

Network Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: October 14, 2017

D. Farinacci  
lispers.net  
P. Pillay-Esnault  
Huawei Technologies  
W. Haddad  
Ericsson  
April 12, 2017

LISP EID Anonymity  
draft-farinacci-lisp-eid-anonymity-02

Abstract

This specification will describe how ephemeral LISP EIDs can be used to create source anonymity. The idea makes use of frequently changing EIDs much like how a credit-card system uses a different credit-card numbers for each transaction.

Requirements Language

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].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on October 14, 2017.

Copyright Notice

Copyright (c) 2017 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 Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Definition of Terms . . . . .	3
3. Overview . . . . .	3
4. Design Details . . . . .	4
5. Other Types of Ephemeral-EIDs . . . . .	4
6. Interworking Considerations . . . . .	5
7. Multicast Considerations . . . . .	5
8. Performance Improvements . . . . .	5
9. Security Considerations . . . . .	6
10. IANA Considerations . . . . .	6
11. References . . . . .	6
11.1. Normative References . . . . .	6
11.2. Informative References . . . . .	7
Appendix A. Acknowledgments . . . . .	8
Appendix B. Document Change Log . . . . .	8
B.1. Changes to draft-farinacci-lisp-eid-anonymity-02 . . . . .	8
B.2. Changes to draft-farinacci-lisp-eid-anonymity-01 . . . . .	8
B.3. Changes to draft-farinacci-lisp-eid-anonymity-00 . . . . .	8
Authors' Addresses . . . . .	8

## 1. Introduction

The LISP architecture [RFC6830] specifies two namespaces, End-Point IDs (EIDs) and Routing Locators (RLOCs). An EID identifies a node in the network and the RLOC indicates the EID's topological location. Typically EIDs are globally unique so a end-node system can connect to any other end-node system on the Internet. Privately used EIDs are allowed when scoped within a VPN but must always be unique within that scope. Therefore, address allocation is required by network administration to avoid address collisions or duplicate address use. In a multiple namespace architecture like LISP, typically the EID will stay fixed while the RLOC can change. This occurs when the EID is mobile or when the LISP site the EID resides in changes its connection to the Internet.

LISP creates the opportunity where EIDs are fixed and won't change. This can create a privacy problem more so than what we have on the Internet today. This draft will examine a technique to allow a end-node system to use a temporary address. The lifetime of a temporary address can be the same as a lifetime of an address in use today on the Internet or can have traditionally shorter lifetimes, possibly on the order of a day or even change as frequent as new connection attempts.

## 2. Definition of Terms

Ephemeral-EID - is an IP address that is created randomly for use for a temporary period of time. An Ephemeral-EID has all the properties of an EID as defined in [RFC6830]. Ephemeral-EIDs are not stored in the Domain Name System (DNS) and should not be used in long-term address referrals.

Client End-Node - is a network node that originates and consumes packets. It is a system that originates packets or initiates the establishment of transport-layer connections. It does not offer services as a server system would. It accesses servers and attempts to do it anonymously.

## 3. Overview

A client end-node can assign its own ephemeral EID and use it to talk to any system on the Internet. The system is acting as a client where it initiates communication and desires to be an inaccessible resource from any other system. The ephemeral EID is used as a destination address solely to return packets to resources the ephemeral EID connects to.

Here is the procedure a client end-node would use:

1. Client end-node desires to talk on the network. It creates and assigns an ephemeral-EID on any interface.
2. If the client end-node is a LISP xTR, it will register the ephemeral-EID with a globally routable RLOC. If the client end-node is not a LISP xTR, it can send packets on the network where a LISP router xTR will register the ephemeral-EID with its RLOC.
3. The client end-node originates packets with a source address equal to the ephemeral-EID and will receive packets addressed to the ephemeral-EID.
4. When the client end-node decides to stop using the ephemeral-EID, it will deregister it from the mapping system and create and

assign a new ephemeral-EID, or decide to configure a static global address, or participate in DHCP to get assigned a leased address.

Note that the ephemeral-EID can be mobile just like any other EID so if it is initially registered to the mapping system with one or more RLOCs, later the RLOC-set can change as the ephemeral-EID roams.

#### 4. Design Details

This specification proposes the use of the experimental LISP EID-block 2001:5::/32 when IPv6 is used. See IANA Considerations section for a specific sub-block allocation request. When IPv4 is used, the Class E block 240.0.0.0/4 is being proposed.

The client end-node system will use the rest of the host bits to allocate a random number to be used as the ephemeral-EID. The EID can be created manually or via a programatic interface. When the EID address is going to change frequently, it is suggested to use a programatic interface. The probability of address collision is unlikely for IPv6 EIDs but could occur for IPv4 EIDs. A client end-node can create a ephemeral-EID and then look it up in the mapping system to see if it exists. If the EID exists in the mapping system, the client end-node can attempt creation of a new random number for the ephemeral-EID. See Section 8 where ephemeral-EIDs can be preallocated and registered to the mapping system before use.

When the client end-node system is co-located with the RLOC and acts as an xTR, it should register the binding before sending packets. This eliminates a race condition for returning packets not knowing where to encapsulate packets to the ephemeral-EID's RLOCs. See Section 8 for alternatives for fixing this race condition problem. When the client end-node system is not acting as an xTR, it should send some packets so its ephemeral-EID can be discovered by an xTR which supports EID-mobility [I-D.portoles-lisp-eid-mobility] so mapping system registration can occur before the destination returns packets. When the end-node system is acting as an xTR, the EID and RLOC-set is co-located in the same node. So when the EID is created, the xTR can register the mapping versus waiting for packet transmission.

#### 5. Other Types of Ephemeral-EIDs

When IPv6 Ephemeral-EIDs are used, an alternative to a random number can be used. For example, the low-order bits of the IPv6 address could be a cryptographic hash of a public-key. Mechanisms from [RFC3972] could be used for EIDs. Using this approach allows the sender with a hashed EID to be authenticated. So packet signatures

can be verified by the corresponding public-key. When hashed EIDs are used, the EID can change frequently as rekeying may be required for enhanced security.

## 6. Interworking Considerations

If a client end-node is communicating with a system that is not in a LISP site, the procedures from [RFC6832] should be followed. The PITR will be required to originate route advertisements for the ephemeral-EID sub-block [I-D.draft-ietf-lisp-eid-block] so it can attract packets sourced by non-LISP sites destined to ephemeral-EIDs. However, in the general case, the coarse block from [I-D.draft-ietf-lisp-eid-block] will be advertised which would cover the sub-block. For IPv4, the 240.0.0.0/4 must be advertised into the IPv4 routing system.

## 7. Multicast Considerations

A client end-node system can be a member of a multicast group fairly easily since its address is not used for multicast communication as a receiver. This is due to the design characteristics of IGMP [RFC3376] [RFC2236] [RFC1112] and MLD [RFC2710] [RFC3810].

When a client end-node system is a multicast source, there is ephemeral (S,G) state that is created and maintained in the network via multicast routing protocols such as PIM [RFC4602] and when PIM is used with LISP [RFC6802]. In addition, when [I-D.draft-ietf-lisp-signal-free-multicast] is used, ephemeral-EID state is created in the mapping database. This doesn't present any problems other than the amount of state that may exist in the network if not timed out and removed promptly.

However, there exists a multicast source discovery problem when PIM-SSM [RFC4607] is used. Members that join (S,G) channels via out of band mechanisms. These mechanisms need to support ephemeral-EIDs. Otherwise, PIM-ASM [RFC4602] or PIM-Bidir [RFC5015] will need to be used.

## 8. Performance Improvements

An optimization to reduce the race condition between registering ephemeral-EIDs and returning packets as well as reducing the probability of ephemeral-EID address collision is to preload the mapping database with a list of ephemeral-EIDs before using them. It comes at a expense of rebinding all of registered ephemeral-EIDs when there is an RLOC change. There is work in progress to consider adding a level of indirection here so a single entry gets the RLOC update and the list of ephemeral-EIDs point to the single entry.

## 9. Security Considerations

When LISP-crypto [I-D.draft-ietf-lisp-crypto] is used the EID payload is more secure through encryption providing EID obfuscation of the ephemeral-EID as well as the global-EID it is communicating with. But the obfuscation only occurs between xTRs. So the randomness of a ephemeral-EID inside of LISP sites provide a new level of privacy.

## 10. IANA Considerations

This specification is requesting the sub-block 2001:5:ffff::/48 for ephemeral-EID usage.

## 11. References

### 11.1. Normative References

- [RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, DOI 10.17487/RFC1112, August 1989, <<http://www.rfc-editor.org/info/rfc1112>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, DOI 10.17487/RFC2236, November 1997, <<http://www.rfc-editor.org/info/rfc2236>>.
- [RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, DOI 10.17487/RFC2710, October 1999, <<http://www.rfc-editor.org/info/rfc2710>>.
- [RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, DOI 10.17487/RFC3376, October 2002, <<http://www.rfc-editor.org/info/rfc3376>>.
- [RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <<http://www.rfc-editor.org/info/rfc3810>>.
- [RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <<http://www.rfc-editor.org/info/rfc3972>>.

- [RFC4602] Pusateri, T., "Protocol Independent Multicast - Sparse Mode (PIM-SM) IETF Proposed Standard Requirements Analysis", RFC 4602, DOI 10.17487/RFC4602, August 2006, <<http://www.rfc-editor.org/info/rfc4602>>.
- [RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <<http://www.rfc-editor.org/info/rfc4607>>.
- [RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, DOI 10.17487/RFC5015, October 2007, <<http://www.rfc-editor.org/info/rfc5015>>.
- [RFC6802] Baillargeon, S., Flinta, C., and A. Johnsson, "Ericsson Two-Way Active Measurement Protocol (TWAMP) Value-Added Octets", RFC 6802, DOI 10.17487/RFC6802, November 2012, <<http://www.rfc-editor.org/info/rfc6802>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <<http://www.rfc-editor.org/info/rfc6830>>.
- [RFC6832] Lewis, D., Meyer, D., Farinacci, D., and V. Fuller, "Interworking between Locator/ID Separation Protocol (LISP) and Non-LISP Sites", RFC 6832, DOI 10.17487/RFC6832, January 2013, <<http://www.rfc-editor.org/info/rfc6832>>.

## 11.2. Informative References

- [I-D.draft-ietf-lisp-crypto]  
Farinacci, D. and B. Weis, "LISP Data-Plane Confidentiality", draft-ietf-lisp-crypto-03 (work in progress).
- [I-D.draft-ietf-lisp-eid-block]  
Iannone, L., Lewis, D., Meyer, D., and V. Fuller, "LISP EID Block", draft-ietf-lisp-eid-block-13.txt (work in progress).
- [I-D.draft-ietf-lisp-signal-free-multicast]  
Farinacci, D. and V. Moreno, "Signal-Free LISP Multicast", draft-ietf-lisp-signal-free-multicast-00.txt (work in progress).

[I-D.meyer-lisp-mn]

Farinacci, D., Lewis, D., Meyer, D., and C. White, "LISP Mobile Node", draft-meyer-lisp-mn-16 (work in progress), December 2016.

[I-D.portoles-lisp-eid-mobility]

Portoles-Comeras, M., Ashtaputre, V., Moreno, V., Maino, F., and D. Farinacci, "LISP L2/L3 EID Mobility Using a Unified Control Plane", draft-portoles-lisp-eid-mobility-02 (work in progress), April 2017.

#### Appendix A. Acknowledgments

The author would like to thank the LISP WG for their review and acceptance of this draft.

#### Appendix B. Document Change Log

[RFC Editor: Please delete this section on publication as RFC.]

##### B.1. Changes to draft-farinacci-lisp-eid-anonymity-02

- o Posted April 2017.
- o Added section describing how ephemeral-EIDs can use a public key hash as an alternative to a random number.
- o Indciate when an EID/RLOC co-located, that the xTR can register the EID when it is configured or changed versus waiting for a packet to be sent as in the EID/RLOC separated case.

##### B.2. Changes to draft-farinacci-lisp-eid-anonymity-01

- o Posted October 2016.
- o Update document timer.

##### B.3. Changes to draft-farinacci-lisp-eid-anonymity-00

- o Posted April 2016.
- o Initial posting.

#### Authors' Addresses



Dino Farinacci  
lispers.net  
San Jose, CA  
USA

Email: farinacci@gmail.com

Padma Pillay-Esnault  
Huawei Technologies  
San Clara, CA  
USA

Email: padma@huawei.com

Wassim Haddad  
Ericsson  
San Clara, CA  
USA

Email: wassim.haddad@ericsson.com

Network Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: December 9, 2017

D. Farinacci  
lispers.net  
P. Pillay-Esnault  
Huawei Technologies  
June 7, 2017

LISP Predictive RLOCs  
draft-ietf-lisp-predictive-rlocs-00

Abstract

This specification will describe a method to achieve near-zero packet loss when an EID is roaming quickly across RLOCs.

Requirements Language

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].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on December 9, 2017.

Copyright Notice

Copyright (c) 2017 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 Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Definition of Terms . . . . .	3
3. Overview . . . . .	3
4. Design Details . . . . .	5
4.1. RLE Encoding . . . . .	5
4.2. Packet Delivery Optimizations . . . . .	6
4.3. Trading Off Replication Cost . . . . .	7
5. Directional Paths with Intersections . . . . .	8
6. Multicast Considerations . . . . .	9
7. Multiple Address-Family Considerations . . . . .	10
8. Scaling Considerations . . . . .	10
9. Security Considerations . . . . .	11
10. IANA Considerations . . . . .	11
11. References . . . . .	11
11.1. Normative References . . . . .	11
11.2. Informative References . . . . .	12
Appendix A. Acknowledgments . . . . .	12
Appendix B. Document Change Log . . . . .	13
B.1. Changes to draft-ietf-lisp-predictive-rlocs-00.txt . . .	13
B.2. Changes to draft-farinacci-lisp-predictive-rlocs-02.txt .	13
B.3. Changes to draft-farinacci-lisp-predictive-rlocs-01.txt .	13
B.4. Changes to draft-farinacci-lisp-predictive-rlocs-00.txt .	13
Authors' Addresses . . . . .	13

## 1. Introduction

The LISP architecture [RFC6830] specifies two namespaces, End-Point IDs (EIDs) and Routing Locators (RLOCs). An EID identifies a node in the network and the RLOC indicates the EID's topological location. When an node roams in the network, its EID remains fixed and unchanged but the RLOCs associated with it change to reflect its new topological attachment point. This specification will focus EIDs and RLOCs residing in separate nodes. An EID is assigned to a host node that roams while the RLOCs are assigned to network nodes that stay stationary and are part of the network topology. For example, a set of devices on an aircraft are assigned EIDs, and base stations on the ground attached to the Internet infrastructure are configured as LISP xTRs where their RLOCs are used for the bindings of the EIDs on the aircraft up in the air.

The scope of this specification will not emphasize general physical roaming as an aircraft would do in the sky but in a direction that is more predictable such as a train traveling on a track or vehicle that travels along a road.

## 2. Definition of Terms

Roaming-EID - is a network node that moves from one topological location in the network to another. The network node uses the same EID when it is roaming. That is, the EID address does not change for reasons of mobility. A roaming-EID can also be a roaming EID-prefix where a set of EIDs covered by the prefix are all roaming and fate-sharing the same set of RLOCs at the same time.

Predictive RLOCs - is a set of ordered RLOCs in a list each assigned to LISP xTRs where the next RLOC in the list has high probability it will be the next LISP xTR in a physical path going in a single predictable direction.

Road-Side-Units (RSUs) - is a network node that acts as a router, more specifically as a LISP xTR. The xTR automatically discovers roaming-EIDs that come into network connectivity range and relays packets to and from the roaming-EID. RSUs are typically deployed along a directional path like a train track or road and are in connectivity range of devices that travel along the directional path.

## 3. Overview

The goal of this specification is to describe a make-before-break EID-mobility mechanism that offers near-zero packet loss. Offering minimal packet loss, not only allows transport layers to operate more efficiently, but because an EID does not change while moving, transport layer session continuity is maintained. To achieve these requirements, a mechanism that reacts to the mobility event is necessary but not sufficient. So the question is not that there isn't a reaction but when it happens. By using some predictive algorithms, we can guess with high probability where the EID will roam to next. We can achieve this to a point where packet data will be at the new location when the EID arrives.

First we should examine both the send and receive directions with respect to the roaming-EID. Refer to Figure 1 for discussion. We show a network node with a fixed EID address assigned to a roaming-EID moving along a train track. And there are LISP xTRs deployed as Road-Side-Units to support the connectivity between the roaming-EID and the infrastructure or to another roaming-EID.

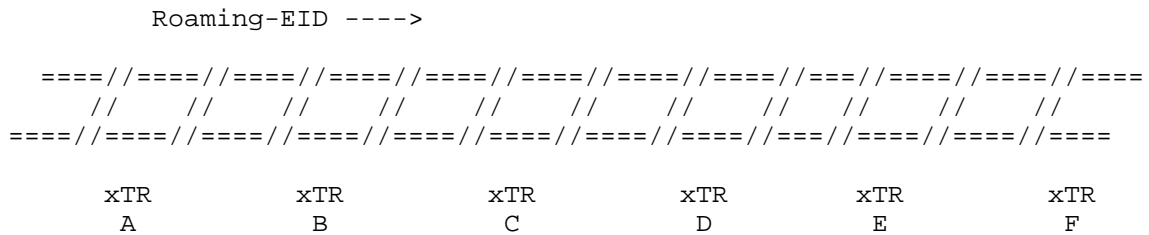


Figure 1: Directional Mobility

For the send direction from roaming-EID to any destination can be accomplish as a local decision. As long as the roaming-EID is in signal range to any xTR along the path, it can use it to forward packets. The LISP xTR, acting as an ITR, can forward packets to destinations in non-LISP sites as well as to stationary and roaming EIDs in LISP sites. This is accomplished by using the LISP overlay via dynamic packet encapsulation. When the roaming-EID sends packets, the LISP xTR must discover the EID and MAY register the EID with a set of RLOCs to the mapping system [I-D.portoles-lisp-eid-mobility]. The discovery process is important because the LISP xTR, acting as an ETR for decapsulating packets that arrive, needs to know what local ports or radios to send packets to the roaming-EID.

Much of the focus of this design is on the packet direction to the roaming-EID. And how remote LISP ITRs find the current location (RLOCs) quickly when the roaming-EID is moving at high speed. This specification solves the fast roaming with the introduction of the Predictive-RLOCs algorithm.

Since a safe assumption is that the roaming-EID is going in one direction and cannot deviate from it allows us to know a priori the next set of RLOCs the roaming-EID will pass by. Referring to Figure 1, if the roaming-EID is in range near xTR-A, then as it moves, it will at some point pass by xTR-B and xTR-C, and so on. As the roaming-EID moves, one could time when the EID is mapped to RLOC A, and when it should change to RLOC B and so on. However, the speed of movement of the roaming-EID won't be constant and the variables involved in consistent timing cannot be relied on. Furthermore, timing the move is not a make-before-break algorithm, meaning the reaction of the binding happens at the time the roaming-EID is discovered by an xTR. One cannot achieve fast hand-offs when message signaling will be required to inform remote ITRs of the new binding.

The Predictive RLOCs algorithm allows a set of RLOCs, in an ordered list, to be provided to remote ITRs so they have the information

available and local for when they need to use it. Therefore, no control-plane message signaling occurs when the roaming-EID is discovered by LISP xTRs.

#### 4. Design Details

Predictive RLOCs accommodates for encapsulated packets to be delivered to Road-Side-Unit LISP xTRs regardless where the roaming-EID is currently positioned.

Referring to Figure 1, the following sequence is performed:

1. The Predictive RLOCs are registered to the mapping system as a LCAF encoded Replication List Entry (RLE) Type [I-D.ietf-lisp-lcaf]. The registration can happen by one or more RSUs or by a third-party. When registered by an RSU, and when no coordination is desired, they each register their own RLOC with merge-semantics so the list can be created and maintained in the LISP Map-Server. When registered by a third-party, the complete list of RLOCs can be included in the RLE.
2. There can be multiple RLEs present each as different RLOC-records so a remote ITR can select one RLOC-record versus the other based in priority and weight policy [RFC6830].
3. When a remote ITR receives a packet destined for a roaming-EID, it encapsulates and replicates to each RLOC in the RLE thereby delivering the packet to the locations the roaming-EID is about to appear. There are some cases where packets will go to locations where the roaming-EID has already been, but see Section 4.2 for packet delivery optimizations.
4. When the ETR resident RSU receives an encapsulated packet, it decapsulates the packet and then determines if the roaming-EID had been previously discovered. If the EID has not been discovered, the ETR drops the packet. Otherwise, the ETR delivers the decapsulated packet on the port interface the roaming-EID was discovered on.

##### 4.1. RLE Encoding

The LCAF [I-D.ietf-lisp-lcaf] Replication List Entry (RLE) will be used to encode the Predictive RLOCs in an RLOC-record for Map-Registers, Map-Reply, and Map-Notify messages [RFC6830].

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
AFI = 16387										Rsvd1										Flags																			
Type = 13										Rsvd2										4 + n																			
Rsvd3										Rsvd4										Level Value																			
AFI = x										RTR/ETR #1 ...																													
Rsvd3										Rsvd4										Level Value																			
AFI = x										RTR/ETR #n ...																													

When the RLOC-record contains an RLE with RLOC entries all with the same level value, it means the physical order listed is the directional path of the RSUs. This will typically be the result of a third-party doing the registration where it knows ahead of time the RSU deployment.

When each RSU is registering with merge-semantics on their own, the level number is used to place them in an ordered list. Since the registrations come at different times and therefore arrive in different order than the physical RSU path, the level number creates the necessary sequencing. Each RSU needs to know its position in the path relative to other RSUs. For example, in xTR-B, it would register with level 1 since it is after xTR-A (and before xTR-C). So if the registration order was xTR-B with level 1, xTR-C with level 2, and xTR-A with level 0, the RLE list stored in the mapping system would be (xTR-A, xTR-B, xTR-C). It is recommended that level numbers be assigned in increments of 10 so latter insertion is possible.

The use of Geo-Prefixes and Geo-Points can be used to compare the physical presence of each RSU with respect to each other, so they can choose level numbers to sequence themselves. Also if the xTRs register with a Geo-Point in an RLOC-record, then perhaps the Map-Server could sequence the RLE list.

#### 4.2. Packet Delivery Optimizations

Since the remote ITR will replicate to all RLOCs in the RLE, a situation is created where packets go to RLOCs that don't need to. For instance, if the roaming-EID is along side of xTR-B and the RLE is (xTR-A, xTR-B, xTR-C), there is no reason to replicate to xTR-A since the roaming-EID has passed it and the the signal range is weak or lost. However, replicating to xTR-B and xTR-C is important to

deliver packets to where the roaming-EID resides and where it is about to go to.

A simple data-plane option, which converges fairly quickly is to have the remote xTR, acting as an ETR, when packets are sent from the roaming-EID, examine the source RLOC in the outer header of the encapsulated packet. If the source RLOC is xTR-B, the remote xTR can determine that the roaming-EID has moved past xTR-A and no longer needs to encapsulate packets to xTR-A's RLOC.

In addition, the remote ITR can use RLOC-probing to determine if each RLOC in the RLE is reachable. And if not reachable, exclude from the list of RLOCs to replicate to.

This solution also handles the case where xTR-A and xTR-B may overlap in radio signal range, but the signal is weak from the roaming-EID to xTR-A but stronger to xTR-B. In this case, the roaming-EID selects xTR-B to send packets that inform the remote xTR that return packets should not be encapsulated to xTR-A.

There are also situations where the RSUs are in signal range of each other in which case they could report reachability status of each other. The use of the Locator-Status-Bits of the LISP encapsulation header could be used to convey this information to the remote xTR. This would only occur when the roaming-EID was discovered by both xTR-A and xTR-B so it was possible for either xTR to reach the roaming-EID. Either an IGP like routing protocol would be required to allow each xTR to know the other could reach the roaming-EID or a path trace tool (i.e. traceroute) could be originated by one xTR targeted for the roaming-EID but MAC-forwarded through the other xTR. These and other roaming-EID reachability mechanisms are work in progress and for further study.

#### 4.3. Trading Off Replication Cost

If RLE lists are large, packet replication can occur to locations well before the roaming-EID arrives. Making RLE lists small is useful without sacrificing hand-off issues or incurring packet loss to the application. By having overlapping RLEs in separate RLOC-records we have a simple mechanism to solve this problem. Here is an example mapping entry to illustrate the point:

```
EID = <roaming-EID>, RLOC-records:
  RLOC = (RLE: xTR-A, xTR-B)
  RLOC = (RLE: xTR-B, xTR-C, xTR-D, xTR-E)
  RLOC = (RLE: xTR-E, xTR-F)
```



When the remote ITR is encapsulating to xTR-B as a decision to use the first RLOC-record, it can decide to move to use the second RLOC-record because xTR-B is the last entry in the first RLOC-record and the first entry in the second RLOC-record. When there are overlapping RLEs, the remote ITR can decide when it is more efficient to switch over. For example, when the roaming-EID is in range of xTR-A, the remote ITR uses the first RLOC-record so the wasted replication cost is to xTR-B only versus a worse cost when using the second RLOC-record. But when the roaming-EID is in range of xTR-B, then replicating to the other xTRs in the second RLOC-record may be crucial if the roaming-EID has increased speed. And when the roaming-EID may be at rest in a parked mode, then the remote ITR encapsulates to only xTR-F using the third RLOC-record since the roaming-EID has moved past xTR-E.

In addition, to eliminate unnecessary replication to xTRs further down a directional path, GEO-prefixes [I-D.farinacci-lisp-geo] can be used so only nearby xTRs that the roaming-EID is about to come in contact with are the only ones to receive encapsulated packets.

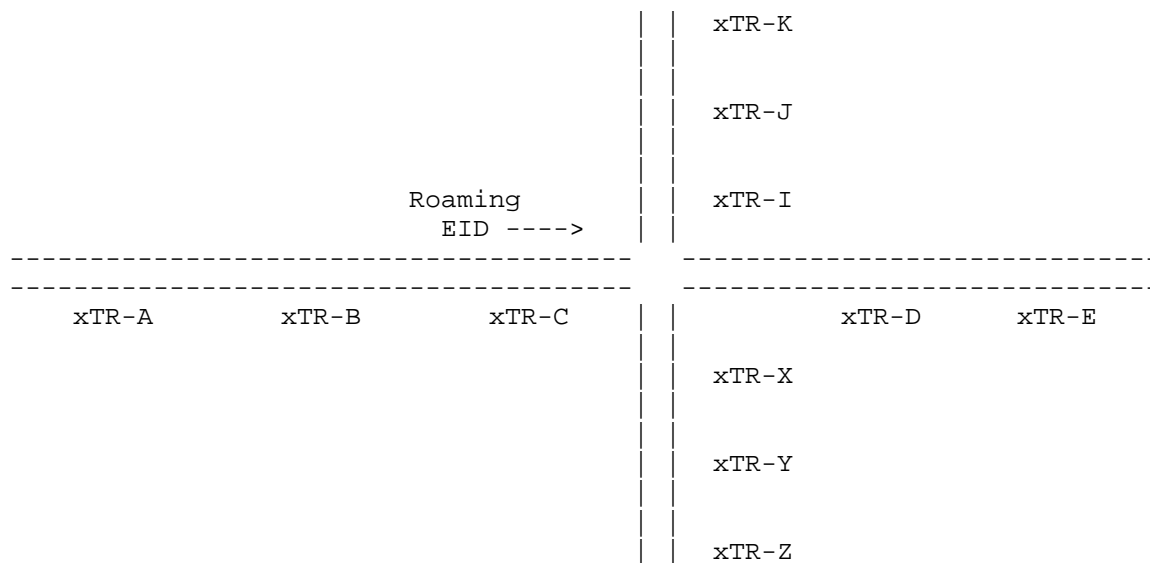
Even when replication lists are not large, we can reduce the cost of replication that the entire network bears by moving the replicator away from the the source (i.e. the ITR) and closer to the RSUs (i.e. the ETRs). See the use of RTRs for Replication Engineering techniques in [I-D.ietf-lisp-signal-free-multicast].

## 5. Directional Paths with Intersections

A roaming-EID could be registered to the mapping system with the following nested RLE mapping:

```
EID = <roaming-EID>, RLOC-records:
  RLOC = (RLE: xTR-A, xTR-B, xTR-C, (RLE: xTR-X, xTR-Y, xTR-Z),
          (RLE: xTR-I, xTR-J, xTR-K), xTR-D, xTR-E)
```

The mapping entry above describes 3 directional paths where the ordered list has encoded one-level of two nested RLEs to denote intersections in a horizontal path. Which is drawn as:



When the roaming-EID is on the horizontal path, the remote-ITRs typically replicate to the rest of the xTRs in the ordered list. When a list has nested RLEs, the replication should occur to at least the first RLOC in a nested RLE list. So if the remote-ITR is replicating to xTR-C, xTR-D, and xTR-E, it should also replicate to xTR-X and xTR-I anticipating a possible turn at the intersection. But when the roaming-EID is known to be at xTR-D (a left or right hand turn was not taken), replication should only occur to xTR-D and xTR-E. Once either xTR-I or xTR-X is determined to be where the roaming-EID resides, then the replication occurs on the respective directional path only.

When nested RLEs are used it may be difficult to get merge-semantics to work when each xTR registers itself. So it is suggested a third-party registers nested RLEs. It is left to further study to understand better how to automate this.

## 6. Multicast Considerations

In this design, the remote ITR is receiving a unicast packet from an EID and replicating and encapsulating to each RLOC in an RLE list. This form of replication is no different than a traditional multicast replication function. So replicating multicast packets in the same fashion is a fallout from this design.

If there are multiple roaming-EIDs joined to the same multicast group but reside at different RSUs, a merge has to be done of any pruned RLEs used for forwarding. So if roaming-EID-1 resides at xTR-A and

roaming-EID-2 resides at xTR-B and the RLE list is (xTR-A, xTR-B, xTR-C), and they are joined to the same multicast group, then replication occurs to all of xTR-A, xTR-B, and xTR-C. Even since roaming-EID-2 is past xTR-A, packets need to be delivered to xTR-A for roaming-EID-1. In addition, packets need to be delivered to xTR-C because roaming-EID-1 and roaming-EID-2 will get to xTR-C (and roaming-EID-1 may get there sooner if it is traveling faster than roaming-EID-2).

When a roaming-EID is a multicast source, procedures from [I-D.ietf-lisp-signal-free-multicast] are used to deliver packets to multicast group members anywhere in the network. The solution requires no signaling to the RSUs. When RSUs receive multicast packets from a roaming-EID, they do a (roaming-EID,G) mapping database lookup to find the replication list of ETRs to encapsulate to.

## 7. Multiple Address-Family Considerations

Note that roaming-EIDs can be assigned IPv6 EID addresses while the RSU xTRs could be using IPv4 RLOC addresses. Any combination of address-families can be supported as well as for multicast packet forwarding, where (S,G) are IPv6 addresses entries and replication is done with IPv4 RLOCs in the outer header.

## 8. Scaling Considerations

One can imagine there will be a large number of roaming-EIDs. So there is a strong desire to efficiently store state in the mapping database and the in remote ITRs map-caches. It is likely, that roaming-EIDs may share the same path and move at the same speed (EID devices on a train) and therefore share the same Predictive RLOCs. And since EIDs are not reassigned for mobility purposes or may be temporal, they will not be topologically aggregatable, so they cannot compress into a single EID-prefix mapping entry that share the same RLOC-set.

By using a level of indirection with the mapping system this problem can be solved. The following mapping entries could exist in the mapping database:

```
EID = <eid1>, RLOC-records:
  RLOC = (afi=<dist-name>: "am-train-to-paris")
EID = <eid2>, RLOC-records:
  RLOC = (afi=<dist-name>: "am-train-to-paris")
EID = <eid3>, RLOC-records:
  RLOC = (afi=<dist-name>: "am-train-to-paris")

EID = "am-train-to-paris", RLOC-records:
  RLOC = (afi=lcaf/RLE-type: xTR-A, xTR-B, xTR-C)

EID = "am-train-to-paris-passengers", RLOC-records:
  RLOC = (afi=lcaf/afi-list-type: <eid1>, <eid2>, <eid3>)
```

Each passenger that boards a train has their EID registered to point to the name of the train "am-train-to-paris". And then the train with EID "am-train-to-paris" stores the Predictive RLOC-set. When a remote-ITR wants to encapsulate packets for an EID, it looks up the EID in the mapping database gets the name "am-train-to-paris" returned. Then the remote-ITR does another lookup for the name "am-train-to-paris" to get the RLE list returned.

When new EIDs board the train, the RLE mapping entry does not need to be modified. Only an EID-to-name mapping is registered for the specific new EID. Optionally, another name "am-train-to-paris-passengers" can be registered as an EID to allow mapping to all specific EIDs which are on the train. This can be used for inventory, billing, or security purposes.

This optimization comes at a cost of a 2-stage lookup. However, if both sets of mapping entries are registered to the same Map-Server, a combined RLOC-set could be returned. This idea is for further study.

## 9. Security Considerations

LISP has procedures for supporting both control-plane security [I-D.ietf-lisp-sec] and data-plane security [I-D.ietf-lisp-crypto].

## 10. IANA Considerations

At this time there are no requests for IANA.

## 11. References

### 11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <<http://www.rfc-editor.org/info/rfc6830>>.

## 11.2. Informative References

- [I-D.farinacci-lisp-geo]  
Farinacci, D., "LISP Geo-Coordinate Use-Cases", draft-farinacci-lisp-geo-03 (work in progress), April 2017.
- [I-D.ietf-lisp-crypto]  
Farinacci, D. and B. Weis, "LISP Data-Plane Confidentiality", draft-ietf-lisp-crypto-10 (work in progress), October 2016.
- [I-D.ietf-lisp-lcaf]  
Farinacci, D., Meyer, D., and J. Snijders, "LISP Canonical Address Format (LCAF)", draft-ietf-lisp-lcaf-22 (work in progress), November 2016.
- [I-D.ietf-lisp-sec]  
Maino, F., Ermagan, V., Cabellos-Aparicio, A., and D. Saucez, "LISP-Security (LISP-SEC)", draft-ietf-lisp-sec-12 (work in progress), November 2016.
- [I-D.ietf-lisp-signal-free-multicast]  
Moreno, V. and D. Farinacci, "Signal-Free LISP Multicast", draft-ietf-lisp-signal-free-multicast-04 (work in progress), May 2017.
- [I-D.portoles-lisp-eid-mobility]  
Portoles-Comeras, M., Ashtaputre, V., Moreno, V., Maino, F., and D. Farinacci, "LISP L2/L3 EID Mobility Using a Unified Control Plane", draft-portoles-lisp-eid-mobility-02 (work in progress), April 2017.

## Appendix A. Acknowledgments

The author would like to thank the LISP WG for their review and acceptance of this draft.

## Appendix B. Document Change Log

[RFC Editor: Please delete this section on publication as RFC.]

## B.1. Changes to draft-ietf-lisp-predictive-rlocs-00.txt

- o Posted June 2017.
- o Make this specification a working group document. It is a copy of draft-farinacci-lisp-predictive-rlocs-02.

## B.2. Changes to draft-farinacci-lisp-predictive-rlocs-02.txt

- o Posted May 2017 to update document timer.

## B.3. Changes to draft-farinacci-lisp-predictive-rlocs-01.txt

- o Posted November 2016 to update document timer.

## B.4. Changes to draft-farinacci-lisp-predictive-rlocs-00.txt

- o Initial post April 2016.

## Authors' Addresses

Dino Farinacci  
lisppers.net  
San Jose, CA  
USA

Email: farinacci@gmail.com

Padma Pillay-Esnault  
Huawei Technologies  
San Clara, CA  
USA

Email: padma@huawei.com

LISP Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: January 4, 2018

V. Ermagan  
A. Rodriguez-Natal  
F. Coras  
C. Moberg  
R. Rahman  
Cisco Systems  
A. Cabellos-Aparicio  
Technical University of Catalonia  
F. Maino  
Cisco Systems  
July 3, 2017

LISP YANG Model  
draft-ietf-lisp-yang-05

Abstract

This document describes a YANG data model to use with the Locator/ID Separation Protocol (LISP).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on January 4, 2018.

Copyright Notice

Copyright (c) 2017 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 Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. LISP Module . . . . .	2
2.1. Module Structure . . . . .	3
2.2. Module Definition . . . . .	5
3. LISP-ITR Module . . . . .	13
3.1. Module Structure . . . . .	13
3.2. Module Definition . . . . .	17
4. LISP-ETR Module . . . . .	20
4.1. Module Structure . . . . .	20
4.2. Module Definition . . . . .	22
5. LISP-Map-Server Module . . . . .	26
5.1. Module Structure . . . . .	26
5.2. Module Definition . . . . .	32
6. LISP-Map-Resolver Module . . . . .	37
6.1. Module Structure . . . . .	37
6.2. Module Definition . . . . .	38
7. LISP-Address-Types Module . . . . .	39
7.1. Module Definition . . . . .	39
8. Acknowledgments . . . . .	54
9. IANA Considerations . . . . .	54
10. Security Considerations . . . . .	54
11. Normative References . . . . .	54
Authors' Addresses . . . . .	55

## 1. Introduction

The Locator/ID Separation Protocol (LISP) defines several network elements subject to be configured. This document presents the YANG data models required for basic configuration of all major LISP [RFC6830] elements. The models also capture some essential operational data elements as well.

## 2. LISP Module

This module is the base LISP module that is augmented in multiple models to represent various LISP device roles.



## 2.1. Module Structure

```
module: ietf-lisp
```

```

  +--rw lisp
    +--rw locator-sets
      +--rw locator-set* [locator-set-name]
        +--rw locator-set-name      string
        +--rw (locator-type)?
          +--:(local-interface)
            +--rw interface* [interface-ref]
              +--rw interface-ref    if:interface-ref
              +--rw priority?         uint8
              +--rw weight?           uint8
              +--rw multicast-priority? uint8
              +--rw multicast-weight? uint8
          +--:(general-locator)
            +--rw locator* [id]
              +--rw id                string
              +--rw locator-address
                +--rw address-type    lisp-address-family-ref
                +--rw virtual-network-id? instance-id-type
                +--rw (address)?
                  +--:(no-address)
                    | +--rw no-address?          empty
                  +--:(ipv4)
                    | +--rw ipv4?                inet:ipv4-address
                  +--:(ipv4-prefix)
                    | +--rw ipv4-prefix?         inet:ipv4-prefix
                  +--:(ipv6)
                    | +--rw ipv6?                inet:ipv6-address
                  +--:(ipv6-prefix)
                    | +--rw ipv6-prefix?         inet:ipv6-prefix
                  +--:(mac)
                    | +--rw mac?                 yang:mac-address
                  +--:(distinguished-name)
                    | +--rw distinguished-name?   distinguished-name
                -type
                  +--:(as-number)
                    | +--rw as-number?           inet:as-number
                  +--:(null-address)
                    | +--rw null-address
                    |   +--rw address?          empty
                  +--:(afi-list)
                    | +--rw afi-list
                    |   +--rw address-list*     simple-address
                  +--:(instance-id)
                    | +--rw instance-id
                    |   +--rw iid?              instance-id-type
                    |   +--rw mask-length?      uint8

```

```

|         +--rw address?          simple-address
+---:(as-number-lcaf)
|         +--rw as-number-lcaf
|         +--rw as?              inet:as-number
|         +--rw address?        simple-address
+---:(application-data)
|         +--rw application-data
|         +--rw address?          simple-address
|         +--rw protocol?         uint8
|         +--rw ip-tos?           int32
|         +--rw local-port-low?   inet:port-number
|         +--rw local-port-high?  inet:port-number
|         +--rw remote-port-low?  inet:port-number
|         +--rw remote-port-high? inet:port-number
+---:(geo-coordinates)
|         +--rw geo-coordinates
|         +--rw latitude?         bits
|         +--rw latitude-degrees? uint8
|         +--rw latitude-minutes? uint8
|         +--rw latitude-seconds? uint8
|         +--rw longitude?        bits
|         +--rw longitude-degrees? uint16
|         +--rw longitude-minutes? uint8
|         +--rw longitude-seconds? uint8
|         +--rw altitude?         int32
|         +--rw address?          simple-address
+---:(nat-traversal)
|         +--rw nat-traversal
|         +--rw ms-udp-port?      uint16
|         +--rw etr-udp-port?     uint16
|         +--rw global-etr-rloc?  simple-address
|         +--rw ms-rloc?          simple-address
|         +--rw private-etr-rloc? simple-address
|         +--rw rtr-rlocs*        simple-address
+---:(explicit-locator-path)
|         +--rw explicit-locator-path
|         +--rw hop* [hop-id]
|         +--rw hop-id           string
|         +--rw address?         simple-address
|         +--rw lrs-bits?        bits
+---:(source-dest-key)
|         +--rw source-dest-key
|         +--rw source?          simple-address
|         +--rw dest?            simple-address
+---:(key-value-address)
|         +--rw key-value-address
|         +--rw key?             simple-address
|         +--rw value?           simple-address

```

```

|                                     +--:(service-path)
|                                     +--rw service-path
|                                     +--rw service-path-id?  service-path-id-type
|                                     +--rw service-index?    uint8
|                                     +--rw priority?         uint8
|                                     +--rw weight?          uint8
|                                     +--rw multicast-priority? uint8
|                                     +--rw multicast-weight?  uint8
+--rw lisp-router-instances
  +--rw lisp-router-instance* [lisp-router-instance-id]
    +--rw lisp-router-instance-id  int32
    +--rw lisp-role* [lisp-role-type]
      | +--rw lisp-role-type  lisp-role-ref
    +--rw lisp-router-id
      +--rw site-id?  uint64
      +--rw xtr-id?   lisp:xtr-id-type

```

## 2.2. Module Definition

```

<CODE BEGINS> file "ietf-lisp@2017-07-01.yang"
module ietf-lisp {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp";
  prefix lisp;
  import ietf-interfaces {
    prefix if;
  }
  import ietf-lisp-address-types {
    prefix lacf;
  }
  import ietf-yang-types {
    prefix yang;
  }
  organization
    "IETF LISP (Locator/ID Separation Protocol) Working Group";
  contact
    "lisp@ietf.org";
  description
    "This YANG module defines the generic parameters for LISP.
     The module can be extended by vendors to define vendor-specific
     LISP parameters and policies."

```

Copyright (c) 2014 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions

Relating to IETF Documents  
(<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC 6338; see  
the RFC itself for full legal notices.  
";

```
revision 2017-07-01 {
  description
    "Initial revision.";
  reference
    "https://tools.ietf.org/html/rfc6830";
}
identity lisp-role {
  description
    "LISP router role.";
}
identity itr {
  base lisp-role;
  description
    "LISP ITR.";
}
identity pitr {
  base lisp-role;
  description
    "LISP PITR.";
}
identity etr {
  base lisp-role;
  description
    "LISP ETR.";
}
identity petr {
  base lisp-role;
  description
    "LISP PETR.";
}
identity mapping-system {
  description
    "Mapping System interface";
}
identity single-node-mapping-system {
  base mapping-system;
  description
    "logically singular Map Server";
}
typedef mapping-system-ref {
  type identityref {
```

```
    base mapping-system;
  }
  description
    "Mapping System reference";
}

typedef lisp-role-ref {
  type identityref {
    base lisp-role;
  }
  description
    "LISP role reference";
}

typedef map-reply-action {
  type enumeration {
    enum no-action {
      value 0;
      description
        "Mapping is kept alive and no encapsulation occurs.";
    }
    enum natively-forward {
      value 1;
      description
        "Matching packets are not encapsulated or dropped but
        natively forwarded.";
    }
    enum send-map-request {
      value 2;
      description
        "Matching packets invoke Map-Requests.";
    }
    enum drop {
      value 3;
      description
        "Matching packets are dropped.";
    }
  }
  description
    "Defines the lisp map-cache ACT type";
  reference "https://tools.ietf.org/html/rfc6830#section-6.1.4";
}

typedef eid-id {
  type string;
  description
    "Type encoding of lisp-addresses to be generally used in EID
    keyed lists.";
}

typedef auth-key-type {
```

```
type enumeration {
  enum none {
    value 0;
    description
      "No authentication.";
  }
  enum hmac-sha-1-96 {
    value 1;
    description
      "HMAC-SHA-1-96 (RFC2404) authentication is used.";
  }
  enum hmac-sha-256-128 {
    value 2;
    description
      "HMAC-SHA-256-128 (RFC4868) authentication is used.";
  }
}
description
  "Enumeration of the authentication mechanisms supported by
  LISP.";
reference
  "https://tools.ietf.org/html/rfc6830#section-6.1.6";
}
typedef xtr-id-type {
  type binary {
    length "16";
  }
  description
    "128 bit xTR identifier.";
}

grouping locator-properties {
  description
    "Properties of a RLOC";
  leaf priority {
    type uint8;
    description
      "Locator priority.";
  }
  leaf weight {
    type uint8;
    description
      "Locator weight.";
  }
  leaf multicast-priority {
    type uint8;
    description
      "Locator's multicast priority";
  }
}
```

```
    }
    leaf multicast-weight {
      type uint8;
      description
        "Locator's multicast weight";
    }
  }

  grouping locators-grouping {
    description
      "Group that defines a list of LISP locators.";
    list locator {
      key "id";
      description
        "List of routing locators";
      leaf id {
        type string {
          length "1..64";
        }
        description
          "Locator id";
      }
      container locator-address {
        uses lcaf:lisp-address;
        description
          "The locator address provided in LISP canonincal
            address format.";
      }
      uses locator-properties;
    }
  }

  grouping local-locators-grouping {
    description
      "Group that defines a list of LISP locators.";
    list interface {
      key "interface-ref";
      description
        "The address type of the locator";
      leaf interface-ref {
        type if:interface-ref;
        description
          "The name of the interface supporting the locator.";
      }
      uses locator-properties;
    }
  }
}
```

```
grouping mapping {
  description
    "Group that defines a LISP mapping.";
  container eid {
    uses lcaf:lisp-address;
    description
      "End-host Identifier (EID) to be mapped to a list of
      locators";
  }
  leaf time-to-live {
    type uint32;
    units minutes;
    description
      "Mapping validity period in minutes.";
  }
  leaf creation-time {
    type yang:date-and-time;
    description
      "Time when the mapping was created.";
  }
  leaf authoritative {
    type bits {
      bit A {
        description
          "Authoritative bit.";
      }
    }
    description
      "Bit that indicates if mapping comes from an
      authoritative source.";
  }
  leaf static {
    type boolean;
    default "false";
    description
      "This leaf should be true if the mapping is static.";
  }
  choice locator-list {
    description
      "list of locartors are either negative, or positive.";
    case negative-mapping {
      leaf map-reply-action {
        type map-reply-action;
        description
          "Forwarding action for a negative mapping.";
      }
    }
    case positive-mapping {
```



```
        container rlocs {
            uses locators-grouping;
            description
                "List of locators for a positive mapping.";
        }
    }
}

grouping mappings {
    description
        "Group that defines a list of LISP mappings.";
    list virtual-network {
        key "vni";
        description
            "Virtual network to which the mappings belong.";
        leaf vni {
            type lcaf:instance-id-type;
            description
                "Virtual network identifier.";
        }
        container mappings {
            description
                "Mappings within the virtual network.";
            list mapping {
                key "id";
                description
                    "List of EID to RLOCs mappings.";
                leaf id {
                    type eid-id;
                    description
                        "Id that uniquely identifies a mapping.";
                }
                uses mapping;
            }
        }
    }
}

container lisp {
    description
        "Parameters for the LISP subsystem.";

    container locator-sets {
        description
            "Container that defines a named locator set which can be
            referenced elsewhere.";
        list locator-set {
```

```
    key "locator-set-name";
    description
        "Multiple locator sets can be defined.";
    leaf locator-set-name {
        type string {
            length "1..64";
        }
        description
            "Locator set name";
    }
    choice locator-type {
        description
            "Locator sets can be based on local interfaces, or
            general locators.";
        case local-interface {
            uses local-locators-grouping;
            description
                "List of locators in this set based on local
                interfaces.";
        }
        case general-locator {
            uses locators-grouping;
            description
                "List of locators in this set based on lisp-address.";
        }
    }
}

container lisp-router-instances {
    description
        "Different LISP routers instantiated in the device";
    list lisp-router-instance {
        key "lisp-router-instance-id";
        description
            "Each entry contains parameters for a LISP router.";
        leaf lisp-router-instance-id {
            type int32;
            description
                "Arbitrary lisp-router id.";
        }
        list lisp-role {
            key lisp-role-type;
            description
                "List of lisp device roles such as MS, MR, ITR,
                PITR, ETR or PETR.";
            leaf lisp-role-type {
                type lisp-role-ref;
            }
        }
    }
}
```

```

        description
            "The type of LISP device - identity derived from the
             'lisp-device' base identity.";
    }
}
container lisp-router-id {
    when "../lisp-role/lisp-role-type = 'itr' or
         ../lisp-role/lisp-role-type = 'pitrr' or
         ../lisp-role/lisp-role-type = 'etr' or
         ../lisp-role/lisp-role-type = 'petrr'" {
        description "Only when ITR, PITR, ETR or PETR.";
    }
    description
        "Site-ID and xTR-ID of the device.";
    leaf site-id {
        type uint64;
        description "Site ID";
    }
    leaf xtr-id {
        type lisp:xtr-id-type;
        description "xTR ID";
    }
}
}
}
}
}
<CODE ENDS>
```

### 3. LISP-ITR Module

This module captures the configuration data model of a LISP ITR. The model also captures some operational data elements.

### 3.1. Module Structure

```

module: ietf-lisp-itr
augment /lisp:lisp/lisp:lisp-router-instances/lisp:lisp-router-instance:
  +--rw itr!
    +--rw rloc-probing!
      | +--rw interval?          uint16
      | +--rw retries?          uint8
      | +--rw retries-interval?  uint16
    +--rw itr-rlocs?            -> /lisp:lisp/locator-sets/locator-set/locator-set
-name
  +--rw map-resolvers
    | +--rw map-resolver*      inet:ip-address
  +--rw proxy-etrs
    | +--rw proxy-etr-address*  inet:ip-address

```

```

+--rw map-cache
  +--rw virtual-network* [vni]
    +--rw vni          lcaf:instance-id-type
    +--rw mappings
      +--rw mapping* [id]
        +--rw id          eid-id
        +--rw eid
          +--rw address-type          lisp-address-family-ref
          +--rw virtual-network-id?   instance-id-type
          +--rw (address)?
            +--:(no-address)
              | +--rw no-address?          empty
            +--:(ipv4)
              | +--rw ipv4?          inet:ipv4-address
            +--:(ipv4-prefix)
              | +--rw ipv4-prefix?    inet:ipv4-prefix
            +--:(ipv6)
              | +--rw ipv6?          inet:ipv6-address
            +--:(ipv6-prefix)
              | +--rw ipv6-prefix?    inet:ipv6-prefix
            +--:(mac)
              | +--rw mac?            yang:mac-address
            +--:(distinguished-name)
              | +--rw distinguished-name? distinguished-name-ty
          +--:(as-number)
            | +--rw as-number?          inet:as-number
          +--:(null-address)
            | +--rw null-address
            |   +--rw address?          empty
          +--:(afi-list)
            | +--rw afi-list
            |   +--rw address-list*    simple-address
          +--:(instance-id)
            | +--rw instance-id
            |   +--rw iid?              instance-id-type
            |   +--rw mask-length?     uint8
            |   +--rw address?         simple-address
          +--:(as-number-lcaf)
            | +--rw as-number-lcaf
            |   +--rw as?               inet:as-number
            |   +--rw address?         simple-address
          +--:(application-data)
            | +--rw application-data
            |   +--rw address?          simple-address
            |   +--rw protocol?         uint8
            |   +--rw ip-tos?           int32
            |   +--rw local-port-low?   inet:port-number
            |   +--rw local-port-high?  inet:port-number

```

```

|         +--rw remote-port-low?      inet:port-number
|         +--rw remote-port-high?     inet:port-number
+---:(geo-coordinates)
|   +--rw geo-coordinates
|     +--rw latitude?                  bits
|     +--rw latitude-degrees?          uint8
|     +--rw latitude-minutes?          uint8
|     +--rw latitude-seconds?          uint8
|     +--rw longitude?                 bits
|     +--rw longitude-degrees?          uint16
|     +--rw longitude-minutes?          uint8
|     +--rw longitude-seconds?          uint8
|     +--rw altitude?                  int32
|     +--rw address?                   simple-address
+---:(nat-traversal)
|   +--rw nat-traversal
|     +--rw ms-udp-port?                uint16
|     +--rw etr-udp-port?                uint16
|     +--rw global-etr-rloc?             simple-address
|     +--rw ms-rloc?                     simple-address
|     +--rw private-etr-rloc?            simple-address
|     +--rw rtr-rlocs*                   simple-address
+---:(explicit-locator-path)
|   +--rw explicit-locator-path
|     +--rw hop* [hop-id]
|       +--rw hop-id                     string
|       +--rw address?                     simple-address
|       +--rw lrs-bits?                     bits
+---:(source-dest-key)
|   +--rw source-dest-key
|     +--rw source?                       simple-address
|     +--rw dest?                         simple-address
+---:(key-value-address)
|   +--rw key-value-address
|     +--rw key?                           simple-address
|     +--rw value?                         simple-address
+---:(service-path)
|   +--rw service-path
|     +--rw service-path-id?              service-path-id-type
|     +--rw service-index?                 uint8
+--rw time-to-live?                       uint32
+--rw creation-time?                       yang:date-and-time
+--rw authoritative?                       bits
+--rw static?                             boolean
+--rw (locator-list)?
|   +---:(negative-mapping)
|   |   +--rw map-reply-action?           map-reply-action
|   +---:(positive-mapping)

```

```

+--rw rlocs
  +--rw locator* [id]
    +--rw id string
    +--rw locator-address
      | +--rw address-type lisp-address-fa
mily-ref
      | +--rw virtual-network-id? instance-id-ty
e
      | +--rw (address)?
      |   +--:(no-address)
      |     | +--rw no-address? empty
      |   +--:(ipv4)
      |     | +--rw ipv4? inet:ipv4
-address
      |   +--:(ipv4-prefix)
      |     | +--rw ipv4-prefix? inet:ipv4
-prefix
      |   +--:(ipv6)
      |     | +--rw ipv6? inet:ipv6
-address
      |   +--:(ipv6-prefix)
      |     | +--rw ipv6-prefix? inet:ipv6
-prefix
      |   +--:(mac)
      |     | +--rw mac? yang:mac-
address
      |   +--:(distinguished-name)
      |     | +--rw distinguished-name? distingui
shed-name-type
      |   +--:(as-number)
      |     | +--rw as-number? inet:as-n
umber
      |   +--:(null-address)
      |     | +--rw null-address
      |     |   +--rw address? empty
      |   +--:(afi-list)
      |     | +--rw afi-list
      |     |   +--rw address-list* simple-address
      |   +--:(instance-id)
      |     | +--rw instance-id
      |     |   +--rw iid? instance-id-type
      |     |   +--rw mask-length? uint8
      |     |   +--rw address? simple-address
      |   +--:(as-number-lcaf)
      |     | +--rw as-number-lcaf
      |     |   +--rw as? inet:as-number
      |     |   +--rw address? simple-address
      |   +--:(application-data)
      |     | +--rw application-data
      |     |   +--rw address? simple-addr
ess
      |   +--rw protocol? uint8
      |   +--rw ip-tos? int32
      |   +--rw local-port-low? inet:port-n
umber
      |   +--rw local-port-high? inet:port-n
umber
      |   +--rw remote-port-low? inet:port-n
umber
      |   +--rw remote-port-high? inet:port-n
umber
      |   +--:(geo-coordinates)

```



```

+---rw geo-coordinates
+---rw latitude? bits
+---rw latitude-degrees? uint8
+---rw latitude-minutes? uint8
+---rw latitude-seconds? uint8
+---rw longitude? bits
+---rw longitude-degrees? uint16
+---rw longitude-minutes? uint8
+---rw longitude-seconds? uint8
+---rw altitude? int32
+---rw address? simple-add

ress

+---:(nat-traversal)
+---rw nat-traversal
+---rw ms-udp-port? uint16
+---rw etr-udp-port? uint16
+---rw global-etr-rloc? simple-addr

ess

+---rw ms-rloc? simple-addr

ess

+---rw private-etr-rloc? simple-addr

ess

+---rw rtr-rlocs* simple-addr

+---:(explicit-locator-path)
+---rw explicit-locator-path
+---rw hop* [hop-id]
+---rw hop-id string
+---rw address? simple-address
+---rw lrs-bits? bits
+---:(source-dest-key)
+---rw source-dest-key
+---rw source? simple-address
+---rw dest? simple-address
+---:(key-value-address)
+---rw key-value-address
+---rw key? simple-address
+---rw value? simple-address
+---:(service-path)
+---rw service-path
+---rw service-path-id? service-path

-id-type

+---rw service-index? uint8
+---rw priority? uint8
+---rw weight? uint8
+---rw multicast-priority? uint8
+---rw multicast-weight? uint8

```

### 3.2. Module Definition

```
<CODE BEGINS> file "ietf-lisp-itr@2017-07-01.yang"
module ietf-lisp-itr {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp-itr";
  prefix lisp-itr;
```



```
import ietf-lisp {
  prefix lisp;
}
import ietf-inet-types {
  prefix inet;
}
organization
  "IETF LISP (Locator/ID Separation Protocol) Working Group";
contact
  "lisp@ietf.org";
description
  "This YANG module defines the generic parameters for a LISP
  ITR. The module can be extended by vendors to define
  vendor-specific parameters and policies."
```

Copyright (c) 2015 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC 6338; see the RFC itself for full legal notices.

```
";
revision 2017-07-01 {
  description
    "Initial revision.";
  reference
    "https://tools.ietf.org/html/rfc6830";
}
augment "/lisp:lisp/lisp:lisp-router-instances/lisp:lisp-router-instance" {
  when "lisp:lisp-role/lisp:lisp-role-type = 'lisp:itr' or
        lisp:lisp-role/lisp:lisp-role-type = 'lisp:pitr'" {
    description
      "Augment is valid when LISP role type is ITR or PITR.";
  }
  description
    "This augments LISP devices list with (P)ITR specific
    parameters.";
  container itr {
    presence "LISP (P)ITR operation enabled";
    description
      "ITR parameters";
    container rloc-probing {
```

```
presence "RLOC probing active";
description
  "RLOC-probing parameters";
leaf interval {
  type uint16;
  units "seconds";
  description
    "Interval in seconds for resending the probes";
}
leaf retries {
  type uint8;
  description
    "Number of retries for sending the probes";
}
leaf retries-interval {
  type uint16;
  units "seconds";
  description
    "Interval in seconds between retries when sending probes.
    The action taken if all retries fail to receive is
    impementation specific.";
}
}
leaf itr-rlocs {
  type leafref {
    path "/lisp:lisp/lisp:locator-sets/lisp:locator-set/"
      + "lisp:locator-set-name";
  }
  description
    "Reference to a locator set that the (P)ITR includes in
    Map-Requests";
}
container map-resolvers {
  description
    "Map-Resolvers that the (P)ITR uses.";
  leaf-list map-resolver {
    type inet:ip-address;
    min-elements 1;
    description
      "Each Map-Resolver within the list of Map-Resolvers.";
  }
}
container proxy-etr {
  when "../lisp:lisp-role/lisp:lisp-role-type = 'lisp:itr'" {
    description
      "Container exists only when LISP role type is ITR";
  }
  description

```

```

        "Proxy ETRs that the ITR uses.";
    leaf-list proxy-etr-address{
        type inet:ip-address;
        description
            "Proxy ETR RLOC address.";
    }
}
container map-cache{
    uses lisp:mappings;
    description
        "EID to RLOCs mappings cache.";
}
}
}
}
<CODE ENDS>

```

#### 4. LISP-ETR Module

This module captures the configuration data model of a LISP ETR. The model also captures some operational data elements.

##### 4.1. Module Structure

```

module: ietf-lisp-etr
augment /lisp:lisp/lisp:lisp-router-instances/lisp:lisp-router-instance:
  +--rw etr!
    +--rw map-servers
      |   +--rw map-server* [ms-address]
      |   |   +--rw ms-address          inet:ip-address
      |   |   +--rw auth-key?           string
      |   |   +--rw auth-key-type?      lisp:auth-key-type
    +--rw local-eids
      +--rw virtual-network* [vni]
      +--rw vni              lcaf:instance-id-type
      +--rw eids
        +--rw local-eid* [id]
          +--rw id                      lisp:eid-id
          +--rw eid-address
            +--rw address-type          lisp-address-family-ref
            +--rw virtual-network-id?   instance-id-type
            +--rw (address)?
            |   +--:(no-address)
            |   |   +--rw no-address?   empty
            |   +--:(ipv4)
            |   |   +--rw ipv4?         inet:ipv4-address
            |   +--:(ipv4-prefix)
            |   |   +--rw ipv4-prefix?  inet:ipv4-prefix

```

pe

```

+---:(ipv6)
|   +--rw ipv6?                               inet:ipv6-address
+---:(ipv6-prefix)
|   +--rw ipv6-prefix?                         inet:ipv6-prefix
+---:(mac)
|   +--rw mac?                                yang:mac-address
+---:(distinguished-name)
|   +--rw distinguished-name?                  distinguished-name-ty

+---:(as-number)
|   +--rw as-number?                           inet:as-number
+---:(null-address)
|   +--rw null-address
|       +--rw address?    empty
+---:(afi-list)
|   +--rw afi-list
|       +--rw address-list*    simple-address
+---:(instance-id)
|   +--rw instance-id
|       +--rw iid?              instance-id-type
|       +--rw mask-length?      uint8
|       +--rw address?           simple-address
+---:(as-number-lcaf)
|   +--rw as-number-lcaf
|       +--rw as?                inet:as-number
|       +--rw address?           simple-address
+---:(application-data)
|   +--rw application-data
|       +--rw address?            simple-address
|       +--rw protocol?           uint8
|       +--rw ip-tos?             int32
|       +--rw local-port-low?     inet:port-number
|       +--rw local-port-high?    inet:port-number
|       +--rw remote-port-low?    inet:port-number
|       +--rw remote-port-high?   inet:port-number
+---:(geo-coordinates)
|   +--rw geo-coordinates
|       +--rw latitude?           bits
|       +--rw latitude-degrees?   uint8
|       +--rw latitude-minutes?   uint8
|       +--rw latitude-seconds?   uint8
|       +--rw longitude?          bits
|       +--rw longitude-degrees?   uint16
|       +--rw longitude-minutes?   uint8
|       +--rw longitude-seconds?   uint8
|       +--rw altitude?           int32
|       +--rw address?            simple-address
+---:(nat-traversal)
|   +--rw nat-traversal

```

```

|         +--rw ms-udp-port?          uint16
|         +--rw etr-udp-port?         uint16
|         +--rw global-etr-rloc?      simple-address
|         +--rw ms-rloc?              simple-address
|         +--rw private-etr-rloc?     simple-address
|         +--rw rtr-rlocs*            simple-address
|     +---:(explicit-locator-path)
|     |   +--rw explicit-locator-path
|     |   |   +--rw hop* [hop-id]
|     |   |   |   +--rw hop-id        string
|     |   |   |   +--rw address?      simple-address
|     |   |   |   +--rw lrs-bits?     bits
|     |   +---:(source-dest-key)
|     |   |   +--rw source-dest-key
|     |   |   |   +--rw source?       simple-address
|     |   |   |   +--rw dest?        simple-address
|     |   +---:(key-value-address)
|     |   |   +--rw key-value-address
|     |   |   |   +--rw key?          simple-address
|     |   |   |   +--rw value?       simple-address
|     |   +---:(service-path)
|     |   |   +--rw service-path
|     |   |   |   +--rw service-path-id?  service-path-id-type
|     |   |   |   +--rw service-index?    uint8
|     +--rw rlocs?                        -> /lisp:lisp/locator-sets/loc
ator-set/locator-set-name
|     +--rw record-ttl?                   uint32
|     +--rw want-map-notify?             boolean
|     +--rw proxy-reply?                 boolean
|     +--rw registration-interval?       uint16

```

#### 4.2. Module Definition

```

<CODE BEGINS> file "ietf-lisp-etr@2017-07-01.yang"
module ietf-lisp-etr {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp-etr";
  prefix lisp-etr;
  import ietf-lisp {
    prefix lisp;
  }
  import ietf-lisp-address-types {
    prefix lacf;
  }
  import ietf-inet-types {
    prefix inet;
  }
  organization
    "IETF LISP (Locator/ID Separation Protocol) Working Group";
  contact

```

```
"lisp@ietf.org";
description
  "This YANG module defines the generic parameters for a LISP
  ETR. The module can be extended by vendors to define
  vendor-specific parameters and policies.

  Copyright (c) 2015 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC 6338; see
  the RFC itself for full legal notices.
  ";
revision 2017-07-01 {
  description
    "Initial revision.";
  reference
    "https://tools.ietf.org/html/rfc6830";
}
augment "/lisp:lisp/lisp:lisp-router-instances/lisp:lisp-router-instance" {
  when "lisp:lisp-role/lisp:lisp-role-type = 'lisp:etr' or
        lisp:lisp-role/lisp:lisp-role-type = 'lisp:petr'" {
    description
      "Augment is valid when LISP device type is (P)ETR.";
  }
  description
    "This augments LISP devices list with (P)ETR specific
    parameters.";
  container etr {
    presence "LISP (P)ETR operation enabled";
    description
      "(P)ETR parameters.";

    container map-servers {
      when "../lisp:lisp-role/lisp:lisp-role-type = 'lisp:etr'" {
        description
          "Container exists only when LISP device type is ETR.";
      }
      description
        "Map-Servers that the ETR uses.";
      list map-server {
        key "ms-address";
```

```
description
  "Each Map-Server within the list of Map-Servers.";
leaf ms-address {
  type inet:ip-address;
  description
    "Map-Server address.";
}
leaf auth-key {
  type string;
  description
    "Map-Server authentication key.";
}
leaf auth-key-type {
  type lisp:auth-key-type;
  description
    "Map-Server authentication type.";
}
}
}

container local-eids {
  when "../lisp:lisp-role/lisp:lisp-role-type = 'lisp:etr'" {
    description
      "Container exists only when LISP device type is ETR.";
  }
  description
    "Virtual networks served by the ETR.";
  list virtual-network {
    key "vni";
    description
      "Virtual network for local-EIDs.";
    leaf vni {
      type lisp:instance-id-type;
      description
        "Virtual network identifier.";
    }
  }
  container eids {
    description
      "EIDs served by the ETR.";
    list local-eid {
      key "id";
      min-elements 1;
      description
        "List of local EIDs.";
      leaf id {
        type lisp:eid-id;
        description
          "Unique id of local EID.";
      }
    }
  }
}
```

```

    }
    container eid-address {
        uses lcaf:lisp-address;
        description
            "EID address in generic LISP address format.";
    }
    leaf rlocs {
        type leafref {
            path "/lisp:lisp/lisp:locator-sets/lisp:locator-set/"
                + "lisp:locator-set-name";
        }
        description
            "Locator set mapped to this local EID.";
    }
    leaf record-ttl {
        type uint32;
        units minutes;
        description
            "Validity period of the EID to RLOCs mapping provided
            in Map-Replies.";
    }
    leaf want-map-notify {
        type boolean;
        default "true";
        description
            "Flag which if set in a Map-Register requests that a
            Map-Notify be sent in response.";
    }
    leaf proxy-reply {
        type boolean;
        default "false";
        description
            "Flag which if set in a Map-Register requests that the
            Map-Server proxy Map-Replies for the ETR.";
    }
    leaf registration-interval {
        type uint16;
        units "seconds";
        default "60";
        description
            "Interval between consecutive Map-Register messages.";
    }
}
}
}
}
}
}

```



```

}
<CODE ENDS>

```

## 5. LISP-Map-Server Module

This module captures the configuration data model of a LISP Map Server [RFC6833]. The model also captures some operational data elements.

### 5.1. Module Structure

```

module: ietf-lisp-mapserver
augment /lisp:lisp/lisp-router-instances/lisp:lisp-router-instance:
  +--rw map-server!
    +--rw sites
      +--rw site* [site-id]
        +--rw site-id      uint64
        +--rw auth-key
          +--rw auth-key-value?  string
          +--rw auth-key-type*   lisp:auth-key-type
    +--rw virtual-network-ids
      +--rw virtual-network-identifier* [vni]
        +--rw vni          lcaf:instance-id-type
        +--rw mappings
          +--rw mapping* [eid-id]
            +--rw eid-id          lisp:eid-id
            +--rw eid-address
              +--rw address-type   lisp-address-family-ref
              +--rw virtual-network-id?  instance-id-type
              +--rw (address)?
                +--:(no-address)
                | +--rw no-address?      empty
                +--:(ipv4)
                | +--rw ipv4?            inet:ipv4-address
                +--:(ipv4-prefix)
                | +--rw ipv4-prefix?     inet:ipv4-prefix
                +--:(ipv6)
                | +--rw ipv6?            inet:ipv6-address
                +--:(ipv6-prefix)
                | +--rw ipv6-prefix?     inet:ipv6-prefix
                +--:(mac)
                | +--rw mac?             yang:mac-address
                +--:(distinguished-name)
                | +--rw distinguished-name? distinguished-name-ty
                |
                | +--:(as-number)
                | | +--rw as-number?     inet:as-number
                +--:(null-address)
                | +--rw null-address

```

```

|         +--rw address?    empty
+---:(afi-list)
|   +--rw afi-list
|     +--rw address-list*   simple-address
+---:(instance-id)
|   +--rw instance-id
|     +--rw iid?            instance-id-type
|     +--rw mask-length?   uint8
|     +--rw address?       simple-address
+---:(as-number-lcaf)
|   +--rw as-number-lcaf
|     +--rw as?            inet:as-number
|     +--rw address?       simple-address
+---:(application-data)
|   +--rw application-data
|     +--rw address?       simple-address
|     +--rw protocol?      uint8
|     +--rw ip-tos?        int32
|     +--rw local-port-low? inet:port-number
|     +--rw local-port-high? inet:port-number
|     +--rw remote-port-low? inet:port-number
|     +--rw remote-port-high? inet:port-number
+---:(geo-coordinates)
|   +--rw geo-coordinates
|     +--rw latitude?       bits
|     +--rw latitude-degrees? uint8
|     +--rw latitude-minutes? uint8
|     +--rw latitude-seconds? uint8
|     +--rw longitude?      bits
|     +--rw longitude-degrees? uint16
|     +--rw longitude-minutes? uint8
|     +--rw longitude-seconds? uint8
|     +--rw altitude?       int32
|     +--rw address?       simple-address
+---:(nat-traversal)
|   +--rw nat-traversal
|     +--rw ms-udp-port?    uint16
|     +--rw etr-udp-port?   uint16
|     +--rw global-etr-rloc? simple-address
|     +--rw ms-rloc?        simple-address
|     +--rw private-etr-rloc? simple-address
|     +--rw rtr-rlocs*      simple-address
+---:(explicit-locator-path)
|   +--rw explicit-locator-path
|     +--rw hop* [hop-id]
|       +--rw hop-id        string
|       +--rw address?       simple-address
|       +--rw lrs-bits?      bits

```

			<pre> +---:(source-dest-key)     +--rw source-dest-key         +--rw source?    simple-address         +--rw dest?      simple-address +---:(key-value-address)     +--rw key-value-address         +--rw key?       simple-address         +--rw value?     simple-address +---:(service-path)     +--rw service-path         +--rw service-path-id?  service-path-id-type         +--rw service-index?    uint8 +--rw site-id*                  uint64 +--rw more-specifics-accepted?  boolean +--rw mapping-expiration-timeout? int16 +--rw mapping-records     +--rw mapping-record* [xtr-id]         +--rw xtr-id          lisp:xtr-id-type         +--rw site-id?        uint64         +--rw eid             +--rw address-type          lisp-address-family-r </pre>
ef			<pre> +--rw virtual-network-id?  instance-id-type +--rw (address)?     +---:(no-address)         +--rw no-address?          empty     +---:(ipv4)         +--rw ipv4?                inet:ipv4-addre </pre>
ss			
x			<pre> +---:(ipv4-prefix)     +--rw ipv4-prefix?          inet:ipv4-prefi </pre>
ss			<pre> +---:(ipv6)     +--rw ipv6?                inet:ipv6-addre </pre>
x			<pre> +---:(ipv6-prefix)     +--rw ipv6-prefix?          inet:ipv6-prefi </pre>
s			<pre> +---:(mac)     +--rw mac?                  yang:mac-addres </pre>
ame-type			<pre> +---:(distinguished-name)     +--rw distinguished-name?    distinguished-n </pre>
			<pre> +---:(as-number)     +--rw as-number?             inet:as-number +---:(null-address)     +--rw null-address         +--rw address?          empty +---:(afi-list)     +--rw afi-list         +--rw address-list*     simple-address +---:(instance-id)     +--rw instance-id         +--rw iid?              instance-id-type </pre>

```

|         +--rw mask-length?    uint8
|         +--rw address?        simple-address
+---:(as-number-lcaf)
|         +--rw as-number-lcaf
|         +--rw as?             inet:as-number
|         +--rw address?        simple-address
+---:(application-data)
|         +--rw application-data
|         +--rw address?        simple-address
|         +--rw protocol?       uint8
|         +--rw ip-tos?         int32
|         +--rw local-port-low? inet:port-number
|         +--rw local-port-high? inet:port-number
|         +--rw remote-port-low? inet:port-number
|         +--rw remote-port-high? inet:port-number
+---:(geo-coordinates)
|         +--rw geo-coordinates
|         +--rw latitude?       bits
|         +--rw latitude-degrees? uint8
|         +--rw latitude-minutes? uint8
|         +--rw latitude-seconds? uint8
|         +--rw longitude?      bits
|         +--rw longitude-degrees? uint16
|         +--rw longitude-minutes? uint8
|         +--rw longitude-seconds? uint8
|         +--rw altitude?       int32
|         +--rw address?        simple-address
+---:(nat-traversal)
|         +--rw nat-traversal
|         +--rw ms-udp-port?     uint16
|         +--rw etr-udp-port?    uint16
|         +--rw global-etr-rloc? simple-address
|         +--rw ms-rloc?         simple-address
|         +--rw private-etr-rloc? simple-address
|         +--rw rtr-rlocs*       simple-address
+---:(explicit-locator-path)
|         +--rw explicit-locator-path
|         +--rw hop* [hop-id]
|         |         +--rw hop-id    string
|         |         +--rw address?  simple-address
|         |         +--rw lrs-bits? bits
+---:(source-dest-key)
|         +--rw source-dest-key
|         +--rw source?          simple-address
|         +--rw dest?            simple-address
+---:(key-value-address)
|         +--rw key-value-address
|         +--rw key?             simple-address

```

					+--rw value?	simple-address
					+++:(service-path)	
					+--rw service-path	
					+--rw service-path-id?	service-path-id-ty
pe					+--rw service-index?	uint8
					+++rw time-to-live?	uint32
					+++rw creation-time?	yang:date-and-time
					+++rw authoritative?	bits
					+++rw static?	boolean
					+++rw (locator-list)?	
					+++:(negative-mapping)	
					+--rw map-reply-action?	map-reply-action
					+++:(positive-mapping)	
					+--rw rlocs	
					+--rw locator* [id]	
					+--rw id	string
					+--rw locator-address	
ess-family-ref					+--rw address-type	lisp-addr
id-type					+--rw virtual-network-id?	instance-
					+--rw (address)?	
					+++:(no-address)	
ty					+--rw no-address?	emp
					+--:(ipv4)	
t:ipv4-address					+--rw ipv4?	ine
					+--:(ipv4-prefix)	
t:ipv4-prefix					+--rw ipv4-prefix?	ine
					+--:(ipv6)	
t:ipv6-address					+--rw ipv6?	ine
					+--:(ipv6-prefix)	
t:ipv6-prefix					+--rw ipv6-prefix?	ine
					+--:(mac)	
g:mac-address					+--rw mac?	yan
					+--:(distinguished-name)	
tinguished-name-type					+--rw distinguished-name?	dis
					+--:(as-number)	
t:as-number					+--rw as-number?	ine
					+--:(null-address)	
					+--rw null-address	
					+--rw address?	empty
					+--:(afi-list)	
					+--rw afi-list	
dress					+--rw address-list*	simple-ad
					+--:(instance-id)	
					+--rw instance-id	
d-type					+--rw iid?	instance-i
					+--rw mask-length?	uint8
ress					+--rw address?	simple-add
					+--:(as-number-lcaf)	



			+--rw as-number-lcaf	
			+--rw as? inet:as-number	
			+--rw address? simple-address	
			+++:(application-data)	
			+--rw application-data	
e-address			+--rw address?	simpl
			+--rw protocol?	uint8
port-number			+--rw ip-tos?	int32
			+--rw local-port-low?	inet:
port-number			+--rw local-port-high?	inet:
			+--rw remote-port-low?	inet:
port-number			+--rw remote-port-high?	inet:
port-number				
			+++:(geo-coordinates)	
			+--rw geo-coordinates	
			+--rw latitude?	bits
8			+--rw latitude-degrees?	uint
			+--rw latitude-minutes?	uint
8			+--rw latitude-seconds?	uint
			+--rw longitude?	bits
			+--rw longitude-degrees?	uint
16			+--rw longitude-minutes?	uint
8			+--rw longitude-seconds?	uint
8			+--rw altitude?	int3
2			+--rw address?	simp
le-address				
			+++:(nat-traversal)	
			+--rw nat-traversal	
6			+--rw ms-udp-port?	uint1
			+--rw etr-udp-port?	uint1
6			+--rw global-etr-rloc?	simpl
e-address			+--rw ms-rloc?	simpl
e-address			+--rw private-etr-rloc?	simpl
e-address			+--rw rtr-rlocs*	simpl
e-address				
			+++:(explicitLocatorPath)	
			+--rw explicitLocatorPath	
			+--rw hop* [hop-id]	
			+--rw hop-id	string
ress			+--rw address?	simple-add
			+--rw lrs-bits?	bits
			+++:(sourceDestKey)	
			+--rw sourceDestKey	
			+--rw source?	simple-address
			+--rw dest?	simple-address
			+++:(keyValueAddress)	

				<pre> +--rw key-value-address +--rw key?      simple-address +--rw value?    simple-address +--:(service-path) +--rw service-path </pre>
--	--	--	--	------------------------------------------------------------------------------------------------------------------------------------------



[illegible]

## 5.2. Module Definition

```
<CODE BEGINS> file "ietf-lisp-mapserver@2017-07-01.yang"
module ietf-lisp-mapserver {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp-mapserver";
  prefix lisp-ms;
  import ietf-lisp {
    prefix lisp;
  }
  import ietf-lisp-address-types {
    prefix lcaf;
  }
  import ietf-yang-types {
    prefix yang;
    revision-date 2013-07-15;
  }

  organization
    "IETF LISP (Locator/ID Separation Protocol) Working Group";
  contact
```

```
"lisp@ietf.org";
description
  "This YANG module defines the generic parameters for a LISP
  Map-Server. The module can be extended by vendors to define
  vendor-specific parameters and policies.

  Copyright (c) 2015 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC 6338; see
  the RFC itself for full legal notices.
  ";

revision 2017-07-01 {
  description
    "Initial revision.";
  reference
    "https://tools.ietf.org/html/rfc6833";
}

identity ms {
  base lisp:lisp-role;
  description
    "LISP Map-Server.";
}

grouping ms-counters {
  description "Group that defines map-server counters.";
  container counters {
    config false;
    description "Container for the counters";

    leaf map-registers-in {
      type yang:counter32;
      description "Number of incoming Map-Register messages";
    }

    leaf map-registers-in-auth-failed {
      type yang:counter32;
      description
        "Number of incoming Map-Register messages failed
  ";
    }
  }
}
```

```
        authentication";
    }

    leaf map-notify-records-out {
        type yang:counter32;
        description
            "Number of outgoing Map-Notify records";
    }

    leaf proxy-reply-records-out {
        type yang:counter32;
        description
            "Number of outgoing proxy Map-Reply records";
    }

    leaf map-requests-forwarded-out {
        type yang:counter32;
        description
            "Number of outgoing Map-Requests forwarded to ETR";
    }
}

augment "/lisp:lisp/lisp:lisp-router-instances"
+ "/lisp:lisp-router-instance" {
    when "lisp:lisp-role/lisp:lisp-role-type = 'lisp-ms:ms'" {
        description
            "Augment is valid when LISP device type is Map-Server.";
    }
    description
        "This augments LISP devices list with Map-Server specific
        parameters.";
    container map-server {
        presence "LISP Map-Server operation enabled";
        description
            "Map-Server parameters.";
        container sites{
            description
                "Sites to accept registrations from.";
            list site {
                key site-id;
                description
                    "Site that can send registrations.";
                leaf site-id {
                    type uint64;
                    description "Site ID";
                }
                container auth-key {
```

```
        description
        "Site authentication key.";
        leaf auth-key-value {
            type string;
            description
            "Clear text authentication key";
        }
        leaf-list auth-key-type {
            type lisp:auth-key-type;
            description
            "Authentication key type.";
        }
    }
}

container virtual-network-ids {
    description
    "Sites for which the Map-Server accepts registrations.";
    list virtual-network-identifier {
        key "vni";
        description
        "Virtual network instances in the Map-Server.";
        leaf vni {
            type lcaf:instance-id-type;
            description
            "Virtual network identifier.";
        }
    }
    container mappings {
        description
        "EIDs registered by device.";
        list mapping {
            key "eid-id";
            description
            "List of EIDs registered by device.";
            leaf eid-id {
                type lisp:eid-id;
                description
                "Id of the EID registered.";
            }
            container eid-address {
                uses lcaf:lisp-address;
                description
                "EID in generic LISP address format registered
                with the Map-Server.";
            }
        }
        leaf-list site-id {
            type uint64;
            description "Site ID";
        }
    }
}
```

```

    }
    leaf more-specifics-accepted {
        type boolean;
        default "false";
        description
            "Flag indicating if more specific prefixes
             can be registered.";
    }
    leaf mapping-expiration-timeout {
        type int16;
        units "seconds";
        default "180"; //3 times the mapregister int
        description
            "Time before mapping is expired if no new
             registrations are received.";
    }
    container mapping-records {
        description
            "Datastore of registred mappings.";
        list mapping-record {
            key xtr-id;
            description
                "Registered mapping.";
            leaf xtr-id {
                type lisp:xtr-id-type;
                description "xTR ID";
            }
            leaf site-id {
                type uint64;
                description "Site ID";
            }
            uses lisp:mapping;
        }
    }
    }
    }
    }
    uses ms-counters;
}
leaf mapping-system-type {
    type lisp:mapping-system-ref;
    description
        "A reference to the mapping system";
}

container summary {
    config false;
    description "Summary state information";
}

```

```

    leaf number-configured-sites {
        type uint32;
        description "Number of configured LISP sites";
    }
    leaf number-registered-sites {
        type uint32;
        description "Number of registered LISP sites";
    }
    container af-datum {
        description "Number of configured EIDs per each AF";

        list af-data {
            key "address-type";
            description "Number of configured EIDs for this AF";
            leaf address-type {
                type lcaf:lisp-address-family-ref;
                description "AF type";
            }
            leaf number-configured-eids {
                type uint32;
                description "Number of configured EIDs for this AF";
            }
            leaf number-registered-eids {
                type uint32;
                description "Number of registered EIDs for this AF";
            }
        }
    }
}
uses ms-counters;
}
}
}
<CODE ENDS>

```

## 6. LISP-Map-Resolver Module

This module captures the configuration data model of a LISP Map Resolver [RFC6833]. The model also captures some operational data elements.

### 6.1. Module Structure

```

module: ietf-lisp-mapresolver
augment /lisp:lisp/lisp:lisp-router-instances/lisp:lisp-router-instance:
  +--rw map-resolver!
    +--rw mapping-system-type?    lisp:mapping-system-ref
    +--rw ms-address?             inet:ip-address

```

## 6.2. Module Definition

```
<CODE BEGINS> file "ietf-lisp-mapresolver@2017-07-01.yang"
module ietf-lisp-mapresolver {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp-mapresolver";
  prefix lisp-mr;
  import ietf-lisp {
    prefix lisp;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization
    "IETF LISP (Locator/ID Separation Protocol) Working Group";
  contact
    "lisp@ietf.org";
  description
    "This YANG module defines the generic parameters for a LISP
    Map-Resolver. The module can be extended by vendors to define
    vendor-specific parameters and policies."

    Copyright (c) 2015 IETF Trust and the persons identified as
    authors of the code. All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (http://trustee.ietf.org/license-info).

    This version of this YANG module is part of RFC 6338; see
    the RFC itself for full legal notices.
  ";
  revision 2017-07-01 {
    description
      "Initial revision.";
    reference
      "https://tools.ietf.org/html/rfc6833";
  }
  identity mr {
    base lisp:lisp-role;
    description
      "LISP Map-Resolver.";
  }
  augment "/lisp:lisp/lisp:lisp-router-instances"
    +"/lisp:lisp-router-instance" {
```

```

    when "lisp:lisp-role/lisp:lisp-role-type = 'lisp-mr:mr'" {
      description
        "Augment is valid when LISP device type is Map-Resolver.";
    }
  description
    "This augments LISP devices list with Map-Resolver specific
    parameters.";
  container map-resolver {
    presence "LISP Map-Resolver operation enabled";
    description
      "Map-Resolver parameters.";
    leaf mapping-system-type {
      type lisp:mapping-system-ref;
      description
        "A reference to the mapping system";
    }
    leaf ms-address {
      when "../mapping-system-type='lisp-mr:single-node-mapping-system'";
      type inet:ip-address;
      description
        "address to reach the Map Server when "
        + "lisp-mr:single-node-mapping-system is being used.";
    }
  }
}
}
<CODE ENDS>

```

## 7. LISP-Address-Types Module

This module captures the various LISP address types, and is an essential building block used in other LISP modules.

### 7.1. Module Definition

```

<CODE BEGINS> file "ietf-lisp-address-types@2015-11-05.yang"
module ietf-lisp-address-types {
  namespace "urn:ietf:params:xml:ns:yang:ietf-lisp-address-types";
  prefix laddr;
  import ietf-inet-types {
    prefix inet;
    revision-date 2013-07-15;
  }
  import ietf-yang-types {
    prefix yang;
    revision-date 2013-07-15;
  }
  organization

```



```
"IETF LISP (Locator/ID Separation Protocol) Working Group";
contact
  "lisp@ietf.org";
description
  "This YANG module defines the LISP Canonical Address Formats
  (LCAF) for LISP. The module can be extended by vendors to
  define vendor-specific parameters.

  Copyright (c) 2014 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC 6338; see
  the RFC itself for full legal notices.

  ";
revision 2015-11-05 {
  description
    "Initial revision.";
  reference
    "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10";
}
identity lisp-address-family {
  description
    "Base identity from which identities describing LISP address
    families are derived.";
}
identity no-address-afi {
  base lisp-address-family;
  description
    "IANA Reserved.";
}
identity ipv4-afi {
  base lisp-address-family;
  description
    "IANA IPv4 address family.";
}
identity ipv4-prefix-afi {
  base lisp-address-family;
  description
    "IANA IPv4 address family prefix.";
}
```

```
identity ipv6-afi {
  base lisp-address-family;
  description
    "IANA IPv6 address family.";
}
identity ipv6-prefix-afi {
  base lisp-address-family;
  description
    "IANA IPv6 address family prefix.";
}
identity mac-afi {
  base lisp-address-family;
  description
    "IANA MAC address family.";
}
identity distinguished-name-afi {
  base lisp-address-family;
  description
    "IANA Distinguished Name address family.";
}
identity as-number-afi {
  base lisp-address-family;
  description
    "IANA AS Number address family.";
}
identity lcaf {
  base lisp-address-family;
  description
    "IANA LISP Canonical Address Format address family.";
}
identity null-address-lcaf {
  base lcaf;
  description
    "Null body LCAF type.";
}
identity afi-list-lcaf {
  base lcaf;
  description
    "AFI-List LCAF type.";
}
identity instance-id-lcaf {
  base lcaf;
  description
    "Instance-ID LCAF type.";
}
identity as-number-lcaf {
  base lcaf;
  description
```

```
        "AS Number LCAF type.";
    }
    identity application-data-lcaf {
        base lcaf;
        description
            "Application Data LCAF type.";
    }
    identity geo-coordinates-lcaf {
        base lcaf;
        description
            "Geo-coordinates LCAF type.";
    }
    identity opaque-key-lcaf {
        base lcaf;
        description
            "Opaque Key LCAF type.";
    }
    identity nat-traversal-lcaf {
        base lcaf;
        description
            "NAT-Traversal LCAF type.";
    }
    identity nonce-locator-lcaf {
        base lcaf;
        description
            "Nonce-Locator LCAF type.";
    }
    identity multicast-info-lcaf {
        base lcaf;
        description
            "Multicast Info LCAF type.";
    }
    identity explicit-locator-path-lcaf {
        base lcaf;
        description
            "Explicit Locator Path LCAF type.";
    }
    identity security-key-lcaf {
        base lcaf;
        description
            "Security Key LCAF type.";
    }
    identity source-dest-key-lcaf {
        base lcaf;
        description
            "Source/Dest LCAF type.";
    }
    identity replication-list-lcaf {
```

```
    base lcaf;
    description
      "Replication-List LCAF type.";
  }
  identity json-data-model-lcaf {
    base lcaf;
    description
      "JSON Data Model LCAF type.";
  }
  identity key-value-address-lcaf {
    base lcaf;
    description
      "Key/Value Address LCAF type.";
  }
  identity encapsulation-format-lcaf {
    base lcaf;
    description
      "Encapsulation Format LCAF type.";
  }
  identity service-path-lcaf {
    base lcaf;
    description
      "Service Path LCAF type.";
  }
  typedef instance-id-type {
    type uint32 {
      range "0..16777215";
    }
    description
      "Defines the range of values for an Instance ID.";
  }
  typedef service-path-id-type {
    type uint32 {
      range "0..16777215";
    }
    description
      "Defines the range of values for a Service Path ID.";
  }
  typedef distinguished-name-type {
    type string;
    description
      "Distinguished Name address.";
    reference
      "http://www.iana.org/assignments/address-family-numbers/
      address-family-numbers.xhtml";
  }
  typedef simple-address {
    type union {
```

```
    type inet:ip-address;
    type inet:ip-prefix;
    type yang:mac-address;
    type distinguished-name-type;
    type inet:as-number;
  }
  description
    "Union of address types that can be part of LCAFs.";
}

typedef lisp-address-family-ref {
  type identityref {
    base lisp-address-family;
  }
  description
    "LISP address family reference.";
}
typedef lcac-ref {
  type identityref {
    base lcac;
  }
  description
    "LCAF types reference.";
}

grouping lisp-address {
  description
    "Generic LISP address.";
  leaf address-type {
    type lisp-address-family-ref;
    mandatory true;
    description
      "Type of the LISP address.";
  }
  leaf virtual-network-id {
    type instance-id-type;
    description
      "Virtual Network Identifier (instance-id) of the address.";
  }
  choice address {
    description
      "Various LISP address types, including IP, MAC, and LCAF.";

    leaf no-address {
      when "../address-type = 'laddr:no-addr-afi'" {
        description
          "When AFI is 0.";
      }
    }
  }
}
```

```
    type empty;
    description
        "No address.";
}
leaf ipv4 {
    when "../address-type = 'laddr:ipv4-afi'" {
        description
            "When AFI is IPv4.";
    }
    type inet:ipv4-address;
    description
        "IPv4 address.";
}
leaf ipv4-prefix {
    when "../address-type = 'laddr:ipv4-prefix-afi'" {
        description
            "When AFI is IPv4.";
    }
    type inet:ipv4-prefix;
    description
        "IPv4 prefix.";
}
leaf ipv6 {
    when "../address-type = 'laddr:ipv6-afi'" {
        description
            "When AFI is IPv6.";
    }
    type inet:ipv6-address;
    description
        "IPv6 address.";
}
leaf ipv6-prefix {
    when "../address-type = 'laddr:ipv6-prefix-afi'" {
        description
            "When AFI is IPv6.";
    }
    type inet:ipv6-prefix;
    description
        "IPv6 address.";
}
leaf mac {
    when "../address-type = 'laddr:mac-afi'" {
        description
            "When AFI is MAC.";
    }
    type yang:mac-address;
    description
        "MAC address.";
```

```
}
leaf distinguished-name {
  when "../address-type = 'laddr:distinguished-name-afi'" {
    description
      "When AFI is distinguished-name.";
  }
  type distinguished-name-type;
  description
    "Distinguished Name address.";
}
leaf as-number {
  when "../address-type = 'laddr:as-number-afi'" {
    description
      "When AFI is as-number.";
  }
  type inet:as-number;
  description
    "AS Number.";
}
container null-address {
  when "../address-type = 'laddr:null-address-lcaf'" {
    description
      "When LCAF type is null.";
  }
  description
    "Null body LCAF type";
  leaf address {
    type empty;
    description
      "AFI address.";
  }
}
container afi-list {
  when "../address-type = 'laddr:afi-list-lcaf'" {
    description
      "When LCAF type is AFI-List.";
  }
  description
    "AFI-List LCAF type.";
  reference
    "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10#section-4.16.1";
  leaf-list address-list {
    type simple-address;
    description
      "List of AFI addresses.";
  }
}
```

```
container instance-id {
  when "../address-type = 'laddr:instance-id-lcaf'" {
    description
      "When LCAF type is Instance-ID";
  }
  description
    "Instance ID LCAF type.";
  reference
    "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
    #section-4.2";
  leaf iid {
    type instance-id-type;
    description
      "Instance ID value.";
  }
  leaf mask-length {
    type uint8;
    description
      "Mask length.";
  }
  leaf address {
    type simple-address;
    description
      "AFI address.";
  }
}
container as-number-lcaf {
  when "../address-type = 'laddr:as-number-lcaf'" {
    description
      "When LCAF type is AS-Number.";
  }
  description
    "AS Number LCAF type.";
  reference
    "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
    #section-4.3";
  leaf as {
    type inet:as-number;
    description
      "AS number.";
  }
  leaf address {
    type simple-address;
    description
      "AFI address.";
  }
}
container application-data {
```



```
when "../address-type = 'laddr:application-data-lcaf'" {
  description
    "When LCAF type is Application Data.";
}
description
  "Application Data LCAF type.";
reference
  "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
  #section-4.4";
leaf address {
  type simple-address;
  description
    "AFI address.";
}
leaf protocol {
  type uint8;
  description
    "Protocol number.";
}
leaf ip-tos {
  type int32;
  description
    "Type of service field.";
}
leaf local-port-low {
  type inet:port-number;
  description
    "Low end of local port range.";
}
leaf local-port-high {
  type inet:port-number;
  description
    "High end of local port range.";
}
leaf remote-port-low {
  type inet:port-number;
  description
    "Low end of remote port range.";
}
leaf remote-port-high {
  type inet:port-number;
  description
    "High end of remote port range.";
}
}
container geo-coordinates {
  when "../address-type = 'laddr:geo-coordinates-lcaf'" {
    description
```

```
        "When LCAF type is Geo-coordinates.";
    }
    description
        "Geo-coordinates LCAF type.";
    reference
        "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
        #section-4.5";
    leaf latitude {
        type bits {
            bit N {
                description
                    "Latitude bit.";
            }
        }
        description
            "Bit that selects between North and South latitude.";
    }
    leaf latitude-degrees {
        type uint8 {
            range "0 .. 90";
        }
        description
            "Degrees of latitude.";
    }
    leaf latitude-minutes {
        type uint8 {
            range "0..59";
        }
        description
            "Minutes of latitude.";
    }
    leaf latitude-seconds {
        type uint8 {
            range "0..59";
        }
        description
            "Seconds of latitude.";
    }
    leaf longitude {
        type bits {
            bit E {
                description
                    "Longitude bit.";
            }
        }
        description
            "Bit that selects between East and West longitude.";
    }
}
```

```
leaf longitude-degrees {
  type uint16 {
    range "0 .. 180";
  }
  description
    "Degrees of longitude.";
}
leaf longitude-minutes {
  type uint8 {
    range "0..59";
  }
  description
    "Minutes of longitude.";
}
leaf longitude-seconds {
  type uint8 {
    range "0..59";
  }
  description
    "Seconds of longitude.";
}
leaf altitude {
  type int32;
  description
    "Height relative to sea level in meters.";
}
leaf address {
  type simple-address;
  description
    "AFI address.";
}
}
container nat-traversal {
  when "../address-type = 'laddr:nat-traversal-lcaf'" {
    description
      "When LCAF type is NAT-Traversal.";
  }
  description
    "NAT-Traversal LCAF type.";
  reference
    "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
    #section-4.6";
  leaf ms-udp-port {
    type uint16;
    description
      "Map-Server UDP port (set to 4342).";
  }
  leaf etr-udp-port {
```

```
        type uint16;
        description
            "ETR UDP port.";
    }
    leaf global-etr-rloc {
        type simple-address;
        description
            "Global ETR RLOC address.";
    }
    leaf ms-rloc {
        type simple-address;
        description
            "Map-Server RLOC address.";
    }
    leaf private-etr-rloc {
        type simple-address;
        description
            "Private ETR RLOC address.";
    }
    leaf-list rtr-rlocs {
        type simple-address;
        description
            "List of RTR RLOC addresses.";
    }
}
container explicit-locator-path {
    when "../address-type = 'laddr:explicit-locator-path-lcaf'" {
        description
            "When LCAF type type is Explicit Locator Path.";
    }
    description
        "Explicit Locator Path LCAF type.";
    reference
        "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
        #section-4.9";
    list hop {
        key "hop-id";
        ordered-by user;
        description
            "List of locator hops forming the explicit path.";
        leaf hop-id {
            type string {
                length "1..64";
            }
            description
                "Unique identifier for the hop.";
        }
        leaf address {
```

```

        type simple-address;
        description
            "AFI address.";
    }
    leaf lrs-bits {
        type bits{
            bit lookup {
                description
                    "Lookup bit.";
            }
            bit rloc-probe {
                description
                    "RLOC-probe bit.";
            }
            bit strict {
                description
                    "Strict bit.";
            }
        }
        description
            "Flag bits per hop.";
    }
}

container source-dest-key {
    when "../address-type = 'laddr:source-dest-key-lcaf'" {
        description
            "When LCAF type type is Source/Dest.";
    }
    description
        "Source/Dest LCAF type.";
    reference
        "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10#section-4.11";
    leaf source {
        type simple-address;
        description
            "Source address.";
    }
    leaf dest {
        type simple-address;
        description
            "Destination address.";
    }
}

container key-value-address {
    when "../address-type = 'laddr:key-value-address-lcaf'" {
        description

```

```
        "When LCAF type type is Key/Value Address.";
    }
    description
        "Key/Value Address LCAF type.";
    reference
        "http://tools.ietf.org/html/draft-ietf-lisp-lcaf-10
        #section-4.11";
    leaf key {
        type simple-address;
        description
            "Address as Key.";
    }
    leaf value {
        type simple-address;
        description
            "Address as Value.";
    }
}
container service-path {
    when "../address-type = 'laddr:service-path-lcaf'" {
        description
            "When LCAF type service path identifier.";
    }
    description
        "Service Path LCAF type.";
    reference
        "http://tools.ietf.org/html/draft-ermagan-lisp-nsh-00";
    leaf service-path-id {
        type service-path-id-type;
        description
            "Service path identifier for the path for NSH header";
    }
    leaf service-index {
        type uint8;
        description
            "Service path index for NSH header";
    }
}
}
}
```

<CODE ENDS>

## 8. Acknowledgments

The tree view and the YANG model shown in this document have been formatted with the 'pyang' tool.

## 9. IANA Considerations

This memo includes no request to IANA.

## 10. Security Considerations

Security Considerations TBD

## 11. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <<http://www.rfc-editor.org/info/rfc6830>>.
- [RFC6832] Lewis, D., Meyer, D., Farinacci, D., and V. Fuller, "Interworking between Locator/ID Separation Protocol (LISP) and Non-LISP Sites", RFC 6832, DOI 10.17487/RFC6832, January 2013, <<http://www.rfc-editor.org/info/rfc6832>>.
- [RFC6833] Fuller, V. and D. Farinacci, "Locator/ID Separation Protocol (LISP) Map-Server Interface", RFC 6833, DOI 10.17487/RFC6833, January 2013, <<http://www.rfc-editor.org/info/rfc6833>>.
- [RFC6836] Fuller, V., Farinacci, D., Meyer, D., and D. Lewis, "Locator/ID Separation Protocol Alternative Logical Topology (LISP+ALT)", RFC 6836, DOI 10.17487/RFC6836, January 2013, <<http://www.rfc-editor.org/info/rfc6836>>.
- [RFC8060] Farinacci, D., Meyer, D., and J. Snijders, "LISP Canonical Address Format (LCAF)", RFC 8060, DOI 10.17487/RFC8060, February 2017, <<http://www.rfc-editor.org/info/rfc8060>>.

[RFC8111] Fuller, V., Lewis, D., Ermagan, V., Jain, A., and A. Smirnov, "Locator/ID Separation Protocol Delegated Database Tree (LISP-DDT)", RFC 8111, DOI 10.17487/RFC8111, May 2017, <<http://www.rfc-editor.org/info/rfc8111>>.

#### Authors' Addresses

Vina Ermagan  
Cisco Systems  
San Jose, CA  
USA

Email: [vermagan@cisco.com](mailto:vermagan@cisco.com)

Alberto Rodriguez-Natal  
Cisco Systems  
San Jose, CA  
USA

Email: [natal@cisco.com](mailto:natal@cisco.com)

Florin Coras  
Cisco Systems  
San Jose, CA  
USA

Email: [fcoras@cisco.com](mailto:fcoras@cisco.com)

Carl Moberg  
Cisco Systems  
San Jose, CA  
USA

Email: [camoberg@cisco.com](mailto:camoberg@cisco.com)

Reshad Rahman  
Cisco Systems  
Canada

Email: [rrahman@cisco.com](mailto:rrahman@cisco.com)



Albert Cabellos-Aparicio  
Technical University of Catalonia  
Barcelona  
Spain

Email: [acabello@ac.upc.edu](mailto:acabello@ac.upc.edu)

Fabio Maino  
Cisco Systems  
San Jose, CA  
USA

Email: [fmaino@cisco.com](mailto:fmaino@cisco.com)

LISP Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: January 4, 2018

A. Rodriguez-Natal  
V. Ermagan  
A. Smirnov  
V. Ashtaputre  
Cisco Systems  
D. Farinacci  
lispers.net  
July 3, 2017

Vendor Specific LCAF  
draft-rodrigueznatal-lisp-vendor-lcaf-00

Abstract

This document describes a new LCAF for LISP, the Vendor Specific LCAF. This LCAF enables organizations to have internal encodings for LCAF addresses.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on January 4, 2018.

Copyright Notice

Copyright (c) 2017 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 Simplified BSD License.

## Table of Contents

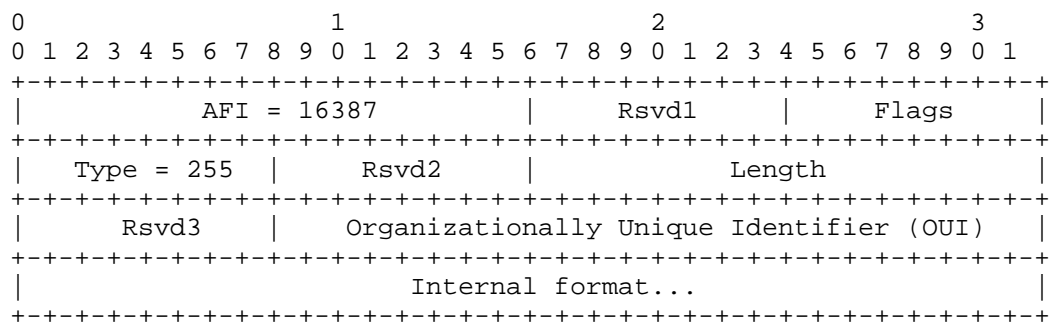
1. Introduction . . . . .	2
2. Vendor Specific LCAF . . . . .	2
3. Security Considerations . . . . .	3
4. Acknowledgments . . . . .	3
5. IANA Considerations . . . . .	3
6. Normative References . . . . .	3
Authors' Addresses . . . . .	4

## 1. Introduction

The LISP Canonical Address Format [RFC8060] defines the format and encoding for different address types that can be used on LISP [RFC6830] deployments. However, certain deployments require specific format encodings that may not be applicable outside of the use-case for which they are defined. The Vendor Specific LCAF allows organizations to create LCAF addresses to be internally-only used on particular LISP deployments.

## 2. Vendor Specific LCAF

The Vendor Specific LCAF relies on using the IEEE Organizationally Unique Identifier (OUI) [IEEE.802\_2001] to prevent collisions across vendors or organizations using the LCAF. The format of the Vendor Specific LCAF is provided below.



### Vendor Specific LCAF

The Vendor Specific LCAF has the following fields.

Rsvd3: This 8-bit field is reserved for future use. It MUST be set to 0 on transmit and MUST be ignored on receipt.

Organizationally Unique Identifier (OUI): This is a 24-bit field that carries the IEEE OUI [IEEE.802\_2001] of the organization.

Internal format: This is a variable length field that is left undefined on purpose. Each vendor or organization can define its own internal format(s) to use with the Vendor Specific LCAF.

The definition for the rest of the fields can be found in [RFC8060].

The Vendor Specific LCAF type SHOULD not be used in deployments where different organizations interoperate. If a LISP device receives a LISP message containing an Vendor Specific LCAF with an OUI it does not understand, it SHOULD drop the message and a log action MUST be taken.

### 3. Security Considerations

TBD.

### 4. Acknowledgments

TBD.

### 5. IANA Considerations

Following the guidelines of [RFC5226], this document requests IANA to update the "LISP Canonical Address Format (LCAF) Types" Registry defined in [RFC8060] to allocate the following assignment:

Value #	LISP LCAF Type Name	Reference
255	Vendor Specific	Section 2

Table 1: Vendor Specific LCAF assignment

### 6. Normative References

- [IEEE.802\_2001]  
 IEEE, "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture", IEEE 802-2001, DOI 10.1109/ieeestd.2002.93395, July 2002, <<http://ieeexplore.ieee.org/servlet/opac?punumber=7732>>.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <<http://www.rfc-editor.org/info/rfc6830>>.
- [RFC8060] Farinacci, D., Meyer, D., and J. Snijders, "LISP Canonical Address Format (LCAF)", RFC 8060, DOI 10.17487/RFC8060, February 2017, <<http://www.rfc-editor.org/info/rfc8060>>.

## Authors' Addresses

Alberto Rodriguez-Natal  
Cisco Systems  
170 Tasman Drive  
San Jose, CA  
USA

Email: natal@cisco.com

Vina Ermagan  
Cisco Systems  
170 Tasman Drive  
San Jose, CA  
USA

Email: vermagan@cisco.com

Anton Smirnov  
Cisco Systems  
170 Tasman Drive  
San Jose, CA  
USA

Email: asmirnov@cisco.com

Vrushali Ashtaputre  
Cisco Systems  
170 Tasman Drive  
San Jose, CA  
USA

Email: vrushali@cisco.com

Dino Farinacci  
lispers.net  
San Jose, CA  
USA

Email: farinacci@gmail.com