DMSC Working Group
Internet-Draft
Intended status: Standards Track
Expires: 30 May 2026

X. Li
A. Wang
W. Wang
China Telecom
D. Kutscher
HKUST(GZ)

26 November 2025

Distributed Micro Service Communication architecture based on Content Semantic
draft-li-dmsc-architecture-01

Abstract

This draft introduces a novel communication architecture, called Distributed Micro Service Communication architecture (DMSC). It includes multiple aspects of microservice communication, such as service registration, service discovery, service routing, service scheduling, and more , Which to achieve all the essential functionalities provided by current centralized service networks. By incorporating content-semantic communication, DMSC significantly enhances the performance, scalability, and reliability of microservice communication. It provides a robust architecture for managing the complex communication requirements of distributed microservices, ensuring data integrity, security, and efficient resource utilization. Furthermore, DMSC provides a reference direction for the transition of the service mesh infrastructure from a location-based model to a content- and service-centric paradigm.

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 30 May 2026.

Table of Contents

1. Introduction

Microservices [Microservices] represent a paradigm for building complex software applications by breaking them down into small, independent, and loosely coupled units called microservices. This architecture improves application flexibility, maintainability, and scalability by enabling each microservice [microservice] to focus on a specific business capability and operate autonomously. With the rise of cloud computing and cloud-native applications, the number of services or components within applications has grown exponentially, posing significant challenges for microservice communication. Ensuring efficient service discovery and interaction in cloud-native environments has become a critical issue.

To address these challenges, service mesh [ServiceMesh] has emerged. Service mesh aims to meet microservice communication requirements, including service registration, discovery, traffic distribution, quality monitoring, and secure communication. However, existing service mesh architectures like Istio [Istio] and Linkerd face architectural limitations (regarding the issues faced by the service mesh, they will be compiled into a separate draft in the future). For instance, the centralized management of sidecars in Istio's control plane (as shown in figure 1) increases update costs and frequency, thereby restricting scalability and flexibility in large-scale distributed systems.

Traditional IP networks are designed around hosts and connections, which limits their ability to address the emerging needs of modern technologies like cloud computing, the Internet of Things (IoT), edge computing, and microservice architectures. These technologies demand a shift from connection-centric design to a content- and service-centric model, enabling intelligent handling of service traffic, dynamic resource allocation, and adaptive routing.

In response to these challenges, this draft proposes a Distributed Micro Service Communication architecture (DMSC) based on content-semantic. The DMSC architecture aims to address the large-scale communication needs of microservices in cloud-native environments while meeting the requirements for service delivery, flexibility, security, and scalability. It also provides a forward-looking design for the the transition of the service mesh infrastructure, supporting its evolution towards a content- and service-centric architecture.

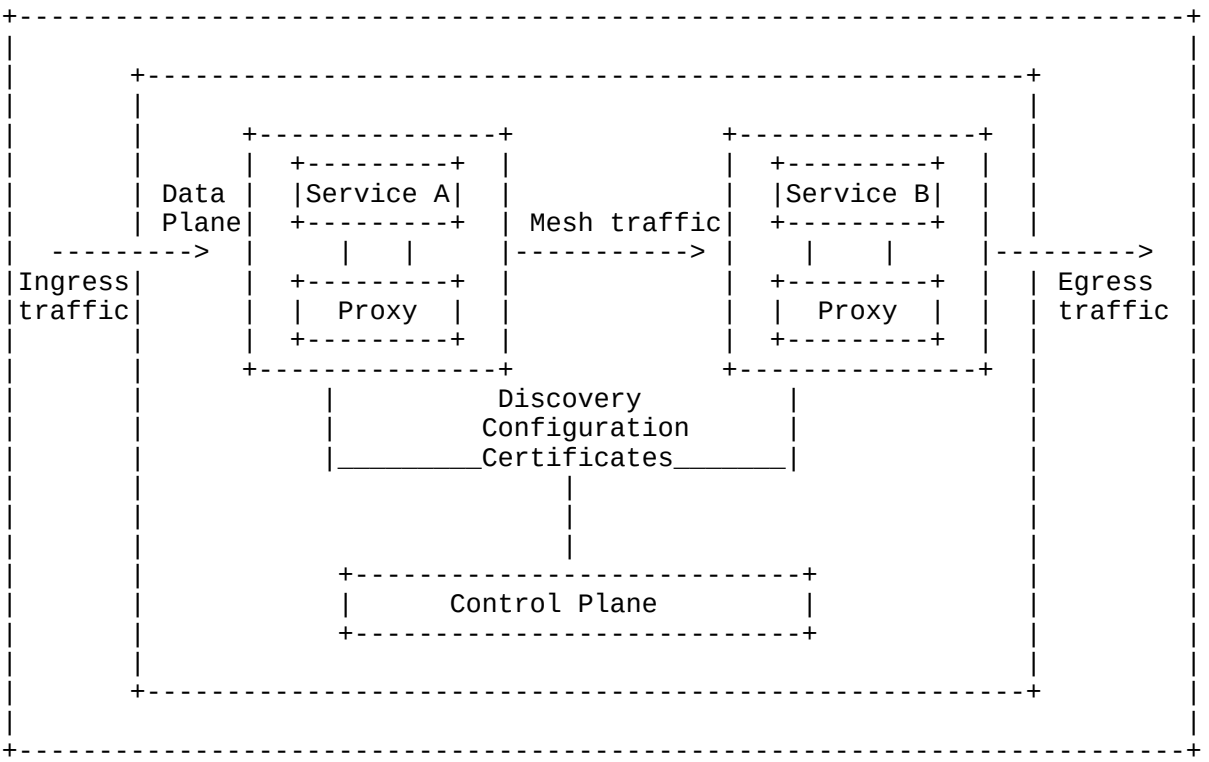

Figure 1 The architecture of Istio

2. Conventions used in this document

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. Terminology

The following terms are defined in this draft:

* DMSC: Distributed Micro Service Communication architecture , a distributed architecture for microservice communication defined in this draft.

* Service: It is a component or microservices in the application.

* Pod: the Pod of Microservices, in the context of microservices, a "Pod" refers to a group or collection of closely related microservices that are co-located and deployed together within a

shared execution environment. Pod is the foundation of all business types, a collection of one or more microservices that share storage, networks, and namespaces, as well as specifications for how to operate, defined in [Istio]

* SG: Service Gateway, it is an important component in the DAMC architecture. It is located between the Pod of Microservices and the entire communication architecture, and is responsible for handling the communication and coordination between Microservices.

* SPA: Service Prefixes Authentication, it is a key component in the DAMC architecture, which is used to verify the validity and validity of the service prefix declared by the Pod to which the Microservices belongs.

* SR: Service Router, it is a network device or component that is responsible for managing and processing the routing and forwarding of communication traffic between Microservices.

* SCSC: Service Mesh Communication Scheduling Center, it is a key component in the DAMC architecture responsible for coordinating and managing the communication within the service mesh.

* FIB: Forwarding Information Base.

* RIB: Routing Information Base.

* LSP: Link State Message LSP (Link State PDUs) is used to exchange link state information.

* LSDB: Link State Database. The state of all links in the network constitutes the link state database.

4. The overview of DMSC

This section provides an overview of the DMSC architecture and an introduction to its key features. The overall architecture of DMSC is shown in Figure 2. It primarily consists of three types of devices shown in Figure 2: Service Gateway (SG), Service Router (SR), and Service Prefix Authentication (SPA) entities,along with a centrally deployed Service Mesh Communication Scheduling Center (SCSC), organized by domain. By integrating content semantics into this architecture, the DMSC architecture aims to overcome the limitations of host-based communication in traditional IP networks and the constraints of centralized control planes in existing service mesh architectures. It uses service prefix and content-based addressing to uniquely identify and locate microservices, so as to

achieve efficient and flexible communication. Service gateway (SG) plays a vital role in service registration, discovery, routing and quality control, while service router promotes the optimal forwarding of communication traffic between microservices. The service prefix authentication (SPA) entity ensures the legitimacy and authorization of the service prefix declared by the microservices, and enhances the security in the distributed microservices environment. In addition, the service mesh communication scheduling center (SCSC) provides effective coordination and allocation of customized forwarding policies between the service gateway and service router (SR) based on the quality detection results reported by SG. This enables intelligent routing decisions to ensure that communication traffic is directed to the most capable and efficient microservices node.

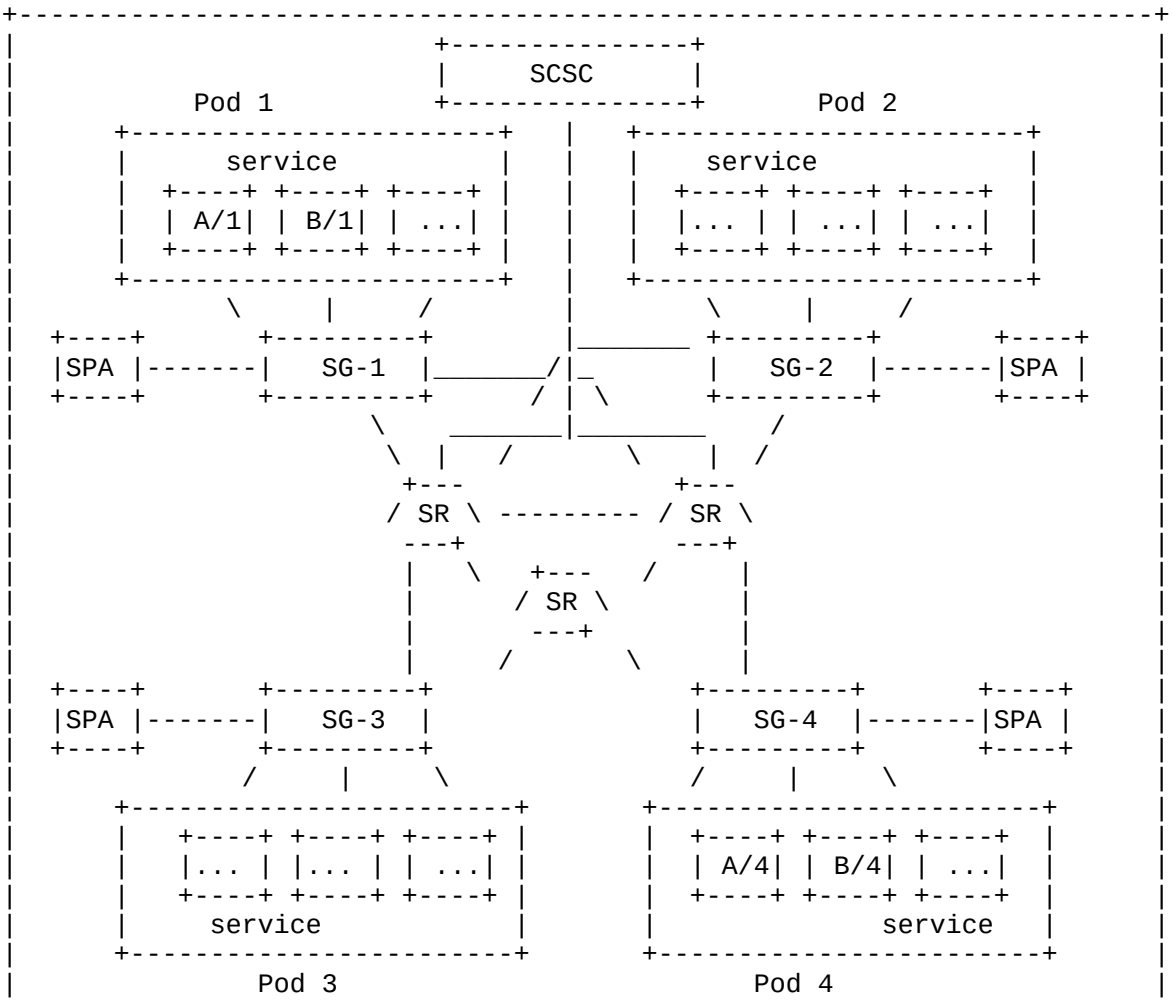

Figure 2 The overall distributed architecture for DMSC

4.1. The control signaling messages design of DMSC

In this draft, we defines a novel communication architecture, called Distributed Micro Service Communication architecture (DMSC). It encompasses all the critical functionalities offered by existing centralized service networks. In the DMSC, we also define multiple communication entities and clearly define their control signaling message types and functions. These communicating entities play a key role in the architecture, ensuring efficient communication between microservices. The DMSC includes the following communication entities:

* Pod

* Service Gateway (SG)

* Service Prefixes Authentication (SPA)

* Service Router (SR)

* Service Mesh Communication Scheduling Center (SCSC)

The types and functions of control signaling messages required for communication between entities are shown in Figure 3. The following is an explanation of each signaling:

* Service Prefix Announcement: Type 1 signaling. This signaling is used in microservices communication. The microservices in Pod notifies the service prefix (namespace) it uses to the connected service gateway (SG). Through this signaling, each Microservices can notify the SG of the service prefix it uses to ensure that the communication between microservices is correctly located and identified.

* Service Prefix LSA (Link State Advertisement): Type 2 signaling. The signaling type exchanged between SG and Service Router (SR), used for advertising service prefixes and topology connection relationships. Through this signaling, SG and SR can share the service prefixes they can reach and their connection relationships, thereby constructing a Link State Database (LSDB) about service prefixes. This provides topology information for routing decisions.

* Service Prefix Authentication: Type 3 signaling. This signaling is used by the Service Gateway (SG) to send authentication requests to the Service Prefix Authentication Entity (SPA). SG requests SPA to verify the legality of the service prefix declared by a Pod. After receiving the authentication request, SPA confirms whether the requested service prefix is legal, thus enhancing the security and reliability in the distributed Microservices environment.

* Service QoS Telemetry/Service QoS Policy: This is the signaling type exchanged between SG, SR and Service Mesh Communication Scheduling Center (SCSC), used to report the communication quality between Microservices and customize the quality of service policy. Through this signaling, SG and SR can automatically detect the quality of target Microservices on a regular basis and report the detection results to the SCSC. Based on these results, SCSC makes decisions to adjust forwarding strategies, optimize traffic routing, ensure that requests are processed quickly and reliably, and maximize the utilization of available resources.

Num	Communication Entities	Control Signaling Message Types	Control Signaling Function
1	Pod/SG	Service Prefixes (Name Space) Announcement	Microservices within each Pod communicate their used Service Prefix (Namespace) to the SG.
2	SG/ SR	Service Prefixes LSA	SG and SR advertise the Service Prefix and topology link relationship they can reach.
3	SG/SPA	Service Prefixes Authentication	The SG authenticates to the SPA requested by the Pod is legal.
4	SG /SR and SCSC	Service QoS Telemetry/Service QoS Policy	Communication quality reporting policies between microservices.

Figure 3 Key communication message types and functions of DMSC

4.2. The key process of DMSC

The key process of the DMSC defined in this draft is as follows:

1) The Pod sends the notification signaling to the service gateway (SG), and the notification signaling is used to notify the service gateway linked to each of the pods of the service identification information. The service identification information is the service prefix or namespace of the pod, and the notification signaling is the first signaling defined in the fourth section of the draft. When the microservices Pod needs to send signaling to the service gateway or other microservices, it can send corresponding packets or messages to the target service gateway or microservices through gRPC calls.

2) The Service Gateway (SG) sends an authentication request to the Service Prefix Authentication entity (SPA) through service prefix authentication signaling, which is used to authenticate whether the service identification information requested by the Pod is legal. If authentication fails, the service prefix authentication (SPA) entity sends an authentication failure notification to the service gateway (SG). If the authentication is passed, perform the operation of sending service prefix LSA signaling from the service gateway (SG) to the service router (SR).

3)Acting as the default gateway for the Pod hosting the service, the Service Gateway (SG) absorbs and forwards all Service traffic emitted from that Service Pod. When any microservices in the microservices pod initiates a request or response, all communication traffic will pass through the service gateway, which is responsible for processing and forwarding it to the target microservices or external service. The service gateway plays a key role in the Microservices Pod. It acts as the entrance and exit of communication between all microservices, ensuring the smoothness and reliability of communication.

3) After the service prefix authentication (SPA) authentication is passed, the service gateway sends service prefix LSA signaling to the service router, which is used to notify the topology link relationship and service identification information under various interfaces of the service gateway linked to the Pod to the service router.

4) The Service Gateway and Service Router (SR) exchange information about the Service Prefixes they can reach and the interconnectivity topology. Based on the preset algorithm and the topological link relationship, each service gateway and each service router obtain a forwarding information base (FIB) that based on the service identification information, forwarding communication packets according to the forwarding information base (FIB).

5) In order to ensure the quality and reliability of communication between microservices, the service gateway needs to understand the health and availability of each target Microservices. To this end, the service gateway will send a detection signaling to each target microservice, requiring it to self detect. Each service gateway receives the detection results reported by the target Microservices, and reports the detection results to the service mesh centralized scheduling center. The Service Gateway and Service Router perform traffic forwarding based on Service Prefixes and the service policies distributed by the Service Mesh Scheduling Center.

DMSC, a distributed architecture for microservices communication, addresses the intricate communication demands among microservices in the future, while encompassing all the critical functionalities offered by existing centralized service networks.

5. The implementation of DMSC’s key functions

Based on the DMSC, the key functions required for communication between microservices are implemented as follows:

1) Service registration

The Pods send the Service identification information to the Service Gateway (SG).The service identification information is the Service Prefix or Namespace owned by the pod. The Service Gateway facilitates proxy registration by utilizing Service Prefix Announcements. This approach allows the Service Gateway to announce the Service Prefixes it can reach, along with relevant information, enabling other components or services to become aware of and register with the appropriate Service Prefix.

2) Service discovery

The Microservices information is authenticated to the service prefix authentication (SPA) through the service gateway (SG). After the successful authentication through the service prefix authentication (SPA), the synchronous distribution of Microservices information is realized between the Service Gateway (SG) and the service router (SR) through the distributed protocol. In this process, the Service Gateway (SG) acts as the central node of authentication to achieve secure and reliable communication between Microservices. The service router (SR) plays a crucial role in facilitating the distribution of these verified Microservices information, so as to achieve efficient routing and seamless interaction between Microservices. In this architecture, Service Prefix Authentication (SPA), Service Gateway (SG) and Service Router (SR) jointly establish a powerful and synchronized system, complete the work of agents in the traditional service mesh architecture, and ensure the integrity and consistency of Microservices communication in the distributed service mesh.

3) Service measurement

The service gateway (SG) detects the target Microservices regularly and automatically according to the demand. The purpose of these probes is to collect information and evaluate the availability of target Microservices. The service gateway (SG) records the detection results and reports them to the Service Mesh Communication Scheduling Center (SCSC). If the detection result does not meet the preset

conditions, the service mesh centralized scheduling center generates forwarding policies based on the detection result, and issues the forwarding policies to the service gateway and service router corresponding to the paths of each target Microservices. The detection results do not meet the preset conditions, including at least one of the following:

* The bandwidth of the corresponding path of the target Microservices is less than or equal to the preset bandwidth threshold.

* The delay of the corresponding path of the target Microservices is greater than or equal to the preset delay threshold.

* The jitter of the corresponding path of the target Microservices is greater than or equal to the preset jitter threshold.

By actively and regularly detecting the target Microservices, the service gateway (SG) ensures the latest and accurate assessment of its status. This proactive approach can identify any potential problems or exceptions, so that timely actions can be taken to ensure the overall reliability and performance of Microservices.

4) Service scheduling

The Service Mesh Communication Scheduling Center (SCSC) utilizes the quality detection results reported by various service gateways (SG) to distribute customized forwarding policies to service gateways (SG) and service routers (SR) on the communication path. These forwarding policies aim to select the optimal service node from the Microservices pool that provides the same function. By distributing these customized forwarding policies throughout the entire service mesh, the system can intelligently route traffic to the most capable and efficient service node. This ensures that requests are directed to Microservices that exhibit excellent performance, responsiveness, and overall quality. The Service Mesh Communication Scheduling Center (SCSC) plays a crucial role in coordinating the distribution of these customized forwarding policies, ensuring effective utilization of available resources by the network and optimizing overall system performance. By making wise decisions based on the quality detection results, the scheduling center enables the service mesh to dynamically adjust and guide traffic to the most appropriate Microservices, enhance the overall user experience, and maximize the utilization of available resources.

Upon receiving requests from clients, the Service Mesh Communication Scheduling Center (SCSC) employs a path selection algorithm, such as Dijkstra’s algorithm, to determine the optimal service path based on

the target microservice and the current network topology. This ensures that requests are routed through the shortest or most efficient path from the client to the destination microservice. In cases where a microservice's performance deteriorates or a service node experiences a failure, the SCSC leverages the quality detection results to adjust the forwarding strategy. By rerouting requests away from the problematic microservice node and redirecting them to a better-performing node, the SCSC ensures that requests are reliably and efficiently handled. Moreover, during periods of high traffic load or network congestion, the SCSC dynamically adapts the forwarding strategy based on both the topology information and quality detection results. This optimization allows the SCSC to efficiently route traffic, ensuring that requests are promptly processed even under challenging network conditions. By synergizing topology information and quality detection results, the SCSC makes intelligent routing decisions and dynamically adapts the forwarding strategy. This approach guarantees that the service mesh's communication flow is efficiently guided and managed, leading to enhanced overall system performance and improved user experience.

In conclusion, the distributed architecture for microservice communication defined in this draft depends on various components, such as service gateway (SG), service router (SR) and Service Mesh Communication Scheduling Center (SCSC). It realizes efficient and reliable communication between Microservices by using service prefix registration, service prefix authentication, quality detection and customized forwarding policies. This architecture overcomes the limitations of traditional centralized methods and provides improved performance, scalability, and reliability. By using these components and mechanisms, the system can effectively manage the complex communication requirements of large-scale Microservices deployment, and ensure the optimal routing of traffic, thus enhancing the overall system performance and user experience.

6. The operation process of DMSC

The distributed architecture for microservices communication consists of the multiple Pods, multiple service gateways, multiple service routers, one Pod linked to one service gateway. The communication process of DMSC is jointly completed by the control plane and the forwarding plane. The control plane is responsible for managing and controlling the policy, routing and security of Microservices communication, while the forwarding plane is responsible for the actual packet forwarding and flow control.

The communication process starts from the control plane, where various components such as service gateway, service prefix authentication, and service router interact through defined signaling

types. For example, the service gateway can send an authentication request to the service prefix authentication to verify whether the Microservices has a legal service prefix. These signaling messages are transmitted in the control plane to ensure Microservices authentication, topology information collection and routing policy distribution. In DMSC, the service gateway and the service router use LSP to build the LSDB and generate the RIB (Routing Information Base) based on the service prefix by routing protocol such as IS-IS. In the data plane, service router downloads the preferred routes in RIB to the FIB (Forwarding Information Base) and forwards data using the FIB.

Once the components in the control plane complete coordination and decision-making, the forwarding plane begins to take effect. When a packet arrives at the service gateway, it selects the best path and the next hop according to the rules in the Forwarding information base (FIB) to forward the packet to the target Microservices. In this way, the service gateway and service router can work together to forward data packets along the optimal path, ensuring efficient transmission and correct routing of traffic.

6.1. Control plane process

1) Service A located within Pod uses the defined type 1 signaling, Service Prefixes (Name Space) Announcement, to notify the connected Service Gateway (SG) of its service prefix or namespace. By adopting this service prefix naming convention, each Microservice can establish its unique identity.

2) When the Service Gateway (SG-1) receives a service prefix notification, it initiates the verification process by sending an authentication request to the Service Prefix Authentication (SPA) unit using a defined signaling type (referred to as type 3 signaling). The purpose of this authentication request is to verify and confirm that Pod indeed has the specified service prefix. By participating in this authentication process, the service gateway ensures that only legitimate Pods with authorized service prefixes are granted access to the network. This robust authentication mechanism adds an additional layer of security and integrity to communication in the service mesh environment.

3) When the service gateway (SG-1) completes the authentication of the service prefix, it will use type 2 signaling to communicate with its connected service router (SR) to announce topology relationships and service prefixes under each interface. By using this specific signaling type, the Service Gateway (SG) transmits information about the entire service mesh topology to the service router, including the connection relationship between the service gateway and the service

router, as well as the service prefix associated with each interface. This notification process ensures that various components in the service mesh can accurately understand the topological relationships between each other, providing an important foundation for subsequent traffic forwarding and service routing. In this way, the service gateway and service router can work together to achieve intelligent traffic management and routing decisions, thus providing efficient and reliable Microservices communication.

4) Other microservices and service gateways will also adopt a similar process for notification. They will communicate with the Service Gateway (SG) and Service Router (SR) using the corresponding signaling types to inform them of their service prefix information and topology relationships. Through this notification process, various components in the entire service mesh can understand each other's existence and functionality, and establish a consistent communication foundation.

5) After the Service Gateway (SG) receives Service Prefix LSA from other components, a Link State Database (LSDB) is generated between the Service Gateway and the Service Router based on the service identification information and the topological link relationship. Then, the service gateway and service router will run SPF algorithm (Shortest Path First) for routing calculation to form the RIB. FIB will be used to guide traffic forwarding and routing decisions, ensuring that each service node can receive and forward traffic according to the optimal path.

6.2. Forwarding plane process

In the DMSC, when Microservices Service A/1 needs to communicate with Service B/4, the communication process is facilitated through the service gateways (SG). The service gateways act as intermediaries responsible for managing and controlling communication traffic in the Microservices architecture.

1) Service A/1 initiates communication by invoking methods on Service B/4 using the remote method invocation (RMI) pattern. The RMI pattern enables seamless interaction between Microservices as if they were co-located within the same process, regardless of their physical location in the distributed environment.

2)The communication message from Service A/1 is directed to the default service gateway (SG-1). The service gateway (SG-1) and service routers utilize the FIB to determine the next destination for the communication message. The FIB contains forwarding rules and microservice prefixes for guiding messages through the network. Packet routing in DMSC is no longer based on traditional IP addresses and ports, but on unique microservice prefixes.

3) The service gateway (SG-1) forwards the communication message to subsequent service routers based on the FIB table’s rules. Each service router in the path takes a step-by-step approach, forwarding the message closer to its destination. Ultimately, the communication message reaches the service gateway (SG-4) associated with the location of Service B/4.

4) Upon receiving the communication message, service gateway (SG-4) processes it based on the information specified in the message, directing it to the appropriate Pod where Service B/4 is located. The same process is applied for backhaul traffic.

Through the above communication process, the DMSC realizes effective cooperation between the control plane and the forwarding plane. The control plane is responsible for managing and controlling the strategy and routing of Microservices communication, while the forwarding plane is responsible for actual packet forwarding and traffic management. This layered architecture enables the system to flexibly adapt to changing communication needs and provide reliable communication services.

7. Conclusion

The Distributed Micro Service Communication architecture (DMSC) addresses the fundamental limitations of traditional IP networks and existing service meshes. Traditional IP networks rely on host-centric communication, which struggles to meet modern demands, while existing service meshes depend on centralized control planes, limiting scalability and flexibility. DMSC is a distributed communication architecture based on four types of communication entities and utilizes content semantics for routing, breaking free from the limitations of traditional IP networks and centralized control planes. This architecture significantly enhances the scalability, performance, and reliability of microservice architectures, catering to the needs of cloud-native environments and large-scale distributed systems. By enabling dynamic service delivery, flexible routing, and efficient resource utilization, DMSC provides an innovative solution for the transition of service mesh infrastructure to a content- and service-centric paradigm.

8. IANA Considerations

TBD

9. Acknowledgement

TBD

10. Contributors

The following people gave substantial contributions to the content of this document and should be considered as coauthors:

Yue Wang

China Telecom

Email: wangy73@chinatelecom.cn

Yiqun Li

China Telecom

Email: liyiq6@chinatelecom.cn

Zhengguang Cui

China Telecom

Email: cuizg@chinatelecom.cn

11. Normative References

[Istio] L, Larsson., "Impact of etcd deployment on kubernetes, istio, and application performance", 2020.

[microservice] A, E., "Guiding architectural decision making on service mesh based microservice architectures", September 2019.

[Microservices] N, D., "Microservices: yesterday, today, and tomorrow", 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[ServiceMesh] Li, W., "Service mesh: Challenges, state of the art, and future research opportunities", 2019.

Authors' Addresses

Xueting Li
China Telecom
Beiqijia Town, Changping District
Beijing
Beijing, 102209
China
Email: lixt2@foxmail.com

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing
Beijing, 102209
China
Email: wangaj3@chinatelecom.cn

Wei Wang
China Telecom
Beiqijia Town, Changping District
Beijing
Beijing, 102209
China
Email: weiwang94@foxmail.com

Dirk Kutscher
HKUST(GZ)
No. 1 Du Xue Road, Nan Sha District
Guangzhou
Guangdong,
China
Email: dku@ust.hk