LISP Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 16, 2014

L. Cheng
M. Sun
ZTE Corporation

July 15, 2013

# draft-cheng-lisp-shdht-04
# LISP Single-Hop DHT Mapping Overlay

## Abstract

This draft specifies the LISP Single-Hop Distributed Hash Table Mapping Database (LISP-SHDHT), a distributed mapping database which consists of a set of SHDHT Nodes to provide mappings from LISP Endpoint Identifiers (EIDs) to Routing Locators (RLOCs). EID namespace is dynamically distributed among SHDHT Nodes based on DHT Hash algorithm. Each SHDHT Node is configured with one or more hash spaces which contain multiple EID-prefixes along with RLOCs of corresponding Map Servers.

## Status of This Memo

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

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

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

This Internet-Draft will expire on January 16, 2014.

Table of Contents



## 1. Introduction

Locator/ID Separation Protocol (LISP) [RFC6830] specifies an architecture and mechanism for replacing the address currently used by IP with two separate name spaces: Endpoint IDs (EIDs), used within LISP sites, and Routing Locators (RLOCs), used on transit networks that make up the Internet infrastructure. To achieve this separation, LISP defines protocol mechanisms for mapping from EIDs to RLOCs. As a result, an efficient database is needed to store and

propagate those mappings globally. Several such mapping databases have been proposed, among them: LISP-NERD [I-D.lear-lisp-nerd], LISP-ALT[RFC6836], LISP-DDT[I-ietf-lisp-ddt], and LISP-DHT [LISP-DHT].





According to databases such like LISP-ALT [RFC6836] and LISP-DDT [I-ietf-lisp-ddt], architectures of these mapping databases are based on announcement/delegation of hierarchically-delegated segments of EID namespace (i.e., prefixes). Therefore, based on these architectures, when a roaming event occurs and a LISP site or a LISP MN receives new RLOCs, the site or MN has to anchor pre-configured map-server to register its new mapping information no matter where the site or MN currently locates, just in order to protect EID prefixes announced aggregately in the database [I-D.meyer-lisp-mn].

As a DHT strategy based mapping database, LISP-DHT [LISP-DHT] exhibits several interesting properties, such as self-configuration, self-maintenance, scalability and robustness that are clearly desirable for a EID-to-RLOC resolution service. However, this database is based on multi-hop Chord DHT. On one hand, inquiries of mapping information in this case need to pass through iterative multi-hop lookup steps which will cause relatively large delay time. On the other hand, load balance between Chord nodes is another essential problem need to be solved.

This draft specifies a LISP Single-Hop Distributed Hash Table Mapping Overlay (LISP-SHDHT) which provides mapping information lookup service for sites running LISP. Main characters of this strategy is that,

1. Each SHDHT Node maintains routing information for all other SHDHT Nodes. Thus, messages interaction between SHDHT Nodes in the same SHDHT overlay just need one hop.

2. Traditionally, Node IDs are used to identify DHT nodes and represent hash space arrangement on DHT nodes. In SHDHT strategy, the two roles are separated. Partition IDs are adopted for hash space arrangement and a build-in load balancing solution is designed.

This draft specifies the outline of SHDHT and the basic application

of LISP SHDHT. In actual deployment of LISP SHDHT, mapping database could be maintained by multiple service providers and could be deployed as collaborative combination of multiple Domain LISP SHDHTs. Details about Domain LISP SHDHT Deployment are introduced in Section 5.

In main context of this draft, SHDHT Mapping database is proposed according to structure requirements of LISP-MS [RFC6833]. This SHDHT strategy provides services for LISP Sites mapping lookup. In Appendix A, a special SHDHT strategy for LISP-MN [I-D.meyer-lisp-mn] scenario is proposed. This SHDHT strategy is not completely match structure requirements of LISP-MS. However, it could provide better





service for LISP-MN scenario, where LISP-MN need not to anchor pre-configured Map Server and could update its mapping information as soon as possible when it roams to a new location.

## 2. Definition of Terms

This draft uses terms defined in [RFC6830]. This section defines some new terms used in this document.

SHDHT: Single-Hop Distributed Hash Table Mapping Overlay.

SHDHT Node: Physical nodes which compose SHDHT overlay's topology. Each SHDHT Node has a unique Node ID and maintains multiple hash space segments which labeled by Partition IDs. Each SHDHT Node maintains a Node Routing Table of local SHDHT Mapping Overlay. SHDHT Nodes locates in the same Mapping Overlay implement hash operation based on the same hash algorithm. SHDHT Nodes hash data object to be a unique Resource ID, and perform put/get/move operations based on the Resource IDs.

Node ID: Node identifier, which is used for maintenance. Each SHDHT Node has a unique Node ID. The ring containing Node IDs indicates overlay's topology.

Partition ID: Partition identifier, which is used for hash space assignment. Partition IDs and Resource IDs share the same hash

space. All Partition IDs in overlay are unique. Each SHDHT Node could have multiple Partition IDs. The ring containing Partition IDs determines how the hash space is partitioned into segments and how these segments are assigned to nodes.

Resource ID: Each data object stored in DHT overlay could be hashed to be a unique Resource ID. In LISP-SHDHT strategy, data objects correspond to the EIDs. Resource IDs share the same hash space with Partition IDs. As a result, SHDHT Nodes perform data objects put/get/remove operations based on these IDs.

Node Routing Table: Routing table of a SHDHT Mapping Overlay which contains all SHDHT Nodes information of this overly, including Node IDs, Partition IDs and Node IP addresses, etc. Each SHDHT Node of this overly will maintain the Routing Table.

SHDHT Map Resolver: A SHDHT Node that also implements Map Resolver functionality (accept Map-Requests, and forward them to corresponding SHDHT Map Servers based on information get from SHDHT mapping database or forward them to corresponding SHDHT Map Servers through SHDHT Nodes, which corresponds to different operation modes of the SHDHT Mapping Database.).

Report Message: Kind of message sent by Map Server to SHDHT Nodes to publish the EIDs/EID prefix information in charge of Map Server onto the LISP-SHDHT mapping overlay.





SHDHT Border Node: SHDHT Border Node locates on the border of a Domain SHDHT Overlay. Each SHDHT Border Node maintains an Inter-Domain Routing Table. SHDHT Border Nodes are used to flood cross domain routing and forward cross domain packets.

Inter-Domain Routing Table: A routing table maintained on a SHDHT Border Node. This routing table contains information of other Domain SHDHT Overlays, such like EID prefixes other domain overlays maintain, IP addresses and ports information of other overlay's Border Nodes.

## 3. SHDHT Overview

### 3.1. Node ID and Partition ID

Most of existing DHTs use node IDs for both maintenance and hash space arrangement. For example, in LISP-DHT[LISP-DHT], each chord node of the DHT ring has a unique k-bits identifier (ChordID). Nodes perform operations such like put/get/remove based on ChordIDs.

Furthermore, ChordIDs are also used to associate nodes with hash space segments that the nodes responsible for.

In SHDHT, two roles of maintenance and hash space arrangement are separated and a new kind identifier called Partition ID is adopted. Each SHDHT node has a unique Node ID which identifies the physical node and multiple Partition IDs which represent hash space segments. All Partition IDs in the overlay are also unique. Node IDs and Partition IDs are mapped into two ring-shaped spaces respectively. The ring containing Node IDs indicates the overlay's topology. The ring containing Partition IDs determines how the hash space is partitioned into segments and how these segments are assigned to nodes. It is noteworthy that SHDHT Nodes could determine number of Partition IDs on them separately and could generate Partition IDs randomly just need to make sure that the generated Partition IDs will not conflict with existing Partition IDs on the SHDHT plane.

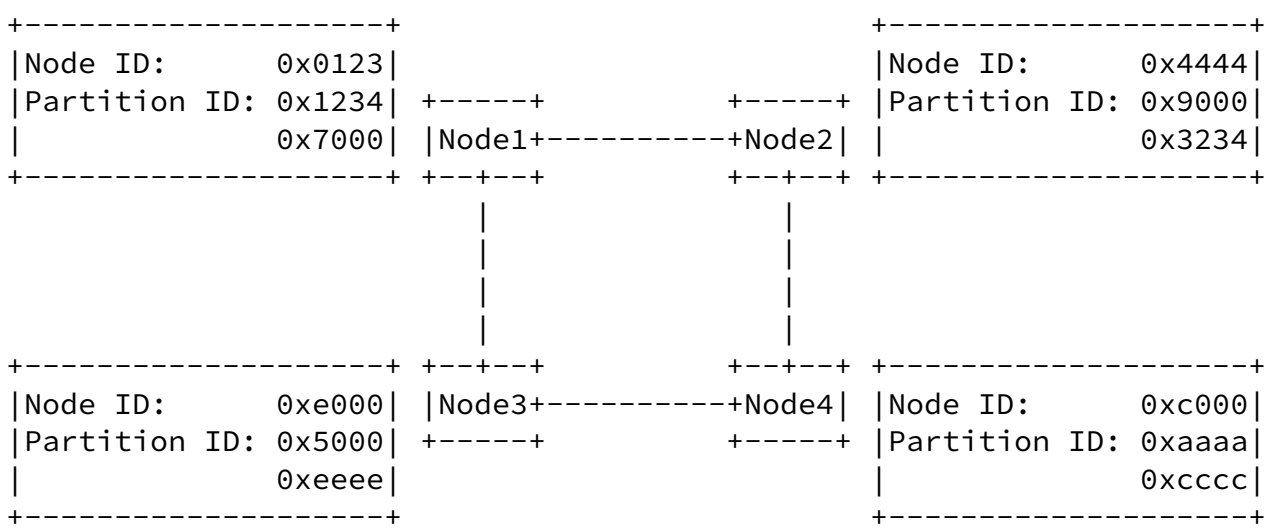



Fig.1 SHDHT Deployment Example

As shown in Fig.1 is an example of SHDHT. This SHDHT overlay is consist of four SHDHT NODEs each has a unique Node ID and maintains two Partition IDs. According to this deployment, hash space is partitioned to be eight segments each is indexed by a Partition ID. From Fig. 1, it could be observed that hash space segments are not required to be partitioned equally. As SHDHT Nodes could generate





Partition IDs separately, when a SHDHT Node gets all hash segments

assignment information of other SHDHT Nodes, it will be able to implement the load balance of SHDHT overlay by generate proper Partition IDs.

In SHDHT, each SHDHT Node stores and maintains data objects. Data objects are indexed by Resource IDs which share the same hash space with Partition IDs and will locate in the hash space segments whose Partition IDs are closest to their Resource IDs.

For example, for a data object whose Resource ID is 0x8213, the Resource ID locates between Partition ID 0x7000 and Partition ID 0x9000. As Partition ID 0x9000 is closer to Resource ID 0x8213, the data object will be stored and maintained on Node2 who is assigned with the hash space segment indexed by Partition ID 0x9000.

## 3.2. Data Storage and Hash Assignment

In traditional DHTs, hash space is partitioned into segments based on node IDs. As a result, data objects are always stored in their root nodes, whose node IDs are "closest" to data objects' Resource IDs.

What does "closest" means? Suppose we have three consecutive Partition IDs a, b and c which are the only Partition IDs in SHDHT for our example, then the range of each hash space segment is defined as follow:

Partition ID a: [id(a)-0.5*d(c,a); id(a)+0.5*d(a,b))

Partition ID b: [id(b)-0.5*d(a,b); id(b)+0.5*d(b,c))

Partition ID c: [id(c)-0.5*d(b,c); id(c)+0.5*d(c,a))

with functions

id(x): value of Partition ID x in hash space

d(x,y): distance between Partition ID x and y in hash space

Replications of data objects in a particular node are always stored in the preceding node or successor node of the root node. The backup preceding node or successor node will automatically become the new closest node if the root node leaves the overlay.

In SHDHT, the whole hash space is partitioned into segments based on partition IDs. The root node of a data object is the node, which has the closest partition ID to the data object's Resource ID. In SHDHT, each node can maintain multiple hash space segments with respective

Partition IDs. As the preceding Partition ID or successor Partition ID may be owned by the same root node. Replication of data objects could still be stored in preceding node or successor node of root node.

## 3.3. Node Routing Table

In SHDHT, each node maintains a Node Routing Table containing routing information of all other SHDHT Nodes locate in the same SHDHT overlay. Table I shows the Node Routing Table on SHDHT Nodes of Fig.1. A Node Routing Table contains all Partition IDs and their associated Node IDs and node addresses. For simplification, Node IDs and Partition IDs shown in the draft are only 16-bit numbers.

When SHDHT Node receives a message points to a particular Resource ID, it could look up Node Routing Table and find out the Partition ID which is closest to the Resource ID. Furthermore, message could be transferred to the corresponding SHDHT Node.

| Partition ID | Node ID | Address:Port |
|---|---|---|
| 0x1234 | 0x0123 | 10.0.0.2:2000 |
| 0x3234 | 0x4444 | 10.0.0.3:2000 |
| 0x5000 | 0xe000 | 10.0.0.4:2000 |
| 0x7000 | 0x0123 | 10.0.0.2:2000 |
| 0x9000 | 0x4444 | 10.0.0.3:2000 |
| 0xaaaa | 0xc000 | 10.0.0.5:2000 |
| 0xcccc | 0xc000 | 10.0.0.5:2000 |
| 0xeeee | 0xe000 | 10.0.0.4:2000 |

TABLE I SHDHT Node Routing Table

For example, if Node 1 (ID: 0x1234) in Fig.1 needs to implement put/get/remove operations for a data object with Resource ID 0x8213. Node 1 first looks up its Node Routing Table and finds out that the closest Partition ID corresponding to this Resource ID is 0x9000. Then Node 1 will send put/get/remove request to the node owns the Partition ID, in Fig.1 is Node2, whose Node ID is 0x4444 and address is 10.0.0.3:2000.

## 4. LISP SHDHT

LISP SHDHT is proposed to provide "EID-to-RLOC(s)" mapping information lookup service for sites running the Locator/ID Separation Protocol (LISP).

As shown in Fig.2, LISP SHDHT is consists of SHDHT Nodes in which some nodes play roles of Map Resolver. And in the draft, term "SHDHT-MR" is adopted to identify these nodes.

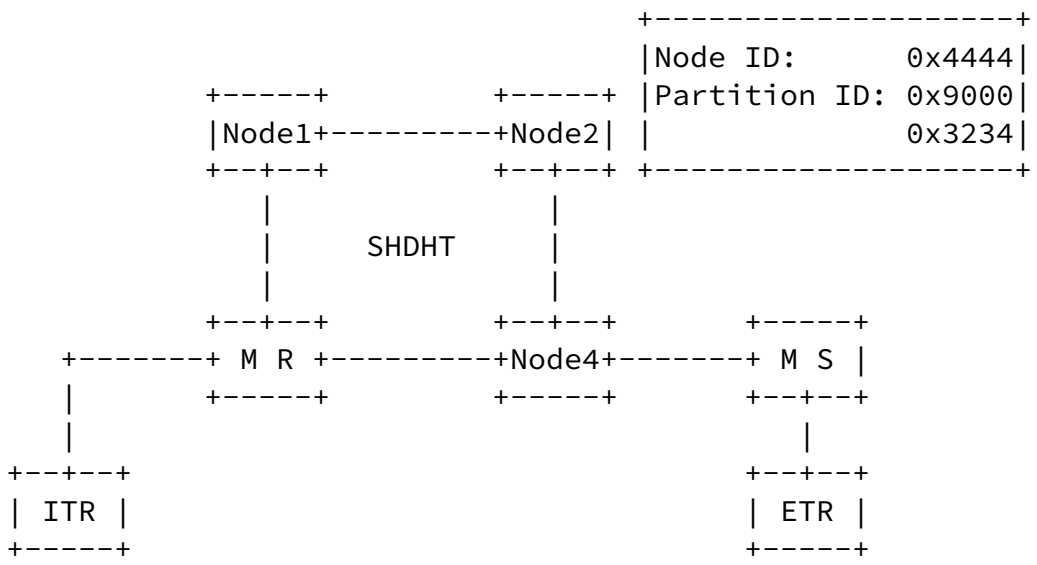



Fig.2 LISP-SHDHT Deployment Example

Map Server publishes EIDs/EID prefixes information it responsible to onto SHDHT Mapping overlay. All EIDs/EID prefixes entries are stored in SHDHT Nodes as data objects. EIDs/EID prefixes in mapping entries can be hashed as Resource IDs of data objects. All SHDHT Nodes in the same SHDHT overlay perform hash operation based on the same hash algorithm.

In this draft, LISP-SHDHT can run in two distinct modes: i) SHDHT Forward Mode and ii) Recursive Lookup Mode.

### 4.1. ITR Operation

According to LISP-MS [RFC6833], LISP ITRs use Map Resolvers as proxy to send control messages, such like encapsulated Map-Requests and Map-Replies.

In Scenario of LISP SHDHT, an ITR send Map-Requests directly to the SHDHT Node which is selected to play roles of SHDHT Map Resolver for that ITR.





## 4.2. ETR Operation

LISP ETR plays the same role as defined in LISP-MS [RFC6833]; it registers mapping information onto the Map Server by sending Map-Register messages.

## 4.3. SHDHT Map Server Operation

When Map Server receives Map Request messages, it forwards the messages to ETRs or send Map-Reply messages directly based on M-bits of ETRs' Map Registers. That's to say, Map Server performs the same operation as defined in LISP-MS[RFC6833] .

Extended in LISP SHDHT, Map Server needs to publish EIDs/EID prefixes information onto the LISP-SHDHT mapping overlay.

For example, as shown in figure 2, when a Map Server needs to publish EIDs information such like EID "1.1.1.1" onto LISP-SHDHT, following operations will be performed.

1. Map Server sends a report message to the nearest SHDHT Node, i.e. Node 4. Map Server contains the information of EID "1.1.1.1" in the report message, along with Map Server's routable RLOCs. Report message may not be a kind of new defined message. For example, Map Server could advertise EID information onto SHDHT mapping database based on BGP protocol.

2. When Node 4 receives the report message, it extracts EID information from the message, i.e. EID "1.1.1.1". Node 4 then hashes the report EID to be a Resource ID.

3. Suppose Node 4 hash EID "1.1.1.1" to be Resource ID 0x8956. Then

it checks the Nodes Routing Table to find out which Node maintains the hash space whose Partition ID matches the Resource ID. In this example, the match Partition ID is 0x9000, and the corresponding hash space is maintained by Node 2.

4. Node 4 forwards the report message to Node 2. Node 2 stores the (key, value) pair, where key is the Resource ID 0x8956, and value contains reported EID "1.1.1.1" along with corresponding Map Server's RLOCs.

Other SHDHT Nodes now could contact Node2 to get which Map Server is responsible to the EID "1.1.1.1".





Note: In previous example, we suppose that Map-Server reports an EID address onto LISP-SHDHT. In practical deployment, Map Server reports EID prefixes at most time. Details about EID prefix report will be illustrated in Section 4.6.

## 4.4. SHDHT Map Resolver Operation

As previous mentioned, LISP-SHDHT can run in two distinct modes: i) SHDHT Forward Mode and ii) Recursive Lookup Mode.

As shown in Fig.2, suppose SHDHT Map Resolver receives a Map-Request message target at EID 1.1.1.1. SHDHT Map Resolver operations under two different modes are illustrated in following sections.

### 4.4.1. SHDHT Map Resolver Operation under SHDHT Forward Mode

Under SHDHT Forward Mode, SHDHT Map Resolver performs the following operaion.

1. SHDHT Map Resolver extracts destination EID address "1.1.1.1" from the Map-Request message.

2. SHDHT Map Resolver hashes the EID address to be Resource ID 0x8956 based on the shared hash algorithm.

3. SHDHT Map Resolver looks up Node Routing Table and finds out the Partition ID 0x9000 which matches the Resource ID.

4. SHDHT Map Resolver forwards Map-Request message to the corresponding SHDHT Node 2 who maintains the hash space labeled by matched Partition ID 0x9000.

### 4.4.2. SHDHT Map Resolver Operation under Recursive Lookup Mode

Under Recursive Lookup Mode, SHDHT Map Resolver performs the following operation.

1. SHDHT Map Resolver receives the Map-Request message and stores it in local catch.

2. SHDHT Map Resolver hash destination EID of Map-Request to be Resource ID 0x8956.

3. SHDHT Map Resolver looks up Node Routing Table and finds out that Node 2 maintains the corresponding hash space and stores the data object indexed by 0x8956.





4. SHDHT Map Resolver query Node 2 to get data object indexed by 0x8956, i.e. get EID information and RLOCs of the Map Server who maintains the destination EID.

5. SHDHT Map Resolver forwards Map-Request message to corresponding Map Server based on data object.

Under Recursive Lookup Mode, SHDHT Map Resolve could catch information of the Map Server, including EID prefix Map Server responsible for and Map Server's RLOCs. When receives other Map Requests whose destination EIDs covered by Map Server's EID prefix, Map Resolver could forward Map Requests directly to Map Server.

## 4.5. SHDHT Nodes Operation

As specified in Section 4.3, when SHDHT Nodes receive a report message, it will hash the report EID/EID prefix to be Resource ID, and check which Node should store the data object. If itself is the

responsible Node, it will store the (key, value) pair, otherwise, it forward report message to corresponding Node.

As specified in Section 4.4.1, under SHDHT Forward Mode, when a SHDHT Node receives a Map-Request message from a Map Resolver, it hashes the requested EID to be a Resource ID to get the data object stored in its hash space. Then SHDHT Node forwards the Map-Request to Map Server based on data object information.

As specified in Section 4.4.2, under Recursive Lookup Mode, when a SHDHT Node receives a query message from Map Resolver, it replies with the data object Map Resolver requested.

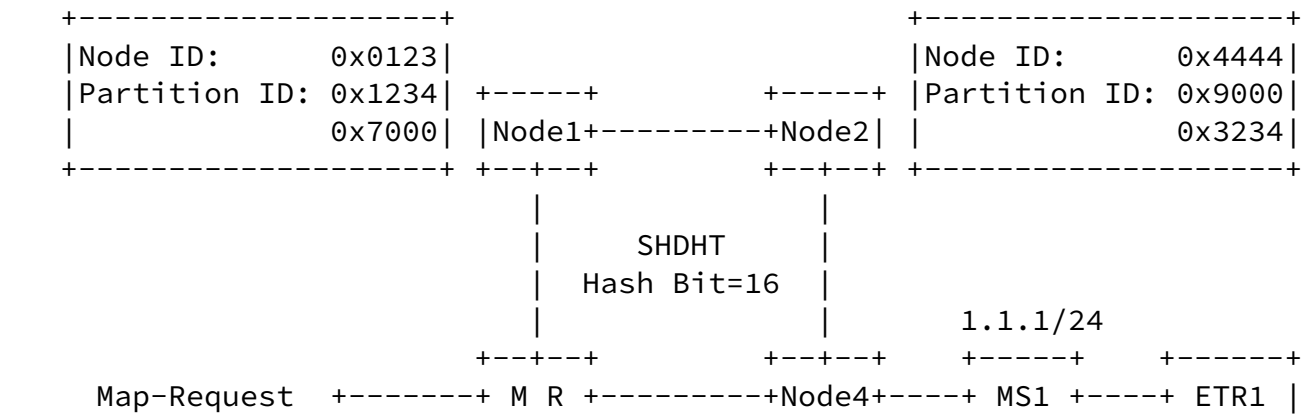
## 4.6. EID prefixes Report onto LISP-SHDHT

In LISP-SHDHT, Map Server always report EID prefixes onto the SHDHT Mapping overlay. However, Map-Request message always targets at a specific EID address. How to hash the requested EID and the EID prefix covered this EID to be the same Resource ID? Each LISP-SHDHT Mapping overlay could configure a "Hash Bit" to solve this problem.

As shown in Fig.3, suppose the LISP-SHDHT Mapping overlay configures the "Hash Bit" to be 16 bits.







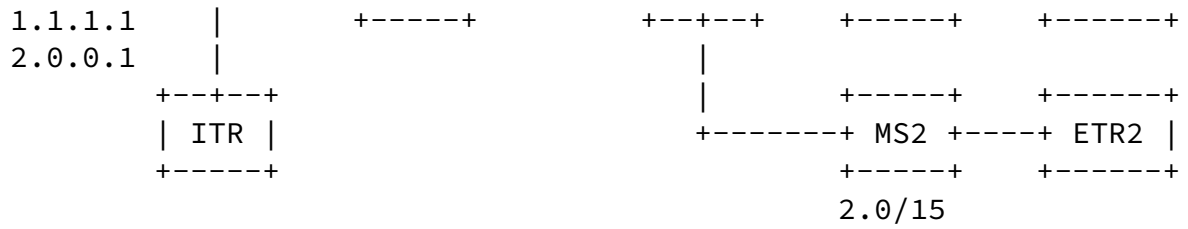



Fig.3 EID Prefix Report Example

Example 1: MS1 reports EID prefix 1.1.1/24 onto the LISP-SHDHT.

1. When Node4 receives the report message from MS1, it extracts the EID prefix 1.1.1/24.

2. Node4 hashes the EID prefix to be Resource ID based on Hash Bit. In this case, Hash Bit is 16 bits, as a result Node4 hashes the /16 prefix of reported EID prefix. That's to say, Node4 hashes 1.1/16 to be a Resource ID, suppose to be 0x8560.

3. Node 4 checks Node Routing Table and forwards the report message to Node 2 who maintains the corresponding hash space with Partition ID 0x9000.

In this example, when another Map Server advertises EID prefix such like 1.1.2/24, this prefix will also be hashed to be Resource ID 0x8560 and the report message will be forwarded to Node 2.

Node 2 maintains (key, value) pair, where key is 0x8560 and value contains all EIDs/EID prefixes information covered by 1.1/16.

Example 2: MS2 reports EID prefix 2.0/15 onto the LISP-SHDHT.

1. When Node4 receives the report message from MS1, it extracts the EID prefix 2.0/15.

2. Node4 hashes the EID prefix to be Resource ID based on Hash Bit. In this case, Hash Bit is 16 bits, Node4 first splits the EID





prefix 2.0/15 to be two 16 bits sub-prefixes, 2.0/16 and 2.1/16. Suppose Node 4 hashes prefix 2.0/16 to be Resource ID 0x1210 and hashes prefix 2.1/16 to be Resource ID 0x3200.

3. As Shown in Fig. 3, data objects with Resource ID 0x1210 and 0x3200 should be stored on Node 1 and Node 2 separately. Node 4 will copy the report message and forward report message both to Node 1 and Node 2.

In this example, Node 1 and Node 2 maintains (key, value) pairs with different keys (0x1210 and 0x3200), but the value both contain the same EID prefix 2.0/15.

In practical deployment, SHDHT service providers could configure proper Hash Bits, in order to avoid the scenario which needs to split a shorter EID prefix to be multiple longer prefixes.

Example 3: ITR sends Map Requests onto LISP-SHDHT.

1. When ITR sends a Map-Request target at EID 1.1.1.1 as shown in Fig.3, SHDHT Map Resolver hashes the EID based on Hash Bit, i.e. SHDHT Map Resolver hashes EID prefix 1.1/16 to get the Resource ID. SHDHT Map Resolver judges the corresponding data object is stored on Node 2 (according to Example 1), then SHDHT Map Resolver could forward the Map-Request to Node 2 (based on SHDHT Forward Mode) or get information about the best matched EID prefix 1.1.1/24 from Node 2 (based on Recursive Lookup Mode).

2. When ITR sends a Map-Request target at EID 2.0.0.1, SHDHT Map Resolver hashes EID prefix 2.0/16 to get the Resource ID. SHDHT Map Resolver judges the corresponding data object is stored on Node 1 (according to Example 2), then SHDHT Map Resolver could forward the Map-Request to Node 1 (based on SHDHT Forward Mode) or get information about the best matched EID prefix 2.0/15 from Node 1 (based on Recursive Lookup Mode).

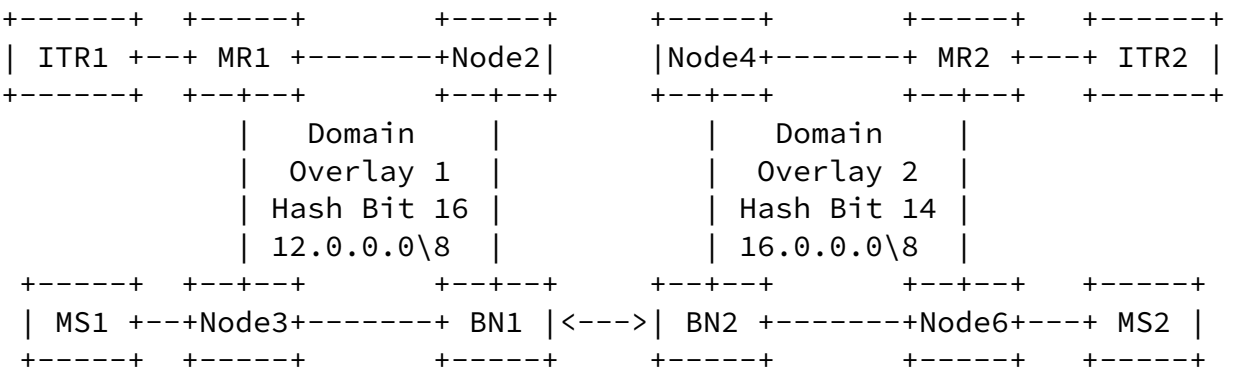
## 5. Domain LISP SHDHT Deployment

LISP is a global architecture. In order to make LISP SHDHT meets requirements of LISP mapping database better, LISP SHDHT should perform better scalability and distribution attributes. Especially in practical deployment, LISP mapping database may be operated by different ISPs, when a new mapping service provider join or leave the mapping database, all other providers should not be influenced to be re-assigned.

In practical deployment, LISP SHDHT mapping overlay could be consist of multiple Domain SHDHT overlays which are operated by different mapping service providers. These Domain SHDHT overlays communicate through SHDHT Border Nodes of each other.

As shown in Fig.4, there are two Domain LISP SHDHT Overlays which communicate through BN1 (Border Node1) and BN2.

In domain LISP SHDHT deployment, different domain overlays maintain EID-to-RLOC mapping information covered by different EID prefixes. As in example of Fig.4, Domain 1 maintains mapping information according to EID prefix 12.0.0.0/8, and Domain 2 maintains mapping information covered by EID prefix 16.0.0.0/8. Furthermore, different Domain Overlay could configure their Hash Bits separately.




* BN: SHDHT Border Node
* MR: SHDHT Map Resolver
* MS: Map Server

Fig.4 Domain SHDHT Deployment Example

### 5.1. SHDHT Border Node

Each Domain SHDHT Overlay has one or more Border Nodes which are not only perform like normal SHDHT Nodes, but also be used to flood cross domain routing and forward the cross domain packets.

Each SHDHT Border Node maintains an Inter-Domain Routing Table, which contains information of all other domain overlays, such like EID prefixes other domain overlays maintain, IP addresses and ports information of other overlays' Border Nodes.

At the beginning, Inter-Domain Routing Table could be configured on SHDHT Border Nodes. Then, a SHDHT Border Node will flood cross domain routing periodically to trigger other Border Nodes update their Inter-Domain Routing Tables.

## 5.2. EIDs/EID Prefixes Report onto Domain SHDHT Overlay

All SHDHT Nodes of a Domain SHDHT Overlay must be noticed the EID prefixes that local domain overlay responsible for. When a SHDHT Node of a domain overlay receives a report message, it checks if the registered EIDs/EID prefixes are covered by local domain overlay's EID prefixes.

If local domain overlay is responsible for reported EIDs/EID prefixes, SHDHT Node who receives report message will process the message as procedures listed in Section 4.3 and 4.6

Otherwise, if local domain overlay is not responsible for reported EIDs/EID prefixes, SHDHT Node who receives report message will forward it directly to local domain overlay's Border Nodes. Then, Border Nodes will forward the message to corresponding domain overlay based on the Inter-Domain Routing Table.

Suppose in Fig.4, MS2 reports EID prefix 12.2.0/24 to Node 6 of Domain 2.

1. Node6 extracts the EID prefix from report message and finds that the reported EID prefix is 12.2.0/24.

2. Node6 determines that EID prefix 12.2.0/24 is not covered by Domain 2's prefix 16.0.0.0/8.

3. Node6 forwards the report message to BN2.

4. BN2 looks up Inter-Domain Routing Table to find that Domain 1 is

responsible for EID prefix 12.2.0/24. BN2 forwards report message to Domain 1's Border Node (BN1).

5. BN1 processes the report message based on procedures introduced in Section 4.3 and 4.6.

## 5.3. Mapping Request Lookup onto Domain SHDHT Overlay





When SHDHT Map Resolver receives a Map-Request message, it checks if the requested EID is covered by local domain overlay's EID prefixes, i.e. if the requested mapping entry is stored on local domain overlay.

If local domain overlay is responsible for requested EID, SHDHT Map Resolver processes the message based on procedures introduced in Section 4.4 and 4.6.

Otherwise, if the requested EID entry is not stored on local domain overlay, under SHDHT Forward Mode, SHDHT Map Resolver directly forwards the Map-Request to Border Nodes. Border Nodes of local domain overlay then forwards it to corresponding domain overlay based on Inter-Domain Routing Table.

Suppose in Fig.4, ITR2 sends a Map-Request message to SHDHT MR 2 of Domain 2 to get mapping information of EID 12.2.0.1.

1. SHDHT MR2 extracts requested EID from the Map-Request message.

2. SHDHT MR2 determines that requested EID 12.2.0.1 is not covered by Domain 2's prefix 16.0.0.0/8.

3. SHDHT MR2 forwards the Map-Request message to BN2.

4. BN2 extracts requested EID and looks for Inter-Domain Routing Table to find corresponding domain overlay of EID 12.2.0.1.

5. BN2 finds out Domain 1 is responsible for EID 12.2.0.1. BN2 forwards Map-Request message to Domain 1's Border Node (BN1).

6. BN1 processes the Map-Request message based on procedures introduced in Section 4.4 and 4.6.

If the requested EID entry is not stored on local domain overlay, under Recursive Lookup Mode, SHDHT Map Resolver catch the Map-Request message, and send query message to Border Nodes. Border Nodes of local domain overlay query Border Nodes of the corresponding domain overlay responsible for requested EID entry to get related information.

Suppose in Fig.4, ITR2 sends Map-Request message to SHDHT MR 2 to get mapping information of 12.2.0.1.

1. SHDHT MR2 determines the requested EID 12.2.0.1 is not covered by Domain 2's prefix 16.0.0.0/8.





2. SHDHT MR2 catches the Map-Request message and sends a query message for EID 12.2.0.1 to BN2.

3. BN2 finds out Domain 1 is responsible for the requested EID. BN2 sends query message to BN1.

4. BN1 hashes 12.2/16 to be Resource ID to get which SHDHT Node in Domain 1 now maintains information of EIDs covered by EID prefix 12.2/16.

5. Suppose the responsible Node in Domain 1 is Node 2. Node 2 maintains information of EID prefix 12.2.0/24 along with MS2's RLOC information. BN2 queries Node 2 to get the information and sends them to BN1.

6. BN1 sends relative information to SHDHT MR 2.

7. After the recursive lookup procedures, SHDHT MR 2 sends the Map-Request directly to MS2.

## 6. Security Considerations

TBD

## 7. IANA Considerations

This document makes no requests to IANA.

# 8. References

## 8.1. Normative References

[I-D.meyer-lisp-mn]

Farinacci, D., Lewis, D., Meyer, D., and C. White, "LISP Mobile Node", October 2012.

[I-ietf-lisp-ddt]
Fuller, V., Lewis, D., and V. Ermagan, "LISP Delegated Database Tree", October 2012.

[LISP-DHT]
Mathy, L. and L. Iannone, "LISP-DHT: Towards a DHT to map identifiers onto locators, http://dl.acm.org/citation.cfm?id=1544073", December 2008.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "Locator/ID Separation Protocol (LISP)", January 2013.

[RFC6833] Fuller, V. and D. Farinacci, "LISP Map Server Interface", January 2013.

[RFC6836] Fuller, V., Farinacci, D., Meyer, D., and D. Lewis, "LISP Alternative Topology (LISP+ALT)", January 2013.

## 8.2. Informational References

[I-D.lear-lisp-nerd]
Lear, E., "NERD: A Not-so-novel EID to RLOC Database", April 2012.

## Appendix A. Acknowledgments

The authors with to express their thanks to Michael Hoefling for work on Hash space segment of SHDHT overlay. Thanks also go to Dino Farinacci and Darrel Lewis for their suggestions about database structure deployment.

## Authors' Addresses

Li Cheng
ZTE Corporation
R&D Building 1, Zijinghua Road No.68
Nanjing, Yuhuatai District 210012
P.R.China

Email: cheng.li2@zte.com.cn

Mo Sun
ZTE Corporation
R&D Building 1, Zijinghua Road No.68
Nanjing, Yuhuatai District  210012
P.R.China

Email: sun.mo@zte.com.cn