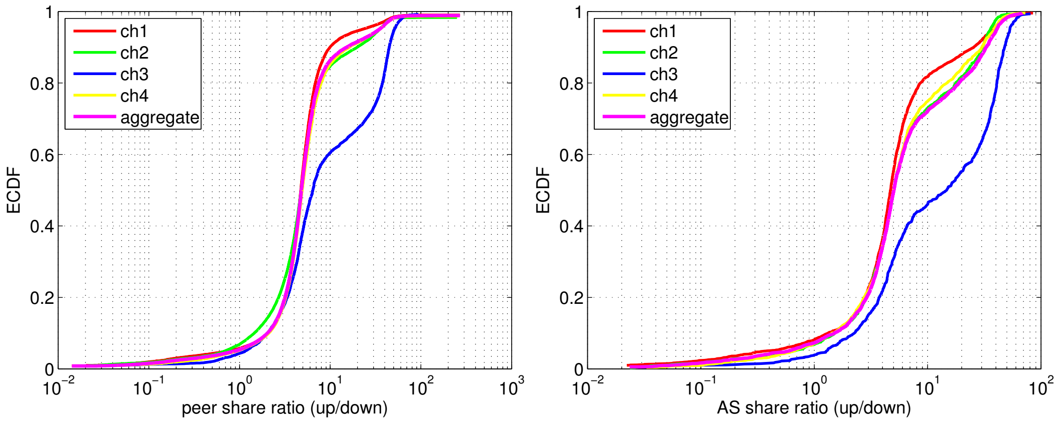
A.2 Distribution of clients in ASes

The distribution of clients in ASes is depicted in Fig. A.2 as both cumulative distribution function (CDF) and complementary cumulative distribution function (CCDF). Additionally, we have added a curve depicting the distribution of clients in ASes for the aggregated clients sets of all channels. It can be seen that the plots have a similar shape and the differences between them are only due to the inter-channel client variations. This also holds for the aggregate distribution which is slightly different from the curves pertaining to $ch1$ and $ch4$ and which account for the largest client populations.

As we have seen in the previous section, channels tend to have non-overlapping client populations. Therefore the reasons behind the similarity of the curves have to do with a more subtle phenomenon probably related to user behavior and localized user interest. Figure A.2a depicts the CCDFs of the distributions and from them we can deduce that clients roughly distribute in autonomous systems according to a power law. The reasoning being that power laws have as CCDF representation a straight line.

A.3 Collaboration between peers

Within P2P systems, it is the responsibility of the peers to replicate content to other members. This, in fact, being the fundamental requisite for the scaling of P2P overlays. In this section we study the amount of traffic exchanged by our nodes



(a) Peer share ratio (b) AS share ratio

Fig. A.3 Peer and AS share ratios

with their peers. To evaluate their level of collaboration, we define and compute for each overlay member a *sharing ratio*. Specifically, for each peer, the sharing ratio is computed by dividing its volume of uploaded traffic by the download one.

For all channels, Fig. A.3 shows the CDF of the sharing ratios (upload/download) observed at our nodes at peer and AS level. The results are aggregated for channels and then for the whole experiment. A ratio larger than 1 means that one of our nodes has acted altruistically with a peer (it provided more data that it requested) whereas one lower means that the peer has provided content to one of our nodes. Of course, a value of 1 means that the peer is in traffic balance with one of our nodes.

Save for *ch*3 our nodes in all channels behave similarly and this holds also for the global (aggregate) traffic exchange. We speculate that the curve for *ch*3 has a different shape in the zone of larger share ratios due to the lower number of peers in the overlay and a higher need for nodes with high upload capacities.

In absolute terms, the plots show that our nodes acted as seeds for the overlay as they typically obtained the content from under 5% of their peers and uploaded content for 95%. Similarly, at AS level content is downloaded from around 7.5% of the peer ASes and replicated towards the rest of 92.5%. The fact that for all vantage vantage points their AS curve grows slower than the IP share ratio indicates that our nodes exchange large amounts of traffic with several nodes within the same AS. This is suggestive of an inefficient intra-domain replication of traffic, due to a lack of peer collaboration.

A.4 Peer and content locality

In order to understand the traffic requirements for ISPs involved in SOPCast overlays, we investigated the inter-AS traffic exchanges. Figure A.4 depicts the distribution of both upload and download traffic for vantage points in $ch1$ and $ch2$.

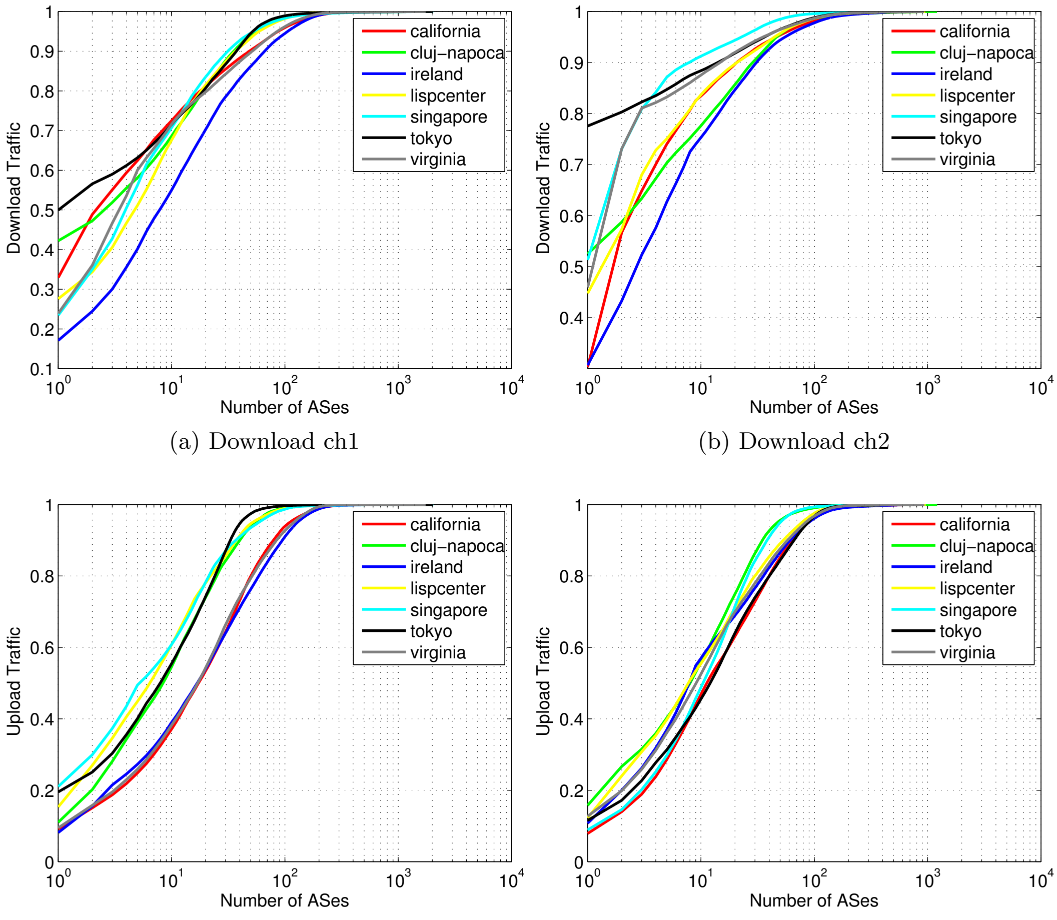
Although the number of ASes with contacted peers is above $1k$ for $ch1$ and above 700 in the case of $ch2$, both for upload and download, the greater part of traffic is exchanged with a restricted AS subset. Nevertheless, its size is not small. In the case of $ch1$, on average, more than 50 AS were contacted to download 90% of the streamed content. Similarly, the vantage points receiving $ch2$, to obtain 90% of the content, had to connect to more than 20 ASes. For both channels, the vantage points replicate the content to a slightly larger set of ASes than the one from which they had obtained it. However, it is interesting to note here that two different kinds of seeding behaviors were observed. They are easily distinguishable in Fig. A.4c for $ch1$ where we see that a first group of the vantage points replicate content towards a limited number of ASes, actually comparable with the one from which they received their content. The second group replicates content to a larger set of ASes however still smaller than 100 if we consider the destinations of 90% of the traffic. This behavior is not that easily distinguishable in the case of $ch2$ except as the number of ASes increases.

Overall, we could conclude that the number of ASes towards which a large part (90%) of the content gets replicated is only slightly larger than the one from which content is obtained. Then, if we consider the large replication factors of each vantage point seen in Table A.1 we can infer that our nodes generally replicate content to a large number of peers in each AS. A rather inefficient use of inter-AS bandwidth that is probably caused by the inability of nodes to identify close-by (same AS) peers.

A.4.1 Geographic location of peers

To better understand the criteria based on which traffic is replicated between peers and ASes, we studied the geographic distribution of participating hosts and the volume of traffic exchanged at country level. ASes were mapped to their corresponding country with the help of the cymru [1] whois service. Although an AS can span many countries, for our intended purposes this mapping was accurate enough.

Figures A.5 and A.6 show our results for $ch1$ and $ch2$. We did not distinguish between peers used for uploading and those for downloading as these two groups tend to have a high percentage of overlap. Due to readability constraints, the stacked



(a) Download ch1

(b) Download ch2

(c) Upload ch1

(d) Upload ch2

Fig. A.4 Inter-AS traffic

histograms present just the first 20 countries, when ranked by number of clients. The legend entry *OTHER* stands as an aggregate for the rest of the countries that could not be shown.

The country with the largest population in both channels is Germany. This is somewhat unexpected as both football teams involved were Spanish. In fact, Spanish peers make up just a small part of the *ch*1 overlay and are not present in the top 20 of *ch*2. Additionally, at no vantage point did we observe any sort of biasing of the peer population towards the country where the measurement was performed. In effect, the *Peer* bars, in both figures, indicate that all vantage points observe almost the same distribution of peers in countries. This corroborates the conclusion that our nodes had an accurate view of the channel population.

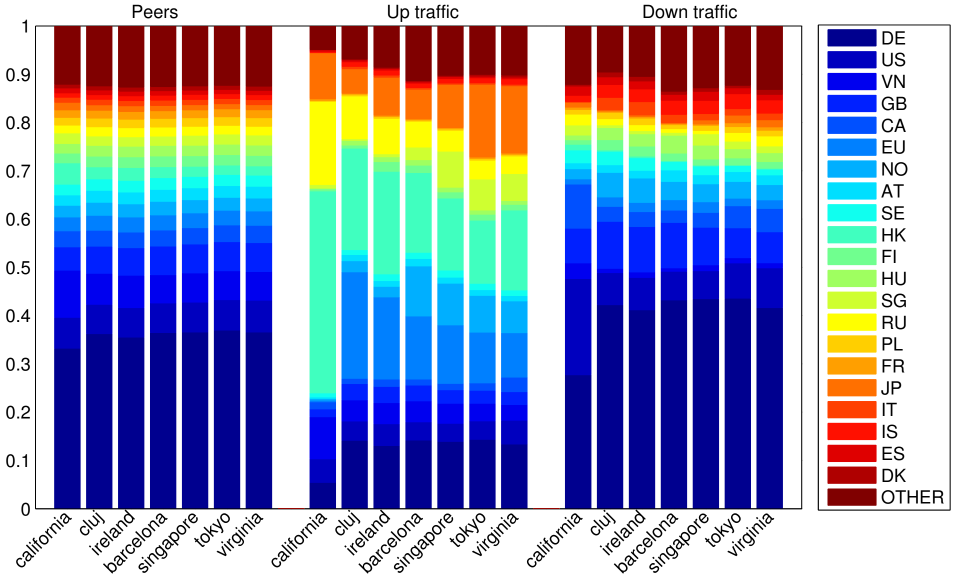


Fig. A.5 Geographic location of peers in ch1

Surprisingly, the *Download traffic* bar plots are also very similar although, with more important differences than the *Peer* ones. This suggests that the vantage points are completely unbiased (from geographic perspective) in their choice of upstream peers. For $ch1$, traffic is mainly obtained from a handful of countries in Northern Europe and America: Germany, USA, GB, Canada and Iceland. Similarly, a large part of the content for $ch2$ is obtained from a limited set of countries: Germany, GB, Hungary, Egypt, France. As it may be seen, again the majority are placed in Europe in spite of the fact that the channel is of Chinese origin.

Similarities between the upload patterns of the vantage points are not as obvious as the one for download but they are still discernible. For $ch1$, save for the fact that the vantage point in California uploaded little content to EU (ASes registered with the European Union country code) and Norway customers, the typical node behavior is to upload mainly to clients in Germany, Hong Kong, EU, Norway and Italy. The fact that traffic is resent to Germany after it was mainly received from there, speaks again against the efficiency of the routing which does not seem to be aware of the peer locality. The bouncing between the countries is even more detrimental if one considers that the content is delay sensitive. Same observations hold for $ch2$ just that the destination countries are different. The predilect destinations for the uploaded traffic were China, Hong Kong and Japan. A considerable amount of content has been uploaded to destinations with few peers and from which a low volume of content

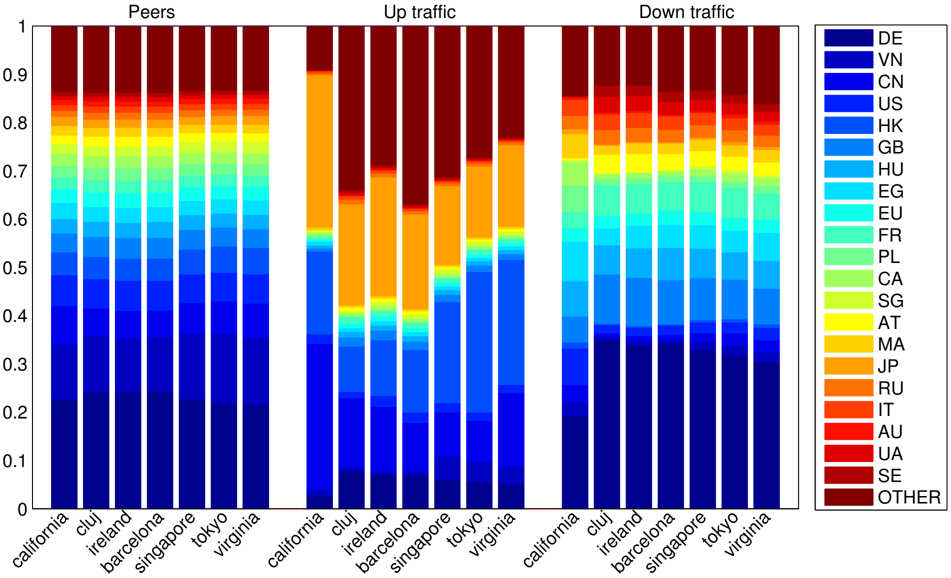


Fig. A.6 Geographic location of peers in ch2

has been downloaded.

In summary, the analysis has shown that the routing in the overlay is not aware of peer locality. As a result, nodes interact in a similar fashion, in this sense, our nodes observe the same distribution of peers in countries, obtain their content from the same sources and roughly upload to the same destinations.

A.5 Conclusions

Throughout the study we quantified several relevant parameters for live streaming overlays. From traffic volume perspective we characterize both global and peer level exchanges and provide information about upload/download ratios and respectively peer share ratios. Furthermore, we also analyze and provide information about traffic locality at AS level. Finally, we study the geographic location of peers and the traffic exchange at country level.

Our results indicate that SOPCast does not consider peer geographic location and generally exchanges traffic arbitrarily between a large set of ASes. As a result the same content may be exchanged multiple times between two ASes and intra-domain traffic exchanges are not prioritized. We did not observe a correlation between the streamed traffic and the geographical location of clients and the application seemed to be used world-wide indiscriminately of continent.

Appendix B

Complete List of Publications

B.1 Related Publications

- F. Coras, A. Cabellos-Aparicio, J. Domingo-Pascual. An Analytical Model for the LISP Cache Size. *In Proc. of IFIP-TC6 Networking*, Prague, Czech Republic, May 2012

- F. Coras, D. Saucez, L. Jakab, A. Cabellos-Aparicio, J. Domingo-Pascual. Implementing a BGP-Free ISP Core with LISP, *IEEE Global Communications Conference (GLOBECOM)*, Anaheim, USA, December 2012

- F. Coras, J. Domingo-Pascual, F. Maino, D. Farinacci, A. Cabellos-Aparicio, Lcast: Software- defined inter-domain multicast, *Elsevier Computer Networks*, vol. 59, pp. 153- 170, February 2014

- F. Coras, J. Domingo-Pascual, D. Lewis, A. Cabellos. An Analytical Model For Loc/ID Mappings Caches, *IEEE/ACM Transactions on Networking* (To appear)

Under Submission

- F. Coras, J. Domingo-Pascual, A. Cabellos. On the Scalability of LISP Mappings Caches, under review

Internet Drafts

- F. Coras, A. Cabellos, J. Domingo-Pascual, F. Maino, D. Farinacci. LISP Replication Engineering, draft-coras-lisp-re (Work in progress)

Technical Reports

- F. Coras, J. Domingo-Pascual, D. Lewis, A. Cabellos-Aparicio. An Analytical Model for Loc/ID Mappings Caches, *CoRR arXiv:1312.1378*, December 2013
- F. Coras, T. Silverston, J. Domingo-Pascual, A. Cabellos-Aparicio. A Measurement Study of SOPCast. Tech. Rep. UPC-DAC-RR-CBA-2012-2, Universitat Politécnica de Catalunya, 2012

Talks

- *"Lcast: LISP-based Single-Source Inter-Domain Multicast"*, 83rd IETF Meeting, LISP Working Group, Paris, France, March 2012
- *"LISP Replication Engineering"*, 84th IETF Meeting, LISP Working Group, Vancouver, Canada, August 2012
- *"LISP Replication Engineering"*, 89th IETF Meeting, LISP and MBONED Working Groups, London, United Kingdom, March 2014

Code

- I am one of the LISPmob maintainers, my main contribution being the restructuring of the code to enable better modularization. This has mainly led to a better decoupling of control and data plane functionalities, the implementation of RTR and MS LISP devices and the implementation of *liblisp*, a library for parsing LISP control plane messages. `https://github.com/LISPmob/lispmob`
- My implementation of LISP-RE in LISPmob, `https://github.com/LISPmob/lispmob/tree/lisp-re`

B.2 Other Publications

- 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
- L. Jakab, A. Cabellos-Aparicio, F. Coras, D. Saucez and O. 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

- L. Jakab, A. Cabellos-Aparicio, T. Silverston, M. Solé, F. Coras and J. Domingo-Pascual. CoreCast: How Core/Edge Separation Can Help Improving Inter-Domain Live Streaming, *Elsevier Computer Networks*, vol. 54, no. 18, pp. 3388-3401, December 2010

- D. Papadimitriou, F. Coras and A. Cabellos, Path-vector Routing Stability Analysis. *SIGMETRICS Performance Evaluation Review*, vol. 39, 2011

- D. Papadimitriou, F. Coras, A. Rodriguez, V. Carela, D. Careglio, L. Fàbrega, P. Vilà, and P. Demeester. Iterative Research Method Applied to the Design and Evaluation of a Dynamic Multicast Routing Scheme, *Lecture Notes in Computer Science*, vol. 7586, pp. 107–126, Springer, 2012

- D. Papadimitriou, A. Cabellos, and F. Coras. Stability Metrics and criteria for path- vector routing, *International Conference on Computing, Networking and Communications (ICNC)*, January 2013

- F. Coras, D. Saucez, L. Iannone, B. Donnet. On the Performance of the LISP-Beta Network, *In Proc. of IFIP-TC6 Networking*, June 2014

Requests for Comments

- L. Jakab, A. Cabellos-Aparicio, F. Coras, J. Domingo-Pascual and D. Lewis, LISP Network Element Deployment Considerations, RFC 7215

Internet Drafts

- D. Saucez, L. Iannone, F. Coras, LISP Impact, draft-saucez-lisp-impact (Work in progress).