ALTO
Internet-Draft
Intended status: Informational
Expires: 15 April 2024

L. M. Contreras
Telefonica
S. Randriamasy
Nokia Bell Labs
J. Ros-Giralt
Qualcomm Europe, Inc.
D. Lachos
Benocs
C. Rothenberg
Unicamp

13 October 2023

Use of ALTO for Determining Service Edge
draft-contreras-alto-service-edge-10

Abstract

Service providers are starting to deploy computing capabilities across the network for hosting applications such as AR/VR, vehicle networks, IoT, and AI training, among others. In these distributed computing environments, knowledge about computing and communication resources is necessary to determine the proper deployment location of each application. This document proposes an initial approach towards the use of ALTO to expose such information to the applications and assist the selection of their deployment locations.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at draft-contreras-alto-service-edge.html">https://giralt.github.io/draft-contreras-alto-service-edge/#go.draft-draft-contreras-alto-service-edge.html">contreras-alto-service-edge.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-contreras-alto-service-edge/.

Discussion of this document takes place on the WG Working Group mailing list (mailto:alto@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/alto/. Subscribe at https://www.ietf.org/mailman/listinfo/alto/.

Source for this draft and an issue tracker can be found at https://github.com/giralt/draft-contreras-alto-service-edge.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

With the advent of network virtualization, operators can make use of dynamic instantiation of network functions and applications by using different techniques on top of commoditized computation infrastructures, permitting a flexible and on-demand deployment of services, aligned with the actual needs as demanded by the users.

Operators are starting to deploy distributed computing environments in different parts of the network with the objective of addressing different service needs including latency, bandwidth, processing capabilities, storage, etc. This is translated in the emergence of a number of data centers (both in the cloud and at the edge) of different sizes (e.g., large, medium, small) characterized by distinct dimension of CPUs, memory, and storage capabilities, as well as bandwidth capacity for forwarding the traffic generated in and out of the corresponding data center.

The proliferation of the edge computing paradigm further increases the potential footprint and heterogeneity of the environments where a function or application can be deployed, resulting in different unitary cost per CPU, memory, and storage. This increases the complexity of deciding the location where a given function or application should be best deployed, as this decision should be influenced not only by the available resources in a given computing environment, but also by the network capacity of the path connecting the traffic source with the destination.

It is then essential for a network operator to have mechanisms assisting this decision by considering a number of constraints related to the function or application to be deployed, understanding how a given decision on the computing environment for the service edge affects the transport network substrate. This would enable the integration of network capabilities in the function placement decision and further optimize performance of the deployed application.

This document proposes the use of ALTO [RFC7285] for assisting such a decision.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Computing Needs

The compute needs for an application or service function are typically set in terms of certain requirements in terms of processing capabilities (i.e., CPU), as well as volatile memory (i.e., RAM) and storage capacity. There are two ways of considering those needs: either as individual values for each of the resources, or as a pre-defined bundle of them in the form of instance or compute flavor.

3.1. Working with compute resource values

Compute resource values can be obtained from cloud manager systems able to provide information about compute resources in a given compute node. These cloud manager systems typically provide information about total resources in the compute node and either the used or the allocatable resources. This information can be leveraged for taking application or service function placement decisions from the compute capability perspective. Each cloud management system has their own schema of information to be provided. In general terms, information about CPU, memory and storage can be provided, together with the identifiers of the compute node and some other system information. Examples of these cloud management systems are Kubernetes, OpenStack and OpenNebula. The following table summarizes some of the obtainable information from these cloud management systems.

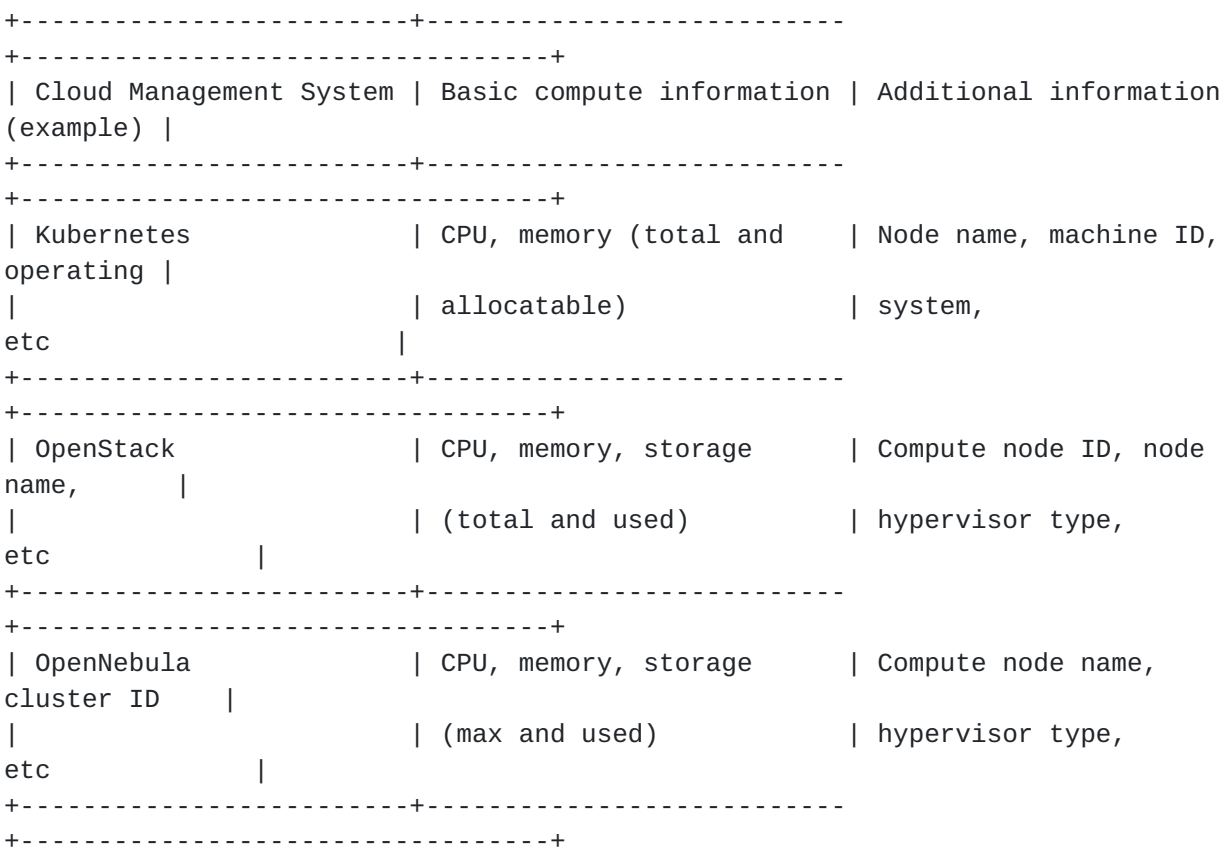

Figure 1: Compute resource information from well known cloud managers

3.2. Working with compute flavors

Cloud computing providers such as Amazon Web Services Microsoft Azure, or Google Cloud Platform, typically structure their offerings of computing capabilities by bundling CPU, RAM and storage units as quotas, instances, and flavors that can be consumed in an ephemeral fashion, during the actual lifetime of the required function or application. In the case of Amazon Web Services such grouping of compute resources is denominated instance type, being the same concept named as virtual machine size in Microsoft Azure and machine type in Google Cloud Platform.

This same approach is being taken for characterizing bundles of resources on the so-called Network Function Virtualization Infrastructure Points of Presence (NFVI-PoPs) being deployed by the telco operators. For instance, the Common Network Function Virtualization Infrastructure Telecom Taskforce (CNTT) [CNTT], jointly hosted by GSMA [GSMA] and the Linux Foundation, intends to harmonize the definition of instances and flavors for abstracting capabilities of the underlying NFVI, facilitating a more efficient utilization of the infrastructure and simplifying the integration and

certification of functions. (Here certification means the assessment of the expected behavior for a given function according to the level of resources determined by a given flavor.) An evolution of this initiative is Anuket [Anuket], which works to consolidate different architectures for well-known tools such as OpenStack and Kubernetes.

Taking CNTT as an example, the flavors or instances can be characterized according to:

* _Type of instance (T):_ Used to specify the type of instances, which are characterized as B (Basic), or N (Network Intensive). The latter includes network acceleration extensions for offloading network intensive operations to hardware.

* _Interface Option (I):_ Used to specify the associated bandwidth of the network interface.

* _Compute flavor (F):_ Used to specify a given predefined combination of resources in terms of virtual CPU, RAM, disk, and bandwidth for the management interface.

* _Optional storage extension (S):_ Used to specify additional storage capacity.

* _Optional hardware acceleration characteristics (A):_ Used to specify acceleration capabilities for improving the performance of the application.

The naming convention of an instance is thus encoded as TIFSA.

3.3. ALTO to support abstracted compute information

Examples in sections 3.1 in particular Figure 1 stress the need to abstract these values for the sake of harmonization and ALTO could provide the information services to do so. Abstraction is also needed to (i) aggregate values for simplicity or scalability and (ii) support potential confidentiality needs of data center management entity. To specify the ALTO metrics relevant to compute capabilities, an exercise similar to the RFC-9439 to be [I-D.ietf-alto-performance-metrics] would be useful. The initial metrics could be taken from different standardization bodies or cloud providers or IETF working groups. Besides, metrics reflecting energy consumption of application deployment footprint also need to be considered, given the expected massive usage of ML/AI and the current context urging to optimize energy consumption.

4. Usage of ALTO for Service Placement

ALTO can assist the deployment of a service on a specific flavor or instance of the computing substrate by taking into consideration network cost metrics. A generic and primary approach is to take into account metrics related to the computing environment, such as availability of resources, unitary cost of those resources, etc. Nevertheless, the function or application to be deployed on top of a given flavor must also be interconnected outside the computing environment where it is deployed, therefore requiring the necessary network resources to satisfy application performance requirements

such as bandwidth or latency.

The objective then is to leverage ALTO to provide information about the more convenient execution environments to deploy virtualized network functions or applications, allowing the operator to get a coordinated service edge and transport network recommendation.

4.1. Integrating Compute Information in ALTO

CNTT proposes the existence of a catalogue of compute infrastructure profiles collecting the computing capability instances available to be consumed. Such a catalogue could be communicated to ALTO or even incorporated to it.

ALTO server queries could support TIFSA encoding in order to retrieve proper maps from ALTO. Additionally, filtered queries for particular characteristics of a flavor could also be supported.

4.2. Association of Compute Capabilities to Network Topology

It is required to associate the location of the available instances with topological information to allow ALTO construct the overall map. The expectation is that the management of the network and cloud capabilities will be performed by the same entity, producing an integrated map to handle both network and compute abstractions jointly. While this can be straightforward when an ISP owns both the network and the cloud infrastructure, it can in general require collaboration between multiple administrative domains. Details on potential scenarios will be provided in future versions of this document.

At this stage, four potential solutions could be considered:

* To leverage (and possibly extend) [I-D.ietf-teas-sf-aware-topo-model] for disseminating topology information together with the notion of function location (that would require to be adapted to the existence of available compute capabilities). A recent effort in this direction can be found in [I-D.llc-teas-dc-aware-topo-model].

* To extend BGP-LS [RFC7752], which is already considered as a mechanism for feeding topology information in ALTO, in order to also advertise computing capabilities as well.

* To combine information from the infrastructure profiles catalogue with topological information by leveraging the IP prefixes allocated to the gateway providing connectivity to the NFVI PoP.

* To integrate with Cloud Infrastructure Managers that could expose cloud infrastructure capabilities as in [CNTT] and [GSMA].

The viability of these options will be explored in future versions of this document.

4.3. ALTO Architecture for Determining Serve Edge

The following logical architecture defines the usage of ALTO for determining service edges.

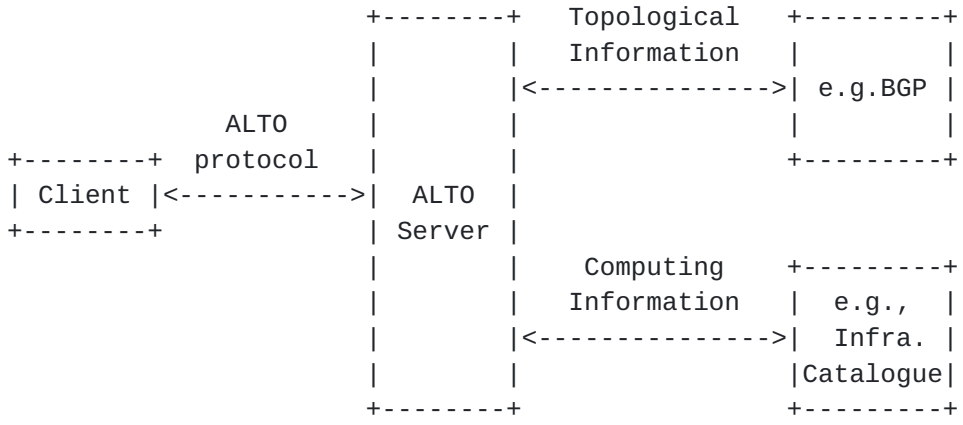

Figure 2: Service Edge Information Exchange.

In order to select the optimal edge server from both the network (e.g., the path with lower latency and/or higher bandwidth) and the cloud perspectives (e.g., number of CPUs/GPUs, available RAM and storage capacity), there is a need to see the edge server as both an IP entity (as in [RFC7285]) and an Abstract Network Element (ANE) entity (as in [RFC9275]).

Currently there is no mechanism (neither in [RFC9275] nor [RFC9240]) to see the same edge server as an entity in both domains. The design of ALTO, however, allows extensions that could be used to identify that an entity can be defined in several domains. These different domains and their related properties can be specified in extended ALTO property maps, as proposed in the next sections.

5. ALTO Design Considerations for Determining Service Edge

This section is in progress and gathers the ALTO features that are needed to support the exposure of both networking capabilities and compute capabilities in ALTO Maps.

In particular, ALTO Entity Property Maps defined in [RFC9240] can be extended. [RFC9240] generalizes the concept of endpoint properties to domains of other entities through property maps. Entities can be defined in a wider set of entity domains such as IPv4, IPv6, PID, AS, ISO3166-1 country codes or ANE. In addition, RFC 9240 specifies how properties can be defined on entities of each of these domains.

5.1. Example of Entity Definition in Different Domains

As there can be applications that do not necessarily need both compute and networking information, it is fine to keep the entity domains separate, each with their own native properties. However, some applications need information on both topics, and a scalable and flexible solution consists in defining an ALTO property type, that:

* Indicates that an entity can be defined in several domains;

* Specifies, for an entity, the other domains where this entity is defined.

For instance, one possible approach is to introduce entity properties that list other entity domains where an edge server is identified. This property type should be usable in all entity domains types. The following provides an example where the property "entity domain mapping" lists the other domains in which an entity is defined.

Suppose an edge server is identified as follows:

* In the IPV4 domain, with an IP address, e.g., ipv4:1.2.3.4;

* In the ANE domain, with an identifier, e.g., ane:DC10-HOST1.

To get information on this edge server as an entity in the "ipv4" entity domain, an ALTO client can query the properties "pid" and "entity-domain-mappings" and obtain a response as follows:

```
POST /propmap/lookup/dc-ip HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: TBC
Content-Type: application/alto-propmapparams+json
{
"entities" : ["ipv4:1.2.3.4"],
"properties" : [ "pid", "entity-domain-mappings"]
}
```

```
HTTP/1.1 200 OK
Content-Length: TBC
Content-Type: application/alto-propmap+json
{
"meta" : {},

"property-map":  {
    "ipv4:1.2.3.4" :
      {"pid" : DC10,
        "entity-domain-mappings" : ["ane"]}
    }
}
```

To get information on this edge server as an entity in the "ane" entity domain, an ALTO client can query the properties "entity-domain-mappings" and "network-address" and obtain a response as follows:

```
POST /propmap/lookup/dc-ane HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: TBC
Content-Type: application/alto-propmapparams+json
{
"entities" : ["ane:DC10-HOST1"],
"properties" : [ "entity-domain-mappings", "network-address"]
}

HTTP/1.1 200 OK
Content-Length: TBC
Content-Type: application/alto-propmap+json
{
"meta" : {},

"property-map":  {
  "ane:DC10-HOST1"
      {"entity domain mappings :  ["ipv4"]",
        "network-address" :  ipv4:1.2.3.4}
      }
}
```

Thus, if the ALTO Client sees the edge server as an entity with a network address, it knows that it can see the server as an ANE on which it can query relevant properties.

Further elaboration will be provided in future versions of this document.

5.2. Definition of Flavors in ALTO Property Map

The ALTO Entity Property Maps [RFC9240] generalize the concept of endpoint properties to domains of other entities through property maps. The term "flavor" or "instance" refers to an abstracted set of computing resources, with well-specified properties such as CPU, RAM and Storage. Thus, a flavor can be seen as an ANE with properties defined in terms of TIFSA. A flavor or instance is a group of 1 or more elements that can be reached via one or more network addresses. So an instance can also be seen as a PID that groups one or more IP addresses. In a context such as the one defined in CNTT, an ALTO property map could be used to expose TIFSA information of potential candidate flavors, i.e., potential NFVI-PoPs where an application or service can be deployed.

Figure 3 below depicts an example of a typical edge-cloud scenario [RFC9275] where each ANE represents a flavor/instance that resides on different cloud servers. Flavors on the "on-premise" edge nodes have

limited resources (but are closer to the end hosts), and flavors on the site-radio edge node and access central office (CO) have more available resources.

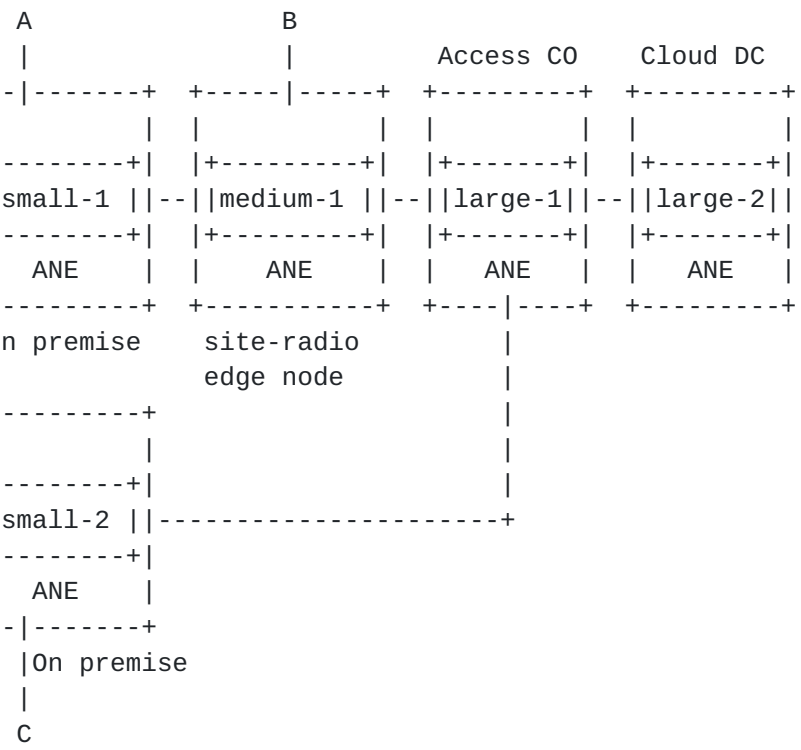

Figure 3: Example Use Case for Service Edge.

Based on the reference scenario (Figure 3), Figure 4 shows fictitious TIFSA property types for entities of domain type "ane":

flavor -name	type (T)	inter face (I)	f-c pu (F)	f-ra m (F)	f-di sk (F)	f-b w (F)	S	A
small-1	basic	1 Gbps	1	512 MB	1 GB	1 G bps		
small-2	network-intensive	9 Gbps	1	512 MB	1 GB	1 G bps		
medium-1	network-intensive	25 Gbps	2	4 GB	40 GB	1 G bps		
large-1	compute-intensive	50 Gbps	4	8 GB	80 GB	1 G bps		
large-2	compute-intensive	100 Gbps	8	16 GB	160 GB	1 G bps		

Figure 4: ALTO Property Map.

Subsequently, an ALTO client may request flavor(s) information from source [A] to destinations [B,C]. The following is a simplified example of an ALTO client request and the corresponding response. Note that the response consists of two parts: (i) ANE array for each source and destination pair (out of the scope of this document), and (ii) the requested properties of ANEs.

```
POST /costmap/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;type=application/alto-costmap+json,
        application/alto-error+json
Content-Length: 163
Content-Type: application/alto-costmapfilter+json

{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "pids": {
    "srcs": [ "A" ],
    "dsts": [ "B", "C" ]
  },
  "ane-property-names": [
    "type",
```

```
      "cpu",
      "ram",
      "disk"
    ]
  }

  HTTP/1.1 200 OK
  Content-Length: 952
  Content-Type: multipart/related; boundary=example-1;
                type=application/alto-costmap+json

  Content-ID: <costmap@alto.example.com>
  Content-Type: application/alto-costmap+json

  {
    "meta": {
      "vtag": {
        "resource-id": "filtered-cost-map-pv.costmap",
        "tag": "d827f484cb66ce6df6b5077cb8562b0a"
      },
      "dependent-vtags": [
        {
          "resource-id": "my-default-networkmap",
          "tag": "c04bc5da49534274a6daeee8ea1dec62"
        }
      ],
      "cost-type": {
        "cost-mode": "array",
        "cost-metric": "ane-path"
      }
    },
    "cost-map": {
      "A": {
        "B": [ "small-1", "medium-1"],
        "C": [ "small-1", "medium-1", "large-1", "small-2" ]
      }
    }
  }

  Content-ID: <propmap@alto.example.com>
  Content-Type: application/alto-propmap+json

  {
      "meta" : {
      },
      "property-map": {
        ".ane:small-1":
          {"type" : "basic", "cpu" : 1,
```

```
            "ram" : "512MB", "disk" : 1GB},
        ".ane:small-2":
          {"type" : "network-intensive", "cpu" : 1,
            "ram" : "512MB", "disk" : 1GB},
        ".ane:medium-1":
          {"type" : "compute-intensive", "cpu" : 2,
            "ram" : "4GB", "disk" : 40GB},
        ".ane:large-1":
         {"type" : "compute-intensive", "cpu" : 4,
           "ram" : "8GB", "disk" : 80GB}
      }    }
```

6. Use Cases

6.1. Open Abstraction for Edge Computing

As shown in this document, modern applications such as AR/VR, V2X, or IoT, require bringing compute closer to the edge in order to meet strict bandwidth, latency, and jitter requirements. While this deployment process resembles the path taken by the main cloud providers (notably, AWS, Facebook, Google and Microsoft) to deploy their large-scale datacenters, the edge presents a key difference: datacenter clouds (both in terms of their infrastructure and the applications run by them) are owned and managed by a single organization, whereas edge clouds involve a complex ecosystem of operators, vendors, and application providers, all striving to provide a quality end-to-end solution to the user. This implies that, while the traditional cloud has been implemented for the most part by using vertically optimized and closed architectures, the edge will necessarily need to rely on a complete ecosystem of carefully designed open standards to enable horizontal interoperability across all the involved parties. This document envisions ALTO playing a role as part of the ecosystem of open standards that are necessary to deploy and operate the edge cloud.

As an example, consider a user of an XR application who arrives at his/her home by car. The application runs by leveraging compute capabilities from both the car and the public 5G edge cloud. As the user parks the car, 5G coverage may diminish (due to building interference) making the home local Wi-Fi connectivity a better choice. Further, instead of relying on computational resources from the car and the 5G edge cloud, latency can be reduced by leveraging computing devices (PCs, laptops, tablets) available from the home edge cloud. The application's decision to switch from one domain to another, however, demands knowledge about the compute and communication resources available both in the 5G and the Wi-Fi domains, therefore requiring interoperability across multiple industry standards (for instance, IETF and 3GPP on the public side,

and IETF and LF Edge [LF-EDGE] on the private home side). ALTO can be positioned to act as an abstraction layer supporting the exposure of communication and compute information independently of the type of domain the application is currently residing in.

Future versions of this document will elaborate further on this use case.

6.2. Optimized placement of microservice components

Current applications are transitioning from a monolithic service architecture towards the composition of microservice components, following cloud-native trends. The set of microservices can have associated SLOs which impose constraints not only in terms of required compute resources (CPU, storage, ...) dependent on the compute facilities available, but also in terms of performance indicators such as latency, bandwidth, etc, which impose restrictions in the networking capabilities connecting the computing facilities. Even more complex constrains, such as affinity among certain microservices components could require complex calculations for selecting the most appropriate compute nodes taken into consideration both network and compute information.

Thus, service/application orchestrators can benefit from the information exposed by ALTO at the time of deciding the placement of the microservices in the network.

6.3. Distributed AI Workloads

Generative AI is a technological feat that opens up many applications such as holding conversations, generating art, developing a research paper, or writing software, among many others. Yet this innovation comes with a high cost in terms of processing and power consumption. While data centers are already running at capacity, it is projected that transitioning current search engine queries to leverage generative AI will increase costs by 10 times compared to traditional search methods [DC-AI-COST]. As (1) computing nodes (CPUs and GPUs) are deployed to build the edge cloud through technologies like 5G and (2) with billions of mobile user devices globally providing a large untapped computational platform, shifting part of the processing from the cloud to the edge becomes a viable and necessary step towards enabling the AI-transition. There are at least four drivers supporting this trend:

* Computational and energy savings: Due to savings from not needing large-scale cooling systems and the high performance-per-watt efficiency of the edge devices, some workloads can run at the edge at a lower computational and energy cost [EDGE-ENERGY], especially when considering not only processing but also data transport.

* Latency: For applications such as driverless vehicles which require real-time inference at very low latency, running at the edge is necessary.

* Reliability and performance: Peaks in cloud demand for generative AI queries can create large queues and latency, and in some cases even lead to denial of service. In some cases, limited or no connectivity requires running the workloads at the edge.

* Privacy, security, and personalization: A "private mode" allows users to strictly utilize on-device (or near-the-device) AI to enter sensitive prompts to chatbots, such as health questions or confidential ideas.

These drivers lead to a distributed computational model that is hybrid: Some AI workloads will fully run in the cloud, some will fully run in the edge, and some will run both in the edge and in the cloud. Being able to efficiently run these workloads in this hybrid, distributed, cloud-edge environment is necessary given the aforementioned massive energy and computational costs. To make optimized service and workload placement decisions, information about both the compute and communication resources available in the network is necessary too.

Consider as an example a large language model (LLM) used to generate text and hold intelligent conversations. LLMs produce a single token per inference, where a token is almost equivalent to a word. Pipelining and parallelization techniques are used to optimize inference, but this means that a model like GPT-3 could potentially go through all 175 billion parameters that are part of it to generate a single word. To efficiently run these computational-intensive workloads, it is necessary to know the availability of compute resources in the distributed system. Suppose that a user is driving a car while conversing with an AI model. The model can run inference on a variety of compute nodes, ordered from lower to higher compute power as follows: (1) the user's phone, (2) the computer in the car, (3) the 5G edge cloud, and (4) the datacenter cloud. Correspondingly, the system can deploy four different models with different levels of precision and compute requirements. The simplest model with the least parameters can run in the phone, requiring less compute power but yielding lower accuracy. Three other models ordered in increasing value of accuracy and computational complexity

can run in the car, the edge, and the cloud. The application can identify the right trade-off between accuracy and computational cost, combined with metrics of communication bandwidth and latency, to make the right decision on which of the four models to use for every inference request. Note that this is similar to the resolution/ bandwidth trade-off commonly found in the image encoding problem, where an image can be encoded and transmitted at different levels of resolution depending on the available bandwidth in the communication channel. In the case of AI inference, however, not only bandwidth is a scarce resource, but also compute. ALTO extensions to support the exposure of compute resources would allow applications to make optimized decisions on selecting the right computational resource, supporting the efficient execution of hybrid AI workloads.

7. Security Considerations

TODO Security

8. IANA Considerations

This document has no IANA actions.

9. Conclusions

Telco networks will increasingly contain a number of interconnected data centers and edge clouds of different sizes and characteristics, allowing flexibility in the dynamic deployment of functions and applications for advanced services. The overall objective of this document is to begin a discussion in the ALTO WG regarding the suitability of the ALTO protocol for determining where to deploy a function or application in these distributed computing environments. The result of these discussions will be reflected in future versions of this draft.

10. References

10.1. Normative References

[RFC7285] Alimi, R., Ed., Penno, R., Ed., Yang, Y., Ed., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, DOI 10.17487/RFC7285, September 2014, <https://www.rfc-editor.org/info/rfc7285>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

[RFC9275] Gao, K., Lee, Y., Randriamasy, S., Yang, Y., and J. Zhang, "An Extension for Application-Layer Traffic Optimization (ALTO): Path Vector", RFC 9275, DOI 10.17487/RFC9275, September 2022, <https://www.rfc-editor.org/info/rfc9275>.

[RFC9240] Roome, W., Randriamasy, S., Yang, Y., Zhang, J., and K. Gao, "An Extension for Application-Layer Traffic Optimization (ALTO): Entity Property Maps", RFC 9240, DOI 10.17487/RFC9240, July 2022, <https://www.rfc-editor.org/info/rfc9240>.

[I-D.ietf-alto-performance-metrics] Wu, Q., Yang, Y. R., Lee, Y., Dhody, D., Randriamasy, S., and L. M. Contreras, "ALTO Performance Cost Metrics", Work in Progress, Internet-Draft, draft-ietf-alto-performance-metrics-28, 21 March 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-alto-performance-metrics-28>.

[I-D.ietf-teas-sf-aware-topo-model] Bryskin, I., Liu, X., Lee, Y., Guichard, J., Contreras, L. M., Ceccarelli, D., Tantsura, J., and D. Shytyi, "SF Aware TE Topology YANG Model", Work in Progress, Internet-Draft, draft-ietf-teas-sf-aware-topo-model-11, 12 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-teas-sf-aware-topo-model-11>.

[I-D.llc-teas-dc-aware-topo-model] Lee, Y., Liu, X., and L. M. Contreras, "DC aware TE topology model", Work in Progress, Internet-Draft, draft-llc-teas-dc-aware-topo-model-03, 10 July 2023, <https://datatracker.ietf.org/doc/html/draft-llc-teas-dc-aware-topo-model-03>.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[CNTT] "Cloud Infrastructure Telco Taskforce Reference Model", https://github.com/cntt-n/CNTT/wiki , June 2022.

[GSMA] "Cloud Infrastructure Reference Model, Version 2.0", https://www.gsma.com/newsroom/wp-content/uploads//NG.126-v2.0-1.pdf , October 2021.

[Anuket] "Anuket Project", https://wiki.anuket.io/ , October 2022.

[LF-EDGE] "Linux Foundation Edge", https://www.lfedge.org/ , March 2023.

[EDGE-ENERGY] "Estimating energy consumption of cloud, fog, and edge computing infrastructures", IEEE Transactions on Sustainable Computing , 2019.

[DC-AI-COST] "Generative AI Breaks The Data Center - Data Center Infrastructure And Operating Costs Projected To Increase To Over $76 Billion By 2028", Forbes, Tirias Research Report , 2023.

Acknowledgments

The work of Luis M. Contreras has been partially funded by the European Union under the Horizon Europe project CODECO (COgnitive, Decentralised Edge- Cloud Orchestration), grant number 101092696.

Authors' Addresses

Luis M. Contreras
Telefonica
Email: luismiguel.contrerasmurillo@telefonica.com

Sabine Randriamasy
Nokia Bell Labs
Email: sabine.randriamasy@nokia-bell-labs.com

Jordi Ros-Giralt
Qualcomm Europe, Inc.
Email: jros@qti.qualcomm.com

Danny Lachos
Benocs
Email: dlachos@benocs.com

Christian Rothenberg
Unicamp
Email: chesteve@dca.fee.unicamp.br