MAP-CACHE SYNCHRONIZATION FOR LOCATOR/ID SPLIT PROTOCOL

Vladimír Veselý

Doctoral Degree Programme (4. VTI), FIT BUT

E-mail: xvesel38@stud.fit.vutbr.cz

Supervised by: Miroslav Švéda

E-mail: sveda@fit.vutbr.cz

Abstract: Nowadays, Locator/ID Split Protocol is discussed as one of the possible solutions for architectural troubles of the current Internet – mobility, multi-homing and partitioning of address space. The main idea of this protocol is separation of localization and identification functions of IP address using map-and-encap principle. This paper characterizes the problem of maintaining up-to-date map-cache between multiple Ingress Tunnel Router devices in high availability scenarios. Moreover as the contribution it proposes possible solution and describes its implementation and resulting impact on topology.

Keywords: Locator/ID Split, LISP, RLOC, map-cache, ITR, synchronization

1. INTRODUCTION

1.1. OPEN PROBLEMS OF THE INTERNET

The IAB meeting in the year 2006 opened discussion on conceptual flaws of the TCP/IP stack and architectural limitations of the Internet. Its aim was to propose possible solutions to the problems that had started to appear as more and more devices were connected to network. It deals with the following issues:

- *Multi-homing and default-free zone routing table growth*: Usually a customer network is a part of autonomous system (AS) of its internet service provider (ISP). A customer should establish its own AS and use BGP for routing information exchange whenever it wants to have redundant Internet connectivity to more ISPs. The default-free zone (DFZ) experiences massive growth for each multi-homing AS, which increases load, CPU and memory requirements to core routers.
- *Mobility*: Under the term device mobility in computer networks we understand the ability of a device to change access connections to different networks preferably without any outage. The TCP/IP stack itself does not have any option to guarantee mobility. Hence, it was additionally amended with initiatives like Mobile IP, Mobile IPv6 or HMIPv6.
- *Device identity*: If a device is connected with only one network interface card (NIC) then IP address could be used to uniquely identify this device and localize it in the network. But if the device is using two or more NICs then each address is only so called point of attachment (PoA) to the target end-network. This implies that: (1) address cannot be used to uniquely identify the device and (2) more than one route through the network exists and is returned during the attempt to localize device.

1.2. INTRODUCING LOCATOR/ID SPLIT PROTOCOL

Locator/ID Split Protocol (LISP) addresses the above problems; among the main goals are to reduce DFZ routing table growth and to allow multi-homing without BGP. Deploying LISP does not cause any reconfiguration to end-stations and it is without any changes to DNS.

Functionality of IP address is currently overloaded: it is used both for localization and for identification of devices. On the one hand, localization is usually affected by ISP and its hierarchical addressing aptness. On the other hand, endpoint addressing (used for identification) obeys intentions of end-network administrator. Negative effect of this overloading is impossibility to build scalable and effective routing architecture in DFZ.

LISP splits those two functions. Nowadays, addresses on Internet form one flat namespace. Instead of it, LISP creates two addressing systems:

- **Routing Locators (RLOC)** space – Its addresses serve as network locators, or to put them differently: those addresses are PoAs of target end-network.
- **Endpoint Identifiers (EID)** space (a.k.a. LISP site) – Each address represents unique address of one device, which identifies it among others.

Also Non-LISP space exists for those parts of Internet that do not understand LISP or do not want to adopt LISP. Special routers running LISP implementation are needed to facilitate communication between RLOC, EID and Non-LISP spaces.

LISP communication uses map-and-encap principle first introduced in ENCAPS [1]. If a packet leaves from EID space to RLOC space then border router performs mapping of destination EID address to destination RLOC address. The packet is then wrapped with LISP + UDP headers and encapsulated with a new outer IP header. Subsequently, if packet reaches LISP site then on receiving border router packet is decapsulated (outer IP header is removed and LISP + UDP headers stripped off).

The LISP mapping system is analogous to DNS. It benefits the same pull model where information is available in distributed hierarchical database. Devices query this database only when they need particular information, which they can store in local cache (called **map cache**). Data-driven query against LISP mapping system is the resolving process of target EID address to RLOC address. This kind of approach makes LISP mapping system massively scalable.

All headstone components of LISP are summarized in the following list:

- **Ingress Tunnel Router (ITR)** – ITR is an ingress point to the RLOC space. It performs encapsulation of data traffic leaving LISP site. It maintains own map cache of EID-to-RLOC mappings by generating queries to the mapping system.
- **Egress Tunnel Router (ETR)** – ETR is an egress point from the RLOC space. It performs decapsulation of data traffic when it is entering the LISP site. ETR advertises available RLOCs to the mapping system for each EID prefix that is responsible for. ETR responds to mapping request.
- **Map Resolver (MR)** – Each ITR router uses configured MR that delegates ITR's map requests to appropriate MS with requested mapping information.
- **Map Server (MS)** – Each ETR router has preconfigured MS which gathers mapping advertisements and maintains mapping database (what EIDs are bond with target ETR's RLOCs). Also MS mediates answers to the mapping request; either by delegating requests to suitable ETR, or by replying directly instead of ETR.

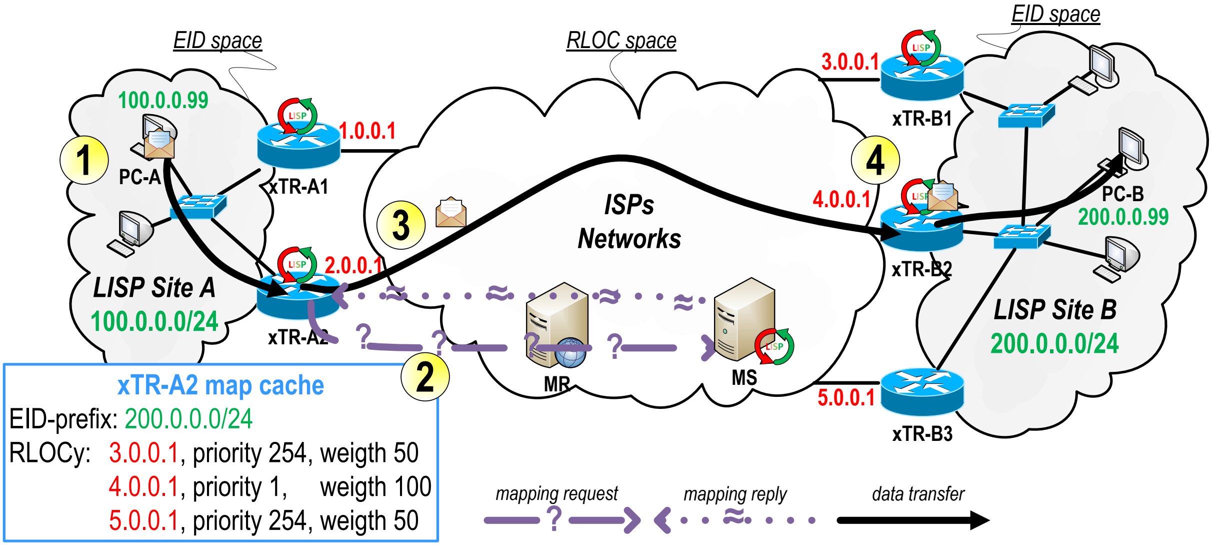
ITR and ETR functionality is dual to each other and usually performed by same device, so called **xTR**. Whole LISP architecture is depicted on the Figure 1 as simple unicast transfer scenario with following description:

(1) Two computers (one in the *LISP Site A* and another in *B*) want to communicate with each other. Thus, computer *PC-A* initiates data transfer.

(2) Packet arrives on *xTR-A2* where it should be encapsulated. ITR does not have relevant EID-to-RLOC mapping. Hence, it queries mapping system which consists of asking map resolver *MR* and map server *MS* with *LISP Map-Request* message + obtaining solicited response to this query called *LISP Map-Reply* message. Subsequently *xTR-A2* receives mapping reply and populates its map cache with up to date information.

(3) ITR *xTR-A2* choses the RLOC with the best parameters (the lowest priority). Then it encapsulates original packet from *PC-A* with the new LISP + outer IP headers. Following next it forwards the packet through outgoing interface.

(4) ETR *xTR-B2* receives the packet. It performs decapsulation and sends original packet to end-station (computer *PC-B*).



xTR-A2 map cache
EID-prefix: 200.0.0.0/24
RLOCy: 3.0.0.1, priority 254, weigth 50
4.0.0.1, priority 1, weigth 100
5.0.0.1, priority 254, weigth 50

Figure 1: LISP architecture and demonstration unicast transfer

The whole process repeats – steps (1), (2), (3) and (4) – upon communication answer from *PC-B* because also *xTR-B** needs to populate its map cache. Until map caches contain valid EID-to-RLOC mappings only steps (1), (3) and (4) take place during LISP map-and-encap procedure.

Detail signalization process of LISP is unfortunately behind the scope of this paper, but all relevant information could be found in recently standardized RFCs – namely RFC 6830, RFC 6832 and RFC 6833.

2. PROBLEM STATEMENT AND MOTIVATION

The ITR's map cache records creation is driven by LISP data traffic. The performance of encapsulation process at ITR depends on the fact whether its map cache contains EID-to-RLOC record or not. In the second case all affected data flows incoming to ITR are discarded until appropriate mapping is discovered by signalization to LISP mapping system (generating *LISP Map-Request/Reply* message exchange). This behavior is similar to ARP throttling [2].

However, link and device failures might occur and affect ITR capabilities to route and forward LISP traffic. None communication and packet loss is allowed in data centers with mission critical traffic. When using LISP in such scenarios, high availability of ITRs is usually done by devices running first-hop redundancy protocols (i.e. HSRP, GLBP) and load-balancing traffic of user – just like *LISP Site A* and its multiple ITR routers *xTR-A1/2* on Figure 1. Nevertheless if LISP data traffic is load-balanced then ITR's map caches are different because of EID-to-RLOC mappings are created on-demand. Imagine that the *xTR-A1* goes down then the remaining ITR *xTR-A2* must update its map cache for additional LISP data traffic. This situation could lead to initial packet loss when the mapping system is being queried for a new mapping.

The main motivation behind our research is that currently LISP protocol itself and no existing LISP control plane implementation do not have any option how to address this issue.

3. CONTRIBUTION

This paper deals with previously described problem and this section introduces our contribution. We have decided to implement the synchronization of map caches that uses independent TCP transfers among predefined **synchronization set (SS)** of ITRs. This approach guarantees that remaining devices could forward rerouted LISP data traffic without packet loss or interruption in case of any ITR failure. Basically, any solicited *LISP Map-Reply* received by ITR triggers the synchronization process. Each record in the map cache is equipped by a time-to-live (TTL) parameter. TTL expresses for how long the record is considered to be valid and is used for encapsulation. Map caches must maintain the same TTL on shared records; otherwise a loss of synchronization might occur (on some ITRs, identical records could expire because of no demand).

We have implemented two modes of synchronization behavior:

1) *Naïve*: During synchronization the whole content of map cache is transferred to SS. All mappings are then updated according to the new content and TTLs are reset. This approach works fine but obviously introduces significant transfer overhead.
2) *Smart*: We created following policy for mapping record expiration based on propositions in [3]. When TTL expires the ITR must verify its usage in last minute. If the mapping record has not been used then it is removed from the cache. Otherwise, its state is refreshed by generating *LISP Map-Request* and the result (mapping update) of each *LISP Map-Reply* response is then transferred to other members of SS.

We implemented previous behavior as an extension of the existing OpenLISP [4] `mapd` daemon. The map cache is exported to a file and watched by our program. The whole (naïve mode) or just difference (smart mode) is transferred via the TCP connection(s) to preconfigured set of IP addresses (members of SS) upon change of file event. Then destination file is updated and all changes are integrated to `mapd`.

4. TESTING AND RESULTS

We have joined LISP Beta-Network [5] to be able to test LISP in non-lab environment as the first site in Czech Republic ever. We were assigned with EID block 153.16.48.112/28.

Figure 2 shows our testing topology which should simulate high-availability scenario – used RLOCs are configured for LISP 50:50 ingress load-balancing. We virtualize all hosts using VirtualBox 4.2.6 on the network transparent system. Border xTRs *ITR1* and *ITR2* are running FreeBSD 8.2 with our add-on and altered OpenLISP implementation. Between the same devices also VRRP runs to guarantee gateway redundancy to user end-stations. Those machines are running OpenSUSE 12.2 and generate scripted ICMP traffic into LISP Beta-Network – *USR1* is periodically pinging 20 destinations with single RLOC, *USR2* another 10 destinations. Default TTL is 24 hours that is rather long to consider RLOC reachability as appropriate. It is wise to configure shorter TTL to keep records up-to-date. Lower TTL yields: (1) smaller map cache size because unused mappings expired sooner, (2) more synchronization triggers during renewal of old records, (3) additional transfer overhead. We use 10 minutes long TTL as compromise guaranteeing up-to-date RLOC reachability and also good performance.

We have conducted two tests (named *Test 1* and *Test 2*), each one with two variants when map cache synchronization is disabled and enabled. The difference between tests is in configuration of default gateway. For *Test 1* all end-stations use *ITR1* as the master gateway. In *Test 2* traffic load-balances between *ITR1* and *ITR2* using VLANs – *ITR1* is the gateway for *USR1* and *ITR2* for *USR2*. Each test is scheduled with *ITR1* failure (accomplished with interface shutdown) after which VRRP takes place and *ITR2* is elected as the new default-gateway – for both *USR1* and *USR2* in case of *Test 1*; for *USR1* in case of *Test 2*.

The results are summarized in the table below. For each variant we measure number of map cache records (columns "*ITR1/2* MC" where the value means <total records> / <records on-demanded by

USR1> / <records on-demanded by *USR2*>) and number of *ITR2* cache misses during TTL period (column "*ITR2* misses") when default-gateway switchover occur.

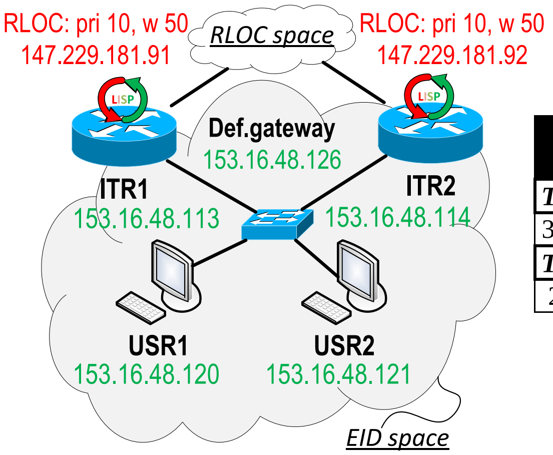


Figure 2: Testing topology

ITR1 MC	*ITR2* MC	*ITR2* misses	*ITR1* MC	*ITR2* MC	*ITR2* misses
***Test 1* w/o synchronization**			***Test 1* with synchronization**		
30/20/10	15/10/5	15	30/20/10	30/20/10	0
***Test 2* w/o synchronization**			***Test 2* with synchronization**		
25/20/5	20/10/10	5	25/20/10	25/20/10	0

Table 1: Comparison of impact on topology using ITR map cache synchronization

Table 1 reveals that in case of *ITR1* failure without map synchronization turned on *ITR2* experiences map cache misses that are followed by ICMP packet drops for affected flows – that is because for destination EIDs map cache is missing records. Map cache synchronization comes with increase of overall cache size, but this cost is relatively small comparing to prevention of unnecessary data loss due to cache misses for the traffic that already is mapped by someone in SS. Hence, ITR's map cache synchronization proves itself as usable technique how to improve data delivery in high-availability mission critical environments.

5. CONCLUSION AND FUTURE WORK

In this paper we provide basic motivation for LISP existence and inform about its architecture components. We provide implementation details of our software contribution capable of transfers of map caches among set of ITRs. We discuss the results and impact of deployed solution, which shows that synchronized map caches of ITRs reduce data loss.

Our next step is to create OMNeT++ simulation models for a LISP testbed that would be independent on any LISP implementation. Moreover, we expect that our rather simple synchronization approach is not suitable for scenarios with many ITRs or for map caches that are quiet large (thousands of records). Hence, for those use-cases we plan to develop more versatile synchronization technique based on the epidemic algorithms for replicated database maintenance.

This work was partially supported by the BUT FIT grant FIT-S-11-1 and the research plan MSM 0021630528.

REFERENCES

[1] Hinden, R. *New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG*. IETF, http://tools.ietf.org/html/rfc1955, June 1996.

[2] Froom, R., Frahim, E., and Sivasubramanian, B. *CCNP Self-Study: Understanding and Configuring Multilayer Switching*. Cisco Press, http://www.ciscopress.com/articles/article.asp?p=425816&seqNum=2, 2005.

[3] Saucez, D., Kim, J., Iannone, L., Bonaventure, O., and Filsfils, C. A Local Approach to Fast Failure Recovery of LISP Ingress Tunnel Routers. (2012), IFIP Networking 2012.

[4] Iannone, L. *The OpenLISP Project*. http://www.openlisp.org/, [online], October 2011.

[5] LISP4.NET/LISP6.NET. *LISP BetaNetwork*. http://www.lisp4.net/beta-network/, [online].