Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 30, 2018

I. Bryskin
Huawei Technologies
X. Liu
Jabil
V. Beeram
Juniper Networks
T. Saad
Cisco Systems Inc

October 27, 2017

ONF/T-API Services vs. IETF/YANG Models and Interfaces
draft-bryskin-teas-yang-ietf-vs-onf-01

Abstract

This document compares IETF YANG TE (Traffic Engineering) data model and ONF/T-API model.

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 http://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 April 30, 2018.

Table of Contents

1. Introduction

The success of T-SDN as an architecture depends to a large degree on the quality and widespread adoption of open standardized interfaces to/from T-SDN controllers, linking them flexibly into various hierarchies and confederations. Currently, the two most popular such interfaces are:

1. T-API developed by ONF;

2. RESTCONF/YANG [RFC7950] based on TE Topology and TE Tunnels models defined in [I-D.ietf-teas-yang-te-topo] and [I-D.ietf-teas-yang-te] documents respectively, the product of IETF TEAS WG.

The two interfaces have the close attention of network operators and vendors. There is a lot of confusion about their respective technical merits and "marketing" strengths, applications they can support, use cases they cover, and so on. Do they compete or could they somehow complement each other?

This memo is limited to a strictly technical comparison with the special focus on the models supporting the two interfaces, in particular, the semantics, relationships, informational flows and services they define. Our analysis suggests that the IETF models provide for implementation of powerful hierarchical T-SDN controller systems, supporting a broad range of client systems and use cases, and that in some identifiable respects, T-API appears to fall relatively shorter. This memo is largely organized around considering the identified "gaps".

2. Topology Service

2.1. Constrained Nodes

The T-API Topology service does not support the notion of blocking/constrained nodes. This means that if a T-API Topology service provider exposes to a client a topology with at least one node with constrained connectivity, e.g. the node can switch a potential TE path/connection, say, from interface (NodeEdge point) A to B, but not from A to C; there is no way for the provider to communicate the connection limitations to the client, thus making the provided TE topology unfit for the client's path computations. This is a serious issue because many transport physical switches and virtually all abstract composite nodes should be treated as blocking nodes.

Likewise, if a potential path source/destination node is constrained in such a way that the path may leave/enter the source/destination node over a link from a subset of (but not all) same-layer links connected to the node, the T-API Topology service provider has no way of communicating such a circumstance to the client.

The described issue is addressed in the IETF TE Topology model. A TE node's Connectivity Matrix attribute (Figure 1) fully describes the node's TE path/connection switching limitations, while a TE Tunnel Termination Point's (TTP's) Local Link Connectivity List attribute (Figure 2) describes the node's TE path/connection termination limitations with respect to each TTP hosted by the node in question.

```
   Basic Connectivity Matrix:          Detailed Connectivity Matrix:

   LTP-6/label-x <=> LTP-1/label-y     LTP-6/label-x <=> LTP-1/label-y
                                        (Cost c, Delay d, SRLB s, ...)
   LTP-5/label-x <=> LTP-2/label-y     LTP-5/label-x <=> LTP-2/label-y
                                        (Cost c, Delay d, SRLB s, ...)
   LTP-5/label-x <=> LTP-4/label-y     LTP-5/label-x <=> LTP-4/label-y
                                        (Cost c, Delay d, SRLB s, ...)
   LTP-4/label-x <=> LTP-1/label-y     LTP-4/label-x <=> LTP-1/label-y
                                        (Cost c, Delay d, SRLB s, ...)
   LTP-3/label-x <=> LTP-2/label-y     LTP-3/label-x <=> LTP-2/label-y
   ...                                 ...
```

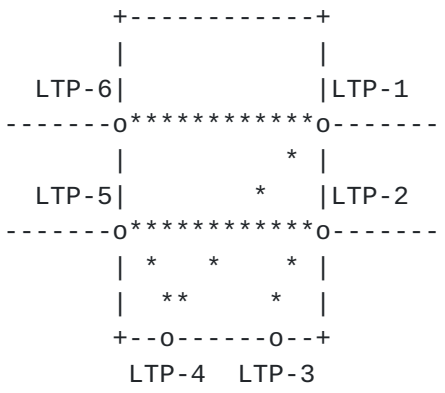

Figure 1: TE Node Connectivity Matrix

TTP-1 Basic LLCL:

```
TTP-1 <=> {LTP-5/label-x,
           LTP-2/label-y}
```

TTP-1 Detailed LLCL:

```
TTP-1 <=> {
 LTP-5/label-x,
  (Cost c, Delay d, SRLB s, ...),
 LTP-2/label-y,
  (Cost c, Delay d, SRLB s, ...)
 }
```

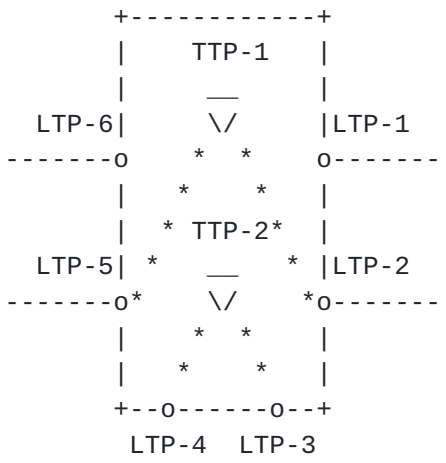

Figure 2: TTP Local Link Connectivity List

2.2. Intra-node Metrics

There is no good way for a T-API Topology service provider to articulate to the client what it would cost for a potential path (e.g., in terms of delay) to cross a node from interface (NodeEdge point) A to interface B. Because nodes (especially composite abstract nodes) may contribute to overall path costs much more than links connecting the nodes along the path, this fact makes the provided topology unfit for the client's path selection optimizations. [Note: To be fair, the T-API Topology service does allow a composite abstract node (representing a group of inter-connected nodes) to refer to the topology describing the abstract node's internals (node's encapTopology attribute). Hence the client may in theory apply path computation algorithms on the abstract node's internal/encapsulated topology to figure out whether the abstract node can switch a path between a given pair of the abstract node's NodeEdge points, as well as the cost penalties the path will accrue by doing so. However, such a technique defeats the whole purpose of creating the abstract node in the first place, which is hiding multiple topological elements behind the abstract node, so that the top level topology becomes smaller and easier to use in path computations. In other words, if the client has to "dive" into the abstract node's internal topology every time the client needs to

understand whether and how a path can cross the abstract node, the client would be better off if the abstract node were not provided, and instead, the node's internals were presented directly in the top level topology.]

This issue does not exist in the IETF TE Topology model. A TE node's Detailed Connectivity Matrix attribute (Figure 1, upper right) associates with each (abstract or physical) node's connectivity matrix entry a vector of costs (in terms of generic TE cost, delay, intra-node SRLGs, etc.) that a potential TE path will have to add to its end-to-end costs should the path select the entry to cross the node. Likewise, a TE path's source/destination TTP's Detailed Local Link Connectivity List attribute (Figure 2, upper right) indicates what it would cost for the path to start/stop on a given first/last link. [Note: In the IETF TE topology model an abstract TE node also points to the encapsulated TE topology describing the node's internals. However, the client is expected to peruse the node's encapsulated TE topology only in exceptional situations (e.g. during trouble shooting), rather than under normal conditions, such as routine path computations.]

2.3. Topology Updates

Suppose that a T-API Topology service client has requested and received a topology from one of its providers (for example, the topology presented in Figure 3). It is imperative that as soon as this done the provider starts updating the client (continuously and in unsolicited way) with changes happening to the topological elements and their attributes that the client has expressed interest in - otherwise, the client would be forced to make decisions on stale information.

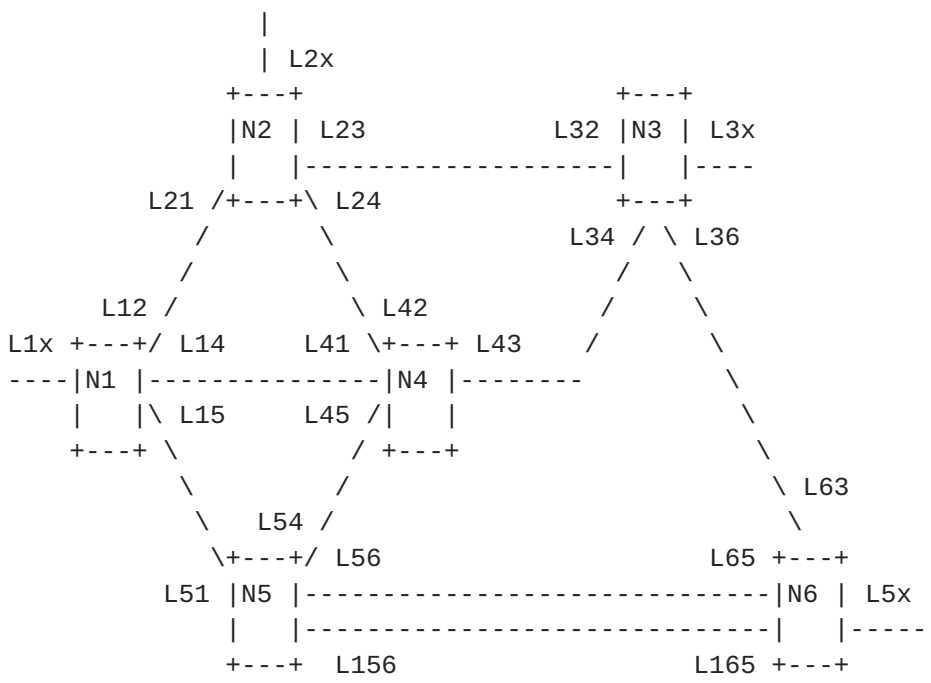

Figure 3: Topology presented to T-API Topology service client

The only way this could be done in T-API is via using T-API Notification service, specifically, the Attribute Value Change (AVC) Notification service, which in a nutshell works as follows:

- Provider registers with the service the types of pre-defined AVC events it is willing and capable of providing notifications for, along with the set of pre-defined object types that may comprise the notification contents;

- Client discovers the registered notifications it can subscribe to and subscribes to some of them, specifying filters to tailor the notifications to its needs.

There are two problems with this paradigm:

1. The client has a very limited way to express which notifications it is interested in, as well as the contents, triggers and frequency of such notifications. Note that even for the same topology element type (e.g., link) different clients may need to know different things, at different scopes and granularities, with respect to the attribute changes. For example, one client may want to hear about links that experienced changes in any attribute, while another client may be interested only in links with changes in specific attribute(s). One client may want to learn about link attribute modifications across all provided topologies, while another client may want to know only about such

links that belong to one or more specific (but not other) topologies. One client may want to receive in the notification the entire set of link attributes, while another client would want to learn only about incremental changes (i.e., changes that happened since the previous notification); some clients are interested not in just any attribute change, but rather, want to know when the attribute has reached a specific threshold, etc. As mentioned, a T-API client has only the option to discover what the provider is willing to offer (without the provider really knowing what their clients want to learn) and to subscribe to a subset of that;

2. In order for the client to understand/interpret the notifications registered by the provider, all notification event types, as well as the types of objects comprising the notification content, must be explicitly pre-defined. Considering the sheer number of, say, link attributes (especially, combinations of them) that different clients may be interested in, and the possible scopes, granularities and triggers of the notifications; explicit pre-definition of notifications is awkward, limited and impractical (if not infeasible).

In sharp contrast, the IETF TE topology model requires no explicit definition of notifications. When the client subscribes to a TE topology update notification it:

a. defines the notification event type by specifying the YANG XPath from the TE topology data store root to the data store node(s) associated with link attribute(s) encompassing the client's points of interest;

b. specifies another XPath pointing to the data store's sub-tree, node or group of nodes to identify the content of the notification and whether the entire new state or incremental changes must be provided;

c. defines the trigger for the notification, which could be any change in the node(s) of interest or a specific increment in value or the value hitting a specific threshold;

d. optionally defines the highest notification frequency at which the client wants to receive the notifications.

To illustrate this assume that the IETF TE Topology model client wants to be notified about all TE links whose available capacity has dropped below 10G, with the notification carrying the actual link's available capacity. In this case the client will:

a. specify root->all TE topologies -> all TE links->linkAttributes->bandwidth XPath as the notification type;

b. specify the same XPath to define the desired notification content;

c. define the notification trigger by specifying the low and high thresholds (e.g. 10G and 15 G respectively);

d. optionally specify the highest frequency of updates the client is capable/willing to consume.

Note that no explicit definitions for the notification were required. After the client registers with the provider the defined subscription, the latter knows exactly what the former wants to be notified about and how. Similar notifications are possible to register with the provider with respect to any TE topology element attribute or combination of thereof.

2.4. Topology Telemetry Collection

Topology service clients (which in the T-SDN context could be various controllers or applications, such as multi-domain coordinators, IP/transport integrators, orchestrators, big data collectors, analytics processors, network planners, etc.) are hungry for accurate real time network state information (a.k.a. network telemetry). This knowledge is instrumental for a client in keeping the network under its control healthy, stable and optimized under conditions of fiber cuts, hardware and software failures. In particular, network telemetry streams provided by the client's providers allow for the client to identify/predict failing network resources and route the provided transport/connectivity services away from them; to identify/predict points of congestion and eliminate/mitigate the congestion by deploying extra network capacity in a timely manner and so forth. Network telemetry is a valuable source of information useful for network planning, trouble shooting and many other things. Network telemetry is especially important for topology service clients because topologies represent - in an abstracted way - the physical network resources.

[Note: At the time of writing of this memo there were no known TAPI design/modeling activities related to telemetry streaming for any of the T-API services].

Topology telemetry collection is similar in nature to receiving updates on topology attribute changes. Per the description in section 1.3, T-API Notification service, State Change (SC) Notification service is the only mechanism theoretically (i.e. after

all the necessary modeling concepts and attributes, such as statistics counters, are in place) available for the client to subscribe and for the provider to stream the requested network telemetry. T-API SC Notification service has the same drawbacks as the AVC Notification service, specifically:

a. limited capability for the client to articulate what telemetry (event type, content, granularity, etc.) it seeks to receive;

b. necessity for explicit definition of the telemetry events and notification messages.

These issues do not exist in the network telemetry streaming machinery offered by the IETF Topology model. Let's consider, for example, that the client wants to identify "flipping" TE links (i.e. TE links frequently changing their UP/DOWN operational status) and obtain in the notification the entire state information for such TE links. In order to achieve this the client needs to:

a. specify root->all TE topologies -> all TE links->linkStatistics->linkUPCounter XPath as the notification type;

b. specify root->all TE topologies -> all TE links->linkState XPath to describe the desired notification content;

c. define the notification trigger by specifying the number the model data state node of interest (the linkUPCounter) must increment by for the next notification to be issued;

d. optionally specify the highest frequency of notifications of this type the client is capable/willing to consume.

2.5. Topology Name/Address Spaces

T-API topologies are required to have each node and link assigned a globally unique UUID. This means that all T-API Topology service clients and providers have to resolve potential UUID collisions via allocating the UUIDs from a universal name space governed by a centralized authority (in a similar way to how global IP addresses are assigned in IP networks).

The IETF TE Topology model allows for all TE topologies to have independent name spaces for the TE node, link and SRLG IDs, which not only eliminates the problem of ID collisions, but also greatly simplifies the design and implementation of network applications such as L0/L1 VPNs.

In Figure 4 a TE topology provider exposes its native (i.e. real, physical) TE topology as separate abstract TE topologies to two clients, each one customized separately on per client basis. According to the IETF TE Topology model each of the three depicted TE topologies may have an independent name space for their respective TE node, link and SRLG IDs.

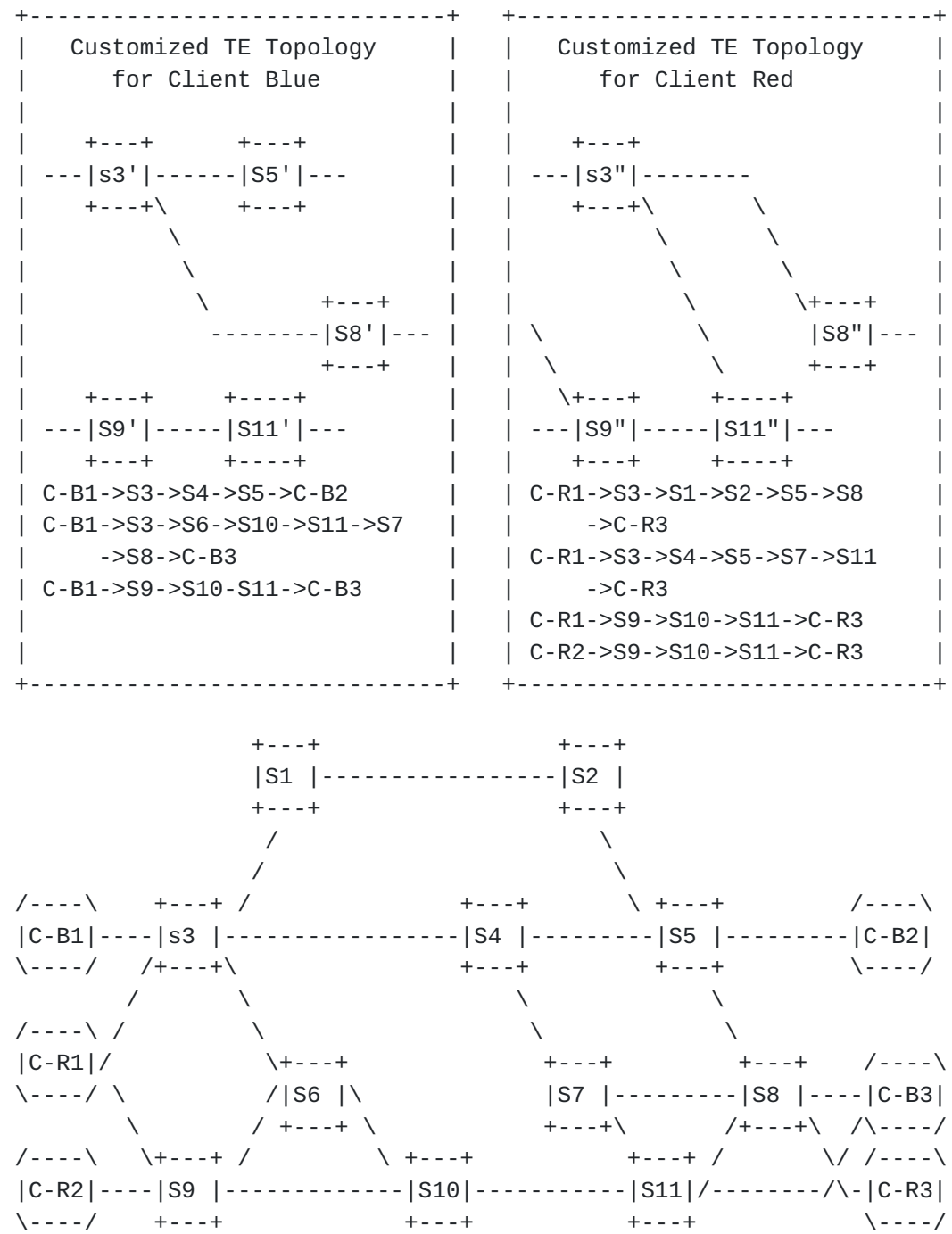

Figure 4: Abstract TE topologies customized for different clients

2.6. Topology Relationships

An IETF TE Topology model provider may expose to the same client multiple TE topologies, which:

- could be native (as known to the provider, unmodified) or abstract (generated by the provider as overlays based on native or lower level abstract TE topologies);
- could describe different layer networks in accordance with distinct layer-specific model augmentations;
- abstract TE topologies could be of a different type (e.g. single node, link mesh, etc.) and of a different hierarchy level;
- abstract TE topologies could be optimized based on different optimization criteria (e.g. smallest cost, shortest delay, best link protection, etc.)

The provider can convey to the client the TE topology optimization criteria, as well as the provider's preference as to the order in which the provided TE topologies are to be used via topology scope attributes specifically designed for this purpose. Furthermore, the TE Topology model defines various inter-topology relationships designed to describe abstract TE topology hierarchies, client-server layer network (vertical) relationships and domain neighboring (horizontal) relationships. The defined inter-topology relationships are as follows:

- TE node underlay topology: A composite abstract TE node of a higher hierarchy level TE topology X, representing a group of inter-connected TE nodes that belong to a lower hierarchy level TE topology Y, has an attribute pointing to Y (i.e., ID of the abstract TE node's internal/encapsulated TE topology);
- TE link underlay topology: A TE link of a TE topology X can point to TE Topology Y which was used by the provider to compute primary and backup TE paths that are (or are to be) used by the actual or potential TE tunnel (transport connectivity) supporting the TE link in question. The TE paths themselves could be provided in the same TE link attribute;
- Supporting node/link topology: A given TE node or link may show up in multiple TE topologies catered by the provider to the client. In order for the provider not to provide/update (and for the client not to consume) multiple identical sets of attributes, the model allows for providing/updating only for one (original) TE node/link, and having the "twins" point to the original TE mode/

link, as well as to the TE topology where the original TE node/ link could be found;

- Source node/link topology: A given TE node or link catered by the provider as a part of a TE topology to the client may be provided to the provider by one of its own providers. In such case the TE node/link in question can point to the original TE node/link, as well as to the TE topology where the original is defined, thus allowing for multi-level multi-provider TE topology hierarchies (see Figure 5);

- Inter-layer lock: This is the relationship/attribute that associates TE links of a higher layer network TE topology with TE Tunnel Termination Points (TTPs) of one or more lower layer network TE topology(ies) to articulate to the client inter-topology /inter-layer adaptation capabilities, to lock the TE topologies describing separate layer networks vertically, thus allowing for client multi-layer path computations and other multi-layer TE applications;

- Inter-domain plug: This is a relationship modeled via an inter-domain TE link attribute that allows for a client managing interconnected multi-domain networks (with each domain served by a separate provider) to identify neighboring domains and to lock the TE topologies provided by all providers horizontally, thus producing TE topologies homogeneously describing the entire multi-domain network and allowing for end-to-end path computations across the network.

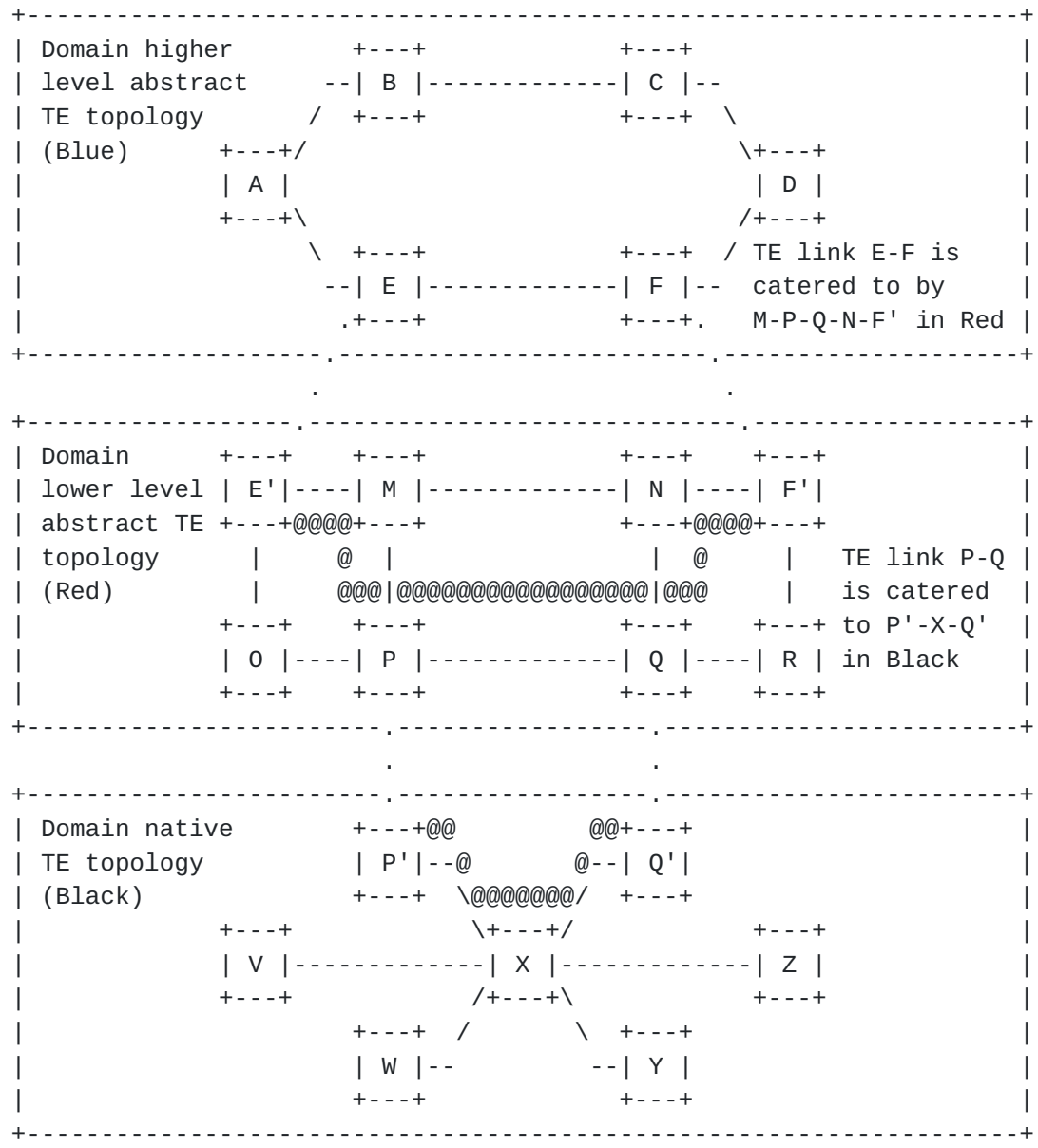

Figure 5: Hierarchical multi-provider abstract TE topologies

A T-API Topology service provider is also allowed to expose multiple topologies to the client. The only inter-topology relationship defined is the Node's encapTopology (which is effectively the same as the IETF's TE node underlay topology relationship described above). Otherwise, all the provided topologies are independent. It is not clear for the client what is the purpose of each of them, what is the provider's preference as to how and in which order they are supposed to be used, and why several same layer topologies, rather than one, were provided to the client in the first place.

2.7. Topology Attributes

Compared to the IETF TE Topology model, T-API Topology nodes and links are missing some important attributes. Specifically, T-API nodes, as mentioned in section 1.1, have no analogs to the Connectivity matrix attribute and the TE TTP container describing nodes switching and termination capabilities/limitations respectively. Furthermore, the T-API Topology service does not have a concept of TTP, which in the context of the IETF TE Topology model conveys to the client various important edge characteristics for a TE tunnel that could be provided by the network described by a given TE topology. Such characteristics include:

- Potential TE tunnel protection capabilities (e.g., whether 1+1 protection could or could not be supported for the tunnel edge);

- Adaptation capacities (i.e., which higher layer network payload types and from which higher layer link termination points can be adopted on the TE tunnel edge, the amount of adaptation bandwidth still available, etc.);

- Technology-specific TTPs describe technology specific properties (e.g. TTP representing an OCh layer transponder can announce whether the transponder's receiver/transmitter is fixed or tunable, and in the latter case what is the range and resolution of the tunability; supported FECs and signal modulation modes, transmit/acceptable optical signal power levels and OSNRs, etc.)

The T-API Topology link is missing the following attributes:

- Administrative groups (administrative colors) - an attribute describing the link's association with pre-defined groups of links; such groups could be used as constraints in the client's path selection/optimization algorithms to mandate/disallow or encourage/discourage the resulting paths to follow/avoid links related to the specified groups;

- Link protection/restoration capability - an attribute that could be also used as a path computation constraint or path optimization criterion, for example, to force or encourage the resulting paths to follow sufficiently protected links;

- Link properties defining whether the link is:

  A. actual (with committed network resources) or potential;

  B. static (with pre-established and always-in-place server layer connectivity supporting the link) or dynamic (for which the

connectivity is dynamically put in place if/when the link is used by at least one client connection and is dynamically released as soon as the link is used by none of the client's connections);

- Link's underlay primary and backup paths and ID of the topology used for their computations.

2.8. Topology Service Relationships with Other Services

IETF TE topology and TE tunnel models are related. For example, a TE link can point via the Supporting Tunnel ID attribute to the lower layer network TE tunnel providing the transport connectivity for the TE link. Likewise, a TE tunnel has an attribute pointing to the TE link it supports, as well as the TE topology which the TE link is part of. These cross-references are instrumental for the client in terms of understanding which network resources a given TE link represents, especially useful at the times of trouble shooting. Additionally, IETF TE tunnel defines and supports the concept of Hierarchical TE links and tunnels. Hierarchical TE tunnels automatically insert dynamic hierarchical TE links into the specified TE topologies as soon as the tunnels are successfully set up (and remove the hierarchical TE links from the respective TE topologies when released). [Note: Hierarchical TE tunnels and links are instrumental in multi-layer traffic engineering].

Furthermore, both TE topology and TE tunnel models are tightly coupled with the IETF YANG based notification machinery, which allows the client to retrieve any telemetry or attribute change updates as long as those telemetry/attribute changes are defined as data state nodes or sub-trees in the respective models.

In contrast, all T-API services (i.e. Topology, Connectivity, Path computation, Virtual Network and Notification) are independent from each other.

2.9. Topology Negotiation and (Re-)configuration

When a client of the IETF TE Topology model/interface receives one or more abstract TE topologies from one of its providers, it may accept the topologies as-is and merge then into one or more of its own native TE topologies. Alternatively, the client may choose to request a re-configuration of one, some or all abstract TE topologies provided by the providers. Specifically, with respect to a given abstract TE topology, some of its TE nodes/links may be requested to be removed, while additional ones may be requested to be added. It is also possible that existing TE nodes/links may be asked to be re-configured (e.g., TE links may be requested to be SRLG disjoint).

Furthermore, the topology-wide optimization criteria may be requested to be changed. For example, underlay TE paths supporting the abstract TE links, currently optimized to be shortest (least-cost) paths, may be requested to be re-optimized based on the minimal delay criteria. Additionally, the client may request the providers to configure entirely new abstract TE topologies and/or to remove existing ones. Furthermore, future periodic or one-time additions, removals and/or re-configurations of abstract TE topologies, topological elements and/or their attributes could be (re-)scheduled by the client ahead of time.

It is the responsibility of the client to implement the logic behind the above-described abstract TE topology negotiation. It is expected that the logic is influenced by the client's local configuration/ templates, policies conveyed by the client's clients, input from the network planning process, telemetry processor, analytics systems and/ or direct human operator commands. Figure 6 exemplifies the abstract TE topology negotiation process. As shown in the Figure, the original abstract TE topology exposed by a provider was requested to be re-configured. Specifically, one of the abstract TE links was asked to be removed, while three new ones were asked to be added to the abstract TE topology.

The ONF T-API Topology service client has no say as to how the abstract topologies exposed to the client by its providers should look like. The only option for the client is to consume the provided topologies as offered. This is a serious disadvantage because it is the client (not providers) that knows which topologies suite best the client's needs.

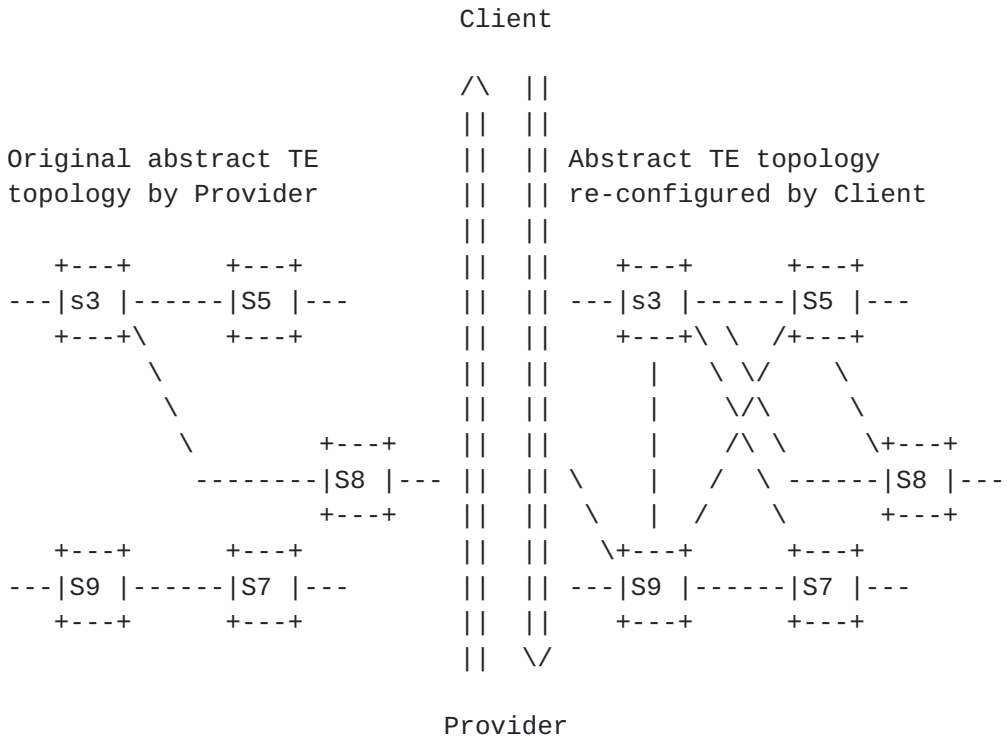

Figure 6: Provider-Client abstract TE topology negotiation

2.10. Integration with IP/MPLS

The IETF TE Topology model is naturally and intimately integrated with IP/MPLS layer models defined for IP/MPLS layer traffic engineering. For example, currently Segment Routing (SR) and Service Function Chaining (SFC) technologies heavily rely on and actively use the TE Topology model. Specifically, SR combines the TE topology model with layer 3 (IP reachability) topology model to facilitate path computations that account for either or both TE and IP reachability information. Likewise, SFC makes use of the TE topology model for computing service function chains optimized according to the combined criteria of real/virtual network function location and best available (possibly in different layers) TE paths to connect the network functions.

It is not clear how the ONF T-API Topology service can fit in and to what extent it can be integrated into the IP/MPLS layer traffic engineering.

3. Connectivity Service

3.1. Connectivity Service Protection

It is not possible for a T-API Connectivity service client to request from a provider a protected service like, for example, the one presented in Figure 7. In the Figure a connectivity service is supported by two disjoint connections - primary (solid blue) and backup (broken yellow), with the client traffic normally carried over the primary connection, but which could be quickly and dynamically switched onto the backup connection as soon as a network failure affecting the primary connection is detected.

The inability to request protected connectivity services from a provider leaves the T-API Connectivity service client with the problem of protecting its own traffic against the network's failures. Admittedly, the client can address this with the following sequence of operations:

1. The client requests a primary connectivity service connecting the desired pair of client device ports over the network managed by the T-API Connectivity service provider;

2. The client requests a secondary connectivity service connecting the same pair of client device ports, which is sufficiently diverse from the primary service (incidentally, this could be problematic due to the independent nature of the path computations carried out by the provider. Specifically, the path selected for the primary service may block disjoint paths for the secondary service. This is a known issue related to sequential/ independent path computations, which could be solved via concurrent path computation for both services);

3. The client binds at both ends the two connectivity services in accordance with the desired protection scheme;

4. From then on the client is constantly monitoring the performance and health of both services;

5. In case the primary service is affected by a network failure (while the secondary service remaining healthy), the client coordinates the protection switchover;

6. In case it is detected that the previously broken primary connectivity service is repaired, the client coordinates the protection reversion (i.e. reversion to the normal forwarding of the client traffic).

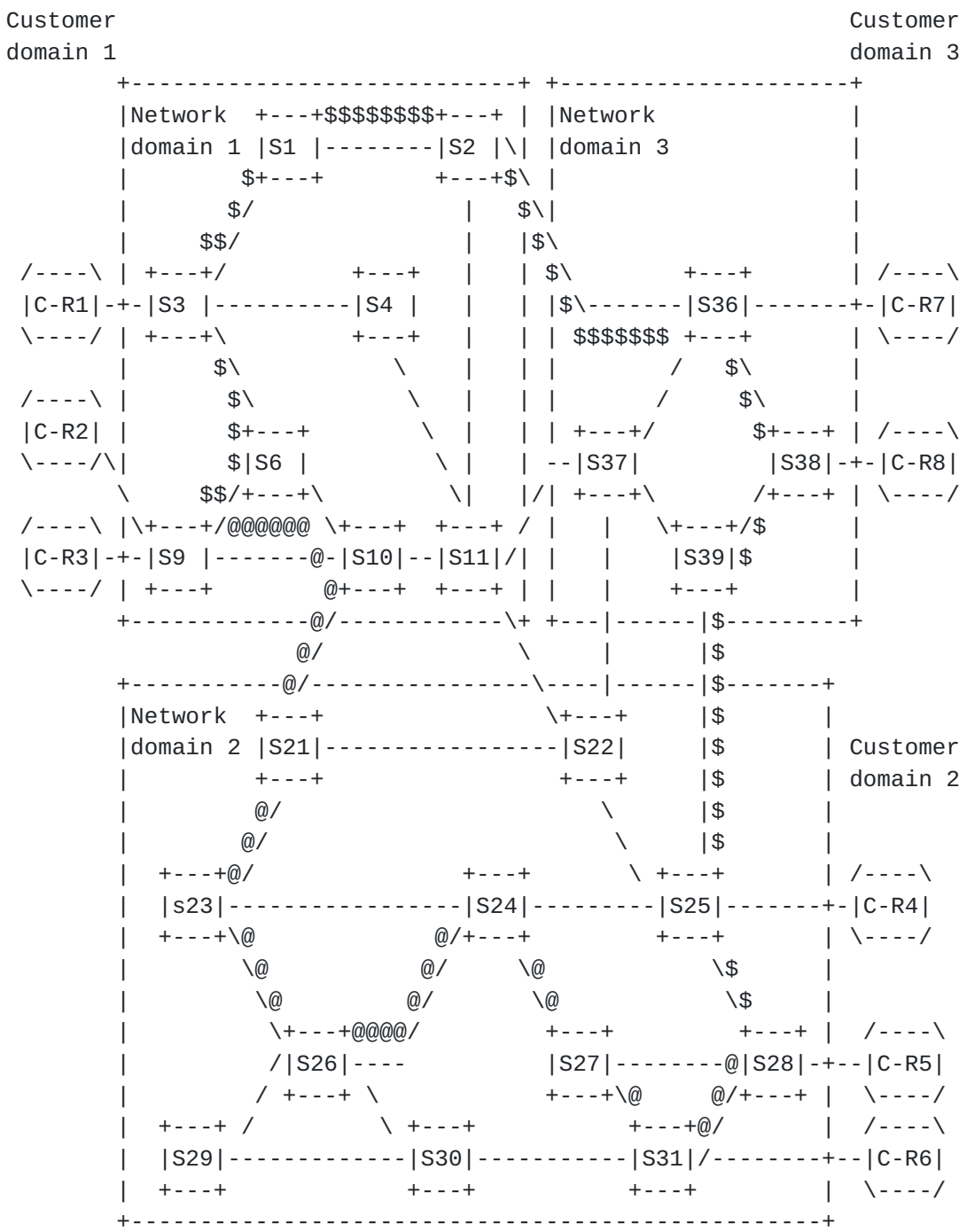

Figure 7: Protected connectivity service

In contrast, an IETF TE tunnel model client normally delegates all the described above operations to the provider by simply configuring the requested transport service (i.e. TE tunnel or a single-domain segment of a multi-domain TE tunnel) to be protected. In doing so the client specifies the required protection type, as well as the level of primary/backup connections disjointedness. Additionally,

the client may specify a set of constraints common for both connections, as well as constraints (e.g. inclusions, exclusions, etc) specific to each connection. Furthermore, the client may even specify, for a given transport service, multiple sets of such constraints in descending preference order for the provider to try before notifying the client about the setup failure. For example, the client may request in this way for a TE tunnel that the primary and backup connections must be SRLG disjoint, and, if this proves to be not possible, to relax the disjointedness criterion to link-disjoint.

3.2. Hierarchical Connectivity Service

A transport network provider may control a multi-layer (e.g. Ethernet/ODUk/OCh) network. On such a network the provider has flexibility to dynamically set up connectivity/transport services in one or more lower layer networks to augment a higher layer topology that is otherwise insufficient for provisioning of a connectivity service requested by the client.

In the top-to-bottom approach the client simply requests a connectivity service in the desired layer network. While processing the request, the provider:

- performs its internal multi-layer path computation,
- identifies one or more lower layer connectivity services required for the successful provisioning of the requested service;
- dynamically (and unknowingly to the client) sets up the so-identified lower layer connections;
- sets up the connection(s) supporting the connectivity service requested by the client.

Both T-API Connectivity service and IETF TE Topology model/interface support the described top-to-bottom multi-layer connectivity services. The approach is simple for the client; however it does not work in many multi-domain use cases. Consider, for example, a multi-domain transport network presented in Figure 8. Consider further that a Multi-Domain Service Coordinator is requested to set up Ethernet layer connectivity service (marked in blue) across three domains, each of which is controlled by a separate provider. Assume also that in order to satisfy the request an underlay ODUk layer TE tunnel (marked as red) also spanning multiple domains needs to be provisioned. This could be achieved via a bottom-to-top multi-layer connectivity service provisioning approach, which includes the following:

- the client (i.e. the Multi-Domain Coordinator) performs its own multi-layer path computation on a network wide TE topology (a product of merging the TE topologies exposed by all providers);

- the client identifies one or more lower layer TE tunnels required for the successful provisioning of the requested service;

- the client coordinates the multi-domain setup of each of the identified lower layer TE tunnels;

- the client instructs each lower layer TE tunnel's first and last domain provider to add a dynamic TE link in their respective higher layer TE topologies;

- the client triggers and coordinates the setup of the connection(s) supporting the requested connectivity service, constraining the connection path(s) to follow the dynamic TE links supported by the lower layer TE tunnels;

- the client adds into its own (network-wide) TE topology, dynamic TE links supported by the lower layer TE tunnels to make the remaining capacity on the tunnels available for path computations for other higher layer connectivity services.

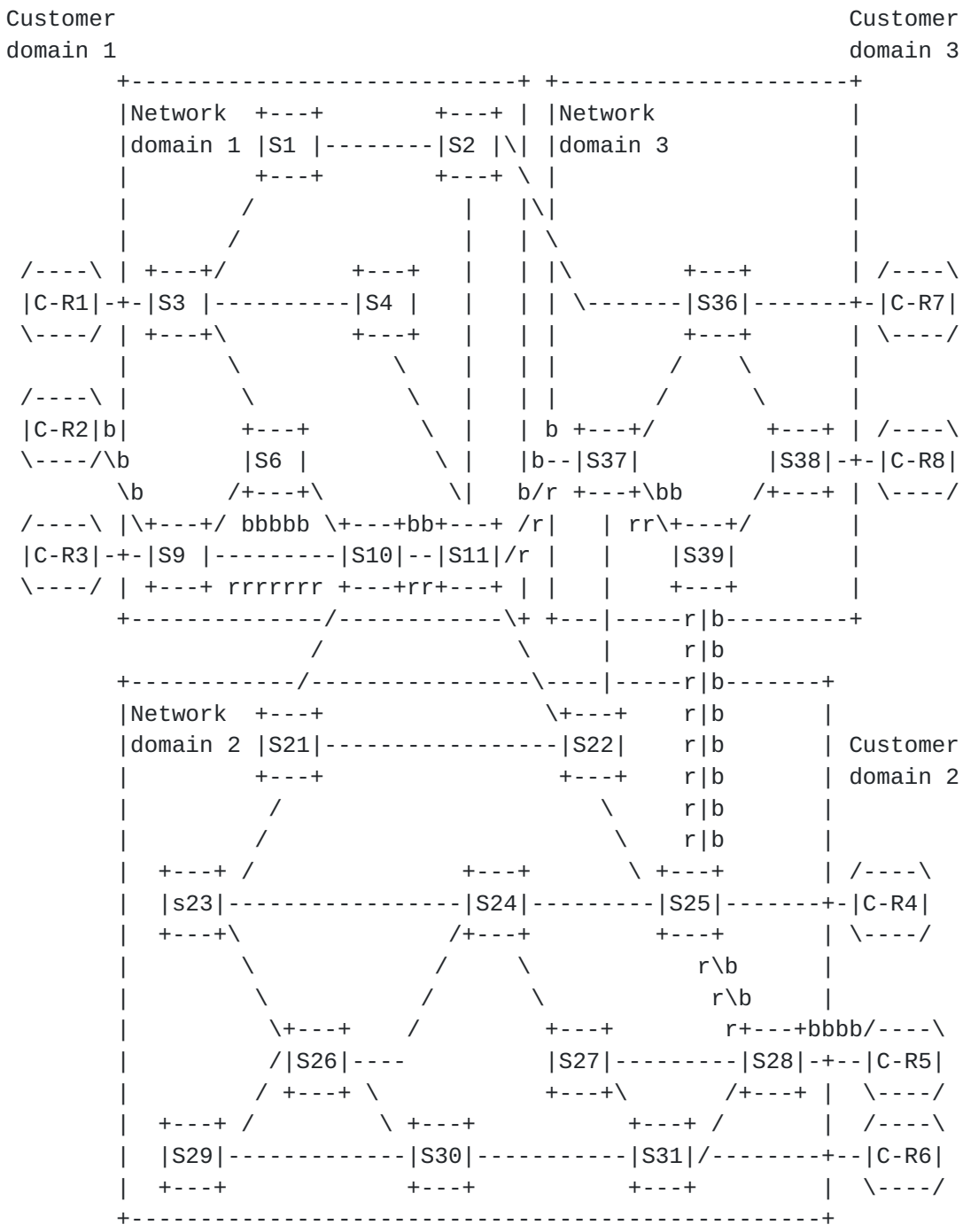

Figure 8: Hierarchical connectivity service

The IETF TE topology model supports the described bottom-to-top multi-layer connectivity service provisioning paradigm via Hierarchy TE tunnels. A hierarchy TE tunnel, once successfully set up, automatically adds into the specified TE topology a TE link it supports and withdraws the TE link from the TE topology if/when released.

T-API Connectivity and Topology services do not support the concept of hierarchical connectivity/dynamic links.

3.3. Connectivity Service Re-optimization

An IETF TE tunnel model/interface client, when requesting a transport service from a provider, can control - via a designed for this purpose knob (lockDown attribute) - whether the connection(s) supporting the service must be "pinned" to their respective original paths (the paths selected at the setup stage), or whether the provider may occasionally perform a service re-optimization, resulting in service connection replacement toward more optimal paths. This knob is especially useful in conjunction with a connectivity scheduling service (see section 2.6), allowing for the client to specify time intervals at which the re-optimization of a given transport service (and subsequent potential traffic hits) is acceptable for the client. For example, the client may configure a transport service to get "unpinned" every Saturday at 1 am for service re-optimization procedures and to get "re-pinned" after that for another week.

T-API Connectivity service clients have no way of controlling of connectivity service re-optimization operations.

3.4. Connectivity Service Templates

The IETF TE tunnel model defines containers of named transport service configuration sets that could be shared by multiple services. This not only simplifies for the client the process of transport service configuration, but also allows manipulation of multiple services by a single configuration change. For example, a client may define a set of constraints named Foo that forces a transport service primary path to go through a node X. If, later, the client modifies Foo by substituting node X with node Y, all transport services configured with the constraint set Foo will (be attempted to) be re-placed onto path(s) going through node Y.

The T-API Connectivity service model does not have a similar concept.

3.5. Connectivity Service Attribute Change Update Notifications and Telemetry Streaming

Both T-API and IETF modeling rely on respective notification tools universal across all interfaces. Therefore, connectivity service attribute change notifications and telemetry streaming is no different from the topology notifications and telemetry streaming discussed in sections 2.3 and 2.4

3.6. Connectivity Scheduling

T-API Connectivity service has the _schedule attribute that includes just two parameters: startTime and endTime. This allows for a client to schedule at a specified time and for a specified period of time a one-time kickoff of a service configured initially (presumably) as disabled. It is not possible to schedule multi-time (periodic) kickoffs. Furthermore, the scheduling granularity is connectivity service as a whole. In particular, it is not possible to schedule re-configurations of one or several service parameters (e.g. bandwidth requirement, inclusion/exclusion path, etc.).

There is an ongoing effort in IETF to produce a generic scheduling tool that could be applied to any of YANG models. Similar to the notification subscription tool - allowing for the client to subscribe on notifications with respect to any data state (CONFIG=FALSE) node defined in any supported by the provider data store - the scheduling tool will allow for the client to schedule periodic and/or one-time modification of any configuration (CONFIG=TRUE) leaf of any supported data store. For example, if it is required to schedule a re-configuration of the bandwidth requirement for one or more selected services, the client will specify an XPath pointing to the configured bandwidth attribute of the services of interest and convey the new bandwidth requirement and the timetable for the service bandwidth re-configuration. [Note: At time intervals outside of the scheduled range, the service configured bandwidth will remain/be restored to the value provided during initial service configuration.]

3.7. Potential Connectivity Service

The IETF TE topology model defines a number of "unconventional" configuration modes to be specified by a client and supported by a provider of transport services. One of those modes is the COMPUTE_ONLY mode. When a provider processes a request for a transport service configured in the COMPUTE_ONLY mode, it performs the normal path computation for the service, but does not trigger setup of the connection(s) supporting the service. Instead, the computed paths are returned to the client as a part of normal service attribute change notification. Furthermore, when the provider detects a change in the managed network potentially affecting the returned paths, it may re-evaluate the paths and notify the client if they have become infeasible or more optimal paths are available.

The concept of COMPUTE_ONLY transport services makes a good foundation for Path computation service/interface between the Client and the Provider (see more in section 4).

4. Path Computation Service

A client of a transport network can discover the network resources available for the client in one of the two ways:

- by requesting from the network provider , via a topology interface, one or more topologies describing the network with respect to its availability to the client;

or

- by requesting, via a path computation interface, that the provider identify potential paths that could connect various client device ports across the network.

To support the latter option, ONF T-API has introduced a Path computation service dedicated to the purpose. A T-API Path computation service client can issue a path computation request specifying the identities of the required path source and destination end points, the layer network in which the paths are to be determined, the required mutual diversity of the resulting paths, various path computation constraints (e.g., bandwidth requirements, inclusions, exclusions, etc.) and path selection optimization criteria (e.g., smallest cost, shortest delay, etc.). A T-API Path computation service provider is expected to satisfy the request by running a path computation algorithm and responding to the client with zero, one or more resulting paths.

In contrast, IETF modeling does not offer a dedicated mechanism/model to support the Client<=>Provider path computation interface. Instead, it is suggested to use the YANG TE tunnel model and request and manipulate path computations in the form of COMPUTE_ONLY TE tunnels as described in section 2.7. This approach has some important advantages as compared to the T-API Path computation service:

Simplicity: provided that both the client and the provider know how to request, manipulate and support transport services, there is no additional interface/model for the client to learn how to use and functionality for the provider to support;

Accuracy: T-API Path computation and Connectivity services are not related. It cannot be guaranteed that the set of path computation constraints conveyed by a T-API Path computation service client will match the set of path computation constraints internally generated by a T-API Connectivity service provider even when the configuration parameters - source/destination, layer network,

bandwidth and others - match. There are many reasons for that, including:

A. additional constraints could be imposed by the provider based on some internal and possibly proprietary knowledge about the network (unknown to the client);

B. various internal policies could relax, harden or overwrite other constraints;

C. various internal policies could modify or overwrite the requested optimization criteria;

D. etc.

Furthermore, the provider may even use different path computation engines to provide the Path computation and connectivity services. All this may result in the paths returned to the Path computation service client being different from the paths taken by the corresponding (same source/destination and other constraints) connectivity services. The difference may be in path costs, delay and fate sharing characteristics, etc. In extreme cases the Path computation service client may even receive unprovisionable and hence useless paths.

IETF COMPUTE_ONLY TE tunnels, on the other hand, do not have such problems. It is inherently guaranteed that the client will be notified/updated with paths which are exactly the same as the ones that would be taken by connections of "conventional" TE tunnels for the same configuration inputs;

Path staleness: paths returned to the T-API Path computation service client may become unfeasible at some later time because of changes in the network's state. There is no way for the Path computation service provider to convey this fact to the client. In contrast, IETF COMPUTE_ONLY TE tunnel provider can use the intrinsic attribute change notifications to let the client know that previously provided paths have changed, have become unfeasible or that better, more optimal paths have become available.

5. Virtual Network Service

A client of a transport network may want to limit the transport network connectivity of a particular type and quality to defined subsets of its device ports interconnected across the network. Furthermore, a given transport network may serve more than one client. In this case some or all clients may want to ensure the availability of transport network resources in case dynamic

(re-)connection of their device ports across the network is envisioned. In all such cases a client may want to set up one or more Virtual Networks over the provided transport network.

ONF T-API has introduced a dedicated service for this purpose - the Virtual Network service (VNS). A VNS client can request creation of a VNS specifying the layer network of the VNS and the Traffic Matrix requirement. The client has no control over the requested VN beyond that. In particular, it is up to the provider to decide which network resources will support the VN in question. The client has no say as to how the underlying network topology should look, how the topology needs to be optimized for the VN (e.g. shortest delay rather than smallest cost), what is the required level of the topology link protection and mutual diversity, and so forth.

As in case of the path computation interface, IETF modeling does not offer a separate model to support VNS. Instead, it encourages using the TE topology model - leveraging the IETF abstract TE topology's ability to be configured by the client. In a nutshell, the client configures and manipulates a VN as a customized abstract TE topology based on the TE topologies already exposed by the provider. In the simplest case the client requests a single node ("black box") abstract TE topology with desired attributes. In more complex cases the client may opt to construct, for the VN, a separate multi-node/ link arbitrary abstract TE topology. In doing so, the client may "borrow" into the VN's topology TE nodes and links from other topologies. Additionally the client may add new composite abstract TE nodes specifying the IDs of TE topologies the nodes will encapsulate, connected by abstract TE links pointing to the respective underlay TE topologies to be used for computation and provisioning of the TE tunnels supporting them. The client/provider negotiation of a"so-cooked" TE topology is described in 1.9. In short, the client is able to manipulate the VN's topology at the granularity of individual topological elements (such as TE nodes and links).

6. Data Modeling Language

Today YANG is a very popular data modeling language. It is a product of IETF NETMOD WG. It is not the only data modeling language produced by IETF (for example, FORCES WG has developed one of its own, arguably - in some aspects - superior to YANG). YANG is neither stable nor perfect. It is constantly evolving with the sole objective to make IETF models more scalable, efficient, inclusive, information-rich: better in all aspects. Supporting non-IETF (e.g. ONF) data models is not a priority. Therefore It is not clear why ONF, while investing a lot of effort in designing Core Information Models, is devoting no effort to designing a data modeling language

of its own that would closely suit support of its CIM. Nor it is clear what would happen if the IETF NETMOD WG decides, for whatever reason to obsolete some of the YANG features/properties/capabilities that ONF models rely upon.

Furthermore, writing CIMs in UML and having them mechanically translated into YANG has its own issues, which includes the following:

- Many useful YANG features that do not have analogs in UML are not used. For example, T-API YANG models use only non-extendible enumeration type, rather than extendible identity type. This prevents T-API YANG models from being easily extendible via augmentation;

- T-API YANG models heavily overuse and often misuse YANG RPCs for operations that could be handled simpler and more efficiently by NETCONF/RESTCONF protocol via native edit-config and get operations;

- T-API YANG models unnecessarily define their own notification subscription/streaming and scheduling mechanisms, instead of leveraging the NETCONF/RESTCONF machinery easily applicable to all YANG models;

- T-API YANG models make no use of YANG templates and defaults designed to simplify for the client the provider's data store (re-)configuration;

- T-API YANG models follow the conventions inherited from UML and previously defined REST APIs. As a consequence. the models sometimes are not compatible with the current best practices recommended for YANG model writers and do not always follow YANG model guidelines defined in [I-D.ietf-netmod-rfc6087bis]

7. Security Framework

ONF T-API does not have a security framework of its own. It simply assumes that the proper security could be inherently provided by the underlying protocols. IETF TEAS interfaces, on the other hand, take the security considerations very seriously. They rely on the generic framework ([RFC6241], [RFC8040], [RFC6536], and [I-D.ietf-netconf-rfc6536bis]) allowing for the provider to configure in a universal way various strength AAA protection for any YANG modeled data store accessible via NETCONF or RESTCONF protocol. In particular, said framework allows for the client authentication, identification of the client's privileges with respect to the

information access, required filtering and scoping of the provided information, as well as secure client-provider communication.

8. IANA Considerations

This document has no actions for IANA.

9. Security Considerations

This document does not define networking protocols and data, hence are not directly responsible for security risks.

This document compares two interface technologies of T-SDN controllers. For each specific technology discussed in the document, security framework has been described and compared in the corresponding section.

10. Acknowledgements

The authors would like to thank Christopher Jenz, Diego Caviglia, Aihua Guo, Fatai Zhang, and Italo Busi for their helpful comments and valuable contributions.

11. References

11.1. Normative 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>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo-12 (work in progress), July 2017.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-08 (work in progress), July 2017.

[I-D.ietf-netconf-rfc6536bis]
Bierman, A. and M. Bjorklund, "Network Configuration Access Control Module", draft-ietf-netconf-rfc6536bis-08 (work in progress), October 2017.

11.2. Informative References

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-14 (work in progress), September 2017.

Authors' Addresses

Igor Bryskin
Huawei Technologies

EMail: Igor.Bryskin@huawei.com

Xufeng Liu
Jabil

EMail: Xufeng_Liu@jabil.com

Vishnu Pavan Beeram
Juniper Networks

EMail: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc

EMail: tsaad@cisco.com