Workgroup: Network Working Group
Internet-Draft:
draft-mendes-rtgwg-rosa-use-cases-01
Published: 9 July 2023
Intended Status: Standards Track
Expires: 10 January 2024
Authors: P. Mendes J. Finkhaeuser LM. Contreras
Airbus Interpeer gUG Telefonica
D. Trossen
Huawei Technologies

Use Cases and Problem Statement for Routing on Service Addresses

Abstract

The proliferation of virtualization, microservices, and serverless architectures has made the deployment of services possible in more than one network location, alongside long practised replication within single network locations, such as within a CDN datacentre. This necessitates the potential need to coordinate the steering of (client-initiated) traffic towards different services and their deployed instances across the network.

The term 'service-based routing' (SBR) captures the set of mechanisms for said traffic steering, positioned as an anycast problem, in that it requires the selection of one of the possibly many choices for service execution at the very start of a service transaction, followed by the transfer of packets to that chosen service endpoint.

This document provides typical scenarios for service-based routing, particularly for which a more dynamic and efficient (in terms of both latency and signalling overhead) selection of suitable service execution endpoints would not exhibit the overheads and thus latency penalties experienced with existing explicit discovery methods. Related drafts introduce the design for an in-band service discovery method instead, named Routing on Service Addresses (ROSA), based on the insights from the use case and problem discussion in this draft.

Status of This Memo

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 10 January 2024.

Table of Contents

1. Introduction

Service provisioning in recent years has been largely driven by two trends. Firstly, virtualization has enabled service provisioning in

more than one network location, progressing from virtual machines to containers, thus enabling sub-second service execution availability. Secondly, the cloud-native paradigm postulates agile development and integration of code, decomposing applications into smaller micro-services, to be deployed and scaled independently, yet chained towards a larger common objective. Micro-service deployment may be done following a serverless model where a third-party provider allocates resources to the micro-services in different network locations when an application is triggered and re-assigning them elsewhere when the application is no longer active. Such deployment flexibility allows to bring services 'closer' to consumers, but also poses challenges such as the need for a service discovery and selection process that aligns with the needed dynamicity in selecting suitable service endpoints, with a particular emphasis on minimizing the latency from the initiating client request to the actual service response.

Service-level communication, captured through the term 'service-based routing' (SBR) throughout this document, has been realized with a decades-old DNS-based model to map service domains onto one of a set of IP addresses, often based on load or geo-information. Those IP addresses and port assignments identify network interfaces and sockets for service access. Contrasting against the aforementioned trends of evolved resource availability, deployment flexibility and location independence, those assignments typically remain static.

We recognize that the Internet community has developed solutions to cope with the limitations of the DNS+IP model, such as Global Server Load Balancing (GSLB) [GSLB], DNS over HTTPS [RFC8484], HTTP indirection [RFC7231] or, more recently, at transport level through QUIC-LB [I-D.ietf-quic-load-balancers]. At the routing level, [TIES2021] outlines a solution to map URL-based services onto a small set of IP addresses, utilizing virtual hosting techniques at the incoming Point-Of-Presence (PoP) to suitably distribute the request to the computational resource that may serve it. However, such solutions compound the centrality of service provisioning through Content Delivery Networks (CDNs).

This centralization of Internet services has been well observed, not just in IETF discussions [Huston2021] [I-D.nottingham-avoiding-internet-centralization], but also in other efforts that aim to quantify the centralization, using methods such as the Herfindahl-Hirschman Index [HHI] or the Gini coefficient [Gini]. Dashboards of the Internet Society [ISOC2022] confirm the dominant role of CDNs in service delivery beyond just streaming services, both in centralization as well as resulting market inequality, which has been compounded through the global CV19 pandemic [CV19].

While we recognize that many of the existing Internet services are well served with existing solutions, it is our key observation in this draft is that those existing solutions and overall developments equally create pain points for use cases, where the dynamic selection among the set of possible choices is a key requirement, together with the need to reduce service completion time, and thus minimize latencies for explicit resolution steps, while possibly improve resource utilization across all deployed service endpoints.

In the remainder of this document, we first introduce a terminology in Section 2 that provides the common language used throughout this document and all related drafts. We then follow with the use cases in Section 3, each one structured along a description of the experienced service functionality and the aforementioned pain paints that may arise when utilizing existing service discovery and selection capabilities. We then summarize those pain points in Section 4, finally leading us to the formulation of a problem statement for service-based routing in Section 5.

2. Terminology

The following terminology is used throughout the remainder of this document, as well as all the related drafts:

Service: A monolithic functionality that is provided according to the specification for said service.

Composite Service: A composite service can be built by orchestrating a combination of monolithic (or other composite) services. From a client perspective, a monolithic or composite nature cannot be determined, since both will be identified in the same manner for the client to access.

Service Instance: A running environment (e.g., a node, a virtual instance) that supports the execution of the expected service. One service may be deployed in several instances running within the same ROSA network at different network locations.

Service Address: An identifier for a specific service.

Service Instance Address: A locator for a specific service instance.

Service Request: A request for a specific service, addressed to a specific service address, which is directed to at least one of possibly many service instances.

Affinity Request: An invocation of a specific service, following an initial service request, requiring steering to the same service instance chosen during the initial service request.

Service Transaction:
A request for the execution of a specific service, encompassing at least one service request, and zero or more affinity requests.

Service Affinity: Preservation of a relationship between a client and one service instance, created with an affinity request.

ROSA Provider: Entity realizing the ROSA-based traffic steering capabilities over at least one infrastructure provider by deploying and operating the ROSA components within the defined ROSA domain.

ROSA Domain: Domain of reachability for services supported by a single ROSA provider.

ROSA Endpoint: A node accessing or providing one or more services through one or more ROSA providers.

ROSA Client: A ROSA endpoint accessing one or more services through one or more ROSA providers, thus issuing services requests directed to one of possible many service instances that have previously announced the service addresses used by the ROSA client in the service request.

Service Address Router (SAR): A node supporting the operations for steering service requests to one of possibly many service instances, following the procedures outlined in a separate architecture document.

Service Address Gateway (SAG): A node supporting the operations for steering service requests to service addresses of suitable endpoints in the Internet or within other ROSA domains.

3. Deployment and Use Case Scenarios

In the following, we outline examples of use cases that exhibit a degree of service distribution in which a service management scheme through explicit mapping and/or gatewaying may become complex and a possible hindrance for service performance. The following sections illustrate several examples, which complement other work, such as the BBF Metro Compute Networking (MCN) [MCN], which have developed similar but also additional use cases.

3.1. CDN Interconnect and Distribution

Video streaming has been revealed nowadays as the main contributing service to the traffic observed in operators' networks. Multiple stakeholders, including operators and third party content providers, have been deploying Content Distribution Networks (CDNs), formed by a

number of cache nodes spread across the network with the purpose of serving certain regions or coverage areas with a proper quality level. In such a deployment, protection schemas are defined in order to ensure the service continuity even in the case of outages or starvation in cache nodes.

In addition to that, novel schemes of CDN interconnection [RFC6770] [SVA] are being defined allowing a given CDN to leverage the installed base of another CDN to complement its overall footprint.

As result, several caches are deployed in different PoPs in the network. This means that for a given content requested by an end user, several of those caches could be candidate nodes for data delivery. From a service perspective (a service being defined either at the level of a video service, expressed as a service domain name or at the level of individual content streams), specific caches represent service instances, i.e., possible candidates to serve the content and thus realize the desired service.

Currently, the choice of the cache node to serve the customer relies solely on the content provider logic, considering only a limited set of conditions to apply. For instance, the usage of cache-control [RFC7234] allows data origins to indicate caching rules downstream. For instance, the Targeted Cache Control (TCC) [RFC9213] defines a convention for HTTP response header fields that allows cache directives to be targeted at specific caches or classes of caches. The original intent was quite limited: to operate between the data source and the data consumer (browser).

We can observe the following pain points when realizing such scenario in today's available systems:

1. Time-to-first-byte: There exist several aspects that cause latencies and thus increase the time-to-first-byte at the consumer end. Firstly, the service name needs resolution, thus involving, e.g., DNS services, to map the service name to the routing locator. This, however, assumes a traditional end-to-end model for providing the video stream. The insertion of caches changes this model in making a decision at the CDN ingress node as to which cache shall serve the incoming request for content, assigning a specific cache to serve requests. Once a cache is found, the delivery will directly commence from this caching point. Depending on the nature of the cache, however, additional possibly application-level operations, including the decryption of the HTTP request, may happen to direct the incoming request more fine-grained to the specific cache as well as decide upon the availability of the requested content in the cache. This, in addition, may incur latencies. Interpreting video services or even specific (e.g., highly

popular) content as service instances in a service routing system could be seen as a way to reduce some of this complexity and thus the latencies incurred.

2. Dynamicity: Decisions on which caches to be used best may be dynamic and may even change during the lifetime of the overall service, thus requiring to revisit the process to decide about the most appropriate CDN node, thus worsening the latency issue observed in the previous point. An example encompasses the usage of satellites to enhance the content distribution efficiency in cooperation with terrestrial networks. Combining satellites with CDNs may not only leverage the mobility of Low Earth Orbit (LEO) satellites to deliver content among different static caches in terrestrial CDNs, but also include mobile satellites serving as couriers. Furthermore, the AR/VR use case that will follow in Section 3.6 represents a case where frequent change of the cache, in case of several caches available for the desired content, may be desirable for improving on the deliver latency variance experienced at the end user.

3. Service-specific cache/service selection: The performance can be improved by considering further conditions in the decision on which cache node to be selected. Thus, the decision can depend not only on the requested content and the operational conditions of the cache itself, but also on the network status or any other valuable, often service-specific, semantic for reaching those nodes, such data validity, end to end delays, or even video analytics. The latter is relevant since as the number of video files grows, so does the need to easily and accurately search and retrieve specific content found within them.

4. Security: The decision on whether and wherefrom to retrieve the cached content may require decryption operations, depending on the nature of the used cache. This, in turn, may require suitable certificate sharing arrangements between content owner and CDN, which may raise security (as well as privacy) issues.

3.2. Distributed user planes for mobile and fixed access

5G networks natively facilitate the decoupling of control and user plane. The 5G User Plane Function (UPF) connects the actual data coming over the Radio Area Network (RAN) to the Internet. Being able to quickly and accurately route packets to the correct destination on the internet is key to improving efficiency and user satisfaction. For this, the UPF terminates the tunnel carrying end user traffic over the RAN permitting to route such traffic in the 5G network

towards its destination, e.g., providing reachability to edge computing facilities.

Currently, the UPF is planned to be deployed in two parts of the (5G) cellular system, namely in the Core Network and at the Edge inside a Multi-Access Edge Controller (MEC). However, in a future 6G network, it is envisioned that several UPFs can be deployed in a more distributed manner, not only for covering different access areas, but also with the attempt of providing access to different types of services, linked with the idea of network slicing as means for tailored service differentiation, while also allowing for frontloading services to minimize latency.

For instance, some UPFs could be deployed very close to the access for services requiring either low latency or very high bandwidth, while others, requiring less service flows, could be deployed in a more centralized manner. Furthermore, multiple service instances could be deployed in different UPFs albeit scaled up and down differently, depending on the demand in a specific moment at the specific UPF (and its serving area).

Similarly to mobile access networks, fixed access solutions are proposing schemas for the separation of control and user plane for Broadband Network Gateway (BNG) elements [I-D.wadhwa-rtgwg-bng-cups] [BBF]. From the deployment point of view, different instances can be deployed based on different metrics such as coverage, and temporary demand.

As a complement to both mobile and fixed access scenarios, edge computing capabilities are expected to complement the deployments for hosting service and applications of different purposes, both for services internal to the operator as well as third party services.

We can observe the following pain points when realizing such scenario based on today's available solutions:

1. Time-to-first-byte: Low latency in finding suitable service instances, and thus the (distributed) UPF where the chosen service instance is located, is crucial for many of the envisioned (e.g., mobile edge) scenarios that 5G networks envision. Furthermore, the mobile nature of many of the envisioned scenarios also pose specific requirements on service session initiation time, thus the initiation time is key to an acceptable service experience. Thus, the latencies involved in resolving service names into the appropriate routing locator are a key issue.

2. Dynamicity: The mobile nature of many scenarios for, e.g., mobile edge computing and other application areas for 5G

systems, necessitates dynamic decisions, particularly over the runtime of the overall application use case. For instance, a video session with an initial selection of a UPF and associated video server may quickly deteriorate due to, e.g., increasing delay to the initial selection of the video server caused by the user's movement. Also, demands on edge resources may fluctuate with the ephemeral nature of mobile users joining and leaving, while at the same time those edge resources are often more limited in capacity in comparison to centralized resources, consequently requiring a more frequent and, thus, dynamic revisiting of the initial selections of service instances for traffic engineering and thus ensuring a suitable user experience.

3. Service-specific selection: Either for both selection of the specific user plane termination instance, or from that point on, selection of the service instance connected to that user plane function, service-specific semantics (and enabling mechanisms) for the selection choice may be required.

3.3. Multi-homed and multi-domain services

Corporate services usually define requirements in terms of availability and resiliency. This is why multi-homing is common in order to diversify the access to services external to the premises of the corporation, or for providing interconnectivity of corporate sites (and access to internal services such as databases, etc).

A similar scenario in which external services need to be reached from within a specific location, is the Connected Aircraft. Solutions that allow for the exploitation of multi-connected aircrafts (e.g., several satellite connections, plus air-to-ground connectivity) are important to improve passenger experience, while helping make the crew more productive with networking solutions that enable seamless, high-speed broadband. Managing a multi-connected Aircraft would benefit from mechanisms that would enable the selection of the best connection points based on service-specific semantics, besides the traffic related parameters considered by solutions such as SD-WAN, which aims to automate traffic steering in an application-driven manner, based on the equivalent of a VPN service between well defined points.

Multi-homing issues in connection with aircrafts also extend to Unmanned Aircraft Systems (UAS). Rather than focusing on passenger experience, multi-homing over commercial off-the-shelf (COTS) communications modules such as 5G or IEEE 802.11 provide command, control and communications (C3) capabilities to Unmanned Aerial Vehicles (UAV; drones). Here, regulatory frameworks mandate fail-over

and minimum response times that require active management of connectivity to the aircraft.

An architectural approach common to the Connected Aircraft as well as UAS is to view network functions physically located on the aircraft as services, which are multi-homed due to the communications fail-over capabilities of the aircraft. Additionally, objects in flight will regularly change network attachment points for the same physical link, which may require updates to service routing information.

The diversity of providers implies to consider service situations in a multi-domain environment, because of the interaction with multiple administrative domains.

From the service perspective, it seems necessary to ensure a common understanding of the service expectations and objectives independently of the domain traversed or the domain providing such a service. Common semantics can facilitate the assurance of the service delivery and a quick adaptation to changing conditions in the internal of a domain, or even across different domains.

The pain points for multi-homed and multi-domain services are:

1. Time-to-first-byte: A service often requires a short completion time, often constrained by regulatory requirements. Hence, explicit resolution steps may present a challenge to meet those completion times, particularly when being additionally met with a dynamicity in the network conditions, as discussed next.

2. Dynamicity: In the afore discussed multi-homing environments, paths may become entirely unavailable or desirable to change due to new network attachment points becoming available or network conditions dynamically changing. Decisions on which service instance to utilize (exposed through different routing locators on different network attachments) may thus need to become highly dynamic so to ensure restoration of a service to or from an endpoint. This does not only require fast decision making, questioning the use of explicit resolution mechanisms, but also mandates a fast update to the conditions that drive the selection of the right instance (and thus locator in the multi- homed environment) being used for completition of the service.

3. Reliability: Many of the aforementioned scenarios for a multi-homed environments require high reliability irrespective of the dynamicity of the environment in which it operates (some domains impose regulatory requirements on that reliability). Overall, reliability is the constraining requirement in these scenarios. Hence, while multi-homing is a means by which

reliability may be achieved, any solution exploiting multi-homing must take the scenario's specific dynamicity into account.

3.4. Micro-service Based Mobile Applications

Mobile applications usually install a monolithic implementation of the device-specific functionality, where this functionality may explicitly utilize remote service capabilities, e.g., provided through cloud-based services.

Application functionality may also be developed based on a micro-service architecture, breaking down the application into independent functions (services) that can work and communicate together. When such services are jointly deployed (i.e., installed) at the mobile device, its overall functionality resembles that of existing applications.

However, the services may also be invoked on network devices other than the mobile device itself, utilizing service-based routing capabilities to forward the service request (and its response) to the remote entity, effectively implementing an 'off-loading' capability. Efforts such as the BBF MCN work [MCN] capture this aspect as 'edge-to-edge collaboration', where in our case here the edge does include the end user devices themselves.

A distributed system developed based on a micro-service architecture inevitably introduces additional complexity as multiple independent services need to be synchronized in a way that allows them to work as a unified software system. If services are split across servers that multi-faceted infrastructure will need to be provisioned not just in resource allocation but also in its steering of traffic across those resources. This is where a service-centric network solution able to coordinate the chain of such services could play an important role.

The work in [I-D.sarathchandra-coin-appcentres] proposes such micro-service approach for mobile applications. The simple example in [I-D.sarathchandra-coin-appcentres] outlines the distribution of video reception, processing, and displaying capabilities as individual services across many network locations. As a result, display service instances may be switched very quickly based on, e.g., gaze control mechanisms, providing display indirection capabilities that utilize display hardware other than the original device's one, while image processing may be offloaded to one or more processing service instances; given the possible stateless nature of the processing, each individual video frame may be processed by another processing service instance to improve overall latency variance, as shown in [OnOff2022].

As also discussed in [I-D.sarathchandra-coin-appcentres], such micro-service design may well be integrated into today's application development frameworks, where a device-internal service registry would allow for utilizing device-local service instances first before directing the service invocation to the network, the latter relying on a service-based routing capability to steer the request to a 'suitable' service endpoint.

We can observe the following pain points when realizing such scenarios based on explicit discovery mechanisms:

1. Time-to-first-byte: Steering service requests requires up-to-date service instance information. A dedicated resolution service, such as the DNS or even a purely local mDNS system, would add several milliseconds (in CDN systems, [OnOff2022] cites 15 to 45ms for such latency) to the completion time for a request. Performing such resolution (repeatedly) for every request is thus not possible for services such as those outlined in [I-D.sarathchandra-coin-appcentres] where the request arrival time corresponds to framerates in a video scenario. The resulting violation of the available delay budget (defined through the framework) would thus impact the time-to-first-byte for every single (frame) request and ultimately negatively impact the user experience.

2. Dynamicity: User interaction may be one driver for dynamicity in those scenarios. For instance, the aforementioned display indirection may take place at high frequency, triggered by sensory input (e.g., gaze control) to decide which instance is best to direct the video stream to. This may be beneficial for new, e.g., gaming experiences that utilize immersive device capabilities. Other examples may include the offloading of processing capabilities (in case of 'better', i.e., more capable, processing being available elsewhere). This requires service instances to be switched over quickly, either through provisioning new ones or by deciding to use an available yet previously unused service instance, such as in the aforementioned display indirection scenario. Utilizing a newly deployed service instance may be needed for efficiency purposes, e.g., moving the client from a loaded instance to another one available. Even if utilizing a switch-over mechanism, in which the 'old' service instance would used (if this is possible) before switching over to the new one requires that the mapping information is updated in a suitably timely manner, thus needing to align the desired switchover time with the possible mapping update time. Given that DNS updates, even in local environments, can take seconds, while ranging towards minutes or even longer in remote DNS environments, switchover to newly available service instances would be significantly

limited. With this, the micro-service based applications would be executed over rather static sets of deployed service instances, not utilizing the possible computing diversity that the edge computing environment possibly provides them with.

3. Service-specific selection: The choice of service instance may be highly dependent on the application, e.g., driven by user interaction specific to the realized application, and its specific micro-services that are executed in the distributed environment. While network parameters like latency and bandwidth are useful for instance selection, they are also limiting when instance- and service-specific criteria are key. For instance, the processing micro-service in our application example above may be realized across N service instances, instead just one, allowing to have a sequence of frames being processed in a round robin fashion with the result of reducing the latency variance of the processed frame, as shown albeit in a different scenario in [OnOff2022]. Embodying this service-specific selection beyond purely network-centric metrics is key, while linking back to the dynamicity pain point in that those decisions may occur at high frequency, here at every frame request.

4. Distributed network locations for service instances: Service instances may be highly distributed, driven by the chained nature of the overall application experience and its realization in separate service (chain) instances. In turn, the service instance locations may not reside in a single, e.g., edge network, but span access networks and technologies alike, while also relying on (central) cloud-based resources or even remotely located resources provided by users directly (e.g., in visiting scenarios where users may rely services executed in their home network, e.g., for file retrieval).

5. Diversity of application identifiers: While, for instance, a REST-based model of service invocation may be used, thus positioning URIs as the key application identifier, the possible integration into an application framework, such as for Android or iOS, may also favour more application-specific identifiers, which are used for what effectively constitutes a procedure call in the (now distributed) application. Thus, a single application identifier scheme may not exist, thus requiring suitable, possibly separate, mapping schemes beyond the DNS to resolve onto a suitable network locator.

3.5. Constrained Video Delivery

Chunk-based video delivery is often constrained to, e.g., latency or playout requirements, while the content itself may be distributed as

well as replicated across several network locations. Thus, it is required to steer client requests for specific content under specific constraints to one of the possibly many network locations at which the respective content may reside.

The work in [I-D.jennings-moq-quicr-arch] proposes a publish-subscribe metaphor that connects clients to a fixed infrastructure of relays for delivering the desired content under specific constraints. Within our context of service-based routing, the relays realize the selection of the 'right' service instance, deployed by different content providers, where this selection is being constrained by the requirements for the video's delivery to the client. However, the publish/subscribe operations in [I-D.jennings-moq-quicr-arch] manifest an explicit discovery step, plus require the deployment of an explicit relay overlay across possibly many network provider domains.

We can observe the following pain points when realizing such scenario through explicit overlays such as those proposed by QUICr:

1. Time-to-first-byte: [I-D.jennings-moq-quicr-arch] aligns with well-established service routing capabilities in that it still relies on an explicit discovery step through the pub/sub operation in order to 'find' the appropriate relay that may serve or point to a serving endpoint. This incurs additional latency before the actual end-to-end data transfer may commence.

2. Dynamicity: Due to the explicit pub/sub-based discovery step, dynamic changes of serving endpoints will repeatedly incur the aforementioned latency for the brokering between client and serving endpoint. With that, there will likely be a tendency to aggregate content at the level, e.g., of a movie, or at least larger number of chunks. Thus, video provisioning may well be distributed, but the delivery of a selected piece of content will still be limited to few or just a single serving endpoint for the duration of the content delivery.

3. Distributed network locations for the serving endpoints: Although QUICr acknowledges the need for distributing the serving endpoints, it relies on a fixed hierarchy of overlay relays/brokers with a single point of failure in the root relay. Instead a routing-based approach may provide the needed resilience against overlay changes and/or failures, thus not disrupting the video discovery capability of the system.

4. Diversity of application identifiers: QUICr is a very good example for a system that introduces, here for efficiency purposes, its own application identifier scheme (a 128bit

identifier, comprised of user, group and content information) instead of relying on long URIs used to express the desired content. However, this in turn requires the QUICr overlay to not just direct client requests but also provide an application-specific mapping from those identifiers onto the routing locators of the service endpoint.

3.6. AR/VR through Replicated Storage

AR/VR scenarios often utilize stored content for delivering immersive experiences, albeit with interaction capabilities stemming from the nature of the used equipment, e.g., headsets. This interaction may lead to varying content retrieval patterns, e.g., due to early termination of an ongoing content retrieval caused by a user moving the headset and thus changing the field of view.

In addition, AR/VR underlies stringent latency requirements. Among others, [I-D.liu-can-ps-usecases] outlines typical delay budgets for such scenarios. Thus, minimizing latencies for the overall delivery for each chunk is desirable.

Furthermore, the delivery of content to a group of clients often uses replicated storage, i.e., clients may be served from one of possibly many replicated content storages throughout the network. Given the stateless nature of content chunk retrieval in such replicated setup, it may be desirable to make decisions of where to send a client request at EVERY chunk request per client.

Expressed in notations of a queuing system, a system of N clients is suggested to be retrieving content chunks from k service instances, where each chunk request is directed to any of the possible k instances; given the stateless nature of this service, any of the k instances is able to serve the chunk without knowledge of any previous one.

Current systems usually employ a load balancing system, which determines which content storage to use at the beginning of a session as part of the DNS lookup for the video server, using techniques such as Global Server Load Balancing (GSLB [GSLB]). In the notation of a queuing system, only one server exists but serving N/k clients, if there are k replicas and N clients overall.

We can observe the following pain points when realizing such scenario in today's available systems that utilize per-session load balancing solution:

1. Time-to-first-byte: Explicit lookup systems incur latencies, often lying between 15 to 45ms (or significantly more for services not being resolved by the first hop resolver) [OnOff2022]. As outlined in [I-D.liu-can-ps-usecases], the

delay budgets for AR/VR are small in their constituents, requiring not just delivery but storage retrieval, decoding, rendering and other aspects to come together in time. Thus, explicit discovery lookups are to be avoided, pushing the system towards linking a client to a single replica at the start of the session, therefore avoiding any needed lookup for the session remainder.

2. Dynamicity: As shown in [OnOff2022], a retrieval that utilizes any of the k replicas significantly reduces the variance of the retrieval latency experienced by any of the N clients compared to groups of N/k clients retrieving content from only one replica each. Such reduced variance positively impacts the user experience through less buffering applied at the client side but also better adhering to the overall latency budget (often in the range of 100ms in AR/VR scenarios with pre-emptive chunk retrieval). Although pre-emptive retrieval is also possible in systems with explicit lookup operations, the involved latencies pose a problem, as discussed in the previous point.

3. Distributed network locations for content replica: the consequence of the two previous points on latency and dynamicity is the centralization of video delivery in a single network location (e.g., a Point-of-Presence DC), in which service provisioning platforms such as K8S may be used to dynamically select one of the possibly many assigned replica resources in the data centre. Such centralization, however, poses an economic and social problem to many content producers in that it, possibly unduly, increases the economic power of content delivery platforms. Instead, federated and distributed platforms may be preferable by some communities, such as those represented by the 'fediverse', albeit wanting similar traffic steering capabilities within the distributed network system in which content replica may be deployed.

3.7. Cloud-to-Thing Serverless Computing

The computing continuum is a crucial enabler of 5G and 6G networks as it supports the requirements of new applications, such as latency and bandwidth critical ones, using the available infrastructure. With the advent of new networks deployed beyond the edge, such as vehicular and satellite networks, researchers have begun investigating solutions to support the cloud-to-thing continuum, in which applications distribute logic (services) across the network following a micro-service architecture. In this scenario storage, computing and networking resources are managed in a decentralized way between cloud, the edge (most liked MEC) and the adhoc network of moving devices, such as aircraft and satellites.

In this scenario, a serverless-based service architecture may be beneficial for the deployment and management of interdependent distributed computing functions, whose behavior and location can be redefined in real-time in order to ensure the continuous operation of the application. Serverless architecture is closely related to micro-services. The latter is a way to design an application and the former a way to run all or part of an application. That is the key to their compatibility. It is possible to code a micro-service and run it as a serverless function.

The combination of a microservice architecture with a serverless model is a driver for dynamicity in Cloud-to-Thing scenarios where a third-party cloud provider takes care of the deployment od all services encompassing each application. In this situation as soon as the application code is triggered, the server allocates resources to all its services in different locations and draws them back when the application is no longer active.

The consideration of serverless architectures is important for the Cloud-to-Thing continuum, since resources beyond the edge, in the adhoc part of the continuum, may be constraint and intermittently available. Hence it makes sense to leverage a serverless architecture in which applications consists of a set of functions (services) that are not permanently available. On contrary, services have a lifecycle as they are triggered, called, executed, runs and is then removed as soon as it is no longer needed. Serverless services only run when they are needed, potentially saving significant resources.

In this scenario, the combination of a service oriented data plan with a model capable of delegating and adapting serverless services in a Cloud-to-Thing continuum is important. The former need to be aware of the presence of different services/functions in order to be able to execute applications based on the correct selection and invocation of different services, within their lifetime. Most importantly, this awareness of the servies is likely to be highly dynamic in the nature of its distribution across network-connected nodes.

We can observe the following pain points when realizing such scenario in today's available systems based on explicit mapping and/or gatewaying:

1. Time-to-first-byte: The computing continuum aims to support the requirements of new applications, including latency and bandwidth critical ones, using the available infrastructure. However, in a cloud-to-thing scenario high latency may occur due to the need to resolve service names in faraway servers (e.g. DNS). Hence, performing DNS resolution for every request in a cloud-to-thing continuum in which the far edge may be

intermittently connected is not desirable. The violation of the available delay budget would impact the time-to-first-byte for every single request over the Cloud-to-Thing continuum, having a negative impact on the user experience.

2. Dynamicity: In a Cloud-to-Thing scenario, a serverless-based service architecture may be beneficial for the deployment and management of interdependent distributed computing functions, whose behavior and location can be redefined in real-time in order to ensure the continuous operation of the application based on the dynamic behaviour of the network. Service awareness is likely to be highly dynamic due to its distribution in a set of heterogeneous network-connected nodes.

3. Service-specific selection: A serverless architecture brings benefits to a Cloud-to-Thing continuum, where resources beyond the edge may be intermittently available, since applications consist of a set of services that are not permanently deployed. In this scenario and due to the intermittent characteristics of the network, different instances may be deployed in different places. In this context the choice of service instance may be highly dependent on serverless functions currently deployed in a distributed fashion, as well as of the network conditions.

4. Distributed network locations for service instances: In a Cloud-to-thing scenario, the usage of a service oriented data plan to delegate and adapt serverless services is important and needs to be aware of the distributed presence of different services, potentially spanning different networks, in order to be able to execute applications based on the correct selection and invocation of different services, within their lifetime.

3.8. Metaverse

Large-scale interactive and networked real-time rendered tree dimension Extended Reality (XR) spaces, such as the Metaverse, follow the assumption that applications will be hosted on platforms, similarly to current web and social media applications. However, the Metaverse is supposed to be more than the participation in isolated three dimension XR spaces. The Metaverse is supposed to allow the internetworking among a large number of XR spaces, although some problems have been observed such as lock-in effects, centralization, and cost overheads.

In spite of the general understanding about potential internetworking limitations, current technical discussions are ignoring the networking challenges altogether. From a networking perspective, it is expected that the Metaverse will challenge traditional client-server inspired web models, centralized security trust anchors and

server-style distributed computing, due to the need to take into account interoperability among a large number of XR spaces, low latency and the envisioned Metaverse pervasiveness.

Current Metaverse platforms rely on web protocols and cloud services, but suffer from performance limitations when interconnecting XR spaces. Some of the challenges pass by consistent throughput to handle high resolution XR applications, and fast response times to computational requests. This leads to the need to bring cloud computing and storage resources towards the edge to reduce long round trip times.

To support Metaverse low latency requirements taking into account the constrained resource of heterogeneous devices in the Cloud-to-Thing continuum, a service-centric networking framework should be based on micro-services executed as serverless services/functions inside selected devices. The motivation to look at serverless functions is related to their capability to simplify service management on heterogeneous devices.

In this context, an open and decentralized Metaverse, able to allow the internetworking of a large number of XR spaces, may be supported by intertwining distributed computing and networking. Hence it is expected that Metaverse applications may gain from a service-centric network framework able to support the execution of services while taking advantage of storage, networking, and computing resources located as close as possible from users, with a dynamic assignment of client requests to those resources.

While the usage of isolated XR spaces is currently a reality, the deployment of a large scale Metaverse should relies in a decentralized networking framework, of which Distributed Ledger Technology (DLT) is a major driver, facilitating the deployment of several Metaverse features such as streaming of payments and NFTs that make digital ownership possible in the Metaverse. Moreover, DLT makes it possible to identify oneself in a secure way in the Metaverse, being also a major web3.0 building block. The Web3.0 builds Internet services on decentralized platforms, being the ownership of the platform tokenized and the users' own tokens are calculated based on their contribution to the platform. For instance Web3.0 domain names are DLT-based DNS addresses that allow users to create and manage their own personalized domains.

Development of DLT based on a service-centric networking approach brings several benefits. To start with, designing DLT applications as microservices allow many software engineering initiatives to run in parallel, reduce dependencies between software development, and allow for the support of multiple technologies, languages and frameworks. Moreover developing DLT on a service-centric networking framework may

help to solve the DLT scalability problem, allowing the implementation of data sharding techniques, in which the storage of the ledger and/or the data used to recreate the ledger is divided across many shards, which are distirbuted between different devices. This process reduces individual nodes storage requirements at any given time to that of a single shard or small set of shards. a service-centric networking approach may also support the need for data availability sampling, providing a method for the network to check that data is available without putting too much strain on any individual node. This will allow DLT to prove that historical data needed to reconstruct part of the ledger was available at one point (i.e. when the block was produced) without nodes actually having to download all the data themselves.

We can observe the following pain points when realizing such scenario in today's available systems based on explicit mapping and/or gatewaying:

1. Time-to-first-byte: Massive interactive immersive spaces based on virtual reality, augmented reality, mixed reality or spatial computing will have more demanding network requirements than current applications, especially latency-wise. On the other hand, Internet technologies induce significant end-to-end latency, such as explicit lookup systems and congestion control. The former incur latencies, often between 15 to 45ms, or significantly more for services not being resolved by the first hop resolver. On the other hand, end-to-end congestion control relies on inducing latency. Additionally, the internet runs on shared infrastructure or frequencies, and Internet Services Providers have no incentives or simple means to change that. Hence, it will be beneficial to run web-based Metaverse service on top of a framework able to avoid explicit lookup systems and end-to-end traffic.

2. Dynamicity: To fulfill the user experience and Quality-of-Service (QoS) requirements, the Metaverse indeed requires extremely intensive and dynamic resource demands that have never been seen before. To address the Metaverse resource management challenge, multi-tier computing architectures can be considered, in which case we need to deploy a system able to select a proper set of services to run Metaverse applications, handling the dynamic needs of different applications over time.

3. Distributed network locations for service instances: An open and decentralized Metaverse, able to allow the internetworking of a large number of XR spaces, may be supported by intertwining distributed computing and networking. In this scenario, computing intensive tasks, e.g. of real-time graphic and audio, rendering from different metaverse services may be

processed in different network locations based on a collaborative computing paradigm, which will benefit from a system able to find the most suitable service instances in a distributed networking environment.

4. Service-specific selection: The choice of service instance may be highly dependent on the metaverse application, and they may be located in different places in the network. Hence there is the need to find not only the closest service instance, but the one that fullfils the needs of specific applications.

5. Diversity of application identifiers: A metaverse application may encompass a significant set of heterogeneous services, such as video, 3D models, spatial sound, voice, IoT, each of which with a specific set of identifiers and semantics. Thus, a single application identifier scheme may not exist, thus requiring suitable, possibly separate, mapping schemes beyond the DNS to resolve onto a suitable network locator.

6. Selection sovereignty: Utilizing a global resolution system may not be desirable in the case of Metaverse applications, since a centralizing DNS resolution system may run significantly counter the desire to not reveal service usage patterns to large corporations. Distributing also the service selection itself, maybe even governed under a regional/national or organizational body more directly associated to the service category itself, may also address the sovereignty concerns of those service providers and users alike.

3.9. Popularity-based Services

The BBF MCN use case report [MCN] outlines 'popularity' as a criteria to move from current explicit indirection-based approaches (such as DNS, GSLB, or Alto) to active service-based routing approaches.

Here, popularity, e.g., measured in service usage over a period of time, is being used as a trigger to announce a popular service to an active service-based routing platform, while less popular services continue to be served via existing (e.g., DNS-based) methods. Equally, services may be unannounced, thus retracted, from the service-based routing overlay to better control the overall cost for the provisioning of the service-based routing overlay.

With this, one could foresee the provisioning of a service-based routing overlay, such as ROSA, as an optimization for a CDN platform provider, either through commercially interfacing to a separate ROSA provider or providing the ROSA domain itself.

We can observe the following pain points when realizing such scenario in today's available systems based on explicit mapping and/or gatewaying:

1. Time-to-first-byte: Popular services desire low latency in delivering their responses. Such popular services may be popular videos (e.g., routing based on the video title), but also popular elements in webpages with the aim to reduce the overall page loading time. Resolution latency adds to the time-to-first-byte, thus removing or reducing that latency is key. Particularly for webpages, the latency incurred for objects that reside on popular albeit distinct websites may compound the overall latency penalty due to the distinct resolution required to be performed.

2. Dynamicity: Popularity may vary as a function for different types of content, e.g., being time dependent for video content while being type-specific for webpages (of certain categories). Most importantly, the popularity may change based on that function, requiring the system to adjust its announcement into the active service routing platform. Furthermore, the service routing capability for those popular service may not just foresee to serve the popular service from dedicated resources but even dynamically assign the specific resource to be used. This aligns dynamicity here with that observed in the use case of Section 3.6, e.g., wanting to serve popular content from a set of replicated resources, possibly distributed across more than one network site.

3. Distributed network locations for the serving endpoints: Continuing from the previous point, popular services must not just be served from dedicated resources but distributed ones. More so, the assignment policy may depend not just on the service but the network region in which requests are being initiated.

3.10. Data and Processing Sovereignty

Data access of any kind, be it for personal as well as curated content or for social media, has become essential to our lives, yet its implementation is fraught with problems. Content as well as service hosts are forced to use CDNs to effectively distribute their data, or choose to rely on one of the big platforms entirely. As a result, the transport from host to receiver is overseen by a conglomerate of giant multi-national corporations, as also observed in various Internet metrics like the GINI of HHI metric. For an end user, data governance but also realization of the significant (often cloud) infrastructure of those corporations are thus difficult to oversee as a result.

As a result, this mode of organizing data transport has created structural inefficiencies in our service provisioning infrastructure, e.g., for those distributed end user created video content. In contrast, a public video streaming infrastructure, which takes content from various hosts and distributes it in an efficient fashion without involvement of a centralized entity, may be preferable from a data governance and ownership standpoint, while still wanting to maintain the desired service quality. Yet, dominant video streaming providers are not incentivized to develop such technologies, since it reduces the barrier of entry for competitors. Instead, if necessary technologies were developed, big economic blocks like the EU could commission the creation of such an infrastructure on their territory even incentivize its use to foster decentralization and localized data governance. Such an undertaking could both possibly reduce the resource footprint for service provisioning as well as open the heavily concentrated market of service provisioning platforms.

We envision, for instance for accessing a video, that a user would access a service address, which in turn would be resolved to a regional service instance. This instance would either use local caches or connect to the wider video streaming infrastructure to retrieve the requested video in the most efficient manner. Within the video streaming infrastructure, techniques such as proximal caching or multicasting could be used to minimize resource usage.

Key here is not the ability to build such service provisioning infrastructure per se, but link the resolution of the service address to an IP address to a service category specific resolution overlay that is not just reducing the latencies experienced in today's DNS systems but allows for being deployed entirely independent from large corporations but instead from decentralized communities, such as for instance the 'fediverse'.

We can observe the following pain points when realizing such scenario in today's available POP-based systems:

1. Dynamicity: Decentralization of infrastructure may increase the dynamicity of assignments between executing service entities, not just from clients to initial services but also among (chained) services. This dynamicity may serve the localization of data traffic but also result from permissionless participation in the service, such as for blockchain or similar services.

2. Distributed network locations for the serving endpoints: Data localization, as one consequence for increasing national and/or regional data and processing sovereignty, may lead to a higher distribution of serving endpoints in the network and thus will

need support in the respective service endpoint selection methods.

3. Service-specific selection: The localization requirements may differ from one service to another, hence a one-size-fits-all, e.g., through geo-locating, will not suffice. Instead, services may want to employ their specific choice of selection.

4. Diversity of application identifiers: While domain services have proliferated in service provisioning, many particularly local services may rely on application-specific identifiers, thus not relying on the DNS and its associated governance of the namespace.

5. Selection sovereignty: Utilizing a global resolution system may not be desirable for localized, including community driven services. But more so, the drive to centralizing DNS resolution through CDN provider based HTTP-over-DNS solutions, may run significantly counter the desire to not reveal service usage patterns to large corporations. Distributing also the service selection itself, maybe even governed under a regional/national or organizational body more directly associated to the service category itself (e.g., for fediverse social media), may also address the sovereignty concerns of those service providers and users alike.

3.11. Web Browsing

Web browsing remains an important usage of the Internet, including during mobile use. Whether it is browsing through pages of places, e.g., linked through mapping services, or view the results of a search performed before, users often view and thus access pages on the Internet through the HTTP protocol suite. This is unlike, e.g., social media or over-the-top video services, which often underlie strict traffic engineering to ensure a superior user experience and are mainly accessed through dedicated, e.g., mobile, applications. However, for web browsing as outlined here, content delivery networks (CDNs) may be used for frequently visited websites, utilizing CDNs as large web caches to improve page loading times.

Key to the browsing experience is that webpages include links, often to other sites, for additional content. For instance, in 2019, a web page loaded on a desktop included on average 70 resources (75 for as mobile page) [MACHMETRIC], many of which may require their own DNS resolution if pointing to other URLs than those previously resolved (within the browsed page or in other pages visited before). Further, according to [MACHMETRIC], the time to first bye (TTFB) was 1.28s for a desktop and 2.59s for mobile pages in the same year, while it took

on average about 4.7s to load the overall page, with 11.9s for a mobile page .

Key here is that the DNS latency for resolving one URL may significantly accumulate due to the many objects a web page may include. While CDNs reduce page loading time, Internet-based resources (thus those not hosted by the local CDN), still require resolving the URL, often at significantly higher latency than the CDN-based resolver; with [OnOff2022] positioning Internet resources at more than 100ms to resolve through the DNS, while CDN-hosted resources may be resolved within 15 to 45ms.

We can observe the following pain points when realizing such scenario in today's available POP-based systems:

1. Time-to-first-byte (TTFB): A lot of emphasis is given in web design on improving the TTFB, particularly to render the initial information for the end user. However, as observed above, that TTFB remains high, which may also be a factor of users not just browsing popular sites, which often are very well traffic engineered, but encountering websites, e.g., in mapping applications, that are hosted outside the CDN, i.e., within the wider Internet.

2. Accumulated latency: While we have recognized the impact of resolution latency in the different use cases of this document, web browsing often exhibits a strong accumulated effect of individual DNS resolutions needing to happen. Sure, this effect is highly dependent on the linked character of the resources on the web page. For instance, if rendering a media gallery with images stored at the same server that provides the initial frame layout, no further DNS resolution is required since all resources reside within the same URL. But if the same 'gallery' experience were to show images from distributed websites, additional DNS resolution, possibly for every image, would be required, thus significantly worsening the latency experienced by the end user.

From the above, we can identify the explicit resolution step, requiring a lookup request with response, before the actual HTTP-based transfer may commence, as a key source for impacting the page retrieval time (we note that other aspects like client rendering and server performance are impacting the overall page loading time but this lies outside the scope of the discussions here).

With the above in mind, we postulate that an in-band signalling of URL to IP mapping requests may significantly reduce the overall page retrieval time, particularly for those scenarios in which no other

traffic engineering methods, such as the careful balancing between CDN caches, is applied, as it is usual for popular sites.

In a preliminary evaluation of such in-band benefits, we positioned the in-band element, realizing the functionalities outlined in [I-D.trossen-rtgwg-rosa-arch] as the Service Access Router and the Service Access Gateway, at the CDN ingress. This enables access to ROSA-hosted resources as well as resources hosted by both the CDN and the wider Internet through the same CDN ingress point.

We assumed a client-CDN RTT of 20ms and we were able to show a reduction for up to 60% of page retrieval time in a simple model where a single page is being retrieved, followed by a parallelized retrieval of all objects included in the initial page. Further, the time-to-first-byte (i.e., the retrieval of the initial object of up to 14kB size) was reduced by up to 70% for CDN-hosted objects. Although those results are preliminary, they outline the potential that moving from explicit resolution to in-band resolution could bring.

4. Issues Observed Across the Use Cases

Several observations can be drawn from the use case examples in the previous section in what concerns their technical needs:

1. Anycast behaviour: Service instances for a specific service may exist in more than one network location, e.g., for replication purposes to serve localized demand, while reducing latency, as well as to increase service resilience.

2. Dynamic decisions: Selections of the 'right' or 'best' service instance in the aforementioned anycast behaviour may be highly dynamic under the given service-specific decision policy and thus may change frequently with demand patterns driven by the use case. For instance, in our examples of Distributed Mobile applications (Section 3.4) and Metaverse (Section 3.8), human interaction may drive the requirement for selecting a suitable service instance down to few tens of milliseconds only, thus creating a need for high frequency updates on the to-be-chosen service instance. As a consequence, traffic following a specific network path from a client to one service instance, may need to follow another network path or even utilize an entirely different service instance as a result of re-applying the decision policy.

3. Ephemeral Service Instances: While the deployment of service instances may follow a longer term planning cycle, e.g., based on demand/supply patterns of content usage, it may also have an ephemeral nature, e.g., scaling in and out dynamically to cope

with temporary load situations as well as with the temporary nature of serverless functions. In existing methods, that impose significant delays in updating the mappings between service name and IP locator, those newly established resources may often remain unused since updated mapping are not available in due course.

4. Latency: Minimizing the latency from the initiating client request to the actual service response arriving back at the client is crucial in many of our scenarios. Any improvement on utilizing the best service instance as quickly as possible, thus taking into account any 'better' alternative to the currently used one, may have a direct contribution to reducing latency. With this, the latencies incurred by explicit resolution steps may often add a significant amount to the available delay budget, often even exceeding it, as discussed in Section 3.6. The work in [OnOff2022] outlines the possible impact of reducing the use of explicit resolution method, thus removing the frequent latency imposed by them. Furthermore, the latency for DNS resolution may be accumulative, as discussed in our browsing use cases in Section 3.11, possibly significantly worsening the latency impact on the overall user experience.

5. Service-specific selection: Knowing which are the best locations to deploy a service instance is crucial and may depend on service-specific demands, realizing a specific service level agreement (with an underlying decision policy) that is tailored to the service and agreed upon between the service platform provider and the communication service provider.

6. Support for service distribution: Typical application or also L4-level solutions, such as GSLB, QUIC-based indirection, and others, lead effectively to egress hopping when performed in a multi-site deployment scenario in that the client request will be routed first to an egress as defined either through the DNS resolution or the indirection through a central server, from which the request is now resolved or redirected to the most appropriate DC site. In deployments with a high degree of distribution across many (e.g., smaller edge computing) sites, this leads to inefficiencies through path stretch and additional signalling that will increase the request completion time. Instead, it would be desirable to have a more direct traffic towards the site where the service will eventually be executed.

7. Namespace mapping: The namespace for services and applications is separate from that of routable identifiers used to reach the implementing endpoints, i.e., the service instances. Resolution

and gateway services are often required to map between those namespace, adding management and thus complexity overhead, an observation also made in [Namespaces2022].

8. Service chaining: A specific service may require the execution of more than one service instance, in an intertwining way, which in turn requires the coordination of the right service instances, each of which can have more than one replica in the network.

We can conclude from our observations above that (i) distribution (of service instances), (ii) dynamicity in the availability of and choosing the 'best' service instance, and (iii) efficiency in utilizing the best possible service instance are crucial issues for our use cases.

5. Problem Statement

This document presented a number of use cases for service-based routing. Common across all those use cases is the inherent need for a dynamic anycast decision, i.e., the frequent (re-)assignment of service instances among a set of possible service endpoints.

Additionally, this (re-)assignment is driven by service-specific policies that capture not just performance-oriented metrics but also possible user-centric interactions with other services, which are jointly composed towards a larger, chained experience.

Existing methods, such as DNS, Alto, and others, already handle the (re-)assignment between service name and routing locator. For this, they employ an out-of-band resolution step, initiated by the client in relation to whatever service the client may want to use and resulting in returning the chosen IP address to the client, after which the latter initiates a direct communication with the now resolved IP address of the chosen service instance. This method has been well proven for the many services as they exist in the Internet today.

However, we must also note that those resolution steps incur explicit resolution latencies that add to the end-to-end communication between client and service instance. Furthermore, solution-specific lags may exist in updating the name-locator assignments, while each resolution solution supports its specific application identifier domain, such as domain names (DNS), URLs (ALTO) or others. In our use cases, these issues, together with others, cause problems to the realization and performance of the use cases and/or the user experience they set out to offer.

WHAT IF a similar end-to-end procedure of data communication between a client and a 'best' choice of service instances (out of set of

possibly many) existed that significantly reduced the aforementioned latency, while it allowed for updating the assignments at rates that are more aligned with the possibility to establish new service instances in distributed locations?

We assert that the following problems need to be addressed in providing such improved procedure:

1. How can we make decisions on anycast-based service instance assignments at high rate, even down to every service request, raising the question on how to possibly remove the need for an explicit out-of-band discovery step, which incurs additional latencies before any data transfer can commence?

2. How could we improve on the update speed for the assignments between service name and 'best' IP locator for the service instance to be used, e.g., using insights into routing-based approaches, where one desirable capability would to align the rate of the possible anycast assignment update with that of the possible availability of the service instance resource?

3. How could we allow for incorporating service-specific policies into the anycast selection mechanism?

4. How can we support any application identifier space (within the governance defined for that identifier space) beyond domain names?

5. How could the chaining of more than one service be realized without explicit discovery latency incurred?

6. Most current SBR methods, specifically the DNS, are initiated by the client in sending an explicit resolution request, followed by subsequent IP-based transfer of the data, that transfer being constrained through the routing policies defined by the (possibly multi-domain) networks across which those IP packets will traverse. This leaves transaction management entirely to the endpoints, driven by a repeated resolution, if renewed decisions are needed. How can we possibly preserve such client-driven operation, and thus avoid transaction state in the network?

We argue that existing solutions do not provide adequate answers to the above problems, which we will separately deepen in our separate gap analysis, leading us to formulate requirements for possible answers in the same draft, with a first proposal for a solution framework and architecture in a separate document.

6. Conclusions

Flexible and even highly dynamic service-based routing is key for a number of emerging and existing use cases, as we outlined in this draft.

As we outlined with a range of use cases, there exist a number of issues when realizing those use cases, leading us to formulate a problem statement for needed work in the IETF to identify adequate answers. In our companion documents, we present our current understanding on the shortcomings of existing solutions to SBR, together with requirements for a possible improved answer to those problems.

7. Security Considerations

To facilitate the decision between service information (i.e., the service address) and the IP locator of the selected service instance, information needs to be provided to the ROSA service address routers. This is similar to the process of resolving domain names to IP locators in today's solutions, such as the DNS. Similar to the latter techniques, the preservation of privacy in terms of which services the initiating client is communicating with, needs to be preserved against the traversing underlay networks. For this, suitable encryption of sensitive information needs to be provided as an option. Furthermore, we assume that the choice of ROSA overlay to use for the service to locator mapping is similar to that of choosing the client-facing DNS server, thus is configurable by the client, including to fall back using the DNS for those cases where services may be announced to ROSA methods and DNS-like solutions alike.

8. IANA Considerations

This draft does not request any IANA action.

9. Acknowledgements

Many thanks go to Ben Schwartz, Mohamed Boucadair, Tommy Pauly, Joel Halpern, Daniel Huang, Peng Liu, Hannu Flinck, and Russ White for their comments to the text to clarify several aspects of the motiviation for and technical details of ROSA.

10. Contributors

Johann Schoepfer
Email: johann.alban.schoepfer@gmail.com

Emilia Ndilokelwa Weyulu
Email: emilia.ndilokelwa.weyulu@huawei.com

11. Informative References

[BBF] ""Control and User Plane Separation for a disaggregated BNG"", Technical Report-459 Broadband Forum (BBF), 2020.

[CV19] Feldmann, A., Gasser, O., Lichtblau, F., Pujol, E., Poese, I., Dietzel, C., Wagner, D., Wichtlhuber, M., Tapiador, J., Vallina-Rodriguez, N., Hohlfeld, O., and G. Smaragdakis, "A Year in Lockdown: How the Waves of COVID-19 Impact Internet Traffic", Paper Communications of ACM 64, 7 (2021), 101-108, 2021.

[Gini] "Gini Coefficient", Technical Report Wikipedia, 2022, <https://en.wikipedia.org/wiki/Gini_coefficient>.

[GSLB] "What is GSLB?", Technical Report Efficient IP, 2022, <https://www.efficientip.com/what-is-gslb/>.

[HHI] "Herfindahl-Hirschman index", Technical Report Wikipedia, 2022, <https://en.wikipedia.org/wiki/Herfindahl-Hirschman_index>.

[Huston2021] Huston, G., "Internet Centrality and its Impact on Routing", Technical Report IETF side meeting on 'service routing and addressing', 2021, <https://github.com/danielkinguk/sarah/blob/main/conferences/ietf-112/materials/Huston-2021-11-10-centrality.pdf>.

[I-D.ietf-quic-load-balancers] Duke, M., Banks, N., and C. Huitema, "QUIC-LB: Generating Routable QUIC Connection IDs", Work in Progress, Internet-Draft, draft-ietf-quic-load-balancers-16, 21 April 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-load-balancers-16>.

[I-D.jennings-moq-quicr-arch] Jennings, C. F. and S. Nandakumar, "QuicR - Media Delivery Protocol over QUIC", Work in Progress, Internet-Draft, draft-jennings-moq-quicr-

arch-01, 11 July 2022, <https://datatracker.ietf.org/doc/html/draft-jennings-moq-quicr-arch-01>.

[I-D.liu-can-ps-usecases] Liu, P., Eardley, P., Trossen, D., Boucadair, M., Contreras, L. M., Li, C., and Y. Li, "Computing-Aware Networking (CAN) Problem Statement and Use Cases", Work in Progress, Internet-Draft, draft-liu-can-ps-usecases-00, 23 October 2022, <https://datatracker.ietf.org/doc/html/draft-liu-can-ps-usecases-00>.

[I-D.nottingham-avoiding-internet-centralization] Nottingham, M., "Centralization, Decentralization, and Internet Standards", Work in Progress, Internet-Draft, draft-nottingham-avoiding-internet-centralization-11, 1 July 2023, <https://datatracker.ietf.org/doc/html/draft-nottingham-avoiding-internet-centralization-11>.

[I-D.sarathchandra-coin-appcentres] Trossen, D., Sarathchandra, C., and M. Boniface, "In-Network Computing for App-Centric Micro-Services", Work in Progress, Internet-Draft, draft-sarathchandra-coin-appcentres-04, 26 January 2021, <https://datatracker.ietf.org/doc/html/draft-sarathchandra-coin-appcentres-04>.

[I-D.trossen-rtgwg-rosa-arch] Trossen, D., Contreras, L. M., Finkhäuser, J., and P. Mendes, "Architecture for Routing on Service Addresses", Work in Progress, Internet-Draft, draft-trossen-rtgwg-rosa-arch-00, 27 June 2023, <https://datatracker.ietf.org/doc/html/draft-trossen-rtgwg-rosa-arch-00>.

[I-D.wadhwa-rtgwg-bng-cups] Wadhwa, S., Shinde, R., Newton, J., Hoffman, R., Muley, P., and S. Pani, "Architecture for Control and User Plane Separation on BNG", Work in Progress, Internet-Draft, draft-wadhwa-rtgwg-bng-cups-03, 11 March 2019, <https://datatracker.ietf.org/doc/html/draft-wadhwa-rtgwg-bng-cups-03>.

[ISOC2022] "Internet Centralization", Technical Report ISOC Dashboard, 2022, <https://pulse.internetsociety.org/centralization>.

[MACHMETRIC] "Average Page Load Times for 2020-Are you faster?", Technical Report-459 Broadband Forum (BBF), 2020,

<https://machmetrics.com/speed-blog/average-page-load-times-for-2020/>.

[MCN] ""Metro Compute Networking: Use Cases and High Level Requirements"", Technical Report-466 Broadband Forum (BBF), 2021.

[Namespaces2022] Reid, A., Eardley, P., and D. Kutscher, "Namespaces, Security, and Network Addresses", Paper ACM SIGCOMM workshop on Future of Internet Routing and Addressing (FIRA), 2022.

[OnOff2022] Khandaker, K., Trossen, D., Yang, J., Despotovic, Z., and G. Carle, "On-path vs Off-path Traffic Steering, That Is The Question", Paper ACM SIGCOMM workshop on Future of Internet Routing and Addressing (FIRA), 2022.

[RFC6770] Bertrand, G., Ed., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, DOI 10.17487/RFC6770, November 2012, <https://www.rfc-editor.org/info/rfc6770>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <https://www.rfc-editor.org/info/rfc7234>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC9213] Ludin, S., Nottingham, M., and Y. Wu, "Targeted HTTP Cache Control", RFC 9213, DOI 10.17487/RFC9213, June 2022, <https://www.rfc-editor.org/info/rfc9213>.

[SVA] ""Optimizing Video Delivery With The Open Caching Network"", Technical Report Streaming Video Alliance, 2018.

[TIES2021]
Giotsas, V., Kerola, S., Majkowski, M., Odinstov, P., Sitnicki, J., Chung, T., Levin, D., Mislove, A., Wood, C. A., Sullivan, N., Fayed, M., and L. Bauer, "The Ties that un-Bind: Decoupling IP from web services and sockets for

robust addressing agility at CDN-scale", Paper ACM SIGCOMM, 2021.

Authors' Addresses

Paulo Mendes
Airbus
82024 Taufkirchen
Germany

Email: paulo.mendes@airbus.com
URI: http://www.airbus.com

Jens Finkhaeuser
Interpeer gUG
86926 Greifenberg
Germany

Email: ietf@interpeer.io
URI: https://interpeer.io/

Luis M. Contreras
Telefonica
Ronda de la Comunicacion, s/n
Sur-3 building, 1st floor
28050 Madrid
Spain

Email: luismiguel.contrerasmurillo@telefonica.com
URI: http://lmcontreras.com/

Dirk Trossen
Huawei Technologies
80992 Munich
Germany

Email: dirk.trossen@huawei.com
URI: https://www.dirk-trossen.de