

MUST

IP SDN Controller SBI Technical Requirements

OOPT



Authors

Oscar Gonzalez de Dios

IP & Transport Networks Technology and planning Expert, Telefonica.

oscar.gonzalezdedios@telefonica.com

Ndifor Luc-Fabrice Ngwa

Senior Engineer, Fixed Networks and Technology Management, MTN.

luc-fabrice.ndifor@mtn.com

Luis Ángel Muñoz

IP and SDN Network Architect, Vodafone.

luis-angel.munoz@vodafone.com

María Jesús Vázquez

IP and SDN Network Architect, Vodafone.

maria.vazquez@vodafone.com

Lloyd Mphahlele

General Manager, Transport, Transport, MTN.

Lloyd.Mphahlele@mtn.com

Philippe Niger

Network architect at Orange Labs (the R&D branch of Orange)

philippe.niger@orange.com

Samier Barguil

IP SDN Technical Leader, Telefonica.

samier.barguilgiraldo.ext@telefonica.com

Carlos Rodriguez

IP/MPLS Senior Network Engineer, Telefonica

carlos.rodriuezcampos.ext@telefonica.com

Fritz-Joachim Westphal

Senior Project Manager, DT

Fritz-Joachim.Westphal@telekom.de



Version Control

Date	Revision	Author(s)	Comment
23/02/2021	V1.0	All Authors	

Table of Contents

Scope	1
Authors	2
Contributors	3
Version Control	4
Table of Contents	5
List of Figures	7
List of Tables	7
1. Scope	11
1.1 Use Cases	11
2 SBI Communication Protocols	15
2.1 NETCONF	15
2.2 BGP-LS	17
2.3 PCEP	17
3 Openconfig Data Models	21
4 Service Provisioning Use cases (category 1)	28
4.1 L3VPN Structure and Classification	28
4.2 Workflow for L3VPN Creation	29
4.2.1 Create VPN (VRF instance definition)	30
4.2.2 Add VPN import/export BGP policies (VRF route target)	31
4.2.3 Configure Interfaces to access to VPN	34
4.2.4 Bind interfaces/subinterfaces to VRF (VPN End Points)	38
4.2.5 Redistribute routing protocols on client connection (CPE) inside VRF	39
4.2.6 Append additional configuration for QoS	43
4.3 Use case 1.1 L3VPN for 3G/4G Services. [L3VPN/Dot1Q/None/None]	49
5 Inventory Support Use Cases (category 2)	58
5.1 Structure and Classification	58
5.2 Use case 2.1: Retrieve Logical Interfaces Inventory	58

5.3	Network Inventory Atomic Operations	59	
6	Topology and Discovery Use Cases (category 3)	61	
6.1	Structure and Classification	61	
6.2	Use case 3.1 Obtain and export of end to end ip topology using IP domain controllers (IGP Topology)	62	
6.3	Use case 3.2 L2 Topology		65
6.4	Use case 3.3 UNI-Topology		69
7	Traffic Engineering Use Cases (category 4)	72	
7.1	TE and PCE for SBI	72	
7.2	Structure and Classification	76	
7.3	Openconfig Model for TE	77	
7.4	TE Uses Cases	79	
7.4.1	Use case 4.1 LSP Creation, modify and delete with RSVP-TE		79
7.4.2	Use case 4.2 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) – SIGNALLING: SR		83
7.4.3	Use case 4.3 - LSP create, modify and delete with constraints (delay, bandwidth and hop count)		84
7.4.4	Use case 4.4 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) and explicit Path (strict and loose)		89
7.4.5	Use case 4.5 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) – PROTECTION: Redundancy 1+1		93
8	References	99	
9	Glossary	100	
10	TIP Document License	102	
	Disclaimers	103	



List of Figures

Figure 1. Uses Cases Categories	12
Figure 2. Openconfig Modules Example	24
Figure 3. Openconfig module relationship	25
Figure 4. Workflow for L3VPN creation using Openconfig yang model + Netconf in the SBI	30
Figure 5. Generic network topology for L3VPN 3G/4G service	50
Figure 6. Configuration process flow for L3VPN 3G/4G service	51
Figure 7. Openconfig LLDP Parameters	67
Figure 8. L2/L3 topology collection architecture	69
Figure 9. PCEP Architecture	73
Figure 10. PCE function in the E2E IP SDN Domain Controller	73
Figure 11. PCE function in the E2E IP MUST SDN Domain Controller	75
Figure 12. PCC-initiated LSP workflow	76
Figure 13. PCE-initiated LSP workflow	76
Figure 14. OC Modules for MPLS	78
Figure 15. LSP network diagram with 1+1 redundancy	94
Figure 2: Secpnd Example	108

List of Tables

Table 1 Use Cases covered in MUST IP SBI D1.1	6
Table 2 Opeconfig parameters for step "Add VPN import/export BGP policies (VRF route target) "	33
Table 3 Opeconfig parameters for step "Configure Interfaces to access to VPN"	38
Table 4 Opeconfig parameters for step "Bind interfaces/subinterfaces to VRF (VPN End Points)"	39
Table 5 Opeconfig parameters for step "Redistribute routing protocols on client connection (CPE) inside VRF"	43
Table 6 Opeconfig parameters for step "Append additional configuration for QoS"	49
Table 7 Definition of Use Case 1.1	49
Table 8 Use case 1.1 VPN scenario	49
Table 9 Example of parameters for VRF Instance creation	52
Table 10 Atomic Operations for Use Case 1.1 (1)	53
Table 11 Atomic Operations for Use Case 1.1 (2)	54
Table 12 Atomic Operations for Use Case 1.1 (3)	55
Table 13 Atomic Operations for Use Case 1.1 (4)	56
Table 14 Atomic Operations for Use Case 1.1 (5)	57
Table 15 Inventory Use Cases covered in D1.1	58
Table 16 Definition of Use Case 2.1	58
Table 17 Network Inventory Atomic Operations	59
Table 18 Definition of Use Case 3.1	62
Table 19 Definition of Use Case 3.2	65
Table 20 LLDP TLVs for Use Case 6.2	68
Table 21 Atomic Operations for Topology Use case 6.3.2	69
Table 22 Definition of Use Case 3.3	70
Table 23 Atomic Operations for Use Case 6.3	70
Table 24 Definition of Use Case 4.1	79
Table 25 Atomic Operations for Traffic Engineering (1)	83
Table 26 Definition of Use Case 4.2	83
Table 27 Definition of Use Case 4.3	84

Table 28 Atomic Operations for Traffic Engineering (2)	89
Table 29 Definition of Use Case 4.4	89
Table 30 Atomic Operations for Traffic Engineering (3)	93
Table 31 Definition of Use Case 4.5	93
Table 32 Atomic Operations for Traffic Engineering (4)	98

1

Scope

Implementing a complete standard specification is a time-consuming process. To accelerate the adoption of the specifications, the TIP MUST OOPT subgroup has **compiled the needs of multiple operators**, selecting the **most relevant common use cases ...**





1 Scope

Implementing a complete standard specification is a time-consuming process. To accelerate the adoption of the specifications, the TIP MUST OOPT subgroup has **compiled the needs of multiple operators**, selecting the **most relevant common use cases** [1]. Based on this information, technical requirements are prepared and shared with the industry in an open manner. The produced specifications will be incorporated as part of each operator processes.

This document provides the specification of the SDN Interfaces mandatory to be exposed by IP/MPLS Network Elements (routers) and consumed South bound of IP SDN Controllers. In this document these set of interfaces is shortened as SBI. The IP/MPLS equipment must support, in addition to the data and control plane protocols:

- **Netconf** with, minimum, the Yang data models defined in this document.
- PCEP to interact with a stateful PCE function with the minimum support defined in the document.
- BGP-LS to export topology and TE with the minimum support defined in the document.
- gRPC (gNMI) to stream telemetry.

1.1 Use Cases

MUST follows an **incremental approach based on use cases**. For each use case, the needs in terms of interfaces is described in order to facilitate the implementations by Equipment and Controller vendors. The use cases in MUST are classified into several categories, as depicted in the next figure (see MUST Open Transport SDN Architecture Whitepaper[1]):

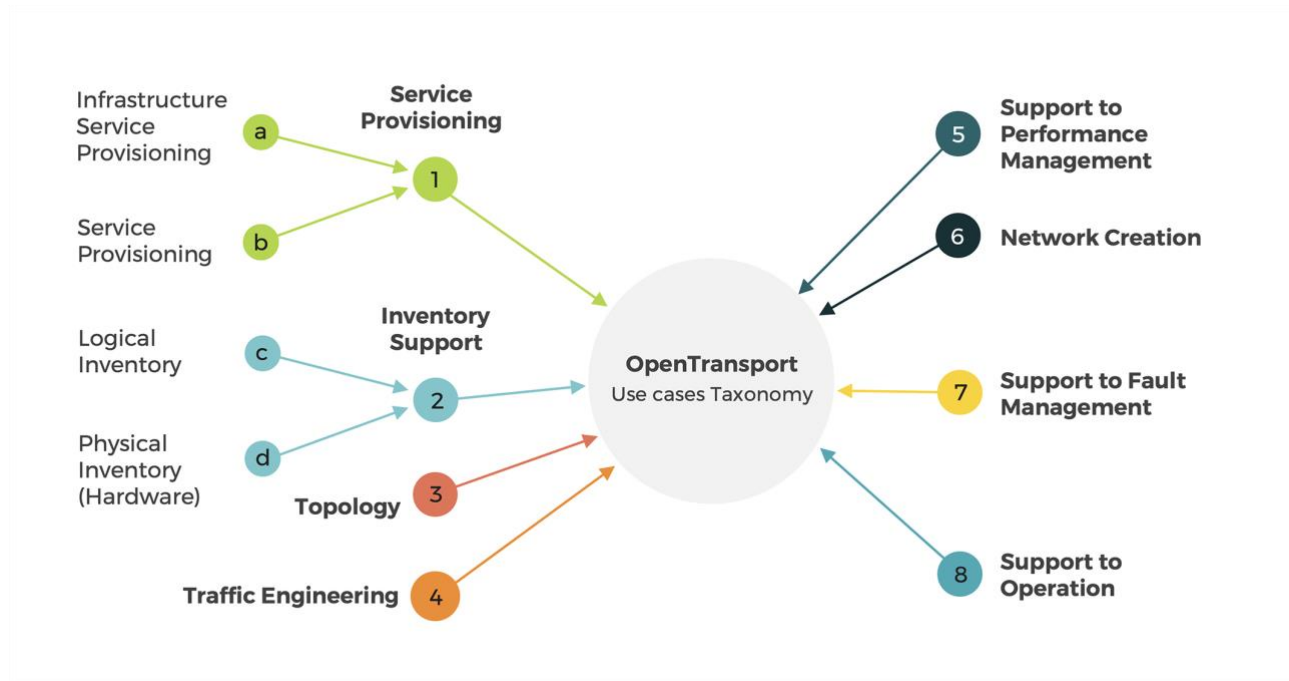


Figure 1. Uses Cases Categories

The purpose of this document is to identify, describe and compile the set of use cases as well as define the requirements, flows and details of each use cases in Network Element (router) and IP Domain controller. MUST Deliverables are incremental in terms of specified use cases.

Within MUST, all member operators have agreed on a first prioritization about the different use cases. The result is the selected use cases included in the following table and covered in D1.1:

ID	Service Provisioning	Section
1.1	L3VPN for 2G/3G/4G Services [L3VPN/Dot1Q/None/None]	4.3
ID	Inventory	Section
2.2	Retrieve logical [Interfaces] Inventory	5.2
ID	Network Topology	Section
3.1	Obtain and export of end to end ip topology using ip domain controllers (IGP Topology)	6.2
3.2	Obtain and export of L2 topology using ip domain controllers (Ethernet links between routers)	6.3
3.3	Export potential service end points in IP topology (UNI Topology)	6.4
ID	Traffic Engineering	Section
4.1	LSP create, modify and delete (no constraints) and RSVP-TE signaling	7.4.1

4.2	LSP create, modify and delete (no constraints) and SR	7.4.2
4.3	LSP create, modify and delete with constraints (delay, bandwidth and hop count) – SIGNALLING: (RSVP & SR)	7.4.3
4.4	LSP create, modify and delete with constraints (delay, bandwidth and hop count) – SIGNALLING: EXPLICIT ROUTE (strict and loose hops) (RSVP & SR)	7.4.4
4.5	LSP create, modify and delete with constraints (delay, bandwidth and hop count) Protection: Redundancy 1+1 (RSVP & SR)	7.4.5

2

SBI Communication Protocols

The Southbound Interface (SBI), is the interface, based on standards, between the SDN Domain Controller and the Network Elements. The MUST project has several objectives ...

2 SBI Communication Protocols

The Southbound Interface (SBI), is the interface, based on standards, between the SDN Domain Controller and the Network Elements. The MUST project has several objectives. One of them includes Southbound standardization. The standardization must include a communication protocol that will depend on the use case. Thus, for the use cases described in this document the standard communication protocols requested are:

- For Service L3 and L2 Service Provisioning: **NETCONF** [2]
- For Inventory support: **NETCONF** [2]
- For Network Topology:
 - **BGP-LS** [3]
 - **NETCONF** to collect **LLDP information**. Note that LLDP will run between the network elements and not with the controller.
- For Traffic Engineering: **PCEP** [4]
- For telemetry and statistics: **gRPC** [5], **NETCONF**[2]

In this section a description of each the protocols and Its requirements is provided:

2.1 NETCONF

The NETCONF protocol defined in RFC 6241 [2] propose a simple mechanism through which a network device can:

- be managed,
- be queried (configuration retrieved)
- be uploaded and manipulated.

NETCONF uses a simple Remote Procedure Call RPC-based mechanism to facilitate communication between a client and a server. The client can be a script, application running as part of a network domain controller. The server is typically a network device.

The protocol allows the device to expose a full, formal application programming interface (API). MUST has selected SSH as the transport protocol and XML as the encoding mechanism, for this purpose. Applications

can use this straightforward API to send and receive full and partial configuration data sets.

The support of each of the use cases described in this document implies the following actions:

```
Creation
Modification
Deletion
```

Accordingly, in order to be supported as the MUST project the network equipment MUST be compliant with the standard RFC 6241 "Network Configuration Protocol (NETCONF)", including the complete set of base protocol operations (NOTE: Apply to all NE to be deployed regardless of the level in hierarchical architecture)

```
get
get-config
edit-config
copy-config
delete-config
lock
unlock
close-session
kill-session
```

Furthermore, NETCONF defines the existence of one or more configuration datastores and allows configuration operations on them. A configuration datastore is defined as the complete set of configuration data that is required to get a device from its initial default state into a desired operational state. The configuration datastore does not include state data or executive commands.

Thus, the network element MUST support the following datastores (in case some of the datastores are not supported, the vendor should provide the alternative workflow to achieve the use cases):

```
Start-Up
Running
Candidate
```

Any equipment MUST advertise its NETCONF capabilities by sending them during an initial capability exchange as defined in RFC 6241. The equipment MUST implement the following NETCONF capabilities:

```
writable-running
candidate configuration
confirmed-commit
```



```
rollback-on-error
```

```
(*) The validate and Xpath capabilities support shall be considered optional but valuable
```

Finally, the equipment MUST provide mechanisms (e.g. RADIUS) to support mutual authentication and authorization with the SDN controller.

2.2 BGP-LS

BGP-LS [3] shall be used to extract the layer 3 topology from the designated node, and also report the Traffic Engineering (TE) information according to RFC8571 [6].

In particular latency and shall deliver Sub TLV present in the IS-IS TE according to RFC8570 (same as RFC7810, or equivalent in OSPF-TE metric extensions RFC7471), as documented in

- Unidirectional average latency (IS-IS/BGP-LS) SubTLV 33 /1114
- MIN/MAX average latency SubTLV 34 /1115
- Delay variation SubTLV 35 /1116
- Unidirectional Residual Bandwidth SubTLV 37 /1118
- Unidirectional Available Bandwidth SubTLV 38 /1119
- Unidirectional Utilized Bandwidth SubTLV 39 /1120

Parameters in use will depend on the specific use case. Value calculation is subject to vendor implementation.

2.3 PCEP

PCEP (Path Computation Element (PCE) Communication Protocol), as described in RFC4657 and RFC5440 will be the protocol used as a way to exchange candidate paths between the PCE and the nodes (also referred to as PCC(Path Computation Client)).

The PCE component of the controller shall be stateful as per RFC8051 definition.

There are two main ways of definition and modification of paths for a Stateful PCE, with different use cases, that will be required in MUST.

- **Delegation:** an operation to grant a PCE temporary rights to modify a subset of LSP parameters on

one or more LSPs of a PCC ("delegated" LSPs). The (unique) PCC that owns the PCE state for the LSP has the right to delegate it. For intra-domain LSPs, this PCC should be the LSP head end.

- **PCE Initiation:** assuming LSP delegation granted by default, a PCE can issue recommendations to the network.

A stateful PCE shall comply with PCEP protocol extensions as described in RFC8231: Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE. In case the Network uses Segment routing (SR) as transport (RFC8402, Segment Routing Architecture), specific PCEP extensions will be needed as well according to RFC8664. PCEP extensions for automatic bandwidth adjustment when employing an Active Stateful PCE for both PCE-Initiated and PCC-Initiated LSPs are described in RFC8733

The way the PCE shall optimise the paths is described at RFC8232 Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE.

A summary of the MUST requirements within this set of recommendations: The PCE, maintains two sets of information DB for use in path computation. Stateful condition requires reliable state synchronization mechanisms between the PCE DB and the network spokes, the PCCs.

1. **Traffic Engineering Database (TED):** topology and resource state in the network. This information can be obtained by a stateful PCE from the network using mechanisms like BGP-LS, or gRPC Telemetry (in this case a parameter mapping to the topology shall be needed)
2. **LSP State Database (LSP-DB),** attributes of all active LSPs in the network, such as their paths through the network, bandwidth/resource usage, switching types, and LSP constraints

On top of stateful, MUST is selecting an **Active Stateful PCE:** a PCE that may issue recommendations to the network. For example, an Active Stateful PCE may use the Delegation mechanism to update LSP parameters in those PCCs that delegate control over their LSPs to the PCE

Note that a Passive Stateful PCE would not actively update LSP state.

RFC 8281 Describes the creation and deletion of PCE-initiated LSPs under the stateful PCE model using additional extensions to PCEP in a Stateful PCE Model (or latest version).

The security implementation of PCEP is critical and shall depend on every operator requirement, but one of



the options under consideration is the use of TLS. RFC8253 PCEPS: Usage of TLS to Provide a Secure Transport for PCEP

The controller shall be able to steer traffic according to policies across candidate paths that can be either MPLS tunnels or SR paths according to (draft-ietf-spring) Segment Routing Policy Architecture. The candidate paths and the policies association can be done in different ways but PCEP is a good option to make it dynamic and active-stateful (draft-ietf-pce-segment-routing-policy-cp-01). LSP paths association extensions in PCEP are described in (RFC 8697).

3

Openconfig Data Models

As mentioned, one pillar of the Southbound Interface (SBI) is the use of NETCONF protocol with Device Models based on Openconfig (OC) ...

3 Openconfig Data Models

As mentioned, one pillar of the Southbound Interface (SBI) is the use of NETCONF protocol with Device Models based on Openconfig (OC) [7] .

OpenConfig is an informal working group of network operators sharing the goal of moving our networks toward a more dynamic, programmable infrastructure by adopting software-defined networking principles such as declarative configuration and model-driven management and operations. Openconfig models are vendor-neutral data models defined in YANG. They cover actual operational needs from use cases and requirements from multiple network operators.

The main modules used in this IP specification are:

- **bgp**: This set of modules describe the BGP protocol configuration. They are used in the service-related use cases to handle the BGP protocol (in CE-PE routing).
- **interfaces**: Model for managing network interfaces and subinterfaces. For the use cases that are currently defined, it is used to configure the interfaces of the VPNs
- **lldp**: This module defines configuration and operational state data for the LLDP protocol. Used for Topology discovery.
- **lACP**: This module describes configuration and operational state data for Link Aggregation Control Protocol (LACP) for managing aggregate interfaces. It works in conjunction with the OpenConfig interfaces and aggregate interfaces models. In this specification, it is used in case of Link Aggregation (LAG) in the CE-PE connection plus LACP protocol.
- **local-routing**: This module describes configuration and operational state data for routes that are locally generated, i.e., not created by dynamic routing protocols. These include static routes, locally created aggregate routes for reducing the number of constituent routes that must be advertised, summary routes for IGPs, etc. In this specification, local-routing is imported in network-instance to configure the static routes.
- **mpls**: This module provides data definitions for configuration of Multiprotocol Label Switching (MPLS) and associated protocols for signaling and traffic engineering. In this specification it is used to create PCC-Initiated LSPs.
- **network-instance**: An OpenConfig description of a network-instance. This may be a Layer 3 forwarding construct such as a virtual routing and forwarding (VRF) instance, or a Layer 2 instance

such as a virtual switch instance (VSI). Mixed Layer 2 and Layer 3 instances are also supported. In this specification this is the main module for the service instantiation. The openconfig-network-instance imports additional models to fulfill the VPN instance requirements:

- Interfaces
 - VLANs
 - Protocols
- **ospf**: An OpenConfig model for Open Shortest Path First (OSPF) version 2. In this specification, it is used in case OSPF is used as PE-CE protocol.
- **isis**: This module describes a YANG model for ISIS protocol configuration. In this specification this model is used to configure commissioning parameters on network creation use cases.
- **platform**: This module defines a data model for representing a system component inventory, which can include hardware or software elements arranged in an arbitrary structure. The primary relationship supported by the model is containment, e.g., components containing subcomponents. The platform module is the base for the Hardware Inventory use case, and it is composed of the following models:
 - Platform - openconfig-platform.yang – It constitutes the main model to define the hardware components of a network device.
 - CPU - It augments the platform model to add specific parameters of a CPU component.
 - FAN - openconfig-platform-fan.yang – It augments the platform model to add specific parameters of a FAN component.
 - LINECARD - It augments the platform model to add specific parameters of a Linecard component.
 - PORT - It augments the platform model to add specific parameters of a port component.
 - PSU - It augments the platform model to add specific parameters of a PSU (Power Supply Unit) component.
- **policy**: This module describes a YANG model for routing policy configuration. It is a limited subset of all of the policy configuration parameters available in the variety of vendor implementations, but supports widely used constructs for managing how routes are imported, exported, and modified across different routing protocols. The routing policy is used in this specification for the distribution of routes within MP-BGP in the VPN related use cases.
- **qos**: This module defines configuration and operational state data related to network quality-of-

service. In this configuration it is used for setting the use case to overwrite the QoS in a VPN setup.

- **rib**: Defines a data model for representing BGP routing table (RIB) contents.
- **segment-routing**: Configuration and operational state parameters relating to the segment routing. This module defines a number of elements which are instantiated in multiple places throughout the OpenConfig collection of models. In this specification it is used to create SR LSPs.
- **types**: This module contains a set of general type definitions that are used across OpenConfig models. It can be imported by modules that make use of these types.
- **vlan**: This module defines configuration and state variables for VLANs.in addition to VLAN parameters associated with interfaces
- **system**: This module describes the managing system-wide services and functions on network devices such as NTP server, DNS, AAA, etc.
- **telemetry**: This module creates the configuration for the telemetry systems and functions on the device, including destinations, protocols, encodings, sensors and subscriptions.
- **acl**: This module defines configuration and operational state data for network access control lists (i.e., filters, rules, etc.).

These modules and the rest of the OpenConfig initiative yang models can be found at <https://github.com/openconfig/public>, under the folder /release/models. Also models.yang files are provided with this document yang.zip.

All of those modules are used as a whole to configure a device depending of every aspect concerned as depicted in the following diagram:

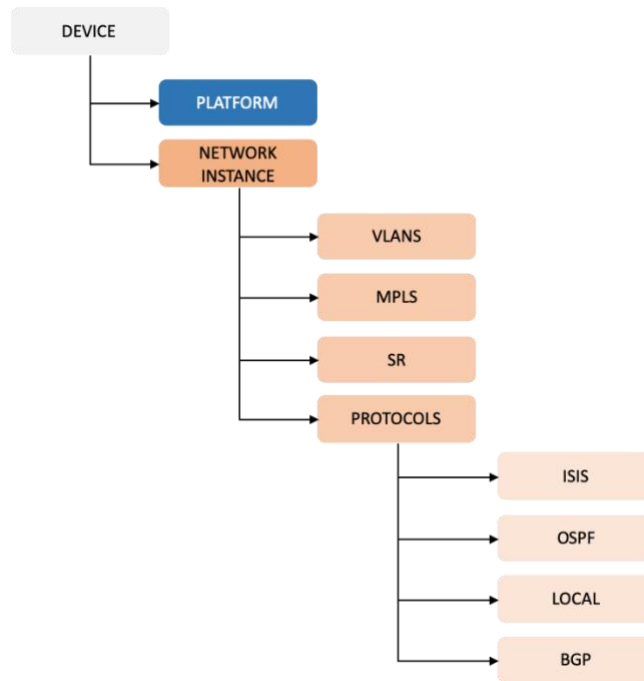


Figure 2. Openconfig Modules Example

Furthermore, the support of certain openconfig modules is highly related with the use case. For example, in the case of a configuration of a L3VPN just certain modules must be supported by the network device. As depicted in this example, the Network instance module has cross-relations with Interfaces and QoS.

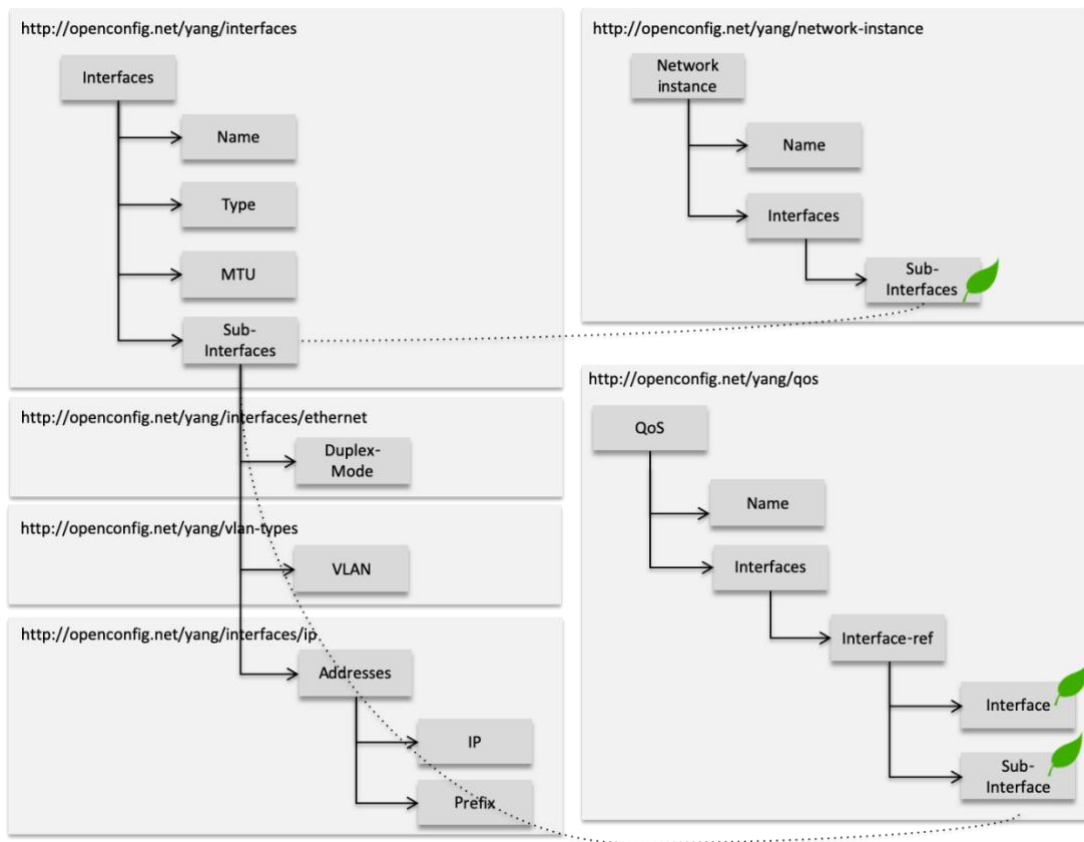


Figure 3. Openconfig module relationship

Openconfig models are currently the preferred options based on the implementation maturity. Nevertheless, the MUST operators will keep monitoring IETF standards and will include them when the level of maturity reaches the same level.

The detailed version of the Openconfig Yang modules which shall be used in this SBI specification and its complementary modules (IETF or OC) implicitly imported within these main modules, along with the complete list of Xpath for those parameters used in each specific use case, can be found on the auxiliary Excel document MUST_IP_SBI_Deliverable_D.1.1_Annex_2_Use_Cases_vs_OC_Model_Params.xlsx [8]

Furthermore, a set of examples of NETCONFIG operations (RPC calls) to be executed so as to implement each use case as described in the corresponding chapter throughout this document is provided on the annex document MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations.docx [9]



4

Service Provisioning Use cases (category 1)

The L3VPNs are widely used to deploy 3G/4G, fixed and enterprise services mostly due to the fact that several traffic discrimination policies can be applied in the network to transport and guarantee the right SLAs to the mobile customers ...



4 Service Provisioning Use cases (category 1)

4.1 L3VPN Structure and Classification

The L3VPNs are widely used to deploy 3G/4G, fixed and enterprise services mostly due to the fact that several traffic discrimination policies can be applied in the network to transport and guarantee the right SLAs to the mobile customers. By definition the L3VPN creates a virtual routing network instance (VRF) in each of the nodes involved in service deployment. This routing instance allows the routing information propagation between the sites involved in the service.

The L3VPN services can be classified using its operational characteristics, thus the following structure have been defined to identify each possible variation of the VPN service. Additionally, it has been included as an identifier of the Use case (**Use case Definition**) to map the relationship between the commercial offer and the real deployment:

- **Use case Definition:** Type of services deployed in the network.
- **Functional Parameters:** Used to find common structures in the configured Services. This classification has the following structure:
 - a. **VPN Service Type:** Type of service configured
 - i) L3VPN
 - b. **End Point Connection Type:** Encapsulation details used in the CE-PE connection.
 - i) None
 - ii) Dot1q
 - iii) QinQ
 - iv) L2VPN (ipipe/epipe/VLL)
 - v) Loopback
 - c. **Routing Protocol used to connect the CE:** Routing details used in the CE-PE connection.
 - i) Direct
 - ii) Static
 - iii) BGP
 - iv) OSPF

- v) ISIS
- vi) RIP
- d. QoS policies applied in the service:
 - i) None
 - ii) QoS Overwriting
 - iii) CIR / PIR policies

4.2 Workflow for L3VPN Creation

In general, the parameters needed to configure devices via SBI (Netconf) through standard Openconfig model will have the following six steps:

1. Create VPN (VRF instance definition)
2. Add VPN import/export BGP policies
3. Configure interfaces and subinterfaces to access L3VPN
4. Bind access interfaces/subinterfaces to VRF (L3VPN End Points)
5. Redistribute routing protocols on client connection (CE) inside VRF
6. Append additional configuration related to:
 - a) QoS Policy
 - b) Service LSP binding for VPN traffic (policy forwarding)
 - c) ...

Thus, the corresponding general workflow, valid for all VPNs, is as follows:

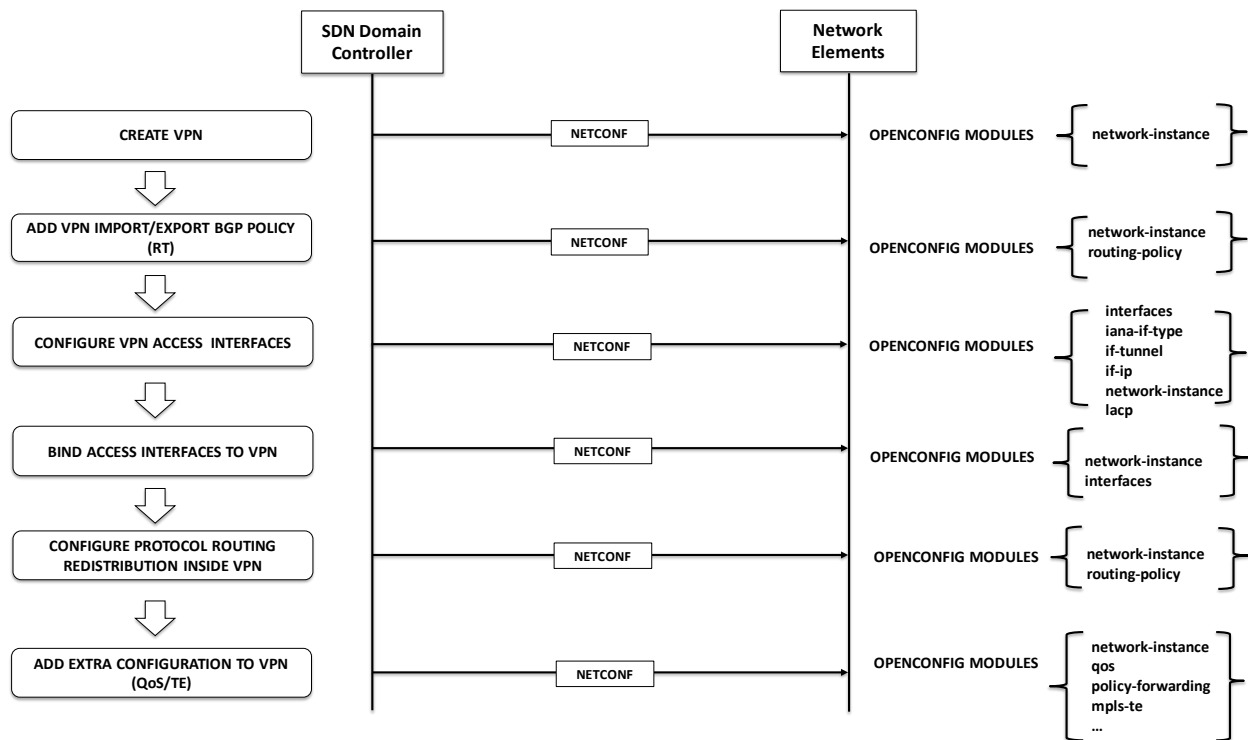


Figure 4. Workflow for L3VPN creation using Openconfig yang model + Netconf in the SBI

In the following subsections, all the parametrization needed for L3VPN will be described in detail.

4.2.1 Create VPN (VRF instance definition)

The set of parameters required for the creation of the new L3VPN service on Netconf SBI are taken from “openconfig-network-instance” module. The information includes a name, description and RD, together with address-family and label-allocation mode for the VPN/VRF. The `xpath` and description of the parameters used are described below:

- 1) `/network-instances/network-instance/config/name` → A unique name identifying the network instance
- 2) `/network-instances/network-instance/config/description` → A free-form string to be used by the network operator to describe the function of this network instance
- 3) `/network-instances/network-instance/config/type` → The type of network instance. The value

of this leaf indicates the type of forwarding entries that should be supported by this network instance. Signaling protocols also use the network instance type to infer the type of service they advertise. Options: *DEFAULT_INSTANCE* (Global Table), *L3VRF*, *L2VSI*, *L2P2P*, *L2L3*

- 4) */network-instances/network-instance/config/enabled* → Whether the network instance should be configured to be active on the network element
- 5) */network-instances/network-instance/config/router-id* → An identifier for the local network instance – typically used within associated routing protocols or signalling routing information in another network instance
- 6) */network-instances/network-instance/config/route-distinguisher* → The route distinguisher that should be used for the local VRF or VSI instance when it is signalled via BGP.
- 7) */network-instances/network-instance/config/enabled-address-families* → The address families that are to be enabled for this network instance. Options: *L2-ETHERNET*, *IPv4*, *IPv6*, *MPLS*
- 8) */network-instances/network-instance/encapsulation/config/encapsulation-type* → The on-the-wire encapsulation that should be used when sending traffic from this network instance. Options: *MPLS*, *VXLAN*
- 9) */network-instances/network-instance/encapsulation/config/label-allocation-mode* → The label allocation mode to be used for L3 entries in the network instance. Options: *PER_PREFIX*, *PER_NEXTHOP*, *INSTANCE_LABEL*

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(1) name	string	-	controller	rw
(2) description	string	-	controller	rw
(3) type	identityref	-	controller	rw
(4) enabled	boolean	-	controller	rw
(5) router-id	string	-	controller	rw
(6) route-distinguisher	union	-	controller	rw
(7) enabled-address-families	identityref	-	controller	rw
(8) encapsulation-type	identityref	-	controller	rw
(9) label-allocation-mode	identityref	-	controller	rw

4.2.2 Add VPN import/export BGP policies (VRF route target)

Furthermore, configuration related to RT and policies to be applied to routes imported and exported to/from VRF are added. As a general rule, first you have to create an export and import policy within

“**openconfig-routing-policy**” module and then use it within “**openconfig-network-instance**” module. The **xpath** and description of the parameters used are described below.

- 1) */network-instances/network-instance/inter-instance-policies/apply-policy/config/import-policy*
→ List of policy names in sequence to be applied on receiving a routing update in the current context
- 2) */network-instances/network-instance/inter-instance-policies/apply-policy/config/export-policy*
→ List of policy names in sequence to be applied on sending a routing update in the current context
- 3) */routing-policy/defined-sets/oc-bgp-pol:bgp-defined-sets/oc-bgp-pol:ext-community-sets/oc-bgp-pol:ext-community-set/oc-bgp-pol:config/oc-bgp-pol:ext-community-set-name* → Name / label of the extended community set -- this is used to reference the set in match conditions
- 4) */routing-policy/defined-sets/oc-bgp-pol:bgp-defined-sets/oc-bgp-pol:ext-community-sets/oc-bgp-pol:ext-community-set/oc-bgp-pol:config/oc-bgp-pol:ext-community-member* → Members of the extended community set
- 5) */routing-policy/policy-definitions/policy-definition/config/name* → Name of the top-level policy definition -- this name is used in references to the current policy
- 6) */routing-policy/policy-definitions/policy-definition/statements/statement/config/name* → name of the policy statement
- 7) */routing-policy/policy-definitions/policy-definition/statements/statement/conditions/config/install-protocol-eq* → Condition to check the protocol / method used to install the route into the local routing table. Options: BGP, OSPF, ISIS, OSPFv3, STATIC, DIRECTLY_CONNECTED, LOCAL_AGGREGATE, IGMP, PIM.
- 8) */routing-policy/policy-definitions/policy-definition/statements/statement/conditions/oc-bgp-pol:bgp-conditions/oc-bgp-pol:match-ext-community-set/oc-bgp-pol:config/oc-bgp-pol:ext-community-set* → References a defined extended community set
- 9) */routing-policy/policy-definitions/policy-definition/statements/statement/conditions/oc-bgp-pol:bgp-conditions/oc-bgp-pol:match-ext-community-set/oc-bgp-pol:config/oc-bgp-pol:match-set-options* → Optional parameter that governs the behavior of the match operation. Options: ANY, ALL, INVERT (NOTE: As from "2019-02-01" version of "openconfig-bgp-policy" module BGP community match-set-options have been moved from policy-definitions/statements/.../bgp-conditions to defined-sets/bgp-defined-sets/community-set. Use new this new location and model (or later) instead).
- 10) */routing-policy/policy-definitions/policy-definition/statements/statement/actions/config/policy-*

result → Select the final disposition for the route, either accept or reject. Options:
ACCEPT_ROUTE, REJECT_ROUTE

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(10) import-policy	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:policy-definitions/oc-rpol:policy-definition/oc-rpol:name	-	controller	rw
(11) export-policy	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:policy-definitions/oc-rpol:policy-definition/oc-rpol:name	-	controller	rw
(12) ext-community-set-name	string	-	controller	rw
(13) ext-community-member	union	-	controller	rw
(14) name	string	-	controller	rw
(15) name	string	-	controller	rw
(16) install-protocol-eq	identityref	-	controller	rw
(17) ext-community-set	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:defined-sets/oc-bgp-pol:bgp-defined-sets/oc-bgp-pol:ext-community-sets/oc-bgp-pol:ext-community-set/oc-bgp-pol:ext-community-set-name	-	controller	rw
(18) match-set-options	enumeration	ANY	controller	rw
(19) policy-result	enumeration	-	controller	rw

Table 1 Opeconfig parameters for step "Add VPN import/export BGP policies (VRF route target) "

Apart from these parameters, in some cases, it could be necessary to complete VPN configuration with information related to MP-iBGP policies for a specific neighbor, in addition to those define for the import/export (RT) within VRF as detailed above. For example, to add/delete/modify a BGP attribute (community, as-path, MED, local-pref, etc) to the routes exchanged with neighbors. To do so, again, we first create a policy by using "routing-policy" module, and then we assign that policy to address-family VPNv4 for the corresponding MP-iBGP neighbors.

4.2.3 Configure Interfaces to access to VPN

Before to be associated to VRF, the PE router access interface connecting node B or Enterprise CPE on client side shall be configured, as described in the figure below. The interface could be a physical Ethernet port or a logical interface, like a port+Vlan subinterface, or an aggregation of several physical ports (LAG). Also, in some scenarios, the interface used to access and be associated to the VRF is not a physical interface, but a “logical” interface like a GRE tunnel or a PseudoWire (VLL). In these cases, first the GRE tunnel or PW has to be defined and then associated to the VRF as if there was a traditional physical interface.

The configuration includes L2/L3 parameters like vlan-id (single or double tagging) or ip-address for interface, subinterface, LAG or GRE tunnel and PW, using yang modules “**iana-if-type**”, “**openconfig-interfaces**”, “**openconfig-if-ip**”, “**openconfig-if-tunnel**”, “**openconfig-lacp**”.

The **xpath** and description of the parameters used are described below.

- 1) `/interfaces/interface/config/name` → The name of the interface.
- 2) `/interfaces/interface/config/type` → The type of the interface. Options: Based on types defined in `ietf-if:interface-type`.
 - For all Ethernet-like interfaces, regardless of speed, as per RFC 3635 use “`ianaift:ethernetCsmacd`”
 - For Loopback Interface use “`ianaift:softwareLoopback`”
 - For GRE Tunnel interfaces use “`ianaift:tunnel`”
 - For L2VLL-L3VPN PW stitching use “`ianaift:ifPwType`”
 - For LAG link aggregation use “`ieee8023adLag`”
- 3) `/interfaces/interface/oc-eth:ethernet/oc-eth:config/oc-eth:duplex-mode` → When auto-negotiate is TRUE, this optionally sets the duplex mode that will be advertised to the peer. If unspecified, the interface should negotiate the duplex mode directly (typically full-duplex). When auto-negotiate is FALSE, this sets the duplex mode on the interface directly. Options: FULL, HALF
- 4) `/interfaces/interface/oc-eth:ethernet/oc-eth:config/oc-eth:port-speed` → When auto-negotiate is TRUE, this optionally sets the port-speed mode that will be advertised to the peer for negotiation. If unspecified, it is expected that the interface will select the highest speed available based on negotiation. When auto-negotiate is set to FALSE, sets the link speed to a fixed value -- supported

values are defined by `ETHERNET_SPEED` identities. Options: `SPEED_10MB`, `SPEED_100MB`, `SPEED_1G`, `SPEED_25G`, `SPEED_40G`, `SPEED_50GB`, `SPEED_100GB`.

- 5) `/interfaces/interface/oc-eth:ethernet/oc-eth:config/oc-eth:auto-negotiate` → Set to `TRUE` to request the interface to auto-negotiate transmission parameters with its peer interface. When set to `FALSE`, the transmission parameters are specified manually.
- 6) `/interfaces/interface/config/mtu` → Set the max transmission unit size in octets for the physical interface. If this is not set, the mtu is set to the operational default -- e.g., 1514 bytes on an Ethernet interface.
- 7) `/interfaces/interface/config/description` → A textual description of the interface
- 8) `/interfaces/interface/config/enabled` → This leaf contains the configured, desired state of the interface. Systems that implement the IF-MIB use the value of this leaf in the 'running' datastore to set IF-MIB.ifAdminStatus to 'up' or 'down' after an ifEntry has been initialized, as described in RFC 2863. Changes in this leaf in the 'running' datastore are reflected in ifAdminStatus, but if ifAdminStatus is changed over SNMP, this leaf is not affected.
- 9) `/interfaces/interface/config/loopback-mode` → When set to true, the interface is logically looped back, such that packets that are forwarded via the interface are received on the same interface.
- 10) `/interfaces/interface/config/oc-vlan:tpid` → Optionally set the tag protocol identifier field (TPID) that is accepted on the VLAN. Options: `TPID_0X8100`, `TPID_0X88A8`, `TPID_0X9100`, `TPID_0X9200`, `TPID_ANY`.
 - Use “`TPID_0X8100`” for 802.1q (by default)
 - Use “`TPID_0X88A8`” for Q-n-Q
- 11) `/interfaces/interface/subinterfaces/subinterface/config/index` → The index of the subinterface, or logical interface number. On systems with no support for subinterfaces, or not using subinterfaces, this value should default to 0, i.e., the default subinterface.
- 12) `/interfaces/interface/subinterfaces/subinterface/config/description` → A textual description of the interface.
- 13) `/interfaces/interface/subinterfaces/subinterface/config/enabled` → This leaf contains the configured, desired state of the interface.
- 14) `/interfaces/interface/subinterfaces/subinterface/oc-vlan:vlan/oc-vlan:config/oc-vlan:vlan-id` → VLAN id for the subinterface -- specified inline for the case of a local VLAN. The id is scoped to the subinterface and could be repeated on different subinterfaces. NOTE: This parameter has been deprecated in the “openconfig-vlan.yang” model, so 34) should be used instead.

- 15) `/interfaces/interface/subinterfaces/subinterface/oc-vlan:vlan/oc-vlan:match/oc-vlan:single-tagged/oc-vlan:config/oc-vlan:vlan-id` → VLAN identifier for single-tagged packets. Used for 802.1q ethernet frames
- 16) `/interfaces/interface/subinterfaces/subinterface/oc-vlan:vlan/oc-vlan:match/oc-vlan:double-tagged/oc-vlan:config/oc-vlan:inner-vlan-id` → Inner VLAN identifier for double-tagged packets. Used for Q-n-Q ethernet frames (C-Vlan)
- 17) `/interfaces/interface/subinterfaces/subinterface/oc-vlan:vlan/oc-vlan:match/oc-vlan:double-tagged/oc-vlan:config/oc-vlan:outer-vlan-id` → Outer VLAN identifier for double-tagged packets. Used for Q-n-Q ethernet frames (S-Vlan)
- 18) `/interfaces/interface/subinterfaces/subinterface/oc-ip:ipv4/oc-ip:addresses/oc-ip:address/oc-ip:config/oc-ip:ip` → The IPv4 address on the interface
- 19) `/interfaces/interface/subinterfaces/subinterface/oc-ip:ipv4/oc-ip:addresses/oc-ip:address/oc-ip:config/oc-ip:prefix-length` → The length of the subnet prefix
- 20) `/interfaces/interface/oc-eth:ethernet/oc-eth:config/oc-lag:aggregate-id` → Specify the logical aggregate interface to which this interface belongs
- 21) `/interfaces/interface/oc-lag:aggregation/oc-lag:config/oc-lag:lag-type` → Sets the type of LAG, i.e., how it is configured / maintained. Options: LACP, STATIC
- 22) `/interfaces/interface/oc-lag:aggregation/oc-lag:config/oc-lag:min-links` → Specifies the minimum number of member interfaces that must be active for the aggregate interface to be available
- 23) `/lacp/interfaces/interface/config/name` → Reference to the interface on which LACP should be configured. The type of the target interface must be `ieee8023adLag`
- 24) `/lacp/interfaces/interface/config/interval` → Set the period between LACP messages -- uses the `lacp-period-type` enumeration.
- 25) `/lacp/interfaces/interface/config/lacp-mode` → ACTIVE is to initiate the transmission of LACP packets, PASSIVE is to wait for peer to initiate the transmission of LACP packets.
- 26) `/lacp/interfaces/interface/config/system-priority` → Sytem priority used by the node on this LAG interface. Lower value is higher priority for determining which node is the controlling system
- 27) `/interfaces/interface/oc-tun:tunnel/oc-tun:config/oc-tun:src` → The source address that should be used for the tunnel.
- 28) `/interfaces/interface/oc-tun:tunnel/oc-tun:config/oc-tun:dst` → The destination address for the tunnel.

- 29) `/interfaces/interface/oc-tun:tunnel/oc-tun:config/oc-tun:tll` → The time-to-live (or hop limit) that should be utilised for the IP packets used for the tunnel transport
- 30) `/interfaces/interface/oc-tun:tunnel/oc-tun:ipv4/oc-tun:unnumbered/oc-tun:config/oc-tun:enabled` → Indicates that the subinterface is unnumbered. By default, the subinterface is numbered, i.e., expected to have an IP address configuration.
- 31) `/interfaces/interface/oc-tun:tunnel/oc-tun:ipv4/oc-tun:addresses/oc-tun:address/oc-tun:config/oc-tun:ip` → The IPv4 address on the interface
- 32) `/interfaces/interface/oc-tun:tunnel/oc-tun:ipv4/oc-tun:addresses/oc-tun:address/oc-tun:config/oc-tun:prefix-length` → The length of the subnet prefix.
- 33) `/interfaces/interface/oc-tun:tunnel/oc-tun:ipv4/oc-tun:config/oc-tun:mtu` → The size, in octets, of the largest IPv4 packet that the interface will send and receive. The server may restrict the allowed values for this leaf, depending on the interface's type. If this leaf is not configured, the operationally used MTU depends on the interface's type.

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(20) name	string	-	controller	rw
(21) type	identityref	-	controller	rw
(22) oc-eth:duplex-mode	enumeration	-	controller	rw
(23) oc-eth:port-speed	identityref	-	controller	rw
(24) oc-eth:auto-negotiate	boolean	TRUE	controller	rw
(25) mtu	uint16	-	controller	rw
(26) description	string	-	controller	rw
(27) enabled	boolean	TRUE	controller	rw
(28) loopback-mode	boolean	FALSE	controller	rw
(29) oc-vlan:tpid	identityref	TPID_0X8100	controller	rw
(30) index	uint32	0	controller	rw
(31) description	string	-	controller	rw
(32) enabled	boolean	TRUE	controller	rw
(33) oc-vlan:vlan-id	union	-	controller	rw
(34) oc-vlan:vlan-id	uint16	-	controller	rw
(35) oc-vlan:inner-vlan-id	uint16	-	controller	rw
(36) oc-vlan:outer-vlan-id	uint16	--	controller	rw
(37) oc-ip:ip	string	-	controller	rw
(38) oc-ip:prefix-length	uint8	-	controller	rw
(39) oc-lag:aggregate-id	(Leafref Path) /oc-if:interfaces/oc-if:interface/oc-if:name	-	controller	rw
(40) oc-lag:lag-type	enumeration	-	controller	rw

(41) oc-lag:min-links	uint16	-	controller	rw
(42) name	(Leafref Path) /oc-if:interfaces/oc-if:interface/oc-if:name	-	controller	rw
(43) interval	lacp-period-type	-	controller	rw
(44) lacp-mode	lacp-activity-type	-	controller	rw
(45) system-priority	uint16	-	controller	rw
(46) oc-tun:src	union	-	controller	rw
(47) oc-tun:dst	union	-	controller	rw
(48) oc-tun:ttl	uint8	-	controller	rw
(49) oc-tun:enabled	boolean	FALSE	controller	rw
(50) oc-tun:ip	string	-	controller	rw
(51) oc-tun:prefix-length	uint8	-	controller	rw
(52) oc-tun:mtu	uint16	-	controller	rw

Table 2 Opeconfig parameters for step “Configure Interfaces to access to VPN”

4.2.4 Bind interfaces/subinterfaces to VRF (VPN End Points)

The interface and subinterface configured in previous step is not yet associated to a L3VPN service and, therefore, not included into a VRF in the network element. Now, as depicted below, we should bind the interface or subinterface (or LAG or tunnel or PW) previously defined to the network instance as client interface by using “**openconfig-network-instance**” and “**openconfig-interface**” modules.

The **xpath** and description of the parameters used are described below.

- 34) */network-instances/network-instance/interfaces/interface/config/id* → A unique identifier for this interface - this is expressed as a free-text string
- 35) */network-instances/network-instance/interfaces/interface/config/interface* → Reference to a base interface. If a reference to a subinterface is required, this leaf must be specified to indicate the base interface.
- 36) */network-instances/network-instance/interfaces/interface/config/subinterface* → Reference to a subinterface -- this requires the base interface to be specified using the interface leaf in this container. If only a reference to a base interface is required, this leaf should not be set.

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(53) id	string	-	controller	rw

(54) interface	(Leafref Path) /oc-if:interfaces/oc-if:interface/oc-if:name	-	controller	rw
(55) subinterface	(Leafref Path) /oc-if:interfaces/oc-if:interface[oc-if:name=current()/../interface]/oc-if:subinterfaces/oc-if:subinterface/oc-if:index	-	controller	rw

Table 3 Opeconfig parameters for step “Bind interfaces/subinterfaces to VRF (VPN End Points)”

4.2.5 Redistribute routing protocols on client connection (CPE) inside VRF

Additionally, route redistribution inside VRF from different protocols (and associated policies if apply) can be configured. OpenConfig provides a set of tables within a VPN/VRF. The tables represent various per-protocol RIBs that can exist in a network instance. Further, OpenConfig models allow for the interconnection of RIBs through an explicit table-connection list within the network-instance model. Thus, in order to redistribute routes within protocol A to protocol B, an explicit connection between the tables corresponding to protocol A and protocol B should be created.

Accordingly, within Openconfig “network-instance” module, first it would be necessary to define the protocols involved in the redistribution (direct connections, static routes, OSPF or BGP). Then, associate a table per each protocol and finally create table-connections as required from a source protocol to destination protocol (Note: when a BGP instance is created with IPv4 and IPv6 address families enabled, the protocol=BGP, address-family=IPv4 table is created by the system). Although “ACCEPT_ROUTE” is set as default-import-policy, if needed, a policy could be also added to explicit routes being imported into the destination protocol's RIB.

The **xpath** and description of the parameters used are described below. It includes parametrization needed for BGP, OSPF(v2) and static routes, taken from “**openconfig-network-instance**” and “**openconfig-routing-policy**” modules.

- 37) `/network-instances/network-instance/protocols/protocol/config/identifier` → The protocol identifier for the instance. Options: DIRECTLY_CONNECTED, STATIC, OSPF, BGP, ISIS
- 38) `/network-instances/network-instance/protocols/protocol/config/name` → A unique name for the protocol instance

-
- 39) */network-instances/network-instance/protocols/protocol/config/enabled* → A boolean value indicating whether the local protocol instance is enabled.
 - 40) */network-instances/network-instance/tables/table/config/protocol* → Reference to the protocol that the table is associated with.
 - 41) */network-instances/network-instance/tables/table/config/address-family* → The address family (IPv4, IPv6) of the table's entries
 - 42) */network-instances/network-instance/table-connections/table-connection/config/src-protocol* → The source protocol for the table connection
 - 43) */network-instances/network-instance/table-connections/table-connection/config/address-family* → The address family associated with the connection. This must be defined for the source protocol. The target address family is implicitly defined by the address family specified for the source protocol
 - 44) */network-instances/network-instance/table-connections/table-connection/config/dst-protocol* → The destination protocol for the table connection
 - 45) */network-instances/network-instance/table-connections/table-connection/config/import-policy* → List of policy names in sequence to be applied on receiving a routing update in the current context, e.g. for the current peer group, neighbor, address family, etc.
 - 46) */network-instances/network-instance/table-connections/table-connection/config/default-import-policy* → Explicitly set a default policy if no policy definition in the import policy chain is satisfied.
Options: ACCEPT_ROUTE, REJECT_ROUTE
 - 47) */network-instances/network-instance/protocols/protocol/bgp/global/config/as* → Local autonomous system number of the router. Uses the 32-bit as-number type from the model in RFC 6991
 - 48) */network-instances/network-instance/protocols/protocol/bgp/global/config/router-id* → Router id of the router - an unsigned 32-bit integer expressed in dotted quad notation
 - 49) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/neighbor-address* → Address of the BGP peer, either in IPv4 or IPv6
 - 50) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/peer-as* → AS number of the peer.
 - 51) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/local-as* → The local autonomous system number that is to be used when establishing sessions with the remote peer or peer group, if this differs from the global BGP router autonomous system number
 - 52) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/peer-type* → Explicitly designate the peer or peer group as internal (iBGP) or external (eBGP).
 - 53) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/remove-private-as* → Remove private AS numbers from updates sent to peers – when this leaf is not specified, the AS_PATH attribute should be sent to the peer unchanged. Options: PRIVATE_AS_REMOVE_ALL, PRIVATE_AS_REPLACE_ALL.
 - 54) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/config/description* → An optional textual description (intended primarily for use with a peer or group)
-

- 55) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/ebgp-multihop/config/enabled* → When enabled the referenced group or neighbors are permitted to be indirectly connected - including cases where the TTL can be decremented between the BGP peers
- 56) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/ebgp-multihop/config/multihop-ttl* → Time-to-live value to use when packets are sent to the referenced group or neighbors and ebgp-multihop is enabled
- 57) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/apply-policy/config/import-policy* → List of policy names in sequence to be applied on receiving a routing update in the current context, e.g., for the current peer group, neighbor, address family, etc.
- 58) */network-instances/network-instance/protocols/protocol/bgp/neighbors/neighbor/apply-policy/config/export-policy* → List of policy names in sequence to be applied on sending a routing update in the current context, e.g., for the current peer group, neighbor, address family, etc.
- 59) */network-instances/network-instance/protocols/protocol/static-routes/static/config/prefix* → Destination prefix for the static route, either IPv4 or IPv6.
- 60) */network-instances/network-instance/protocols/protocol/static-routes/static/next-hops/next-hop/config/next-hop* → The next-hop that is to be used for the static route - this may be specified as an IP address, an interface or a pre-defined next-hop type - for instance, DROP or LOCAL_LINK. When this leaf is not set, and the interface-ref value is specified for the next-hop, then the system should treat the prefix as though it is directly connected to the interface
- 61) */network-instances/network-instance/protocols/protocol/static-routes/static/next-hops/next-hop/config/metric* → A metric which is utilised to specify the preference of the next-hop entry when it is injected into the RIB. The lower the metric, the more preferable the prefix is. When this value is not specified the metric is inherited from the default metric utilised for static routes within the network instance that the static routes are being instantiated. When multiple next-hops are specified for a static route, the metric is utilised to determine which of the next-hops is to be installed in the RIB. When multiple next-hops have the same metric (be it specified, or simply the default) then these next-hops should all be installed in the RIB
- 62) */network-instances/network-instance/protocols/protocol/ospfv2/global/config/router-id* → A 32-bit number represented as a dotted quad assigned to each router running the OSPFv2 protocol. This number should be unique within the autonomous system.
- 63) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/config/identifier* → An identifier for the OSPFv2 area - described as either a 32-bit unsigned integer, or a dotted-quad
- 64) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/config/id* → An operator-specified string utilised to uniquely reference this interface
- 65) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/config/network-type* → The type of network that OSPFv2 should use for the specified interface. Options: BROADCAST_NETWORK, NON_BROADCAST_NETWORK, POINT_TO_POINT_NETWORK

- 66) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/config/passive* → When this leaf is set to true, the interface should be advertised within the OSPF area but OSPF adjacencies should not be established over the interface
- 67) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/config/authentication-type* → The type of authentication that should be used on this interface
- 68) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/interface-ref/config/interface* → Reference to a base interface. If a reference to a subinterface is required, this leaf must be specified to indicate the base interface
- 69) */network-instances/network-instance/protocols/protocol/ospfv2/areas/area/interfaces/interface/interface-ref/config/subinterface* → Reference to a subinterface -- this requires the base interface to be specified using the interface leaf in this container. If only a reference to a base interface is required, this leaf should not be set.

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(56) identifier	identityref	-	controller	rw
(57) name	string	-	controller	rw
(58) enabled	boolean	-	controller	rw
(59) protocol	(Leafref Path) ../..../protocols/protocol/config/identifier	-	controller	rw
(60) address-family	identityref	-	controller	rw
(61) src-protocol	(Leafref Path) ../..../tables/table/config/protocol	-	controller	rw
(62) address-family	(Leafref Path) ../..../tables/table[protocol=current()]/../src-protocol/config/address-family	-	controller	rw
(63) dst-protocol	(Leafref Path) ../..../tables/table/config/protocol	-	controller	rw
(64) import-policy	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:policy-definitions/oc-rpol:policy-definition/oc-rpol:name	-	controller	rw
(65) default-import-policy	enumeration	REJECT_ROUTE	controller	rw
(66) as	uint32	-	controller	rw
(67) router-id	string	-	controller	rw
(68) neighbor-address	union	-	controller	rw
(69) peer-as	uint32	-	controller	rw

(70) local-as	uint32	-	controller	rw
(71) peer-type	enumeration	-	controller	rw
(72) remove-private-as	identityref	-	controller	rw
(73) description	string	-	controller	rw
(74) enabled	boolean	FALSE	controller	rw
(75) multihop-ttl	uint8		controller	rw
(76) import-policy	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:policy-definitions/oc-rpol:policy-definition/oc-rpol:name	-	controller	rw
(77) export-policy	(Leafref Path) /oc-rpol:routing-policy/oc-rpol:policy-definitions/oc-rpol:policy-definition/oc-rpol:name	-	controller	rw
(78) prefix	union	-	controller	rw
(79) next-hop	union		controller	rw
(80) metric	uint32	-	controller	rw
(81) router-id	string	-	controller	rw
(82) identifier	union	-	controller	rw
(83) id	string	-	controller	rw
(84) network-type	identityref	-	controller	rw
(85) passive	boolean	-	controller	rw
(86) authentication-type	string	-	controller	rw
(87) interface	(Leafref Path) /oc-if:interfaces/oc-if:interface/oc-if:name	-	controller	rw
(88) subinterface	(Leafref Path) /oc-if:interfaces/oc-if:interface[oc-if:name=current()/../interface]/oc-if:subinterfaces/oc-if:subinterface/oc-if:index	-	controller	rw

Table 4 Opeconfig parameters for step “Redistribute routing protocols on client connection (CPE) inside VRF”

4.2.6 Append additional configuration for QoS

Apart from previous steps, in some cases, it could be necessary to complete VPN configuration with complementary parametrization related QoS policy to apply to inbound/outbound traffic within VPN. Furthermore, could also be needed to add some kind of TE constraint in order to steer VPN traffic for a specific path.

Thus, for QoS the corresponding Openconfig parametrization is detailed in the next table. It includes the

different stages of the QoS traffic process:

- **Classification:** Organizes traffic on the basis of whether or not it matches a specific criteria by identifying all key packet fields: CoS, DSCP, IP precedence, or MPLS EXP field of the incoming packets.
- **Policing:** Determines whether a packet is in or out of profile by comparing the rate of the incoming traffic to the configured policer CIR and PIR. A set of actions should be performed on packets that conform to the CIR and PIR (conform-action), packets that conform to the PIR, but not the CIR (exceed-action), and packets that exceed the PIR value (violate-action).
- **Marking:** All packets that belong to a classification can be remarked. Furthermore, when you configure a policer, packets that meet or exceed the permitted bandwidth requirements (CIR/PIR) can be conditionally passed through, dropped, or marked.
- **Queueing:** Differentiates traffic classes and regulates the queue size based on the classification to avoid network bottlenecks. It uses some algorithms for congestion avoidance such as tail drop, RED or WRED by providing a drop precedence.
- **Scheduling:** Manages congestion by providing a guaranteed bandwidth to a particular class of traffic while also serving other traffic in a fair way. It uses different mechanisms (traffic shaping, WFQ, priority queueing) to limit the maximum bandwidth that can be consumed by a particular class of traffic and ensure that delay-sensitive traffic in a low-latency queue is sent before traffic in other queues.

Accordingly, the xpath and description of the parameters used from “openconfig-qos” module are described below.

70) /qos/queues/queue/config/name → User-defined name of the queue

71) /qos/queues/queue/config/queue-type → Sets the type of the queue. Options: DROP_TAIL, RED, WRED. The minimum threshold parameter for a RED-managed queue. When the average queue length is less than minth, all packets are admitted to the queue.

72) /qos/queues/queue/red/config/minth → The minimum threshold parameter for a RED-managed queue. When the average queue length is less than minth, all packets are admitted to the queue

73) /qos/queues/queue/red/config/maxth → The maximum threshold parameter for a RED-managed queue. When the average queue length exceeds the maxth value, all packets are dropped (or marked if ECN is enabled).

- 74) `/qos/queues/queue/wred/config` → Configuration data for WRED. Container created but not developed in the qos.yang module. Note: Pending of definition in Openconfig standard model.
- 75) `/qos/scheduler-policies/scheduler-policy/config/name` → Name for the scheduler policy
- 76) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/config/sequence` → Sequence number for the scheduler within the scheduler policy. Schedulers are processed from lowest sequence to highest.
- 77) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/config/type` → Sets the type of scheduler, i.e. the scheduling algorithm used to serve inputs. Options: ONE_RATE_TWO_COLOR, TWO_RATE_THREE_COLOR
- 78) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/config/priority` → Priority of the scheduler within the scheduler policy. Options: STRICT
- 79) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/inputs/input/config/id` → User-defined identifier for the scheduler input
- 80) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/inputs/input/config/input-type` → Describes the type of input source for the scheduler. Options: QUEUE, IN_PROFILE, OUT_PROFILE
- 81) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/inputs/input/config/queue` → Reference to a queue that is an input source for the scheduler
- 82) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/inputs/input/config/weight` → For priority schedulers, this indicates the priority of the corresponding input. Higher values indicate higher priority. For weighted round-robin schedulers, this leaf indicates the weight of the corresponding input.
- 83) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/one-rate-two-color/config/cir` → Committed information rate for the single-rate token bucket scheduler. This value represents the rate at which tokens are added to the bucket.
- 84) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/one-rate-two-color/config/bc` → Committed burst size for the single-rate token bucket scheduler. This value represents the depth of the token bucket.
- 85) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/one-rate-two-color/config/queuing-behavior` → The type of scheduler that is being configured. Options: SHAPE, POLICE
- 86) `/qos/scheduler-policies/scheduler-policy/schedulers/scheduler/one-rate-two-color/config/max-queue-depth-bytes` → When the scheduler is specified to be a shaper – the maximum depth of the queue in bytes is the value specified by this leaf.

- 87) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/config/cir* → Committed information rate for the dual-rate token bucket policer. This value represents the rate at which tokens are added to the primary bucket.
- 88) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/config/pir* → Peak information rate for the dual-rate token bucket policer. This value represents the rate at which tokens are added to the secondary bucket.
- 89) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/config/bc* → Committed burst size for the dual-rate token bucket policer. This value represents the depth of the tokenbucket.
- 90) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/config/be* → Excess burst size for the dual-rate token bucket policer. This value represents the depth of the secondary bucket..
- 91) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/exceed-action/config/set-dscp* → Sets the 6-bit DSCP (differentiated services code point) value in the IP packet header.
- 92) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/exceed-action/config/set-dot1p* → Sets the 3-bit class-of-service value in the Ethernet packet header for 802.1Q VLAN-tagged packets, also known as PCP (priority code point).
- 93) */qos/scheduler-policies/scheduler-policy/schedulers/scheduler/two-rate-three-color/exceed-action/config/drop* → If set to true, packets within this context are dropped.
- 94) */qos/classifiers/classifier/config/name* → User-assigned name of the classifier
- 95) */qos/classifiers/classifier/config/type* → Type of classifier. Options: IPV4, IPV6, ETHERNET, MPLS
- 96) */qos/classifiers/classifier/terms/term/config/id* → Identifier for the match term
- 97) */qos/classifiers/classifier/terms/term/conditions/ipv4/config/source-address* → Source IPv4 address prefix
- 98) */qos/classifiers/classifier/terms/term/conditions/ipv4/config/destination-address* → Destination IPv4 address prefix.
- 99) */qos/classifiers/classifier/terms/term/conditions/ipv4/config/dscp* → Value of diffserv codepoint.
- 100) */qos/classifiers/classifier/terms/term/actions/config/target-group* → References the forwarding group or class to which the matched packets should be assigned
- 101) */qos/forwarding-groups/forwarding-group/config/name* → Name of the forwarding group
- 102) */qos/forwarding-groups/forwarding-group/config/output-queue* → Queue for packets in this forwarding group.

- 103) `/qos/interfaces/interface/config/interface-id` → Identifier for the interface.
- 104) `/qos/interfaces/interface/interface-ref/config/interface` → Reference to a base interface. If a reference to a subinterface is required, this leaf must be specified to indicate the base interface.
- 105) `/qos/interfaces/interface/interface-ref/config/subinterface` → Reference to a subinterface -- this requires the base interface to be specified using the interface leaf in this container. If only a reference to a base interface is required, this leaf should not be set.
- 106) `/qos/interfaces/interface/input/queues/queue/config/name` → Reference to the queue associated with this interface. A queue may be explicitly configured, or implicitly created by the system based on default queues that are instantiated by a hardware component or are assumed to be default on the system.
- 107) `/qos/interfaces/interface/input/classifiers/classifier/config/name` → Reference to the classifier to be applied to ingress traffic on the interface
- 108) `/qos/interfaces/interface/input/classifiers/classifier/config/type` → Type of packets matched by the classifier. Options: IPV4, IPV6, MPLS
- 109) `/qos/interfaces/interface/input/scheduler-policy/config/name` → Reference to the queue associated with this interface. A queue may be explicitly configured, or implicitly created by the system based on default queues that are instantiated by a hardware component, or are assumed to be default on the system
- 110) `/qos/interfaces/interface/output/queues/queue/config/name` → Reference to the queue associated with this interface. A queue may be explicitly configured, or implicitly created by the system based on default queues that are instantiated by a hardware component or are assumed to be default on the system.
- 111) `/qos/interfaces/interface/output/scheduler-policy/config/name` → The scheduler policy to be applied to traffic on this interface.

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(89) name	string	-	controller	rw
(90) queue-type	identityref	-	controller	rw
(91) minth	uint64	-	controller	rw
(92) maxth	uint64	-	controller	rw
(93) config	container	-	controller	rw
(94) name	string	-	controller	rw
(95) sequence	uint32	-	controller	rw
(96) type	identityref	-	controller	rw

(97) priority	enumeration	-	controller	rw
(98) id	string	-	controller	rw
(99) input-type	enumeration	-	controller	rw
(100) queue	(Leafref Path) .././.././.././../queues/queue/name	-	controller	rw
(101) weight	uint64	-	controller	rw
(102) cir	uint64	-	controller	rw
(103) bc	uint32	-	controller	rw
(104) queuing-behavior	enumeration	-	controller	rw
(105) max-queue-depth-bytes	uint32	-	controller	rw
(106) cir	uint64	-	controller	rw
(107) pir	uint64	-	controller	rw
(108) bc	uint32	-	controller	rw
(109) be	uint32	-	controller	rw
(110) set-dscp	uint8	-	controller	rw
(111) set-dot1p	uint8	-	controller	rw
(112) drop	boolean	-	controller	rw
(113) name	string	-	controller	rw
(114) type	enumeration	-	controller	rw
(115) id	string	-	controller	rw
(116) source-address	string	-	controller	rw
(117) destination-address	string	-	controller	rw
(118) dscp	uint8	-	controller	rw
(119) target-group	(Leafref Path) .././.././.././../forwarding-groups/forwarding-group/config/name	-	controller	rw
(120) name	string	-	controller	rw
(121) output-queue	(Leafref Path) .././.././../queues/queue/config/name	-	controller	rw
(122) interface-id	string	-	controller	rw
(123) interface	(Leafref Path) /oc-if:interfaces/oc-if:interface/oc-if:name	-	controller	rw
(124) subinterface	(Leafref Path) /oc-if:interfaces/oc-if:interface[oc-if:name=current()../interface]/oc-if:subinterfaces/oc-if:subinterface/oc-if:index	-	controller	rw
(125) name	string	-	controller	rw
(126) name	string	-	controller	rw
(127) type	enumeration	-	controller	rw
(128) name	(Leafref Path) .././.././../scheduler-policies/scheduler-policy/config/name	-	controller	rw
(129) name	string	-	controller	rw

(130) name	(Leafref Path) ../..../scheduler-policies/scheduler-policy/config/name	-	controller	rw
------------	---	---	------------	----

Table 5 Opeconfig parameters for step “Append additional configuration for QoS”

This section has described the general procedure. In the next sections, the details for each particular L3VPN use case is described.

4.3 Use case 1.1 L3VPN for 3G/4G Services. [L3VPN/Dot1Q/None/None]

4.3.1.1 Use case definition

Number	1.1
Name	L3VPN for 3G/4G Services. [L3VPN/Dot1Q/None/None]
Brief description	Backhaul mobile network represents the interconnection between the data network and the mobile network. L3VPN services are widely deployed in the IP/MPLS networks. It supports the L3 transport of 3G and 4G services, offering specific traffic treatment through QoS policies applied within L3VPN. These services consume several logical resources (RD, RT, VLANs, IP Address, etc.) to be deployed correctly. The maintenance is done in the network daily and several areas get involved.

Table 6 Definition of Use Case 1.1

4.3.1.2 Description

The scenario that would apply for this use case definition is outlined in the following table.

Service Type	L3VPN
End Point Connection Type	Dot1Q
Routing Protocol used to Connect the CPE	Direct
QoS policies applied in the service	None

Table 7 Use case 1.1 VPN scenario

In the **L3VPN for 3G/4G Services** the nodeB/eNodeB are directly connected to the cell site gateway layer. The cell site gateway acts as a first access layer for nodes that shares the same geographical location. The L3VPN can be implemented over a multiple IGP network, in the example two IGP for backhaul and the one core. IP network is therefore structured in Hierarchy Levels where the first border router receives traffic

from the cell site gateway collecting rings of the same geographical location and aggregates them towards the border router to connect with the Core layer.

From the backhaul network point of view, a L3VPN is created on cell site gateway where, as a general rule, and depending on the operator, several sub-interfaces are created, each one with its specific IP Address and VLAN range. Node B interfaces can be assigned to the same or different VPN depending on the specific traffic constraints, such as:

- **S1 Interface:** Control plane and user plane Traffic (S1-MME & S1-U respectively)
- **Sync Interface:** Synchronization Plane Traffic (clock signaling)
- **O&M Interface:** Management Plane Traffic

Finally, the L3VPN created reach the core IGP (HL3 routers) where 3G/4G Core platforms are allocated. Accordingly, the following diagram depicts a generic network scenario to be used as reference for the configuration of this kind of services:

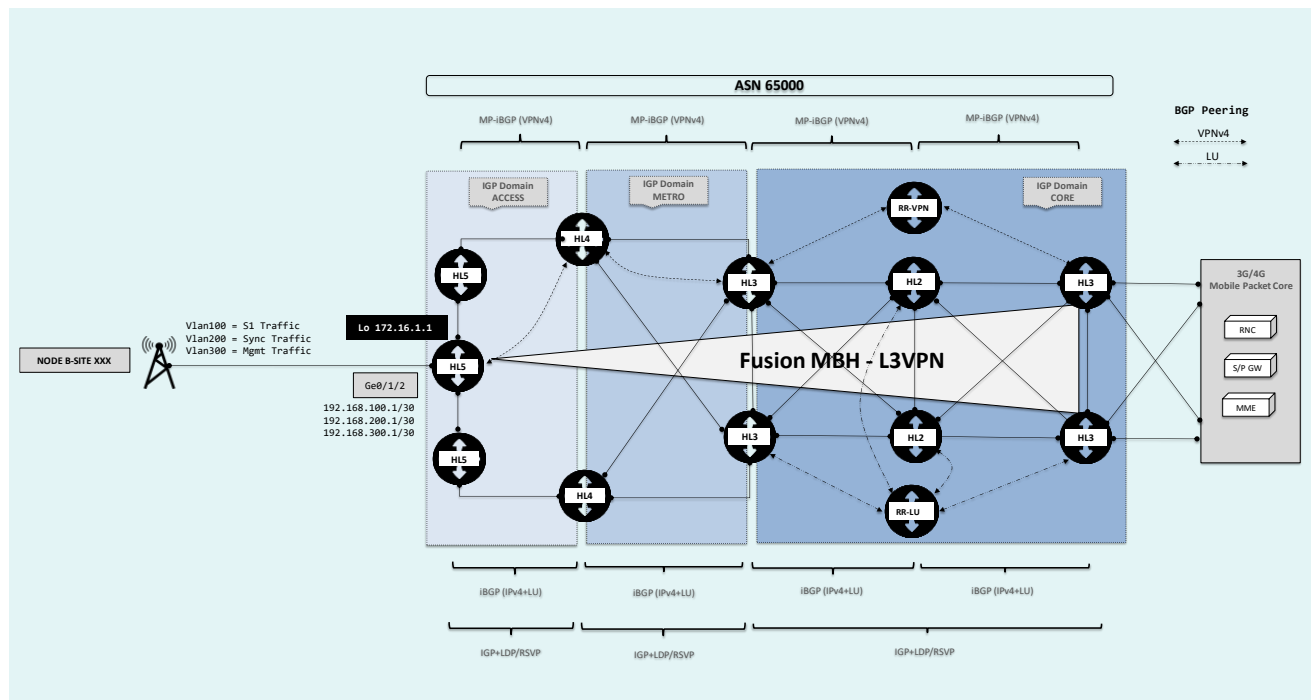


Figure 5. Generic network topology for L3VPN 3G/4G service

To deploy this service several conditions should be evaluated, each of them triggering processes and related configurations, each consuming a subset of the data L3VPN related models, following a flow diagram like this:

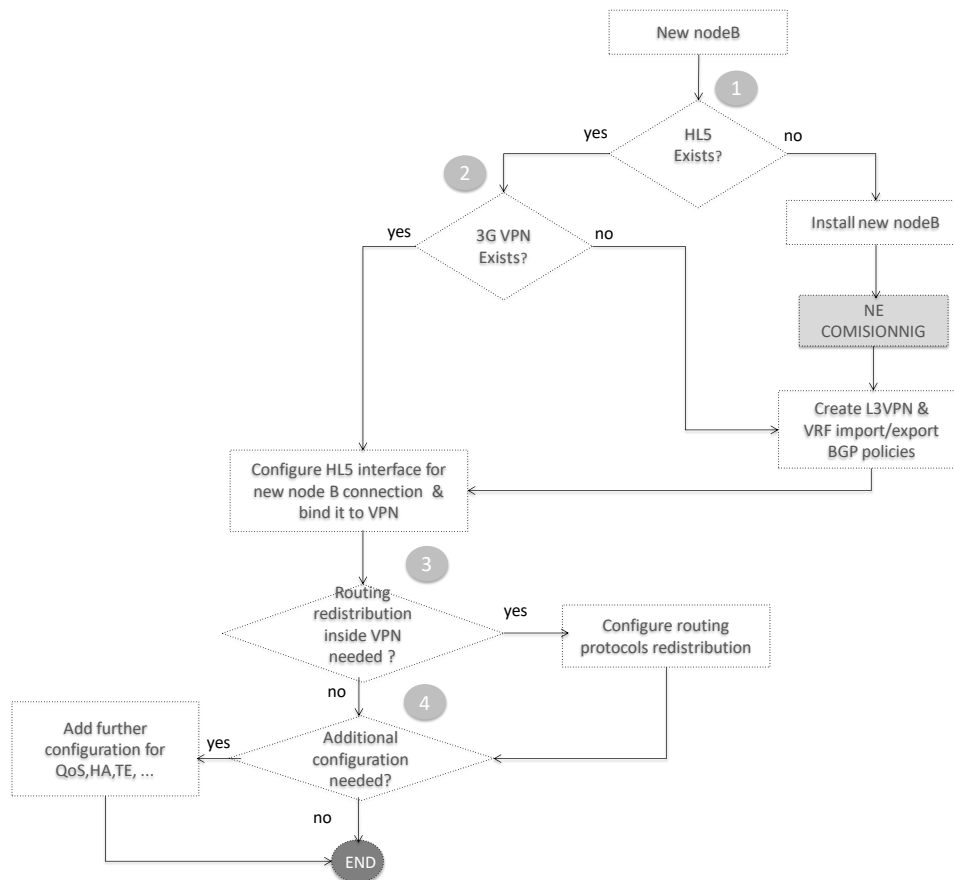


Figure 6. Configuration process flow for L3VPN 3G/4G service

1 → Is the HL5 (Cell Site gateway) already deployed in the network?

- **Yes.** Continue to the next step.
- **No.** Some commissioning process may be done. This Includes Physical/Logical activation of the device, interfaces, and protocols.

2 → Is the L3VPN already configured at the HL5 Cell Site gateway) receiving the nodeB?

- **Yes.** Just the interface parameters and additional configuration must be included

- **No.** L3VPN initial parametrization must be done.
- 3 →** Routing protocols redistribution needed?
 - **Yes.** Append corresponding configuration
 - **No.** Continue with the next step
- 4 →** Do the L3VPN need additional configuration?
 - **Yes.** Applicable new parameters for QoS, HA, TE, etc. should be included in the configuration
 - **No.** End

4.3.1.3 SBI Configuration example

An example (according to diagram depicted in Figure 6. Configuration process flow for L3VPN 3G/4G service) of the configuration workflow process, as described in section 4.2, with the set of Netconf parameters to be configured are detailed below. The required steps would be the following:

Notes:

1. It's assumed that the L3VPN configuration part on HL3 is already completed. Thus, the focus is to automate the HL5 configuration part.
2. In real operational networks, VPNID, VLANIDs, Route Targets, Route Distinguisher, IPs and the rest of logical resources needed for VPN configuration shall be derived by the network planning team during the service design from standard naming conventions applicable in each country.

- [Create VPN \(VRF instance definition\)](#)

The following table contains example values for each one of the parameters above described:

Parameter	Value (example)
(1) name	"MBH_VPN_S1"
(2) description	"Fusion L3VPN S1 TRAFFIC VLAN100"
(3) type	L3VRF
(4) enabled	TRUE
(5) router-id	172.16.1.1
(6) route-distinguisher	65000:100
(7) enabled-address-families	IPV4
(8) encapsulation-type	MPLS
(9) label-allocation-mode	INSTANCE_LABEL

Table 8 Example of parameters for VRF Instance creation

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case. (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Service Provisioning-1	Create L3VPN network-instance	Configure the set of general parameters required for the creation of the new L3VPN service, including name, description and RD, together with address-family and label-allocation mode for the VRF.

Table 9 Atomic Operations for Use Case 1.1 (1)

- Add VPN import/export BGP policies (VRF route target)

The following table contains example values for each one of the parameters above described:

NOTE: For this example, an unique RT for Export and Import is used.

Parameter	Value (example)
(10) import-policy	"RT_IMPORT_POLICY"
(11) export-policy	"RT_EXPORT_POLICY"
(12) ext-community-set-name	"RT_COMMUNITY_SET"
(13) ext-community-member	65000:100
(14) name	"RT_IMPORT_POLICY"
(15) name	"RT_IMPORT_POLICY_STM_1"
(16) install-protocol-eq	BGP
(17) ext-community-set	"RT_COMMUNITY_SET"
(18) match-set-options	ANY
(19) policy-result	ACCEPT_ROUTE
(14) name	"RT_EXPORT_POLICY"
(15) name	"RT_EXPORT_POLICY_STM_1"
(16) install-protocol-eq	BGP
(17) ext-community-set	"RT_COMMUNITY_SET"
(18) match-set-options	ANY
(19) policy-result	ACCEPT_ROUTE

On the other hand, the following table represents the set of NETCONF operations that must be supported

by the SBI of the SDN controller for implementing this use case (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Service Provisioning-2	Define BGP Route Target for a L3VPN network-instance	Configuration of RT (BGP extended communities) to be used in the VRF export and import policies
Service Provisioning-3	Create BGP Routing Policies Import/Export for a L3VPN network-instance	Create routing policies to be applied to routes imported and exported to/from the VRF.
Service Provisioning-4	Apply BGP Import/export Policy (Route Target) to L3VPN network instance	Assign BGP import and export policies to network-instance

Table 10 Atomic Operations for Use Case 1.1 (2)

- **Configure Interfaces and Sub-interfaces to access VPN**

The following table contains example values for each one of the parameters above described:

NOTE: According of diagram (Fig 8), only one 802.1Q subinterface for VLAN=100 is configured. For the rest subinterfaces, the same procedure shall be followed.

Parameter	Value (example)
(20) name	GE0/1/2
(21) type	ianaift:ethernetCsmacd
(25) mtu	1600
(26) description	“NODE_B_SITE_XXX”
(27) enabled	TRUE
(29) oc-vlan:tpid	TPID_0X8100
(30) index	100
(31) description	“NODE_B_SITE_XXX_VLAN_100_S1”
(32) enabled	TRUE
(33) oc-vlan:vlan-id	100
(34) oc-vlan:vlan-id	100
(37) oc-ip:ip	192.168.100.1
(38) oc-ip:prefix-length	30

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the

complementary annex sample RPC calls are documented.

ID	Operation	Description
Service Provisioning-5	Configure interfaces/subinterfaces L1/L2 parameters (Ethernet)	Before to be associated to VRF, PE access interface connecting node B or Enterprise CPE on client side shall be configured with Ethernet L1 physical (speed, duplex,etc) and L2 parameters (no tag, tagged, double tagged, vlan-id,...).
Service Provisioning-6	Configure interfaces/subinterfaces L3 parameters (IP address & mask)	Specify IP address and mask of the PE interface/subinterface
Service Provisioning-11	Create a logical aggregate interface (LAG) with/without LACP	Create a LAG (static or LACP type) joining a number of physical interfaces as belonging to a single aggregate interface.

Table 11 Atomic Operations for Use Case 1.1 (3)

- Bind interfaces/sub-interfaces to VRF (VPN End Points)

The following table contains example values for each one of the parameters above described:

NOTE: According of diagram (Fig 8), only one 802.1Q subinterface for VLAN=100 is configured. For the rest subinterfaces, the same procedure shall be followed.

Parameter	Value (example)
(1) name	"MBH_VPN_S1"
(52) id	"NODE_B_SITE_XXX_VLAN_100_S1"
(53) interface	GE0/1/2
(54) subinterface	100

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see "**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**"). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Service Provisioning-7	Add interfaces (endpoint) to L3VPN network instance	Associate (bind) the physical or logical interface (Vlan subinterface/LAG interface/ GRE tunnel/PW) previously defined to the VRF as VPN client interface.

Table 12 Atomic Operations for Use Case 1.1 (4)

- Configure redistribution of routing protocols on client connection (Node B) inside VRF

The following table contains example values for each one of the parameters above described:

NOTE: For this example, static and directly connected routes will be redistributed into BGP.

Parameter	Value (example)
(56) identifier	DIRECTLY_CONNECTED
(57) name	"DIRECTLY_CONNECTED"
(58) enabled	TRUE
(59) protocol	DIRECTLY_CONNECTED
(60) address-family	IPV4
(56) identifier	BGP
(57) name	"BGP"
(58) enabled	TRUE
(59) protocol	BGP
(60) address-family	IPV4
(61) src-protocol	DIRECTLY_CONNECTED
(62) address-family	IPV4
(63) dst-protocol	BGP
(64) import-policy	"REDISTRIBUTE_CONNECTED"
(65) default-import-policy	ACCEPT_ROUTE

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see "**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**"). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Service Provisioning-8	Define routing protocols used within L3VPN network instance	Define the protocols involved in the redistribution (direct connections, static routes, OSPF or BGP) within the VRF.
Service Provisioning-9	Assign a routing table (RIB) for each protocol within a L3VPN network instance	Create and associate a RIB table per each protocol within the network instance
Service Provisioning-10	Create protocol redistribution policies within a L3VPN network instance	In order to redistribute routes within VRF between protocol A to protocol B, an explicit connection for the tables corresponding to protocol A and protocol B should be created.

5

Inventory Support Use cases (category 2)

The set of uses cases related to inventory support on the SBI interface apply to recover information and state about hardware components of NE and the logical configuration of the device....

5 Inventory Support Use Cases (category 2)

5.1 Structure and Classification

The set of uses cases related to inventory support on the SBI interface apply to recover information and state about hardware components of NE and the logical configuration of the device. In this deliverable, one inventory use case is covered. Note that the inventory use cases are aimed at reading information from the device, without

ID	Use case Title	Section
2.1	Retrieve logical [Interfaces] Inventory	5.2

Table 14 Inventory Use Cases covered in D1.1

5.2 Use case 2.1: Retrieve Logical Interfaces Inventory

5.2.1 Use Case Definition

Number	2.1
Name	Retrieve logical interfaces inventory
Brief description	The use case focuses on retrieving the information regarding all the logical or virtual interfaces of a specific equipment such as subinterfaces, vlan interfaces, tunnel interfaces and other non-physical interfaces.

Table 15 Definition of Use Case 2.1

5.2.2 Description

This use case aims at inventorying logical resources and associated parameters of a previously

commissioned NE through base Openconfig SBI <get> operation. Specifically, parameters concerning logical interfaces inventory would look like those used for configuring interfaces to access to VPN, as explained in section 4.2.3.

5.3 Network Inventory Atomic Operations

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing the inventory support use cases (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**” for). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Network Inventory - 1	Retrieve logical interfaces configuration from one node	Retrieve all the parameters related to interface configuration (description, IP addressing, vlan, etc.) of a node. In this operation information about all the interfaces is received.

Table 16 Network Inventory Atomic Operations

6

Topology and Discovery Use cases (category 3)

A set of abstractions have been defined in order to represent several views of the network topology. However, to gather this information and represent it correctly, the SDN controller must be able to collect the required information from the Network Devices ...



6 Topology and Discovery Use Cases (category 3)

6.1 Structure and Classification

A set of abstractions have been defined in order to represent several views of the network topology. However, to gather this information and represent it correctly, the SDN controller must be able to collect the required information from the Network Devices. Thus, in this chapter all the SDN-Controller SBI requirements are defined to collect and export the network topology information.

The set of topology views defined at the controller level are:

- **UNI-Topology:** This layer should contain all the client/user side ports (used and unused) of the network elements.
- **Layer 1 Topology:** This layer contains the physical and virtual devices of the network and the interconnection between the IP and Optical domain.
- **Layer 2 Topology:** This level should represent the Link-Layer connection between network nodes in NNI side. Even though the Layer 2 network topology should be generic and applicable to Layer 2 networks built with different L2 technologies, it will be mainly used to describe both the physical and the logical (virtual) Ethernet network topologies. Accordingly, parameters of this Layer are the following:
 - MAC address information
 - Port Speed/duplex configuration
 - Port administrative state
 - Maximum Transmission Unit (MTU)
 - Link Aggregation (LAG)
 - VLAN ID (single and double tagging)
 - Pseudowires (PW) and Pseudowire terminations
 - LLDP Relationships between nodes.
- **Layer 3 Topology:** The L3 layer represents the Network at IP layer. As in the L2, at L3 the following information can be found:

- Links IP address in both sides
- Router-id
- Link IGP metric
- Maximum Bandwidth / Available Bandwidth
- Latency

Inside this layer several protocols views can be generated. For example, IGP-topology or BGP topology.

In addition to pure network topology, logical topologies related to BGP, Multicast or Service/LSP to be exported to SDN Controller are also covered in this section.

The set of uses cases related to topology support considered on the SBI interface in this deliverable are listed in the following table:

ID	Use case Title	Section
3.1	Obtain and export of end to end ip topology using ip domain controllers (IGP Topology)	6.2
3.2	Obtain and export of L2 topology using ip domain controllers (ethernet links between routers)	6.3
3.3	Export potential service end points in IP topology (UNI Topology)	6.4

6.2 Use case 3.1 Obtain and export of end to end ip topology using IP domain controllers (IGP Topology)

Number	3.1
Name	Obtain and export of end to end ip topology using ip domain controllers (IGP Topology)
Brief description	L3 topology must represent the IP links in the network, including specific parameters to support IP Addressing, Metrics and IGP information. The L3 (nodes and Links) are supported in L2 topology (nodes and links).

Table 17 Definition of Use Case 3.1

6.2.1.1 Use Case Definition

The L3 topology information can be used by different OSS/BSS applications to draw the network topology and relate the network services with the lower layers. Moreover, in the case of the deployment of a central

controller like PCE in the network, the L3 Topology information is fundamental to calculate end-to-end optimal paths. The topology information can be updated dynamically as the network changes occur i.e. a node or link goes down. However, this update process is out of the scope of the current specification.

The L3 topology in this use case must represent the IP links in the network, including specific parameters to support IP Addressing, Metrics and IGP information. The L3 (nodes and Links) are supported by L2 topology (nodes and links).

The L3 Network information MUST be gathered from the network by the SDN domain controller. This section presents the details of how the IP SDN domain controllers MUST get L3 Topology related information from the network devices and be able to build the NBI abstract representation.

6.2.1.2 L3 Network Topology Discovery

There are several options to obtain the end-to-end topology information. One way is by peering passively with an IGP node to get all the link state information, with all its drawbacks:

- A practical PCE or SDN controller code needs to support both OSPF and IS-IS.
- IGP tends to send many updates, so the controller will spend some time processing all those messages.
- In the cases where the network consists of multiple IGP domains across geographic areas then it could be a challenging on where to place the central controller which peers with IGP nodes.

The target and right alternative approach is to use BGP-LS protocol which creates a new NLRI carrying all the IGP info over BGP. In this approach we can leverage a BGP speaker that is already participating in the IGP, thus retrieving info from IGP LSDBs and distributing it to a PCE or SDN controller. The BGP speaker can apply any filter before sending the info northbound to the controller. Accordingly, compared with the IGP peering, the advantages of this approach would be:

- Controller implementation has to only support BGP.
- BGP tends to produce less updates messages.
- In a network with multiple IGP domains, extending peering over BGP is much more feasible compared to IGP.

Therefore, by establishing a BGP-LS peering with one selected router (or more for redundancy) PCE or SDN

controller, the topology information needed can be obtained from the real network and used it for LSP path computation or to visualize the network topology, respectively. This use case will be only focused on how SDN controller can get L3 IP network topology related information to be stored and exported over its NBI to third parties (SDN hierarchical levels or OSS/BSS).

Since IGP consists of topology and IP reachability information and if we want to reconstruct an IGP Topology view at the controller based on the data received over BGP-LS, then BGP-LS must have some way to represent Topology and IP reachability information in its database. For that, BGP-LS specification contains two parts:

- Definition of a new BGP NLRI type which is essentially sets of TLV's that define three objects:
 1. Nodes
 2. Links
 3. IP Prefixes

So, with the combination of Node and Link objects one can construct a topology info and IP Prefix object will provide IP reachability information.

- Definition of a new BGP path attribute (BGP-LS attribute) which is optional Non-transitive attribute. It encodes the properties of the objects (link, node and prefix). For instance, it could be Node-names, IGP metric, TE-metric, latency, Available BW etc.

For this specific use case, the expected L3 network topology information collected by SDN Domain controller through BGP-LS to MUST include:

1. Router-id
2. Link IP addresses in both sides
3. Link IGP metric
4. Maximum Bandwidth / Available Bandwidth
5. Link latency

A link described by the Link descriptor TLVs actually is a “half-link”, a unidirectional representation of a logical link. In order to fully describe a single logical link, two originating routers advertise a half-link each, i.e. two link NLRIs are advertised for a given point-to-point link. As regards IGP further information such as IP prefixes advertised by each node, area distribution and routers roles are not represented in the topology

for the current version of the document (to be discussed).

6.3 Use case 3.2 L2 Topology

This section presents the details of how the Network Controller MUST gather the information from the SBI to export the L2 topology (ethernet links between routers).

6.3.1.1 Use Case Definition

Number	3.2
Name	Obtain and export of L2 topology using IP domain controllers (ethernet links between routers)
Brief description	This topology stored the parameters to support L2 Ethernet network topology representation including nodes, termination points (ports and interfaces) as well as links and their operational state.

Table 18 Definition of Use Case 3.2

6.3.1.2 L2 Network Topology Discovery

The L2 Network information MUST be gathered from the network by the SDN domain controller using LLDP. LLDP is a Layer 2 protocol that allows a network device to advertise its identity and capabilities on the local network providing topological information. The protocol is defined in the IEEE standard 802.1AB.

- NETCONF/Yang using the “OpenConfig-LLDP” data model: Configuration and state variables are defined in this module to report generic protocol information and the per interface neighbors and status. Note that the model allows to modify parameters like chassis-id or system-description.

```

module: openconfig-lldp
  +--rw lldp
    +--rw config
      | +--rw enabled?                boolean
      | +--rw hello-timer?           uint64
      | +--rw suppress-tlv-advertisement* identityref
      | +--rw system-name?           string
      | +--rw system-description?    string
      | +--rw chassis-id?            string
      | +--rw chassis-id-type?       oc-lldp-types:chassis-id-type

```

```

+--ro state
| +--ro enabled?                boolean
| +--ro hello-timer?            uint64
| +--ro suppress-tlv-advertisement* identityref
| +--ro system-name?            string
| +--ro system-description?     string
| +--ro chassis-id?             string
| +--ro chassis-id-type?        oc-lldp-types:chassis-id-type
| +--ro counters
| | ...
+--rw interfaces
+--rw interface* [name]
+--rw name                      -> ../config/name
+--rw config
| ...
+--ro state
| +--ro name?                    oc-if:base-interface-ref
| +--ro enabled?                boolean
| +--ro counters
| ...
+--ro neighbors
+--ro neighbor* [id]
+--ro id                        -> ../state/id
+--ro config
+--ro state
| +--ro system-name?            string
| +--ro system-description?     string
| +--ro chassis-id?             string
| +--ro chassis-id-type?        oc-lldp-types:chassis-id-
type
| +--ro id?                     string
| +--ro age?                    uint64
| +--ro last-update?            int64
| +--ro ttl?                    uint16
| +--ro port-id?                string
| +--ro port-id-type?           oc-lldp-types:port-id-type
| +--ro port-description?       string
| +--ro management-address?     string
| +--ro management-address-type? string
+--ro custom-tlvs
| +--ro tlv* [type oui oui-subtype]
| +--ro type                     -> ../state/type
| +--ro oui                     -> ../state/oui
| +--ro oui-subtype              -> ../state/oui-subtype
| +--ro config
| +--ro state
| +--ro type?                   int32
| +--ro oui?                   string
| +--ro oui-subtype?            string
| +--ro value?                 binary
+--ro capabilities
| ....

```

Figure 7. Openconfing LLDP Parameters

Both models support the usage of basic and custom TLVs (type-length-values). A TLV is the basic information unit of an LLDP-Data Unit (DU). TLVs that can be encapsulated into an LLDPDU include basic TLVs, TLVs defined by IEEE 802.1 and TLVs defined by IEEE 802.3. The table below details each type of TLV where those labeled as non-mandatory could be suppressed, so that only TLVs needed for representing L2 topology will be advertised with the LLDP PDUs.

TLV Category	TLV Name	TLV Type Value	Description	Mandatory
BASIC	End of LLDPDU	0	End of an LLDPDU.	Yes
BASIC	Chassis ID	1	Bridge MAC address of the transmit device.	Yes
BASIC	Port ID	2	Number of a transmit interface of a device.	Yes
BASIC	Time To Live	3	Timeout period of local device information on neighboring devices.	Yes
BASIC	Port Description	4	String describing an Ethernet interface.	No
BASIC	System Name	5	Device name.	No
BASIC	System Description	6	System description.	No
BASIC	System Capabilities	7	Primary functions of the system and whether these primary functions are enabled.	No
BASIC	Management Address	8	Management address.	No
BASIC	Reserved	9-126	Reserved for special use.	No
Defined by IEEE 802.1	Reserved	0	Reserved for special use.	No
Defined by IEEE 802.1	Port VLAN ID	1	VLAN ID on an interface.	No
Defined by IEEE 802.1	Port And Protocol VLAN ID	2	Protocol VLAN ID on an interface.	No
Defined by IEEE 802.1	VLAN Name	3	VLAN name on an interface.	No
Defined by IEEE 802.1	Protocol Identity	4	Protocol type that an interface supports.	No
Defined by IEEE 802.3	Reserved	0	Reserved for special use.	No
Defined by IEEE 802.3	MAC/PHY Configuration/Status	1	Duplex and bit-rate capability, current duplex and bit-rate settings, and auto-negotiation status.	No

Defined by IEEE 802.3	Power Via MDI	2	Power supply capability of an interface, that is, whether an interface supports PoE and whether an interface supplies or requires power.	No
Defined by IEEE 802.3	Link Aggregation	3	Link aggregation status.	No
Defined by IEEE 802.3	Maximum Frame Size	4	Maximum frame length supported by interfaces. The maximum transmission unit (MTU) of an interface is used.	No
Defined by IEEE 802.3	Reserved	5-255	Reserved for special	No

Table 19 LLDP TLVs for Use Case 6.2

NOTE: For security reasons, in order to use LLDP, it has to be enabled only in that interfaces strictly necessary. A generic recommendation is to enable it on NNI interfaces of the Core and Aggregation routers, but never on UNI interfaces where client CPE are connected. Also, LLDP for Media Endpoint Devices (LLDP-MED) must be disabled in all interfaces of the whole network.

As it is depicted in the following example, when the basic and one optional TLVs are enabled between the **NODE A** and **NODE B**:

- System Name (Optional)
- Chassis ID
- Port ID
- TTL

the SDN controller, based on the information above, could find out and represent the different links between the nodes and create the whole L2 topology.

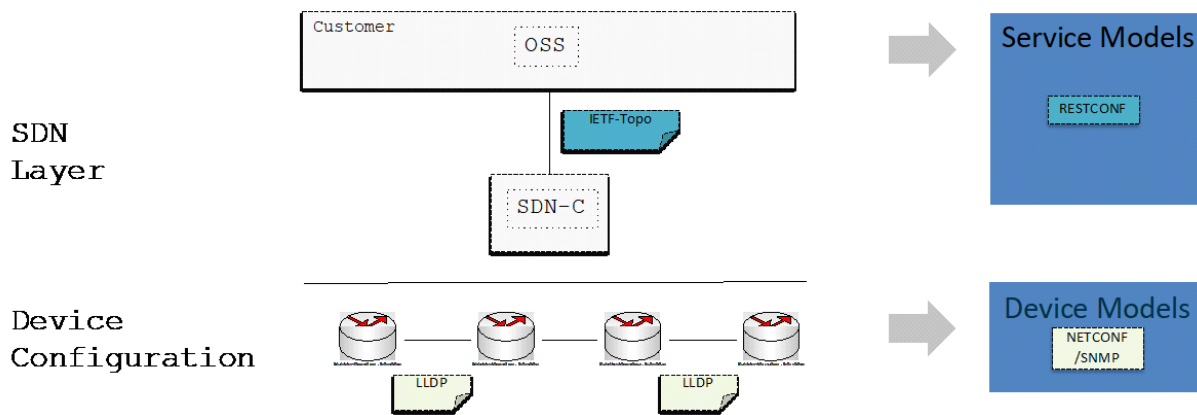


Figure 8. L2/L3 topology collection architecture

Finally, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (all of them related to LLDP use, see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Topology - 1	Retrieve interfaces in a node on which LLDP is enabled	Obtain information about which ports of a node have LLDP protocol enabled
Topology - 2	Retrieve L2 neighbors of a node in all ports	Obtain information about L2 neighbors of a node for all its ports
Topology - 3	Retrieve L2 neighbors of a node in a specific port	Obtain information about L2 neighbors of a node for a specific port
Topology - 4	Enable LLDP in a interface	Activate LLDP protocol in a interface given the name of the interface
Topology - 5	Modify LLDP parameters	Edit global LLDP configuration on a device

Table 20 Atomic Operations for Topology Use case 6.3.2

6.4 Use case 3.3 UNI-Topology

In this section, the details to obtain and export potential end points in the network are the following:

6.4.1.1 Use Case Definition

Number	3.3
Name	Export potential service end points in IP topology (UNI Topology)
Brief description	The UNI topology represents the feasibility topology and has all the potential end point in the network.

Table 21 Definition of Use Case 3.3

6.4.1.2 UNI Topology Discovery

The aim of this use case is to determinate a procedure to obtain those interfaces available in the NE that could be used by the user as UNI ports to be bound to LxVPN services. Such information should be exposed to the SDN Controller.

One way of doing so might be through a netconfig get operation filtering those interfaces that not had mpls activated and export them to SDN controller. It would mean that are susceptible to be utilised as UNI ports.

/network-instances/network-instance/mpls/global/interface-attributes/interface/config/mpls-enabled=FALSE

In any case the vendor is requested to come up with any alternative proposal or enhancement that address the use case objective.

The following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
Topology - 6	Retrieve UNI interfaces of a node	Obtain information about which ports of a node have MPLS enabled and therefore can b considered as UNI ports.

Table 22 Atomic Operations for Use Case 6.3

7

Traffic Engineering Use cases (category 4)

Traffic engineering (TE) allows the enforcement of traffic steering flows by leveraging onto MPLS tunnels or Segment Routing paths ...

7 Traffic Engineering Use Cases (category 4)

7.1 TE and PCE for SBI

Traffic engineering (TE) allows the enforcement of traffic steering flows by leveraging onto MPLS tunnels or Segment Routing paths. This permits to increase the efficiency on the use of the network resources by properly mapping the traffic flows to the available resources, and improve network management, including troubleshooting, to overcome difficult failure situations. Increasingly complex network scenarios such as large single domain environments, multi-domain or multi-layer networks require the usage of algorithms for efficiently computing end-to-end paths.

This complexity is driving the need for a dedicated SDN controller, which will perform path computations and be adaptive to network changes. Currently, only online and real-time constraint-based routing path computation is provided in an MPLS RSVP-TE or Segment Routing networks. Each router performs constraint-based routing calculations independent of the other routers in the network. These calculations are based on currently available topology information, information that is usually recent, but not completely accurate. LSP placements are locally optimized, based on current network status.

On the contrary, a PCE has a global view of the bandwidth demand in the network and maintains a traffic-engineered database to perform path computations. In order to perform the computation of a path for a TE LSP the PCE shall gather the required information (including network topology and resource information) by participating in routing protocols updates as any other NE, usually through BGP-LS as covered in **RFC 8571**. Besides, it even could perform statistics collection from all the routers in the MPLS domain using SNMP, NETCONF or other methods, which also would provide a mechanism for offline control of the TE LSPs.

Technically speaking, a Path Computation Element (PCE) is an entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints. A Path Computation Client (PCC) is any client application requesting a path computation to be performed by a PCE. The Path Computation Element Protocol (PCEP) enables communications between a PCC and a PCE, or between two PCEs, as describe in figure below.

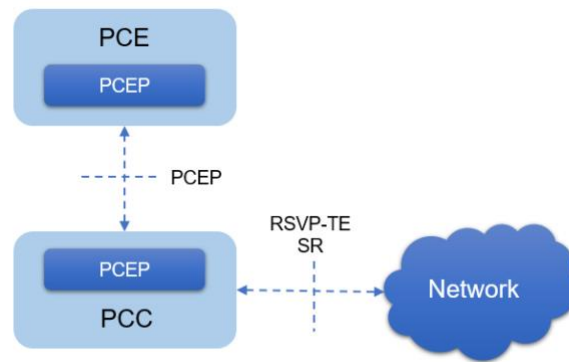


Figure 9. PCEP Architecture

Note: PCE is covered by IETF in different documents, the main are:

- **RFC 4655** “A Path Computation Element (PCE)-Based Architecture”
- **RFC 5440** “Path Computation Element (PCE) Communication Protocol (PCEP)”
- **RFC 8231** “Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE”
- **RFC 8281** “Path Computation Element Communication Protocol (PCEP) Extensions for PCE-Initiated LSP Setup in a Stateful PCE Model”

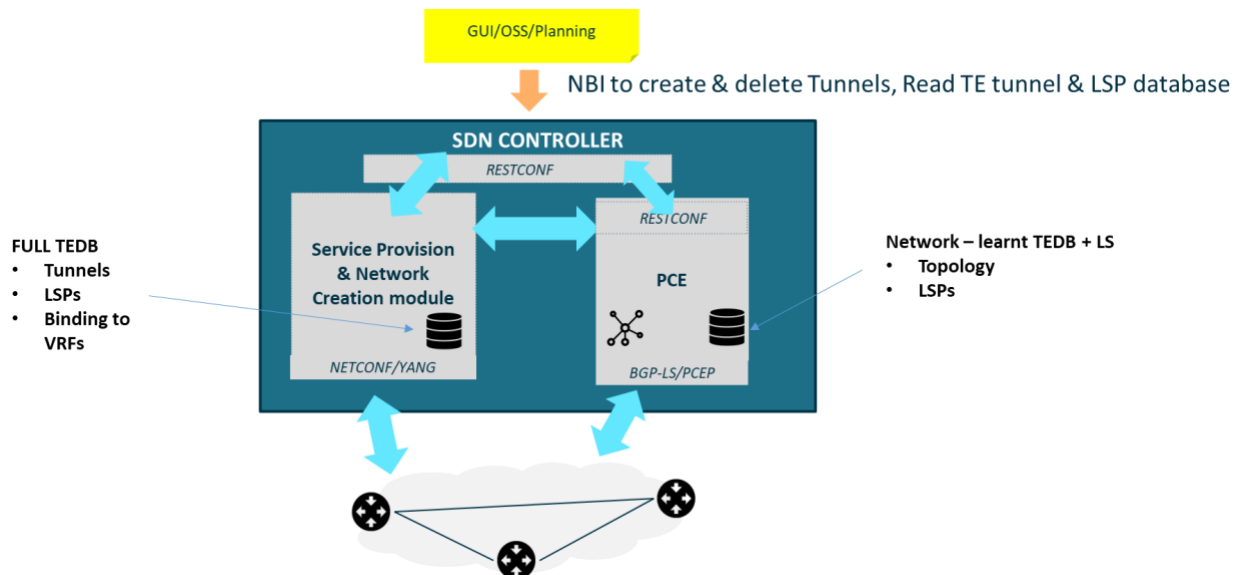


Figure 10. PCE function in the E2E IP SDN Domain Controller



The PCC initiates the PCEP session and stays connected to the PCE for the duration of the PCEP session. During the PCEP session, the PCC requests LSP parameters from the stateful PCE. On receiving one or more LSP parameters from the PCE, the PCC re-signals the TE LSP.

A PCE can either be stateful or stateless. A stateful PCE maintains strict synchronization between the PCE and network states (in terms of topology and resource information), along with the set of computed paths and reserved resources in use in the network. A stateless PCE does not remember any computed path, and each set of requests is processed independently of each other. For this specification only active stateful PCE model are covered, where PCE learns about the PCC LSP states and actively modifies the PCC LSPs.

Thus, these are the functions that a PCC-PCE system using PCEP shall include:

- **LSP state synchronization.** A PCC takes a snapshot of the state of its LSPs and sends a state report containing the snapshot to the PCE. This reporting activity is done for each LSP. PCE would only send a notification back to the PCE if any LSP can not be synchronized. Either the PCE or the PCC may terminate the session using the PCEP.
- **LSP delegation.** After the PCE and the PCC have indicated that they support LSP Update, then the PCC may choose to grant the PCE a temporary right to update attributes from one or more LSPs. PCE can return the LSP delegation at any time if it no longer wishes to update the LSP's state and also, PCE may reject a delegation based on a local policy.
- **LSP operation.** After LSPs have been delegated to the PCE, the PCE can modify LSP parameters of delegated LSPs. To update an LSP, the PCE sends a request to the PCC with the parameters needed to build the LSP. When PCC has built the LSP, PCC sends a report message to the PCE indicating the result of the operation (whether the LSP is build and up or down). Note that PCC will send the operation state report to each stateful PCE that is connected to it.

From an overall NBI architectural point of view, the PCE module is part of the E2E IP MUST Domain controller (see Figure 12).

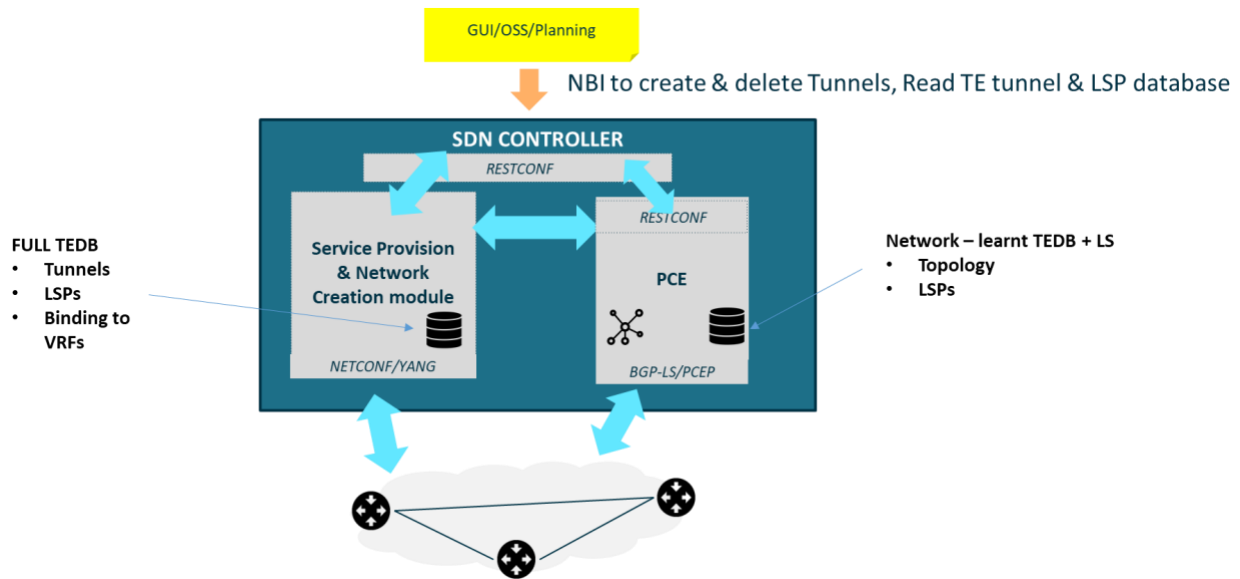


Figure 11. PCE function in the E2E IP MUST SDN Domain Controller

Depending on the entity which initiated the tunnel two scenarios for LSP creation and management shall be considered for this specification.

- **PCC-initiated LSPs.** The PCC delegates the PCC-initiated LSPs to the main PCE for external path computation. The PCC informs the PCE about the configured parameters of a PCE-controlled LSP, such as bandwidth, ERO, and priorities. It also informs the PCE about the actual values used for these parameters to set up the LSP including the RRO, when available.

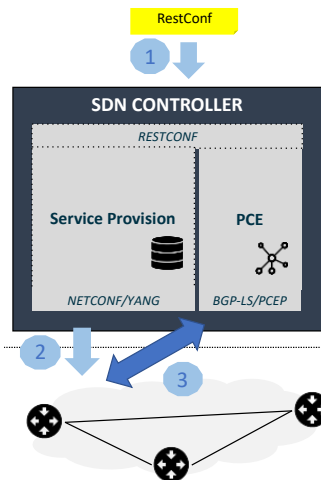


Figure 12. PCC-initiated LSP workflow

- PCE-initiated LSPs.** A PCE-initiated LSP is dynamically created by an external PCE; as a result, there is no LSP configuration present on the PCC. The PCC creates the PCE-initiated LSP using the parameters provided by the PCE, and automatically delegates the LSP to the PCE.

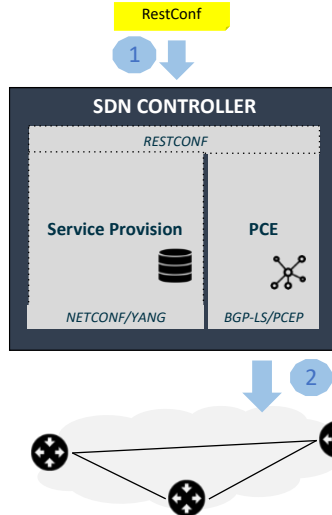


Figure 13. PCE-initiated LSP workflow

7.2 Structure and Classification

Based on previously described functionalities, the main purpose of traffic engineering use cases in the MUST Project is to reduce overall operating costs through more efficient use of network resources, including link occupation, traffic rerouting, network availability, and other components. Essentially, traffic engineering actions address the need to prevent situations where some parts of the network are overloaded, while other parts of the network remain underused and thus ensure the most appropriate path for network traffic and allow the implementation of mechanisms for protecting traffic against network failures.

The set of uses cases related to TE support considered on the SBI interface are listed in the following table.

ID	Use case Title	Section
4.1	LSP Creation, modify and delete with RSVP-TE	7.4.1
4.2	LSP create, modify and delete with constraints (delay, bandwidth and hop count) Signaling: RSVP	7.4.2
4.3	LSP create, modify and delete with constraints (delay, bandwidth and hop count) Signaling: SR	7.4.3
4.4	LSP create, modify and delete with constraints (delay, bandwidth and hop count) Signaling: Explicit Path	7.4.4
4.5	LSP create, modify and delete with constraints (delay, bandwidth and hop count) Protection: Redundancy 1+1	7.4.5

This set of use cases addresses different situations or options for establishing an LSP, from the simplest with no constraint at all, only taking into account routing optimal path or setting the path explicit to the more comprehensive which allow to specify a number of constraints such as delay, bandwidth or hop limit for LSPs, using different types of protection approaches. These constraints will be processed by PCE that will return attributes needed to establish LSP, as explained before. All of this, using RSVP-TE or SR at the signaling plane. Finally, advanced uses cases for LSPs TE optimization and enhancement are covered.

7.3 Openconfig Model for TE

The Openconfig model used for LSP configuration in the SBI will be “openconfig-mpls”. This module is imported into “network-instance” module at the Xpath: /oc-netinst:network-instances/oc-netinst:network-instance/mpls. It's not a standalone root in the hierarchy. Accordingly, each LSP configured will be bounded to the L2VPN or L3VPN instance in which the LSP tunnel has been created. However, it is also possible to create a generic LSP not implicitly associated with a specific network-instance. In order to do that,

Openconfig enables the use of “DEFAULT_INSTANCE”¹ option in order to handle LSPs decoupled from any network instance.

This “openconfig-mpls” module provides data definitions for configuration of Multiprotocol Label Switching (MPLS) and associated protocols for signaling and traffic engineering. It consists of several modules and submodules as shown below. The top-level MPLS module describes the overall framework where the following types of LSPs are supported:

- Traffic-engineered (or constrained-path)
- IGP-congruent (LSPs that follow the IGP path)
- Static LSPs which are not signaled

The structure of each of these LSP configurations is defined in corresponding submodules as depicted below:

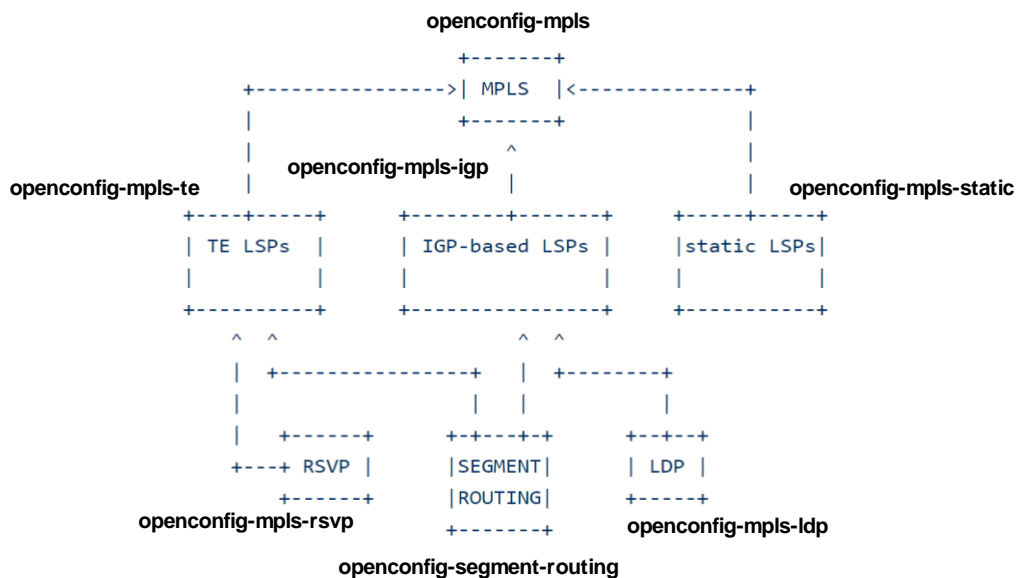


Figure 14. OC Modules for MPLS

¹ The value “DEFAULT_INSTANCE” can be established to the parameter /network-instances/network-instance/config/type

As noted above, for TE LSP creation both RSVP or SR modules can be used, whereas for IGP-based unconstraint LSP, LDP as well as SR modules are considered. The Telefonica Traffic Engineering SBI specification considers TE-LSPs which use either RSVP-TE or SR as the signaling protocol for all the cases described in document (even for the use case of LSPs with no constraints at all or with an explicit path).

7.4 TE Uses Cases

This section presents the details of how to manage LSP establishment and TE information on SBI between NE (routers) and the IP SDN domain controller, which includes a PCE module.

The set of parameters required for the creation of all types of LSP for the use cases explained in this chapter on Netconf SBI are taken from “**openconfig-network-instance**” module that import the mpls module “**openconfig-mpls**” as a container when network-instance type = DEFAULT_INSTANCE (Global Table), as stated in previous section 10.3.

7.4.1 Use case 4.1 LSP Creation, modify and delete with RSVP-TE

7.4.1.1 Use Case Definition

Number	4.1
Name	LSP Creation, modify and delete with RSVP-TE
Brief description	This use case expects the creation of LSPs without any type of constrain calculation or protection. The LSP generally follows the shortest path as dictated by the local routing table, usually taking the same path as destination-based, best-effort traffic. These paths automatically reroute themselves whenever a change or link occurs in a routing table or in the status of a node or link. The LSP won't have any kind of protection either.

Table 23 Definition of Use Case 4.1

7.4.1.2 Description

This use case would be the easiest where there isn't any kind of constraint or requirement to create the LSP, even though RSVP is selected as signaling protocol. Only basic parameters related with source and

destination of the tunnel or priorities shall be taken into account. (Note: This use case don't cover any type of protection for LSP, so only a primary path should be establish).

Depending on the entity which initiated the tunnel two scenarios are considered:

- **4.1a) PCC-initiated LSP**

As of those base requirements the head-end (ingress) router will try to establish the RSVP tunnel to a tail-end (egress) router according to the optimal shortest- path based on IGP (or TE) metric. After the LSP state information is synchronized, the NE acting as PCC delegates the LSP just created to one PCE, which is the main active stateful PCE. At that point, LSP is under external control of the PCE that could ask the PCC could to re-signals the LSP based on a best path it receives from a PCE.

The **xpath** and description of the parameters used are described below.

- 1) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/name` → The tunnel name.
- 2) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/type` → Tunnel type. Options: P2P, P2MP
- 3) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/signaling-protocol` → Signaling protocol used to set up this tunnel. Options: PATH_SETUP_RSVP, PATH_SETUP_SR, PATH_SETUP_LDP.
- 4) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/description` → Optional text description for the tunnel
- 5) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/admin-status` → TE tunnel administrative state (UP or DOWN)
- 6) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/preference` → Specifies a preference for this tunnel. A lower number signifies a better preference
- 7) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/metric-type` → The type of metric specification that should be used to set the LSP(s) metric. Options: LSP_METRIC_RELATIVE, LSP_METRIC_ABSOLUTE, LSP_METRIC_INHERITED
- 8) `/network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/metric` → The value of the metric that should be specified.

- 9) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/protection-style-requested* → Style of mpls fr protection desired: can be link, link-node or unprotected. Options: =UNPROTECTED, LINK_PROTECTION_REQUIRED, LINK_NODE_PROTECTION_REQUESTED
- 10) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/reoptimize-timer* → Frequency of reoptimization of a traffic engineered LSP
- 11) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/soft-preemption* → Enables RSVP soft-preemption on this LSP
- 12) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/setup-priority* → RSVP-TE preemption priority during LSP setup, lower is higher priority; default 7 indicates that LSP will not preempt established LSPs during setup
- 13) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/hold-priority* → preemption priority once the LSP is established, lower is higher priority; default 0 indicates other LSPs will not preempt the LSPs once established
- 14) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/config/source* → RSVP-TE tunnel source address
- 15) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/config/destination* → P2P tunnel destination address

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(1) name	string	-	controller	rw
(2) type	identityref	-	controller	rw
(3) signaling-protocol	identityref	-	controller	rw
(4) description	string	-	controller	rw
(5) admin-status	identityref	-	controller	rw
(6) preference	uint8	-	controller	rw
(7) metric-type	identityref	-	controller	rw
(8) metric	int32	-	controller	rw
(9) protection-style-requested	identityref	-	controller	rw
(10) reoptimize-timer	uint16	-	controller	rw
(11) soft-preemption	boolean	FALSE	controller	rw
(12) setup-priority	uint8	-	controller	rw
(13) hold-priority	uint8	-	controller	rw
(14) source	inet:ip-address	-	controller	rw
(15) destination	inet:ip-address	-	controller	rw

The following table contains example values for each one of the parameters above described:

Parameter	Value (example)
(1) name	"TUNNEL1001"
(2) type	P2P
(3) signaling-protocol	PATH_SETUP_RSVP
(4) description	"TUNNEL_A_TO_B"
(5) admin-status	ADMIN_UP
(6) preference	1
(7) metric-type	LSP_METRIC_INHERITED
(8) metric	10
(9) protection-style-requested	UNPROTECTED
(10) reoptimize-timer	300
(11) soft-preemption	FALSE
(12) setup-priority	7
(13) hold-priority	0
(14) source	172.16.100.1
(15) destination	172.16.200.1

Traffic Engineering Use case 4.1 Atomic operations

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see "**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**"). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
TE -2	Create RSVP or SR signaled LSP without constraint	Creation of a RSVP or SR LSP without any type of constrain calculation or protection, only basic parameters like source and destination
TE -8	Enable LSP PCE Delegation	Delegate the path computation responsibility for a LSP to the PCE
TE -10	Delete RSVP or SR signaled LSP	Deletion of a RSVP or SR LSP without any type of constrain calculation or protection
TE-13	Disable PCE Delegation for a LSP	Un-delegate the path computation responsibility for a LSP to the PCE

TE-14	Retrieve state of an existing LSP	Show information of a LSP to know if the tunnel was successfully created by checking the "operational-state"
TE-15	Modify basic parameters of an existing LSP	Change the description or administrative status of a LSP

Table 24 Atomic Operations for Traffic Engineering (1)

- **4.1.b) PCE-initiated LSP**

This scenario would be equivalent to PCC-initiated where only a Explicit Route Object (ERO) object has to be sent from the PCE to the PCC. The PCC creates the LSP using the parameters provided by the PCE, assigns the PCE-initiated LSP a unique LSP identifier, and automatically delegates the LSP to the PCE. A PCE-initiated LSP is dynamically created by an external PCE, as a result there is no LSP configuration present on the PCC. For this reason, there is no need of Openconfig parametrization in NE either.

A PCC cannot revoke the delegation for the PCE-initiated LSPs for an active PCEP session. When a PCEP session terminates due to a PCE failure, the PCC shouldn't immediately delete the PCE-initiated LSPs, but wait the applicable timers to expire (Redelegation Timeout Interval and State Timeout Interval, see RFC8281) in order to avoid services disruption. Thus, only when the second timer expires and no other PCE has acquired control over the LSPs from the failed PCE, the PCC deletes all the LSPs provisioned by the failed PCE. PCE is designed to work in high availability in order to avoid this scenario to ever happen.

7.4.2 Use case 4.2 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) – SIGNALLING: SR

7.4.2.1 Use Case Definition

Number	4.2
Name	LSP Creation, modify and delete with SR no constraints
Brief description	In this use case the LSP is establish based on constraint-based routing performed by the PCC (router/NE) or a external PCE. The LSP will use SEGMENT ROUTING as signaling protocol and won't have any kind of protection.

Table 25 Definition of Use Case 4.2

7.4.2.2 Description

For this use case would apply all the considerations mentioned for previous use case, but using segment routing as signaling-protocol in MPLS yang model node as indicated in the below table (the rest of the parameters would be equivalent to the previous use case and they are not repeated here)

Parameter	Value (example)
(1) name	"TUNNEL1001"
(2) type	P2P
(3) signaling-protocol	PATH_SETUP_SR
...	...

Likewise, the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case are equivalent to ones described for the previous use case..

7.4.3 Use case 4.3 - LSP create, modify and delete with constraints (delay, bandwidth and hop count)

7.4.3.1 Use Case Definition

Number	4.3
Name	LSP create, modify and delete with constraints (delay, bandwidth and hop count)
Brief description	In this use case the LSP is establish based on constraint-based routing performed by the PCC (router/NE) or a external PCE. The result of Constrained Shortest Path First (CSPF) algorithm will be the path that meets the constraints or the set (combination) of constraints considered, in this case delay, BW and hop limit. The LSP will use RSVP or SR as signalling protocol and won't have any kind of protection.

Table 26 Definition of Use Case 4.3

7.4.3.2 Description

The path computation takes into account information provided by the topology information from link-state routing protocol, the current network resource utilization determined by RSVP, and the resource requirements and constraints of the LSP. The output of the CSPF (Constrained Shortest Path First) calculation is an explicit route consisting of a sequence of router addresses that provides the shortest path through the network that meets the constraints. For signaled constrained-path LSPs to work, either IS-IS or

OSPF protocols and IS-IS or OSPF traffic engineering extensions must be enabled on all routers.

In this particular use case, the constraints to be taken into account for CSPF path computation process would be **delay**, **BW** and/or **hop limit**. (Note: This use case doesn't cover any type of protection for LSP, so only a primary path should be established).

Depending on the entity which initiated the tunnel two scenarios are considered.

- **4.3a) PCC-initiated LSP**

As of defined constraints, the head-end (ingress) router will try to establish the RSVP-TE tunnel to a tail-end (egress) router according to the path derived from Constrained Shortest Path First (CSPF) computation. After the LSP state information is synchronized, the NE acting as PCC delegates the LSP just created to one PCE, which is the main active stateful PCE. At that point, LSP is under external control of the PCE that could ask the PCC to re-signal the LSP based on a best path it receives from a PCE.

The set of additional parameters to be considered for a constrained LSP configuration in accordance to required values for BW, delay or hop-count taken as constraints

The **xpath** and description of the parameters to be considered for a constrained LSP configuration in accordance to required values for BW, delay or hop-count taken as constraints, are additional to general for tunnel definition as described in previous section.

NOTE: It has been identified an issue with Openconfig for MPLS. The model does permit to specify Bandwidth as a constraint for the path, but NOT delay or hop-limit attributes as constraints. This issue has been raised to Openconfig team. It is expected to be clarified the forthcoming version of the specification.

16) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/name* → Path name

17) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/path-computation-method* → The method used for computing the path, either locally computed, queried from a server or not computed at all (explicitly configured). Options: `LOCALLY_COMPUTED`, `EXTERNALLY_QUERIED`, `EXPLICITLY_DEFINED`

18) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/use-cspf* → Flag to enable CSPF for locally computed LSPs

- 19) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/cspf-tiebreaker* → The cspf tiebreaking method when the path is locally computed. Options: RANDOM, LEAST FILL, MOST FILL
- 20) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/path-computation-server* → Address of the external path computation server
- 21) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/explicit-path-name* → Reference to a defined path
- 22) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/preference* → Specifies a preference for this path. The lower the number higher the preference
- 23) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/setup-priority* → RSVP-TE preemption priority during LSP setup, lower is higher priority; default 7 indicates that LSP will not preempt established LSPs during setup
- 24) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/hold-priority* → preemption priority once the LSP is established, lower is higher priority; default 0 indicates other LSPs will not preempt the LSPs once established
- 25) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/config/retry-timer* → Sets the time between attempts to establish the LSP
- 26) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/bandwidth/config/specification-type* → The method used for setting the bandwidth, either explicitly specified or configured. Options: SPECIFIED, AUTO
- 27) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/bandwidth/config/set-bandwidth* → The bandwidth value when bandwidth is explicitly specified
- 28) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/autobandwidth/config/enabled* → enables mpls auto-bandwidth on the lsp
- 29) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/autobandwidth/config/min-bw* → Set the minimum bandwidth in Kbps for an auto-bandwidth LSP

- 30) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/autobandwidth/config/max-bw* → Set the maximum bandwidth in Kbps for an auto-bandwidth LSP
- 31) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/autobandwidth/config/adjust-interval* → Time in seconds between adjustments to LSP bandwidth
- 32) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/autobandwidth/config/adjust-threshold* → Percentage difference between the LSP' specified bandwidth and its current bandwidth allocation -- if the difference is greater than the specified percentage, auto-bandwidth adjustment is triggered

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(16) name	string	-	controller	rw
(17) path-computation-method	identityref	-	controller	rw
(18) use-cspf	boolean	-	controller	rw
(19) cspf-tiebreaker	typedef	-	controller	rw
(20) path-computation-server		-	controller	rw
(21) explicit-path-name	(Leafref) ../..../named-explicit-paths/named-explicit-path/config/name	-	controller	rw
(22) preference	inet:ip-address	-	controller	rw
(23) setup-priority	uint8	-	controller	rw
(24) hold-priority	uint8	-	controller	rw
(25) retry-timer	uint8	-	controller	rw
(26) specification-type	typedef	-	controller	rw
(27) set-bandwidth	oc-mplst:bandwidth-kbps	-	controller	rw
(28) enabled	boolean	FALSE	controller	rw
(29) min-bw	uint64	-	controller	rw
(30) max-bw	uint64	-	controller	rw
(31) adjust-interval	uint32	-	controller	rw
(32) adjust-threshold	uint8	-		

The following table contains example values for each one of the parameters above described:

Parameter	Value (example)
(1) name	"TUNNEL1001"
(2) type	P2P
(3) signaling-protocol	PATH_SETUP_RSVP

(4) description	"TUNNEL_A_TO_B"
(5) admin-status	ADMIN_UP
(6) preference	1
(7) metric-type	LSP_METRIC_INHERITED
(8) metric	10
(9) protection-style-requested	UNPROTECTED
(10) reoptimize-timer	300
(11) soft-preemption	FALSE
(12) setup-priority	7
(13) hold-priority	0
(14) source	172.16.100.1
(15) destination	172.16.200.1
(16) name	"LSP_BW_WITH_CONSTRAIN"
(17) path-computation-method	EXTERNALLY_QUERIED
(18) use-cspf	FALSE
(19) cspf-tiebreaker	LEAST_FILL
(20) path-computation-server	100.100.100.1
(21) explicit-path-name	-
(22) preference	1
(23) setup-priority	7
(24) hold-priority	0
(25) retry-timer	60
(26) specification-type	SPECIFIED
(27) set-bandwidth	5000
(28) enabled	FALSE
(29) min-bw	-
(30) max-bw	-
(31) adjust-interval	-
(32) adjust-threshold	-

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see "**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**"). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
TE -3	Create RSVP or SR signaled LSP with constraint (BW)	Creation of RSVP or SR LSP taken into account for CSPF path computation process a BW constraint
TE -4	Create RSVP or SR signaled LSP with constraint (Delay)	Creation of RSVP or SR LSP taken into account for CSPF path computation process a Delay constrain NOT SUPPORTED BY OPENCONFIG MODELS YET.
TE -5	Create RSVP or SR signaled LSP with	Creation of RSVP or SR LSP taken into account

	constraint (Hop-Limit)	for CSPF path computation process a Hop Limit constraint ❌ NOT SUPPORTED BY OPENCONFIG MODELS YET.
TE -8	Enable LSP PCE Delegation	Delegate the path computation responsibility for a LSP to the PCE
TE -10	Delete RSVP or SR signaled LSP	Deletion of a RSVP or SR LSP without any type of constrain calculation or protection
TE -13	Disable LSP PCE Delegation	Un-delegate the path computation responsibility for a LSP to the PCE
TE-14	Retrieve state of an existing LSP	Show information of a LSP to know if the tunnel was successfully created by checking the “operational-state”
TE-15	Modify basic parameters of an existing LSP	Change the description or administrative status of a LSP
TE-16	Modify constraints of an existing RSVP or SR signaled LSP	Change the value of a constraint or add a new one so that LSP re-signal with the new configuration (at the moment only apply to BW)

Table 27 Atomic Operations for Traffic Engineering (2)

- **4.3b) PCE-initiated LSP**

The same considerations as in use case 4.1 applies. As the path is PCE initiated, there is no need to send the constraints to the node, just the ERO. It is the PCE that makes all calculations using constraints received via any app (northbound or operator demand),. The path must be continuously updated, as in active stateful mode.

7.4.4 Use case 4.4 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) and explicit Path (strict and loose)

7.4.4.1 Use Case Definition

Number	4.4
Name	LSP create, modify and delete with constraints (delay, bandwidth and hop count) and explicit Path (strict and loose)
Short description	An explicit-path (either SR or RVSP-TE) is triggered from an operator action. Once the operator triggers the creation of the path, it is initiated using the explicitly specified path which comes from a planning process. If the path is topologically not feasible, either because the network is partitioned or insufficient resources are available along some parts of the path, the LSP creation will fail. No alternative paths can be used. If the setup succeeds, the LSP stays on the defined path indefinitely. The initiated path in this use case does not consider protection. The explicit path can contain strict or loose hops.

Table 28 Definition of Use Case 4.4

7.4.4.2 Description

In the case of explicit-path LSPs, all intermediate hops of the tunnel are pre-computed (operator, planning tool...) without neither the PCE nor the PCC performs a constrained path calculation. The intermediate hops can be strict, loose, or any combination of the two. When a strict hop is configured, it identifies an exact path through which the LSP must be routed. (Note: This use case does not cover any type of protection for LSP, so only a primary path should be established).

Depending on the entity which initiated the tunnel two scenarios are considered.

- **4.4a) PCC-initiated LSP**

An explicit-path LSP is initiated by NE only along the explicitly specified path. After the LSP state information is synchronized, the NE acting as PCC delegates the LSP just created to the main active stateful PCE.

In the table below are detailed **an example** of how it would be a LSP configuration using **Explicit Route** with 3 strict hops (Note: Using Openconfig mpls module “openconfig-mpls” as a container within module “network-instance” for type “DEFAULT_INSTANCE, as stated in previous section).

The **xpath** and description of the parameters to be considered for a configuration using Explicit Route with 3 strict hops are additional to general for tunnel definition as described in previous section.

33) */network-instances/network-instance/mpls/lsp/constrained-path/named-explicit-paths/named-explicit-path/config/name* → A string name that uniquely identifies an explicit path

34) */network-instances/network-instance/mpls/lsp/constrained-path/named-explicit-paths/named-explicit-path/explicit-route-objects/explicit-route-object/config/address* → Router hop for the LSP path

35) */network-instances/network-instance/mpls/lsp/constrained-path/named-explicit-paths/named-explicit-path/explicit-route-objects/explicit-route-object/config/hop-type* → Strict or loose hop

36) */network-instances/network-instance/mpls/lsp/constrained-path/named-explicit-paths/named-explicit-path/explicit-route-objects/explicit-route-object/config/index* → Index of this explicit route object to express the order of hops in the path

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(33) name	string	-	controller	rw
(34) address	union	-	controller	rw
(35) hop-type	enumeration	-	controller	rw
(36) index	uint8	-	controller	rw

The following table contains example values for each one of the parameters above described:

The following table contains example values for each one of the parameters above described:

Parameter	Value (example)
(1) name	"TUNNEL1001"
(2) type	P2P
(3) signaling-protocol	PATH_SETUP_RSVP
(4) description	"TUNNEL_A_TO_B"
(5) admin-status	ADMIN_UP
(6) preference	1
(7) metric-type	LSP_METRIC_INHERITED
(8) metric	10
(9) protection-style-requested	UNPROTECTED
(10) reoptimize-timer	300
(11) soft-preemption	FALSE
(12) setup-priority	7
(13) hold-priority	0
(14) source	172.16.100.1
(15) destination	172.16.200.1
(16) name	"LSP_EXPLICIT_PATH"
(17) path-computation-method	EXPLICITLY_DEFINED
(18) use-cspf	FALSE
(19) cspf-tiebreaker	LEAST_FILL
(20) path-computation-server	N/A (LSP EXPLICIT NO COMPUTATION)
(21) explicit-path-name	"EXPLICIT_PATH_1_2_3"

(22) preference	1
(23) setup-priority	7
(24) hold-priority	0
(25) retry-timer	60
(26) specification-type	SPECIFIED
(27) set-bandwidth	5000
(28) enabled	FALSE
(29) min-bw	-
(30) max-bw	-
(31) adjust-interval	-
(32) adjust-threshold	-
(33) name	"EXPLICIT_PATH_1_2_3"
(34) address	172.16.101.1
(35) hop-type	STRICT
(36) index	1
(34) address	172.16.102.2
(35) hop-type	STRICT
(36) index	2
(34) address	172.16.103.3
(35) hop-type	STRICT
(36) index	3

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
TE -1	Create Explicit-Path LSP	Creation of a LSP where all intermediate hops are manually configured
TE -9	Delete Explicit-Path LSP	Deletion of a LSP where all intermediate hops are manually configured
TE -8	Enable LSP PCE Delegation	Delegate the path computation responsibility for a LSP to the PCE
TE-13	Disable PCE Delegation for a LSP	Un-delegate the path computation responsibility for a LSP to the PCE

TE-14	Retrieve state of an existing LSP	Show information of a LSP to know if the tunnel was successfully created by checking the "operational-state"
TE-15	Modify basic parameters of an existing LSP	Change the description or administrative status of a LSP
TE-17	Modify an existing Explicit-Path LSP hops	Change, add or erase intermediate nodes to a Explicit-Path LSP
TE-19	Retrieve hops of an existing Explicit-Path LSP	Show information of intermediate nodes used by a Explicit-Path LSP

Table 29 Atomic Operations for Traffic Engineering (3)

- **4.4b) PCE-initiated LSP**

The same considerations as in use case 4.1 applies. As the path is PCE initiated, the explicit path is sent via PCEP protocol.

7.4.5 Use case 4.5 - LSP create, modify and delete with constraints (delay, bandwidth and hop count) – PROTECTION: Redundancy 1+1

7.4.5.1 Use Case Definition

Number	4.5
Name	LSP create, modify and delete with constraints (delay, bandwidth and hop count) – PROTECTION: Redundancy 1+1
Short description	This use case refers to 1+1 protection where there is one working LSP and one protection LSP

Table 30 Definition of Use Case 4.5

7.4.5.2 Description

This use case refers to 1+1 protection where there is one working LSP and one protection LSP. Based on this, a pre-signaled protecting LSP over dedicated resources is deployed at the very same time with the working LSP. At the ingress node, in normal situation, the traffic is sent through working LSP and in case of a failure the two nodes R5 and R1 (egress/ingress) should be coordinated to switch the traffic from the primary to the protection LSP.

Translated to the SBI, this scenario would match with a MPLS Path Protection scheme as depicted in the **Figure 15**. LSP network diagram with 1+1 redundancy:

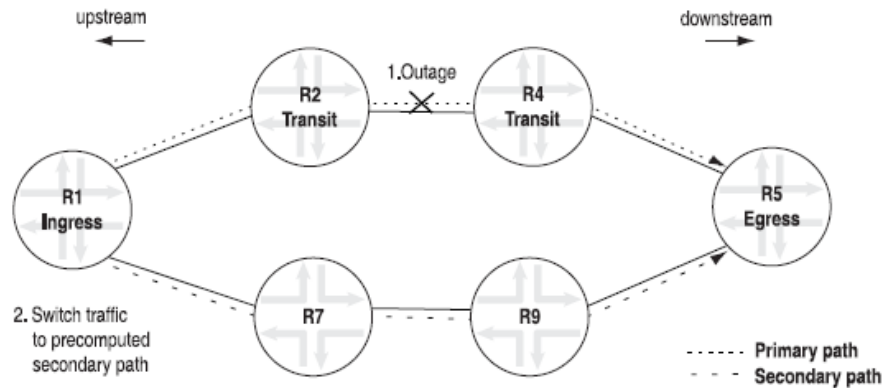


Figure 15. LSP network diagram with 1+1 redundancy

Depending on the entity which initiated the tunnel two scenarios are considered.

- **4.5a) PCC-initiated LSP**

Based on the protection required for the use case, the table below describes **an example** of what parameters or attributes should be considered for a constrained **LSP with Path Protection** with two LSP, Primary and Secondary working together in a hot standby behavior, where the secondary path remains up indefinitely to provide instant switchover if connectivity problems in Primary path occur. The protection path can be disjoint and use Node, Link or SRLG restrictions to deploy the working and protecting LSPs. (Note: Using Openconfig mpls module “openconfig-mpls” as a container within module “network-instance” for type “DEFAULT_INSTANCE, as stated in previous section 9.3).

As can be observed, within Openconfig mpls model the tunnel configuration container includes a node called “protection-style-requested” that support 3 options:

- UNPROTECTED
- LINK_PROTECTION_REQUIRED
- LINK_NODE_PROTECTION_REQUESTED

where the two last ones are used for FRR (Fast Reroute) protection model. For this particular use case we

can select either of them so that we could complement Path Protection end-to-end (Primary and Secondary LSPs) with FRR enabled (LINK_PROTECTION_REQUIRED or LINK_NODE_PROTECTION_REQUESTED) or not enabled (UNPROTECTED).

The **xpath** and description of the parameters to be considered for a configuration with 1+1 LSP redundancy includes specific additional p2p secondary path configuration, additional to general for tunnel and primary LSP definition as described in previous section.

- 37) *network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/name* → Path name
- 38) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/path-computation-method* → The method used for computing the path, either locally computed, queried from a server or not computed at all (explicitly configured). Options: `LOCALLY_COMPUTED`, `EXTERNALLY_QUERIED`, `EXPLICITLY_DEFINED`
- 39) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/use-cspf* → Flag to enable CSPF for locally computed LSPs
- 40) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/cspf-tiebreaker* → The cspf tiebreaking method when the path is locally computed. Options: `RANDOM`, `LEAST_FILL`, `MOST_FILL`
- 41) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/path-computation-server* → Address of the external path computation server
- 42) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/explicit-path-name* → Reference to a defined path
- 43) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/preference* → Specifies a preference for this path. The lower the number higher the preference
- 44) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/setup-priority* → RSVP-TE preemption priority during LSP setup, lower is higher priority; default 7 indicates that LSP will not preempt established LSPs during setup
- 45) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/hold-priority* → preemption priority once

the LSP is established, lower is higher priority; default 0 indicates other LSPs will not preempt the LSPs once established

- 46) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-secondary-path/p2p-primary-path/config/retry-timer* → Sets the time between attempts to establish the LSP
- 47) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/candidate-secondary-paths/candidate-secondary-path/config/secondary-path* → A reference to the secondary path that should be utilized when the containing primary path option is in use
- 48) */network-instances/network-instance/mpls/lsp/constrained-path/tunnels/tunnel/p2p-tunnel-attributes/p2p-primary-path/p2p-primary-path/candidate-secondary-paths/candidate-secondary-path/config/priority* → The priority of the specified secondary path option. Higher priority options are less preferable - such that a secondary path reference with a priority of 0 is the most preferred

In the following table are detailed the features of each parameter:

Parameter	Yang type	Default value	Provided by	Mode
(37) name	string	-	controller	rw
(38) path-computation-method	identityref	-	controller	rw
(39) use-cspf	boolean	-	controller	rw
(40) cspf-tiebreaker	typedef	-	controller	rw
(41) path-computation-server	union	-	controller	rw
(42) explicit-path-name	(Leafref) ../..../named-explicit-paths/named-explicit-path/config/name	-	controller	rw
(43) preference	inet:ip-address	-	controller	rw
(44) setup-priority	uint8	-	controller	rw
(45) hold-priority	uint8	-	controller	rw
(46) retry-timer	uint8	-	controller	rw
(47) secondary-path	(Leafref) ../..../p2p-secondary-paths/p2p-secondary-path/config/name	-	controller	rw
(48) priority	uint16	-	controller	rw

The following table contains example values for each one of the parameters above described:

Parameter	Value (example)
(1) name	"TUNNEL1001"
(2) type	P2P
(3) signaling-protocol	PATH_SETUP_RSVP
(4) description	"TUNNEL_A_TO_B"
(5) admin-status	ADMIN_UP
(6) preference	1
(7) metric-type	LSP_METRIC_INHERITED
(8) metric	10
(9) protection-style-requested	UNPROTECTED
(10) reoptimize-timer	300
(11) soft-preemption	FALSE
(12) setup-priority	7
(13) hold-priority	0
(14) source	172.16.100.1
(15) destination	172.16.200.1
(16) name	"LSP_PRIMARY"
(17) path-computation-method	EXTERNALLY_QUERIED
(18) use-cspf	FALSE
(19) cspf-tiebreaker	LEAST_FILL
(20) path-computation-server	100.100.100.1
(21) explicit-path-name	-
(22) preference	1
(23) setup-priority	7
(24) hold-priority	0
(25) retry-timer	60
(37) name	"LSP_SECONDARY"
(38) path-computation-method	EXTERNALLY_QUERIED
(39) use-cspf	FALSE
(40) cspf-tiebreaker	LEAST_FILL
(41) path-computation-server	100.100.100.1
(42) explicit-	-

path-name	
(43) preference	2
(44) setup-priority	7
(45) hold-priority	0
(46) retry-timer	60
(47) secondary-path	"LSP_SECONDARY"
(48) priority	0

On the other hand, the following table represents the set of NETCONF operations that must be supported by the SBI of the SDN controller for implementing this use case (see “**Annex 1: MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations**”). A deeper level detail of the operations (for further study) would explain how to execute each specific function (RPC calls and XML). In the complementary annex sample RPC calls are documented.

ID	Operation	Description
TE -6	Create LSP with Path Protection (Primary Path + Secondary Path)	Creation of path protection where there is one working LSP and one protection LSP.
TE -7	Create LSP with Node or Link Protection (FRR)	Creation of FRR node or link protection for a LSP
TE -8	Enable LSP PCE Delegation	Delegate the path computation responsibility for a LSP to the PCE
TE -11	Delete LSP with Path Protection	Deletion of path protection where there is one working LSP and one protection LSP
TE -12	Disable Node or Link Protection (FRR)	Deletion of FRR node or link protection for a LSP
TE -13	Disable LSP PCE Delegation	Un-delegate the path computation responsibility for a LSP to the PCE
TE-14	Retrieve state of an existing LSP	Show information of a LSP to know if the tunnel was successfully created by checking the “operational-state”
TE-15	Modify basic parameters of an existing LSP	Change the description or administrative status of a LSP
TE-18	Retrieve Secondary Path of a Primary Path for an existing LSP	Show the Secondary Path (protection) associated to a Primary Path (working) LSP

Table 31 Atomic Operations for Traffic Engineering (4)

• 4.5b) PCE-initiated LSP

In this case, the protection LSP can be also initiated by a stateful PCE, which retains the control of the LSP. The PCE is responsible for computing the path of the LSP and updating to the PCC with the information about the path

8 References

- [1] Telecom Infra Project, "Open Transport SDN Architecture Whitepaper", available online at https://cdn.brandfolder.io/D8DI15S7/at/jh6nnbb6bjvn7w7t5jbgm5n/OpenTransportArchitecture-Whitepaper_TIP_Final.pdf last seen on 01-February-2021.
- [2] "Network Configuration Protocol (NETCONF)". RFC 6241
- [3] "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP". RFC 7752
- [4] "Path Computation Element (PCE) Communication Protocol (PCEP)". RFC 5440
- [5] "draft-openconfig-rtgwg-gnmi-spec - gRPC Network Management Interface (gNMI)
- [6] BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions". RFC 8571
- [7] Openconfig, Vendor-neutral, model-driven network management designed by users, <https://openconfig.net/>
- [8] MUST_IP_SBI_Deliverable_D.1.1_Annex_2_Use_Cases_vs_OC_Model_Params.xlsx
- [9] MUST_IP_SBI_Deliverable_D1.1_Annex_1_Atomic_Operations.docx

9 Glossary

ACL	Access List
BGP	Border Gateway Protocol
BGP-LS	Border Gateway Protocol Link State
BSS	Business Support System
BW	Bandwidth
CE	Customer Edge
CIR	Committed Information Rate
CoS	Class of Service
CPE	Customer Premises Equipment
CSPF	Constrained Shortest Path First
DSCP	Differentiated Service CodePoint
ECN	Explicit Congestion Notification
ERO	Explicit Route Object
GRE	Generic Routing Encapsulation
gRPC	Remote Procedure Call developed by Google.
IANA	Internet Assigned Numbers Authority
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISIS	Intermediate System - Intermediate System
LACP	Link Aggregation Control Protocol
LAG	Link AGgregation
LLDP	Link Layer Discovery Protocol
LSP	Link State Path
MDI	Media Dependent Interface
MED	Multi Exit Discriminator
MIB	Management Information Base
MPLS	Multiprotocol Label Switching
MTU	Maximum Transmission Unit
MUST OOPT	Mandatory Use Cases for SDN for Transport / Open Optical & Packet Transport
NBI	Northbound Interface

NE	Network Element
NETCONF	Network Configuration
NLRI	Network Layer Reachability Information
OC	OpenConfig
OSPF	Open Shortest Path First
OSS	Operations Support System
PCC	Path Computational Client
PCE	Path Computational Element
PCEP	Path Computation Element Protocol
PDU	Protocol Data Unit
PE	Provider Edge
PIR	Peak Information Rate
QoS	Quality of Service
RD/RT	Route Distinguisher / Route Target
RED	Random Early Detection
RFC	Request For Comments
RPC	Remote Procedure Call
RRO	Record Route Object
RSVP	Resource Reservation Protocol
SBI	Southbound Interface
SDN	Software Defined Networks
SLA	Service Level Agreement
SR	Segment Routing
TE	Traffic Engineering
TED	Traffic Engineering Database
TLV	Type Length Value
TTL	Time To Live
UNI	User-to-Network Interface
VLL	Virtual Leased Line
VPN	Virtual Private Network
VSI	Virtual Switch Interface
VXLAN	Virtual Extensible Local Area Network
WFQ	Weighted Fair Queuing
WRED	Weighted Random Early Detection
XML	Extensible Markup Language
YANG	Yet Another Next Generation



10 TIP Document License

By using and/or copying this document, or the TIP document from which this statement is linked, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions:

Permission to copy, display and distribute the contents of this document, or the TIP document from which this statement is linked, in any medium for any purpose and without fee or royalty is hereby granted under the copyrights of TIP and its Contributors, provided that you include the following on ALL copies of the document, or portions thereof, that you use:

1. A link or URL to the original TIP document.
2. The pre-existing copyright notice of the original author, or if it doesn't exist, a notice (hypertext is preferred, but a textual representation is permitted) of the form: "Copyright 2019, TIP and its Contributors. All rights Reserved"
3. When space permits, inclusion of the full text of this License should be provided. We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

No right to create modifications or derivatives of TIP documents is granted pursuant to this License. except as follows: To facilitate implementation of software or specifications that may be the subject of this document, anyone may prepare and distribute derivative works and portions of this document in such implementations, in supporting materials accompanying the implementations, PROVIDED that all such materials include the copyright notice above and this License. HOWEVER, the publication of derivative works of this document for any other purpose is expressly prohibited.

For the avoidance of doubt, Software and Specifications, as those terms are defined in TIP's Organizational Documents (which may be accessed at <https://telecominfraproject.com/organizational-documents/>), and components thereof incorporated into

the Document are licensed in accordance with the applicable Organizational Document(s).


Disclaimers

THIS DOCUMENT IS PROVIDED "AS IS," AND TIP MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

TIP WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name or trademarks of TIP may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with TIP and its Contributors.

This TIP Document License is based, with permission from the W3C, on the W3C Document License which may be found at <https://www.w3.org/Consortium/Legal/2015/doc-license.html>.



Copyright © 2020 Telecom Infra Project, Inc. A TIP Participant, as that term is defined in TIP's Bylaws, may make copies, distribute, display or publish this Specification solely as needed for the Participant to produce conformant implementations of the Specification, alone or in combination with its authorized partners. All other rights reserved.

The Telecom Infra Project logo is a trademark of Telecom Infra Project, Inc. (the "Project") in the United States or other countries and is registered in one or more countries. Removal of any of the notices or disclaimers contained in this document is strictly prohibited.