IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 November 2023, accepted 20 November 2023, date of publication 23 November 2023, date of current version 29 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3335994

RESEARCH ARTICLE

Cloud-Native Service Mesh Readiness for 5G and Beyond

SAIDULU ALDAS[1] **AND ANDREW BABAKIAN**[2]
[1]Intel Corporation, Santa Clara, CA 95054, USA
[2]VMware Inc., Palo Alto, CA 94304, USA

Corresponding author: Andrew Babakian (ababakian@vmware.com)

ABSTRACT The 3GPP has defined the 5G network infrastructure differently from earlier generations, utilizing a Service-Based Architecture (SBA) to accommodate scalability and ensure cost-effectiveness in a cloud and microservices-friendly design. Each 5G Network Function (NF) operates as a service, seamlessly scaling to support an on-demand load and latency requirements. However, this introduces significant complexity to the infrastructure due to an influx of messages across 5G services, making request handling and resiliency challenging. While the 3GPP has provided guidelines and recommendations to address this complexity, it is left to the vendors to develop a working model and design choices within the cloud-native framework. This paper explores the intricate challenges that 5G encounters within the current cloud-native deployment landscapes, with a focus on service-to-service communication, service discovery, and network programmability. It highlights the essential role of Service Mesh in fulfilling 3GPP's vision, proposing enhancements tailored for 5G deployment scenarios. In addition, this work provides architectural guidance for enhancing Service Mesh frameworks to support 5G Core Signalling, extending the Envoy proxy to meet 5G's unique semantic requirements. Finally, this paper sheds light on the future requirements for developing optimal Service Mesh infrastructure that minimizes performance implications while maintaining seamless scalability for cloud-native environments.

INDEX TERMS Cloud-native, 3GPP, 5G core, network functions virtualization, service communication proxy, service mesh, service discovery, identifier system.

I. INTRODUCTION

5G architecture represents a significant step forward in network technology. It intelligently harnesses the vast capacity of the cloud, the dynamic control of Software Defined Networking (SDN), the strategic resource management of Network Function Virtualization (NFV), and the modular strategy intrinsic to microservices architecture [4]. Central to this modular strategy is the concept of a microservice: a self-contained service that fulfills a specific business function that can be independently developed, maintained, and scaled [5]. The 5G architecture can offer scalable services and flexible deployment models by adopting this microservices-based approach combined with the aforementioned technologies.

The associate editor coordinating the review of this manuscript and approving it for publication was Tiago Cruz.

Irrespective of the innovative design, the 5G Core Network Functions (NFs) faced unique challenges in the initial stages of 5G Development, specifically in reliably and efficiently managing service requests. Direct communication between NFs led to significant communication overhead and increased latencies. Furthermore, the system grappled with a lack of load-balancing capabilities, no unified approach to observability, and a complex operational landscape, amplifying the difficulties in achieving seamless network connectivity. The Service Communication Proxy (SCP), introduced as part of 3GPP's release 16 [6], [7], was created as a key component of the 5G Service Based Architecture (SBA) to address these challenges introduced by microservice architectures to provide reliable message delivery through efficient routing, load balancing, and traffic monitoring between NFs.

As defined by the 3GPP standards, the functionality of the SCP primarily provides a guideline and outlines the

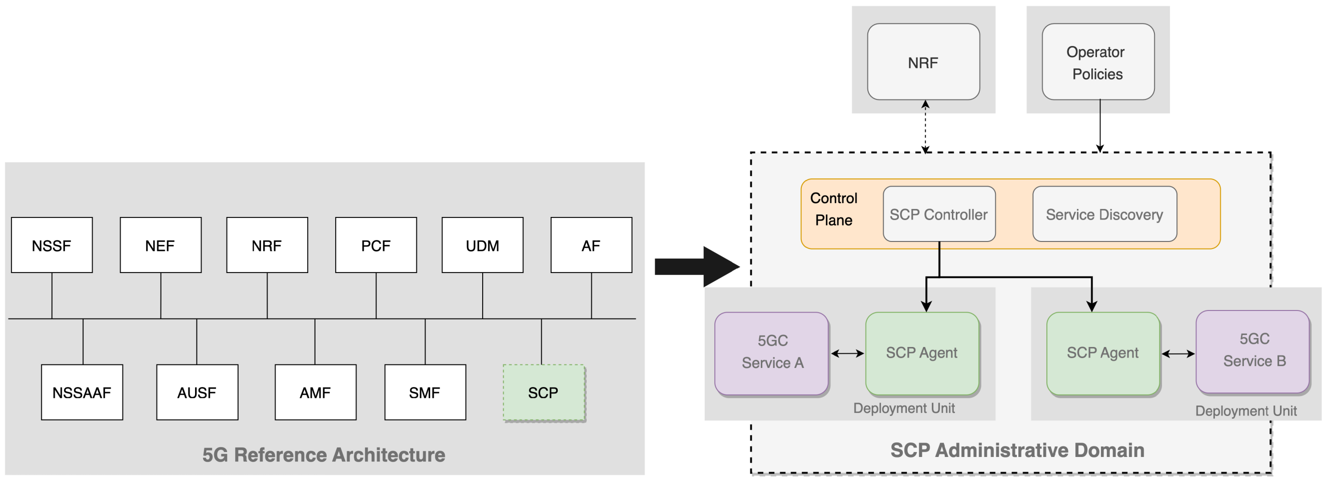

FIGURE 1. **5G control plane reference architecture and the inclusion of SCP.**

protocol definition rather than specifying an implementation for Service Mesh deployments. Instead, vendors are tasked to develop and implement following the 3GPP specification. Service Mesh, a promising technology, offers an avenue to subsume the SCP functionality outlined by 3GPP. By definition, a Service Mesh is a dedicated infrastructure layer for handling service-to-service communication [14]. It is responsible for the reliable delivery of requests through the complex topology of services that comprise a modern, cloud-native application. The deployment model of a Service Mesh is implemented as a collection of proxies deployed alongside an application to handle failure detection and streamline service-to-service communication. Furthermore, deploying 5G services within a Service Mesh enacting as SCP presents unique design considerations that necessitate meticulous evaluation and standardization. These include considerations around 5G signaling routing, service discovery, network programmability, traffic management, observability, and load balancing.

The paper aims to provide architectural guidance on enhancing Service Mesh environments for the efficient execution of 5G Core Signaling functions. This research acknowledges the need for advanced capabilities in service mesh architectures. It outlines the essential extensions within the Service Mesh proxy and control path, particularly in the Envoy, to integrate 5G semantics effectively. By exploring the adaptability of the Envoy proxy for 5G use cases, the study proposes solution scenarios designed to embed 5G routing requirements into the service data path, effectively bridging the gap between traditional Service Mesh functions and the specific demands of 5G signaling. Furthermore, this paper presents a comprehensive outlook on deployment strategies within a Service Mesh infrastructure, offering substantial improvements, facilitating a discussion on various design scenarios, and evaluating the considerations pertinent to each.

Finally, this paper explores the future work integral to the Service Mesh to improve the latency of service communication through Service Discovery and 5G custom message handling. This investigation contributes a fresh perspective to the ongoing discourse on optimizing 5G deployment within Service Mesh infrastructures, thereby setting the stage for future exploration in service-to-service communication in complex 5G architectures.

II. CONSIDERATIONS AND REQUIREMENTS

In a scaled 5GC environment, the challenges with direct communication between NFs include communication overhead and increased latencies. Mitigating this challenge in the 5G space requires delineating the responsibilities between the NF and SCP. This approach entails the NF focusing on the core network functions of the service while the SCP, considered an external service, manages the processing tasks previously handled by the NF. These tasks encompass encryption, load balancing, and circuit breaking, all of which must be carefully considered.

In order to have the Service Mesh assume the role of SCP, modifications are required in the framework to support 5G Core Signaling use cases. Service Mesh deployments primarily cater to traditional enterprise applications, not 3GPP deployments. This discrepancy highlights a gap in meeting specific needs, thus creating additional design considerations for Service Mesh. Attaining optimal scalability and seamless end-to-end performance necessitates specific requirements customized to 5G deployments within the Service Mesh infrastructure. The following section delineates these vital considerations:

1) Optimized Low-Latency Communication: Given the necessity for efficient communication between NFs in 5G deployments, designing and tailoring Service Mesh models explicitly for low-latency operations is essential.
2) Scalability with Performance Maintenance: The Service Mesh model should support the scalability of NFs without negatively impacting performance or increasing latency. This requirement is critical given the inherent communication complexities among 5G NFs.

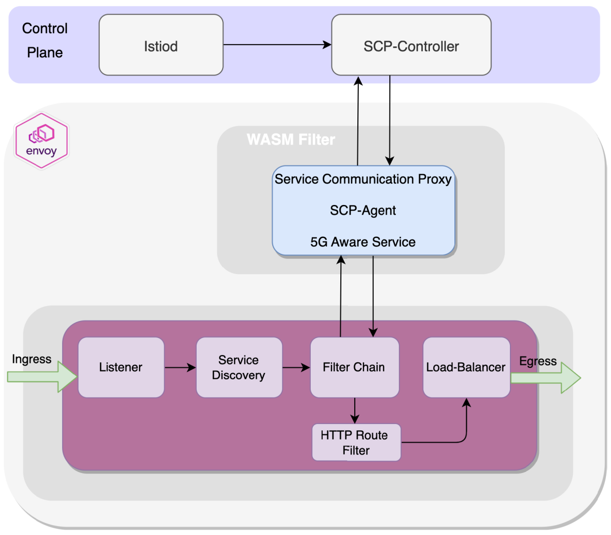

FIGURE 2. **Enhanced envoy architecture.**

3) Efficient Service Discovery: The Service Mesh infrastructure should offer scalable, low-latency service discovery, tailored to the specific needs of 5G NFs, enhancing deployment scalability and reducing end-to-end communication latency.
4) Comprehensive and Unified Observability: The Service Mesh infrastructure should provide 5G-specific and comprehensive infrastructure-wide observability for monitoring and troubleshooting across the system.

Addressing the specified requirements and achieving the targeted Key Performance Indicators (KPIs) requires careful consideration and standardization of various design choices. The following sections explore the high-level solution framework as well as examining these design considerations. Central to this exploration is the enhancement of the Service Mesh architecture, incorporating SCP adaptations to address 5G-specific semantics effectively. Concurrently, these enhancements blend the existing functionalities integral to the Service Mesh. This includes maintaining optimized routing, effective load-balancing, and mutual TLS encryption, thereby ensuring that the core advantages of the Service Mesh are fully leveraged in the 5G context.

III. HIGH-LEVEL SOLUTION

In 3GPP Release-15, the advent of 5GC introduced a Service-Based Architecture (SBA) [1], [9]. In this architecture, NFs interact with the Network Repository Function (NRF), a central repository for all 5G core NFs. This enables all 5G Network NFs to discover other NFs and the services and capabilities each exposes. Subsequent interactions between NFs are carried out directly using the HTTPS/HTTP2 protocol [19] via the Service-Based Interface (SBI).

Due to 5GC's microservices-based design, the communication volume between NFs is anticipated to increase significantly compared to previous generations. Combined with millions of concurrent signaling messages, this architectural model is not conducive to large-scale real-time deployments, as the probability of partial failure grows exponentially with system size [2].

As described in the 3GPP Release-16, the SCP was introduced to address the challenges of extensive signaling traffic among the NFs. Designed to support large deployments, the SCP efficiently manages millions of concurrent transactions, prioritizes signaling traffic, and ensures balanced load distribution across NF instances. Building on this foundation, the SCP architecture, as detailed in 3GPP's Release-16, is composed of two primary components: the SCP Controller and the SCP Agent. Firstly, the SCP Controller is the central control plane, responsible for managing and coordinating the SCP Agents. It also synchronizes with the Network Repository Function (NRF) to obtain the current status of the network functions. The SCP Controller and SCP Agent communicate via an internal channel. The controller manages and distributes configurations to the agents and monitors their health through regular intervals, which involves extracting performance data and monitoring heartbeats from the SCP Agent. If the SCP Controller determines that an SCP Agent has failed, it initiates corrective remediation measures.

Functionally, the SCP Agent operates as a data plane, performing the role of HTTP intermediary between service consumers and service providers. The SCP acts as a mediator or proxy to facilitate communication between NFs. The SCP Agent's primary functions include a load balancer, circuit breaker, encryption, and HTTP router, directing traffic between the service consumer and provider. As illustrated in Figure 1, the SCP Agent interposes the communication path, determining the route for the consumer traffic. It selects an appropriate target based on the consumer's request and directs traffic to the targeted SCP Agent representing the provider. The SCP Controller orchestrates this routing and selection policy, utilizing information sourced from the NRF, and subsequently disseminates routing and service selection directives to the SCP Agents.

The following section will outline the implementation of these SCP guidelines within the Service Mesh architecture to cater to 5G requirements.

A. EXTENDING SERVICE MESH: A CONTRIBUTION TO 5G

The Service Mesh is a dedicated infrastructure to facilitate service-to-service communication within microservice architectures central to the open-source project Istio [12], [18]. This framework consists of two primary components: the control and data plane. The Envoy proxy is central to the data plane and is utilized as a reverse proxy in microservice architectures. In Kubernetes environments, Envoy operates predominantly as a sidecar proxy, mediating all network communication between microservices while managing tasks such as load-balancing, circuit-breaking, performing mutual TLS encryption, and unified observability.

Figure 2 provides an overview of Envoy's data plane operations, illustrating the filter chain where each filter contains a unique role. The listener operates at the socket level, tracking incoming connections. Attached to the listener are the network filters responsible for routing decisions. These filters consist of rules that map virtual hosts to clusters. The metadata within the request, such as the HTTP Host header, is essential to serve as a lookup key in the routing table and a namespace that distinguishes between distinct target instances. The entries in this table determine the destination service to which the request will be forwarded. The load balancer and associated clusters use specific policies to dictate the load-balancing algorithm and select hosts for inclusion in the load-balancing pool. These policies also include retries, timeouts, circuit breakers, and other mechanisms for outlier detection related to service-level objectives.

In Service Mesh implementations, Istio was chosen for this solution for two primary reasons. Firstly, Istio's transition from its traditional role to embrace the new position of SCP is majorly attributed to the extensibility of its data plane, particularly the inclusion of WebAssembly (WASM) [11]. WASM is a general-purpose virtual machine instruction set architecture that provides a bytecode that can be executed on native machine architectures. One of the engines that can interpret and run this bytecode is V8, a high-performance JavaScript and WebAssembly engine written in C++ [11], [15]. Envoy fundamentally embeds a subset of the V8 engine, traditionally used in Chrome and Node.js. This extensibility enables developers to write filters as separate modules in their preferred languages. Once written, Envoy can load these filters and are executed at runtime, which influences the proxy's behavior. For instance, this WASM integration could facilitate the inspection and manipulation of traffic as it flows through the proxy [13]. This approach negates the need for developers to infuse this functionality directly into microservices or be dependent on specific libraries. Instead, the feature is introduced as an inherent part of the sidecar, marking a distinct advantage of embedding SCP functionality into the Service Mesh.

1) INTEGRATION OF SCP WITH ENVOY

From the extensibility of Envoy's integration with WASM, two contributions to the Service Mesh control path architecture have emerged: The inclusion of the SCP Agent and the SCP Controller.

The SCP Agent operates within the Envoy's data plane and is implemented as a WASM filter. This SCP Agent offers three primary functionalities:

1) **Synchronization with the SCP Controller**: The SCP Agent consistently syncs with the SCP Controller. This ensures that policy decisions and routing rules remain current with the latest network dynamics.
2) **Handling of NF Instances Discovery Requests**: When discovery requests are received, the SCP Agent communicates with the SCP Controller, which interacts with the NRF service. The NRF's response data is relayed to the SCP Agent via the SCP Controller. This response typically encompasses Uniform Resource Identifier (URI) [3], application protocol, and service parameters.
3) **Integration with Envoy for Routing and Load Balancing**: Paired with Envoy's existing capabilities, the solution actively monitors downstream HTTP/2 messages. It then leverages 5G header data to determine the routing for requests, ensuring they reach the intended target provider. The pseudocode below illustrates the SCP Agent's request handling function for a detailed understanding of the overall process.

SCP Agent: Request Handling

```
1: for each incoming HTTP request to SCP Agent do
2:     Identify target NF, e.g., AMF, SMF, etc.
3:     if request is for NF discovery and origin is free5gc then
4:         Redirect request to SCP agent for handling
5:     else if request targets a specific NF (other than NRF) and origin is free5gc then
6:         Modify and append 3GPP protocol headers and forward request to the designated target NF
7:     end if
8: end for
```

The SCP Controller is an integral component of the Service Mesh control plane and has been developed as an extension to Istio. Its responsibilities include:

- Registering with the NRF to receive updates about NF details and their operational status.
- Constructing network topologies and formulating dynamic 5G routing policies.
- Regularly updating the data plane with recent policy and routing decisions.
- Discovering available NF Profiles from the NRF and registering its NF Profile.
- Subscribing to notifications from the NRF.
- Conducting health checks and, upon startup or receipt of notifications from the NRF, disseminating available NF Profiles to the SCP Agent.

2) MULTI-CLOUD SUPPORT

In the context of multi-cluster and multi-cloud deployments, the imperative to broaden administrative domains across heterogeneous cloud environments emerges as a critical factor for ensuring secure connectivity among distributed services. VMware's adaptation of Istio addresses this need by enabling the extension of administrative boundaries by implementing a Global Namespace (GNS) [20], [21]. As described in Figure 3, a GNS establishes a unified framework for defining service boundaries spanning multiple cloud environments.

This adaptation to Istio maps Kubernetes namespaces from multiple clusters into a single, unified namespace within the Service Mesh. This abstraction streamlines operations

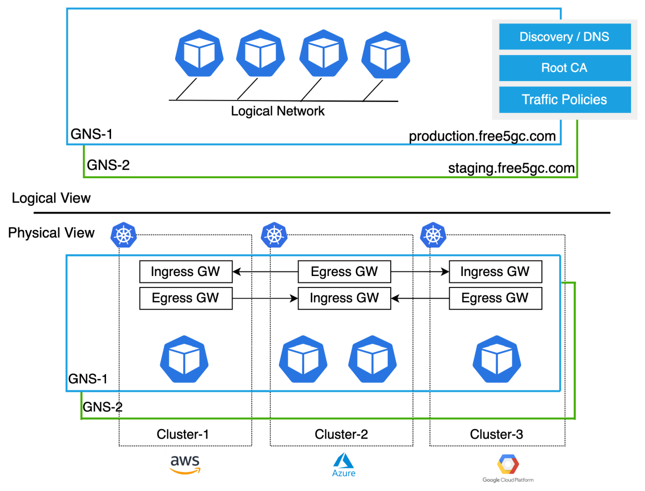

FIGURE 3. **Global namespace overview.**

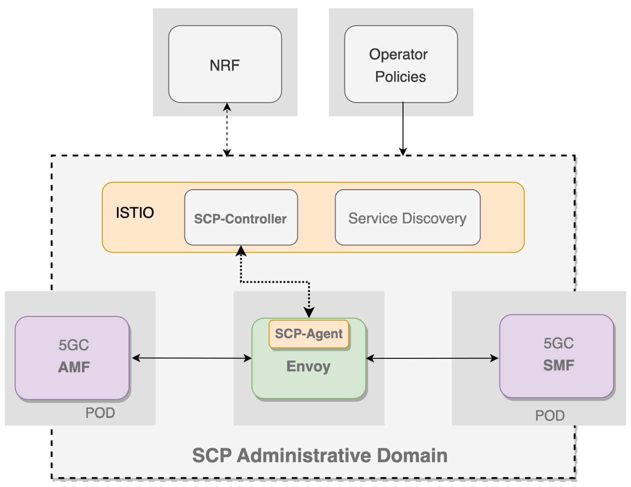

FIGURE 4. **Scenario-2: SCP as Standalone.**

by eliminating the need to manage individual namespaces per cluster, simplifying cross-cluster interactions and policy enforcement.

Within the GNS, each cloud-specific Kubernetes cluster is rendered interchangeable, providing a flexible approach to bind services at the global level and to include or remove Kubernetes clusters as needed. The key features of this configuration likely pertain to the benefits or capabilities introduced by utilizing the GNS, including:

- Certificate Authority Integration: Within a multi-cloud framework, all workloads subscribing to a GNS inherently share a common trust chain, ensuring consistent identity verification and secure inter-service communication via mutual TLS (mTLS), irrespective of their cloud platform.
- Service Discovery and Domain Name System: Within a GNS, scoped identifiers are essential for service discovery and the Domain Name System (DNS) resolution. Services within the GNS are assigned unique identifiers, which ensure that any service discovery requests or DNS queries are strictly contained within the GNS. This confinement of identifiers to the GNS ensures precise service resolution and maintains operational integrity. Furthermore, scoped identifiers within the GNS play an essential role in maintaining isolation boundaries for services spanning diverse cloud platforms.

These combined efforts position Envoy as a valuable complement to the additional features brought by SCP. Detailed elaborations of these enhancements are provided in the design scenarios discussed in Sections III and V. With an observation on the evolving requirements of 5G, Section VI shifts the focus to future work. It details advanced enhancements for Envoy within the Service Mesh model, including the further extension of Envoy filters and service discovery mechanisms. Additionally, this section explores the application of GNS in control-plane slicing, highlighting key areas for ongoing and future research.

IV. INTEGRATING THE ENHANCED ENVOY IN 5G DEPLOYMENT MODELS

Establishing a robust baseline architecture and deployment model is crucial for achieving the requirements and design goals. This section explores different architectural and deployment models that aim to efficiently and scalably meet these requirements. Different SCP deployment and interworking design choices with Service Mesh are discussed to balance scalability and communication latencies in the overall deployment of 5G services. Three distinct scenarios are considered: first, direct communication without the use of SCP; second, discovery through a standalone SCP; and third, employing SCP as an integrated component within a distributed Envoy proxy setup.

Scenario-1: This implies no involvement of proxy or SCP, and all discovery requests are served directly by the NRF. While this model minimizes proxy-induced latency, it introduces substantial communication overhead and fails to leverage the benefits of Service Mesh infrastructure.

Scenario-2: Within the standalone SCP model, each NF is deployed as a microservice within the Service Mesh environment. In this setup, the Envoy filter encapsulates the SCP capability, as shown in Figure 4. Its main functions include managing outbound traffic for NFs and overseeing the service discovery process. In this setup, the control plane extension of the SCP Controller is tasked with maintaining an up-to-date representation of the core network's state. It provides essential routing and discovery information to the SCP Agent via the eXtensible Discovery Service (XDS) API, a protocol Envoy uses for dynamic service discovery and configuration.

All NFs, including the SCP Agent, are registered with the NRF. Uniquely, the SCP Agent subscribes to notifications about the status alterations of NFs, including events such as registration, updates, and deregistration. This mechanism allows the SCP to oversee all discovery and routing requests. Upon receiving a request from an NF consumer, the SCP's role is to conduct service discovery and direct the request to the appropriate consumer.

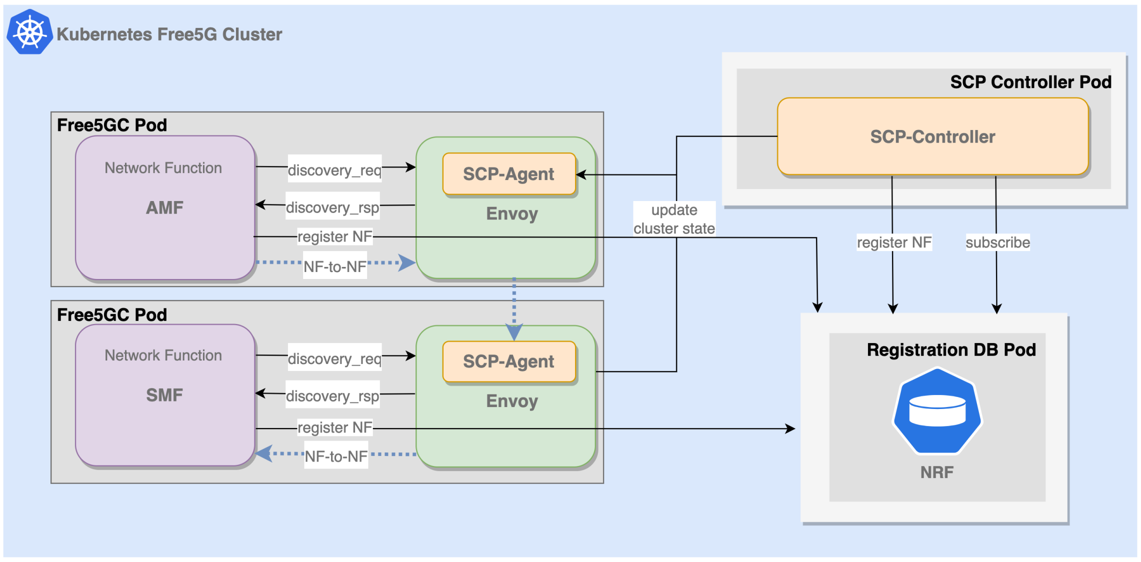

FIGURE 5. Scenario-3: SCP as distributed envoy proxy.

The embedding of the SCP as an Envoy filter within the Service Mesh infrastructure brings many advantages forward. This includes secure inter-service communication and observability without necessitating additional implementation in the SCP or NFs. Realizing the standalone SCP model leverages two pivotal components: a control plane extension that tracks the network state and a data plane or Envoy extension responsible for executing 5G-specific service discovery, routing, and visibility.

Scenario-3: In the SCP as a distributed Envoy proxy model, SCP is deployed as a sidecar within the same pod as the NFs, facilitating in-pod service discovery and routing within the microservice pod. As illustrated in Figure 5, NFs, such as the AMF (Access and Mobility Management Function) and SMF (Session Management Function), along with SCP Controllers, register with the NRF. Additionally, the SCP Controller subscribes to NRF events to keep the state and information about the NFs updated. A source NF, such as the AMF, sends an HTTP signaling message for communication. This message contains a header field named '3gpp-Sbi-Target-apiRoot' with the target URI value set to 'SMF,' indicating the destination NF, which is the SMF. This communication is facilitated by Envoy proxies present on both the source and destination sides. The source SCP Agent, which operates as part of the AMF pod, interprets the incoming request. It then conducts routing discovery and forwards the request to the appropriate SMF instance. Conversely, the SCP Agent within the SMF Pod listens for incoming HTTP messages. Upon receiving a message, it parses the content, verifies the necessary 3GPP message semantics, and then delivers the message to the SMF.

This architecture's core is the SCP WASM filter, as detailed in Section III. It is implemented as an essential Envoy filter, working with the SCP Controller. The SCP Controller is critical in this architecture, as it continually monitors the state of the network and updates the SCP Agent about any changes through the XDS API. By integrating the SCP's functionalities into the Envoy sidecar, the SCP Agent can effectively manage outbound traffic from the NF and efficiently handle the service discovery process. This approach offers a scalable solution that significantly reduces additional latency, thus enhancing communication efficiency.

V. DESIGN CHOICES AND PERFORMANCE CHARACTERIZATION

Each of the design scenarios mentioned earlier presents its own set of advantages and disadvantages:

1) **Scenario-1: Direct Communication Without SCP** - This deployment model is straightforward but lacks the advantages of a Service Mesh. It is characterized by significant communication overhead and limited scalability. In this scenario, each request from the source NF involves a multi-step process for service discovery. Firstly, the source NF sends a request to the NRF to obtain discovery information. Upon receiving the discovery response from the NRF, the source NF sends the communication request to the destination NF. This process introduces multiple additional requests due to the interaction with the NRF, which is hosted on a separate machine in most deployments. This additional layer of communication complicates the process and contributes to the increased overhead and reduced scalability.
2) **Scenario-2: Standalone SCP** - This model benefits from Service Mesh infrastructure; the deployment model is scalable. The SCP is deployed as a separate microservice in the Service Mesh environment with a centralized proxy, i.e., Envoy. The SCP functionality is implemented as a virtual container service that controls outbound traffic and handles discovery requests. In this

scenario, the source NF requests the standalone SCP. The SCP then performs service discovery and forwards the request to the appropriate destination NF instance. Notably, in this configuration, the SCP may be running on a separate machine, which could have implications for the routing and latency of network communications.

3) **Scenario-3: SCP as Distributed Envoy Proxy** - This model also benefits from Service Mesh infrastructure and is scalable. The SCP is deployed as a WASM filter within the Envoy sidecar, sharing a pod with the NF. The SCP handles both discovery requests and communication. In this scenario, the communication request from the source to the destination NF is intercepted and processed by the SCP Agent. The SCP Agent interprets the service discovery information and directs the request to the appropriate destination NF. The key distinction between scenarios two and three lies in the location and mechanism of discovery realization: in Scenario-3, all discovery processes are executed within the pod itself, resulting in minimal latency. In contrast, Scenario-2 involves the request traversing the network to a separate physical machine hosting the SCP agent, potentially requiring one or more network hops, which could significantly increase latencies compared to Scenario-3.

Systematic experiments were conducted to evaluate the deployment models' effectiveness. These aim to measure the discovery and communication latencies across NFs in the three scenarios. The experimental setup involved the following:

- **Workload:**
  Free5GC [8] was deployed to emulate a comprehensive suite of 5G core network functions (NFs). This included the Access and Mobility Management Function (AMF), Session Management Function (SMF), Policy Control Function (PCF), Authentication Server Function (AUSF), Network Slice Selection Function (NSSF), User Plane Function (UPF), Unified Data Management (UDM), Unified Data Repository (UDR), and the supporting MongoDB database. As an open-source project compliant with 3GPP Release 15 and beyond, Free5GC served as an effective solution for the core network in our simulations. It accurately mirrored end-to-end core NF communication scenarios. The choice of Free5GC for the experiments was driven by its comprehensive capabilities as an open-source solution for simulating 5G core workloads, filling the gap in the absence of commercial options.
- **Simulation:** UERANSIM, acting as a RAN and UE simulator, was employed as a load generator to benchmark the latencies of control path intercommunication within the 5G core network. Deployed on one of the servers, UERANSIM was instrumental in simulating network activities such as registration, deregistration, and PDN session establishment. Specifically, the deregistration procedure was initiated using the UERANSIM to generate SCTP messages towards the Access and Mobility Function (AMF), which in turn communicated with other NFs over the Service-Based Interface (SBI) using HTTP/2 protocols, thereby facilitating a comprehensive end-to-end procedural test.
- **Service Mesh infrastructure:** Istio and Envoy were integral in establishing the Service Mesh environment. Istio acted as the control plane, while Envoy served as the data plane, equipped with 5G-specific extensions described in Section III and IV to manage the unique communication requirements of the 5G Core network. The experimental framework utilized a single Kubernetes (k8s) cluster to orchestrate the deployment of network functions, with Docker serving as the container runtime. Each NF was housed in its container, with an Envoy proxy container co-existing within the same pod to enable Service Mesh communication features specifically for Scenario-3.
- **Hardware Configuration:** The tests ran on a hardware setup that included two servers powered by 3rd Gen Intel® Xeon® Scalable Processors, connected through a 100G networking fabric to enable the high-speed data transfer required for 5G operations.

Benchmark data:
The benchmarking process involved triggering the deregistration procedure using UERANSIM, which generated SCTP messages directed towards the AMF. Subsequently, AMF communicated with other NFs, as described in the workload section, over the Service-Based Interface (SBI) using HTTP/2 protocols. This setup facilitated a comprehensive end-to-end procedural test. The focus of latency measurements was between two specific network functions: the UDM and UDR. These NFs were selected because their interactions with the database are anticipated to be more latency-oriented in the overall test scenario, given the nature of data management and repository functions.

TABLE 1. Comparisons of deployment scenarios.

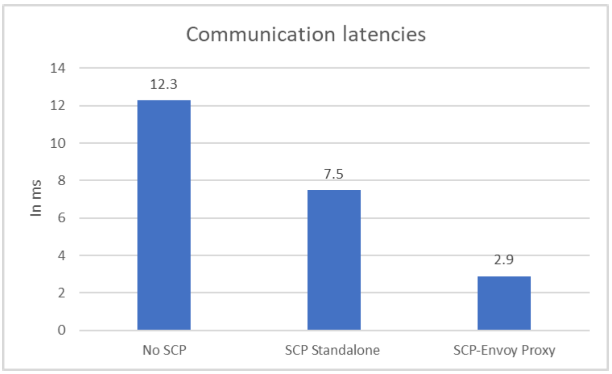

Table 1 illustrates the communication latencies observed across three deployment models: without SCP, with standalone SCP, and with SCP as a distributed Envoy proxy. The analysis reveals that Scenario-2, which employs a centralized SCP, exhibits higher latencies than Scenario-3, where SCP is embedded within the NF pod, directly facilitating discovery and communication. Both scenarios demonstrate their potential for scalable and low-latency communication within a 5G context.

In Scenario-2, although scalability is emphasized, it necessitates augmented allocation of resources, specifically in terms of memory and computation. However, this scenario fails to attain the reduced latency levels observed in Scenario-3, where the SCP is seamlessly integrated within each NF pod.

Refinements in the experimental approach have led to notable latency improvements. Particularly, the integration of the SCP with the distributed Envoy proxy, as detailed in Scenario-3 of Section IV, has been instrumental in achieving these latency reductions. Embedding the SCP within the NF pod and utilizing Envoy's advanced routing capabilities, Scenario-3 meets the sub-millisecond latency benchmarks crucial for high data rate 5G communications. This performance is consistent with the stringent requirements specified in 3GPP TS 22.261 Section 6.4.2.1 and Annex A [17], reflecting the standard's emphasis on resource-use efficiency for both control and user planes and supporting the targeted latencies for 5G Core Signaling networks.

VI. FUTURE PERSPECTIVES AND NECESSARY DEVELOPMENTS

The 5G cloud-native architecture is undergoing continuous evolution, and integrating an enhanced Envoy proxy brings several key areas into the spotlight for future investigation. Among these, service discovery is integral to achieving efficient routing, effective load balancing, and low latency. The following subsections will focus on exploring avenues for future work. This includes addressing the challenges associated with service discovery, emphasizing the integration of custom 5G message handling, and limiting environments specifically to the control-plane network slice.

A Service Mesh solution maintains records and meta-information about the services and endpoints in a cluster to handle routing, policy management, and load balancing. For instance, Istio utilizes its service registry and discovery system, which populates from the platform discovery system. Furthermore, the Envoy proxy, responsible for routing decisions, communicates with the Istio control plane through the XDS API to acquire discovery and routing details. The Envoy service discovery obtains DNS information on the endpoints and services. However, to support 5G-specific requirements, the discovery and routing mechanisms must be enhanced to accommodate 5G routing and load balancing scenarios adequately. For instance, consider the following scenarios, which offer limited literature examples related to 5G service discovery:

Network Slicing-Specific Discovery: Network slicing enables optimal use of network resources by employing virtualization technologies on shared physical infrastructure [16]. The Envoy SCP architecture must consider the service discovery semantics of singling messages related to specific network slices (NSSAI) to satisfy the spec requirements and achieve optimal routing of messages to the destination/producer NF.

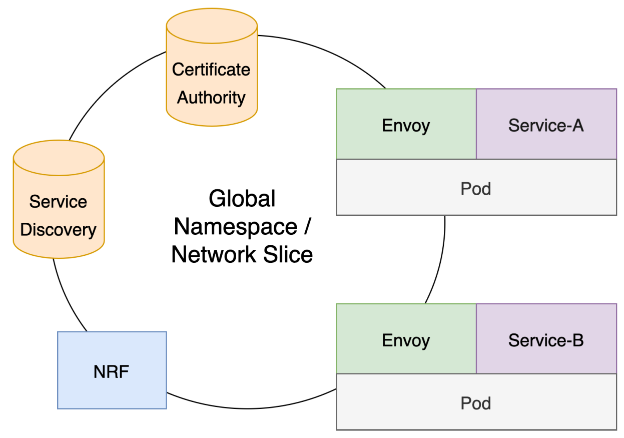

FIGURE 6. Global namespace per network slice.

The Global Namespace (GNS), as illustrated in Figure 6, could enhance the network in governing connectivity in VMware's adaptation of Istio [20]. As described in Section III, Global Namespace (GNS) is a logical abstraction that establishes an application boundary spanning multiple environments and clouds. Within the Service Mesh framework, once Envoy becomes a member of a particular GNS, it is confined to the scope of that GNS, essentially its administrative boundary.

In the examined scenarios, where the control plane is segmented, service discovery query names and certificate authorities are confined within the boundaries of specific Global Namespaces. This arrangement ensures authenticated and secure communication within each network slice. While deploying a dedicated NRF for each slice is feasible, particularly in situations involving distinct tenancy requirements, such as those between independent operators or in distinguishing staging from production environments, a more practical approach is identified. This approach involves the utilization of a common NRF that serves multiple slices. The NRF functions as an area border function in such a framework, intersecting with the GNS boundaries. This intersection enables the NRF to discriminate access based on subscriptions effectively. This method aligns with maintaining efficient and secure network slicing, providing a controlled yet flexible environment within the 5G architecture.

Domain-Specific Discovery Semantics: As part of the discovery process, the Envoy SCP Agent must consider 5G-specific semantics in the discovery mechanisms. This includes rejecting consumer requests with appropriate 3GPP-specific error codes if no NF Service Producer supports the requested features. To address these needs, the conventional service discovery method must shift from providing a universal answer to one tailored for each consumer. This entails customizing responses for reasons such as access control, policy, or to optimize the connection between the consumer and provider. A strategy involves enhancing queries with meta-information, enabling clients to exchange their details for more accurate service information [2]. The service discovery resolution can then fine-tune its answers, offer improved estimations, or even decline connections, as illustrated.

By enforcing identifiers to behave uniquely based on the querying entity, the resolution system could enable the implementation of access controls that adhere to domain-specific discovery semantics. In this context, service discovery is set to transition from a mere dictionary-based search to a system that enriches queries with added details, allowing personalized responses based on the queriers' identity and metadata.

Low-Latency and Highly Dynamic Service Discovery Mechanism to Support Time-Sensitive Communication: One key challenge in any 5G deployment in the cloud is providing low-latency communication infrastructure to meet 5G KPI requirements [10]. The discovery mechanism must maintain a repository of parameters related to the destination NF reachability latencies and load.

Enhancing the service discovery system to provide more than just IP necessitates a shift from modeling hosts to services. Services demand additional details than just IP, including port and protocol data. This also involves understanding the precise needs and intentions of the service consumer's request. A client and service discovery broker exchanging information could streamline the search results, enhancing performance in establishing connections without additional steps to gather connection information. For instance, in the context of the upcoming 6G, the service discovery could supply connection profiles hinting at protocol support (e.g., HTTP/3 for 6G) alongside port details [22]. This ensures fast connections without latency during service negotiations.

Furthermore, application namespaces could be included as a parameter in the Service Discovery resolution. As NFs consolidate onto a shared platform, name-based routing distinguishes application sessions on a common host. Within this discovery process, the subscriber service may populate the name within the application protocol during the initial connection with the provider service. 3GPP's release 16 also suggests that a name-based method is another viable approach for discovery and routing [6], [7].

Furthermore, Service Discovery systems often direct clients based on configured priorities and weights to enhance connection speeds and address time sensitivity. This method, aimed at evenly distributing traffic, might not always account for the queriers' preference for faster response times. This mismatch can conflict between the server's choice of an 'optimal' service and the client's performance needs. Instead of providing a single answer, the service discovery system could offer multiple candidate answers, allowing the client to choose based on specific needs. This adaptable method aligns closely with consumer interests, enabling a tailored service selection and potential for further optimization. While client-side resolution computation is often less favorable, a possible implementation with Envoy could handle this task, ensuring no added burden on the NF. Envoy could facilitate the logic to identify the best candidate through a series of synthetic transactions on behalf of the consumer to determine the optimal target. This approach represents another promising direction for service discovery evolution.

Custom 5G Message Handling:

In this framework designed for the SCP architecture, provisions have been made for future extensions in custom message handling. A key focus is integrating the desired 3GPP functionalities, leveraging custom headers. This enhancement empowers Envoy to efficiently process 5G signaling messages encapsulated in HTTP headers and apply transformation rules accordingly. Such advanced processing is crucial in guiding informed routing decisions. The framework's advanced capabilities are crucial in making informed choices about data routing. Moreover, they ensure that vital information is reliably sent to the main controlling entity, which helps maintain a consistent state for all the NFs across the system.

For 5G aware load balancing, immediate enhancements warrant consideration. The framework has considered the support of custom messages, notably the **3gpp-Sbi-Oci**. This message is critical for indicating overload control notifications and receiving responses from consumers and providers. The load balancer can harness this data as a component of its algorithm, potentially selecting alternate service candidates for request routing. Additionally, the framework accommodates the **3gpp-Sbi-Message-Priority**, a custom message vital for signaling when a specific NF necessitates traffic prioritization over other NFs. This can be used to provide congestion control. By synergistically leveraging these messages, there is an opportunity to elevate performance and reduce latency within the Service Mesh design.

These concepts improve how we perceive service discovery, routing, and load balancing. As the paper outlines, the Service Mesh is the foundational infrastructure layer, facilitating a programmable data plane. This not only establishes a foundation for a system that's more agile and nuanced but also sets the stage for future explorations. In addition to the contributions mentioned, efforts will persist in improving and refining the platform, ensuring it is primed and 5G-ready. This enhanced system is uniquely positioned to cater to the needs of both providers and consumers in cloud-native 5G environments.

VII. CONCLUSION

In conclusion, the 5G Service-Based Architecture (SBA) has facilitated the cloudification of the 5G network and deployments, but not without its inherent challenges. The existing cloud infrastructure, designed predominantly to run general applications, faces difficulties when hosting specific 5G workloads, such as 5G core NFs. This situation leads to sub-optimal utilization of cloud resources, fragmented deployment models, and intensive resource consumption to fulfill the rigorous SLAs and KPIs.

The enhancement of the Envoy proxy to become 5G-aware offers a tailored solution, augmenting the service communication proxy mediating between NFs, as evidenced by the deployment strategies delineated in this article. The ongoing evolution of the Service Mesh infrastructure uncovers a promising path toward addressing the distinctive requirements of 5G as the industry advances towards 6G. It shows a future marked by more flexible, scalable, and dependable network architectures. In the near term, service discovery support will be essential for discoverability and ensuring the integrity of the communication system, especially when addressing the multifaceted challenges of the identifier system and transactional integrity across administrative domains.

With the potential extensibility of Envoy, there is an opportunity to weave innovations such as AI/ML, network efficiency, and cost reduction that align with the outlines of 6G specifications. These advancements in Envoy could play a pivotal role in shaping the next generation of network architectures. This progression also positions the Service Mesh to serve as a conduit for a seamless transition from 5G to 6G networks.

The hope is that these insights contribute to the ongoing development within the telecommunications industry, offering a perspective that may assist in realizing the significant potential of next-generation networking technologies. This work aspires to set the stage for a future characterized by enhanced connectivity, adaptability, and efficiency.

ACKNOWLEDGMENT

The authors would like to thank Intel and VMware for their partnership in the Go Big Multi-Cloud Networking Initiative. This joint venture has been central to the research and development process, leading to notable advancements in the service mesh project for 5G technologies. This collaboration fostered innovation and shared expertise in shaping the future of telecommunications. They also appreciate the engineers, researchers, and support staff involved in this project.

(Saidulu Aldas and Andrew Babakian contributed equally to this work.)

REFERENCES

[1] 3GPP, "System architecture for the 5G system," 3rd Gener. Partnership Project (3GPP), Sophia Antipolis, France, Tech. Rep. TR 23.501, Dec. 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/16.06.00_60/ts_123501v160600p.pdf

[2] A. Babakian, P. Monclus, R. Braun, and J. Lipman, "A retrospective on workload identifiers: From data center to cloud-native networks," *IEEE Access*, vol. 10, pp. 105518–105527, 2022.

[3] T. Berners-Lee, R. T. Fielding, and L. M. Masinter, "Uniform resource identifier (URI): Generic syntax," RFC Editor, RFC 3986, Jan. 2005, p. 61. [Online]. Available: https://www.rfc-editor.org/info/rfc3986, doi: 10.17487/RFC3986.

[4] B. Dab, I. Fajjari, M. Rohon, C. Auboin, and A. Diquélou, "Cloud-native service function chaining for 5G based on network service mesh," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–7.

[5] N. Dragoni, I. Lanese, S. T. Larsen, M. Mazzara, R. Mustafin, and L. Safina, "Microservices: How to make your application scale," in *Perspectives of System Informatics: 11th International Andrei P. Ershov Informatics Conference, PSI 2017*, Moscow, Russia, Cham, Switzerland: Springer, Jun. 2018, pp. 95–104.

[6] ETSI, "5G; 5G system; technical realization of service based architecture; stage 3," ETSI, Sophia Antipolis, France, Tech. Rep. 129 500, Apr. 2021. [Online]. Available: https://www.etsi.org/deliver/etsits/129500129599/129500/16.07.0060/ts129500v160700p.pdf

[7] ETSI, "5G; procedures for the 5G system (5Gs)," ETSI, Sophia Antipolis, France, Tech. Rep. 23 502, Jul. 2021. [Online]. Available: https://www.etsi.org/deliver/etsits/123500123599/123502/16.09.0060/ts123502v160900p.pdf

[8] (2019). *Free5GC*. [Online]. Available: https://free5gc.org/

[9] A. Ghosh, A. Maeder, M. Baker, and D. Chandramouli, "5G evolution: A view on 5G cellular technology beyond 3GPP release 15," *IEEE Access*, vol. 7, pp. 127639–127651, 2019.

[10] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 55–61, Mar. 2020.

[11] A. Haas, A. Rossberg, D. L. Schuff, B. L. Titzer, M. Holman, D. Gohman, L. Wagner, A. Zakai, and J. Bastien, "Bringing the web up to speed with webassembly," in *Proc. 38th ACM SIGPLAN Conf. Program. Lang. Design Implement.*, 2017, pp. 185–200.

[12] (Aug. 2016). *ISTIO*. [Online]. Available: https://www.istio.io

[13] S. M. Jain and S. M. Jain, "Extending ISTIO with webassembly," in *WebAssembly for Cloud: A Basic Guide for Wasm-Based Cloud Apps*. Berlin, Germany: Springer, 2022, pp. 151–160.

[14] W. Li, Y. Lemieux, J. Gao, Z. Zhao, and Y. Han, "Service mesh: Challenges, state of the art, and future research opportunities," in *Proc. IEEE Int. Conf. Service-Oriented Syst. Eng. (SOSE)*, Apr. 2019, pp. 122–1225.

[15] J. Ménétrey, M. Pasin, P. Felber, and V. Schiavoni, "Webassembly as a common layer for the cloud-edge continuum," in *Proc. 2nd Workshop Flexible Resource Appl. Manage. Edge*, 2022, pp. 3–8.

[16] K. Samdanis, A. Prasad, M. Chen, and K. Hwang, "Enabling 5G verticals and services through network softwarization and slicing," *IEEE Commun. Standards Mag.*, vol. 2, no. 1, pp. 20–21, Mar. 2018.

[17] 3GPP, "Service requirements for the 5G system; stage 1 (release 17)," 3rd Generation Partnership Project (3GPP), Sophia Antipolis, France, Tech. Specification 22.261, Sep. 2022.

[18] O. Sheikh, S. Dikaleh, D. Mistry, D. Pape, and C. Felix, "Modernize digital applications with microservices management using the istio service mesh," in *Proc. 28th Annu. Int. Conf. Comput. Sci. Softw. Eng.*, 2018, pp. 359–360.

[19] M. Thomson and C. Benfield, *HTTP/2*, document RFC 9113, Jun. 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9113

[20] VMware. *Vmware Tanzu Service Mesh Documentation—Global Namespaces*. Accessed: Nov. 13, 2023. [Online]. Available: https://docs.vmware.com/en/VMware-Tanzu-Service-Mesh/services/concepts-guide/GUID-9E3F1F90-4310-415B-98C8-C06E59B8A5EE.html

[21] *Vmware Tanzu Service Mesh Usage Guide*. Accessed: Nov. 13, 2023. [Online]. Available: https://docs.vmware.com/en/VMware-Tanzu-Service-Mesh/services/using-tanzu-service-mesh-guide/GUID-8D483355-6F58-4AAD-9EAF-3B8E0A87B474.html

[22] J. Zirngibl, P. Sattler, and G. Carle, "A first look at SVCB and HTTPS DNS resource records in the wild," in *Proc. Int. Workshop Traffic Meas. Cybersecurity*, 2023, pp. 470–474.

SAIDULU ALDAS is currently a Principal Engineer and a Lead Software Architect with the Software and Advanced Technology Group (SATG), Intel Corporation. He is a more than 20 years veteran in wireless, cloud networking, and network virtualization technologies. He has a deep understanding of various technologies and strong business acumen to lead and transform innovation into customer-winning products.

ANDREW BABAKIAN is currently pursuing the Ph.D. degree in engineering with the University of Technology Sydney, Australia. He is also a Technologist and the Senior Director of the Modern Application Platform Business Unit, VMware, driving innovation efforts in software-defined networking and identity technologies for cloud-native architectures. His current research interests include naming systems and the evolution of internet identifiers.

• • •