Workgroup: cats
Internet-Draft: draft-ldbc-cats-framework-01
Published: 10 March 2023
Intended Status: Informational
Expires: 11 September 2023

Authors: C. Li, Ed. Z. Du
Huawei Technologies China Mobile
M. Boucadair, Ed. L. M. Contreras
Orange Telefonica
J. Drake G. Huang G. Mishra
Juniper Networks, Inc. ZTE Verizon Inc.

A Framework for Computing-Aware Traffic Steering (CATS)

Abstract

This document describes a framework for Computing-Aware Traffic Steering (CATS). Particularly, the document identifies a set of CATS components, describes their interactions, and exemplifies the workflow of the control and data planes.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
2. Terminology
3. Framework and Components
   3.1. Assumptions
   3.2. CATS Identifiers
   3.3. CATS Components
      3.3.1. Edge Sites and Services Instances
      3.3.2. CATS Service Metric Agent (C-SMA)
      3.3.3. The CATS Network Metric Agent (C-NMA)
      3.3.4. CATS Path Selector (C-PS)
      3.3.5. CATS Traffic Classifier (C-TC)
      3.3.6. Overlay CATS-Routers
      3.3.7. Underlay Infrastructure
   3.4. Deployment Considerations
4. CATS Framework Workflow
   4.1. Provisioning of CATS Components
   4.2. Service Announcement
   4.3. Metrics Distribution
   4.4. Service Demand Processing
   4.5. Service Instance Affinity
5. Security Considerations
6. Privacy Considerations
7. IANA Considerations
8. Informative References
Appendix A. Acknowledgements
Contributors
Authors' Addresses

1. Introduction

Edge computing architectures have been expanding from single edge nodes to multiple, sometimes collaborative, edge nodes to address various issues (e.g., long response times or suboptimal service and network resource usage).

The underlying networking infrastructures that include edge computing resources usually provide relatively static service dispatching (that is, the selection of the sevice instances that will be invoked for a request). In such infrastructures, service-specific traffic is often directed to the closest edge resource from a routing perspective without considering the actual network state (e.g., traffic congestion conditions).

As described in [I-D.yao-cats-ps-usecases], traffic steering that takes into account computing resource metrics would benefit several services, including latency-sensitive service like immersive services that rely upon the use of augmented reality or virtual reality (AR/VR) techniques. This document provides an architectural framework that aims at facilitating the making of compute- and network-aware traffic steering decisions in networking environments where edge computing resources are deployed.

The Computing-Aware Traffic Steering (CATS) framework assumes that there may be multiple service instances running on different edge nodes, globally providing one given service. A single edge node may have limited computing resources available at a given time, whereas the various edge nodes may experience different resource availability issues over time. A single edge node may also host multiple instances of a service or just one service instance.

The CATS framework is an ingress-based overlay framework for the selection of the suitable service instance(s) from a set of instance candidates. The exact characterization of 'suitable' will be determined by a combination of networking and computing metrics. To that aim, the CATS framework assumes that edge nodes collaborate with each other under a single administrative domain to achieve a global objective of dispatching service demands (and thereby optimizing their processing by the most relevant edge computing resources) over the various and available edge computing resources, by taking into account both service instance status and network state (e.g., reachability considerations, path cost, and traffic congestion conditions).

Also, this document describes a workflow of the main CATS procedures that are executed in both the control and data planes.

2. Terminology

This document makes use of the following terms:

Client: An endpoint that is connected to a service provider network.

Computing-Aware Traffic Steering (CATS): A traffic engineering approach [I-D.ietf-teas-rfc3272bis] that takes into account the dynamic nature of computing resources and network state to optimize service-specific traffic forwarding towards a given service instance. Various relevant metrics may be used to enforce such computing-aware traffic steering policies.

CATS Service ID (CS-ID): An identifier representing a service, which the clients use to access it. See Section 3.2.

CATS Binding ID (CB-ID): An identifier of a single service instance or location of a given service instance (CS-ID). See Section 3.2.

Service: An offering provided by a service provider and which is delivered using one or more service functions [RFC7665].

Service instance: A run-time environment (e.g., a server or a process on a server) that makes a service instance available (i.e., up and running). One service can be accessed through multiple instances running at the same or different locations.

Service demand: The demand for a service identified by a CATS Service ID (CS-ID).

Service request: The request for a specific service instance.

CATS-Router: A network device (usually located at the edge of the network) that makes forwarding decisions based on CATS information to steer traffic specific to a service demand towards a corresponding yet selected service instance. The selection of a service instance relies upon a multi-metric CATS-based path computation. A CATS router may behave as Ingress or Egress CATS-Router.

Ingress CATS-Router: A node that serves as a service access point for CATS clients. It steers service-specific traffic along a CATS-computed path that leads to an Egress CATS-Router that connects to the most suitable edge site that hots the service instance selected to satisfy the initial service demand.

Egress CATS-Router: A node that is located at the end of a CATS-computed path and which connects to a CATS-serviced site.

CATS Service Metric Agent (C-SMA): An agent that is responsible for collecting service capabilities and status, and for reporting them to a CATS Path Selector (C-PS). See Section 3.3.2.

CATS Network Metric Agent (C-NMA): A functional entity that is responsible for collecting network capabilities and status, and for reporting them to a C-PS. See Section 3.3.3.

CATS Path Selector (C-PS): A computation logic that calculates and selects paths towards service locations and instances and which accommodates the requirements of service demands. Such a path computation engine takes into account the service and network status information. See Section 3.3.4.

CATS Traffic Classifier (C-TC): A functional entity that is responsible for determining which packets belong to a traffic flow for a particular service demand. It is also responsible for forwarding such packets along the C-PS computed path that leads to the relevant service instance. See Section 3.3.5.

3. Framework and Components

3.1. Assumptions

CATS assumes that there are multiple service instances running on different edge nodes, and which provide a given service that is represented by the same service identifier (see Section 3.2).

3.2. CATS Identifiers

CATS introduces the following identifiers:

CATS Service ID (CS-ID): An identifier representing a service, which the clients use to access it. Such an ID identifies all the instances of a given service, rgardless of their location. The CS-ID is independent of which service instance serves the service demand. Service demands are spread over the service instances that can accommodate them, considering the location of the initiator of the service demand and the availability (in terms of resource/traffic load, for example) of the service instances resource-wise among other considerations like traffic congestion conditions.

CATS Binding ID (CB-ID): An identifier of a single service instance or location of a given service instance (CS-ID).

3.3. CATS Components

The network nodes make forwarding decisions for a given service demand that has been received from a client according to both service instances and network status information. The main CATS functional elements and their interactions are shown in Figure 1.

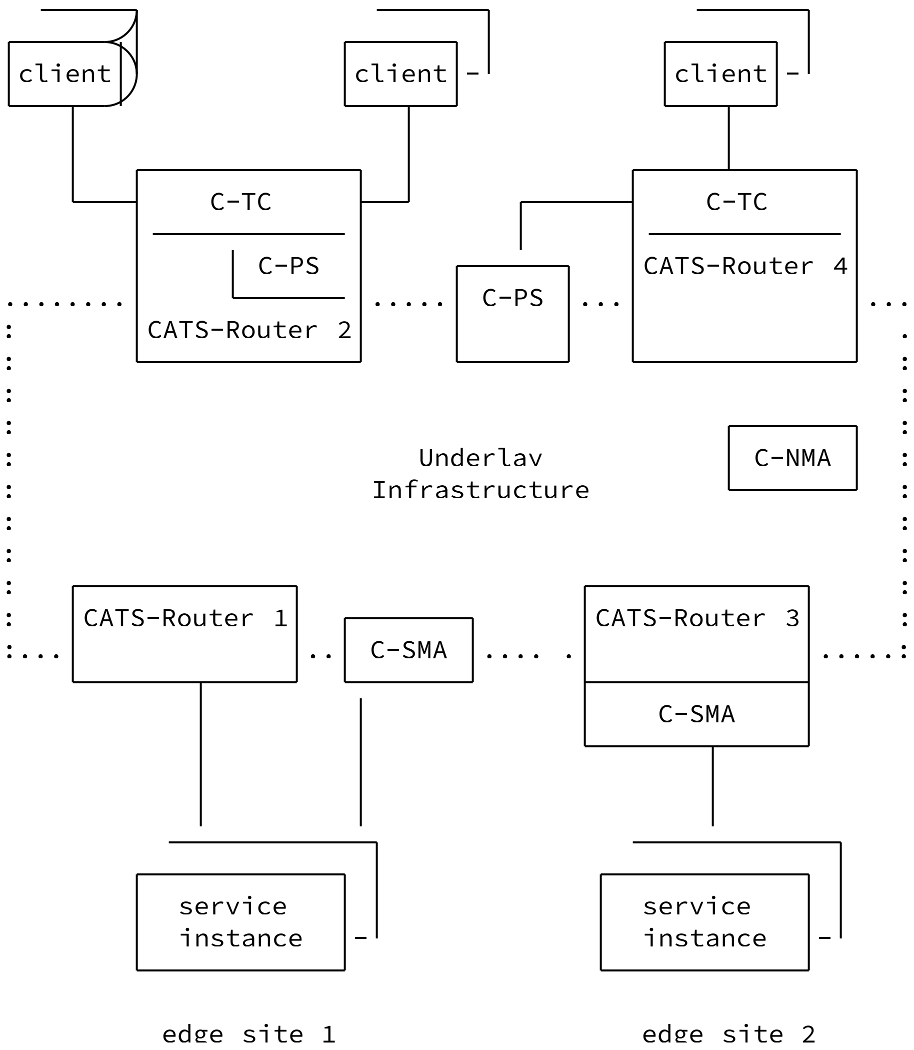

Figure 1: CATS Functional Components

3.3.1. Edge Sites and Services Instances

Edge sites (or edges for short) are the premises that provide access to edge computing resources. As mentioned in Section 3.2, a compute service (e.g., for face recognition purposes or a game server) is uniquely identified by a CATS Service IDentifier (CS-ID).

Service instances can be instantiated and accessed through different edge sites so that a single service can be represented and accessed by several instances that run in different regions of the network.

Figure 1 shows two edge nodes ("CATS-Router 1" and "CATS-Router 3") that provide access to service instances. These nodes behave as Egress CATS-Routers (Section 3.3.6).

> Note: "Egress" is used here in reference to the direction of the service request placement. The directionality is called to explicitly identify the exit node of the CATS infrastructure.

3.3.2. CATS Service Metric Agent (C-SMA)

The CATS Service Metric Agent (C-SMA) is a functional component that gathers information about edge sites and server resources, as well as the status of the different service instances. The C-SMAs are located adjacent to the service instances and can be hosted by the Egress CATS-Routers (Section 3.3.6) or located next to them.

Figure 1 shows one C-SMA embedded in "CATS-Router 3", and another C-SMA that is adjacent to "CATS-Router 1".

3.3.3. The CATS Network Metric Agent (C-NMA)

The CATS Network Metric Agent (C-NMA) is a functional component that gathers information about the state of the network. The C-NMAs may be implemented as standalone components or may be hosted by other components, such as CATS-Routers or CATS Path Selectors (C-PS) (Section 3.3.4).

Figure 1 shows a single, standalone C-NMA within the underlay network. There may be one or more C-NMAs for an underlay network.

3.3.4. CATS Path Selector (C-PS)

The C-SMAs and C-NMAs share the collected information with CATS Path Selectors (C-PSes) that use such information to select the Egress CATS-Routers (and potentially the service instances) where to forward traffic for a given service demand. C-PSes also determine the best paths (possibly using tunnels) to forward traffic, according to various criteria that include network state and traffic congestion conditions. The collected information is encoded into one or more metrics that feed the C-PS path computation logic. Such an information also includes CS-ID and possibly CB-ID identifiers.

There may be one or more C-PSes used to compute CATS paths. They can be integrated into CATS-Routers (e.g., "CATS-Router 2" in Figure 1) or they may be standalone components that communicate with CATS-Routers (e.g., "CATS-Router 4" in Figure 1).

3.3.5. CATS Traffic Classifier (C-TC)

CATS Traffic Classifier (C-TC) is a functional component that is responsible for associating incoming packets with existing service demands. CATS classifiers also ensure that packets that are bound to a specific service instance are all forwarded along the same path that leads to the same service instance, as instructed by a C-PS.

CATS classifiers are typically hosted in CATS routers that are located at the edge of the network.

3.3.6. Overlay CATS-Routers

The Egress CATS-Routers are the endpoints that behave as an overlay egress for service requests that are forwatded over a CATS infrastructure. An edge location that hosts service instances may be connected to one or more Egress CATS routers (that is, multi-homing is of course a design option). If a C-PS has selected a specific service instance and the C-TC has marked the traffic with the CB-ID, the Egress CATS-Router then forwards traffic to the relevant service instance. In some cases, the choice of the service instance may be left open to the Egress CATS-Router (i.e., traffic is marked only with the CS-ID). In such cases, the Egress CATS-Router selects a service instance using its knowledge of service and network capabilities as well as the current load as observed by the CATS router, among other considerations. Absent explicit policy, an Egress CATS-Router must make sure to forward all packets that pertain to a given service demand towards the same service instance.

Note that, depending on the design considerations and service requirements, per-service instance computing-related metrics or aggregated per-site computing related metrics (and a combination thereof) can be used by a C-PS. Using aggregated per-site computing related metrics appears as a privileged option scalability-wise, but relies on Egress CATS-Routers that connect to various service instances to select the proper service instance.

3.3.7. Underlay Infrastructure

The "underlay infrastructure" in Figure 1 indicates an IP/MPLS network that is not necessarily CATS-aware. The CATS paths that are computed by a P-CS will be distributed among the overlay CATS-Routers (Section 3.3.6), and will not affect the underlay nodes.

A CATS implementation may rely upon a control or management plane to distribute service metrics and network metrics - this document does not define a specific solution.

3.4. Deployment Considerations

This document does not make any assumption about how the various CATS functional elements are implemented and deployed. Concretely, whether a CATS deployment follows a fully distributed design or relies upon a mix of centralized (e.g., a C-PS) and distributed CATS functions (e.g., CATS traffic classifiers) is deployment-specific and may reflect the savoir-faire of the (CATS) service provider.

Centralized designs where the computing related metrics from the C-SMAs are collected by a (logically) centralized path computation logic (e.g., a Path Computation Element (PCE) [RFC4655]) that also collects network metrics may be adopted. In the latter case, the CATS computation logic may process incoming service requests to compute and select paths and, therefore, service instances. The outcomes of such a computation process may then be communicated to CATS traffic classifiers (C-TCs).

4. CATS Framework Workflow

The following subsections provide an overview of how the CATS workflow operates assuming a distributed CATS design.

4.1. Provisioning of CATS Components

TBC: --detail required provisioning at CAST elements (booptsrapping, credentials of peer CAST nodes, services, optimization metrics per service, etc.)--

4.2. Service Announcement

A service is associated with a unique identifier called a CS-ID. A CS-ID may be a network identifier, such as an IP address. The mapping of CS-IDs to network identifiers may be learned through a name resolution service, such as DNS [RFC1034].

4.3. Metrics Distribution

As described in Section 3.3, a C-SMA collects both service-related capabilities and metrics, and associates them with a CS-ID that identifies the service. The C-SMA may aggregate the metrics for multiple service instances, or maintain them separately or both. The C-SMA then advertises the CS-IDs along with the metrics to be received by all C-PSes in the network. The service metrics include computing-related metrics and potentially other service-specific metrics like the number of end-users who access the service instance at any given time, their location, etc.

Computing metrics may change very frequently (see [I-D.yao-cats-ps-usecases] for a discussion). How frequently such information is distributed is to be determined as part of the specification of any communication protocol (including routing protocols) that may be used to distribute the information. Various options can be considered, such as (but not limited to) interval-based updates, threshold-triggered updates, or policy-based updates.

Additionally, the C-NMA collects network-related capabilities and metrics. These may be collected and distributed by existing routing protocols, although extensions to such protocols may be required to carry additional information (e.g., link latency). The C-NMA distributes the network metrics to the C-PSes so that they can use the combination of service and network metrics to determine the best Egress CATS-Router to provide access to a service instance and invoke the compute function required by a service demand.

Network metrics may also change over time. Dynamic routing protocols may take advantage of some information or capabilities to prevent the network from being flooded with state change information (e.g., Partial Route Computation (PRC) of OSPFv3 [RFC5340]). C-NMAs should also be configured or instructed like C-SMAs to determine when and how often updates should be notified to the C-PSes.

Figure 2 shows an example of how CATS metrics can be distributed. There is a client attached to the netowrk via "CATS-Router 1". There are three instances of the service with CS-ID "1": two are located at "Edge Site 2" attached via "CATS-Router 2" and have CB-IDs "1" and "2"; the third service instance is located at "Edge Site 3" attached via "CATS-Router 3" and with CB-ID "3". There is also a second service with CS-ID "2" with only one service instance located at "Edge Site 2".

In Figure 2, the C-SMA collocated with "CATS-Router 2" distributes the service metrics for both service instances (i.e., (CS-ID 1, CB-ID 1) and (CS-ID 1, CB-ID 2)). Note that this information may be aggregated into a single advertisement, but in this case, the metrics for each service instance are indicated separately. Similarly, the C-SMA agent located at "Edge Site 2" advertises the service metrics for the two services hosted by "Edge Site 2".

The service metric advertisements are processed by the C-PS hosted by "CATS-Router 1". The C-PS also processes network metric advertisements sent by the C-NMA. All metrics are used by the C-PS to compute and select the most relevant path that leads to the Egress CATS-Router according to the initial client's service demand, the service that is requested ("CS-ID 1" or "CS-ID 2"), the state of the service instances as reported by the metrics, and the state of the network.

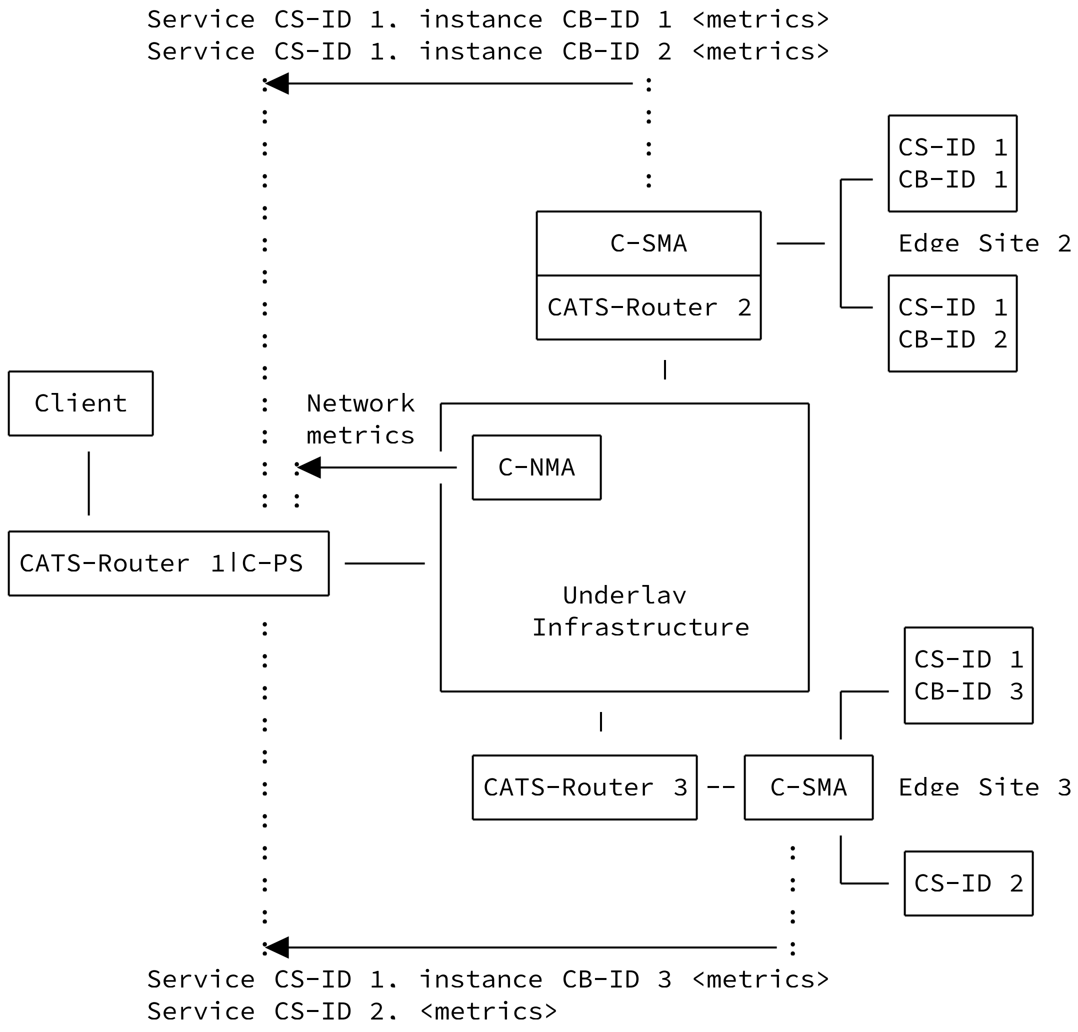

Figure 2: Example CATS Metric Distribution

The example in Figure 2 mainly describes a per-instance computing-related metric distribution. In the case of distributing aggregated per-site computing-related metrics, the per-instance CB-ID information will not be included in the advertisement. Instead, a per-site CB-ID may be used in case multiple sites are connected to the Egress CATS-Router to explicitly indicate the site the aggregated metrics come from.

A CB-ID is not required if the edge site can support consistently service instance selection.

4.4. Service Demand Processing

The C-PS computes paths that lead to Egress CATS-Routers according to the service and network metrics that have been advertised. The C-

PS may be collocated with an Ingress CATS-Router (as shown in Figure 2) or logically centralized.

This document does not specify any algorithm for path computation and selection purposes, but it is expected that a service demand or local policy may feed the C-PS computation logic with Objective Functions that provide some information about the path characteristics (e.g., in terms of maximum latency) and the selected service instance.

In the example shown in Figure 2, when the client sends a service demand to "CATS-Router 1", the router solicits the C-PS to select a service instance hosted by an edge site that can be accessed through a particular Egress CATS-Router. The C-PS also determines a path to that Egress CATS-Router. This information is provided to the Ingress CATS-Router ("CATS-Router 1") so that it can forward packets to their proper destination, as computed by the C-PS.

A service transaction consists of one or more service packets sent by the client to an Ingress CATS-Router to which the client is connected to. The Ingress CATS-Router classifies incoming packets received from clients by soliciting the CATS classifier (C-TC). When a matching classification entry is found for the packets, the Ingress CATS-Router encapsulates and forwards them to the C-PS selected Egress CATS-Router. When these packets reach the Egress CATS-Router, the outer header of the possible overlay encapsulation is removed and inner packets are sent to the relevant service instance.

> Note that multi-homed clients may be connected to multiple CATS domains that may be operated by the same or distinct service providers. This version of the framework does not cover multihoming specifics.

4.5. Service Instance Affinity

Instance affinity means that packets that belong to a flow associated with a service should always be sent to the same Egress CATS-Router which will forward them to the same service instance. Furthermore, packets of a given flow should be forwarded along the same path to avoid mis-ordering and to prevent the introduction of unpredictable latency variations.

The affinity is determined at the time of newly formulated service demands.

Note that different services may have different notions of what constitutes a 'flow' and may, thus, identify a flow differently. Typically, a flow is identified by the 5-tuple transport coordinates (source and destination addresses, source and destination port numbers, and protocol). However, for instance, an RTP video stream may use different port numbers for video and audio channels: in that case, affinity may be identified as a combination of the two 5-tuple flow identifiers so that both flows are addressed to the same service instance.

Hence, when specifying a protocol to communicate information about service instance affinity, a certain level of flexibility for identifying flows should be supported. Or, from a more general perspective, there should be a flexible mechanism to specify and identify the set of packets that are subject to a service instance affinity.

More importantly, the means for identifying a flow for the purpose of ensuring instance affinity should be application-independent to avoid the need for service-specific instance affinity methods. However, service instance affinity information may be configurable on a per-service basis. For each service, the information can include the flow/packets identification type and means, affinity timeout value, etc.

This document does not define any mechanism for defining or enforcing service instance affinity.

5. Security Considerations

The computing resource information changes over time very frequently, especially with the creation and termination of service instances. When such an information is carried in a routing protocol, too many updates may affect network stability. This issue could be exploited by an attacker (e.g., by spawning and deleting service instances very rapidly). CATS solutions must support guards against such misbehaviors. For example, these solutions should support aggregation techniques, dampening mechanisms, and threshold-triggered distribution updates.

The information distributed by the C-SMA and C-NMA agents may be sensitive. Such information could indeed disclose intel about the network and the location of compute resources hosted in edge sites. This information may be used by an attacker to identify weak spots in an operator's network. Furthermore, such information may be modified by an attacker resulting in disrupted service delivery for the clients, up to and including misdirection of traffic to an attacker's service implementation. CATS solutions must support authentication and integrity-protection mechanisms between C-SMAs/C-NMAs and C-PSes, and between C-PSes and Ingress CATS-Routers. Also, C-SMA agents need to support a mechanism to authenticate the services for which they provide information to C-PS computation logics, among other CATS functions.

6. Privacy Considerations

Means to prevent that on-path nodes in the underlay infrastructure to fingerprint and track clients (e.g., determine which client accesses which service) must be supported by CATS solutions. More generally, personal data must not be exposed to external parties by CATS beyond what is carried in the packet that was originally issued by the client.

Since the service will, in some cases, need to know about applications, clients, and even user identity, it is likely that the C-PS computed path information will need to be encrypted if the client/service communication is not already encrypted.

For more discussion about privacy, refer to [RFC6462] and [RFC6973].

7. IANA Considerations

This document makes no requests for IANA action.

8. Informative References

[I-D.ietf-teas-rfc3272bis]
Farrel, A., "Overview and Principles of Internet Traffic Engineering", Work in Progress, Internet-Draft, draft-ietf-teas-rfc3272bis-22, 27 October 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-teas-rfc3272bis-22>.

[I-D.yao-cats-ps-usecases]
Yao, K., Eardley, P., Trossen, D., Boucadair, M., Contreras, L. M., Li, C., Li, Y., and P. Liu, "Computing-Aware Traffic Steering (CATS) Problem Statement and Use Cases", Work in Progress, Internet-Draft, draft-yao-cats-ps-usecases-00, 3 March 2023, <https://datatracker.ietf.org/doc/html/draft-yao-cats-ps-usecases-00>.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/rfc/rfc1034>.

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <https://www.rfc-editor.org/rfc/rfc4655>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/rfc/rfc5340>.

[RFC6462] Cooper, A., "Report from the Internet Privacy Workshop", RFC 6462, DOI 10.17487/RFC6462, January 2012, <https://www.rfc-editor.org/rfc/rfc6462>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/rfc/rfc6973>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/rfc/rfc7665>.

Appendix A. Acknowledgements

The authors would like to thank Joel Halpern, John Scudder, Dino Farinacci, Adrian Farrel, Cullen Jennings, Linda Dunbar, Jeffrey Zhang, Peng Liu, Fang Gao, Aijun Wang, Cong Li, Xinxin Yi, Jari Arkko, Mingyu Wu, Haibo Wang, Xia Chen, Jianwei Mao, Guofeng Qian, Zhenbin Li, and Xinyue Zhang for their comments and suggestions.

Contributors

Huijuan Yao
China Mobile

Email: yaohuijuan@chinamobile.com

Yizhou Li
Huawei Technologies

Email: liyizhou@huawei.com

Dirk Trossen
Huawei Technologies

Email: dirk.trossen@huawei.com

Luigi Iannone
Huawei Technologies

Email: luigi.iannone@huawei.com

Hang Shi
Huawei Technologies

Email: shihang9@huawei.com

Changwang Lin
New H3C Technologies

Email: linchangwang.04414@h3c.com

Xueshun Wang
CICT

Email: xswang@fiberhome.com

Xuewei Wang
Ruijie Networks

Email: wangxuewei1@ruijie.com.cn

Christian Jacquenet
Orange

Email: christian.jacquenet@orange.com

Authors' Addresses

Cheng Li (editor)
Huawei Technologies
China

Email: c.l@huawei.com

Zongpeng Du
China Mobile
China

Email: duzongpeng@chinamobile.com

Mohamed Boucadair (editor)
Orange
France

Email: mohamed.boucadair@orange.com

Luis M. Contreras
Telefonica
Spain

Email: luismiguel.contrerasmurillo@telefonica.com

John E Drake
Juniper Networks, Inc.
United States of America

Email: jdrake@juniper.net

Guangping Huang
ZTE
China

Email: huang.guangping@zte.com.cn

Gyan Mishra
Verizon Inc.
United States of America

Email: hayabusagsm@gmail.com