Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 28 December 2025

C. Perocchio, Ed.
Ericsson

26 June 2025

TE and Path & Placement Computation for Cloud Native Applications
draft-perocchio-rtgwg-path-and-placement-00

Abstract

Currently a path computation is the ability to find the path/s connecting two or more points of the network identified with addresses. The path selection can be driven requiring constraints and metric minimization in traversing the network. The computed paths may be used for realizing a network slice providing connectivity to applications that shall be able to reach addresses of the end-points.

A cloud native application (CNA) becomes a network point only when an instance of it has been deployed in a site fulfilling the requirements for resources, scalability, reliability, security and costs... The process for identifying where to deploy a CNA is disjoined from path computation that is executed in a following step. If the path computation cannot satisfy the request not finding a connectivity solution, the chosen end-points shall be changed restarting the process from the selection of the deployment sites.

The CNA problem can be generalized to all the cases where more end-points can be alternatives for terminating a path. For example, an enterprise site may be attached to the ISP backbone via more nodes in risk diversity belonging to different network segments. How computing the best connectivity in the ISP network including the selection of the best placement for the end-points in the enterprise sites? Where locating end-points of overlay network may be managed as well as an application.

Status of This Memo

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

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

Copyright Notice

Table of Contents

1. Introduction 3
  1.1. Requirements Language 4
  1.2. Terminology 4
2. Requirements 5
3. Applicability of a Path and Placement Computation Element . . 6
4. The Architecture 7
5. Constrained Virtual End-Point 11
6. Cloud Native Application Identifier 11
7. Cloud Native Application Instance Identifier 12
8. Application Registry 12
9. Path and Placement Computation Element 12
10. PCEP Enhancements 14
  10.1. PCEP: Virtual-Endpoint Object Type 14
  10.2. PCEP: VIRTUAL-ENDPOINT TLV 15
  10.3. PCEP: CNA Subobject 16
  10.4. PCEP: Bundle Request 17
  10.5. PCEP: CNA Errors 17
11. BGP-LS Enhancements 18
  11.1. BGP-LS: Cloud Native Application Registry NLRI 18
12. YANG Models 19
13. Feasibility Considerations: an Algorithm 20
  13.1. The Algorithm Explained by Use Case 22
  13.2. The Algorithm Confirmation by Use Case 36
  13.3. Multiple Solutions 39
14. IANA Considerations 39

14.1. PCEP: New Object Type 40
14.2. New PCEP TLV Type Indicators 40
14.3. New PCEP IRO Subobject 40
14.4. New PCEP XRO Subobject 40
14.5. New PCEP Objective Functions 41
14.6. New PCEP-ERROR Object Error Types and Values 41
14.7. New BGP-LS NLRI and Attribute TLV 41
15. Security Considerations 42
16. References 42
16.1. Normative References 42
16.2. Informative References 44
Acknowledgements 44
Contributors 44
Author's Address 45

1. Introduction

Currently the deployment of an application is typically executed only when the clusters/physical locations to use have been already identified. With the introduction of the cloud technology the selected cluster for hosting the application will determine the address of the application. Furthermore with the cluster as a service methodology, at the deployment of the application a new cluster may be created resulting in a new network node with an its own address.

So the endpoints of a slice shall be defined offline for providing them as input for computing the connectivity.

As well it is not possible in a single iteration to identify both the best connectivity and the best placement of the end-points of an overlay networking for a client attached to the ISP backbone via more accesses with different characteristics.

The decoupling of the two decisional processes, the best placement selection in the cloud of the applications and the identification of the best paths across the network, may lead to a complex design flow of the overall slice.

This can evolve in a more dynamic behavior where the resources for the deployment - clusters and physical locations - can be associated to constraints (costs, resources availability, inclusions and exclusions, security risks, vulnerabilities, privacy, power efficiency...), allowing to find out optimal deployment solutions with the possible alternatives resolving the constraints in the dynamicity of the cloud. Doing so, it can be executed with the path selection for an overall optimization.

This may result in a dynamic and reactive mechanism for identifying alternative deployment solutions in case of a site crash or attack. In fact the security is a raising concern in a cloud centric world. It is important to be sure that a CNA involved in a network slice is secure, without any known vulnerabilities and that it does not violate laws, regulations or local policies. The proposed method can find deployments satisfying security constraints.

1.1. Requirements Language

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

1.2. Terminology

This document uses terms from PCEP [RFC5440], [RFC5521], [RFC5541]. The following abbreviations are used in this document:

3GPP:	Third Generation Partnership Project
ACTN:	Abstraction and Control of Transport Networks [RFC8453]
BGP-LS:	Border Gateway Protocol Link-State [RFC9552]
BOM:	Bill of Materials
CNA:	Cloud Native Application
CNF:	Cloud Native Network Function [CNCF-CNF]
DC:	Data Center
E2E:	End to End
ERO:	Explicit Route Object [RFC5440]
ETSI:	European Telecommunications Standards Institute
FOSS:	Free Open Source Software
GOF:	Global Objective Function [RFC5557]
IRO:	Include Route Object [RFC5440]
LSDB:	Link State Database
LSP:	Label Switched Path
LSPA:	LSP Attribute [RFC5440]
N:	Node
NLRI:	Network Layer Reachability Information [RFC9552]
NFV:	Network Function Virtualization
O-CU	O-RAN Centralized Unit defined by O-RAN ALLIANCE
O-DU	O-RAN Distributed Unit defined by O-RAN ALLIANCE
OF:	Objective Function [RFC5541]
P2P:	Point-to-Point
PCEP:	Path Computation Element Communication Protocol [RFC5440]
R:	Router
S:	Service
SBOM:	Software Bill of Materials

SDN:	Software Define Network
SVEC:	Synchronization Vector [RFC5440]
TDB:	Topology Database
TE:	Traffic Engineering
TED:	Traffic Engineering Database
TLV:	Type Length Value
UUID:	Universally Unique Identifier [RFC9562]
XRO:	Exclude Route Object [RFC5521]
VIM:	Virtual Infrastructure Manager [ETSI-GR-NFV-MAN-001]
VNF:	Virtual Network Function [ETSI-GR-NFV-MAN-001]
WAN:	Wide Area Network
WIM:	WAN Infrastructure Manager [ETSI-GS-NFV-IFA-010]

2. Requirements

The ingress and egress points of a path computation request may be virtual: as well the nodes of a network slice.

When virtual, the end-points may be described with attributes and constraints: the path computation shall identify the best solution according the specified metric for placing the virtual end-points in the network and for the requested connectivity.

The virtual end-point may be an application: attributes and constraints shall allow representing requirements like resources and security.

If a path and placement computation is requested to initiate a path or a slice involving one or more virtual end-points, it shall identify the placement solution and it shall address an initiate request with ’network’ end-points to the relevant network node.

The application may or may not be already deployed in the network node. The application may or may not be shared by more than one path or network slice.

Include or exclude applications, software modules not complying to the required security level.

How a cloud native application is deployed or shared is out of scope.

The requested network slice may identify requested CNA deployment and reuse existing connectivity.

3. Applicability of a Path and Placement Computation Element

The cloud is introducing new opportunities for using the virtualization technology not only in few big data centers but also in small nodes at the edge of the network. In this scenario the finding an optimal placement where to deploy a virtual application satisfying constraints both for latency and security may be not an easy job if the design of the overall slice considering network topology, security requirements, cloud resources availability and software choice are done by different teams with different instruments on different tables.

The virtualization defined by ETSI is not limited to the infrastructure provided by data centers, but it includes also network services for providing communication to the virtual network functions. For example, ETSI IFA 032 introduces the multi-site-connectivity and represents network services as in the following picture (reporting a simplified view of the original one):

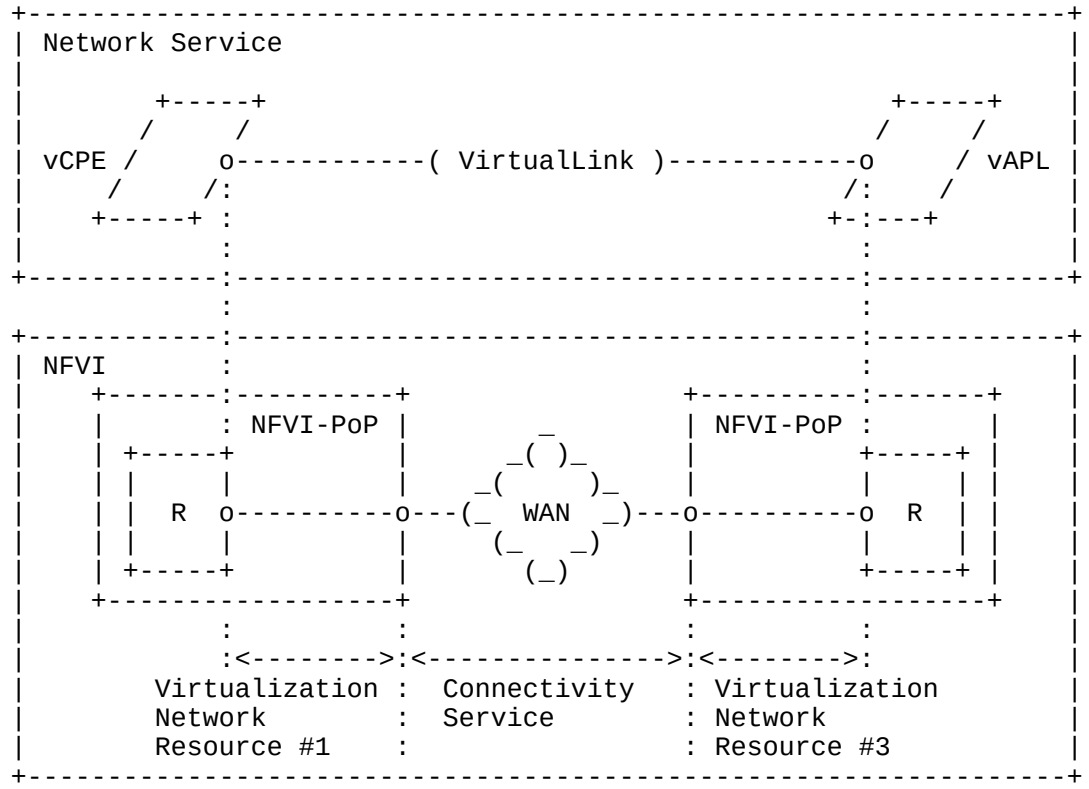


Figure 1: ETSI Network Service

The ETSI SOL 017 binds the interface of the WIM (WAN Infrastructure Manager) to the ACTN (Abstraction and Control of TE Networks) architecture, defined in [RFC8453], that can use PCEP and BGP-LS, as stated in [RFC8637].

The O-RAN WG5 in the Transport Specifications demands requirements and definitions for the transport network to the other standardization bodies (that is IETF for IP), as described in the next picture (a summarized view of how O-RAN WG5 Transport envisions the IP connectivity).

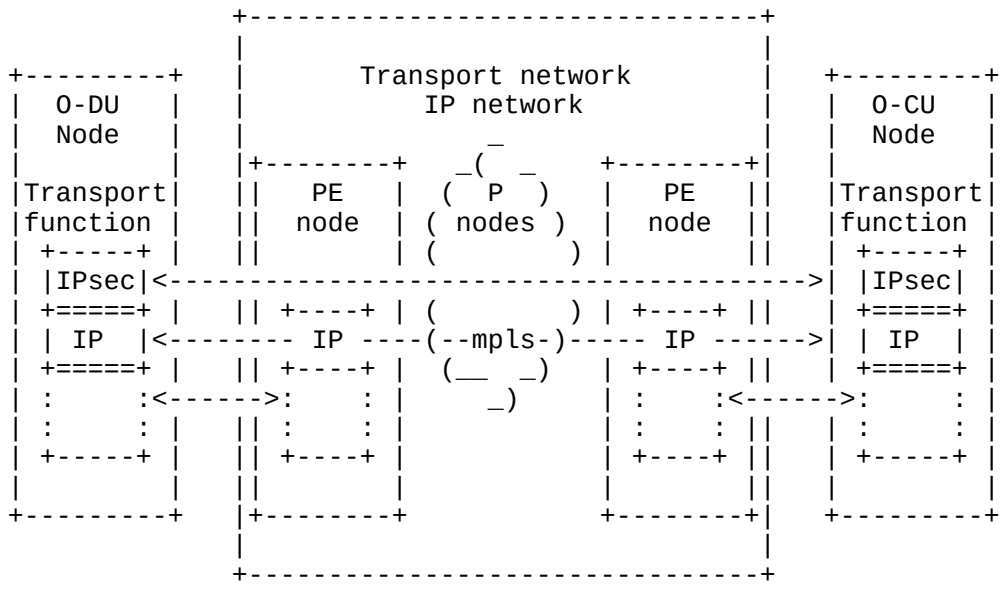


Figure 2: O-RAN O-DU - O-DU IP connectivity

4. The Architecture

The architecture proposed in this document requires additions to PCEP and BGP-LS. The proposed changes extend the protocols maintaining them back-compatible with the previous versions, and they should make possible mixed solutions where network slices are required with a fixed endpoint - perhaps the location of the network management center - and a constrained virtual end-point (see Section 5) on the other side with a specific cloud native application to control. The PCEP is extended for supporting constraints across diverse applications domains, including but not limited to FOSS compliance, SBOM, known vulnerabilities, and deployment location.

A network controller collects the node capabilities through the enhanced BGP. The node capability is a list of CNAs that a node can execute. A new network slice is requested using the enhanced PCEP protocol. The receiving controller runs a new algorithm to find a deployment solution satisfying the required constraints, for example regarding security that is a raising concern in a cloud centric world: it is important to be sure that a CNA involved in a network slice is secure without any known vulnerability and that it does not violate any law, regulation or local policy and the proposed method can find deployments satisfying security constraints (but not only).

If the connectivity of the nodes is already provided, the proposed method will use the provided connections finding a deployment solution according to it and the additional security constraints.

The figure below describes a typical network scenario: nodes, a network controller, a request for a new network slice connecting some 5G NFV. The controller collects the network status from BGP-LS that carries also which VNFs are available (for sharing or for deployment) on each node of the network. Each node reports the list of the available VNF with the specific version.

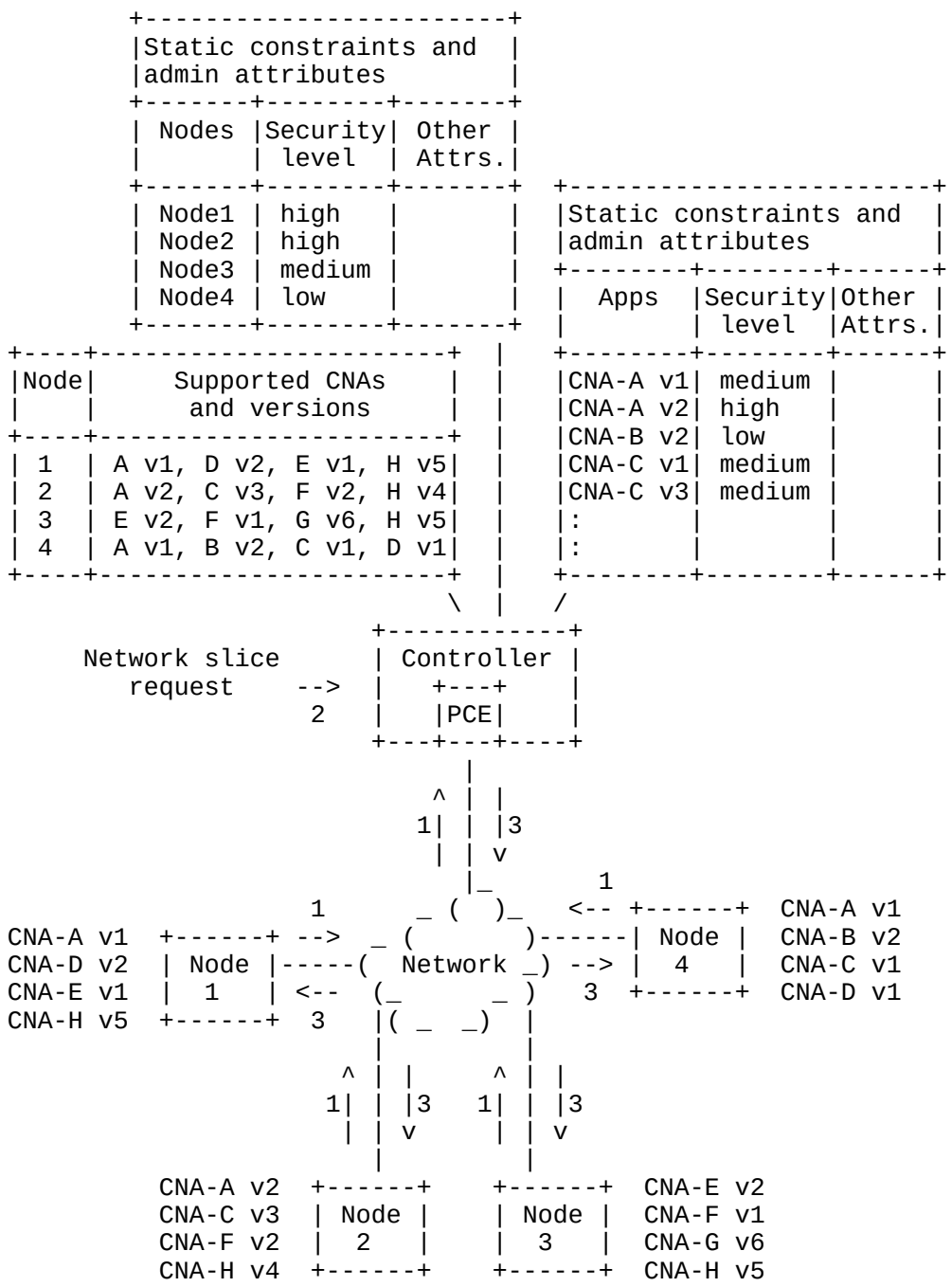


Figure 3: A possible architecture

Legend:

1. BGP-LS reports the supported and available applications in the network nodes to the controller
2. Enhanced PCEP network slice request for path and placement computation
3. Realization in the network of the slice eventually using PCEP too

The administrator of the network controller configures the static constraints and the administrative attributes for the new network slice. The static constraints may define what is the security level of each VNF and the maximum level of security supported by a node. For example, the security level of VNFs may consider different factors like known vulnerabilities, the reliability of the vendors support and others. While the security level supported by a node may consider for instance the place where the node is (restricted area or not), the model version, the security patches applied on the nodes and other factors.

The network controller looks for a solution to the incoming request using both the dynamic constraints of the node taken from the network and the configured static constraints. The goal is to find a deployment of the requested 5G network slice, the best one according to the requested metric, fulfilling all the constraints, both for applications placement and traffic path routing.

Every node (physical network element or virtualized cloud native cluster) is assumed to already have onboard a list of versioned CNAs. A versioned CNA is an implementation of a Network Function Virtualization as a specific Virtual Network Function with its version.

During a setup phase or in a runtime configuration update, the administrator of the network controller provides the administrative attributes and the static constraints for the nodes and the applications, such as the security level supported by the nodes, the security level required by the versioned CNA to run on a specific node.

The versioned CNAs may be either already available on the nodes in this phase or deployed during the network slice provisioning triggered by the network controller.

The network controller collects the topology via BGP-LS, as represented in the picture by the flow 1. The protocol is extended for introducing a new specific TLV associated to the node containing the list of versioned CNAs available for running on the node, in this document it is referred as CNA Registry. See Section 11.1 for protocol changes details.

The network controller collects the capability of the nodes to run all the versioned CNAs.

At runtime the client of the path and placement computation element requests the network slice computation using PCEP updated for introducing the new concept of the virtual end-point, see Section 5. it only refers to the type of the cloud native application that shall participate to the slice communicating over it and that may be still to be deployed. The path required via PCEP can be between 2 CNAs instead of 2 nodes: the ingress point could be one of the nodes on which the required "from" CNA is present and the egress point is one of the nodes on which the "to" CNA is present.

The network controller computes the optimal deployment and path by evaluating the service request, network topology, static constraints, and the application registry which provides metadata on available application instances, their deployment options, and associated requirements. The computation element of the network controller uses techniques typical of the routing protocols as explained in the following sections.

5. Constrained Virtual End-Point

As already described in previous sections, the constrained virtual end-point is the enabler of the path and placement computation. A Virtual End-Point may be an end-point not anchored yet to a specific network node: it represents a point of the network not yet identified by addressing that may or may not exist already, it can be described in terms of attributes and constraints. The path and placement computation element uses requested attributes and constraints for identifying where it should be located in the network. When not existing yet, the path and placement computation element reports the network node able to host it.

6. Cloud Native Application Identifier

This document introduces the definition of the identifier for the applications but does not provide indications on how the identifiers are assigned because it may depend on objectives of the network administrator.

The chosen format is a UUID that is an universally unique identifier 128 bits (16 bytes) long, because UUID is a common practice for representing a software component within a SBOM. In the context of this document the UUID value shall be unique in the scope of the path and placement computation.

An application identifier may represent not only the whole application but also components part of it, like a FOSS. As well the identifier may represent a type of application, for example implementing a network function of the 5G standard. The granularity of the representation in the registry, that is whether representing application components and application types, is a decision of the network administrator.

7. Cloud Native Application Instance Identifier

For representing an instance of a cloud native application a 32 bits (4 bytes) value is used. Instance identifiers shall be unique in the scope of the slice: they may be assigned by the client of the path and placement computation element. Since it is they are a client info, they are not included in the registry reported by BGP-LS.

8. Application Registry

In this document the set of the info of applications reported by BGP-LS and the database used for collecting them (in a node or in the network controller) is called Application Registry: how the info are organized in the database is an implementation decision. For example, in Figure 3 the application repository of the network controller is integrated with two tables with the static constraints and admin attributes of the nodes and of the applications.

The cloud native application registry contains at least a unique identifier for each cloud native application. This document does not provide indications whether the registry shall include the used software components and the type of the application. It is a decision of the network administrator.

This document describes only how application registry info are passed per node via BGP-LS providing a parent-child relationship, see Section 11.1.

9. Path and Placement Computation Element

The path and placement computation element is a classic PCE that supports the feature for identifying a network node fulfilling the constraints of virtual end-points, as defined in this document in Section 10.1.

For example, when the controller must deploy a new network slice including NFV (like in case of 5G), it addresses a request for a bundle of path computations specifying as ingress and egress endpoints only the NFVs with the constraints, security included. On successful computation the slice design is returned to the controlled with the selected nodes and paths. The constraints can be defined in a generic way to include application characteristics. Obviously security is very important and a specific claim to identify FOSS to include or to exclude in the CNA to deploy, has been identified.

In the classic path computation requests ingress and egress nodes (or node interfaces) shall be specified as addresses (IPv4 or IPv6 or Unnumbered) specifying the label limitations (mainly for photonics limitations) with the requested path attributes. Supposing to have to create a network slice connecting 3 services Y-X-Z with latency constraints in the area 10.0.0.0, the nodes shall be identified according to the availability of the requested services X, Y, Z that shall not use FOSS A (for example, the nodes 10.0.0.1, 10.0.0.2 and 10.0.0.3) and then the path computation request shall be issued, for example with two independent path computation requests:

```
Request #1: {Compute, from 10.0.0.1, to 10.0.0.2, latency 5}
Request #2: {Compute, from 10.0.0.1, to 10.0.0.3, latency 15}
```

or with a bundled request:

```
Request: [
  {Compute, from 10.0.0.1, to 10.0.0.2, latency 5},
  {Compute, from 10.0.0.1, to 10.0.0.3, latency 15},
]
```

while in case of path and placement computation the ingress and egress addresses can be substituted with the software identifiers of the applications implementing the services, whose selection can be restricted adding constraints like for the exclusion of the FOSS-A:

```
Innovative bundled request: [
  {Compute, from:  CNA-X {include [area: 10.0.0.0] ...},
            to:    CNA-Y {include [area: 10.0.0.0],
                          exclude [FOSS-A], ...},
            latency: 5}
  {Compute, from:  CNA-X {include [area: 10.0.0.0] ...},
            to:    CNA-Z {include [area: 10.0.0.0],
                          exclude [FOSS-A], ...},
            latency: 15}
]
```

The output of the request is the design of the slice with the sites where CNA-X. CNA-Y and CNA-Z can be deployed according to the constraints.

The path and placement computation element may help in finding the proper CNA involved in the network slice fulfilling the security constraints, that is without known vulnerabilities and that does not violate any law, regulation or local policy.

10. PCEP Enhancements

10.1. PCEP: Virtual-Endpoint Object Type

For representing the endpoints of a path of a network slice as a set of constraints a new Object Type is introduced in the PCEP Object Class 4 - END-POINTS: this proposal allocates value 6 with the name Virtual-Endpoint.

Existing object types for END-POINTS specify the requested endpoint with an address or a router-id with an interface-id, identifying it uniquely. Currently it may contain label restrictions too.

The proposed Virtual-Endpoint contains also other TLVs such as Include Route Object for represent the list of nodes that shall be considered for the deployment, the eXclude Route Object (XRO) for representing the nodes that shall not be used, the SRLG to include or exclude all nodes with links sharing a reported risk, the LSPA to include or exclude all nodes with links associated with the administrative colors (aka affinities). Other TLVs can be introduced in the object, but it is up to the receiving Path Computation Element to fulfil them. In this set there could be those representing the required level of security of a node. The TLVs present in the Virtual-Endpoint follow the grammar:

```
<virtual-endpoint-tlvs> ::= <virtual-p2p-endpoints>

<virtual-p2p-endpoints> ::= <endpoint-choice>
                            <endpoint-choice>

<endpoint-choice> ::=
    <IPV4-ADDRESS> | <IPV6-ADDRESS> | <virtual-constrained-endpoint>

<virtual-constrained-endpoint> ::=
    <virtual-endpoint> [<virtual-endpoint-attribute-list>]

<virtual-endpoint-attribute-list> ::= [<LSPA>]
                                      [<IRO>]
                                      [<XRO>]
                                      [<metric-list>]
                                      [<OF>]
```

where:

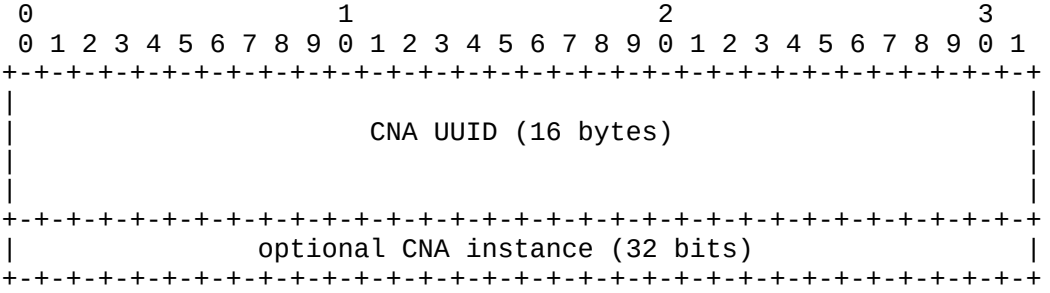
```
<virtual-endpoint> ::= <VIRTUAL-ENDPOINT>
```

is a new definition while LSPA, IRO, XRO, metric-list, OF, IPV4-ADDRESS and IPV6-ADDRESS are already available in IETF standardization.

The endpoint-choice has been provided not to supersede existing definitions, but to make possible to request the computation when only one between source and destination is a CNA.

10.2. PCEP: VIRTUAL-ENDPOINT TLV

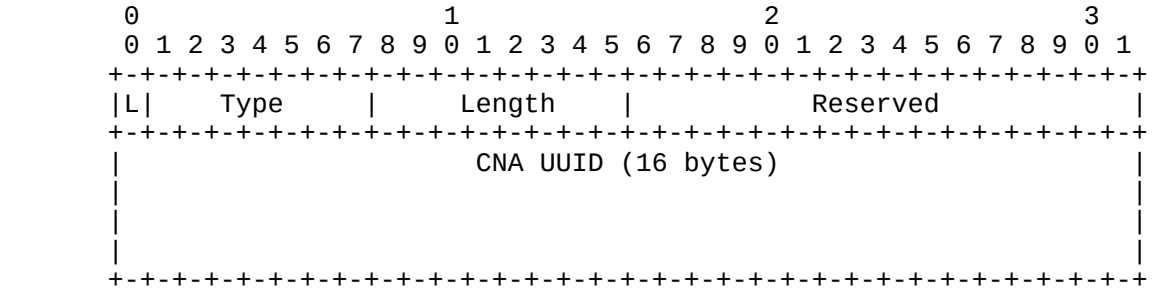
The VIRTUAL-ENDPOINT TLV (this proposal allocates the type value 80) shall represent the CNA in the network slice. The same CNA may run multiple in multiple instances in the same network slice, like in case of geo-redundancy reasons. The CNA is represented as a UUID (16 bytes), that is generally used to represent software at low level (not for humans) with an optional 32-bits number for the instance.



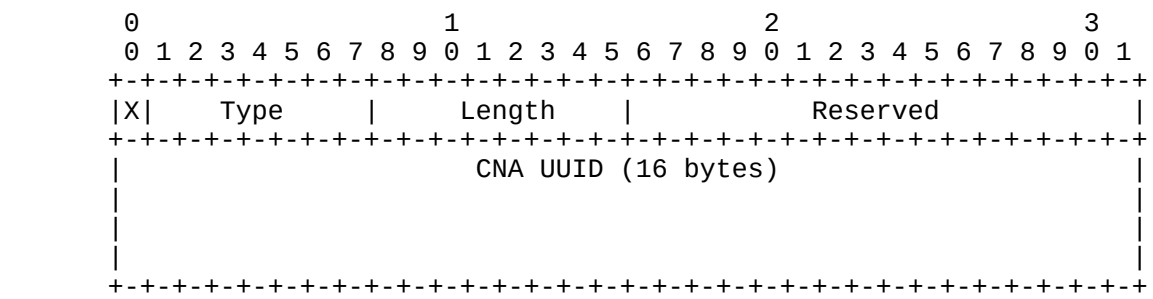
The body of the VIRTUAL-ENDPOINT TLV has the length of 16 bytes of the UUID identifying the CNA plus 4 optional bytes when the instance of the CNA is reported. The CNA UUID can be used to represent a specific cloud native application or an its version or its type or a part of the application software: so it may stand for either a VNF or its version or the NFV that the application implements or a FOSS that the application uses. The assignment of the UUID and of the instance and how they are assigned is not part of this proposal.

10.3. PCEP: CNA Subobject

The CNA subobject is a new definition that can be used both in XRO and IRO to include or to exclude a specific software. If a FOSS shall not be used by the applications running on the network slice, the request can be addressed to the PCE with the XRO with this subobject to exclude the undesired software. While, if a FOSS shall be preferred to others, the request can use the IRO with this subobject for including it. The CNA subobject for IRO as:



referring to [RFC3209] for L, Type and Length. Similarly, for XRO is defined as:



referring to [RFC5521] for the definitions of X and Length. This proposal allocates the value 66 for the type to represent the CNA Subobject both in IRO and in XRO.

10.4. PCEP: Bundle Request

In general, the network slice topology is composed of more than two nodes connected by only one link: each single path of the slice may be requested specifying the ASSOCIATION object reporting the Virtual Network Identifier in the VIRTUAL-NETWORK-TLV. The PCE request allows the concatenation of multiple requests for path computations that can be synchronized with SVEC (Synchronization VECtor) object whose svec-list may include the OF (as global-objective function, GOF). The following new objective-functions are introduced by this proposal:

Function Code	Description
19	Maximize CNA deployment security
20	Minimize CNA deployment cost
21	Maximize CNA deployment diversity
22	Maximize CNA scaling

Table 1

The OF (Objective Function) List TLV at virtual-endpoint level or at global level, may be included to specify a set of objective functions for choosing a solution: new objective functions may be introduced for new use cases.

10.5. PCEP: CNA Errors

The PCEP-ERROR object is used to report the errors:

Error-Type	Meaning	Error-value
34	CNA Error	0: Unassigned
		1: The PCE cannot satisfy the request, no CNA END-POINTS
		2: unknown CNA UUID
		3-255: Unassigned

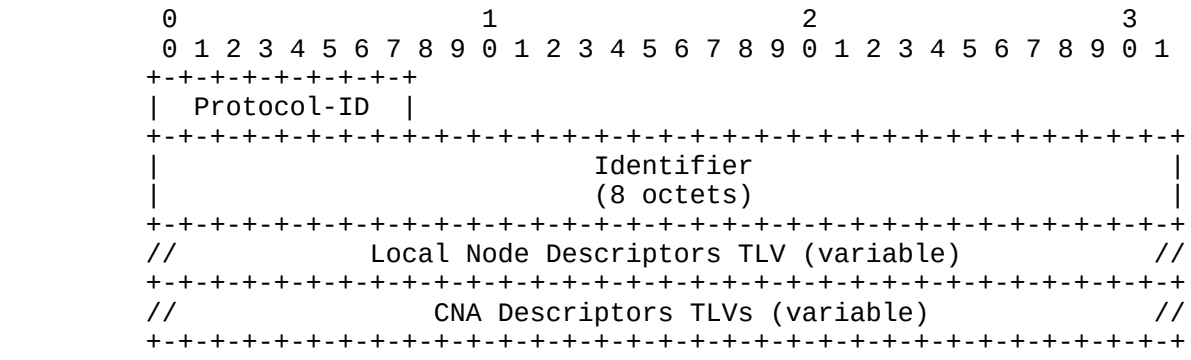
Table 2

11. BGP-LS Enhancements

11.1. BGP-LS: Cloud Native Application Registry NLRI

In this proposal nodes can advertise supported and available applications using BGP. BGP has already been extended for conveying the link-state information stored in the TDB (the topology database either LSDB or TED) of the routers to a controller of the network for feeding a Path Computation Element: the NLRI format has been used. This proposal introduces the Cloud Native Application NLRI.

The Cloud Native Application Registry NLRI is introduced by this proposal and has the following format:



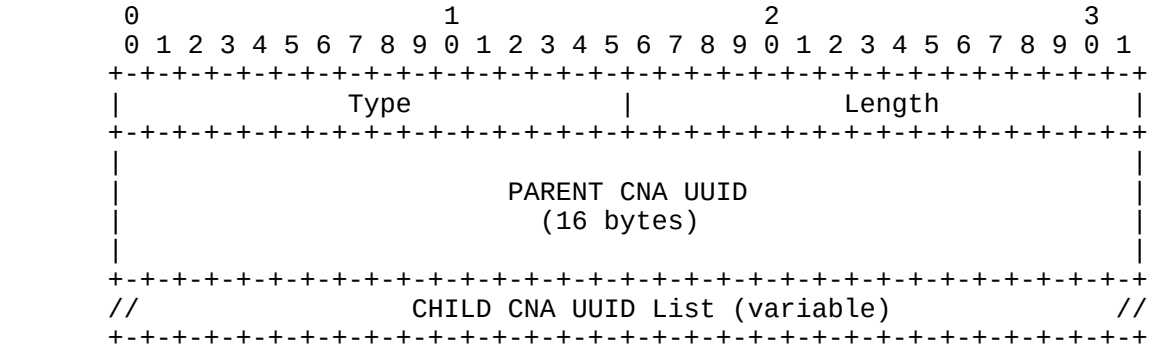
If BGP uses local information, the Protocol-ID shall be 'Static' (protocol-id value 5, [RFC9552]) as in already available in IETF definitions. The Identifier is the BGP-LS Instance Identifier (Instance-ID), as already defined.

For the local node descriptor [RFC9552], a new sub-TLV code point is introduced:

Sub TLV Code Point	Description	Length
519	Cluster VIM Address	Variable * 4 bytes for IPv4 * 16 bytes for IPv6

Table 3

The CNA Descriptor TLV is here defined as:



The CNA UUID may represent a specific application or an its version or its type. A CNA type (like an NFV) may have more than one implementation in the network, and each implementation may have more versions of it: the relationship parent-child is used for the purpose. The child list can be empty.

12. YANG Models

For supporting the case when a network controller is used, as described in Section 4, the relevant YANG models should be updated for reflecting the info and the operations necessary for a Path and Placement Computation.

The PCEP proposed updates may require augmentation to the following YANG models:

* for requesting path and placement computation for [I-D.ietf-teas-yang-path-computation]
* for including placement computation capabilities, the protocol updates and the LSP representation for [I-D.ietf-pce-pcep-yang]
* for representing applications in tunnels and LSPs for [I-D.ietf-teas-yang-te]
* for eventual new TE data types for [RFC8776]

while the te-topology [RFC8795] may host a per node view of the application repository information collected by BGP-LS.

The opportunity to augment the models reported in this paragraph shall be evaluated with the relevant design teams.

13. Feasibility Considerations: an Algorithm

This proposal reports in short a simple algorithm for adding the placement capability to a path computation element. This algorithm example is meant to demonstrate the feasibility of a path and placement computation element. Its implementation is not requested since developers can identify more efficient alternatives.

The example algorithm can be summarized in 5 steps:

Step 1: the constraints specified for each CNA of the slice is resolved in a set of nodes of the network fulfilling them, here we call them "deployment options of the CNA".

Step 2: the connectivity matrix of the slice reports the connections between CNAs, each row can be substituted with the combinations of the of the deployment options of the 2 CNAs; in this phase a full mesh connectivity can be considered between nodes assuming that any connectivity can be realized, otherwise the full mesh can be reduced removing unfeasible or unavailable connectivity.

Step 3: for each node "eligible" to be a "deployment option" a tree is considered where it is the root and all other "eligible" nodes are the leaves: the tree has a single level of depth.

Step 4: the connectivity matrix of the slice is applied to the tree of the "eligible" node. Only the matrix rows that connect the root to the leaves are considered. The matrix rows representing the connections between leaves are discarded (to avoid loops).

Step 5: the remaining rows in each connectivity matrix of the trees can be used to check whether a deployment option is possible.

The above algorithm finds the motivation in how L2 networks of bridges work using Spanning Tree, Multiple Spanning Tree, Generic Attribute Registration Protocol and multihoming. The Spanning Tree cuts loops in the network blocking ports. The Multiple Spanning Tree allows to use more than one tree at a time. The Generic Attribute Registration Protocol allows to identify the connectivity between two edge ports of the Spanning Tree. Multihoming allows a client to be connected to more edge ports of the bridged network: only one of them will be open at a time.

In Step 1 the provided constraints for each end-point of the slice can be used to filter and to identify a set of nodes matching the constraints.

In Step 2 the "eligible" nodes are the bridges of the network. Each CNA can be considered as a client of the bridged network attached to one edge port of each "eligible" node: when there is more than one "eligible" node, the CNA can be viewed as a client in multihoming scenario. The actual connectivity of the network allows considering the bridged network as a full mesh.

In Step 3 the full mesh bridged network can be "drawn" as a polygon, convex and regular, where the vertices are the "eligible" nodes and the edges together with the diagonals are the links: on this representation of the bridged network a multiple spanning tree that is configured to run as many instances as the "eligible" nodes are and play the root bridge role with all the others are directly connected designated bridges, that is its leaves. This means that each tree has as active links only the 2 edges and the diagonals of the polygon starting from the vertex standing for the "eligible" node playing the root bridge role.

Step 4 applies some protocol well known mechanism to the network slices deployment scenario: each node announces the CNAs that is connected to on the slice, as in traditional bridged networks an edge port would announce the configured VLAN via GVRP (GARP for VLAN). In an actual bridged network, the GARP message used for the announcement is multicast and it is forwarded over root and designated ports: when a port of the tree is crossed in both direction by announcements for the same configuration, the port is opened for the connectivity. In the network slice scenario, all the communication work aligned to the described mechanism with an exception: the traffic cannot be forwarded from one tunnel to another tunnel so the announcement will not work for edge-edge communication, aka leaf-leaf communication in the new context.

In Step 5 the connections identified by the multiple spanning tree shall be considered in the perspective requiring only one access point to the bridged network for each client at a time (as in the well-known multihoming use case).

13.1. The Algorithm Explained by Use Case

Considering the example where the endpoints of the following slice:

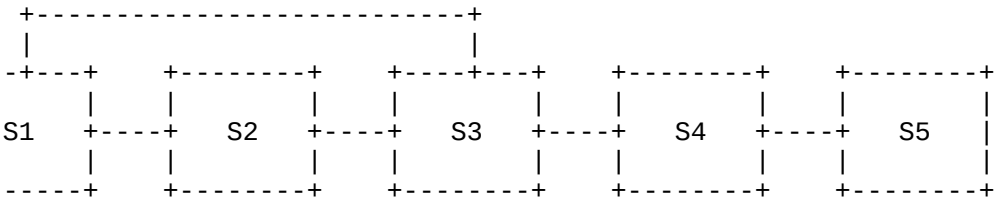


Figure 4: Example of a network of cloud native applications

shall be located and deployed in the following network:

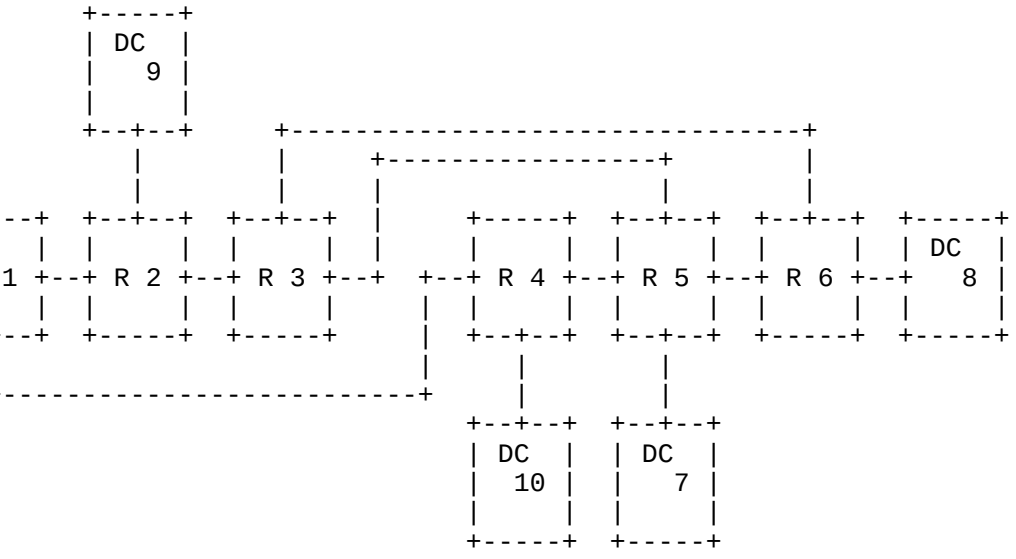


Figure 5: The network providing connectivity to the cloud native applications

with the following constraints (Step 1):

* CNA S1 can be only on node DC-7 or DC-8.

* CNA S3 only on node DC-10.

* CNA S5 only on node DC-9.

* CNAs S2 and S4 have no constraints.

Not all nodes support CNAs deployment, in the considered case only DC nodes (7, 8, 9, 10) that are data centers.

The following table reports a summary of the scenario:

------NETWORK------	-------SLICE-------
* Router	* Services
R1	S1
R2	S2
R3	S3
R4	S4
R5	S5
R6	
	* Connectivity
* Data Centers	
	S1-S2
DC7	S2-S3
DC8	S1-S3
DC9	S3-S4
DC10	S4-S5
* Network Links	* Deployment Constraints
R1-R2	
R1-R4	S1 -> DC7 or DC8
R2-R3	S2 -> any node
R2-DC9	S3 -> DC10
R3-R5	S4 -> any node
R3-R6	S5 -> DC9
R4-R5	
R4-DC10	
R5-R6	
R5-DC7	
R6-DC8	

Table 4: Resources of the use case

The resulting connectivity matrix of the network slice can be represented as:

	S1	S2	S3	S4	S5
S1-S2:	DC-7 or 8	any DC	-	-	-
S1-S3:	DC-7 or 8	-	DC-10	-	-
S2-S3:	-	any DC	DC-10	-	-
S3-S4:	-	-	DC-10	any DC	-
S4-S5:	-	-	-	any DC	DC-9

and the slice representation on the actual network including deployment constraints as:

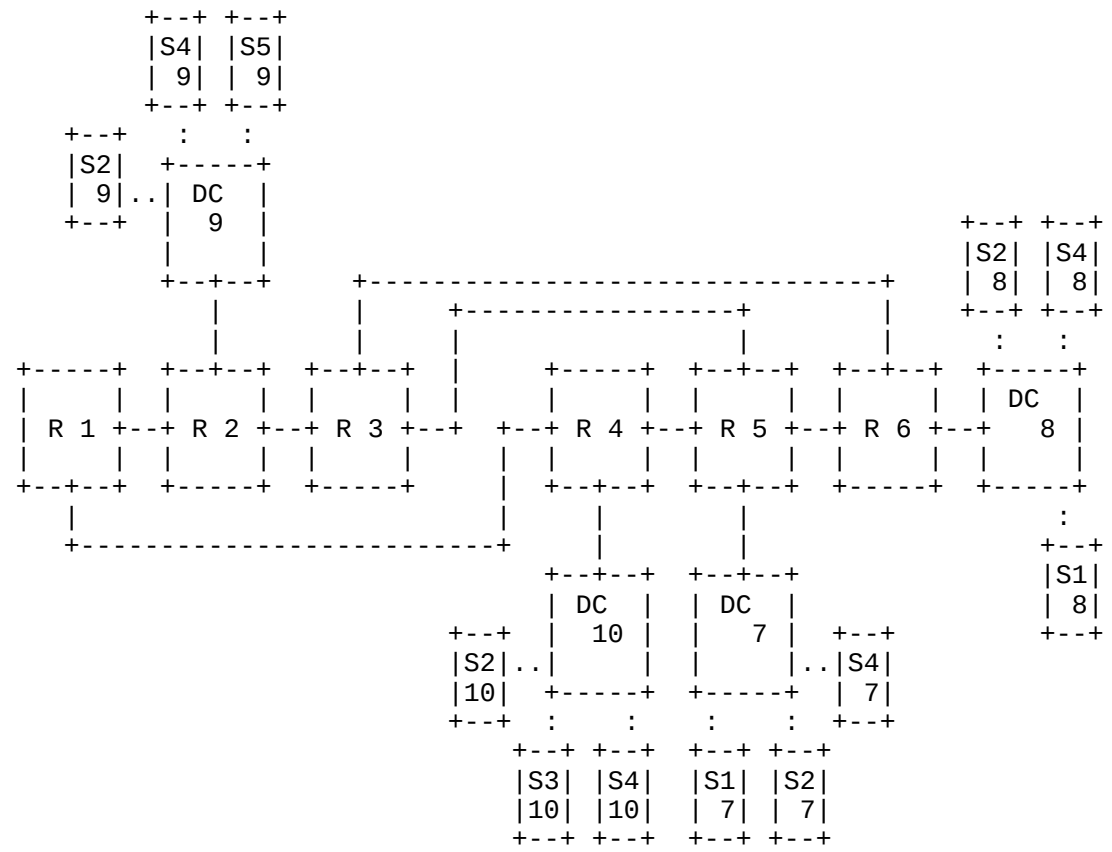


Figure 6: App deployment alternatives in the example network

In Step 2 the algorithm focuses on slice connectivity and the slice in the network can be represented ignoring the complexity of the network and including only "eligible" nodes, slice connectivity and services of the slice with their accesses point as:

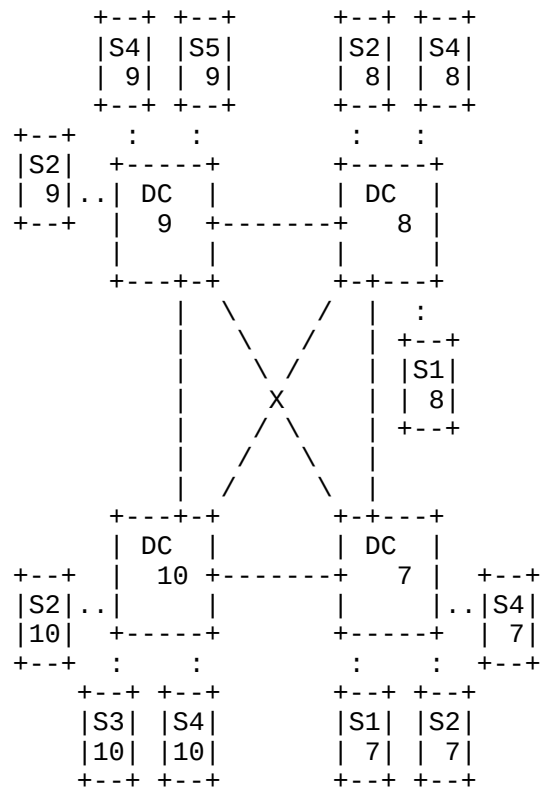
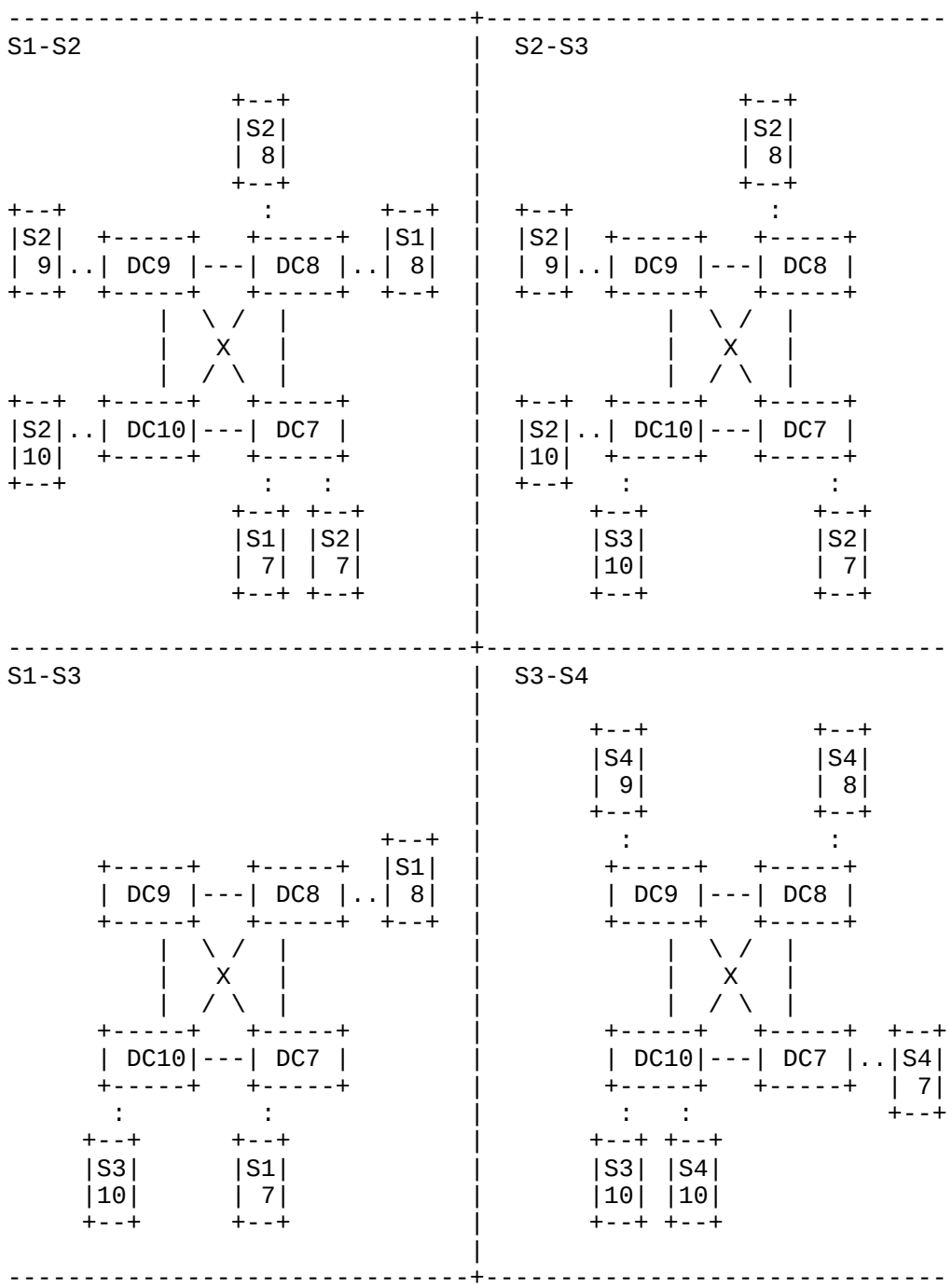


Figure 7: Summarization of the network with app deployment alternatives

Each link of the slice can be considered as follow:


S1-S2
S2-S3
S1-S3
S3-S4
DC9
DC8
DC10
DC7

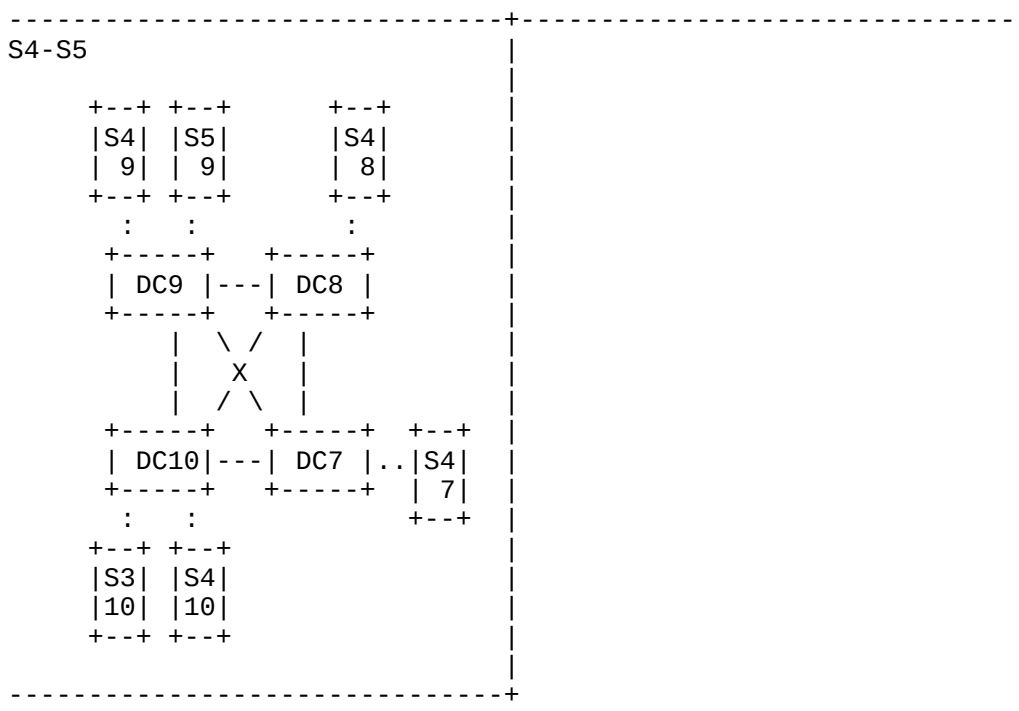


Figure 8: Per slice path the summarization of the network with app deployment alternatives

In Step 3 "eligible" nodes shall be considered as root of a spanning tree, here in case of node 7 as root:

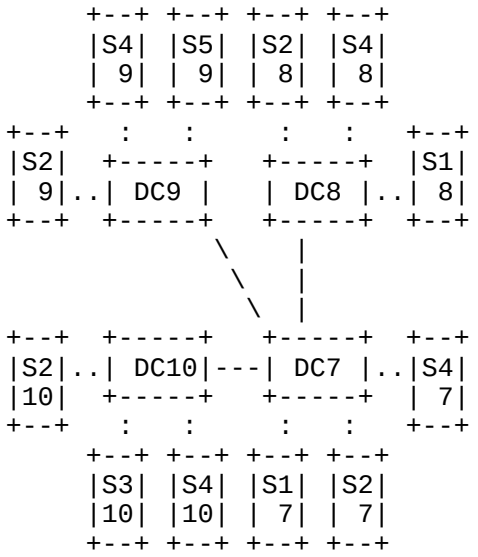
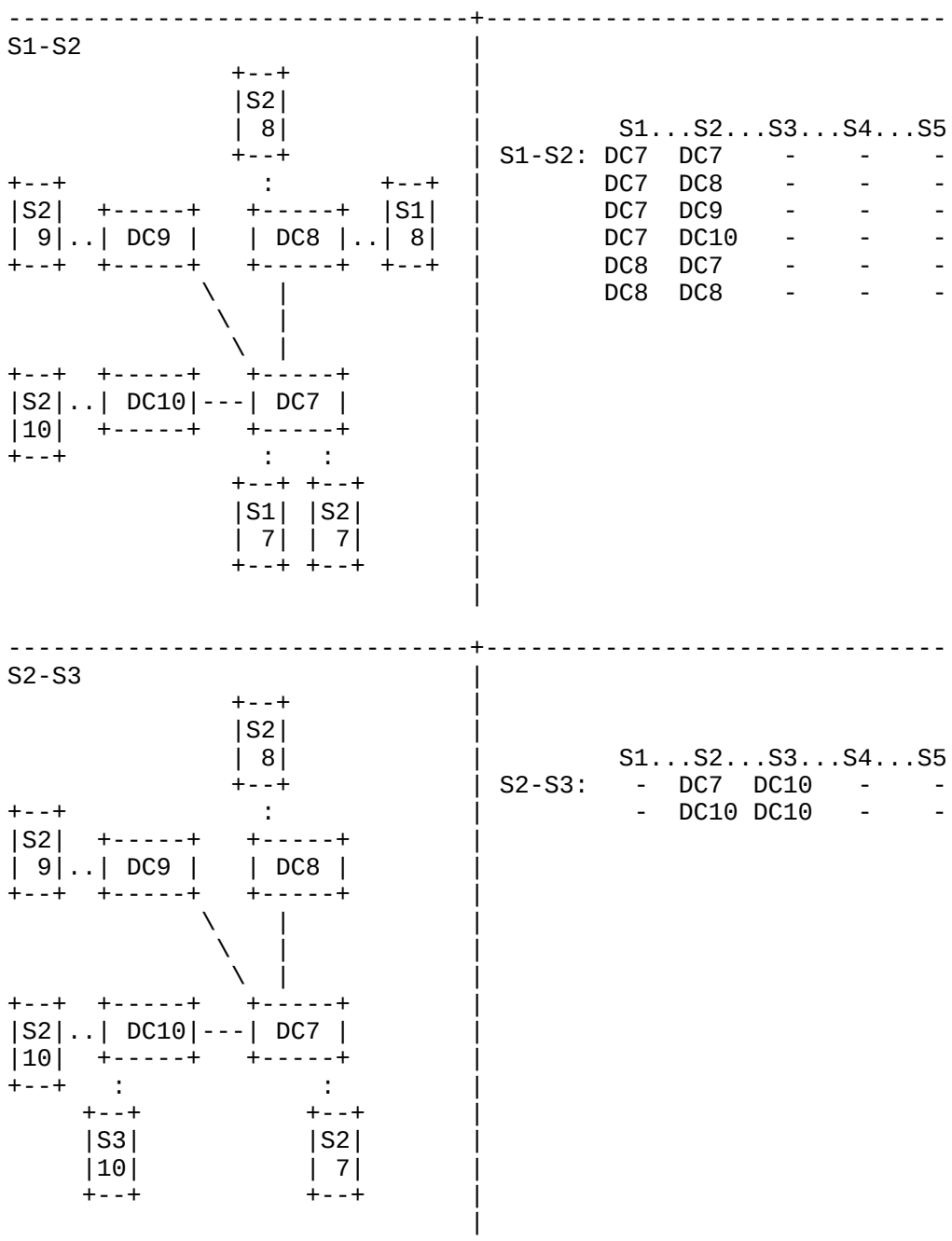


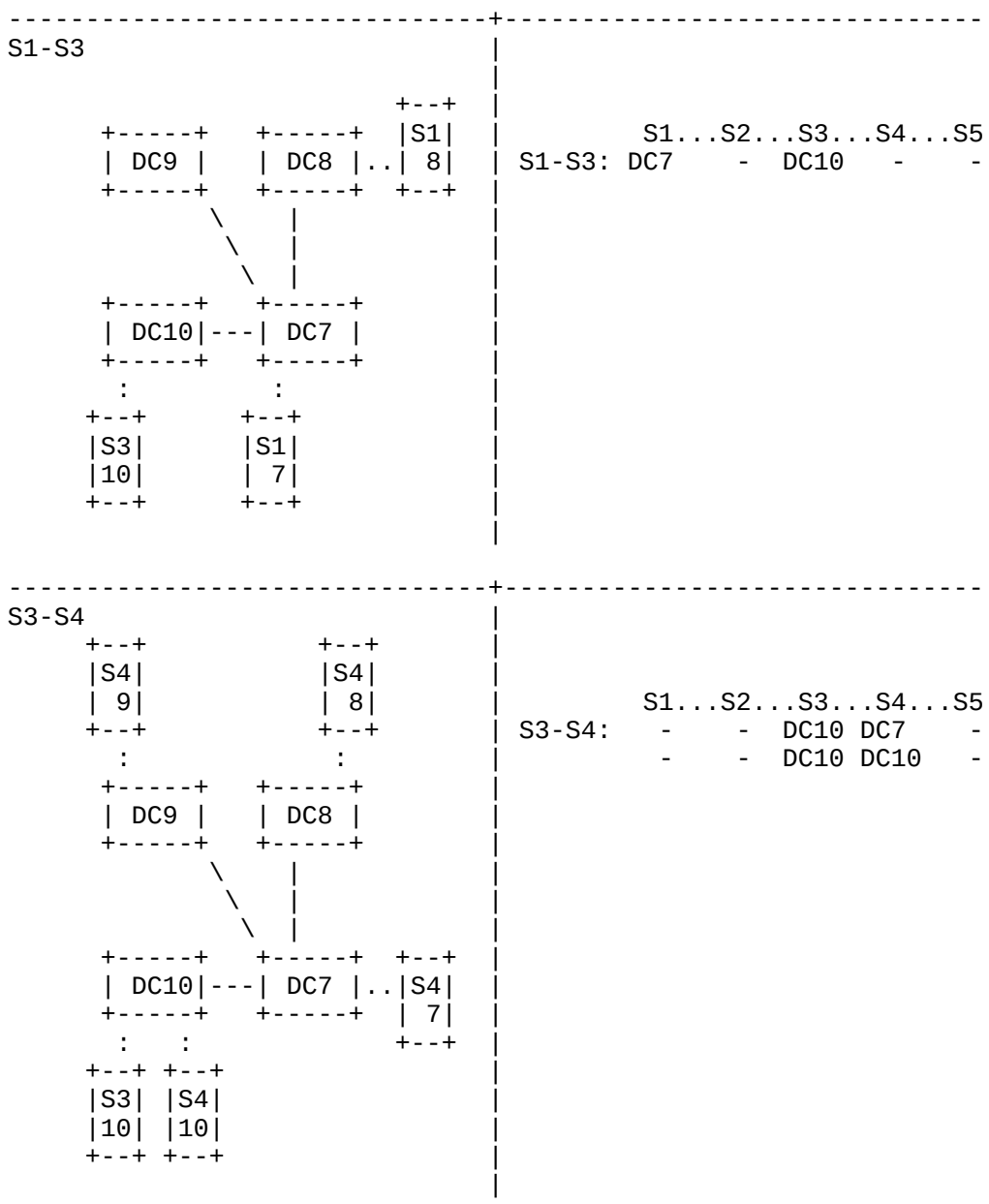
Figure 9: Node 7 root of the tree network with app deployment alternatives

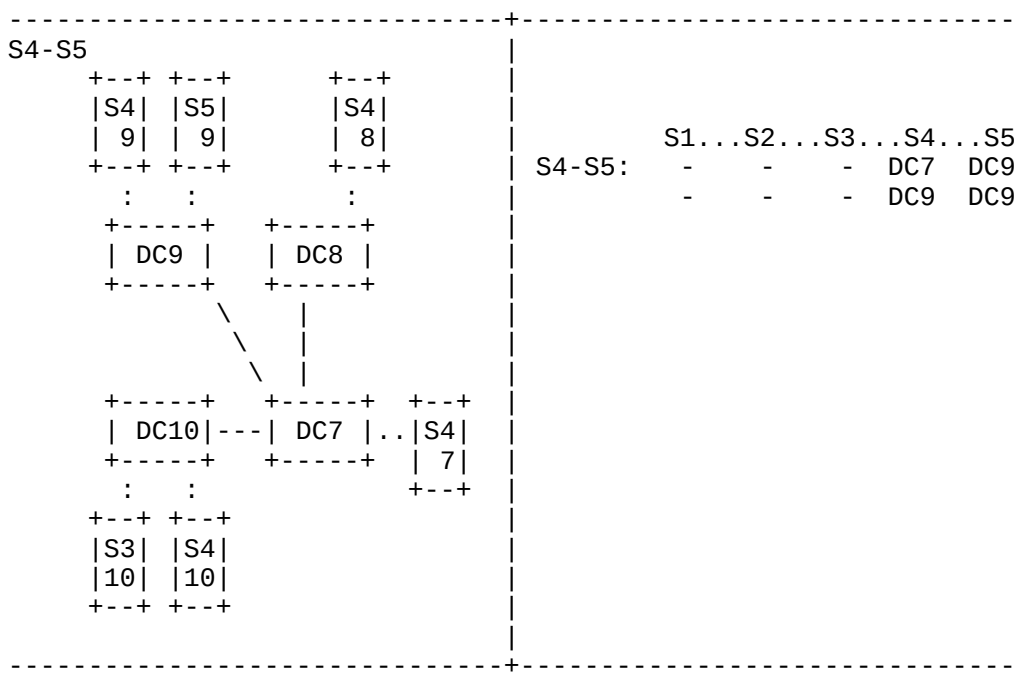
obtaining in Step 4 the following connectivity matrix with all deployment options for each application:

```
        ..S1.....S2.....S3.....S4.....S5
S1-S2:  DC7     DC7       -       -       -    root-root -> ok
        DC7     DC8       -       -       -    root-leaf -> ok
        DC7     DC9       -       -       -    root-leaf -> ok
        DC7     DC10      -       -       -    root-leaf -> ok
        DC8     DC7       -       -       -    leaf-root -> ok
        DC8     DC8       -       -       -    same leaf -> ok
        DC8     DC9       -       -       -    leaf-leaf -> pruning
        DC8     DC10      -       -       -    leaf-leaf -> pruning
S1-S3:  DC7       -     DC10      -       -    root-leaf -> ok
        DC8       -     DC10      -       -    leaf-leaf -> pruning
S2-S3:    -     DC7     DC10      -       -    root-leaf -> ok
          -     DC8     DC10      -       -    leaf-leaf -> pruning
          -     DC9     DC10      -       -    leaf-leaf -> pruning
          -     DC10    DC10      -       -    same leaf -> ok
S3-S4:    -       -     DC10    DC7       -    leaf-root -> ok
          -       -     DC10    DC8       -    leaf-leaf -> pruning
          -       -     DC10    DC9       -    leaf-leaf -> pruning
          -       -     DC10    DC10      -    same leaf -> ok
S4-S5:    -       -       -     DC7     DC9  root-leaf -> ok
          -       -       -     DC8     DC9  leaf-leaf -> pruning
          -       -       -     DC9     DC9  same leaf -> ok
          -       -       -     DC10    DC9  leaf-leaf -> pruning
```

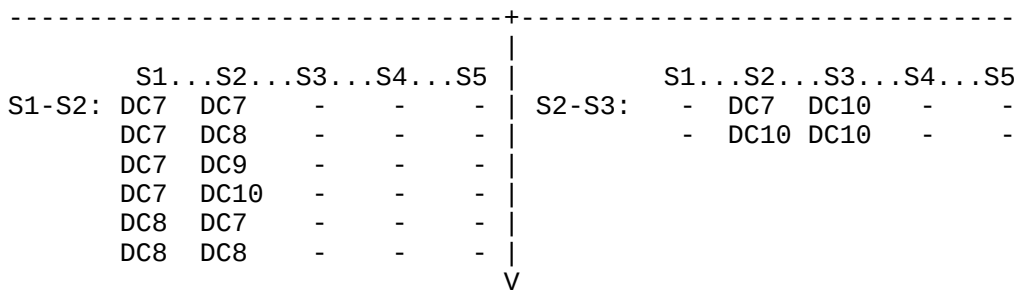
In Step 5 each connectivity information is combined with the others, so in this phase each topology with its connectivity matrix is compared for checking common nodes:


S1-S2
S2-S3
S1...S2...S3...S4...S5
S1-S2: DC7 DC7 - - -
DC7 DC8 - - -
DC7 DC9 - - -
DC7 DC10 - - -
DC8 DC7 - - -
DC8 DC8 - - -
S2-S3: - DC7 DC10 - -
- DC10 DC10 - -
|S2| 9|..| DC9 |
|S1| 8|
|S2| 10|..| DC10|
|S1| 7|
|S2| 7|
|S3| 10|
|S2| 7|
DC8
DC7


S1-S3
+--+
|S1|
+-----+ +-----+ 8|
| DC9 | | DC8 |..|
+-----+ +-----+ +--+
\ |
\ |
\ |
+-----+ +-----+
| DC10|---| DC7 |
+-----+ +-----+
: :
+--+ +--+
|S3| |S1|
|10| | 7|
+--+ +--+
S1...S2...S3...S4...S5
S1-S3: DC7 - DC10 - -
S3-S4
+--+ +--+
|S4| |S4|
| 9| | 8|
+--+ +--+
: :
+-----+ +-----+
| DC9 | | DC8 |
+-----+ +-----+
\ |
\ |
\ |
+-----+ +-----+ +--+
| DC10|---| DC7 |..|S4|
+-----+ +-----+ | 7|
: : +--+
+--+ +--+
|S3| |S4|
|10| |10|
+--+ +--+
S1...S2...S3...S4...S5
S3-S4: - - DC10 DC7 -
- - DC10 DC10 -



And then combining the matrices, common services (a column in both matrices with non-zero values) are considered keeping only common values while other columns (with zero values in at least one of the two matrices) are 'indifferent'. Considering at the beginning S1-S2 and S2-S3 matrices S2 is the only one in common, so "eligible" nodes for S2 shall be present in both matrices:



```
                  S1....S2....S3....S4....S5
          S1-S2: DC7   DC7     -     -     -
                 DC7   DC10    -     -     -
                 DC8   DC7     -     -     -
          S2-S3:   -   DC7   DC10    -     -
                   -   DC10  DC10    -     -
```

Combining the previous result S1-S3 matrices, S1 and S3 services are present in both matrices, so only common "eligible" nodes shall be considered.

For example, DC8 as deployment option for service S1 is discarded because not present in S1-S3 matrix

```
--------------------------------+--------------------------------
                                |
 from the previous combination  |         S1...S2...S3...S4...S5
                                | S1-S3: DC7    -  DC10   -    -
                                V
```

```
                  S1....S2....S3....S4....S5
          S1-S2: DC7   DC7     -     -     -
                 DC7   DC10    -     -     -
          S2-S3:   -   DC7   DC10    -     -
                   -   DC10  DC10    -     -
          S1-S3: DC7    -    DC10    -     -
```

Going on combining with S3-S4:

```
--------------------------------+--------------------------------
                                |
 from the previous combination  |         S1...S2...S3...S4...S5
                                | S3-S4:   -    -  DC10 DC7    -
                                |          -    -  DC10 DC10   -
                                V
```

```
                  S1....S2....S3....S4....S5
          S1-S2: DC7   DC7     -     -     -
                 DC7   DC10    -     -     -
          S2-S3:   -   DC7   DC10    -     -
                   -   DC10  DC10    -     -
          S1-S3: DC7     -   DC10    -     -
          S3-S4:   -     -   DC10  DC7     -
                   -     -   DC10  DC10    -
```

And finally, with S4-S5

```
--------------------------------+--------------------------------
                                |
 from the previous combination  |          S1...S2...S3...S4...S5
                                | S4-S5:    -    -    -  DC7  DC9
                                |           -    -    -  DC9  DC9
                                V

                  S1....S2....S3....S4....S5
          S1-S2: DC7   DC7     -     -     -
                 DC7   DC10    -     -     -
          S2-S3:   -   DC7   DC10    -     -
                   -   DC10  DC10    -     -
          S1-S3: DC7     -   DC10    -     -
          S3-S4:   -     -   DC10  DC7     -
          S4-S5:   -     -     -   DC7   DC9
```

The combination of the matrices produces the results in case of tree topology with "eligible" node 7 as root:

```
                   S1....S2....S3....S4....S5
                  DC7   DC7   DC10  DC7   DC9
                  DC7   DC10  DC10  DC7   DC9
```

that is requiring network connectivity between DC-7 and DC-9 and between DC-7 and DC10, with the attributes and constraints according to hosted services:

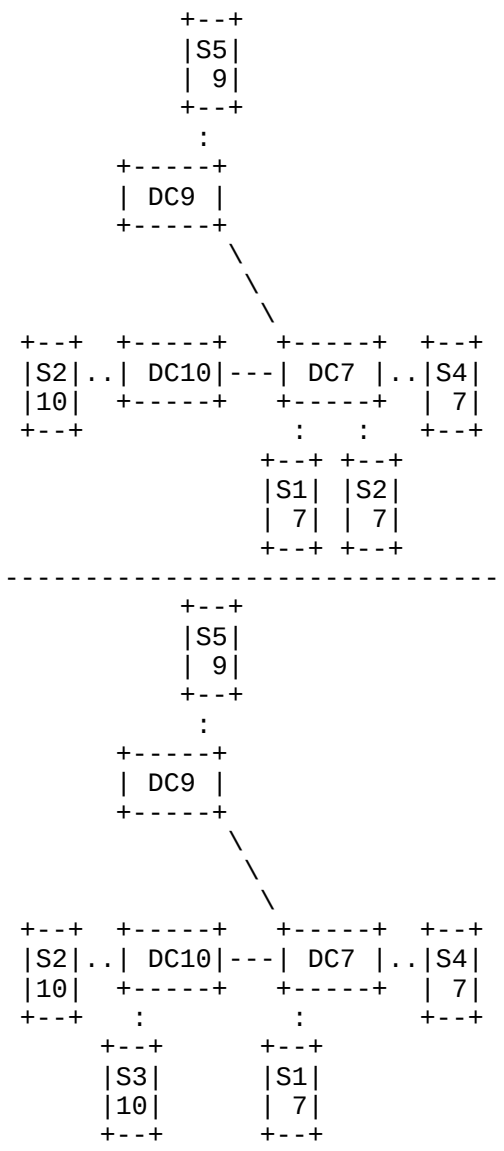


Figure 10: Deployment solutions for node 7 as root of the tree network

A tree may produce no solution.

The solutions found for each tree can be combined as per multiple spanning tree.

A requested metric or an objective function may drive the selection of the best deployment option privileging the deployment or the network metrics, or combining them.

13.2. The Algorithm Confirmation by Use Case

Considering a new use case where a network slice is requested with services connected in ring topology and with strict deployment constraints, each single tree topology cannot provide a solution to the problem, but using the multiple spanning tree final combination the only one possible solution can be found by the algorithm.

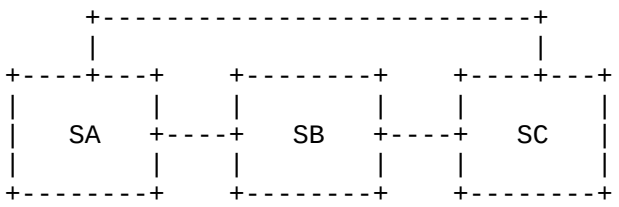


Figure 11: Another example of a network of cloud native applications

The deployment constraints are:

* the application S-A deployable only on N1 node
* S-B available only on N2 node
* S-C only on N3

and the network view is:

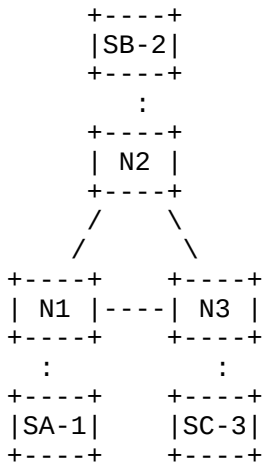


Figure 12: The network with apps single deployment option

so the slice requires the following paths:

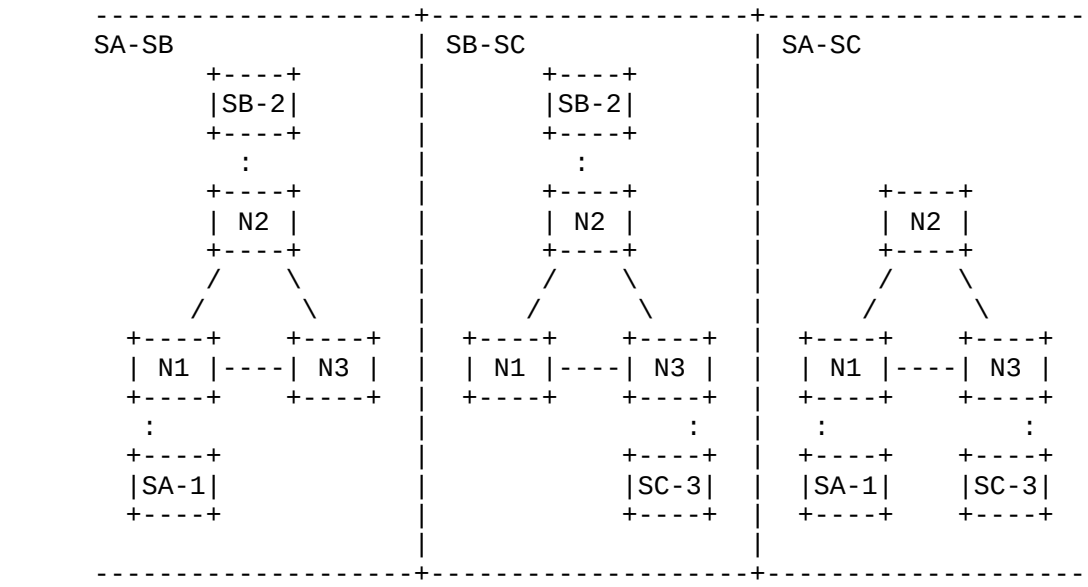


Figure 13: Paths of the slice in the network between apps single deployment option

N1 as root topology is not suitable for fulfilling SB-SC connectivity:

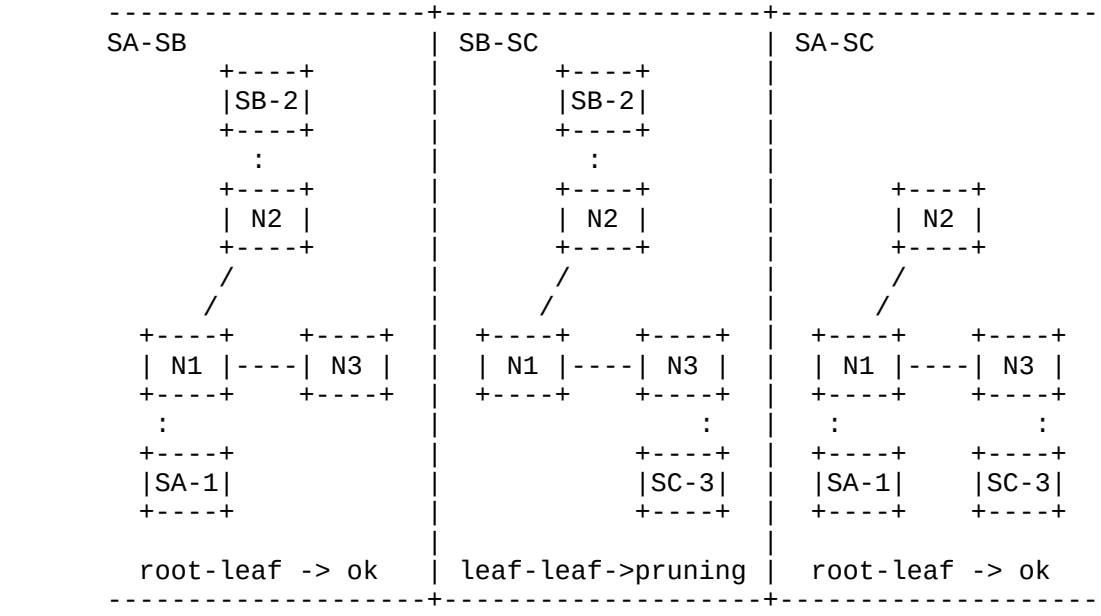


Figure 14: N1 as root of the tree network

as well N2 for SA-SC:

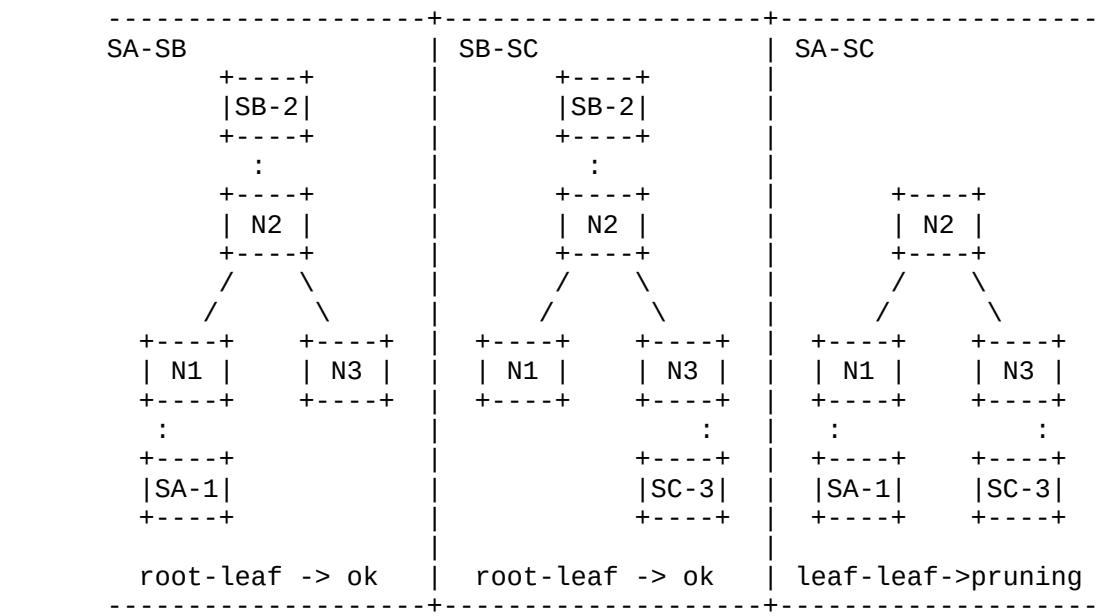


Figure 15: N2 as root of the tree network

and the same in case of N3 as root of the tree network for SA-SB:

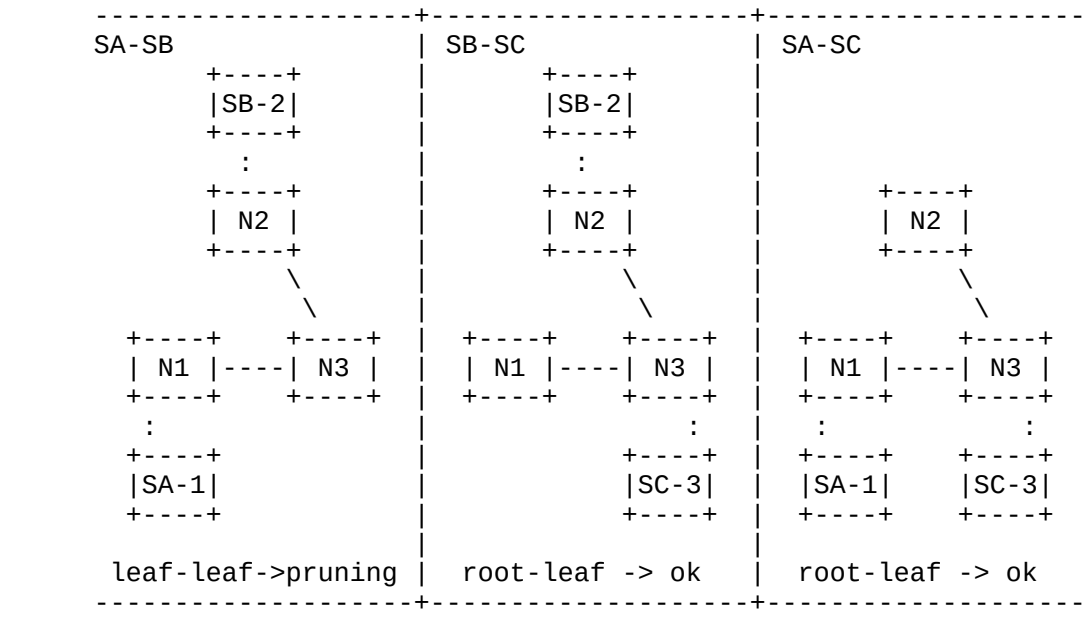


Figure 16: N3 as root of the tree network

Only combining all matrices as in the previous use case, it is possible to find the only one apps deployment solution.

13.3. Multiple Solutions

The proposed algorithm could provide multiple solutions: using the Objective Functions defined in PCEP protocol is possible to select the best one among them. If the Objective Functions are not provided in the PCEP request, in case of multiple deployment solutions, a default mechanism would select the first one.

14. IANA Considerations

IANA is asked to register the following entries:

14.1. PCEP: New Object Type

Object Class Value	Name	Object-Type	Reference
4	END-POINTS	6: Virtual-Endpoint	this document

Table 5

14.2. New PCEP TLV Type Indicators

Value	Description	Reference
80	VIRTUAL-ENDPOINT	this document

Table 6

14.3. New PCEP IRO Subobject

Value	Description	Reference
66	CNA	this document

Table 7

14.4. New PCEP XRO Subobject

Value	Description	Reference
66	CNA	this document

Table 8

14.5. New PCEP Objective Functions

Code Point	Name	Reference
19	Maximize CNA deployment security	this document
20	Minimize CNA deployment cost	this document
21	Maximize CNA deployment diversity	this document
22	Maximize CNA scaling	this document

Table 9

14.6. New PCEP-ERROR Object Error Types and Values

Error-Type	Meaning	Error-value	Reference
34	CNA Error	0: Unassigned	this document
		1: The PCE cannot satisfy the request, no CNA END-POINTS	this document
		2: unknown CNA UUID	this document
		3-255: Unassigned	this document

Table 10

14.7. New BGP-LS NLRI and Attribute TLV

TLV Code Point	Description	Reference
519	Cluster VIM Address	this document

Table 11

15. Security Considerations

This document should not affect the security of the Internet. [CHECK]

16. References

16.1. Normative References

[I-D.ietf-pce-pcep-yang]
Dhody, D., Beeram, V. P., Hardwick, J., and J. Tantsura, "A YANG Data Model for Path Computation Element Communications Protocol (PCEP)", Work in Progress, Internet-Draft, draft-ietf-pce-pcep-yang-30, 26 January 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-pce-pcep-yang-30>.

[I-D.ietf-teas-yang-path-computation]
Busi, I., Belotti, S., de Dios, O. G., Sharma, A., and Y. Shi, "A YANG Data Model for requesting path computation", Work in Progress, Internet-Draft, draft-ietf-teas-yang-path-computation-24, 13 February 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-teas-yang-path-computation-24>.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V. P., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels, Label Switched Paths and Interfaces", Work in Progress, Internet-Draft, draft-ietf-teas-yang-te-38, 29 May 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-teas-yang-te-38>.

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

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC5521] Oki, E., Takeda, T., and A. Farrel, "Extensions to the Path Computation Element Communication Protocol (PCEP) for Route Exclusions", RFC 5521, DOI 10.17487/RFC5521, April 2009, <https://www.rfc-editor.org/info/rfc5521>.

[RFC5541] Le Roux, JL., Vasseur, JP., and Y. Lee, "Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP)", RFC 5541, DOI 10.17487/RFC5541, June 2009, <https://www.rfc-editor.org/info/rfc5541>.

[RFC5557] Lee, Y., Le Roux, JL., King, D., and E. Oki, "Path Computation Element Communication Protocol (PCEP) Requirements and Protocol Extensions in Support of Global Concurrent Optimization", RFC 5557, DOI 10.17487/RFC5557, July 2009, <https://www.rfc-editor.org/info/rfc5557>.

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

[RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for Abstraction and Control of TE Networks (ACTN)", RFC 8453, DOI 10.17487/RFC8453, August 2018, <https://www.rfc-editor.org/info/rfc8453>.

[RFC8637] Dhody, D., Lee, Y., and D. Ceccarelli, "Applicability of the Path Computation Element (PCE) to the Abstraction and Control of TE Networks (ACTN)", RFC 8637, DOI 10.17487/RFC8637, July 2019, <https://www.rfc-editor.org/info/rfc8637>.

[RFC8776] Saad, T., Gandhi, R., Liu, X., Beeram, V., and I. Bryskin, "Common YANG Data Types for Traffic Engineering", RFC 8776, DOI 10.17487/RFC8776, June 2020, <https://www.rfc-editor.org/info/rfc8776>.

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

[RFC9552] Talaulikar, K., Ed., "Distribution of Link-State and Traffic Engineering Information Using BGP", RFC 9552, DOI 10.17487/RFC9552, December 2023, <https://www.rfc-editor.org/info/rfc9552>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/info/rfc9562>.

16.2. Informative References

[CNCF-CNF] Cloud Native Computing Foundation, "Cloud Native Thinking for Telecommunications", 2020, <https://github.com/cncf/telecom-user-group/blob/master/whitepaper/cloud_native_thinking_for_telecommunications.md>.

[ETSI-GR-NFV-MAN-001] ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration; Report on Management and Orchestration Framework", 2021, <https://www.etsi.org/deliver/etsi_gr/NFV-MAN/001_099/001/01.02.01_60/gr_nfv-man001v010201p.pdf>.

[ETSI-GS-NFV-IFA-010] ETSI, "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Functional requirements specification", 2023, <https://www.etsi.org/deliver/etsi_gs/nfv-ifa/001_099/010/04.05.01_60/gs_nfv-ifa010v040501p.pdf>.

Acknowledgements

To the projects PAROMA-MED (founded by the European Horizon 2020 Program for research, technological development and demonstration under grant agreement n属 101070222) and 6Green (founded by the European Union’s Horizon Europe, grant agreement No 101096925) for setting communication requirements that contributed to originate and to generalize the solution.

To Francesco Lazzeri (former Ericsson, now retired) for identifying the terms of the problem.

To Orazio Toscano (Ericsson), Rosario Colica (Ericsson), Luca Piccinini (Ericsson), Joel Halpern (Ericsson) for the support.

Contributors

Manuela Scarella
Ericsson
Via Melen 77
16152 Genoa GE
Italy
Email: manuela.scarella@ericsson.com

URI: www.ericsson.com

Domenico Cotroneo
Ericsson
Via Melen 77
16152 Genoa GE
Italy
Email: domenico.cotroneo@ericsson.com
URI: www.ericsson.com

Author’s Address

Carlo G. Perocchio (editor)
Ericsson
Via Melen 77
16152 Genoa GE
Italy
Email: carlo.perocchio@ericsson.com
URI: www.ericsson.com