Workgroup: Network Working Group
Internet-Draft: draft-kjsun-lisp-dyncast-03
Published: 5 November 2022
Intended Status: Informational
Expires: 9 May 2023
Authors: K. Sun Y. Kim
ETRI Soongsil University

LISP for Computing-Aware Networking

Abstract

When a service has been distributed over many locations in the network, providing service by utilizing computing resources hosted in various servers is required to consider supporting delay-sensitive service and optimizing network loads. For that reason, Computing-Aware Networking (CAN) is a new routing approach to balance services using service-specific metrics instead of simply dispatching the service request in a static way or optimizing solely connectivity metrics [draft-liu-can-ps-usecases]. Currently, CAN solutions are discussed with both existing technologies (e.g., DNS, L7 Load Balancer, etc.) and a new approach. In this document, it describes the LISP-based CAN approach and related standard works to meet requirements.

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 9 May 2023.

Table of Contents

1. Introduction

When a service has been distributed over many locations in the network, providing service by utilizing computing resources hosted in various servers is required to consider supporting delay-sensitive service and optimizing network loads. For that reason, Computing-Aware Networking (CAN) is a new routing approach to balance services using service-specific metrics instead of simply dispatching the service request in a static way or optimizing solely connectivity metrics [draft-liu-can-ps-usecases]. With enhancing current technologies such as DNS or L7 Load Balancer, one of CAN solutions is to take into account computing as well as service-specific metrics in the distribution decision seen as dynamic anycast ("dyncast", for short).

The main feature of the dyncast described in [draft-liu-dyncast-ps-usecases] is that a unique service identifier that can be assigned to multiple instances in multiple edge environments should be able to be mapped as an actual routable unicast address. Since this concept is similar to the Location/ID separation method already used in the LISP design basis, the LISP protocol can be considered as one of the candidate protocols that can implement dyncast. This draft is proposed to design the LISP-based CAN approach with dyncast and analyze the extension method of

LISP to meet the requirements defined in [draft-liu-can-gap-reqs] for realizing dynamic anycasting between different LISP sites.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document is to be interpreted as described in [RFC2119]. This document uses the terminology described in [RFC6830], [draft-liu-dyncast-ps-usecases], [draft-liu-dyncast-reqs]. Detailed definition of terminologies are written below.

Dyncast : As defined in [draft-liu-dyncast-ps-usecases], Dynamic Anycast, taking the dynamic nature of computing resource metrics into account to steer an anycast routing decision.

D-Router: A node supporting Dyncast functionalities as described in this document. Namely it is able to understand both network-related and service-instances-related metrics, take forwarding decision based upon and manitain instance affinity, i.e., forwards packets belonging to the same service demand to the same instance.

Dyncast Metric Agent (D-MA): A dyncast specific agent able to gather and send metric updates (from both network and instance prespective) but not performing forwarding decisions. May run on a D-Router, but it can be also implementated as a separate module (e.g., a software library) collocated with a service instance.

Dyncast Service Endpoint ID (DSEID) : Anycast IP address assigned to the service running on distributed locations. DSEID cannot be routed globally, and it is unique for specific service. Multiple service instances which are same service have a same DSEID.

D-BID: Dyncast Binding D-Node, an address to reach a service instance for a given DSEID. It is usually a unicast IP where service instances are attached. Different service instances provide the same service identified through D-SID but with different Dyncast Binding IDs. In the LISP architecture, D-BIDs of same service are replaced to RLOC-set of DSEID.

3. Overview of LISP use-case for Computing-Aware Networking

As described in [draft-liu-can-ps-usecases], for guaranteeing quality of service across multiple edge sites, service providers can ensure functional equivalency by deploying instances for the same service across multiple edge sites. For that, Computing-Aware Networking (CAN) considers steering traffic dynamically to the "best" service instance not only in terms of network performance but also computing capability. Figure 1 describes the LISP use-case for Computing-Aware Networking. In the LISP architecture [RFC9299], each

edge network has one or more LISP routers deployed. If we consider where services are running on the edge, each service is regarded the same as endpoint that has its globally unique EID and RLOC depending on its location, it is hard to provide service equivalency to the user which is hard to discover the same service across multiple edges. One possible approach is allocating anycast EID for the same service instances, which is based on dynamic anycast (Dyncast) approach described in the [draft-li-dyncast-architecture]. [RFC6830] defines that anycast address can be assigned for both Endpoint ID (EID) and Routing Locator (RLOC) within each of their address spaces. In order to forward a packet destined for an anycast EID between LISP edges, the addresses of the LISP Egress Tunnel Router (ETR) are used as RLOC-set, which exposes locations where equivalent services are running. Each RLOC address in RLOC-set is routable in the underlay, but it is not unique value per each service instance. When multiple services are running on the same LISP site, they can be assigned the same RLOC which is the xTR of their LISP site. Map-server/resolver of the LISP control plane can manage mapping information for Anycast EID-to-RLOC-set mappings together with existing EID-to-RLOC mappings.

For resource-efficient forwarding decisions across multiple service instances, [draft-li-dyncast-architecture] defines Dyncast Metric Agent (D-MA) which collects metrics related to network and service instances. Actual packet forwarding is handled in the Dyncast Router (D-Router) based on collected metrics with maintaining instance affinity. In the LISP architecture, the D-Router and D-MA function can be implemented on each LISP ETR, or can be deployed as separate components within the edge for managing service instances. The LISP control plane is logically centralized and it provides an interface with each LISP router to exchange mapping information. However, it does not mean that the LISP control plane is located in a single physical location, several mechanisms for distributing the mapping system already have been defined.

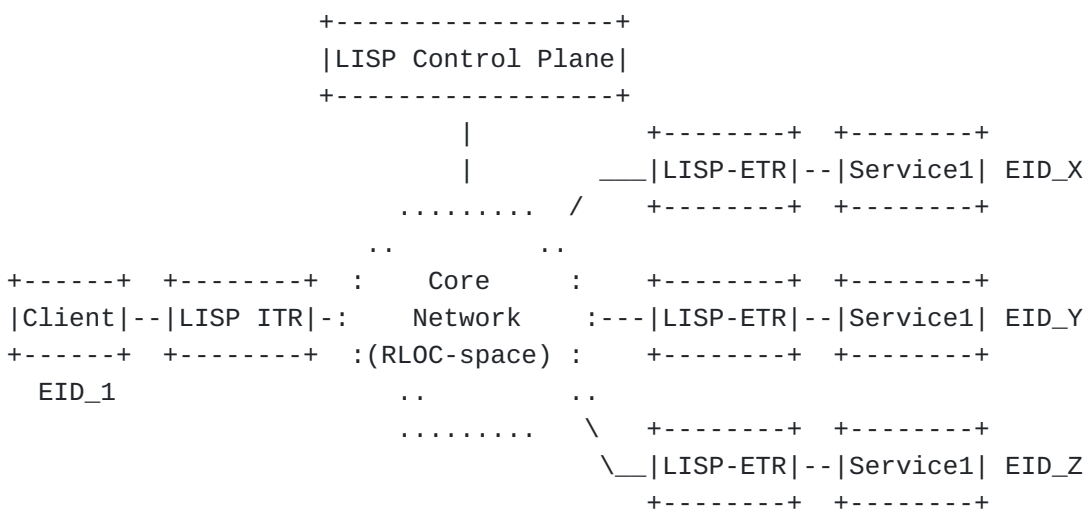

Figure 1: LISP use-case for Dynamic anycast

Figure 3 shows an example of LISP-based dyncast deployment where two services each deployed two instances at different edges. In this scenario, two services are assigned an RLOC according to the ETR address of the LISP site. Both Service_A and Service_B instances connected to ETR_2 are assigned RLOC2, which is the RLOC of ETR_2, as a binding ID. According to this figure, Anycast EID-to-RLOC-set mappings can be configured as an example below.

Anycast EID	RLOC-set
EID_A	RLOC-set_A ({RLOC2, metric}, {RLOC3, metric})
EID_B	RLOC-set_B ({RLOC2, metric}, {RLOC3, metric})

Figure 2: DSEID-to-RLOC-set Example

In addition to these examples, the RLOC-set can also be used in the form of Explicit Locator Path (ELP) or Run-Length Encoding (RLE) for the encap-path between ETR and ITR.

In the case of the edge where ETR_2 is located, as an edge composed only of service instances, the LISP Router function can be operated by being strongly coupled to the edge computing server. Mapping information update for Anycast EID is performed through the LISP protocol Map-Register message, and service-instance-related metrics can be delivered through the LISP protocol header or other methods. A method of inserting service-instance-related metric information

into the LISP protocol will be discussed later. When the ITR_1 receives a packet destined for the EID of the service by service request from the Host_1, the ITR can acquire the RLOC-set of the requested EID from the LISP control plane through the Map-Request message. At the control plane, it may select a proper RLOC on the collected metric information and return it to the ITR or return the RLOC-set of multiple service instances with metric information to the ITR so the ITR selects the proper RLOC in the set. A method for determining an appropriate RLOC will be discussed later.

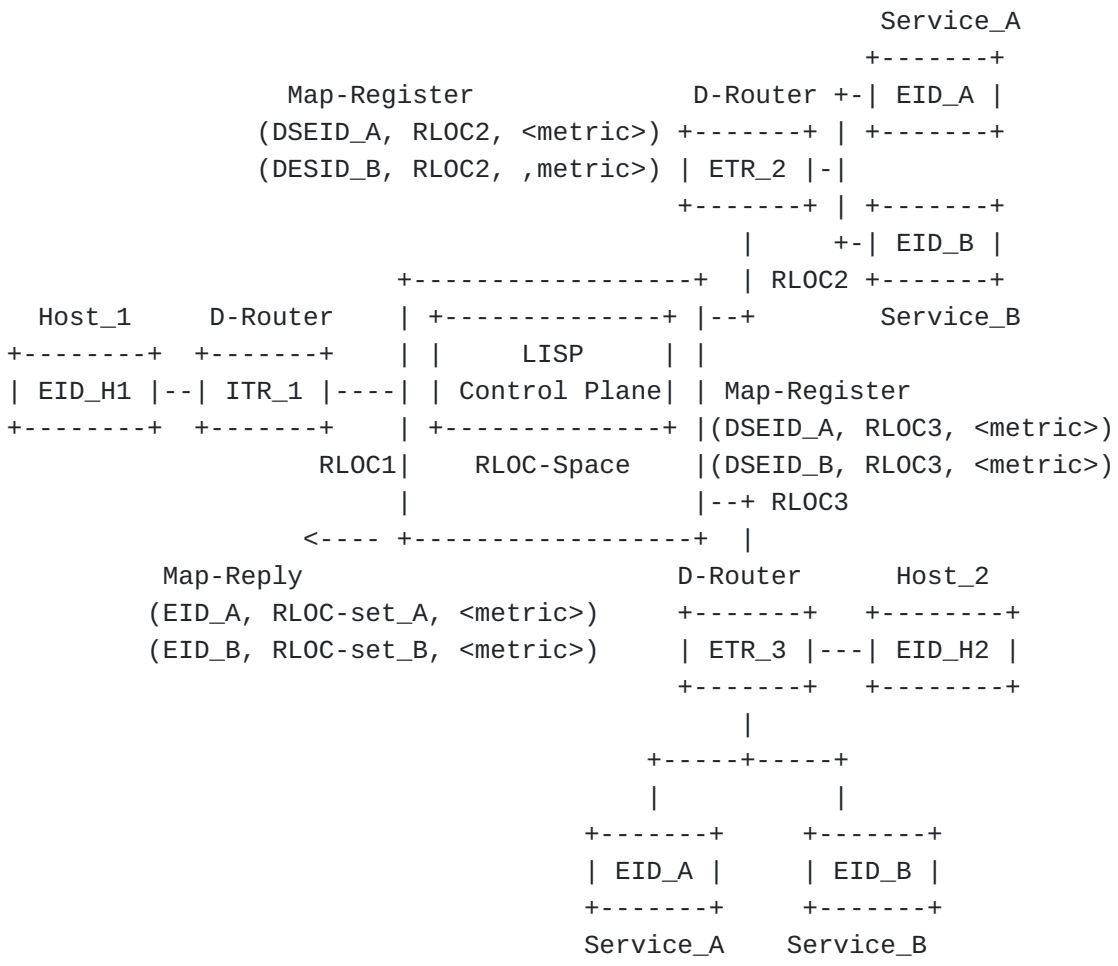

Figure 3: LISP-based Dyncast Example Scenario

4. Addressing CAN Requirements with LISP

In the following, we analyze for the LISP-based CAN approach to solve requirements described in [draft-liu-can-gap-reqs]

4.1. Dynamic and effective selection among multiple service instances

To support dyncast selection, the system must provide a method for searching a service identifier and mapping it to a specific unicast address. From this point of view, the LISP is a suitable protocol for separating ID/Location of service and managing mapping information. When the system allocates the same EID to each service instance for service equivalency, the LISP can define an anycast address space for the Anycast EID and assign it to service instances created across multiple sites. Also, the D-BID can be replaced to an RLOC address of LISP xTR that can be routed between edges as unicast. That is, it is necessary to define a separate space for anycast address within the existing EID space and to allocate it in advance so that it can be used in all edge networks where the service instances are located. In the LISP definition, the EID assigned to each service has a globally unique value and, in particular, [RFC6830] defines that anycast address can be assigned within an EID or RLOC block spaces. In each LISP site, same as the EID which is defined to enable internal routing, the DSEID can be able to be routed without the RLOC encapsulation process to the EID within a single site.

One of alternative addressing solution is to use anycast-EID-to-anycast-RLOC mapping. Using this, it is required to register from one place (an SDN controller) or each ETR registering the same RLOC without any merge semantics. So the service is chosen by destination address in a packet (the anycast-EID) which maps to an anycast-RLOC where the underlay takes you to the "closest" LISP site. However, in the dyncast, routing selection is not depending on just distance but also computing resources of each service location. Depending on dynamics of these metrics, anycast-RLOC should be registered/deregistered at the ETR depending on the absence of specific anycast-EID. Further discussion is required which is more efficient rather than using indirection mapping and update it with unicast-RLOC with metric information.

4.2. Support session and service continuity

To maintain "Instance Affinity", it is required to provide routing to the same service instance for the same flow. In LISP, the RLOC mapping information for the destination EID is stored in a local cache called Map-cache in the ITR for a certain period of time, and it is maintained for a set time-to-live (TTL) time. Therefore, mapping information for a specific service once requested from a client is generally maintained in the ITR until the corresponding session expires and can be delivered to the RLOC stored in the map-cache entry. However, in order to have a flexible selection of service instances between different flows at the same point, it is additionally required to assign different RLOCs for different flows

depending on metrics dynamically changed. For that, it is necessary to enhance ITR Map-cache to maintain destination RLOC for each flow. In [draft-rodrigueznatal-lisp-multi-tuple-eids], it can be supported to store Multi-Tuple Extend-EID mappings. With Multi-Tuple EID mappings, it is possible to provide RLOC affinity depending on its destination EID as well as other information such as source EID, protocol or port number. For that, it is required to support multi-stage lookup process, where the multi-tuple EID mappings that point to an DSEID and then there is a anycast EID mapping that points to RLOC-set.

In addition, although the general TTL value in LISP ITR is defined as 24 hours, in dyncast the system requires a shorter TTL time for changing network path depending on dynamically updated network-related and service-instance-related metrics. The LISP support to send a refresh Map-Request before removing map-cache entry. If it needs a shorter TTL to update the map-cache, two options are possible. First option is to send Solicit Map-Request(SMR) for refreshing cache, and another option is to use Pub/Sub which is described in [draft-ietf-lisp-pubsub].

4.3. Support metrics reqresentation and distribution

The one of most important requirements is that it should be able to collect various metrics of service-instances-related as well as network-related, and include them in-network routing decisions. For that, it is necessary to define how to collect these metrics and forward them, and also where to make a decision. In the LISP environment, since that the entire EID-RLOC mapping information is managed in the control plane, one possible scenario is that the controller function which collects service-instance-related metrics updates them to the DSEID mapping entry in the LISP control plane. For that, it can be used an encoding method proposed in [draft-ietf-lisp-name-encoding] that defines to insert specific information such as parameters for a specific EID or RLOC using an ASCII string. Using that, it is possible to encode a string that is pre-defined of a specific metric to interpret in the control plane and send a Map-Request message so that the control plane can select an appropriate RLOC based on it. Another possible option is to use policy distribution by a network controller, which is proposed in [draft-kowal-lisp-policy-distribution]. Using network controller, the ITR could receive and apply the QoS policies that would shape traffic to the correct rate on each ITR RLOC interface. In order to insert service-instance-related metrics from the EID side, the agent function may be needed to forward the metrics of the requested service to the LISP ITR so that the metric can be inserted into the header of the Map-Register message. This metric information encoded into the Map-Register message can help the LISP control plane to make multi-tuple mapping entry and sent it to the requested ITR.

Once the requested ITR receives these information, it can make a routing decision based on the multi-tuple parameters.

4.4. Support flexible use of metrics

CAN system is required that in must make routing decisions for all service requests, and this must be done under an understanding of all metrics. Routing decisions in the LISP can be done with two options which is done in the control plane or ITR by specifying priority and weight values for each RLOC. In case that routing decisions are made in the control plane, the Map-Resolver dynamically sets the priority and weight values of each mapped RLOCs, selects a proper RLOC based on them, and forward it to the requested ITR using the Map-Reply message. However, since this centralized approach may not be calculated based on point of requested ITR, the actual routing path may not be optimal. In case that routing decision is determined at the ITR, the LISP control plane may return one or more RLOC values for the requested EID to the ITR, including priority and weight values based on the collected metrics. After receiving these metrics, the ITR stores them in map-cache entry and selects an appropriate one to forward the data packet. For that, a mechanism for estimating appropriate priority and weight values based on both network-related and service-instance-related metrics is required for the control plane or ITR. When EID-to-RLOC-set mapping is used, it is noted that if RLOCs in the set have equal priority, the ITR can load-split traffic across RLOCs and that cause to break session connection. So, an ITR that is configured that a particular EID in its map-cache is an EID, it should be cared to use an RLOC-set above with each RLOC priority=1.

In the dyncast architecture described in [draft-li-dyncast-architecture], the D-Router collects metrics by exchanging metric information of the service identifier between another edge D-Routers and make a decision itself. This approach can minimize the signaling for routing decisions by decentralizing the authority for the anycast routing decision to an entity in the actual packet path, but the signaling for collecting metrics between each D-Router is bound to increase. In contrast, when the LISP is used, it can reduce effectively signaling of collecting metrics from the ITR since that the mapping information for EID and RLOC-set can be managed in a centralized control plane. However, if the metrics change too much then the contents of the RLOC-set changes which requires more frequent map-cache updates. So analyzing in depth of this tradeoff remains further studies.

For service dynamism, the system should support different selections for each flow according to a dynamically changing metric while considering various requirements in the selection of a service instance. As mentioned in Section 4.2,

[draft-rodrigueznatal-lisp-multi-tuple-eids] can provide the map-cache to be maintained for each flow, so the forwarding path can be dynamically changed to the different service instances by allocating target RLOC to the map-cache entry per-flow according to dynamic changes of metrics. In order to refresh the EID-to-RLOC-set mapping upon changing metric, the Solicit Map-Request(SMR) message can be used to update so that the ITR can update the weight and priority for the RLOC which is already received from the Map-server. Additionally, as proposed in [draft-farinacci-lisp-telemetry], telemetry data can be collected between Encapsulating/Decapsulating xTRs of the current flow, which is expected to be used for dynamic service path reselection.

5. Security Considerations

TBD

6. References

6.1. Informative References

[draft-farinacci-lisp-telemetry] Farinacci, D., Ouissal, S., and E. Nordmark, "LISP Data-Plane Telemetry", May 2022, <https://datatracker.ietf.org/doc/draft-farinacci-lisp-telemetry/>.

[draft-ietf-lisp-name-encoding] Farinacci, D., "LISP Distinguished Name Encoding", September 2022, <https://datatracker.ietf.org/doc/draft-ietf-lisp-name-encoding/>.

[draft-ietf-lisp-pubsub] Rodrigues-Natal, A., Ermagan, V., Cabellos, A., Barkai, S., and M. Boucadair, "Publish/Subscribe Functionality for LISP", June 2021, <https://datatracker.ietf.org/doc/draft-ietf-lisp-pubsub/>.

[draft-kowal-lisp-policy-distribution] Kowal, M., Portoles, M., Jain, A., and D. Farinacci, "LISP Transport for Policy Distribution", September 2022, <https://datatracker.ietf.org/doc/draft-kowal-lisp-policy-distribution/>.

[draft-li-dyncast-architecture] Li, Y., Iannone, L., Trossen, D., Liu, P., and C. Li, "Dynamic-Anycast Architecture", July 2022, <https://datatracker.ietf.org/doc/draft-li-dyncast-architecture/>.

[draft-liu-can-gap-reqs] Liu, P., Jiang, T., Eardley, P., Trossen, D., Li, C., and G. Huang, "Computing-Aware Networking (CAN) Gap Analysis ans Requirements", October 2022,

<https://datatracker.ietf.org/doc/draft-liu-can-ps-usecases/>.

[draft-liu-can-ps-usecases]
Liu, P., Eardley, P., Trossen, D., Boucadair, M., Contreras, LM., Li, C., and Y. Li, "Computing-Aware Networking (CAN) Problem Statement and Use Cases", October 2022, <https://datatracker.ietf.org/doc/draft-liu-can-ps-usecases/>.

[draft-liu-dyncast-ps-usecases]
Liu, P., Eardley, P., Trossen, D., Boucadair, M., Contreras, LM., Li, C., and Y. Li, "Dynamic-Anycast (Dyncast) Use Cases; Problem Statement", July 2022, <https://datatracker.ietf.org/doc/draft-liu-dyncast-ps-usecases/>.

[draft-liu-dyncast-reqs] Liu, P., Jiang, T., Willis, P., Trossen, D., and C. Li, "Dynamic-Anycast (Dyncast) Requirements", January 2022, <https://datatracker.ietf.org/doc/draft-liu-dyncast-reqs/>.

[draft-rodrigueznatal-lisp-multi-tuple-eids]
Rodrigues-Natal, A., Cabellos-Aparicio, A., Barkai, S., Ermagan, V., Lewis, D., Maino, F., and D. Farinacci, "LISP support for Multi-Tuple EIDs", October 2021, <https://datatracker.ietf.org/doc/draft-rodrigueznatal-lisp-multi-tuple-eids/>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997, <https://datatracker.ietf.org/doc/rfc2119/>.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, January 2013, <https://datatracker.ietf.org/doc/rfc6830/>.

[RFC9299] Cabellos, A. and D. Saucez, "An Architectural Introduction to the Locator/ID Separation Protocol (LISP)", October 2022, <https://datatracker.ietf.org/doc/rfc9299/>.

Authors' Addresses

Kyoungjae Sun
ETRI
218, Gajeong-ro, Yuseung-gu
Dajeon
34065
Republic of Korea

Phone: +82 10 3643 5627
Email: kjsun@etri.re.kr

Younghan Kim
Soongsil University
369, Sangdo-ro, Dongjak-gu
Seoul
06978
Republic of Korea

Phone: +82 10 2691 0904
Email: younghak@ssu.ac.kr