

LISP Working Group
Internet-Draft
Intended status: Informational
Expires: April 18, 2013

J. N. Chiappa
Yorktown Museum of Asian Art
October 15, 2012

An Architectural Perspective on the LISP
Location-Identity Separation System
draft-ietf-lisp-architecture-00

Abstract

LISP upgrades the architecture of the IPvN internetworking system by separating location and identity, current intermingled in IPvN addresses. This is a change which has been identified by the IRTF as a critically necessary evolutionary architectural step for the Internet. In LISP, nodes have both a 'locator' (a name which says where in the network's connectivity structure the node is) and an 'identifier' (a name which serves only to provide a persistent handle for the node). A node may have more than one locator, or its locator may change over time (e.g. if the node is mobile), but it keeps the same identifier.

This document gives additional architectural insight into LISP, and considers a number of aspects of LISP from a high-level standpoint.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to format it for publication as an RFC or to translate it into languages other than English.

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 April 18, 2013.

Copyright Notice

Copyright (c) 2012 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.	Goals of LISP
2.1.	Reduce DFZ Routing Table Size
2.2.	Deployment of New Namespaces
2.3.	Future Development of LISP
3.	Architectual Perspectives
3.1.	Another Packet-Switching Layer
3.2.	'Double-Ended' Approach
4.	Architectual Aspects
4.1.	Critical State
4.2.	Need for a Mapping System
4.3.	Piggybacking of Control on User Data
5.	Namespaces
5.1.	LISP EIDs
5.1.1.	Residual Location Functionality in EIDs
5.2.	RLOCs
5.3.	Overlapping Uses of Existing Namespaces
5.4.	LCAFs
6.	Scalability
6.1.	Demand Loading of Mappings
6.2.	Caching of Mappings
6.3.	Amount of State
6.4.	Scalability of The Indexing Subsystem
7.	Security
7.1.	Basic Philosophy
7.2.	Design Guidance
7.2.1.	Security Mechanism Complexity
7.3.	Security Overview
7.3.1.	Securing Lookups
7.3.2.	Securing The Indexing Subsystem
7.3.3.	Securing Mappings
7.4.	Securing the xTRs
8.	Robustness
9.	Fault Discovery/Handling
10.	Optimization
11.	Open Issues
11.1.	Local Open Issues
11.1.1.	Missing Mapping Packet Queueing
11.1.2.	Mapping Cache Management Algorithm
11.2.	Systemic Open Issues
11.2.1.	Mapping Database Provider Lock-in
11.2.2.	Automated ETR Synchronization
11.2.3.	EID Reachability
11.2.4.	Detect and Avoid Broken ETRs
12.	Acknowledgments
13.	IANA Considerations
14.	Security Considerations
15.	References
15.1.	Normative References
15.2.	Informative References
	Appendix A. Glossary/Definition of Terms
	Appendix B. Other Appendices

1. Introduction

This document begins by introducing some high-level architectural perspectives which have proven useful for thinking about the LISP location-identity separation system. It then discusses some

architectural aspects of LISP (e.g. its namespaces). The balance (and bulk) of the document contains architectural analysis of the LISP system; that is, it reviews from a high-level standpoint various aspects of that system; e.g. its scalability, security, robustness, etc.

NOTE: This document assumes a fair degree of familiarity with LISP; in particular, the reader should have a good 'high-level' understanding of the overall LISP system architecture, such as is provided by [Introduction], "An Introduction to the LISP System".

By "system architecture" above, the restricted meaning used there is: 'How the system is broken up into subsystems, and how those subsystems interact; when does information flows from one to another, and what that information is.' There is obviously somewhat more to architecture (e.g. the namespaces of a system, in particular their syntax and semantics), and that remaining architectural content is covered here.

2. Goals of LISP

As previously stated in the abstract, broadly, the goal of LISP is to be a practically deployable architectural upgrade to IPvN which performs separation of location and identity. But what is the value of that? What will it allow us to do?

The answer to that obviously starts with the things mentioned in the "Initial Applications" section of [Introduction], but there are other, longer-range (and broader) goals as well.

2.1. Reduce DFZ Routing Table Size

One of the main design drivers for LISP, as well as other location-identity separation proposals, is to decrease the overhead of running global routing system. In fact, it was this aspect that led the IRTF Routing RG to conclude that separation of location and identity was a key architectural underpinning needed to control the growth of the global routing system. [RFC6115]

As noted in [Introduction], many of the practical needs of Internet users are today met with techniques that increase the load on the global routing system (Provider Independent addresses for the provision of provider independence, multihoming, etc; more-specific routes for TE; etc.) Provision of these capabilities by a mechanism which does not involve extra load on the global routing system is therefore very desirable.

A number of factors, including the use of these techniques, has led to a great increase in the fragmentation of the address space, at least in terms of routing table entries. In particular, the growth in demand for multi-homing has been forseen as driving a large increase in the size of the global routing tables.

In addition, as the IPv4 address space becomes fuller and fuller, there will be an inevitable tendency to find use in smaller and smaller 'chunks' of that space. [RFC6127] This too would tend to increase the size of the global routing table.

LISP, if successful and widely deployed, offers an opportunity to use separation of location and identity to control the growth of the size of the global routing table. (A full examination of this topic is beyond the scope of this document - see {{find reference}}.)

2.2. Deployment of New Namespaces

Once the mapping system is widely deployed and available, it should make deployment of new namespaces (in the sense of new syntax, if not new semantics) easier. E.g. if someone wishes in the future to devise a system which uses native MPLS [RFC3031] for a data carriage system joining together a large number of xTRs, it would be easy enough to arrange to have the mappings for destinations attached to those xTRs be some sort of MPLS-specific name.

More broadly, the existence of a binding layer, with support for multiple namespaces built into the interface on both sides (see Section 5) is a tremendously powerful evolutionary tool; one can introduce a new namespace (on one side) more easily, if it is mapped to something which is already deployed (on the other). Then, having taken that step, one can invert the process, and deploy yet another new namespace, but this time on the other.

2.3. Future Development of LISP

Speculation about long-term future developments which are enabled by the deployment of LISP is not really proper for this document. However, interested readers may wish to consult [Future] for one person's thoughts on this topic.

3. Architectural Perspectives

This section contains some high-level architectural perspectives which have proven useful in a number of ways for thinking about LISP. For one, when trying to think of LISP as a complete system, they provide a conceptual structure which can aid analysis of LISP. For another, they can allow the application of past analysis of, and experience with, similar designs.

3.1. Another Packet-Switching Layer

When considering the overall structure of the LISP system at a high level, it has proven most useful to think of it as another packet-switching layer, run on top of the original internet layer - much as the Internet first ran on top of the ARPANET.

All the functions that a normal packet switch has to undertake - such as ensuring that it can reach its neighbours, and that they are still up - the devices that make up the LISP overlay also have to do, along with the 'tunnels' which connect them to other LISP devices.

There is, however, one big difference: the fanout of a typical LISP ITR will be much larger than most classic physical packet switches. (ITRs only need to be considered, as the LISP tunnels are all effectively unidirectional, from ITR to ETR - an ETR needs to keep no per-tunnel state, etc.)

LISP is, fundamentally, a 'tunnel' based system. Tunnel system designs do have their issues (e.g. the high inter-'switch' fan-out), but it's important to realize that they also can have advantages, some of which are listed below.

3.2. 'Double-Ended' Approach

LISP may be thought of as a 'double-ended' approach to enhancing the architecture, in that it uses pairs of devices, one at each end of a

communication stream. In particular, to interact with the population of 'legacy' hosts (which will be, inevitably, the vast majority, in the early stages of deployment) it requires a LISP device at both ends of the 'tunnel'.

This is in distinction to, say, NAT systems ([RFC1631]), which only need a device deployed at one end: the host at the other end doesn't need a matching device at its end to massage the packets, but can simply consume them on its own, as any packets it receives are fully normal packets. This allows any site which deploys such a 'single-ended' device to get the full benefit, whilst acting entirely on its own. [Wasserman]

The issue is not that LISP uses tunnels. Designs like HIP ([RFC4423]) and ILNP ([ILNP]), which do not involve tunnels, inhabit a similar space to tunnel-based designs like LISP, in that unless both ends are upgraded - or there is a proxy at the un-upgraded end - one doesn't get any benefits. So it's really not the tunnel which is the key aspect, it's the 'all at one end' part which is key. Whether the system is tunnel, versus non-tunnel, is not that important.

However, the double-ended approach of LISP does have advantages, as well as costs. To put it simply, the 'feature' of the alternative approach, that there's only a box at one end, has a 'bug': there's only a box at one end. There are things which such a design cannot accomplish, because of that.

To put it another way, does the fact that the packet thus necessarily has only a single 'name' in it for the entities at each end (i.e. the IPvN source and destination addresses), because it is a 'normal' packet, present a limitation? Put that way, it would seem natural that it should cause certain limits.

To compile a complete list of the things that can be done, when two separate 'names' are in the packet, is beyond the scope of this document. However, one example of the kind of thing that can be done is mobility with open connections, without needing to 'triangle route' the packets through some sort of 'base station' at the original location. Another is that it is possible to automatically tunnel IPv6 traffic over IPv4 infrastructure, or vice versa, invisibly to the hosts on both ends.

In the longer term, having having tunnel boxes will allow (and is allowing) us to explore other kinds of wrappings. For example, we can transport 'raw' local-network packets (such as Ethernet MAC frames) across an IPvN infrastructure.

One could also wrap packets in non-IPvN formats: perhaps to take direct advantage of the capabilities of underlying switching fabrics (e.g. MPLS [RFC3031]); perhaps to deploy new carriage protocols, etc, where non-standard packet formats will allow extended semantics.

4. Architectural Aspects

LISP does take some novel architectural approaches in a number of ways: e.g. its use of a separate mapping system, etc, etc. This section contains some commentary on some of the high-level architectural aspects of LISP.

4.1. Critical State

LISP does have 'critical state' in the network (i.e. state which, if

if lost, causes the communication to fail). However, because LISP is designed as an overall system, 'designing it in' allows for a 'systems' approach to its state issues. In LISP, this state has been designed to be maintained in an 'architected' way, so it does not produce systemic brittleness in the way that the state in NATs does.

For instance, throughout the system, provisions have been made to have redundant copies of state, in multiple devices, so that the loss of any one device does not necessarily cause a failure of an ongoing connection.

4.2. Need for a Mapping System

LISP does need to have a mapping system, which brings design, implementation, configuration and operational costs. Surely all these costs are a bad thing? However, having a mapping system have advantages, especially when there is a mapping layer which has global visibility (i.e. other entities know that it is there, and have an interface designed to be able to interact with it). This is unlike, say, the mappings in NAT, which are 'invisible' to the rest of the network.

In fact, one could argue that the mapping layer is LISP's greatest strength. Wheeler's Axiom* ('Any problem in computer science can be solved with another level of indirection') indicates that the binding layer available with the LISP mapping system will be of great value. Again, it is not the job of this document to list them all - and in any event, there is no way to foresee them all.

The author of this document has often opined that the hallmark of great architecture is not how well it does the things it was designed to do, but how well it does things it was never expected to have to handle. Providing such a powerful and generic binding layer is one sure way to achieve the sort of lasting flexibility and power that leads to that outcome.

[Footnote *: This Axiom is often mis-attributed to Butler Lampson, but Lampson himself indicated that it came from David Wheeler.]

4.3. Piggybacking of Control on User Data

LISP piggybacks control transactions on top of user data packets. This is a technique that has a long history in data networking, going back to the early ARPANET. [McQuillan] It is now apparently regarded as a somewhat dubious technique, the feeling seemingly being that control and user data should be strictly segregated.

It should be noted that none of the piggybacking of control functionality in LISP is architecturally fundamental to LISP. All of the functions in LISP which are performed with piggybacking could be performed almost equally well with separate control packets.

The "almost" is solely because it would cause more overhead (i.e. control packets); neither the response time, robustness, etc would necessarily be affected - although for some functions, to match the response time observed using piggybacking on user data would need as much control traffic as user data traffic.

This technique is particularly important, however, because of the issue identified at the start of this section - the very large fanout of the typical LISP switch. Unlike a typical router, which will have control interactions with only a few neighbours, a LISP switch could

eventually have control interactions with hundreds, or perhaps even thousands (for a large site) of neighbours.

Explicit control traffic, especially if good response times are desired, could amount to a very great deal of overhead in such a case.

5. Namespaces

One of the key elements in any architecture, or architectural analysis, are the namespaces involved: what are their semantics and syntax, what are the kinds of things they name, etc.

LISP has two key namespace, EIDs and RLOCs, but it must be emphasized that on an architectural level, neither the syntax, or, to a lesser degree, the semantics, of either are absolutely fixed. There are certain core semantics which are generally unchanging (such as the notion that EIDs provide only identity, whereas RLOCs provide location), but as we will see, there is a certain amount of flexibility available for the long-term.

In particular, all of LISP's key interfaces always include an Address Family Identifier (AFI) [AFI] for all names, so that new forms can be introduced at any time the need is felt. Of course, in practise such an introduction would not be a trivial exercise - but neither is it impossibly painful, as is the case with IPv4's 32-bit addresses, which are effectively impossible to upgrade.

5.1. LISP EIDs

A 'classic' EID is defined as a subset of the possible namespaces for endpoints. [Chiappa] Like most 'proper' endpoint names, as proposed there, they contain contain no information about the location of the endpoint. EIDs are the subset of possible endpoint names which are: fixed length, 'reasonably' short', binary (i.e. not intended for direct human use), globally unique (in theory), and allocated in a top-down fashion (to achieve the former).

LISP EIDs are, in line with the general LISP deployment philosophy, a reuse of something already existing - i.e. IPvN addresses. For those used as in LISP as EIDs, LISP removes much (or, in some cases, all) of the location-naming function of IPvN addresses.

In addition, the goal is to have EIDs name hosts (or, more properly, their end-end communication stacks), whereas the other LISP namespace group (RLOCs) names interfaces. The idea is not just to have two namespaces (with different semantics), but also to use them to name different classes of things - classes which currently do not have clearly differentiated names. This should produce even more functionality.

5.1.1. Residual Location Functionality in EIDs

LISP retains, especially in the early stages of the deployment, in many cases some residual location-naming functionality in EIDs. This is to allow the packet to be correctly routed/forwarded to the destination node, once it has been unwrapped by the ETR - and this is a direct result of LISP's deployment philosophy (see [Introduction], Section "Deployment").

Clearly, if there are one or more unmodified routers between the ETR and the destination node, those routers will have to perform a routing

step on the packet, for which it will need some information as to the location of the destination.

One can thus view such LISP EIDs, which retain 'stub' location information, as 'addresses' (in the definition of the generic sense of this term, as used here), but with the location information restricted to a limited, local scope.

This retention of some location functionality in LISP EIDs, in some cases, has led some people to argue that use of the name 'EID' is improper. In response, it was suggested that LISP use the term 'LEID', to distinguish LISP's 'bastardized' EIDs from 'true' EIDs, but this usage has never caught on.

It has also been suggested that one usage mode for LISP EIDs, in existing software loads, is to assign them as the address on an internal virtual interface; all the real interfaces would have RLOCs only. [Templin] This would make such LISP EIDs functionally equivalent to 'real' EIDs - they are names which are purely identity, have no location information of any kind in them, and cannot be used to make any routing decisions anywhere outside the host.

It is true that even in such cases, the EID is still not a 'pure' EID, as it names an interface, not the end-end stack directly. However, to do a perfect job here (or on separation of location and identity) is impossible without modifying existing hosts (which are, inevitably, almost always one end of an end-end communication) - and that has been ruled out, for reasons of viable deployment.

The need for interoperation with existing unmodified hosts limits the semantic changes one can impose, much as one might like to provide a cleaner separation. (Future evolution can bring us toward that state, however: see [Future].)

5.2. RLOCs

RLOCs are basically pure 'locators' [RFC1992], although their syntax and semantics is restricted at the moment, because in practise the only forms of RLOCs supported are IPv4 and IPv6.

5.3. Overlapping Uses of Existing Namespaces

It is in theory possible to have a block of IPvN namespace used as both EIDs and RLOCs. In other words, EIDs from that block might map to some other RLOCs, and that block might also appear in the DFZ as the locators of some other ETRs.

This is obviously potentially confusing - when a 'bare' IPvN address from one of these blocks, is it the RLOC, or the EID? Sometimes it is obvious from the context, but in general one could not simply have a (hypothetical) table which assigns all of the address space to either 'EID' or 'RLOC'.

In addition, such usage will not allow interoperation of the sites named by those EIDs with legacy sites, using the PITR mechanism ([Introduction], Section "Proxy Devices"), since that mechanism depends on advertizing the EIDs into the DFZ, although the LISP-NAT mechanism should still work ([Introduction], Section "LISP-NAT").

Nevertheless, as the IPv4 namespace becomes increasingly used up, this may be an increasingly attractive way of getting the 'absolute last drop' out of that space.

5.4. LCAFs

```
{{To be written.}}
```

```
--- Key-ID
```

```
--- Instance-IDs
```

6. Scalability

As with robustness, any global communication system must be scalable, and scalable up to almost any size. As previously mentioned ([xref target="Perspectives-Packet"/](#)), the large fanouts to be seen with LISP, due to its 'overlay' nature, present a special challenge.

One likely saving grace is that as the Internet grows, most sites will likely only interact with a limited subset of the Internet; if nothing else, the separation of the world into language blocks means that content in, say, Chinese, will not be of interest to most of the rest of the world. This tendency will help with a lot of things which could be problematic if constant, full, N^2 connectivity were likely on all nodes; for example the caching of mappings.

6.1. Demand Loading of Mappings

One question that many will have about LISP's design is 'why demand-load mappings - why not just load them all'? It is certainly true that with the growth of memory sizes, the size of the complete database is such that one could reasonably propose keeping the entire thing in each LISP device. (In fact, one proposed mapping system for LISP, named NERD, did just that. [NERD])

A 'pull'-based system was chosen over 'push' for several reasons; the main one being that the issue is not just the pure size of the mapping database, but its dynamicity. Depending on how often mappings change, the update rate of a complete database could be relatively large.

It is especially important to realize that, depending on what (probably unforeseeable) uses eventually evolve for the identity->location mapping capability LISP provides, the update rate could be very high indeed. E.g. if LISP is used for mobility, that will greatly increase the update rate. Such a powerful and flexible tool is likely be used in unforeseen ways (Section 4.2), so it's unwise to make a choice that would preclude any which raise the update rate significantly.

Push as a mechanism is also fundamentally less desirable than pull, since the control plane overhead consumed to load and maintain information about unused destinations is entirely wasted. The only potential downside to the pull option is the delay required for the demand-loading of information.

(It's also probably worth noting that many issues that some people have with the mapping approach of LISP, such as the total mapping database size, etc are the same - if not worse - for push as they are for pull.)

Finally, for IPv4, as the address space becomes more highly used, it will become more fragmented - i.e. there will tend to be more, smaller, entries. For a routing table, which every router has to hold, this is problematic. For a demand-loaded mapping table, it is

not bad. Indeed, this was the original motivation for LISP ([RFC4984]) - although many other useful and desirable uses for it have since been enumerated (see [Introduction], Section "Applications").

For all of these reasons, as long as there is locality of reference (i.e. most ITRs will use only a subset of the entire set), it makes much more sense to use the a pull model, than the classic push one heretofore seen widely at the internetwork layer (with a pull approach thus being somewhat novel - and thus unsettling to many - to people who work at that layer).

It may well be that some sites (e.g. large content providers) may need non-standard mechanisms - perhaps something more of a 'push' model. This remains to be determined, but it is certainly feasible.

6.2. Caching of Mappings

It should be noted that the caching spoken of here is likely not classic caching, where there is a fixed/limited size cache, and entries have to be discarded to make room for newly needed entries. The economics of memory being what they are, there is no reason to discard mappings once they have been loaded (although of course implementations are free to chose to do so, if they wish to).

This leads to another point about the caching of mappings: the algorithms for management of the cache are purely a local issue. The algorithm in any particular ITR can be changed at will, with no need for any coordination. A change might be for purposes of experimentation, or for upgrade, or even because of environmental variations - different environments might call for different cache management strategies.

The local, unsynchronized replacability of the cache management scheme is the architectural aspect of the design; the exact algorithm, which is engineering, is not.

6.3. Amount of State

```
{{To be written.}} [Iannone]

-- Mapping cache size
--- Mention studies
-- Delegation cache size (in MRs)
--- Mention studies
-- Any others?
```

6.4. Scalability of The Indexing Subsystem

LISP initially used an indexing subsystem called ALT. [ALT] ALT was relatively easy to construct from existing tools (GRE, BGP, etc), but it had a number of issues that made it unsuitable for large-scale use. ALT is now being superseded by DDT. [DDT]

The basic structure and operation of DDT is identical to that of TREE, so the extensive simulation work done for TREE applies equally to DDT, as do the conclusions drawn about TREE's superiority to ALT. [Jakab]

From an architectural point of view, the main advantage of DDT is that it enables client side caching of information about intermediate nodes in the resolution hierarchy, and also enables direct

communication with them. As a result, DDT has much better scaling properties than ALT.

The most important result of this change is that it avoids a concentration of resolution request traffic at the root of the indexing tree, a problem which by itself made ALT unsuitable for a global-scale system. The problem of root concentration (and thus overload) is almost unavoidable in ALT (even if masses of 'bypass' links are created).

ALT's scalability also depends on enforcing an intelligent organization that increases aggregation. Unfortunately, the current backbone routing BGP system shows that there is a risk of an organic growth of ALT, one which does not achieve aggregation. DDT does not display this weakness, since its organization is inherently hierarchical (and thus inherently aggregable).

The hierarchical organization of DDT also reduces the possibility for a configuration error which interferes with the operation of the network (unlike the situation with the current BGP DFZ). DDT security mechanisms can also help produce a high degree of robustness, both against misconfiguration, and deliberate attack. The direct communication with intermediate nodes in DDT also helps to quickly locate problems when they occur, resulting in better operational characteristics.

Next, since in ALT mapping requests must be transmitted through an overlay network, a significant share of requests can see substantially increased latencies. Simulation results in the TREE work clearly showed, and quantified, this effect.

The simulations also showed that the nodes composing the ALT and DDT networks for a mapping database of full Internet size could have thousands of neighbours. This is not an issue for DDT, but would almost certainly have been problematic for ALT nodes, since handling that number of simultaneous BGP sessions would likely to be difficult.

7. Security

LISP does not yet have an overarching security architecture. Many parts of the system have been hardened, but more on a case-by case basis, rather than from an overall perspective. (This is in part due to the 'just enough' approach to security initially taken in LISP; see [Introduction], Section "Just Enough Security".)

This section represents an attempt to produce a more broadly-based view of security in LISP; it mostly resulted from an attempt to add security to the DDT indexing system ([DDT]), but the analysis is general enough to apply to LISP broadly.

The good thing about the Internet is that it brings the world to your doorstep - masses of information from all around the world are instantly available on your computing device. The bad thing about the Internet is that it brings the world to your doorstep - including legions of crackers, thieves, and general scum and villainy. Thus, any node may be the target of fairly sophisticated attack - often automated (thereby reducing the effort required of the attacker to spread their attack as broadly as possible).

Security in LISP faces many of the same challenges as security for other parts of the Internet: good security usually means work for the

users, but without good security, things are vulnerable.

The Internet has seen many very secure systems devised, only to see them fail to reach wide adoption; the reasons for that are complex, and vary, but being too much work to use is a common thread. It is for this reason that LISP attempts to provide 'just enough' security (see [Introduction], Section "Just Enough Security").

7.1. Basic Philosophy

To square this circle, of needing to have very good security, but of it being too difficult to use very good security, the general concept is for LISP to have a series of 'graded' security measures available, with the 'ultimate' security mechanisms being very high-grade indeed.

The concept is to devise a plan in which LISP can simultaneously attempt to have not just 'ultimate' security, but also one or more 'easier' modes, ones which will be easier to configure and use. This 'easier' mode can be both an interim system (with the full powered system available for when it is needed), as well as the system used in sections of the network where security is less critical (following the general rule that the level of any security should generally be matched to what is being protected).

The challenge is to do this in a way that does not make the design more complex, since it has to include both the 'full strength' mechanism(s), and the 'easier to configure' mechanism(s). This is one of the fundamental tradeoffs to struggle with: it is easy to provide 'easier to configure' options, but that may make the overall design more complex.

As far as making it hard to implement to begin with (also something of a concern initially, although obviously not for the long term): we can make it 'easy' to deploy initially by simply not implementing/configuring the heavy-duty security early on. (Provided, of course, that the packet formats, etc, needed to support such security are all included in the design to begin with.)

7.2. Design Guidance

In designing the security, there are a small number of key points that will guide the design:

- Design lifetime
- Threat level

How long is the design intended to last? If LISP is successful, a minimum of a 50-year lifetime is quite possible. (For comparison, IPv4 is now 34 at the time of writing this, and will be around for at least several decades yet, if not longer; DNS is 28, and will probably last indefinitely.)

How serious are the threats it needs to meet? As mentioned above, the Internet can bring the worst crackers from anywhere to any location, in a flash. Their sophistication level is rising all the time: as the easier holes are plugged, they go after others. This will inevitably eventually require the most powerful security mechanisms available to counteract their attacks.

Which is not to say that LISP needs to be that secure right away. The threat will develop and grow over a long time period. However, the basic design has to be capable of being securable to the

expanded degree that will eventually be necessary. However, eventually it will need to be as securable as, say, DNS - i.e. it can be secured to the same level, although people may chose not to secure their LISP infrastructure as well as DNSSEC potentially does. [RFC4033]

In particular, it should be noted that historically many systems have been broken into, not through a weakness in the algorithms, etc, but because of poor operational mechanics. (The well-known 'Ultra' breakins of the Allies were mostly due to failures in operational procedure. [Welchman]) So operational capabilities intended to reduce the chance of human operational failure are just as important as strong algorithms; making things operationally robust is a key part of 'real' security.

7.2.1. Security Mechanism Complexity

Complexity is bad for several reasons, and should always be reduced to a minimum. There are three kinds of complexity cost: protocol complexity, implementation complexity, and configuration complexity. We can further subdivide protocol complexity into packet format complexity, and algorithm complexity. (There is some overlap of algorithm complexity, and implementation complexity.)

We can, within some limits, trade off one kind of complexity for others: e.g. we can provide configuration options which are simpler for the users to operate, at the cost of making the protocol and implementation complexity greater. And we can make initial (less capable) implementations simpler if we make the protocols slightly more complex (so that early implementations don't have to implement all the features of the full-blown protocol).

It's more of a question of some operational convenience/etc issues - e.g. 'How easy will it be to recover from a cryptosystem compromise'. If we have two ways to recover from a security compromise, one which is mostly manual and a lot of work, and another which is more automated but makes the protocol more complicated, if compromises really are very rare, maybe the smart call is to go with the manual thing - as long as we have looked carefully at both options, and understood in some detail the costs and benefits of each.

7.3. Security Overview

First, there are two different classes of attack to be considered: denial of service (DoS, i.e. the ability of an intruder to simply cause traffic not to successfully flow) versus exploitation (i.e. the ability to cause traffic to be 'highjacked', i.e. traffic to be sent to the wrong location).

Second, one needs to look at all the places that may be attacked. Again, LISP is a relatively simple system, so there are not that many parts to examine. The following are the things we need to secure:

- Lookups
- Indexing
- Mappings

7.3.1. Securing Lookups

{{To be written.}} Nonces, [SecurityReq]

7.3.2. Securing The Indexing Subsystem

It is envisioned that DDT will be highly securable, with all the delegations cryptographically secured via public-private signatures, very similar to the way DNS is ([RFC4033]).

The detailed mechanisms will be based on DNS's; this has the obvious benefit that all the lessons of DNS's years of practical experience with deployment, operations, etc, as well as the improvements to the basic design of DNS Security to provide a secure but usable system can be taken into account. However, DDT's security will also apply the thinking above, about making a 'versio' which is easier to use available.

{{To be written.}}

7.3.3. Securing Mappings

There are two approaches to securing the provision of mappings. The first, which is of course not completely satisfactory, is to only secure the channel between the ITR and the entities involved in providing mappings for it. (See above, Section 7.3.1)

The second is to secure the mappings themselves, by signing them 'at birth' (much the same way in which DNS Security operates). [RFC4033]. There was an attempt early on to suggest such a system for LISP ([SecurityAuth]), but it was not adopted (although the particular proposal was rather complex).

In the long run, the latter approach would obviously be superior, since it would be almost immune to any compromises of the mapping distribution system. {{Tie-in to space allocation security}}

7.4. Securing the xTRs

- Cache management
- Unsolicited Map-Replies are very bad - must go through mapping system to verify that the sender is authoritative for that range of EIDs

8. Robustness

- Depends on deployment as well as design
- Architected, visible replication of state/data
- Overlapping mechanisms (ref redundancy as key for robustness)

9. Fault Discovery/Handling

Any global communication system must be robust, and to be robust, it must be able to discover and handle problems. LISP's general philosophy of robustness is usually to have overlapping, simple mechanisms to discover and repair problems.

10. Optimization

- Philosophy
- Piggybacking
- 'Wiretapping' return mappings
- Security is an issue on that

11. Open Issues

Although much work has been done on LISP, and it operates satisfactorily in a reasonably large initial deployment, there are a few potentially problematic issues which remain. It is not clear if they will be issues which need to be dealt, since they have not proven to be obstacles so far, but it is worth listing them.

We can divide them in _local_ issues, i.e. ones which can be solved on a node-by-node basis, without requiring co-ordinated change, and systemic issues, which are obviously more problematic, since they could require co-ordinated changes to the protocols.

11.1. Local Open Issues

11.1.1. Missing Mapping Packet Queueing

Currently, some (all?) ITRs discard packets when they need a mapping, but have not loaded one yet, thereby causing the application to have to retransmit their opening packet. True, many ARP implementations use the same strategy, but the average ARP cache will only ever contain a few mappings, so it will not be so noticeable as with the mapping cache in an ITR, which will likely contain thousands.

Obviously, they could queue the packets while waiting to load the mapping, but this presents a number of subtle implementation issues: the ITR must make sure that it does not queue too many packets, etc.

In particular, if such packets are queued, this presents a potential DoS attack vector, unless the code is carefully written with that possibility in mind.

11.1.2. Mapping Cache Management Algorithm

Relatively little work has been done on sophisticated mapping cache management algorithms; in particular, the issue of which mapping(s) to drop if the cache reaches some maximum allowed size.

This particular issue has also been identified as another potential DoS attack vector.

11.2. Systemic Open Issues

11.2.1. Mapping Database Provider Lock-in

This refers to the fact that if one does not like the entity which is providing the indexing for the part of the address space which one's EIDs are allocated out of, there isn't probably isn't any way to switch to an alternative provider.

It is not clear that this is a real problem, though - the fact that all DNS top-level zones only have a single registry has not been a problem, nor has the fact that if one doesn't like the service the registry offers, one can't take one's DNS name to another registry.

Doing anything about it would also be difficult. Although it is _technically_ possible to duplicate any node in the delegation tree, and in theory such duplicates could be provided by different providers, it is not clear that such an arrangement would make _business_ sense.

For instance, if the holder of 10.1.1/24 decides they do not like the

entity providing indexing for 10.1/16 (call them E1), and ask another entity (E2) to provide alternative service for 10.1/16, two problems arise. First, E1 is still going to have to maintain the correct data for 10.1.1/24, and response to queries asking about them. Second, E2 will similarly have to maintain data for, and reply to queries about, all the other space-holders in 10.1/16 - even though they will likely not have any business relationship with them.

11.2.2. Automated ETR Synchronization

LISP requires that all the ETRs which are authoritative for the mappings for a particular address block return the same mapping data. In particular, their idea of the 'liveness' of all the ETRs should be identical, and correct.

At the moment, this is mostly a manual process, although liveness information can be currently be gathered from some IGPs.

11.2.3. EID Reachability

At the moment, LISP assumes that if an ETR is reachable from a given ITR, all destination EIDs behind that ETR are reachable from that ETR. There is no way to detect if any are not, nor to switch to an alternate ETR.

It is not clear that this is a problem that needs attention. The same has been true for all border routers for many years now, and there does not seem to be any general mechanism to deal with it (Although some BGP implementations may advertize changes in reachability status if what they are seeing from their IGP changes.)

11.2.4. Detect and Avoid Broken ETRs

{{To be written}}

12. Acknowledgments

The author would like thank all the members of the core LISP group for their willingness to allow him to add himself to their effort, and for their enthusiasm for whatever assistance he has been able to provide. He would also like to thank (in alphabetical order) Vina Ermagan, Vince Fuller, and Joel Halpern for their careful review of, and helpful suggestions for, this document. Grateful thanks also to Vince Fuller for help with XML.

A final thanks is due to John Wrocklawski for the author's organizational affiliation. This memo was created using the xml2rfc tool

13. IANA Considerations

This document makes no request of the IANA.

14. Security Considerations

This memo does not define any protocol and therefore creates no new security issues.

15. References

15.1. Normative References

- [DDT] V. Fuller, D. Lewis, and D. Farinacci, "LISP Delegated Database Tree", draft-fuller-lisp-ddt-01 (work in progress), March 2012.
- [Future] J. N. Chiappa, "Potential Long-Term Developments With the LISP System", draft-chiappa-lisp-evolution-00 (work in progress), July 2012.
- [Introduction] J. N. Chiappa, "An Introduction to the LISP Location-Identity Separation System", draft-ietf-lisp-introduction-00 (work in progress), October 2012.
- [SecurityAuth] R. Gagliano, "A Profile for Endpoint Identifier Origin Authorizations (IOA)", draft-rgaglian-lisp-iao-00 (work in progress), March 2009.
- [SecurityReq] F. Maino, V. Ermagan, A. Cabellos, D. Saucez, and O. Bonaventure, "LISP-Security (LISP-SEC)", draft-ietf-lisp-sec-02 (work in progress), March 2012.
- [AFI] IANA, "Address Family Indicators (AFIs)", Address Family Numbers, January 2011, <<http://www.iana.org/assignments/address-family-numbers>>.

15.2. Informative References

- [RFC1631] K. Egevang and P. Francis, "The IP Network Address Translator (NAT)", RFC 1631, May 1994.
- [RFC1992] I. Castineyra, J. N. Chiappa, and M. Steenstrup, "The Nimrod Routing Architecture", RFC 1992, August 1996.
- [RFC3031] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.
- [RFC4033] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS Security: Introduction and Requirements", RFC 4033, March 2005.
- [RFC4423] R. Moskowitz and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.
- [RFC4984] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing", RFC 4984, September 2007.
- [RFC6115] T. Li, Ed., "Recommendation for a Routing Architecture", RFC 6115, February 2011.

Perhaps the most ill-named RFC of all time; it contains nothing that could truly be called a 'routing architecture'.
- [RFC6127] J. Arkko and M. Townsley, "IPv4 Run-Out and IPv4-IPv6 Co-Existence Scenarios", RFC 6127, May 2011.
- [ALT] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "LISP Alternative Topology (LISP-ALT)",

- draft-ietf-lisp-alt-10 (work in progress),
December 2011.
- [NERD] E. Lear, "NERD: A Not-so-novel EID to RLOC Database",
draft-lear-lisp-nerd-09 (work in progress),
April 2012.
- [ILNP] R.J. Atkinson and S.N. Bhatti, "ILNP Architectural
Description", draft-irtf-rrg-ilnp-arch-05 (work in
progress), May 2012.
- [Chiappa] J. N. Chiappa, "Endpoints and Endpoint Names: A
Proposed Enhancement to the Internet Architecture",
Personal draft (work in progress), 1999,
<<http://www.chiappa.net/~jnc/tech/endpoints.txt>>.
- [Jakab] L. Jakab, A. Cabellos-Aparicio, F. Coras, D. Saucez,
and O. Bonaventure, "LISP-TREE: A DNS Hierarchy to
Support the LISP Mapping System", in 'IEEE Journal on
Selected Areas in Communications', Vol. 28, No. 8,
pp. 1332-1343, October 2010.
- [Iannone] L. Iannone and O. Bonaventure, "On the Cost of
Caching Locator/ID Mappings", in 'Proceedings of the
3rd International Conference on emerging Networking
EXperiments and Technologies (CoNEXT'07)', ACM, pp.
1-12, December 2007.
- [McQuillan] J. M. McQuillan, W. R. Crowther, B. P. Cosell,
D. C. Walden, and F. E. Heart, "Improvements in the
Design and Performance of the ARPA Network",
Proceedings AFIPS 1972 FJCC, Vol. 40, pp. 741-754.
- [Templin] F. Templin, "LISP WG", LISP WG list
message, Message-ID: 39C363776A4E8C4A94691D2BD9D1C9A1
05B0AC71@XCH-NW-7V2.nw.nos.boeing.com, 13
March 2009,, <[http://www.ietf.org/mail-archive/web/
lisp/current/msg00269.html](http://www.ietf.org/mail-archive/web/lisp/current/msg00269.html)>.
- [Wasserman] M. Wasserman, "IPv6 networking: Bad news for small
biz", IETF list message, Message-Id:
D11C4A34-7362-423E-A60E-476FC5D61D37@lilacglade.org,
5 April 2012, <[https://www.ietf.org/ibin/
c5i?mid=6&rid=49&gid=0&k1=933&k2=62733&
tid=1340933524](https://www.ietf.org/ibin/c5i?mid=6&rid=49&gid=0&k1=933&k2=62733&tid=1340933524)>.
- [Welchman] G. Welchman, "The Hut Six Story", Allen Lane,
London, pg. 3, 1982.

A truly monumental book; the ground it covers ranges
from his work helping break German codes in World War
II to his experience with securing data packet
networks!

Appendix A. Glossary/Definition of Terms

- Address
- Locator
- EID
- RLOC
- ITR
- ETR

- xTR
- Pitr
- Petr
- MR
- MS
- DFZ

Appendix B. Other Appendices

- Location/Identity Separation Brief History
- LISP History
- Old models (LISP 1, LISP 1.5, etc)
- Different mapping distribution models (e.g. LISP-NERD)
- Different mapping indexing models (LISP-ALT forwarding/overlay model),
LISP-TREE DNS-based, LISP-CONS)

Author's Address

J. Noel Chiappa
Yorktown Museum of Asian Art
Yorktown, Virginia
USA

EMail: jnc@mit.edu

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: July 22, 2017

V. Fuller

D. Lewis
V. Ermagan
Cisco Systems
A. Jain
Juniper Networks
A. Smirnov
Cisco Systems
January 18, 2017

LISP Delegated Database Tree
draft-ietf-lisp-ddt-09

Abstract

This document describes the LISP Delegated Database Tree (LISP-DDT), a hierarchical, distributed database which embodies the delegation of authority to provide mappings from LISP Endpoint Identifiers (EIDs) to Routing Locators (RLOCs). It is a statically-defined distribution of the EID namespace among a set of LISP-speaking servers, called DDT nodes. Each DDT node is configured as "authoritative" for one or more EID-prefixes, along with the set of RLOCs for Map Servers or "child" DDT nodes to which more-specific EID-prefixes are delegated.

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 July 22, 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	3
2. Requirements Language	4
3. Definition of Terms	5
4. Database organization	7
4.1. XEID prefixes	7
4.2. DDT database tree structure	7
4.3. Configuring prefix delegation	8
4.3.1. The root DDT node	9
5. DDT Map-Request	9
6. The Map-Referral message	10
6.1. Action codes	10
6.2. Referral set	11
6.3. Incomplete flag	11
6.4. Map-Referral Message Format	11
6.4.1. SIG section	14
7. DDT network elements and their operation	15
7.1. DDT node	15
7.1.1. Match of a delegated prefix (or sub-prefix)	15
7.1.2. Missing delegation from an authoritative prefix	16
7.2. DDT Map Server	16
7.3. DDT client	17
7.3.1. Queuing and sending DDT Map-Requests	17
7.3.2. Receiving and following referrals	18
7.3.3. Handling referral errors	20
7.3.4. Referral loop detection	20
8. Pseudo Code and Decision Tree diagrams	21
8.1. Map Resolver processing of ITR Map-Request	21
8.1.1. Pseudo-code summary	21
8.1.2. Decision tree diagram	21
8.2. Map Resolver processing of Map-Referral message	22
8.2.1. Pseudo-code summary	22
8.2.2. Decision tree diagram	24
8.3. DDT Node processing of DDT Map-Request message	25
8.3.1. Pseudo-code summary	25
8.3.2. Decision tree diagram	27
9. Example topology and request/referral following	27

9.1.	Lookup of 2001:db8:0103:1::1/128	30
9.2.	Lookup of 2001:db8:0501:8:4::1/128	31
9.3.	Lookup of 2001:db8:0104:2::2/128	32
9.4.	Lookup of 2001:db8:0500:2:4::1/128	32
9.5.	Lookup of 2001:db8:0500::1/128 (non-existent EID)	33
10.	Securing the database and message exchanges	34
10.1.	XEID-prefix Delegation	34
10.2.	DDT node operation	35
10.2.1.	DDT public key revocation	35
10.3.	Map Server operation	36
10.4.	Map Resolver operation	36
11.	Open Issues and Considerations	37
12.	IANA Considerations	37
13.	Security Considerations	37
14.	References	38
14.1.	Normative References	38
14.2.	Informative References	38
Appendix A.	Acknowledgments	39
Authors' Addresses	40

1. Introduction

LISP [RFC6830] specifies an architecture and mechanism for replacing the addresses currently used by IP with two separate name spaces: Endpoint Identifiers (EIDs), used end-to-end for terminating transport-layer associations, and Routing Locators (RLOCs), which are bound to topological location, and are used for routing and forwarding through the Internet infrastructure.

[RFC6833] specifies an interface between database storing EID-to-RLOC mappings and LISP devices which need this information for packet forwarding. Internal organization of such database is out the scope of [RFC6833]. Multiple architectures of the database have been proposed, each having its advantages and disadvantages (see for example [RFC6836] and [RFC6837]).

This document specifies architecture for database of LISP EID-to-RLOC mappings with emphasis on high scalability. LISP-DDT is a hierarchical distributed database, which embodies the delegation of authority to provide mappings, i.e. its internal structure mirrors the hierarchical delegation of address space. It also provides delegation information to Map Resolvers, which use the information to locate EID-to-RLOC mappings. A Map Resolver, which needs to locate a given mapping, will follow a path through the tree-structured database, contacting, one after another, the DDT nodes along that path until it reaches the leaf DDT node(s) authoritative for the mapping it is seeking.

LISP offers a general-purpose mechanism for mapping between EIDs and RLOCs. In organizing a database of EID to RLOC mappings, this specification extends the definition of the EID numbering space by logically prepending and appending several fields for purposes of defining the database index key: Database-ID (DBID, 16 bits), Instance identifier (IID, 32-bits), Address Family Identifier (16 bits), and EID-prefix (variable, according to AFI value). The resulting concatenation of these fields is termed an "Extended EID prefix" or XEID-prefix.

LISP-DDT defines a new device type, the DDT node, that is configured as authoritative for one or more XEID-prefixes. It also is configured with the set of more-specific sub-prefixes that are further delegated to other DDT nodes. To delegate a sub-prefix, the "parent" DDT node is configured with the RLOCs of each child DDT node that is authoritative for the sub-prefix. Each RLOC either points to a DDT Map Server to which an Egress Tunnel Router (ETR) has registered that sub-prefix or points to another DDT node in the database tree that further delegates the sub-prefix. See [RFC6833] for a description of the functionality of the Map Server and Map Resolver. Note that the target of a delegation must always be an RLOC (not an EID) to avoid any circular dependency.

To provide a mechanism for traversing the database tree, LISP-DDT defines a new LISP message type, the Map-Referral, which is returned to the sender of a Map-Request when the receiving DDT node can refer the sender to another DDT node that has more detailed information. See Section 6 for the definition of the Map-Referral message.

To find an EID-to-RLOC mapping, a LISP-DDT client, usually a DDT Map Resolver, starts by sending an Encapsulated Map-Request to a preconfigured DDT node RLOC. The DDT node responds with a Map-Referral message that either indicates that it will find the requested mapping to complete processing of the request or that the DDT client should contact another DDT node that has more-specific information; in the latter case, the DDT node then sends a new Encapsulated Map-Request to the next DDT node and the process repeats in an iterative manner.

Conceptually, this is similar to the way that a client of the Domain Name System (DNS) follows referrals (DNS responses that contain only NS records) from a series of DNS servers until it finds an answer.

2. 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].

3. Definition of Terms

Extended EID (XEID): a LISP EID extended with data uniquely identifying address space to which it belongs, such as LISP instance ID, Address Family etc. See Section 4.1 for detailed description of XEID data.

XEID-prefix: Extended EID-prefix (XEID-prefix) is a LISP EID-prefix prepended with XEID data. An XEID-prefix is used as a key index into the DDT database. XEID prefixes are used to describe database organization and are not seen as a single entity in protocol messages, though messages contain individual fields constituting XEID prefix.

DDT node: a network infrastructure component responsible for specific XEID-prefix(es) and for delegation of more-specific sub-prefixes to other DDT nodes.

DDT client: a network infrastructure component that sends DDT Map-Request messages and implements the iterative following of Map-Referral results. Typically, a DDT client will be a Map Resolver (as defined by [RFC6833]), but it is also possible for an ITR to implement DDT client functionality.

DDT Map Server: a DDT node that also implements Map Server functionality (forwarding Map-Requests and/or returning Map-Replies if offering proxy Map-Reply service) for a subset of its delegated prefixes. Map Server functions including proxying Map-Replies are described in [RFC6833].

DDT Map Server peers: list of all DDT Map Servers performing Map Server functionality for the same prefix. If peers are configured on a DDT Map Server then the latter will provide complete information about the prefix in its Map-Replies; otherwise the Map Server will mark returned reply as potentially incomplete.

DDT Map Resolver: a network infrastructure element which implements both the DDT client functionality and Map Resolver functionality as defined by [RFC6833]. DDT Map Resolver accepts Map-Requests from ITRs, sends DDT Map-Requests to DDT nodes and implements iterative following of Map-Referrals. Note that Map Resolvers do not respond to clients which sent Map-Requests, they only ensure that the Map-Request has been forwarded to a LISP device (ETR or proxy Map-Server) which will provide authoritative response to the original requestor. A DDT Map Resolver will typically maintain a cache of previously received Map-Referral message results containing RLOCs for DDT nodes responsible for XEID- prefixes of interest (termed the "referral cache").

Encapsulated Map-Request: a LISP Map-Request carried within an Encapsulated Control Message, which has an additional LISP header prepended. Sent to UDP destination port 4342. The "outer" addresses are globally-routable IP addresses, also known as RLOCs. Used by an ITR when sending to a Map Resolver and by a Map Server when forwarding a Map-Request to an ETR as documented in LISP-MS [RFC6833].

DDT Map-Request: an Encapsulated Map-Request sent by a DDT client to a DDT node. The "DDT-originated" flag is set in the encapsulation header indicating that the DDT node should return Map-Referral messages if the Map-Request EID matches a delegated XEID-prefix known to the DDT node. Section 7.3.1 describes how DDT Map-Requests are sent. Section 5 defines position of the "DDT-originated" flag in the Encapsulated Control Message header.

Authoritative XEID-prefix: an XEID-prefix delegated to a DDT node and for which the DDT node may provide further delegations of more-specific sub-prefixes.

Map-Referral: a LISP message sent by a DDT node in response to a DDT Map-Request for an XEID that matches a configured XEID-prefix delegation. A non-negative Map-Referral includes a "referral", a set of RLOCs for DDT nodes that have information about the more specific XEID prefix covering requested XEID; a DDT client "follows the referral" by sending another DDT Map-Request to one of those RLOCs to obtain either an answer or another referral to DDT nodes responsible for even more specific XEID-prefix. See Section 7.1 and Section 7.3.2 for details on the sending and processing of Map-Referral messages.

Negative Map-Referral: an answer from an authoritative DDT node that there is no mapping for the requested XEID. Negative Map-Referral is a Map-Referral sent in response to a DDT Map-Request that matches an authoritative XEID-prefix but for which there is no delegation configured (or no ETR registration if sent by a DDT Map-Server).

Pending Request List: the set of outstanding requests for which a DDT Map Resolver has received encapsulated Map-Requests from its clients seeking EID-to-RLOC mapping for a XEID. Each entry in the list contains additional state needed by the referral following process, including the XEID, requestor(s) of the XEID (typically, one or more ITRs), saved information about the last referral received and followed (matching XEID-prefix, action code, RLOC set, index of last RLOC queried in the RLOC set), and any LISP-SEC information ([I-D.ietf-lisp-sec]) that was included in the DDT client Map-Request. An entry in the list may be interchangeably

termed a "pending request list entry" or simply a "pending request".

For definitions of other terms, notably Map-Request, Map-Reply, Ingress Tunnel Router (ITR), Egress Tunnel Router (ETR), Map Server, and Map Resolver, please consult the LISP specification [RFC6830] and the LISP Mapping Service specification [RFC6833].

4. Database organization

4.1. XEID prefixes

DDT database is indexed by Extended EID-prefixes (XEID-prefixes). XEID-prefix is LISP EID-prefix together with data extending it to uniquely identify address space of the prefix. XEID-prefix is composed as binary encoding of five fields, in order of significance: DBID (16 bits), Instance Identifier (IID, 32 bits), Address Family Identifier (AFI, 16 bits), and EID-prefix (variable, according to AFI value). The fields are concatenated, with the most significant fields as listed above.

DBID is LISP-DDT database ID, a 16-bit field provided to allow the definition of multiple databases. In this version of DDT DBID MUST always be set to zero. Other values of DBID are reserved for future use.

Instance ID (IID) is 32-bit value describing context of EID prefix if the latter is intended for use in an environment where addresses may not be unique, such as on a Virtual Private Network where [RFC1918] address space is used. See "Using Virtualization and Segmentation with LISP" in [RFC6830] for more discussion of Instance IDs. Encoding of the instance ID (IID) is specified by [I-D.ietf-lisp-lcaf].

Address Family Identifier (AFI) is a 16-bit value defining syntax of EID-prefix. AFI values are assigned by IANA ([AFI]).

4.2. DDT database tree structure

LISP-DDT database of each DDT node is organised as a tree structure that is indexed by XEID prefixes. Leaves of the database tree describe delegation of authority between DDT nodes (see more on delegation and information kept in the database entries in Section 4.3).

DDT Map-Requests sent by the DDT client to DDT nodes always contain specific values for DBID, IID and AFI; never a range or unspecified

value for any of these fields. Thus XEID prefix used as key for search in the database tree will have length of at least 64 bits.

DDT node may, for example, be authoritative for a consecutive range of 3-tuples (DBID, IID, AFI) and all associated EID prefixes; or only for a specific EID prefix of a single 3-tuple. Thus XEID prefixes/keys of the database tree leaves may have length less, equal or more than 64 bits.

It is important to note that LISP-DDT does not store actual EID-to-RLOC mappings; it is, rather, a distributed index that can be used to find the devices (ETRs which registered their EIDs with DDT Map Servers) that can be queried with LISP to obtain those mappings. Changes to EID-to-RLOC mappings are made on the ETRs which define them, not to any DDT node configuration. DDT node configuration changes are only required when branches of the database hierarchy are added, removed, or modified.

4.3. Configuring prefix delegation

Every DDT node is configured with one or more XEID-prefixes for which it is authoritative along with a list of delegations of XEID-prefixes to other DDT nodes. A DDT node is required to maintain a list of delegations for all sub-prefixes of its authoritative XEID-prefixes; it also may list "hints", which are prefixes that it knows about that belong to its parents, to the root, or to any other point in the XEID-prefix hierarchy. A delegation (or hint) consists of an XEID-prefix, a set of RLOCs for DDT nodes that have more detailed knowledge of the XEID-prefix, and accompanying security information (for details of security information exchange and its use see Section 10). Those RLOCs are returned in Map-Referral messages when the DDT node receives a DDT Map-Request with an XEID that matches a delegation. A DDT Map Server will also have a set of sub-prefixes for which it accepts ETR mapping registrations and for which it will forward (or answer, if it provides proxy Map-Reply service) Map-Requests.

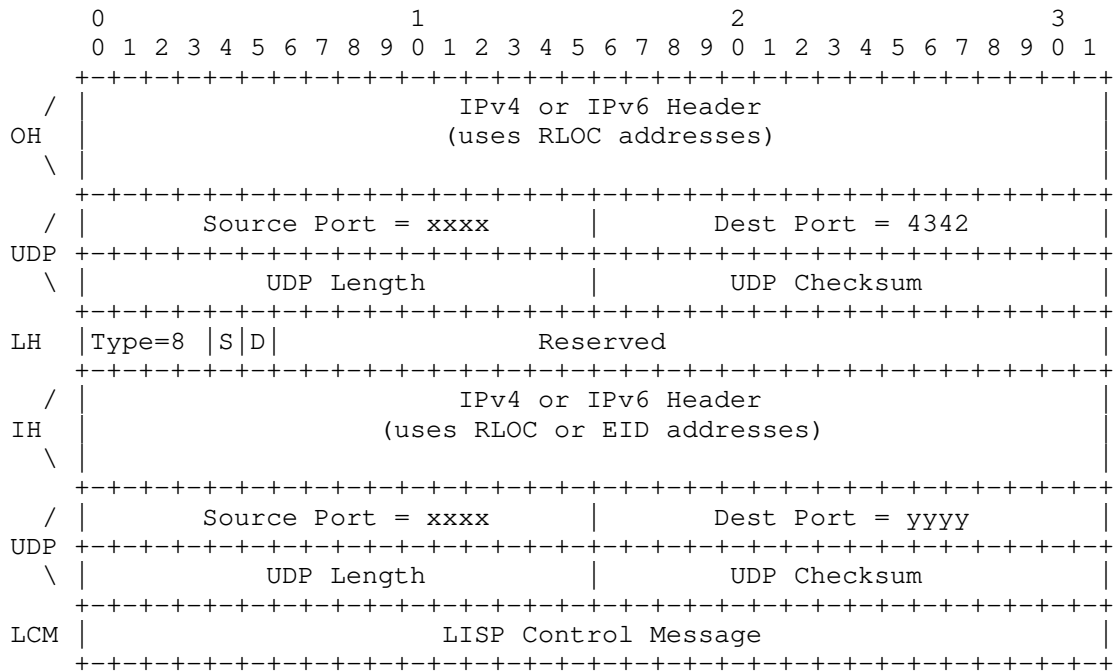
XEID prefix (or prefixes) for which DDT node is authoritative and delegation of authority for sub-prefixes is provided as configuration of the DDT node. Implementations will likely develop a language to express this prefix authority and delegation. Since DDT configuration is static (i.e. not exchanged between DDT nodes as part of the protocol itself) such language is implementation-dependant and is outside the scope of this specification.

4.3.1. The root DDT node

The root DDT node is the logical "top" of the distributed database hierarchy. It is authoritative for all XEID prefixes, that is for all valid 3-tuples (DBID, IID, AFI) and their EID prefixes. A DDT Map-Request that matches no configured XEID-prefix will be referred to the root node (see Section 8 for formal description of conditions when DDT Request is forwarded to the root node). The root node in a particular instantiation of LISP-DDT therefore MUST be configured with delegations for at least all defined IIDs and AFIs.

5. DDT Map-Request

A DDT client (usually a Map Resolver) uses LISP Encapsulated Control Message (ECM) to send Map-Request to a DDT node. Format of the ECM is defined by [RFC6830]. This specification adds to ECM flag "DDT-originated".



D: The "DDT-originated" flag. It is set by a DDT client to indicate that the receiver SHOULD return Map-Referral messages as appropriate. Use of the flag is further described in Section 7.3.1. This bit is allocated from LISP message header bits marked as Reserved in [RFC6830].

6. The Map-Referral message

This specification defines a new LISP message, the Map-Referral. It is sent by a DDT node to a DDT client in response to a DDT Map-Request message. See Section 6.4 for a detailed layout of the Map-Referral message fields.

The message consists of an action code along with delegation information about the XEID-prefix that matches the requested XEID.

6.1. Action codes

The action codes are as follows:

NODE-REFERRAL (0): indicates that the replying DDT node has delegated an XEID-prefix that matches the requested XEID to one or more other DDT nodes. The Map-Referral message contains a "map-record" with additional information, most significantly the set of RLOCs to which the prefix has been delegated, that is used by a DDT client to "follow" the referral.

MS-REFERRAL (1): indicates that the replying DDT node has delegated an XEID-prefix that matches the requested XEID to one or more DDT Map Servers. It contains the same additional information as a NODE-REFERRAL, but is handled slightly differently by the receiving DDT client (see Section 7.3.2).

MS-ACK (2): indicates that the replying DDT Map Server received a DDT Map-Request that matches an authoritative XEID-prefix for which it has one or more registered ETRs. This means that the request has been forwarded to one of those ETRs to provide an answer to the querying ITR.

MS-NOT-REGISTERED (3): indicates that the replying DDT Map Server received a Map-Request for one of its configured XEID-prefixes which has no ETRs registered.

DELEGATION-HOLE (4): indicates that the requested XEID matches a non-delegated sub-prefix of the XEID space. This is a non-LISP "hole", which has not been delegated to any DDT Map Server or ETR. See Section 7.1.2 for details. Also sent by a DDT Map Server with authoritative configuration covering the requested EID, but for which no specific site ETR is configured.

NOT-AUTHORITATIVE (5): indicates that the replying DDT node received a Map-Request for an XEID for which it is not authoritative and has no configured matching hint referrals. This can occur if a cached referral has become invalid due to a change in the database

hierarchy. However, if such a DDT node has a "hint" delegation covering the requested EID, it MAY choose to return NODE-REFERRAL or MS-REFERRAL as appropriate. When returning action code NOT-AUTHORITATIVE DDT node MUST provide EID-prefix received in the request and the TTL MUST be set to 0.

6.2. Referral set

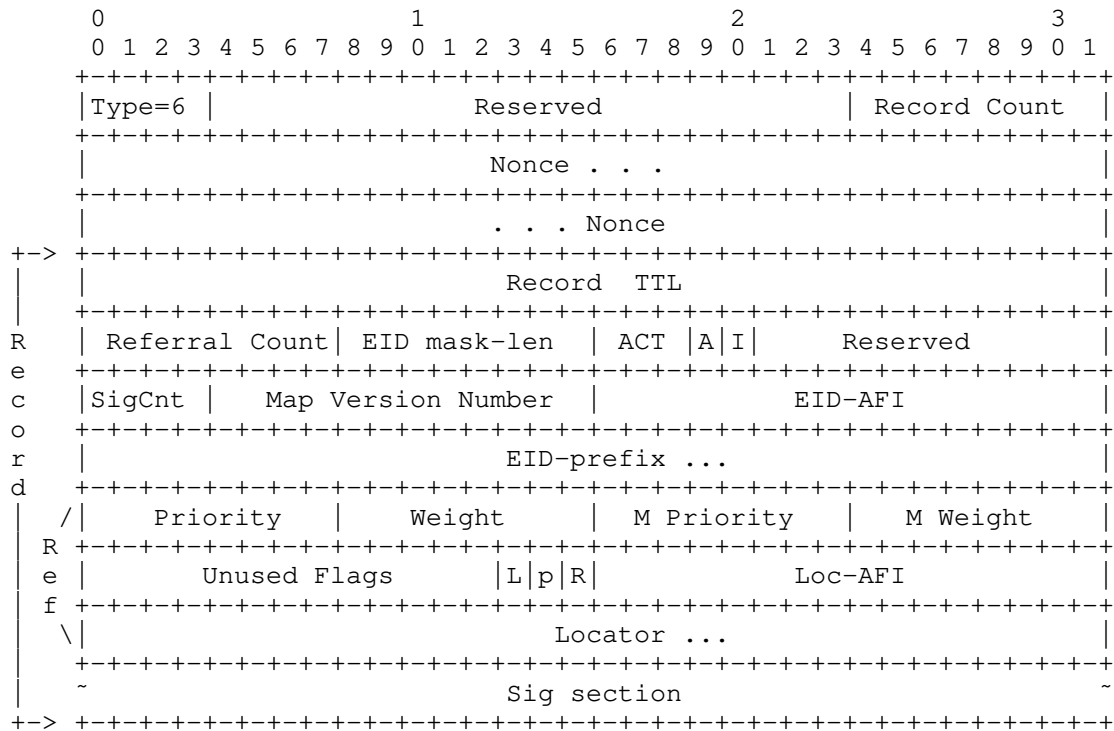
For "positive" action codes (NODE-REFERRAL, MS-REFERRAL, MS-ACK), a DDT node MUST include in the Map-Referral message a list of RLOCs for DDT nodes that are authoritative for the XEID-prefix being returned; a DDT client uses this information to contact one of those DDT nodes as it "follows" a referral.

6.3. Incomplete flag

A DDT node sets the "Incomplete" flag in a Map-Referral message if the Referral Set is incomplete; this is intended to prevent a DDT client from caching a referral with incomplete information. A DDT node MUST set the "incomplete" flag in the following cases:

- o If it is setting action code MS-ACK or MS-NOT-REGISTERED but the matching XEID-prefix is not flagged in configuration as "complete". XEID-prefix configuration on DDT Mapping Server SHOULD be marked as "complete" when configuration of the XEID-prefix lists all "peer" DDT nodes that are also authoritative for the same XEID-prefix or when it is known that local DDT node is the only one authoritative for the XEID-prefix.
- o If it is setting action code NOT-AUTHORITATIVE.

6.4. Map-Referral Message Format



Type: Type value 6 was reserved for future use in RFC6830, this document allocates this value to identify Map-Referral messages.

ACT: The "action" field of the mapping record in a Map-Referral message encodes one of the 6 action types: NODE-REFERRAL, MS-REFERRAL, MS-ACK, MS-NOT-REGISTERED, DELEGATION-HOLE, NOT-AUTHORITATIVE. See Section 6.1 for description of their meaning.

Incomplete: The "I" bit indicates that a DDT node's referral-set of locators is incomplete and the receiver of this message SHOULD NOT cache the referral. A DDT sets the "incomplete" flag, the TTL, and the Action Type field as follows:

Type	(Action field)	Incomplete Referral-set		TTL values
0	NODE-REFERRAL	NO	YES	1440
1	MS-REFERRAL	NO	YES	1440
2	MS-ACK	*	*	1440
3	MS-NOT-REGISTERED	*	*	1
4	DELEGATION-HOLE	NO	NO	15
5	NOT-AUTHORITATIVE	YES	NO	0

*: The "Incomplete" flag setting on Map Server originated referral of MS-ACK and MS-NOT-REGISTERED types depend on whether the Map Server has the full peer Map Server configuration for the same prefix and has encoded the information in the mapping record. Incomplete bit is not set when the Map Server has encoded the information, which means the referral-set includes all the RLOCs of all Map Servers that serve the prefix. It MUST be set when configuration of the Map Server does not flag matching prefix as configured with the complete information about "peer" Map Servers or when the Map Server does not return all configured locators.

Referral Count: number of RLOCs in the current Referral set, it is equal to the number of Ref sections in the message.

SigCnt: Indicates the number of signatures (sig section) present in the Record. If SigCnt is larger than 0, the signature information captured in a sig section as described in Section 6.4.1 will be appended to the end of the record. The number of sig sections at the end of the Record MUST match the SigCnt. Note that bits occupied by SigCnt were Reserved in Records embedded into messages defined by [RFC6830] and were required to be set to zero.

Loc-AFI: AFI of the Locator field. If AFI value is different from LCAF AFI, security keys are not included in the record. If AFI is equal to the LCAF AFI, the contents of the LCAF depend on the Type field of the LCAF. LCAF Type 11 is used to store security material along with the AFI of the locator. DDT nodes and DDT Map Servers can use this LCAF Type to include public keys associated with their Child DDT nodes for a XEID-prefix referral record. LCAF types and formats are defined in [I-D.ietf-lisp-lcaf].

Locator: RLOC of a DDT node the DDT client is being referred to. Length of this variable-length field is determined by the Loc-AFI.

All other fields and their descriptions are equivalent to those in the Map-Reply message, as defined in LISP [RFC6830]. Note, though, that the set of RLOCs correspond to the DDT node to be queried as a result of the referral not the RLOCs for an actual EID-to-RLOC mapping.

6.4.1. SIG section

SigCnt counts the number of signature sections that appear at the end of the Record. Format of the signature section is described below.

```

      +-----+-----+-----+-----+-----+-----+-----+-----+
      /|                                     Original Record TTL          |
      / +-----+-----+-----+-----+-----+-----+-----+-----+
      / |                                     Signature Expiration         |
      | +-----+-----+-----+-----+-----+-----+-----+-----+
s  |                                     Signature Inception             |
i  |                                     Key Tag                          |
g  |                                     Sig Length                      |
  | +-----+-----+-----+-----+-----+-----+-----+-----+
  | Sig-Algorithm |   Reserved   |   Reserved   |
  \ +-----+-----+-----+-----+-----+-----+-----+-----+
  \ ~                                     Signature                        ~
      +-----+-----+-----+-----+-----+-----+-----+-----+

```

Original Record TTL: The original Record TTL for this record that is covered by the signature. Record TTL is in minutes.

Signature Expiration and Inception: Specify the validity period for the signature. The signature MUST NOT be used for authentication prior to the inception date and MUST NOT be used for authentication after the expiration date. Each field specifies a date and time in the form of a 32-bit unsigned number of seconds elapsed since 1 January 1970 00:00:00 UTC, ignoring leap seconds, in network byte order.

Key Tag: An identifier to specify which key is used for this signature if more than one valid key exists for the signing DDT node.

Sig Length: The length of the Signature field in bytes.

Sig-Algorithm: The identifier of the cryptographic algorithm used for the signature. Sig-Algorithm values defined in this specification are listed in Table 1. Implementation conforming to this

specification MUST implement at least RSA-SHA256 for DDT signing. Sig-Algorithm type 1 RSA-SHA1 is deprecated and SHOULD NOT be used.

Sig-Algorithm	Name	Reference	Notes
1	RSA-SHA1	[RFC3447]	DEPRECATED
2	RSA-SHA256	[RFC3447]	MANDATORY

Table 1: Sig-Algorithm Values

Reserved: This field MUST be set to 0 on transmit and MUST be ignored on receipt.

Signature: Contains the cryptographic signature that covers the entire referral record that this signature belongs to. The Record TTL is set to Original Record TTL and the sig fields are Signature field is set to zero for the purpose of computing the Signature.

7. DDT network elements and their operation

As described above, DDT introduces a new network element, the "DDT node", extends the functionality of Map Servers and Map Resolvers to send and receive Map-Referral messages. The operation of each of these devices is described as follows.

7.1. DDT node

When a DDT node receives a DDT Map-Request, it compares the requested XEID against its list of XEID-prefix delegations and its list of authoritative XEID-prefixes and acts as follows:

7.1.1. Match of a delegated prefix (or sub-prefix)

If the requested XEID matches one of the DDT node's delegated prefixes, then a Map-Referral message is returned with the matching more-specific XEID-prefix and the set of RLOCs for the referral target DDT nodes including associated security information (see Section 10 for details on security). If at least one DDT node of the delegation is known to be a DDT Map Server, then the Map-Referral message SHOULD be sent with action code MS-REFERRAL to indicate to the receiver that LISP-SEC information (if saved in the pending request) SHOULD be included in the next DDT Map-Request; otherwise, the action code NODE-REFERRAL SHOULD be used.

Note that a matched delegation does not have to be for a sub-prefix of an authoritative prefix; in addition to being configured to delegate sub-prefixes of an authoritative prefix, a DDT node may also be configured with other XEID-prefixes for which it can provide referrals to DDT nodes anywhere in the database hierarchy. This capability to define "shortcut hints" is never required to be configured, but may be a useful heuristic for reducing the number of iterations needed to find an EID, particular for private network deployments.

Referral hints, if used properly, may reduce number of referrals a DDT client needs to follow to locate DDT Map Server authoritative for XEID prefix being resolved. On the other hand, incorrect use of hints may create circular dependencies between DDT nodes (or "referral loops"). DDT client MUST be prepared to handle such circular referrals. See Section 7.3.4 for discussion of referral loops and measures DDT client must implement in order to detect circular referrals and prevent infinite looping.

Another danger with use of hints is when DDT deployment spans multiple administrative domains (i.e. different authorities manage DDT nodes in the same DDT database). In this case operator managing a DDT node may not be aware of the fact that the node is being referred to by hints. Locator addresses in hints may become stale when referred DDT nodes are taken out of service or change their locator addresses.

7.1.2. Missing delegation from an authoritative prefix

If the requested XEID did not match a configured delegation but does match an authoritative XEID-prefix, then the DDT node MUST return a negative Map-Referral that uses the least-specific XEID-prefix that does not match any XEID-prefix delegated by the DDT node. The action code is set to DELEGATION-HOLE; this indicates that the XEID is not a LISP destination.

If the requested XEID did not match either a configured delegation, an authoritative XEID-prefix or a "hint", then negative Map-Referral with action code NOT-AUTHORITATIVE MUST be returned.

7.2. DDT Map Server

When a DDT Map Server receives a DDT Map-Request, its operation is similar to that of a DDT node with additional processing as follows:

- o If the requested XEID matches a registered XEID-prefix, then the Map-Request is forwarded to one of the destination ETR RLOCs (or the Map Server sends a Map-Reply, if it is providing proxy Map-

Reply service) and a Map-Referral with the MS-ACK action MUST be returned to the sender of the DDT Map-Request.

- o If the requested XEID matches a configured XEID-prefix for which no ETR registration has been received then a negative Map-Referral with action code MS-NOT-REGISTERED MUST be returned to the sender of the DDT Map-Request.

7.3. DDT client

A DDT client queries one or more DDT nodes and uses an iterative process of following returned referrals until it receives one with action code MS-ACK (or an error indication). MS-ACK indicates that the Map-Request has been sent to a Map Server that will forward it to an ETR that, in turn, will provide a Map-Reply to the locator address in the Map-Request.

DDT client functionality will typically be implemented by DDT Map Resolvers. Just as any other Map Resolver, a DDT Map Resolver accepts Map-Requests from its clients (typically, ITRs) and ensures that those Map-Requests are forwarded to the correct ETR, which generates Map-Replies. Unlike a Map Resolver that uses the ALT mapping system (see [RFC6836]), however, a DDT Map Resolver implements a DDT client functionality to find the correct ETR to answer a Map-Request; this requires a DDT Map Resolver to maintain additional state: a Map-Referral cache and pending request list of XEIDs that are going through the iterative referral process.

DDT client functionality may be implemented on ITRs. In this case the DDT client will not receive Map-Request from another network element; instead, equivalent information will be provided to the DDT client by the means of programming interface.

7.3.1. Queuing and sending DDT Map-Requests

When a DDT client receives a request to resolve XEID (in case of DDT Map Resolver this will be in the form of received encapsulated Map-Request), it first performs a longest-match search for the XEID in its referral cache. If no match is found or if a negative cache entry is found, then the destination is not in the database; a negative Map-Reply MUST be returned and no further processing is performed by the DDT client.

If a match is found, the DDT client creates a pending request list entry for the XEID and saves the original request (in case of DDT Map-Resolved, original Map-Request minus the encapsulation header) along with other information needed to track progress through the iterative referral process; the "referral XEID-prefix" is also

initialized to indicate that no referral has yet been received. The DDT client then creates a DDT Map-Request (which is an encapsulated Map-Request with the "DDT-originated" flag set in the message header) for the XEID but without any authentication data that may have been included in the original request. It sends the DDT Map-Request to one of the RLOCs in the chosen referral cache entry. The referral cache is initially populated with one or more statically-configured entries; additional entries are added when referrals are followed, as described below. A DDT client is not absolutely required to cache referrals, but it doing so decreases latency and reduces lookup delays.

Note that in normal use on the public Internet, the statically-configured initial referral cache for a DDT client should include a "default" entry with RLOCs for either the DDT root node or one or more DDT nodes that contain hints for the DDT root node. If a DDT client does not have such configuration, it will return a Negative Map-Reply if it receives a query for an EID outside the subset of the mapping database known to it. While this may be desirable on private network deployments or during early transition to LISP when few sites are using it, this behavior is not appropriate when LISP is in general use on the Internet. If DDT message exchange is authenticated as described in Section 10 then DDT client MUST also be configured with public keys of DDT nodes pointed to by the "default" cache entry. In this case the "default" entry will typically be for the DDT root node.

7.3.2. Receiving and following referrals

After sending a DDT Map-Request, a DDT client expects to receive a Map-Referral response. If none occurs within the timeout period, the DDT client retransmits the request, sending to the next RLOC in the referral cache entry if one is available. If all RLOCs have been tried and the maximum number of retransmissions has occurred for each, then the pending request list entry is dequeued and discarded. In this case DDT client returns no response to sender of the original request.

A DDT client processes a received Map-Referral message according to each action code:

NODE-REFERRAL: The DDT client checks for a possible referral loop as described in Section 7.3.4. If no loop is found, the DDT client saves the prefix returned in the Map-Referral message in the referral cache, updates the saved prefix and saved RLOCs in the pending request list entry, and follows the referral by sending a new DDT Map-Request to one of the DDT node RLOCs listed

in the Referral Set; security information saved with the original Map-Request SHOULD NOT be included.

MS-REFERRAL: The DDT client processes an MS-REFERRAL in the same manner as a NODE-REFERRAL except that security information saved with the original Map-Request MUST be included in the new Map-Request sent to a Map Server (see Section 10 for details on security).

MS-ACK: This is returned by a DDT Map Server to indicate that it has one or more registered ETRs that can answer a Map-Request for the XEID and the request has been forwarded to one of them (or if the Map Server is doing proxy service for the prefix then reply has been sent to the querying ITR). If the pending request did not include saved LISP-SEC information or if that information was already included in the previous DDT Map-Request (sent by the DDT client in response to either an MS-REFERRAL or a previous MS-ACK referral), then the pending request for the XEID is complete; processing of the request stops and all request state can be discarded. Otherwise, LISP-SEC information is required and has not yet been sent to the authoritative DDT Map-Server; the DDT client MUST re-send the DDT Map-Request with LISP-SEC information included and the pending request queue entry remains until another Map-Referral with MS-ACK action code is received. If the "incomplete" flag is not set, the prefix is saved in the referral cache.

MS-NOT-REGISTERED: The DDT Map Server queried could not process the request because it did not have any ETRs registered for a matching, authoritative XEID-prefix. If the DDT client has not yet tried all of the RLOCs saved with the pending request, then it sends a Map-Request to the next RLOC in that list. If all RLOCs have been tried, then the destination XEID is not registered and is unreachable. The DDT client MUST return a negative Map-Reply to the requester (in case of DDT Map Resolver to the sender of original Map-Request); this Map-Reply contains the least-specific XEID-prefix in the range for which this DDT Map Server is authoritative and no registrations exist and whose TTL value SHOULD be set to one minute. A negative referral cache entry is created for the prefix (whose TTL also SHOULD be set to one minute) and processing of the request stops.

DELEGATION-HOLE: The DDT Map Server queried did not have an XEID-prefix defined that matched the requested XEID so it does not exist in the mapping database. The DDT client MUST return a negative Map-Reply to the requester (in case of DDT Map Resolver to the sender of original Map-Request); this Map-Reply SHOULD indicate the least-specific XEID-prefix matching the requested

XEID for which no delegations exist and SHOULD have a TTL value of 15 minutes. A negative referral cache entry is created for the prefix (which also SHOULD have TTL of 15 minutes) and processing of the pending request stops.

NOT-AUTHORITATIVE: The DDT Map Server queried is not authoritative for the requested XEID. This can occur if a cached referral has become invalid due to a change in the database hierarchy. If the DDT client receiving this message can determine that it is using old cached information, it MAY choose to delete that cached information and re-try the original Map-Request, starting from its "root" cache entry. If this action code is received in response to a query that did not use a cached referral information, then it indicates a database synchronization problem or configuration error. The pending request is silently discarded, i.e. all state for the request that caused this answer is removed and no answer is returned to the original requestor.

7.3.3. Handling referral errors

Other states are possible, such as a misconfigured DDT node (acting as a proxy Map Server, for example) returning a Map-Reply to the DDT client; they should be considered errors and logged as such. It is not clear exactly what else the DDT client should do in such cases; one possibility is to remove the pending request list entry and send a negative Map-Reply to the requester (in case of DDT Map Resolver to the sender of original Map-Request). Alternatively, if a DDT client detects unexpected behavior by a DDT node, it could mark that node as unusable in its referral cache and update the pending request to try a different DDT node if more than one is listed in the referral cache. In any case, any prefix contained in a Map-Referral message that causes a referral error (including a referral loop) is not saved in the DDT client referral cache.

7.3.4. Referral loop detection

In response to a Map-Referral message with action code NODE-REFERRAL or MS-REFERRAL, a DDT client is directed to query a new set of DDT node RLOCs that are expected to have more-specific XEID-prefix information for the requested XEID. To prevent a possible "iteration loop" (following referrals back-and-forth among a set of DDT nodes without ever finding an answer), a DDT client saves the last received referral XEID-prefix for each pending request and checks that a newly received NODE-REFERRAL or MS-REFERRAL message contains a more-specific referral XEID-prefix; an exact or less-specific match of the saved XEID-prefix indicates a referral loop. If a loop is detected, the DDT Map Resolver handles the request as described in Section 7.3.3. Otherwise, the DDT client saves the most recently

received referral XEID-prefix with the pending request when it follows the referral.

As an extra measure to prevent referral loops, it is probably also wise to limit the total number of referrals for any request to some reasonable number; the exact value of that number will be determined during experimental deployment of LISP-DDT, but is bounded by the maximum length of the XEID.

Note that when a DDT client adds an entry to its lookup queue and sends an initial Map-Request for an XEID, the queue entry has no previous referral XEID-prefix; this means that the first DDT node contacted by a DDT Map Resolver may provide a referral to anywhere in the DDT hierarchy. This, in turn, allows a DDT client to use essentially any DDT node RLOCs for its initial cache entries and depend on the initial referral to provide a good starting point for Map-Requests; there is no need to configure the same set of root DDT nodes on all DDT clients.

8. Pseudo Code and Decision Tree diagrams

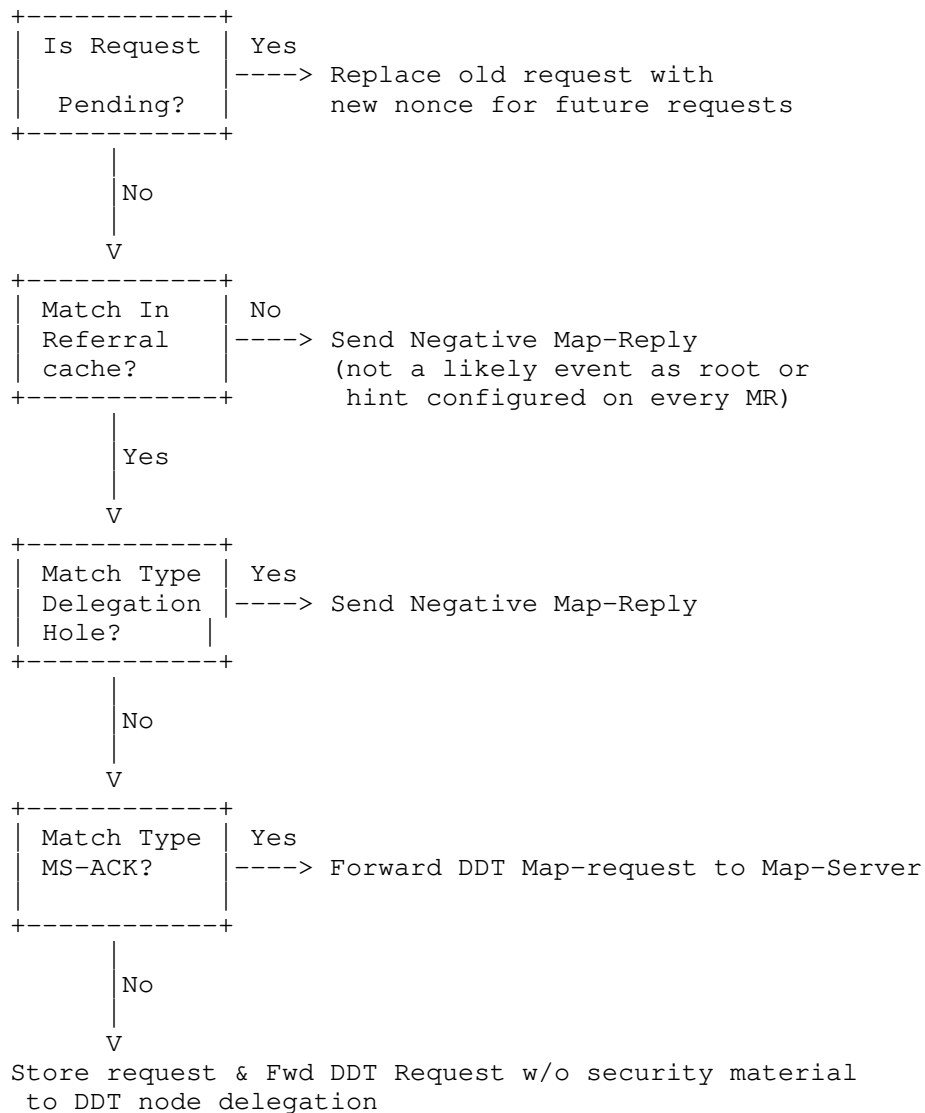
To illustrate DDT algorithms described above and to aid in implementation, each of the major DDT Map Server and DDT Map Resolver functions are described below, first using simple "pseudo-code" and then in the form of a decision tree.

8.1. Map Resolver processing of ITR Map-Request

8.1.1. Pseudo-code summary

```
if ( request pending i.e., (ITR,EID) of request same ) {  
    replace old request with new & use new request nonce  
    for future requests  
} else if ( no match in refcache ) {  
    return negative map-reply to ITR  
} else if ( match type delegation hole ) {  
    return negative map-reply to ITR  
} else if ( match type ms-ack ) {  
    fwd DDT request to map-server  
} else {  
    store & fwd DDT request w/o security material to node delegation  
}
```

8.1.2. Decision tree diagram



8.2. Map Resolver processing of Map-Referral message

8.2.1. Pseudo-code summary

```

if ( authentication signature validation failed ) {
    silently drop
}

if ( no request pending matched by referral nonce ) {

```

```
        silently drop
    }

    if ( pfx in referral less specific than last referral used ) {
        if ( gone through root ) {
            silently drop
        } else {
            send request to root
        }
    }
}

switch (map_referral_type) {

    case NOT_AUTHORITATIVE :
        if ( gone through root ) {
            return negative map-reply to ITR
        } else {
            send request to root
        }

    case DELEGATION_HOLE:
        cache & send negative map-reply to ITR

    case MS_REFERRAL:
        if ( referral equal to last used ) {
            if ( gone through root ) {
                return negative map-reply to ITR
            } else {
                send request to root
            }
        } else {
            cache
            follow the referral, include security material
        }

    case NODE_REFERRAL:
        if ( referral equal to last used ) {
            if ( gone through root ) {
                return negative map-reply to ITR
            } else {
                send request to root
            }
        } else {
            cache
            follow the referral, strip security material
        }

    case MS_ACK:
```

```

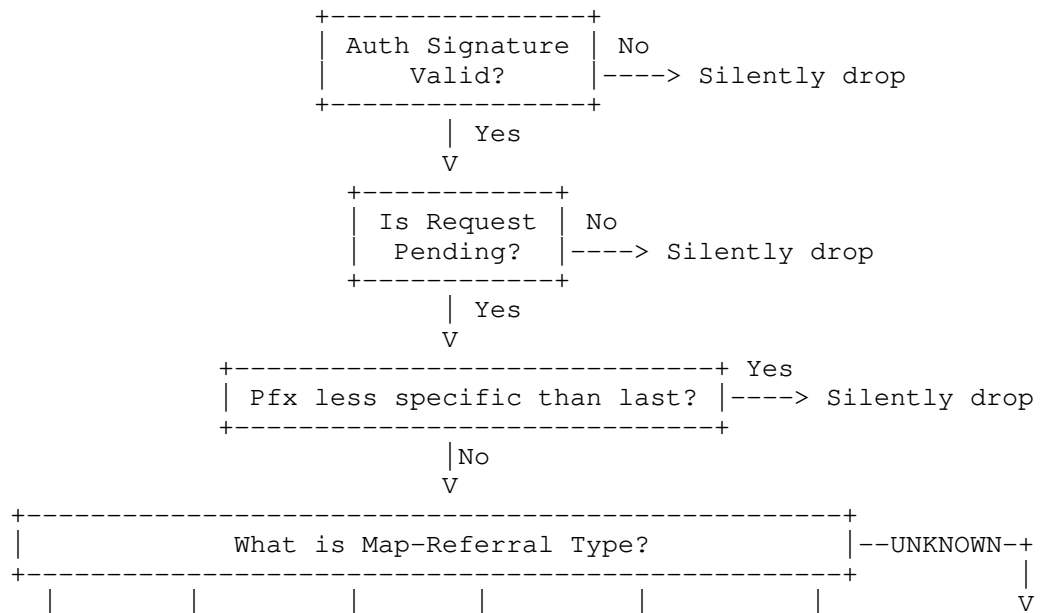
    if ( security material stripped ) {
        resend request with security material
        if { !incomplete } {
            cache
        }
    }

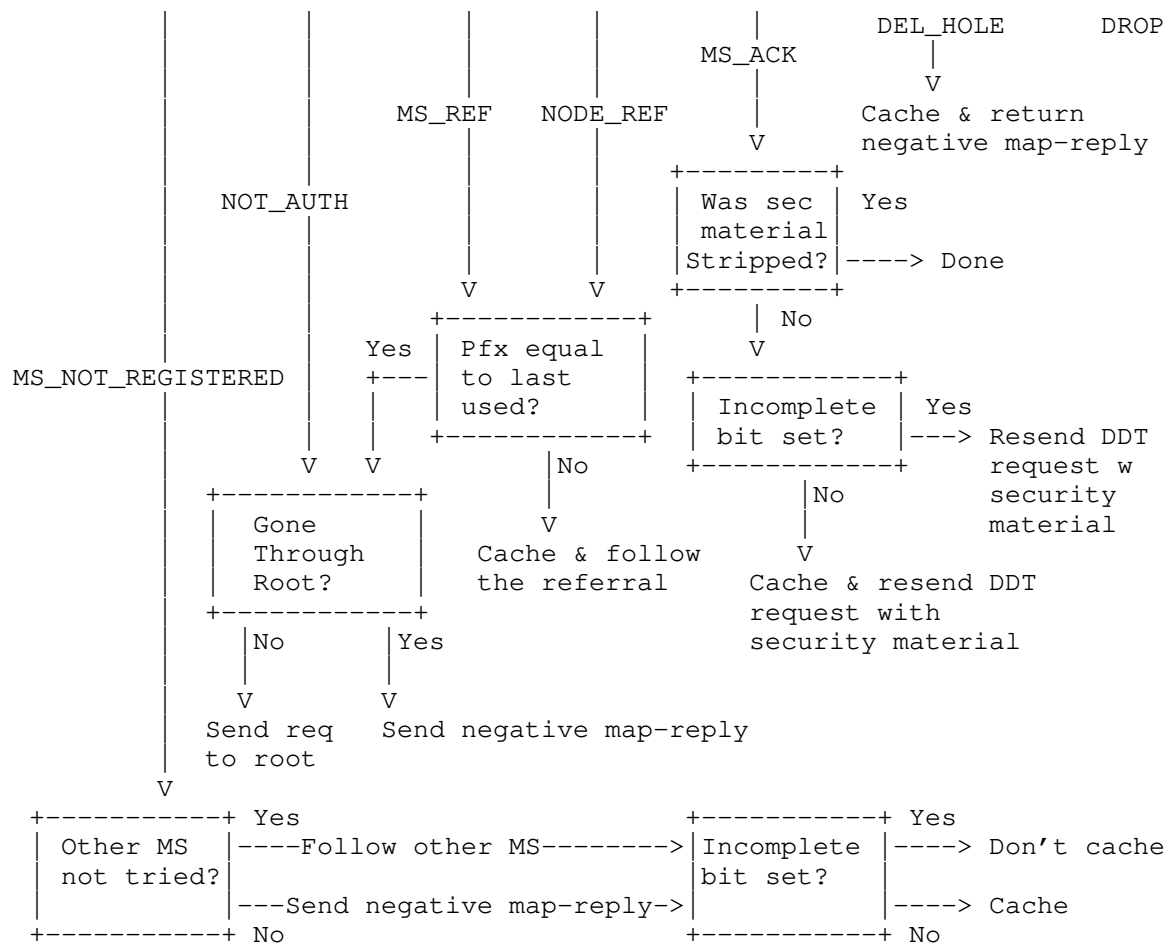
case MS_NOT_REGISTERED:
    if { all map-server delegations not tried } {
        follow delegations not tried
        if ( !incomplete ) {
            cache
        }
    } else {
        send negative map-reply to ITR
        if { !incomplete } {
            cache
        }
    }

case DEFAULT:
    drop
}

```

8.2.2. Decision tree diagram





8.3. DDT Node processing of DDT Map-Request message

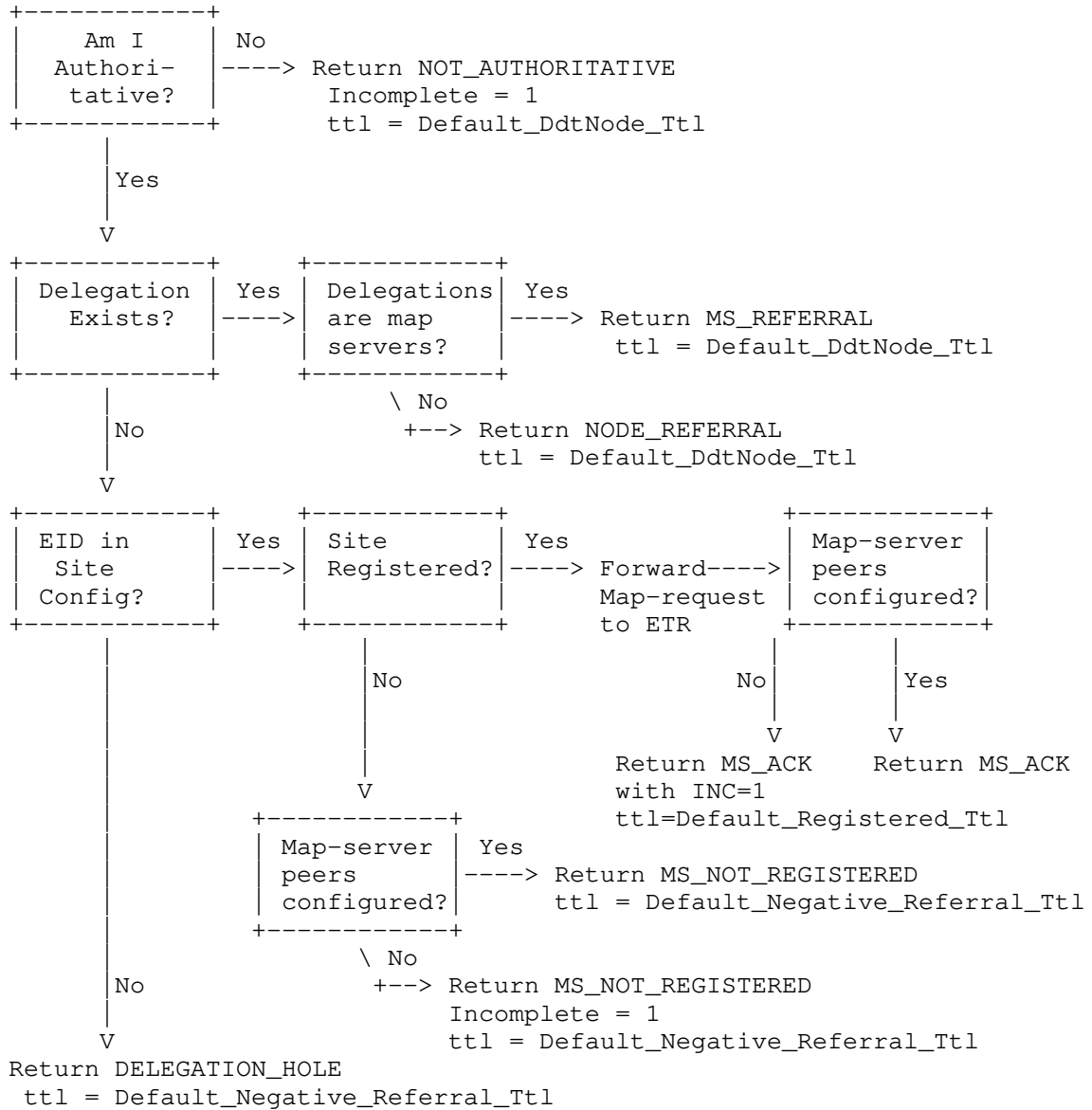
8.3.1. Pseudo-code summary

```
if ( I am not authoritative ) {
    send map-referral NOT_AUTHORITATIVE with
        incomplete bit set and ttl 0
} else if ( delegation exists ) {
    if ( delegated map-servers ) {
        send map-referral MS_REFERRAL with
            ttl 'Default_DdtNode_Ttl'
    } else {
        send map-referral NODE_REFERRAL with
            ttl 'Default_DdtNode_Ttl'
    }
} else {
    if ( eid in site ) {
        if ( site registered ) {
            forward map-request to etr
            if ( map-server peers configured ) {
                send map-referral MS_ACK with
                    ttl 'Default_Registered_Ttl'
            } else {
                send map-referral MS_ACK with
                    ttl 'Default_Registered_Ttl' and incomplete bit set
            }
        } else {
            if ( map-server peers configured ) {
                send map-referral MS_NOT_REGISTERED with
                    ttl 'Default_Configured_Not_Registered_Ttl'
            } else {
                send map-referral MS_NOT_REGISTERED with
                    ttl 'Default_Configured_Not_Registered_Ttl'
                    and incomplete bit set
            }
        }
    } else {
        send map-referral DELEGATION_HOLE with
            ttl 'Default_Negative_Referral_Ttl'
    }
}
```

where architectural constants for TTL are set as follows:

Default_DdtNode_Ttl	1440 minutes
Default_Registered_Ttl	1440 minutes
Default_Negative_Referral_Ttl	15 minutes
Default_Configured_Not_Registered_Ttl	1 minute

8.3.2. Decision tree diagram

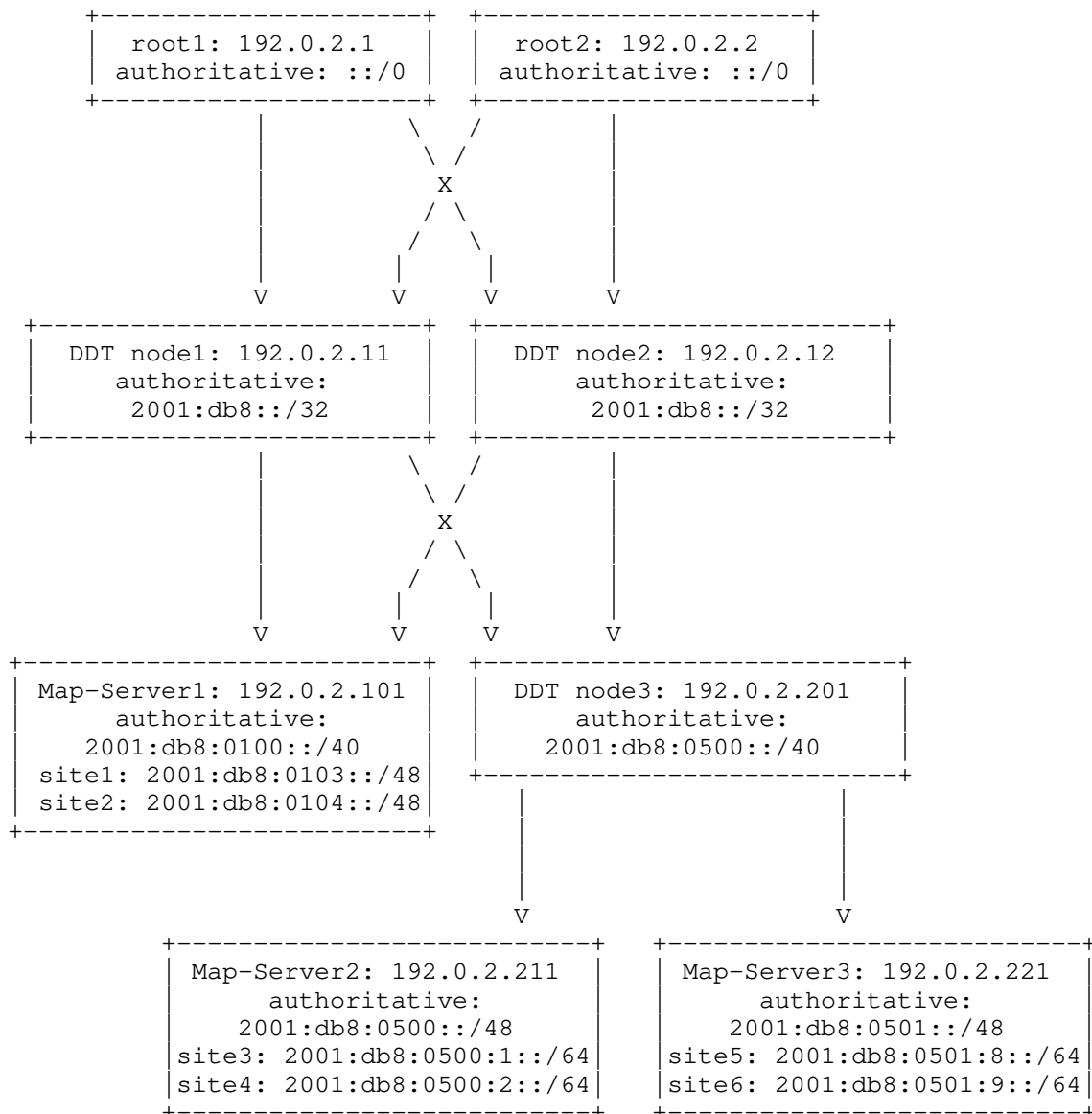


9. Example topology and request/referral following

This chapter shows example DDT tree and several possible scenarios of Map-Requests coming to a Map Resolver and subsequent iterative DDT referrals. For the sake of example RLOCs of DDT nodes are shown in

IPv4 address space while the EIDs are in IPv6 AF. The same principle of hierarchical delegation and pinpointing referrals is equally applicable to any AF whose address hierarchy can be expressed as a bitstring with associated length. DDT tree of IPv4 prefixes is another AF with immediate practical value.

To show how referrals are followed to find the RLOCs for a number of EIDs, consider the following example EID topology for DBID=0, IID=0, AFI=2, and EID=0/0



DDT nodes are configured for this "root" at IP addresses 192.0.2.1 and 192.0.2.2. DDT Map Resolvers are configured with default referral cache entries to these addresses.

The root DDT nodes delegate 2001:db8::/32 to two DDT nodes with IP addresses 192.0.2.11 and 192.0.2.12.

The DDT nodes for 2001:db8::/32 delegate 2001:db8:0100::/40 to a DDT Map Server with RLOC 192.0.2.101

The DDT Map Server for 2001:db8:0100::/40 is configured to allow ETRs to register the sub-prefixes 2001:db8:0103::/48 and 2001:db8:0104::/48

The DDT nodes for 2001:db8::/32 also delegate 2001:db8:0500::/40 to a DDT node with RLOC 192.0.2.201

The DDT node for 2001:db8:0500::/40 is further configured to delegate 2001:db8:0500::/48 to a DDT Map Server with RLOC 192.0.2.211 and 2001:db8:0501::/48 to a DDT Map Server with RLOC 192.0.2.221

The DDT Map Server for 2001:db8:0500::/48 is configured to allow ETRs to register the sub-prefixes 2001:db8:0500:1::/64 and 2001:db8:0500:2::/64

The DDT Map Server for 2001:db8:0501::/48 is configured to allow ETRs to register the sub-prefixes 2001:db8:0501:8::/64 and 2001:db8:0501:9::/64

9.1. Lookup of 2001:db8:0103:1::1/128

The first example shows a DDT Map Resolver following a delegation from the root to a DDT node followed by another delegation to a DDT Map Server.

ITR1 sends an Encapsulated Map-Request for 2001:db8:0103:1::1 to one of its configured (DDT) Map Resolvers. The DDT Map Resolver proceeds as follows:

1. Send DDT Map-Request (for 2001:db8:0103:1::1) to one of the root DDT nodes, 192.0.2.1 or 192.0.2.2
2. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8::/32, action code NODE-REFERRAL, RLOC set (192.0.2.11, 192.0.2.12)
3. Send DDT Map-Request to 192.0.2.11 or 192.0.2.12
4. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8:0100::/40, action code MS-REFERRAL, RLOC set (192.0.2.101)
5. Send DDT Map-Request to 192.0.2.101; if the ITR-originated Encapsulated Map-Request had a LISP-SEC signature, it is included

6. DDT Map Server at 192.0.2.101 decapsulates the DDT Map-Request and forwards to a registered site1 ETR for 2001:db8:0103::/48
 7. DDT Map Server at 192.0.2.101 sends a Map-Referral message for EID-prefix 2001:db8:0103::/48, action code MS-ACK to the DDT Map Resolver
 8. DDT Map Resolver receives Map-Referral message and dequeues the pending request for 2001:db8:0103:1::1
 9. site1 ETR for 2001:db8:0103::/48 receives Map-Request forwarded by DDT Map Server and sends Map-Reply to ITR1
- 9.2. Lookup of 2001:db8:0501:8:4::1/128

The next example shows a three-level delegation: root to first DDT node, first DDT node to second DDT node, second DDT node to DDT Map Server.

ITR2 sends an Encapsulated Map-Request for 2001:db8:0501:8:4::1 to one of its configured (DDT) Map Resolvers, which are different from those for ITR1. The DDT Map Resolver proceeds as follows:

1. Send DDT Map-Request (for 2001:db8:0501:8:4::1) to one of the root DDT nodes, 192.0.2.1 or 192.0.2.2
2. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8::/32, action code NODE-REFERRAL, RLOC set (192.0.2.11, 192.0.2.12)
3. Send DDT Map-Request to 192.0.2.11 or 192.0.2.12
4. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8:0500::/40, action code NODE-REFERRAL, RLOC set (192.0.2.201)
5. Send DDT Map-Request to 192.0.2.201
6. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8:0501::/48, action code MS-REFERRAL, RLOC set (192.0.2.221)
7. Send DDT Map-Request to 192.0.2.221; if the ITR-originated Encapsulated Map-Request had a LISP-SEC signature, it is included
8. DDT Map Server at 192.0.2.221 decapsulates the DDT Map-Request and forwards to a registered site5 ETR for 2001:db8:0501:8::/64

9. DDT Map Server at 192.0.2.221 sends a Map-Referral message for EID-prefix 2001:db8:0501:8::/64, action code MS-ACK, to the DDT Map Resolver
10. DDT Map Resolver receives Map-Referral(MS-ACK) message and dequeues the pending request for 2001:db8:0501:8:4::1
11. site5 ETR for 2001:db8:0501:8::/64 receives Map-Request forwarded by DDT Map Server and sends Map-Reply to ITR2

9.3. Lookup of 2001:db8:0104:2::2/128

This example shows how a DDT Map Resolver uses a saved referral cache entry to skip the referral process and go directly to a DDT Map Server for a prefix that is similar to one previously requested.

In this case, ITR1 uses the same Map Resolver used in example Section 9.1. It sends an Encapsulated Map-Request for 2001:db8:0104:2::2 to that (DDT) Map Resolver. The DDT Map-Resolver finds an MS-REFERRAL cache entry for 2001:db8:0100::/40 with RLOC set (192.0.2.101) and proceeds as follows:

1. Send DDT Map-Request (for 2001:db8:0104:2::2) to 192.0.2.101; if the ITR-originated Encapsulated Map-Request had a LISP-SEC signature, it is included
2. DDT Map Server at 192.0.2.101 decapsulates the DDT Map-Request and forwards to a registered site2 ETR for 2001:db8:0104::/48
3. DDT Map Server at 192.0.2.101 sends a Map-Referral message for EID-prefix 2001:db8:0104::/48, action code MS-ACK to the DDT Map Resolver
4. DDT Map Resolver receives Map-Referral(MS-ACK) and dequeues the pending request for 2001:db8:0104:2::2
5. site2 ETR for 2001:db8:0104::/48 receives Map-Request and sends Map-Reply to ITR1

9.4. Lookup of 2001:db8:0500:2:4::1/128

This example shows how a DDT Map Resolver uses a saved referral cache entry to start the referral process at a non-root, intermediate DDT node for a prefix that is similar to one previously requested.

In this case, ITR2 asks the same Map Resolver used in example Section 9.2. It sends an Encapsulated Map-Request for 2001:db8:0500:2:4::1 to that (DDT) Map Resolver, which finds a NODE-

REFERRAL cache entry for 2001:db8:0500::/40 with RLOC set (192.0.2.201). It proceeds as follows:

1. Send DDT Map-Request (for 2001:db8:0500:2:4::1) to 192.0.2.201
 2. Receive (and save in referral cache) Map-Referral for EID-prefix 2001:db8:0500::/48, action code MS-REFERRAL, RLOC set (192.0.2.211)
 3. Send DDT Map-Request to 192.0.2.211; if the ITR-originated Encapsulated Map-Request had a LISP-SEC signature, it is included
 4. DDT Map Server at 192.0.2.211 decapsulates the DDT Map-Request and forwards to a registered site4 ETR for 2001:db8:0500:2::/64
 5. DDT Map Server at 192.0.2.211 sends a Map-Referral message for EID-prefix 2001:db8:0500:2::/64, action code MS-ACK to the DDT Map Resolver
 6. DDT Map Resolver receives Map-Referral (MS-ACK) and dequeues the pending request for 2001:db8:0500:2:4::1
 7. site4 ETR for 2001:db8:0500:2::/64 receives Map-Request and sends Map-Reply to ITR2
- 9.5. Lookup of 2001:db8:0500::1/128 (non-existent EID)

This example uses the cached MS-REFERRAL for 2001:db8:0500::/48 learned above to start the lookup process at the DDT Map-Server at 192.0.2.211. The DDT Map Resolver proceeds as follows:

1. Send DDT Map-Request (for 2001:db8:0500::1) to 192.0.2.211; if the ITR-originated Encapsulated Map-Request had a LISP-SEC signature, it is included
2. DDT Map Server at 192.0.2.211, which is authoritative for 2001:db8:0500::/48, does not have a matching delegation for 2001:db8:0500::1. It responds with a Map-Referral message for 2001:db8:0500::/64, action code DELEGATION-HOLE to the DDT Map Resolver. The prefix 2001:db8:0500::/64 is used because it is the least-specific prefix that does match the requested EID, but does not match one of configured delegations (2001:db8:0500:1::/64 and 2001:db8:0500:2::/64).
3. DDT Map Resolver receives the delegation, adds a negative referral cache entry for 2001:db8:0500::/64, dequeues the pending request for 2001:db8:0500::1, and returns a negative Map-Reply to ITR2.

10. Securing the database and message exchanges

This section specifies the DDT security architecture that provides data origin authentication, data integrity protection, and XEID-prefix delegation. Global XEID-prefix authorization is out of the scope of this document.

Each DDT node is configured with one or more public/private key pair(s) that are used to digitally sign referral records for XEID-prefix(es) that the DDT node is authoritative for. In other words, each public/private key pair is associated with the combination of a DDT node and a XEID-prefix that it is authoritative for. Every DDT node is also configured with the public keys of its children DDT nodes. By including public keys of target child DDT nodes in the Map-Referral records, and signing each record with the DDT node's private key, a DDT node can securely delegate sub-prefixes of its authoritative XEID-prefixes to its children DDT nodes. A DDT node configured to provide hints must also have the public keys of the DDT nodes that its hints point to. DDT node keys can be encoded using LCAF type 11 to associate the key to the RLOC of the referred DDT node. If a node has more than one public key, it should sign its records with at least one of these keys. When a node has N keys, it can sustain up to N-1 key compromises. Revocation mechanism is described in Section 10.2.1.

Map Resolvers are configured with one or more trusted public keys referred to as trust anchors. Trust anchors are used to authenticate the DDT security infrastructure. Map Resolvers can discover a DDT node's public key either by having it configured as a trust anchor, or by obtaining it from the node's parent as part of a signed Map-Referral. When a public key is obtained from a node's parent, it is considered trusted if it is signed by a trust anchor, or if it is signed by a key that was previously trusted. Typically, in a Map Resolver, the root DDT node public keys should be configured as trust anchors. Once a Map Resolver authenticates a public key it locally caches the key along with the associated DDT node RLOC and XEID-prefix for future use.

10.1. XEID-prefix Delegation

In order to delegate XEID sub-prefixes to its children, a parent DDT node signs its Map-Referrals. Every signed Map-Referral MUST also include the public keys associated with each child DDT node. Such a signature indicates that the parent node is delegating the specified XEID-prefix to a given child DDT node. The signature is also authenticating the public keys associated with the children nodes, and authorizing them to be used by the children DDT nodes to provide origin authentication and integrity protection for further

delegations and mapping information of the XEID-prefix allocated to the DDT node.

As a result, for a given XEID-prefix, a Map Resolver can form an authentication chain from a configured trust anchor (typically the root DDT node) to the leaf nodes (Map Servers). Map Resolvers leverage this authentication chain to verify the Map-Referral signatures while walking the DDT tree until they reach a Map Server authoritative for the given XEID-prefix.

10.2. DDT node operation

Upon receiving a Map-Request, the DDT node responds with a Map-Referral as specified in Section 7. For every record present in the Map-Referral, the DDT node also includes the public keys associated with the record's XEID-prefix and the RLOCs of the children DDT nodes. Each record contained in the Map-Referral is signed using the DDT node's private key.

10.2.1. DDT public key revocation

The node that owns a public key can also revoke that public key. For instance if a parent node advertises a public key for one of its child DDT nodes, the child DDT node can at a later time revoke that key. Since DDT nodes do not keep track of the Map Resolvers that query them, revocation is done in a pull model, where the Map Resolver is informed of the revocation of a key only when it queries the node that owns that key. If the parent DDT is configured to advertise this key, the parent node must also be signaled to remove the key from the records it advertises for the child DDT node; this is necessary to avoid further distribution of the revoked key.

To securely revoke a key, the DDT node creates a new Record for the associated XEID-prefix and locator, including the revoked key with the R bit set. The DDT node must also include a signature in the Record that covers this record; this is computed using the private key corresponding to the key being revoked. Such a record is termed a "revocation record". By including this record in its Map-Referrals, the DDT node informs querying Map Resolvers about the revoked key. A digital signature computed with a revoked key can only be used to authenticate the revocation, and SHOULD NOT be used to validate any data. To prevent a compromised key from revoking other valid keys, a given key can only be used to sign a revocation for that specific key; it cannot be used to revoke other keys. This prevents the use of a compromised key to revoke other valid keys as described in [RFC5011]. A revocation record MUST be advertised for a period of time equal to or greater than the TTL value of the Record that initially advertised the key, starting from the time that the

advertisement of the key was stopped by removal from the parent DDT node.

10.3. Map Server operation

Similar to a DDT node, a Map Server is configured with one (or more) public/private key pairs that it must use to sign Map-Referrals.

However unlike DDT nodes, Map Servers do not delegate prefixes and as a result they do not need to include keys in the Map-Referrals they generate.

10.4. Map Resolver operation

Upon receiving a Map-Referral, the Map Resolver MUST first verify the signature(s) by using a trust anchor, or a previously authenticated public key, associated with the DDT node sending the Map-Referral. If multiple authenticated keys are associated with the DDT node sending this Map-Referral, the Key Tag field of the signature can be used to select the right public key for verifying the signature. If the key tag matches more than one key associated with that DDT node, the Map Resolver MUST try verifying the signature with all matching keys. For every matching key that is found the Map Resolver MUST also verify that the key is authoritative for the XEID-prefix in the Map-Referral record. If such a key is found, the Map Resolver MUST use it to verify the associated signature in the record. If no matching key is found, or if none of the matching keys is authoritative for the XEID-prefix in the Map-Referral record, or if such a key is found but the signature is not valid the Map-Referral record is considered corrupted and MUST be discarded. This may be due to expired keys. The Map Resolver MAY try other siblings of this node if there is an alternative node authoritative for the same prefix. If not, the Map Resolver CAN query the DDT node's parent to retrieve a valid key. It is good practice to use a counter or timer to avoid repeating this process if the resolver cannot verify the signature after several trials.

Once the signature is verified, the Map Resolver has verified the XEID-prefix delegation in the Map-Referral, and authenticated the public keys of the children DDT nodes. The Map Resolver must add these keys to the authenticated keys associated with each child DDT node and XEID-prefix. These keys are considered valid for the duration specified in the record's TTL field.

11. Open Issues and Considerations

There are a number of issues with the organization of the mapping database that need further investigation. Among these are:

- o Defining an interface to implement interconnection and/or interoperability with other mapping databases, such as LISP+ALT.
- o Additional key structures for use with LISP-DDT, such as to support additional EID formats as defined in [I-D.ietf-lisp-lcaf]
- o Management of the DDT Map Resolver referral cache, in particular, detecting and removing outdated entries.
- o Best practices for configuring hint referrals (or vice verse avoiding using them).

Operational experience will help answer open questions surrounding these and other issues.

12. IANA Considerations

This document makes no request of the IANA.

13. Security Considerations

Section 10 describes a DDT security architecture that provides data origin authentication, data integrity protection, and XEID-prefix delegation within the DDT Infrastructure.

Global XEID-prefix authorization is beyond the scope of this document, but the SIDR working group [RFC6480] is developing an infrastructure to support improved security of Internet routing. Further work is required to determine if SIDR's public key infrastructure (PKI) and the distributed repository system it uses for storing and disseminating PKI data objects may also be used by DDT devices to verifiably assert that they are the legitimate holders of a set of XEID prefixes.

This document specifies how DDT security and LISP-SEC ([I-D.ietf-lisp-sec]) complement one another to secure the DDT infrastructure, Map-Referral messages, and the Map-Request/Map-Reply protocols. In the future other LISP security mechanisms may be developed to replace LISP-SEC. Such future security mechanisms should describe how they can be used together with DDT to provide similar levels of protection.

LISP-SEC can use the DDT public key infrastructure to secure the transport of LISP-SEC key material (the One-Time Key) from a Map-Resolver to the corresponding Map-Server. For this reason, when LISP-SEC is deployed in conjunction with a LISP-DDT mapping database and the path between Map-Resolver and Map-Server needs to be protected, DDT security as described in Section 10 should be enabled as well.

14. References

14.1. Normative References

- [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.
- [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>>.
- [RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, DOI 10.17487/RFC3447, February 2003, <<http://www.rfc-editor.org/info/rfc3447>>.
- [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>>.
- [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>>.

14.2. Informative References

- [AFI] "Address Family Identifier (AFIs)", IANA , Febuary 2007, <<http://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml>>.
- [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.

[LISP-TREE]

Jakab, L., Cabellos-Aparicio, A., Coras, F., Saucez, D., and O. Bonaventure, "LISP-TREE: a DNS hierarchy to support the lisp mapping system", Selected Areas in Communications, IEEE Journal , 2010, <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5586446>.

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<http://www.rfc-editor.org/info/rfc1918>>.

[RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", STD 74, RFC 5011, DOI 10.17487/RFC5011, September 2007, <<http://www.rfc-editor.org/info/rfc5011>>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <<http://www.rfc-editor.org/info/rfc6480>>.

[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>>.

[RFC6837] Lear, E., "NERD: A Not-so-novel Endpoint ID (EID) to Routing Locator (RLOC) Database", RFC 6837, DOI 10.17487/RFC6837, January 2013, <<http://www.rfc-editor.org/info/rfc6837>>.

Appendix A. Acknowledgments

The authors would like to express their thanks to Lorand Jakab, Albert Cabellos-Asparicio, Florin Coras, Damien Saucez, and Olivier Bonaventure for their work on LISP-TREE [LISP-TREE] and LISP iterable mappings that inspired the hierarchical database structure and lookup iteration approach described in this document. Thanks also go to Dino Farinacci and Isidor Kouvelas for their implementation work; to Selina Heimlich and Srin Subramanian for testing; to Fabio Maino for work on security processing; and to Job Snijders, Glen Wiley, Neel Goyal, and Mike Gibbs for work on operational considerations and initial deployment of a prototype database infrastructure. Special thanks go to Jesper Skriver, Andrew Partan, and Noel Chiappa; all of whom have participated in (and put up with) seemingly endless hours of discussion of mapping database ideas, concepts, and issues.

Authors' Addresses

Vince Fuller

Email: vaf@vaf.net

Darrel Lewis
Cisco Systems

Email: darlewis@cisco.com

Vina Ermagan
Cisco Systems

Email: vermagan@cisco.com

Amit Jain
Juniper Networks

Email: atjain@juniper.net

Anton Smirnov
Cisco Systems

Email: as@cisco.com

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: July 21, 2014

L. Jakab
Cisco Systems
A. Cabellos-Aparicio
F. Coras
J. Domingo-Pascual
Technical University of
Catalonia
D. Lewis
Cisco Systems
January 17, 2014

LISP Network Element Deployment Considerations
draft-ietf-lisp-deployment-12.txt

Abstract

This document is a snapshot of different Locator/Identifier Separation Protocol (LISP) deployment scenarios. It discusses the placement of new network elements introduced by the protocol, representing the thinking of the LISP working group as of Summer 2013. LISP deployment scenarios may have evolved since. This memo represents one stable point in that evolution of understanding.

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 July 21, 2014.

Copyright Notice

Copyright (c) 2014 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	3
2. Tunnel Routers	4
2.1. Deployment Scenarios	4
2.1.1. Customer Edge	4
2.1.2. Provider Edge	6
2.1.3. Tunnel Routers Behind NAT	7
2.1.3.1. ITR	7
2.1.3.2. ETR	8
2.1.3.3. Additional Notes	8
2.2. Functional Models with Tunnel Routers	8
2.2.1. Split ITR/ETR	8
2.2.2. Inter-Service Provider Traffic Engineering	10
2.3. Summary and Feature Matrix	12
3. Map Resolvers and Map Servers	13
3.1. Map Servers	13
3.2. Map Resolvers	15
4. Proxy Tunnel Routers	16
4.1. P-ITR	16
4.2. P-ETR	17
5. Migration to LISP	18
5.1. LISP+BGP	18
5.2. Mapping Service Provider (MSP) P-ITR Service	19
5.3. Proxy-ITR Route Distribution (PITR-RD)	19
5.4. Migration Summary	22
6. Security Considerations	22
7. IANA Considerations	23
8. Acknowledgements	23
9. References	23
9.1. Normative References	23
9.2. Informative References	23
Appendix A. Step-by-Step Example BGP to LISP Migration Procedure	24
A.1. Customer Pre-Install and Pre-Turn-up Checklist	24
A.2. Customer Activating LISP Service	26
A.3. Cut-Over Provider Preparation and Changes	27
Authors' Addresses	27

1. Introduction

The Locator/Identifier Separation Protocol (LISP) is designed to address the scaling issues of the global Internet routing system identified in [RFC4984] by separating the current addressing scheme into Endpoint IDentifiers (EIDs) and Routing LOCators (RLOCs). The main protocol specification [RFC6830] describes how the separation is achieved, which new network elements are introduced, and details the packet formats for the data and control planes.

LISP assumes that such separation is between the edge and core and uses mapping and encapsulation for forwarding. While the boundary between both is not strictly defined, one widely accepted definition places it at the border routers of stub autonomous systems, which may carry a partial or complete default-free zone (DFZ) routing table. The initial design of LISP took this location as a baseline for protocol development. However, the applications of LISP go beyond just decreasing the size of the DFZ routing table, and include improved multihoming and ingress traffic engineering (TE) support for edge networks, and even individual hosts. Throughout the document we will use the term LISP site to refer to these networks/hosts behind a LISP Tunnel Router. We formally define the following two terms:

Network element: Facility or equipment used in the provision of a communications service over the Internet [TELCO96].

LISP site: A single host or a set of network elements in an edge network under the administrative control of a single organization, delimited from other networks by LISP Tunnel Router(s).

Since LISP is a protocol which can be used for different purposes, it is important to identify possible deployment scenarios and the additional requirements they may impose on the protocol specification and other protocols. Additionally, this document is intended as a guide for the operational community for LISP deployments in their networks. It is expected to evolve as LISP deployment progresses, and the described scenarios are better understood or new scenarios are discovered.

Each subsection considers an element type, discussing the impact of deployment scenarios on the protocol specification. For definition of terms, please refer to the appropriate documents (as cited in the respective sections).

This experimental document describing deployment considerations and the LISP specifications have areas that require additional experience and measurement. LISP is not recommended for deployment beyond experimental situations. Results of experimentation may lead to

modifications and enhancements of the LISP protocol mechanisms. Additionally, at the time of this writing there is no standardized security to implement. Beware that there are no counter measures for any of the threads identified in [I-D.ietf-lisp-threats]. See Section 15 [of RFC 6830] for specific, known issues that are in need of further work during development, implementation, and experimentation, and [I-D.ietf-lisp-threats] for recommendations to ameliorate the above-mentioned security threats.

2. Tunnel Routers

The device that is the gateway between the edge and the core is called a Tunnel Router (xTR), performing one or both of two separate functions:

1. Encapsulating packets originating from an end host to be transported over intermediary (transit) networks towards the other end-point of the communication
2. Decapsulating packets entering from intermediary (transit) networks, originated at a remote end host.

The first function is performed by an Ingress Tunnel Router (ITR), the second by an Egress Tunnel Router (ETR).

Section 8 of the main LISP specification [RFC6830] has a short discussion of where Tunnel Routers can be deployed and some of the associated advantages and disadvantages. This section adds more detail to the scenarios presented there, and provides additional scenarios as well. Furthermore this section discusses functional models, that is, network functions that can be achieved by deploying Tunnel Routers in specific ways.

2.1. Deployment Scenarios

2.1.1. Customer Edge

The first scenario we discuss is customer edge, when xTR functionality is placed on the router(s) that connect the LISP site to its upstream(s), but are under its control. As such, this is the most common expected scenario for xTRs, and this document considers it the reference location, comparing the other scenarios to this one.

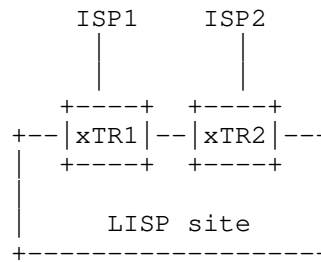


Figure 1: xTRs at the customer edge

From the LISP site perspective the main advantage of this type of deployment (compared to the one described in the next section) is having direct control over its ingress traffic engineering. This makes it easy to set up and maintain active/active, active/backup, or more complex TE policies, adding ISPs and additional xTRs at will, without involving third parties.

Being under the same administrative control, reachability information of all ETRs is easier to synchronize, because the necessary control traffic can be allowed between the locators of the ETRs. A correct synchronous global view of the reachability status is thus available, and the Locator Status Bits (Loc-Status-Bits, defined in [RFC6830]) can be set correctly in the LISP data header of outgoing packets.

By placing the tunnel router at the edge of the site, existing internal network configuration does not need to be modified. Firewall rules, router configurations and address assignments inside the LISP site remain unchanged. This helps with incremental deployment and allows a quick upgrade path to LISP. For larger sites with many external connections, distributed in geographically diverse points of presence (PoPs), and complex internal topology, it may however make more sense to both encapsulate and decapsulate as soon as possible, to benefit from the information in the IGP to choose the best path (see Section 2.2.1 for a discussion of this scenario).

Another thing to consider when placing tunnel routers is MTU issues. Encapsulation increases the amount of overhead associated with each packet. This added overhead decreases the effective end-to-end path MTU (unless fragmentation and reassembly is used). Some transit networks are known to provide larger MTU than the typical value of 1500 bytes of popular access technologies used at end hosts (e.g., IEEE 802.3 and 802.11). However, placing the LISP router connecting to such a network at the customer edge could possibly bring up MTU issues, depending on the link type to the provider as opposed to the following scenario. See [RFC4459] for MTU considerations of tunneling protocols on how to mitigate potential issues. Still, even

with these mitigations, path MTU issues are still possible.

2.1.2. Provider Edge

The other location at the core-edge boundary for deploying LISP routers is at the Internet service provider edge. The main incentive for this case is that the customer does not have to upgrade the CE router(s), or change the configuration of any equipment. Encapsulation/decapsulation happens in the provider's network, which may be able to serve several customers with a single device. For large ISPs with many residential/business customers asking for LISP this can lead to important savings, since there is no need to upgrade the software (or hardware, if it's the case) at each client's location. Instead, they can upgrade the software (or hardware) on a few PE routers serving the customers. This scenario is depicted in Figure 2.

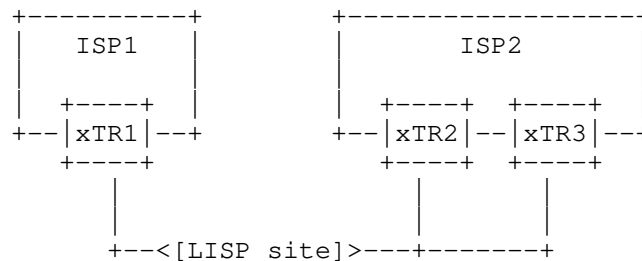


Figure 2: xTR at the PE

While this approach can make transition easy for customers and may be cheaper for providers, the LISP site loses one of the main benefits of LISP: ingress traffic engineering. Since the provider controls the ETRs, additional complexity would be needed to allow customers to modify their mapping entries.

The problem is aggravated when the LISP site is multihomed. Consider the scenario in Figure 2: whenever a change to TE policies is required, the customer contacts both ISP1 and ISP2 to make the necessary changes on the routers (if they provide this possibility). It is however unlikely, that both ISPs will apply changes simultaneously, which may lead to inconsistent state for the mappings of the LISP site. Since the different upstream ISPs are usually competing business entities, the ETRs may even be configured to compete, either to attract all the traffic or to get no traffic. The former will happen if the customer pays per volume, the latter if the connectivity has a fixed price. A solution could be to configure the Map Server(s) to do proxy-replying and have the Mapping Service Provider (MSP) apply policies.

Additionally, since xTR1, xTR2, and xTR3 are in different administrative domains, locator reachability information is unlikely to be exchanged among them, making it difficult to set Loc-Status-Bits (LSB) correctly on encapsulated packets. Because of this, and due to the security concerns about LSB described in [I-D.ietf-lisp-threats] their use is discouraged (set the L-bit to 0). Mapping versioning is another alternative [RFC6834].

Compared to the customer edge scenario, deploying LISP at the provider edge might have the advantage of diminishing potential MTU issues, because the tunnel router is closer to the core, where links typically have higher MTUs than edge network links.

2.1.3. Tunnel Routers Behind NAT

NAT in this section refers to IPv4 network address and port translation.

2.1.3.1. ITR

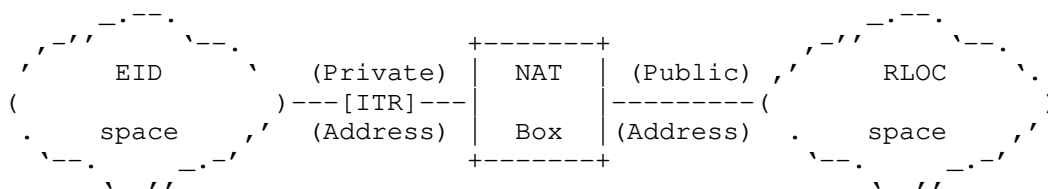


Figure 3: ITR behind NAT

Packets encapsulated by an ITR are just UDP packets from a NAT device's point of view, and they are handled like any UDP packet, there are no additional requirements for LISP data packets.

Map-Requests sent by an ITR, which create the state in the NAT table, have a different 5-tuple in the IP header than the Map-Reply generated by the authoritative ETR. Since the source address of this packet is different from the destination address of the request packet, no state will be matched in the NAT table and the packet will be dropped. To avoid this, the NAT device has to do the following:

- o Send all UDP packets with source port 4342, regardless of the destination port, to the RLOC of the ITR. The most simple way to achieve this is configuring 1:1 NAT mode from the external RLOC of the NAT device to the ITR's RLOC (Called "DMZ" mode in consumer broadband routers).

- o Rewrite the ITR-AFI and "Originating ITR RLOC Address" fields in the payload.

This setup supports only a single ITR behind the NAT device.

2.1.3.2. ETR

An ETR placed behind NAT is reachable from the outside by the Internet-facing locator of the NAT device. It needs to know this locator (and configure a loopback interface with it), so that it can use it in Map-Reply and Map-Register messages. Thus support for dynamic locators for the mapping database is needed in LISP equipment.

Again, only one ETR behind the NAT device is supported.

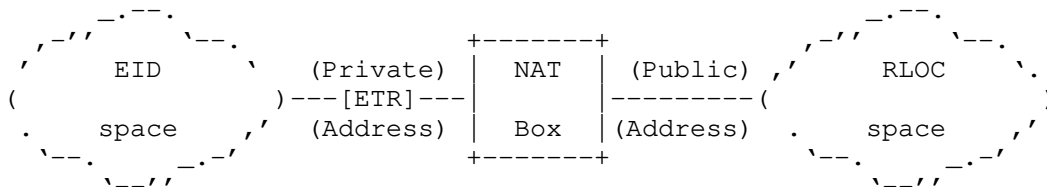


Figure 4: ETR behind NAT

2.1.3.3. Additional Notes

An implication of the issues described above is that LISP sites with xTRs can not be behind carrier based NATs, since two different sites would collide on the port forwarding. An alternative to static hole-punching to explore is the use of the Port Control Protocol (PCP) [RFC6887].

We only include this scenario due to completeness, to show that a LISP site can be deployed behind NAT, should it become necessary. However, LISP deployments behind NAT should be avoided, if possible.

2.2. Functional Models with Tunnel Routers

This section describes how certain LISP deployments can provide network functions.

2.2.1. Split ITR/ETR

In a simple LISP deployment, xTRs are located at the border of the LISP site (see Section 2.1.1). In this scenario packets are routed inside the domain according to the EID. However, more complex

networks may want to route packets according to the destination RLOC. This would enable them to choose the best egress point.

The LISP specification separates the ITR and ETR functionality and allows both entities to be deployed in separated network equipment. ITRs can be deployed closer to the host (i.e., access routers). This way packets are encapsulated as soon as possible, and egress point selection is driven by operational policy. In turn, ETRs can be deployed at the border routers of the network, and packets are decapsulated as soon as possible. Once decapsulated, packets are routed based on destination EID, according to internal routing policy.

In the following figure we can see an example. The Source (S) transmits packets using its EID and in this particular case packets are encapsulated at ITR_1. The encapsulated packets are routed inside the domain according to the destination RLOC, and can egress the network through the best point (i.e., closer to the RLOC's AS). On the other hand, inbound packets are received by ETR_1 which decapsulates them. Then packets are routed towards S according to the EID, again following the best path.

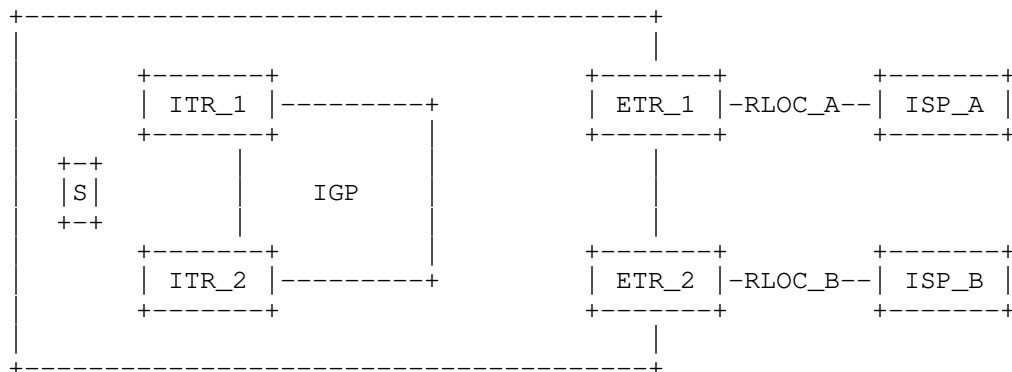


Figure 5: Split ITR/ETR Scenario

This scenario has a set of implications:

- o The site must carry more specific routes in order to choose the best egress point, and typically BGP is used for this, increasing the complexity of the network. However, this is usually already the case for LISP sites that would benefit from this scenario.
- o If the site is multihomed to different ISPs and any of the upstream ISPs are doing uRPF filtering, this scenario may become impractical. ITRs need to determine the exit ETR, for setting the

correct source RLOC in the encapsulation header. This adds complexity and reliability concerns.

- o In LISP, ITRs set the reachability bits when encapsulating data packets. Hence, ITRs need a mechanism to be aware of the liveness of all ETRs serving their site.
- o MTU within the site network must be large enough to accommodate encapsulated packets.
- o In this scenario, each ITR is serving fewer hosts than in the case when it is deployed at the border of the network. It has been shown that cache hit ratio grows logarithmically with the amount of users [CACHE]. Taking this into account, when ITRs are deployed closer to the host the effectiveness of the mapping cache may be lower (i.e., the miss ratio is higher). Another consequence of this is that the site may transmit a higher amount of Map-Requests, increasing the load on the distributed mapping database.
- o By placing the ITRs inside the site, they will still need global RLOCs, and this may add complexity to intra-site routing configuration, and further intra-site issues when there is a change of providers.

2.2.2. Inter-Service Provider Traffic Engineering

At the time of this writing, if two ISPs want to control their ingress TE policies for transit traffic between them, they need to rely on existing BGP mechanisms. This typically means deaggregating prefixes to choose on which upstream link packets should enter. This is either not feasible (if fine-grained per-customer control is required, the very specific prefixes may not be propagated) or increases DFZ table size.

Typically, LISP is seen applicable only to stub networks, however the LISP protocol can be also applied in a recursive manner, providing service provider ingress/egress TE capabilities without impacting the DFZ table size.

In order to implement this functionality with LISP consider the scenario depicted in Figure 6. The two ISPs willing to achieve ingress/egress TE are labeled as ISP_A and ISP_B, they are servicing Stub1 and Stub2 respectively, both are required to be LISP sites with their own xTRs. In this scenario we assume that Stub1 and Stub2 are communicating with each other and thus, ISP_A and ISP_B offer transit for such communications. ISP_A has RLOC_A1 and RLOC_A2 as upstream IP addresses while ISP_B has RLOC_B1 and RLOC_B2. The shared goal

among ISP_A and ISP_B is to control the transit traffic flow between RLOC_A1/A2 and RLOC_B1/B2.

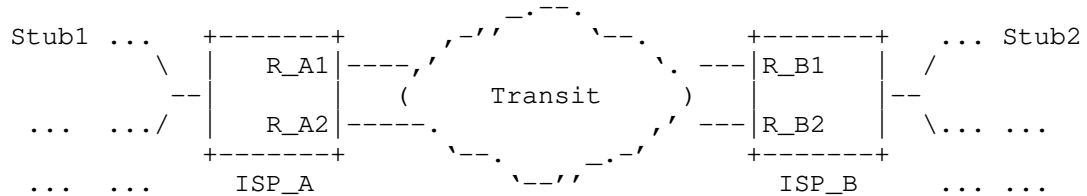


Figure 6: Inter-Service provider TE scenario

Both ISPs deploy xTRs on RLOC_A1/A2 and RLOC_B1/B2 respectively and reach a bilateral agreement to deploy their own private mapping system. This mapping system contains bindings between the RLOCs of Stub1 and Stub2 (owned by ISP_A and ISP_B respectively) and RLOC_A1/A2 and RLOC_B1/B2. Such bindings are in fact the TE policies between both ISPs and the convergence time is expected to be fast, since ISPs only have to update/query a mapping to/from the database.

The packet flow is as follows. First, a packet originated at Stub1 towards Stub2 is LISP encapsulated by Stub1's xTR. The xTR of ISP_A recursively encapsulates it and, according to the TE policies stored in the private mapping system, the ISP_A xTR chooses RLOC_B1 or RLOC_B2 as the outer encapsulation destination. Note that the packet transits between ISP_A and ISP_B double-encapsulated. Upon reception at the xTR of ISP_B the packet is decapsulated and sent towards Stub2 which performs the last decapsulation.

This deployment scenario, which uses recursive LISP, includes three important caveats. First, it is intended to be deployed between only two ISPs. If more than two ISPs use this approach, then the xTRs deployed at the participating ISPs must either query multiple mapping systems, or the ISPs must agree on a common shared mapping system. Furthermore, keeping this deployment scenario restricted to only two ISPs maintains the solution scalable, given that only two entities need to agree on using recursive LISP, and only one private mapping system is involved.

Second, the scenario is only recommended for ISPs providing connectivity to LISP sites, such that source RLOCs of packets to be recursively encapsulated belong to said ISP. Otherwise the participating ISPs must register prefixes they do not own in the above mentioned private mapping system. This results in either requiring complex authentication mechanisms or enabling simple traffic redirection attacks. Failure to follow these recommendations may lead to operational security issues when deploying this scenario.

And third, recursive encapsulation models are typically complex to troubleshoot and debug.

Besides these recommendations, the main disadvantages of this deployment case are:

- o Extra LISP header is needed. This increases the packet size and requires that the MTU between both ISPs accommodates double-encapsulated packets.
- o The ISP ITR must encapsulate packets and therefore must know the RLOC-to-RLOC binding. These bindings are stored in a mapping database and may be cached in the ITR's mapping cache. Cache misses lead to an additional lookup latency, unless a push based mapping system is used for the private mapping system.
- o The operational overhead of maintaining the shared mapping database.

2.3. Summary and Feature Matrix

When looking at the deployment scenarios and functional models above, there are several things to consider when choosing the appropriate one, depending on the type of the organization doing the deployment.

For home users and small site who wish to multihome and have control over their ISP options, the "CE" scenario offers the most advantages: it's simple to deploy, in some cases it only requires a software upgrade of the CPE, getting mapping service, and configuring the router. It retains control of TE and choosing upstreams by the user. It doesn't provide too many advantages to ISPs, due to the lessened dependence on their services in case of multihomed clients. It is also unlikely that ISP wishing to offer LISP to their customers will choose the "CE" placement: they need to send a technician to each customer, and potentially a new CPE. Even if they have remote control over the router, and a software upgrade could add LISP support, the operation is too risky.

For a network operator a good option to deploy is the "PE" scenario, unless a hardware upgrade is required for its edge routers to support LISP (in which case upgrading CPEs may be simpler). It retains control of TE, choice of PETR, and MS/MR. It also lowers potential MTU issues, as discussed above. Network operators should also explore the "Inter-SP TE" (recursive) functional model for their TE needs.

Large organizations can benefit the most from the "Split ITR/ETR" functional model, to optimize their traffic flow.

The following table gives a quick overview of the features supported by each of the deployment scenarios discussed above (marked with an "x") in the appropriate column: "CE" for customer edge, "PE" for provider edge, "Split" for split ITR/ETR, and "Recursive" for inter-service provider traffic engineering. The discussed features include:

Control of ingress TE: The scenario allows the LISP site to easily control LISP ingress traffic engineering policies.

No modifications to existing int. network infrastructure: The scenario doesn't require the LISP site to modify internal network configurations.

Loc-Status-Bits sync: The scenario allows easy synchronization of the Locator Status Bits.

MTU/PMTUD issues minimized: The scenario minimizes potential MTU and Path MTU Discovery issues.

Feature	CE	PE	Split	Recursive	NAT
Control of ingress TE	x	-	x	x	x
No modifications to existing int. network infrastructure	x	x	-	-	x
Loc-Status-Bits sync	x	-	x	x	-
MTU/PMTUD issues minimized	-	x	-	-	-

3. Map Resolvers and Map Servers

Map Resolvers and Map Servers make up the LISP mapping system and provide a means to find authoritative EID-to-RLOC mapping information, conforming to [RFC6833]. They are meant to be deployed in RLOC space, and their operation behind NAT is not supported.

3.1. Map Servers

The Map Server learns EID-to-RLOC mapping entries from an authoritative source and publishes them in the distributed mapping database. These entries are learned through authenticated Map-Register messages sent by authoritative ETRs. Also, upon reception of a Map-Request, the Map Server verifies that the destination EID matches an EID-prefix for which it is authoritative for, and then re-encapsulates and forwards it to a matching ETR. Map Server functionality is described in detail in [RFC6833].

The Map Server is provided by a Mapping Service Provider (MSP). The MSP participates in the global distributed mapping database infrastructure, by setting up connections to other participants, according to the specific mapping system that is employed (e.g., ALT [RFC6836], DDT [I-D.ietf-lisp-ddt]). Participation in the mapping database, and the storing of EID-to-RLOC mapping data is subject to the policies of the "root" operators, who should check ownership rights for the EID prefixes stored in the database by participants. These policies are out of the scope of this document.

The LISP DDT protocol is used by LISP Mapping Service providers to provide reachability between those providers' Map-Resolvers and Map-Servers. The DDT Root is currently operated by a collection of organizations on an open basis. See [DDT-ROOT] for more details. Similarly to the DNS root, it has several different server instances using names of the letters of the Greek alphabet (alpha, delta, etc.), operated by independent organizations. When this document was published, there were 5 such instances, one of them being anycasted. The Root provides the list of server instances on their web site and configuration files for several map server implementations. The DDT Root, and LISP Mapping Providers both rely on and abide by existing allocation policies by Regional Internet Registries to determine prefix ownership for use as EIDs.

It is expected that the DDT root organizations will continue to evolve in response to experimentation with LISP deployments for Internet edge multi-homing and VPN use cases.

In all cases, the MSP configures its Map Server(s) to publish the prefixes of its clients in the distributed mapping database and start encapsulating and forwarding Map-Requests to the ETRs of the AS. These ETRs register their prefix(es) with the Map Server(s) through periodic authenticated Map-Register messages. In this context, for some LISP sites, there is a need for mechanisms to:

- o Automatically distribute EID prefix(es) shared keys between the ETRs and the EID-registrar Map Server.
- o Dynamically obtain the address of the Map Server in the ETR of the AS.

The Map Server plays a key role in the reachability of the EID-prefixes it is serving. On the one hand it is publishing these prefixes into the distributed mapping database and on the other hand it is encapsulating and forwarding Map-Requests to the authoritative ETRs of these prefixes. ITRs encapsulating towards EIDs under the responsibility of a failed Map Server will be unable to look up any of their covering prefixes. The only exception are the ITRs that

already contain the mappings in their local cache. In this case ITRs can reach ETRs until the entry expires (typically 24 hours). For this reason, redundant Map Server deployments are desirable. A set of Map Servers providing high-availability service to the same set of prefixes is called a redundancy group. ETRs are configured to send Map-Register messages to all Map Servers in the redundancy group. The configuration for fail-over (or load-balancing, if desired) among the members of the group depends on the technology behind the mapping system being deployed. Since ALT is based on BGP and DDT was inspired from the Domain Name System (DNS), deployments can leverage current industry best practices for redundancy in BGP and DNS. These best practices are out of the scope of this document.

Additionally, if a Map Server has no reachability for any ETR serving a given EID block, it should not originate that block into the mapping system.

3.2. Map Resolvers

A Map Resolver is a network infrastructure component which accepts LISP encapsulated Map-Requests, typically from an ITR, and finds the appropriate EID-to-RLOC mapping by consulting the distributed mapping database. Map Resolver functionality is described in detail in [RFC6833].

Anyone with access to the distributed mapping database can set up a Map Resolver and provide EID-to-RLOC mapping lookup service. Database access setup is mapping system specific.

For performance reasons, it is recommended that LISP sites use Map Resolvers that are topologically close to their ITRs. ISPs supporting LISP will provide this service to their customers, possibly restricting access to their user base. LISP sites not in this position can use open access Map Resolvers, if available. However, regardless of the availability of open access resolvers, the MSP providing the Map Server(s) for a LISP site should also make available Map Resolver(s) for the use of that site.

In medium to large-size ASes, ITRs must be configured with the RLOC of a Map Resolver, operation which can be done manually. However, in Small Office Home Office (SOHO) scenarios a mechanism for autoconfiguration should be provided.

One solution to avoid manual configuration in LISP sites of any size is the use of anycast RLOCs [RFC4786] for Map Resolvers similar to the DNS root server infrastructure. Since LISP uses UDP encapsulation, the use of anycast would not affect reliability. LISP routers are then shipped with a preconfigured list of well know Map

Resolver RLOCs, which can be edited by the network administrator, if needed.

The use of anycast also helps improve mapping lookup performance. Large MSPs can increase the number and geographical diversity of their Map Resolver infrastructure, using a single anycasted RLOC. Once LISP deployment is advanced enough, very large content providers may also be interested running this kind of setup, to ensure minimal connection setup latency for those connecting to their network from LISP sites.

While Map Servers and Map Resolvers implement different functionalities within the LISP mapping system, they can coexist on the same device. For example, MSPs offering both services, can deploy a single Map Resolver/Map Server in each PoP where they have a presence.

4. Proxy Tunnel Routers

4.1. P-ITR

Proxy Ingress Tunnel Routers (P-ITRs) are part of the non-LISP/LISP transition mechanism, allowing non-LISP sites to reach LISP sites. They announce via BGP certain EID prefixes (aggregated, whenever possible) to attract traffic from non-LISP sites towards EIDs in the covered range. They do the mapping system lookup, and encapsulate received packets towards the appropriate ETR. Note that for the reverse path LISP sites can reach non-LISP sites simply by not encapsulating traffic. See [RFC6832] for a detailed description of P-ITR functionality.

The success of new protocols depends greatly on their ability to maintain backwards compatibility and inter-operate with the protocol(s) they intend to enhance or replace, and on the incentives to deploy the necessary new software or equipment. A LISP site needs an interworking mechanism to be reachable from non-LISP sites. A P-ITR can fulfill this role, enabling early adopters to see the benefits of LISP, similar to tunnel brokers helping the transition from IPv4 to IPv6. A site benefits from new LISP functionality (proportionally with existing global LISP deployment) when going LISP, so it has the incentives to deploy the necessary tunnel routers. In order to be reachable from non-LISP sites it has two options: keep announcing its prefix(es) with BGP, or have a P-ITR announce prefix(es) covering them.

If the goal of reducing the DFZ routing table size is to be reached, the second option is preferred. Moreover, the second option allows

LISP-based ingress traffic engineering from all sites. However, the placement of P-ITRs significantly influences performance and deployment incentives. Section 5 is dedicated to the migration to a LISP-enabled Internet, and includes deployment scenarios for P-ITRs.

4.2. P-ETR

In contrast to P-ITRs, P-ETRs are not required for the correct functioning of all LISP sites. There are two cases, where they can be of great help:

- o LISP sites with unicast reverse path forwarding (uRPF) restrictions, and
- o Communication between sites using different address family RLOCs.

In the first case, uRPF filtering is applied at their upstream PE router. When forwarding traffic to non-LISP sites, an ITR does not encapsulate packets, leaving the original IP headers intact. As a result, packets will have EIDs in their source address. Since we are discussing the transition period, we can assume that a prefix covering the EIDs belonging to the LISP site is advertised to the global routing tables by a P-ITR, and the PE router has a route towards it. However, the next hop will not be on the interface towards the CE router, so non-encapsulated packets will fail uRPF checks.

To avoid this filtering, the affected ITR encapsulates packets towards the locator of the P-ETR for non-LISP destinations. Now the source address of the packets, as seen by the PE router is the ITR's locator, which will not fail the uRPF check. The P-ETR then decapsulates and forwards the packets.

The second use case is IPv4-to-IPv6 transition. Service providers using older access network hardware, which only supports IPv4 can still offer IPv6 to their clients, by providing a CPE device running LISP, and P-ETR(s) for accessing IPv6-only non-LISP sites and LISP sites, with IPv6-only locators. Packets originating from the client LISP site for these destinations would be encapsulated towards the P-ETR's IPv4 locator. The P-ETR is in a native IPv6 network, decapsulating and forwarding packets. For non-LISP destination, the packet travels natively from the P-ETR. For LISP destinations with IPv6-only locators, the packet will go through a P-ITR, in order to reach its destination.

For more details on P-ETRs see [RFC6832].

P-ETRs can be deployed by ISPs wishing to offer value-added services

to their customers. As is the case with P-ITRs, P-ETRs too may introduce path stretch (the ratio between the cost of the selected path and that of the optimal path). Because of this the ISP needs to consider the tradeoff of using several devices, close to the customers, to minimize it, or few devices, farther away from the customers, minimizing cost instead.

Since the deployment incentives for P-ITRs and P-ETRs are different, it is likely they will be deployed in separate devices, except for the CDN case, which may deploy both in a single device.

In all cases, the existence of a P-ETR involves another step in the configuration of a LISP router. CPE routers, which are typically configured by DHCP, stand to benefit most from P-ETRs. Autoconfiguration of the P-ETR locator could be achieved by a DHCP option, or adding a P-ETR field to either Map-Notifys or Map-Replies.

5. Migration to LISP

This section discusses a deployment architecture to support the migration to a LISP-enabled Internet. The loosely defined terms of "early transition phase", "late transition phase", and "LISP Internet phase" refer to time periods when LISP sites are a minority, a majority, or represent all edge networks respectively.

5.1. LISP+BGP

For sites wishing to go LISP with their PI prefix the least disruptive way is to upgrade their border routers to support LISP, register the prefix into the LISP mapping system, but keep announcing it with BGP as well. This way LISP sites will reach them over LISP, while legacy sites will be unaffected by the change. The main disadvantage of this approach is that no decrease in the DFZ routing table size is achieved. Still, just increasing the number of LISP sites is an important gain, as an increasing LISP/non-LISP site ratio may decrease the need for BGP-based traffic engineering that leads to prefix deaggregation. That, in turn, may lead to a decrease in the DFZ size and churn in the late transition phase.

This scenario is not limited to sites that already have their prefixes announced with BGP. Newly allocated EID blocks could follow this strategy as well during the early LISP deployment phase, depending on the cost/benefit analysis of the individual networks. Since this leads to an increase in the DFZ size, the following architecture should be preferred for new allocations.

5.2. Mapping Service Provider (MSP) P-ITR Service

In addition to publishing their clients' registered prefixes in the mapping system, MSPs with enough transit capacity can offer them P-ITR service as a separate service. This service is especially useful for new PI allocations, to sites without existing BGP infrastructure, that wish to avoid BGP altogether. The MSP announces the prefix into the DFZ, and the client benefits from ingress traffic engineering without prefix deaggregation. The downside of this scenario is adding path stretch.

Routing all non-LISP ingress traffic through a third party which is not one of its ISPs is only feasible for sites with modest amounts of traffic (like those using the IPv6 tunnel broker services today), especially in the first stage of the transition to LISP, with a significant number of legacy sites. This is because the handling of said traffic is likely to result in additional costs, which would be passed down to the client. When the LISP/non-LISP site ratio becomes high enough, this approach can prove increasingly attractive.

Compared to LISP+BGP, this approach avoids DFZ bloat caused by prefix deaggregation for traffic engineering purposes, resulting in slower routing table increase in the case of new allocations and potential decrease for existing ones. Moreover, MSPs serving different clients with adjacent aggregatable prefixes may lead to additional decrease, but quantifying this decrease is subject to future research study.

5.3. Proxy-ITR Route Distribution (PITR-RD)

Instead of a LISP site, or the MSP, announcing their EIDs with BGP to the DFZ, this function can be outsourced to a third party, a P-ITR Service Provider (PSP). This will result in a decrease of the operational complexity both at the site and at the MSP.

The PSP manages a set of distributed P-ITR(s) that will advertise the corresponding EID prefixes through BGP to the DFZ. These P-ITR(s) will then encapsulate the traffic they receive for those EIDs towards the RLOCs of the LISP site, ensuring their reachability from non-LISP sites.

While it is possible for a PSP to manually configure each client's EID routes to be announced, this approach offers little flexibility and is not scalable. This section presents a scalable architecture that offers automatic distribution of EID routes to LISP sites and service providers.

The architecture requires no modification to existing LISP network elements, but it introduces a new (conceptual) network element, the

EID Route Server, defined as a router that either propagates routes learned from other EID Route Servers, or it originates EID Routes. The EID-Routes that it originates are those that it is authoritative for. It propagates these routes to Proxy-ITRs within the AS of the EID Route Server. It is worth to note that a BGP capable router can be also considered as an EID Route Server.

Further, an EID-Route is defined as a prefix originated via the Route Server of the mapping service provider, which should be aggregated if the MSP has multiple customers inside a single large continuous prefix. This prefix is propagated to other P-ITRs both within the MSP and to other P-ITR operators it peers with. EID Route Servers are operated either by the LISP site, MSPs or PSPs, and they may be collocated with a Map Server or P-ITR, but are a functionally discrete entity. They distribute EID-Routes, using BGP, to other domains, according to policies set by participants.

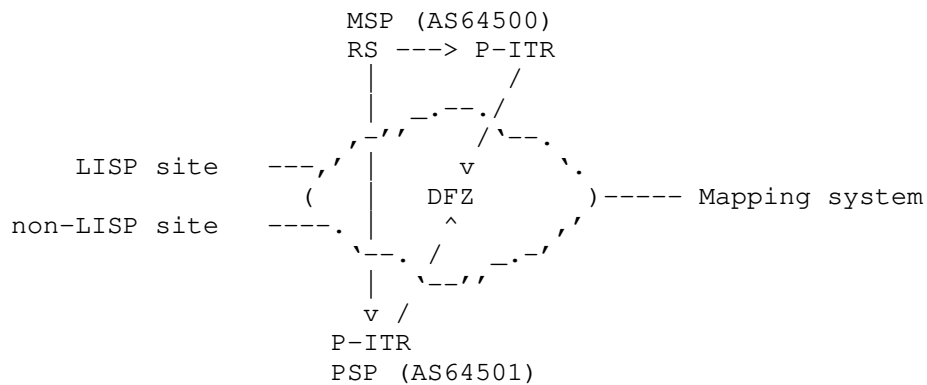


Figure 7: The P-ITR Route Distribution architecture

The architecture described above decouples EID origination from route propagation, with the following benefits:

- o Can accurately represent business relationships between P-ITR operators
- o More mapping system agnostic
- o Minor changes to P-ITR implementation, no changes to other components

In the example in the figure we have a MSP providing services to the LISP site. The LISP site does not run BGP, and gets an EID allocation directly from a RIR, or from the MSP, who may be a LIR. Existing PI allocations can be migrated as well. The MSP ensures the

presence of the prefix in the mapping system, and runs an EID Route Server to distribute it to P-ITR service providers. Since the LISP site does not run BGP, the prefix will be originated with the AS number of the MSP.

In the simple case depicted in Figure 7 the EID-Route of LISP site will be originated by the Route Server, and announced to the DFZ by the PSP's P-ITRs with AS path 64501 64500. From that point on, the usual BGP dynamics apply. This way, routes announced by P-ITR are still originated by the authoritative Route Server. Note that the peering relationships between MSP/PSPs and those in the underlying forwarding plane may not be congruent, making the AS path to a P-ITR shorter than it is in reality.

The non-LISP site will select the best path towards the EID-prefix, according to its local BGP policies. Since AS-path length is usually an important metric for selecting paths, a careful placement of P-ITR could significantly reduce path-stretch between LISP and non-LISP sites.

The architecture allows for flexible policies between MSP/PSPs. Consider the EID Route Server networks as control plane overlays, facilitating the implementation of policies necessary to reflect the business relationships between participants. The results are then injected to the common underlying forwarding plane. For example, some MSP/PSPs may agree to exchange EID-Prefixes and only announce them to each of their forwarding plane customers. Global reachability of an EID-prefix depends on the MSP the LISP site buys service from, and is also subject to agreement between the mentioned parties.

In terms of impact on the DFZ, this architecture results in a slower routing table increase for new allocations, since traffic engineering will be done at the LISP level. For existing allocations migrating to LISP, the DFZ may decrease since MSPs may be able to aggregate the prefixes announced.

Compared to LISP+BGP, this approach avoids DFZ bloat caused by prefix deaggregation for traffic engineering purposes, resulting in slower routing table increase in the case of new allocations and potential decrease for existing ones. Moreover, MSPs serving different clients with adjacent aggregatable prefixes may lead to additional decrease, but quantifying this decrease is subject to future research study.

The flexibility and scalability of this architecture does not come without a cost however: A PSP operator has to establish either transit or peering relationships to improve their connectivity.

5.4. Migration Summary

Registering a domain name typically entails an annual fee that should cover the operating expenses for publishing the domain in the global DNS. The situation is similar with several other registration services. A LISP mapping service provider (MSR) client publishing an EID prefix in the LISP mapping system has the option of signing up for P-ITR services as well, for an extra fee. These services may be offered by the MSP itself, but it is expected that specialized P-ITR service providers (PSPs) will do it. Clients not signing up become responsible for getting non-LISP traffic to their EIDs (using the LISP+BGP scenario).

Additionally, Tier 1 ISPs have incentives to offer P-ITR services to non-subscribers in strategic places just to attract more traffic from competitors, thus more revenue.

The following table presents the expected effects of the different transition scenarios during a certain phase on the DFZ routing table size:

Phase	LISP+BGP	MSP P-ITR	P-ITR-RD
Early transition	no change	slower increase	slower increase
Late transition	may decrease	slower increase	slower increase
LISP Internet	considerable decrease		

It is expected that P-ITR-RD will co-exist with LISP+BGP during the migration, with the latter being more popular in the early transition phase. As the transition progresses and the MSP P-ITR and P-ITR-RD ecosystem gets more ubiquitous, LISP+BGP should become less attractive, slowing down the increase of the number of routes in the DFZ.

Note that throughout Section 5 we focused on the effects of LISP deployment on the DFZ route table size. Other metrics may be impacted as well, but to the best of our knowledge have not been measured as of yet.

6. Security Considerations

All security implications of LISP deployments are to be discussed in separate documents. [I-D.ietf-lisp-threats] gives an overview of LISP threat models, including ETR operators attracting traffic by overclaiming an EID-prefix (Section 4.4.3). Securing mapping lookups is discussed in [I-D.ietf-lisp-sec].

7. IANA Considerations

This memo includes no request to IANA.

8. Acknowledgements

Many thanks to Margaret Wasserman for her contribution to the IETF76 presentation that kickstarted this work. The authors would also like to thank Damien Saucez, Luigi Iannone, Joel Halpern, Vince Fuller, Dino Farinacci, Terry Manderson, Noel Chiappa, Hannu Flinck, Paul Vinciguerra, Fred Templin, Brian Haberman, and everyone else who provided input.

9. References

9.1. Normative References

- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, January 2013.
- [RFC6832] Lewis, D., Meyer, D., Farinacci, D., and V. Fuller, "Interworking between Locator/ID Separation Protocol (LISP) and Non-LISP Sites", RFC 6832, January 2013.
- [RFC6833] Fuller, V. and D. Farinacci, "Locator/ID Separation Protocol (LISP) Map-Server Interface", RFC 6833, January 2013.

9.2. Informative References

- [CACHE] Jung, J., Sit, E., Balakrishnan, H., and R. Morris, "DNS performance and the effectiveness of caching", 2002.
- [DDT-ROOT] "DDT Root", <<http://ddt-root.org/>>.
- [I-D.ietf-lisp-ddt] Fuller, V., Lewis, D., Ermagan, V., and A. Jain, "LISP Delegated Database Tree", draft-ietf-lisp-ddt-01 (work in progress), March 2013.
- [I-D.ietf-lisp-sec] Maino, F., Ermagan, V., Cabellos-Aparicio, A., Saucez, D., and O. Bonaventure, "LISP-Security (LISP-SEC)", draft-ietf-lisp-sec-05 (work in progress), October 2013.

- [I-D.ietf-lisp-threats]
Saucez, D., Iannone, L., and O. Bonaventure, "LISP Threats Analysis", draft-ietf-lisp-threats-08 (work in progress), October 2013.
- [RFC4459] Savola, P., "MTU and Fragmentation Issues with In-the-Network Tunneling", RFC 4459, April 2006.
- [RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, December 2006.
- [RFC4984] Meyer, D., Zhang, L., and K. Fall, "Report from the IAB Workshop on Routing and Addressing", RFC 4984, September 2007.
- [RFC6834] Iannone, L., Saucez, D., and O. Bonaventure, "Locator/ID Separation Protocol (LISP) Map-Versioning", RFC 6834, January 2013.
- [RFC6836] Fuller, V., Farinacci, D., Meyer, D., and D. Lewis, "Locator/ID Separation Protocol Alternative Logical Topology (LISP+ALT)", RFC 6836, January 2013.
- [RFC6887] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, April 2013.
- [TELCO96] "Telecommunications Act of 1996", 1996.

Appendix A. Step-by-Step Example BGP to LISP Migration Procedure

To help the operational community deploy LISP, this informative section offers a step-by-step guide for migrating a BGP based Internet presence to a LISP site. It includes a pre-install/pre-turn-up checklist, and customer and provider activation procedures.

A.1. Customer Pre-Install and Pre-Turn-up Checklist

1. Determine how many current physical service provider connections the customer has and their existing bandwidth and traffic engineering requirements.

This information will determine the number of routing locators, and the priorities and weights that should be configured on the xTRs.

2. Make sure customer router has LISP capabilities.

- * Check OS version of the CE router. If LISP is an add-on, check if it is installed.

This information can be used to determine if the platform is appropriate to support LISP, in order to determine if a software and/or hardware upgrade is required.

- * Have customer upgrade (if necessary, software and/or hardware) to be LISP capable.

3. Obtain current running configuration of CE router. A suggested LISP router configuration example can be customized to the customer's existing environment.

4. Verify MTU Handling

- * Request increase in MTU to 1556 or more on service provider connections. Prior to MTU change verify that 1500 byte packet from P-xTR to RLOC with do not fragment (DF-bit) bit set.
- * Ensure they are not filtering ICMP unreachable or time-exceeded on their firewall or router.

LISP, like any tunneling protocol, will increase the size of packets when the LISP header is appended. If increasing the MTU of the access links is not possible, care must be taken that ICMP is not being filtered in order to allow for Path MTU Discovery to take place.

5. Validate member prefix allocation.

This step is to check if the prefix used by the customer is a direct (Provider Independent), or if it is a prefix assigned by a physical service provider (Provider Aggregatable). If the prefixes are assigned by other service providers then a Letter of Agreement is required to announce prefixes through the Proxy Service Provider.

6. Verify the member RLOCs and their reachability.

This step ensures that the RLOCs configured on the CE router are in fact reachable and working.

7. Prepare for cut-over.

- * If possible, have a host outside of all security and filtering policies connected to the console port of the edge router or switch.
- * Make sure customer has access to the router in order to configure it.

A.2. Customer Activating LISP Service

1. Customer configures LISP on CE router(s) from service provider recommended configuration.

The LISP configuration consists of the EID prefix, the locators, and the weights and priorities of the mapping between the two values. In addition, the xTR must be configured with Map Resolver(s), Map Server(s) and the shared key for registering to Map Server(s). If required, Proxy-ETR(s) may be configured as well.

In addition to the LISP configuration, the following:

- * Ensure default route(s) to next-hop external neighbors are included and RLOCs are present in configuration.
 - * If two or more routers are used, ensure all RLOCs are included in the LISP configuration on all routers.
 - * It will be necessary to redistribute default route via IGP between the external routers.
2. When transition is ready perform a soft shutdown on existing eBGP peer session(s)
 - * From CE router, use LIG to ensure registration is successful.
 - * To verify LISP connectivity, find and ping LISP connected sites. If possible, find ping destinations that are not covered by a prefix in the global BGP routing system, because PITRs may deliver the packets even if LISP connectivity is not working. Traceroutes may help discover if this is the case.
 - * To verify connectivity to non-LISP sites, try accessing a landmark (e.g., a major Internet site) via a web browser.

A.3. Cut-Over Provider Preparation and Changes

1. Verify site configuration and then active registration on Map Server(s)
 - * Authentication key
 - * EID prefix
2. Add EID space to map-cache on proxies
3. Add networks to BGP advertisement on proxies
 - * Modify route-maps/policies on P-xTRs
 - * Modify route policies on core routers (if non-connected member)
 - * Modify ingress policers on core routers
 - * Ensure route announcement in looking glass servers, RouteViews
4. Perform traffic verification test
 - * Ensure MTU handling is as expected (PMTUD working)
 - * Ensure proxy-ITR map-cache population
 - * Ensure access from traceroute/ping servers around Internet
 - * Use a looking glass, to check for external visibility of registration via several Map Resolvers

Authors' Addresses

Lorand Jakab
Cisco Systems
170 Tasman Drive
San Jose, CA 95134
USA

Email: lojakab@cisco.com

Albert Cabellos-Aparicio
Technical University of Catalonia
C/Jordi Girona, s/n
BARCELONA 08034
Spain

Email: acabello@ac.upc.edu

Florin Coras
Technical University of Catalonia
C/Jordi Girona, s/n
BARCELONA 08034
Spain

Email: fcoras@ac.upc.edu

Jordi Domingo-Pascual
Technical University of Catalonia
C/Jordi Girona, s/n
BARCELONA 08034
Spain

Email: jordi.domingo@ac.upc.edu

Darrel Lewis
Cisco Systems
170 Tasman Drive
San Jose, CA 95134
USA

Email: darlewis@cisco.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 24 March 2022

A. Cabellos
UPC-BarcelonaTech
D. Saucez (Ed.)
Inria
20 September 2021

An Architectural Introduction to the Locator/ID Separation Protocol
(LISP)
draft-ietf-lisp-introduction-15

Abstract

This document describes the architecture of the Locator/ID Separation Protocol (LISP), making it easier to read the rest of the LISP specifications and providing a basis for discussion about the details of the LISP protocols. This document is used for introductory purposes, more details can be found in [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis], the protocol specifications.

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 <https://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 24 March 2022.

Copyright Notice

Copyright (c) 2021 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 (<https://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	3
2. Definition of Terms	4
3. LISP Architecture	5
3.1. Design Principles	5
3.2. Overview of the Architecture	6
3.3. Data-Plane	8
3.3.1. LISP Encapsulation	9
3.3.2. LISP Forwarding State	10
3.4. Control-Plane	10
3.4.1. LISP Mappings	10
3.4.2. Mapping System Interface	11
3.4.3. Mapping System	12
3.5. Internetworking Mechanisms	14
4. LISP Operational Mechanisms	15
4.1. Cache Management	15
4.2. RLOC Reachability	16
4.3. ETR Synchronization	17
4.4. MTU Handling	17
5. Mobility	18
6. Multicast	19
7. Use Cases	20
7.1. Traffic Engineering	20
7.2. LISP for IPv6 Co-existence	20
7.3. LISP for Virtual Private Networks	21
7.4. LISP for Virtual Machine Mobility in Data Centers	21
8. Security Considerations	21
9. IANA Considerations	23
10. Acknowledgements	23
11. References	23
11.1. Normative References	23
11.2. Informative References	26
Appendix A. A Brief History of Location/Identity Separation	27
A.1. Old LISP Models	28
Authors' Addresses	28

1. Introduction

This document introduces the Locator/ID Separation Protocol (LISP) architecture ([I-D.ietf-lisp-rfc6830bis], [I-D.ietf-lisp-rfc6833bis]), its main operational mechanisms and its design rationale. Fundamentally, LISP is built following a well-known architectural idea: decoupling the IP address overloaded semantics. Indeed and as pointed out by Noel Chiappa [RFC4984], currently IP addresses both identify the topological location of a network attachment point as well as the node's identity. However, nodes and routing have fundamentally different requirements. On the one hand, routing systems require that addresses are aggregatable and have topological meaning, on the other hand, nodes require to be identified independently of their current location [RFC4984].

LISP creates two separate namespaces, EIDs (End-host IDentifiers) and RLOCs (Routing LOCators), both are syntactically identical to the current IPv4 and IPv6 addresses. However, EIDs are used to uniquely identify nodes irrespective of their topological location and are typically routed intra-domain. RLOCs are assigned topologically to network attachment points and are typically routed inter-domain. With LISP, the edge of the Internet (where the nodes are connected) and the core (where inter-domain routing occurs) can be logically separated. LISP-capable routers interconnect the two logical spaces. LISP also introduces a database, called the Mapping System, to store and retrieve mappings between identity and location. LISP-capable routers exchange packets over the Internet core by encapsulating them to the appropriate location.

In summary:

- * RLOCs have meaning only in the underlay network, that is the underlying core routing system.
- * EIDs have meaning only in the overlay network, which is the encapsulation relationship between LISP-capable routers.
- * The LISP edge maps EIDs to RLOCs
- * Within the underlay network, RLOCs have both locator and identifier semantics
- * An EID within a LISP site carries both identifier and locator semantics to other nodes within that site
- * An EID within a LISP site carries identifier and limited locator semantics to nodes at other LISP sites (i.e., enough locator information to tell that the EID is external to the site)

The relationship described above is not unique to LISP and it is common to other overlay technologies.

The initial motivation in the LISP effort is to be found in the routing scalability problem [RFC4984], where, if LISP were to be completely deployed, the Internet core would be populated with RLOCs while Traffic Engineering mechanisms would be pushed to the Mapping System. In such scenario RLOCs are quasi-static (i.e., low churn), hence making the routing system scalable [Quoitin], while EIDs can roam anywhere with no churn to the underlying global routing system. [RFC7215] discusses the impact of LISP on the global routing system during the transition period. However, the separation between location and identity that LISP offers makes it suitable for use in additional scenarios such as Traffic Engineering (TE), multihoming, and mobility among others.

This document describes the LISP architecture and its main operational mechanisms as well as its design rationale. It is important to note that this document does not specify or complement the LISP protocol. The interested reader should refer to the main LISP specifications [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis], as well as the complementary documents [RFC6831], [RFC6832], [I-D.ietf-lisp-6834bis], [RFC6835], [RFC6836], [RFC7052] for the protocol specifications along with the LISP deployment guidelines [RFC7215].

2. Definition of Terms

Endpoint Identifier (EID): EIDs are addresses used to uniquely identify nodes irrespective of their topological location and are typically routed intra-domain.

Routing Locator (RLOC): RLOCs are addresses assigned topologically to network attachment points and typically routed inter-domain.

Ingress Tunnel Router (ITR): A LISP-capable router that encapsulates packets from a LISP site towards the core network.

Egress Tunnel Router (ETR): A LISP-capable router that decapsulates packets from the core of the network towards a LISP site.

xTR: A router that implements both ITR and ETR functionalities.

Map-Request: A LISP signaling message used to request an EID-to-RLOC mapping.

Map-Reply: A LISP signaling message sent in response to a Map-Request that contains a resolved EID-to-RLOC mapping.

Map-Register: A LISP signaling message used to register an EID-to-RLOC mapping.

Map-Notify: A LISP signaling message sent in response of a Map-Register to acknowledge the correct reception of an EID-to-RLOC mapping.

This document describes the LISP architecture and does not introduce any new term. The reader is referred to [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis], [RFC6831], [RFC6832], [I-D.ietf-lisp-6834bis], [RFC6835], [RFC6836], [RFC7052], [RFC7215] for the complete definition of terms.

3. LISP Architecture

This section presents the LISP architecture, it first details the design principles of LISP and then it proceeds to describe its main aspects: data-plane, control-plane, and internetworking mechanisms.

3.1. Design Principles

The LISP architecture is built on top of four basic design principles:

- * **Locator/Identifier split:** By decoupling the overloaded semantics of the current IP addresses the Internet core can be assigned identity meaningful addresses and hence, can use aggregation to scale. Devices are assigned with relatively opaque topologically meaningful addresses that are independent of their topological location.
- * **Overlay architecture:** Overlays route packets over the current Internet, allowing deployment of new protocols without changing the current infrastructure hence, resulting into a low deployment cost.
- * **Decoupled data-plane and control-plane:** Separating the data-plane from the control-plane allows them to scale independently and use different architectural approaches. This is important given that they typically have different requirements and allows for other data-planes to be added. Even though the data-plane and the control-plane are decoupled, they are not completely isolated because the LISP data-plane may trigger control-plane activity.
- * **Incremental deployability:** This principle ensures that the protocol interoperates with the legacy Internet while providing some of the targeted benefits to early adopters.

3.2. Overview of the Architecture

LISP splits architecturally the core from the edge of the Internet by creating two separate namespaces: Endpoint Identifiers (EIDs) and Routing LOCators (RLOCs). The edge consists of LISP sites (e.g., an Autonomous System) that use EID addresses. EIDs are IPv4 or IPv6 addresses that uniquely identify communication end-hosts and are assigned and configured by the same mechanisms that exist at the time of this writing. EIDs do not contain inter-domain topological information and because of this, EIDs are usually routable at the edge (within LISP sites) but not in the core; see Section 3.5 for discussion of LISP site internetworking with non-LISP sites and domains in the Internet.

LISP sites (at the edge) are connected to the interconnecting core by means of LISP-capable routers (e.g., border routers). LISP sites are connected across the interconnecting core using tunnels between the LISP-capable routers. When packets originated from a LISP site are flowing towards the core network, they ingress into an encapsulated tunnel via an Ingress Tunnel Router (ITR). When packets flow from the core network to a LISP site, they egress from an encapsulated tunnel to an Egress Tunnel Router (ETR). An xTR is a router which can perform both ITR and ETR operations. In this context ITRs encapsulate packets while ETRs decapsulate them, hence LISP operates as an overlay on top of the current Internet core.

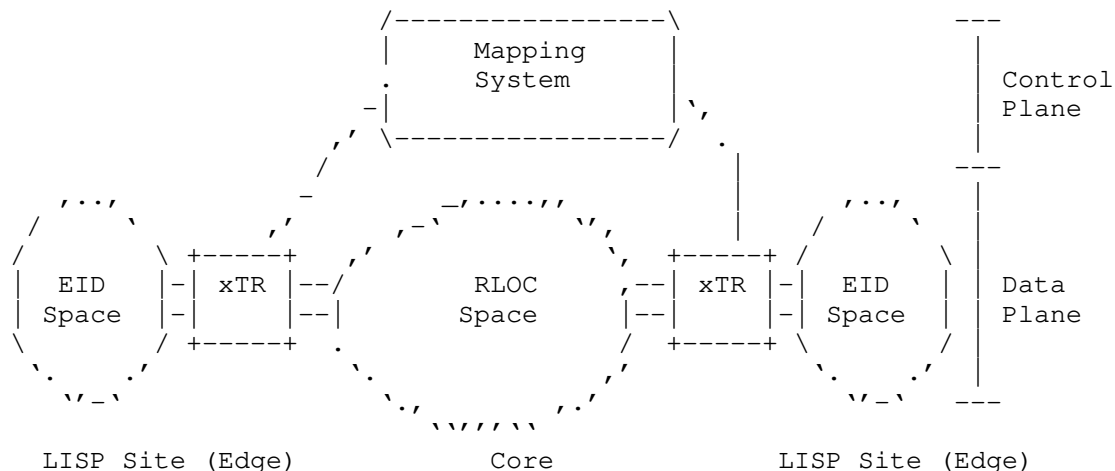


Figure 1: A schema of the LISP Architecture.

With LISP, the core uses RLOCs, an RLOC is an IPv4 or IPv6 address assigned to an core-facing network interface of an ITR or ETR.

A database which is typically distributed, called the Mapping System, stores mappings between EIDs and RLOCs. Such mappings relate the identity of the devices attached to LISP sites (EIDs) to the set of RLOCs configured at the LISP-capable routers servicing the site. Furthermore, the mappings also include traffic engineering policies and can be configured to achieve multihoming and load balancing. The LISP Mapping System is conceptually similar to the DNS where it is organized as a distributed multi-organization network database. With LISP, ETRs register mappings while ITRs retrieve them.

Finally, the LISP architecture emphasizes incremental deployment. Given that LISP represents an overlay to the current Internet architecture, end hosts as well as intra and inter-domain routers remain unchanged, and the only required changes to the existing infrastructure are to routers connecting the EID space with the RLOC space. Additionally, LISP requires the deployment of an independent Mapping System, such distributed database is a new network entity.

The following describes a simplified packet flow sequence between two nodes that are attached to LISP sites. Please note that typical LISP-capable routers are xTRs (both ITR and ETR). Client HostA wants to send a packet to server HostB.

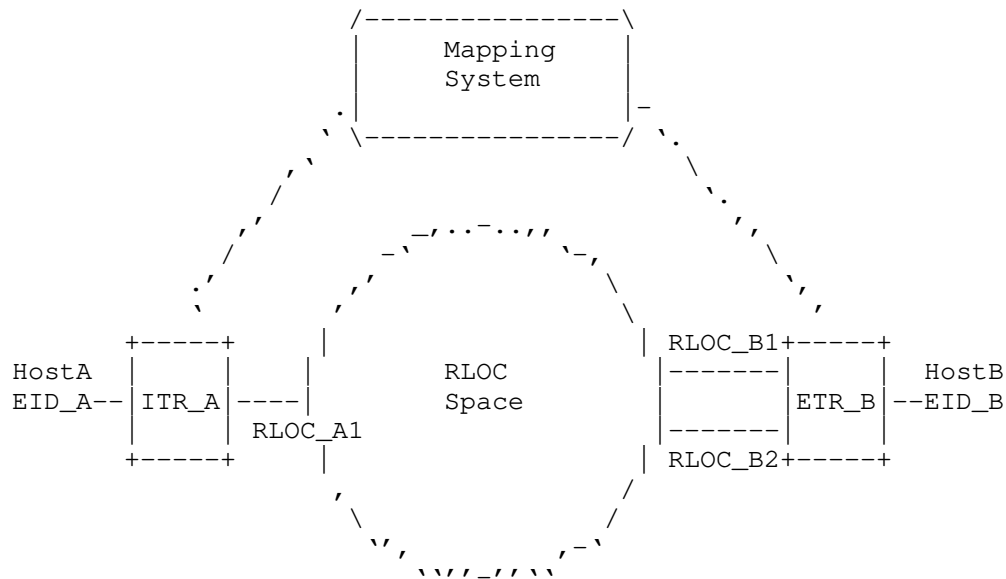


Figure 2: Packet flow sequence in LISP.

1. HostA retrieves the EID_B of HostB, typically querying the DNS and obtaining an A or AAAA record. Then it generates an IP packet as in the Internet, the packet has source address EID_A and destination address EID_B.
2. The packet is forwarded towards ITR_A in the LISP site using standard intra-domain mechanisms.
3. ITR_A upon receiving the packet queries the Mapping System to retrieve the locator of ETR_B that is servicing HostB's EID_B. In order to do so it uses a LISP control message called Map-Request, the message contains EID_B as the lookup key. In turn it receives another LISP control message called Map-Reply, the message contains two locators: RLOC_B1 and RLOC_B2 along with traffic engineering policies: priority and weight per locator. Note that a Map-Reply can contain more locators if needed. ITR_A can cache the mapping in a local storage to speed-up forwarding of subsequent packets.
4. ITR_A encapsulates the packet towards RLOC_B1 (chosen according to the priorities/weights specified in the mapping). The packet contains two IP headers, the outer header has RLOC_A1 as source and RLOC_B1 as destination, the inner original header has EID_A as source and EID_B as destination. Furthermore ITR_A adds a LISP header, more details about LISP encapsulation can be found in Section 3.3.1.
5. The encapsulated packet is forwarded over the interconnecting core as a normal IP packet, making the EID invisible from the core.
6. Upon reception of the encapsulated packet by ETR_B, it decapsulates the packet and forwards it to HostB.

3.3. Data-Plane

This section provides a high-level description of the LISP data-plane, which is specified in detail in [I-D.ietf-lisp-rfc6830bis]. The LISP data-plane is responsible for encapsulating and decapsulating data packets and caching the appropriate forwarding state. It includes two main entities, the ITR and the ETR, both are LISP capable routers that connect the EID with the RLOC space (ITR) and vice versa (ETR).

3.3.1. LISP Encapsulation

ITRs encapsulate data packets towards ETRs. LISP data packets are encapsulated using UDP (port 4341), the source port is usually selected by the ITR using a 5-tuple hash of the inner header (so to be consistent in case of multi-path solutions such as ECMP [RFC2992]) and ignored on reception. LISP data packets are often encapsulated in UDP packets that include a zero checksum [RFC6935] [RFC6936] that may not be verified when it is received, because LISP data packets typically include an inner transport protocol header with a non-zero checksum. The use of UDP zero checksums over IPv6 for all tunneling protocols like LISP is subject to the applicability statement in [RFC6936]. If LISP data packets are encapsulated in UDP packets with non-zero checksums, the outer UDP checksums are verified when the UDP packets are received, as part of normal UDP processing.

LISP-encapsulated packets also include a LISP header (after the UDP header and before the original IP header). The LISP header is prepended by ITRs and stripped by ETRs. It carries reachability information (see more details in Section 4.2) and the Instance ID field. The Instance ID field is used to distinguish traffic to/from different tenant address spaces at the LISP site and that may use overlapped but logically separated EID addressing.

Overall, LISP works on 4 headers, the inner header the source constructed, and the 3 headers a LISP encapsulator prepends ("outer" to "inner"):

1. Outer IP header containing RLOCs as source and destination addresses. This header is originated by ITRs and stripped by ETRs.
2. UDP header (port 4341), usually with zero checksum. This header is originated by ITRs and stripped by ETRs.
3. LISP header that contains various forwarding-plane features (such as reachability) and an Instance ID field. This header is originated by ITRs and stripped by ETRs.
4. Inner IP header containing EIDs as source and destination addresses. This header is created by the source end-host and is left unchanged by LISP data plane processing on the ITR and ETR.

Finally, in some scenarios Re-encapsulating and/or Recursive tunnels are useful to choose a specified path in the underlay network, for instance to avoid congestion or failure. Re-encapsulating tunnels are consecutive LISP tunnels and occur when a decapsulator (an ETR action) removes a LISP header and then acts as an encapsulator (an ITR

action) to prepend another one. On the other hand, Recursive tunnels are nested tunnels and are implemented by using multiple LISP encapsulations on a packet. Such functions are implemented by Reencapsulating Tunnel Routers (RTRs). An RTR can be thought of as a router that first acts as an ETR by decapsulating packets and then as an ITR by encapsulating them towards another locator, more information can be found at [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis].

3.3.2. LISP Forwarding State

In the LISP architecture, ITRs keep just enough information to route traffic flowing through them. Meaning that, ITRs retrieve from the LISP Mapping System mappings between EID-prefixes (blocks of EIDs) and RLOCs that are used to encapsulate packets. Such mappings are stored in a local cache called the LISP Map-Cache for subsequent packets addressed to the same EID prefix. Note that, in case of overlapping EID-prefixes, following a single request, the ITR may receive a set of mappings, covering the requested EID-prefix and all more-specifics (cf., Section 5.5 [I-D.ietf-lisp-rfc6833bis]). Mappings include a (Time-to-Live) TTL (set by the ETR). More details about the Map-Cache management can be found in Section 4.1.

3.4. Control-Plane

The LISP control-plane, specified in [I-D.ietf-lisp-rfc6833bis], provides a standard interface to register and request mappings. The LISP Mapping System is a database that stores such mappings. The following first describes the mappings, then the standard interface to the Mapping System, and finally its architecture.

3.4.1. LISP Mappings

Each mapping includes the bindings between EID prefix(es) and set of RLOCs as well as traffic engineering policies, in the form of priorities and weights for the RLOCs. Priorities allow the ETR to configure active/backup policies while weights are used to load-balance traffic among the RLOCs (on a per-flow basis).

Typical mappings in LISP bind EIDs in the form of IP prefixes with a set of RLOCs, also in the form of IP addresses. IPv4 and IPv6 addresses are encoded using the appropriate Address Family Identifier (AFI) [RFC3232]. However LISP can also support more general address encoding by means of the ongoing effort around the LISP Canonical Address Format (LCAF) [RFC8060].

With such a general syntax for address encoding in place, LISP aims to provide flexibility to current and future applications. For instance LCAFs could support MAC addresses, geo-coordinates, ASCII names and application specific data.

3.4.2. Mapping System Interface

LISP defines a standard interface between data and control planes. The interface is specified in [I-D.ietf-lisp-rfc6833bis] and defines two entities:

Map-Server: A network infrastructure component that learns mappings from ETRs and publishes them into the LISP Mapping System. Typically Map-Servers are not authoritative to reply to queries and hence, they forward them to the ETR. However, they can also operate in proxy-mode, where the ETRs delegate replying to queries to Map-Servers. This setup is useful when the ETR has limited resources (e.g., CPU or power).

Map-Resolver: A network infrastructure component that interfaces ITRs with the Mapping System by proxying queries and in some cases responses.

The interface defines four LISP control messages which are sent as UDP datagrams (port 4342):

Map-Register: This message is used by ETRs to register mappings in the Mapping System and it is authenticated using a shared key between the ETR and the Map-Server.

Map-Notify: When requested by the ETR, this message is sent by the Map-Server in response to a Map-Register to acknowledge the correct reception of the mapping and convey the latest Map-Server state on the EID to RLOC mapping. In some cases a Map-Notify can be sent to the previous RLOCs when an EID is registered by a new set of RLOCs.

Map-Request: This message is used by ITRs or Map-Resolvers to resolve the mapping of a given EID.

Map-Reply: This message is sent by Map-Servers or ETRs in response to a Map-Request and contains the resolved mapping. Please note that a Map-Reply may contain a negative reply if, for example, the queried EID is not part of the LISP EID space. In such cases the ITR typically forwards the traffic natively (non encapsulated) to the public Internet, this behavior is defined to support incremental deployment of LISP.

3.4.3. Mapping System

LISP architecturally decouples control and data-plane by means of a standard interface. This interface glues the data-plane - routers responsible for forwarding data-packets - with the LISP Mapping System - a database responsible for storing mappings.

With this separation in place, the data and control-plane can use different architectures if needed and scale independently. Typically the data-plane is optimized to route packets according to hierarchical IP addresses. However the control-plane may have different requirements, for instance and by taking advantage of the LCAF, the Mapping System may be used to store non-hierarchical keys (such as MAC addresses), requiring different architectural approaches for scalability. Another important difference between the LISP control- and data- planes is that, as a result of the local mapping cache available at ITR, the Mapping System does not need to operate at line-rate.

Many of the existing mechanisms to create distributed systems have been explored and considered for the Mapping System architecture: graph-based databases in the form of LISP+ALT [RFC6836], hierarchical databases in the form of LISP-DDT [RFC8111], monolithic databases in the form of LISP-NERD [RFC6837], flat databases in the form of LISP-DHT [I-D.cheng-lisp-shdht], [Mathy], and a multicast-based database [I-D.curran-lisp-emacs]. Furthermore it is worth noting that, in some scenarios such as private deployments, the Mapping System can operate as logically centralized. In such cases it is typically composed of a single Map-Server/Map-Resolver.

The following focuses on the two mapping systems that have been implemented and deployed (LISP+ALT and LISP-DDT).

3.4.3.1. LISP+ALT

The LISP Alternative Topology (LISP+ALT) [RFC6836] was the first Mapping System proposed, developed and deployed on the LISP pilot network. It is based on a distributed BGP overlay participated by Map-Servers and Map-Resolvers. The nodes connect to their peers through static tunnels. Each Map-Server involved in the ALT topology advertises the EID-prefixes registered by the serviced ETRs, making the EID routable on the ALT topology.

When an ITR needs a mapping it sends a Map-Request to a Map-Resolver that, using the ALT topology, forwards the Map-Request towards the Map-Server responsible for the mapping. Upon reception the Map-Server forwards the request to the ETR that in turn, replies directly to the ITR.

3.4.3.2. LISP-DDT

LISP-DDT [RFC8111] is conceptually similar to the DNS, a hierarchical directory whose internal structure mirrors the hierarchical nature of the EID address space. The DDT hierarchy is composed of DDT nodes forming a tree structure, the leafs of the tree are Map-Servers. On top of the structure there is the DDT root node, which is a particular instance of a DDT node and that matches the entire address space. As in the case of DNS, DDT supports multiple redundant DDT nodes and/or DDT roots. Finally, Map-Resolvers are the clients of the DDT hierarchy and can query either the DDT root and/or other DDT nodes.

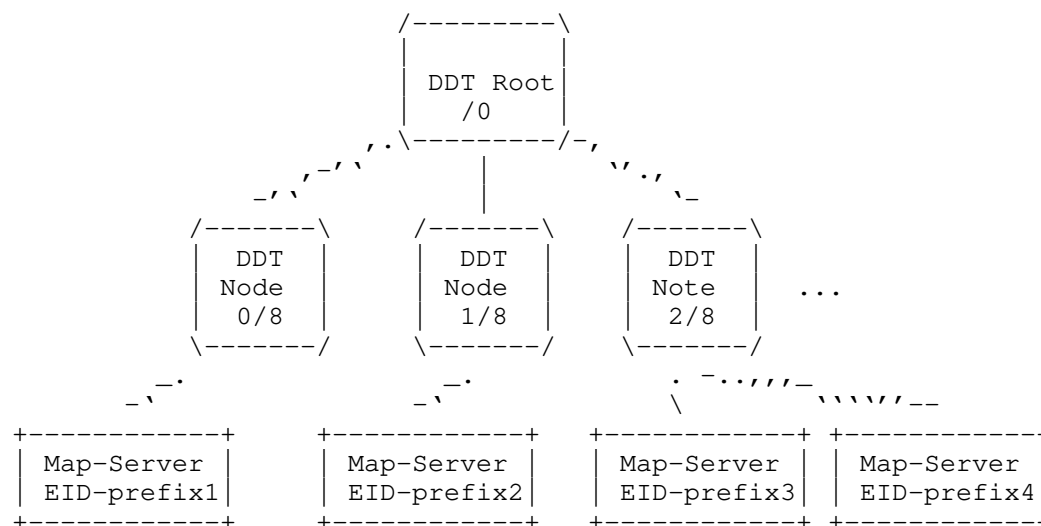


Figure 3: A schematic representation of the DDT tree structure, please note that the prefixes and the structure depicted should be only considered as an example.

The DDT structure does not actually index EID-prefixes but eXtended EID-prefixes (XEID). An XEID-prefix is just the concatenation of the following fields (from most significant bit to less significant bit): Database-ID, Instance ID, Address Family Identifier and the actual EID-prefix. The Database-ID is provided for possible future requirements of higher levels in the hierarchy and to enable the creation of multiple and separate database trees.

In order to resolve a query LISP-DDT operates in a similar way to the DNS but only supports iterative lookups. DDT clients (usually Map-Resolvers) generate Map-Requests to the DDT root node. In response

they receive a newly introduced LISP-control message: a Map-Referral. A Map-Referral provides the list of RLOCs of the set of DDT nodes matching a configured XEID delegation. That is, the information contained in the Map-Referral points to the child of the queried DDT node that has more specific information about the queried XEID-prefix. This process is repeated until the DDT client walks the tree structure (downwards) and discovers the Map-Server servicing the queried XEID. At this point the client sends a Map-Request and receives a Map-Reply containing the mappings. It is important to note that DDT clients can also cache the information contained in Map-Referrals, that is, they cache the DDT structure. This is used to reduce the mapping retrieving latency [Jakab].

The DDT Mapping System relies on manual configuration. That is Map-Resolvers are configured with the set of available DDT root nodes while DDT nodes are configured with the appropriate XEID delegations. Configuration changes in the DDT nodes are only required when the tree structure changes itself, but it doesn't depend on EID dynamics (RLOC allocation or traffic engineering policy changes).

3.5. Internetworking Mechanisms

EIDs are typically identical to either IPv4 or IPv6 addresses and they are stored in the LISP Mapping System, however they are usually not announced in the routing system beyond the local LISP domain. As a result LISP requires an internetworking mechanism to allow LISP sites to speak with non-LISP sites and vice versa. LISP internetworking mechanisms are specified in [RFC6832].

LISP defines two entities to provide internetworking:

Proxy Ingress Tunnel Router (PITR): PITRs provide connectivity from the legacy Internet to LISP sites. PITRs announce in the global routing system blocks of EID prefixes (aggregating when possible) to attract traffic. For each incoming packet from a source not in a LISP site (a non-EID), the PITR LISP-encapsulates it towards the RLOC(s) of the appropriate LISP site. The impact of PITRs in the routing table size of the Default-Free Zone (DFZ) is, in the worst-case, similar to the case in which LISP is not deployed. EID-prefixes will be aggregated as much as possible both by the PITR and by the global routing system.

Proxy Egress Tunnel Router (PETR): PETRs provide connectivity from LISP sites to the legacy Internet. In some scenarios, LISP sites may be unable to send encapsulated packets with a local EID address as a source to the legacy Internet. For instance when Unicast Reverse Path Forwarding (uRPF) is used by Provider Edge routers, or when an intermediate network between a LISP site and a

non-LISP site does not support the desired version of IP (IPv4 or IPv6). In both cases the PETR overcomes such limitations by encapsulating packets over the network. There is no specified provision for the distribution of PETR RLOC addresses to the ITRs.

Additionally, LISP also defines mechanisms to operate with private EIDs [RFC1918] by means of LISP-NAT [RFC6832]. In this case the xTR replaces a private EID source address with a routable one. At the time of this writing, work is ongoing to define NAT-traversal capabilities, that is xTRs behind a NAT using non-routable RLOCs.

PITRs, PETRs and, LISP-NAT enable incremental deployment of LISP, by providing significant flexibility in the placement of the boundaries between the LISP and non-LISP portions of the network, and making it easy to change those boundaries over time.

4. LISP Operational Mechanisms

This section details the main operational mechanisms defined in LISP.

4.1. Cache Management

LISP's decoupled control and data-plane, where mappings are stored in the control-plane and used for forwarding in the data-plane, requires a local cache in ITRs to reduce signaling overhead (Map-Request/Map-Reply) and increase forwarding speed. The local cache available at the ITRs, called Map-Cache, is used by the router to LISP-encapsulate packets. The Map-Cache is indexed by (Instance ID, EID-prefix) and contains basically the set of RLOCs with the associated traffic engineering policies (priorities and weights).

The Map-Cache, as any other cache, requires cache coherence mechanisms to maintain up-to-date information. LISP defines three main mechanisms for cache coherence:

Record Time-To-Live (TTL): Each mapping record contains a TTL set by the ETR, upon expiration of the TTL the ITR can't use the mapping until it is refreshed by sending a new Map-Request.

Solicit-Map-Request (SMR): SMR is an explicit mechanism to update mapping information. In particular a special type of Map-Request can be sent on demand by ETRs to request refreshing a mapping. Upon reception of a SMR message, the ITR must refresh the bindings by sending a Map-Request to the Mapping System. Further uses of SMRs are documented in [I-D.ietf-lisp-rfc6833bis].

Map-Versioning: This optional mechanism piggybacks in the LISP

header of data-packets the version number of the mappings used by an xTR. This way, when an xTR receives a LISP-encapsulated packet from a remote xTR, it can check whether its own Map-Cache or the one of the remote xTR is outdated. If its Map-Cache is outdated, it sends a Map-Request for the remote EID so to obtain the newest mappings. On the contrary, if it detects that the remote xTR Map-Cache is outdated, it sends a SMR to notify it that a new mapping is available. Further details are available in [I-D.ietf-lisp-6834bis].

Finally it is worth noting that in some cases an entry in the map-cache can be proactively refreshed using the mechanisms described in the section below.

4.2. RLOC Reachability

In most cases LISP operates with a pull-based Mapping System (e.g., DDT), this results in an edge to edge pull architecture. In such scenario the network state is stored in the control-plane while the data-plane pulls it on demand. This has consequences concerning the propagation of xTRs reachability/liveness information since pull architectures require explicit mechanisms to propagate this information. As a result LISP defines a set of mechanisms to inform ITRs and PITRs about the reachability of the cached RLOCs:

Locator Status Bits (LSB): LSB is a passive technique, the LSB field is carried by data-packets in the LISP header and can be set by a ETRs to specify which RLOCs of the ETR site are up/down. This information can be used by the ITRs as a hint about the reachability to perform additional checks. Also note that LSB does not provide path reachability status, only hints on the status of RLOCs as such they must not be used over the public Internet and should be coupled with Map-Versioning to prevent race conditions where LSB are interpreted as referring to different RLOCs than intended.

Echo-nonce: This is also a passive technique, that can only operate effectively when data flows bi-directionally between two communicating xTRs. Basically, an ITR piggybacks a random number (called nonce) in LISP data packets, if the path and the probed locator are up, the ETR will piggyback the same random number on the next data-packet, if this is not the case the ITR can set the locator as unreachable. When traffic flow is unidirectional or when the ETR receiving the traffic is not the same as the ITR that transmits it back, additional mechanisms are required. The echo-nonce mechanism must be used in trusted environments only, not over the public Internet.

RLOC-probing: This is an active probing algorithm where ITRs send probes to specific locators, this effectively probes both the locator and the path. In particular this is done by sending a Map-Request (with certain flags activated) on the data-plane (RLOC space) and waiting in return a Map-Reply, also sent on the data-plane. The active nature of RLOC-probing provides an effective mechanism to determine reachability and, in case of failure, switching to a different locator. Furthermore the mechanism also provides useful RTT estimates of the delay of the path that can be used by other network algorithms.

It is worth noting that RLOC probing and Echo-nonce can work together. Specifically if a nonce is not echoed, an ITR could RLOC-probe to determine if the path is up when it cannot tell the difference between a failed bidirectional path or the return path is not used (a unidirectional path).

Additionally, LISP also recommends inferring reachability of locators by using information provided by the underlay, in particular:

ICMP signaling: The LISP underlay -the current Internet- uses the ICMP protocol to signal unreachability (among other things). LISP can take advantage of this and the reception of a ICMP Network Unreachable or ICMP Host Unreachable message can be seen as a hint that a locator might be unreachable, this should lead to perform additional checks.

Underlay routing: Both BGP and IGP carry reachability information, LISP-capable routers that have access to underlay routing information can use it to determine if a given locator or path are reachable.

4.3. ETR Synchronization

All the ETRs that are authoritative to a particular EID-prefix must announce the same mapping to the requesters, this means that ETRs must be aware of the status of the RLOCs of the remaining ETRs. This is known as ETR synchronization.

At the time of this writing LISP does not specify a mechanism to achieve ETR synchronization. Although many well-known techniques could be applied to solve this issue it is still under research, as a result operators must rely on coherent manual configuration

4.4. MTU Handling

Since LISP encapsulates packets it requires dealing with packets that exceed the MTU of the path between the ITR and the ETR. Specifically LISP defines two mechanisms:

Stateless: With this mechanism the effective MTU is assumed from the ITR's perspective. If a payload packet is too big for the effective MTU, and can be fragmented, the payload packet is fragmented on the ITR, such that reassembly is performed at the destination host.

Stateful: With this mechanism ITRs keep track of the MTU of the paths towards the destination locators by parsing the ICMP Too Big packets sent by intermediate routers. ITRs will send ICMP Too Big messages to inform the sources about the effective MTU. Additionally ITRs can use mechanisms such as PMTUD [RFC1191] or PLPMTUD [RFC4821] to keep track of the MTU towards the locators.

In both cases if the packet cannot be fragmented (IPv4 with DF=1 or IPv6) then the ITR drops it and replies with a ICMP Too Big message to the source.

5. Mobility

The separation between locators and identifiers in LISP is suitable for traffic engineering purpose where LISP sites can change their attachment points to the Internet (i.e., RLOCs) without impacting endpoints or the Internet core. In this context, the border routers operate the xTR functionality and endpoints are not aware of the existence of LISP. This functionality is similar to Network Mobility [RFC3963]. However, this mode of operation does not allow seamless mobility of endpoints between different LISP sites as the EID address might not be routable in a visited site. Nevertheless, LISP can be used to enable seamless IP mobility when LISP is directly implemented in the endpoint or when the endpoint roams to an attached xTR. Each endpoint is then an xTR and the EID address is the one presented to the network stack used by applications while the RLOC is the address gathered from the network when it is visited. This functionality is similar to Mobile IP ([RFC5944] and [RFC6275]).

Whenever the device changes of RLOC, the xTR updates the RLOC of its local mapping and registers it to its Map-Server, typically with a low TTL value (1min). To avoid the need of a home gateway, the ITR also indicates the RLOC change to all remote devices that have ongoing communications with the device that moved. The combination of both methods ensures the scalability of the system as signaling is strictly limited the Map-Server and to hosts with which communications are ongoing. In the mobility case the EID-prefix can be as small as a full /32 or /128 (IPv4 or IPv6 respectively) depending on the specific use-case (e.g., subnet mobility vs single VM/Mobile node mobility).

The decoupled identity and location provided by LISP allows it to operate with other layer 2 and layer 3 mobility solutions.

6. Multicast

LISP also supports transporting IP multicast packets sent from the EID space, the operational changes required to the multicast protocols are documented in [RFC6831].

In such scenarios, LISP may create multicast state both at the core and at the sites (both source and receiver). When signaling is used to create multicast state at the sites, LISP routers unicast encapsulate PIM Join/Prune messages from receiver to source sites. At the core, ETRs build a new PIM Join/Prune message addressed to the RLOC of the ITR servicing the source. An simplified sequence is shown below

1. An end-host willing to join a multicast channel sends an IGMP report. Multicast PIM routers at the LISP site propagate PIM Join/Prune messages (S-EID, G) towards the ETR.
2. The join message flows to the ETR, upon reception the ETR builds two join messages, the first one unicast LISP-encapsulates the original join message towards the RLOC of the ITR servicing the source. This message creates (S-EID, G) multicast state at the source site. The second join message contains as destination address the RLOC of the ITR servicing the source (S-RLOC, G) and creates multicast state at the core.
3. Multicast data packets originated by the source (S-EID, G) flow from the source to the ITR. The ITR LISP-encapsulates the multicast packets, the outer header includes its own RLOC as the source (S-RLOC) and the original multicast group address (G) as the destination. Please note that multicast group address are logical and are not resolved by the mapping system. Then the multicast packet is transmitted through the core towards the receiving ETRs that decapsulates the packets and sends them using the receiver's site multicast state.

Please note that the inner and outer multicast addresses are in general different, unless in specific cases where the underlay provider implements a tight control on the overlay. LISP specifications already support all PIM modes [RFC6831]. Additionally, LISP can support as well non-PIM mechanisms in order to maintain multicast state.

When multicast sources and receivers are active at LISP sites, and the core network between the sites does not provide multicast support, a signal-free mechanism can be used to create an overlay that will allow multicast traffic to flow between sites and connect the multicast trees at the different sites [RFC8378]. Registrations from the different receiver sites will be merged at the mapping system to assemble a multicast-replication-list inclusive of all Routing Locators (RLOCs) that lead to receivers for a particular multicast group or multicast channel. The replication list for each specific multicast entry is maintained as a database mapping entry in the LISP mapping system.

7. Use Cases

7.1. Traffic Engineering

A LISP site can strictly impose via which ETRs the traffic must enter the LISP site network even though the path followed to reach the ETR is not under the control of the LISP site. This fine control is implemented with the mappings. When a remote site is willing to send traffic to a LISP site, it retrieves the mapping associated to the destination EID via the mapping system. The mapping is sent directly by an authoritative ETR of the EID and is not altered by any intermediate network.

A mapping associates a list of RLOCs to an EID prefix. Each RLOC corresponds to an interface of an ETR (or set of ETRs) that is able to correctly forward packets to EIDs in the prefix. Each RLOC is tagged with a priority and a weight in the mapping. The priority is used to indicate which RLOCs should be preferred to send packets (the least preferred ones being provided for backup purpose). The weight permits to balance the load between the RLOCs with the same priority, proportionally to the weight value.

As mappings are directly issued by the authoritative ETR of the EID and are not altered while transmitted to the remote site, it offers highly flexible incoming inter-domain traffic engineering with even the possibility for a site to support a different mapping policy for each remote site.

7.2. LISP for IPv6 Co-existence

LISP encapsulations allow to transport packets using EIDs from a given address family (e.g., IPv6) with packets from other address families (e.g., IPv4). The absence of correlation between the address family of RLOCs and EIDs makes LISP a candidate to allow, e.g., IPv6 to be deployed when all of the core network may not have IPv6 enabled.

For example, two IPv6-only data centers could be interconnected via the legacy IPv4 Internet. If their border routers are LISP capable, sending packets between the data center is done without any form of translation as the native IPv6 packets (in the EID space) will be LISP encapsulated and transmitted over the IPv4 legacy Internet by the mean of IPv4 RLOCs.

7.3. LISP for Virtual Private Networks

It is common to operate several virtual networks over the same physical infrastructure. In such virtual private networks, it is essential to distinguish which virtual network a packet belongs and tags or labels are used for that purpose. When using LISP, the distinction can be made with the Instance ID field. When an ITR encapsulates a packet from a particular virtual network (e.g., known via the VRF or VLAN), it tags the encapsulated packet with the Instance ID corresponding to the virtual network of the packet. When an ETR receives a packet tagged with an Instance ID it uses the Instance ID to determine how to treat the packet.

The main usage of LISP for virtual private networks does not introduce additional requirements on the underlying network, as long as it runs IP.

7.4. LISP for Virtual Machine Mobility in Data Centers

A way to enable seamless virtual machine mobility in data center is to conceive the datacenter backbone as the RLOC space and the subnet where servers are hosted as forming the EID space. A LISP router is placed at the border between the backbone and each subnet. When a virtual machine is moved to another subnet, it can keep (temporarily) the address it had before the move so to continue without a transport layer connection reset. When an xTR detects a source address received on a subnet to be an address not assigned to the subnet, it registers the address to the Mapping System.

To inform the other LISP routers that the machine moved and where, and then to avoid detours via the initial subnetwork, mechanisms such as the Solicit-Map-Request messages are used.

8. Security Considerations

This section describes the security considerations associated to the LISP protocol.

While in a push mapping system, the state necessary to forward packets is learned independently of the traffic itself, with a pull architecture, the system becomes reactive and data-plane events

(e.g., the arrival of a packet for an unknown destination) may trigger control-plane events. This on-demand learning of mappings provides many advantages as discussed above but may also affect the way security is enforced.

Usually, the data-plane is implemented in the fast path of routers to provide high performance forwarding capabilities while the control-plane features are implemented in the slow path to offer high flexibility and a performance gap of several order of magnitude can be observed between the slow and the fast paths. As a consequence, the way data-plane events are notified to the control-plane must be thought carefully so to not overload the slow path and rate limiting should be used as specified in [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis].

Care must also be taken so to not overload the mapping system (i.e., the control plane infrastructure) as the operations to be performed by the mapping system may be more complex than those on the data-plane, for that reason [I-D.ietf-lisp-rfc6830bis] and [I-D.ietf-lisp-rfc6833bis] recommends to rate limit the sending of messages to the mapping system.

To improve resiliency and reduce the overall number of messages exchanged, LISP offers the possibility to leak information, such as reachability of locators, directly into data plane packets. In environments that are not fully trusted, like the open Internet, control information gleaned from data-plane packets must not be used or must be verified before using it.

Mappings are the centrepiece of LISP and all precautions must be taken to avoid them to be manipulated or misused by malicious entities. Using trustable Map-Servers that strictly respect [I-D.ietf-lisp-rfc6833bis] and the authentication mechanism proposed by LISP-Sec [I-D.ietf-lisp-sec] reduces the risk of attacks to the mapping integrity. In more critical environments, secure measures may be needed. The way security is implemented for a given mapping system strongly depends on the architecture of the mapping system itself and the threat model assumed for the deployment. Thus, the mapping system security has to be discussed in the relevant documents proposing the mapping system architecture.

As with any other tunneling mechanism, middleboxes on the path between an ITR (or PITR) and an ETR (or PETR) must implement mechanisms to strip the LISP encapsulation to correctly inspect the content of LISP encapsulated packets.

Like other map-and-encap mechanisms, LISP enables triangular routing (i.e., packets of a flow cross different border routers depending on their direction). This means that intermediate boxes may have incomplete view on the traffic they inspect or manipulate. Moreover, LISP-encapsulated packets are routed based on the outer IP address (i.e., the RLOC), and can be delivered to an ETR that is not responsible of the destination EID of the packet or even to a network element that is not an ETR. The mitigation consists in applying appropriate filtering techniques on the network elements that can potentially receive un-expected LISP-encapsulated packets

More details about security implications of LISP are discussed in [RFC7835].

9. IANA Considerations

This memo includes no requests to IANA.

10. Acknowledgements

This document was initiated by Noel Chiappa and much of the core philosophy came from him. The authors acknowledge the important contributions he has made to this work and thank him for his past efforts.

The authors would also like to thank Dino Farinacci, Fabio Maino, Luigi Iannone, Sharon Barkai, Isidoros Kouvelas, Christian Cassar, Florin Coras, Marc Binderberger, Alberto Rodriguez-Natal, Ronald Bonica, Chad Hintz, Robert Raszuk, Joel M. Halpern, Darrel Lewis, David Black.

11. References

11.1. Normative References

[I-D.ietf-lisp-6834bis]
Iannone, L., Saucez, D., and O. Bonaventure, "Locator/ID Separation Protocol (LISP) Map-Versioning", Work in Progress, Internet-Draft, draft-ietf-lisp-6834bis-09, 31 August 2021, <<https://www.ietf.org/archive/id/draft-ietf-lisp-6834bis-09.txt>>.

- [I-D.ietf-lisp-rfc6830bis]
Farinacci, D., Fuller, V., Meyer, D., Lewis, D., and A. Cabellos-Aparicio, "The Locator/ID Separation Protocol (LISP)", Work in Progress, Internet-Draft, draft-ietf-lisp-rfc6830bis-36, 18 November 2020, <<https://www.ietf.org/internet-drafts/draft-ietf-lisp-rfc6830bis-36.txt>>.
- [I-D.ietf-lisp-rfc6833bis]
Farinacci, D., Maino, F., Fuller, V., and A. Cabellos-Aparicio, "Locator/ID Separation Protocol (LISP) Control-Plane", Work in Progress, Internet-Draft, draft-ietf-lisp-rfc6833bis-30, 18 November 2020, <<https://www.ietf.org/internet-drafts/draft-ietf-lisp-rfc6833bis-30.txt>>.
- [I-D.ietf-lisp-sec]
Maino, F., Ermagan, V., Cabellos-Aparicio, A., and D. Saucez, "LISP-Security (LISP-SEC)", Work in Progress, Internet-Draft, draft-ietf-lisp-sec-22, 12 January 2021, <<https://www.ietf.org/internet-drafts/draft-ietf-lisp-sec-22.txt>>.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <<https://www.rfc-editor.org/info/rfc1191>>.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. J., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<https://www.rfc-editor.org/info/rfc1918>>.
- [RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path Algorithm", RFC 2992, DOI 10.17487/RFC2992, November 2000, <<https://www.rfc-editor.org/info/rfc2992>>.
- [RFC3232] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, DOI 10.17487/RFC3232, January 2002, <<https://www.rfc-editor.org/info/rfc3232>>.
- [RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <<https://www.rfc-editor.org/info/rfc3963>>.
- [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <<https://www.rfc-editor.org/info/rfc4821>>.

- [RFC4984] Meyer, D., Ed., Zhang, L., Ed., and K. Fall, Ed., "Report from the IAB Workshop on Routing and Addressing", RFC 4984, DOI 10.17487/RFC4984, September 2007, <<https://www.rfc-editor.org/info/rfc4984>>.
- [RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <<https://www.rfc-editor.org/info/rfc5944>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://www.rfc-editor.org/info/rfc6275>>.
- [RFC6831] Farinacci, D., Meyer, D., Zwiebel, J., and S. Venaas, "The Locator/ID Separation Protocol (LISP) for Multicast Environments", RFC 6831, DOI 10.17487/RFC6831, January 2013, <<https://www.rfc-editor.org/info/rfc6831>>.
- [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, <<https://www.rfc-editor.org/info/rfc6832>>.
- [RFC6835] Farinacci, D. and D. Meyer, "The Locator/ID Separation Protocol Internet Groper (LIG)", RFC 6835, DOI 10.17487/RFC6835, January 2013, <<https://www.rfc-editor.org/info/rfc6835>>.
- [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, <<https://www.rfc-editor.org/info/rfc6836>>.
- [RFC6837] Lear, E., "NERD: A Not-so-novel Endpoint ID (EID) to Routing Locator (RLOC) Database", RFC 6837, DOI 10.17487/RFC6837, January 2013, <<https://www.rfc-editor.org/info/rfc6837>>.
- [RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, DOI 10.17487/RFC6935, April 2013, <<https://www.rfc-editor.org/info/rfc6935>>.
- [RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, DOI 10.17487/RFC6936, April 2013, <<https://www.rfc-editor.org/info/rfc6936>>.

- [RFC7052] Schudel, G., Jain, A., and V. Moreno, "Locator/ID Separation Protocol (LISP) MIB", RFC 7052, DOI 10.17487/RFC7052, October 2013, <<https://www.rfc-editor.org/info/rfc7052>>.
- [RFC7215] Jakab, L., Cabellos-Aparicio, A., Coras, F., Domingo-Pascual, J., and D. Lewis, "Locator/Identifier Separation Protocol (LISP) Network Element Deployment Considerations", RFC 7215, DOI 10.17487/RFC7215, April 2014, <<https://www.rfc-editor.org/info/rfc7215>>.
- [RFC7835] Saucez, D., Iannone, L., and O. Bonaventure, "Locator/ID Separation Protocol (LISP) Threat Analysis", RFC 7835, DOI 10.17487/RFC7835, April 2016, <<https://www.rfc-editor.org/info/rfc7835>>.
- [RFC8060] Farinacci, D., Meyer, D., and J. Snijders, "LISP Canonical Address Format (LCAF)", RFC 8060, DOI 10.17487/RFC8060, February 2017, <<https://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, <<https://www.rfc-editor.org/info/rfc8111>>.
- [RFC8378] Moreno, V. and D. Farinacci, "Signal-Free Locator/ID Separation Protocol (LISP) Multicast", RFC 8378, DOI 10.17487/RFC8378, May 2018, <<https://www.rfc-editor.org/info/rfc8378>>.

11.2. Informative References

- [I-D.cheng-lisp-shdht]
Cheng, L. and J. Wang, "LISP Single-Hop DHT Mapping Overlay", Work in Progress, Internet-Draft, draft-cheng-lisp-shdht-04, 15 July 2013, <<http://www.ietf.org/internet-drafts/draft-cheng-lisp-shdht-04.txt>>.
- [I-D.curran-lisp-emacs]
Brim, S., Farinacci, D., Meyer, D., and J. Curran, "EID Mappings Multicast Across Cooperating Systems for LISP", Work in Progress, Internet-Draft, draft-curran-lisp-emacs-00, 9 November 2007, <<http://tools.ietf.org/html/draft-curran-lisp-emacs-00>>.

- [Jakab] Jakab, L., Cabellos, A., Saucez, D., and O. Bonaventure, "LISP-TREE: A DNS Hierarchy to Support the LISP Mapping System, IEEE Journal on Selected Areas in Communications, vol. 28, no. 8, pp. 1332-1343", October 2010.
- [Mathy] Mathy, L., Iannone, L., and O. Bonaventure, "LISP-DHT: Towards a DHT to map identifiers onto locators. The ACM ReArch, Re-Architecting the Internet. Madrid (Spain)", December 2008.
- [Quoitin] Quoitin, B., Iannone, L., Launois, C., and O. Bonaventure, "Evaluating the Benefits of the Locator/Identifier Separation" in Proceedings of 2Nd ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture", 2007.

Appendix A. A Brief History of Location/Identity Separation

The LISP architecture for separation of location and identity resulted from the discussions of this topic at the Amsterdam IAB Routing and Addressing Workshop, which took place in October 2006 [RFC4984].

A small group of like-minded personnel spontaneously formed immediately after that workshop, to work on an idea that came out of informal discussions at the workshop and on various mailing lists. The first Internet-Draft on LISP appeared in January, 2007.

Trial implementations started at that time, with initial trial deployments underway since June 2007; the results of early experience have been fed back into the design in a continuous, ongoing process over several years. LISP at this point represents a moderately mature system, having undergone a long organic series of changes and updates.

LISP transitioned from an IRTF activity to an IETF WG in March 2009, and after numerous revisions, the basic specifications moved to becoming RFCs at the start of 2013 (although work to expand and improve it, and find new uses for it, continues, and undoubtedly will for a long time to come). The LISP WG was rechartered in 2018 to continue work on the LISP base protocol and produce standard-track documents.

A.1. Old LISP Models

LISP, as initially conceived, had a number of potential operating modes, named 'models'. Although they are no used anymore, one occasionally sees mention of them, so they are briefly described here.

LISP 1: EIDs all appear in the normal routing and forwarding tables of the network (i.e. they are 'routable'); this property is used to 'bootstrap' operation, by using this to load EID->RLOC mappings. Packets were sent with the EID as the destination in the outer wrapper; when an ETR saw such a packet, it would send a Map-Reply to the source ITR, giving the full mapping.

LISP 1.5: Similar to LISP 1, but the routability of EIDs happens on a separate network.

LISP 2: EIDs are not routable; EID->RLOC mappings are available from the DNS.

LISP 3: EIDs are not routable; and have to be looked up in in a new EID->RLOC mapping database (in the initial concept, a system using Distributed Hash Tables). Two variants were possible: a 'push' system, in which all mappings were distributed to all ITRs, and a 'pull' system in which ITRs load the mappings they need, as needed.

Authors' Addresses

Albert Cabellos
UPC-BarcelonaTech
c/ Jordi Girona 1-3
08034 Barcelona Catalonia
Spain

Email: acabello@ac.upc.edu

Damien Saucez (Ed.)
Inria
2004 route des Lucioles BP 93
06902 Sophia Antipolis Cedex
France

Email: damien.saucez@inria.fr

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: June 1, 2017

D. Farinacci
lispers.net
D. Meyer
Brocade
J. Snijders
NTT
November 28, 2016

LISP Canonical Address Format (LCAF)
draft-ietf-lisp-lcaf-22

Abstract

This document defines a canonical address format encoding used in LISP control messages and in the encoding of lookup keys for the LISP Mapping Database System.

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 June 1, 2017.

Copyright Notice

Copyright (c) 2016 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	3
2. Definition of Terms	4
3. LISP Canonical Address Format Encodings	5
4. LISP Canonical Address Applications	8
4.1. Segmentation using LISP	8
4.2. Carrying AS Numbers in the Mapping Database	9
4.3. Assigning Geo Coordinates to Locator Addresses	11
4.4. NAT Traversal Scenarios	13
4.5. Multicast Group Membership Information	15
4.6. Traffic Engineering using Re-encapsulating Tunnels	17
4.7. Storing Security Data in the Mapping Database	18
4.8. Source/Destination 2-Tuple Lookups	20
4.9. Replication List Entries for Multicast Forwarding	22
4.10. Applications for AFI List Type	23
4.10.1. Binding IPv4 and IPv6 Addresses	23
4.10.2. Layer-2 VPNs	24
4.10.3. ASCII Names in the Mapping Database	25
4.10.4. Using Recursive LISP Canonical Address Encodings	26
4.10.5. Compatibility Mode Use Case	27
5. Experimental LISP Canonical Address Applications	28
5.1. Convey Application Specific Data	29
5.2. Generic Database Mapping Lookups	30
5.3. PETR Admission Control Functionality	32
5.4. Data Model Encoding	33
5.5. Encoding Key/Value Address Pairs	34
5.6. Multiple Data-Planes	35
6. Security Considerations	37
7. IANA Considerations	38
8. References	39
8.1. Normative References	39
8.2. Informative References	40
Appendix A. Acknowledgments	42
Appendix B. Document Change Log	42
B.1. Changes to draft-ietf-lisp-lcaf-22.txt	42
B.2. Changes to draft-ietf-lisp-lcaf-21.txt	43
B.3. Changes to draft-ietf-lisp-lcaf-20.txt	43
B.4. Changes to draft-ietf-lisp-lcaf-19.txt	43
B.5. Changes to draft-ietf-lisp-lcaf-18.txt	43

IPv6 Encoded Address:

```

      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 = 2               | IPv6 Address ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               ... IPv6 Address ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               ... IPv6 Address ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               ... IPv6 Address ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               ... IPv6 Address     |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

This document describes the currently-defined AFIs the LISP protocol uses along with their encodings and introduces the LISP Canonical Address Format (LCAF) that can be used to define the LISP-specific encodings for arbitrary AFI values.

Specific detail uses for the LCAF types defined in this document can be found in the use-case documents that use them. The same LCAF type may be used by more than one use-case document. As an experimental specification, this work is by definition, incomplete. The LCAF types defined in this document are to support experimentation and intended for cautious use in self-contained environments in support of the corresponding use-case documents. This document provides assignment for an initial set of approved LCAF Types (registered with IANA) and additional unapproved LCAF Types [RFC6830]. The unapproved LCAF encodings are defined to support further study and experimentation.

2. Definition of Terms

Address Family Identifier (AFI): a term used to describe an address encoding in a packet. Address families are defined for IPv4 and IPv6. See [AFI] and [RFC3232] for details. The reserved AFI value of 0 is used in this specification to indicate an unspecified encoded address where the length of the address is 0 bytes following the 16-bit AFI value of 0.

Unspecified Address Format:

```

      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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               |                               <no address follows>
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Endpoint ID (EID): a 32-bit (for IPv4) or 128-bit (for IPv6) value used in the source and destination address fields of the first (most inner) LISP header of a packet. The host obtains a destination EID the same way it obtains a destination address today, for example through a DNS lookup or SIP exchange. The source EID is obtained via existing mechanisms used to set a host's "local" IP address. An EID is allocated to a host from an EID-prefix block associated with the site where the host is located. An EID can be used by a host to refer to other hosts.

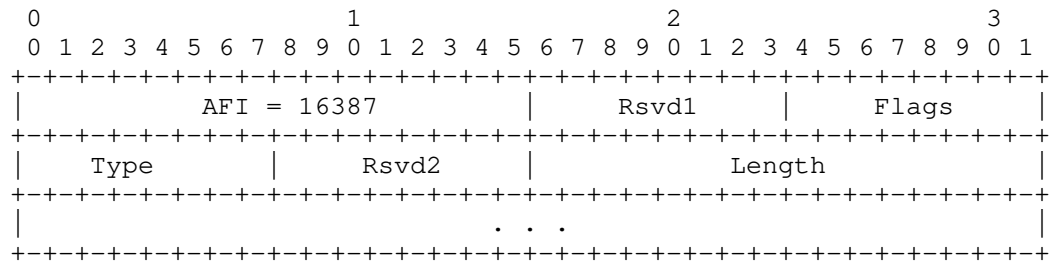
Routing Locator (RLOC): the IPv4 or IPv6 address of an egress tunnel router (ETR). It is the output of a EID-to-RLOC mapping lookup. An EID maps to one or more RLOCs. Typically, RLOCs are numbered from topologically aggregatable blocks that are assigned to a site at each point to which it attaches to the global Internet; where the topology is defined by the connectivity of provider networks, RLOCs can be thought of as Provider-Assigned (PA) addresses. Multiple RLOCs can be assigned to the same ETR device or to multiple ETR devices at a site.

3. LISP Canonical Address Format Encodings

IANA has assigned AFI value 16387 (0x4003) to the LISP architecture and protocols. This specification defines the encoding format of the LISP Canonical Address (LCA). This section defines all types for which an initial allocation in the LISP-LCAF registry is requested. See IANA Considerations section for the complete list of such types.

The Address Family AFI definitions from [AFI] only allocate code-points for the AFI value itself. The length of the address or entity that follows is not defined and is implied based on conventional experience. When the LISP protocol uses LCAF definitions from this document, the AFI-based address lengths are specified in this document. When new LCAF definitions are defined in other use case documents, the AFI-based address lengths for any new AFI encoded addresses are specified in those documents.

The first 6 bytes of an LISP Canonical Address are followed by a variable number of fields of variable length:



Rsvd1/Rsvd2: these 8-bit fields are reserved for future use and MUST be transmitted as 0 and ignored on receipt.

Flags: this 8-bit field is for future definition and use. For now, set to zero on transmission and ignored on receipt.

Type: this 8-bit field is specific to the LISP Canonical Address formatted encodings. Currently allocated (both approved and unapproved) values are:

- Type 0: Null Body Type
- Type 1: AFI List Type
- Type 2: Instance ID Type
- Type 3: AS Number Type
- Type 4: Application Data Type
- Type 5: Geo Coordinates Type
- Type 6: Opaque Key Type
- Type 7: NAT-Traversal Type
- Type 8: Nonce Locator Type
- Type 9: Multicast Info Type
- Type 10: Explicit Locator Path Type
- Type 11: Security Key Type
- Type 12: Source/Dest Key Type

Type 13: Replication List Entry Type

Type 14: JSON Data Model Type

Type 15: Key/Value Address Pair Type

Type 16: Encapsulation Format Type

Length: this 16-bit field is in units of bytes and covers all of the LISP Canonical Address payload, starting and including the byte after the Length field. When including the AFI, an LCAF encoded address will have a minimum length of 8 bytes when the Length field is 0. The 8 bytes include the AFI, Flags, Type, Rsvd1, Rsvd2, and Length fields. When the AFI is not next to an encoded address in a control message, then the encoded address will have a minimum length of 6 bytes when the Length field is 0. The 6 bytes include the Flags, Type, Rsvd1, Rsvd2, and Length fields.

[RFC6830] states RLOC records based on an IP address are sorted when encoded in control messages so the locator-set has consistent order across all xTRs for a given EID. The sort order is based on sort-key {afi, RLOC-address}. When an RLOC based on an IP address is LCAF encoded, the sort-key is {afi, LCAF-Type}. Therefore, when a locator-set has a mix of AFI records and LCAF records, they are ordered from smallest to largest AFI value.

4. LISP Canonical Address Applications

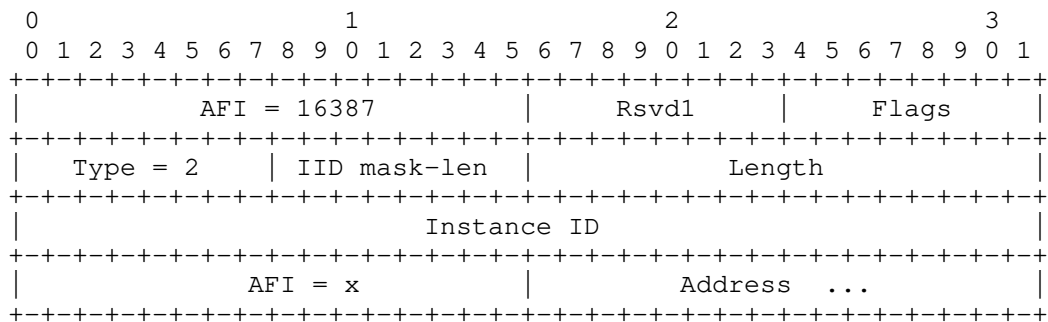
The following sections define the LCAF for the currently approved initial set of Type values.

4.1. Segmentation using LISP

When multiple organizations inside of a LISP site are using private addresses [RFC1918] as EID-prefixes, their address spaces must remain segregated due to possible address duplication. An Instance ID in the address encoding can aid in making the entire AFI-based address unique.

Another use for the Instance ID LISP Canonical Address Format is when creating multiple segmented VPNs inside of a LISP site where keeping EID-prefix based subnets is desirable.

Instance ID LISP Canonical Address Format:



IID mask-len: if the AFI is set to 0, then this format is not encoding an extended EID-prefix but rather an instance-ID range where the 'IID mask-len' indicates the number of high-order bits used in the Instance ID field for the range. The low-order bits of the Instance ID field must be 0.

Length: length in bytes starting and including the byte after this Length field.

Instance ID: the low-order 24-bits that can go into a LISP data header when the I-bit is set. See [RFC6830] for details. The reason for the length difference is so that the maximum number of instances supported per mapping system is 2^{32} while conserving space in the LISP data header. This comes at the expense of limiting the maximum number of instances per xTR to 2^{24} . If an xTR is configured with multiple instance-IDs where the value in

the high-order 8 bits are the same, then the low-order 24 bits MUST be unique.

AFI = x: x can be any AFI value from [AFI].

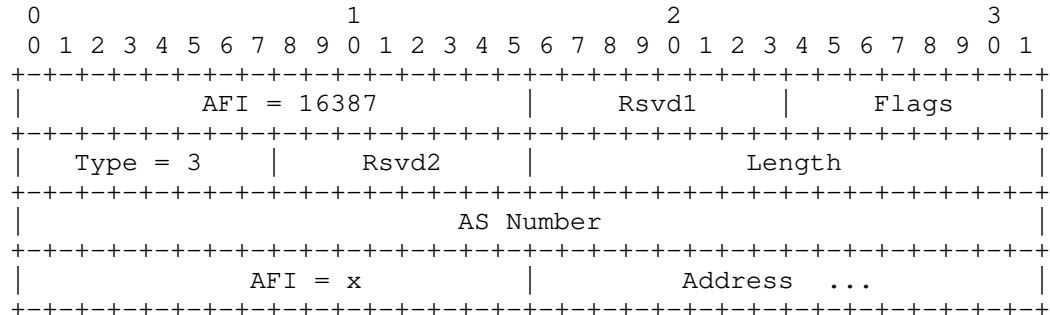
This LISP Canonical Address Type can be used to encode either EID or RLOC addresses.

Usage: When used as a lookup key, the EID is regarded as an extended-EID in the mapping system. This encoding is used in EID records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages.

4.2. Carrying AS Numbers in the Mapping Database

When an AS number is stored in the LISP Mapping Database System for either policy or documentation reasons, it can be encoded in a LISP Canonical Address.

AS Number LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

AS Number: the 32-bit AS number of the autonomous system that has been assigned to either the EID or RLOC that follows.

AFI = x: x can be any AFI value from [AFI].

The AS Number Canonical Address Type can be used to encode either EID or RLOC addresses. The former is used to describe the LISP-ALT AS number the EID-prefix for the site is being carried for. The latter is used to describe the AS that is carrying RLOC based prefixes in the underlying routing system.

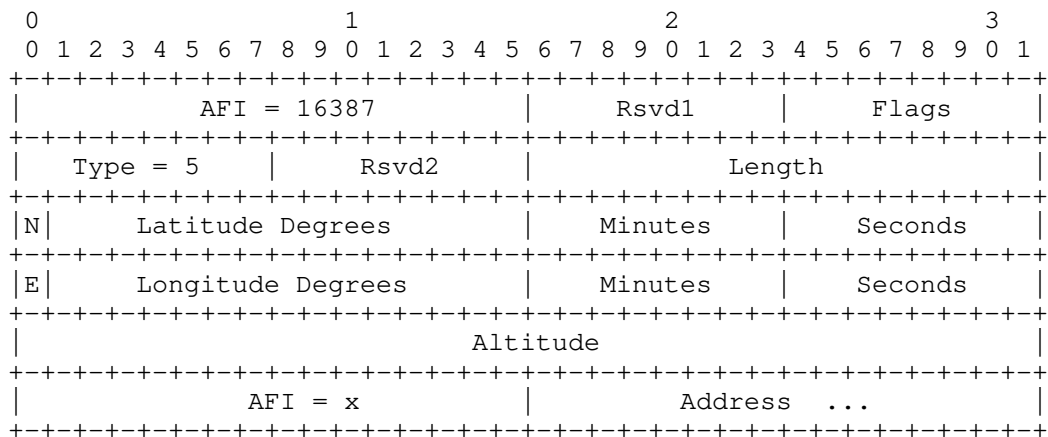
Usage: This encoding can be used in EID or RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages.

4.3. Assigning Geo Coordinates to Locator Addresses

If an ETR desires to send a Map-Reply describing the Geo Coordinates for each locator in its locator-set, it can use the Geo Coordinate Type to convey physical location information.

Coordinates are specified using the WGS-84 (World Geodetic System) reference coordinate system [WGS-84].

Geo Coordinate LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

N: When set to 1 means North, otherwise South.

Latitude Degrees: Valid values range from 0 to 90 degrees above or below the equator (northern or southern hemisphere, respectively).

Latitude Minutes: Valid values range from 0 to 59.

Latitude Seconds: Valid values range from 0 to 59.

E: When set to 1 means East, otherwise West.

Longitude Degrees: Valid values are from 0 to 180 degrees right or left of the Prime Meridian.

Longitude Minutes: Valid values range from 0 to 59.

Longitude Seconds: Valid values range from 0 to 59.

Altitude: Height relative to sea level in meters. This is a two's complement signed integer meaning that the altitude could be below sea level. A value of 0x7fffffff indicates no Altitude value is encoded.

AFI = x: x can be any AFI value from [AFI].

The Geo Coordinates Canonical Address Type can be used to encode either EID or RLOC addresses. When used for EID encodings, you can determine the physical location of an EID along with the topological location by observing the locator-set.

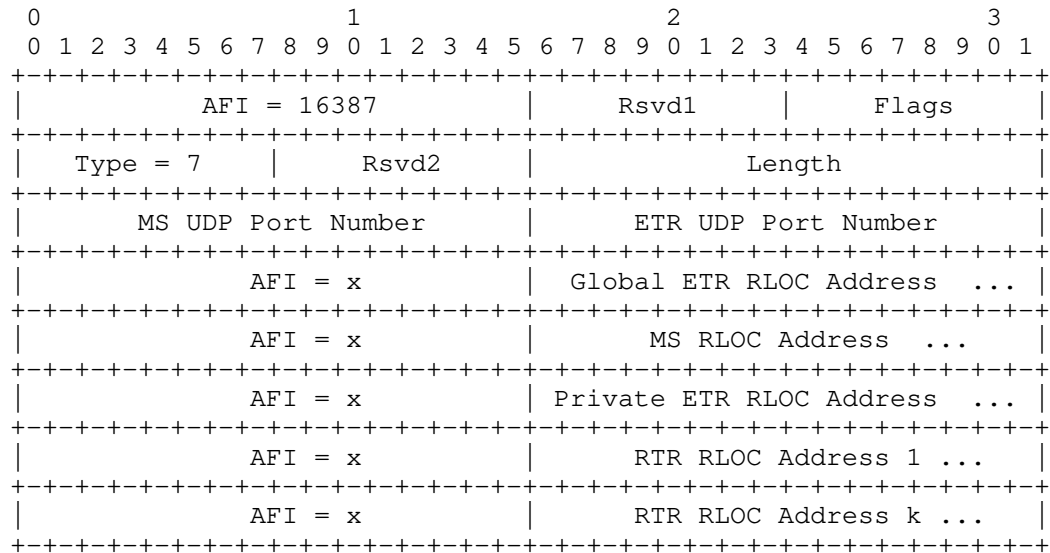
Usage: This encoding can be used in EID or RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages.

The use of the Geo-Coordinates LCAF encoding raises privacy issues as location information is privacy sensitive, and possibly unexpectedly privacy sensitive information may be conveyed, e.g. if the location information corresponds to a router located in a person's home. Therefore, this encoding should not be used unless needed for operation of a LISP deployment. Before electing to utilize this encoding, care should be taken to ensure the appropriate policies are being used by the EID for controlling the conveyed information.

4.4. NAT Traversal Scenarios

When a LISP system is conveying global address and mapped port information when traversing through a NAT device, the NAT-Traversal LCAF Type is used. See [I-D.ermagan-lisp-nat-traversal] for details.

NAT-Traversal Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

MS UDP Port Number: this is the UDP port number of the Map-Server and is set to 4342.

ETR UDP Port Number: this is the port number returned to a LISP system which was copied from the source port from a packet that has flowed through a NAT device.

AFI = x: x can be any AFI value from [AFI].

Global ETR RLOC Address: this is an address known to be globally unique built by NAT-traversal functionality in a LISP router.

MS RLOC Address: this is the address of the Map-Server used in the destination RLOC of a packet that has flowed through a NAT device.

Private ETR RLOC Address: this is an address known to be a private address inserted in this LCAF by a LIISP router that resides on the private side of a NAT device.

RTR RLOC Address: this is an encapsulation address used by an ITR or PITR which resides behind a NAT device. This address is known to have state in a NAT device so packets can flow from it to the LIISP ETR behind the NAT. There can be one or more NAT Reencapsulating Tunnel Router (RTR) [I-D.ermagan-lisp-nat-traversal] addresses supplied in these set of fields. The number of RTRs encoded is determined by parsing each field. When there are no RTRs supplied, the RTR fields can be omitted and reflected by the LCAF length field or an AFI of 0 can be used to indicate zero RTRs encoded.

Usage: This encoding can be used in Info-Request and Info-Reply messages. The mapping system does not store this information. The information is used by an xTR and Map-Server to convey private and public address information when traversing NAT and firewall devices.

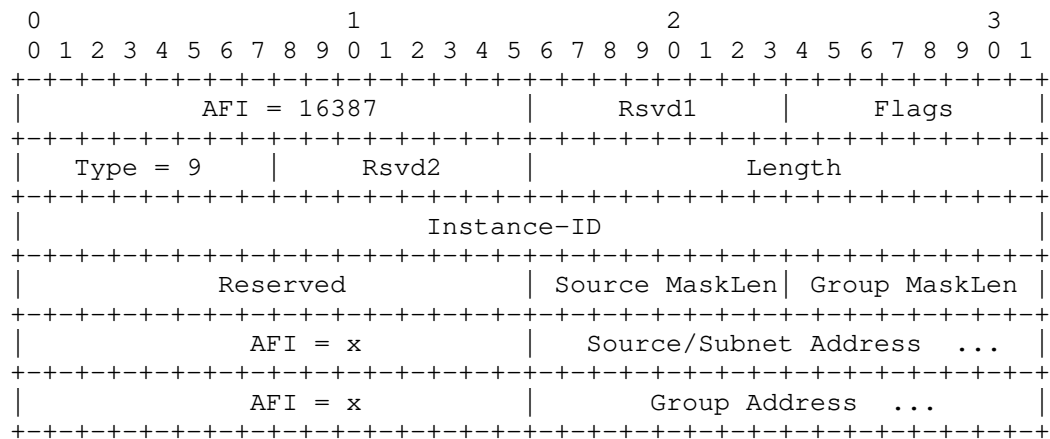
Care should be taken to protect privacy against the adverse use of a Global or Private ETR RLOC Address by ensuring policy controls are used during EID registrations that use this LCAF Type in RLOC-records. Refer to the use case documents for additional information.

4.5. Multicast Group Membership Information

Multicast group information can be published in the mapping database. So a lookup on a group address EID can return a replication list of RLOC group addresses or RLOC unicast addresses. The intent of this type of unicast replication is to deliver packets to multiple ETRs at receiver LISP multicast sites. The locator-set encoding for this EID record type can be a list of ETRs when they each register with "Merge Semantics". The encoding can be a typical AFI-encoded locator address. When an RTR list is being registered (with multiple levels according to [I-D.coras-lisp-re]), the Replication List Entry LCAF type is used for locator encoding.

This LCAF encoding can be used to send broadcast packets to all members of a subnet when an EID is away from its home subnet location.

Multicast Info Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

Reserved: must be set to zero and ignored on receipt.

Instance ID: the low-order 24-bits that can go into a LISP data header when the I-bit is set. See [RFC6830] for details. The use of the Instance-ID in this LCAF type is to associate a multicast forwarding entry for a given VPN. The instance-ID describes the VPN and is registered to the mapping database system as a 3-tuple of (Instance-ID, S-prefix, G-prefix).

Source MaskLen: the mask length of the source prefix that follows.
The length is the number of high-order mask bits set.

Group MaskLen: the mask length of the group prefix that follows.
The length is the number of high-order mask bits set.

AFI = x: x can be any AFI value from [AFI]. When a specific address family has a multicast address semantic, this field must be either a group address or a broadcast address.

Source/Subnet Address: is the source address or prefix for encoding a (S,G) multicast entry.

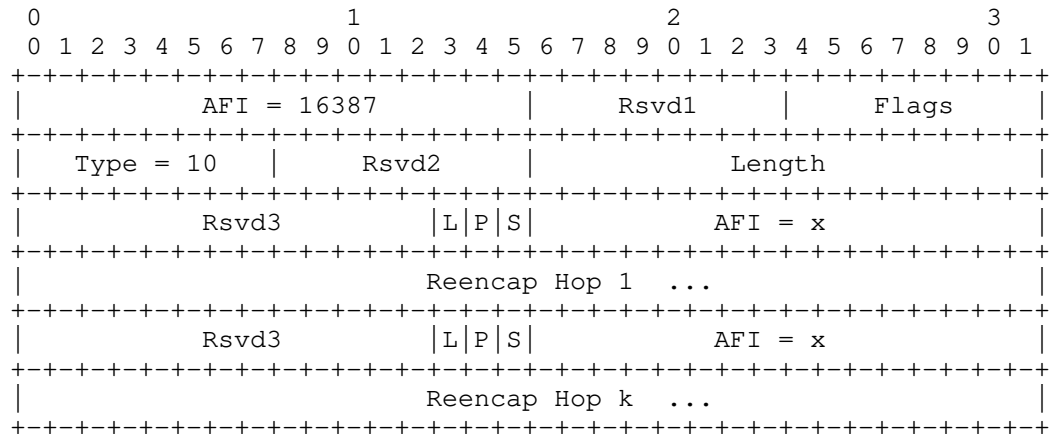
Group Address: is the group address or group prefix for encoding (S,G) or (*,G) multicast entries.

Usage: This encoding can be used in EID records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages.

4.6. Traffic Engineering using Re-encapsulating Tunnels

For a given EID lookup into the mapping database, this LCAF can be returned to provide a list of locators in an explicit re-encapsulation path. See [I-D.farinacci-lisp-te] for details.

Explicit Locator Path (ELP) Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

Rsvd3: this field is reserved for future use and MUST be transmitted as 0 and ignored on receipt.

Lookup bit (L): this is the Lookup bit used to indicate to the user of the ELP to not use this address for encapsulation but to look it up in the mapping database system to obtain an encapsulating RLOC address.

RLOC-Probe bit (P): this is the RLOC-probe bit which means the Reencap Hop allows RLOC-probe messages to be sent to it. When the R-bit is set to 0, RLOC-probes must not be sent. When a Reencap Hop is an anycast address then multiple physical Reencap Hops are using the same RLOC address. In this case, RLOC-probes are not needed because when the closest RLOC address is not reachable another RLOC address can be reachable.

Strict bit (S): this is the strict bit which means the associated Reencap Hop is required to be used. If this bit is 0, the reencapsulator can skip this Reencap Hop and go to the next one in the list.

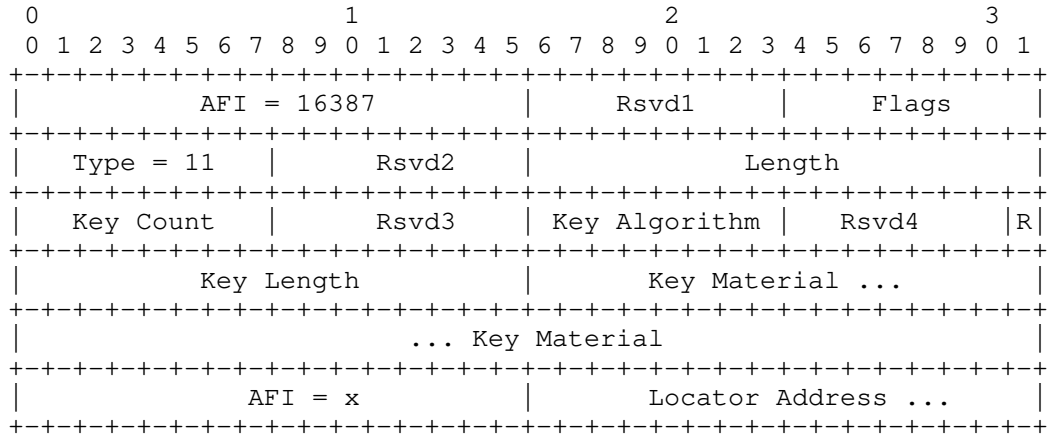
AFI = x: x can be any AFI value from [AFI]. When a specific AFI has its own encoding of a multicast address, this field must be either a group address or a broadcast address.

Usage: This encoding can be used in RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. This encoding does not need to be understood by the mapping system for mapping database lookups since this LCAF type is not a lookup key.

4.7. Storing Security Data in the Mapping Database

When a locator in a locator-set has a security key associated with it, this LCAF will be used to encode key material. See [I-D.ietf-lisp-ddt] for details.

Security Key Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

Key Count: the Key Count field declares the number of Key sections included in this LCAF. A key section is made up of "Key Length" and "Key Material" fields.

Rsvd3: this field is reserved for future use and MUST be transmitted as 0 and ignored on receipt.

Key Algorithm: the Algorithm field identifies the key's cryptographic algorithm and specifies the format of the Public Key field. Refer to the [I-D.ietf-lisp-ddt] and [I-D.ietf-lisp-crypto] use cases for definitions of this field.

Rsvd4: this field is reserved for future use and MUST be transmitted as 0 and ignored on receipt.

R bit: this is the revoke bit and, if set, it specifies that this Key is being Revoked.

Key Length: this field determines the length in bytes of the Key Material field.

Key Material: the Key Material field stores the key material. The format of the key material stored depends on the Key Algorithm field.

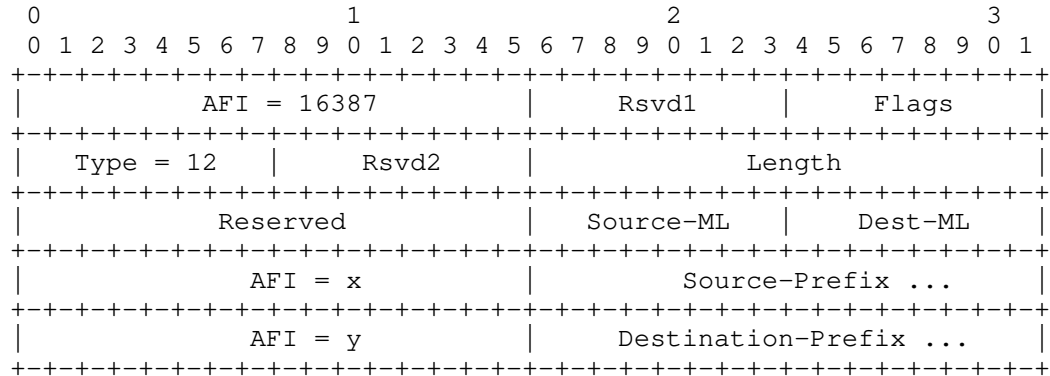
AFI = x: x can be any AFI value from [AFI]. This is the locator address that owns the encoded security key.

Usage: This encoding can be used in EID or RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages.

4.8. Source/Destination 2-Tuple Lookups

When both a source and destination address of a flow need consideration for different locator-sets, this 2-tuple key is used in EID fields in LISP control messages. When the Source/Dest key is registered to the mapping database, it can be encoded as a source-prefix and destination-prefix. When the Source/Dest is used as a key for a mapping database lookup the source and destination come from a data packet.

Source/Dest Key Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

Reserved: must be set to zero and ignore on receipt.

Source-ML: the mask length of the source prefix that follows. The length is the number of high-order mask bits set.

Dest-ML: the mask length of the destination prefix that follows. The length is the number of high-order mask bits set.

AFI = x: x can be any AFI value from [AFI].

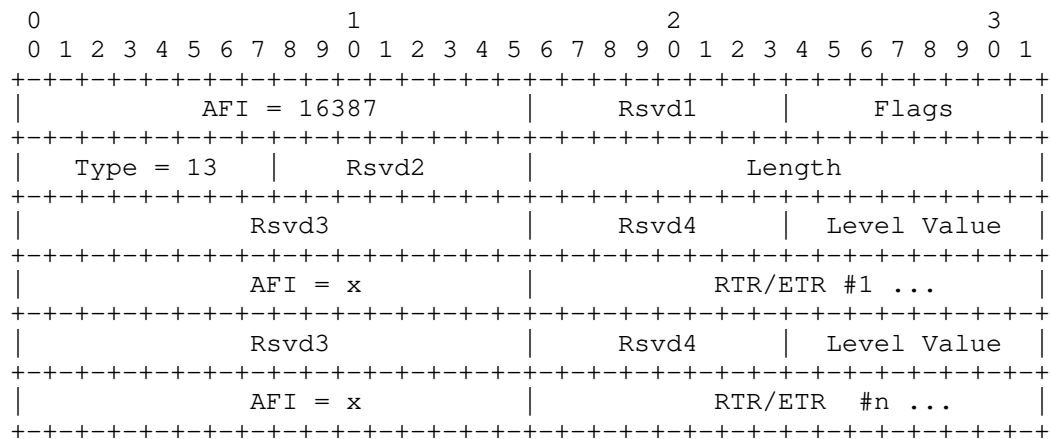
AFI = y: y can be any AFI value from [AFI]. When a specific address family has a multicast address semantic, this field must be either a group address or a broadcast address.

Usage: This encoding can be used in EID records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages. Refer to [I-D.farinacci-lisp-te] for usage details of this LCAF type.

4.9. Replication List Entries for Multicast Forwarding

The Replication List Entry LCAF type is an encoding for a locator being used for unicast replication according to the specification in [I-D.coras-lisp-re]. This locator encoding is pointed to by a Multicast Info LCAF Type and is registered by Re-encapsulating Tunnel Routers (RTRs) that are participating in an overlay distribution tree. Each RTR will register its locator address and its configured level in the distribution tree.

Replication List Entry Address Format:



Length: length in bytes starting and including the byte after this Length field.

Rsvd3/Rsvd4: must be set to zero and ignore on receipt.

Level Value: this value is associated with the level within the overlay distribution tree hierarchy where the RTR resides. The level numbers are ordered from lowest value being close to the ITR (meaning that ITRs replicate to level-0 RTRs) and higher levels are further downstream on the distribution tree closer to ETRs of multicast receiver sites.

AFI = x: x can be any AFI value from [AFI]. A specific AFI has its own encoding of either a unicast or multicast locator address. For efficiency reasons, all RTR/ETR entries for the same level should be combined together by a Map-Server to avoid searching through the entire multi-level list of locator entries in a Map-Reply message.

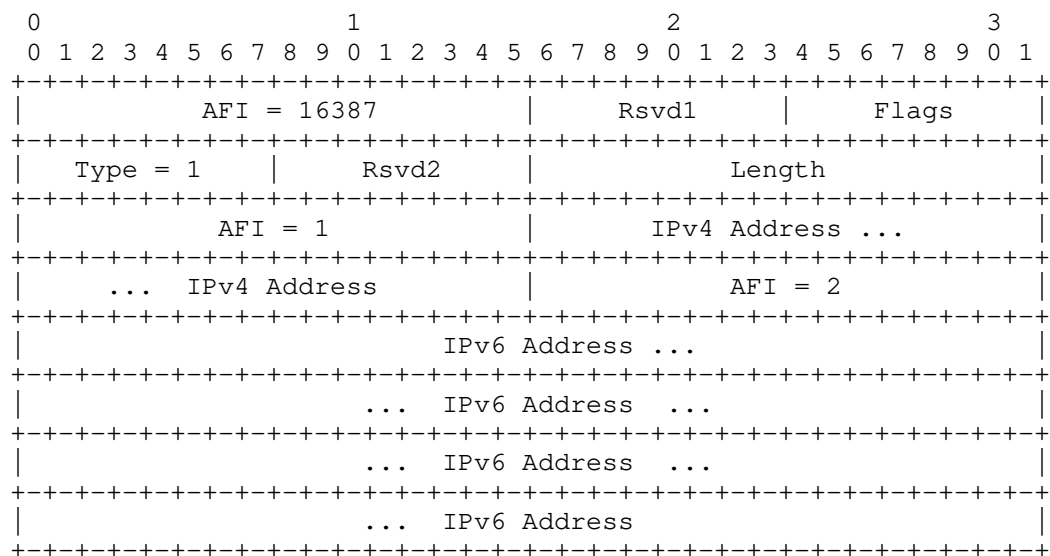
Usage: This encoding can be used in RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages.

4.10. Applications for AFI List Type

4.10.1. Binding IPv4 and IPv6 Addresses

When header translation between IPv4 and IPv6 is desirable a LISP Canonical Address can use the AFI List Type to carry a variable number of AFIs in one LCAF AFI.

Address Binding LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

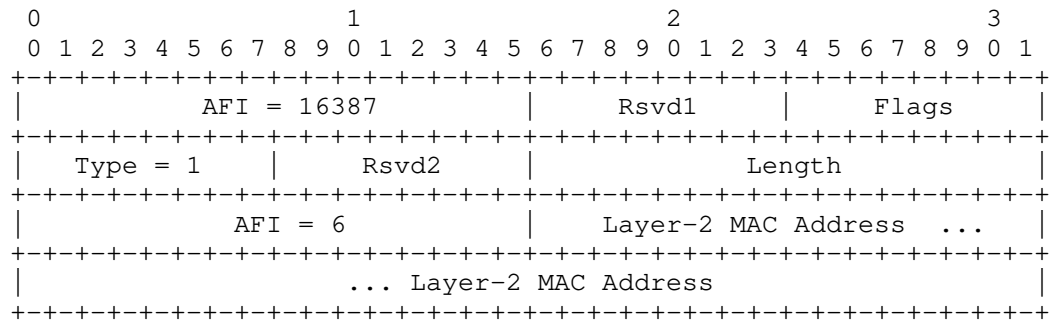
This type of address format can be included in a Map-Request when the address is being used as an EID, but the Mapping Database System lookup destination can use only the IPv4 address. This is so a Mapping Database Service Transport System, such as LISP-ALT [RFC6836], can use the Map-Request destination address to route the control message to the desired LISP site.

Usage: This encoding can be used in EID or RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. See subsections in this section for specific use cases.

4.10.2. Layer-2 VPNs

When MAC addresses are stored in the LISP Mapping Database System, the AFI List Type can be used to carry AFI 6.

MAC Address LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

This address format can be used to connect layer-2 domains together using LISP over an IPv4 or IPv6 core network to create a layer-2 VPN. In this use case, a MAC address is being used as an EID, and the locator-set that this EID maps to can be an IPv4 or IPv6 RLOCs, or even another MAC address being used as an RLOC. See [I-D.portoles-lisp-eid-mobility] for how layer-2 VPNs operate when doing EID mobility.

Care should be taken to protect privacy against the adverse use of a Layer-2 MAC Address by ensuring policy controls are used during EID registrations that use AFI=6 encodings in RLOC-records. Refer to the use case documents for additional information.

4.10.3. ASCII Names in the Mapping Database

If DNS names [RFC1035] or URIs [RFC3986] are stored in the LISP Mapping Database System, the AFI List Type can be used to carry an ASCII string.

ASCII LISP Canonical Address Format:

```

      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 = 1   |   Rsvd2   |               Length               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|               AFI = 17               |   DNS Name or URI   ...   |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

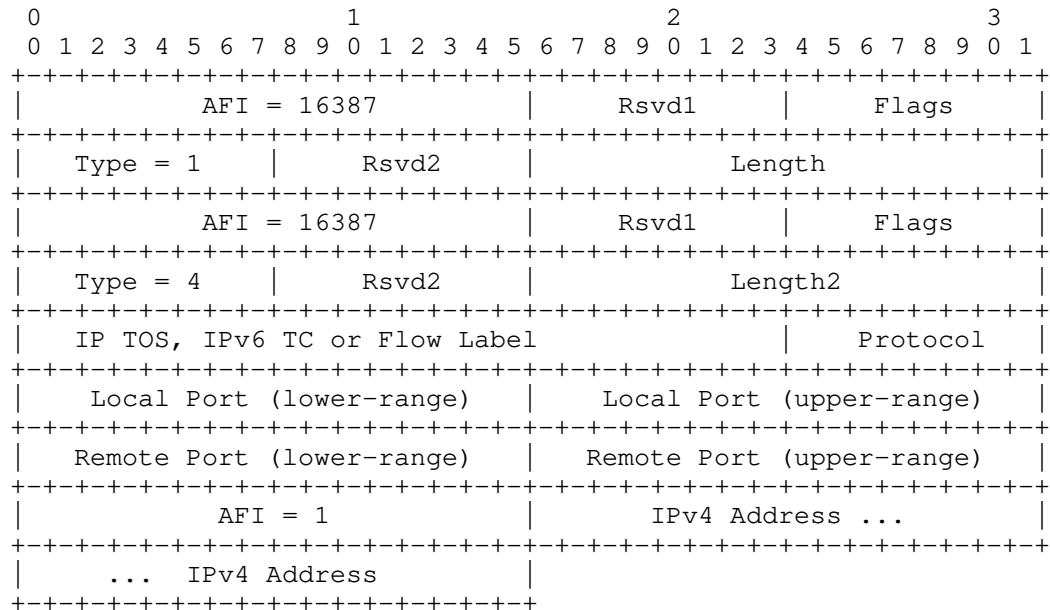
Length: length in bytes starting and including the byte after this Length field.

An example for using DNS names is when an ETR registers a mapping with an EID-record encoded as (AFI=1, 10.0.0.0/8) with a RLOC-record (AFI=17, "router.abc.com").

4.10.4. Using Recursive LISP Canonical Address Encodings

When any combination of above is desirable, the AFI List Type value can be used to carry within the LCAF AFI another LCAF AFI (for example, Application Specific Data see Section 5.1).

Recursive LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

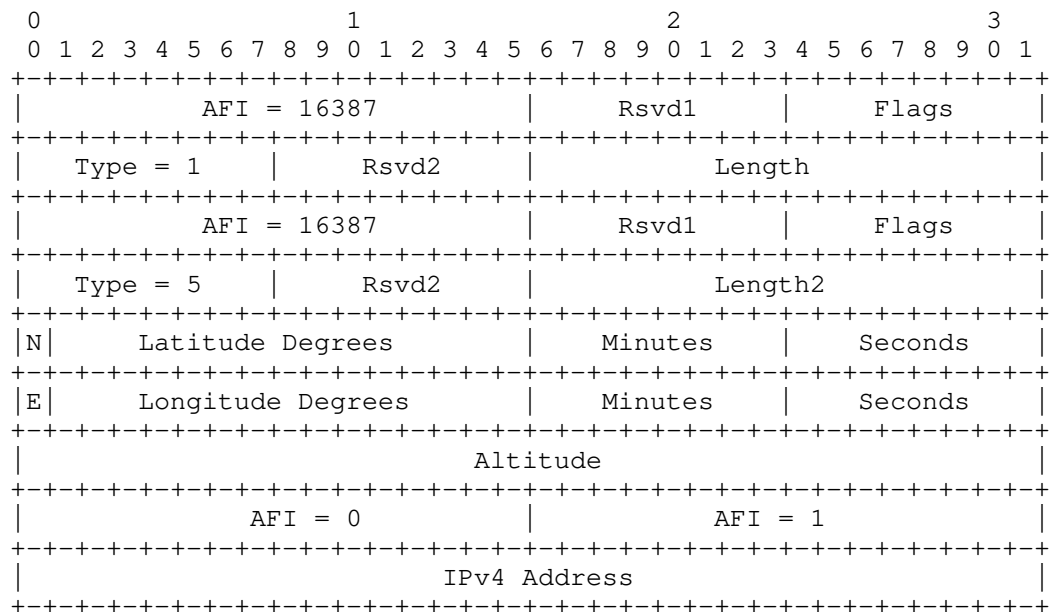
Length2: length in bytes starting and including the byte after this Length2 field.

This format could be used by a Mapping Database Transport System, such as LISP-ALT [RFC6836], where the AFI=1 IPv4 address is used as an EID and placed in the Map-Request destination address by the sending LISP system. The ALT system can deliver the Map-Request to the LISP destination site independent of the Application Data Type AFI payload values. When this AFI is processed by the destination LISP site, it can return different locator-sets based on the type of application or level of service that is being requested.

4.10.5. Compatibility Mode Use Case

A LISP system should use the AFI List Type format when sending to LISP systems that do not support a particular LCAF Type used to encode locators. This allows the receiving system to be able to parse a locator address for encapsulation purposes. The list of AFIs in an AFI List LCAF Type has no semantic ordering and a receiver should parse each AFI element no matter what the ordering.

Compatibility Mode Address Format:



Length: length in bytes starting and including the byte after this Length field.

Length2: length in bytes starting and including the byte after this Length2 field.

If a system does not recognized the Geo Coordinate LCAF Type that is accompanying a locator address, an encoder can include the Geo Coordinate LCAF Type embedded in a AFI List LCAF Type where the AFI in the Geo Coordinate LCAF is set to 0 and the AFI encoded next in the list is encoded with a valid AFI value to identify the locator address.

A LISP system is required to support the AFI List LCAF Type to use this procedure. It would skip over 10 bytes of the Geo Coordinate

LCAF Type to get to the locator address encoding (an IPv4 locator address). A LISP system that does support the Geo Coordinate LCAF Type can support parsing the locator address within the Geo Coordinate LCAF encoding or in the locator encoding that follows in the AFI List LCAF.

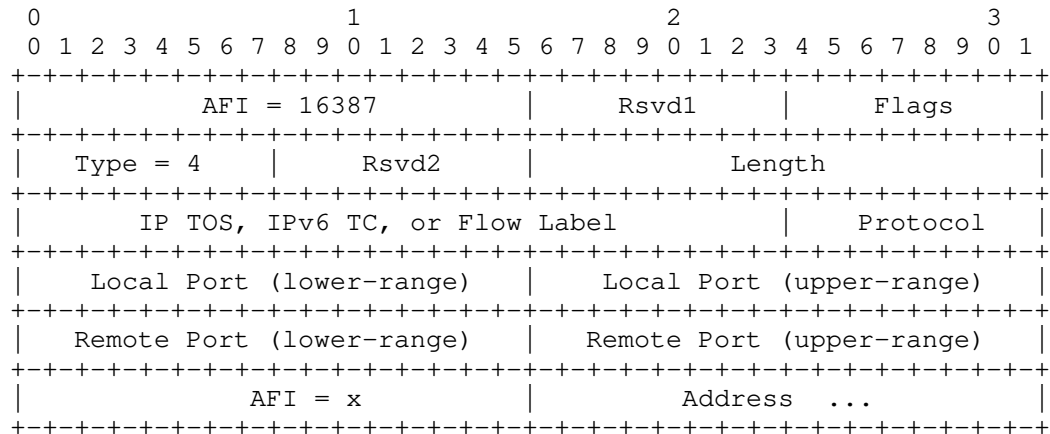
5. Experimental LISP Canonical Address Applications

The following sections describe experimental LCAF encodings. These LCAF Types are not approved (registered with IANA). The inclusion of these encodings in this document are in support of further study and experimentation to determine whether these encodings are functional, if there is a demand for these use cases, and better understand deployment considerations. As noted previously, these LCAF Types are restricted to cautious use in self-contained environments in support of the corresponding use-case documents.

5.1. Convey Application Specific Data

When a locator-set needs to be conveyed based on the type of application or the Per-Hop Behavior (PHB) of a packet, the Application Data Type can be used.

Application Data LISP Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

IP TOS, IPv6 TC, or Flow Label: this field stores the 8-bit IPv4 TOS field used in an IPv4 header, the 8-bit IPv6 Traffic Class or Flow Label used in an IPv6 header.

Local Port/Remote Port Ranges: these fields are from the TCP, UDP, or SCTP transport header. A range can be specified by using a lower value and an upper value. When a single port is encoded, the lower and upper value fields are the same.

AFI = x: x can be any AFI value from [AFI].

The Application Data Canonical Address Type is used for an EID encoding when an ITR wants a locator-set for a specific application. When used for an RLOC encoding, the ETR is supplying a locator-set for each specific application it has been configured to advertise.

Usage: This encoding can be used in EID records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages. When LISP-DDT [I-D.ietf-lisp-ddt] is used as the mapping system mechanism, extended EIDs are used in Map-Referral messages. This LCAF type is used as a

lookup key to the mapping system that can return a longest-match or exact-match entry.

5.2. Generic Database Mapping Lookups

When the LISP Mapping Database system holds information accessed by a generic formatted key (where the key is not the usual IPv4 or IPv6 address), an opaque key may be desirable.

Opaque Key LISP Canonical Address Format:

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	2	3	4	5	6	7	8	9																								
AFI = 16387																Rsvd1																Flags																															
Type = 6								Rsvd2								Length																																															
Key Field Num								Key Wildcard Fields																Key . . .																																							
. . . Key																																																															

Length: length in bytes starting and including the byte after this Length field.

Key Field Num: the value of this field is the number of "Key" sub-fields minus 1, the "Key" field can be broken up into. So if this

field has a value of 0, there is 1 sub-field in the "Key". The width of the sub-fields are fixed length. So for a key size of 8 bytes, with a Key Field Num of 3, allows 4 sub-fields of 2 bytes each in length. Allowing for a reasonable number of 16 sub-field separators, valid values range from 0 to 15.

Key Wildcard Fields: describes which fields in the key are not used as part of the key lookup. This wildcard encoding is a bitfield. Each bit is a don't-care bit for a corresponding field in the key. Bit 0 (the low-order bit) in this bitfield corresponds the first field, the low-order field in the key, bit 1 the second field, and so on. When a bit is set in the bitfield it is a don't-care bit and should not be considered as part of the database lookup. When the entire 16-bits is set to 0, then all bits of the key are used for the database lookup.

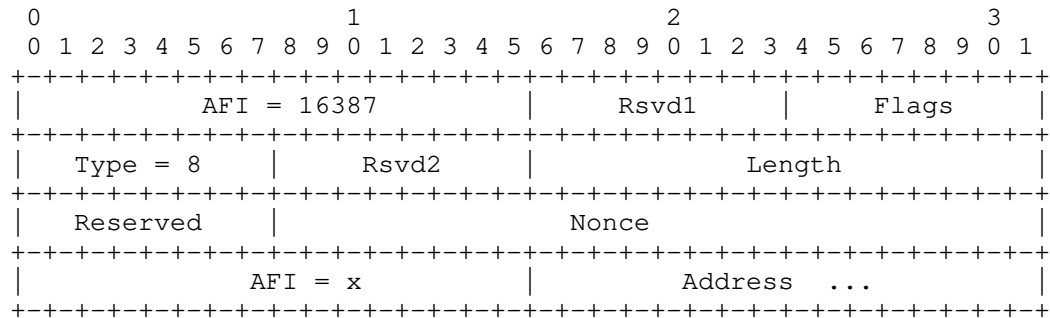
Key: the variable length key used to do a LISP Database Mapping lookup. The length of the key is the value n (as shown above).

Usage: This is an experimental type where the usage has not been defined yet.

5.3. PETR Admission Control Functionality

When a public PETR device wants to verify who is encapsulating to it, it can check for a specific nonce value in the LISP encapsulated packet. To convey the nonce to admitted ITRs or PITRs, this LCAF is used in a Map-Register or Map-Reply locator-record.

Nonce Locator Canonical Address Format:



Length: length in bytes starting and including the byte after this Length field.

Reserved: must be set to zero and ignore on receipt.

Nonce: this is a nonce value returned by an ETR in a Map-Reply locator-record to be used by an ITR or PITR when encapsulating to the locator address encoded in the AFI field of this LCAF type. This nonce value is inserted in the nonce field in the LISP header encapsulation.

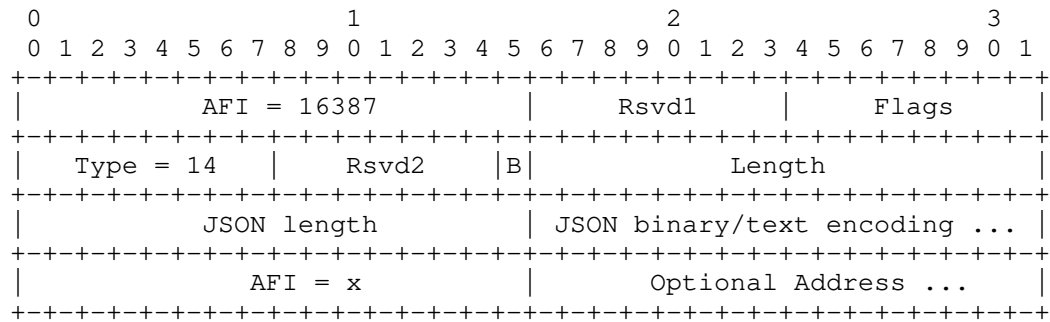
AFI = x: x can be any AFI value from [AFI].

Usage: This is an experimental type where the usage has not been defined yet.

5.4. Data Model Encoding

This type allows a JSON data model to be encoded either as an EID or RLOC.

JSON Data Model Type Address Format:



Length: length in bytes starting and including the byte after this Length field.

B bit: indicates that the JSON field is binary encoded according to [JSON-BINARY] when the bit is set to 1. Otherwise the encoding is based on text encoding according to [RFC7159].

JSON length: length in octets of the following 'JSON binary/text encoding' field.

JSON binary/text encoding field: a variable length field that contains either binary or text encodings.

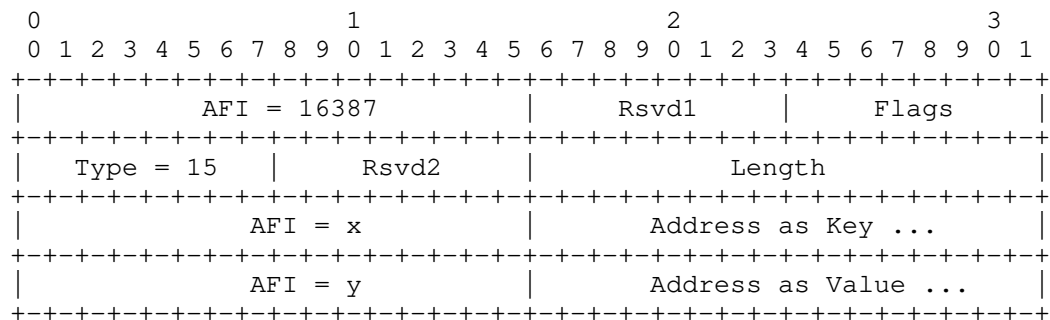
AFI = x: x can be any AFI value from [AFI]. A specific AFI has its own encoding of either a unicast or multicast locator address. All RTR/ETR entries for the same level should be combined together by a Map-Server to avoid searching through the entire multi-level list of locator entries in a Map-Reply message.

Usage: This is an experimental type where the usage has not been defined yet. An example mapping is an EID-record encoded as a distinguished-name "cpe-rotuer" and a RLOC-record encoded as a JSON string "{ "router-address" : "1.1.1.1", "router-mask" : "8" }".

5.5. Encoding Key/Value Address Pairs

The Key/Value pair is, for example, useful for attaching attributes to other elements of LISP packets, such as EIDs or RLOCs. When attaching attributes to EIDs or RLOCs, it's necessary to distinguish between the element that should be used as EID or RLOC, and hence as the key for lookups, and additional attributes. This is especially the case when the difference cannot be determined from the types of the elements, such as when two IP addresses are being used.

Key/Value Pair Address Format:



Length: length in bytes starting and including the byte after this Length field.

AFI = x: x is the "Address as Key" AFI that can have any value from [AFI]. A specific AFI has its own encoding of either a unicast or multicast locator address. All RTR/ETR entries for the same level should be combined together by a Map-Server to avoid searching through the entire multi-level list of locator entries in a Map-Reply message.

Address as Key: this AFI-encoded address will be attached with the attributes encoded in "Address as Value" which follows this field.

AFI = y: y is the "Address of Value" AFI that can have any value from [AFI]. A specific AFI has its own encoding of either a unicast or multicast locator address. All RTR/ETR entries for the same level should be combined together by a Map-Server to avoid searching through the entire multi-level list of locator entries in a Map-Reply message.

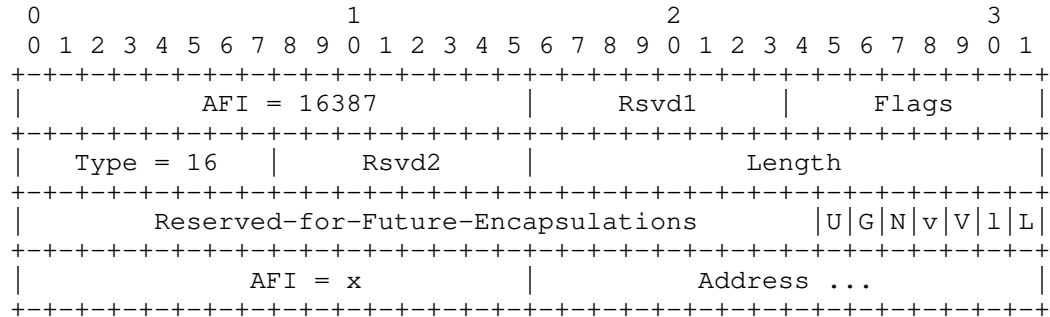
Address as Value: this AFI-encoded address will be the attribute address that goes along with "Address as Key" which precedes this field.

Usage: This is an experimental type where the usage has not been defined yet.

5.6. Multiple Data-Planes

Overlays are becoming popular in many parts of the network which have created an explosion of data-plane encapsulation headers. Since the LIISP mapping system can hold many types of address formats, it can represent the encapsulation format supported by an RLOC as well. When an encapsulator receives a Map-Reply with an Encapsulation Format LCAF Type encoded in an RLOC-record, it can select an encapsulation format, that it can support, from any of the encapsulation protocols which have the bit set to 1 in this LCAF type.

Encapsulation Format Address Format:



Length: length in bytes starting and including the byte after this Length field.

Reserved-for-Future-Encapsulations: must be set to zero and ignored on receipt. This field will get bits allocated to future encapsulations, as they are created.

L: The RLOCs listed in the AFI-encoded addresses in the next longword can accept layer 3 LISP encapsulation using destination UDP port 4341 [RFC6830].

l: The RLOCs listed in the AFI-encoded addresses in the next longword can accept layer 2 LISP encapsulation using destination UDP port 8472 [I-D.smith-lisp-layer2].

V: The RLOCs listed in the AFI-encoded addresses in the next longword can accept VXLAN encapsulation using destination UDP port 4789 [RFC7348].

v: The RLOCs listed in the AFI-encoded addresses in the next longword can accept VXLAN-GPE encapsulation using destination UDP port 4790 [I-D.quinn-vxlan-gpe].

N: The RLOCs listed in the AFI-encoded addresses in the next longword can accept NV-GRE encapsulation using IPv4/ IPv6 protocol number 47 [RFC7637].

G: The RLOCs listed in the AFI-encoded addresses in the next longword can accept GENEVE encapsulation using destination UDP port 6081 [I-D.gross-geneve].

U: The RLOCs listed in the AFI-encoded addresses in the next longword can accept GUE encapsulation using destination UDP port TBD [I-D.herbert-gue].

Usage: This encoding can be used in RLOC records in Map-Requests, Map-Replies, Map-Registers, and Map-Notify messages.

6. Security Considerations

This document is classified as Experimental. The LCAF encodings defined in this document are intended to be used with their corresponding use cases and in self-contained environments. Users

should carefully consider how the [I-D.ietf-lisp-sec] threat model applies to their particular use case.

The use of the Geo-Coordinates LCAF Type may raise physical privacy issues. Care should be taken when configuring the mapping system to use specific policy parameters so geo-location information is not returned gratuitously. It is recommended that any documents that specify the use of the Geo-Coordinates LCAF Type should consider the applicability of the BCP160 [RFC6280] for location-based privacy protection.

Additional privacy concerns have arisen since publication of BCP160, and future work on LISP should examine potential threats beyond BCP160 and address improving privacy and security for LISP deployments.

7. IANA Considerations

This document defines a canonical address format encoding used in LISP control messages and in the encoding of lookup keys for the LISP Mapping Database System. Such address format is based on a fixed AFI (16387) and a LISP LCAF Type field.

The LISP LCAF Type field is an 8-bit field specific to the LISP Canonical Address formatted encodings, for which IANA is to create and maintain a new registry (as outlined in [RFC5226]) entitled "LISP LCAF Type". Initial values for the LISP LCAF Type registry are given below. Future assignments are to be made based on specification required. Assignments consist of a LISP LCAF Type name and its associated value:

Value	LISP LCAF Type Name	Definition
0	Null Body Type	Section 3
1	AFI List Type	Section 3
2	Instance ID Type	Section 3
3	AS Number Type	Section 3
5	Geo Coordinates Type	Section 3
7	NAT-Traversal Type	Section 3
9	Multicast Info Type	Section 3
10	Explicit Locator Path Type	Section 3
11	Security Key Type	Section 3
12	Source/Dest Key Type	Section 3
13	Replication List Entry Type	Section 3

Table 1: LISP LCAF Type Initial Values

8. References

8.1. Normative References

- [BCP160] "An Architecture for Location and Location Privacy in Internet Applications", Best Current Practices
<https://www.rfc-editor.org/bcp/bcp160.txt>, July 2011.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<http://www.rfc-editor.org/info/rfc1035>>.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<http://www.rfc-editor.org/info/rfc1918>>.
- [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>>.
- [RFC3232] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, DOI 10.17487/RFC3232, January 2002, <<http://www.rfc-editor.org/info/rfc3232>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<http://www.rfc-editor.org/info/rfc3986>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC6280] Barnes, R., Lepinski, M., Cooper, A., Morris, J., Tschofenig, H., and H. Schulzrinne, "An Architecture for Location and Location Privacy in Internet Applications", BCP 160, RFC 6280, DOI 10.17487/RFC6280, July 2011, <<http://www.rfc-editor.org/info/rfc6280>>.
- [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>>.

- [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>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<http://www.rfc-editor.org/info/rfc7159>>.
- [RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <<http://www.rfc-editor.org/info/rfc7348>>.
- [RFC7637] Garg, P., Ed. and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <<http://www.rfc-editor.org/info/rfc7637>>.

8.2. Informative References

- [AFI] IANA, , "Address Family Identifier (AFIs)", ADDRESS FAMILY NUMBERS <http://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml?>, February 2007.
- [I-D.coras-lisp-re] Coras, F., Cabellos-Aparicio, A., Domingo-Pascual, J., Maino, F., and D. Farinacci, "LISP Replication Engineering", draft-coras-lisp-re-08 (work in progress), November 2015.
- [I-D.ermagan-lisp-nat-traversal] Ermagan, V., Farinacci, D., Lewis, D., Skriver, J., Maino, F., and C. White, "NAT traversal for LISP", draft-ermagan-lisp-nat-traversal-11 (work in progress), August 2016.
- [I-D.farinacci-lisp-te] Farinacci, D., Kowal, M., and P. Lahiri, "LISP Traffic Engineering Use-Cases", draft-farinacci-lisp-te-11 (work in progress), September 2016.
- [I-D.gross-geneve] Gross, J., Sridhar, T., Garg, P., Wright, C., Ganga, I., Agarwal, P., Duda, K., Dutt, D., and J. Hudson, "Geneve: Generic Network Virtualization Encapsulation", draft-gross-geneve-02 (work in progress), October 2014.

- [I-D.herbert-gue]
Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", draft-herbert-gue-03 (work in progress), March 2015.
- [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-ddt]
Fuller, V., Lewis, D., Ermagan, V., Jain, A., and A. Smirnov, "LISP Delegated Database Tree", draft-ietf-lisp-ddt-08 (work in progress), September 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.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-01 (work in progress), October 2016.
- [I-D.quinn-vxlan-gpe]
Quinn, P., Manur, R., Kreeger, L., Lewis, D., Maino, F., Smith, M., Agarwal, P., Yong, L., Xu, X., Elzur, U., Garg, P., and D. Melman, "Generic Protocol Extension for VXLAN", draft-quinn-vxlan-gpe-04 (work in progress), February 2015.
- [I-D.smith-lisp-layer2]
Smith, M., Dutt, D., Farinacci, D., and F. Maino, "Layer 2 (L2) LISP Encapsulation Format", draft-smith-lisp-layer2-03 (work in progress), September 2013.
- [JSON-BINARY]
"Universal Binary JSON Specification",
URL <http://ubjson.org>.
- [WGS-84]
Geodesy and Geophysics Department, DoD., "World Geodetic System 1984", NIMA TR8350.2, January 2000, <<http://earth-info.nga.mil/GandG/publications/tr8350.2/wgs84fin.pdf>>.

Appendix A. Acknowledgments

The authors would like to thank Vince Fuller, Gregg Schudel, Jesper Skriver, Luigi Iannone, Isidor Kouvelas, and Sander Steffann for their technical and editorial commentary.

The authors would like to thank Victor Moreno for discussions that lead to the definition of the Multicast Info LCAF type.

The authors would like to thank Parantap Lahiri and Michael Kowal for discussions that lead to the definition of the Explicit Locator Path (ELP) LCAF type.

The authors would like to thank Fabio Maino and Vina Ermagan for discussions that lead to the definition of the Security Key LCAF type.

The authors would like to thank Albert Cabellos-Aparicio and Florin Coras for discussions that lead to the definition of the Replication List Entry LCAF type.

Thanks goes to Michiel Blokzijl and Alberto Rodriguez-Natal for suggesting new LCAF types.

Thanks also goes to Terry Manderson for assistance obtaining a LISP AFI value from IANA.

And finally, the authors thank Stephen Farrell (Security Area Director) and Deborah Brungard (Routing Area Director) for their suggested text to get the document through IESG review.

Appendix B. Document Change Log

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

B.1. Changes to draft-ietf-lisp-lcaf-22.txt

- o Submitted November 2016.
- o Take into account RTG area director Deborah Brungard's comments suggestions.
- o The changes put in should clear Stephen's DISCUSS comments on RLOC-record ordering and privacy concerns with the Geo-Coordinate LCAF type.

B.2. Changes to draft-ietf-lisp-lcaf-21.txt

- o Submitted November 2016.
- o Reflect Alexey's DISCUSS comments.
- o Add text to intro section that says the details for any LCAF type can be found in other use-case documents.
- o Provide general examples for JSON and DNS LCAF types.

B.3. Changes to draft-ietf-lisp-lcaf-20.txt

- o Submitted October 2016.
- o Put in references to DNS names and URIs per Alexey's comment.

B.4. Changes to draft-ietf-lisp-lcaf-19.txt

- o Submitted October 2016.
- o Make it more clear that any use-case documents that use the Geo-Coordinates LCAF type should discuss RFC6280 compliance.

B.5. Changes to draft-ietf-lisp-lcaf-18.txt

- o Submitted October 2016 after October 13th telechat.
- o Addressed comments from Ben Campbell, Jari Arrko, Stephen Farrel, Peter Yee, Dale Worley, Mirja Kuehlewind, and Suresh Krishnan.

B.6. Changes to draft-ietf-lisp-lcaf-17.txt

- o Submitted October 2016.
- o Addressed comments from Gen-ART reviewer Peter Yee.
- o Addressed IESG last-call comments from Suresh Krishnan.

B.7. Changes to draft-ietf-lisp-lcaf-16.txt

- o Submitted October 2016.
- o Addressed comments from Security Directorate reviewer David Mandelberg.

B.8. Changes to draft-ietf-lisp-lcaf-15.txt

- o Submitted September 2016.
- o Addressed comments from Routing Directorate reviewer Stig Venass.

B.9. Changes to draft-ietf-lisp-lcaf-14.txt

- o Submitted July 2016.
- o Fix IDnits errors and comments from Luigi Iannone, document shepherd.

B.10. Changes to draft-ietf-lisp-lcaf-13.txt

- o Submitted May 2016.
- o Explain the Instance-ID LCAF Type is 32-bits in length and the Instance-ID field in the LISP encapsulation header is 24-bits.

B.11. Changes to draft-ietf-lisp-lcaf-12.txt

- o Submitted March 2016.
- o Updated references and document timer.
- o Removed the R, J, and L bits from the Multicast Info Type LCAF since working group decided to not go forward with draft-farinacci-lisp-mr-signaling-03.txt in favor of draft-ietf-lisp-signal-free-00.txt.

B.12. Changes to draft-ietf-lisp-lcaf-11.txt

- o Submitted September 2015.
- o Reflecting comments from Prague LISP working group.
- o Ready document for a LISP LCAF registry, RFC publication, and for new use cases that will be defined in the new charter.

B.13. Changes to draft-ietf-lisp-lcaf-10.txt

- o Submitted June 2015.
- o Fix coauthor Job's contact information.

B.14. Changes to draft-ietf-lisp-lcaf-09.txt

- o Submitted June 2015.
- o Fix IANA Considerations section to request a registry to allocate and track LCAF Type values.

B.15. Changes to draft-ietf-lisp-lcaf-08.txt

- o Submitted April 2015.
- o Comment from Florin. The Application Data Type length field has a typo. The field should be labeled "12 + n" and not "8 + n".
- o Fix length fields in the sections titled "Using Recursive LISP Canonical Address Encodings", "Generic Database Mapping Lookups", and "Data Model Encoding".

B.16. Changes to draft-ietf-lisp-lcaf-07.txt

- o Submitted December 2014.
- o Add a new LCAF Type called "Encapsulation Format" so decapsulating xTRs can inform encapsulating xTRs what data-plane encapsulations they support.

B.17. Changes to draft-ietf-lisp-lcaf-06.txt

- o Submitted October 2014.
- o Make it clear how sorted RLOC records are done when LCAFs are used as the RLOC record.

B.18. Changes to draft-ietf-lisp-lcaf-05.txt

- o Submitted May 2014.
- o Add a length field of the JSON payload that can be used for either binary or text encoding of JSON data.

B.19. Changes to draft-ietf-lisp-lcaf-04.txt

- o Submitted January 2014.
- o Agreement among ELP implementors to have the AFI 16-bit field adjacent to the address. This will make the encoding consistent with all other LCAF type address encodings.

B.20. Changes to draft-ietf-lisp-lcaf-03.txt

- o Submitted September 2013.
- o Updated references and author's affiliations.
- o Added Instance-ID to the Multicast Info Type so there is relative ease in parsing (S,G) entries within a VPN.
- o Add port range encodings to the Application Data LCAF Type.
- o Add a new JSON LCAF Type.
- o Add Address Key/Value LCAF Type to allow attributes to be attached to an address.

B.21. Changes to draft-ietf-lisp-lcaf-02.txt

- o Submitted March 2013.
- o Added new LCAF Type "Replication List Entry" to support LISP replication engineering use cases.
- o Changed references to new LISP RFCs.

B.22. Changes to draft-ietf-lisp-lcaf-01.txt

- o Submitted January 2013.
- o Change longitude range from 0-90 to 0-180 in section 4.4.
- o Added reference to WGS-84 in section 4.4.

B.23. Changes to draft-ietf-lisp-lcaf-00.txt

- o Posted first working group draft August 2012.
- o This draft was renamed from draft-farinacci-lisp-lcaf-10.txt.

Authors' Addresses

Dino Farinacci
lispers.net
San Jose, CA
USA

Email: farinacci@gmail.com

Dave Meyer
Brocade
San Jose, CA
USA

Email: dmm@1-4-5.net

Job Snijders
NTT Communications
Theodorus Majofskistraat 100
Amsterdam 1065 SZ
The Netherlands

Email: job@ntt.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 1, 2016

D. Saucez
INRIA
L. Iannone
Telecom ParisTech
O. Bonaventure
Universite catholique de Louvain
January 29, 2016

LISP Threats Analysis
draft-ietf-lisp-threats-15.txt

Abstract

This document provides a threat analysis of the Locator/Identifier 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 August 1, 2016.

Copyright Notice

Copyright (c) 2016 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	3
2. Threat model	3
2.1. Attacker's Operation Modes	4
2.1.1. On-path vs. Off-path Attackers	4
2.1.2. Internal vs. External Attackers	4
2.1.3. Live vs. Time-shifted attackers	5
2.1.4. Control-plane vs. Data-plane attackers	5
2.1.5. Cross mode attackers	5
2.2. Threat categories	5
2.2.1. Replay attack	5
2.2.2. Packet manipulation	6
2.2.3. Packet interception and suppression	6
2.2.4. Spoofing	6
2.2.5. Rogue attack	7
2.2.6. Denial of Service (DoS) attack	7
2.2.7. Performance attack	7
2.2.8. Intrusion attack	7
2.2.9. Amplification attack	7
2.2.10. Passive Monitoring Attacks	8
2.2.11. Multi-category attacks	8
3. Attack vectors	8
3.1. Gleaning	8
3.2. Locator Status Bits	9
3.3. Map-Version	10
3.4. Routing Locator Reachability	11
3.5. Instance ID	12
3.6. Interworking	12
3.7. Map-Request messages	12
3.8. Map-Reply messages	13
3.9. Map-Register messages	15
3.10. Map-Notify messages	15
4. Note on Privacy	15
5. Threats Mitigation	16
6. Security Considerations	17
7. IANA Considerations	17
8. Acknowledgments	17
9. References	17
9.1. Normative References	17
9.2. Informative References	18
Appendix A. Document Change Log (to be removed on publication) .	19
Authors' Addresses	21

1. Introduction

The Locator/ID Separation Protocol (LISP) is specified in [RFC6830]. This document provides an assessment of the potential security threats for the current LISP specifications if LISP is deployed in the Internet (i.e., a public non-trustable environment).

The document is composed of three main parts: the first defines a general threat model that attackers use to mount attacks. The second part, using this threat model, describes the techniques based on the LISP protocol and LISP architecture that attackers may use to construct attacks. The third part discusses mitigation techniques and general solutions to protect the LISP protocol and architecture from attacks.

This document does not consider all the possible uses of LISP as discussed in [RFC6830] and [RFC7215] and does not cover threats due to specific implementations. The document focuses on LISP unicast, including as well LISP Interworking [RFC6832], LISP Map-Server [RFC6833]), and LISP Map-Versioning [RFC6834]. Additional threats may be discovered in the future while deployment continues. The reader is assumed to be familiar with these documents for understanding the present document.

This document assumes a generic IP service and does not discuss the difference, from a security viewpoint, between using IPv4 or IPv6.

2. Threat model

This document assumes that attackers can be located anywhere in the Internet (either in LISP sites or outside LISP sites) and that attacks can be mounted either by a single attacker or by the collusion of several attackers.

An attacker is a malicious entity that performs the action of attacking a target in a network where LISP is (partially) deployed by leveraging the LISP protocol and/or architecture.

An attack is the action of performing an illegitimate action on a target in a network where LISP is (partially) deployed.

The target of an attack is the entity (i.e., a device connected to the network or a network) that is aimed to undergo the consequences of an attack. Other entities can potentially undergo side effects of an attack, even though they are not directly targeted by the attack. The target of an attack can be selected specifically, i.e., a particular entity, or arbitrarily, i.e., any entity. Finally, an

attacker can aim at attacking one or several targets with a single attack.

Section 2.1 specifies the different modes of operation that attackers can follow to mount attacks and Section 2.2 specifies the different categories of attacks that attackers can build.

2.1. Attacker's Operation Modes

In this document attackers are classified according to their modes of operation, i.e., the temporal and spacial diversity of the attacker. These modes are not mutually exclusive, they can be used by attackers in any combination, and other modes may be discovered in the future. Further, attackers are not at all bound by our classification scheme, so implementers and those deploying will always need to do additional risk analysis for themselves.

2.1.1. On-path vs. Off-path Attackers

On-path attackers, also known as Men-in-the-Middle, are able to intercept and modify packets between legitimate communicating entities. On-path attackers are located either directly on the normal communication path (either by gaining access to a node on the path or by placing themselves directly on the path) or outside the location path but manage to deviate (or gain a copy of) packets sent between the communication entities. On-path attackers hence mount their attacks by modifying packets initially sent legitimately between communication entities.

An attacker is called off-path attacker if it does not have access to packets exchanged during the communication or if there is no communication. In order for their attacks to succeed, off-path attackers must hence generate packets and inject them in the network.

2.1.2. Internal vs. External Attackers

An internal attacker launches its attack from a node located within a legitimate LISP site. Such an attacker is either a legitimate node of the site or it exploits a vulnerability to gain access to a legitimate node in the site. Because of their location, internal attackers are trusted by the site they are in.

On the contrary, an external attacker launches its attacks from the outside of a legitimate LISP site.

2.1.3. Live vs. Time-shifted attackers

A live attacker mounts attacks for which it must remain connected as long as the attack is mounted. In other words, the attacker must remain active for the whole duration of the attack. Consequently, the attack ends as soon as the attacker (or the used attack vector) is neutralized.

On the contrary, a time-shifted attacker mounts attacks that remain active after it disconnects from the Internet.

2.1.4. Control-plane vs. Data-plane attackers

A control-plane attacker mounts its attack by using control-plane functionalities, typically the mapping system.

A data-plane attacker mounts its attack by using data-plane functionalities.

As there is no complete isolation between the control-plane and the data-plane, an attacker can operate in the control-plane (or data-plane) to mount attacks targeting the data-plane (or control-plane) or keep the attacked and targeted planes at the same layer (i.e., from control-plane to control-plane or from data-plane to data-plane).

2.1.5. Cross mode attackers

The attacker modes of operation are not mutually exclusive and hence attackers can combine them to mount attacks.

For example, an attacker can launch an attack using the control-plane directly from within a LISP site to which it is able to get temporary access (i.e., internal + control-plane attacker) to create a vulnerability on its target and later on (i.e., time-shifted + external attacker) mount an attack on the data plane (i.e., data-plane attacker) that leverages the vulnerability.

2.2. Threat categories

Attacks can be classified according to the nine following categories. These categories are not mutually exclusive and can be used by attackers in any combination.

2.2.1. Replay attack

A replay attack happens when an attacker retransmits at a later time, and without modifying it, a packet (or a sequence of packets) that

has already been transmitted.

2.2.2. Packet manipulation

A packet manipulation attack happens when an attacker receives a packet, modifies the packet (i.e., changes some information contained in the packet) and finally transmits the packet to its final destination that can be the initial destination of the packet or a different one.

2.2.3. Packet interception and suppression

In a packet interception and suppression attack, the attacker captures the packet and drops it before it can reach its final destination.

2.2.4. Spoofing

With a spoofing attack, the attacker injects packets in the network pretending to be another node. Spoofing attacks are made by forging source addresses in packets.

It should be noted that with LISP, packet spoofing is similar to spoofing with any other existing tunneling technology currently deployed in the Internet. Generally the term "spoofed packet" indicates a packet containing a source IP address that is not the actual originator of the packet. Hence, since LISP uses encapsulation, the spoofed address could be in the outer header as well as in the inner header, this translates to two types of spoofing.

Inner address spoofing: the attacker uses encapsulation and uses a spoofed source address in the inner packet. In case of data-plane LISP encapsulation, that corresponds to spoofing the source EID (End-point IDentifier) address of the encapsulated packet.

Outer address spoofing: the attacker does not use encapsulation and spoofs the source address of the packet. In case of data-plane LISP encapsulation, that corresponds to spoofing the source RLOC (Routing LOCator) address of the encapsulated packet.

Note that the two types of spoofing are not mutually exclusive, rather all combinations are possible and could be used to perform different kinds of attacks. For example, an attacker outside a LISP site can generate a packet with a forged source IP address (i.e., outer address spoofing) and forward it to a LISP destination. The packet is then eventually encapsulated by a PITR (Proxy Ingress

Tunnel Router) so that once encapsulated the attack corresponds to a inner address spoofing. One can also imagine an attacker forging a packet with encapsulation where both inner and outer source addresses are spoofed.

It is important to note that the combination of inner and outer spoofing makes the identification of the attacker complex as the packet may not contain information that allows to detect the origin of the attack.

2.2.5. Rogue attack

In a rogue attack the attacker manages to appear as a legitimate source of information, without faking its identity (as opposed to a spoofing attacker).

2.2.6. Denial of Service (DoS) attack

A Denial of Service (DoS) attack aims at disrupting a specific targeted service to make it unable to operate properly.

2.2.7. Performance attack

A performance attacks aims at exploiting computational resources (e.g., memory, processor) of a targeted node so as to make it unable to operate properly.

2.2.8. Intrusion attack

In an intrusion attack, the attacker gains remote access to a resource (e.g., a host, a router, or a network) or information that it legitimately should not have access. Intrusion attacks can lead to privacy leakages.

2.2.9. Amplification attack

In an amplification attack, the traffic generated by the target of the attack in response to the attack is larger than the traffic that the attacker must generate.

In some cases, the data-plane can be several orders of magnitude faster than the control-plane at processing packets. This difference can be exploited to overload the control-plane via the data-plane without overloading the data-plane.

2.2.10. Passive Monitoring Attacks

An attacker can use pervasive monitoring, which is a technical attack [RFC7258], targeting information about LISP traffic that may or not be used to mount other type of attacks.

2.2.11. Multi-category attacks

Attacks categories are not mutually exclusive and any combination can be used to perform specific attacks.

For example, one can mount a rogue attack to perform a performance attack starving the memory of an ITR (Ingress Tunnel Router) resulting in a DoS (Denial-of-Service) on the ITR.

3. Attack vectors

This section presents attack techniques that may be used by attackers when leveraging the LISP protocol and/or architecture.

3.1. Gleaning

To reduce the time required to obtain a mapping, the optional gleaning mechanism defined for LISP allows an xTR (Ingress and/or Egress Tunnel Router) to directly learn a mapping from the LISP data encapsulated packets and the Map-Request packets that it receives. LISP encapsulated data packets contain a source RLOC, destination RLOC, source EID and destination EID. When an xTR receives an encapsulated data packet coming from a source EID for which it does not already know a mapping, it may insert the mapping between the source RLOC and the source EID in its EID-to-RLOC Cache. The same technique can be used when an xTR receives a Map-Request as the Map-Request also contains a source EID address and a source RLOC. Once a gleaned entry has been added to the EID-to-RLOC cache, the xTR sends a Map-Request to retrieve the actual mapping for the gleaned EID from the mapping system.

If a packet injected by an off-path attacker and with a spoofed inner address is gleaned by an xTR then the attacker may divert the traffic meant to be delivered to the spoofed EID as long as the gleaned entry is used by the xTR. This attack can be used as part of replay, packet manipulation, packet interception and suppression, or DoS attacks as the packets are sent to the attacker.

If the packet sent by the attacker contains a spoofed outer address instead of a spoofed inner address then it can achieve a DoS or a performance attack as the traffic normally destined to the attacker

will be redirected to the spoofed source RLOC. Such traffic may overload the owner of the spoofed source RLOC, preventing it from operating properly.

If the packet injected uses both inner and outer spoofing, the attacker can achieve a spoofing, a performance, or an amplification attack as traffic normally destined to the spoofed EID address will be sent to the spoofed RLOC address. If the attacked LISP site also generates traffic to the spoofed EID address, such traffic may have a positive amplification factor.

A gleaning attack does not only impact the data-plane but can also have repercussions on the control-plane as a Map-Request is sent after the creation of a gleaned entry. The attacker can then achieve DoS and performance attacks on the control-plane. For example, if an attacker sends a packet for each address of a prefix not yet cached in the EID-to-RLOC cache of an xTR, the xTR will potentially send a Map-Request for each such packet until the mapping is installed which leads to an over-utilisation of the control-plane as each packet generates a control-plane event. In order for this attack to succeed, the attacker may not need to use spoofing. This issue can occur even if gleaning is turned off since whether or not gleaning is used as the ITR may need to send a Map-Request in response to incoming packets whose EID is not currently in the cache.

Gleaning attacks are fundamentally involving a time-shifted mode of operation as the attack may last as long as the gleaned entry is kept by the targeted xTR. RFC 6830 [RFC6830] recommends to store the gleaned entries for only a few seconds which limits the duration of the attack.

Gleaning attacks always involve external data-plane attackers but results in attacks on either the control-plane or data-plane.

Note, the outer spoofed address does not need to be the RLOC of a LISP site, it may be any address.

3.2. Locator Status Bits

When the L bit in the LISP header is set to 1, it indicates that the second 32-bits longword of the LISP header contains the Locator Status Bits. In this field, each bit position reflects the status of one of the RLOCs mapped to the source EID found in the encapsulated packet. The reaction of a LISP xTR that receives such a packet is left as operational choice in [RFC6830].

When an attacker sends a LISP encapsulated packet with an illegitimately crafted LSB to an xTR, it can influence the xTR's

choice of the locators for the prefix associated to the source EID. In case of an off-path attacker, the attacker must inject a forged packet in the network with a spoofed inner address. An on-path attacker can manipulate the LSB of legitimate packets passing through it and hence does not need to use spoofing. Instead of manipulating the LSB field, an on-path attacker can also obtain the same result of injecting packets with invalid LSB values by replaying packets.

The LSB field can be leveraged to mount a DoS attack by either declaring all RLOCs as unreachable (all LSB set to 0), or by concentrating all the traffic to one RLOC (e.g., all but one LSB set to 0) and hence overloading the RLOC concentrating all the traffic from the xTR, or by forcing packets to be sent to RLOCs that are actually not reachable (e.g., invert LSB values).

The LSB field can also be used to mount a replay, a packet manipulation, or a packet interception and suppression attack. Indeed, if the attacker manages to be on the path between the xTR and one of the RLOCs specified in the mapping, forcing packets to go via that RLOC implies that the attacker will gain access to the packets.

Attacks using the LSB are fundamentally involving a time-shifted mode of operation as the attack may last as long as the reachability information gathered from the LSB is used by the xTR to decide the RLOCs to be used.

3.3. Map-Version

When the Map-Version bit of the LISP header is set to 1, it indicates that the low-order 24 bits of the first 32 bits longword of the LISP header contain a Source and Destination Map-Version. When a LISP xTR receives a LISP encapsulated packet with the Map-Version bit set to 1, the following actions are taken:

- o It compares the Destination Map-Version found in the header with the current version of its own configured EID-to-RLOC mapping, for the destination EID found in the encapsulated packet. If the received Destination Map-Version is smaller (i.e., older) than the current version, the ETR should apply the SMR (Solicit-Map-Request) procedure described in [RFC6830] and send a Map-Request with the SMR bit set.
- o If a mapping exists in the EID-to-RLOC Cache for the source EID, then it compares the Map-Version of that entry with the Source Map-Version found in the header of the packet. If the stored mapping is older (i.e., the Map-Version is smaller) than the source version of the LISP encapsulated packet, the xTR should send a Map-Request for the source EID.

A cross-mode attacker can use the Map-Version bit to mount a DoS attack, an amplification attack, or a spoofing attack. For instance if the mapping cached at the xTR is outdated, the xTR will send a Map-Request to retrieve the new mapping which can yield to a DoS attack (by excess of signalling traffic) or an amplification attack if the data-plane packet sent by the attacker is smaller, or otherwise uses fewer resources, than the control-plane packets sent in response to the attacker's packet. With a spoofing attack, and if the xTR considers that the spoofed ITR has an outdated mapping, it will send an SMR to the spoofed ITR which can result in performance, amplification, or DoS attack as well.

Map-Version attackers are inherently cross mode as the Map-Version is a method to put control information in the data-plane. Moreover, this vector involves live attackers. Nevertheless, on-path attackers do not have specific advantage over off-path attackers.

3.4. Routing Locator Reachability

The Nonce-Present and Echo-Nonce bits in the LISP header are used to verify the reachability of an xTR. A testing xTR sets the Echo-Nonce and the Nonce-Present bits in LISP data encapsulated packets and include a random nonce in the LISP header of packets. Upon reception of these packets, the tested xTR stores the nonce and echoes it whenever it returns a LISP encapsulated data packets to the testing xTR. The reception of the echoed nonce confirms that the tested xTR is reachable.

An attacker can interfere with the reachability test by sending two different types of packets:

1. LISP data encapsulated packets with the Nonce-Present bit set and a random nonce. Such packets are normally used in response to a reachability test.
2. LISP data encapsulated packets with the Nonce-Present and the Echo-Nonce bits both set. These packets will force the receiving ETR to store the received nonce and echo it in the LISP encapsulated packets that it sends. These packets are normally used as a trigger for a reachability test.

The first type of packets are used to make xTRs think that an other xTR is reachable while it is not. It is hence a way to mount a DoS attack (i.e., the ITR will send its packet to a non-reachable ETR when it should use another one).

The second type of packets could be exploited to attack the nonce-based reachability test. If the attacker sends a continuous flow of

packets that each have a different random nonce, the ETR that receives such packets will continuously change the nonce that it returns to the remote ITR, which can yield to a performance attack. If the remote ITR tries a nonce-reachability test, this test may fail because the ETR may echo an invalid nonce. This hence yields to a DoS attack.

In the case of an on-path attacker, a packet manipulation attack is necessary to mount the attack. To mount such an attack, an off-path attacker must mount an outer address spoofing attack.

If an xTR chooses to periodically check with active probes the liveness of entries in its EID-to-RLOC cache (as described in section 6.3 of [RFC6830]), then this may amplify the attack that caused the insertion of entries being checked.

3.5. Instance ID

LISP allows to carry in its header a 24-bits value called Instance ID and used on the ITR to indicate which local Instance ID has been used for encapsulation, while on the ETR the instance ID decides the forwarding table to use to forward the decapsulated packet in the LISP site.

An attacker (either a control-plane or data-plane attacker) can use the instance ID functionality to mount an intrusion attack.

3.6. Interworking

[RFC6832] defines Proxy-ITR and Proxy-ETR network elements to allow LISP and non-LISP sites to communicate. The Proxy-ITR has functionality similar to the ITR, however, its main purpose is to encapsulate packets arriving from the DFZ (Default-Free Zone) in order to reach LISP sites. A PETR (Proxy Egress Tunnel Router) has functionality similar to the ETR, however, its main purpose is to inject de-encapsulated packets in the DFZ in order to reach non-LISP sites from LISP sites. As a PITR (or PETR) is a particular case of ITR (or ETR), it is subject to similar attacks as ITRs (or ETRs).

As any other system relying on proxies, LISP interworking can be used by attackers to hide their exact origin in the network.

3.7. Map-Request messages

A control-plane off-path attacker can exploit Map-Request messages to mount DoS, performance, or amplification attacks. By sending Map-Request messages at high rate, the attacker can overload nodes involved in the mapping system. For instance sending Map-Requests at

high rate can considerably increase the state maintained in a Map-Resolver or consume CPU cycles on ETRs that have to process the Map-Request packets they receive in their slow path (i.e., performance or DoS attack). When the Map-Reply packet is larger than the Map-Request sent by the attacker, that yields to an amplification attack. The attacker can combine the attack with a spoofing attack to overload the node to which the spoofed address is actually attached.

Note, if the attacker sets the P bit (Probe Bit) in the Map-Request, it will cause legitimately sending the Map-Request directly to the ETR instead of passing through the mapping system.

The SMR bit can be used to mount a variant of these attacks.

For efficiency reasons, Map-Records can be appended to Map-Request messages. When an xTR receives a Map-Request with appended Map-Records, it does the same operations as for the other Map-Request messages and so is subject to the same attacks. However, it also installs in its EID-to-RLOC cache the Map-Records contained in the Map-Request. An attacker can then use this vector to force the installation of mappings in its target xTR. Consequently, the EID-to-RLOC cache of the xTR is polluted by potentially forged mappings allowing the attacker to mount any of the attacks categorized in Section 2.2 (see Section 3.8 for more details). Note, the attacker does not need to forge the mappings present in the Map-Request to achieve a performance or DoS attack. Indeed, if the attacker owns a large enough EID prefix it can de-aggregate it in many small prefixes, each corresponding to another mapping and it installs them in the xTR cache by mean of the Map-Request.

Moreover, attackers can use Map Resolver and/or Map Server network elements to relay its attacks and hide the origin of the attack. Indeed, on the one hand, a Map Resolver is used to dispatch Map-Request to the mapping system and, on the other hand, a Map Server is used to dispatch Map-Requests coming from the mapping system to ETRs that are authoritative for the EID in the Map-Request.

3.8. Map-Reply messages

Most of the security risks associated with Map-Reply messages will depend on the 64 bits nonce that is included in a Map-Request and returned in the Map-Reply. Given the size of the nonce (64 bits), if best current practice is used [RFC4086] and if an ETR does not accept Map-Reply messages with an invalid nonce, the risk of an off-path attack is limited. Nevertheless, the nonce only confirms that the Map-Reply received was sent in response to a Map-Request sent, it does not validate the contents of that Map-Reply.

If an attacker manages to send a valid (i.e., in response to a Map-Request and with the correct nonce) Map-Reply to an ITR, then it can perform any of the attacks categorised in Section 2.2 as it can inject forged mappings directly in the ITR EID-to-RLOC cache. For instance, if the mapping injected to the ITR points to the address of a node controlled by the attacker, it can mount replay, packet manipulation, packet interception and suppression, or DoS attacks, as it will receive every packet destined to a destination lying in the EID prefix of the injected mapping. In addition, the attacker can inject a plethora of mappings in the ITR to mount a performance attack by filling up the EID-to-RLOC cache of the ITR. The attacker can also mount an amplification attack if the ITR at that time is sending a large number of packets to the EIDs matching the injected mapping. In this case, the RLOC address associated to the mapping is the address of the real target of the attacker and so all the traffic of the ITR will be sent to the target which means that with one single packet the attacker may generate very high traffic towards its final target.

If the attacker is a valid ETR in the system, it can mount a rogue attack if it uses prefixes over-claiming. In such a scenario, the attacker ETR replies to a legitimate Map-Request message which it received with a Map-Reply message that contains an EID-Prefix that is larger than the prefix owned by the attacker. For example if the owned prefix is 192.0.2.0/25 but the Map-Reply contains a mapping for 192.0.2.0/24, then the mapping will influence packets destined to other EIDs than the one attacker has authority on. With such technique, the attacker can mount the attacks presented above as it can (partially) control the mappings installed on its target ITR. To force its target ITR to send a Map-Request, nothing prevents the attacker to initiate some communication with the ITR. This method can be used by internal attackers that want to control the mappings installed in their site. To that aim, they simply have to collude with an external attacker ready to over-claim prefixes on behalf of the internal attacker.

Note, when the Map-Reply is in response to a Map-Request sent via the mapping system (i.e., not send directly from the ITR to an ETR), the attacker does not need to use a spoofing attack to achieve its attack as by design the source IP address of a Map-Reply is not known in advance by the ITR.

Map-Request and Map-Reply messages are exposed to any type of attackers, on-path or off-path but also external or internal attackers. Also, even though they are control message, they can be leveraged by data-plane attackers. As the decision of removing mappings is based on the TTL indicated in the mapping, time-shifted attackers can take advantage of injecting forged mappings as well.

3.9. Map-Register messages

Map-Register messages are sent by ETRs to Map Servers to indicate to the mapping system the EID prefixes associated to them. The Map-Register message provides an EID prefix and the list of ETRs that are able to provide Map-Replies for the EID covered by the EID prefix.

As Map-Register messages are protected by an authentication mechanism, only a compromised ETR can register itself to its allocated Map Server.

A compromised ETR can over-claim the prefix it owns in order to influence the route followed by Map-Requests for EIDs outside the scope of its legitimate EID prefix (see Section 3.8 for the list of over-claiming attacks).

A compromised ETR can also de-aggregate its EID prefix in order to register more EID prefixes than necessary to its Map Servers (see Section 3.7 for the impact of de-aggregation of prefixes by an attacker).

Similarly, a compromised Map Server can accept an invalid registration or advertise an invalid EID prefix to the mapping system.

3.10. Map-Notify messages

Map-Notify messages are sent by a Map Server to an ETR to acknowledge the reception and processing of a Map-Register message.

Similarly to the pair Map-Request/Map-Reply, the pair Map-Register/Map-Notify is protected by a nonce making it difficult for an attacker to inject a falsified notification to an ETR to make this ETR believe that the registration succeeded when it has not.

4. Note on Privacy

As reviewed in [RFC6973], universal privacy considerations are difficult to establish as the privacy definitions may vary for different scenarios. As a consequence, this document does not aim at identifying privacy issues related to the LISP protocol but the security threats identified in this document could play a role in privacy threats as defined in section 5 of [RFC6973].

Similar to public deployments of any other control plane protocols, in an Internet deployment, LISP mappings are public and hence provide information about the infrastructure and reachability of LISP sites

(i.e., the addresses of the edge routers). Depending upon deployment details, LISP map replies might or might not provide finer grained and more detailed information than is available with currently deployed routing and control protocols.

5. Threats Mitigation

Most of the above threats can be mitigated with careful deployment and configuration (e.g., filter) and also by applying the general rules of security, e.g. only activating features that are necessary for the deployment and verifying the validity of the information obtained from third parties.

The control-plane is the most critical part of LISP from a security viewpoint and it is worth to notice that the LISP specifications already offer an authentication mechanism for mappings registration ([RFC6833]). This mechanism, combined with LISP-SEC [I-D.ietf-lisp-sec], strongly mitigates threats in non-trustable environments such as the Internet. Moreover, an authentication data field for Map-Request messages and Encapsulated Control messages was allocated [RFC6830]. This field provides a general authentication mechanism technique for the LISP control-plane which future specifications may use while staying backward compatible. The exact technique still has to be designed and defined. To maximally mitigate the threats on the mapping system, authentication must be used, whenever possible, for both Map-Request and Map-Reply messages and for messages exchanged internally among elements of the mapping system, such as specified in [I-D.ietf-lisp-sec] and [I-D.ietf-lisp-ddt].

Systematically applying filters and rate-limitation, as proposed in [RFC6830], will mitigate most of the threats presented in this document. In order to minimise the risk of overloading the control-plane with actions triggered from data-plane events, such actions should be rate limited.

Moreover, all information opportunistically learned (e.g., with LSB or gleaning) should be used with care until they are verified. For example, a reachability change learned with LSB should not be used directly to decide the destination RLOC, but instead should trigger a rate-limited reachability test. Similarly, a gleaned entry should be used only for the flow that triggered the gleaning procedure until the gleaned entry has been verified [Trilogy].

6. Security Considerations

This document provides a threat analysis and proposes mitigation techniques for the Locator/Identifier Separation Protocol.

7. IANA Considerations

This document makes no request to IANA.

8. Acknowledgments

This document builds upon the document of Marcelo Bagnulo ([I-D.bagnulo-lisp-threat]), where the flooding attack and the reference environment was first described.

The authors would like to thank Deborah Brungard, Ronald Bonica, Albert Cabellos, Ross Callon, Noel Chiappa, Florin Coras, Vina Ermagan, Dino Farinacci, Stephen Farrell, Joel Halpern, Emily Hiltzik, Darrel Lewis, Edward Lopez, Fabio Maino, Terry Manderson, and Jeff Wheeler for their comments.

This work has been partially supported by the INFISO-ICT-216372 TRILOGY Project (www.trilogy-project.org).

The work of Luigi Iannone has been partially supported by the ANR-13-INFR-0009 LISP-Lab Project (www.lisp-lab.org) and the EIT KIC ICT-Labs SOFNETS Project.

9. References

9.1. Normative References

- [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>>.

[RFC6834] Iannone, L., Saucez, D., and O. Bonaventure, "Locator/ID Separation Protocol (LISP) Map-Versioning", RFC 6834, DOI 10.17487/RFC6834, January 2013, <<http://www.rfc-editor.org/info/rfc6834>>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<http://www.rfc-editor.org/info/rfc6973>>.

9.2. Informative References

- [I-D.bagnulo-lisp-threat]
Bagnulo, M., "Preliminary LISP Threat Analysis", draft-bagnulo-lisp-threat-01 (work in progress), July 2007.
- [I-D.ietf-lisp-ddt]
Fuller, V., Lewis, D., Ermagan, V., and A. Jain, "LISP Delegated Database Tree", draft-ietf-lisp-ddt-03 (work in progress), April 2015.
- [I-D.ietf-lisp-sec]
Maino, F., Ermagan, V., Cabellos-Aparicio, A., and D. Saucez, "LISP-Security (LISP-SEC)", draft-ietf-lisp-sec-09 (work in progress), October 2015.
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <<http://www.rfc-editor.org/info/rfc4086>>.
- [RFC7215] Jakab, L., Cabellos-Aparicio, A., Coras, F., Domingo-Pascual, J., and D. Lewis, "Locator/Identifier Separation Protocol (LISP) Network Element Deployment Considerations", RFC 7215, DOI 10.17487/RFC7215, April 2014, <<http://www.rfc-editor.org/info/rfc7215>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<http://www.rfc-editor.org/info/rfc7258>>.
- [Trilogy] Saucez, D. and L. Iannone, "How to mitigate the effect of scans on mapping systems", Trilogy Future Internet Summer

School., 2009.

Appendix A. Document Change Log (to be removed on publication)

- o Version 15 Posted January 2016.
 - * Few changes to address Stephen Farrel comments as part of the IESG Review.
- o Version 14 Posted December 2015.
 - * Editorial changes according to Deborah Brungard's (Routing AD) review.
- o Version 13 Posted August 2015.
 - * Keepalive version.
- o Version 12 Posted March 2015.
 - * Addressed comments by Ross Callon on the mailing list (<http://www.ietf.org/mail-archive/web/lisp/current/msg05829.html>).
 - * Addition of a section discussing mitigation techniques for deployments in non-trustable environments.
- o Version 11 Posted December 2014.
 - * Editorial polishing. Clarifications added in few points.
- o Version 10 Posted July 2014.
 - * Document completely remodelled according to the discussions on the mailing list in the thread <http://www.ietf.org/mail-archive/web/lisp/current/msg05206.html> and to address comments from Ronald Bonica and Ross Callon.
- o Version 09 Posted March 2014.
 - * Updated document according to the review of A. Cabellos.
- o Version 08 Posted October 2013.
 - * Addition of a privacy consideration note.
 - * Editorial changes

- o Version 07 Posted October 2013.
 - * This version is updated according to the thorough review made during October 2013 LISP WG interim meeting.
 - * Brief recommendations put in the security consideration section.
 - * Editorial changes
- o Version 06 Posted October 2013.
 - * Complete restructuration, temporary version to be used at October 2013 interim meeting.
- o Version 05 Posted August 2013.
 - * Removal of severity levels to become a short recommendation to reduce the risk of the discussed threat.
- o Version 04 Posted February 2013.
 - * Clear statement that the document compares threats of public LISP deployments with threats in the current Internet architecture.
 - * Addition of a severity level discussion at the end of each section.
 - * Addressed comments from V. Ermagan and D. Lewis' reviews.
 - * Updated References.
 - * Further editorial polishing.
- o Version 03 Posted October 2012.
 - * Dropped Reference to RFC 2119 notation because it is not actually used in the document.
 - * Deleted future plans section.
 - * Updated References
 - * Deleted/Modified sentences referring to the early status of the LISP WG and documents at the time of writing early versions of the document.

- * Further editorial polishing.
- * Fixed all ID nits.
- o Version 02 Posted September 2012.
 - * Added a new attack that combines over-claiming and de-aggregation (see Section 3.8).
 - * Editorial polishing.
- o Version 01 Posted February 2012.
 - * Added discussion on LISP-DDT.
- o Version 00 Posted July 2011.
 - * Added discussion on LISP-MS>.
 - * Added discussion on Instance ID.
 - * Editorial polishing of the whole document.
 - * Added "Change Log" appendix to keep track of main changes.
 - * Renamed "draft-saucez-lisp-security-03.txt".

Authors' Addresses

Damien Saucez
INRIA
2004 route des Lucioles BP 93
06902 Sophia Antipolis Cedex
France

Email: damien.saucez@inria.fr

Luigi Iannone
Telecom ParisTech
23, Avenue d'Italie, CS 51327
75214 PARIS Cedex 13
France

Email: ggx@gigix.net

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

Email: olivier.bonaventure@uclouvain.be

