Analyses on First Packet Drops of LISP in End-to-End Bidirectional Communications

Motoyuki OHMORI
Chikushi Jogakuen University
ohmori@chikushi-u.ac.jp

Koji Okamura
Kyushu University
oka@ec.kyushu-u.ac.jp

Kohei HAYAKAWA
Cisco Systems G.K.
khayakaw@cisco.com

Fuminori TANIZAKI
NTT West Corporation
fuminori.tanizaki@fukuoka.ntt-west.jp

ABSTRACT

In this paper, we empirically present impacts of first packet drop of Locator/ID Separation Protocol (LISP) on initiating end-to-end bidirectional communications. In actual LISP networks, we measure delays that several types of existing end nodes initiate communications including DNS resolutions and establishments of TCP connections. The results of these measurements show that delays caused by LISP first packet drop mainly comprise behaviors of retransmitting DNS queries and TCP SYNs. It also appears that delays to establish TCP connection increase when a DNS entry for a corresponding node is cached. In addition, the results show that some existing nodes cannot even establish TCP connections where first packets from an initiating and a corresponding node are dropped. We then present that these existing nodes can establish TCP connections and delays caused by first packet drop can be reduced by half when piggyback resolution is employed. We also discuss other possible solutions to overcome the delays.

Keywords

LISP, mapping resolution, first packet drop.

1. INTRODUCTION

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

In order to achieve this separation, Locator/ID Separation Protocol (LISP)[3] has been proposed. 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 resolution may impact on end-to-end bidirectional communications because there may be a delay to resolve their mapping. Especially, a first packet drop problem is well known in LISP [3]; a first packet is used for mapping resolution and its packet is dropped. Its impact on end-to-end bidirectional communications is unclear and under discussion.

In this paper, we empirically present impacts of mapping resolutions of LISP on initiating end-to-end bidirectional 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). We measure then 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 and the first packet drop problem. 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 ways to overcome its impact. Finally, we refer to related works and conclude this paper.

2. LISP

2.1 LISP Overview

LISP 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 and/or decapsulate packets with RLOCs when packets traverse interdomains; all packets are routed only

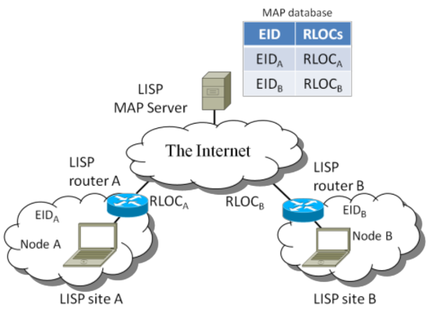

Figure 1 LLISP Overview

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 a mapping server that maintains a map between EIDs and RLOCs in a 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$_B$ while router A drops the received packet.
4. A mapping server resolves RLOC$_B$ from EID$_B$ and replies to router A.
5. Router A then creates a cache entry for a mapping of RLOC$_B$ and EID$_B$.
6. Node A sends another packet destined for EID$_A$.
7. Router A receives the packet and resolves RLOC$_B$ for EID$_B$ from the cache.
8. Router A encapsulates a packet with RLOC$_A$ and RLOC$_B$ as a source and destination IP address, respectively.
9. A packet reaches router B, which is called Egress Tunnel Router (ETR).
10. Router B then decapsulates a packet and forwards it toward node B.

As described above, LISP needs mapping resolutions between EIDs and RLOCs. An established tunnel is then unidirectional; another tunnel should be separately established in case of bidirectional communications. Therefore, bidirectional communications require LISP mapping resolutions at least twice.

2.2 First Packet Drop Problem

As described in section 2.1, an ITR drops a first packet when an ITR does not have a cache entry for a corresponding EID. This problem dropping packets for mapping resolutions is called *first packet drop*. This causes an end node to retransmit a packet; LISP relies on retransmissions of end node. This retransmission causes a delay or a side effect to initiate communications and they may be related to end node behaviors.

Regarding end-to-end bidirectional communications, a delay or impact that a first packet drop causes is more serious. In case of end-to-end bidirectional communications, at least two mapping resolutions are necessary because a tunnel in LISP is unidirectional. That is, each first packet from an initiating and corresponding node is dropped. Hence, a delay to establish end-to-end bidirectional communication may increase due to first packet drops.

2.3 Mitigating First Packet Drop Impacts

2.3.1 Data Probe

It would be apparently better to avoid first packet drop. To this end, LISP+ALT[4] then defines *Data Probe*. *Data Probe* allows first packet to be forwarded to a corresponding ETR without first packet drop. *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][4][5] because it may burden much loads on a mapping server.

2.3.2 Piggyback Resolution

In order to mitigate impacts caused by first packet dropped especially for bidirectional communications, piggyback resolutions is optionally defined in [3]. Regarding bidirectional communications, two mapping resolutions are necessary as described in section 2.2. Piggyback resolutions reduce the number of resolutions by one; piggyback resolutions establish two directional tunnels when an initiating node transmits first packet. Piggyback resolution operates as follows. When an ITR receives a first packet destined to an EID, an ITR resolves a corresponding RLOC as described in section 2.1. At same time, a mapping server then notifies a corresponding ETR of its resolutions. A corresponding ETR then create a cache for an initiating end node. As described above, piggyback resolution enables to establish two directional tunnels with only one mapping resolution.

3. BEHAVIOR OF EXISTING END NODES

As described in section 2.2, impacts of first packet drops may be related to end node behaviors of retransmissions. In this section, we present existing end node behaviors when these nodes initiate end-to-end bidirectional communications. Existing end nodes usually initiate communications from DNS resolutions. These end nodes may also have DNS caches and these end nodes may try to establish TCP connection without DNS resolutions. Therefore, we here focus on their behaviors of retransmissions of DNS queries and TCP SYN packets, and measure their retransmission intervals.

We use several types of existing end nodes as shown in Table 1. We then use Secure Shell (SSH)[6][7] on each node as an application. Each node initiates to establish a connection with non-existent node so that we can measure retransmission intervals. We then capture packets at each node and then calculate retransmission intervals according their packet traces. Note that retransmission intervals of end nodes are default values without any special configurations.

Table 2 shows the number of retransmissions and their intervals of DNS queries of end nodes. In case of Linux and NetBSD, DNS queries are retransmitted only once. Their retransmission intervals are then 5.00 seconds and 5.01 seconds, respectively. On the other hand, Windows retransmits DNS queries at most 4 times and their retransmission intervals are exponentially backed off as shown in Table 2. As described section 2.2, LISP relies on retransmissions of end nodes for first packet. Thus, the shorter intervals are better for LISP. That is, Windows may face the smallest impacts on DNS resolutions while others may have bigger impacts. In addition, at least two mapping resolutions are necessary for end-to-end bidirectional communications as described in section 2.2. This implies that Linux and NetBSD may fail in DNS resolutions because they retransmit DNS queries only once when piggyback resolution is not employed.

Table 1. Specifications of existing end nodes

CPU	Memory	OS
Intel Xeon 2.8GHz	2GB	Linux kernel 2.6.26.6 (32bits) (Fedora core 9)
Intel Xeon 2.4GHz	2.5GB	NetBSD 5.99.22 (32bits)
Intel core i5 2.4GHz	4GB	Windows 7 service pack 1 (32bits)

Table 2. DNS query retransmission of end nodes

Node	Interval per retransmission (sec.)			
	1	2	3	4
Linux	5.00	N/A	N/A	N/A
NetBSD	5.01	N/A	N/A	N/A
Windows	1.00	1.00	2.00	4.00

Table 3 then shows the number of retransmissions and their intervals of TCP SYN of end nodes. The number of retransmissions of Linux, NetBSD and Windows are 5, 3 and 2, respectively. Regarding retransmission intervals of TCP SYN, their intervals exponentially backed-off as defined in [8]. In case of Linux and Windows, the initial retransmission interval is 3.00 second. On the other hand, one of NetBSD is 6.00 second. NetBSD may be then affected more because of a first packet drop.

Table 3. TCP SYN retransmission of end nodes

Node	Interval per retransmission (sec.)				
	1	2	3	4	5
Linux	3.00	6.00	12.00	24.00	48.00
NetBSD	6.00	12.00	24.00	N/A	N/A
Windows	3.00	6.00	N/A	N/A	N/A

4. INITIATING END-TO-END BIDIRECTIONAL COMMUNICATIONS WITH LISP

In this section, we present a delay or a side effect of first packet drops on actual LISP networks. We measure a delay including DNS resolutions where piggyback resolution is not employed. We then measure a delay where its resolution is employed. Note that we do not here present a delay when *Data Probe* is employed because routers in our LISP network do not support it.

4.1 Measurement Environment

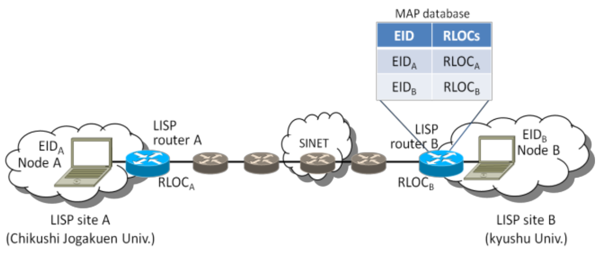

Figure 2 Measurement environment

We establish LISP site A and B in universities as depicted in Figure 2. LISP site A and B are connected via Science Information Network (SINET). There are four intermediate routers between LISP router A and B.

Under this network configuration, a LISP mapping server is co-located in router B. A full DNS resolver for node A is also co-located in node B. This full DNS resolver has a DNS A record for a FQDN of node B. Note that these configurations of a LISP mapping server and a DNS resolver may not be common; in general, LISP mapping server may be separated from LISP routers, ITR and/or ETR. In addition, a DNS resolver for node A may be located in LISP site A and another DNS name server, which has a DNS A record for the FQDN, may be located in not node B but another node in LISP site B. However, we here simplify the configuration and delays shown in later experiments would be less than more realistic environments.

Table 4 then shows each specification of equipments. As shown in Table 4, we use same nodes as shown in Table 1 as node A and respectively measure with each node.

Table 4. Specifications of equipments

Node	Specification
Node A	Same as shown in Table 1
Node B	Intel Xeon 3GHz, 2GB memory, NetBSD 3.99.7 (32bits), bind 9.8.0p2
Router A/B	Cisco ISR2951, IOS Version 15.1(4)M, RELEASE SOFTWARE (fc1)

4.2 Measurement Methodology

In the networks described in section 4.1, we measure a delay to initiate a communications. Throughout measurements, we use SSH as an application. In Figure 2, node A and node B are an initiator and a corresponding node of a SSH communication, respectively. We consider following scenarios:

- a.) No piggyback resolutions
  - b-1.) Without DNS cache
  - b-2.) With DNS cache
- b.) Piggyback resolutions
  - b-1.) Without DNS cache
  - b-2.) With DNS cache

In order to realize protocol sequences, we capture packets at node A, B, and router A, B, respectively. We then calculate each delays that node A transmits or receives each packet after node A sends a first packet according to packet traces captured at node A. We then calculate a total delay between when node A transmits first packet of a DNS query or a TCP SYN and when node A receives a first TCP data segment. In each scenario, we try 10 times and we calculate an average as a delay. In order to realize impacts caused by first packet drop, we clear mapping cache in LISP mapping server and also clear a DNS cache in node A when we try each measurement.

4.3 Delay without Piggyback Resolutions

4.3.1 Delay with no DNS cache

Figure 3 and Table 5 show the protocol sequence and each delay, respectively. As shown in Figure 3, each first packet of a DNS query and response is dropped. As described in 3, Linux and NetBSD core retransmit DNS queries only once. In addition, DNS full resolver on node B does not autonmously retransmit DNS responses to node A. Thus, SSH client on node A here give up establishing connections after second DNS query (first retransmission of DNS query) because node A cannot receive any DNS response. That is, Linux and NetBSD cannot establish connections in this scenario. This means that servers running with Fecora core and NetBSD are affected when LISP is introduced into actual networks. This impact should be precluded and considered.

On the other hand, Windows retransmits DNS queries more than once (i.e. more than two DNS queries in total). Hence, Windows succeeds in establishing a connection with a delay as shown in Table 5. Its total delay to establish a connection is about 2 seconds. The total delay then mainly comprises of a retransmission interval of DNS query, which are about 1 second. We can then say that retransmission intervals of DNS queries of existing end node are important factors that introduce a delay to initiate communications in LISP networks.

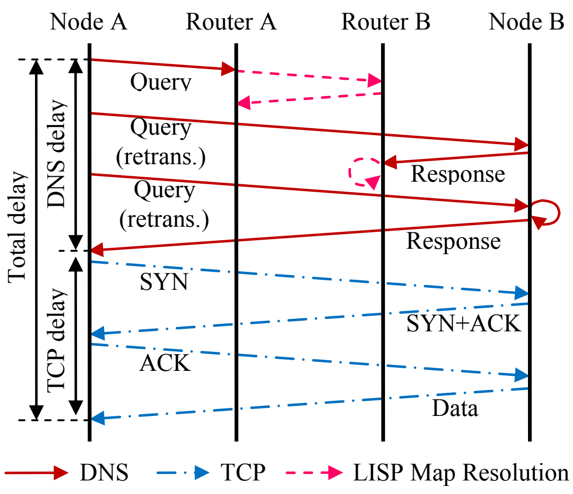

Figure 3 Protocol sequence to establish TCP connection with no DNS cache

Table 5}. Delay with no DNS cache

Packet	Delay (sec.)		
	Linux	NetBSD	Windows
Total	N/A	N/A	2.036306
Query (retrans.)	5.000537	5.00655	1.01
Query (retrans.)	N/A	N/A	1.00
Response	N/A	N/A	0.00230
SYN	N/A	N/A	0.00608
SYN+ACK	N/A	N/A	0.00219
ACK	N/A	N/A	0.00009
Data	N/A	N/A	0.01050

4.3.2 Delay with DNS cache

Figure 4 and Table 6 show the protocol sequence and each delay, respectively. As shown in Figure 4, each first packet of TCP SYN and SYN+ACK is dropped. However, all of Linux, NetBSD and Windows succeed in establishing TCP connections. The total delay to establish connections are about 4, 9 and 6 seconds in Linux, NetBSD, and Windows, respectively. The total delays mainly comprise retransmission intervals of TCP SYN and SYN+ACK. Note that node A retransmits TCP SYN only once (i.e. two TCP SYNs in total) while three DNS queries was necessary in the measurement in section 4.3.1. As shown in Figure 4, node B autonomously retransmits TCP SYN+ACK at an interval of about 3 seconds. That is, retransmission intervals of both of node A and node B are major factors of delays while the retransmission interval of only node A was the major factors in the measurement in section 4.3.1. We can say that retransmission intervals of TCP SYN and TCP SYN+ACK are also important factors that introduce a delay to initiate communications in LISP networks.

In addition, the total delay on Windows in this measurement is larger than one in the measurement in section 4.3.1. This results from the fact that the retransmission intervals of TCP SYN of Windows are larger than one of DNS queries. Thus, we can say that the total delay would be larger when a corresponding DNS entry is cached than when a DNS entry is not cached.

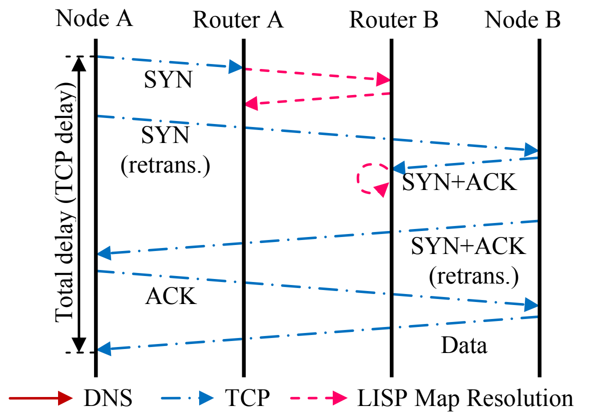

Figure 4 Protocol sequence to establish TCP connection with DNS cache

Table 6. Delay with DNS cache

Packet	**Delay (sec.)**		
	Linux	**NetBSD**	**Windows**
Total	6.006894	9.004188	6.010517
SYN (retrans.)	2.99933	5.99037	3.00490
SYN+ACK (retrans.)	2.99772	3.00241	2.99494
ACK	0.00003	0.00001	0.00013
Data	0.00982	0.01139	0.01055

4.4 Delay with Piggyback Resolutions

4.4.1 Delay with no DNS cache and piggyback resolutions

Figure 5 and Table 7 show the protocol sequence and each delay, respectively. As depicted in Figure 5, only first packet of DNS query from node A is dropped. This results from the fact that a LISP mapping cache is created when router A receives a first DNS query and resolves the mapping between EID_B and $RLOC_B$ via router B. Hence, all of Linux, NetBSD and Windows can here establish connections while Linux and NetBSD could not in section 4.3.1. Hence, we can say that piggyback resolution plays an important role and should be mandatory for Linux and NetBSD while it is defined as optional in the specification [3].

As shown in Table 7, the total delays mainly comprise retransmission intervals of DNS queries of node A. In comparison with the measurement in section 4.3.1, the total delay on Windows decreases by half. This obviously results from the fact that a first packet from node B is not dropped. Hence, the piggyback resolution plays an important role to reduce a delay.

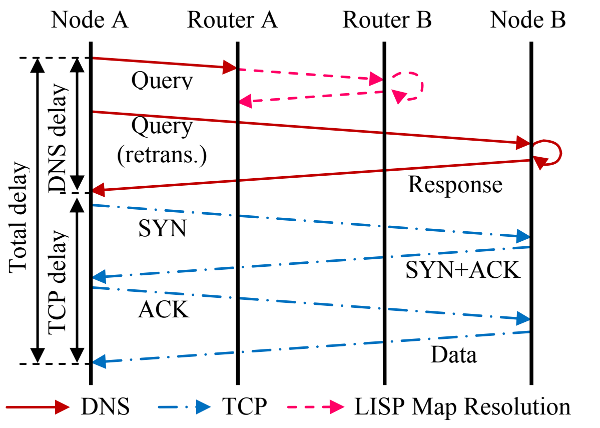

Figure 5 Protocol sequence to establish TCP connection with no DNS cache and piggyback resolution

Table 7. Delay with no DNS cache and piggyback resolution

Packet	**Delay (sec.)**		
	Linux	**NetBSD**	**Windows**
Total	5.014918	5.023592	1.022383
Query (retrans.)	5.00033	5.00743	1.00224
Response	0.00205	0.00253	0.00208
SYN	0.00018	0.00010	0.00618
SYN+ACK	0.00177	0.00240	0.00178
ACK	0.00002	0.00001	0.00009
Data	0.01057	0.01113	0.01002

4.4.2 Delay with DNS cache and piggyback resolutions

Figure 6 and Table 8 show the protocol sequence and each delay, respectively. As depicted in Figure 6, only first packet of TCP SYN from node A is dropped as same as in section 4.4.1. In this measurements, all of Linux, NetBSD and Linux succeed in establishing connections as same as in section 4.4.2.

As shown in Table 8, the total delay mainly comprise retransmission intervals of TCP SYNs of node A while one of TCP SYN+ACK of node B also was a factor as described in section 4.4.2. The total delays are reduced by about 3 seconds, which was a retransmission delay of TCP SYN+ACK of node B. That is, piggyback resolution also plays an important role in this scenario.

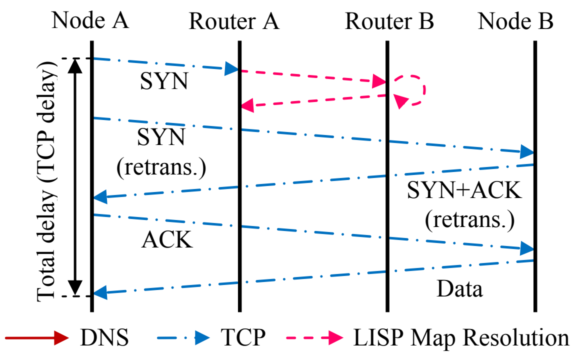

Figure 6 Protocol sequence to establish TCP connection with DNS cache and piggyback resolution

Table 8. Delay with DNS cache and piggyback resolution

Packet	**Delay (sec.)**		
	Linux	**NetBSD**	**Windows**
Total	3.011919	6.010572	3.012793
SYN (retrans.)	2.99930	5.99704	3.00027
SYN+ACK	0.00214	0.00240	0.00186
ACK	0.00002	0.00001	0.00010
Data	0.01045	0.01113	0.01056

5. DISCUSSIONS

In this section, we discuss impacts shown in section 4 on actual services in the Internet. We then discuss possibilities to overcome these impacts. We also refer to how often these impacts may actually occur.

As described in section 4, it appeared that Linux and NetBSD could not even establish connections when they initiate from DNS resolutions. This impact caused by a first packet drop may be undesirable. For example, let us consider mail transfer with Simple Mail Transfer Protocol (SMTP) [9]. When a sending mail server fails to open a connection with a corresponding mail server, a sending mail server will retry after a long delay such as 16 minutes in postfix[10]. It may be controversial if such a long delay is acceptable for mailing services or not. However, a shorter delay would be still preferable. As results shown in section 4, we can preclude such an impact by applying piggyback resolution. Piggyback resolution also improves a delay as described in section 4. Hence, it would be better to employ piggyback resolution if possible.

However, there is a case where piggyback resolution is not applicable. Apparently, piggyback resolution requires ETRs to act as ITRs as well in a LISP site. That is, incoming and outgoing packets for a bidirectional end-to-end communications should go through a same LISP router in a LISP site. Therefore, piggyback resolution cannot be applied when a LISP site has multiple LISP routers and when incoming and outgoing LISP routers are different. In this case, impacts caused by LISP mapping resolutions may be greater as shown section 4. In order to reduce these impacts as much as possible, we can bring two solutions:

1.) Establish symmetric packet path per EID

   This makes piggyback resolutions always possible. In order to achieve this, we can exploit LISP nature. In LISP, a LISP mapping servers provide mappings between EIDs and RLOCs. If a mapping server can always provide a querier with same RLOC for same EIDs, packet paths can be symmetric for the querier. However, this may require a mapping server of more complicated processes such as memorizing a querier.

2.) Locate DNS server outside LISP sites

   As described in section 4, impacts caused by LISP mapping resolutions are worse (i.e. establishments of communications fail) when communications are initiated from DNS resolution over LISP networks. In order to avoid this, we may locate DNS server outside LISP sites. This means that DNS resolutions are done with only RLOCs and there is no mapping resolution. Hence, this avoids at least failures of communication establishments. However, delays to establish communications may be still longer. In addition, this may require an operator in a LISP site of more complicated operations.

These solutions may be effective. However, each solution still has disadvantages and they should be discussed in the future.

In addition, it would be worthy of considering how cache misses occur in LISP mapping resolutions, which is the main focus of this paper. As presented in [11], it has appeared that cache miss rate would be about 0.5% based upon a simulation with actual packet traces. Even at such low cache miss rate, there is still the possibility to introduce longer delays in some services such as mailing services as described above. It would be still better to preclude these delays.

6. RELATED WORKS

There have been several proposals for LISP mapping system [4][12][13]. Quoitin et al. have then evaluated LISP regarding reductions of routing information and multi-path possibilities for redundancies by a simulation based upon traffic traces [14]. On the other hand, Iannone and Bonaventure have also evaluated mapping systems regarding cache sizes, cache hit rates and cache expiry. However, these studies do not consider behaviors of existing end nodes and present actual impacts of LISP on bidirectional end-to-end communication including DNS resolutions.

7. CONCLUDING REMARKS

In this paper, we have empirically presented impacts caused by LISP first packet drop on end-to-end bidirectional communications. Our measurement results have shown that first packet drop would prevent some existing end nodes from establishing communications. We have then shown that this impact would introduce longer delay in some services such as mailing services to deliver mails. In order to preclude this impact, it has been appeared that piggyback resolutions played an important role. However, we have shown that piggyback resolutions might not be applied in some environments. In order to overcome this problem, we have also shown possible solutions. We can finally conclude that more discussions are necessary to avoid LISP first packet drops as much as possible in accordance with existing end node behaviors.

8. ACKNOWLEDGMENTS

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).

9. REFERENCES

[1] Huston, G. The growth of the bgp table - 1994 to present. Available online at: http://bgp.potaroo.net.

[2] Bonaventure, O. Reconsidering the Internet Routing Architecture, Internet draft, 2007. Available online at: http://tools.ietf.org/id/draft-bonaventure-irtf-rrg-rira-00.txt.

[3] Farinacci, D., Fuller, V., Meyer, D. and Lewis, D. Locator/ID Separation Protocol (LISP), Internet draft, Oct. 2011. Available online at: http://tools.ietf.org/id/draft-ietf-lisp-15.txt.

[4] Fuller, V. and Farinacci, D. LISP Map Server. Internet draft, 2011. Available online at: http://tools.ietf.org/id/draft-ietf-lisp-ms-11.txt.

[5] Fuller, V., Farinacci, D., Meyer, D. and Lewis, D. LISP Alternative Topology (LISP+ALT), Internet draft, 2011. Available online at: http://tools.ietf.org/id/draft-ietf-lisp-alt-07.txt.

[6] OpenSSH Project, Available online at: http://www.openssh.com/.

[7] Tatham, S. PUTTY: A Free Telnet/SSH Client. Available on line at: http://www.chiark.greenend.org.uk/~sgtatham/putty/.

[8] Postel, J. Transmission Control Protocol. RFC 793, 1981. Available online at: http://www.ietf.org/rfc/rfc793.txt.

[9] Klensin, J., Simple Mail Transfer Protocol, RFC5321, 2008. Available online at: http://tools.ietf.org/html/rfc5321.

[10] Venema, W. Z. The Postfix Home Page, Available online at http://www.postfix.org/.

[11] Jakab, L., Cabellos, A., Coraş F., Domingo J., Saucez D. and Bonaventure, O. Evaluation of LISP+ALT performance, IETF meeting 75, 2009. Available online at: http://www.ietf.org/proceedings/75/slides/lisp-4.pdf.

[12] Mathy, L. and Iannone, L. LISP-DHT: Towards a DHT to map identifiers onto locators, in Proc. of ACM CoNext, 2008.

[13] Jakab, L., Aparicio, A.C; Coras, F. Saucez, D. and Bonaventure, O. LISP-TREE: A DNS Hierarchy to Support the LISP Mapping System, *IEEE Journals on selected areas in Communications*, Vol. 28, No. 8, 2011, pp. 1332-1342.

[14] Quoitin, B., Iannone, L., Launois, C.D. and Bonaventure, O. Evaluating the Benefits of the Locator/Identifier Separation, in Proc. of ACM MobiArch, 2007.

[15] Iannone, L. and Bonaventure, O. On the Cost of Caching Locator/ID Mappings, in Proc. of ACM CoNext, 2007.