LISP Mapping Resolution Impacts on Initiating Bidirectional End-to-End Communications

Motoyuki OHMORI 1,*, Koji OKAMURA 2, Kohei Hayakawa 3 and Fuminori TANIZAKI 4

1 Chikushi Jogakuen University / 2-12-1, Ishizaka, Dazaifu, Fukuoka, Japan.  
2 Kyushu University / 6-10-1, Hakozaki, Higashi-ku, Fukuoka, Japan.  
3 Cisco Systems G.K. / 3-5-36, Miyahara, Yodogawa-ku, Osaka, Japan.  
4 NTT West Corporation / 3-2-28 Hakataekihigashi, Hakata-ku, Fukuoka, Japan.

E-Mails: ohmori@chikushi-u.ac.jp; oka@ec.kyushu-u.ac.jp; khayakaw@cisco.com; fuminori.tanizaki@fukuoka.ntt-west.jp

* Tel.: +81-92-925-9944; Fax: +81-92-925-3153  
http://dx.doi.org/10.7125/APAN.32.8 ISSN 2227-3026

Abstract: In order to solve scalability problems of routing and addressing architectures of the current Internet, Locator/ID Separation Protocol (LISP) has been proposed. LISP requires mapping resolutions between a locator and an ID and such resolutions may impact on communications. In this paper, we empirically show its impact on initiating end-to-end bidirectional communications based upon actual LISP environment. We then discuss solutions to overcome impacts and imply that Data Probe may be a solution by applying it only to DNS queries, responses, TCP SYN and SYN+ACK.

Keywords: future Internet; LISP; mapping resolution.

1. Introduction

According to rapid growth of the Internet, it has been recognized that the current Internet has architectural flaws of routing and addressing, and is facing scalability problems. Indeed, the number of routes has explosively grown and reached 350,000 [1]. This large number of routes may burden routers in Default Free Zone (DFZ) where all routers have full routes without default route. Hence, it has been considered that the future Internet should reduce the number of routes in DFZ and it can be done by separating an identifier and a locator of an end node [2].

To this end, Locator/ID Separation Protocol (LISP) [3] has been proposed. As its name itself represents, LISP separates an identifier and a locator of an end node. This separation enables
routing packets only with locators between border routers when packets traverse interdomains. Therefore, LISP can reduce the number of routes in intermediate routers in their routing domain such as an Autonomous System (AS) because intermediate routers must have only their own routes in their AS. On the other hand, LISP requires the scheme to map an end node identifier to its locator [4]. Such mapping and its resolutions may impact on bidirectional end-to-end communications because there may be a delay or other side effects to resolve their mapping. These impacts are still under discussions and unclear.

In this paper, we clarify impacts of mapping resolutions of LISP on initiating bidirectional end-to-end communications. We especially focus on impacts to establish a Transmission Control Protocol (TCP) connection after an end node queries a Domain Name System (DNS) server in order to resolve an end node identifier for a Fully Qualified Domain Name (FQDN). To this end, we measure a delay to initiate communications between end nodes on actual LISP environment. We then analyze their results and show that LISP mapping resolutions impacts on initiating communications. According to our analyses, we discuss methods to overcome these impacts.

The rest of this paper is organized as follows. We introduce the overview of LISP operation. We then experiment that end nodes establish connections in LISP networks regarding LISP mapping resolutions. We then analyze their results of initiating communications and clarify impacts of LISP mapping resolution. We then discuss possible way to overcome its impact. Finally, we refer to related works and conclude this paper.

2. LISP

LISP [3, 4] is a simple protocol to establish a unidirectional IP-over-UDP tunnel between LISP sites (e.g. LISP capable ASes) in order to separate IP addresses into two kinds of Routing Locator (RLOC) and Endpoint Identifier (EID). One of main objectives of LISP is to archive its separation with no modification to existing protocol stacks of end nodes. Therefore, end nodes always use only EIDs to communicate. On the other hand, border routers of LISP sites encapsulate/decapsulate packets with RLOCs when packets traverse interdomains; all packets are routed only with RLOCs outside sites. Therefore, routing information of only RLOCs is necessary for interdomain packet routing.
In order to overview LISP operation, let us take an example as depicted in figure 1. In figure 1, EID$_A$ and EID$_B$ are used in site A and B, respectively. RLOC$_A$ and RLOC$_B$ are then assigned to router A and B, respectively. In order to map each EID and RLOC, LISP introduces mapping server that maintains a map between EIDs and RLOCs in mapping database.

Let us assume that node A sends a packet to node B. In this case, a packet reaches at node B as follows:

1. Node A sends a packet with EID$_A$ and EID$_B$ as a source and destination IP address, respectively.
2. A packet reaches router A, which is called Ingress Tunnel Router (ITR).
3. Router A queries mapping server a RLOC for EID$_B$.
4. A mapping server resolves RLOC$_B$ from EID$_B$ and replies to router A.
5. Router A then encapsulates a packet with RLOC$_A$ and RLOC$_B$ as a source and destination IP address, respectively.
6. A packet reaches router B, which is called Egress Tunnel Router (ETR).
7. Router B then decapsulates a packet and forwards it toward node B.

As described above, LISP needs mapping resolutions between EIDs and RLOCs. In addition, a tunnel established described above is unidirectional; another tunnel should be separately established in case of bidirectional communications. Therefore, bidirectional communications require LISP mapping resolutions at least twice. Such resolutions may cause a delay or side effect against bidirectional communications. Note that successive packets are forwarded without mapping resolutions once mapping between RIDs and RLOCs are resolved.

3. Experiments to Initiate End-to-End Communications with LISP

3.1. Experiment Environment

We establish LISP site A and B in universities as depicted in figure 2. Universities are connected via Science Information Network (SINET) in Japan and there are four intermediate routers between LISP router A and B. Under this network configuration, a mapping server is co-located in router B. A full DNS resolver for node A is co-located in node B. A full DNS resolver for node A then has a DNS A record for the FQDN of node B in its DNS database. Note that DNS resolution is here done by UDP not TCP throughout experiments. Table 1 then shows specifications of equipments in figure 2. As shown in table 1, we use three types of nodes as node A in these experiments.

Under this environment, node A initiates to communicate with node B. We here use Secure Shell (SSH) as an application to measure a delay of an establishment to simplify measurements because a SSH server autonomously sends a first data as soon as a TCP connection is established. In figure 2, node A and B act as a SSH client and server, respectively.
Figure 2. Experiment environment to initiate end-to-end communications.

Table 1. Equipment specifications.

<table>
<thead>
<tr>
<th>Node</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node A</td>
<td>Intel Xeon 2.8GHz, 2GB memory, Fedora core 9 (32bits)</td>
</tr>
<tr>
<td></td>
<td>Intel Xeon 2.4GHz, 2.5GB memory, NetBSD 5.99.22 (32bits)</td>
</tr>
<tr>
<td></td>
<td>Intel core i5 2.4GHz, 4GB memory, Windows 7 service pack 1 (32bits)</td>
</tr>
<tr>
<td>Node B</td>
<td>Intel Xeon 3GHz, 2GB memory, NetBSD 3.99.7 (32bits), bind 9.8.0p2</td>
</tr>
<tr>
<td>Router A/B</td>
<td>Cisco ISR2951, IOS Version 15.1(4)M, RELEASE SOFTWARE (fc1)</td>
</tr>
</tbody>
</table>

3.2. Delays to Establish a TCP Connection with LISP

We measure delays between when node A initiates to communicate with node B from a DNS resolution and when node A establishes TCP connections. We here consider two scenarios:

a) A stub resolver in node A has a DNS cache for node B.

b) A stub resolver in node A does not have a DNS cache for node B.

Throughout these experiments, we capture packets by tcpdump [6] at node A. According to time stamps in captured packets, we calculate durations between sent or received packets. We also monitor packets at node B, router A and B in order to realize protocol sequences. We then measure 10 times for each experiment and regards averages as results. In addition, we clear mapping caches in router A and B before experimenting with each scenario and node so that we can clarify impacts caused by first mapping resolutions. Regarding the scenario a), we measure duration as a delay between when node A sends a first DNS query and when node A receives a first TCP data segment. As shown in table 2, node A of Windows successfully establishes a TCP connection within 2.04 seconds of a total delay. In this case, the protocol sequence is as depicted in figure 3 (a). Interestingly, first DNS query from node A and DNS response from node B are dropped at router A and B, respectively. That is, each first packet is used only for its mapping resolutions. Generally speaking, a DNS full resolver does not autonomously retransmit a DNS response. Hence, a DNS stub resolver should retransmit a DNS query until a DNS response is
received. We can then say that at least three DNS queries are necessary so that node A can receive a DNS response. By breaking down its protocol sequence, it also appears that it takes 2.02 seconds from 2.04 seconds of a total delay that node A receives a DNS response from node B after sending a first DNS query. As described above, retransmission intervals of DNS queries may be the important factors for a delay of a connection establishment. On the other hand, surprisingly, node A of Linux or NetBSD can never establish a TCP connection and their connection establishment times out during DNS resolutions. By breaking down its protocol sequence based upon captured packets, it appears that Linux and NetBSD send only two DNS queries. As described above, at least three DNS queries are necessary. Therefore, Linux and NetBSD can never establish TCP connection during DNS resolutions.

Table 2. Delay to establish a TCP connection.

<table>
<thead>
<tr>
<th>OS</th>
<th>Delay without a DNS cache (sec.)</th>
<th>Delay with a DNS cache (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>2.04</td>
<td>6.01</td>
</tr>
<tr>
<td>Linux</td>
<td>N/A</td>
<td>6.01</td>
</tr>
<tr>
<td>NetBSD</td>
<td>N/A</td>
<td>9.00</td>
</tr>
</tbody>
</table>

Figure 3. Protocol sequence to establish TCP connection, (a) Without a DNS cache. (b) With a DNS cache.

Regarding the scenario b), we measure the duration as a delay between when node A sends a first TCP SYN packet and when node A receives a first TCP data segment. As shown in table 2, node A of Windows, Linux and NetBSD successfully established TCP connection. Interestingly, node A of Windows took longer delay than one of the scenario a) even though any DNS query and response were not exchanged between node A and B. Its protocol sequence is then as depicted in figure 3 (b). As shown in figure 3 (b), first TCP SYN and SYN+ACK are dropped at
router A and B, respectively. These behaviors are same as seen in the scenario a). However, interestingly, node A sends only two TCP SYN in this scenario b) while node A sends three DNS queries in the scenario a). These may result from a TCP behavior of each node. That is, TCP retransmission intervals may be longer than ones of DNS query retransmissions and node B autonomously retransmits TCP SYN+ACK before node A retransmits a third TCP SYN. We can then say that TCP retransmission intervals are important factors for a delay of a connection establishment.

3.3. Existing End Node Behaviors

In order to clarify causes of delays shown in section 3.2, we here analyze existing end node behaviors. We focus on behaviors of a DNS query retransmission and a TCP SYN retransmission of Windows, Linux and NetBSD. Throughout this experiment, we capture packets that each node sends and calculate durations. We experiment 10 times for each experiment and averages of these results are shown in table 3. Regarding a DNS query retransmission, the maximum number of DNS query transmissions of Linux and NetBSD is 2 and their query retransmission interval is about 5.00 second as shown in table 3. In LISP, first DNS query and response are dropped as depicted in figure 2 (a) and at least three DNS queries are required as described in section 3.2. Therefore, Linux and NetBSD cannot establish TCP connection when they initiate communications by DNS resolutions. On the other hand, the maximum number of DNS queries of Windows is 5 and their query retransmission intervals are 1.00, 1.00, 2.00 and 4.00 when the number of retransmissions is 1, 2, 3 and 4, respectively; Windows sends three DNS queries within about 2 seconds. Therefore, Windows can receive a DNS response within about 2 second then establish immediately after a DNS response as shown in section 3.2.

Table 3. Existing end nodes behaviors.

<table>
<thead>
<tr>
<th></th>
<th>DNS query retransmission</th>
<th>TCP SYN retransmission</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max #</td>
<td>Interval (sec.)</td>
</tr>
<tr>
<td>Windows</td>
<td>5</td>
<td>1.00 (1,2), 2.00 (3), 4.00 (4)</td>
</tr>
<tr>
<td>Linux</td>
<td>2</td>
<td>5.00</td>
</tr>
<tr>
<td>NetBSD</td>
<td>2</td>
<td>5.01</td>
</tr>
</tbody>
</table>

Regarding a TCP SYN retransmission, the maximum number of TCP SYN sent and their interval are shown in table 3. In addition to table 3, an initial retransmission interval of TCP SYN+ACK of NetBSD 3.99.2, which is used as node B, is 3.00 seconds while its retransmission interval of TCP SYN is 6.00 seconds. Note that these intervals represent for first initial retransmission intervals and successive retransmission intervals are exponentially backed-off. In LISP,
first TCP SYN and SYN+ACK are dropped as depicted figure 3 (b). However, second TCP SYN+ACK is retransmitted at an interval of 3.00 seconds before third TCP SYN is retransmitted at intervals of 6.00, 6.00 and 12.00 seconds for Windows, Linux and NetBSD, respectively. Therefore, only two TCP SYN are required to establish a TCP connection. We can then say that a delay to establish a TCP connection in LISP comprises of a delay to retransmit second TCP SYN and second TCP SYN+ACK.

4. Discussions to Preclude LISP Mapping Resolution Impacts

We here present and discuss a method to overcome issues clarified in section 3. As described in section 3, DNS resolutions may not work especially on NetBSD and Linux. This is caused by the maximum number of DNS query retransmissions. In addition, we have presented delays to establish TCP connection without DNS resolutions on Windows, Linux and NetBSD. These delays may be longer than ones of end nodes on the current Internet and should be shortened.

In order to solve these issues, we can consider utilizing Data Probe of LISP Alternative Topology (LISP+ALT) [5]. Data Probe operates as follows. When an ITR receives a first packet destined to an EID that an ITR does not cache corresponding RLOC, an ITR forward its packet to a mapping server. A mapping server forwards a packet to a corresponding ETR because a mapping server maintains all mappings between EIDs and RLOCs. A mapping server then notifies an ITR of a corresponding RLOC for an EID. As described above, Data Probe avoids a first packet dropped. However, Data Probe is strongly discouraged in [3-5] because it may burden much loads on a mapping server. This may be true when all first packets are forwarded. However, it may still be possible to forward only DNS queries, responses, TCP SYN and SYN+ACK because these packets impacts on much delays and their traffic may be less than ones of successive data packets. This method may preclude all retransmissions from DNS resolutions and establishments of TCP connections even though there may be delay to forward first packets. Therefore, this method may be able to solve all issues described in section 3.

5. Related Works

There have been several proposals for LISP mapping system [5-8]. Quoitin et al. then evaluates LISP regarding reductions of routing information and multi-path possibilities for redundancies by simulation based upon traffic traces [9]. However, these studies do not present actual impacts of LISP on bidirectional end-to-end communication including DNS resolutions.

6. Concluding Remarks

In this paper, we have empirically shown that mapping resolutions between EIDs and RLOCs in LISP impacts on initiating bidirectional end-to-end communications. Mapping resolutions
have impacted especially on DNS resolutions and this caused that Linux and NetBSD could not establish TCP connections when they initiated communications by DNS resolution. In addition, mapping resolutions also caused a delay more than 6 seconds to establish TCP connections where DNS resolutions were cached. We have then discussed above issues and implied that such issues could be solved by applying Data Probe only to a DNS query, response, TCP SYN and TCP SYN+ACK.

Acknowledgements

We deeply appreciate Cisco Systems, Inc. to rent their equipments and advise us to establish LISP environments. This work was also supported in part under Collaborative Research Agreement for Study on collaborative research platform for Asia-Pacific international research network and its application between the Korea Institute of Science & Technology Information (KISTI) and Keio Research Institute at SFC, Keio University (KRIS).

References


© 2011 by the authors; licensee Asia Pacific Advanced Network. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).