Proceedings of IC-NIDC2012

AN OPEN CONTROL-PLANE IMPLEMENTATION FOR LISP NETWORKS

Dung Phung Chi(1,2), Stefano Secci(2), Guy Pujolle(2), Patrick Raad(3), Pascal Gallard(3)

(1) VNU, Hanoi, Vietnam, dungpc@vnu.edu.vn
(2) LIP6/UPMC, Paris, France, firstname.lastname@lip6.fr
(3) NSS, Lognes, France, firstname.lastname@nss.fr

Abstract

Among many options to tackle scalability issues of the current Internet routing architecture, the Locator Identity Separation Protocol (LISP) seems to be a feasible and effective one. LISP brings renewed scale and flexibility to the network, enabling advanced mobility management, with acceptable scalability and security. This paper gives a brief presentation about an open control-plane implementation of LISP currently working in the lisp4.net testbed. Our implementation includes most LISP control-plane functions, and also a module to allow the integration with an OpenLISP data-plane and, therefore, the deployment of a complete standalone Open-Source LISP Tunnel Router interoperable with existing Cisco LISP implementation[1].

Keywords: Control-Plane, Internet Routing, LISP, VM Mobility

1 Introduction

The basic idea of LISP is to implement a two-level routing on the top of BGP/IP, separating transit networks from edge networks, mapping an IP address, or End-point Identifier (EID), to one or many Routing Locators (RLOCs). RLOCs remain globally routable, while EIDs become provider independent and routable beyond RLOCs [2].

At the data-plane, "map-and-encap" is performed using a mapping cache. The control-plane communications (with a mapping system) handle EID-to-RLOC registrations and resolutions. Many RLOCs can be registered for the same EID; priority and weight metrics are associated with each RLOC to decide which one to use (best priority), or how to do load-balancing (if equal priorities) [3]. When a host communicates with another host on another LISP site, the source sends a native IP packet with EID as the destination IP address; the packet reaches a border router, acting as an Ingress Tunnel Router (ITR), which does mapping lookups for EID-to-RLOCs, appends a LISP header and an external IP header with the ITR as source and an RLOC as destination. The destination RLOC, or Egress Tunnel Router (ETR), strips the outer header and sends the native packet to the destination.

OpenLISP [4] is an open-source implementation of the LISP data-plane, in a FreeBSD environment. Standalone, an OpenLISP node is not able to handle all control-plane signaling within a LISP network, so it has to depend on xTR proxies. Our control-plane implementation aims at filling this gap, including an optional module using the OpenLISP mapping socket, yet being independent from OpenLISP, as detailed hereafter. Our control-plane implementation is currently used to interconnect the LIP6, UNIROMA1, VNU Hanoi, U. Prague and INRIA Sophia Antipolis LISP sites to the worldwide testbed operated by Cisco (http://www.lisp4.net).

The rest of paper is organized as follows. Section II presents the design of our control-plane modules. Section III presents control-plane implementation design, and Section IV describes implementation details. Section V surveys existing related work. Section VI presents future work and concludes the paper.

2 LISP control-plane

The LISP mapping system includes two node types: Map Resolver (MR) and Map Server (MS). A MR accepts requests sent from an ITR and resolves the EID-to-RLOC mapping using a mapping database; whereas a MS learns authoritative EID-to-RLOC mappings from an ETR, including them in a mapping database [5]. In [2], the authors describe the format and the different types of the control-plane mapping system messages, without specifying the Mapping System architecture.

1 *This work has been partially funded by the* ***"Investissement d'Avenir - Developpement de l'Economie Numérique"*** *nu@ge French research program*

Two mapping system architectures have been proposed: the Alternative Topology (LISP-ALT, [6]), based on BGP signaling, and the Delegation Data Tree (LISP-DDT, [7]), inspired by Domain Name System (DNS). Currently, DDT is used in the lisp4.net testbed. It is worth mentioning that there is no interdependency between the mapping system architecture and the LISP control-plane architecture, which strictly includes the signaling message between LISP tunnel router (xTR) and mapping system nodes.

The LISP control-plane messages that we included in our open-source implementation are those indispensable for LISP operations. They are:

- **MAP-REGISTER:** message (authenticated using HMAC) sent by an ETR to MS to register one or many EID-to-RLOC mappings, including RLOC priority and weight metrics;
- **MAP-REQUEST:** message sent by an ITR, or relayed by MS, to an ETR, to get the mapping for a given EID (local probes between xTR of same LISP site are possible to verify the reachability);
- **MAP-REPLY:** message sent by an ETR in response to a map-request, including the mapping information;
- **ENCAPSULATED-CONTROL-MESSAGE (ECM)}:** is used to encapsulate control-plane message between xTR and Mapping System. Currently, only map-request is allowed to be encapsulated and used to send between an ITR and MR;
- **SOLICIT-MAP-REQUEST (SMR):** a map-request message soliciting a mapping update to an ITR. It will trigger a map-request from the ITR to the requester and then a map-reply from the requester.

3 Implementation design

Our implementation includes the basic control-plane functions needed to fully operate a LISP site. In order to operate it independently of or integrated with OpenLISP, it needs to be divided into independent modules in terms of functionality. Another important thing is that the control-plane must support both IPv4 and IPv6, not only for EID but also for MS and MR.

MAP-REGISTER module: implemented at the ETR interface; it sends periodically (60 seconds recommend in [2]) information about EID-to-RLOC mapping registration to at least one MS. ETR authenticated with MS using HMAC and a previously granted key.

An ETR in the lisp4.net testbed is assigned only single key, which can be used to authenticate mapping registrations with all the MSs.

In order to allow higher degree of freedom with respect to the basic LISP specification, our control-plane program allows specifying keys for different MS, so as to, for example, allow an ITR joining many difference LISP networks managed by independent mapping systems.

An organization might indeed manage more than one EID-prefix, coming from many mapping systems, so an ETR is made able to chose which EID will be registered with which MS. Then, inside an organization, some other network entity is supported to orchestrate consistent mapping information across the difference mapping system (e.g., to avoid loop).

The registration information needs to be kept in a local database be used by other modules. In the case of our program, the database is loaded from the configuration file and is kept in live memory during its lifetime. In the case the OpenLISP interworking is used, it sharing database with OpenLISP.

MAP-REPLY module: implemented at the ETR interface, it receives and processes map-requests that may come from the ITR or MS, and may be encapsulated in the ECM or not. An ECM map-request can be a map-request IPv4 encapsulated in an IPv6 ECM or vice versa. Therefore, the module must be protocol independent. In our program, this module includes two children modules for IPv4 and IPv6. The children share two common sockets to allow them to process ECMs.

If a map-request message arrived at the ETR is querying for an EID-to-RLOC mapping, a map-reply is generated and sent. As of the specification, the nonce from map-request is copied in the nonce field of the map-reply, and an EID (or more than one in the case an ETR is configured with overlapping EID-prefixes) with a prefix length that is less or equal to the EID being request is included in map-reply. In the case the incoming map-request has the SMR format (special bit set to 1), if our control-plane program is integrated with OpenLISP, SMR processing will be made to called update the database of OpenLISP; in case of no integration with OpenLISP, the map-request will be ignored. Details about the SMR process can be found in [2].

toOPENLISP INTERWORKING module: this module allows our control-plane program to work with the OpenLISP data-plane and to have a working xTR. In order to perform data-plane functions, OpenLISP maintains a Mapping Information Database (MIB) consisting of the LISP Cache (storing short lived mappings in an on-demand fashion), and the LISP Database (storing all "local" mappings, used in selecting the appropriate RLOCs when encapsulating or decapsulating packets [2]). OpenLISP provides also a tool to add or remove mappings from the MIB but only using the command line. The main task of this interworking module is to maintain that database by using its control-plane. More precisely, the main functions are:

- Initial building the LISP Database of xTR;
- Listen to request from OpenLisp through the OpenLISP socket [4];
- Send the request to the mapping system;
- Process the map-reply and update the database cache of OpenLISP.

Our program includes an option to use the MIB as the reference database for both the control-plane and the data-plane. At its current version, OpenLISP only defines add/delete functions to maintain database. Based on them, we build our interworking module to perform database updates and to refresh database upon SMR.

4 Implementation overview

Once started, our control-plane program listens on the UDP LISP control port where LISP control-plane message are sent. It is worth recalling that the program is designed to work in a FreeBSD environment. Moreover, it is worth mentioning that the program runs in the user space, while the data-plane OpenLISP runs in the kernel. This is an important feature as it is a good practice to give higher priority to data-plane functions than to control-plane functions.

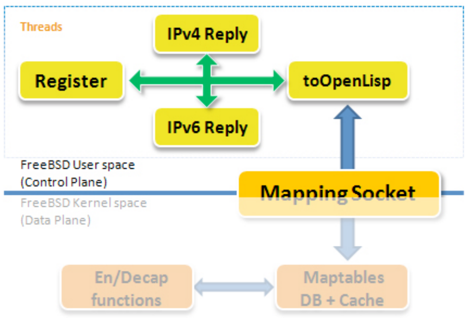

Figure 1. OpenLISP control-plane thread interworking

As depicted in Figure 1, the control-plane program handles four threads: one (register thread) periodically sends a map-register message to the MS; two others (reply threads) handle IPv4 and IPv6 independently, treat map-request messages, and respond with map-reply messages; the last thread (toOpenLISP thread) communicates directly with the OpenLISP data-plane.

Moreover, the reply threads share two sockets (on IPv4 and IPv6) to be able to accept ECMs that may have an IPv4 inner header and an IPv6 outer header, or the other way around.

The control-plane implementation requires basic information in a configuration file: the mapping between the EID-to-RLOC managed locally by the xTR, the IPs of the MSs, the authentication key (optionally, many keys for multiple mapping systems). The file is divided into two main sections: one is for the MS's IPs; the other one can be divided into multiple subsections and handles the EID-to-RLOC mapping database.

The following is an example of the MS section of the configuration file. It contains a list of map-servers (IP address or domain name) with the associated authentication-key ('auth-key' to be replaced with authentication key that the xTRs of the same LISP side use to authenticate with a same mapping system).

Configuration file: mapping-server/resolver part:

```
@MAPSERVER
193.162.145.50                          auth-key
l3-london-mr-ms.rloc.lisp4.net          auth-key
2001:67c:21b4:109::b                    auth-key
@MAPRESOLVER
l3-london-mr-ms.rloc.lisp4.net
```

Each EID-to-RLOC section includes the mapping between one EID and RLOC(s). The first line of the section contains EID's information: EID-prefix, Subnet mask, Time-To-Live (TTL) and Flag. The Flag is used to indicate whether the EID-prefix is registered with the MS or not. Other lines are RLOC's information: RLOC address, Priority, Weight and Local flag. As of the original LISP specification [2], the priority metric is used to prefer one RLOC to another (the lower priority inter value indicates, the higher the priority), and the weight, range from 1 to 100, indicates the load-balancing ratio (if equal priorities). Our program does not handle integrity checks (sum of the weight for all the RLOC with equal priorities equal to 100; priority value between 1 and 100; mask length minor or equal than 32; etc), for the sake of code simplicity. However, these functions may be easily implemented.

Configuration file: EID-to-RLOC part:

```
@EID
#Eid-Prefix  Subnetmask     TTL     Flag
153.16.38.0          25      60        1
#RLOC                 Priority  Weight   Local flag
132.227.62.242            2      100           1
132.227.62.243          1        100           0
2001:660:240::242       5        100           1
2001:660:240::243       3        100           0
```

During its operation, the program also prints debug information to the console output. This feature is very important especially when implementing novel functionalities in LISP and to debug control-plane message errors. The debug information is displayed in two forms: raw package format (in hexadecimal number) and human readable format. Below is an example of console output:

Console output for debug: Map-Register Packet

Raw format:

```
0x3000000459f066a12bc5f51f000100149f5d07b1c234c6
3edba328df82d85492167a9e4c0000000a0220100000000
0199102c750164ff000005000197647a09ff64ff000001000
184e33ef20000000a0120100000000000199102c730164ff0
00005000197647a090000000a0120100000000000199102
c720164ff000005000197647a090000000a012010000000
000199102c760164ff000005000197647a09
```

Human readable format

```
lisp_type              = 3
rloc_probe             = 0
want_map_notify        = 0
record_count           = 4
lisp_nonce             = 0x59f066a1-0x2bc5f51f
key_id                 = 1
key_len                = 20
auth_data =
0x9f5d07b1c234c63edba328df82d85492167a9e4c
#Recodr0
record_ttl             = 10
loc_count              = 2
eid_mask_len           = 32
action                 = 0
auth_bit               = 0xffffffff
lisp_map_version       = 0x0000-0x00000000
eid_afi                = 1
eid_prefix             = 153.16.44.117
RLOC 0: priority=1 weight=100  mpriority=255
mweight=0 rloc_local=1 rloc_probe=0
reach_bit=1loc_afi 1 locator  = 151.100.122.9
RLOC 1: priority=255 weight=100 mpriority=255
mweight=0     rloc_local=0 rloc_probe 0 reach_bit 1
loc_afi 1 locator = 132.227.62.242
```

The control-plane program source code and more detail about research activities on LISP can be found in http://www.lisp.ipv6.lip6.fr (web server accessible via LISP!).

5 Related work

In this section, we review related research about LISP control-plane and related applications.
Significant work has been devoted to the possible deployment LISP for mobility management purposes, in both wireless network and data-center network environments. With a wireless equipment perspective, a specific implementation of the control-plane and the data-plane for mobility nodes exists, called LISP Mobile Node (LISP-MN) [8], developed in the frame of the Lispmob project. The idea is the make a mobile equipment become a complete xTR, with the deployment of a lightweight version of the control-plane and the data-plane into node's operating system (OS). The supported environment is Linux (currently not FreeBSD). When a node moves across LISP sites, it uses control-plane messages to update the mapping between its EID and new RLOC corresponding to its new position. Of course, one possible drawback is the fact that LISP, defined as transit-edge separation architecture, here touches the user equipment, hence affecting the scalability of the solution. A different approach is to maintain the current LISP philosophy and build mechanisms to detect and update the mapping when a node moves across the LISP edge, as presented in [9] for wireless mesh networks; however no details are given about the implementation. Under the first approach, the mobile node can operate and move freely across LISP sites. Under the second approach, a mobile node can only move across LISP sites (remembering that under this approach, the node's OS does not need to be modified).

Some other studies focus on developing mechanisms to manage the caching database of xTR: resolution to refresh caching database when mapping change [2] or cache synchronization when an xTR restarts [11]. Currently, there are three methods are proposed in [2] to refresh caching database: Clock Sweep (based on setting TTL of EID-to-RLOC mapping), SMR (described in previous section), and Database Map Version (each encapsulated package includes a map version field and xTR uses it to detect if one EID-to-RLOC mapping is up-to-date or not). The first method suffers from a long convergence time because lower bounded by the minimum TTL (1 minute) set by the ITR willing to change the mapping and upper bounded the maximum TTL (24h) an ITR shall wait to ensure that all mapping cached in network's ITRs with the previous TTL expire (TTL values as of [2]. The second approach has been already described, the SMR message is sent only to the ITRs present in the mapping cache, asking them to send back a map-request to get the new mapping. The third approach, map versioning, allows an ETR autonomously solicitating ITRs about mapping information update with more recent mappings, using SMRs, hence this works only only with the ITRs still present in the ETR mapping cache, i.e., for which a recent communication exists.

Despite that main purpose behind LISP is to improve the Internet routing scalability and resiliency by decreasing the routing table size and by offering novel traffic engineering features, one of major LISP's fields of application today is data center networking. The mobile node is in this case not wireless equipment, but a virtual machine that can be moved from one network to another, the networks being potentially very far from each other. An advantage in using LISP for virtual machine mobility, rather than DNS or mobile IP, is that the IP address can be the same (not possible with DNS remapping) and no triangular routing is used (not natively possible in mobile IP). There are currently two approaches that we could identify to handle virtual machine mobility (VMM) in a LISP environment: one is based on mapping update upon data-plane traffic detection [10], the other consists in implementing control-plane directly in the mobile node as suggested by [8]. However, the first suffers from security issues (no authentication implies that a hacked zombie virtual machine can generate service interruption redirecting traffic to itself) and low performance for non-streaming VMs (the incoming VM should stream data packets to trigger locator change in the mapping system; otherwise a special process should be run by the VM or the hypervisor). The second approach, as already mentioned, somehow goes against the LISP philosophy of separating transit from edge networks, putting additional burden on the VM in terms of control-plane signaling, and moreover opening the path to security flaws related to OS infection by virus.

6 Future work and perspectives

Our attention is devoted to the lack of security and performance in the current methods to handle VMM in a LISP Cloud environment. Our objective is to define a solution that:

- Implements routing locator priority change upon VM migration;
- Offers the lowest possible convergence time;
- Guarantees a form of authentication and security;

Does not need to modify the VM's OS, hence is run either at the hypervisor level and/or at the switch-router level. The solution should encompass the definition of new control-plane messages in the edge side, from the xTR to the EID, and in the transit side between xTRs and the mapping system. We are interested in defining a solution for VMM in LISP cloud environments that can be readily adopted also for wireless access network environments, so that node mobility functions can be unified for users and machines.

Another open field research is the implementation of LISP traffic engineering modules allowing the definition of specific policies in LISP configuration, e.g., implementing the solution proposed in [4].

Finally, we are currently working in the implementation of an open-source mapping server interface, also compatible with OpenLISP (to date, the MS is implemented only in the Cisco IOS). Our control-plane implementation allows connecting a LISP site to the lisp4.net testbed without the need to buy a router. Moreover, our implementation allows working simultaneously with different mapping systems. From an opensource implementation perspective, the path forward is the implementation of an open-source Mapping Server, to push forward the development of new functionalities for the LISP control plane, and to surround current limitations in the Cisco implementation (e.g., locator count limited to 5 of the 8 available bits).

References

[1] D. Meyer, L. Zhang, K. Fall, "Report from the IAB Workshop on Routing and Addressing", RFC4984, 2007.

[2] D. Farinacci, V. Fuller, D. Meyer, D.Lewis, "Locator/ID Separation Protocol (LISP)", draft-ietf-lisp-23, May, 2012.

[3] S. Secci, K. Liu, G. K. Rao, B. Jabbari, "Resilient Traffic Engineering in a Transit-Edge Separated Internet Routing", in Proc. of IEEE ICC 2011.

[4] D. Saucez, L. Iannone, O. Bonaventure, "OpenLISP: An Open Source Implementation of the (Locator/ID) Separation Protocol", in Proc. of ACM SIGCOMM 2009, Demo paper.

[5] V. Fuller, D. Farinacci, , "LISP Map Server Interface", draft-ietf-lispms-16, March, 2012.

[6] V. Fuller, D. Farinacci, D. Meyer, D. Lewis, "LISP Alternative Topology (LISP+ALT)", draft-lisp-alt-10, December, 2011.

[7] V. Fuller, D. Lewis, V. Ermagan, "LISP Delegated Database Tree", draftfuller-lisp-ddt-01, March, 2012.

[8] D. Farinacci, D. Lewis, D. Meyer, "LISP Mobile Node", draft-meyerlisp-mn-07, October, 2011.

[9] M.F. Almirall, L. Iannone, R. Merz, "Managing Fast Mobility in Wireless Multi-Hop Networks with LISP", 2011.

[10] "Locator ID Separation Protocol (LISP) VM Mobility Solution", Cisco white paper, 2011.

[11] D. Saucez, O. Bonaventure, L. Iannone, C. Filsfils, "LISP ITR Graceful Restart", draft-saucez-lisp-itr-graceful-00, July, 2012.