Future Internet Services based on LISP Technology

Yue LI

Department of INFRES
Telecom Paristech

This dissertation is submitted for the degree of
Doctor of Philosophy

Acknowledgements

Abstract

The *Locator/Identifier Separation Protocol (LISP)* is proposed in 2006 to initially address Internet scalability issues. It is based on a map-encap mechanism to split the *who* (*Endpoint IDentifier (EID)*) and the *where* (*Routing LOCator (RLOC)*) of the current IP addresses. A new network entity called the (*Mapping Distribution System (MDS)*) is introduced to associate RLOCs with EIDs. LISP leverages on a proxy (*Proxy Ingress Egress Tunnel Router (PxTR)*) to communicate with the legacy Internet. Thanks to the Loc/ID separation paradigm, LISP also brings the benefits to the seamless mobility. Although LISP is currently under standardization in IETF and is deployed in the wild by two testbeds at the same time, it is still young. It lacks the thorough measurement works to show its realistic performance in the large-scale networks and to improve itself.

In this dissertation, we assess LISP from the different aspects : (i) the measurements on the MDS: we continuously measured the MDS for seventeen days. The results show that it is stable and consistent over most of time. Nevertheless, instability and inconsistency are observed although they are rare events. We define a new taxonomy to classify them so to facilitate the deeper analysis. (ii) Proposing a comprehensive monitor to supervise the MDS: as these instabilities and inconsistencies have never been observed, we propose a more comprehensive LISP monitoring architecture to dynamically supervise the whole MDS. After one full month of testing, our proposed monitor provides more mapping information, and can show the MDS performance by various metrics. (iii) The assessment of LISP interworking performance with legacy Internet: we also evaluate PxTR by conducting two experiments in the different years. Both results show that LISP is stable although the stretch of PxTR is introduced. The PxTR brings the negative effects for the short trips, but can be ignored for the intercontinental long-distance transmissions. Besides, the position of PxTR either near to the sources or the destinations can decrease the latency a lot. (iv) The evaluation of LISP mobility: we analyze the LISP mobility performance in three scenarios. We calculate the bounds of handover for the different components implementing LISP. Each scenario has its own advantages and shortcomings. Besides, to facilitate the researchers to verify their proposals or test the new features on LISP mobility, we implement LISP mobility extensions under ns-3.27 on an existing simulator.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Motivations

Since the ARPANET (Advanced Research Projects Agency Network) emerged in 1966 [85], the Internet has obtained significant achievements over the last fifty years and developed very fast. At the beginning of 2018, the Internet users in the world are 3.82 billions, i.e., around 40% of the world population has an Internet connection today [10]. Moreover, we come into the era of the mobile Internet. According to the latest research [29], the number of global mobile users has been higher than that of desktop since 2014. Although having a huge success, today's Internet is suffering the overloading of IP address semantic, which results in many shortcomings in the application landscape [54]. In the current TCP/IP protocol stack, the IP address serves both as the *who* to identify the host and the *where* to indicate the attachment point in the network topology, leading to a serious routing scalability problem. As shown in Fig. 1.1, the BGP table size, which is the number of IP prefixes in the BGP (Border Gateway Protocol) routing table of DFZ (Defaut Free Zone) routers, has grown at a super-linear rate in recent years. After full use of IPv6, the routing scalability issue will be aggravated, since it has much larger addressing space than IPv4. Maintaining such routing table is very cumbersome, since once a site changes its routing information, all the routers need to update, which causes a huge BGP churn. In addition, frequent routing information updates result in the increase of convergence time and packet loss rates.

To cope with the current Internet scalability problems, the Loc/ID separation paradigm is proposed, which separates the single IP addressing space into two orthogonal spaces. Many solutions based on this mechanism are proposed, which can be classified into two categories: *Host-based* and *Network-based*. The former one needs to modify the host stack. The representative architectures are: HIP (Host Identity Protocol) [95] [93] [92], Six/One [117], SHIM6 (Site Multihoming by IPv6 Intermediation) [62] [96], ILNP (Identifier-Locator Net-

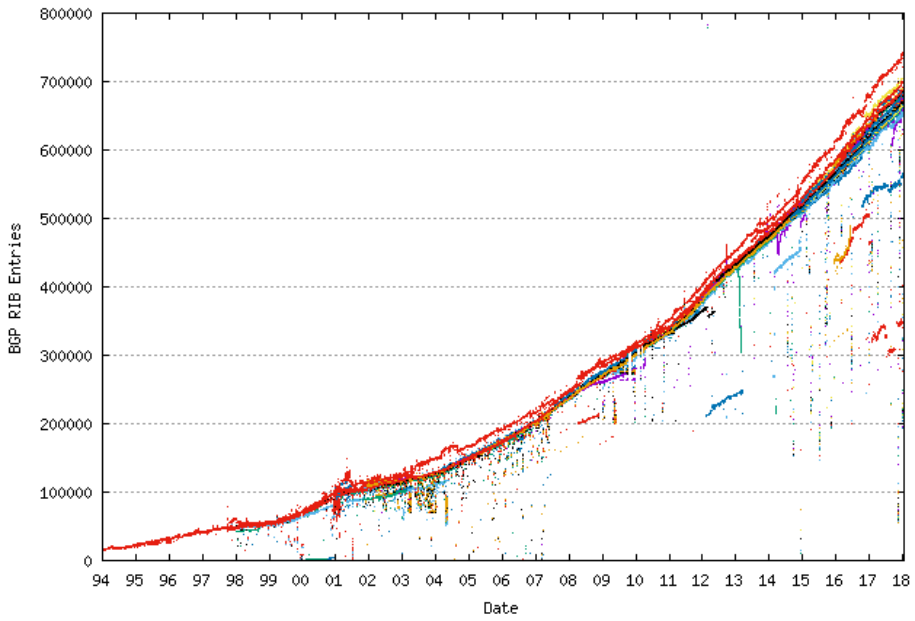

Fig. 1.1 Growth of the BGP Table - 1994 to Present [6]

work Protocol) [38] [35], and LNA (Layered Naming Architecture) [36]. The network-based approach needs to update the routers, and the representative proposals are: GSE (Global, Site, and End-system address elements) [97], LISP (Locator/ID Separation Protocol) [48], and TIDR (Tunneled Inter-domain Routing) [31]. Among them, LISP is outstanding by its relatively comprehensive network systems in both control and data planes. LISP was initially proposed by Cisco in 2006 and has an exclusive IETF WG since 2009.

The key philosophy of LISP is to separate the IP addressing space into two sub-spaces: the *Routing LOCators* (RLOCs) and the *Endpoint IDentifiers* (EIDs). End-to-end communication is based on EIDs, which in an intra-domain context means just performing conventional routing and forwarding. On the contrary, for inter-domain end-to-end communication, RLOCs are used for routing and forwarding. To this end, LISP does a map-and-encap operation, first mapping EIDs to RLOCs (through a DNS-like on-demand Mapping Distribution System (MDS)[1] [56]) and then encapsulating the original packet so to use RLOCs on the outer header. Internetworking between LISP and non-LISP networks is performed through two types of proxies [77]: *Proxy Ingress Tunnel Routers* (PITRs) and *Proxy Egress Tunnel Routers* (PETRs). They are collectively denoted by *PxTR*. PITRs advertise in the BGP routing infrastructure the EID-prefix space on behalf of the LISP sites, so that non-LISP sites can reach them. PETRs are used as exit points from LISP sites when the destination

[1]The terms *Mapping Distribution System (MDS)* and the *mapping system* are used interchangeably in this dissertation.

is not an EID in a LISP site. With help of this mechanism, LISP is able to improve the Internet routing scalability, enable enhanced inter-domain traffic engineering, lightweight multi-tenant carrier-grade VPN, and VMs mobility in Data-Centers [111].

As LISP is a promising technology that brings many benefits to the Internet, it attracts the attention from both academic and industrial world. Many companies, universities and research institutes are now involved in the development of LISP. Hence, the related standardization and research are very active. In terms of deployment, two large-scale experimental LISP platforms: LISP Beta Network and LISP-Lab platform are deployed on the world respectively in 2008 and 2015. Up to date, all the efforts are mainly about theoretic or simulation-based performance analysis, description of LISP architectures, implementations, extensions, applications and so on. However, it lacks of the comprehensive measurement works for LISP technology itself. These performance measurements work will give the research community a clear view about the LISP performance, reveal the points that can be improved, indicate possible research directions, and improve the performance of two platforms. Thus, the large-scale measurements in realistic networks are of paramount importance for the LISP development.

1.2 Contributions

With the goal of evaluating LISP from the different views, in this dissertation we aim at answering the following research questions, which have not been tackled before:

1. Mapping system is essential in LISP, because it decides where the packets will be routed. Thus, does it always provide the identical mapping information for the same destination? Do all of the Map-Resolvers (MRs) of the mapping system offer the same response at the same time? Does the mapping information replies keep consistent if we query them from the different Vantage Points (VPs)?
   We continuously measured LISP Beta Network for seventeen days. Measurements show that the LISP mapping system is stable over most of time. The replies from the different MRs and VPs are also mostly consistent. Nevertheless, instability and inconsistency are observed although they are rare events. We define a new taxonomy to classify these instabilities and inconsistencies so to facilitate a deeper analysis. This work is published in WNM [82].
2. As the results of the last question show that there exists inconsistency between the MRs and VPs, is it possible to implement a comprehensive monitor so to simultaneously supervise the whole mapping system and all the resolvers?

We propose a dynamic LISP monitoring architecture, namely LISP-Views, so to deepen the understandings on LISP and to ease day-to-day operations and troubleshooting. After one full month monitoring the whole LISP mapping system, it demonstrates that LISP-Views provides the complete mapping information. Furthermore, with our proposed monitoring platform, more mapping system performance metrics such as reliability, latency, and configuration issues can be assessed, which helps for further LISP improvements. This work is published in ITC [78].

3. By using proxy approach to communicate with the legacy Internet, a stretch in the path is introduced. How much is it in the real world performance when the platforms integrate with the legacy Internet? Under which conditions such stretch has an important impact on the performance? Conversely, under which situation such overhead is so small that it can be ignored?

   To evaluate LISP interworking performance with legacy Internet, we conduct two measurement campaigns. The results of the two experiments are coherent. Both of them show that LISP is stable although the stretch is introduced due to the use of proxies. We find that the proxy indeed introduces negative effects for the destinations located in Europe and America, but can be ignored for the intercontinental long-distance transmissions to Asia destinations. Besides, we observe that the position of proxies is very important. If it is either near to the sources or the destinations can decrease the latency a lot. This work is published in INFOCOM Student workshop [80], ACM SIGCOMM Student workshop [79], and Elsevier Computer Networks [81].

4. As LISP separates the identifier and the topological attachment point from the traditional IP address, it is able to maintain the communication between two hosts during the handover. LISP can be implemented on the border routers or the terminals or even both of them. Thus, which network element supports LISP showing a better performance, i.e., having a shorter handover delay or introducing less overhead during the mobility? What are the advantages and shortcomings of each case?

   To explore the mobility performance of the LISP implementation on each network element, we design three different scenarios: 1) Only mobile node supports LISP; 2) Only border routers support LISP; 3) Both mobile node and border routers support LISP. The numerical analysis shows that the 1st and 3rd scenarios can achieve the mobile node roam through the different subnets, whereas they require additional permanent EID for each mobile node. The 1st has no help on reducing the BGP table size and the 3rd scenario shows much longer handover delay than the other two's. The 2nd scenario presents the smallest handover delay and needs the least IP addresses, but it can only handle mobility within the same subnet. To verify the numerical anal-

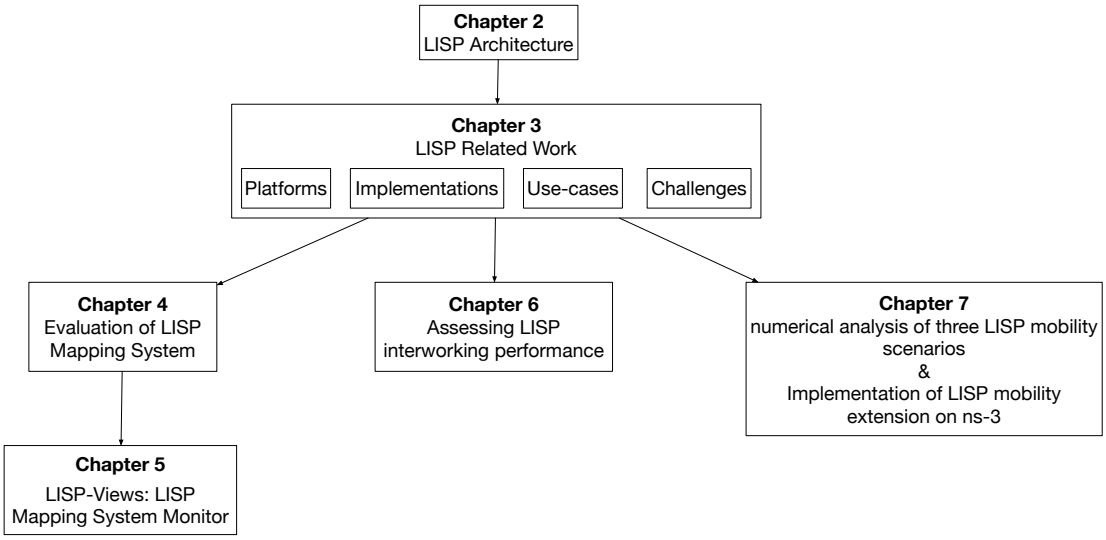

Fig. 1.2 Structure of this dissertation

ysis, we implement a LISP simulator with LISP mobility extensions under ns-3.27 by extending an existing ns-3 based LISP implementation.

1.3 Manuscript organization

The remainder of this dissertation is organized as follows and shown in Fig. 1.2:

- *Chapter 2*: Describes the basic LISP mechanisms, presenting how the packets are exchanged between two LISP-sites and with the legacy Internet. It also shows the different scenario of LISP mobility.
- *Chapter 3*: The challenges brought by LISP, LISP-related studies and missing works are also illustrated. Introduces the LISP current status, such as: the available LISP platforms, the first LISP monitor, the LISP manual management tool, the open source LISP implementations and LISP use-cases.
- *Chapter 4*: This chapter is about the evaluation of LISP Mapping System. In this chapter, we observe that if the mapping system always provides the same mapping information over time, and if we can obtain the identical mapping answers from the different network entities.
- *Chapter 5*: Describes the proposition of LISP-Views: a LISP Mapping System Monitor at large scale. In this chapter, we introduce how this architecture is implemented, how it has been validated and what experimental results it can provide.

- *Chapter 6*: Assesses LISP interworking performance through RIPE Atlas. This chapter presents how the experiments are conducted, how much is the stretch introduced by proxy in the real world, and whether this stretch have a big impact on the performance or it can be ignored.
- *Chapter 7*: Introduces ns-3 Implementation of LISP mobility extension and numerically analyses the performance of different LISP mobility scenarios. In this chapter, we first describe how to implement the LISP mobility extension on the ns-3. Then we define three different LISP mobility scenarios, provide the numerical analysis of each scenario, and compare the advantages and shortcomings between them.
- *Chapter 8*: Concludes the dissertation and discusses perspectives for further work.

Chapter 2

LISP Overview

In this Chapter, the LISP principles and its mechanisms to which we refer to in the reminder of the dissertation are introduced. The basic architecture of LISP is introduced by presenting its Data Plane and Control Plane. We detail how the traffic between two LISP-sites is exchanged by using an example. A new network element, which is used to allow communication between LISP-site and the legacy Internet, is also presented, as well as the sequence of packets exchange between them. Then, two mechanisms used to update the LISP mapping cache are illustrated. Finally, this chapter ends with the explanation of LISP mobile node, i.e., a LISP-enabled mobile terminal, and the introduction about how the LISP operations are performed when the LISP-enabled node is behind a LISP-speaking router.

2.1 LISP Architecture

The *Locator/Identifier Separation Protocol (LISP)* separates the traditional IP address role into two logical sub-spaces: (i) *Endpoint IDentifier (EID)* and (ii) *Routing LOCator (RLOC)*. Both of them are the conventional IPv4 (32-bit) [105] or IPv6 (128-bit) [46] addresses. The EID is the identifier of hosts, which is locally routable in the LISP-site. Normally they are the addresses of source and destination hosts. Within the same LISP-site, the hosts can directly communicate with each other via EIDs as in a legacy IP network. The host gets a destination EID in the same way it obtains the destination IP address today, for instance through a Domain Name System (DNS) [89] lookup or Session Initiation Protocol (SIP) [107]. RLOCs denote the attachment points of EIDs in the Internet topology, i.e., the source/destination address of the edge routers, behind whom the EIDs can be found. RLOCs addresses are used to forward the packets in the core Internet.

When packets are transferred between two different LISP-sites, an encapsulation on the edge router is needed: the source and remote destination EIDs are in the inner packet

header, while the source and destination RLOCs are in the outer header. The EID-prefixes are not necessary to be announced on the Internet core, so the BGP routing table size can be reduced. Similarly, in stub networks, it is not necessary to know the routing information of core Internet any more. The aforementioned encapsulation and forwarding operations are the responsibilities of LISP Data Plane. More details are introduced in Sec. 2.1.1. To bind the EIDs and RLOCs, a new network entity named Mapping Distribution System (MDS) is used, whose front end is consisting of Map Resolver (MR) and Map Server (MS) [57]. The MDS is not only able to provide the mapping information between the EIDs and RLOCs for the LISP-sites (we simply call them LISP Map-Replies), but also indicate which addresses are not LISP sites and have no RLOCs, meaning that are part of the legacy Internet (these are called Negative Map-Replies). The Control Plane is described in Sec. 2.1.2.

2.1.1 LISP Data Plane

The LISP Data plane mainly takes care of the encapsulation and decapsulation operations done at the source and destination networks. LISP basically provides a level of indirection through a tunneling mechanism over the core Internet (the so called *Default-Free Zone* – DFZ), in order to provide end-to-end communication. More specifically, any host willing to communicate with another host residing at remote LISP-site, uses its own EID as source address and the EID of other host as destination address to generate regular IP packets. These packets are first transferred as usual to the border router, called *Ingress Tunnel Router (ITR)*. ITR performs a mapping lookup in its Cache, or in case of miss, it queries the MDS for the mapping information. Then the ITR encapsulates the regular IP packets into a LISP packet by adding a tunnel header [48], where the RLOC of the ITR is used as source address and the destination RLOC as destination address. After encapsulation, the LISP packets are forwarded over the core Internet. When arriving at the destination border router, called *Egress Tunnel Router (ETR)*, the LISP packets are decapsulated (i.e., the tunnel header is removed) and transferred to the destination host. A device which acts as both an ITR and an ETR is denoted as *xTR*.

Mappings are stored in two data structures on the xTRs: the *LISP Database* and the *LISP Cache*. The LISP Database is populated by configuration and stores all known EID-to-RLOC mappings, for which the EID-Prefixes are behind the xTR. On an ITR, this helps select a source RLOC used in encapsulation. While on an ETR, it allows to verify whether itself is the proper ETR connecting to the destination EID, so that such ETR is able to forward the decapsulated packets to the final destination.

The LISP Cache temporally stores mappings for the EID-prefixes of the remote communicating end-points. On an ITR, it is used to select the destination RLOC of the outer

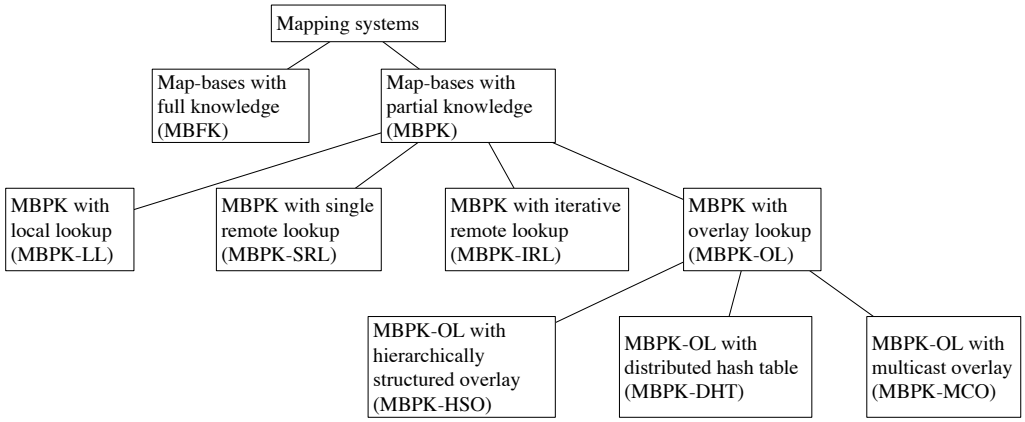

Fig. 2.1 Hierarchical taxonomy of mapping systems (from [66])

header of the LISP-encapsulated packet. While on an ETR, it is used to perform a basic anti-spoof verification. Different from the LISP Database, the LISP Cache is populated on demand. The procedure is triggered by the first packet of a new flow, which can not find a suitable mapping for the destination EID among the mappings stored in the LISP Cache. The mappings in the LISP Cache are purged if not used upon expiration of a time-out ([67, 73, 119, 44]).

2.1.2 LISP Control Plane

The LISP Control plane is mainly tasked to discover the associations of EIDs with RLOCs, i.e., find the *mappings*. A new network entity named *Mapping Distribution System (MDS)* is introduced by LISP to perform such EID-to-RLOC distribution. In particular, a mapping contains an EID prefix and an associated list of RLOC tuples: <*RLOC, Priority, Weight*>. The RLOC refers to the IP address of each xTR interface. Each RLOC is assigned a Priority. The RLOC with the highest priority is preferred. The weight is taken into account only to perform load balancing in the case of several RLOCs having the same priority.

The current BGP-based Internet routing architecture relies on a push model, in which routing information is pushed to the whole Internet. The Locator/Identifier Split-based Internet routing instead relies on a pull model, where the *Mapping Distribution System (MDS)* provides a mapping upon an explicit query,[1] where routing information is pulled only when

[1]The terms *Mapping Distribution System (MDS)* and *mapping system* are used interchangeably in this dissertation.

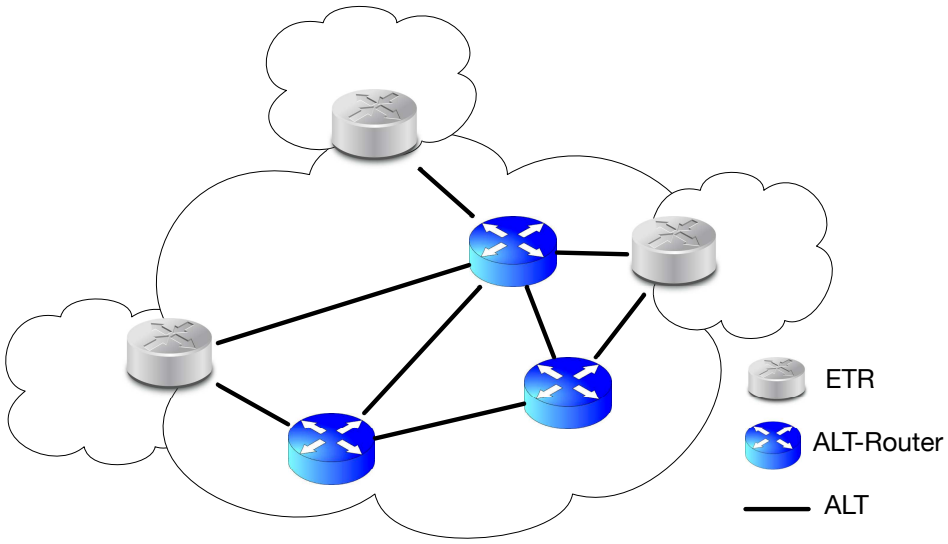

Fig. 2.2 Illustration of the LISP+ALT topology

actually needed. This paradigm shift requires making routing information available in an on-demand manner. Fig. 2.1 shows the current possible taxonomy of mapping systems [66]. So far, several mapping systems have been proposed for LISP, such as: LISP-TREE [70], LISP-NERD (Not-so-novel EID to RLOC Database) [76] and LISP-CONS (Content distribution Overlay Network Service for LISP) [40]. However, only two have been deployed: *LISP Alternative Logical Topology (LISP+ALT)* [58] and *LISP Delegated Database Tree (LISP-DDT)* [59].

LISP+ALT

LISP+ALT was the initial mapping system for LISP, where the ETRs store mappings they are authoritative for [108]. The basic idea is to use BGP to construct an overlay, named Alternative Logical Topology (ALT), to establish reachability between ETRs via tunnels (e.g., GRE tunnels [51]). As shown in Fig. 2.2, the routers in the ALT are called ALT-Routers. Each ALT–Router maintains a BGP session with its neighbor and announces the EID–prefixes it is authoritative for. These EIDs are then routable in the ALT. Note that the ALT–Routers only exchange aggregated EID-prefixes that can be reached through them between neighbors, but they do not contain the mapping information. To get a mapping, an ITR constructs a Map-Request for the queried EID by using its own RLOC as source address and the queried EID as destination address, then sends it to an ALT-Router. The Map-

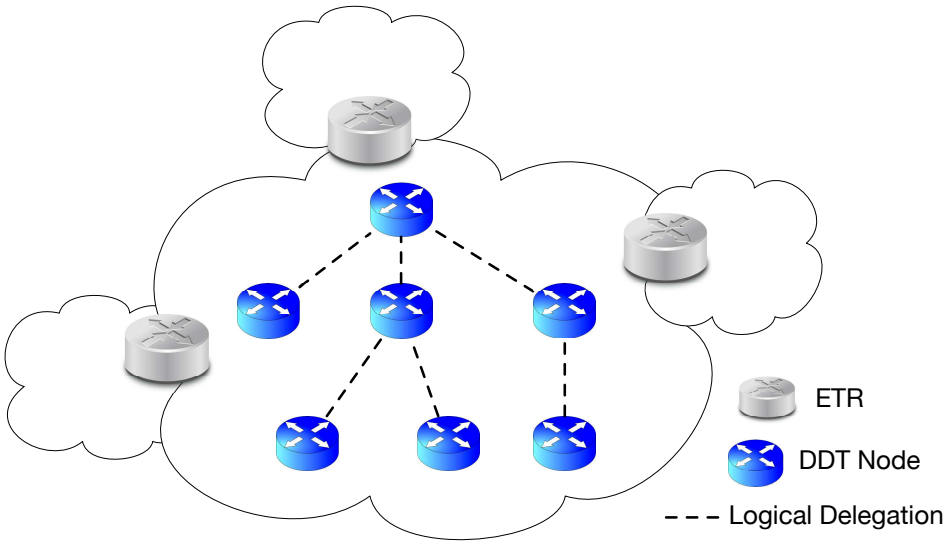

Fig. 2.3 Illustration of the LISP-DDT topology

Request is forwarded over the ALT and eventually reaches an originator ETR for the EID-prefix that matches the destination EID. This ETR resolves the Map-Request and directly sends back a Map-Reply to the inquirer ITR, this time not using the ALT. LISP-ALT belongs to the Map-Bases with Partial Knowledge using Hierarchically Structured Overlay (MBPK-HSO) in Fig. 2.1.

LISP-DDT

LISP-DDT is a hierarchical distributed database that embodies the delegation of authority to provide mappings from LISP EIDs to RLOCs . Its hierarchical organization is shown in Fig. 2.3. It consists of *Map-Resolvers (MRs)* and *Map-Servers (MSes)* to provide a simplified "front end". The MR is a network infrastructure component that accepts LISP Encapsulated Map-Requests from an ITR and resolves the mapping using a DNS-like distributed database. The MS learns of EID-Prefix mapping entries (a list of EID-Prefixes plus a set of RLOC tuples) from ETRs. The hierarchy is maintained as a tree where a root server is responsible for the entire EID space. This space is divided into several portions and each one is managed by one of the root's children. Mapping information is only stored at the tree leaves, which are made of MSes. Intermediate nodes only maintain pointers to their children.

If an ITR cannot find a mapping in its LISP Cache for a new flow, it sends a query, called *Map-Request*, to a MR [57]. The MR determines whether the queried destination IP address is an EID. If not, a *Negative Map-Reply* is directly returned; otherwise, the root server is queried. The root replies with a pointer to its child node responsible for the queried EID. The process is recursively repeated to find the child root of the sub-tree where a mapping can be retrieved for the queried EID. When the leaf is reached, the mapping is then retrieved by sending a Map-Request to the ETR authoritative for the matching EID-prefix. The ETR in turn directly replies a *Map-Reply* message containing the whole requested mapping information to the ITR initiating the query.

When an ETR first contacts an MS after restarting or changing its database, it sends the Map-Register messages to the MS to publish its EID-prefixes as well as the mapping information. For maintaining the association, the registration interval should be expanded to at least 1 minute. If the MS has not received a valid Map-Register within the past 3 minutes, it removes the registration of this ETR. If the "want-map-notify" (M-bit) flag is set in Map-Register, the received MS needs to send back a Map-Notify message to inform the ETR that the Map-Register has been received and processed. If the "proxy Map-Reply" flag (P-bit) is set, the MS answers the Map-Requests to the queried ITR by providing the mapping information on behalf of the ETR. MRs offer an interface to the MDS for the xTRs, so that the whole complex MDS mechanism can be hided.

2.1.3 Communication between two LISP-sites

We use Fig. 2.4 as an example to describe how the LISP packets are processed in details in the case of the packets exchange between two LISP-sites.

When the host in the AS_s communicates with the host in the AS_d, it uses EID_s as source address and EID_d as destination address. After the conventional IP routing to xTR_1 (the function of ITR and ETR are combined together in Fig. 2.4) labeled as *"1"*, ITR_1 first checks in its mapping cache [67] to find out the association between EID_d and its RLOC. If there is no record, it sends a Map-Request to Map-Resolver (MR), labeled as *"2"*. MR searches which Map-Server (MS) stores the mapping information of EID_d and forwards the Map-Request to that MS. If MS is authoritative (i.e., acts as a proxy), it directly returns back a Map-Reply containing the mapping information (this procedure is not labeled in the figure). If not, after forwarding the Map-Request within MDS with the procedure described in Sec. 2.1.2 to MS and then to the destination side (label *"3"*), xTR_1 receives a Map-Reply from one of ETRs of EID_d (label *"4"*). If the Map-Reply shows that the priority of ETR_3 is higher than ETR_4, then the ITR_1 encapsulates the packets by adding $RLOC_{ETR_3}$ as the destination address and $RLOC_{ITR_1}$ as the source address in the outer header and sends the

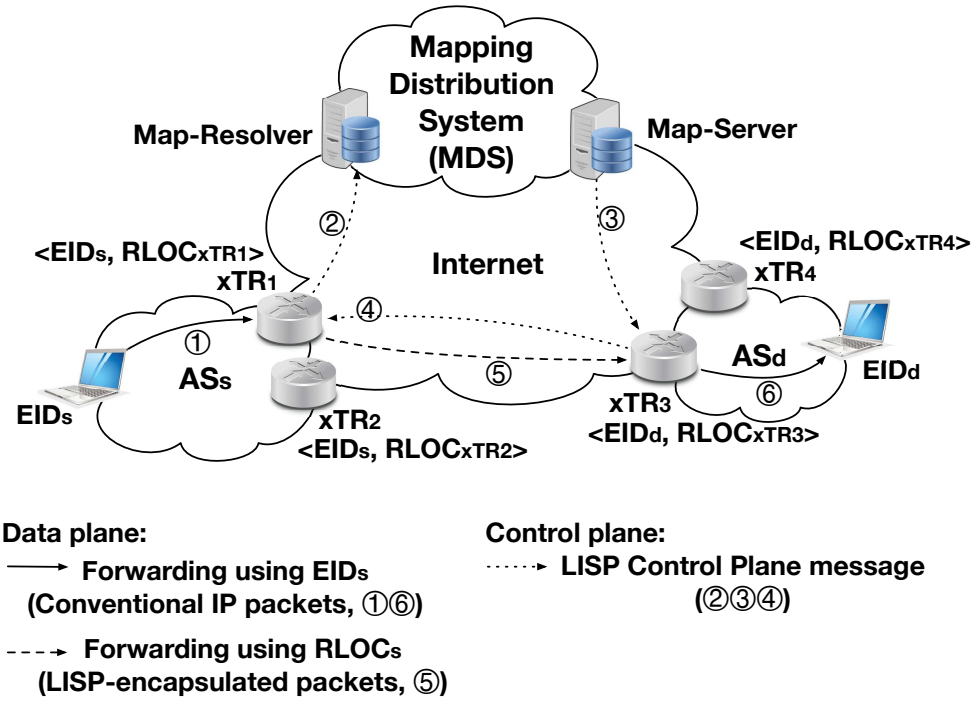

Fig. 2.4 LISP architecture overview

encapsulated packets through the Internet as labeled *"5"* in Fig. 2.4. Once ETR_3 receives these packets, it decapsulates and forwards them by using the original IP packets (label *"6"*). If the destination site is not a LISP-site, the MR directly returns back the Map-Reply without mapping information to ITR_1.

Based on LISP architecture, there are three types of outcomes for a query:

- *LISP Map-Reply*, means that the queried IP address belongs to a LISP site (i.e., EID) and the Map-Reply contains the mapping information for this site, including: the association between the EID-prefix that queried EID belongs to, and the list of RLOC tuples <RLOC, Priority, Weight> of the destination site.
- *Negative Map-Reply*, means that the prefix covering the queried IP address belongs to a non-LISP site (i.e., conventional site), and the Map-Reply contains no mapping information. We consider these two kinds of replies as successful queries.
- *No Map-Reply*, means that the xTR does not receive any reply during a certain time. According to the standard described in [48], the time-out is set to 3 seconds. But this value can be modified in the implementation to adapt the experiment needs. In this case, we consider the query as failed.

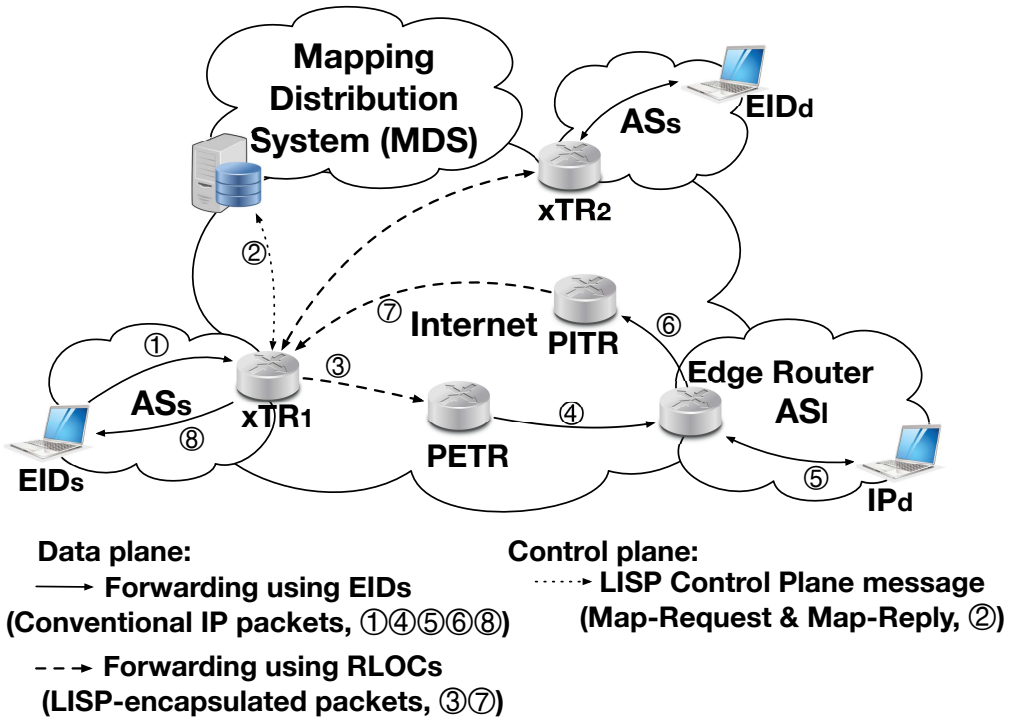

Fig. 2.5 LISP architecture with the packets forwarding sequence

2.2 Interworking With Legency Internet

Sec. 2.1.3 describes how the LISP packets are exchanged between two LISP-sites. This section discusses how the packets are processed when a terminal behind LISP-site communicates with a terminal on the legacy Internet.

Although EIDs are syntactically identical to IPv4 or IPv6 addresses, routes to them are not announced in the global routing system, so an interoperability mechanism is needed for non-LISP-speaking sites to exchange traffic with LISP-speaking sites. Thus, two new network elements are introduced: LISP Proxy Ingress Tunnel Router (PITR) and LISP Proxy Egress Tunnel Router (PETR). The PITR acts as an intermediate LISP ITR for the hosts on legacy Internet. It is in charge of announcing one or more highly aggregated EID-Prefixes on behalf of LISP-sites into the public Internet and receives the traffic from the public Internet to encapsulate them as LISP packets. The PETR acts as an ETR for traffic destined to non-LISP sites. It receives the traffic from the ITRs, decapsulates the LISP packets into conventional IP packets and sends them to the legacy Internet.

Fig. 2.5 shows an example for the packets exchanging between LISP-site and non LISP-site (i.e., legacy Internet). The host with EID_s in the AS_s exchanges the packets with the host with the IP_d in the AS_l. With the same procedure introduced in Sec. 2.1.3 labeled as *"1"* and

"2" in Fig. 2.4, the original IP packets are sent from host to xTR_1, and the later one sends the Map-Request to MDS to query the mapping information. Since the destination is in a non-LISP-Site, the MR directly returns back the Map-Reply without mapping information (i.e., Negative Map-Reply) to xTR_1, the packet can either be sent non-encapsulated or the xTR_1 encapsulates it using the statically configured RLOC of the PETR as the outer destination address and forwards the LISP packets there. When PETR receives the LISP-encapsulated packets, it performs a decapsulation as an ETR and then forwards the inner IP packets to the legacy destination host by using the normal BGP routing, labeled as procedure *"4"* and *"5"* in Fig. 2.5.

When the packets come back, they are forwarded in a traditional way from the host with address IP_d in AS_l to the PITR, denoted as procedure *"5"* and *"6"* in Fig. 2.5. A PITR shares many characteristics with ITRs, as it also queries the mapping information of destination to the MDS and encapsulates the non-LISP Internet traffic into LISP-encapsulated packets and route them to their destination RLOCs (label *"7"*). Target ETR_1 then dencapsulates the packets and send them to the destination host EID_s. The PETR and PITR can be noted together as PxTR if they are located on a same physical router.

2.3 Mapping Cache Update Mechanisms

Given that LISP mapping distribution is based on a pull model, in case that mapping changes occur in ETRs, the only way that makes remote ITRs be aware of such change and update the corresponding mapping information is to request again the mappings to ETRs. Thus, two mechanisms are proposed to notify ITRs that mapping has changed and they need to get the latest mapping: Solicit-Map-Request and Map-Versionning.

2.3.1 Solicit-Map-Request (SMR)

Soliciting a Map-Request is used by ETRs to tell remote ITRs to update the mappings they have cached. When the mappings in the Database of an ETR changes, the ETR sends the Map-Requests with the SMR bit set (called Solicit-Map-Request (SMR) message) for each RLOC in its Cache. A remote ITR that receives the SMR message will take a check in its Cache whether the source RLOC of the SMR message, i.e., the RLOC of ETR with mappings change is in its Cache. If yes, the remote ITR sends a Map-Request to the MR (as shown in the left hand part of Fig. 2.6) or directly to the sender of SMR (as shown in the right hand part of Fig. 2.6). If the RLOC of sender is not in the Cache of ITR any longer, the later drops the SMR and does not send the Map-Request. The later situation occurs when

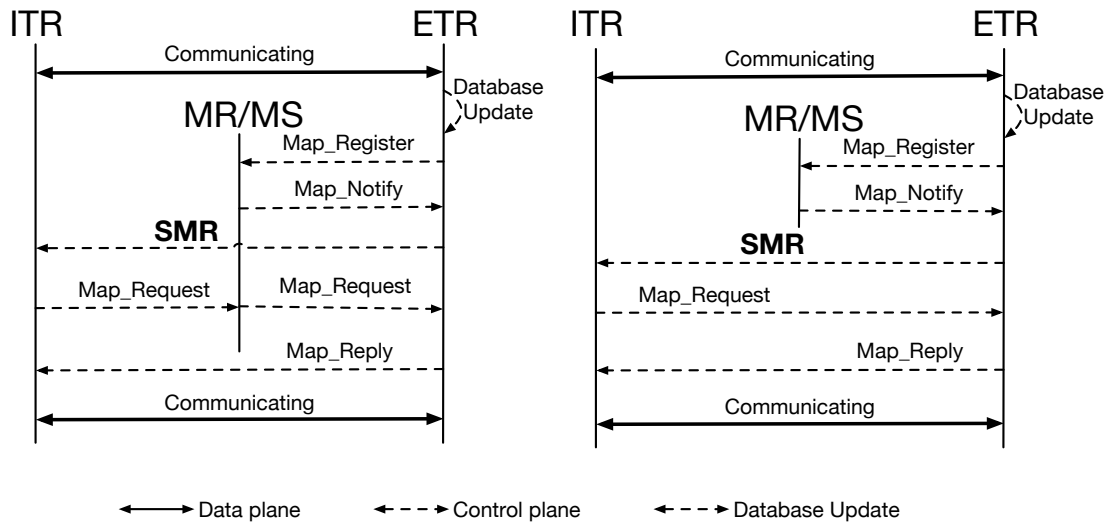

Fig. 2.6 Packet sequence of LISP SMR mechanism

the ETR sends the SMR to all RLOCs stored in its Cache, but the remote ITR actually has not sent the packets to it for a long time. Between sending the SMR message and receiving the new Map-Reply, the ITR continues to use the mapping information previously cached, and that may cause packet loss.

2.3.2 Map-Versioning

Map-Versioning is used to inform through the Data Plane the xTR that the mappings of a remote site changed and update the mapping information with the help of SMR message. Map-Versioning associates a Map-Version number to each LISP mapping information and transports such a version number in the LISP-specific header [68]. When a mapping changes, a new version number, normally incrementally higher than the previous one, is assigned to the updated mapping. When an ITR encapsulates a packet, it selects the version number assigned to the mapping stored in the Database as *source RLOC version number*, and selects the version number assigned to the mapping contained in the Cache as *destination RLOC version number*. When an ETR receives such packets and dencapsulates them, it makes a comparison between the received version number and the local one. If the source map version is higher than the one that ETR stores in its Cache, i.e., the mappings of remote ITR changed, the ETR sends a Map-Request to the MR or directly to the remote ITR to retrieve the latest mapping information (the left figure of Fig. 2.7). If the destination map version is lower than the one contained in the Database of ETR, i.e., the local mappings

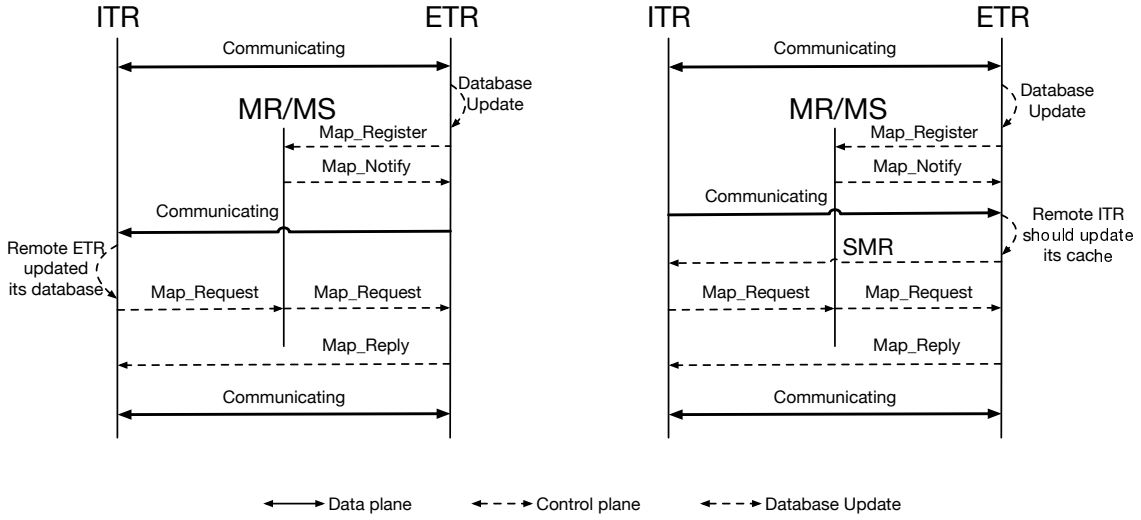

Fig. 2.7 Packet sequence of LISP Map-Versioning mechanism

changed but the remote site does not know, then the ETR sends a SMR to the remote ITR to let it update the cached mapping (the right figure of Fig. 2.7).

2.4 LISP Mobility

Nowadays, the explosion of mobile devices that need the massive mobile Internet traffic requires the evolution of mobile network architecture. Keeping the communication during the roaming without any interruption becomes a key point. Thanks to LISP separating the identifier of a host and the locator of the attachment point, the connection between two hosts can be kept alive while roaming. Because the EID, which is used as the inner address, is permanent, whereas only RLOC (used as the outer address) is changed.

By using LISP mobility, it is possible 1) to allow TCP connections to stay alive while roaming; 2) to provide the shortest bidirectional data paths between a Mobile Node (MN) and Correspondent Node (CN); 3) to allow MN to communicate with another MN when both are roaming; 4) not require fine-grained routes in the core network, nor the home-agent, foreign agent or other data plane network elements to support mobility; 5) there is no triangle routing of data packets as is found in Mobile IP [98]; 6) and there is no IPv6 new extension headers to avoid triangle routing [72]. LISP Mobile Node (LISP-MN) takes advantage of the LISP infrastructure so to overcome the limits imposed by Mobile IP. The implementation of LISP-MN is achieved by OOR (Open Overlay Router) [22], which will be introduced in Sec. 3.3.1.

Table 2.1 Catalogs of mobility

Name of catalogs	MN	Site
LISP-MN in LISP-Site	LISP	LISP
LISP-MN in non-LISP-Site	LISP	non-LISP
MN in LISP-Site	non-LISP	LISP
normal mobility	non-LISP	non-LISP

LISP can be implemented on both border routers and terminals. According to the adoption of LISP and host/router supports LISP, mobility can be divided into 4 categories: LISP-MN in LISP-Site, LISP-MN in non-LISP-Site, MN in LISP-Site, and MN in non-LISP-Site (shown in Tab. 2.1). However, the case "MN in non-LISP-Site" is the current conventional mobility, which has no relationship with LISP. There are some proposals to solve the seamless mobility issues for the traditional handover, such as Mobile IP (MIP) [100], Mobility Support in IPv6 (MIP6) [99] [88], Multipath TCP (MPTCP) [55] and so on. Thus, in the following sections, we will present LISP-MN first and then only describe how the packets exchange when the handover occurs for the first three cases.

2.4.1 LISP Mobile Node

This section introduces a mobile LISP-speaking node, called LISP Mobile Node (LISP-MN). It can reside behind both LISP-Site and non-LISP-Site. It is able to change its attachment point during the communication. The LISP-MN implements a subset of the standard xTR functionality [50]. It can send Map-Request to MR to get the mapping information of the remote host.

When a LISP-MN resides in a LISP-Site, it is assigned an EID taken from the site's EID-prefix as its RLOC, called *Local RLOC (LRLOC)*. Thus, LISP-MN stores not only its permanent unique EID but also the LRLOC and registers this mapping information to the MS. The conventional IP packets produced by the LISP-MN are encapsulated by itself using the permanent EID as the inner source address and LRLOC as the outer source address. The encapsulated LISP packets are forwarded to xTRs and encapsulated again.

When a LISP-MN resides in a non-LISP-Site, it is assigned an IP address taken from the site's prefix as its *LRLOC*. LISP-MN also need to store both its permanent unique EID and the LRLOC and register this mapping information to the MS. The conventional IP packets produced by the LISP-MN are encapsulated by itself using the permanent EID as the inner source address and LRLOC as the outer source address. The encapsulated LISP packets are forwarded to the border router and natively forwarded to the Internet core.

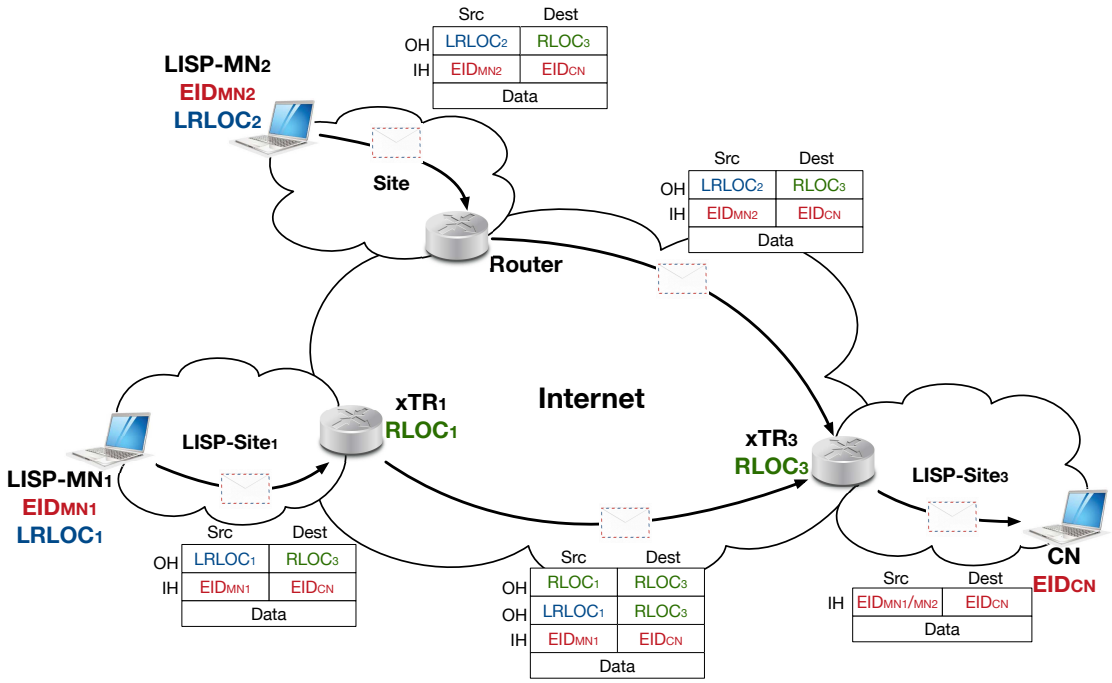

Fig. 2.8 Illustration of LISP-MN in LISP-Site and non-LISP-Site

The packet flow sequence of LISP-MN in LISP-Site and non-LISP-Site are shown in Fig. 2.8. Two LISP-MNs, i.e., LISP-MN$_1$ and LISP-MN$_2$ send the packets to a remote Correspondent Node (CN) . The only difference is that LISP-MN$_1$ resides in a LISP-Site, while LISP-MN$_2$ resides in a non-LISP-Site. The LISP-MN$_1$ produces the traditional IP packets with EID_{MN1} as source address and EID_{CN} as destination address. Then it encapsulates by adding $LRLOC_1$ as outer source address and $RLOC_3$ as outer destination address by itself after querying the mapping information. The LISP packets with one time encapsulation are forwarded to xTR_1. The latter gets the mapping information from MR and encapsulates once again the packets by adding the $RLOC_1$ as the outer source address and $RLOC_3$ as the outer destination address, then sends the double LISP encapsulated packets on core Internet. The xTR_3 needs to dencapsulate the packets twice after receiving and verifying the packets, and finally sends to CN.

The situation for LISP-MN$_2$ is simpler. It also produces the traditional IP packets and encapsulates the packets by itself, then sends to its border router. The *Router* does nothing related to LISP but only forwards the already LISP-encapsulated packets to the Internet core. Thus, the packets are encapsulated only once, whose inner source and destination address is respectively EID_{MN2} and EID_{CN}, outer source and destination address is respectively $LRLOC_2$ and $RLOC_3$.

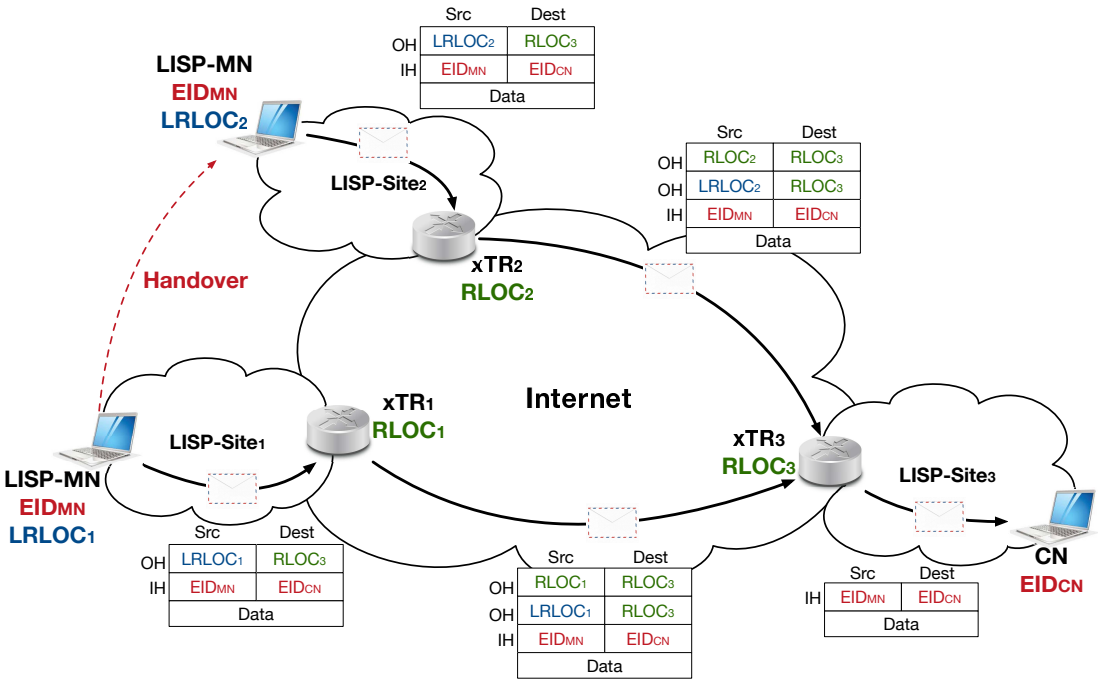

Fig. 2.9 Packet flow of mobility when LISP-MN in LISP-Site

2.4.2 MN mobility in LISP-Site

When the MN is normal host without any LISP function, while the border router supports LISP, i.e., xTR, the packets are exchanged with the remote *CN* as the basic communication between two LISP-Sites introduced in Sec. 2.1.3. The EID of MN is distributed from the EID-prefix of the LISP-Site, if MN moves to a new LISP-Site but keeps the original EID, it can not exchange the packets with its new xTR, since its EID does not belong to the EID-prefix of the new LISP-Site. Thus, in this case, the mobility of MN can only conduct within a subnet instead of roaming through different domains.

2.4.3 LISP-MN mobility in LISP-Site

This section describes how a LISP-MN moves from one LISP-Site to another and the packet flow is shown in Fig. 2.9. At first, packets exchanged between LISP-MN residing in LISP-Site$_1$ and *CN* are double encapsulated LISP packets as mentioned in Sec. 2.4.1. After conducting the handover, LISP-MN still uses its permanent EID_{MN}. However, since it changes the attachment point, xTR_2 distributes a new EID from its EID-prefix to it, and it uses this as its new LRLOC, labeled $LRLOC_2$ in the figure. The LISP-MN should register itself with the new mapping information to the MDS and ask the xTR_3 (xTR of CN) to update

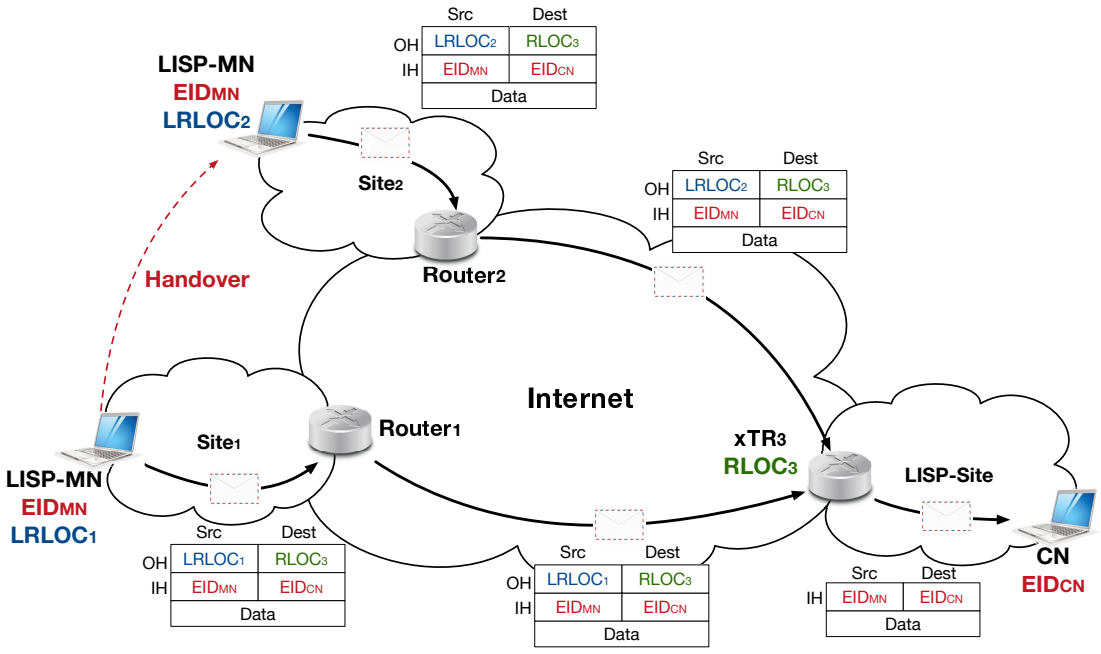

Fig. 2.10 Packet flow of mobility when LISP-MN in non-LISP-Site

its LISP-Cache by *SMR* or *Map-Versioning* (introduced in Sec. 2.3). When receiving the packets from LISP-MN, the xTR_2 conducts the second encapsulation and sends the packets to xTR_3. The benefit of such mobility is that the LISP-MN can roam through the different subnets without interrupting the communication with the remote *CN*, because LISP-MN never changes its EID. However, to conduct double registration may lead to higher delay of handover.

2.4.4 LISP-MN mobility in non-LISP-Site

The Fig. 2.10 illustrates the packet flow of the LISP-MN roaming between two non-LISP-Sites. The $Site_1$ distributes an IP address from its prefix to LISP-MN, and the later uses this as its LRLOC, labeled as $LRLOC_1$ in the figure. The LISP-MN encapsulates the packets by itself and sends to $Router_1$. $Router_1$ just natively forwards the packets to xTR_3. When LISP-MN moves to another site labeled $Site_2$, it receives an IP address from a different prefix and uses it as the new LRLOC. It registers itself to the MDS with the EID_{MN}-to-$LRLOC_2$, and informs the remote xTR_3 (*RLOC* of *CN*) to update the mapping information by sending *SMR* or using *Map-Versioning*. In this scenario, LISP-MN can also permit to roam through the different subnets, but it can also reduce the high latency of double registration during the handover.

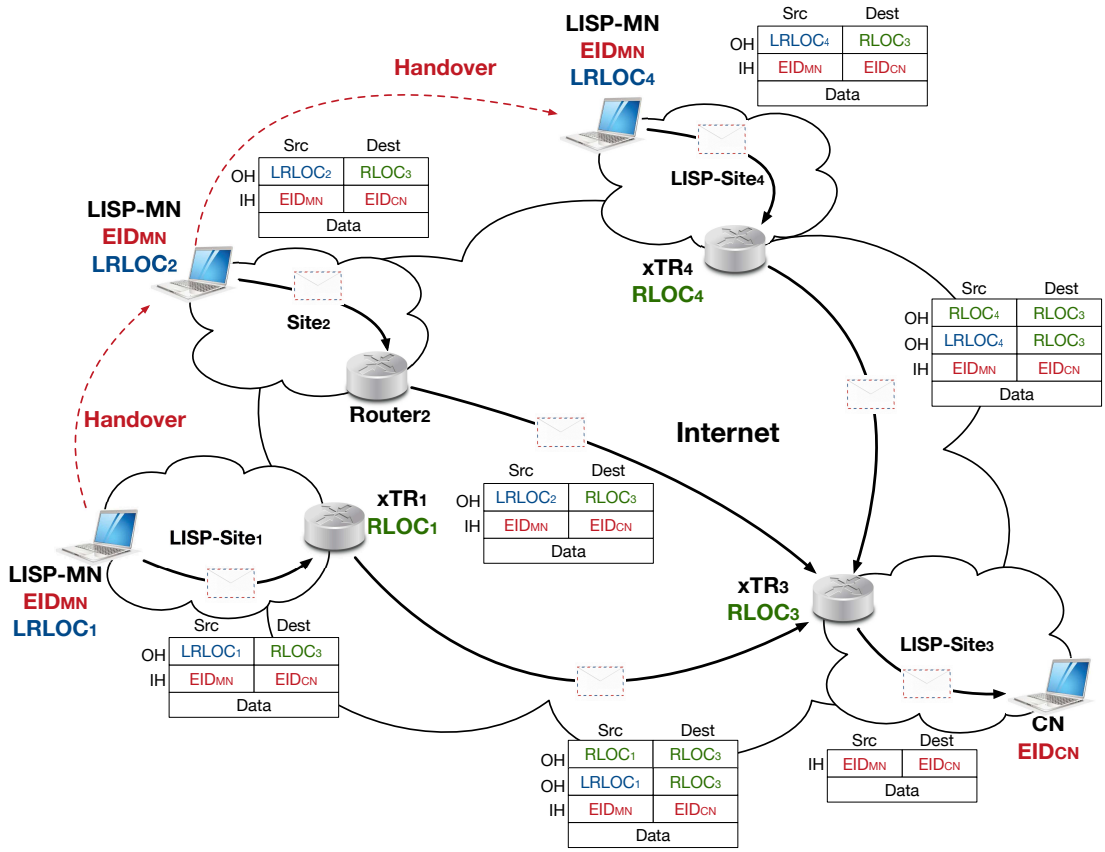

Fig. 2.11 Packet flow of LISP-MN mobility between LISP-Site and non-LISP-Site

2.4.5 LISP-MN mobility between LISP-Site & non-LISP-Site

This section describes how a LISP-MN moves from one LISP-Site (from $LISP\text{-}Site_1$) to non-LISP-Site (to $Site_2$) and vice versa (to $LISP\text{-}Site_4$). Its mechanism combines Sec. 2.4.3 and Sec. 2.4.4, and the packet flow is shown in Fig. 2.11. At first, when the LISP-MN resides in $LISP\text{-}Site_1$, it uses the allocated EID as its $LRLOC$ and exchanges the packets with CN via xTR_1 by double encapsulation. When it moves to the $Site_2$, it receives the assigned IP address from $Router_2$ as its new $LRLOC$. LISP-MN registers its new mapping information $< EID_{MN}, LRLOC_2 >$ to the mapping system and informs the xTR_3 (xTR of CN) to update the mapping in its cache by *SMR* or *Map-Versioning*. The $Router_2$ just natively forwards the packets to xTR_3. When the LISP-MN roams behind xTR_4, it uses the allocated EID from xTR_4 as the latest $LRLOC$ and register this mapping information $< EID_{MN}, LRLOC_4 >$ to the mapping system. Also, it tells xTR_3 ($RLOC$ of CN) to update its mapping by using *SMR* or *Map-Versioning*. Once the handover procedure is completed, the packets exchanged

between LISP-MN and *CN* will be double encapsulated: once on the LISP-MN itself and once on the xTR_4.

2.5 LISP alternative use-cases

Even though LISP is proposed to solve the Internet scalability issues, it also provides the advantages to the other uses, such as: traffic engineering, transition between IPv4 and IPv6, device's mobility at client-side, virtual machine's mobility in data center, LISP SDN, and so on [41].

2.5.1 LISP traffic engineering

For every mapping information, an EID-prefix associates to a list of RLOC tuples <RLOC, Priority, Weight>, where each RLOC is annotated with a priority and a weight. In the case of multiple RLOCs existing, the ITR selects the one with the highest priority and sends all the LISP encapsulated packet to this RLOC. If several RLOCs all have the highest priority, the traffic is balanced proportionally according to their weight among such RLOCs so to route to multiple ETRs. Traffic engineering in LISP thus helps the mapping owner decide the primary and backup path for its incoming packets, and also allows it to control the traffic load among its links. For example, the received Map-Reply shows that an EID-prefix is associated to <RLOC_1, 10, 50>, <RLOC_2, 10, 20>, <RLOC_3, 10, 20>, <RLOC_4, 10, 10>, then the RLOC_1 gets 50% of the traffic, RLOC_2 and RLOC_3 are respectively assigned to 20% of the traffic, and the last 10% of the traffic are balanced by RLOC_4.

Traffic engineering in LISP can achieve one step further than nowadays BPG. Since every Map-Request contains the source EID of the packet, for which causes the ITR mapping miss and triggers the Map-Request. It is possible that a mapping owner (ETR) gives the different Map-Replies according to the sender of Map-Requests. Thus, the traffic engineering policy may be different to different ITRs although they query the same ETR for the same EID-prefix. This functionality is not available today with BGP, because a domain cannot control exactly the incoming routes from the other domains if they are not direct neighbor.

2.5.2 LISP transition between IPv4 and IPv6

The LISP encapsulation mechanism is designed to support any combination of address families for locators and identifiers, such as: IPv4, IPv6, MAC address, or some other arbitrary elements. It is then possible to bind IPv6 EIDs with IPv4 RLOCs and vice versa. This allows transporting IPv6 packets over an IPv4 network (or IPv4 packets over an IPv6 network)

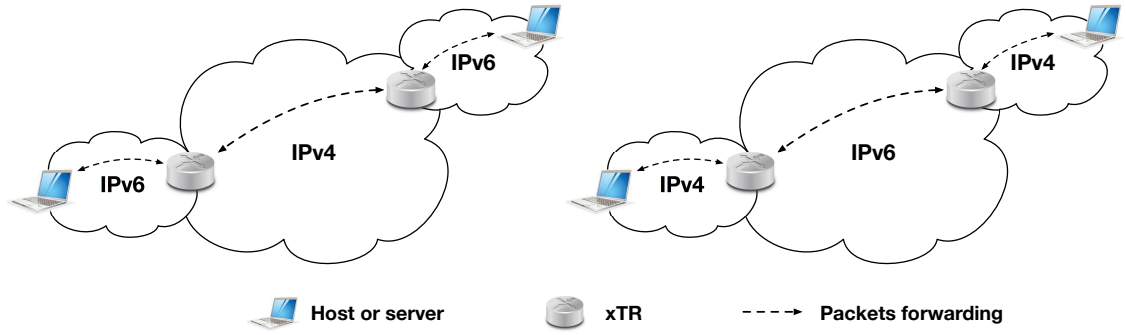

Fig. 2.12 Illustration of LISP transmission between IPv4 and IPv6

by deploying the xTRs at the border of heterogeneous networks, making LISP a valuable mechanism to ease the transition to IPv6.

Examples are as shown in Fig. 2.12, where IPv6 islands are connected via IPv4 infrastructure (on the left) and IPv4 islands are connected through IPv6 infrastructure (on the right). As LISP provides mechanisms for encapsulating IPv6 host packets using IPv4 RLOCs on the xTRs, LISP can then provide IPv6 communication between hosts/servers even though the intermediate network only supports IPv4. Similarly, xTR can encapsulate the IPv4 packets by adding the IPv6 RLOCs to achieve the IPv4 access through the IPv6 infrastructure. In addition, when the LISP infrastructure includes PxTRs, it can provide both IPv4 and IPv6 connectivity to non-LISP IPv4/IPv6 sites (legacy Internet). In summary, it can function in any of four following ways:

- IPv4-to-IPv4 over IPv4
- IPv4-to-IPv4 over IPv6
- IPv6-to-IPv6 over IPv4
- IPv6-to-IPv6 over IPv6

However, LISP does not translate between the IPv4 and IPv6 EIDs.

2.5.3 LISP SDN

As described before, LISP is inherently decoupling Control Plane from Data Plane. LISP moves all control functions onto Mapping System, while keeps data functions at the xTR level.The feature of LISP corresponds to the characteristic of Software Defined Network (SDN), which also decouples Network Control Plane from Data Plane, increases flexibility and development speed of features and functionalists. Thus, this decoupling entitles network operators to build a Software Defined Network (SDN) on top of LISP [106].

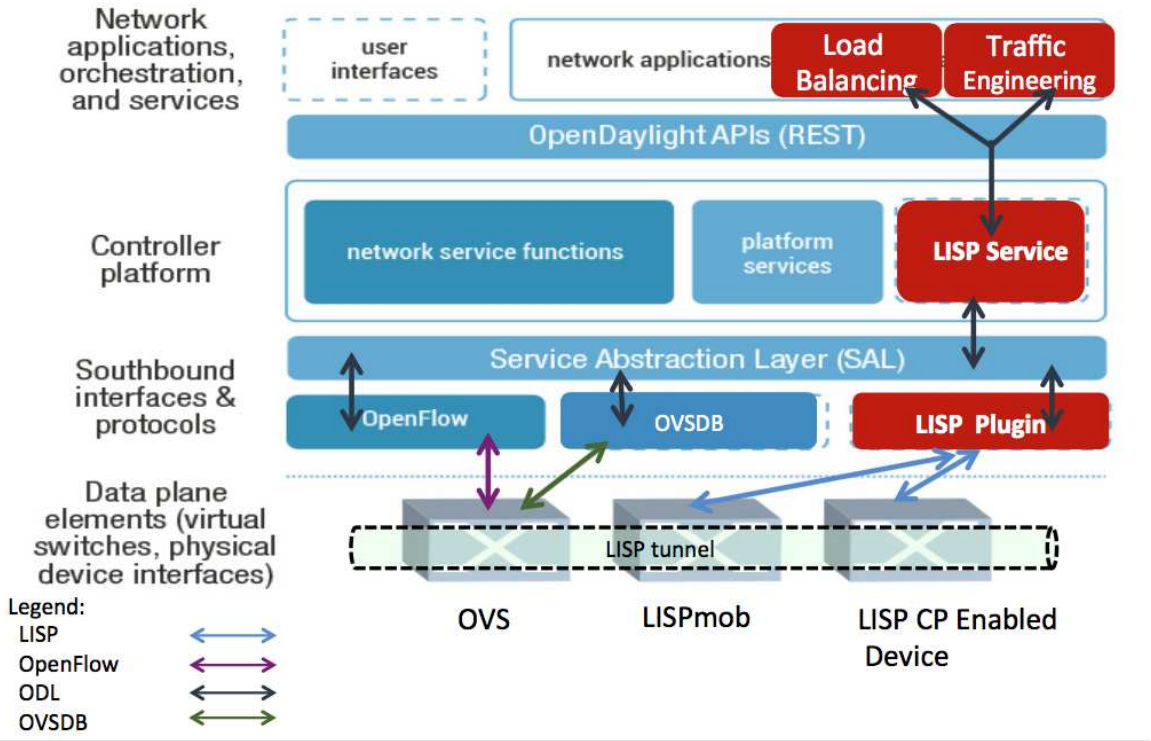

Fig. 2.13 LISP in OpenDaylight (source: [14])

The OpenDaylight Project [23] is a collaborative open source project hosted by The Linux Foundation to promote Software Defined Network (SDN) and Network Functions Virtualization (NFV). It implements the LISP Control Plane functions as shown in Fig 2.13. The data plane elements consist of virtual switches and physical device interfaces, at the aim of constructing the LISP tunnel to transfer the packets. The LISP Plugin is the Southbound interface, which is used to look up LISP policies from LISP Service by sending Map-Request and receiving Map-Reply, and create the LISP tunnel over LISP overlay across the different kinds of data plane devices. The LISP Service is actually implemented as LISP Mapping System to store the various user-defined policies. The only difference is that in OpenDaylight, it is the MS itself that replies to the Map-Requests, instead of forwarding them to any xTR. It is like every mapping has the "proxy Map-Reply" flag (P-bit) set to response on behalf of xTRs. The OpenDaylight APIs (REST) are used to set the policies, such as: the policies for Load Balancing and Traffic Engineering, obtained from the north bound into LISP Service. The policies are defined as the network applications or services by the users.

Network functions such as subscriber-management, content-optimization, security and quality of service, are typically delivered using proprietary hardware appliances embedded

into the network [37]. But the next generation network functions are being implemented as pure software instances running on standard servers. They are unbundled virtualized components of capacity and functionality. LISP-SDN based flow-mapping, dynamically assembles these components to whole solutions by steering the proper traffic in the right sequence to the correct virtual function instance.

Chapter 3

Related Work

This chapter first presents the current LISP status, including: two available LISP platforms, the only LISP monitor, two LISP manual management tools, and some private and open-source implementations. It also introduces the latest research achievements, from the LISP network evolution, the LISP mapping system assessment, the evaluation of interworking between LISP-site and legacy Internet, to the analysis of LISP mobility. Besides presenting the existing works, we also find some limitations in each aspect. Thus, we point out the missing works in the LISP monitor and implementations, as well as in the various sides of LISP performance evaluation. We tackled most of these missing parts and present them in this dissertation. The work that we conducted to complete LISP studies are presented in the following chapters, i.e., from Chapter. 4 to Chapter. 7. Further, we list some other LISP challenges and issues at the end of this chapter. Although these fields are not considered in this dissertation, they are also important to help researchers improve LISP technology.

3.1 LISP platforms

This dissertation is focusing on the evaluation of LISP performance in the realistic environment, thus the large scale experiments on existing platforms are of paramount importance. As such, we introduce first two existing LISP platforms: LISP Beta Network and LISP-Lab platform, before discussing the current research results.

3.1.1 LISP Beta Network

The first global LISP testbed called LISP Beta Network [12] [45], has been deployed since 2008. Started as the only LISP testbed, the LISP Beta network had a steady growth and achieved world-wide experimental deployment [108] with the purpose to gain real-life ex-

Fig. 3.1 International LISP Beta Network High Level Topology – October 2017 [13]

perience on LISP. Initiated by Cisco, the members of the LISP Beta Network are not only major companies and operators, but also academics, research laboratories, and startups offering LISP related services. Participants of this network are located in 34 different countries, with the highest concentration in Europe and North America.

The LISP Beta Network is composed of the mapping system, LISP routers (i.e., xTRs), mobile nodes, Re-encapsulating Tunnel Routers (RTR [47]) and Proxy Tunnel Routers (PxTR [77]). On March 14th 2012, the mapping system used by the LISP Beta Network has switched from LISP+ALT [58] to the more flexible LISP-DDT [59]. Changing the mapping system has considerably reduced the configuration and maintenance burden, while a slight performance degradation has been observed [108]. According to the latest architecture (October 2017) [13] shown in Fig. 3.1, LISP Beta Network has 3 DDT Root, 2 DDT TLD, 7 available MR/MSes and 6 PxTRs. More precisely: 2 MR/MSes and 2 PxTRs are in US, 3 MR/MSes and 3 PxTRs are in Europe, 2 MR/MSes and 1 PxTR are in Asia. It mainly uses two EID addressing spaces, which belongs to Cisco: 153.16.0.0/16 for IPv4 and 2610:00D0::/32 for IPv6 [108], but there are also some other EIDs.

3.1.2 LISP-Lab platform

Due to a monolithic control by one single commercial actor, however, LISP Beta network has limitations to explore the innovative services. Thus, some other projects, such as ANR LISP-Lab project [15], focus on developing new enhanced features.

LISP-Lab project aimed at building an open platform, based on the LISP architecture, providing the environment to perform high quality research and support the design, development, and thorough assessment of new services and use-cases. The range of technical tasks planned in the LISP-Lab project, from cloud networking, to access technology, through inter-domain connectivity, traffic engineering, and mapping management, boosting innovation beyond the LISP technology itself. LISP-Lab platform was solely deployed by the open source LISP implementation, using OpenLISP [24], opened to the external users from 2015. It was coordinated by a French consortium: two academic institutions (UPMC, TPT), two Cloud Networking SME (Alphalink, NSS), two network operators (Renater, Orange), two SMEs on Access/Edge Networking (Border 6, Ucopia) and one Internet eXchange Point (Rezopole). Ten international external partners also joined the platform. At the moment of writing, LISP-Lab consists of 3 MRs/MSes and 2 PxTRs located in France, as well as 13 worldwide xTRs, and has already been interconnected to the LISP Beta Network with an Open-LISP DDT root [60].

3.2 LISP monitoring tools

As the researchers need to leverage on LISP tools to conduct the experiments on the aforementioned LISP platforms, this section introduces the currently existing tools to supervise LISP status or troubleshooting it for some specific purposes.

3.2.1 LISPmon

LISPmon is the only monitoring platform that supervises the current public LISP status, i.e., the mappings between the EID-prefixes and the RLOCs. This platform is developed by the Advanced Broadband Communications Center of UPC (Universitat Politècnica de Catalunya). It scans the whole IPv4 addressing space everyday, normally beginning at 7:00 a.m. (UTC), and queries them by sending the Map-Request to one operator preferred MR of the LISP Beta network (i.e., 7 MRs are candidates to be selected). If this MR does not respond, the LISPmon operator will choose another one as a replacement, and start the queries again, to guarantee that the reception of LISP Map-Replies changes smoothly

between two consecutive days. The available EID-RLOC mapping information is published once per day, ever since the beginning of 2010.

3.2.2 LIG

LISP Internet Groper (LIG) [53] is a LISP-specific manual management tool, used to obtain the mapping information for a certain EID by querying the LISP mapping system. It can be run by all devices that implement LISP, including xTR, PxTR, MR, MS, as well as by a host system at either a LISP-capable or non-LISP-capable site. A possible syntax for a LIG command is:

$$lig < destination > [source < source >][to < map-resolver >]$$

Where <destination>: is either a Fully Qualified Domain Name (FQDN) or a destination EID for a remote LISP site; source <source>: is an optional source EID to be inserted in the 'Source EID' field of the Map-Request; to <map-resolver>: is an optional FQDN or RLOC address for a MR. When LIG is launched, a Map-Request is sent for a destination IP address, and the returned Map-Reply is displayed as follows:

- The EID-prefix for the site that the queried destination EID matches.
- The locator address of the responder.
- The RLOC tuples <RLOC, Priority, Weight>.
- A round-trip-time (RTT) estimate for the Map-Request/Map-Reply exchange.

The remaining work for LIG is about processing of Negative Map-Replies: LIG should be able to clearly indicate how and why a Negative Map-Reply is received. Negative Map-Replies could be sent in the following cases: the LIG request was initiated for a non-EID address or there was rate-limiting on the responder.

3.2.3 RIG

LISP-DDT Referral Internet Groper (RIG) [49] is also a LISP-specific manual management tool, but it is not like LIG, which can be used on any MDS to retrieve RLOCs used for packet encapsulation. RIG is used to recursively find all the MSes serving an EID-prefix, specifically within a LISP-DDT mapping database framework. It performs the same operation as that of a MR. More specifically, when RIG command is run, a Map-Request is sent into the DDT hierarchy for a destination EID, and is forward based on the EID-prefix and delegation referrals contained in the Map-Referral messages until the DDT MS containing the mapping information is reached. It can be run on the same devices as LIG. A simple

syntax for a RIG command is:

$$rig < eid > to < ddt-node > [follow-all-referrals]$$

Where <eid> is the destination EID being queried; <ddt-node> is the RLOC of any DDT node in the DDT hierarchy; when the keyword [follow-all-referrals] is used, all the referral RLOCs will be queried; if this keyword is not used, one of the referral RLOCs will be selected to descend a branch of the DDT hierarchy. The results of returned Map-Referral messages mainly contain the RLOCs of child nodes storing the destination EID mapping information, as well as RTT indicating the time spends for getting the referral DDT nodes.

3.2.4 Monitoring tools shortcomings

LIG and RIG are the manual management tools, they are used to troubleshoot specific issues rather than monitor the LISP platforms. LISPmon as the first and only LISP monitor website and was optimal at the initial stage, but it faces some limitations as the LISP Beta Network growths. LISPmon just publishes the mapping information as the daily once per day. All the published results are queried to one MR of LISP Beta Network and captured only by one VP. Since LISP Beta Network has multiple MRs over the world and join of LISP-Lab platform later, the current monitoring mechanism of LISPmon shows the incompleteness. To move LISP forward, it is necessary to fully understand the performance of every LISP network entity. Beyond only publishing the mapping information between EID-prefix and its RLOCs, automatically conducting the comprehensive experiments from the various aspects will help the LISP experimenters improve their research. Thus, we filled this gap by proposing a new LISP monitor tool for large scale deployments, which will be described in Chapter. 5.

3.3 LISP implementations

The most widely used LISP implementations are the one on Cisco routers and on AVM FRITZ!Box home routers [5]. However, the source codes of their implementations are not available and thus cannot be extended for LISP research purposes. In this section, we focus on other LISP implementations, which can help the researchers verify their enhancements so to move forward LISP technology. Among these implementations, several are proprietary and some are open-source.

3.3.1 Existing implementations

OpenLISP

OpenLISP [24] [102] is an open source implementation of the LISP protocol. The LISP Data Plane, such as: LISP-Cache and LISP-Database, as well as the encapsulation/decapsulation functions are implemented in the kernel of FreeBSD [8] Operating System. LISP Control Plane is written in C in the user space of FreeBSD and Linux. Everything related to the Control Plane is meant to run in the user space. OpenLISP implements a new type of sockets, called the Mapping Sockets providing an API that can be used by any users space process. OpenLISP was implemented by researchers from Universite catholique de Louvain and T-Labs/TU Berlin. At the moment of writing, Pierre et Marie Curie University maintains the Lip6-lisp [11], which is an extended implementation of OpenLISP and it supports fully featured LISP Control-Plane.

Open Overlay Router

Open Overlay Router (OOR) [22], the successor of LISPmob [42], is an open source LISP and LISP mobile node implementation written in C for Linux, Android (only on rooted devices) and OpenWrt that uses LISP or VXLAN [83]/GRE [64] as an overlay for SDN. By interoperating with LISP Beta Network, LISPmob enables mobile hosts to change its network attachment point without losing connectivity, while maintaining the same IP addresses. OOR is maintained by Barcelona Tech University. It focuses on supporting for network function virtualization (NFV), e.g., the integration with OpenDaylight [23]. The extended feature set makes OOR attractive for use.

PyLisp

PyLisp [27] is a small but functional implementation of LISP in Python. There is a package of PyLisp implementation on Github, called pylisp [26] providing the means to parse and create LISP data and control messages, as well as command line tools to perform actions like asking a map resolver for a mapping. The functions of PyLisp are not complete, so the intention to the later stage is the implementations for map server, map resolver and DDT node.

jLISP

jLISP [112], proposed in 2016, is an open source implementation of LISP in Java and runs in the user space. It is platform-independent and object-oriented. jLISP is compatible with

the LISP RFCs and is modular, which facilitates to explicit configurations for every LISP component, such as: xTRs, RTRs, NTRs, and the mapping-system.

LISP on OMNet++

OMNeT++ [21] is an extensible, modular, C++ simulation library and framework, primarily for building network simulators. A LISP simulator is proposed in [75], which integrates LISP into the INET framework [9] for OMNeT++. This simulator implements basic LISP functions, interworking mechanism and LISP-MN architecture. In addition, their work also support NAT traversal. Since this simulation model is proposed in 2012, the only mapping system supported is LISP-ALT instead of nowadays LISP-DDT. In addition, the simulator OMNeT++ is open source, but this LISP extension is not.

Lispers.net

Lispers.net [17] is a Python implementation aiming to implement the complete feature set of LISP and interoperate with all modern proprietary and open source implementations available today. It can run on Ubuntu, Fedora, CentOS, and Debian Linux distros as well as MacOS, Raspbian, and docker containers. The implementation supports the Data-plane components: ITR, ETR, and RTR, and the Control-plane components: MR, MS, and DDT-Node for 10 different EID address types and 7 different RLOC address types. The first version was released in November 2013. Additional non-LISP-specific functions and behaviors are also added, which make lispers.net a general overlay controller. Although the goal of Lispers.net is the world's richest feature set of LISP, it is not open source.

LISP controller in ONOS

The LISP controller in Open Network Operating System (ONOS), which is an open source SDN OS for service providers [63], is one of South Bound Interfaces (SBI) for SDN controllers. The LISP controller in ONOS is proposed to address the OpenFlow issues, such as: scalability, performance, heterogeneity, inter-operability with legacy networks and protocols. An open source SDN controller development project, called OpenDaylight [23] sub-project, is developing the LISP mapping system in SDN environments. The LISP controller centralizes LISP controls into a single entity. It supports LISP as an SDN SBI to work with other protocols such as OpenFlow, and provides APIs for LISP applications. The LISP controller brings many benefits, such as: simplified management, a global view of LISP networks, short information convergence time, and easy development support of LISP based applications.

3.3.2 The LISP simulation gap

At the moment, LISP has some implementations for the different Operation Systems, including either the proprietary source or the open source. However, researchers prefer to test new features on simulations rather than real testbeds, because the simulation environment simplifies the studied problem and allows researchers to concentrate on the most critical issues [90]. So, simulation has a dominant capability over the other evaluation tools, when the researchers want to quickly test new features on a larger scale for different network scenarios. Among the recent network simulators, such as ns-2 [39], ns-3 [65], OMNET++ and OPNET [43], basic LISP features have already been extended on OMNET++. However, it is not open source, which poses the issues for the researchers to easily and flexibly conduct the simulations. In addition, ns-3 is more popular with the researchers than the others [104] and there is an open source project[1] implementing the fundamental LISP on it in 2016. Whereas this project is based on ns-3.24 (the version at that moment), and it does not support features related to LISP mobility. Thus, we decide to leverage on this project, but first adapt to ns-3.27, and realize the extension of LISP mobility. Such contribution will be introduced in Chapter. 7.

3.4 LISP mapping system evaluation

3.4.1 Current studies

The performance of LISP Mapping System is evaluated in [108] and [45] from different aspects. As the LISP Beta Network changed its mapping system from LISP+ALT to LISP-DDT on March 14th 2012, the authors measure the delay of resolving mapping (the latency between sending the Map-Request and receiving the Map-Reply) during this transition period. They find the delay to get the mapping by using LISP+ALT is smaller and more stable than LISP-DDT. However, the manageability of the latter is better than the former, since by using LISP+ALT, maintaining the ALT causes many cumbersome configuration overhead especially as the network growing. Although using LISP-DDT slightly degrades the resolving performance, the delays remain small and the mapping system is able to cope with a significant growth of the addressing space.

[1] Source code is available at: [32]

3.4.2 The incomplete puzzle of MDS evaluation

The previous measurements of the MDS are mainly depended on the delay, including the different resolving latency that each MR needs to use, and comparing which VPs have longer delay to query the mappings. However, the authors have never taken look at the contents of mapping information, i.e., the type of mappings and locators etc.. In addition, there is no work to compare whether the consecutively received Map-Replies change over time. If yes, how they change? If there are any patterns to follow? There are also no comparisons on whether the Map-Replies sent by the different MRs for the same EIDs are identical. Nor to the Map-Replies received by the various VPs. All these evaluations will be discussed in Chapter. 4.

3.5 LISP network evolution

After deploying LISP in LISP Beta Network from 2008, more and more participants joined to the development of LISP. Thus, Saucez, D. and et al. [108] look at how the LISP Beta network evolves. The authors observe that the number of LISP mapping was as low as 20 in January 2010, but it has consistently grown to reach 80 mappings in 2012. The number of LISP mappings is four times the initial number, which indicates that LISP network has a regular growth. Within two months of the second part of 2011, the number of negative mappings was doubled, i.e., from about 150 to about 300. The authors suppose that is due to a more fragmented EID-space. The paper also observes some malformed mappings, which belongs to the LISP mappings but have an empty RLOC set. However, this kind of mappings is never more than 0.5% of the total number of mappings within one day. Further, the paper presents that from January 2010 to May 2012, the percentage of mappings using two or less RLOCs has increased. Moreover, there exist mappings with four RLOCs in 2012, whereas they were not present in January 2010.

3.5.1 Updating LISP measurements

Previous works respectively evaluate the LISP network evolution in 2010 and 2012. However later, LISP mapping system is changed from LISP+ALT to LISP-DDT in 2012. Besides, as mentioned in Sec. 3.1.2 of Chapter. 3, LISP-Lab platform is open since 2015. It brings new LISP-sites over the world and interconnects with LISP Beta Network. Further, LISP Beta Network itself also updated their xTRs and changed its network architecture in 2016. Thus, the number of LISP mappings, the fragment of EID-space, the distributions of

RLOCs are also changed over time. It is interesting to have a look at them in order to know how the LISP network evolves in recent years. This work will be mentioned in Chapter. 5.

3.6 LISP interworking with legacy Internet

LISP has significant impact on the traffic between LISP-site and non-LISP-site. First, to support the interworking between LISP-site and the legacy Internet, the PxTRs should be set up. The configuration on PxTR does not cause any special technical issue, but using PxTR potentially causes path stretch between source and destination. It makes the packet exchange experience a larger delay and potentially leads to higher packet loss rate. Thus, placing the PxTR near the source/destination of traffic allows the communication between the non-LISP site and the LISP site to have the least path stretch (i.e., the least number of forwarding hops when compared to an optimal path between the sites) [77].

When deploying the PxTRs in the experimental infrastructure, it is better to think about first where to deploy the PxTRs, since the placement in topology has a significant impact on the path stretch. Besides, as the number of PxTRs has a direct impact on the traffic load and the recovery success in case of the failure of a PxTR, we need consider how many PxTRs are needed. Further, we need to think if it is necessary that all the PxTRs advertise for the whole EID space, or it only needs that each of them just announces for the different segmented EID-prefixes.

As more and more deployments of LISP, it becomes necessary that the LISP networks communicate with the legacy Internet. The evaluation of the interworking performance is presented in reference [45]. It chooses 200 VPs on the PlanetLab (a global research network that supports the development of new network services from 2003) [25] located in North America, Asia and Europe by the order of diversity of number of VPs. It also selects 116 EIDs, which have at least 1 corresponding RLOC in the mapping from the daily of LISPmon project [19] published on May 17th 2013. The experiment is conducted from November 4th to 7th 2013. The authors find that the interworking generally has a negative impact compared to without using LISP. The results show that there are 70% of the EID-prefixes have at least 20% of increase in the delay (i.e., Round-Trip Time (RTT)). One PxTR behaves the worst that 95% of the EID-prefixes using it suffer from a delay increase. Based on BGP, the selection of PITRs is driven by the economical relationship between autonomous systems, instead of their geographical positions. As a result, even if the LISP Beta Network has 9 available PxTRs at the moment, only 4 are used in the experiment. Further, the most VPs use PxTRs in Asia although they are located in Europe or America. The most popular PxTR is selected 79% of times.

3.6.1 PxTRs evaluation: an incomplete picture

The only evaluation of PxTR used for the interworking between LISP and legacy Internet was conducted in 2013. At that time, the unique LISP platform was the LISP Beta Network. Thus, the measurement was only based on its PxTRs and the VPs on the research network PlanetLab. After the open of LISP-Lab platform in 2015, there is no evaluation about the PxTR anymore. We do not know the recent performance of PxTR on LISP Beta Network and we have no knowledge about if the PxTR of LISP-Lab functions well. Further, the previous assessment was conducted within an experimental/research environment, there is no experience about the interworking performance in the realistic network, especially for the interoperability with the current popular websites on the world. In the Chapter. 6, we will explore all these issues of LISP interworking performance.

3.7 LISP mobility

To support IP mobility, the conventional LISP should be adopted. The first mobility-compatible LISP extension is LISP-MN [50][94]. LISP-MN is an end-host based IP mobility protocol which requires updating the software of the mobile terminal. In other words, the end-host that uses LISP-MN can be regarded as a small LISP-site.

The main drawbacks of LISP-MN are the double encapsulation that may happen and the need to modify software in end-host. Double encapsulation increases mapping latency, overhead and MTU issues. The modification at end-host hinders the deployment of LISP-MN. To overcome these drawbacks, lots of research efforts have been made by the LISP community.

The performance of LISP-MN is investigated by M. Menth and et al. [86]. The authors illustrate the performance degradation due to double encapsulation and the triangle route problem under some conditions. They also propose mechanisms such as location-aware LISP mobility node, local mapping systems to improve the performance of LISP-MN. A simple model that help analyze the handoff procedure in IP mobility is proposed by [101]. Besides LISP-MN, the proposed model can be also used for the handoff analysis for other IP mobility protocols such as MIPv6.

To avoid the modification to end-host required in LISP-MN, a network-based and LISP-based IP mobility protocol: LISP-ROAM is proposed in [61]. Their solution is that the network assigns the same IP address regardless of their network attachment point under the cooperation between some additional/modified network components (DHCP servers, authentication services, LISP xTR and servers). The authors implement LISP-ROAM on top of LISPmob and give an experimental evaluation of the performance of LISP-ROAM. As

an effort to avoid double encapsulation in LISP-MN, LISP-NEMO is proposed in [118]. The basic idea of LISP-NEMO originates from NEMO [71]. It introduces a new network element called mobile router (MR) that manages and represents the mobile network. To mitigate the problems in LISP-NEMO and LISP-ROAM, LISP-HNM is proposed in [113]. The authors introduce an access register protocol and an active notification scheme to conventional LISP so that the latter can support both end-host and network mobility.

The packet loss due to IP mobility is also a concern. To reduce the packet loss, Isah et al. [69] propose an improvement to LISP-MN. The basic idea is that those packets generated during handover procedure are buffered to severs close to the MN's new location and forwarding them to the MN on handover completion.

3.7.1 Open issue in LISP mobility

To achieve the seamless handover, LISP can be implemented on the border router or directly on the host to support the mobility. However, these proposals are only limited at the theoretical level. There are no experimental results showing which solution is better based on some metrics, such as: the smaller handover delay or the less packet loss. No evaluations respectively compare the advantages and the disadvantages of each solution. What's more, no matter which solution is adopted to conduct the mobility, SMR and Map-Versioning are the candidates to say to the remote ITR to update its mapping cache. Thus, it is interesting to make a comparison between these two mechanisms, for example modeling the overhead traffic according to the number of MNs and CNs of each mechanism. The Chapter. 7 will analyze those scenarios.

3.8 Troubleshooting problem

A big issue in the LISP experimentation is the difficulty of troubleshooting. The most frequent used troubleshooting tool is *traceroute*, but it can not provide the exact path that the packets pass between the xTRs or PxTRs. The experimenters only know how many hops between them. When the host traceroutes a destination, it uses each EID as the source/destination IP address. However, when the packets pass to the ITR, it encapsulates the packets, so the following routers think that the ITR is the source of the packet and will give back the ICMP message to the ITR. In this way, the host, which is the real source receives no response. Until the packets reach to ETR or PETR, they decapsulate the packets to know the real source from the inner header and answer to the EID of host.

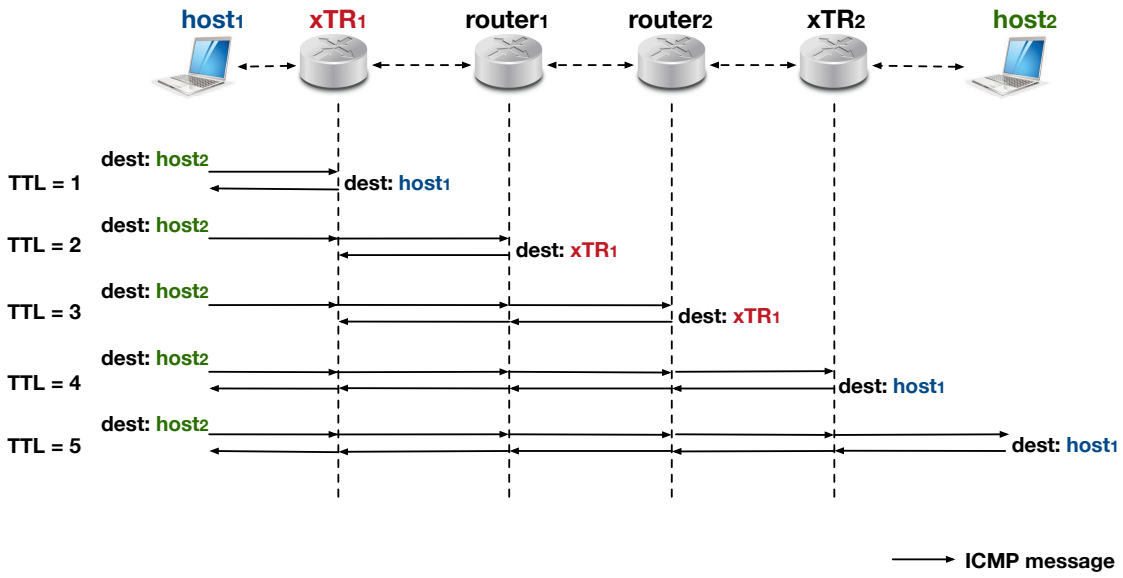

Fig. 3.2 Illustration of LISP troubleshooting

An example of LISP troubleshooting is shown in Fig. 3.2. When $host_1$ traceroutes $host_2$, $host_1$ puts its own EID and the one of $host_2$ respectively as source and destination address. The first UDP message with TTL (time-to-live)=1 expires at xTR_1, then it sends back the ICMP Time Exceeded message to $host_1$. Thus, the $host_1$ knows that its next hop is xTR_1. However, when xTR_1 receives the UDP message with TTL=2, it decreases the TTL value, and encapsulates the UDP message with its own RLOC as source address and forwards to the next hop $router_1$. Since the UDP message that the $router_1$ receives showing xTR_1 is the source instead of $host_1$, the $router_1$ drops the packets and replies back to xTR_1. In this way, the $host_1$ cannot receive any ICMP Time Exceeded messages until the xTR_2 is tracerouted. When the xTR_2 receives the UDP message, it decapsulates the packets and finds that the real traceroute sender is $host_1$. Thus the xTR_2 replies to $host_1$. Same to $host_2$, the received UDP message shows that $host_1$ is the traceroute origin and itself is the destination, so it returns an ICMP Port Unreachable message $host_1$.

To the conventional Internet topology, *traceroute* uses the returned ICMP Time Exceeded messages to build a list of routers that packets traverse until the destination is reached. However, the list of routers between xTRs is hidden to the traceroute sender. It can only know how many hops it needs to arrive the ETR. Thus, when there is a problem in the LISP network, it is hard for the operators to pinpoint the reason as they have only a partial view of the network. The experimenters can leverage on the LISP Cache/Database or querying the MRs to observe the potential problem in the network, but the knowledge remains limited.

This issue brings inconveniences to our experiment, which will be discussed in Chapter. 6. In that experiment, we can only find how many hops between our VPs and the PxTRs, but are not able to specify the exact routing paths.

3.9 Other LISP Related studies

LISP is relatively young and there are many other aspects needed to be taken into consideration, such as: the caching, the LISP security, the MTU issue, and so on. As they are important and necessary to LISP, we list them here. However, since they are not related to this dissertation, they will not be discussed in the following chapters.

3.9.1 LISP Cache performance

A thorough analysis of LISP Cache is presented in [67], [73] and [74]. Their works complement and validate each other. The papers show that although the cache time largely reduces from initial 24-h to as small as 60s, the LISP cache maintains a very high efficiency, with more than 99% hit-ratio. As a result, the cache highly reduces the size, and the traffic volume overhead introduced by both the traffic encapsulation and the control traffic to request missing mappings is limited to 6%. Around 70% of cache miss is generated by UDP protocol regardless the port number. The authors argue that is mainly caused by P2P applications or any other distributed applications generating the same spread traffic pattern. They also compare the original LISP specification, called vanilla LISP with the symmetric LISP model, proposed by Saucez et al., and show that the later model provides higher security. Besides, they evaluate the relationship between the size of cache and the number of users. It shows that when the number of users is increased to three times, the size of cache only doubles. It means that the average size of the LISP Cache is not linear relationship with the number of hosts.

As querying the mapping information to the MDS largely increases packet transmission delay, to store the necessary mappings in the LISP Cache can significantly alleviate it. However, as limited capacity of LISP Cache and dynamic changes of mapping information, it still requires smart algorithms to determine which mappings should be reserved or removed for better performance, especially when there are many expired mappings caused by mobility [54].

3.9.2 LISP security

As LISP is born to address the scalability issues of Internet, it is not designed with security in mind. The possible security threats that LISP faces are given in [109]. Although after the authentication, the MSes only accept the registrations of their compromised ETRs, the mapping system can still be attacked. For example, the compromised ETRs may de-aggregate their EID-prefixes in order to register more EID-prefixes than necessary to their MSes, and this causes the ITRs to send overwhelmed Map-Requests to the mapping system. The compromised ETRs may also overclaim the prefix they own in order to influence the route followed by Map-Requests for EIDs outside the scope of their legitimate EID-prefix. Besides, the Map-Register stage is vulnerable to masquerading and content poisoning attacks, for example: the ETRs register the fake mapping information to the MSes [34] [91] [33]. By sending the Map-Requests at a high rate can overload the nodes in the MDS and cause the flood of Map-Replies as well. Some bits in the LISP control plane messages also have some security issues, for example: the Locator-Status-Bits can cause a DoS (Denial-of-Service), a replay, a packet manipulation, or a packet interception and suppression attack; the Map-Version number can mount a DoS, an amplification, or a spoofing attack; the routing locator reachability is able to mount a packet manipulation and an amplification attack.

A set of security mechanisms (LISP-SEC) that provides authentication, integrity and anti-replay protection for LISP are introduced by Maino, F and et al. [84]. A number of vulnerabilities should be considered before deploying LISP in large scale [103].

The reference [33] analyses the security and functionality of the LISP mapping procedure using a formal methods approach based on Casper/FDR tool. The authors point out several security issues in the protocol such as the lack of data confidentiality and mutual authentication. The reference [34] figures out that the Map-Registration stage is vulnerable to masquerading and content poisoning attacks. Therefore, it presents a new security method for protecting the LISP Registration stage. The proposed method uses the ID-Based Cryptography (IBC), which allows the MDS to authenticate the source of the data.

3.9.3 The Maximum Transmission Unit (MTU) issue

LISP encapsulation adds 36 bytes for IPv4 (i.e., 20 bytes of outer header + 8 bytes of UDP header + 8 bytes of LISP header) and 56 bytes for IPv6 (i.e., 40 bytes of outer header + 8 bytes of UDP header + 8 bytes of LISP header) to the original packet size. As with other protocols that encapsulate or tunnel traffic (e.g., IP Security (IPsec) [114] and generic routing encapsulation (GRE) [52]), it is possible that the resultant encapsulated packet would exceed the MTU somewhere along the transit path. LISP provides both stateful and stateless

mechanisms for handling potential MTU issues [7], with the main goals being (i) preventing packets from being dropped, and (ii) preventing the need for ETRs to perform packet reassembly prior to LISP de-encapsulation.

Based on real Internet experiments, [67] and [74] show that although the encapsulation overhead is not negligible, it is very limited (in the order of few percentage points of the total traffic volume). In addition, no major issue due to the MTU is observed on the LISP Beta Network.

Chapter 4

Evaluation of LISP mapping system

4.1 Introduction

As presented in Chapter 2, to retrieve the routing information (RLOCs) for EIDs, LISP relies on a pull model by explicitly querying the Mapping Distribution System (MDS). The performance of the mapping system is of paramount importance for LISP. Thus, we leverage on *LISP Beta Network* [12], to conduct measurement campaigns about the mapping system. Such a worldwide real playground has been used to assess the state of LISP deployment and evolution over the years [108]. Yet, to our best knowledge, nobody at the moment has evaluated the mapping system from the following two standpoints: Stability and Consistency. We thus propose a much deeper study and a more thorough evaluation, focusing on the exploration of all the mapping systems on LISP Beta Network through active LIG (LISP Internet Groper [53]) measurement. The temporal comparison shows that the mapping system is quite stable over time and the crosswise comparison indicates that the mapping system is consistent between network entities. However, some instability and inconsistency cases are also identified through evaluation, which indicates that the mapping system can be improved. Hence, besides proposing a brand new taxonomy in order to classify them, we carry out an in-depth investigation of such cases to provide hints on how to improve the performance of LISP.

The remainder of this chapter is organized as follows. Sec. 4.2 presents an overview of the experiment campaigns and introduces the metrics that we use to evaluate the performance of mapping system. Sec. 4.3 and Sec. 4.4 respectively describe the mapping system stability and consistency results in details, as well as the instability and inconsistency cases by using the proposed taxonomy, and investigate in depth the causes from different viewpoints. Finally, Sec. 4.5 provides some findings in other scopes and concludes this chapter.

4.2 Methodology

In order to collect data to carry out our analysis, several measurements campaigns have been performed. Hereafter we provide an overview on the measurements and the metrics used in our evaluation.

4.2.1 Measurements Overview

To obtain a comprehensive and in-depth evaluation of LISP Beta Network Performance, we conducted experiments using LIG (LISP Internet Groper [53], see Sec. 3.2.2 for the details). When LIG is launched, a *Map-Request* is sent for a destination IP address, and the returned *Map-Reply* is then stored for further analysis. It exists three possible types of outcome: (i) *LISP Map-Reply*, where the reply contains mapping information to reach the LISP-enabled sites; (ii) *Negative Map-Reply*, where the reply states that the requested IP address belongs to a prefix of a non-LISP site; (iii) *No Map-Reply*, where no answer is received within some amount of time. A query is considered to be successful if either a LISP Map-Reply or a Negative Map-Reply is received. While a query is considered to be failed if no reply is received within 5 seconds.

The experiments are conducted from July 2nd to 18th 2013. From a set of different vantage points (VPs), sending *Map-Request* messages to all the Map-Resolvers of the LISP Beta Network, for all selected destination IP addresses every 30 minutes. The set of VPs was composed by part of academic networks, commercial Internet, PlanetLab and spread across Europe and USA. The destination IP addresses consisted of the first address within each prefix derived from the LISPmon project [19] regardless of the type (i.e., the EID-prefixes or regular prefixes). During this measurement campaign, 5 VPs and 13 MRs/MSes were available and 613 prefixes were queried at each round.

We analyzed the Map-Replies associated with EIDs instead of all regular IP addresses. In the remainder, we use the term *experiment round* to identify a complete *Map-Request*/*Map-Reply* exchange by LIG for all the selected destinations to all the available MRs over all existing VPs. Although No Map-Reply occasionally appears in some experiment rounds, we just focus on LISP Map-Reply and Negative Map-Reply to conduct the analysis. [1]

[1] We use *"simultaneously"* or *"at the same time"* as synonym of *"at a same experiment round"*.

Table 4.1 Map-Reply Differences

Category	Mismatch
Map-Reply Type	Negative Map-Reply and LISP Map-Reply
RLOC set	RLOC number
	RLOC address
	RLOC priority
	RLOC weight
	RLOC state

4.2.2 Metrics

As indicated in Sec. 2.1.2, the LISP Control plane is in charge of offering the mapping information that consists of an EID-prefix associated with a list of $<RLOC, Priority, Weight>$ tuples. Theoretically, unless configuration has changed, if the same EID-prefix is queried, Map-Replies should always contain the same mapping information, no matter from which VP the query is sent, and no matter which MR is queried. During our measurements, however, we found that the Map-Replies are not always identical as they are supposed to be. The reasons causing the Map-Reply changes are summarized in Table 4.1.

The Map-Reply differences can be classified into two broad categories: the first is a difference in terms of *Map-Reply Type*, which means that a mix of Negative Map-Reply and LISP Map-Reply occurs; the other type of difference is in the *RLOC set*, which means that at least one of the attributes associated with RLOC set is different. More specifically, *RLOC number* indicates a difference in the number of RLOC tuples received in the Map-Reply. The others are differences in a RLOC tuple, which consists of RLOC address, RLOC priority, and RLOC weight (presented in Sec. 2.1.2).[2] The RLOC state is derived from the *R-bit* in Map-Reply message, which is set when the sender of a Map-Reply has a route to the RLOC, meaning that from its viewpoint the RLOC is up and running. If at least one of the six parameters shown in Table 4.1 conveyed in a Map-Reply is different from the previous one (over time) or the others (between different VPs or MRs) at the same experiment round, the Map-Reply is marked as different. Based on the criteria of the Map-Reply difference, we propose two metrics to evaluate in more details the performance of the LISP mapping system: (i) *Stability* and (ii) *Consistency*.

A mapping is considered to exhibit *stability* if the Map-Replies for one specific EID obtained from one MR observed at a given VP (briefly denoted as a tuple $<EID, MR, VP>$),

[2]The terms *RLOC address* and *RLOC* are used interchangeably in this chapter.

over time, are identical. A classification taxonomy for unstable cases is proposed later in Sec. 4.3.

A mapping exhibits *consistency* if: *i*) Map-Replies for a specific EID from all MRs observed at a given VP (denoted as a group <$EID, *, VP$>) are identical at the same time (this kind of consistency is referred as *Consistency by MR*); *ii*) Map-Replies for a specific EID-MR pair in all VPs (denoted as a pair <$EID, MR, *$>) are identical at the same time (referred as *Consistency by VP*). A classification taxonomy for non-consistent cases is proposed later in Sec. 4.4.

4.3 Mapping System Stability Evaluation

In this section, we study the stability of the mapping system over time. More precisely, we consider that an EID is *stable* if all mappings retrieved for this EID are identical. If it is not the case, we say there is *instability*.

Overall, when we consider the whole dataset, 91.49% of observations are stable, which indicates that over the full set of observations, changes are infrequent and the MDS is stable. When the stability is considered on a per vantage point basis, we observe, as shown in Fig. 4.1, that the stability is independent of the vantage point since no significant deviation from the overall average can be observed (dashed line in Fig. 4.1). Nevertheless, while these results demonstrate that the mapping system is generally stable, in the following of this section, we investigate the reason why about 8.51% of the observations exhibit instability.

Fig. 4.2 complements Fig. 4.1 by showing, for each tuple <EID, MR, VP>, the frequency $f_{<EID,MR,VP>}$ at which mappings are different, by using the following definition:

$$f_{<EID,MR,VP>} = \frac{\#\text{ Instabilities}}{\#\text{ Experiments}} \times 100\%. \tag{4.1}$$

Thus, if the instability frequency $f_{<EID,MR,VP>}$ is 0%, it means that such a tuple is always stable during the whole experiment, i.e., its Map-Replies are always the same. On the contrary, a tuple <EID, MR, VP> with instability frequency of 100% indicates that its current Map-Reply is never identical to the previous ones, i.e., its Map-Replies change at every experiment round.

Fig. 4.2 shows the CDF of the mapping change frequency (i.e., instability frequency). It shows the probability where the instability frequency equals to zero is 91.49%, which is coherent with the result in Fig. 4.1 (dashed line) and means that the mapping is globally stable. Apart from the stability case, the others ($100.0\% - 91.49\%$) belong to the mapping changes case, i.e., instability. This CDF also shows when the instability frequency equals to

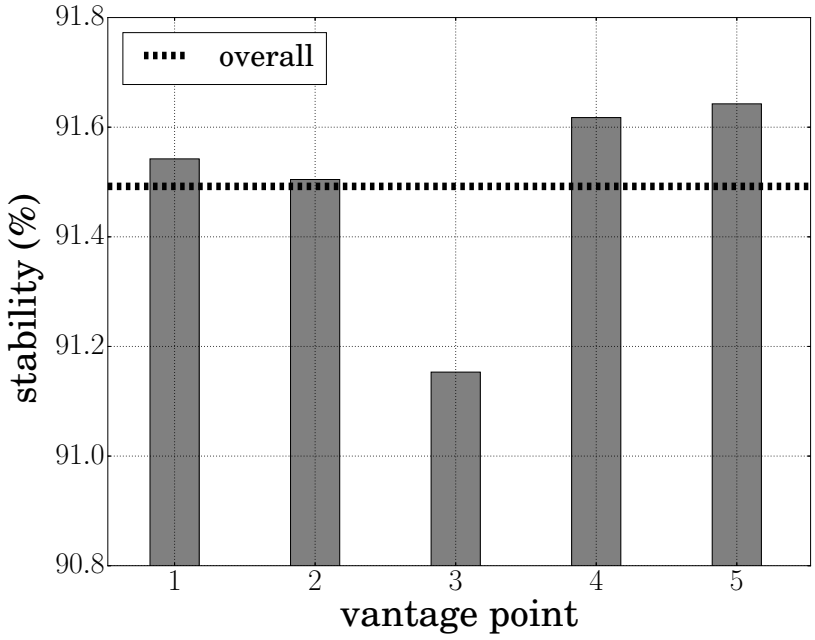

Fig. 4.1 Overall percentage of stability observed at five different VPs.

1%, the CDF reaches to 98.53%, which indicates that among the instability cases, 82.77% ($\frac{98.53-91.49}{100.0-91.49}$%) of all mapping changes events have an instability frequency less than 1%. We can deduce that mapping changes, if they occur, are rare events. The existence of instability frequency around 50% means that, in some cases, even though it is rather rare, the mapping changes occurs almost one time every two experiment rounds.

While Fig. 4.2 indicates that mappings can potentially change frequently, Fig. 4.3 shows the CDF of the stability duration, i.e., for how many experiment rounds a mapping remains stable. We observe that the majority of instabilities has a stability duration of 1 experiment round. This fact shows that the mapping change occurs during a measurement period (i.e., 30 minutes, the interval between two experiment rounds). At the other side of the curve, stability lasts up to 800 experiment rounds are observed, indicating that the mapping changed at the very beginning of or the end of the measurements we performed. In the dataset we identified both cases for such situations. To further understand the instability, we classify it into four categories:

- *New Deployment*: the Map-Replies of a tuple <EID, MR, VP> mix two different types: Negative Map-Reply and LISP Map-Reply. (It is caused more often by turning on/off the xTRs than the real deployment of LISP sites. Since the consequence of Map-Reply changes between Negative Map-Reply and LISP Map-Reply, we make this case named New Deployment.)

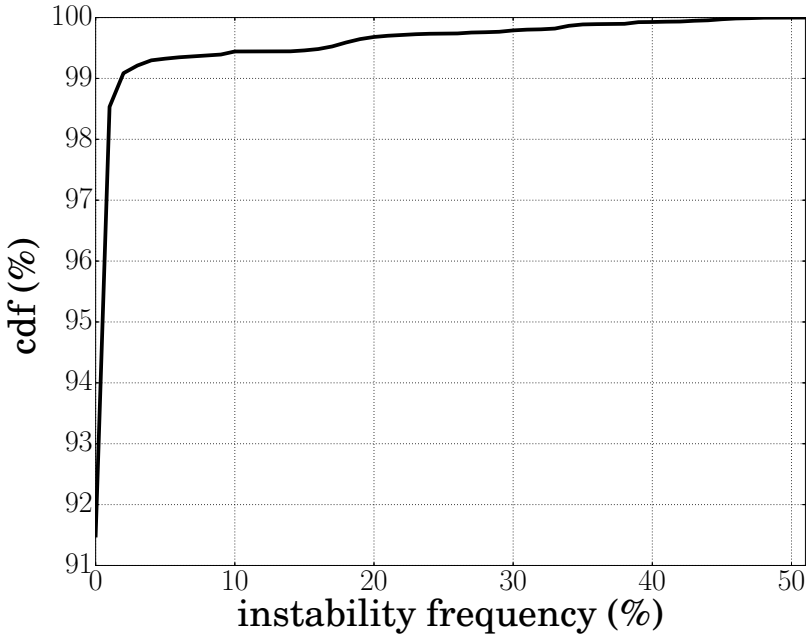

Fig. 4.2 CDF of instability frequency. X-axis is instability frequency, indicating the ratio of instability occurrence number for one <EID, MR, VP> tuple over total experiment number.

Table 4.2 Percentage of Instabilities by Category

New Deployment	Statistical outliers	Mobility	Reconfiguration
72.36%	7.23%	0%	20.41%

- *Statistical outliers*: the Map-Replies of a tuple <EID, MR, VP> change more frequently than usually seen. In our measurements, outliers corresponds to mappings that change at least 3 times during a day.

- *Mobility*: the Map-Replies of a tuple <EID, MR, VP> are explicitly tagged with the mobility bit as specified in RFC6830 [48].

- *Reconfiguration*: all cases that fit neither into the New Deployment case nor the Statistical outliers case.

Tab. 4.2 presents the percentage of observations for the four categories. New Deployments dominate the total unstable cases with a percentage of 72.36%, followed by Reconfiguration with 20.41%, and Statistical outliers with 7.23%. Throughout the dataset, we did not observe any Mobility event, probably due to the fact that the mobility specifications were not yet implemented at the time of measurements. Therefore, we omit the Mobility category in the rest of this chapter.

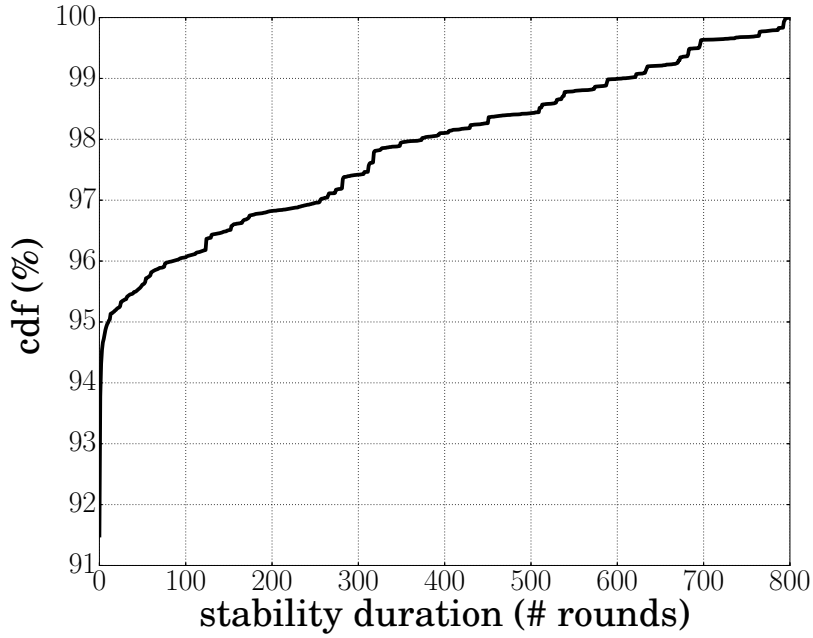

Fig. 4.3 CDF of stability duration (in number of experimental rounds). X-axis is stability duration, indicating during how many experiment rounds that the mapping for one <EID, MR, VP> tuple is stable.

New Deployments are the most common instabilities and we used spectral analysis with Fourier transforms and auto-correlation to understand if such events are correlated between themselves. This analysis shows that New Deployments are equally spread over the whole measurement period, i.e., their occurrence per day is relatively constant. We performed the same study on the Reconfiguration category and Statistical outliers and reached the same conclusion, i.e., the absence of correlations between events.

Since New Deployments are incidental events that can happen at any time, no particular duration is observed for this category of instabilities. Moreover, as New Deployments account for 72.36% of instabilities, it significantly impacts the duration distributions shown in Fig. 4.3, explaining the relatively smooth evolution of the CDF. On the contrary, Statistical outliers are characterized by very frequent changes and they bias the distribution around the lowest duration values, explaining the relatively high number of observations for small duration.

To characterize the mapping changes happening during instabilities, we derived the following *dissimilarity* metric $d(m_i, m_j) = 1 - \frac{|m_i \cap m_j|}{|m_i \cup m_j|}$, based on the Jaccard similarity coefficient, where m_x is the RLOC set of mapping x. A dissimilarity of 1 indicates that no RLOCs are shared between the two compared mappings, while a dissimilarity of 0 indicates that the RLOC sets are the same. Fig. 4.4 shows the CDF of the mappings' dissimilarity for each

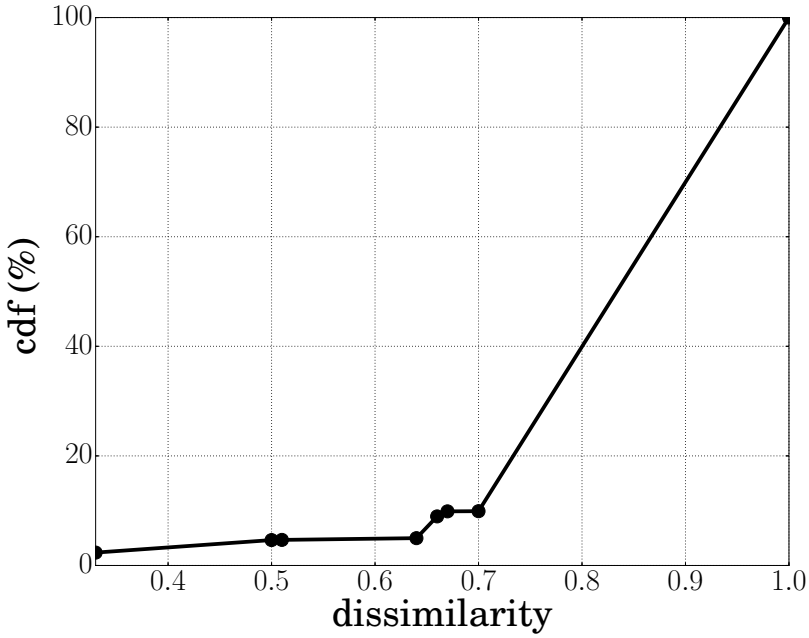

Fig. 4.4 CDF of the dissimilarity among different types of instabilities.

unstable <EID, MR, VP> tuple. In our experiment, the dissimilarity results are discrete (as the discrete points shown in the Fig. 4.4) in a range from 0.33 to 1.0. As one can expect, the most likely dissimilarity is 1 in 90.09% of the total number of instabilities. These observations mostly come from New deployment case where before turning on the xTR, no RLOC is provided with the Negative Map-Reply (RLOC set is empty), hence, switch to a LISP Map-Reply implies a full change in the RLOC set. As 90.09% is larger than the 72.36% of New deployments share, it indicates as well that complete change of the RLOC set have also been observed during Reconfiguration and Statistical outliers. For the rest, no particular trend can be observed showing that changes of RLOC set can actually take any form.

4.4 Mapping System Consistency Evaluation

In this section, we discuss the consistency of the mapping system respectively by MR and by VP. More precisely, we consider that a tuple of <$EID, *, VP$> exhibits consistency, if all the Map-Replies from different MRs, for a certain EID, observed at a given VP are identical at the same time (same experiment round). This kind of consistency is denoted as *consistency by MR*. Similarly, if all the Map-Replies sent by a specific MR, for a certain EID, and observed at different VPs are identical at every experiment round, then we consider that

a tuple of <$EID, MR, *$> is consistent, and denote it as *consistency by VP*. Otherwise, for cases not falling in the above two, we simply say that there exists *inconsistency*.

Overall, throughout the whole dataset, we found that 86.3% of observations of all <$EID, *, VP$> tuples are consistent, which indicates that non-identical Map-Replies from different MRs are uncommon. Similarly, consistency by VP is observed in 90.48% of observations of all <$EID, MR, *$> tuples, showing that the Map-Replies received at different VPs are usually identical. In the following, Sec. 4.4.1 will go deeper in the analysis of consistency by MR, proposing as well several types of inconsistency; then Sec. 4.4.2 will provide thorough analysis of consistency by VP.

4.4.1 Consistency Evaluation by MR

The overall percentage of consistency by MR is 86.3% (see the dashed line on Fig. 4.5), meaning that across the whole experiment traces the inconsistency among the Map-Replies sent by different MRs is not common. Yet, we can also observe from Fig. 4.5 that different VPs show some deviations from the overall consistency by MR, reflecting the fact that while certain VPs receive the same Map-Reply from different MRs, some other VPs receive inconsistent responses. Such deviation is very small (no more than 2.77%) among different vantage points, indicating that it remains a relatively rare event. These results show that the mapping system is consistent among different MRs in general, but the inconsistency exists at the same time with a percentage around 13.7%. In the following of this section, we go deeper this issue, exploring the reasons why the few observations exhibit inconsistency by MR.

We compute the CDF of the inconsistency frequency by MR: $f_{<EID,*,VP>}$, i.e., the probability of inconsistency frequency over all <$EID, *, VP$> tuples, by using the following definition:

$$f_{<EID,*,VP>} = \frac{\text{\# Inconsistencies by MR}}{\text{\# Experiments}} \times 100\%. \tag{4.2}$$

For a <$EID, *, VP$> tuple, an inconsistency frequency of 0% means that all Map-Replies from all of the different MRs for one EID are the same at each experiment round throughout the whole measurement campaign. For a <$EID, *, VP$> tuple, an inconsistency frequency of 100% means that at each experiment round during the whole measurement campaign, there is at least one MR replying a Map-Reply different from the replied by other MRs. The resulting CDF of inconsistency frequency is shown on Fig. 4.6. The value 0% of inconsistency frequency has a probability of 82.38%, indicating that through all the dataset, there are 82.38% of <$EID, *, VP$> tuples that are always consistent by MR at whichever VP. Yet, it is worth noticing that this value is lower than the overall consistency of 86.3% shown in

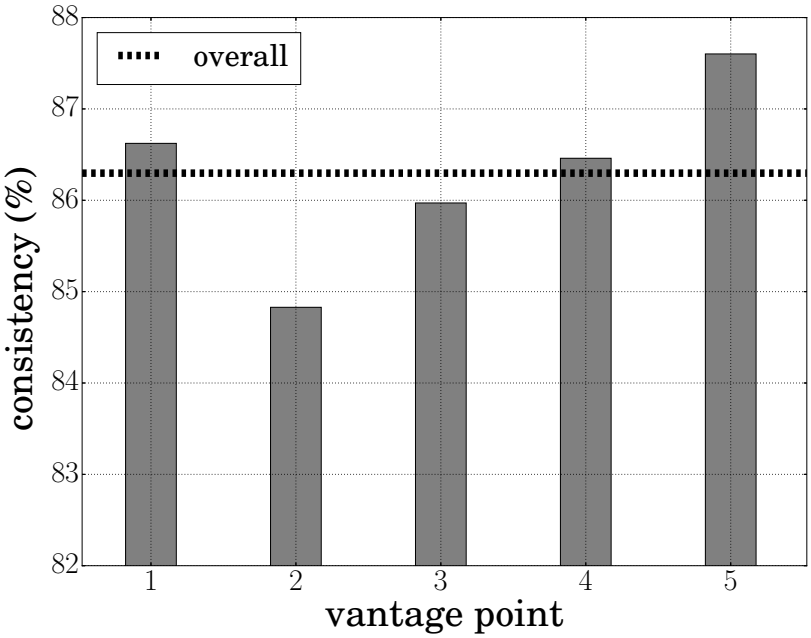

Fig. 4.5 Overall consistency by MR at 5 different Vantage Points.

Fig. 4.5. The reason of such difference lays in the fact that, when we analyze the 0% inconsistency frequency, we just take into account those common EIDs exhibiting a consistency by MR in all VPs. Hence 82.38% represents the subset of the EIDs considered in the former percentage and is the lower bound, guaranteeing that 82.38% of <$EID, *, VP$> tuples always have consistent responses, regardless of the VP. In addition, the inconsistency frequency concentrates at a very low range, since a sharp increase occurs between 0% and 2% with a growth of 16.15%. It indicates that the <$EID, *, VP$> tuples classed as inconsistent do actually receive inconsistent responses rarely, hence, demonstrating that the mapping system is generally consistent. Yet, in very few cases (only 0.5%), the inconsistency between different MRs occurs all the time.

So far, the analysis indicates that the mapping system is consistent by MR in general, but the inconsistency indeed exists. To have a closer look to the inconsistency, we further cluster it into two cases:

- **Map-Reply Type**:, The type of Map-Reply provided by the 13 MRs for a specific <$EID, *, VP$> tuple is different, meaning that some MRs provide LISP Map-Reply while others return a Negative Map-Reply. The difference of received Map-Reply type leads the xTRs to use a different way to send the packets, i.e., xTRs receiving a LISP Map-Reply sends LISP-encapsulated packets by using the obtained RLOCs, while xTRs receiving a Negative Map-Reply will forward the conventional packets (i.e., without LISP-encapsulation).

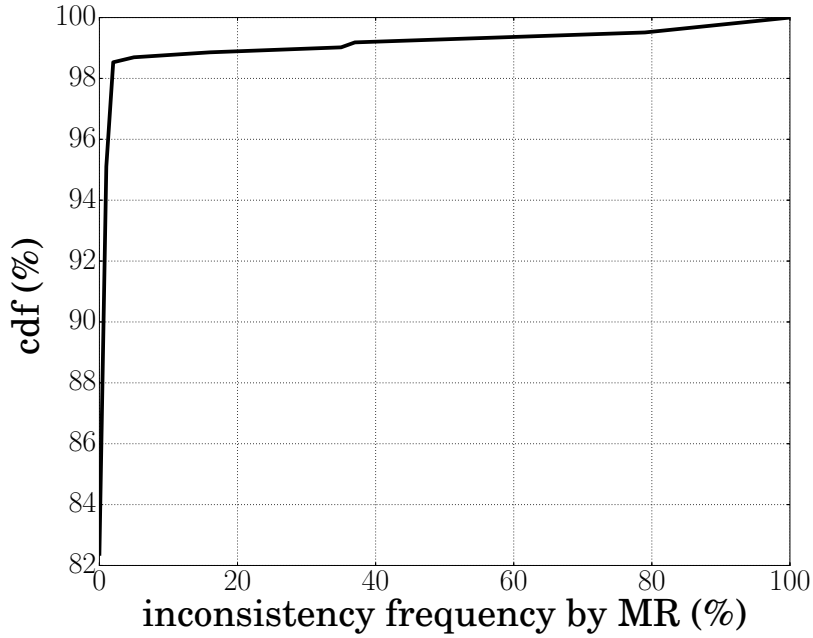

Fig. 4.6 Cumulative Distribution Function of inconsistency frequency by MR. X-axis is inconsistency frequency by VP, indicating the ratio of inconsistency occurrence for each <$EID, *, VP$> tuple over total experiment number.

- **RLOC set**: All the Map-Replies returned by the 13 MRs for a <$EID, *, VP$> tuple belong to LISP Map-Reply type, but the associated attributes (cf. Tab. 4.1 from Sec. 4.2) are different. For instance, some Map-Replies may contain RLOC_1, while some others convey RLOC_2 (i.e., the RLOC address is not identical); or some Map-Replies have only one RLOC, whereas others have multiple RLOCs (i.e., the RLOC number is different), and so on. The difference in such Map-Replies could lead the requesting xTRs to select different destination RLOCs.

The large majority of inconsistencies, around 98.34%, are Map-Reply Type, probably because there is an update of mapping information in the mapping system for the New Deployment case. Not all of the 13 MRs simultaneously update the mapping information, meaning that a convergence time is needed so that the global mapping information maintained in all parts of the mapping system is synchronized. If VPs query the mapping system during this convergence period, some MRs will still provide Negative Map-Replies, while other (already updated) MRs will forward the Map-Requests towards the corresponding ETRs, which in turn will send the updated LISP Map-Replies, and vice versa. This situation contributes to the inconsistency frequency around 1%, which means that there is only once or twice inconsistency between Map-Replies from different MRs for one specific EID, since changing the status of xTR is not frequent.

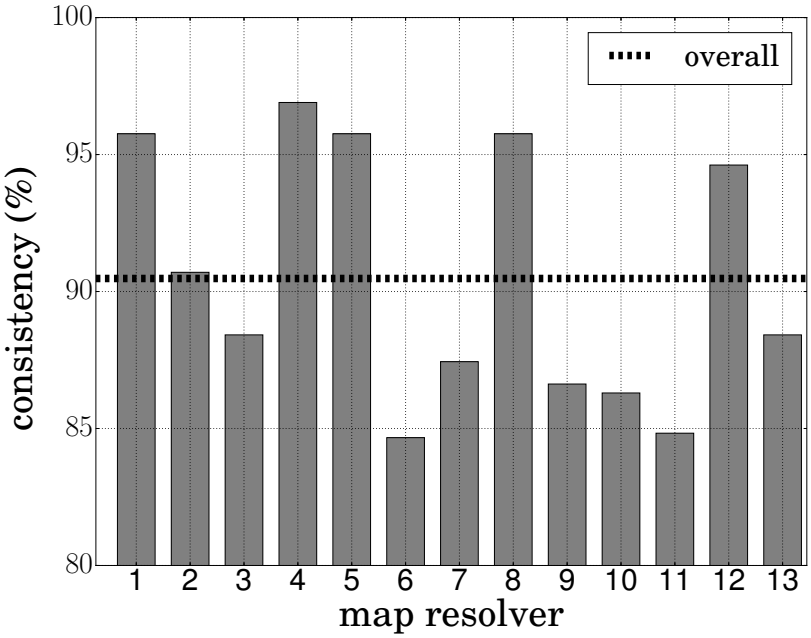

Fig. 4.7 Overall consistency by VP from the different Map-Resolvers.

Just few inconsistencies, only 1.66%, are caused by the RLOC set. This scenario usually occurs for those EIDs whose RLOCs change frequently over time (the case of Statistical outliers instability). So, because of the convergence time of the mapping system, it is very difficult to guarantee that every MR can provide the same LISP Map-Reply at every experiment round. This reason for changes in the RLOC set is most probably due to traffic engineering policies. Similarly as the previous case, when different MRs provide a different RLOC set for a specific EID, the requesting xTRs may end up using different RLOCs to reach the same destination EID.

4.4.2 Consistency Evaluation by VP

The overall percentage of consistency by VP for the whole experiment dataset is 90.48%, presented by the dotted line in Fig.4.7, which indicates that the Map-Replies received at different VPs from the same MR are highly consistent. Yet, we can also observe that different MRs show some deviations from the overall consistency by VP, reflecting the fact that certain MRs send the same Map-Reply to all VPs, but some do not. The deviation among different MRs is however limited (less than 12.23%). Therefore, we can certainly state that the mapping system shows generally a high degree of consistency by VP but presents a non-negligible amount of inconsistencies (9.52%). In the rest of this section, following the same

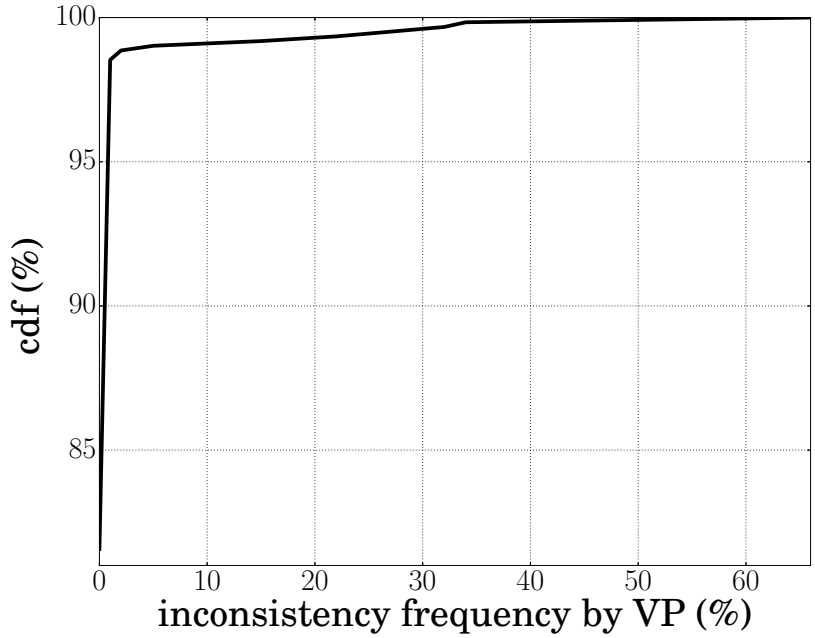

Fig. 4.8 Cumulative Distribution Function of inconsistency frequency by VP. X-axis is inconsistency frequency by VP, indicating the ratio of inconsistency occurrence for each $<EID, MR, *>$ tuple over total experiment number.

approach as for the consistency by MR, we go deeper this issue, exploring the reasons why some observations exhibit inconsistency by VP.

Fig.4.8 shows the CDF of the inconsistency frequency by VP: $f_{<EID,MR,*>}$, i.e., the probability of inconsistency frequency for all $<EID, MR, *>$ tuples, by using the following definition:

$$f_{<EID,*,VP>} = \frac{\text{\# Inconsistencies by VP}}{\text{\# Experiments}} \times 100\%. \tag{4.3}$$

Like for the explanation for Fig. 4.6, here an inconsistency frequency by VP of 0% means that all Map-Replies received by all of the VPs from a same MR are totally consistent at every experiment round during the whole experiment campaign, while an inconsistency frequency of 100% means that at least one of VPs received an inconsistent Map-Reply compared to the other VPs at every experiment round. The probability of consistency by VP (when X-axis is 0% in Fig. 4.8) is similar to that by MR, with a value of 81.57%. It can be noticed that this percentage is slightly lower than the overall percentage of consistency by VP in Fig. 4.7. The reason is that here the inconsistent occurrence with 0% only includes those EIDs for which all VPs always show consistency, no matter which MR is actually queried. A sharp increase with a growth of 16.97% is found when X-axis is 1%, which means that although VPs sometimes receive inconsistent Map-Replies at several experiment

rounds, the occurrence remains pretty low, hence showing that even by VP the mapping system exhibits a very high degree of consistency. Differently from the largest inconsistency frequency by MR, which goes up to reaches 100%, the largest inconsistency frequency by VP is only 66%. This phenomenon indicates a fact that the Map-Replies received by different VPs are not inconsistent all the time, at least there are one third of observations show that their Map-Replies are totally same. Thus the consistency by VP is higher than consistency by MR.

To further explore the inconsistency by VP, we then use same metrics as in Sec. 4.4.1, with the exception that the queried target changes from <$EID, *, VP$> to <$EID, MR, *$> tuples.

The most inconsistent case, 89.06%, is due to a mix of Negative Map-Reply and LISP Map-Reply, i.e., demand for a same <$EID, MR, *$> tuple, at some VPs receive Negative Map-Reply but at others receive LISP Map-Reply. This is caused by the New Deployment case, but differently from the cause for inconsistency by MR, the reason for inconsistency by VP is probably because of Map-Requests sent by different VPs can not be processed exactly at the same time, due to queuing in the MR. Thus, if the queries are sent during the period of mapping information update, some VPs, whose Map-Requests are processed first will have different Map-Reply Type from the latter ones. Inconsistencies caused by different RLOC sets, i.e., different VPs receive LISP Map-Replies with different RLOCs set from the same MR, also exist, contributing for around 10.94%. It is probably caused by a combination of the Statistical outliers instability case and the queuing mechanism in MR. As the RLOC sets provided by a MR for some certain EIDs frequently change, and the Map-Requests sent by different VPs need to be processed one by one, it is very possible that the first processed Map-Requests are replied by the different RLOC sets compared to the latter ones. The difference between Map-Replies is probably due to traffic engineering policies, but the further experiments are needed to explore the deeper reasons for such inconsistency.

4.5 Summary

During the evaluation of LISP mapping system performance, besides the observations about the stability and consistency that we describe in details in the previous sections, we also investigate some interesting LISP-related results. We provide them here so that they can be used to improve LISP performance. We identified that 8.97% of the LISP-sites leverage on multiple RLOCs in order to be multihomed to the Internet, with an average of 3 distinct locators. Interestingly, when we compare the locators with their Autonomous System Number we found that around 50% of the locators in multihoming situations belong to different

ASes, which indicates that LISP is used to transparently interconnect LISP-sites by the intermediate of multiple ISPs.[3] As the control-plane (i.e., MS/MR and mapping system) and the data-plane (i.e., ETR/ITR) are decoupled in LISP, Map-Replies can originate from the LISP-sites directly or from shared utilities constituting the mapping system (i.e., MS/MR). This particularity of LISP is already used in the current deployment, where only 88.8% of the non-negative Map-Replies are originated directly from their LISP-sites, with the remaining 11.2% being originated by delegated MSes replying on behalf of the LISP-sites. This allows reducing the control traffic seen by LISP-sites and help them to scale better.

In addition to mappings, LISP also relies on encapsulation, meaning that with LISP traffic can be transported over IPv4 or IPv6 without requiring end-hosts to be dual-stack. Interestingly, we identified that no less than 8.84% of the locators are IPv6. However, we have not observed cases where a LISP site was reachable strictly using IPv6 while 82.79% of sites are reachable only through IPv4. This discrepancy tends to confirm that network operators still lack of confidence being reachable only through IPv6. As a result, the 17.21% of LISP-sites explicitly using IPv6, always include at least one IPv4 address in order to be reachable.

As a solution to cope with the increase of BGP routing table, traffic engineering, and scalability issues, LISP has been widely deployed on the LISP Beta Network experimental testbed for ten years. To ensure accurate and efficient communications, LISP mapping system should be stable and consistent. However, there is no study evaluating the stability and consistency of LISP mapping system. For such purpose, we continuously measured the LISP Beta Network for seventeen days to assess the mapping system from these two aspects. Measurements show that the LISP mapping system is stable and consistent over time. Nevertheless, instability and inconsistency are observed and for studying them in details we developed a new taxonomy. All in all, instability and inconsistency are rare events. At last, we observe utilization of multi-homing and dual-stack (i.e., IPv4 and IPv6) during the whole experiment and plan to use these results to further improve the LISP performance.

[3] We used the Team Cymru BGP database to associate IP addresses to ASNs (see http://www.team-cymru.org/IP-ASN-mapping.html).

Chapter 5

LISP-Views: LISP mapping system monitor

Currently only two experimental platforms have already been deployed in the Internet: LISP Beta Network and LISP-Lab. However, only the LISP Beta Network is monitored with LISPmon that monitors the mapping system once a day. To accompany the growth of LISP, a dynamic and complete monitoring system is required. Therefore, we propose LISP-Views, a dynamic versatile large scale LISP monitoring architecture. LISP-Views allows to automatically conduct comprehensive and objective measurements. After running LISP-Views in the wild for several months and comparing the monitoring results with LISPmon, we confirm that LISP-Views provides more detailed and accurate information. We observe the different behaviours between every network entity within the mapping system, and also explore the current LISP performance for further improvements.

In the remainder, Sec. 5.2.1 introduces LISPmon and why we propose a new LISP monitor. Sec. 5.2.2 describes our proposed LISP monitoring architecture in details. Sec. 5.3 validates LISP-Views by comparing with LISPmon. Sec. 5.4 provides the snapshot of what kind of further analysis can be done with our proposal. Finally, Sec. 5.5 concludes the chapter.

5.1 Introduction

To promote the development of LISP and boost the related research, large scale flexible platforms are necessary. Two LISP-related platforms have been interconnected so far. The experimental LISP Beta Network testbed [12] is deployed since 2008, and the LISP-Lab platform [15] is open to external experimenters since 2015. Currently, a unique LISP mon-

itoring system called LISPmon [19] supervises the global MDS and publishes the mapping information daily. However, it is known that the mapping information sometimes changes frequently within a day and that the elements constituting the MDS are not always consistent [82]. We hence propose a dynamic versatile LISP monitoring architecture, namely LISP-Views, to overcome these limitations. LISP-Views automatically explores the whole MDS every 2 hours and stores the detailed mapping information, so to facilitate the experimenters to evaluate the LISP comprehensive performance.

We used a one-month long set of traces produced by LISP-Views to evaluate its performance and accuracy, including comparing it with LISPmon. We show that LISP-Views is more accurate than LISPmon since it monitors all the MDS elements in parallel. In addition, thanks to its detailed reporting, LISP-Views allows to assess high level metrics of LISP deployments such as reliability, latency, or configuration issues.

5.2 Proposed Monitoring Architecture

5.2.1 Motivation

In order to move LISP forward, we need to deeply understand the behaviour of the different LISP network entities and since the MDS reflects the status of a LISP network as it stores all the mapping information, it is essential to be able to monitor the MDS. LISPmon was the first step towards a systematic LISP monitoring. However, it monitors the MDS just from one vantage point (VP) once per day and only queries one MR. Upon MR issues, LISPmon must be manually re-configured to monitor another MR. Yet the mapping information may be unstable and inconsistent between MRs, i.e., the mapping information sometimes changes within a day, and the Map-Replies from the different MRs may not coincide at a given time [82]. It is similar to the BGP looking glass servers, which do not always provide the same responses for an IP address as the whole routing system may not have converged or because of routing policies. From such point of view, LISPmon has some limitations since it is not able to detect the changes of mapping information within one day, and is not able to show the differences among MRs. Thus, we propose a new versatile LISP monitoring architecture, called LISP-Views, to monitor public LISP deployments, as well as to enable further performance evaluation of LISP defined by the users themselves. In fact, LISP-Views not only can be used to monitor LISP, but also can be extended to be used in non-public networks, such as VxLAN [83].

LISP-Views is an open source implementation and has been designed to fulfill the following objectives:[1]

1. LISP-Views queries to all the working MRs existing in current MDS, in parallel, while LISPmon just queries one, aiming at building a complete view of the current LISP status.
2. LISP-Views periodically monitors all the MRs with arbitrary intervals,[2] while LISPmon just does it daily, aiming at providing information about the mapping evolution at smaller time granularity.
3. LISP-Views supervises the whole MDS without any manual process, automatically reacting to failing components (e.g., unresponsive MRs).
4. LISP-Views obtains the mapping information from all the MRs of the LISP Beta Network as well as LISP-Lab platform, whereas LISPmon prefers to leverage the MR of the former one.
5. LISP-Views is flexible and configurable, with users able to define different monitoring jobs and get the various measurements, whereas LISPmon just publishes the mapping list daily.
6. LISP-Views is designed to be easily deployed in a distributed manner.

5.2.2 Description of LISP-Views

The architecture of the proposed LISP-Views monitoring tools is depicted in Fig. 5.1. LISP-Views consists of several modules with different functions. The *Measurement*, *Report*, and *Raw Data* modules are deployed on a centralized server while, the *Crawler*, *Sonar*, and *Controller* modules can be deployed on several different *VPs*. As for now, LISP-Views is just deployed on one VP in Paris, France, where the monitor is developed and the centralized module is deployed. Nevertheless, we are planing to implement the distributed version of LISP-Views. For this reason, we will not discuss further how to deploy LISP-Views on multiple VPs. All the modules are implemented in *Python* and described as follows:

Sonar module: the main module with two functions: 1. sending the LISP encapsulated Map-Requests and receiving the Map-Replies to/from all the existing MRs based on the standard [48]; 2. storing the received information in *Report* and *Raw Data* with different purposes.

The IP address that *Sonar* uses to query to MRs is selected either from the output of *Crawler*, or from the recorded information in the previously produced *Report*. The reasons are explained in the corresponding modules.

[1] Source code available on Github: https://github.com/SeleneLI/LISP-Views

[2] This interval can be changed by the experimenters to accord their necessity.

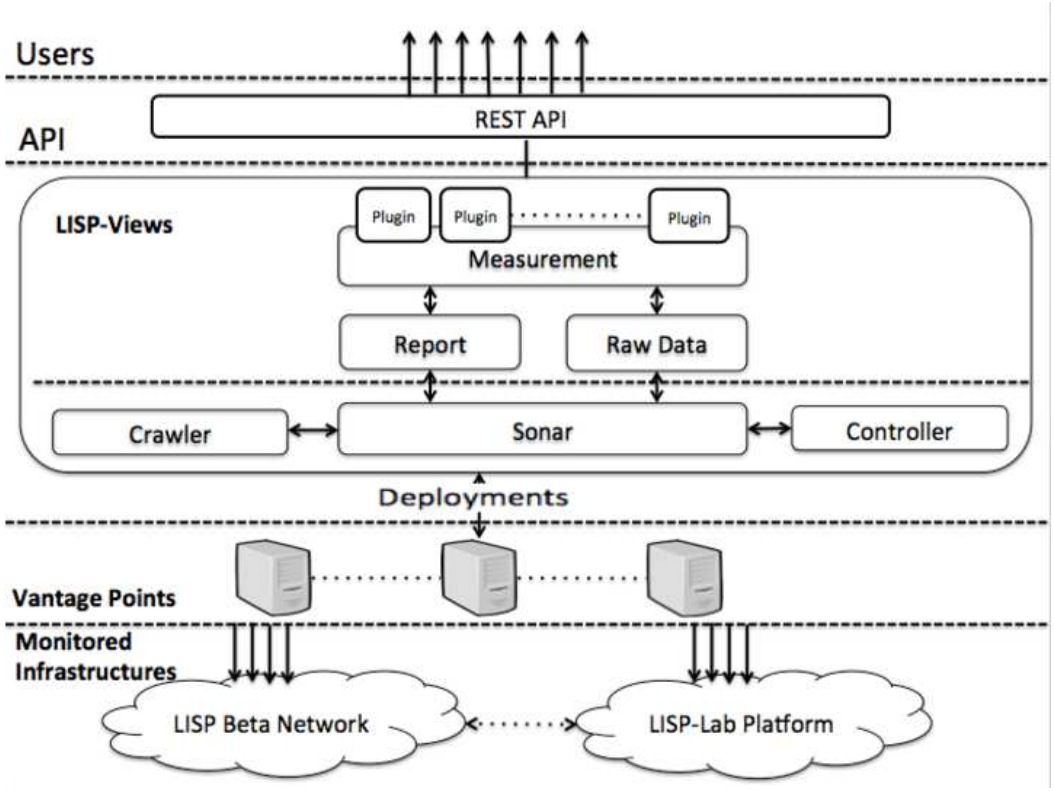

Fig. 5.1 Monitoring Architecture

Crawler module: scans all the existing IPv4 addressing space using *Sonar*. If the queried IP (e.g., 192.0.2.1) has no Map-Reply, *Crawler* increments the IP by one as the next queried IP (i.e., 192.0.2.2) and lets *Sonar* send the Map-Request. If *Sonar* obtains Negative or LISP Map-Reply, since the Reply contains a prefix (mentioned in Sec. 2.1), *Crawler* sets the first IP beyond the returned prefix. For instance if the returned prefix is 192.0.2.0/25, the next queried IP is 192.0.2.128. Scanning the whole IPv4 address space takes very long time, and is mainly caused by time wasted waiting for no Map-Reply. Indeed, no Map-Reply means to wait 3 seconds, and the next queried IP is just increased by one, instead of skipping a block of IP addresses like in the case of Negative/LISP Map-Reply.

Report module: contains the collected EID-prefixes in a list, as well as a list of MRs that answered. As crawling the address space may take long time but we aim to obtain the status of MRs as frequently as possible, *Sonar* sends Map-Requests for the EIDs recorded in *Report* only to MRs that have previously responded. Thus, it decreases the possibility to receive no Map-Reply, so that to get LISP status within a shorter time.

Raw Data module: contains all the detailed information of Map-Replies for each MR, such as Map-Reply type, RLOCs, required EID-Prefix, Round Trip Time (RTT) and the returned source for each round specify with *TimeStamp* (regardless the source of *Sonar*).

So, it can be used to perform thorough performance analysis of MDS. Moreover, since the *Raw Data* is stored according to the MR, it is possible to track the performance of the MRs individually.

Measurement module: provides the composition of requested measurements (i.e., select different measurement plug-ins) based on the analysis of *Raw Data* and *Report*.

REST API: is connected to *Measurement module* so that the users can launch a custom experiment by setting the experiment time, the monitored MRs, and obtain the different aspects of LISP status. We are currently implementing the REST API, so the results of measurements presented here were obtained via the command lines.

Controller module: synchronizes all the modules in LISP-Views and also specifies the start and stop time of producing both *Report* and *Raw Data*. The interval of generating *Report* and *Raw Data* can be changed according to the hardware processing capability. However, since the inconsistency between MRs exits, the interval of producing Report and Raw Data for every MR differs from each other. Thus, the interval set in Control module should cover the slowest MR.

5.3 LISP-Views Validation

5.3.1 Methodology

In order to validate and evaluate the LISP-Views monitoring tool, we used raw data and report collected during one month (from 0:00 September 4th to midnight of October 4th 2016), by deploying LISP-Views on one VP, which is an xTR of LISP-Lab platform.

The interval to produce reports was 6 hours, and the interval to produce raw data was 2 hours. Unfortunately, both MRs of LISP-Lab platform just responded to the Map-Requests at the first time and then stopped. They were not able to handle large number of queries, because the Map-Requests did fill the request queues, resulting a drop in the MRs. Per-se this is a success, because this bug was unknown, and the OpenLISP coders fixed the issue after our measurements. All the results of evaluation used in this section only depend on the 6 working MRs of LISP Beta Network (3 in Europe, 1 in US and 2 in Asia) and the other 6 MRs were down at the moment of conducting the experiment. The aim of the work is to validate LISP-Views by assessing if it provides at least the same results as LISPmon.

5.3.2 LISP-Views vs. LISPmon

In this section, we compare LISPmon and LISP-Views, so to asses if the information provided is comparable between the two monitoring platforms, hence validating LISP-Views.

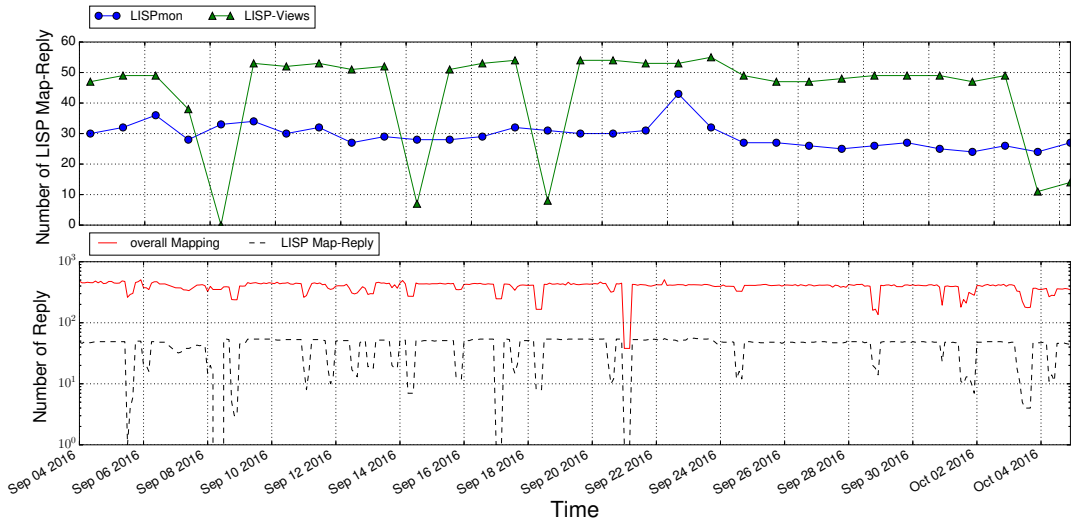

Fig. 5.2 Comparison between LISPmon and LISP-Views. The upper sub-figure shows the number of LISP Map-Reply from LISPmon generally at 7:00 and LISP-Views exactly at 8:00 over days. The bottom sub-figure indicates the LISP Map-Reply and overall Mapping from LISP-Views over time with an interval of 2 hours.

As we indicated in Sec. 5.2.1, at any time LISPmon just queries one MR, on LISP Beta Network once per day and generally begins at 7:00 a.m.. The queried MR is changed sometimes when it has issues. LISP-Views, however, keeps sending Map-Requests to all the 6 MRs with an interval of 2 hours everyday, so to retrieve the actual real status of each MR.

Fig. 5.2 shows the number of LISP Map-Replies received from LISPmon and LISP-Views over time. The upper sub-figure makes a comparison between the 2 monitors. The line with point is obtained from the daily LISPmon publications, indicating the number of LISP Map-Replies returned by one MR in an uncertain time monitoring round every day. To facilitate the comparison, we pick up the results from LISP-Views at 8:00 a.m., which is the nearest experimental round to LISPmon. As presented in the curve with triangle symbols, LISP-Views reflects the fact of a combination of the number of LISP Map-Reply containing the different EID-prefixes from all the MRs at a fixed experiment time. In most cases, LISP-Views receives around 20 LISP Map-Replies more than LISPmon, reflects the fact that the MRs are not coherent at any time due to the existence of convergence time for synchronizing the mapping information among them. The other 5 days, where LISP-Views receives very few LISP Map-Replies, show that there were issues to normally receive LISP status at that time, whereas LISPmon always presents a smoother trend but hides the reality.

The lack of stability can be better appreciated in the bottom sub-figure of Fig. 5.2. Such figure depicts the number of LISP Map-Reply (in dashed line) and the overall Mapping

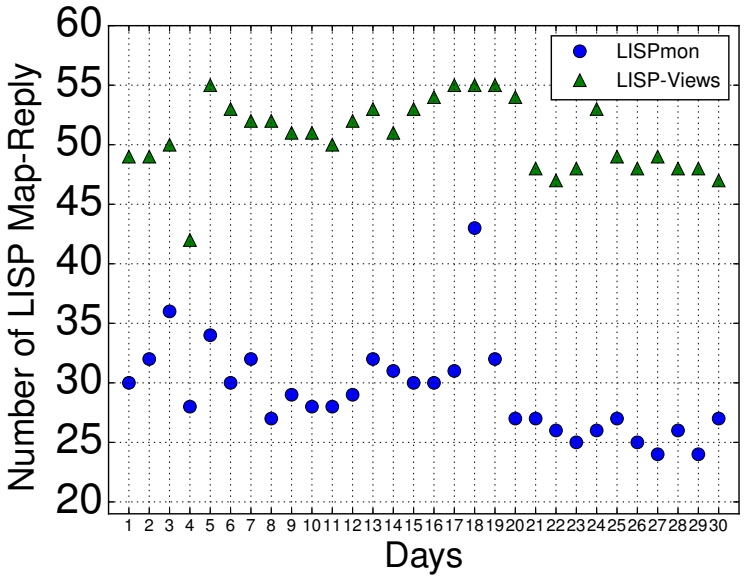

Fig. 5.3 Comparison between the LISP Map-Reply of LISPmon and LISP-Views over days

(LISP and Negative Map-Reply, in solid line) from LISP-Views for every 2 hours monitoring round during the whole experiment. Although the number of Map-Replies presented in the figure is still the combination of results from 6 MRs when the returned EID-prefixes are different, it shows that the MRs are not quite stable. Sometimes the MRs even provide 0 or less than 10 LISP Map-Replies, but in the next experiment round, the number is recovered. Besides, we evaluate the total number of successful queries every 2 hours. It also presents the behavior of instability, where it generally oscillates between 400 and 500, but it occasionally drops lower than 300. Both solid and dashed lines have almost same changing trend, indicating that the change of overall Map-Replies is mainly affected by LISP Map-Replies. Monitoring rounds where the number of LISP Map-Replies approaches 0 but where Negative Map-Replies are still received indicate issues at the MRs, especially for answering the LISP Map-Reply.

We compared the number of received LISP Map-Replies within a whole day between LISPmon and LISP-Views over 30 days. Shown in Fig. 5.3, as LISPmon only publishes one record each day, the number (blue points) is exactly identical to those in Fig. 5.2. LISP-Views, however, provides a combination of LISP Map-Replies not only from all MRs but also from 12 monitoring rounds within one day. We observe that our monitor architecture receives more LISP Map-Replies than LISPmon in all days with a large difference. 83.3% of the time LISP-Views receives more than 20 LISP Map-Replies. The maximum number of LISP Map-Replies that our proposed monitor obtains is 55, while the maximum value

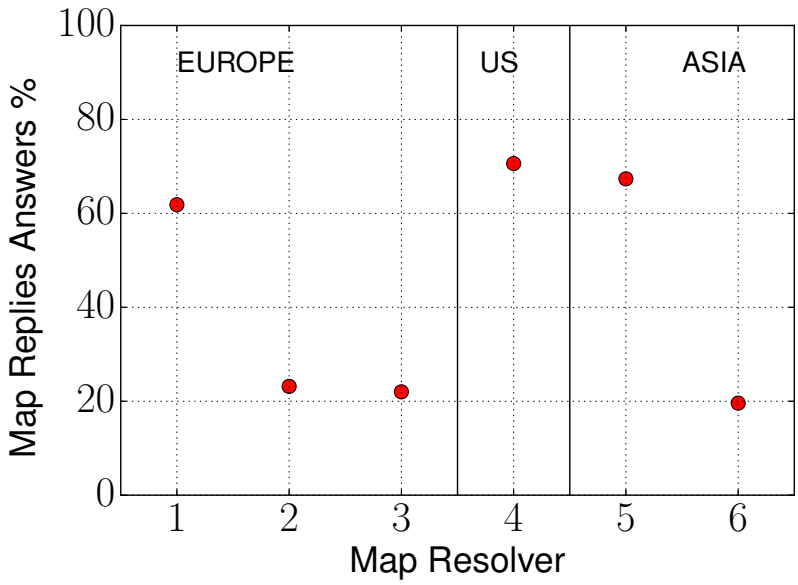

Fig. 5.4 Reliability of each MR

for LISPmon is 43. Differently from the upper sub-figure of Fig. 5.2, where LISP-Views occasionally receives nearly 0 LISP Map-Replies, the total number within a whole day is always more than 40, which illustrates that the MRs may not work normally in some rounds but they are able to recover fast.

Since LISP-Views repeats querying simultaneously to every MRs, it is able to report on the status of each MR at anytime. It provides more complete mapping information compared to LISPmon and is also able to highlight sporadic issues with MRs.

5.4 Dissecting LISP with LISP-Views

After validating LISP-Views in the previous section, this section presents several examples of how LISP-Views can be used to dissect LISP and obtain in-depth results. The experimental dataset used is the one in Sec. 5.3.1.

Fig. 5.4 shows the *Reliability* of each MR during our data collection, by calculating the percentage of successful queries over the total number of Map-Requests. MR1, MR4, and MR5 give the highest reliability values, which are more than 60%. The lowest reliability values, about 20% are observed for MR2, MR3, and MR6. In general, the reliability of each MR is different and low compared to the years of 2012 and 2013 presented in [45], which coincides with the fact that there was a change of the MRs architecture on LISP Beta Network that year, and an updated-software was tested on MRs as well. The low value of

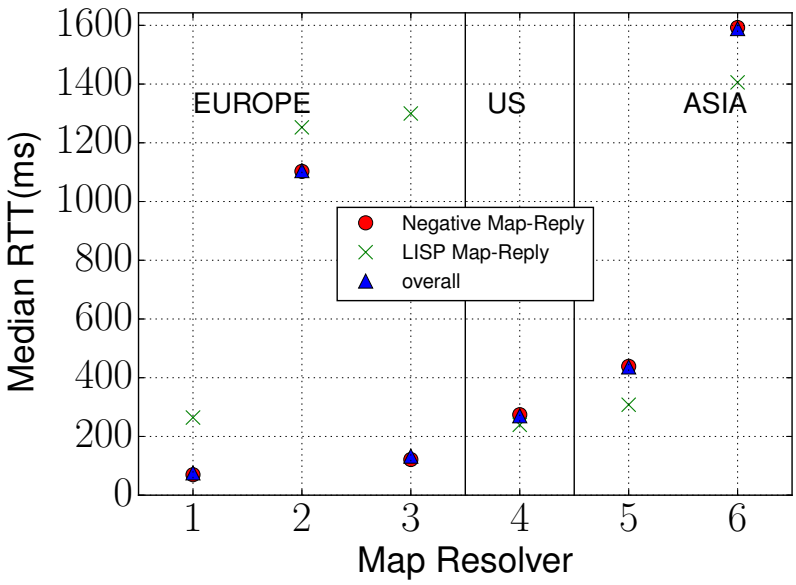

Fig. 5.5 Median RTT per MR (In the most time, the Negative Map-Reply and overall are overlapped.)

reliability is caused by MRs having an unstable behaviour, as shown in Fig. 5.2, the number of Map-Reply changes heavily over time, and especially the number of LISP Map-Reply sometimes drops to 0.

In order to understand the behavior of each MR in terms of latency, we analyze the median RTT obtained from our dataset. The RTT here refers to the Round Trip Time from sending out the Map-Request until receiving the Map-Reply. Fig. 5.5 shows that the best performance come from MR1, MR3, MR4, and MR5; since the overall RTTs are much lower than the others. For MR3, the number of LISP Map-Replies is much higher than the number of Negative Map-Replies, probably because its embedded Map-Server registers less EID-prefixes hence requiring Map-Requests to be forwarded to a remote MS. Then, the Map-Request is forwarded to the xTR, where the remote MS registers and the xTR gives back the Map-Reply. Compared to the Negative Map-Replies, which are normally returned by MR, LISP Map-Replies take longer time, especially in this case. However, MR2 and MR6 present very high latency, particularly for MR2 (located in Europe), but its RTT is even higher than MR5, which is located in Asia. If we focus on the RTT of LISP Map-Replies, only MR1, MR4, and MR5 provide the best behavior, which coincides with the result of Fig. 5.4 on reliability. The high RTTs of LISP Map-Replies are from the other 3 MRs, where the median RTT is around 1300 ms, is caused by the failed queries. Furthermore, it also explains why we can always receive the Negative Map-Replies, but the number of LISP Map-Replies is sometimes quite low in Fig. 5.2. Half of LISP Map-Replies are more

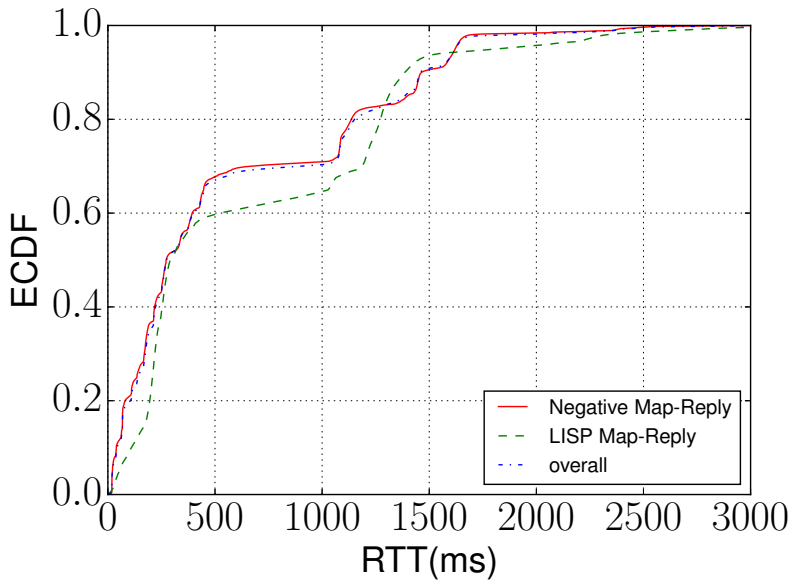

Fig. 5.6 CDF RTT comparison between LISP and Negative Map-Reply

than 1300 ms and partly even more than 3 s. Since our measurement timeout is set to 3 s it implies that some Map-Replies are dropped by our measurement unit. As the RTT of Negative Map-Replies is generally lower, we can receive more of them. This phenomenon may be very biased by the VP location. In addition, Fig. 5.5 presents that the number of Map-Replies returning from the MRs located in Europe and US is indeed higher than the number of Negative Map-Replies, which is expected. But both Asian MRs return LISP Map-Replies faster than Negative Map-Replies. The reason of this observation is unclear and requires further exploration.

We also calculate the Cumulative Distribution Function (CDF) of the RTT for the 6 MRs. Fig. 5.6 provides the CDF of the RTT for Negative, LISP, and overall Map-Replies. As indicated in Sec. 2.1, the LISP MDS needs more time to solve a complete mapping (LISP Map-Reply) than the Negative Map-Reply. We observe that the Negative Map-Reply is indeed faster than the LISP Map-Reply until the RTT reaches 1000 ms, then the behavior changes, i.e., the LISP Map-Reply sometimes becomes faster. What's more, we find that 67.03% of RTTs within 500 ms, 70.33% less than 1000 ms and 90.88% don't exceed 1500 ms for overall Map-Replies. In details, for the Negative Map-Reply, we found that 67.8% of RTT values within 500 ms, for the LISP Map-Reply that 59.7% of RTT values less than 500 ms. This figure almost presents a bi-modal distribution, where 500 ms is a peak and 1300 ms is another one. These two high occurrences of RTTs are exactly the most two frequent latency in Fig. 5.5.

Table 5.1 Percentage of Mapping Source

Map-Reply Type	Map Resolver	xTR	Other
Negative Map-Reply	98.79%	-	1.21%
LISP Map-Reply	0.14%	88.37%	11.49%

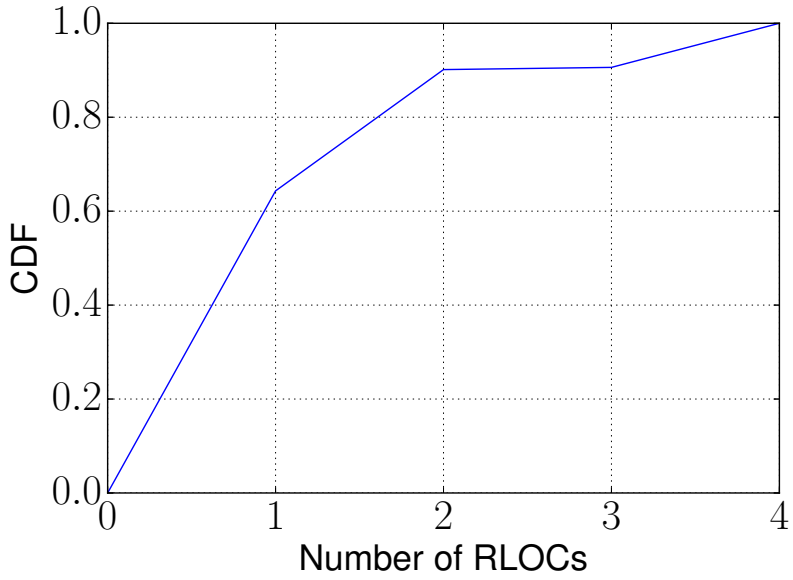

Fig. 5.7 CDF of the number of RLOCs

The dataset obtained with our monitoring architecture also shows the information about *Mapping Source*. We explore the source answering the Map-Replies according to different types (LISP or Negative). In the case of LISP Map-Replies, we expect the source of replies to come either from one of the ETRs or the queried MR. On the contrary, for Negative Map-Replies, replies should just come from the queried MR. Tab. 5.1 presents the percentage of observations for the two types of Map-Replies. For the Negative Map-Reply, 98.79% come from the queried MR and 1.21% come from the other sources. Further, the other sources are actually the other MRs without query and it happens just for a fixed EID-Prefixes, i.e., if we send a Map-Request for one of these EID-Prefixes to a dedicated MR, the Map-Reply always comes from a fixed specific MR. As a conclusion, all the Negative Map-Replies are answered by MRs. For the LISP Map-Reply, we observe that 0.14% come from the queried MR, 88.37% come from one of their ETRs, but 11.49% come from the other sources in different locations. This unexpected behaviour needs further investigation.

The following type of measurement is the distribution of the size of RLOC set, i.e., how many RLOCs are associated to one EID prefix on average. As previously observed, the per-

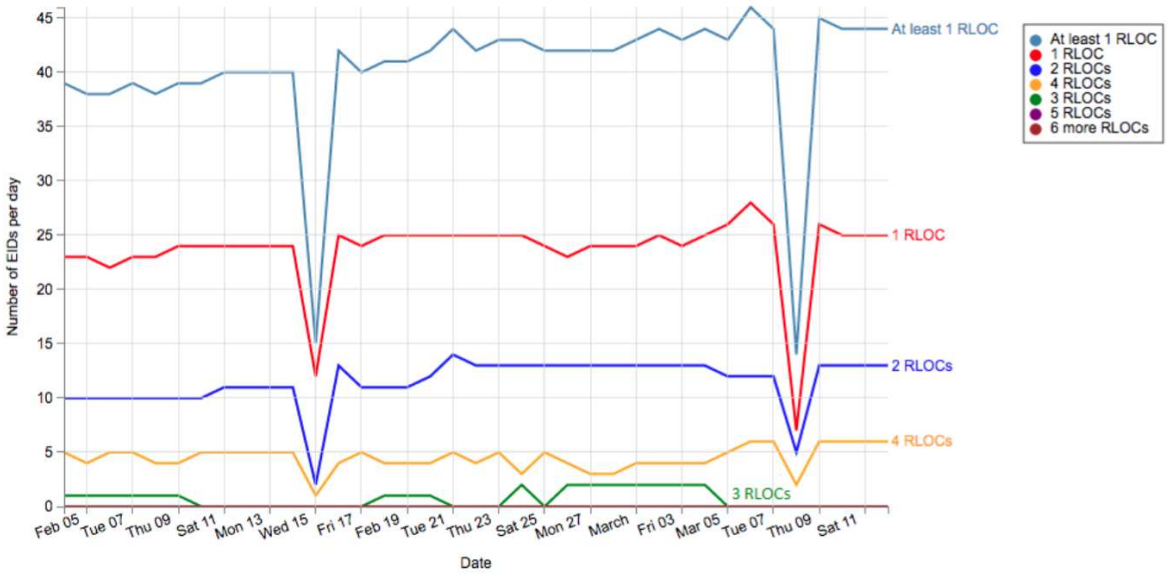

Fig. 5.8 EIDs by Quantity of Associated RLOCs [16]

centage of mappings using two or less RLOCs has increased between 2010 and 2012 [108]. Fig. 5.7 shows that this trend keeps going on, i.e., more mappings use fewer RLOCs and the maximum number of RLOC is 4. An interesting point is that although LISP is a good candidate to support the increasing multi-homing in Internet, more than 60% LISP users are not multi-homed and among them the majority only has 2 RLOCs. Not only the number of RLOCs of each site does not significantly change, but also the RLOCs themselves remains stable. We measured that the stability reaches 99.8% for all the dataset, i.e., once an EID-prefix – to – RLOCs mapping is decided, it rarely changes. We have not found any mobile LISP sites.

All the aforementioned observations are based on the experiment made between September and October 2016 as described in Sec. 5.3.1. Later LISP-Views detects that MR2 and MR6 in Fig. 5.5 with very high overall RTTs are down, and three new other MRs are up. After confirming with the operators of the LISP Beta Network, these two MRs are indeed definitely down. The change of the architecture of MRs also proves the accuracy of LISP-Views, while the LISPmon presents a smooth change in its daily reports, hence hiding these facts. Since February 2017 LISP-Views publicly publishes preliminary daily reporting online [16]. Development efforts are still ongoing to provide a complete and production-level website. Fig. 5.8 is a capture from the website, it is one type of measurement about the number of EID-prefix quantified by the different size of RLOC set. The shown result is an union of all MRs during a whole day, to present the most complete mapping information of the actual LISP deployment. The line on the top indicates the number of LISP Map-Replies,

i.e., the Map-Replies with at least 1 RLOC, which coincides to the results shown in Fig. 5.2 and is mainly affected by the EID-prefix with 1 RLOC. Moreover, the composition of the different sizes of RLOC set is also rather identical to the one in Fig. 5.7. The only difference is that in the previous dataset there is no EID-prefix with 3 RLOCs, while in the latest dataset it appears, but not very stable. In addition, the line with 3 RLOCs is almost complementary to the line with 4 RLOCs. It is probably caused by one LISP-site that has 4 xTRs but one being always down, or the LISP-site having 4 interfaces on a same xTR among which one is down. Besides, the number of observed EIDs per day heavily drops two times in the figure. Since the two valley don't drop to zero, we know that the problem doesn't come directly from the MDS. Instead, the problem comes from issues that occurred within the network of the LISP-Views server itself. This observation highlights the need of distributing VPs.

5.5 Summary

Very little is known about the behavior of LISP in operational environments and it still lacks of troubleshooting tools. Motivated by the only LISP monitor deployed so far, named LISPmon, which records the current LISP status only from a specific MR just once per day, we propose a more dynamic LISP monitoring architecture, namely LISP-Views, so to deepen the understandings on LISP and to ease day-to-day operations and troubleshooting. As LISP-Views aims at being deployed in large scale and dynamic networks, we make a comparison with LISPmon to validate the former one by comparing their behavior during one full month.

It demonstrates that LISP-Views provides more information by discovering more mapping information from all MRs and more complete mapping information of each MR. Furthermore, with our proposed monitoring platform, more mapping system performance metrics such as reliability, latency, and configuration issues can be assessed, which helps for further LISP improvements.

Chapter 6

Assessing LISP interworking performance through RIPE Atlas

Thanks to two testbeds: the LISP Beta Network and the LISP-Lab project, LISP is gradually deployed in the wild. The interworking mechanism is proposed to ensure the communication between LISP-speaking sites and legacy Internet. Although LISP has been evaluated in terms of scalability [108], stability [82], LISP evolution [78], and delay resolving the bindings between EIDs and RLOCs ([108], [45]), LISP is never analyzed from the aspect of the interworking with the legacy Internet at large scale. This dissertation fills this gap by providing a latency evaluation and routing path measurement about LISP interworking mechanism. The work is based on two experiments (one is conducted in 2015, the other one is in 2016) by using RIPE Atlas, which is the largest existing Internet measurement infrastructure for both IPv4 and IPv6.

The experimental results confirm that the use of proxies to connect LISP and non-LISP sites introduce negative effects, which are important for the nearby destinations but can be ignored for the intercontinental long-distance destinations. It also shows that the selection of the proxy location is very important, since having them close either to the sources or to the destinations can decrease a lot the negative stretch. Although LISP introduces some overhead, the performance is quite stable for IPv4. However, the same conclusion does not hold for IPv6. We also observed that the interworking performance of the LISP-Lab platform is more reliable than the LISP Beta Network.

In the remainder, Sec. 6.1.1 and Sec. 6.1.2 introduce the necessary resources on which our experiment leveraged. Sec. 6.2 describes the methodology we used to conduct the experiment. Sec. 6.3 and Sec. 6.4 respectively present the IPv4 ping results obtained from 2015 and 2016. Sec. 6.5 shows the IPv6 ping results and Sec. 6.6 indicates the traceroute results for both IPv4 and IPv6.

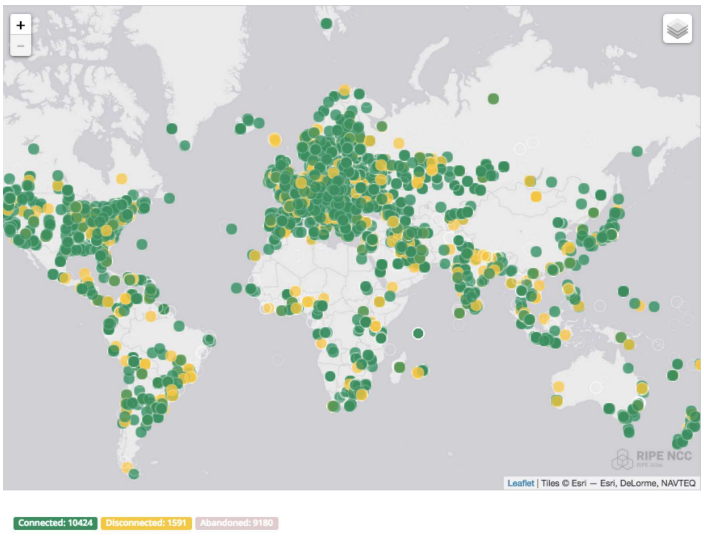

Fig. 6.1 Deployment of probes on RIPE Atlas in 2017

6.1 Related Work

6.1.1 RIPE Atlas

RIPE Atlas [28] is the largest Internet measurement infrastructure, consisting of a global network of more than 9000 probes all over the world that measure Internet connectivity, reachability, and provide an understanding of the state of the Internet in real time. The deployment of worldwide probes is shown in Fig. 6.1. From early 2013, the hardware of the probe is a modified TP-Link wireless router (model TL-MR 3020) with a small USB thumb drive in it, but this probe does not support WiFi. Atlas also has more than 200 worldwide anchors, which are enhanced probes, offering more process capacity and sufficient bandwidth to support the larger number of measurements. Thus, the anchors are normally more stable, and can be used as reference. All the probes and anchors whenever are set up, they automatically start executing a set of pre-defined measurements, called built-in measurements. The set of built-in measurements contain *ping*, *traceroute*, *DNS*, *SSL* and some *HTTP* requests, mostly towards well-known targets such as DNS root servers, but also towards some of the RIPE Atlas infrastructure components. Besides, the probes and anchors also provide the user-defined active measurements with the same measurement types. Volunteers all over the world host these small hardware devices to actively measure Internet performance. To avoid malicious attacks, credits are required to launch experiments and the amount of required credits varies according to experiment types. In addition, the maximum allowed number of measurements towards the same target is 10. RIPE Atlas provides a set of RESTful API with which experiment campaign parameters (e.g.,type of measurement, query source and desti-

nation, the duration and interval of experiment etc.) are passed to probes and measurement traces can be retrieved. Based on the available API, one can schedule experiment campaign in an automatic manner.

6.1.2 Alexa

Alexa [1] [2] is a website created by Amazon which provides commercial web traffic data, global rankings, and other information on 30 million websites. The analytic such as website rankings is based on the data collected by a toolbar developed by Alexa and installed within users web browser. The toolbar provides functions such as popup blocker, a search engine,etc. In early 2015, Alexa stated that there had been 10 million downloads of the toolbar. Alexa ranks sites based primarily on tracking a sample set of Internet traffic—users of its toolbar for the Internet Explorer, Firefox and Google Chrome web browsers. Due to its huge sampling space, the website ranking that it published is widely used to evaluation the popularity of websites.

6.2 Measurement Methodology

The mechanism of LISP interworking is presented in Sec.2.2 and Coras et Al. [45] describe how the use of PxTRs introduces a stretch in the path between LISP and non-LISP networks. Thus, it is necessary to conduct a large scale experiment in the real world to quantify this overhead. The objective is to answer three questions through such an experiment: how much is the negative impact? Under which conditions such stretch has an important impact on the performance? Conversely, under which situation such overhead is so small that it can be ignored? The follows describe how we conduct our experiments.

6.2.1 Dataset 2015

As the purpose of our experiment campaign is to fully obtain the knowledge about the LISP interworking performance with legacy Internet, we deployed a probe (RIPE Atlas probe number is #22341) with IPv4 and IPv6 address on the LISP-Lab platform inside an academic institute in Paris, France to conduct the LISP-enabled active measurements. For IPv4, it uses both PETR and PITR of LISP-Lab to communicate with the legacy Internet, and the connection between the xTR of probe and PxTR is via a MPLS VPN. However, the MPLS tunnel did not support IPv6 at that time. Thus, we configure the ITR of the LISP-Lab probe natively forward the packets towards the Internet core for IPv6 targets (i.e., PETR is not used for IPv6). As a result, the IPv6 packets outgoing from this probe are

Table 6.1 Different configurations of probes in 2015

Name	using LISP	network type	probe/anchor
LISP-Lab	yes	academic	probe
mPlane	no	academic	probe
rmd	no	industrial	probe
FranceIX	no	industrial	anchor

not encapsulated into LISP packets and natively forwarded in the traditional way, but the returned packets still pass through the PITR of LISP-Lab platform.

The mPlane probe (#13842) resides in an academic network and uses the conventional routing. It allows to compare with non-LISP academic networks. While the rmd probe (#16958) resides in industrial network and also uses the conventional routing. It is chosen in order to compare with non-LISP and non-academic networks. Further, a stable probe with much more measurement capacity is necessary as a reference with all other probes. To this end, the only anchor in Paris named FranceIX (#6118) is selected. It resides in an IXP (Internet exchange point) network and does not use LISP. Thus, in this experiment, there are in total 4 probes used as sources to conduct the active measurements. Tab. 6.1 summarizes the different probes.

We are now in a setup phase, where we start with a reduced number of destinations so to first setup the automated experiments. In this first experiment, the selected 4 probes ping to the top 50 Alexa sites every 10 minutes during 6 hours. However, from the experiment we find that there are 14 websites resolve to the same IPv4 addresses. We filtered them, so the assessment presented in Sec. 6.3 are analyzed by the results of 36 top popular websites. With the collected dataset, we evaluate the various performances leveraged on RTT.

6.2.2 Dataset 2016

The new experiment still leverages on the LISP-Lab probe and RIPE Atlas infrastructure, but provides the following new contributions:

- Add another LISP probe connected to the LISP Beta Network as experimental source, to compare LISP-Beta Network and LISP-Lab.
- Enlarge the number of destinations to the first 500 most popular websites on the Alexa ranking [1].
- Consider the performance of IPv6 besides IPv4.
- Use *traceroute*, in complement to latency measurements, in order to have a deeper understanding of the observed behavior.

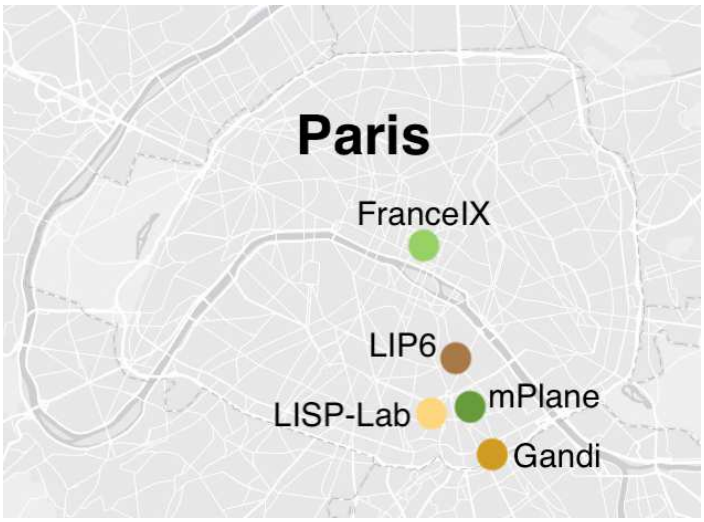

Fig. 6.2 Locations of probes and anchor in 2016

Table 6.2 Location of LISP Beta Network PxTRs in 2016

Number	Continent	Country
1	Europe	Netherlands
2	Europe	Denmark
3	Europe	Norway
4	America	US
5	America	US
6	Asia	Japan

- Extend the experimental span to 15 days so to study potential periodicity of traffic.

The newly added LIP6 probe (#2403) also connects to academic network behind both LISP Beta Network and LISP-Lab platform. When this probe communicates with the hosts in legacy Internet, it uses PETR of LISP-Lab by configuration, while return traffic goes through one of 6 PITRs of LISP Beta Network according to the BGP behavior. Depending on the BGP announcement to the legacy host, a different PITR is used. The more precise location of PITRs on LISP Beta Network is shown in Tab. 6.2: 3 are in Europe (Netherlands, Denmark and Norway), 2 are in US, and 1 is in Asia (Japan). Both LISP probes have a MPLS tunnel from their ITRs to the PETR at Lyon for IPv4, while for IPv6, the ITR of LIP6 probe is configured to use normal BGP routing to reach the PETR in Lyon (i.e., PETR is used but without MPLS VPN). The use of the MPLS tunnel IPv4 packets experience a shorter path compared to the IPv6 BGP-based one. Thus, theoretically the IPv4 packets sending

Table 6.3 Configuration of two LISP probes in 2016

Probe	PETR in Lyon	PITR
LISP-Lab (IPv4)	Via MPLS Tunnel	Lyon (Via MPLS Tunnel)
LIP6 (IPv4)	Via MPLS Tunnel	LISP Beta Network
LISP-Lab (IPv6)	not used	Lyon
LIP6 for (IPv6)	Via BGP Routing	LISP Beta Network

Table 6.4 Different configurations of probes in 2016

Name	using LISP	network type	probe/anchor
LISP-Lab	yes	academic	probe
LIP6	yes	academic	probe
mPlane	no	academic	probe
Gandi	no	industrial	probe
FranceIX	no	industrial	anchor

from the LIP6 probe should arrive to PETR faster than those of IPv6. The configuration of two LISP probes are listed in Tab. 6.3.

Since the rmd probe is not configured with an IPv6 address, we replace it by the Gandi probe (#3141), which also resides in industrial network and also uses the conventional routing for both IPv4 and IPv6. As the mPlane probe and the FranceIX anchor satisfy all the requirements of this experiment, we still keep them as the references. The locations of all the probes and anchor are shown in Fig. 6.2. Tab. 6.4 shows the different configurations of probes.

What we care about is the delay performance and the path of LISP interworking with legacy Internet. Thus, we rely on the ping and traceroute tools provided by RIPE Atlas. As destinations of our measurements we selected the 500 most popular websites (according to worldwide website ranking provided by Alexa [24]), which reply to IPv4 ping and traceroute. Among them, 122 websites are configured with IPv6 address. Thus, in our experiment, 500 IPv4 and 122 IPv6 addresses are the destinations of ping and traceroute. We develop a Python script to schedule the experiment campaign by using the API provided by RIPE Atlas. The parameters of experiment campaign are as follows: the experiment campaign lasts 15 days (from December 15^{th} to 29^{th} 2016). The 5 chosen probes ping and traceroute to 500 IPv4 and 122 IPv6 addresses. The intervals of ping and traceroute measurements are respectively 30 minutes and 60 minutes. We want to evaluate LISP interworking performance as frequently as possible, but each probe sequentially launches 622 traceroute measurements,

which last more than 30 minutes. To avoid the heavy traffic burden and guarantee that all the measurements in a same experimental round can be finished before next round, we set the sampling interval of traceroute at 60 minutes. As a summary, the results presented in this journal come from the following 4 experiment campaigns:

- 5 probes ping to 500 destinations during 15 days with an interval of 30 minutes (IPv4).
- 5 probes ping to 122 destinations during 15 days with an interval of 30 minutes (IPv6).
- 5 probes traceroute to 500 destinations during 15 days with an interval of 60 minutes (IPv4).
- 5 probes traceroute to 122 destinations during 15 days with an interval of 60 minutes (IPv6).

From the experiment, we find that some websites actually resolve to the same IPv4 (34 over 500) or IPv6 (47 over 122) addresses. Further, for some destinations, at least one probe (mainly the LIP6 probe) does not get any responses of *ping* during the whole experiment (42 for IPv4 and 10 for IPv6). In this paper, we filtered the anycast destinations and only consider the destinations having the responses. Thus, after cleaning the traces for the above reasons, all the results of *ping* measurements used in the following subsections consists on the 5 chosen probes as the experimental sources and 424 IPv4 and 65 IPv6 responding addresses as the destinations. Differently from the ping measurements, for the *traceroute* dataset, all the successful *traceroute* responses from each probe are kept for further analysis.

6.3 IPv4 Ping results from Dataset 2015

Fig. 6.3 shows the cumulative distribution function (CDF) of average RTTs towards the selected 50 targets. Since the experimental destinations are located all over the world, the range of observed RTTs varies from few milliseconds (ms) to nearly five hundreds ms. When the RTT values are in the small range, especially when RTT is less than 50ms, the latency from LISP-Lab probe is always higher than the three others and the difference is mainly around 10ms. This latency difference is caused by the stretch introduced by the proxy technology (cf. Sec.2.2), since every packet sent between LISP-Lab probe and the legacy Internet has to pass through the LISP-Lab PxTR, which is located in Lyon (approximately 400km away from the probe). As the RTT increases, actually the difference (surprisingly) decreases. When the RTT around 200ms is reached, all probes show basically the same RTT. Around such RTT values, the destinations concentrate in North America. Going further in the high range RTT values, i.e., more than 350ms, when destinations are mainly located in Asia, the probe in the LISP-Lab domain actually shows the lowest RTT. Thus, it indicates that the network connection between the LISP-Lab PxTR to the (asian) intercontinental des-

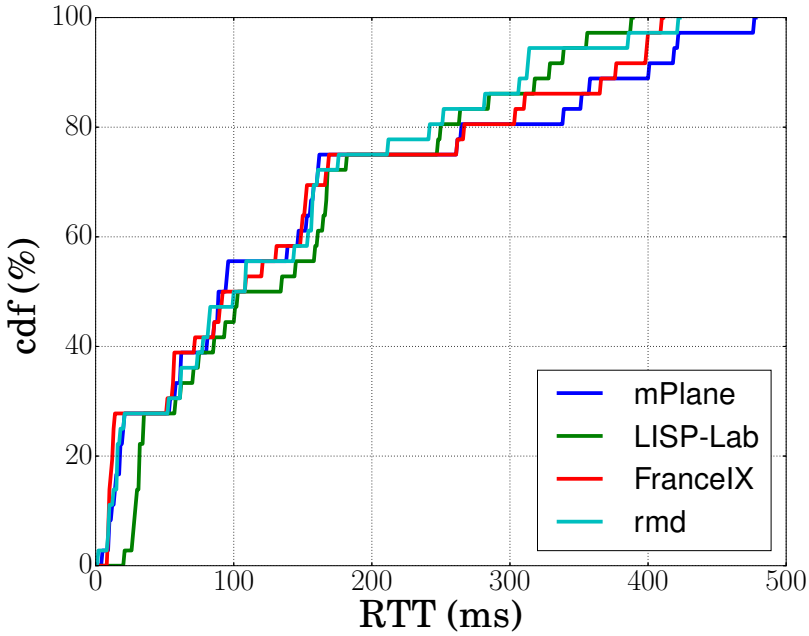

Fig. 6.3 CDF of average RTT between different probes from Dataset 2015

tinations has a better performance and the stretch delay from LISP-Lab probe to PxTR can be ignored. We wanted to use traceroute to further investigate the reasons, but it cannot be natively used at that moment, since the LISP-encapsulation prevents its correct function. We explored a new way to find out what happens for Dataset 2016, which is described in Sec. 6.6.

We tried to quantify the percentage of times that one probe's RTT is the smallest compared to three others, since the destinations that the different probes contact to are exactly the same. The result shows that in 52.8% of the time, FranceIX shows the smallest RTT to reach the destinations. It is normal that its RTTs are the smallest, since FranceIX is an Internet Exchange Point (IXP), hence well connected, and also acts as one of the anchors of RIPE Atlas, thus with a more powerful hardware. While, only in the 5.6% of the cases, the LISP-Lab probe has the smallest RTT. In this small percentage the main contribution comes from Intercontinental destinations, i.e., when the RTT values belong to high range.

Fig. 6.4 depicts the percentage of times that one probe's RTT is the smallest compared to three other probes grouped by continents where the selected targets are located. When the destinations are in Europe and America, FranceIX is the fastest most of the time. Whereas, the percentage of LISP-Lab is always 0. Its higher RTT is caused by the proxy stretch. When the targets are in Asia, LISP-Lab becomes the fastest with a percentage of 20% (mean RTT) and 10% (median RTT). It indicates that such connection from the LISP-Lab PxTR is faster, so that the stretch can be ignored. The performance of FranceIX is not very stable to the

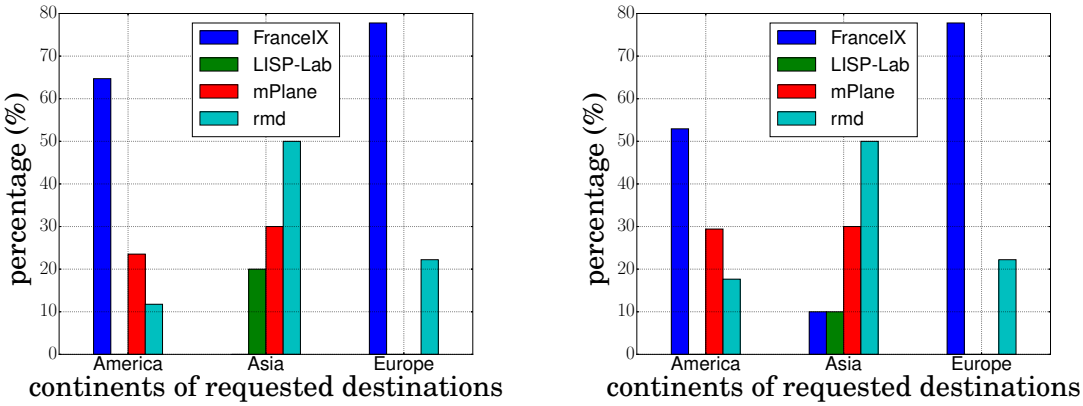

Fig. 6.4 Smallest mean RTT (left) and smallest median RTT (right) grouped by continent from Dataset 2015.

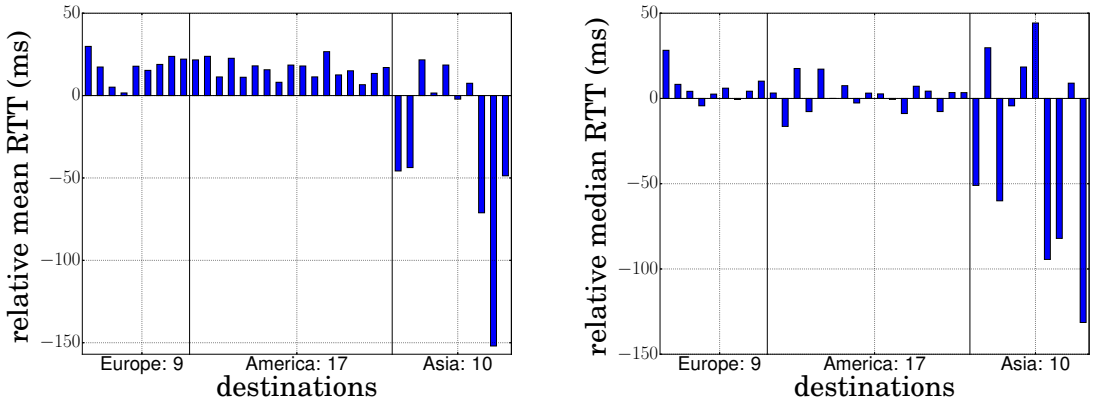

Fig. 6.5 Relative mean (left) and median (right) RTT clustered by different continents from Dataset 2015.

Asian destinations, being the fastest 0% in average, but it is 10% looking at the median RTT. It shows that FranceIX sometimes has extremely high RTT values to Asian destinations.

We also evaluate whether the performance of LISP-Lab is as stable as FranceIX. We define a metric called *Relative RTT (rRTT)* for each destination as:

$$rRTT_{LL}(d) = RTT_{LL}(d) - RTT_F(d) \tag{6.1}$$

where d is the destination, subscriptions LL and F respectively refers to LISP-Lab and FranceIX. The results clustered by continents are shown in Fig. 6.5. The left one shows the mean RTT, while the right one shows the median RTT. For the European and American targets, LISP-Lab is a little slower than FranceIX but with a stable behavior. On the contrary, for half of the Asian destinations, LISP-Lab is significantly faster than FranceIX. It shows that the network connection between LISP-Lab PxTR and Asian destinations has better

Table 6.5 Correlation coefficient to FranceIX from Dataset 2015

	LISP-Lab	mPlane	rmd	FranceIX
Coefficient	0.9733	0.9784	0.9646	1.0

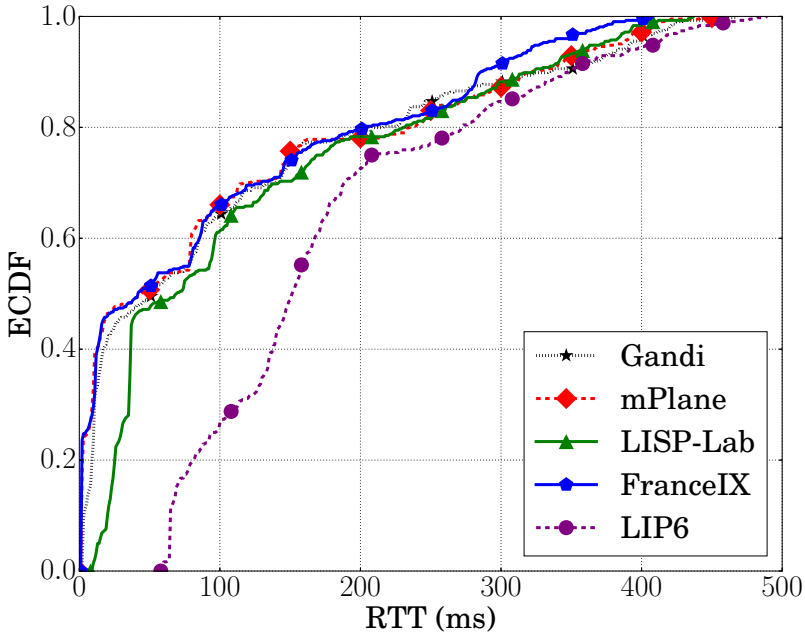

Fig. 6.6 CDF of median RTT between different probes (IPv4) from Dataset 2016

performance. Comparing the two subfigures, there is no negative values at all in Europe and America area in left figure, but there are some in the right one. It indicates that LISP-Lab RTTs to these destinations are very unstable and the variance is quite high.

Since FranceIX shows the best RTT most of the time, we tried to evaluate the correlation between the RTT of the other 3 probes and FranceIX. The purpose is to see if RTT measurements are correlated or totally independent. We compute the correlation coefficient for every RTT series of different probes to all destinations with the ones of FranceIX. Tab.6.5 shows the results. The absolute value of coefficient is 1 in the case of a perfect direct linear relationship, whereas 0 in the case of no correlation at all. The correlation coefficient of LISP-Lab is 0.9733 (> 0.8), showing a very high correlation with FranceIX, like the other 2 probes. This means that the LISP while certainly introducing some overhead, has quite stable performance that does not deviate from normal network operation.

6.4 IPv4 Ping Results from Dataset 2016

Fig. 6.6 shows the CDF of the average RTT toward the selected IPv4 destinations for Dataset 2016. As an anchor, the FranceIX probe outperforms other probes in most cases, hence, showing the smallest RTT. With RTT range $[0, 200]$ms, the latency from two LISP probes, LISP-Lab and LIP6, are respectively 25ms and 60ms slower than three other non-LISP probes. Such a RTT performance degradation is caused by the path length stretch introduced by the the proxy technology (as mentioned in Sec. 2.1), since every packet sent between LISP probe and the legacy Internet has to pass through a PxTR. Although both LISP probes introduce the stretch, LIP6 probe is even 50ms slower than the LISP-Lab probe. Given that both LISP-Lab and LIP6 probes pass through the same PETR located in Lyon, the performance degradation of LIP6 probe is due to the PITR selection for traffic from legacy Internet to LISP network. The PITRs used by LIP6 probe belongs to the LISP Beta Network and there are in total 6 available PITRs. The PITRs announce the LISP Beta prefixes using BGP to their AS peers. So different destinations select different PITRs depending on where the destinations are and lead to the reply goes through the different paths. For example, an Asian destination normally uses the PITR in Japan instead of selecting one in Europe. As listed in Tab. 6.2, LISP Beta Network has not deployed PITR in France, while the PITR of LISP-Lab is in the same country to its probe. Thus, the return path that we measured for the LIP6 probe is longer than for the LISP-Lab probe. As a result, the closer the destination is located to the probes, the bigger the difference of RTT between the two LISP probes.

Within RTT range $[200, 500]$ms, all probes except LIP6 show basically the same RTT. It indicates that LISP-Lab has a better performance in scenario of long distance network connection. The reason is that stretch delay from LISP-Lab probe to PxTR (in Lyon) can be ignored. Fig. 6.6 confirms this situation, within RTT range $[200, 500]$ ms, the difference between two probes becomes significantly smaller than that in range $[0, 200]$ ms.

The aforementioned results and analysis coincide with the Fig. 6.3 from the dataset of 2015, which just covers comparison related to LISP-Lab probe. This indicates that the LISP PxTR performance is stable: the measurement results do not change with the different destinations at the different time.

Even with stretch brought by the PxTR, LISP probes still outperform non-LISP probes in some cases according to the measurement results. To identify these cases, worldwide destinations are classified by continent. Fig. 6.7 depicts the percentage of times that one probe has the smallest RTT compared to the other probes grouped by continents where the selected targets are located. The experimental destinations are located in 4 continents, i.e., Europe, North America, South America, and Asia. The *ping* measurement is repeated 720 times as the experiment lasts 15 days and its interval is 30 minutes. The RTTs of the

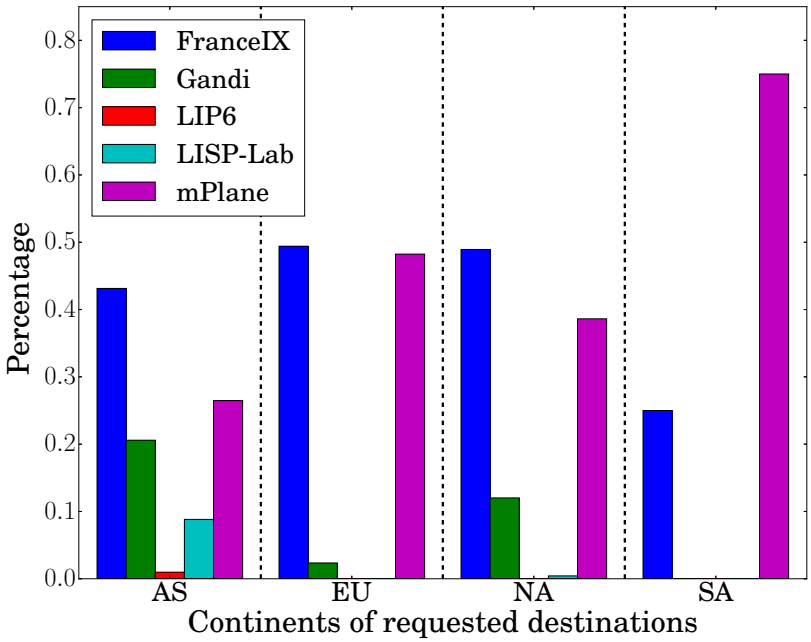

Fig. 6.7 Smallest median RTT grouped by continent (IPv4) from Dataset 2016.

FranceIX anchor and mPlane probe are the smallest most of the time, especially for the European and South American destinations. Only for the Asian destinations the RTT of LISP-Lab probe experiences sometimes the smallest latency for 8.82% of the targets, while the LIP6 probe experiences the smallest latency in 0.98% of the cases. Among the North American destinations, there is just one destination for which the RTT of the LISP-Lab probe is the smallest, while there is no destination for which LIP6 experiences the smallest latency. Fig. 6.6 shows that for RTT above 200*ms* LISP-Lab performs like the other probes, so that we can question whether it actually has the best performance in some cases, and for which destinations. Fig. 6.7 shows the geographic location of destinations for which LISP-Lab actually is the best performing probe. In particular, the LISP-Lab probe has no additional delay compared to other non-LISP probes for the intercontinental transmission to the Asian destinations. It means that the negative effect of LISP stretch delay is evident for communications with European and American destinations, but can be neglected for Asian intercontinental destinations. This phenomenon is similar to the result shown in [79], which shows only in 5.6% of the cases that the LISP-Lab probe experiences the smallest latency and all these destinations are in Asia. It proves again that the negative effect of LISP PxTR can be ignored for the intercontinental transmission to the Asian destinations (when considering the European vantage point).

Also for the Dataset 2016, we try to evaluate the correlation in term of RTT between FranceIX and other probes. The correlation coefficient of LISP-Lab is 0.9766 (> 0.8), show-

Table 6.6 Correlation coefficient to FranceIX (IPv4) from Dataset 2016

	Gandi	mPlane	LISP-Lab	LIP6	FranceIX
Coefficient	0.9647	0.9707	0.9766	0.8547	1.0

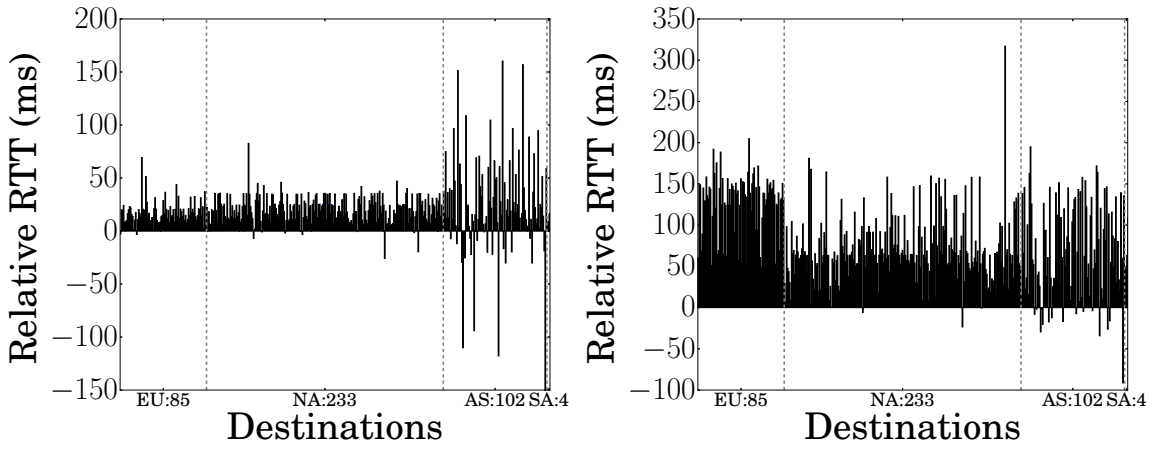

Fig. 6.8 IPv4 Relative median RTT clustered by different continents for LISP-Lab (left) and LIP6 (right) from Dataset 2016

ing a very high correlation level with FranceIX, and even higher than the correlation coefficient of all the other probes. The correlation coefficient of LIP6 is 0.8547, shows a high correlation with FranceIX as well. Both of 2 LISP probes having a high correlation with FranceIX means that while certainly introducing some overheads, LISP has quite stable performance that does not deviate from normal network operation. Such result has two sides for the performance of LISP. Good because LISP shows the stability. Bad because it means that LISP experiences the same performance and failures like the normal Internet.

Another interesting point is to asses whether the two LISP probes have a stable performance as FranceIX according to the destinations. To this end, we leverage on the same metric of *Relative RTT (rRTT)* (formula 6.1), defined in Sec. 6.3 for each destination as:

$$rRTT_{LL}(d) = RTT_{LL}(d) - RTT_F(d) \tag{6.2}$$

$$rRTT_{L6}(d) = RTT_{L6}(d) - RTT_F(d) \tag{6.3}$$

where d is the destination, subscriptions LL, $L6$ and F respectively refers to LISP-Lab, LIP6 and FranceIX. The results clustered by continents are shown in Fig. 6.8, where the relative RTT for LISP-Lab probe is on the left and the relative RTT for LIP6 probe is on the right. The positive relative RTT indicates FranceIX faster, while the negative values indicating that LISP-Lab or LIP6 are faster. In the left side of Fig. 6.8, for the European and American

Table 6.7 Reliability of each probe (IPv4) from Dataset 2016

	Gandi	mPlane	LISP-Lab	LIP6	FranceIX
Reliability (%)	99.66	99.65	99.43	77.78	99.62

targets, the values are between 10ms and 20ms in most cases (i.e., 31.37%), showing that LISP-Lab is a little slower than FranceIX but with a stable behavior. On the contrary, for some Asian destinations, LISP-Lab is significantly either faster (in 20.6% of the cases) or slower than FranceIX and the largest difference reaches 150ms. It shows as well that for the European and American targets, LISP-Lab probe is stable compared to FranceIX, since the difference is mainly caused by the transmission delay between Paris (location of probe) and Lyon (location of PxTR). But when considering the Asian destinations, LISP-Lab does not show a very stable behavior, where the difference might be very large, since the path diversity for long-distance intercontinental transmission is higher, with every path having very different performance. The right figure of Fig. 6.8 shows that for LIP6 probe the difference is almost always positive. For the European destinations, the average relative RTT is around 130ms and much higher than those in the other 3 continents. All European targets show that the LIP6 probe is significantly slower than the FranceIX anchor. For the American destinations, most relative RTT decreases to around 50ms but some of them are around 150ms. For the Asian targets, the relative RTT presents big differences. Some still stay at around 150ms, some drop to below 50ms, and there are 5 destinations even show negative relative RTTs. The reasons causing the relative RTTs for LIP6 probe varying a lot are similar to that for LISP-Lab but not totally the same. The same point is that the longest transmission of traffic in our experiment are the destinations located in Asia. Since the stretch can be ignored in the case of intercontinental transmission, both of two LISP probes are not always slower than FranceIX to Asian targets. The different performance between LISP-Lab probe and LIP6 probe is caused by the location of each PxTR. Although the PxTR of LISP-Lab is not close to its probe, but at least they are in a same country. While the LIP6 probe uses the same PETR to LISP-Lab probe but its PITR is not in France, and is even quite far away. As a result, for the shorter transmission, i.e., to the European destinations, the LIP6 probe introduces extremely higher RTTs than the non-LISP probes and even higher than LISP-Lab probe. Thus, the location of PxTR is very important. The PxTR near to either the sources or the destinations can obviously decrease the stretch. At least the probe having a PxTR in the same country shows a better performance compared to the PxTR just in the same continent.

We also want to evaluate the *Reliability* when having the response to each ping measurement. We measure reliability, in our experiment, by simply calculating the percentage of

replies over the total number of requests. If every ping measurement is successful, i.e., there is a response having the RTT value for a probe at every experiment round, its reliability is 100%. As shown in Tab. 6.7, except for the LIP6 probe with 77.78%, all the other probes are more than 99%. The lower reliability of LIP6 probe indicates that sometimes its ping measurement is not successful. Since it is much lower than the others, to make sure that the LIP6 probe works well and there is no congestion or misconfiguration on the probe or any of its connected routers, we conduct another experiment letting the LIP6 probe use a normal public IPv4 address pinging the same 500 destinations. In this case, the LIP6 probe is non-LISP-speaking. The results show that the reliability of LIP6 probe is 98.91%, which is very close to the other probes. Thus, we eliminate the possibility of the problem on the LIP6 probe itself. As the LIP6 probe uses the same PETR to the LISP-Lab probe, while the reliability of LISP-Lab is very high, the difference is caused by the PITRs. The LISP-Lab probe uses the PITR of LISP-Lab platform but the LIP6 probe leverages one of LISP Beta Network. On one hand, LIP6 probe has longer transmission distance causing more risk of losing the packets and thus leads to more losses. On the other hand, we can conclude that the performance of LISP-Lab PxTR is more reliable than that of LISP Beta Network.

Since the experiment in 2015 just lasts 6 hours and shows no periodicity. Thus, this experiment campaign is extended to conduct with a span of 2 weeks to assess whether the RTT measurements have periodicity. Two methods are used to check: Fast Fourier transform (FFT) and Auto-correlation. However, the analysis results also show that none of RTT series from probes to any destinations exhibits periodicity. It means that the traffic does not periodically fluctuate with a certain interval.

6.5 IPv6 Ping Results

In this section, LISP interworking performance of IPv6 is evaluated with the same metrics as those used in Sec. 6.4.

The CDF of average RTTs to the selected IPv6 destinations is shown on Fig. 6.9, which is generally similar to the Fig. 6.6 for IPv4. The FranceIX anchor still shows the best performance with the smallest delay for almost all the destinations. The RTT of LISP-Lab probe is always higher than the other non-LISP probes with RTT range $[0, 30]ms$ and the difference is around $7ms$. The RTTs are almost the same when the values are more than $30ms$. Hence, the performance of IPv6 is similar to IPv4 that the stretch introduced by PxTR is significant when the range of RTT value is small, but can be ignored when the range of RTT value becomes large.

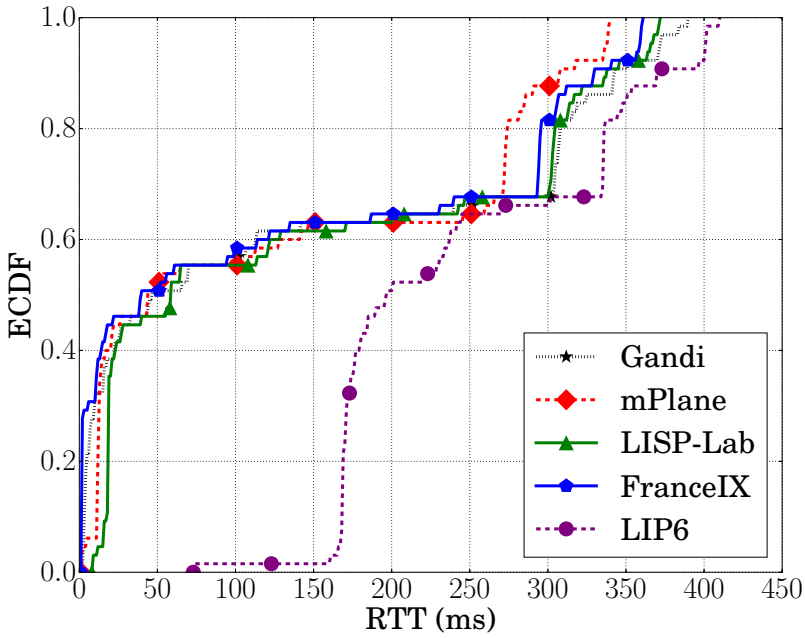

Fig. 6.9 CDF of median RTT between different probes (IPv6) from Dataset 2016

However, in the experiment for IPv6 targets, the traffic produced by LISP-Lab probe are natively forward to the destinations instead of going to PETR at Lyon first, thus the RTT difference compared to the other non-LISP probes becomes smaller compared to those for IPv4. But the latency still exists, caused by the returning traffic that still need to pass through a PITR. Since the latency decreases, it is easier to be ignored for IPv6. That is the reason why the stretch can be ignored when RTT values are higher than just 30*ms* for IPv6 while it needs to be higher than 200*ms* for IPv4. While the LIP6 probe is always the slowest compared to the other probes, the difference is even 160*ms* to the non-LISP probes and 150*ms* to the LISP-Lab probe. The reason having such a big difference is not only caused by the bidirectional traffic should pass through the PxTR, but also caused by the route between xTR of LIP6 probe to the PETR at Lyon, which has no tunnel indicating the used routing path longer than the case of having an MPLS tunnel. Similar to IPv4, when RTT values are higher than 250*ms*, the difference decreases to 40*ms* in most cases.

To further explore why two LISP probes are slower, especially to understand why the LIP6 probe has an extremely large latency compared to the others, we leverage on the Relative RTT between both LISP probes and the FranceIX anchor grouped in four continents as mentioned in Sec. 6.4. In the left figure of Fig. 6.10, which is the relative RTT between LISP-Lab and FranceIX, all the values are positive, indicating that the LISP-Lab probe is slower than the FranceIX anchor no matter to which destination. For the European and North American targets, the relative RTTs are mostly between 5ms and 17ms. But the aver-

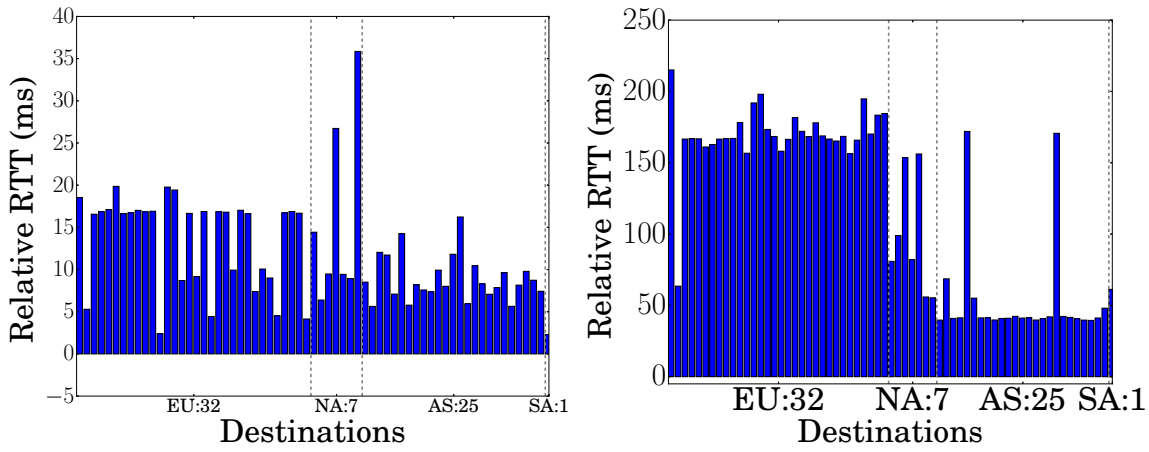

Fig. 6.10 IPv6 Relative median RTT clustered by different continents for LISP-Lab (left) and LIP6 (right) from Dataset 2016

age relative RTTs decrease to 7ms for the 25 Asian destinations. It proves the fact that for the nearby transmission, the delay introduced by PxTR is significant, while it effects less for the intercontinental transmission to the Asian destinations. What's more, the absolute relative RTT values for IPv6 are smaller than those for IPv4, because there is only returning traffic passing PITR for IPv6 so to reduce the latency. Thus, natively forwarding is faster than using PETR, but not so much. From the Fig. 6.10 we also know that there are 32 destinations located in Europe and just 7 in North America. It explains the reason why the curve of CDF for LISP-Lab sharply increases when the RTT is in the small range and mixes with the other non-LISP lines so quickly. It is because a half of destinations are not far away from the probes. Thus, a high percentage (around 50%) consists of small RTT values and these RTTs of LISP-Lab have a consistent delay compared to the other non-LISP probes. The mix part is mainly produced by the Asian destinations, to which the RTT values are large and the delay introduced by only the PITR of LISP-Lab is less significant. The relative RTT for the LIP6 probe is shown on the right of Fig. 6.10. The relative RTTs are all positive and much higher, indicating that the LIP6 probe is always quite slower than the FranceIX anchor to whichever target. Especially for the European destinations, the values are around 170ms. But to the North American and Asian targets, the relative RTTs are obviously smaller. As mentioned in Sec. 6.2.2, the outgoing packets from LIP6 probe still need to go to PETR first and especially the path between its xTR and PETR has no MPLS VPN tunnel. Besides, the main reason that the higher relative RTT values for IPv6 than IPv4 is caused by the LIP6 probe using the PITRs of LISP Beta Network. Different from IPv4 [3], there are only two ASes of IPv6 PITRs and both of them are located in US [4]. As a result, no matter from which destination, all the returning packets need to pass the PITRs in US first and

Table 6.8 Correlation coefficient to FranceIX (IPv6) from Dataset 2016

	Gandi	mPlane	LISP-Lab	LIP6	FranceIX
Coefficient	0.9565	0.968	0.967	0.3734	1.0

Table 6.9 Reliability of each probe (IPv6) from Dataset 2016

	Gandi	mPlane	LISP-Lab	LIP6	FranceIX
Reliability (%)	98.43	99.97	99.98	82.24	99.99

then forward back to the LIP6 probe located in Paris. By consequence, the relative RTTs to the North American destinations are extremely smaller than those to European destinations. Although the traffic returning back from the Asian destinations also pass through the PITRs in US, the distance between the source and destination is quite far, thus it does not effect a lot to the relative RTTs for Asian targets.

Further, in the 84.61% of cases the FranceIX anchor has the smallest RTT in the IPv6 experiment. The LISP-Lab probe and LIP6 are not the fastest to any destinations, even to the Asian targets. It shows that the LISP performance of IPv6 is a little bit worse than IPv4. Since FranceIX has the smallest RTTs in most situations and as an anchor, its higher measurement capacities lead to its higher stability, we also use the correlation between the RTT of the other 4 probes and FranceIX to see if RTT measurements of each probe are as stable as FranceIX. As shown in Tab. 6.8, the correlation coefficient of LISP-Lab is 0.967, which is almost one, indicating although LISP over IPv6 has higher latency than FranceIX caused by the introducing of PxTR, the performance is still stable. However, the correlation coefficient of LIP6 probe is just 0.3734. It is higher than 0.2, but much lower than 0.8, showing that the LIP6 probe is not totally independent from FranceIX, but it has quite low correlation. In fact, the IPv6 RTT series of LIP6 fluctuate a lot over time during the experiment and are not stable.

The Reliability of every probe for IPv6 is almost the same of IPv4 as shown in Tab. 6.9 except for the LIP6 probe. The higher reliability of LISP-Lab probe than the LIP6 probe confirms again that even for IPv6, the PxTR of LISP-Lab is still more reliable than that of LISP Beta Network, although the latter one has 6 world-wide PxTRs in total, from which only 2 can be used for IPv6. The reliability of LISP-Lab probe for IPv6 is a little bit higher than itself for IPv4, showing that using PxTR may decrease the reliability but the effect is very small. The reliability of the LIP6 probe for IPv6 is also higher than itself for IPv4, indicating that the IPv6 performance of PxTR on LISP Beta Network is better than IPv4, mainly thanks to the use of PxTRs in US. Thus from this comparison, we get a conclusion that the two PxTRs in US are more reliable than the other 4 PxTRs of LISP Beta Network.

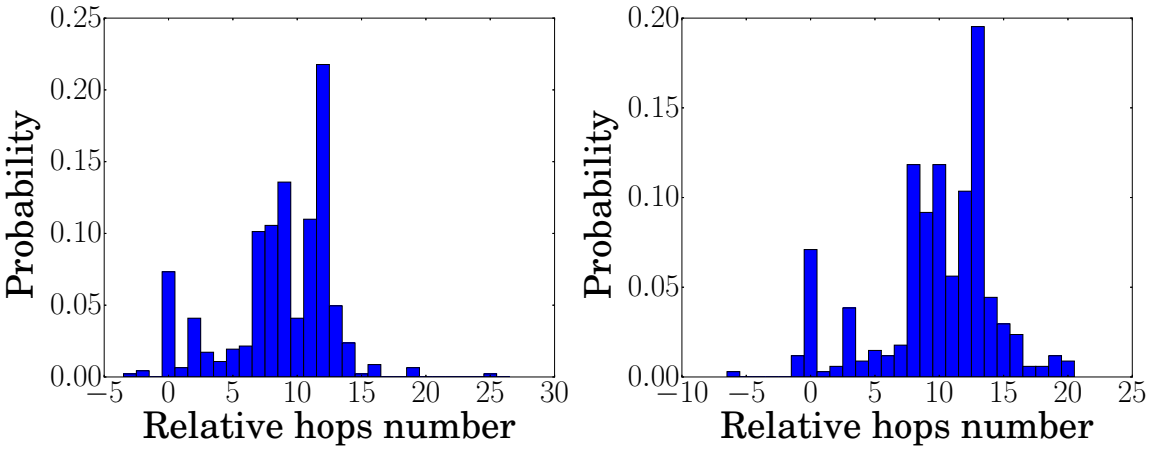

Fig. 6.11 Distribution of IPv4 Relative Hops Number for LISP-Lab (left) and LIP6 (right) from Dataset 2016

The periodicity check for IPv6 is also conducted, but no probes show that their RTT measurements to any destinations have periodicity, either. It indicates that the latency of traffic has no relationship with the specified experiment time, i.e., the RTT measurements generally have the same results regardless the experiment time and duration. Thus, the results of these experiments are not occasional phenomenon with the special results, they are reproducible instead.

6.6 Traceroute-related Results

In this section, we use hops number obtained via traceroute from sources (i.e., probes) to destinations as the metric to further evaluate the LISP performance and also further investigate the reasons for the results presented in the ping-related experiments. Since the traceroute can only present the performance of the outgoing path, i.e., the routing path from probes to destinations, we focus on the use of the PETR and existence of a VPN between the xTR and the PETR.

When the IPv4 targets are in Europe and America, there are very few cases that the hops number of LISP-Lab or LIP6 is the smallest. Precisely, there is 7.65% of cases that LISP-Lab has the shortest path by hops number to the Asian destinations and 6.63% of cases that LIP6 has the shortest path in the experiment. This result coincides to the percentage of the smallest RTT shown in Fig. 6.7. This consistent relationship between smallest RTT and shortest path exists as well for the IPv6 experiment, where there is no destination at all for which the hop number or the RTT are the smallest (for both LISP-Lab and LIP6 probes). For IPv6, it is FranceIX or Gandi that always has the shortest path to all the targets.

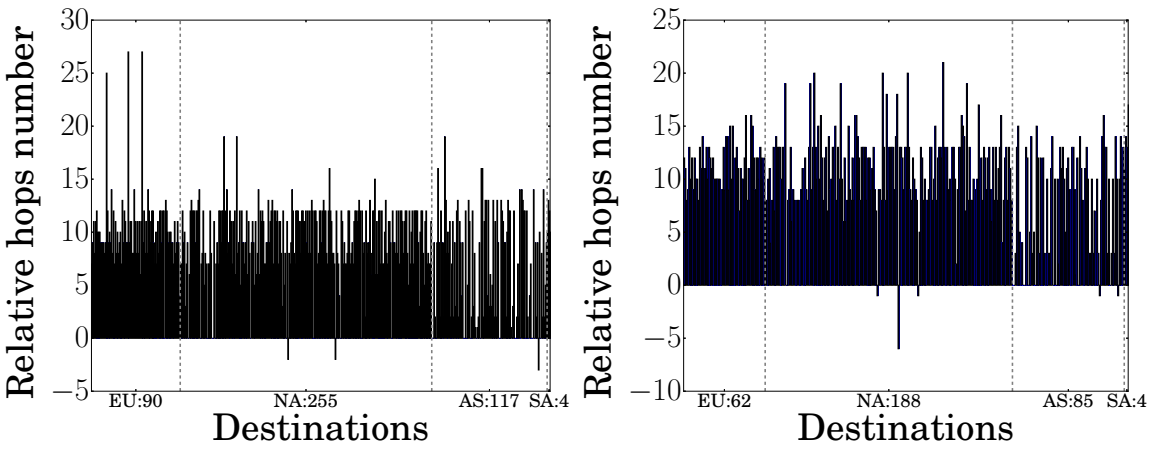

Fig. 6.12 IPv4 Relative hops number clustered by different continents for LISP-Lab (left) and LIP6 (right) from Dataset 2016

As FranceIX is always the most stable probe and has the shortest path in most cases, we define *Relative Hops Number (rHN)* clustered by different continents to look at the difference between the hops number of LISP probes and FranceIX. The definition is as:

$$rHN_{LL}(d) = HN_{LL}(d) - HN_F(d) \tag{6.4}$$

$$rHN_{L6}(d) = HN_{L6}(d) - HN_F(d) \tag{6.5}$$

where d is the destination, subscriptions LL, $L6$ and F respectively refers to LISP-Lab, LIP6 and FranceIX. The positive value indicates that the hops number of FranceIX is smaller than LISP probes. Fig. 6.11 provides an IPv4 distribution of relative hops number for LISP-Lab on the left and for LIP6 on the right. This figure shows the probability of each relative hops number. For the LISP-Lab probe, the most common relative hops number is 12 with a percentage of 21.77%, hence remaining limited in most of the cases. From the ping-related experiment in Sec. 6.4 and Sec. 6.5 we know that using PxTR introduces some overhead. So the Relative Hops Number being generally positive is reasonable. The very large relative hops number just appears once for 19 and 25, so we regard them as outliers. Similar to the LISP-Lab case, for the LIP6 probe, the most frequent relative hops number is 13 in 19.53% of cases.

In order to understand where the most common relative hops number appears, we produce the relative hops number for each destination clustered by different continents in Fig. 6.12 to complete Fig. 6.11. The left figure is for the LISP-Lab probe, indicating that the relative hops number is almost the same for the European and North American destinations and contributes to the relative hops number with high probabilities in Fig. 6.11. The

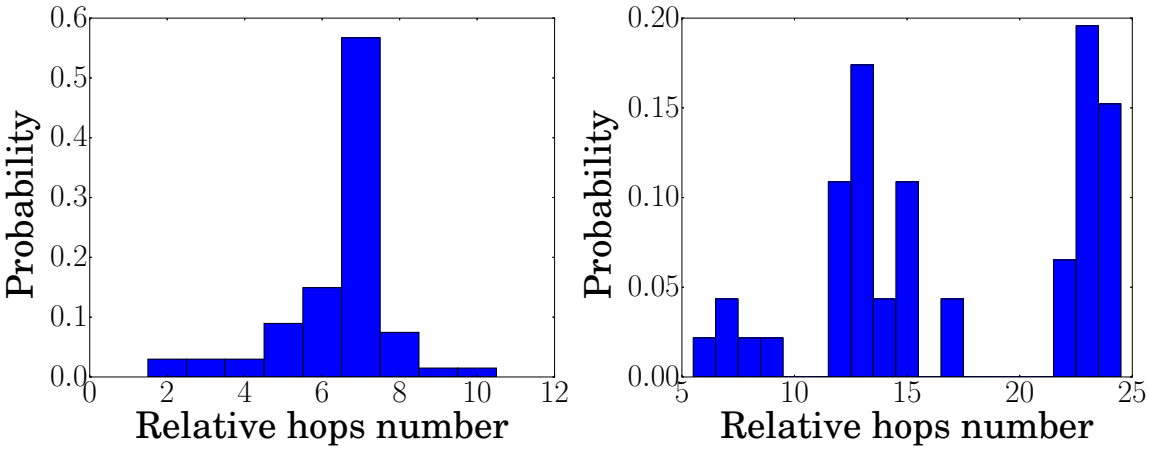

Fig. 6.13 Distribution of IPv6 Relative Hops Number for LISP-Lab (left) and LIP6 (right) from Dataset 2016

small relative hops number is mostly produced from the Asian targets. Whereas the outliers are all from the European destinations. The right figure of Fig. 6.12 is for the LIP6 probe, mainly presenting the same result as the LISP-Lab probe, except that the latter rarely has the relative hops number higher than 16, but LIP6 has more.

With the same evaluation metrics of IPv4, the distribution of Relative Hops Number and the Relative Hops Number clustered by different continents for IPv6 are respectively shown in Fig. 6.13 and Fig. 6.14. The left figure of Fig. 6.13 presents that the most common relative hops number between the FranceIX anchor and the LISP-Lab probe is 7 in 56.72% of cases, more than a half, much higher than the probability of other relative hops numbers. The range of relative hops number is smaller compared to IPv4 and the most common relative hops number is also smaller, since the traffic does not need to pass the PETR, but is just natively forwarded so the overhead is reduced. The left figure of Fig. 6.14 reveals the truth that 7 relative hops numbers are produced by the majority of targets regardless where the destinations are, instead of like the one of IPv4, where the relative hops number for European and American destinations are more than the Asian's. As the relative hops number from PETR is only 3, but 7 for natively forwarding, it is likely that the PxTR has shorter paths to most destinations, leads to the decreasing of hops number for the LISP-Lab probe. The relative RTTs higher for European and American destinations than those for Asia shown in left figure of Fig. 6.10 are mainly caused by the returning path, since it is natively forward for outgoing and there is no difference by continents. For IPv6, the LIP6 probe has extremely high relative hops number in a range from 12 to 24. The majority of relative hops number is 23 in 19.57% of the cases and followed by 13 with a percentage of 17.39%. The higher relative hops number is caused by not using the VPN between xTR and

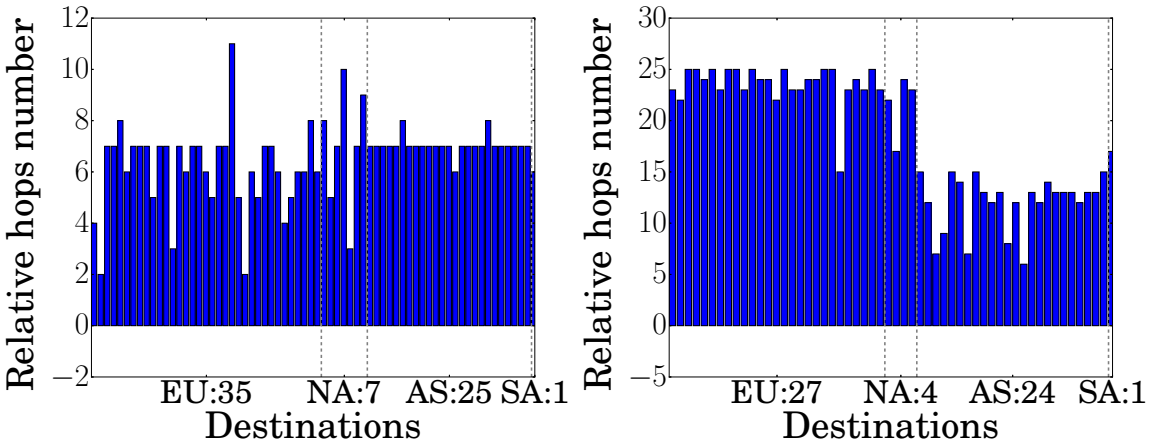

Fig. 6.14 IPv6 Relative hops number clustered by different continents for LISP-Lab (left) and LIP6 (right) from Dataset 2016

PETR compared to the IPv4 case. The right figure of Fig. 6.14 visualizes the 23 relative hops number coming from the European destinations and 13 coming from the Asian targets. The form coincides to the relative RTT for IPv6 shown in the right hand part of Fig. 6.10, indicating that the high relative RTTs are caused by the high relative hops number.

Concerning the RTT performance of the LISP-Lab probe to Asian destinations, which sometimes shows even better than FranceIX, we analyze the AS-path (Autonomous System path) from FranceIX and LISP-Lab to Asian destinations that the packets traverse. Since all the IPv4 packets from LISP-Lab probe are sent to PETR at Lyon first, so the AS-path discussed in the followings is actually from PETR to the Asian destinations. By comparing the AS-path, we find that FranceIX and LISP-Lab very often take different paths, with only the last 1 or 2 hops in common. Even further, by looking at the geographic location of each hop of *traceroute*, it can be observed that FranceIX and LISP-Lab take the path even in the different directions in most of the cases. In particular, there are 11 destinations out of 117 in total (i.e., 9.4%) for which the packets sent by LISP-Lab pass through US, while the packets sent by FranceIX pass through Eastern Europe. There are 68 destinations (i.e., 58.1%), which show exatly the opposite situation, i.e., packets sent by FranceIX pass through US, while packets sent by LISP-Lab pass through Eastern Europe. This last case is when LISP-Lab has smaller RTT (compared to FranceIX), almost all of the times (with only few exceptions).

6.7 Summary

In this chapter, we conduct a six-hours as well as a two-weeks real network measurements with LISP Beta Network, LISP-Lab platform and RIPE Atlas to provide a comprehensive performance evaluation of LISP interworkings. Concretely, we provide a first thorough sight on the performance of LISP PxTR. Since the results of experiment in 2016 coincides to the one in 2015 and are more comprehensive. Thus, we conclude this chapter by mainly presenting the observations of the Dataset 2016.

In our large scale measurement campaign, we take into account 5 probes as sources, 500 IPv4 and 122 IPv6 addresses as destinations, conducting ping and traceroute experiments. We find that the PxTR indeed introduces the negative effects for the destinations located in Europe and America, but the negative impact of PxTR can be ignored for the intercontinental long-distance transmission to Asia destinations. From the experiment, the results show that the position of PxTR is very important. The PxTR either near to the sources or the destinations can decrease the latency a lot. Generally speaking, LISP is stable compared with the reference anchor FranceIX, except for the IPv6 performance of LISP Beta Network. Further, the performance of LISP-Lab PxTR is more reliable than the one of LISP Beta Network, although the latter has 6 worldwide PxTRs used for IPv4 and 2 located in US used for IPv6, whereas LISP-Lab has only 1 PxTR for both IPv4 and IPv6. Compared to leveraging on PxTR, natively forwarding without using LISP decreases the latency, but not much. The traceroute experiment shows that introducing PxTR of course brings more hops, but if the PETR is well configured, so to always have peers to the destinations, there are only 4 more hops compared to the packets being natively forward by xTR without encapsulating with LISP.

Chapter 7

Analysis of LISP mobility and ns-3 Implementation

The *Locator/Identifier Separation Protocol* (LISP) reconstructs the current IP addressing space so to be able to achieve the mobility. LISP Mobile Node (LISP-MN) is based on the basic LISP functionality to provide mobility across networks. Thus, LISP can be implemented either on the border routers or directly on the end hosts to manage mobility. However, there are no experimental results comparing the advantages and disadvantages of each solution. The basic LISP architecture is deployed on LISP Beta Network and LISP-Lab platform to offer the researchers a realistic experimental environment, but both do not support LISP-MN. Some simulation models with LISP extensions are implemented on various simulators, but are not open source. Fortunately, there is an open source project implementing the fundamental LISP on ns-3 in 2016. Providing a free and flexible LISP simulator so to help researchers quickly test new LISP mobility behaviors motivates our work. This chapter analyzes the different LISP mobility scenarios from the respects of handover delay and overhead of LISP Control Plane. It describes the characteristics of each scenario. In addition, this chapter introduces the implementation of basic LISP architecture model and LISP-MN on the simulator ns-3.

The rest of chapter is organized as follows: Sec. 7.1 and Sec. 7.2 respectively introduces ns-3 and the existing LISP simulator on it. Sec. 7.3 analyzes the design and implementation of our prototype, and afterwards, Sec. 7.4 illustrates three different LISP scenarios supporting mobility, presents their traffic schema, modelizes the handover delay and overhead of LISP Control Plane, and compares their advantages and disadvantages. Sec. 7.5 provides some ideas of evaluation based on the proposed simulation for future work.

7.1 NS-3

ns-3 [20] is a popular and free discrete-event network simulator for networking research. It is developed completely in the C++ programming language. The ns-3 architecture is similar to Linux computers with application, TCP/IP protocol stack, network interface, sockets, etc. ns-3 is very well documented and has an active community which facilitate the researches to adapt ns-3 source code for their researches. Besides, ns-3 offers the possibility to visualize the simulation instance so to allow the users to visually confirm the packets flow as they expect.

7.2 Basic LISP implementation on ns-3

Simulation is becoming more important for deploying new technologies or as a proof of concept of new protocols. In the study of LISP, there exist few simulators based on OMNet++ [116, 115, 75] or based on Java [112]. However, these existing simulators are not open-source, which hinders other researchers to modify or adapt the simulator with respect to their own research purposes.

To our best knowledge, the unique open-source LISP simulator that we found in the literature is the one proposed by Agbodjan [32]. The authors implemented a basic LISP simulator under ns-3.24, but this work can be further polished. For example, the encoding of LISP Control Plane messages does not respect RFC 6830 [48] so that the Wireshark [30] can not resolve the captured results in the correct format. More importantly, its implementation has no support for LISP mobility. By leveraging the source code of Agbodjan, we implement an open source LISP simulator with LISP mobility extension. Besides, we also cover the shortage of the original source code. For example, the encoding of LISP control messages is according to RFC 6830 [48] so that Wireshark can correctly decode these messages for analysis. The case of Negative Map-Reply has been covered. Recall that the work of Agbodjan is still under ns-3.24, but ns-3.24 evolved lots from ns-3 to the latest version ns-3.27. This implementation is under ns-3.27, which allows the other researchers to profit the newest functionalities of ns-3 simulator.

7.3 LISP mobility extensions on ns-3

(**[Yue]** *Mention adapt to ns-3.27 and encode for Wireshark.*)(**[qs]** *Personally, I think this kind of words should be put in previous part. When implementing our jobs, we only focus on talking about what we have done. Please refer to previous part to see my modification.*)

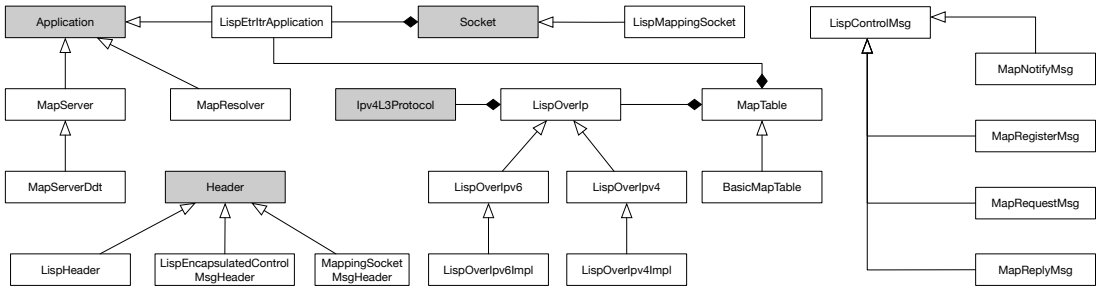

Fig. 7.1 UML diagram of LISP and LISP mobility implementation. The solid arrow refers to a composition relation, while the blank one refers to a inheritance relation.

Our implementation respects LISP RFC 6830 [48] and LISP mobility standards [87]. As a design choice, we implement LISP and LISP mobility functionalities by modifying and extending two already existing modules of ns-3: *internet* and *internet-apps*, instead of by creating a new independent module. The justification behind this design is that LISP protocol and legacy Internet module have an interdependent relationship: an IP layer packet is processed by LISP and then passed to IP protocol again. However, this kind mutually dependent relationship between modules is not supported by ns-3. Our implementation consists of two parts: LISP Data Plane and LISP Control Plane. The communication between LISP Data and Control Plane is achieved via a dedicated socket (i.e. *LispMappingSocket*) that inherits from ns-3 *Socket* class. The Data Plane implementation is in "kernel space" (i.e. ns-3's *TCP/IP stack*) and Control Plane is implemented in "user space" (i.e. ns-3 *Application*). Such a design is inspired by that of OpenLISP [110].

The UML diagram of proposed LISP and LISP mobility implementation is illustrated Fig. 7.1. The white blocks refer to the classes that we added into ns-3, while darker blocks are classes already in ns-3. It should be noted that our implementation keeps the same class names used in the Agbodjan's implementation [32] but rewrite the contents of most classes to support LISP mobility, especially for class *LispEtrItrApplication*, *LispOverIpv4Impl*, and *BasicMapTables*. This work currently only supports IPv4 at time of this writing. The IPv6 support (i.e. the implementation related to IPv6 such as *LispOverIpv6Impl*) is still in process. In addition, the authentication procedure involved in LISP is not considered in our implementation.

7.3.1 Implementation of LISP Data Plane

The implementation of LISP Data Plane mainly consists of *LispOverIp* and *MapTable* classes and their subclasses, along side with some auxiliary classes (e.g. *LispHeader*). In addition, to support LISP functionalites, *Ipv4L3Protcol*'s packet transmission, reception, forward and delivery procedures are accordingly adapted.

LISP Database and cache

Each LISP-speaking node should maintain one LISP database and LISP cache for LISP encapsulation and decapsulation operations. In our implementation, both LISP database and cache are modellized by the same class *MapTable* that stores and manages EID-RLOC mapping information. This class is in charge of CRUD (Create, Retrieve, Update, Delete) operations for mappings. Each mapping entry in LISP database and cache is an instance of *MapEntry*. For the sake of flexibility, the class *MapTable* is an abstract base class. The CRUD methods are implemented in its subclass *BasicMapTable*. The mapping search operation is a straightforward iteration over LISP database or cache. It is possible that for other users to provide their own LISP database and cache implementation, for example, which uses more sophisticated mapping entry look up algorithms, by extending *MapTable* class.

In addition, *MapTable* has a callback which allows to send the buffered packet (either LISP Data Plane or Control Plane)(**[qs]** *Recall that now I add a callback into MapTable and I only implement the resending once the required cache is inserted into cache. WE currently only use this for LISP-control messages (SMR-invoked Map Request). We can easily extend this to support other LISP data plane packets buffering and resending once the required EID-RLOC mapping is obtained.*) due to LISP cache miss event, upon insertion of the required EID-RLOC mapping information into LISP cache.

Implementation of LISP encapsulation and decapsulation

To integrate LISP and LISP mobility into conventional Internet protocol stack, one key technical difficulty is that *Ipv4L3Protcol* should be able to determine when passing a packet being processed to LISP-related procedure and how to retrieve the associated mapping information. To this end, a new class called *LispOverIp* and its extended classes (refer to Fig. 7.1) are added to ns-3 *internet* module. This class is in charge of checking whether it is necessary to do LISP-related operations (*NeedEncapsulation()*, *NeedDecapsulation()*), and encapsulating conventional IP packets (i.e., *LispOutput()*) as well as decapsulating LISP

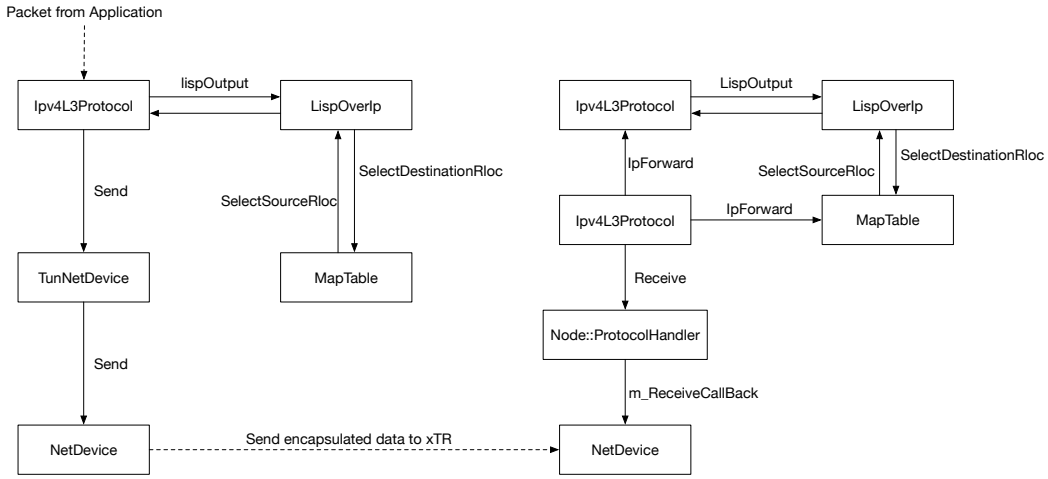

Fig. 7.2 Illustration of LISP encapsulation and decapsulation
([Yue] *Figure is not completed*)(**[qs]** *Good remark. This figure is actually the process of MN, xTR1/2. I forgot to draw to the processing of xTR3 and CN. I will make a draft of the left part and send it to you as soon as possible.*)

packets(*LispInput()*). It contains a smart pointer[1] pointing to the LISP database and LISP cache (e.g. *MapTable*) on which executes mapping search.

We take the double encapsulation example shown in Fig. 2.8 to illustrate how LISP encapsulation and decapsulation is implemented. We assume that the required mapping entries during LISP Data Plane operations are already in LISP-MN cache. A LISP-speaking node behind the xTR_1 needs to communicate with CN behind the xTR_3. Thus, a packet should be encapsulated within the considered node and forwarded to the xTR_1. Subsequently, the packet is further encapsulated and forwarded. At the xTR_3, the received packet is decapsulated twice and forwarded. Thus, the example involving packet transmission, forwarding and reception is illustrated in Fig. 7.2.

With the LISP-MN node, when upper layer calls the *Send()* method of *Ipv4L3Protcol*, a packet comes down to IP layer. The *Send()* method is adapted so that it first verifies whether the *LispOverIp* object is present. If yes, some checks are then conducted to determine that this packet should be processed by *LispOutput()* (to encapsulate the packets) or by conventional packet transmission routine. For example, if both source and destination IP address of this packet belong to the same network, the LISP-related process (e.g., encapsulation)

[1] A smart pointer is an abstract data type introduced in C++ that simulates a pointer while providing additional features, such as automatic memory management or bounds checking.

is skipped and this packet is processed as in a non-LISP network. Otherwise, EID-RLOC mapping information is searched from LISP cache and LISP database on LISP-MN node.

After encapsulation and forwarding, the considered packet is forwarded to xTR_1. Low layer invokes the *Receive()*(**[Yue]** *why* Send() *is small s but Receive() has a capital R?*) (**[qs]** *All send() has been modified as Send(). Remember that in C++, all class method starts with capital letter.*) method of *Ipv4L3Protocol* to pass this packet to IP layer. Since this packet is destinated to a remote CN instead of itself, this packet is processed by patched *IpForward()* method. With this method, LISP encapsulation is verified. *LispOverIp* looks for the source RLOC (RLOC of xTR_1) and destination RLOC (RLOC of xTR_3) for the outer IP header building. Once this step is done, *Ipv4L3Protocol*'s *Send()* method transmits these encapsulated packet to MAC layer for transmission.

When this packet arrives at xTR_3 which servers CN, xTR_3 finds that the destination of this packet is the node itself, the packet is processed by *LocalDelivery()* method in *Ipv4L3Protocol*. Before passing to transport layer, *LocalDelivery* checks if the packet should be decapsulated. If yes, it is passed to *LispInput()* method, in which the packet is decapsulated and re-injected(**[Yue]** *re-injected?*)(**[qs]** *U R right. Modified.*) into the IP stack. That is to say, *Receive()* method is called again after decapsulation operation. If the packet still has LISP header, the aforementioned procedure will repeated until it has no need to be decapsulated. Finally, the packet with source address of LISP-MN and destination address of CN is forwarded to CN.

7.3.2 Implementation of LISP Control Plane

The implementation of LISP Control Plane at least should provide ITR/ETR, MR and MS. In practice, ITR and ETR functionalities are usually placed on a same router called xTR. In our implementation, they are included into class *LispEtrItrApplication*. The functionalities of MR and MS are respectively implemented by class *MapResolver* and *MapServer*. The LISP Control Plane messages (Map-Register, Map-Request, etc.) are represented by the derived classes of *LispControlMsg*. In addition, to communicate with LISP Data Plane, a socket class *LispMappingSocket* is proposed.

Implementation of xTR functionalities

A ns-3 node that runs *LispEtrItrApplication* is a LISP-compatible router. It should be able to communicate with *LispOverIp* on the same node (e.g. inform cache missing event) and other LISP-compatible routers (e.g. Map-Request/Map-Reply). The state transition diagram is illustrated in Fig. 7.3. When destination RLOC is not found in the cache of xTR

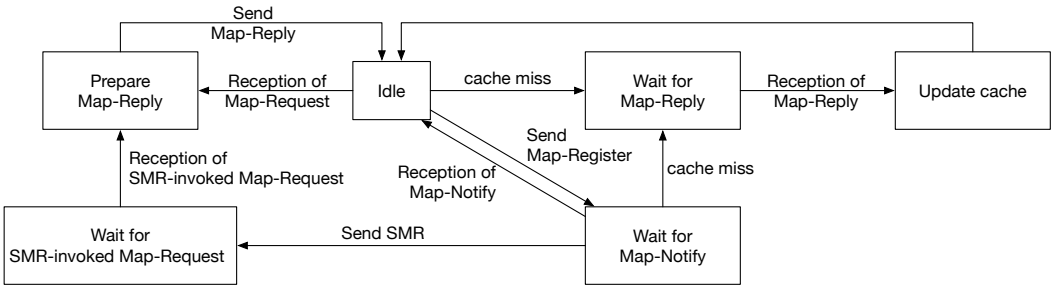

Fig. 7.3 State transition diagram of xTR application

([**Yue**] *why "cache miss" and "send SMR" are sent from "Wait for Map-Notify"?*)([**qs**] *Good remark. First in a state transition diagram, a block refers to a 'state' and an arrow refers to a certain action leading to state change. A state transition diagram identifies all possible states and reflects how a state is transited to another one. Back to this diagram, this state transition diagram is questionable, because this digram reflects the behaviors of the xTR running within LISP-node and the SMR-invoked Map-Request will be sent back to the xTR which previously initiates one SMR. For this scenario, when in state of 'Wait for Map-Notify', it has 2 possible actions: if the required EID-RLOC mapping (to xTR2 for example) is known, it can switch to state of 'Wait for SMR-invoked' map request by action of 'Send SMR'. Otherwise, a cache miss event occurs and this makes the xTR sends Map-Request message and enter into state of 'Wait for Map-Reply'. Conclusion: to be more clear, maybe first change the legend of this figure to "State transition diagram of xTR application running on LISP-MN node. SMR-invoked Map-Request is sent back to the xTR which initiates.". Second, if you think, 'cache miss' is misleading. You can change it as 'Send Map-Request'. In addition, I think modify the state name 'Idle' to 'Listening' is better, because xTR is always listening to a UDP port for the incoming LISP control messages. Up to you to make the change.*)

for a processed packet, the cache miss event occurs and LISP Data Plane (e.g. *LispOverIp*) notifies *LispEtrItrApplication* on the same LISP-MN node([**Yue**] *why the former is xTR but here is LISP-MN?*)([**qs**] *You can delete 'on the same LISP-MN node'. This state transition diagram reflects the behaviors of xTR running on a LISP-MN node.*) of this event via a *LispMappingSocket* socket. Once reception of cache missing event from LISP Data Plane (i.e. *LispOverIp* object), *LispEtrItrApplication* initiates a Map-Request message to LISP mapping system. Once reception of Map-Reply, the received EID-RLOC mapping is inserted into LISP cache. It should be noted that in our implementation, before the reception of Map-Reply message, all transmitted packets with the required RLOC as desination are dropped. There exists one reception: the processing of SMR-invoked Map-Request message. If the RLOC of xTR initiating the SMR (actually a local RLOC) is not found, SMR-invoked Map-Request message is buffered and sent again once the insertion of required mapping

information into LISP cache. This is achievable thanks to a callback associated with LISP cache insert operation.(**[Yue]** *Which one we implement?*)(**[qs]** *We should to drop LISP data plane packets but buffer LISP control plane message (i.e. SMR-invoked Map-Request). I made a modification you can read this sentence again.*)

In case of reception of Map-Request, *LispEtrItrApplication* executes a database look up on *MapTable* and generates the corresponding Map-Reply containing EID-RLOC mapping.

When xTR application starts or LISP database on a node has information update (e.g. during a handover scenario), xTR application sends a Map-Register message and waits for a Map-Notify message. In case of LISP database update, xTR sends a SMR message to all xTR whose RLOC is present in its cache. According to RFC 6830 [48], the xTR receiving the SMR has two possibilities of reactions about sending SMR-invoked Map-Request: towards a mapping system or the xTR initiating this SMR. Both cases are implemented in our work and this can be configured by an attribute of *LispEtrItrApplication*. It should be noted that in double-encapsulation, if SMR-invoked Map-Request is directly sent to LISP-MN node whose LRLOC is unknown for xTR, the first SMR-invoked Map-Request cannot be sent due to cache miss. We implement a callback function within *BasicMapTable* which allows to send immediately the buffered SMR-invoked Map-Request upon insertion of required EID-RLOC mapping into cache. Note that this mechanism is designed only for LISP Control Plane messages. LISP Data Plane is dropped in case of cache miss.

To support LISP-MN feature, *LispEtrItrApplication* also communicates with DHCP client application. For example, once a LISP-MN obtains an IP address from DHCP server, *LispEtrItrApplication* receives the corresponding EID-RLOC mapping and sends a Map-Register message to the Map Server.

Implementation of MR and MS

A node that runs a *MapServer* application is the MS in a LISP-supported network. This class has a smart pointer pointing to a LISP database (i.e. *MapTables*) to store the all EID-RLOC mapping information. This application is always listening to UDP port 4342. Once reception of Map-Register message, it retrieves the EID-RLOC mapping inside, inserts the latter into LISP database and sends a Map-Notify message as response if necessary(**[Yue]** *why if necessary?*)(**[qs]** *Because only a certain flag of LISP header is set as 1, a Map-Notify message is sent as a response to Map-Register message. Otherwise, no map-notify is sent. This is defied by RFC 6830.*). Each time MS receives a Map-Request message, it looks up the required EID within its database. If the EID-RLOC mapping is found, map server forwards this request to the corresponding xTR otherwise sends a Negative Map-Reply message to the querying xTR. It is worth to indicating that class *MapServer* is actually

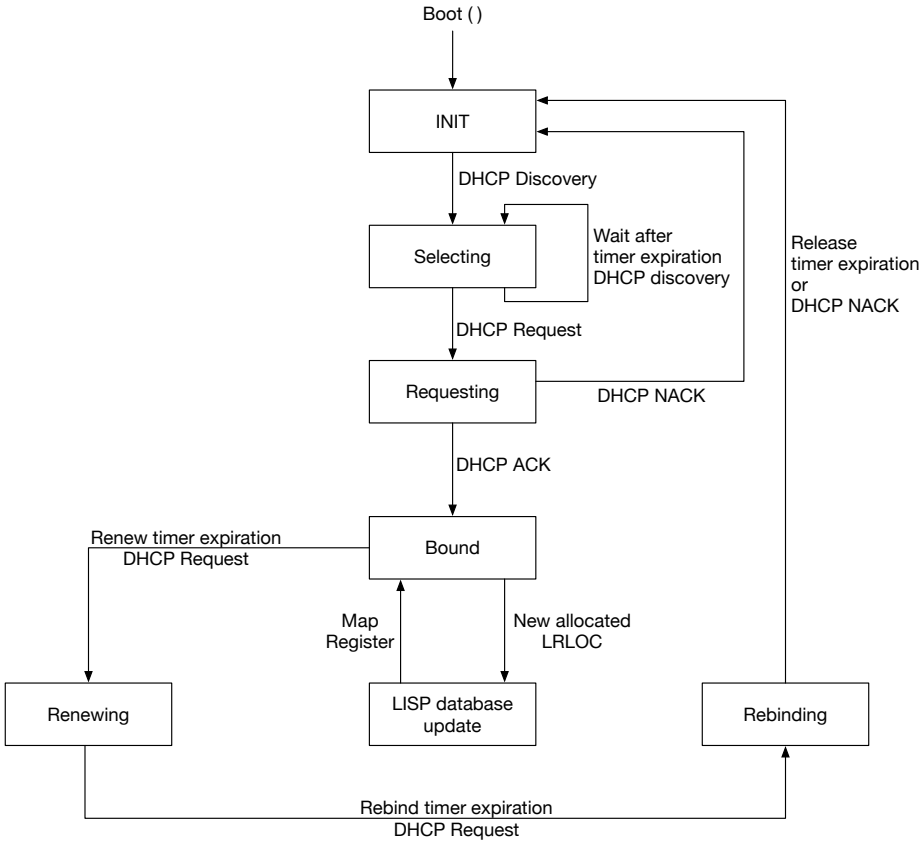

Fig. 7.4 LISP-compatible DHCP client state transition diagram

an abstract base class. The real functionalities are implemented by its subclasses, which in our implementation is *MapServerDdt*. This design allows the researchers to easily integrate and test their own implementations of MS. In current implementation, the role of MR is to receive the Map-Request message from xTR and forward it to the MS.

7.3.3 Modification of DHCP client to support LISP-MN

To support LISP mobility for IPv4, the intervention of DHCP is indispensable. From ns-3.27, DHCP client and server have been implemented in module *internet-apps*. However, the DHCP client of ns-3 is not compatible with LISP. Thus, conventional DHCP client is adapted to support LISP. The state transition diagram of DHCP client is illustrated in Fig. 7.4. When a LISP-MN node roams into the area covered by another xTR, the boot procedure of DHCP client is triggered once that the link state is changed. Afterwards, the DHCP client

sequentially pass 'INIT', 'Selecting', 'Requesting' states and enter into 'Bound' state after the reception of DHCP ACK message from DHCP server. In 'Bound' state, apart from saving the newly obtained IPv4 address (namely LRLOC) and default gateway, DHCP client should be able to check if the LRLOC is different from the one associated with its EID in its LISP database. If LRLOC is changed, DHCP client enters into state of 'LISP database update'. DHCP client is equipped with a dedicated socket of type *LispMappingSocket*. By this socket, DHCP client notifies the *LispEtrItrApplication* by sending a dedicated message that contains the EID-LRLOC mapping. *LispEtrItrApplication* is in charge of populating the received mapping entry into LISP database and sending a Map-Register message to the Map Server.

To be compatible with DHCP, conventional LISP-related process is also modified. For example, to transmit a DHCP Discovery message (application layer message), its source IP address is set as 0.0.0.0. The *Send()* method of *Ipv4L3Protocol* should be modified so that this message is not processed *LispOverIp*.

7.3.4 Integration of TUN net interface card

To support mobility, a LISP-speaking node actually can be regarded as a small LISP-Site. The xTR functionalities and DHCP service should be implemented on the LISP-speaking node. The address of MR and MS should be configured. As a LISP-MN, it has a static permanent EID and dynamic RLOC assigned by the DHCP server. To differentiate with conventional RLOC of xTR interface, such kind of RLOC is referred to as the local RLOC (LRLOC). There exist several possibilities about on which net interface cards (NICs) EID is configured: IP aliasing, loop back device and TUN device. IP aliasing consists of associating more than one IP address to a network interface. In case of LISP mobility, the supplementary IP address is EID. Loop back device can be also configured with EID to support LISP encapsulation. As a design choice, we use the solution based on TUN device, which is also applied by Lispmob [18]. In our implementation, different from a conventional LISP node, at least two NICs should be installed into the node. One is normal NIC such as *WifiNetDevice*. The DHCP client application runs on this kind of card and thus the LRLOC is allocated to this card. The other is a TUN type card. The TUN NIC is a virtual card which should actually invoke *Send()* of another real NIC. The permanent EID is assigned to TUN device.

Recall that after DHCP procedure, the node will be configured a default gateway provided by DHCP server. To make the packets from application layer always use EID as the source IP address of inner IP header, two static route entries with destination prefix length

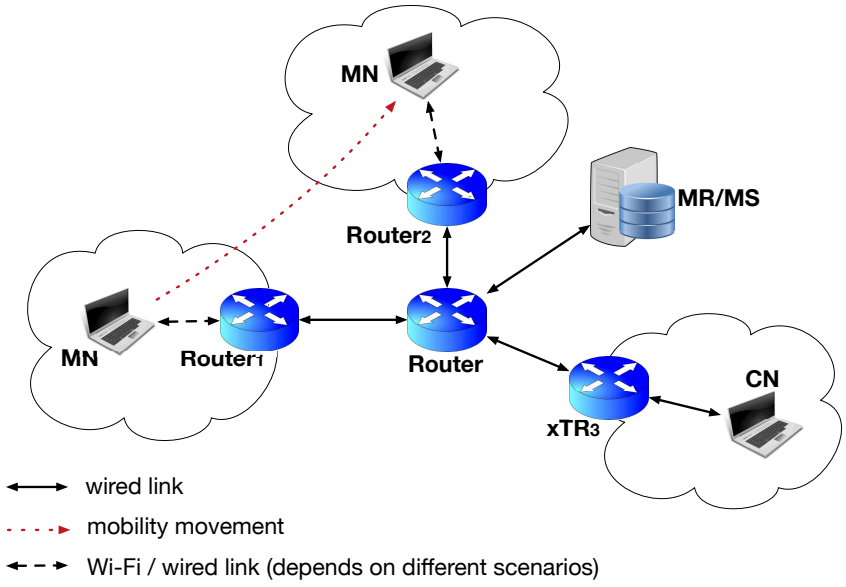

Fig. 7.5 General scenario for LISP mobility architecture

of 1 are added into static route of LISP-MN. Thus, to build the inner IP header, the EID is always used as its source address.

7.4 Theoretical analysis

IP mobility leveraging on LISP can be implemented either on the border router or the end host. To explore the characteristics of each scheme, we propose the following three different scenarios and analyze the overall handover delay and LISP Control Plane overhead.

1. LISP-MN in the non-LISP-Site (i.e., only the end host supports LISP).
2. MN in the LISP-Site (i.e., only the border router supports LISP).
3. LISP-MN in the LISP-Site (i.e., both the border router and the end host support LISP).

The network topology for all the scenarios is shown in Fig. 7.5. An MN is initially in the subnet of $Router_1$. It exchanges packets with a remote stationary node CN situated in the LISP-Site of xTR_3. The roles of $Router_1$ and $Router_2$ are slightly different in different scenarios. This will be respectively specified in the following parts (Sec. 7.4.1 to Sec. 7.4.3). To obtain the estimation of LISP mobility handover, we do not consider the delay due to wireless link switch and use an intermediate router to connect all three routers and the mapping system. The connection between MN and $Router_1$ can be either Wi-Fi or wired link. If Wi-Fi link is used, MN will move into the subnet of $Router_2$ at a constant speed. At a cer-

Table 7.1 Symbols for numerical analysis

Symbols	Explanations
$D_{overall}$	Overall handover delay related to LISP
D_{DHCP}	DHCP address configuration delay
$D_{Register}$	Delay of sending Map-Register
D_{Notify}	Delay of receiving Map-Notify
$D_{Request}$	Delay of sending Map-Request to MDS
D_{Reply}	Delay of receiving Map-Reply
$D_{Resolve}$	Delay of resolving mapping information in MDS
D_{SMR}	Delay of sending SMR
$D_{Request_{SMR}}$	Delay of sending a SMR-invoked Map-Request
D_{Link}	Link delay between two network entities
T_{A-B}	Delay of packet transmission between A and B
$T_{timeout_SMR}$	Timeout of SMR

tain moment, the Wi-Fi link between MN and $Router_1$ is down, which triggers the handover procedure. Afterwards, MN connects to $Router_2$ and re-establishes the communication with CN node. If they use wired link, the handover procedure can be simulated as follows. The MN have two interfaces respectively connected to $Router_1$ and $Router_2$. At a certain time, the connection to $Router_1$ set to be down and the one with $Router_2$ is set to be up. At the same time, the DHCP client on the interface connected to $Router_2$ is run and this triggers the handover procedure. By using wired link between MN and its routers is just to test the different mechanisms in an ideal situation, and it is not a real scenario.

In this chapter, the overall handover delay related to LISP $D_{overall}$ is defined as the time interval between the first and the last LISP packets during the handover procedure. The overall handover overhead $C_{overall}$ is defined as the number of LISP Control Plane messages exchanged during handover procedure. According to the three following scenarios, the handover delay and overhead consist of different parts. All the necessary delay and LISP overhead of LISP Control Plane during the mobility are listed in Tab. 7.1.

7.4.1 LISP-MN in non-LISP-Site

The first scenario is the LISP-MN in non-LISP-Site, where the border routers are the conventional routers and LISP is only implemented on the mobile end host MN. In our simulation, the LISP-MN with permanent EID is initially placed in the subnet of $Router_1$, with the IP address distributed by $Router_1$ as its RLOC. The remote CN is a conventional stationary end host, residing in a LISP-Site of xTR_3. LISP-MN first has a DHCP procedure when MN moves to $Router_2$, so that the later allocates a new IP address as RLOC. Then LISP-MN

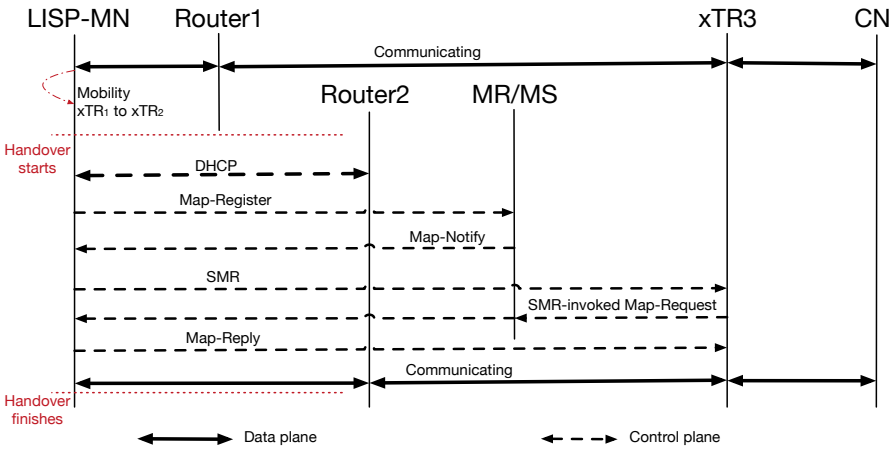

Fig. 7.6 Schema for LISP-MN mobility (SMR-invoked Map-Request is sent to the mapping system)

needs register its new mapping information to the mapping system, and also send a *SMR* to its communicating nodes in its cache (there is only CN in our scenario). xTR_3 sends an SMR-invoked Map-Request to the mapping system, so to obtain the new mapping information of LISP-MN. Afterwards, LISP-MN re-establishes the communication with CN node via $Router_2$. The detailed traffic schema related to the handover procedure is illustrated in Fig. 7.6.

The overall handover delay in this scenario is composed by two parts: the DHCP related delay and LISP related delay. The delays of DHCP procedures consist of LISP-MN sending DHCP Discovery message to $Router_2$, receiving DHCP Offer message, sending DHCP Request message, and receiving DHCP ACK message. The delays of all the LISP procedures are presented in Fig. 7.6. This chapter only focus on the delay caused by LISP. Thus, the overall handover delay $D_{overall}$ related to LISP of LISP-MN in non-LISP-Site is:

$$D_{overall} = D_{Register} + D_{Notify} + D_{SMR} + D_{Request_{SMR}} + D_{Reply} \tag{7.1}$$

To facilitate the comparison between each scenario, we provide the numerical results for the designed topology as shown in Fig. 7.5. Every link between two network entities in this architecture is set to $1ms$. According to the experimental results of Coras et al.'s paper [45], we set the resolving time in the mapping system as $200ms$. Thus, the $D_{overall}$ of the first

scenario in our designed architecture is:

$$\begin{aligned} D_{overall} &= 3T_{MN-MDS} + 2T_{MN-xTR_3} + T_{xTR_3-MDS} + D_{Resolve} \\ &= 3*(3*D_{Link}) + 2*(3*D_{Link}) + 2*D_{Link} + D_{Resolve} \\ &= 17D_{Link} + D_{Resolve} \\ &= 217ms \end{aligned}$$

The handover overhead is 6 messages of LISP Control Plane. It includes 2 signalings of registration when LISP-MN connects to $Router_2$ (Map-Register and Map-Notify), and 4 signalings related to SMR procedure: 1 SMR from LISP-MN to xTR_3, 1 SMR-invoked Map-Request from xTR_3 to the mapping system, 1 Map-Request forwarded by mapping system to LISP-MN, and 1 Map-Reply from LISP-MN to xTR_3.

There are two options when xTR_3 receives *SMR*. It can send the SMR-invoked Map-Request to the mapping system as we described before, or it can directly send the SMR-invoked Map-Request to the source locator address of *SMR* [48]. In this scenario, the source of *SMR* is LISP-MN. Thus, the overall handover delay $D_{overall}$ in this scenario is as follows:

$$D_{overall} = D_{Register} + D_{Notify} + D_{SMR} + D_{Request_{SMR}} + D_{Reply} \tag{7.2}$$

The numerical result is:

$$\begin{aligned} &= 2T_{MN-MDS} + 3T_{MN-xTR_3} \\ &= 2*(3*D_{Link}) + 3*(3*D_{Link}) \\ &= 15D_{Link} \\ &= 15ms \end{aligned}$$

Compared with solution of sending SMR-invoked Map-Request to the mapping system, the overall handover delay of sending the SMR-invoked Map-Request back to the source of *SMR* is smaller, since there is no resolving delay in the mapping system.

In this scenario, the handover overhead associated to LISP Control Plane is 5 messages. Since the SMR-invoked Map-Request is directly sent from xTR_3 to LISP-MN instead of passing the mapping system, it has 1 signaling less than the one sent to the mapping system.

The advantages of this scenario, i.e., LISP-MN in non-LISP-Site, are: 1) it is able to achieve handover through different subnets; 2) the numerical analysis indicates that the overall handover delay is small; 3) so to the overall overhead. The mobility does not cause lots of traffic in LISP Control Plane. However, since the routers are still the normal routers in this scenario, it cannot help to reduce the BGP routing table size, which is the initial purpose

to motivate LISP. Moreover, each LISP-MN needs a permanent IP address as its EID, which increases the burden of IPv4 address allocation. Each permanent EID and its LRLOC needs register to the mapping system, which also increases the size of LISP mapping table.

7.4.2 MN in LISP-Site

The second scenario is the MN in LISP-Site, where the mobile node MN is conventional and LISP are only implemented on the border routers. In our simulation, the MN is initially placed in the subnet of xTR_1, with the assigned IP address as its EID. The remote CN is same to the first scenario. The MN communicates with CN by encapsulating the packets on xTR_1 and decapsulating the packets on xTR_3, and the MN moves into the coverage of xTR_2 after the simulation begins. Since the communication should not be interrupted during the mobility, this scenario limits the movement of MN being only within the same subnet, i.e., one of the EID-prefixes of xTR_1 is same to one of xTR_2's. Similar to the first scenario, the DHCP procedure is necessary so to trigger the registration of new mapping information, but MN still keeps the former IP address as its EID, instead of xTR_2 distributing a new one to it. Then xTR_2 registers the new mapping information to the mapping system. As the mapping system finds out that the EID of MN has been registered and associated by xTR_1, it sends a Map-Notify to both xTRs. The reason for sending to xTR_2 is the acknowledgment of the reception of its Map-Register. Whereas the purpose to xTR_1 is to tell it that the MN is now mapping with xTR_2 and inform the remote CN to update its mapping information. Since MN used xTR_1 to communicate with CN in the past time, only xTR_1 stored in its cache that to whom MN was exchanging the packets instead of xTR_2. Thus, xTR_1 sends a *SMR* to xTR_3, and xTR_3 sends an SMR-invoked Map-Request to the mapping system, so to obtain the new mapping information of MN. Afterwards, MN re-establishes the communication with CN node via xTR_2. The detailed traffic schema related to the handover procedure is shown in Fig. 7.7.

The overall handover delay $D_{overall}$ in this scenario is almost same to the first one:

$$D_{overall} = D_{Register} + D_{Notify} + D_{SMR} + D_{Request_{SMR}} + D_{Reply} \tag{7.3}$$

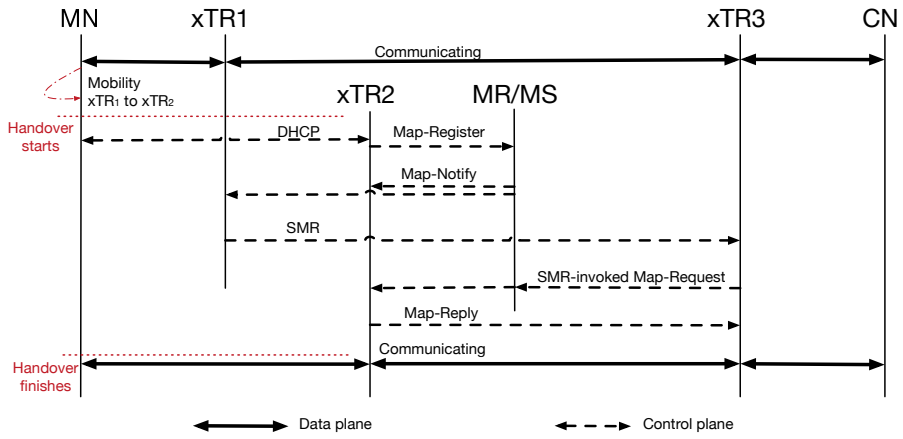

Fig. 7.7 Schema for LISP-MN mobility (SMR-invoked Map-Request is sent to the mapping system)

The numerical result is as follows:

$$\begin{aligned} D_{overall} &= 2T_{xTR_2-MDS} + T_{MDS-xTR_1} + T_{xTR_1-xTR_3} + T_{xTR_3-MDS} + \\ &\quad D_{Resolve} + T_{xTR_2-xTR_3} \\ &= 2*(2*D_{Link}) + 2*D_{Link} + 2*D_{Link} + 2*D_{Link} + D_{Resolve} + 2*D_{Link} \\ &= 12D_{Link} + D_{Resolve} \\ &= 212ms \end{aligned}$$

The handover overhead in this scenario is 7. Besides 4 messages used for SMR procedure, it needs 3 signalings to complete the registration. There are 1 Map-Register from xTR_2 to the mapping system and 2 messages of Map-Notify: 1 to the xTR_2 and 1 to the xTR_1.

Same to the first scenario having two options, when xTR_3 receives the *SMR* from xTR_1, it can also directly send SMR-invoked Map-Request back to xTR_1, which implies that the xTR_1 puts the new mappings into its database. The overall handover delay $D_{overall}$ is as follows:

$$D_{overall} = D_{Register} + D_{Notify} + D_{SMR} + D_{Request_{SMR}} + D_{Reply} \tag{7.4}$$

The numerical result is:

$$\begin{aligned} &= T_{xTR_2-MDS} + T_{MDS-xTR_1} + T_{xTR_1-xTR_3} + 2T_{xTR_2-xTR_3} \\ &= 2*D_{Link} + 2*D_{Link} + 2*D_{Link} + 2*(2*D_{Link}) \\ &= 10D_{Link} \\ &= 10ms \end{aligned}$$

Since the SMR-invoked Map-Request is not sent to the mapping system, there is no resolving delay. However, in this scenario, as xTR_1 is no longer in charge of MN, how long it stores the CNs for MN in its cache is an important point to discuss. If the expired time is set too long, it wastes the source of xTR_1 and is not necessary. Whereas if the time is too short, there is the risk that remote xTRs of CNs like xTR_3 in our scenario, do not have enough time to request the new mapping information.

Same to the first scenario that directly responding to the source locator of SMR has 1 signaling less than the one requesting to the mapping system first. Thus, the overall handover overhead in this case is 6.

Since the routers support LISP, i.e., are the xTRs in this scenario, it can help to reduce the BGP routing table size, which is the initial motivation of proposing LISP. Besides, the analysis hints that the overall handover delay of this scenario is the shortest. However, it can only provide the mobility within a same subnet, which means that it is not able to offer the handover through different subnets. Thus, this scenario is more suitable to be applied for the mobility of virtual machines in the Data Center.

7.4.3 LISP-MN in LISP-Site

The third scenario is the LISP-MN in LISP-Site, where both the border routers and the mobile node MN are implemented LISP. In our simulation, the LISP-MN with permanent EID is initially placed in the subnet of xTR_1, with the IP address allocated by xTR_1 as its LRLOC. The remote CN is still same to the first two scenarios, which is a conventional stationary end host, residing in a LISP-Site of xTR_3. The LISP-MN communicates with CN by double encapsulation. The first encapsulation is on itself and the second time is on the xTR_1. When the LISP packets arrive at xTR_3, it needs decapsulate them twice. LISP-MN first has a DHCP procedure with xTR_2, so that the later assigns it a new IP address as its LRLOC. Then LISP-MN needs register its new mapping information to the mapping system, and also send a *SMR* to all the xTRs of CNs in its cache (actually is to the xTR of CN, i.e., xTR_3 in our scenario). Once xTR_3 receives a *SMR*, it sends an SMR-invoked Map-Request to the mapping system, so to obtain the new mapping information of LISP-MN. Actually,

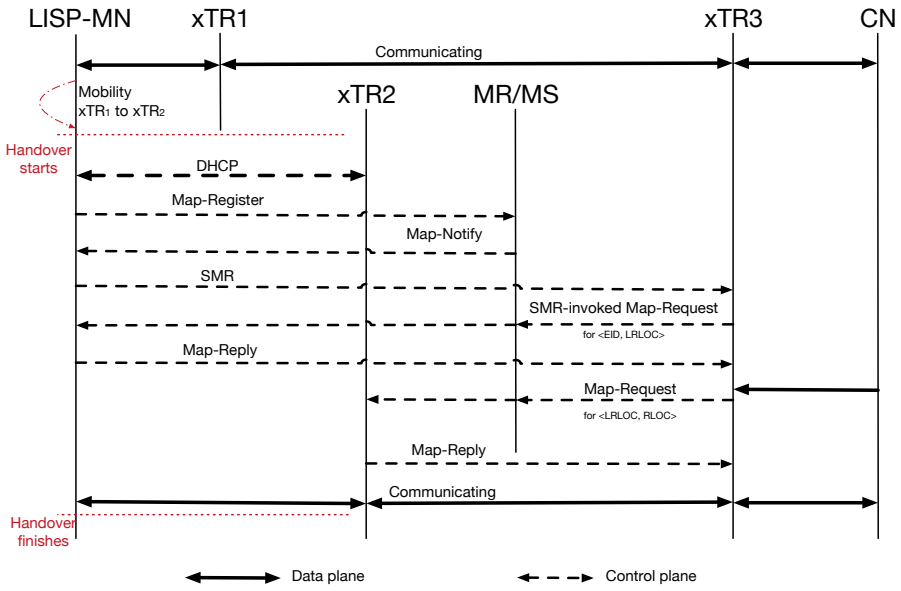

Fig. 7.8 Schema for LISP-MN in LISP-Site mobility (SMR-invoked Map-Request is sent to the mapping system)

this mapping information that xTR_3 obtains is the $< EID_MN, LRLOC >$ for LISP-MN. It is not able to send the packets to LISP-MN at the moment, since it does not know how to route the packets to the *LRLOC* of LISP-MN, i.e., it lacks the mapping information for *LRLOC*. Only until xTR_3 receives the packets from CN to LISP-MN, which triggers the Map-Request procedure to the mapping system, can xTR_3 know the mapping information of LRLOC $< LRLOC, RLOC_{xTR_2} >$. Now xTR_3 gets the double mapping information for LISP-MN. Afterwards, LISP-MN re-establishes the communication with CN node by passing xTR_2. The detailed traffic schema related to the handover procedure is illustrated in Fig. 7.8.

Since this scenario is double encapsulation that xTR_3 needs to know both inner and outer mapping information of LISP-MN for sending the packets. The overall handover delay $D_{overall}$ in this scenario is larger than the first two scenarios. The $D_{overall}$ is:

$$\begin{aligned} D_{overall} &= D_{Register} + D_{Notify} + D_{SMR} + D_{Request_{SMR}} + D_{Reply} + \\ &\quad D_{Request} + D_{Reply} \end{aligned} \tag{7.5}$$

The numerical result is as follows:

$$\begin{aligned}
&= 3T_{MN-MDS} + 2T_{MN-xTR_3} + 2T_{xTR_3-MDS} + 2D_{Resolve} + \\
&\quad T_{MDS-xTR_2} + T_{xTR_2-xTR_3} \\
&= 3*(3*D_{Link}) + 2*(3*D_{Link}) + 2*(2*D_{Link}) + 2D_{Resolve} + \\
&\quad 2*D_{Link} + 2*D_{Link} \\
&= 23D_{Link} + 2D_{Resolve} \\
&= 423ms
\end{aligned}$$

The double encapsulation causes not only longer handover delay but also more handover overhead related to LISP Control Plane. Two messages are needed for the registration, 3 signalings are used to obtain the *LRLOC* of *EID*. Besides, two more Map-Requests (one from xTR_3 to the mapping system, one from the mapping system to xTR_2) and one more Map-Reply are required to get the *RLOC* of *LRLOC*. Thus, the handover overhead is 9, which has 3 messages more than the first scenario.

Differently from the first two scenarios, when xTR_3 directly sends the SMR-invoked Map-Request back to the source of *SMR* has smaller overall handover delay, this solution for the third scenario has bigger delay instead. It is caused by the double encapsulation in this scenario while the first two scenarios have only single encapsulation. When xTR_3 receives the *SMR* from LISP-MN, it wants to send SMR-Invoked Map-Request to the LISP-MN for the mapping information of $< EID_MN, LRLOC >$, but it does not know how to reach to LISP-MN, i.e., it lacks the mapping information of $< LRLOC, RLOC_{xTR_2} >$. Thus, it discards the *SMR* and sends the Map-Request to the mapping system first. Then, the mapping information of $< LRLOC, RLOC_{xTR_2} >$ is stored in its cache. It waits for the next *SMR* so to send the SMR-invoked Map-Request to the LISP-MN. The traffic schema is shown in Fig. 7.9.

The overall handover delay $D_{overall}$ by sending the SMR-invoked Map-Request to the source of *SMR* in this scenario is as follows, where $T_{timeout\_SMR}$ is the interval to re-send the *SMR* in case of nothing received. We set it to $1s$ in the simulation:

$$\begin{aligned}
D_{overall} &= D_{Register} + D_{Notify} + D_{SMR} + T_{timeout_SMR} + D_{SMR} + \\
&\quad D_{Request_{SMR}} + D_{Reply}
\end{aligned} \tag{7.6}$$

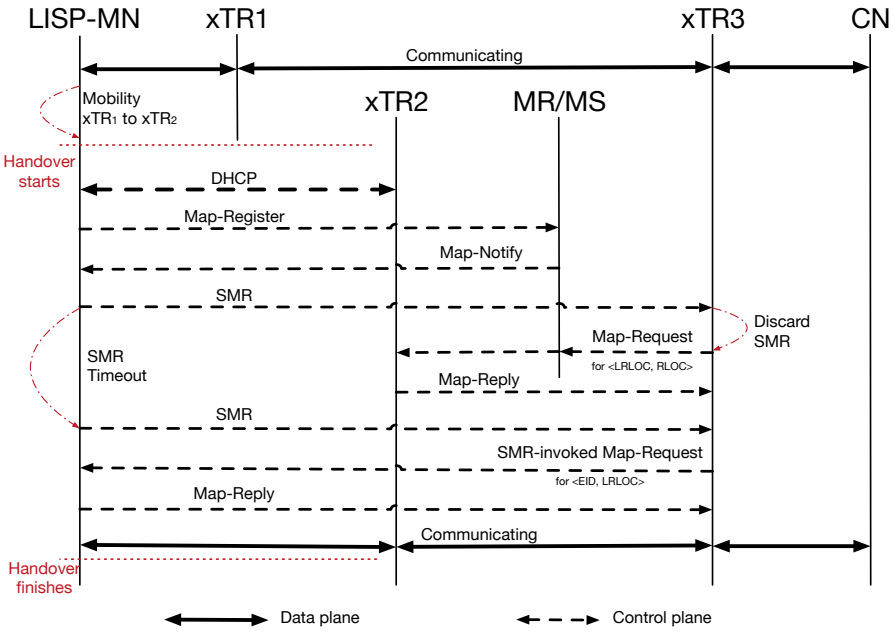

Fig. 7.9 Schema for LISP-MN in LISP-Site mobility (SMR-invoked Map-Request is sent to the source of SMR)

The numerical result is as follows:

$$
\begin{aligned}
&= 2T_{MN-MDS} + 4T_{MN-xTR_3} + T_{timeout_SMR} \\
&= 2*(3*D_{Link}) + 4*(3*D_{Link}) + T_{timeout\_SMR} \\
&= 18D_{Link} + T_{timeout\_SMR} \\
&= 1018ms
\end{aligned}
$$

Although when the SMR-invoked Map-Request directly sent to the source of SMR has the different traffic schema from the one sent to the mapping system, the overall handover overhead is still 9. The difference between them is only the order of getting mapping information on xTR_3, where this case is to obtain the outer mapping first but the last case is to get the inner mapping first.

Since both the MN and the border routers support LISP, the advantages of this scenario, i.e., LISP-MN in the LISP-Site, are: 1) it can help to reduce the BGP routing table size; 2) it is able to achieve handover through different subnets. However, same to the shortcomings for the first scenario, each LISP-MN needs a permanent IP address as its EID, which increases the burden of IPv4 address allocation. Each permanent EID and its LRLOC needs register to the mapping system, which also increases the size of LISP mapping table. Be-

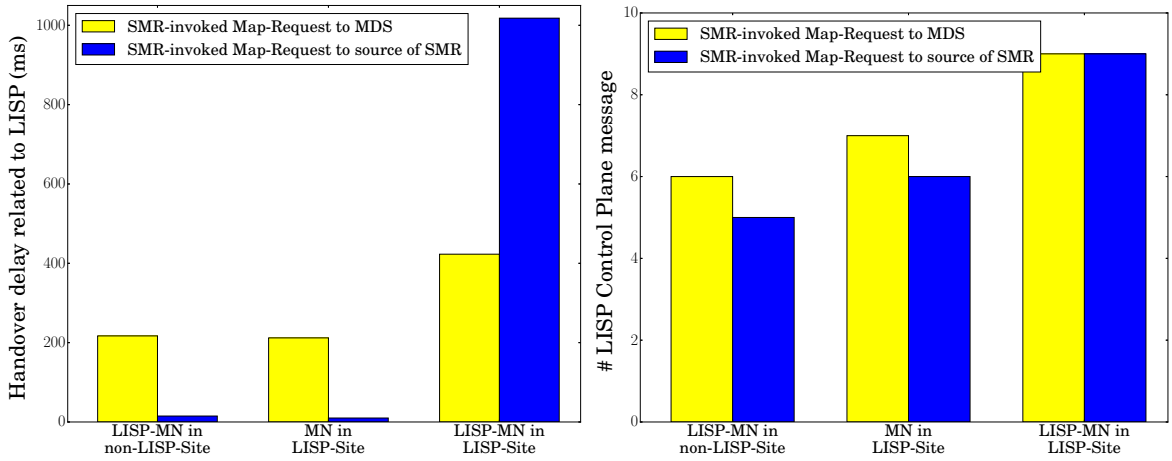

Fig. 7.10 The handover delay related to LISP (left) and the handover overhead (right) grouped by three LISP mobility scenarios.

sides, the numerical analysis indicates that the overall handover delay is much longer than the other two scenarios due to its double encapsulation.

All the analyzed results about the overall handover delay related to LISP and the handover overhead are presented in Fig. 7.10.

7.5 Summary

There exist some LISP simulation implementations, but they are proprietary or they do not support the extension of LISP mobility. Further, although measurements on LISP-testbeds can provide real time performance, due to the complicated topological structure, it is somewhat like a black box test, which hinders us to find the exact explanation for some results. This highlights the importance to have an open source simulator for LISP in particular to support LISP mobility functionality. In this chapter, based on an implementation of basic LISP on ns-3.24, we adapt it to ns-3.27 first (the latest version at the moment of writing). To facilitate the researchers to deeply track the exchange of LISP packets, we encode in LISP Data Plane packets so that the Wireshark can resolve them. Finally we implement the LISP mobility extensions on it. As there are three methods to support mobility in LISP: host-based (i.e. LISP-MN), network-based (i.e., xTR), both host-based and network-based (i.e., LISP-MN behinds xTR) mobility. We analyze the overall handover delay and the overhead of LISP Control Plane among them, compare the performance among them by listing their advantages and shortcomings.

Chapter 8

Conclusion and Future Work

In this chapter, we summarize our major contributions and discuss future research directions.

8.1 Major Contributions

The objective of this dissertation is to thoroughly evaluate *Locator/Identifier Separation Protocol (LISP)* from different aspects: from measuring its performance by conducting experiments in the large-scale real networks, numerically analyzing the bounds, up to implementing LISP mobility in ns-3. The measurement works are related to the evaluation of mapping system and the assessment of interworking performance with the legacy Internet. Due to the new findings in the mapping system, we propose a more comprehensive LISP monitor tool to address the additional requirements. The theoretical analysis is about the LISP mobility. More precisely, we analyze the low bounds of handover for the different network components supporting LISP. Besides, we implement LISP mobility extensions on ns-3.27 by leveraging an existing implementation. Concretely, the main contributions can be summarized as follows:

8.1.1 Evaluation of LISP Mapping System

We continuously measured the LISP Beta Network for seventeen days to assess the stability and consistency of the mapping system. The experiments are conducted from July 2nd to 18th 2013, from 5 Vantage Points (VPs), sending *Map-Request* messages to all the 13 Map-Resolvers (MRs) of the LISP Beta Network, for all selected 613 destination IP addresses every 30 minutes. The VPs were composed by part of academic networks, commercial Internet, PlanetLab and spread across Europe and USA. The destination IP addresses consisted of the first address within each prefix derived from the LISPmon project [19] re-

gardless of the type (i.e., the EID-prefixes or regular prefixes). The results of measurements show that the mapping system is stable during 91.49% of time, is consistent respectively by MR in 86.3% and by VP in 90.48% of the cases. As the instabilities and inconsistencies are observed and for studying them in details we developed a new taxonomy. It shows that Map-Reply only changes once in the most instability cases, and the inconsistency is mainly caused by the different types of received Map-Replies (the mix of Negative Map-Reply and LISP Map-Reply). All in all, instabilities and inconsistencies are rare events. At last, the utilization of multi-homing and dual-stack (i.e., IPv4 and IPv6) are observed during the whole experiment. This work is published in WNM [82].

8.1.2 LISP-Views: LISP Mapping System Monitor

The evaluation of the mapping system shows that there exist instability and inconsistency, which are hard to be found by the only current LISP monitor LISPmon. A dynamic and complete monitoring system is required to accompany the growth of LISP. We propose LISP-Views, a versatile large scale LISP monitoring architecture. LISP-Views allows to automatically conduct comprehensive and objective measurements. Users can deploy it on whichever VP and define to monitor which MRs. They are also able to define the type of measurements. All these operations can be conducted via the provided API. At the moment of writing, LISP-Views is deployed on one VP to supervise all the existing MRs on both LISP-Beta Network and LISP-Lab Platform. After running LISP-Views in the wild for several months and comparing the monitoring results with LISPmon, we confirm that LISP-Views provides more detailed and accurate information. We observe that every MR has different behaviours compared to others, such as in the reliability and resolving time. Besides, the measurements helped the OpenLISP coders fix a previously unknown bug in the MR of LISP-Lab platform. The preliminary graphic results, such as the number of LISP Map-Reply, the number of RLOCs contained in each LISP Map-Reply, and the responsiveness of each MR are shown on the website of LISP-Views [16]. This work is published in ITC [78].

8.1.3 Assessing LISP interworking performance through RIPE Atlas

We conducted two experiments to evaluate LISP interworking performance with legacy Internet. It is the first thorough insight on the performance of LISP Proxy Ingress Egress Tunnel Router (PxTR). As a setup phase, we selected 4 probes ping to the top 50 Alexa sites every 10 minutes during 6 hours on November 6th 2015. In the second experiment, we chose 5 probes ping and traceroute to top 500 Alexa IPv4 and 122 IPv6 addresses re-

spectively every 30 minutes and 60 minutes. Both experiments show that the PxTR indeed introduces negative effects for nearby destinations, but the negative impact of PxTR can be ignored for the intercontinental long-distance transmission. The results also show that the position of PxTR is very important. The PxTR positioned either near to the sources or the destinations can decrease the latency a lot. LISP is generally stable, except for the IPv6 performance of LISP Beta Network. Further, the performance of LISP-Lab PxTR is more reliable than the one of LISP Beta Network, although at the time of this writing the latter has 6 worldwide PxTRs used for IPv4 and 2 located in US for IPv6, whereas LISP-Lab has only 1 PxTR for both IPv4 and IPv6. Natively forwarding without using LISP decreases the latency, but not much. The traceroute experiment of LISP-Lab shows that introducing PxTR of course brings more hops, but if the PxTR is well configured, so to always have peers to the destinations, there are only 4 more hops compared to the packets being natively forward by xTR without encapsulating with LISP. This work is published in INFOCOM Student workshop [80], ACM SIGCOMM Student workshop [79], and Elsevier Computer Networks [81].

8.1.4 Analysis of LISP mobility and ns-3 Implementation

There are three methods to support LISP mobility: host-based (i.e. LISP Mobile Node (LISP-MN)), network-based (i.e., Ingress Egress Tunnel Router (xTR)), both host-based and network-based (i.e., LISP-MN behinds xTR) mobility. In the first scenario, there is only MN supporting LISP (LISP-MN in the non-LISP-Site); in the second scenario, there is only border router supporting LISP (MN in the LISP-Site); and in the third scenario, both MN and border routers supporting LISP (LISP-MN in the LISP-Site). We analyze the overall handover delay and the overhead of the LISP control plane among them. The second scenario has the smallest handover delay, but the MN can only roam within the same subnet. The first and third scenarios support the MN seamless roaming through the different subnets, but the first scenario does not help in reducing the BGP routing table, and the third scenario introduces a much longer handover delay. Based on an implementation of basic LISP on ns-3.24, we adapt it to ns-3.27 first (the latest version at the moment of writing). To facilitate the researchers to deeply track the exchange of LISP packets, we encode every bit in LISP Data Plane packets so that the Wireshark can resolve them. Finally we implement the LISP mobility extensions on it.

8.2 Discussion & Future work

8.2.1 Evaluation of LISP Mapping System

There are two aspects to be done in the future. One is to conduct a new experiment to see if something in the mapping system has changed. The other one is using LISP-DDT Referral Internet Groper (RIG) to have a deeper look at the LISP Delegated Database Tree (LISP-DDT).

The experiment was conducted in 2013, while there is a growth of LISP in the following years. The architecture of the mapping system has also changed, in which there were 13 MRs during the measurement but only 7 MRs in 2017. Besides, the LISP Beta Network is the only LISP testbed at that time. However, the LISP-Lab platform is open to the experimenters in 2015. The later consists of 3 MRs and interconnects with the mapping system of LISP Beta Network. Thus, it is very necessary to conduct a new experiment in the future to evaluate the latest *Stability* and *Consistency* performance of LISP mapping system. The results can be compared to the results in Chapter. 4 so to get the LISP developing trend.

The mapping information should be identical on all the MRs at the same time, but the experimental results show that there exist the inconsistencies between the MRs and VPs. The reason is difficult to explore by such kind of experiment, since the procedures of forwarding Map-Requests within the mapping system is transparent to the experimenters. Based on the theory, it is probably caused by the cache of each MR storing the mapping information at the different time. Thus, leveraging on RIG to troubleshoot the inconsistent issues within the mapping system is useful. If the assumption is correct, to propose a synchronized mechanism should be important to the mapping system.

8.2.2 LISP-Views: LISP Mapping System Monitor

LISP-Views is still in the first phase, it is deployed on only one VP and executed the measurements by using the command lines. Thus, it is expected to deploy the LISP-Views on multiple VPs to reproduce the consistency of the VPs as described in Chapter. 4. Besides, the command lines are not efficient to conduct the experiments, to implement a REST API on LISP-Views for setting the experiment time, the monitored MRs, and obtain the different aspects of LISP status should be done in the future. At the moment of writing this dissertation, all the monitored IP addresses via LISP-Views are IPv4, it should be interesting to test IPv6 behaviors.

8.2.3 Assessing LISP interworking performance through RIPE Atlas

In our experiment, as LIP6 probe does not receive any *ping* responses for 42 IPv4 and 10 IPv6 destinations, to cluster these destinations by geography and explore the reasons can be possible future works. As the IPv6 packets sent from the LISP-Lab probe are natively forwarded instead of encapsulating into LISP packets during our experiment, the configurations should be modified so that PETR of LISP-Lab allows to receive the IPv6 traffic from LISP-Lab probe. Then the LISP IPv6 performance can be evaluated. Besides, the path between the LIP6 probe and its PETR is BGP routing and is not configured with tunnel. Thus, it will be interesting to compare the performance with and without tunnel. Furthermore, as the degradation of performance observed in Chapter. 6 is mainly due to the PITR selection by the destinations. It will be helpful to explore the interaction between BGP anycast and LISP interworking. If the destinations leverage on the *anycast* to select the nearest PITR, we suppose that the additional delay introduced by LISP will be decreased a lot.

8.2.4 Analysis of LISP mobility and ns-3 Implementation

There are some directions for the further work. For the next step, we plan to validate the numerical analysis on the simulator. As *Map-Versioning* [68] is another Mapping Cache update mechanism, we can also compare the performance between it and *SMR* that we present in this Chapter. 7 by our implemented simulator.

The second scenario has the shortest handover delay, but it does not support the MN roaming through the different subnets. Thus, to use MAC address as the EID can be an option so that the MN is able to change the subnet without the interruption of the communication. Further, since the third scenario uses double encapsulation, which causes a big increase in the delay, if will be interesting if the mapping system can differently processes SMR-invoked Map-Request and the normal Map-Request. When mapping system receives an SMR-invoked Map-Request, if it finds that the *RLOC* of the queried EID is actually a *LRLOC*, e.g., the *LRLOC* of LISP-MN in our third scenario, it then forwards the Map-Request not only to the LISP-MN, but also to the xTR_2, so that LISP-MN replies to the xTR_3 with the mapping information of $<EID_{LISP-MN}, LRLOC_{LISP-MN}>$, and xTR_2 replies to the xTR_3 with the mapping information of $<LRLOC_{LISP-MN}, RLOC_{xTR_2}>$. In this way, xTR_3 only needs query to the mapping system once, so the handover delay due to the LISP related procedure can reduce a lot. The future work for the implementation of LISP mobility extensions is intended to support IPv6.

References

[1] Alexa. http://www.alexa.com/.

[2] Alexa top sites ranking. https://www.alexa.com/topsites.

[3] As3943 ipv4 route propagation. http://bgp.he.net/AS3943#_graph4.

[4] As3943 ipv6 route propagation. http://bgp.he.net/AS3943#_graph6.

[5] AVM's FRITZ!Box Configuration and Operation. See https://en.avm.de/fileadmin/user_upload/EN/Manuals/FRITZ_Box/Manual_FRITZ_Box_3490-en.pdf.

[6] BGP Routing Table Analysis Reports. See https://bgp.potaroo.net/.

[7] Cisco Locator/ID Separation Protocol (LISP) Q&A. See https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/locator-id-separation-protocol-lisp/qa_c67-582925.pdf.

[8] FreeBSD (website). https://www.freebsd.org/.

[9] INET Framework (Homepage). See https://inet.omnetpp.org/.

[10] Internet live stats. Accessed on Jan. 7, 2018. [Online]. Available: http://www.internetlivestats.com/internet-users/.

[11] LIP6-LISP (github). https://github.com/lip6-lisp.

[12] LISP Beta Network. See http://www.lisp4.net.

[13] LISP Beta Network Latest Architecture. See https://www.lisp4.net//images/lisp-ddt.pdf.

[14] LISP for SDN and NFV (OpenDaylight Submmit). See https://events.linuxfoundation.org/sites/events/files/slides/LISP_ODLSummit_2014.pdf.

[15] LISP-Lab. See http://www.lisp-lab.org/.

[16] LISP-Views (website). See http://hunter.lisp-views.org/.

[17] Lispers.net. See https://www.lispers.net/.

[18] LISPmob (website). https://www.openoverlayrouter.org/lispmob/.

[19] LISPmon. See http://www.lispmon.net.

[20] ns-3. See http://https://www.nsnam.org/.

[21] OMNet++ (Homepage). See https://www.omnetpp.org/.

[22] Open Overlay Router (website), the successor of LISPmob. See https://www.openoverlayrouter.org/.

[23] OpenDaylight (website). See https://www.opendaylight.org/.

[24] OpenLISP (website). http://www.openlisp.org/.

[25] Planetlab (website). https://www.planet-lab.org/.

[26] pylisp (Github). See https://github.com/steffann/pylisp.

[27] pylisp (Reference). See https://www.biostat.wisc.edu/~annis/creations/PyLisp/pylisp.html.

[28] RIPE Atlas. https://atlas.ripe.net/.

[29] Smartinsights. Accessed on Jan. 7, 2018. [Online]. Available: http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/.

[30] The official website of Wireshark. See https://www.wireshark.org/.

[31] J. Adan. Tunneled inter-domain routing (tidr). *Work in Progress*, 2006.

[32] L. Agbodjan. Towards a lisp simulator. Master's thesis, Université de Liège, 2016. Unpublished master's thesis. The source code of Basic LISP implementation under ns-3.24 is on the bitbucket: https://bitbucket.org/Lionel_Agbodjan/tfe__towards_a_lisp_simulator.

[33] M. Aiash. A novel security protocol for resolving addresses in the location/id split architecture. In *International Conference on Network and System Security*, pages 68–79. Springer, 2013.

[34] M. Aiash, A. Al-Nemrat, and D. Preston. Securing address registration in location/id split protocol using id-based cryptography. In *International Conference on Wired/Wireless Internet Communication*, pages 129–139. Springer, 2013.

[35] R. Atkinson, S. Bhatti, and S. Hailes. Evolving the internet architecture through naming. *IEEE Journal on Selected Areas in Communications*, 28(8):1319–1325, 2010.

[36] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. A layered naming architecture for the internet. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 343–352. ACM, 2004.

[37] S. Barkai, D. Farinacci, D. Meyer, F. Maino, V. Ermagan, A. Rodriguez-Natal, and A. Cabellos-Aparicio. Lisp based flowmapping for scaling nfv. Technical report, Internet-Draft draft-barkai-lisp-nfv-10, Internet Engineering Task Force. Work in Progress, 2017.

[38] S. Bhatti and R. Atkinson. Identifier-locator network protocol (ilnp) architectural description. 2012.

[39] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, et al. Advances in network simulation. *Computer*, 33(5):59–67, 2000.

[40] S. Brim, N. Chiappa, D. Farinacci, V. Fuller, and D. Lewis. Lisp-cons: A content distribution overlay network service for lisp. draft-meyer-lisp-cons-04, 2008.

[41] A. Cabellos and F. Coras. Internet engineering task force (ietf) d. saucez request for comments: 7834 inria category: Informational l. iannone. 2016.

[42] A. Cabellos, A. R. Natal, L. Jakab, V. Ermagan, P. Natarajan, and F. Maino. Lispmob: mobile networking through lisp. *LISPmob white paper*, 2011.

[43] X. Chang. Network simulations with opnet. In *Simulation Conference Proceedings, 1999 Winter*, volume 1, pages 307–314. IEEE, 1999.

[44] F. Coras, A. Cabellos-Aparicio, and J. Domingo-Pascual. An analytical model for the LISP cache size. In *Proc. IFIP Networking*, May 2012.

[45] F. Coras, D. Saucez, L. Iannone, and B. Donnet. On the performance of the LISP beta network. In *Proc. IFIP Networking*, June 2014.

[46] S. Deering and R. Hinden. Rfc 2460: Internet protocol, 1998.

[47] V. Ermagan, D. Farinacci, C. White, J. Skriver, D. Lewis, and F. Maino. Nat traversal for lisp. 2016.

[48] D. Farinacci and et al. The locator/ID separation protocol (LISP). RFC 6830, Internet Engineering Task Force, January 2013.

[49] D. Farinacci, A. Jain, I. Kouvelas, and D. Lewis. Locator/id separation protocol delegated database tree (lisp-ddt) referral internet groper (rig). Technical report, 2017.

[50] D. Farinacci, D. Lewis, D. Meyer, and C. White. Lisp mobile node, draft-ietf-lisp-mn-01. Technical report, institution=IETF Internet draft, Apr. 2017.[Online]. Available: https://tools.ietf.org/html/draft-ietf-lisp-mn-01, 2017.

[51] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. Rfc 2784-generic routing encapsulation (gre). *IETF, March 2000*, 2000.

[52] D. Farinacci, T. Li, S. Hanks, D. Meyer, P. Traina, and R. Glenn. Generic routing encapsulation. *IETF (The Internet Engineering Task Force) Request for Comments RFC*, 2784, 2000.

[53] D. Farinacci and D. Meyer. The locator/ID separation protocol Internet groper (LIG). RFC 6835, Internet Engineering Task Force, 2013.

[54] B. Feng, H. Zhang, H. Zhou, and S. Yu. Locator/identifier split networking: A promising future internet architecture. *IEEE Communications Surveys & Tutorials*, 2017.

[55] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. Tcp extensions for multipath operation with multiple addresses. Technical report, 2013.

[56] V. Fuller. LISP-DDT: A new mapping database for LISP, June 2012. Nanog 55 Talk.

[57] V. Fuller and et al. Locator/ID separation protocol (LISP) map-server interface. RFC 6833, Internet Engineering Task Force, January 2013.

[58] V. Fuller, D. Farinacci, D. Meyer, and D. Lewis. Locator/ID separation protocol alternative logical topology (LISP+ALT). RFC 6836, Internet Engineering Task Force, January 2013.

[59] V. Fuller, D. Lewis, and A. Ermagan, V. Jain. LISP Delegated Database Tree. Internet Draft, Work in Progress draft-ietf-lisp-ddt-01.txt, Internet Engineering Task Force, March 2013.

[60] V. Fuller, D. Lewis, V. Ermagan, and A. Jain. Lisp delegated database tree. *draft-ietf-lisp-ddt-09, Internet Engineering Task Force*, 2017.

[61] A. Galvani, A. Rodriguez-Natal, A. Cabellos-Aparicio, and F. Risso. Lisp-roam: network-based host mobility with lisp. In *Proceedings of the 9th ACM workshop on Mobility in the evolving internet architecture*, pages 19–24. ACM, 2014.

[62] A. García-Martínez, M. Bagnulo, and I. Van Beijnum. The shim6 architecture for ipv6 multihoming. *IEEE Communications Magazine*, 48(9), 2010.

[63] Y. Han, S. Ryu, Y.-J. Suh, and J. W.-K. Hong. Design and implementation of lisp controller in onos. In *NetSoft Conference and Workshops (NetSoft), 2016 IEEE*, pages 417–422. IEEE, 2016.

[64] S. Hanks, D. Meyer, D. Farinacci, and P. Traina. Generic routing encapsulation (gre). 2000.

[65] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena. Network simulations with the ns-3 simulator. *SIGCOMM demonstration*, 14(14):527, 2008.

[66] M. Hoefling, M. Menth, and M. Hartmann. A survey on mapping systems for locator/identifier split internet routing. *IEEE Communications Surveys and Tutorials*, 15(4):1842–1858, November 2013.

[67] L. Iannone and O. Bonaventure. On the cost of caching Locator/ID mappings. In *Proc. ACM SIGCOMM CoNEXT*, December 2007.

[68] L. Iannone, D. Saucez, and O. Bonaventure. Locator/id separation protocol (lisp) map-versioning. Technical report, 2013.

[69] M. Isah, S. Simpson, Y. Sani, and C. Edwards. Towards zero packet loss with lisp mobile node. In *Computing, Networking and Communications (ICNC), 2017 International Conference on*, pages 265–271. IEEE, 2017.

[70] 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*, 28(8):1332 – 1343, October 2010.

[71] S. Jeon, J. Jang, and Y.-H. Kim. Network mobility support in the proxy mobile ipv6 domain. *Network*, 2010.

[72] D. Johnson, C. Perkins, and J. Arkko. RFC 3775. *Mobility support in IPv6*, 2004.

[73] J. Kim, L. Iannone, and A. Feldmann. A deep dive into the LISP cache. In *Proc. IFIP Networking*, May 2011.

[74] J. Kim, L. Iannone, and A. Feldmann. Caching locator/id mappings: An experimental scalability analysis and its implications. *Computer Networks*, 57(4):897–909, 2013.

[75] D. Klein, M. Hartmann, M. Hoefling, and M. Menth. Integration of lisp and lisp-mn into inet. In *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, pages 299–306. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2012.

[76] E. Lear. Nerd: A not-so-novel endpoint id (eid) to routing locator (rloc) database. 2013.

[77] D. Lewis, D. Meyer, D. Farinacci, and V. Fuller. Interworking between locator/id separation protocol (LISP) and non-LISP sites. RFC 6832, Internet Engineering Task Force, January 2013.

[78] Y. Li, A. Abouseif, L. Iannone, and D. Saucez. Lisp-views: Monitoring lisp at large scale. In *Teletraffic Congress (ITC 29), 2017 29th International*, volume 1, pages 178–186. IEEE, 2017.

[79] Y. Li and et al. Performance evaluation of locator/identifier separation protocol through ripe atlas. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference*, pages 561–562. ACM, 2016.

[80] Y. Li and L. Iannone. Using ripe atlas to evaluate the locator/id separation protocol. In *Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on INFOCOM 2016 Conference*, pages 696–697. IEEE, 2016.

[81] Y. Li and L. Iannone. Assessing locator/identifier separation protocol interworking performance through RIPE Atlas. *Computer Networks*, pages –, 2017.

[82] Y. Li, D. Saucez, L. Iannone, and B. Donnet. Stability and consistency of the lisp pull routing architecture. In *Proc. Workshop on Network Measurement (WNM)*, November 2016.

[83] M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright. Virtual extensible local area network (vxlan): A framework for overlaying virtualized layer 2 networks over layer 3 networks. Technical report, 2014.

[84] F. Maino, V. Ermagan, A. Cabellos, and D. Saucez. Lisp-security (lisp-sec). *draft-ietf-lisp-sec-14, Internet Engineering Task Force*, 2017.

[85] T. Marill and L. G. Roberts. Toward a cooperative network of time-shared computers. In *Proceedings of the November 7-10, 1966, fall joint computer conference*, pages 425–431. ACM, 1966.

[86] M. Menth, D. Klein, and M. Hartmann. Improvements to lisp mobile node. In *Teletraffic Congress (ITC), 2010 22nd International*, pages 1–8. IEEE, 2010.

[87] D. Meyer, D. Lewis, D. Farinacci, and C. White. Lisp mobile node. *draft-meyer-lisp-mn-16. txt, Internet Engineering Task Force*, 2016.

[88] D. Minoli. Security considerations for mipv6. *Mobile Video with Mobile IPv6*, pages 173–189.

[89] P. Mockapetris. Rfc 1034: Domain names-concepts and facilities, 1987. *URL: ftp://ftp. isi. edu/in-notes/rfc1034. txt*.

[90] M. Möller, R. Bye, K. Bsufka, S. A. Camtepe, and S. Albayrak. From simulation to emulation: an integrated approach for network security evaluation. *Lecture Notes in Informatics*, 192, 2011.

[91] D. Montero, M. Siddiqui, R. Serral-Gracia, X. Masip-Bruin, and M. Yannuzzi. Securing the lisp map registration process. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 2145–2151. IEEE, 2013.

[92] R. Moskowitz, T. Heer, P. Jokela, and T. Henderson. Host identity protocol version 2 (hipv2). Technical report, 2015.

[93] R. Moskowitz and M. Komu. Host identity protocol architecture. *draft-ietf-hip-rfc4423-bis-18*, 2017.

[94] A. R. Natal, L. Jakab, M. Portolés, V. Ermagan, P. Natarajan, F. Maino, D. Meyer, and A. C. Aparicio. Lisp-mn: mobile networking through lisp. *Wireless personal communications*, 70(1):253–266, 2013.

[95] P. Nikander, A. Gurtov, and T. R. Henderson. Host identity protocol (hip): Connectivity, mobility, multi-homing, security, and privacy over ipv4 and ipv6 networks. *IEEE Communications Surveys & Tutorials*, 12(2):186–204, 2010.

[96] E. Nordmark and M. Bagnulo. Shim6: Level 3 multihoming shim protocol for ipv6. Technical report, 2009.

[97] M. O'DELL. Gse-an alternate addressing architecture for ipv6. *http://www. watersprings. org/pub/id/draft-ietf-ipngwggseaddr-00. txt*, 2006.

[98] C. Perkins. RFC 3344. *IP Mobility Support for IPv4*, 8, 2002.

[99] C. Perkins, D. Johnson, and J. Arkko. Mobility support in ipv6. Technical report, 2011.

[100] C. E. Perkins. Mobile ip. *IEEE communications Magazine*, 35(5):84–99, 1997.

[101] D. Phoomikiattisak and S. N. Bhatti. Control plane handoff analysis for ip mobility. In *Wireless and Mobile Networking Conference (WMNC), 2016 9th IFIP*, pages 65–72. IEEE, 2016.

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

[103] A. Raheem, A. Lasebae, M. Aiash, and J. Loo. Supporting communications in the iots using the location/id split protocol: a security analysis. In *Future Generation Communication Technology (FGCT), 2013 Second International Conference on*, pages 143–147. IEEE, 2013.

[104] M. K. Rana, B. Sardar, S. Mandal, and D. Saha. Implementation and performance evaluation of a mobile ipv6 (mipv6) simulation model for ns-3. *Simulation Modelling Practice and Theory*, 72:1–22, 2017.

[105] I. RFC0791. Internet protocol. *J. Postel. September*, 1981.

[106] A. Rodriguez-Natal, S. Barkai, V. Ermagan, D. Lewis, F. Maino, and D. Farinacci. Software defined networking extensions for the locator/id separation protocol. *IETF LISP Working Group Internet-Draft*, 2014.

[107] J. Rosenberg et al. Sip: Session initiation protocol (jun. 2008) retrieved at http://tools. ietf. org/html/rfc3261. *Relevant pages provided*.

[108] D. Saucez and et al. 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.

[109] D. Saucez and L. Iannone. O. bonaventure," locator/id separation protocol (lisp) threat analysis. Technical report, RFC 7835, DOI 10.17487/RFC7835, April 2016,< http://www. rfc-editor. org/info/rfc7835.

[110] D. Saucez, L. Iannone, O. Bonaventure, et al. Openlisp: An open source implementation of the locator/id separation protocol. *ACM SIGCOMM Demos Session*, pages 1–2, 2009.

[111] D. Saucez, L. Iannone, O. Bonaventure, and D. Farinacci. Designing a deployable internet: The locator/identifier separation protocol. *IEEE Internet Computing*, 16(6):14–21, 2012.

[112] A. Stockmayer, M. Schmidt, and M. Menth. jlisp: An open, modular, and extensible java-based lisp implementation. In *Teletraffic Congress (ITC 28), 2016 28th International*, volume 1, pages 205–208. IEEE, 2016.

[113] Z. Tang, Y. Zhou, W. Deng, and B. Wang. Lisp-hnm: Integrated fast host and network mobility control in lisp networks. In *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2017 15th International Symposium on*, pages 1–6. IEEE, 2017.

[114] R. Thayer, N. Doraswamy, and R. Glenn. Rfc 2411: Ip security document roadmap. *Internet standard, November*, 1998.

[115] V. Veselỳ, M. Marek, O. Ryšavỳ, and M. Švéda. Multicast, trill and lisp extensions for inet. *International Journal on Advances in Networks and Services Volume 7, Number 3 & 4, 2014*, 2014.

[116] V. Veselỳ and O. Ryšavỳ. Locator/id split protocol improvement for high-availability environment. *ICNS 2015*, page 75, 2015.

[117] C. Vogt. Six/one: A solution for routing and addressing in ipv6. *INTERNET DRAFT, draft-vogt-rrg-six-one-01. txt*, 2007.

[118] Y. Wu, K. Chen, K. Xue, and D. Ni. Nemo-based mobility management in lisp network. In *Wireless Communications and Signal Processing (WCSP), 2014 Sixth International Conference on*, pages 1–6. IEEE, 2014.

[119] H. Zhang, M. Chen, and Y. Zhu. Evaluating the performance on IDLoc mapping. In *Proc. IEEE Global Communications Conference (GLOBECOM)*, November 2008.

List of Publications

- Y. Li, D. Saucez, L. Iannone, and B. Donnet. Stability and consistency of the lisp pull routing architecture. In *Proc. Workshop on Network Measurement (WNM)*, November 2016
- Y. Li and L. Iannone. Using ripe atlas to evaluate the locator/id separation protocol. In *Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on INFOCOM 2016 Conference*, pages 696–697. IEEE, 2016
- Y. Li and et al. Performance evaluation of locator/identifier separation protocol through ripe atlas. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference*, pages 561–562. ACM, 2016
- Y. Li, A. Abouseif, L. Iannone, and D. Saucez. Lisp-views: Monitoring lisp at large scale. In *Teletraffic Congress (ITC 29), 2017 29th International*, volume 1, pages 178–186. IEEE, 2017
- Y. Li and L. Iannone. Assessing locator/identifier separation protocol interworking performance through RIPE Atlas. *Computer Networks*, pages –, 2017

Appendix A

Source Code

Some of our implementations are originally written and some are modified from previous open source projects. So we upload related codes to GitHub in my personal repositories as https://github.com/SeleneLI. It is convenient to download and explore the codes. A brief introduction of each repository is provided as follows.

A.1 LISP Stability and Consistency analyzer

https://github.com/SeleneLI/TracesAnalyzer

This project is totally originally written to analyze the stability and consistency of LISP mapping system.

A.2 LISP-Views

https://github.com/SeleneLI/LISP-Views

This project is the original codes for LISP-Views architecture.

A.3 LISP measurements on Atlas

https://github.com/SeleneLI/Atlas

This project is used to conduct the experiment on RIPE Atlas and analyze the ping / traceroute results for LISP interworking performance.

A.4 LISP implementation on ns-3

https://github.com/SeleneLI/LISP_ns-3

This project is an implementation of LISP mobility extensions on an existing LISP simulator under ns-3. It also provides the scripts evaluating LISP mobility performance.