OpenLISP: An Open Source Implementation of the Locator/ID Separation Protocol

L. Iannone
TUB – Deutsche Telekom Laboratories AG,
luigi@net.t-labs.tu-berlin.de

D. Saucez
Université catholique de Louvain
damien.saucez@uclouvain.de

O. Bonaventure
Université catholique de Louvain
olivier.bonaventure@uclouvain.de

***Abstract*—The network research community has recently started to work on the design of an alternate Internet Architecture aiming at solving some scalability issues that the current Internet is facing. The Locator/ID separation paradigm seems to well fit the requirements for this new Internet Architecture. Among the various solutions, LISP (Locator/ID Separation Protocol), proposed by Cisco, has gained attention due to the fact that it is incrementally deployable. In the present paper we give a short overview on OpenLISP, an open-source implementation of LISP. Beside LISP's basic specifications, OpenLISP provides a new socket-based API, namely the *Mapping Sockets*, which makes OpenLISP an ideal experimentation platform for LISP, but also other related protocols.**

I. Introduction and Motivation

The idea of improving the Internet Architecture with some form of separation between the identity of end-systems and their location in the Internet topology dates from the mid-90s ([1], [2], [3]). The Locator/ID separation paradigm has recently gained momentum due to the increasing concern on some scalability issues with the current Internet Architecture ([4]) and the benefits that such a paradigm provides ([5]). A fair amount of activity has been ongoing in the Routing Research Group (RRG) of the IRTF (Internet Research Task Force), which has been expressly rechartered for this purpose ([6]).

The Locator/ID separation paradigm has several implications, concerning the necessity to map IDs into locators, storing and distributing these mappings, and perform tunnneling or address translation operations in order to forward packets in the core Internet. There are several solution that have been proposed insofar, however, while interesting and promising, these have the drawback of not being incrementally deployable. Such lack of deployability is an obstacle to large scale experimentations.

The Locator/ID Separation Protocol (LISP), proposed by Farinacci et al. [7], and based on a map-and-encap approach, has the main advantage of being incremental deployable on border routers of edge networks, thus limiting the number of systems that need to be upgraded.

The OpenLISP project[1], our open-source implementation of LISP, aims at providing an open and flexible platform for experimentation. To this end, with OpenLISP we went further than the LISP specifications. LISP has a detailed description of the encapsulation and decapsulation operations, the forwarding operation and offer several options as mapping system. Nevertheless there is no specification of an API to allow the mapping system to interact with the forwarding engine. In OpenLISP we proposed and implemented a new socket based solution in order to overcome this issue: the *Mapping Sockets*. Mapping sockets make OpenLISP an open and flexible solution, where different approaches for the locator/ID separation paradigm can be experimented, even if not strictly related to LISP. To the best of our knowledge, OpenLISP is the only existing effort in developing an open source Locator/ID separation approach. The development and the experimentation done with OpenLISP had also an impact on the original LISP specifications, allowing to correct some original design shortcomes and improve some engineering solutions [8].

[1]This work has been partially supported by Cisco and the European Commission within the IST-027609 AGAVE project (www.ist-agave.org) and the INFSO-ICT-216372 TRILOGY Project (www.trilogy-project.org).

II. LISP in a nutshell

LISP is meant to be deployed on border routers whose upstream IP address is used as Routing LOCator (RLOC) for the end-systems of the local domain. End-systems still communicate using legacy IP addresses, which in the LISP terminology are called Endpoint IDentifiers (EIDs). EIDs and RLOCs are both IP addresses, however, while EIDs have only a local scope, thus not routable outside the local domain, RLOCs are only used for inter-domain routing and cannot be used as endpoint identifiers for host-to-host connections. EIDs can be actually associated to a set of RLOCs, since a domain can be multi-homed, *i.e.*, having several border routers. LISP tunnels the packets in the core Internet, using an IP-over-UDP approach, from one of the RLOCs of the source EID to one of the RLOCs of the destination EID. In order to perform such a tunneling, LISP needs to know when to encapsulate or decapsulate a packet and what to put exactly in the header. For this purpose, LISP uses two data structures: the LISP Database and the LISP Cache.

The LISP Database is used to select the source RLOC for outgoing packets and to determine whether an incoming packet needs to be decapsulated. It consists of all EID-Prefix-to-RLOC mappings that are "owned locally". A LISP router owns a mapping if its upstream interface (toward the provider), is in the set of RLOCs associated to the EID-Prefix used as addressing space downstream (*i.e.*, inside the local domain).

The LISP Cache is used to select the destination RLOC for

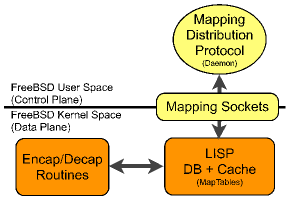

Fig. 1. OpenLISP Architecture.

```
freebsd% map get -inet 192.0.2.1

      Mapping for EID: 192.0.2.1
        EID: 192.0.2.0
        EID mask: 255.255.255.128
        RLOC Addr: inet6 2001:DB8::1
        RLOC Addr: inet  192.0.2.129
        flags: <UP,STATIC,DONE>
```

Fig. 2. Example of `map` usage.

```
freebsd% mapstat -X
 Mapping tables

 Internet:
 EID            Flags  Refs    # RLOC(s)
 192.0.2.0/17   US        1    1 2001:DB8::1
                               2 192.0.2.129
```

Fig. 3. Example of `mapstat` usage.

outgoing packets and contains mappings for EID-Prefixes that are not owned locally. Entries in the LISP Cache are short-lived and subject to timeout, *i.e.*, when a mapping is not used for a certain period the entry is deleted. The LISP Cache is populated in an on-demand fashion. When a packet generates a cache-miss, the mapping system is queried to retrieve the missing mapping.

The purpose of the mapping system is to provide a lookup infrastructure. It usually consists in a Mapping Distribution Protocol, providing such a functionality. There are several of such protocols proposed insofar. A list as well as a comparison can be found in [9].

There is a fair amount of activity on LISP in both the IRTF and the IETF (Internet Engineering Task Force), which is planning to set up a new experimental working group. Cisco, in collaboration with other companies and research institutes, has already deployed its implementation on a testbed (*http://www.lisp4.net*) scattered world-wide, using the LISP-ALT mapping system [10].

III. Overview on OpenLISP

OpenLISP, whose high-level architecture is depicted in figure 1, is our implementation of LISP in the FreeBSD operating system. The forwarding engine of OpenLISP, which includes functions for encapsulation and decapsulation, has been implemented directly in the kernel space, along with both LISP's cache and database. In OpenLISP the two databases are merged in a single radix tree data structure [11] called MapTable. Radix trees provide efficient and fast indexing for all the EID-Prefixes that need to be stored in the system. EID-Prefixes that are part of the LISP Database are tagged with a "local" flag. This allows to maintain a logical separation between the LISP Cache and the LISP Database, since, when performing lookups, it is possible to limit the search scope only to entries tagged in a specific way.

Concerning the mapping system, OpenLISP does not provide any specific Mapping Distribution Protocol. The reason of this choice is because our aim was to develop a flexible and extensible platform providing support for future experimentation of both new and existing Mapping Distribution Protocols. Nonetheless, we provided OpenLISP with some simple tools to have access and to control OpenLISP from a shell terminal. In particular, the `map` utility allows to manipulate the networks' MapTables. Figure 2 shows an example of usage of `map` to perform a lookup for the EID 192.0.2.1. The `mapstat` command, instead, allows retrieving and displaying various contents of network-related LISP data structures. Figure 3 shows an example of usage of the `mapstat` tools to dump the content of the MapTables. The description of both `map` and `mapstat` can be found in the appendix of [8].

The interaction between user space and kernel space is possible thanks to the new socket API that we developed in OpenLISP, namely the *Mapping Sockets*. Mapping sockets allow Mapping Distribution Protocols (or tools like `map` and `mapstat`) running in the user space to send messages to the kernel space in order to perform operations and modify kernel's data structure and receive confirmation messages. Moreover, mapping sockets also offer signaling functionality the other way around, allowing the kernel to notify daemons in user space of specific events related to LISP (*e.g.*, cache-miss).

Work is ongoing to integrate the Cisco testbed with OpenLISP boxes. The advantage of OpenLISP is that it is not bound to the LISP-ALT, but it is open for development and, more importantly, experimenting new mapping systems.

IV. Conclusion

Thanks to the mapping sockets, OpenLISP provides an open and extensible platform to experiment new mapping distribution protocols and traffic engineering techniques. OpenLISP is freely available at: *http://inl.info.ucl.ac.be/OpenLISP*.

References

[1] J. Saltzer, "On the Naming and Binding of Network Destinations," IETF Network Working Group, RFC 1498, August 1993.
[2] R. Hiden, "New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG," IETF Network Working Group, RFC 1955, June 1996.
[3] M. O'Dell, "GSE - An Alternate Addressing Architecture for IPv6," IETF Network Working Group, Internet Draft *draft-ietf-ipngwg-gseaddr-00.txt*, February 1997.
[4] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," IETF Network Working Group, Tech. Rep. RFC 4984, September 2007.
[5] B. Quoitin, L. Iannone, C. de Launois, and O. Bonaventure, "Evaluating the Benefits of the Locator/Identifier Separation," *Proc. 2nd ACM SIGCOMM Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, August 2007.
[6] T. Li, "Design Goals for Scalable Internet Routing," Internet Research Task Force, Tech. Rep., July 2007.
[7] D. Farinacci, V. Fuller, D. Oran, D. Meyer, and S. Brim, "Locator/ID Separation Protocol (LISP)," IETF Network Working Group, Internet Draft *draft-farinacci-lisp-11.txt*, December 2008.
[8] L. Iannone, D. Saucez, and O. Bonaventure, "OpenLISP Implementation Report," IETF Network Working Group, Internet Draft *draft-iannone-openlisp-implementation-01.txt*, July 2008.
[9] L. Mathy and L. Iannone, "LISP-DHT: Towards a DHT to map identifiers onto locators," in *Proc. of ReArch'08 - Re-Architecting the Internet.*, December 2008.
[10] D. Farinacci, V. Fuller, and D. Meyer, "LISP Alternative Topology (LISP-ALT)," IETF Network Working Group, Internet Draft *draft-fuller-lisp-alt-03.txt*, October 2008.
[11] G. Wright and W. Stevens, *TCP/IP Illustrated Volume 2, The Implementation*, ser. Professional Computing Series. Addison-Wesley, 1995.