

Network Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: January 13, 2011

L. Iannone  
TU Berlin - Deutsche Telekom  
Laboratories AG  
D. Saucez  
O. Bonaventure  
Universite catholique de Louvain  
July 12, 2010

LISP Map-Versioning  
draft-iannone-lisp-mapping-versioning-02.txt

## Abstract

This document describes the LISP Map-Versioning mechanism. This is mechanism to provide in-packet information about EID-to-RLOC mappings used to encapsulate LISP data packets. The proposed approach is based on associating a version number to EID-to-RLOC mappings and transport such a version number in the LISP specific header of LISP-encapsulated packets. LISP Map-Versioning is particularly useful to inform communicating xTRs about modification of the mappings used to encapsulate packets. Note that, in the LISP encapsulation and in the Map Records, bits used for Map-Versioning can be safely ignored by xTRs that do not support the mechanism.

## Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on January 13, 2011.

Internet-Draft

LISP Map-Versioning

July 2010

## Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">2.</a>	Requirements notation . . . . .	<a href="#">3</a>
<a href="#">3.</a>	EID-to-RLOC Map-Version number . . . . .	<a href="#">4</a>
<a href="#">3.1.</a>	The special Map-Version 0 . . . . .	<a href="#">4</a>
<a href="#">4.</a>	Dealing with Map-Version numbers . . . . .	<a href="#">5</a>
<a href="#">4.1.</a>	Handling Destination Map-Version Number . . . . .	<a href="#">5</a>
<a href="#">4.2.</a>	Handling Source Map-Version Number . . . . .	<a href="#">6</a>
<a href="#">5.</a>	LISP header and Map-Version numbers . . . . .	<a href="#">7</a>
<a href="#">6.</a>	Map Record and Map-Version . . . . .	<a href="#">9</a>
<a href="#">7.</a>	Benefits and case studies for Map-Versioning . . . . .	<a href="#">10</a>
<a href="#">7.1.</a>	Synchronization of different xTRs . . . . .	<a href="#">10</a>
<a href="#">7.2.</a>	Map-Versioning and unidirectional traffic . . . . .	<a href="#">11</a>
<a href="#">7.3.</a>	Map-Versioning and interworking . . . . .	<a href="#">11</a>
<a href="#">7.4.</a>	Graceful RLOC shutdown/withdraw . . . . .	<a href="#">12</a>
<a href="#">7.5.</a>	Map-Version for lightweight LISP implementation . . . . .	<a href="#">12</a>
<a href="#">8.</a>	Incremental deployment and implementation status . . . . .	<a href="#">13</a>
<a href="#">9.</a>	Security Considerations . . . . .	<a href="#">13</a>
<a href="#">9.1.</a>	Map-Versioning against traffic disruption . . . . .	<a href="#">13</a>
<a href="#">9.2.</a>	Map-Versioning against reachability information DoS . . . . .	<a href="#">14</a>
<a href="#">10.</a>	Acknowledgements . . . . .	<a href="#">14</a>
<a href="#">11.</a>	References . . . . .	<a href="#">14</a>
<a href="#">11.1.</a>	Normative References . . . . .	<a href="#">14</a>
<a href="#">11.2.</a>	Informative References . . . . .	<a href="#">15</a>
<a href="#">Appendix A.</a>	Map-Version wrap-around . . . . .	<a href="#">15</a>
	Authors' Addresses . . . . .	<a href="#">16</a>

## 1. Introduction

This document describes the Map-Versioning mechanism used to provide information on changes in the EID-to-RLOC mappings used in the LISP ([\[I-D.ietf-lisp\]](#)) context to perform encapsulation. The mechanism is totally transparent to xTRs not supporting such a functionality. It is not meant to replace any existing LISP mechanism, but rather to complete them providing new functionalities. The basic mechanism is to associate Map-Version numbers to each LISP mapping and transport such a version number in the LISP specific header. When a mapping changes, a new version number is assigned to the updated mapping. A change in an EID-to-RLOC mapping can be a change in the RLOCs set, by adding or removing one or more RLOCs, but it can also be a change in the priority or weight of one or more RLOCs.

When Map-Versioning is used, LISP-encapsulated data packets contain the version number of the mappings used to select the RLOCs in the outer header (both source and destination). These version numbers are encoded in the 24 low-order bits of the first longword of the LISP header and indicated by a specific bit in the flags (first 8 high-order bits of the first longword of the LISP header). Note that not all packets need to carry version numbers.

When an ITR encapsulates a data packet, with a LISP header containing the Map-Versions, it puts in the LISP-specific header two version numbers:

1. The version number assigned to the mapping (contained in the EID-to-RLOC Database) used to select the source RLOC.
2. The version number assigned to the mapping (contained in the EID-to-RLOC Cache) used to select the destination RLOC.

This operation is two-fold. On the one hand it enables the ETR receiving the packet to know if the ITR that sent it is using the latest mapping for the destination EID. If it is not the case the ETR can send to the ITR a Map-Request containing the updated mapping or invoking a Map-Request from the ITR (both cases are already defined in [\[I-D.ietf-lisp\]](#)). In this way the ITR can update its

cache. On the other hand, it enables an xTR receiving such a packet to know if it has in its cache the latest mapping for the source EID (in case of bidirectional traffic). If it is not the case a Map-Request can be send.

## [2.](#) Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

## [3.](#) EID-to-RLLOC Map-Version number

The EID-to-RLLOC Map-Version number consists in an unsigned 12-bits integer. The version number is assigned in a per-mapping fashion, meaning that different mappings will have assigned a different version number, which is also updated independently. An update in the version number (i.e., a newer version) consist in incrementing by one the older version number. [Appendix A](#) contains a rough estimation of the wrap-around time for the Map Version number.

The space of version numbers has a circular order where half of the version numbers is greater than the current Map-Version number and the other half is smaller than current Map-Version number. In a more formal way, assuming we have two version numbers  $V1$  and  $V2$  and that the numbers are expressed on  $N$  bits, the following three cases may happen:

$V1 = V2$  : This is the exact match case.

$V1 < V2$  : True if and only if  $V1 < V2 < (V1 + 2^{*(N-1)})$ .

$V1 > V2$  : True if and only if  $V1 > V2 > (V1 - 2^{*(N-1)})$ .

Using 12 bits, as defined in this document, and assuming a Map-Version value of 69, Map-Versions in  $[70; 69 + 2047]$  are greater and versions in  $[69 + 2048; (69 + 4095) \bmod 4096]$  are smaller.

The initial Map-Version number of a new mapping can be randomly generated. However, it MUST NOT be zero (0) because it has a special meaning (see section [Section 3.1](#)).

### [3.1.](#) The special Map-Version 0

The value 0 (zero) is not a valid Map-Version Number. The only valid use of Map-Version number equal to 0 is in the Map Records. Map Records that have Map-Version number equal 0 indicate that there is no Map-Version number associated with the mapping. This means that LISP encapsulated packets, destined to the EID-Prefix the Map Record refers to, MUST never contain Map-Version number (i.e., V bit MUST always be 0). In other words, Map-Version number equal to 0 signal to the requester of the mapping that the Map-Versioning is not supported, or even if supported it must not be used for that specific EID-Prefix. Any value different from zero means that Map-Versioning is supported and can be used.

For LISP encapsulated packets with the V-bit set, if the Source Map-Version is 0, it means that the version number must be ignored and no checks (described in [Section 4](#)) need to be performed.

The fact that the 0 value has a special meaning for the Map-Version

number implies that, when updating a Map-Version number because of a change in the mapping, if the next value is 0 then Map-Version number must be incremented by 2 (i.e., set to 1, the next valid value).

## [4.](#) Dealing with Map-Version numbers

The main idea of using Map-Version numbers is that whenever there is a change in the mapping (e.g., adding/removing RLOCs, a change in the weights due to TE policies, or a change in the priorities) or an ISP realizes that one or more of its own RLOCs are not reachable anymore from a local perspective (e.g., through IGP, or policy changes) the ISP updates the mapping with a new Map-Version number.

In order to announce in a data-driven fashion that the mapping has been updated, Map-Version numbers used to create the outer IP header of the LISP encapsulated packet are embedded in the LISP specific header. This means that the header needs to contain two Map-Version numbers:

- o A first one from the EID-to-RLOC mapping in the EID-to-RLOC Database used to select the source RLOC, and called Source Map-Version Number.

- o A second one from the EID-to-RLOC mapping in the EID-to-RLOC Cache used to select the destination RLOC, and called Destination Map-Version Number.

By embedding both Source Map-Version Number and Destination Map-Version Number an ETR can perform the following checks:

1. The ITR has an up-to-date mapping in its cache for the destination EID and is performing encapsulation correctly.
2. In case of bidirectional traffic, the mapping in the local xTR cache for the source EID is up-to-date.

If one or both of the above conditions do not hold, the xTR can send a Map-Request either to make the ITR aware that a new mapping is available (see [Section 4.1](#)) or to update local mapping in the cache (see section [Section 4.2](#)).

#### [4.1](#). Handling Destination Map-Version Number

When an ETR receives a packet, the Destination Map Version number relates to the mapping for the destination EID for which the ETR is a RLOC. This mapping is part of the ETR LISP Database. Since the ETR is authoritative for the mapping, it has the correct and up-to-date Destination Map-Version number. A check on this version number is done, where the following cases can arise:

- o The packets arrive with the same Destination Map Version number stored in the EID-to-RLOC Database. This is the regular case. The ITR sending the packet has in its EID-to-RLOC Cache an up-to-date mapping. No further actions are needed.

- o The packet arrives with a Destination Map-Version number greater (i.e., newer) than the one stored in the EID-to-RLOC Database. Since the ETR is authoritative on the mapping, this means that someone is not behaving correctly w.r.t. the specifications, thus the packets carries a not valid version number and can be silently dropped.
- o The packets arrive with an Destination Map-Version number smaller (i.e., older) than the one stored in the EID-to-RLOC Database. This means that the ITR sending the packet has an old mapping in its EID-to-RLOC Cache containing stale information. Further actions are needed. The ITR sending the packet must be informed that a newer mapping is available. This is done with a Map-Request message sent back to the ITR. The Map-Request will either trigger a Map-Request back using the SMR bit or it will piggy-back the newer mapping. These are not new mechanisms; how to SMR or

piggy-back mappings in Map-Request messages is already described in [[I-D.ietf-lisp](#)], while their security is discussed in [[I-D.saucez-lisp-security](#)]. These Map-Request message should be rate limited (rate limitation policies are also described in [[I-D.ietf-lisp](#)]). The gain introduced by Map-Version Numbers is that after a certain number of retries, if the Destination Map-Version Number in the packets is not updated, packet can be silently dropped because either the ITR is refusing to use the mapping for which the ETR is authoritative or it might be some form of attack. Note that the rule can be even more restrictive. If the mapping has been the same for a period of time as long as the TTL (defined in LISP [[I-D.ietf-lisp](#)]) of the previous version of the mapping, all packets arriving with an old Map-Version should be silently dropped right away without issuing any Map-Request. Indeed, if the new mapping with the updated version number has been stable for at least the same time as the TTL of the older mapping, all the entries in the caches of ITRs must have expired. If packets with old Map-Version number are still received, the reason is that either someone has not respected the TTL, or it is a form of spoof/attack. In both cases this is not valid behavior w.r.t. the specifications and the packet can be silently dropped.

#### [4.2.](#) Handling Source Map-Version Number

When an xTR receives a packet, the Source Map-Version Number relates to the mapping for the source EID for which the ITR is authoritative. If the xTR has an entry in its LISP Cache a check is performed and the following cases can arise:

- o The packet arrives with the same Source Map-Version number stored in the LISP Cache. This is the correct regular case. The xTR has in its cache an up-to-date copy of the mapping. No further actions are needed.

- o The packet arrives with a Source Map-Version number greater (i.e., newer) than the one stored in the local LISP Cache. This means that xTR has in its cache a mapping that is stale and needs to be updated. The packet is considered valid but further actions are needed. In particular a Map-Request must be sent to get the new mapping for the source EID. This is a normal Map-Request message sent through the mapping system and must respect the specifications in [[I-D.ietf-lisp](#)], including rate limitation

policies.

- o The packet arrives with a Source Map-Version number smaller (i.e., older) than the one stored in the local LISP Cache. Such a case is not valid w.r.t. the specifications. Indeed, if the mapping is already present in the LISP Cache, this means that an explicit Map-Request has been sent and a Map-Reply has been received from an authoritative source. Assuming that the mapping system is not corrupted anyhow, the Map-Version in the LISP Cache is the correct one, hence the packet is not valid and can be silently dropped.

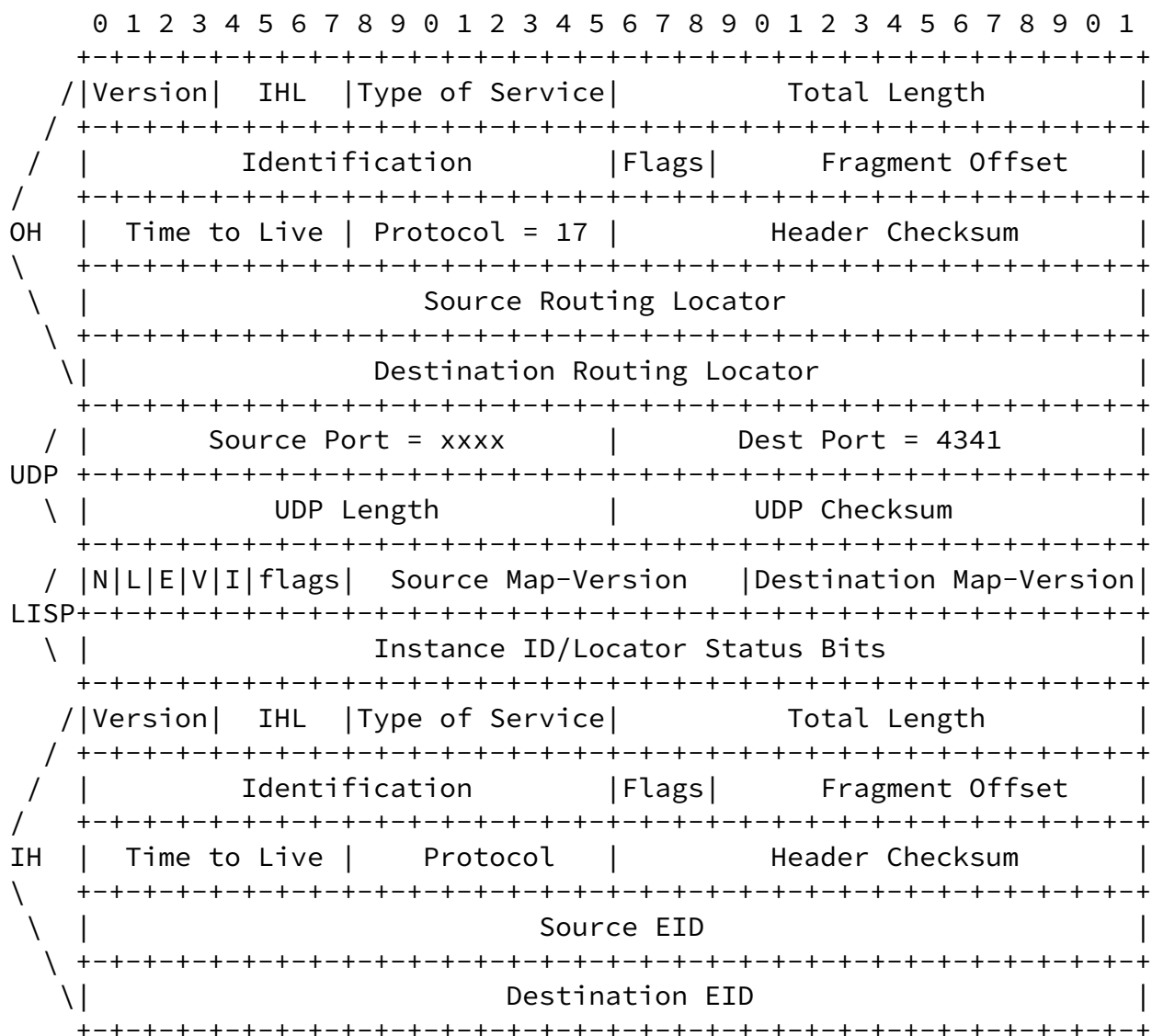
Otherwise, if the xTR does not have an entry in its cache (e.g. unidirectional traffic) the Source Map-Version can be safely ignored.

## [5.](#) LISP header and Map-Version numbers

In order for the versioning approach to work, the LISP specific header has to carry both Source Map-Version Number and Destination Map-Version Number. This is done by setting the V-bit in the LISP specific header. When the V-bit is set the low-order 24-bits of the first longword (which usually contains the nonce) are used transport both source and destination Map-Versions. In particular the first 12 bits are used for Source Map-Version and the second 12 bits for the Destination Map-Version.

Hereafter is the example of LISP header carrying version numbers in the case of IPv4-in-IPv4 encapsulation. The same setting can be used for any other case (IPv4-in-IPv6, IPv6-in-IPv4, IPv6-in-IPv6). The authoritative document for LISP packet format is [[I-D.ietf-lisp](#)], the following example is proposed only for explanation purposes.





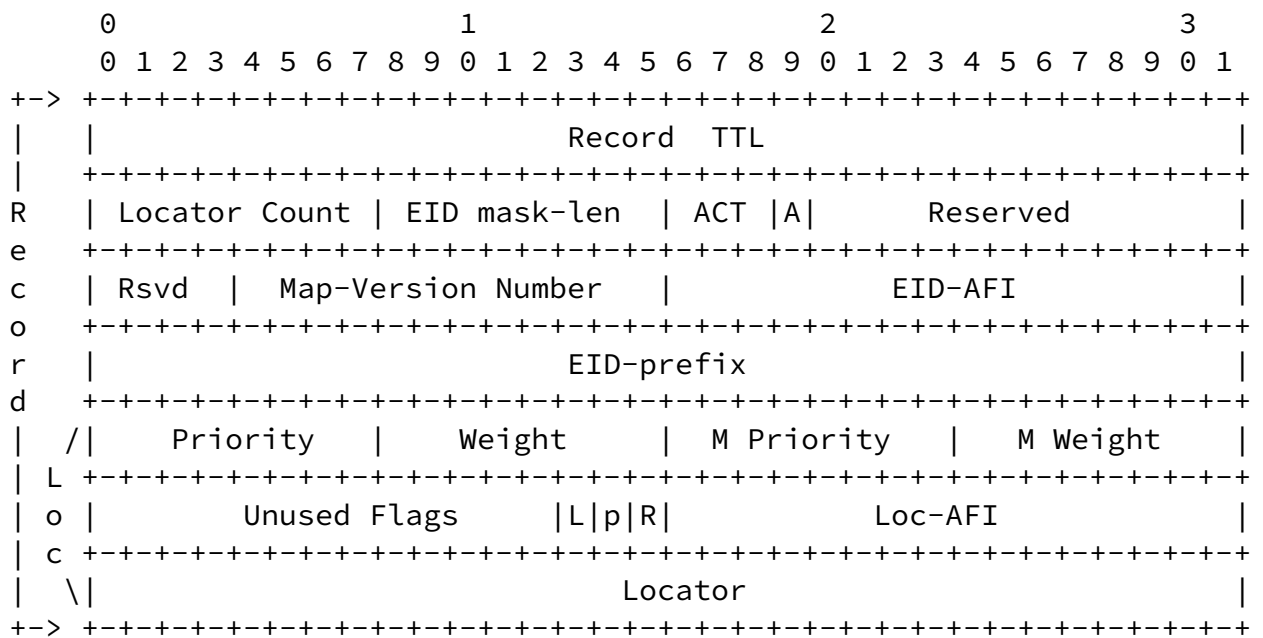
- V: this is the Map-Version bit as defined in [[I-D.ietf-lisp](#)]. When this bit is set to 1 the low-order 24-bits of the first longword of the LISP header contain Map-Version numbers.
- Source Map-Version number (12 bits): Map-Version of the mapping used by the ITR to select the RLOC present in the "Source Routing Locator" field. Note that the mapping used for such a selection is determined by the Source EID through a search in the LISP Database of the ITR.
- Destination Map-Version Number (12 bits): Map-Version of the mapping used by the ITR to select the RLOC present in the "Destination Routing Locator" field. Note that the mapping used for such a selection is determined by the Destination EID, used as lookup key in the LISP Cache of the ITR.

Not all of the LISP encapsulated packets need to carry version numbers. When Map-Version number are carried the V bit must be set

to 1. All legal combination of the flags, when the V-bit is set to 1 are described in [[I-D.ietf-lisp](#)]. As a recall and in summary, Map-Version cannot be used with the Echo-Nonce feature (E = 1) and the Nonce feature (N = 1), since they use the same bitfield.

## 6. Map Record and Map-Version

To accommodate the proposed mechanism, the Map Records that are transported on Map-Request/Map-Reply messages need to carry the Map-Version number as well. For this purpose the 12-bits before the EID-AFI field in the Record that describe a mapping is used. This is defined in [[I-D.ietf-lisp](#)] and reported here as example.



**Map-Version Number:** Map-Version of the mapping contained in the Record. As explained in [Section 3.1](#) this field can be zero (0), meaning that no Map-Version is associated to the mapping, hence LISP encapsulated packet must not contain Map-Version in the LISP specific header.

Note that this packet format works perfectly with xTRs that do not support Map-Versioning, since they can simply ignore those bits. Furthermore, existing and future mapping distribution protocol (e.g., ALT [[I-D.ietf-lisp-alt](#)]) are able to carry version numbers without needing any modification. The same applies to the LISP Map Server ([[I-D.ietf-lisp-ms](#)]) which will still work without any change since reserved bits are simply ignored.

## 7. Benefits and case studies for Map-Versioning

In the following sections we provide more discussion on various aspects and use of the Map-Versioning. Security observations are instead grouped in [Section 9](#).

### 7.1. Synchronization of different xTRs

Map-Versioning does not require additional synchronization mechanism compared to the normal functioning of LISP without Map-Versioning. Clearly all the ETRs have to reply with the same Map-Version number, otherwise there can be an inconsistency that creates additional control traffic, instabilities, traffic disruptions.

As an example, let's consider the topology of Figure 1 where ITR A.1 of domain A is sending unidirectional traffic to the xTR B of domain B, while xTR A.2 of domain A and xTR B of domain B exchange bidirectional traffic.

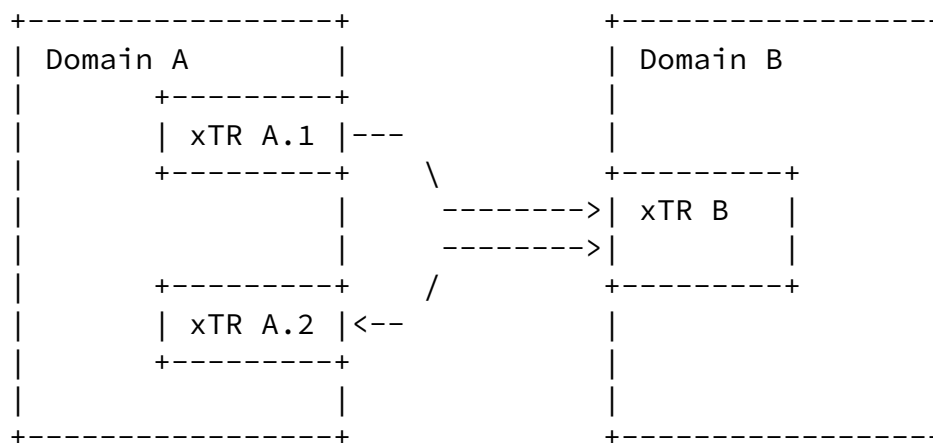


Figure 1

Obviously in the case of Map-Versioning both xTRs of domain A must use the same value otherwise the xTR of domain B will start to send Map-Requests.

The same problem can, however, arise without Map-Versioning. For instance if the two xTRs of domain A send different Loc Status Bits. In this case either the traffic is disrupted, if the xTR B trusts the Locator Status Bits, or it xTR B will start sending Map-Requests to confirm the each change in the reachability.

So far, LISP does not provide any specific synchronization mechanism, but assumes that synchronization is provided by configuring the different xTRs consistently. The same applies for Map-Versioning.

If in the future any synchronization mechanism is provided, Map-Versioning will take advantage of it automatically since it is included in the Record format, as described in [Section 6](#).

## [7.2](#). Map-Versioning and unidirectional traffic

When using Map-Versioning the LISP specific header carries two Map-Version numbers, for both source and destination mapping. This can raise the question on what will happen in the case of unidirectional flows, like for instance in the case presented in Figure 2, since LISP specification do not mandate for ETR to have a mapping for the source EID.

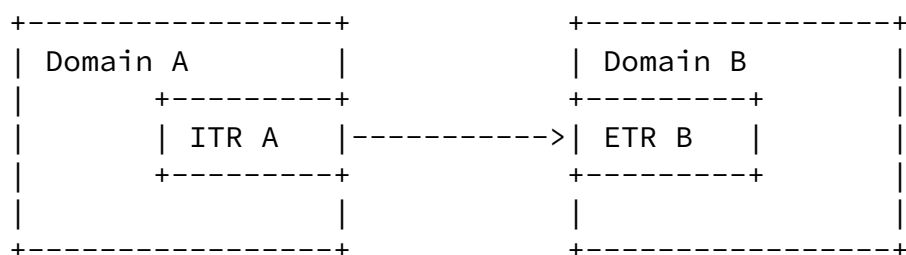


Figure 2

For what concerns the ITR, it is able to put both source and destination version number in the LISP header since the Source Map-Version number is in ITR's database, while the Destination Map-Version number is in ITR's cache.

For what concerns the ETR, it simply checks only the Destination Map-Version number in the same way as described in [Section 4](#), ignoring the Source Map-Version number.

### [7.3.](#) Map-Versioning and interworking

Map-Versioning works in the context of interworking between LISP and IPv4 and IPv6 ([\[I-D.ietf-lisp-interworking\]](#)) in the following way.

The case of proxy-ITR encapsulating packet for LISP sites is basically the same as the unidirectional traffic case presented in the previous section. The same rules can be applied. The only difference that arises is the fact that a proxy-ITR does not have any mapping, since it just encapsulate packets arriving from non-LISP site, thus it has no Source Map-Version. In this case, the proxy-ITR will just put the special value 0 (zero) as Source Map-Version number, while the receiving ETR will ignore the field.

### [7.4.](#) Graceful RLOC shutdown/withdraw

Map-Versioning can be even used to perform a graceful shutdown or withdraw of a specific RLOC. This is achieved by simply issuing a new mapping, with an updated Map-Version number, where the specific RLOC to be shut down is withdrawn or announced as unreachable (R bit in the Map Record, see [\[I-D.ietf-lisp\]](#)), but without actually turning it off.

Once no more traffic is received by the RLOC, because all sites have updated the mapping, it can be shut down safely.

### [7.5.](#) Map-Version for lightweight LISP implementation

The use of Map-Versioning can help in simplifying the implementation of LISP. This comes with the price of not supporting Loc-Status-Bit, which are useful in some contexts.

In the current LISP specifications the set of RLOCs must always be maintained ordered and consistent with the content of the Loc Status Bits (see section 6.5 of [\[I-D.ietf-lisp\]](#)). With Map-Versioning such type of mechanisms can be avoided. When a new RLOC is added to a mapping, it is not necessary to "append" new locators to the existing ones as explained in Section 6.5 of [\[I-D.ietf-lisp\]](#). A new mapping with a new Map-Version number will be issued, and since the old

locators are still valid the transition will be disruptionless. The same applies for the case a RLOC is withdrawn. There is no need to maintain holes in the list of locators, as is the case when using Locator Status Bits, for sites that are not using the RLOC that has been withdrawn the transition will be disruptionless.

All of these operations, as already stated, do not need to maintain any consistency among Locator Status Bits, and the way RLOC are stored in the cache. This eases implementation.

Further, Map-Version can be used to substitute the "clock sweep" operation described in Section 6.5.1 of [[I-D.ietf-lisp](#)]. Indeed, every LISP site communicating to a specific LISP site that has updated the mapping will be informed of the available new mapping in a data-driven manner.

Note that what proposed in the present section is just a case study and MUST NOT be considered as specification for a lightweight LISP implementation.

## [8.](#) Incremental deployment and implementation status

Map-Versioning can be incrementally deployed without any negative impact on existing LISP xTRs. Any LISP element that does not support Map-Versioning can safely ignore them. Further, there is no need of any specific mechanism to discover if an xTR supports or not Map-Versioning. This information is already included in the Map Record.

Map-Versioning is currently implemented in OpenLISP [[I-D.iannone-openlisp-implementation](#)].

Note that the reference document for LISP implementation and interoperability tests remains [[I-D.ietf-lisp](#)].

## [9.](#) Security Considerations

Map-Versioning does not introduces any new security issue concerning

both the data-plane and the control-plane. On the contrary, as described in the following, if Map-Versioning is used also to update mappings in case of change in the reachability information (i.e., instead of the Locator Status Bits) it is possible to reduce the effects of some DoS or spoofing attacks that can happen in an untrusted environment.

A thorough security analysis of LISP is documented in [\[I-D.saucez-lisp-security\]](#).

### 9.1. Map-Versioning against traffic disruption

An attacker can try to disrupt ongoing communications by creating LISP encapsulated packets with wrong Locator Status Bits. If the xTR blindly trusts the Locator Status Bits it will change the encapsulation accordingly, which can result in traffic disruption.

This does not happen in the case of Map-Versioning. As described in [Section 4](#), upon a version number change the xTR first issues a Map-Request. The assumption is that the mapping distribution system is sufficiently secure that Map-Request and Map-Reply messages and their content can be trusted. Security issues concerning specific mapping distribution system are out of the scope of this document. Note also that in the case of Map-Versioning the attacker should "guess" a valid version number that triggers a Map-Request, as described in [Section 4](#), otherwise the packet is simply dropped.

Note that a similar level of security can be obtained with Loc Status Bits, by simply making mandatory to verify any change through a Map-Request. However, in this case Locator Status Bits lose their

meaning, because, it does not matter anymore which specific bits has changed, the xTR will query the mapping system and trust the content of the received Map-Reply. Furthermore there is no way to perform filtering as in the Map-Versioning in order to drop packets that do not carry a valid Map-Version number. In the case of Locator Status Bits, any random change can trigger a Map-Request (unless rate limitation is enabled which raise another type of attack discussed in [Section 9.2](#)).

### 9.2. Map-Versioning against reachability information DoS

Attackers can try to trigger a large amount of Map-Request by simply forging packets with random Map-Version or random Locator Status Bits. In both cases the Map-Requests are rate limited as described in [[I-D.ietf-lisp](#)]. However, differently from Locator Status Bit where there is no filtering possible, in the case of Map-Versioning is possible to filter not valid version numbers before triggering a Map-Request, thus helping in reducing the effects of DoS attacks. In other words the use of Map-Versioning enables a fine control on when to update a mapping or when to notify that a mapping has been updated.

It is clear, that Map-Versioning does not protect against DoS and DDoS attacks, where an xTR loses processing power doing checks on the LISP header of packets sent by attackers. This is independent from Map-Versioning and is the same for Loc Status Bits.

## [10.](#) Acknowledgements

The authors would like to thank Pierre Francois, Noel Chiappa, Dino Farinacci for their comments and review.

This work has been partially supported by the INFSO-ICT-216372 TRILOGY Project ([www.trilogy-project.org](http://www.trilogy-project.org)).

## [11.](#) References

### [11.1.](#) Normative References

[[I-D.ietf-lisp](#)]

Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "Locator/ID Separation Protocol (LISP)", [draft-ietf-lisp-07](#) (work in progress), April 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

Iannone, et al.

Expires January 13, 2011

[Page 14]

---

Internet-Draft

LISP Map-Versioning

July 2010

### [11.2.](#) Informative References

[[I-D.iannone-openlisp-implementation](#)]

Iannone, L., Saucez, D., and O. Bonaventure, "OpenLISP



Implementation Report",  
[draft-iannone-openlisp-implementation-01](#) (work in progress), July 2008.

[I-D.ietf-lisp-alt]

Fuller, V., Farinacci, D., Meyer, D., and D. Lewis, "LISP Alternative Topology (LISP+ALT)", [draft-ietf-lisp-alt-04](#) (work in progress), April 2010.

[I-D.ietf-lisp-interworking]

Lewis, D., Meyer, D., Farinacci, D., and V. Fuller, "Interworking LISP with IPv4 and IPv6", [draft-ietf-lisp-interworking-00](#) (work in progress), May 2009.

[I-D.ietf-lisp-ms]

Fuller, V. and D. Farinacci, "LISP Map Server", [draft-ietf-lisp-ms-05](#) (work in progress), April 2010.

[I-D.saucez-lisp-security]

Saucez, D., Iannone, L., and O. Bonaventure, "Notes on LISP Security Threats and Requirements", [draft-saucez-lisp-security-00](#) (work in progress), October 2009.

## [Appendix A](#). Map-Version wrap-around

The present section proposes an estimation of the wrap-around time for proposed 12 bits size for the Map-Version Number. Using a granularity of seconds and assuming as worst case that a new version is issued each second, it takes slightly more than 1 hour before the version wraps around. Note that the granularity of seconds is in line with the rate limitation policy for Map-Request messages, as proposed in the LISP main specifications ([[I-D.ietf-lisp](#)]). Alternatively a granularity of minutes can also be used, as for the TTL of the Map-Reply ([[I-D.ietf-lisp](#)]). Using a granularity of minutes leads to a much longer time before wrap-around. In particular, when using 12 bits, the wrap-around time is almost 3 days.

For general information, hereafter there is a table with a rough estimation of the time before wrap-around happens considering different sizes of the Map-Version Number and different time

granularity.

Version Number		Time before wrap around			
Size (bits)					
		Granularity: Minutes		Granularity: Seconds	
32		8171	Years	136	Years
30		2042	Years	34	Years
24		31	Years	194	Days
16		45	Days	18	Hours
15		22	Days	9	Hours
14		11	Days	4	Hours
13		5.6	Days	2.2	Hours
12		2.8	Days	1.1	Hours

Figure 3: Estimation of time before wrap-around

#### Authors' Addresses

Luigi Iannone  
 TU Berlin – Deutsche Telekom Laboratories AG  
 Ernst-Reuter Platz 7  
 Berlin  
 Germany

Email: [luigi@net.t-labs.tu-berlin.de](mailto:luigi@net.t-labs.tu-berlin.de)

Damien Saucez  
 Universite catholique de Louvain  
 Place St. Barbe 2  
 Louvain la Neuve  
 Belgium

Email: [damien.saucez@uclouvain.be](mailto:damien.saucez@uclouvain.be)

Olivier Bonaventure  
 Universite catholique de Louvain  
 Place St. Barbe 2  
 Louvain la Neuve  
 Belgium

Email: [olivier.bonaventure@uclouvain.be](mailto:olivier.bonaventure@uclouvain.be)

