TEAS
Internet-Draft
Intended status: Informational
Expires: 23 April 2026

L. M. Contreras
Telefonica
X. Liu
Alef Edge
20 October 2025

DC aware TE topology model
draft-llc-teas-dc-aware-topo-model-05

Abstract

This document proposes the extension of the TE topology model for including information related to data center resource capabilities. For that purpose, it defines a YANG module to augment TE topologies with awareness of data-center computing resources.

Although the model is designed to be compatible with TE aware topologies, it can also be applied to non-TE networks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 23 April 2026.

Table of Contents

1. Introduction

More and more service providers are deploying cloud computing facilities in order to host different kinds of services and applications. Such facilities can be generally referred as Datacenter Points of Presence (DC-PoPs). Those DCs will consist of a number of servers and networking elements for connecting all of them with the transport network. Depending on the number of servers in the data center, there will be distinct capabilities in terms of CPUs, memory and storage available for deploying and running the aforementioned services.

In such distributed and interconnected DC-PoPs, both computing and topological information are of interest for determining the optimal DC where to deploy a given service or application.

This document proposes a DC-aware extension for the topology model. This model is intended to co-exist and interoperate with existing traffic engineering (TE) topology models (e.g. ietf-te-topology [RFC8795] and technology-specific augmentations). It does not re-define the TE topology, but rather links cloud/DC resource entities to the network via attachment points / circuits, providing a bridge between the DC-aware model and the TE domain.

Although the model is designed to be compatible with TE aware topologies, it can also be applied to non-TE networks. The TE-related leafrefs are optional and only used when integration with TE models is required.

2. Datacenter information

The relevant information for datacenter capabilities can be described in different ways. One potential manner is to describe resource capabilities such as CPU, memory, storage, etc. This can be done in terms of total, used and free capacity for each of the parameters of interest. Cloud management systems allow to obtain such kind of information. For instance, in the case of Kubernetes it is possible to retrieve information about the total and allocatable resources in a compute node. Alternatively, in the case of OpenStack it is possible to collect information about the total resources and those currently in use from such total.

Another form of populating the information is by describing those resource capabilities as a bundled, usually referred as quota or flavor. Well known cloud computing providers like Amazon Web Services, Microsoft Azure or Google Cloud Platform follow such schema, bundling CPU, RAM and storage units as flavors. In the case of Amazon Web Services the bundle is known as “instance type”, while in Microsoft Azure is termed “virtual machine size” and in Google Cloud Platform it is called “machine type”.

Additional information to consider in both cases could refer to the management capabilities of the compute infrastructure, such as hypervisor details or virtualization technologies available.

The recent trend of cloud-native approach for the instantiation and deployment of service functions has positioned Kubernetes as the de-facto standard to manage containerized software in data centers, but it is also increasingly being used by telecommunication operators to manage compute resources at the edge.

Finally, all can be complemented with information related to the networking details for reaching the aforementioned compute capabilities (IP addresses, bandwidth, etc).

3. Relationship between TE and DC Resource Models

The modules proposed in this document define a data structure that allows a TE or network topology model to associate network nodes and links with data-center (DC) computing resources. The intent is to expose the awareness of available compute capabilities within the network view, without embedding full compute-domain models.

The design follows these principles:

* Separation of concerns: the ietf-dc-aware-topology module provides references towards cloud technology-specific models such as ietf-openstack-info and ietf-kubernetes-info. This helps to augment the model as long as the cloud-related technology evolves along the time.

* Attachment circuit abstraction [RFC9834]: Each DC reachable from the network is represented through one or more attachment circuits (ACs). The AC serves as the conceptual interface between the TE node and the computing domain. An attachment circuit may map to a physical port, a virtual link, or a service-level endpoint. This document adopts the AC semantics defined in [RFC9834].

* Integration with TE topology models: The model augments TE nodes and termination points defined in [RFC8795] (ietf-te-topology) with optional containers that reference DC resources connected through an attachment circuit. This allows a TE-aware orchestrator to take into account computing capacity, storage, or service location when performing path computation or placement decisions.

* Generic applicability: Although this module integrates with TE models, it is applicable to non-TE topologies as well. The augmentation is optional and can be ignored when TE-awareness is not required.

4. Model structure

This section provides the YANG modules that describe how cloud resources which can be associated with nodes in a TE topology. Three modules are defined:

* ietf-openstack-info: describes OpenStack-based systems.

* ietf-kubernetes-info: describes Kubernetes-based systems.

* ietf-dc-aware-topology: provides the augmentation of the TE topology model to expose attachment circuits from network nodes to the data center systems defined in the previous two modules.

4.1. Cloud technology-specific models

According to the distinct approaches for managing cloud-based resources different options could exist.

4.1.1. Hypervisor-based cloud solutions

A model structure for hypervisor-based cloud solutions (e.g., OpenStack) can be described in the following manner.

```
module: ietf-openstack-info
  +--rw dcpop
     +--rw dcpop-id?   string
     +--rw dc* [id]
        +--rw id              string
        +--rw attachment-circuit
        |  +--rw ac* [ac-id]
        |     +--rw ac-id            string
        |     +--rw node-ref?        string
        |     +--rw interface-name?  string
        |     +--rw bandwidth?       uint64
        |     +--rw status?          enumeration {up, down, degraded}
        +--rw openstack
           +--rw system
           |  +--rw system-id     string
           |  +--rw name          string
           |  +--rw region        string
           +--rw nodes
           |  +--rw node* [name]
           |     +--rw name       string
           |     +--rw cpu
           |     |  +--rw total        uint64
           |     |  +--rw allocated    uint64
           |     |  +--rw used         uint64
           |     +--rw memory
           |     |  +--rw total        uint64
           |     |  +--rw allocated    uint64
           |     |  +--rw used         uint64
           |     +--rw workloads
           |        +--rw max          uint32
           |        +--rw running      uint32
           +--rw workloads
              +--rw workload* [id]
                 +--rw id              string
                 +--rw name            string
                 +--rw project-id      string
                 +--rw cpu
                 |  +--rw allocated    uint64
                 |  +--rw limit        uint64
                 |  +--rw used         uint64
                 +--rw memory
                 |  +--rw allocated    uint64
                 |  +--rw limit        uint64
                 |  +--rw used         uint64
                 +--rw status
                    +--rw state        enumeration {active, stopped, error}
                    +--rw conditions*  string
```

4.1.2. Container-based cloud solutions

A model structure for container-based cloud solutions (e.g., Kubernetes) can be described in the following manner.

```
module: ietf-kubernetes-info
  +--rw dcpop
     +--rw dcpop-id?   string
     +--rw dc* [id]
        +--rw id               string
        +--rw attachment-circuit
        |  +--rw ac* [ac-id]
        |     +--rw ac-id            string
        |     +--rw node-ref?        string
        |     +--rw interface-name?  string
        |     +--rw bandwidth?       uint64
        |     +--rw status?          enumeration {up, down, degraded}
        +--rw kubernetes
           +--rw system
           |  +--rw system-id   string
           |  +--rw name        string
           |  +--rw location    string
           +--rw cluster
              +--rw cluster-id  string
              +--rw name        string
              +--rw location    string
              +--rw nodes
              |  +--rw node* [name]
              |     +--rw name      string
              |     +--rw cpu
              |     |  +--rw capacity      uint64
              |     |  +--rw allocatable   uint64
              |     |  +--rw usage         uint64
              |     +--rw memory
              |     |  +--rw capacity      uint64
              |     |  +--rw allocatable   uint64
              |     |  +--rw usage         uint64
              |     +--rw workloads
              |        +--rw max           uint32
              |        +--rw running       uint32
              +--rw workloads
                 +--rw workload* [id]
                    +--rw id              string
                    +--rw namespace       string
                    +--rw name            string
                    +--rw cpu
                    |  +--rw request      uint64
                    |  +--rw limit        uint64
```

```
                    |  +--rw usage        uint64
                    +--rw memory
                    |  +--rw request      uint64
                    |  +--rw limit        uint64
                    |  +--rw usage        uint64
                    +--rw status
                       +--rw phase        enumeration {pending, running, failed}
                       +--rw conditions*  string
```

4.2. Integration with TE model

The following structure provides the augmentation of the TE topology model to indicate the attachment of compute capabilities.

```
module: ietf-dc-aware-topology
  augment /nw:networks/nw:network/te:te-topology/te:node:
    +--rw dc-awareness?                          // presence container
       +--rw attachment-circuit* [ac-id]
          +--rw ac-id             string
          +--rw (dc-system-ref)?
             +--:(openstack)
             |  +--rw openstack-ref?    -> /os:dcpop/os:dc/os:openstack/os:system/os:syst
em-id
             +--:(kubernetes)
                +--rw kubernetes-ref?   -> /k8s:dcpop/k8s:dc/k8s:kubernetes/k8s:system/k8
s:system-id
```

The ietf-dc-aware-topology module augments the TE topology model so that a TE node can indicate awareness of attached data-center systems. Each attachment circuit (AC) represents a logical or physical link between the network domain and a data center of any type (i.e., central cloud, edge facilities, etc). In this way the network becomes aware of computing capabilities.

5. Security and operational considerations

The model is designed to be accessed via NETCONF [RFC6241], thus the security considerations for the NETCONF protocol are applicable here.

6. Informative References

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC8795] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Gonzalez de Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", RFC 8795, DOI 10.17487/RFC8795, August 2020, <https://www.rfc-editor.org/info/rfc8795>.

[RFC9834] Boucadair, M., Ed., Roberts, R., Ed., Gonzalez de Dios, O., Barguil, S., and B. Wu, "YANG Data Models for Bearers and Attachment Circuits as a Service (ACaaS)", RFC 9834, DOI 10.17487/RFC9834, September 2025, <https://www.rfc-editor.org/info/rfc9834>.

Acknowledgements

The work of L.M. Contreras has been partially funded by the European Union under the Horizon Europe projects NEMO (NExt generation Meta Operating system) grant number 101070118, and CODECO (COgnitive, Decentralised Edge-Cloud Orchestration), grant number 101092696.

Young Lee contributed to initial versions of this draft.

Authors' Addresses

Luis M. Contreras
Telefonica
Ronda de la Comunicacion, s/n
28050 Madrid
Spain
Email: luismiguel.contrerasmurillo@telefonica.com
URI: http://lmcontreras.com

Xufeng Liu
Alef Edge
Email: xufeng.liu.ietf@gmail.com