Evaluating the Performance of LISP Mapping Systems*

Loránd Jakab†, Albert Cabellos†, Florin Coras‡, Damien Saucez§, Olivier Bonaventure§
†UPC, Barcelona, Spain ‡TUCN, Cluj-Napoca, Romania §UCL, Louvain-la-Neuve, Belgium
{ljakab,acabello}@ac.upc.edu, florin.coras@net.utcluj.ro,
{FirstName.LastName}@uclouvain.be

ABSTRACT

The Locator/Identifier Separation Protocol (LISP) is currently being developed within the IETF as an incrementally deployable new Internet architecture that should scale better than the current one. In this architecture end hosts use endpoint identifiers that are not announced in the global routing tables and LISP relies on a mapping system to map these identifiers onto the routing locators that allow to reach them. The performance of the chosen mapping system will be an essential factor in LISP deployment. In this paper, we first review two mapping system proposals: LISP+ALT, considered by the LISP IETF working group and LISP-DHT. We then describe CoreSim, our simulator that allows to evaluate the performance of mapping systems by replaying packet traces collected on border router links. We use it with two traces collected at two different campuses and compare LISP+ALT and LISP-DHT in terms of mapping delay, mapping load, and packet buffering. Our evaluation reveals several drawbacks of LISP-DHT compared to LISP+ALT and shows that the design of a mapping system influences its performance.

1. INTRODUCTION

During the last years, several operators and researchers have expressed concerns about the continued growth of the BGP routing tables in the Default Free Zone [8]. Several factors contribute to this growth, including multihoming, traffic engineering and current address allocation policies among others [21]. This is not the first time that the Internet has to face a problematic growth of its BGP routing tables. In the early nineties, the BGP routing tables were growing at an alarming rate and the available routers could not all cope with this growth. Furthermore, the class-B addresses were becoming exhausted. The IETF developed two different solutions to solve these problems. The short-term solution, Classless Interdomain Routing [24] was quickly deployed and allowed to reduce the growth of the BGP routing tables during about ten years [8]. In parallel, Network Address Translation (NAT) was proposed and became quickly successful. The long-term solution was (and still is) IPv6.

Today’s Internet is facing similar problems. The IPv4 addressing space will be fully used within about two years [9] and the growth of the BGP routing tables is again on the agenda. IPv6 is still the long-term solution, but a pure IPv6 Internet would also likely suffer from growing BGP routing tables due to factors such as multihoming, traffic engineering and the pressure for provider-independent address allocations [22].

The Routing Research Group (RRG) of the Internet Research Task Force (IRTF) has been chartered to develop an Internet architecture that scales better than the current one. Among the proposals that have been discussed in this working group [12, 5, 20], the Locator/Identifier Separation Protocol (LISP) [5, 20] is the most advanced one. Like many of the proposals to better scale the Internet architecture, LISP separates the two roles of addresses. LISP considers two different types of addresses: Endpoint Identifiers (EIDs) and Routing Locators (RLOCs). EIDs are allocated to sites in a provider-independent manner, but they are not advertised in the global BGP routing tables. The global BGP routing tables only contain the RLOCs and these are highly aggregated by transit network providers to help scale the BGP routing tables. Furthermore, such a separation between locators and identifiers provides both smaller BGP routing tables and added benefits in terms of performance [23] or traffic engineering capabilities [25]. A mapping system must be used to map
The identifiers on the RLOCs that allow to reach them. The mapping systems are a key part of the LISP solution. The overall performance of LISP will heavily depend on the performance of the mapping systems. Besides a first analysis of the mapping cache [10] there has been no evaluation of the performance of the proposed LISP mapping systems. Our contributions are twofold. First, we propose CoreSim, a simulator that allows to evaluate the performance of LISP mapping systems. CoreSim is extensible and can support different types of mapping systems. Second, we use CoreSim to compare the performance of two LISP mapping systems: LISP+ALT and LISP-DHT.

This paper is organised as follows. Sec. 2 gives an overview of LISP and the two mapping systems under study. It is followed by the description of CoreSim, the simulator used for the comparison in Sec. 3. The simulation results are presented and discussed in Sec. 4, and later contrasted with the related work in Sec. 5. Finally, the paper is concluded in Sec. 6.

2. LISP IN A NUTSHELL

To forward packets between endhosts, LISP relies on mapping and encapsulation. Each LISP site contains at least one Ingress/Egress Tunnel Router (xTR) that has at least one RLOC. For example, in Fig. 1, the site at the bottom has two xTRs: R1 and R2. In this example, we use IPv6 addresses as EIDs and IPv4 addresses as RLOCs. The EIDs belonging to the reserved 0100::/8 prefix are not advertised in the BGP routing tables.

When an endhost in the bottom LISP site in Fig. 1, e.g. 0100:FE::1234, needs to contact remote endhost, e.g. 0100:DD:1234 it sends a normal (IPv6 in this case) packet with the destination EID as destination. This packet is intercepted by the site's ITR (i.e., R1 or R2). To forward the packet, this ITR needs to obtain the RLOCs of the xTRs attached to the destination EID. For this, the ITR queries the mapping system. Several mapping systems [15, 3, 6, 19] are being developed for LISP. We discuss two of them in more details in section 2.1. The mapping system will return to the ITR a set of RLOCs that allows to reach the destination EID. A priority and a weight are associated to each RLOC, to allow a LISP site to control its incoming traffic. Usually, the mapping system will return a mapping that is valid for an EID prefix. For example, in Fig. 1, the mapping system could return a mapping that associates 0100:DD/48 to 3.1.1.2 and 2.2.1.2. Once the mapping has been received, it is installed in the mapping cache of the ITR and the packet is sent through a tunnel to the RLOC of the destination ETR. The destination ETR decapsulates the packet and forwards it to the destination endhost. Subsequent packets sent to this EID will be forwarded based on the cached mapping.

2.1 Mapping systems

The mapping system is a major component of the Locator/Identifier Separation Protocol as it is the mapping system that makes the association between an identifier and its locators. Typically, when an ITR needs a mapping for an EID, it sends a Map-Request for that EID to the mapping system [5]. The mapping system then forwards the request to a node that is able to provide a mapping for the EID and a Map-Reply is sent back to the ITR. The Map-Reply contains a list of RLOC that can be used to reach the requested EID. Usually, the mapping associates an EID prefix to a set of RLOCs. This implies that a single mapping can be used to reach all the EIDs of the prefix. The received mappings are stored in a mapping cache maintained by the ITR. Mappings also have a Time To Live (TTL). The mapping cannot be used for a duration longer than its TTL. After the expiration of the mapping TTL, it must be either refreshed or removed from the mapping cache. The suggested default TTL for the mappings is 24h [5]. Map-Requests and Map- Replies are detailed in [5].

Different types of mapping systems are possible. In Push-based mapping systems, the ITR receives and stores all the mappings for all EID prefixes even if it does not contact them. Push-based mapping systems have thus similarities with today's BGP. NERD [15] is a Push-based mapping systems that was proposed within the IETF. In Pull-based mapping systems, the ITR sends queries to the mapping system every time it needs to contact a remote EID and has no mapping for it. The mapping system then returns a mapping.
for a prefix that contains this EID. Pull-based mapping systems have thus similarities with today’s DNS. Proposed Pull-based mapping systems include LISP+ALT [6], LISP-CONS [3] and LISP-DHT [19]. In this paper, we focus on LISP+ALT and LISP-DHT that we describe in more details in Sec. 2.1.1 and Sec. 2.1.2, respectively.

An advantage of the Push-based model is that, at any time, each ITR knows the mapping of any possible EID meaning that it can encapsulate any packet that has a valid destination EID. Drawbacks of this approach are that each ITR has to maintain a full mapping database and that every change in the database has to be advertised to all the ITRs. On the other hand, in the Pull-based model, the ITRs maintain only the mapping they need. This makes ITRs scale with the actual traffic they carry. Unfortunately, when a mapping is not known for a particular EID, a Map-Request has to be sent and all the packets towards this EID must be discarded or buffered until a mapping has been received.

2.1.1 LISP+ALT

The LISP Alternative Topology (LISP+ALT) [6] is a mapping system distributed over an overlay. Each node participating in the overlay is connected to some others with a GRE tunnel. The routing table of the overlay is built thanks to BGP. The BGP part of LISP+ALT is used to announce EIDs prefixes making the EIDs routable in the overlay. The mappings are not advertised by BGP. When an ITR needs a mapping it sends a Map-Request to an ALT node. The destination address of the request is the EID for which the mapping has to be retrieved and the source address is the RLOC of the ITR. The ALT nodes then forward the Map-Request on the overlay by inspecting their ALT routing tables. When the Map-Request reaches the ALT node responsible for the mapping, it generates a Map-Reply that is directly sent to the ITR’s RLOC, without using the ALT overlay.

2.1.2 LISP-DHT

LISP-DHT [19] is a mapping system based on a Distributed Hash Table (DHT). The LISP-DHT mapping system uses an overlay network derived from Chord [26]. In a traditional Chord DHT, nodes chose their identifier randomly. In LISP-DHT, a node is associated to an EID prefix and its Chord identifier is the highest EID in the associated EID prefix. This provides mapping locality that ensures that a mapping is always stored by a node chosen by the owner of the EID prefix.

When an ITR needs a mapping, it sends a Map-Request through the LISP-DHT overlay with the source address being its RLOC. Each node routes the request according to its finger table (a table that associates a next hop to a portion of the space covered by the ring). The Map-Reply is sent directly to the ITR via its RLOC.

3. SIMULATION MODEL

Work on the LISP specification and mapping system proposals is currently underway in the LISP WG of the IETF. The main protocol and LISP+ALT, one of the mapping systems, already have an experimental implementation for Cisco NX-Os, deployed in a testbed of about 20 nodes at the time of this writing. An open source implementation called OpenLISP also exists for the FreeBSD operating system [11]. These implementations help validate the proposed protocols. However, it is difficult to evaluate their operation at a large scale. For the purpose of such as evaluation we developed CoreSim, an Internet-scale LISP deployment simulator. CoreSim has a hybrid event/trace based architecture and is able to replay a packet trace to simulate the behaviour of an ITR and the associated Mapping system.

3.1 The CoreSim Simulator

CoreSim works on top of a topology based on measurements performed by the iPlane infrastructure, which provides Internet topology information and the latency between arbitrary IP addresses. The simulator and reports mapping lookup latency, the load imposed on each node of the mapping system and cache performance statistics.

CoreSim is composed of two main building blocks (see Fig. 2): the first one simulates an ITR with its associated operations (sending Map-Requests, buffering packets, caching Map-Replies, and forwarding packets), while the second is responsible for the mapping system (path taken and latency of the Map-Requests). In our hybrid architecture the ITR simulation block is event based: each packet arrival generates an event that has to be processed. In contrast, the mapping system block processes Map-Requests, calculates the appropriate metrics and returns the results instantly, without generating an event after the lookup latency time elapses. The blocks are described in details in Sec. 3.2 and following.

Topology.

The first element to model the behaviour of mapping systems in a large scale Internet is the network itself. For CoreSim, we chose to use the current Internet as the reference topology.

Tunnel Routers.

More precisely, the topology used in the simulator is composed of all the points of presence (PoPs) as defined by the iPlane measurements set [18]. This dataset

1See http://lisp4.net for more details.
2Available at http://iplane.cs.washington.edu/
associates several IP prefixes addresses with each PoP and provides the links that it has to other PoPs. For our simulations we assume that one LISP tunnel router (xTR) will be deployed at the border router of each autonomous system. We use the PoP connectivity information to select xTRs for our simulated topology, as follows. First, we determine the AS number of all PoPs, and count the number of links that each PoP has to PoPs in other ASes (inter-domain links). We consider as the AS border router and LISP xTR the PoP with the highest number of inter-domain connections. The iPlane dataset yielded 14340 such PoPs, which is almost half of the ASes participating in the Internet today.

EID Prefixes.
Choosing the xTRs is the first step in building the LISP topology the simulator uses. To complete the topology, each xTR is assigned several EID prefixes. The iPlane dataset contains the prefixes currently advertised in the Default Free Zone (DFZ), along with the AS that originates them. We removed from the iPlane dataset the more specific prefixes because many of them are advertised for traffic engineering (TE) purposes. Since LISP has native support for TE by means of priorities and weights, these prefixes need not be advertised separately in the mapping system. Using the list of the xTRs for each AS determined in the previous step, a total number of 112,233 prefixes are assigned based on originating AS to their respective xTR.

Delays.
To be able to simulate the time required to obtain a mapping from a mapping system, CoreSim must have information about the delays between the nodes in the modelled topology. For this, we rely also on the iPlane platform. It combines BGP looking glass information with a large number of daily traceroutes from over 300 vantage points [17, 18]. iPlane includes an Internet latency lookup service that returns an estimated latency between arbitrary IP addresses. Unfortunately, iPlane does not provide all possible delay pairs. When iPlane provides the delay between a pair of nodes, CoreSim uses the iPlane delay. This corresponds to about 65% of the pairs in our simulations. For the other 35% pairs, we developed a latency estimator that fills the iPlane gaps. The estimator correlates the geographical distance between IP addresses as reported by the MaxMind database \(^3\) with the latency, based on a training dataset from iPlane (see details in [4]).

Since the simulator works with PoPs and iPlane resolves latencies for pair of IP addresses, whenever calculating the latency between two PoPs we use all their associated IP addresses, and start lookup up all combinations, until a result is returned. Since iPlane doesn’t have 100% coverage, this increases the chances that we get a result. We use our estimator only if none of the combinations yields a response.

Note that iPlane returns unidirectional latency estimates, and differentiates between the source and destination IP address. For our simulations we decided to consider Internet latencies symmetrical, for the same reason of improving iPlane latency lookup rate. When we need the latency from \(A \rightarrow B\) we prefer to use the iPlane result for \(A \leftarrow B\) rather than estimate it ourselves, especially since it was shown that that roughly 80% of the end-to-end paths in the Internet are symmetric [7].

3.2 Modelling ITRs
CoreSim simulates how an ITR would behave in today’s Internet with today’s traffic. To achieve this, we select one PoP from the border PoP list described in the previous section based on the location where the trace was captured. When the ITR module is bound to a topological location we start feeding it with the packet trace.

The ITR is composed of three modules: the mapping cache module (mapCache), the packet buffer module (packetBuffer), and the module responsible of tracking ongoing map requests (inFlightBuffer). They work as follows:

mapCache The mapping cache module maintains a list of all prefixes that are currently in the cache after a successful Map-Request, along with the timestamp when they will expire from the cache. For each hit on a prefix, the timestamp is updated to \(t_{\text{packet}} + T\) where \(t_{\text{packet}}\) is the packet timestamp and \(T\) the minimum lifetime of an entry in

---

the cache. We used a 3 minutes timeout $T$ for the simulations in this paper. This approach permits to maintain entries in the cache if there are packets more than once every $T$ minutes. It is important to notice that we considered that mappings have a TTL longer than the duration of the trace meaning that entries are never refreshed with a Map-Request once they enter the cache.

**packetBuffer** The packet buffer module holds the buffered packets that caused a cache miss and must wait for the Map-Reply to be received before they can be sent. application

**inFlightBuffer** This module keeps track of all ongoing mapping requests, so that the second and the following packets destined to the particular prefix can be buffered.

Algorithm 1 describes the details of the ITR module implementation. For each arriving packet the simulator first evaluates the timestamp, removes from the buffer packets for which a Map-Reply arrived in the last inter-packet interval, and writes statistics to disk. Next, the ITR examines if there is an ongoing Map-Request for the destination of the packet, in which case the packet is buffered. Otherwise, the RLOC for the destination is is buffered. For a cache hit the entry’s timeout value is updated to $t_{packet} + T$. In the case of a cache miss, the mapping system module is used to determine the lookup latency and path, an entry is added to the cache, the packet is buffered and the inFlightBuffer is updated to save state. Packets with destinations without a valid mapping are dropped.

**Algorithm 1 ITR packet processing**

```plaintext
Prune expired inFlightBuffer entries
Send outstanding packets from buffer
if ∃ ongoing Map-Request for destination EID then
    /* inFlight Hit */
    Buffer packet
else if ∃ Mapping in mapCache for destination EID then
    /* mapCache Hit */
    Update Timeout ← $t_{packet} + T$
else
    /* mapCache Miss */
    Perform lookup
    if ∃ Mapping then
        /* Mapping found */
        Create new mapCache entry
        Create new inFlightBuffer entry
        Set Timeout ← $t_{packet} + T$
        Buffer the packet
    else /* No mapping found */
        Discard packet
end if
end if
```

### 3.3 Mapping Systems

In CoreSim, a LISP Mapping System is modelled as an overlay between nodes of the topology. The organisation of the overlay depends on the considered mapping system. The simulator routes Map-Requests from the node associated to the ITR to the node responsible for the mapping. The path of the query is recorded and the respective latencies are added together, to compute path length (number of hops and time) and record statistics on the number of queries routed by individual nodes.

#### 3.3.1 LISP+ALT model

The LISP+ALT draft [6] was used as starting point to implement the LISP+ALT mapping system. This document describes the characteristics that ALT Routers must have, and envisages a hierarchical topology built with GRE tunnels. However, there is no detailed proposal on the exact organisation of a LISP+ALT overlay and the current lisp4.net testbed is too small to be used as a model.

A LISP+ALT overlay can be organised in many different ways. It could be a star-shaped network with all ITRs connected to a central node. It could also be a full-mesh topology or anything in between such as the current Internet. To build the LISP+ALT topology, we took an optimistic assumption and assumed that to preserve the scalability of the LISP+ALT topology it would have a hierarchical structure. Such a structure would allow ALT nodes to use aggregation when advertising EID prefixes on the overlay. We assume that there are no link failures that affect the topology of ALT overlay and assume it to be static during the entire simulation.

As pointed out before, CoreSim proposes to simulate how a LISP mapping system would behave in today’s Internet. For that reason, it uses the current set of prefixes existing in the DFZ, filtered according to the description in Sec. 3.1. Based on discussions on the IETF mailing list, a three-layer hierarchy was decided (see Figure 3). In this hierarchy, the bottom leaf nodes are ALT routers belonging to a certain domain, which may or may not be the same physical entity as the xTR (the figure depicts them as different entities). The upper two layers are dedicated ALT routers, which may be offered as a commercial service by providers or registries. For example, RIPE currently hosts one ALT node for the lisp4.net testbed. Each of these routers is responsible for a certain aggregated prefix, connecting to all leaf nodes included in that prefix. Our hierarchical topology consists of 16 root layer ALT routers, responsible for the eight /3 prefixes and 256 second layer ALT routers, each responsible for a /8 prefix. For each /3 prefix two ALT routers at different locations are used and each lower layer ALT router connects to the one with lowest latency. All these 16 routers are connected

---

3.3 Mapping Systems

In CoreSim, a LISP Mapping System is modelled as an overlay between nodes of the topology. The organisation of the overlay depends on the considered mapping system. The simulator routes Map-Requests from the node associated to the ITR to the node responsible for the mapping. The path of the query is recorded and the respective latencies are added together, to compute path length (number of hops and time) and record statistics on the number of queries routed by individual nodes.

3.3.1 LISP+ALT model

The LISP+ALT draft [6] was used as starting point to implement the LISP+ALT mapping system. This document describes the characteristics that ALT Routers must have, and envisages a hierarchical topology built with GRE tunnels. However, there is no detailed proposal on the exact organisation of a LISP+ALT overlay and the current lisp4.net testbed is too small to be used as a model.

A LISP+ALT overlay can be organised in many different ways. It could be a star-shaped network with all ITRs connected to a central node. It could also be a full-mesh topology or anything in between such as the current Internet. To build the LISP+ALT topology, we took an optimistic assumption and assumed that to preserve the scalability of the LISP+ALT topology it would have a hierarchical structure. Such a structure would allow ALT nodes to use aggregation when advertising EID prefixes on the overlay. We assume that there are no link failures that affect the topology of ALT overlay and assume it to be static during the entire simulation.

As pointed out before, CoreSim proposes to simulate how a LISP mapping system would behave in today’s Internet. For that reason, it uses the current set of prefixes existing in the DFZ, filtered according to the description in Sec. 3.1. Based on discussions on the IETF mailing list, a three-layer hierarchy was decided (see Figure 3). In this hierarchy, the bottom leaf nodes are ALT routers belonging to a certain domain, which may or may not be the same physical entity as the xTR (the figure depicts them as different entities). The upper two layers are dedicated ALT routers, which may be offered as a commercial service by providers or registries. For example, RIPE currently hosts one ALT node for the lisp4.net testbed. Each of these routers is responsible for a certain aggregated prefix, connecting to all leaf nodes included in that prefix. Our hierarchical topology consists of 16 root layer ALT routers, responsible for the eight /3 prefixes and 256 second layer ALT routers, each responsible for a /8 prefix. For each /3 prefix two ALT routers at different locations are used and each lower layer ALT router connects to the one with lowest latency. All these 16 routers are connected.
to each other with a fully meshed topology.

To calculate the lookup latency in ALT, the simulator uses the destination address of the packet that triggers a Map-Request and routes the packet via the shortest path through the hierarchy starting from the bottom layer. Each node through which the Map-Request passes has a FIB with the prefixes that it “sees”: all prefixes of the child nodes and aggregated prefixes from the parent node. Packet forwarding proceeds according to the FIB entries and the path is recorded. The topology module then computes the latency of all segments that the Map-Request traversed, and all are added together. A Map-Request that traverses the path $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$ will have the latency calculated as

$$L_{MR} = L_{AB} + L_{BC} + L_{CD} + L_{DE} + L_{EA},$$

where $L_{AB}$ is the latency between routers $A$ and $B$. The last segment, $L_{EA}$, corresponds to the Map-Reply.

### 3.3.2 LISP-DHT model

LISP-DHT uses a slightly modified Chord protocol, with a 32 bit ChordID namespace for IPv4 address compatibility. Whenever searching for keys (EIDs to be looked up), no hashing of the key takes place unlike in the original protocol. Nodes entering the Chord ring (Mapping Servers) choose as ChordID the highest IP address in their prefix. If a domain has several non-contiguous prefixes, their mapping server will participate in the DHT with a separate instance for each such prefix.

The iPlane dataset contained 112,233 prefixes after filtering, requiring a Chord ring of the same size. To the best of our knowledge, all available Chord simulators use an event-based approach, which cannot scale to such large number of nodes. Using 112,233 instances of a real Chord implementation isn’t feasible either. To solve the issue, we made the assumption that there are not link and node failures and thus LISP-DHT is modelled as a static Chord overlay. This implies that we consider no LISP site connects to or is removed from the LISP-DHT overlay during a simulation. Handling churn in such a large scale Mapping System is left for further work.

Since the LISP-DHT is stable for the entire simulation, our LISP-DHT model starts from all the EID prefixes and computes the finger tables of all nodes on the overlay. This is possible as in Chord the finger tables only depend on the ChordIDs of the nodes and not on when the nodes have joined the ring.

To look up a destination EID in the DHT, the regular Chord protocol is used: the originating node looks for the successor of the EID in its finger table and sends the Map Request to that node. Our model considers two modes for the queries: recursive and iterative modes. In recursive mode, the originating node sends the Map Request to the closest node according to its finger table. The closest node then forwards the Map Request. This continues until the node responsible for the queried EID prefix receives the Map Request and replies directly. In iterative mode, the same nodes are involved in message routing, but each node replies to the originating node with the ID of the next hop, until it learns the RLOC of the node responsible for the queried EID. In both cases, the path is recorded and the lookup latency is calculated. Suppose the Map-Request traverses nodes $A, B, C, D,$ and $E$, with $A$ being the originating node.

For simulating LISP-DHT recursive mode the latency is calculated as

$$L_{MR} = L_{AB} + L_{BC} + L_{CD} + L_{DE} + L_{EA},$$

In the case of iterative mode, we have:

$$L_{MR} = 2 \cdot (L_{AB} + L_{AC} + L_{AD} + L_{AE}).$$

As previously noted, the topology is considered symmetrical in terms of latency, and the round-trip times are computed by doubling the latency from the originating node to each node the query passes through.

The simulator source code and all the results are publicly available from [http://www.cba.upc.edu/lisp](http://www.cba.upc.edu/lisp).

### 4. SIMULATION RESULTS

This section presents an evaluation of LISP+ALT and LISP-DHT with the CoreSim simulator. We first present the traces used for the simulations and then discuss the mapping lookup latency and the load supported by the mapping system nodes. We finally show the feasibility of packet buffering during a miss.

#### 4.1 Experimental Datasets

In order to evaluate the performance of the mapping systems presented above we used traffic traces collected at the border routers of two university campuses, the place where a LISP tunnel router would most likely be deployed. The first trace was captured at Université catholique de Louvain (UCL) in NetFlow format, and
the second is a packet trace from Universitat Politècnica de Catalunya (UPC).

4.1.1 UCL

The UCL campus is connected to the Internet with a 1Gbps link via the Belgian national research network (Belnet). This trace consists of a one day full NetFlow trace collected on March 23, 2009. For this paper, only the outgoing traffic is considered, representing 752GB of traffic and 1200 million packets for an average bandwidth of 69Mbps. A total number of 4.3 million different IP addresses in 123,804 different BGP prefixes have been contacted by 8,769 different UCL hosts during the 24 hours of the trace. The UCL campus is accessible to more than 26,000 users. TCP carries 96% of the volume in the UCL trace.

NetFlow generates level 4 flow traces but the simulator requires packet traces. This is why the NetFlow trace collected at UCL has been converted into a packet trace: for each flow, we generated the number of packets specified in the NetFlow record, distributed evenly across the flow duration and the size of the flow. Throughout the rest of the paper, the term UCL trace corresponds to the emulated packet trace from the NetFlow trace collected at UCL.

4.1.2 UPC

The second unidirectional trace we used was captured at the 2Gbps link connecting the UPC campus network to the Catalan Research Network (CESCA) with the help of the CoMo infrastructure [2]. It consists of the egress traffic on May 26, 2009 between 08:00-11:49 local time, and contains about 1200 million packets accounting for 463GB of traffic with an average bandwidth of 289Mbps. 4.3 million distinct destination IP addresses from 111,492 BGP prefixes were observed in the trace. UPC Campus has more than 36,000 users and TCP carries 94% of the total volume.

Fig. 4 shows the evolution of the bandwidth (bps) with the time for both UCL and UPC but also the packet rate (pps) and the number of different destinations per second exchanging traffic with the campuses (ip/s). The UCL traffic is less important than the UPC one and contacts fewer destinations. The shape of the packet rate (in packets per second) differs between the two campuses. At 11:00 the packet rate suddenly increases at UPC but not the bandwidth meaning that small packets have been sent at a high rate during this period. The short bandwidth peaks that appear at regular intervals in the UCL trace are caused by regular data transfers of well connected scientific servers on the campus. It is worth to note that the working hours start at 8:30 AM (7:30 AM GMT) and finish at 6 PM (5 PM GMT) at UCL. UCL exhibits a 1 hour peak at the beginning of the workday. It is caused by tasks that are done once in the morning by the people, like reading the news and replying in batch to their emails. The UPC is too short in time to observe this kind of phenomenon.

Throughout in the paper, the two traces are illustrated. However, due to space limitations and unless stated otherwise, the UCL trace is considered in the discussions.

4.2 Mapping Lookup Latency

The mapping lookup latency is particularly important in mapping systems as it defines the time required to get a Map-Reply after sending a Map-Request. When an ITR waits for a mapping for an EID, no packet can be sent to this EID. If this delay is too long, the traffic can be severely affected.

Fig. 5 shows the cumulative distribution of the mapping lookup latencies for LISP+ALT and LISP-DHT. The x-axis is the latency expressed in milliseconds and the y-axis is the cumulative distribution function of these latencies. The RTT delay curves give the round-trip-time distribution between the campus and each node involved in the mapping system as given by iPlane. The round-trip-time is the mapping lookup latency that would be observed if the Map-Request was sent directly to the mapping server without using any mapping system overlay.

Fig. 5 shows that the hierarchical deployment used for LISP+ALT achieves better performances than the Chord approach used in LISP-DHT. 95% of the mapping lookup latencies are sub-second with LISP+ALT which is the same order of magnitude than what is observed for DNS in [13]. On the contrary, up to 55% of the mapping lookup latencies are higher than one
Furthermore, the path followed by a Map-Request is the highest EID of the EID prefix it is responsible for. The Chord identifier of a node is the overlay in LISP-DHT only depends on the Chord with the topology. On the opposite, the construction of the paths followed by the queries tend to be congruent thanks to the replication of layer-1 nodes. Moreover, the number of layer-1 nodes with the topology are as close as possible to the querying nodes. Therefore, the exchange of messages in LISP-DHT does not follow the topology and, for example, a Map-Request from a node in Paris to another node in Paris can transit via London, Tokyo and New-York. This lack of locality is a well known problem in P2P and research to tackle this issue is ongoing [27].

Fig. 6 helps to better understand the mapping latency difference between the mapping systems. Fig. 6 shows the cumulative distribution of the number of hops necessary for a Map-Request to be replied. The number of hops is not directly related to the mapping latency but each hop induces latency and it is likely that if the number of hops increases, the total latency increases too.

LISP+ALT in the simulator limits the maximum number of hops to 6 but, in 95% of the cases, this maximum number of hops is observed. In order to have a shorter path, the destination EID should be in one of the /8 prefixes that doesn’t have a more specific part announced separately. This corresponds to the second layer of our simulator’s LISP+ALT hierarchy and a Map-Request would reach the responsible server after 5 hops in this case. If the ITR belongs to one of these /8 networks, we would have a maximum limit of 5 hops, but this is not the case for either of our traces. These networks only represent a small proportion of the Map-Requests. This explains the hop number distribution in LISP+ALT. On the other hand, the maximum number of hops is 17 with LISP-DHT Recursive and 34 with LISP-DHT Iterative. It is even worse to see that LISP-DHT, independently of the querying mode, needs more hops than LISP+ALT to resolve a mapping. This means that more nodes are involved in the resolution of a mapping when using LISP-DHT than when using LISP+ALT.

The major difference between the recursive and the iterative mode in LISP-DHT comes from fact that in the recursive mode, messages are sent from predecessors to predecessors until the Map-Request arrives to the mapping server. With the iterative mode, when a node receives a Map-Request for an EID it does not own, it sends the identifier of the next hop on the overlay path back to the ITR and the ITR sends the Map-Request to that next hop via the overlay, until the Map-Request arrives to the mapping server. Each hop thus requires an exchange with the ITR. The nodes involved in the resolution of the mapping are the same in iterative and recursive mode, it is thus normal for the iterative mode to have more mapping lookup latency than the recursive mode.

It is worse to notice that the RTT and the mapping lookup latency are almost the same for the UCL trace and the UPC one. The difference between UCL and UPC in the LISP-DHT mostly comes from their position on the ring and the position of the mapping resolver on the ring. Therefore, the exchange of messages in LISP-DHT do not follow the topology and, for example, a Map-Request from a node in Paris to another node in Paris can transit via London, Tokyo and New-York. This lack of locality is a well known problem in P2P and research to tackle this issue is ongoing [27].
the ring.

The findings of this study can be leveraged for further research on the LISP mapping systems because there are several properties that can be modelled based on the latency. We provide on the simulator web page a standalone latency modelling tool, which generates latencies following the distributions that we obtained for each mapping systems. The results can help LISP researchers, designers and developers.

In the following of this paper, we will not consider anymore the iterative mode in LISP-DHT as it dramatically increases the mapping lookup latency.

4.3 Node Load

We define node load as the total number of Map-Request messages processed by participants in the mapping system. For a more thorough analysis, we differentiate between the load caused by messages forwarded by the node (transit load) and the load due the messages for which the node is the final destination (mapping load). Mapping load only depends on the observed traffic (distribution of destination EIDs) and is the same for both LISP+ALT and LISP-DHT. Transit load however is mapping system specific and is determined by its routing mechanism.

The average mapping load is 41 for a median of 17. The difference between the average and the median shows that most of the nodes have a small mapping load but few have a higher one. This is confirmed by the first and third quartiles of 4 and 67 respectively and a maximum mapping load of 407.

Due to the architectural differences, the mapping systems under study exhibit different transit load characteristics. Indeed, in the case LISP-DHT all participating nodes are both transit and destination nodes, while in LISP+ALT the nodes from layer 1 and layer 2 are dedicated transit nodes. Figure 7 shows the cumulative distribution of the load in four categories of nodes: LISP+ALT layer 1 and layer 2 transit-only nodes, finger nodes and non-finger nodes of the ITR in LISP-DHT.

For LISP+ALT, the load among the nodes in layer-1 is more unequal than in layer-2. Layer-1 consists of 7 logical nodes for the seven /3 covering the routable prefixes. The most loaded node is 10 times more loaded than the second two most loaded nodes. And those second most loaded nodes are 10 times more loaded than the four least loaded nodes. This inequality comes from the approach we followed to build the LISP+ALT hierarchy. We considered each IP address being queries with the same likelihood and thus uniformly decomposed the IP space with height virtual nodes, each one responsible of a particular /3. In practise, some part of the IP space are more contacted than other. Nevertheless, having a unequal repartition of the load among the different /3 is not an issue, the logical node can be sustained by more or less nodes depending on the expected load, this is already the case in DNS where \texttt{.com} is more important than \texttt{.be}.

Figure 8 shows a different perspective on the transit load in LISP-DHT. Darker lines represent fingers of the ITR at UCL, who was originating the Map-Requests. From the total number of 112,233 nodes participating in the DHT, only 82,637 nodes have routed or received Map-Request messages and are depicted in the figure. We can observe a sharp surge (even on a log scale) after a load value of 1000, that accounts for about 1.8% of the total number of nodes from the DHT. As expected we find many of the ITR’s fingers among these hotspots. Of the 2,365,797 Map-Requests initiated, more than half pass via the last finger and one third via the second last finger. But from the top ten most loaded nodes 7 are not
fingers. Note that a single physical xTR may participate with several different prefixes, but our analysis refers to the virtual Chord nodes.

The last finger of a Chord node is responsible for half of the key space on the Chord ring, so our results are to be expected. We extended our analysis to the last finger of all nodes participating in LISP-DHT to see whether there are hot spots for the global overlay, not just for the UCL traffic. Only 12% of the nodes show up as last finger in some other node’s finger table, and more than half of them in only one node. But there is one node which is last finger for 5.6% of DHT participants, the node responsible for prefix 4.0.0.0/8. This is the first prefix from the IPv4 space present in the iPlane dataset, and because Chord is organised on a ring and wraps around, this node becomes responsible for the EID space of class D and E, because those are not represented in the overlay. For this reason, this prefix must bear the biggest load in the case of LISP-DHT deployment.

The likelihood of a node becoming a popular finger in LISP-DHT is proportional to the IP space for which it is responsible in the Chord ring. This IP space, apart from the size of the prefix advertised depends also on the existence of unallocated blocks just before. E.g., if a /24 prefix is preceded by two unallocated /8 blocks, it has a higher chance to become a hotspot than a /8 with an allocated neighbouring prefix. Note that even if there is no traffic to the unallocated space, the /24 from our example is still a hotspot for transit traffic, because it is present in many finger tables.

The way transit traffic is distributed in LISP-DHT may be desirable for peer-to-peer networks, but is an important disadvantage in a mapping system. Since the transit route is defined only by the Chord routing algorithm, there is no way an entity can control how Map-Request reach its site. This may result in severe performance issues, because the possibility exists that the mapping server of a very popular destination is reached through a small site’s limited capacity node. Malicious attacks, when a middleman selectively or completely drops all Map-Request traffic for some destinations are even worse and require other mechanisms to mitigate. From this point of view, the three-layer LISP+ALT approach is a safer choice, because the upper two layers could be offered as a commercial service and providers can be forced by service level agreements to maintain good quality of service.

4.4 Packet Buffer

Whenever a cache miss occurs, the ITR has the option of either dropping the packet, or storing it in a buffer until the associated mapping has been retrieved. This is an important design decision, and the feasibility of using a buffer depends greatly on its expected size. As described in Sec. 3.2, CoreSim implements a packet buffer and estimates its maximum occupancy at each moment in time.

It is not possible to give exact numbers, because the simulator can only replay the trace, and does not emulate transport or application layer behaviour in the presence of LISP. Specifically, it cannot account for the influence of the mapping latency on the initial TCP three-way-handshake in the case of cache misses. In a real network, after sending a SYN packet the sender would wait for the corresponding SYN + ACK before sending anything else. The trace however may contain the following packets of the TCP stream before the mapping would have returned, since it was captured under normal conditions. When such packets are received, CoreSim adds them to the buffer. This approach thus provides an upper bound for the buffer size.

UDP is an unreliable protocol, without transport layer congestion control. Still, many applications implement their own at the application layer, and these algorithms may be affected by the presence of LISP as well. Using the same approach as in the case of TCP we account for the worst case scenario for UDP too. We believe that providing worst case values for buffer size is useful to get an insight on the feasibility of the buffering approach.

Fig. 9 presents the evolution of the packet buffer occupancy. In the case of the UCL trace we plot it in terms of maximum amount of bytes and packets observed during one minute bins. The median values are 86 packets and 65 KBytes for LISP+ALT, 136 packets and 114 KBytes from LISP-DHT, and even lower for UPC. These low values obtained for a Gigabit link can be explained by the high cache hit ratio of 99.5%. The large spike after 11:00 is caused by a burst of UDP packets, which increases the number of packets in the buffer by several orders of magnitude compared to normal traffic. Interestingly, it isn’t as prominent on the octets plot, where several peaks can be observed.

The results show that LISP+ALT performs better than LISP-DHT, with consistently lower values of buffer occupancy. The amount of time a packet spends in the buffer depends on the mapping latency, so there is a direct relation between these metrics. Since LISP+ALT has lower latency, the buffer occupancy is smaller. The difference is less than an order of magnitude, and cannot be considered as an important drawback of LISP-DHT.

Traffic anomalies (malicious or benign) have an important impact on observed values. In order to have good performance of the packet buffer, a strategy for handling these anomalies should be devised.

Our study also revealed, that in most cases a host only initiates one traffic flow to a given destination within the mapping lookup time window. In the case of LISP+ALT, where lookups are quicker, 94% of the des-
A miss can be observed during an established flow when the mapping has been removed or the ITR in charge of the flow has been changed. Mapping entries can be removed from ITR’s cache if the ITR is running out of memory or simply because no traffic has been seen for a while for the mapping. ITR should be provisioned to support the load and cache replacement algorithms have to remove the mapping with the least probability of being used again. However, no technique exists to avoid mapping removals with the data-driven model.

In large networks with multiple ITRs, flows can be moved from an ITR to another, for traffic engineering reasons or after the outage of the ITR. The impact of such re-routing can be mitigated by replicating the mapping to all the ITRs.

5. RELATED WORK

Several mapping systems have been proposed for LISP [6, 3, 15, 19], but to our knowledge none of these systems have been completely evaluated in details. Iannone and Bonaventure have analysed in [10] the mapping cache used by ITRs. Using trace-driven simulation, they showed that the number of entries in this cache grows with the TTL of the cache and that the mapping system should provide mappings for EID prefixes and not individual EIDs. CoreSim also models the mapping cache on the ITR, but furthermore it models the entire mapping system and provides detailed information about its performance in terms of delay, load on mapping nodes, ... Other researchers [14, 1] have analysed the utilisation of caches or similar techniques to reduce the size of FIB tables on routers.

Luo et al. propose in [16] another LISP mapping systems that relies on the CAN DHT. This mapping system provides mappings for individual EIDs instead of EID prefixes as [6, 3, 15, 19]. Using such flat EIDs is unlikely to scale. A trace-driven evaluation of this mapping system is provided in [16]. The evaluation mainly focuses on the size of the mapping cache and the number of hops through the CAN DHT. CoreSim models delays and is not dedicated to a single mapping system.

6. CONCLUSION

The Locator/Identifier Separation Protocol (LISP) is being developed within the IETF as a way to better scale the Internet routing architecture. A key part of LISP is the mapping system that allows LISP routers to obtain the routing locators associated to remote endpoint identifiers.

Our main contribution is CoreSim, a mixed event/trace-based simulator that allows to efficiently model the operation of LISP mapping systems. For this, CoreSim relies on topology and delay information from iPlane. The
current version of CoreSim contains models of LISP+ALT which is the mapping system that is currently favoured by the IETF LISP working group and LISP-DHT, an adaptation of the Chord Distributed Hash Table. Our simulations with these two mapping systems showed that a hierarchical organisation of the LISP+ALT overlay would produce shorter delays than LISP-DHT. With these short mapping delays, it appears possible to buffer packets that are waiting for mapping replies in LISP routers. Our simulations also show that the load on the intermediate nodes in the mapping system must be taken into account when choosing such a system. With LISP+ALT, the nodes with the higher load are those at the top of the hierarchy. They can thus be dimensioned to sustain this load. However, it appears more difficult to dimension LISP+DHT as the load of a node will depend on its position on the ring and on the position of the nodes sending many requests.

7. REFERENCES