2014

Pablo Aguilar Cabadas

Universidad Carlos III
de Madrid

[PCE PROTOTYPE WITH SEGMENT ROUTING AND BGPLS SUPPORT]

Agradecimientos

En primer lugar me gustaría dar las gracias a mi familia, en especial a mis padres por haberme dado la oportunidad de llegar a donde estoy ahora. Sin su esfuerzo y su dedicación nunca hubiera sido de capaz de recorrer este camino.

Mención especial merecen mis compañeros de carrera Ricky y Junqui con los que he salido vivo de esta dura travesía. Sin el apoyo que nos hemos brindado mutuamente y el trabajo que hemos puesto posiblemente no hubiéramos llegado a junio de 2014.

A mi familia de Telefónica, concretamente a Víctor López del que he aprendido innumerables cualidades tanto a nivel personal como profesional. Ha sido en todo momento una referencia a seguir y un guía en mi primera aventura en el mundo laboral.

Agradecer a Arturo Azcorra su predisposición y su ayuda en múltiples materias. Sus consejos y su experiencia me han ayudado muchísimo en la toma de muchas de las decisiones que he tenido que afrontar a lo largo de mi trayectoria a nivel académico y profesional, desde mi etapa en el colegio hasta ahora.

Finalmente, a mis dos apoyos morales: M.M y V.S. El mundo me dio la oportunidad de conocer a estas dos increíbles personas y ellas han sido parte insustituible en esta etapa que hoy termina. Gracias por haber estado siempre.

"When you want to succeed as bad as you want to breathe then you will be successful".

Abstract

This project presents two contributions to the PCE implementation in Telefonica I+D: Segment Routing and the upgrade of the BGP-LS protocol to the 3rd version of the draft to support MPLS and GMPLS scenarios.

Regarding the first contribution, this document is intended to assess the use of Segment Routing in centralised traffic-engineering scenarios. It will attempt to make a validation of such technology using the available IETF drafts and publications and trying, at all time, to back-up the use cases with experimental demonstrations.

Moreover, the 3rd version of the BGP-LS protocol draft was implemented. This protocol opens the possibility to export the network's topology and its Traffic Engineering parameters to external entities. The BGP-LS extensions developed enables to retrieve the TE parameters for MPLS and GMPLS networks.

The development of the project was done in Telefonica R&D's facilities within the Core Network Evolution group. The code extends Telefonica's PCE and network protocols to support Segment Routing and the new version for BGP-LS. As such, both the PCEP and the BGP-LS protocols were enhanced with the latest IETF drafts that define the technology.

Once the code was developed and debugged, a series of tests were run in order to validate that the format used followed all the proposed standards. These tests have been defined following the sections that constitute each draft in an attempt to proof the use of each protocol in the most exhaustive possible way. It is important to remark that the validation tests are done not only with Telefonica code, but also with external prestigious entities like Cisco, Telecom Italia, Centre Tecnològic Telecomunicacions Catalunya or Consorzio Nazionale Interuniversitario per le Telecomunicazioni.

KEY WORDS

SEGMENT ROUTING – BPG-LS – PCE – TRAFFIC-ENGINEERING

Table of Content

1 Introduction 7
1.1 Motivation 7
1.2 Objectives 8
1.3 Document Structure 8
2 Segment Routing 10
2.1 Introduction: Historic network evolution 10
2.1.1 The early days 10
2.1.2 The rise of the Stupid Network 11
2.1.3 Intelligent networking v2.0 12
2.2 What is Segment Routing? 13
2.3 Segment Routing Architecture 14
2.4 Segment Routing use cases 17
2.4.1 Class of Service based Traffic Engineering 17
2.4.2 SDN controller use in Segment Routing 19
3 Path Computation Element 21
3.1 The PCE Architecture 21
3.2 Use Cases 22
3.3 Workflow of the basic PCE model 23
3.4 Hierarchical PCE architecture: Inter-domain path computation 25
3.5 Stateful, Stateless and Active PCEs 26
3.5.1 Stateless PCE 26
3.5.2 Stateful PCE 27
3.5.3 Active Stateful PCE 27
3.6 Path Computation Element Protocol (PCEP) 28
4 Border Gateway Protocol with Link State Capability 31
4.1 Application scenario 31
4.2 BGP-LS protocol 33
4.2.1 BGP-LS NLRI 36
4.2.2 The Link-State attribute 38

4.3 Translation from OSPF to BGP-LS 40
4.3.1 Mapping BGPLS NLRI descriptors from OSPF 41
4.3.2 Mapping of OSPF-TE parameters into the BGP-LS attribute 43
4.4 Validation Tests 44
4.4.1 Test-bed scenario 44
4.4.2 BGP-LS extensions 45
4.4.3 OSPF to BGP-LS translation 50
5 Segment Routing Extensions 55
5.1 PCEP Extensions for Segment Routing support 55
5.1.1 The Segment Routing PCE Capability TLV 55
5.1.2 The Path Setup Type TLV 56
5.1.3 The SR-ERO Object 57
5.2 PCEP extensions for SR validation scenario 57
5.2.1 SR Capability Negotiation 59
5.2.2 Path Computation Request-Reply Message Exchange 61
5.2.3 Initiating a SR-Path from the PCE 67
5.2.4 Error Reporting 72
5.3 BGP-LS Extensions for Segment Routing support 73
5.3.1 SR Node Attribute TLVs 74
5.3.2 SR Link Attribute TLVs 74
6 Conclusions 76
References 79
Acronyms 80
Annex A: Steps to test Segment Routing using Telefonica R&D's PCE .. 81
Annex B: Steps to tests Segment Routing with TID-PCE & TID-PCC 85

Table of Figures

Figure 1: Backbone bandwidth growth [3] 11
Figure 2: Example of network scenario 14
Figure 3: Packet P at ingress node I 15
Figure 4: Label switching in SR 16
Figure 5: Example of Shortest Path Routing in SR 17
Figure 6: SR database [5] 17
Figure 7: CoS-based TE example 18
Figure 8: SDN use case example 20
Figure 9: Basic PCE architecture 22
Figure 10: Functional description of a PCE based architecture 24
Figure 11: H-PCE scenario example 25
Figure 12: Generic PCEP session 29
Figure 13: TE distribution with BGP-LS 32
Figure 14: TE distribution using BGPLS in an H-PCE scenario 32
Figure 15: BGP message exchange and Finite State Machine 34
Figure 16: Update Message Format [7] 36
Figure 17: TLV format [8] 36
Figure 18: Node NLRI format [8] 37
Figure 19: Link NLRI format [8] 37
Figure 20: Node descriptor sub-TLVs [8] 38
Figure 21: Link descriptors [8] 38
Figure 22: Node Attributes [8] 39
Figure 23: Link Attributes [8] 40
Figure 24: OSPF to BGPLS Node Descriptors 42
Figure 25: OSPF to BGPLS Link Descriptors 43
Figure 26: OSPF LSA to BGP-LS Attribute 43
Figure 27: BGP-LS draft 04 testing scenario 44
Figure 28: BGP-LS message trace 46
Figure 29: Test-bed Update message format 46
Figure 30: Test-bed BGPLS NLRI format 47
Figure 31: Test-bed Node Descriptors TLV format 48
Figure 32: Test-bed Link Descriptors TLV format 48
Figure 33: Test-bed Link State attribute format 49
Figure 34: Traffic Engineering Database format 50
Figure 35: OSPF to BGP-LS message trace 51
Figure 36: Test-bed OSPF LS Update message 52
Figure 37: OSPF Available Labels sub-TLV 53
Figure 38: Link State Attribute with Available Labels 54
Figure 39: SR-PCE-Capability TLV [11] 55
Figure 40: LSP-Setup-Type TLV [12] 56
Figure 41: SR-ERO Sub-Object [11] 57
Figure 42: Test-Bed Scenario 58
Figure 43: Capability Negotiation 60
Figure 44: Capability Negotiation with SR-capable Cisco Router 60
Figure 45: Stateful Capability TLV format 61
Figure 46: General PCReq-PCRep message exchange 62

Figure 47: Emulated SR scenario 62
Figure 48: Request-Response message trace 63
Figure 49: Resulting SR path using node SIDs 64
Figure 50: Source Routing using Node SIDs 65
Figure 51: Route provisioning using adjacency SIDs 66
Figure 52: Path Provisioning using Node and Adjacency SIDs combined 67
Figure 53: PCE Initiated SR-Path using Node SIDs 68
Figure 54: PCE Initiated SR path using node SIDs 69
Figure 55: CISCO_1 to CISCO4 shortest path example 69
Figure 56: PCE Initiated SR-Path using AdjSIDs 70
Figure 57: PCE Initiated SR path using Adjacency SIDs 71
Figure 58: CISCO_1 SR-Path to 4 using Adjacency SIDs 71
Figure 59: Bad Label Value Error 72
Figure 60: Unsupported Number of SIDs Error 73
Figure 61: SR Node Attributes [13] 74
Figure 62: SR Link Attributes [14] 74

1 Introduction

1.1 Motivation

This project exploits the idea of centralized traffic engineering through the use of a Path Computation Element together with the most innovative Traffic Engineering technique called Segment Routing.

Since its introduction in October 2012 at a Cisco hosted conference with operators, Segment Routing has seen increased its popularity by several orders of magnitude within the development community. Six months later, in March 2013, the first public presentation on Segment Routing was made in the MPLS world congress. This presentation included the first working code developed by Cisco for the use case defined in the segment-routing-use-cases-00 draft.

The expectation rose so fast that just immediately after the use case was presented, two IETF drafts where published (ospf-segment-routing-extensions-00 and sivabalan-pce-segment-routing-00). At this point, Alcatel and Ericsson joined the project. In July 2013, supporting the IETF standardization process and seeking interoperability between the different vendors, Cisco released yet another draft (segment-routing-use-cases-01) and at that point Juniper joined all the protocol extensions draft.

By March 2014, one year after the first public presentation of the technology, 15 IETF drafts were available with all the major vendors participating in the development process following the guidelines set by the operators in order to fulfil their requirements.

Up until now, many use cases have been defined meaning that operators are really finding future ways to exploit the benefits of Segment Routing for their particular purpose. In addition, in order to implement these use cases very few protocol extensions in ISIS, OSPF and PCEP are needed which translates into easier implementation and rapid deployment.

This project was born to take centralized traffic engineering to yet another level by combining the use of Segment Routing together with a Path Computation Element leveraging the use of source routing and tunnelling paradigms. In addition, it sets the scenario to make yet another stride in the world of Software Defined Networking (SDN) by including the latest BGP-LS protocol as one of the PCE's supported features. In other words, the BGP-LS [1]

used by Telefonica has been updated in order to make it compatible with the latest releases and use cases defined for the protocol.

We will use a centralized TE controller (PCE) developed by the Core Network Evolution team of Telefonica R&D. This PCE will be used against the latest Cisco OS development version containing the latest Segment Routing features in order to validate the use case with the company that originally defined the technology (Cisco).

This project will not only validate the idea behind the Segment Routing technology but will also leverage the use of Software Define Networking and Traffic Engineering. Once it is concluded we will be able to safely say that a new Traffic Engineering solution is finally among us.

1.2 Objectives

The following list summarizes the main objectives of this research project.

- Define the Segment Routing technology.
- Analyze the applicability of SR in different scenarios.
- Analyze the different use cases defined for the Path Computation Element in order to apply them to SR.
- Extend the PCE Protocol in order to carry Segment Routing.
- Study how such extensions carry Segment Routing information.
- Test Segment Routing with a Path Computation Element using PCEP.
- Implement the 3rd version of the BGP-LS draft.
- Develop the extensions to support optical parameters in the BGP-LS protocol.
- Validate the exportation of topology using BGP-LS.
- Present extensions to the BGP-LS protocol to make it compatible with SR.

1.3 Document Structure

This memory studies the possible applications of Segment Routing in a Software Defined Networking architecture using a centralized TE controller, the Path Computation Element (PCE).

The project is divided into six chapters which collect all the necessary information to make the proof of concept. The contents of each chapter present the different functional elements that are necessary to implement such technology.

Every chapter represents a key element in our architecture. First of all, Segment Routing is presented as a whole with the most up-to-date information about the technology. Everything written about it is contrasted with the latest drafts and articles which are made available by the design leaders of this technology.

In the first place, chapter 2 describes the Segment Routing technology in a generalized manner making special emphasis on those characteristics that will be addressed throughout this project.

In chapter 3, the Path Computation Element is studied in detail paying especial attention to those features that make our implementation possible.

Chapter 4 describes the BGP-LS protocol included in our Path Computation Element. We consider that this protocol will be of great importance in the days to come as it opens many possibilities in end-to-end path computation tasks. Together with the study, working code is tested validating the use of this protocol.

Chapter 5 proofs the use of PCEP with SR extensions in order to exploit the use of a Segment Routing capable PCE. The extensions are explained in detail and a working PCE is tested against SR-capable Cisco routers to test the use of such protocol extensions.

Finally, Chapter 6 concludes this work. The annexes attached at the end contain detailed information about how to duplicate the tests carried out in this project.

2 Segment Routing

2.1 Introduction: Historic network evolution

2.1.1 The early days

Back at the beginning, the network was created based on a series of assumptions that, as long as they were not violated frequently, proved to be greatly efficient. Some of the main assumptions made by the classic telephone company included:

- Rare and expensive infrastructure should be shared to offer low priced premium services.
- Human voice generates the majority of the traffic.
- Circuit-switched technologies are the *alpha dog* of the ICT industry.
- The telephone company is in control of its network.

Many things have changed since those days. Voice is no longer the predominant source of traffic. Data in all its different forms is now the main issue and the classic telephone network has switched to a scenario where many different technologies coexist. In addition, Internet has made the details of network operation irrelevant and therefore the control of it has been transferred mostly to the end user.

The origin of the so call Intelligent Network resided in the four points mentioned above. The goal was not customer service but the development of some new features to encourage vendor independence, better automation, and some new 'intelligent' services into the existing network architecture.

Intelligent Network specs tried to push vendors into interoperating, designing their equipment to work in a multi-vendor environment. Consequently, the freshly engineered products were able to adapt to the business systems of certain customers, but only through restricted and carefully designed interfaces. An example of a classic Intelligent Network service was giving the caller different options during a phone call (e.g. "Please push one if you are calling from a mobile phone"). All these new 'intelligent' features were, in theory, meant to encourage new business opportunities by opening new roads to meet customer needs.

The main drawback to this approach was the great wall that separated the possible business opportunity from the entrepreneur trying to address it. This wall was the telephone company who owned the network. In order for an idea to get implemented, it had to get through all the different filters a company has (getting the attention from the decision makers, getting approval from the business case study, establishing the developing plan, budget…) which constitute a mayor hold back almost impossible to surpass.

When the Internet made it finally to the homes of most of us, the scenario changed drastically. The telephone companies lost their design hegemony and those barriers mentioned above started to collapse rapidly. As the Internet Protocol works at the level where the user software controls the session it puts the company out of play. The Stupid Network described by Isenberg was finally here [2].

2.1.2 The rise of the Stupid Network

Internet broke the operator control over the network and shifted management to the end user by making the underlying network layers opaque. In this way, anyone could write an end-user application to make use of those business opportunities that previously had to go through the company's bureaucracy.

The *dumb network* allowed you to stream your bits at one end independently of the type of traffic without getting caught by the company's legislation. Consequently, the companies realized that it was no longer profitable to continue to invest in scarce, costly, network gear as the above mentioned assumptions no longer held. In addition, bandwidth became not an issue anymore with fibre backbone bandwidth increasing exponentially throughout the century.

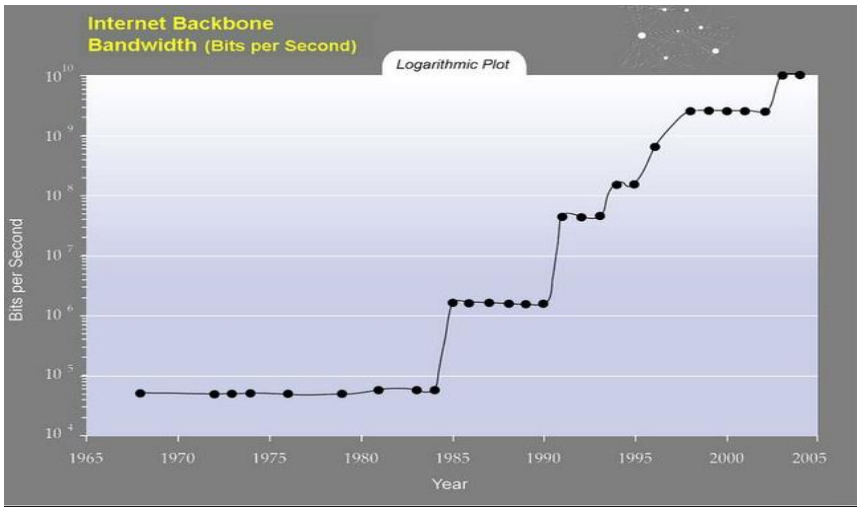


Figure 1: Backbone bandwidth growth [3]

Centralized control was quickly replaced by end-user intelligence making the network become just the vehicle to deliver the bits. Furthermore, the network was intended to be as simple as possible making use of routing algorithms that took as little computational cost as possible.

The network would be engineered simply to "Deliver the Bits, Stupid" not for fancy network routing or "smart" number translation. [2]

Essentially, data tells the network its destination not the other way around. Instead of sophisticated network routing mechanisms, intelligent end-user devices are connected to high bandwidth access networks constantly checking for packages directed to their area of influence.

This almost gets us to our case of study but we still have one more network evolution step to tell.

2.1.3 Intelligent networking v2.0

We have seen that network complexity was shifted from the network itself to the endpoints as "plentiful bandwidth" and "high computational capacity" became the primary engineering premises for the communication world.

Somehow the perception that the network is becoming too complex again in today's scenario is gaining more and more adepts. Therefore it is time to examine the current network state that has to be maintained once again and to try and turn the complexity in yet another direction.

At some point engineers started to search for technics that allowed them to fully characterize data traffic and treat each stream of bits in a customized manner to fully utilize the network features. As so, we no longer have a Stupid Network that only delivers the bits to the end user but a Stupid Network that responds to the intelligent endpoints adding special treatment to fulfil the different business cases.

Nowadays we rely on the network to provide high efficiency, manage link utilization, resist to DDOS attacks, apply selective performance and provide with high availability just to name a couple of examples. All of these features are very difficult to obtain in pure IP networks. Therefore technologies such as MPLS or GMPLS gained great value within the networking world. However, as expected, it comes with a cost. That is, the requirement to maintain more and more network states in a much more complex control plain.

And, finally we have reached our question of interest. Is there a way so we could reduce the network complexity and somehow maintain the so many control plain protocols? Yes, Segment Routing!

2.2 What is Segment Routing?

Segment Routing enables any node to select any path (explicit or derived from Interior Gateway Protocol (IGP) Shortest Path Tree (SPT) computations) for each of its traffic classes [4]. This path does not depend on any specific per-hop signalling technic. In addition neither Level Distribution Protocol (LDP) nor Resource Reservation Protocol with Traffic Engineering extensions (RSVP-TE) is used for distribution purposes. It depends on a set of segments that are advertised through the correspondent IGP routing protocol (OSPF/IS-IS) which are later combined together to create the desired path to the destination.

Two types of segments are defined: node and adjacency segments. A node segment is a path to a node while an adjacency segment represents a one-hop path to the target. SR's control plane is fully compatible with both IPv6 and MPLS data planes. For instance, a node segment to node 'A' would be translated as an LSP through the shortest-path to the node while an adjacency would be a cross-connect entry pointing to a specific point of egress. Only three types of operations are defined: push, continue (swap) and next (pop). We will further illustrate a few examples to show how they are used.

The main goal of SR is to make things easier for operators by improving scalability and simplifying network operations. As it is thought to be perfectly compatible with the MPLS data plane, maintaining its existing label structure, it provides excellent leverage over all the services supported in MPLS (explicit routing, FRR, VPNv4/6, etc.).

Operators have been asking for a drastic improvement in routing simplicity and number of protocol interactions. The point they make is that there are currently too many protocols that exchange network state and consequently the data bases that maintain it become far too costly. SR addresses this issue by avoiding the use of LDP or RSVP-TE for label distribution. Consequently we safe a huge amount of labels in the LDP database, TE LSPs in the network and tunnels to configure.

To sum up, Segment Routing brings simple (less protocols), more scalable (less states in the network, less labels kept in the router and less tunnel to maintain) and highly responsive (no waiting for new path signalling and programmability) way of networking.

However, these advantages come with some drawbacks. Without network state the network can be more difficult to troubleshoot, there are legacy services working with current protocols and this responsiveness is not a critical issue for current services. Besides, path establishment can be a benefit for TE as the resources are reserved.

2.3 Segment Routing Architecture

Let's describe the Segment Routing mode of operation using Figure 2 as the reference. In this example we have an ingress node I and an egress node E corresponding to a certain autonomous system (AS1). Nodes A, B, C, D and F are all intra-domain nodes of AS1.

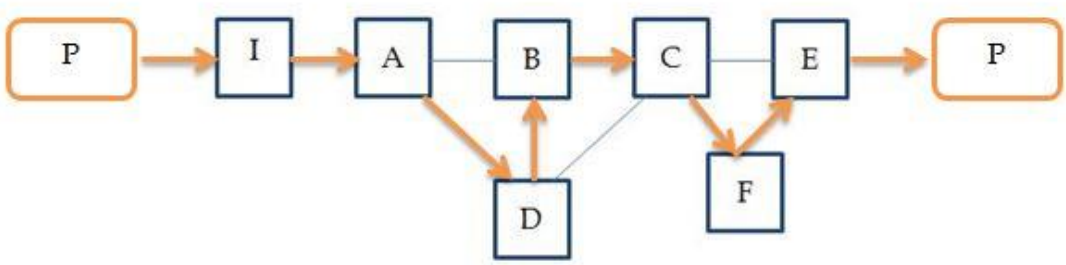


Figure 2: Example of network scenario

Consider a packet P entering the network at ingress node I. Now assume we set a few constraints on how that packet should be treated:

- Node B must apply a local service S to packet P.
- Links AB and CE must not be used to transport P.
- As a security measure any external packet coming from outside the operator's domain can be dropped at ingress.
- Any node along the traversed path of packet P must be capable of determining the ingress and egress nodes of the SR domain.

By adding a simple SR abstract header with the adequate labels all these properties can be achieved. The header contains the desired path encoded as a set of segments, a pointer and the ingress and egress Segment Routing edge nodes.

Figure 3: Packet P at ingress node I

A segment identifier (SID) is a 4 byte number that stands for either a topological instruction or a service instruction. This SID can be either local (adjacency segment) or global (node segment). A local SID is interpreted only by the node that originated it while a global SID is executed by any node in the domain (assuming SR-capable).

In Figure 3 node SIDs (global) SD, SB, SF, SI and SE represent the shortest path to the corresponding node while adjacency SID (local) SSB identifies a local service that must be provided by B. All segments have been previously flooded through an IGP protocol (OSPF or ISIS) and saved in the corresponding SR forwarding table of each node.

With all this in mind, let us illustrate the scenario described in Figure 2: Example of network scenario and how packet P is routed. At ingress, node 'I' pushes the header shown in Figure 3 and sets the pointer to the first tag (SD). SD belongs to the forwarding table of all the nodes in the SR domain and makes packet P follow the shortest path to D. The same thing can be said about SB, SF and SE respectively. At each node the pointer is incremented so it is set to the following tag (instruction) to be executed by the next element. For example, node 'A' would execute SD and update the pointer to SB (pop SD) so that 'D' forwards the packet as desired. Each of the following nodes would do the same in their corresponding context.

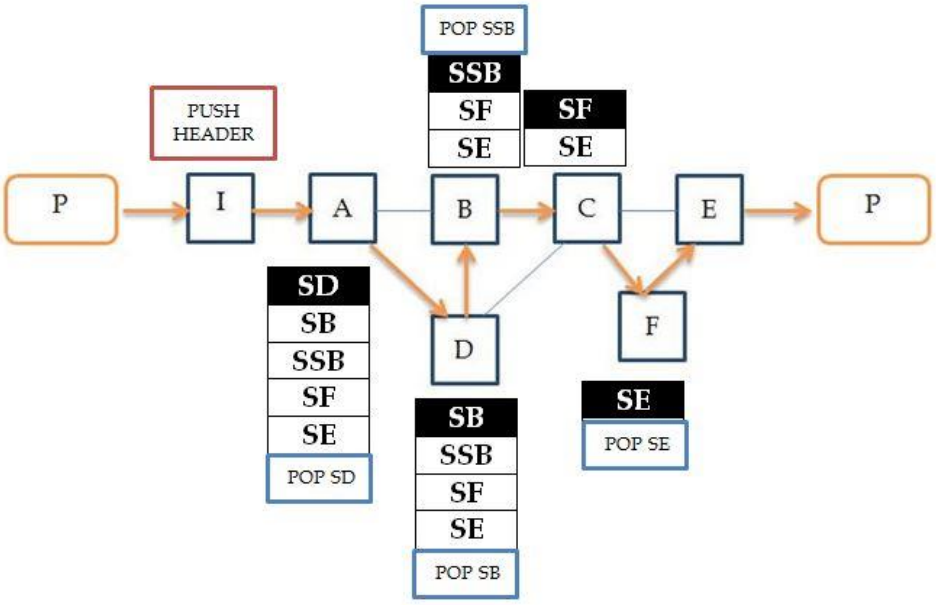


Figure 4: Label switching in SR

Tag SSB represents a local SID to be interpreted only by B. Adjacency SIDs are usually utilized to force a packet through a path different to that announced by OSPF enhancing network programmability. Finally, at egress the packet no longer is preceded by the SR header and can, therefore, continue its way to its final destination without any constraint.

As we have seen all the initial constraints (see page 6) on our proposed case of study have been fulfilled using Segment Routing. In addition, only node 'I' stores *per-flow* state of packet P (e.g. how to route packet P of traffic class T to the destination). Intermediate nodes only save states of the corresponding labels distributed through IGP (local and global) improving scalability. Each node stores "N" (node SID) + "A" (adjacency SID) entries in its Forwarding Information Base (FIB) in comparison with an order N^2 if RSVP-TE was used.

Now, let's imagine we do not push a label per-hop as in the example above but instead we just push SE at ingress. As we already know Segment Routing only defines three types of operations (push, swap and pop). The swap operation leaves the header unchanged if the operation cannot be completed (e.g. the packet is not delivered to 'E'). In addition, as previously mentioned the definition of a node SID is the shortest path to the node. Consequently a packet with just SE in its SR header will follow the shortest path to 'E' (e.g. I-A-B-C-E).

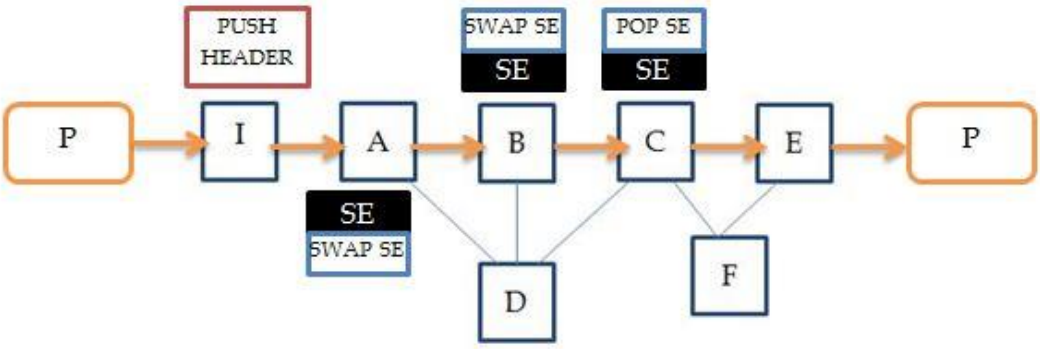


Figure 5: Example of Shortest Path Routing in SR

To finish the section, an example of how a SR database looks like is shown in Figure 6. The instruction associated with each SID contains at least the next-hop to be followed by the packet and the corresponding SR operation to be performed by the node. Each SR node maintains its own database. Its entries can be derived from a local configuration or via IGP advertisements.

Segment	Next-Hop	SR Header operation
Sk	M	CONTINUE
Sj	N	NEXT
Sl	NAT Srvc	NEXT
Sm	FW srvc	NEXT
Sn	Q	NEXT
etc.	etc.	etc.

Figure 6: SR database [5]

2.4 Segment Routing use cases

Next, a couple of Segment Routing use cases will be shown. The current use cases defined by the IETF can be found in the [filsfils-spring-segment-routing-use-cases-00] draft.

2.4.1 Class of Service based Traffic Engineering

A very common approach in the traffic engineering world is the definition of per-flow Class of Service (CoS) routing policies. In other words, different classes of service need different path characteristics. Usually the two common parameters used by traffic engineers to determine a routing policy are bandwidth (BW) and latency (usually round-trip time).

Figure 7: CoS-based TE example

In the example above, let us say there are paths to route traffic from Barcelona to Lisbon, via Madrid or via Seville. The first one has cheap and plentiful network capacity while the second one has higher cost but offers premium low latency service.

In this case IGP metrics would be tuned in order to carry most of the load via Madrid and this would be perfectly fine for most common applications. However, it may not fulfil the more demanding real time applications such as VoIP. In this case the operator would, most certainly, want to separate both flows (e.g. data and VoIP) to be able to provide for its customers.

In order to do this the operator would configure:

- The IGP metric such that the shortest path from Barcelona to Lisbon is via Madrid.
- Seville's core routers to announce its corresponding SID (extensible to larger networks: any-cast SID[1]).
- The IGP metric such that the traffic received by Seville's core routers is sent via the shortest path to Lisbon using the premium low latency channel.

With these prerequisites in mind, the operator would configure the following policies to its router in Barcelona for the traffic directed to Lisbon in order to apply successful CoS traffic engineering:

[1] **Any-cast SID**: A SID which does not identify a specific router but a set of routers. The packet would be received by the closest router part of the any-cast SID and then it will be sent to the destination via the ECMP-aware shortest path.

- VoIP traffic: SIDs {999, 600}
- Data traffic: SID {600}

SID 999 would route high quality voice traffic through Seville which will then follow the configured SP to Lisbon hence fulfilling the low latency requirements. All other data will be carried through the ECMP-aware shortest path to Lisbon (via Madrid).

This SR technic would provide the desired traffic engineering behaviour while at the same time maintaining simplicity and enhancing resiliency.

- Zero per-flow state and signalling at intermediate and egress nodes
- Traffic engineering policy would not be attached to a particular core node eliminating the single point of failure scenario.

2.4.2 SDN controller use in Segment Routing

This use case shows the main application scenario of SR to Software Defined Networking. Its implementation depends directly on the SDN controller that could perfectly be a classic path computation element (PCE). In this example the PCE is responsible of accepting or denying new flows and how to route them. In addition, it monitors the topology looking for possible problematic situations like the one represented where a congestion issue has to be solved.

Let us assume that all the labels have been flooded into the network and collected in the SR database. By default any flow with destination D is admitted and set to traverse the network following the shortest path to D (e.g. ABD) by pushing label 101 at ingress.

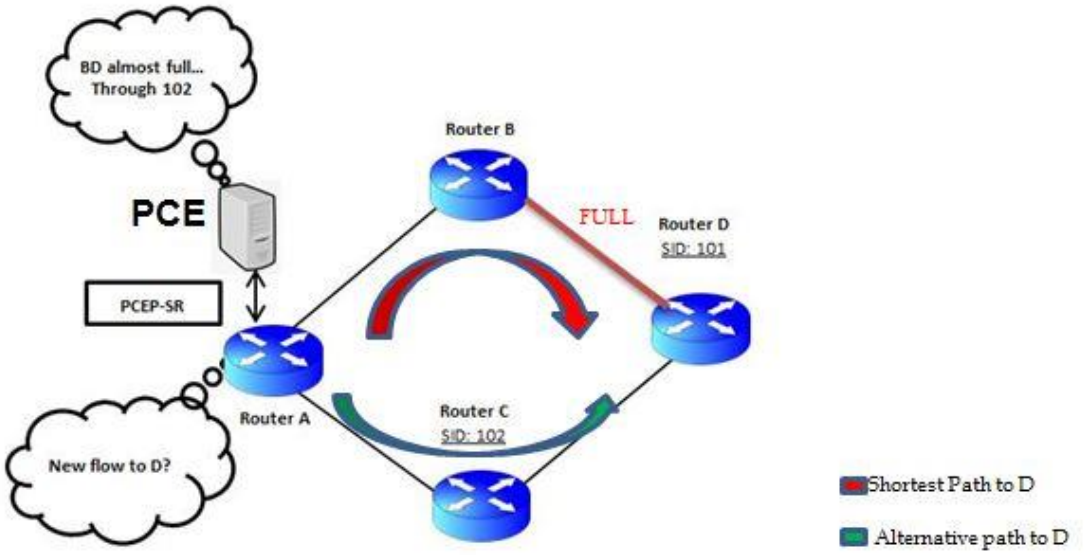


Figure 8: SDN use case example

Now, our SDN controller which is constantly monitoring the SR database as shown in Figure 8, detects a possible congestion risk due to limited available capacity in link BD. Automatically it will set up a SR-policy where any path computation request to D will be re-routed through C hence pushing {102, 101} as the SR header.

This use case will be our main focus of interest and the one we will implement later on in the document as we believe it brings important benefits that could be highly useful in future network development. Some of them are:

- Explicit source routing capability with no per-hop signalling
- Highly responsive network as the SDN controller can apply any policy by just pushing de adequate header.
- Simple midpoints: no further complexity is introduced at the intermediate nodes as all the computation is performed by the SDN.
- The state is only maintained at ingress (PCE).
- It makes use of the capabilities offered by protocols such as BGPLS, PCEP or OPENFLOW.

We will further analyse the different protocol extensions that are necessary in order to leverage from Segment Routing.

3 Path Computation Element

IETF: "*a PCE is an entity that is capable of computing a network path or route based on a network graph, and of applying computational constraints during the computation*" [6].

This chapter is going to introduce the different types of Path Computation Elements and their possible utilization scenarios.

Traditionally backbone networks where managed in a centralized way, the network elements where configured statically. Lately, large strides have been made in order to create a common control plane by standardizing GMPLS which allows a dynamic and distributed configuration of the optical layer. Nevertheless, path computation in optical networks is a complex task due to the additional constraints that optical network elements present. If such task is left to the GMPLS controllers, these controllers must be provided with the sufficient computational capacity making the network increasingly expensive.

This is where the Path Computation Element gets into consideration. The main objective is to free up network resources by moving the tough path computation tasks away from the nodes. In addition, a PCE could apply modern traffic engineering technics to constitute the desired path. These technics could be looked upon as the way of monitoring and manage the network behaviour to provide optimum quality of service.

The PCE is defined as a network entity that contains topology information and is consulted by the different network nodes to determine the path to be followed by a packet from ingress to destination. At the same time, it uses abstract information about the domain to work out the optimum path to destination.

In all, the PCE eliminates the necessity of computing routes inside the nodes and, consequently, reducing its cost.

3.1 The PCE Architecture

In Figure 9, the basic principle of the PCE architecture is shown. This model includes, at least, one PCE per domain. Nevertheless, a domain could have multiple PCEs to help with load balance and single point of failure prevention. The PCE receives path computation requests from the Path Computation Clients (PCC). To attend them, it needs to maintain up-to-date

information about the state of the network. This information is saved in the Traffic Engineering Database and updated periodically.

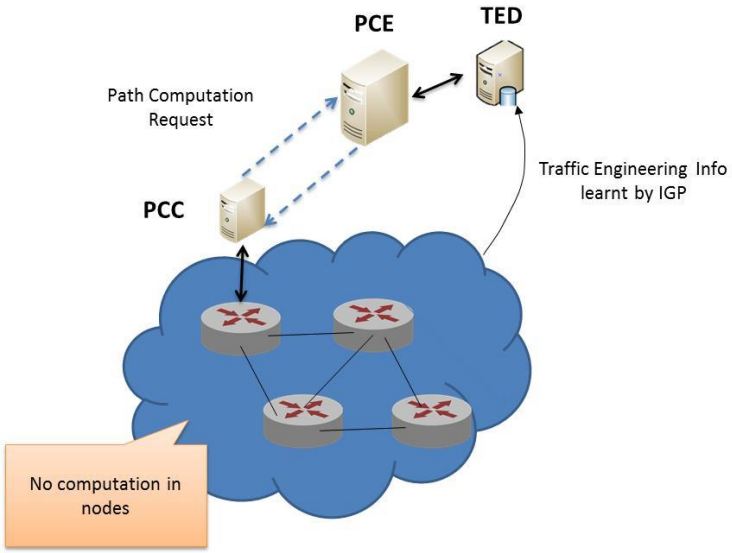


Figure 9: Basic PCE architecture

Such architecture provides with the necessary functionality to calculate the optimum path using traffic engineering technics. Furthermore, it liberates the nodes from heavy computational tasks, while enhancing security and reliability.

Being a centralized element, a PCE is capable of calculating optimum paths not only locally (inside its domain) but also in collaboration with other PCEs to determine the optimum inter-domain route in multi-domain scenarios. This characteristic is the key to finding a global optimal solution between different administrative groups as it is known that merging locally optimal solutions will derive in a sub-optimal one when put all together.

Using Constrained Shortest Path First is a paramount characteristic for traffic engineering in MPLS networks. This process is extremely consuming when dealing with large multi-domain networks and it requires the use of special computational equipment together with inter-domain element collaboration.

3.2 Use Cases

There exist different situations where it is appropriate to use a PCE-based architecture. This does not obsolete other technics, it simply highlights certain scenarios where using a PCE could be extremely beneficial. Some of these are:

- **High resource consumption:** It is possible that a route calculation is so consuming that the Label Switch Router (LSR) does not have the necessary resources to deal with it. In this case, a PCE could be of great use.
- **Limited visibility:** There are many situations where an LSR does not have the minimum information to set up a path to the target. This could possibly be a multi-domain scenario where the edge router does not have access to the other domains' information. This brings us to the situation of having multiple PCEs collaborating with each other to determine the route.
- **TED absence:** Maintaining a traffic engineering database can require high memory and resource consumption as multiple threads would be interacting at the same time. At the same time there might be a situation where the intermediate nodes do not support traffic engineering for the different network protocols. In this case it would be necessary that a PCE was supplying this TE information having its own TED.
- **Control plane absence or routing capability in a network element:** Many times in optical networks it is common that a network element does not have a control plane or is not routing capable. In such cases all its connections are handled by the managing plane. A PCC would interact with the PCE to establish the desired route.
- **Computing alternate paths:** A PCE can be configured to calculate back-up paths for security reasons in case there is a failure in the network.

3.3 Workflow of the basic PCE model

Once the basic architecture has been explained, let us show how an LSP calculation takes place. In Figure 10 a basic TE LSP calculation sequence is shown:

1. A new traffic flow arrives at a certain domain (usually MPLS) with destination D. The LER (Label Edge Router) must then initiate a path computation request with the desired traffic engineering restrictions to D.
2. In our scenario the LER behaves as a PCC. As such it asks the PCE for the best possible route inside its domain. They use PCEP to communicate between them.
3. The PCE can calculate the path by itself or in collaboration with other PCEs belonging to the same domain. To do that it checks

whether the requested TE constraints can be fulfilled based on the information stored in its TED and its local policies or not.

4. The PCE uses a local computational algorithm. This is totally up to the network administrator and there is no standard that addresses it.
5. Finally, when the path is calculated, the PCE notifies the PCC and the latter creates the new route using the configured reservation protocol (e.g. RSVP-TE)

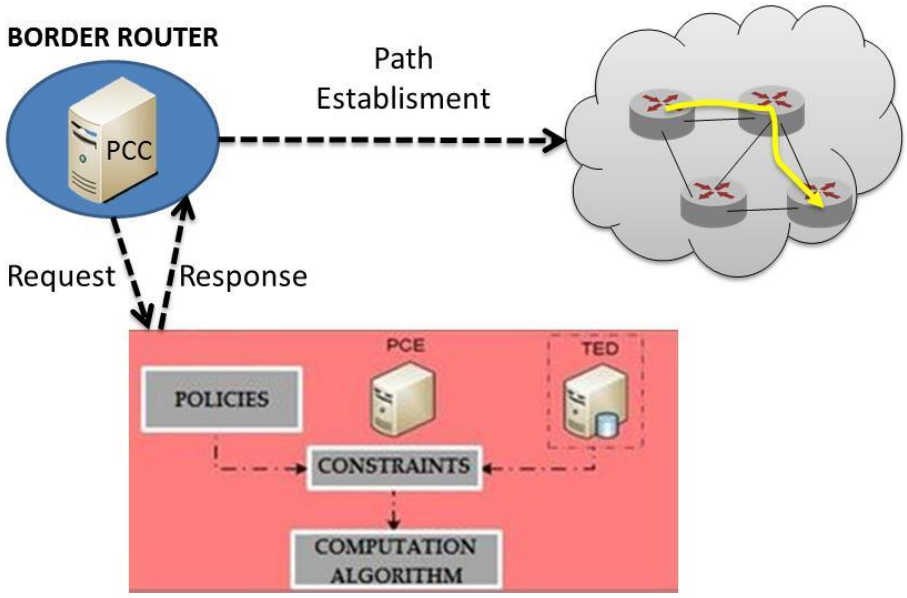


Figure 10: Functional description of a PCE based architecture

A PCE based architecture works based on a computational model that can be, either, distributed or centralized. In a centralized model, a single PCE is in charge of dealing with the requests from all the PCCs of its domain. This carries the additional risk of having a single point of failure, so an appropriate resiliency measured would be to set up a back-up PCE in case the first one fails.

In a distributed model several PCEs can participate in the path computation tasks either by calculating different path segments or by interacting between them to find the optimal solution. This model allows a PCC to initiate a request to a certain PCE but if this PCE is not capable of providing a complete path to destination a second PCE may be addressed to provide it. This can be the case where the returned path is either a segment or a *loose path.*

It is worth mentioning that a PCC does not distinguish between a centralized and a distributed model with inter-PCE communication. The PCC simply sends a path computation request and receives a complete path or a segment of it, without knowing if several PCEs have been involved in the process.

Many times, certain services may require the calculation of several routes for the same flow (e.g. load balance). If this is the case, the PCC has two options:

- To send multiple individual path requests to a PCE. In this case the petitions would not be synchronized and different individual routes would be created.
- To send a single petition to a PCE requesting several paths in a synchronized or unsynchronized manner. The PCE will then, perform simultaneous or individual computation of the set of desired paths.

3.4 Hierarchical PCE architecture: Inter-domain path computation

Using a PCE to compute a path between nodes belonging to the same domain is pretty simple and is mainly described above. Calculation of an end-to-end route when the ingress and egress nodes belong to different domains requires co-operation between multiple PCEs, each having jurisdiction in its own domain.

The model that we are going to describe in this section is the Hierarchical PCE (H-PCE) implementation as it is the one with which we will be working with later on in our simulations. This method is oriented to work with small collections of domains never with large multi-domain networks like the internet.

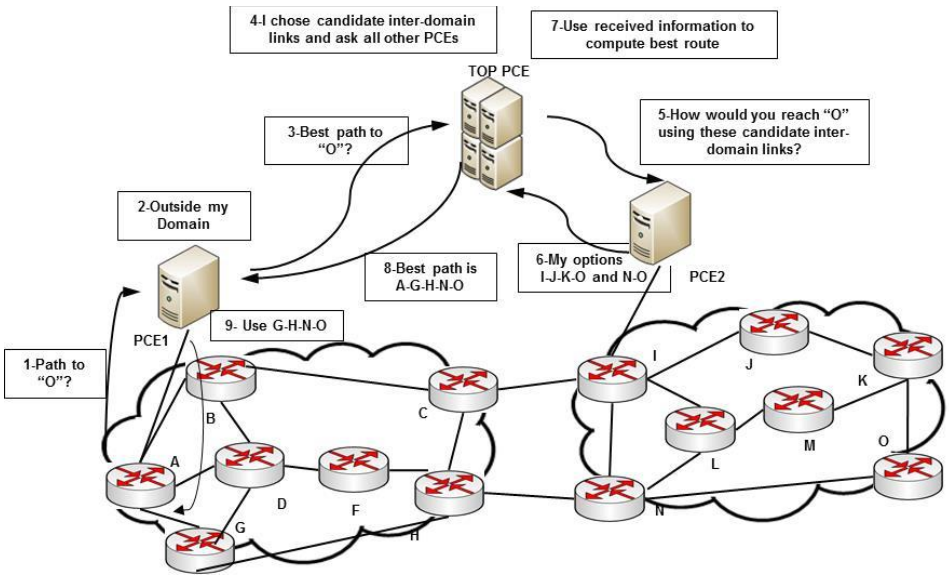


Figure 11: H-PCE scenario example

In Figure 11 we can see a simple H-PCE scenario. In this example, a parent PCE (p-PCE) holds a map of the multi-domain topology where the

nodes represent the two different domains and the connections are the different inter-domain links that exist. The p-PCE (PCE 5) does not have any intra-domain information of any of the child domains; the parent does not have any knowledge about resource availability or link state in the above shown domains. The only info it has is the TE information of the interconnections between the domains as they belong to its jurisdiction.

To preserve confidentiality of the different Autonomous Systems (AS), the p-PCE does not contain any resource information about the links or the nodes inside the child domains. Furthermore, each child PCE (c-PCE) only has awareness of the topology within its own domain and the links that connect there domain to their neighbour.

Each c-PCE must be configured with the IP of its p-PCE. There may be more than one parent. At the same time a p-PCE needs to know about each of the c-PCE of all the domains in its topology. This information can be configured either statically or dynamically.

The basic workflow in an H-PCE is as follows:

1. At ingress, the c-PCE sends a path computation request to the parent using the Path Computation Element Protocol (PCEP)
2. The parent then selects a set of possible domain paths based on the state of the interconnections between domains.
3. The p-PCE sends path requests for the c-PCE responsible for each of the candidate inter-domain paths.
4. Each child selects a set of candidate per-hop paths through its domain and forwards them to the parent.
5. The parent concatenates the received candidate paths to create the optimal end-to-end route which is forwarded to the c-PCE that initiated the request which passes it to the ingress node (PCC).

This procedure based on a hierarchical architecture of the PCEs requires the use of certain PCEP extensions to work that will be detailed further on.

3.5 Stateful, Stateless and Active PCEs

A PCE can be either stateless or stateful, and active or passive. Let us summarize how each of them is defined and their possible application scenario.

3.5.1 Stateless PCE

A stateless PCE does not have any information about the LSPs in the network so its function is limited to dealing with the path requests but without any control over the set of paths already created. Each petition is processed

independently of each other as no LSP information is maintained in the PCE's database.

A stateless PCE computes paths based on the current state of the TE database which could be out of date with respect to the actual network state. This is why a PCC can include a set of previously computed paths in the request in order for the PCE to take them into consideration. Updating the TED is done through the link state information carried in the different IGP messages or by some other set of protocols (e.g. BGP-LS).

3.5.2 Stateful PCE

A stateful PCE, on the other hand, has a complete knowledge about, not only the current network state (resources and topology information), but also the set of already established LSPs and available resources in the network. This PCE uses traffic engineering information together with the set of computed paths in the network when dealing with a new request.

This model may deliver the optimum solution but it requires huge amount of pre-computation and a reliable synchronization mechanism. The latter can be non-trivial if more than one PCE is involved in the path computation tasks as each PCE should notify the rest when a new LSP is created and how many resources have been allocated. This could derive in a more complex control plane and significant overhead.

The main application scenario of this type of PCE is that in which we have a single centralized environment, one PCE processing all the path computation requests. In such a situation maintaining an LSP database is easier as all the paths are being managed by that PCE and no PCE-PCE signalling is necessary.

3.5.3 Active Stateful PCE

An active stateful PCE is an extension of Passive Stateful PCE, in which the PCE is given the capability of issuing recommendations to the network. It gives the PCC the possibility of delegating management of certain LSPs to an active stateful PCE giving the latter the power to update LSP parameters in those LSPs that were delegated to it.

In other words, the main advantage of having an active stateful PCE is the possibility to take control of the LSPs. Such ability includes the capacity to re-optimize and restore previously torn down LSPs, create additional protection or establish priorities. In addition, it allows the stateful PCE to actively instantiate a path request to the PCC using the Path Computation Initiate (PCInit) message.

Even though LSP management is delegated to the PCE, the PCC remains in full control of its LSPs as it may revoke this delegation at any time during the lifetime of the LSP. The PCC may revoke this delegation issuing a notification message to the PCE in charge of the LSP at the time. Furthermore, a PCE may return an LSP delegation at any point during the lifetime of the PCEP session.

3.6 Path Computation Element Protocol (PCEP)

PCEP is a protocol based in the request/response model and utilized for the communication between a PCC and a PCE or between two PCEs.

It operates over the transport protocol TCP using client-server sessions. In order to perform in such a way it uses seven different types of messages: *Open, Request, Reply, Keepalive, Notification, Error* and *Close.* A PCC can establish multiple PCEP sessions with different PCEs the same way a PCE can open a session with multiple PCCs but only one PCEP session can be up at a given time between PCEP peers.

The different possible states in a PCEP session are as follows:

- **Initialization stage:** There are two consecutive steps that conform this stage:
  - TCP connection establishment between the PCEP peers.
  - PCEP session creation over the previously set up TCP connection.

  Once the TCP session is up both peers negotiate different parameters to configure the PCEP session. These parameters are sent in the *Open* message and they include timers *Keepalive* and *Deadtimer,* together with some additional information that determines the conditions under which a *Request* message must be sent to the PCE.
- **Keepalive session:** when a session is established, both PCEP peers need to know if the other side of the communication is still up and running. They use two timers for this purpose:
  - *Keepalive:* Every time a PCEP message is received this timer resets. In case it expires, the PCEP peer sends a *keepalive* message to keep the session up.
  - *Deadtimer:* If a PCEP message is not received before this timer expires the session is considered as dead and the connection is closed.
- **Path Computation Request (PCReq):** When a PCEP session is up and a new flow arrives at ingress, the PCC can send a Path

Computation Request (PCReq) to the PCE to route the packet to destination. A PCReq message is identified by a petition id number and contains several attributes that are used to perform the path computation. Among other things, the elements contained in a PCReq message are the origin and destination IP addresses the requested bandwidth or the priority. Furthermore, a PCC may include several requests inside the same message the same way a PCE may provide with several paths in a response.

- **Path Computation Reply (PCRep):** When a request is received, the PCE uses its internal algorithms to resolve it, if possible, and provide with the optimum path. In case the computation is successful applying the desired TE constraints, the resulting path or set of paths are sent back to the PCC using a *Reply* message. If the path computation is not feasible a "No Path" is sent back to the PCC with the possibility of indicating which conditions could not be satisfied.
- **Ending of the PCEP session:** When one of the PCEP peers wants to terminate a session, it must send a *Close* message before closing the corresponding TCP connection.

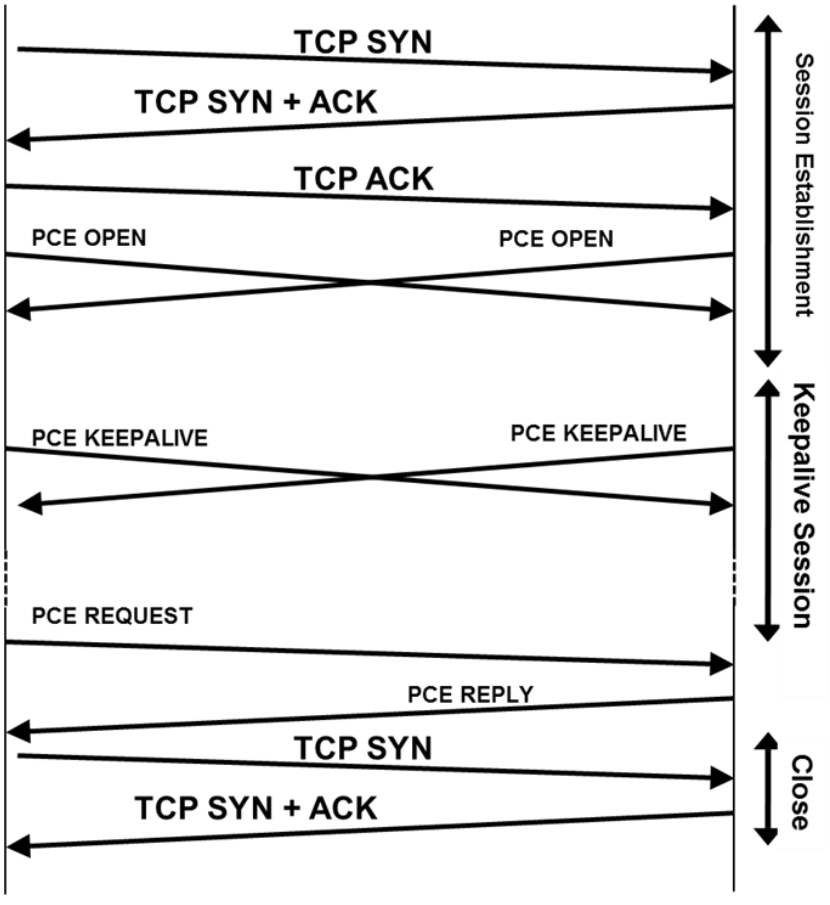


Figure 12: Generic PCEP session

It must be noted that a PCEP session can be either permanent or intermittent. A permanent session would require both peers to continuously exchange *Keepalive* messages to keep the connection up while an intermittent one would mean that a new connection is established every time a path computation operation is required. To choose one over the other would depend on the frequency of requests and scalability.

4 Border Gateway Protocol with Link State Capability

As we have seen throughout this document, the tendency is to transfer the computational load away from the network and into dedicated equipment (e.g. PCEs). These elements would be called upon to perform computations based on the current network state and topology taking into account the necessary traffic engineering constraints imposed by the network administrator. This information is usually flooded using IGP routing protocols, but IGP imposes restrictions on where to place these central entities.

This section will describe how to collect and distribute the necessary link state and traffic engineering information to external elements using the Border Gateway Protocol (BGP) with Link State (LS) extension. This is possible thanks to a new BGP Network Layer Reachability Information (NLRI) and the corresponding Link State attribute (BGP-LS attribute).

4.1 Application scenario

The contents of a traffic engineering database are defined to retrieve the network status. In some cases, optimization of end-to-end inter-domain paths could be drastically improved by having visibility outside the Autonomous System (AS). This visibility could be tuned to satisfy the different policies from the different network agents.

As we have seen in the previous section a Path Computation Element (PCE) is a way of achieving end-to-end computation of TE paths across multiple domains and requires coordinated action. The PCE needs to collect information about the network's characteristics in order to be able to provide with the adequate route.

A router keeps at least one database for storing link-state (LS) information about nodes and links in a given domain. If BGP is enabled, this node can collect this information and distribute it to its peers using the newly defined LS attributes. A diagram of this process can be seen in Figure 13.

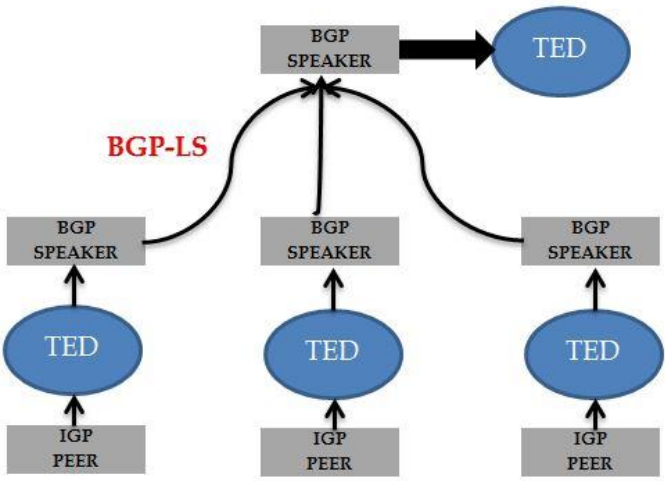


Figure 13: TE distribution with BGP-LS

In Figure 13 the BGP speakers exporting the topology could represent child PCEs (c-PCE) belonging to different domains while the one importing the information could be the parent (p-PCE) learning about the network. This would bring as back to the hierarchical model described in section 3.4 where the p-PCE is used to compute end-to-end paths through multiple domains.

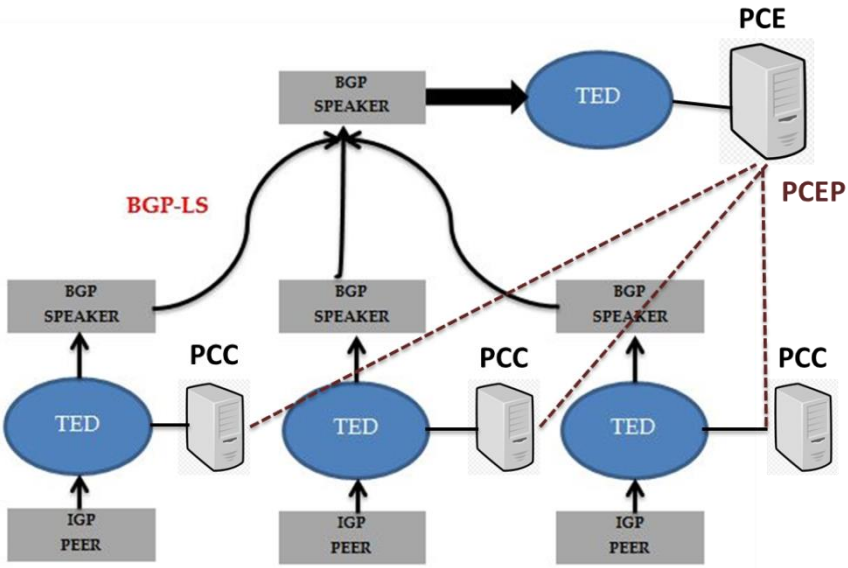


Figure 14: TE distribution using BGPLS in an H-PCE scenario

In this model, Figure 14, a c-PCE would provide enhanced computational power to provide optimal paths through its domain. As it has no visibility of the rest of the network, in case a packet arrives with a destination node outside its scope this c-PCE would send a path request to the p-PCE to obtain the needed set of hops. This is possible due to BGP extending its applicability to carry, not only reachability data, but LS information too.

Previously implemented solutions would use per-domain path computation as the way of solving the issue of having to compute a multi-domain path. This would mean that the router at the ingress domain would have to compute the optimum path for that domain, the same for the second domain and so on for the following hops. This usually derives in a sub-optimal solution as the network is not fully analysed.

4.2 BGP-LS protocol

There are two main modifications introduced in the Border Gateway Protocol (BGP) to carry link-state information:

- The appearance of a new BGP Network Layer Reachability Information (NLRI) that describes links, nodes and prefixes distributed through IGP.
- The definition of a new BGP path attribute (BGP-LS attribute) characterizing those elements (e.g. carrying TE properties).

There are 4 types of messages defined in the BGP protocol:

- *Open*: Used to establish a peering session.
- *Keepalive*: Handshake at regular interval to prevent the session from dying.
- *Notification:* Shuts down a peering session due to error.
- *Update:* Announces or withdraws new routes. Every announcement consists in a descriptor plus some attribute values.

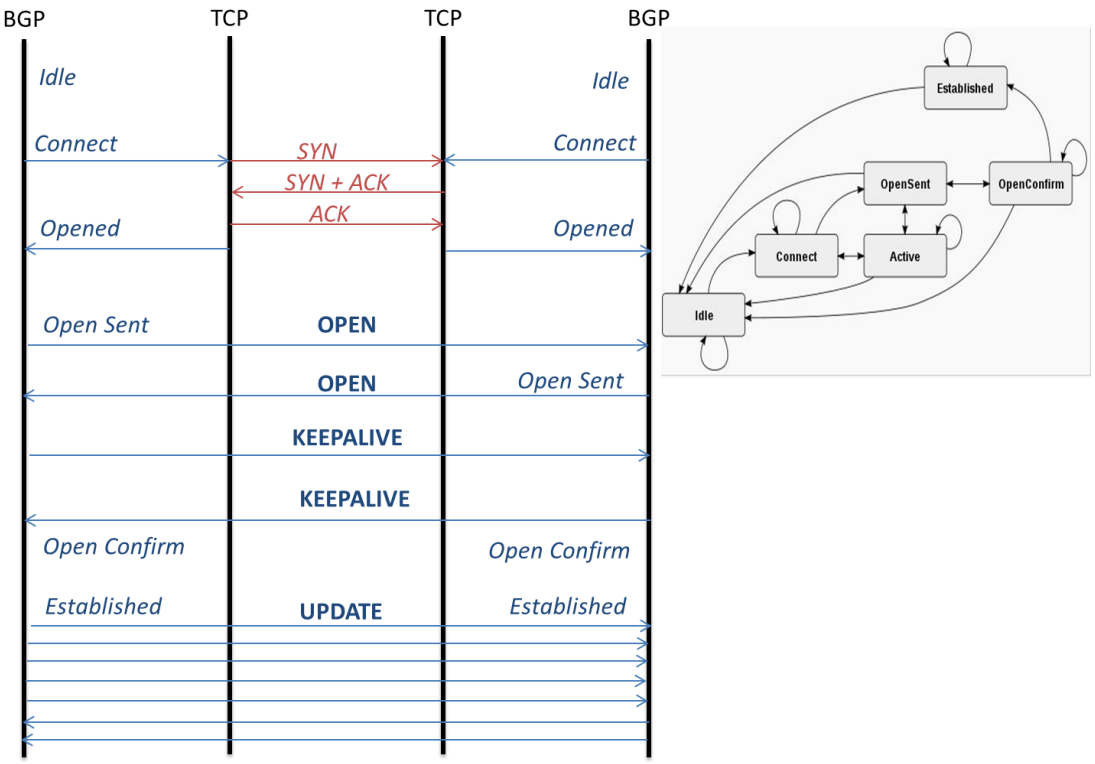


Figure 15: BGP message exchange and Finite State Machine

In order to make decisions in its operations with peers, a BGP peer uses a simple Finite State Machine (FSM) that consists of six states: Idle; Connect; Active; OpenSent; OpenConfirm; and Established. For each peer-to-peer session, a BGP implementation maintains a state variable that tracks which of these six states the session is in. The BGP defines the messages that each peer should exchange in order to change the session from one state to another.

All BGP peers transit through several states before becoming adjacent neighbours and exchanging routing information. During each of the states, the peers must send and receive messages, process message data, and initialize resources before proceeding to the next state. This process is known as the BGP *Finite-State Machine (FSM)*. If the process fails at any point, the session is torn down and the peers both transition back to an Idle state and begin the process again. Each time a session is torn down, all routes from the peer who is not up will be removed from the tables, which causes downtime. If configuration issues exist on one of the BGP peers, the peering routers continuously transition between unestablished states until the issue has been resolved.

The first state that a router enters when configured for BGP is the Idle state. In the Idle state, the BGP-speaking router refuses incoming BGP session requests. At this point, the router has not allocated any resources to the BGP process and does not do so until a BGP start event has either been initiated by the router's BGP process or by manual user intervention

The second state is "Connect". In the "Connect" state, the router waits for the TCP connection to complete and transitions to the "OpenSent" state if successful; after sending the Open message to its peers. If unsuccessful, it starts the ConnectRetry timer and transitions to the "Active" state upon expiration.

In the "Active" state, the router resets the ConnectRetry timer to zero and returns to the "Connect" state.

In the OpenSent state, the BGP peer waits for an OPEN message from its peer. After an OPEN message has been received, it is checked for validity. At this time, all fields in the OPEN message are checked against the local BGP configuration. Any fields that do not match the expected values cause an OPEN message error to occur. At this time, the BGP peer also checks to verify that a connection collision has not occurred. If the message is valid, the peer sends a KEEPALIVE message to its peer, sets the KEEPALIVE timer, sets the hold timer, and transitions to the OpenConfirm state.

In the OpenConfirm state, the local router is waiting for the receipt of a KEEPALIVE message from its peer. Upon receipt of a KEEPALIVE message, the BGP session transitions to the Established state.

BGP peers reach the Established state after they have successfully exchanged OPEN and KEEPALIVE messages. After the peers reach the Established state, they begin to send UPDATE messages containing routing information and KEEPALIVE messages to verify the TCP Connection state. If an error is encountered at any time while a peer is in the Established state, the local peer sends a NOTIFICATION message with the reason for the error and transitions back to the Idle state.

BGP-LS Update messages (Figure 16) are composed of path attributes (see Figure 16 in orange) containing the different BGP metrics. Path attributes can be classified as "*well-known*" or "*optional*". Well-known attributes must be recognized by all compliant implementations while optional are expected not to be recognized by all.

The new BGP-LS NLRI is included as a path attribute (MP_Reach attribute) that is a mandatory attribute in those messages that do not carry a classic NLRI (see Figure 16 in red). Together with the MP_Reach attribute a BGP_LS (LS) attribute may or not be present. The latter is considered an optional, non-transitive BGP attribute which means that it is not mandatory in all BGP Update messages and that it may not be advertised to other peers in the case of not having been recognized by the former.

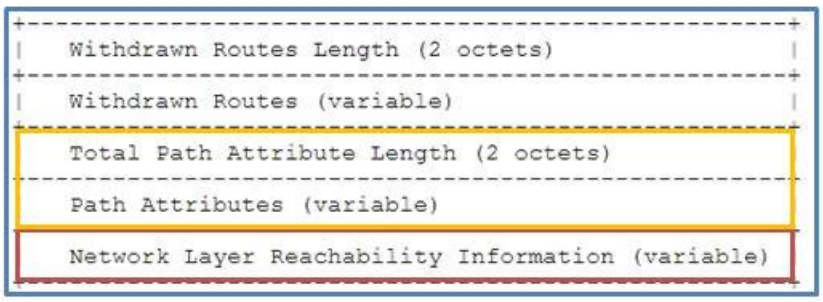


Figure 16: Update Message Format [7]

These two new BGP features will be described in detail in the next section as they are the key features of the new BGP extensions for Traffic Engineering and Link State distribution.

4.2.1 BGP-LS NLRI

Information in the new link-state NLRI is encoded as a triplet in a Type, Length and Value format (TLV). This format is shown Figure 17. The type field contains the code for the field that is going to be described in the value section. The length field represents the number of octets of the value field. Finally, the value section contains the descriptor of the element (node, link or prefix) that is going to be characterized. In addition to the descriptor field, the protocol through which the information has been learnt and an instance identifier are also carried.

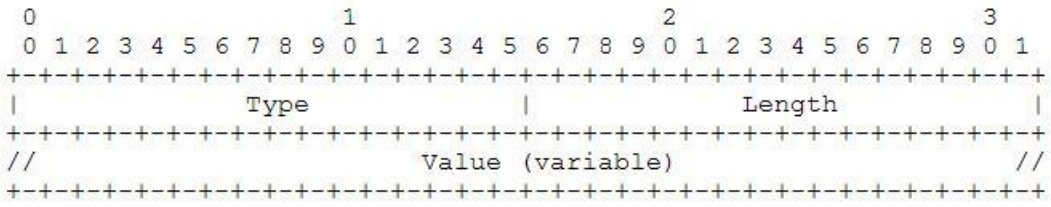


Figure 17: TLV format [8]

As stated before there are three types of descriptors: node (type 1), link (type 2) and prefix (types 3 and 4). Each descriptor is a TLV or set of TLVs, carrying the information of the type in question. For our purpose we are going to focus only on the first two: nodes and links.

4.2.1.1 *Node NLRI*

Type one belongs to a node NLRI. The value field of this NLRI type will contain a local node descriptor (type 256) field that defines a node (e.g. router) in the network. The node NLRI format can be seen in Figure 18.

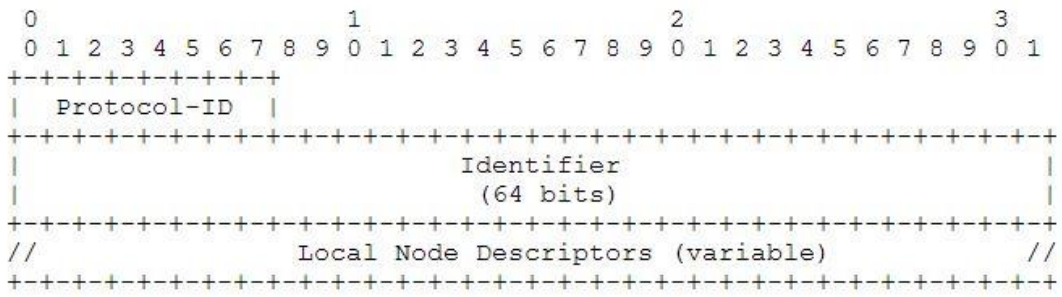


Figure 18: Node NLRI format [8]

4.2.1.2 *Link NLRI*

Type two represents a link in the network. It is described through a set of local node, remote node and link descriptors. The local (type 256) and remote node descriptors (257) are the extreme points and the link descriptor is the interconnection of both. An example of this NLRI type is shown in Figure 19.

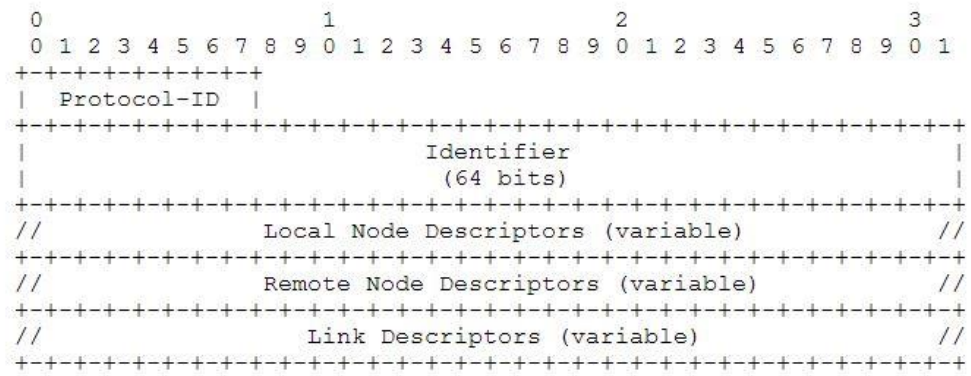


Figure 19: Link NLRI format [8]

4.2.1.3 *Descriptor types*

The two types of descriptors we are going to be concerned about are node and link descriptors. They both represent a TLV or set of TLVs carrying the necessary information to fully describe a node or a link. The amount of shared information will depend on the administrator's policies as all of these TLV are defined as fully optional.

4.2.1.3.1 Node descriptors

Each link needs to be characterized by a couple of uniquely identified nodes. For this purpose we have a globally unique IGP router ID carried inside the correspondent TLV inside a node descriptor. The problem resides when router IDs are assigned following private-IP numbering. In this case, we need additional fields such as the Autonomous System (AS) number or the BGP-LS identifier to individually name the node. A list of such descriptor and its TLV types can be seen in Figure 20. It must be noted that node descriptors are

carried inside the value field of the local/remote node descriptor container as it constitutes a TLV itself.

Sub-TLV Code Point	Description	Length
512	Autonomous System	4
513	BGP-LS Identifier	4
514	Area-ID	4
515	IGP Router-ID	Variable

Figure 20: Node descriptor sub-TLVs [8]

4.2.1.3.2 Link descriptors

The link descriptor field is a set of TLVs uniquely identifying a *unidirectional* connection between a pair of adjacent nodes. In other words, in order to fully characterize a bidirectional link two different Update messages would be needed.

There are two main ways of describing a link: Through its local and remote (neighbour) interface addresses or through unnumbered interfaces (code number 258) assuming that the IPs of both the local and remote nodes are carried in the node descriptors within the same NLRI. The set of up-to-date link descriptor sub-TLVs code points is shown in Figure 21. IPv6 and Multi-Topology technology are out of the scope of this study.

TLV Code Point	Description	IS-IS TLV /Sub-TLV	Value defined in:
258	Link Local/Remote Identifiers	22/4	[RFC5307]/1.1
259	IPv4 interface address	22/6	[RFC5305]/3.2
260	IPv4 neighbor address	22/8	[RFC5305]/3.3
261	IPv6 interface address	22/12	[RFC6119]/4.2
262	IPv6 neighbor address	22/13	[RFC6119]/4.3
263	Multi-Topology Identifier	---	Section 3.2.1.5

Figure 21: Link descriptors [8]

4.2.2 The Link-State attribute[2]

As stated in the section above, this optional BGP path attribute is used to carry the parameters that are necessary to fully characterize the elements described using the NLRIs (node or link) mentioned above. It is again a set of TLV triplets using the same format as that in Figure 17.

[2] The name BGP-LS attribute and LS attribute will be used indistinctively throughout this section.

For our purpose, we are going to focus on node attributes and link attributes.

4.2.2.1 *Node attributes*

Node attribute TLVs may be included in the BGP-LS attribute accompanying a node NLRI. The following set of node attributes are defined by the IETF but most of them are left unused for the purpose of this document:

TLV Code Point	Description	Length	Value defined in:
263	Multi-Topology Identifier	variable	Section 3.2.1.5
1024	Node Flag Bits	1	Section 3.3.1.1
1025	Opaque Node Properties	variable	Section 3.3.1.5
1026	Node Name	variable	Section 3.3.1.3
1027	IS-IS Area Identifier	variable	Section 3.3.1.2
1028	IPv4 Router-ID of Local Node	4	[RFC5305]/4.3
1029	IPv6 Router-ID of Local Node	16	[RFC6119]/4.1

Figure 22: Node Attributes [8]

The main TLV we are going to pay attention to for nodes is the IPv4 Router ID. This TLV uniquely links the node NLRI with the IP of the node in question fulfilling the identification information of the node. It must be noted that node attributes are more of a descriptive complement of the node, as these nodes do not present TE parameters themselves.

4.2.2.2 *Link attributes*

Just as happens with node attributes, link attributes are a set of triplets of the same format as Figure 17. They are presented together with the correspondent link NLRI describing the link. Link attributes can be sourced by any of the extensions for the IGP routing protocols (IS-IS/OSPF).

There are many link attributes defined by the IETF (Figure 23) but as policies are local to every autonomous system, one can choose which to share. In our implementation we are going to pay attention only to four of them:

- **Maximum Link Bandwidth:** This TLV contains the maximum bandwidth that can be used by this link on this direction. [9]
- **Maximum Reservable Link Bandwidth:** This TLV contains the maximum quantity of bandwidth that can be reserved on this link in this direction [9].
- **Unreserved Bandwidth:** This TLV contains the actual reservable bandwidth at this point on this link [9].
- **Metric:** This TLV carries the IGP metric for the link. [8]

With TLVs like these, the network operator can apply local policies and special treatment to traffic allowing for better path optimization and network state awareness. Such information is necessary for the optimizing algorithms inside the Path Computation Elements.

TLV Code Point	Description	IS-IS TLV /Sub-TLV	Defined in:
1028	IPv4 Router-ID of Local Node	134/---	[RFC5305]/4.3
1029	IPv6 Router-ID of Local Node	140/---	[RFC6119]/4.1
1030	IPv4 Router-ID of Remote Node	134/---	[RFC5305]/4.3
1031	IPv6 Router-ID of Remote Node	140/---	[RFC6119]/4.1
1088	Administrative group (color)	22/3	[RFC5305]/3.1
1089	Maximum link bandwidth	22/9	[RFC5305]/3.3
1090	Max. reservable link bandwidth	22/10	[RFC5305]/3.5
1091	Unreserved bandwidth	22/11	[RFC5305]/3.6
1092	TE Default Metric	22/18	[RFC5305]/3.7
1093	Link Protection Type	22/20	[RFC5307]/1.2
1094	MPLS Protocol Mask	---	Section 3.3.2.2
1095	Metric	---	Section 3.3.2.3
1096	Shared Risk Link Group	---	Section 3.3.2.4
1097	Opaque link attribute	---	Section 3.3.2.5
1098	Link Name attribute	---	Section 3.3.2.6

Figure 23: Link Attributes [8]

4.3 Translation from OSPF to BGP-LS

In order for the PCE to carry out the path computation tasks it first needs a detailed image of the topology under its jurisdiction. This topology is learnt through OSPF-TE. The Traffic Engineering extensions allow OSPF to carry link state information that can be used in optimizing technics such as the PCE algorithms.

Carrying TE information in OSPF is a well-known standardized feature (RFC 3630). The problem arises when we must export this topology and these TE parameters outside our domain (e.g. Hierarchical PCE architecture). This is where BGP-LS comes to play.

As shown in Figure 13, the intra-domain topology information is learnt through IGP (OSPF mainly) and stored in the traffic engineering database. This database is accessed by the c-PCE to compute optimum paths within its domain. If optimum inter-domain paths shall be computed, this topology information must be exported to the p-PCE through BGP-LS. In order to perform this action, one must first translate the OSPF information into BGP-LS.

BGP-LS extends the BGP Update messages to advertise link-state topology thanks to the new BGP Network Layer Reachability Information (NLRI) and BGP-LS attribute.

The BGP NLRI carries the descriptors used to define the element in question (e.g. link or node) and the BGP-LS attribute carries the chosen parameters to characterize the described element. Information is codified using multiple TLV triplets just as the ones used in OSPF-TE making it easy to integrate.

For the purpose of this section we are only going to consider a scenario where we have an origin (router) with the correspondent IPv4, a destination with its IPv4 and a link having the following TE parameters: maximum BW, maximum reservable BW and unreserved BW.

4.3.1 Mapping BGPLS NLRI descriptors from OSPF

To illustrate this example we will use a Link NLRI as shown in Figure 19.

4.3.1.1 *Node Descriptors*

In the OSPF packet we will find two fields that tell us the origin and destination node IDs. The origin IP will be the Source OSPF Router ID in the OSPF header and this will be mapped into the IGP Router ID subTLV inside the Local Node Descriptors field.

The destination IP will be found as the Link ID field in the MPLS LSA in OSPF. This will be mapped into the correspondent IGP Router ID in the Remote Node Descriptors field.

There are other subTLVs inside the Local/Remote Node Descriptors but they will not be taken into consideration in this section.

Figure 24 shows how this mapping is done.

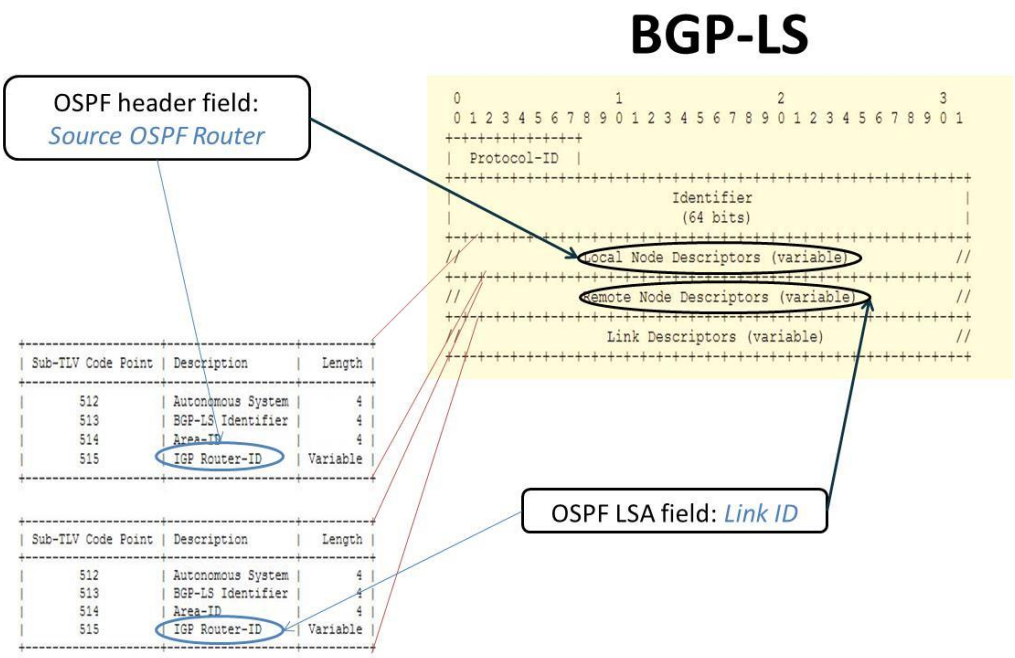


Figure 24: OSPF to BGPLS Node Descriptors

4.3.1.2 *Link Descriptors*

In the Link Descriptors field the only two TLVs that are going to be mapped from OSPF are the local and remote interface addresses. This information will be mapped directly from the Local/Remote Interface address TLV carried in the MPLS LSA of OSPF into the Local/Remote Interface address subTLV of the Link Descriptors field as shown in Figure 25.

In case of unnumbered interfaces being used, the same procedure must be applied but utilizing the Link Local/Remote Identifiers TLV.

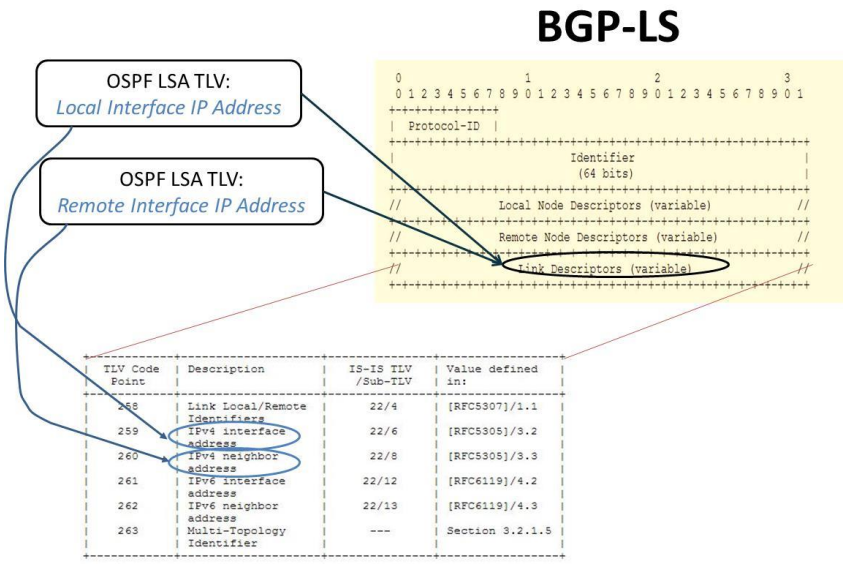


Figure 25: OSPF to BGPLS Link Descriptors

4.3.2 Mapping of OSPF-TE parameters into the BGP-LS attribute

The BGP-LS attribute will be a set of TLV triplets carrying the desired TE parameters learnt by OSPF. For this section we will be using bandwidth related parameters to illustrate the example but they are many more.

In Figure 26 the procedure on how the BGP-LS attribute is mapped is illustrated. The TLVs carried in the MPLS-TE LSA in OSPF are directly translated into the equivalent TLVs in BGP-LS. As such, the Unreserved BW TLV in OSPF is mapped into the Unreserved BW TLV in BGP-LS. The same happens with the Maximum BW TLV and the Maximum Reservable BW TLV.

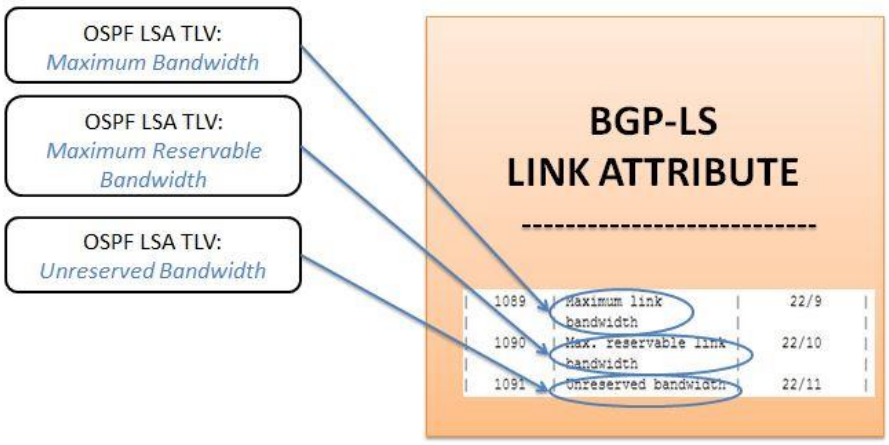


Figure 26: OSPF LSA to BGP-LS Attribute

4.4 Validation Tests

This section contains the validation tests for the protocol. To do so, the test-bed is explained firstly and, later, the experimental validation.

4.4.1 Test-bed scenario

In this section the test scenario will be presented in order to show how the implemented BGP-LS protocol works and validate its theoretical use cases.

Figure 27 shows the chosen test-bed scenario. It contains an IGP domain composed by four nodes each of them using OSPF to flood Traffic Engineering (TE) information. The IGP peer represents the broadcast address of OSPF. All OSPF messages within the domain are flooded by the different nodes onto this address and collected by the TED storing the TE information.

A possible utilization of this topology module is that a PCE can access to such TED and retrieve the necessary information to compute a path within the IGP domain.

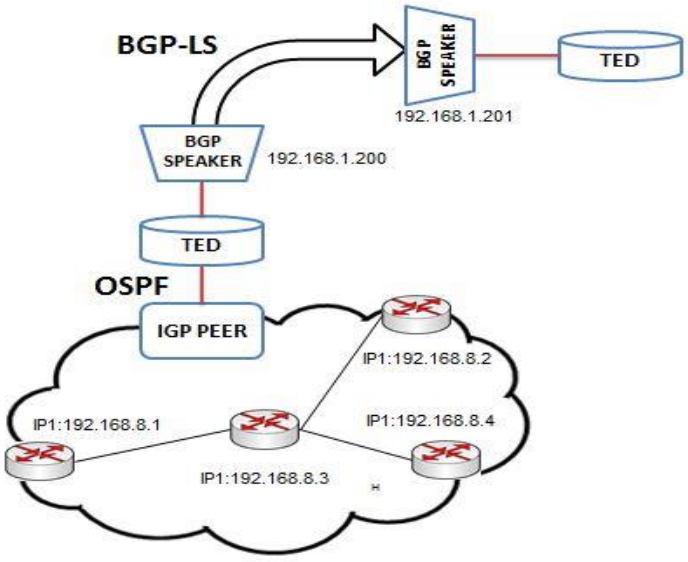


Figure 27: BGP-LS draft 04 testing scenario

Attached to the TED, a BGP speaker is placed, having access to the information stored in it. Using the new Link State extensions for BGP the speaker behaves as a route reflector, adapting the information learnt by OSPF into a BGP Update message and exporting it to an external BGP speaker.

The external BGP speaker is connected to a TED that stores the received data. The external BGP speaker maintains a session with the speaker in the domain, constantly updating the external TED. Hence, a PCE could use this information without being part of the IGP domain. Another scheme to use the

topological information is in a Hierarchical PCE (H-PCE) scenario. The child-PCEs (cPCE) and the parent-PCE (pPCE) can exchange topology using BGP-LS.

Policies can be applied to the local BGP speaker to restrict the amount of information being exported. If the configuration is to be used in an H-PCE scenario, the minimum amount of exported information is the inter-domain links making it possible for the PCE to create the domain topology map. On the other hand, the local BGP speaker can be configured to send a complete list of all the intra-domain and inter-domain links.

4.4.2 BGP-LS extensions

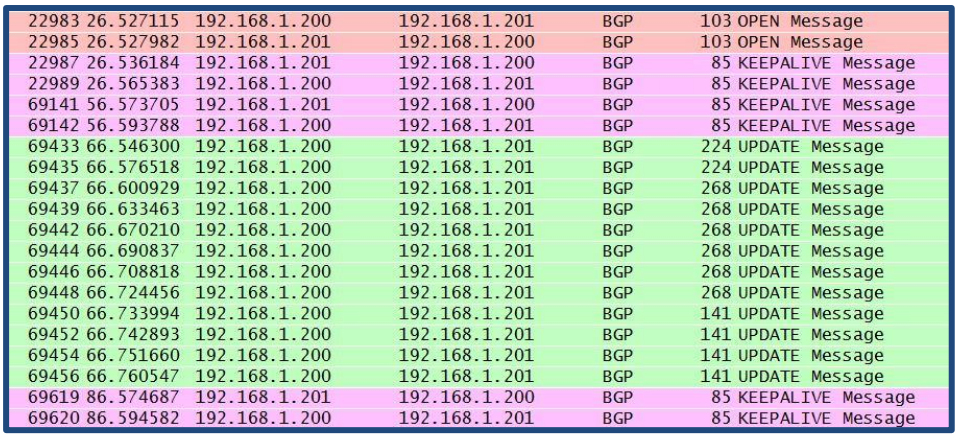
In order to create such a scenario, extensions to the Border Gateway Protocol have been implemented. Such extensions have been codified following the '*North-Bound Distribution of Link-State and TE Information using BGP*' IETF draft (draft-ietf-idr-ls-distribution-04). This draft extends the BGP4 protocol to carry Link State information by creating two new BGP path attributes as explained in 4.2.

Table 1 shows a summary of all the supported BGP-LS features in our current implementation and their section in the draft.

Table 1: Supported BGP-LS Features

Draft Section	Supported BGP-LS Feature	Feature Summary
3.1	TLV Format	Type, Length and Value triplets
3.2	AFI/SAFI = 16388/71	Address Families supporting the new LS attributes
3.2	Node NLRI	NLRI carrying node descriptors
3.2	Link NLRI	NLRI carrying the link descriptors
3.2.1	Node Descriptors	Node Descriptors TLVs used to describe the nodes
3.2.2	Link Descriptors	Link Descriptors TLVs used to describe the links
3.3	Link State Attribute	New Attribute carrying the Node/Link attributes
3.3.1	Node Attributes	TLVs used to characterize a Node (supported types: 1024, 1026, 1028)
3.3.2	Link Attributes	TLVs used to characterize a Link (supported types:1028, 1030, 1088, 1089, 1090,1091, 1092)

In Figure 28 the message exchange between the BGP peers is shown. The local BGP speaker initiates the conversation through an Open message carrying the Multiprotocol Extension Capabilities field announcing that it supports Link State distribution. The external BGP peer sends an Open message back also announcing BGP-LS support through the Multiprotocol Extension Capabilities field. In this manner, the BGP session enters the '*Open Confirm*' state waiting for the *keepalive* acknowledgement.



22983	26.527115	192.168.1.200	192.168.1.201	BGP	103	OPEN Message
22985	26.527982	192.168.1.201	192.168.1.200	BGP	103	OPEN Message
22987	26.536184	192.168.1.201	192.168.1.200	BGP	85	KEEPALIVE Message
22989	26.565383	192.168.1.200	192.168.1.201	BGP	85	KEEPALIVE Message
69141	56.573705	192.168.1.201	192.168.1.200	BGP	85	KEEPALIVE Message
69142	56.593788	192.168.1.200	192.168.1.201	BGP	85	KEEPALIVE Message
69433	66.546300	192.168.1.200	192.168.1.201	BGP	224	UPDATE Message
69435	66.576518	192.168.1.200	192.168.1.201	BGP	224	UPDATE Message
69437	66.600929	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69439	66.633463	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69442	66.670210	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69444	66.690837	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69446	66.708818	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69448	66.724456	192.168.1.200	192.168.1.201	BGP	268	UPDATE Message
69450	66.733994	192.168.1.200	192.168.1.201	BGP	141	UPDATE Message
69452	66.742893	192.168.1.200	192.168.1.201	BGP	141	UPDATE Message
69454	66.751660	192.168.1.200	192.168.1.201	BGP	141	UPDATE Message
69456	66.760547	192.168.1.200	192.168.1.201	BGP	141	UPDATE Message
69619	86.574687	192.168.1.201	192.168.1.200	BGP	85	KEEPALIVE Message
69620	86.594582	192.168.1.200	192.168.1.201	BGP	85	KEEPALIVE Message

Figure 28: BGP-LS message trace

In Figure 28 the message exchange between the BGP peers is shown. The local BGP speaker initiates the conversation through an Open message carrying the Multiprotocol Extension Capabilities field announcing that it supports Link State distribution. The external BGP peer sends an Open message back also announcing BGP-LS support through the Multiprotocol Extension Capabilities field. In this manner, the BGP session enters the '*Open Confirm*' state waiting for the *keepalive* acknowledgement.

Following the BGP4 protocol, two Keepalive messages must be interchanged before switching into the '*Established*' state leading to the Update message exchange.

Once the session is fully functional the Update message exchange starts.

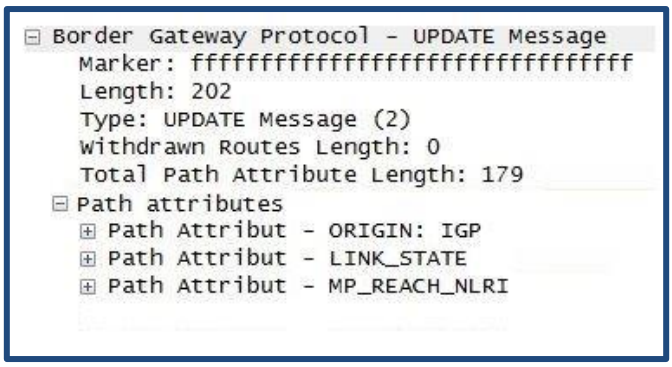


Figure 29: Test-bed Update message format

Figure 29 shows a capture of an exchanged Update message displaying the path attributes that have been used. The Origin attribute is a mandatory BGP4 attribute. In our case, the announced route was originated using IGP, thus the Origin attribute is set to IGP.

The other two, are the new attributes used to carry the LS information. As they are the main focus point of this section, two sub-sections are dedicated to fully explain how these two new features are used in our test-bed scenario.

4.4.2.1 *BGP-LS NLRI (MPREACH attribute)*

As described in section 4.2.1, the new BGP Network Layer Reachability Information is used to carry topological information about links and nodes in the network through two different TLVs: node and link descriptors.

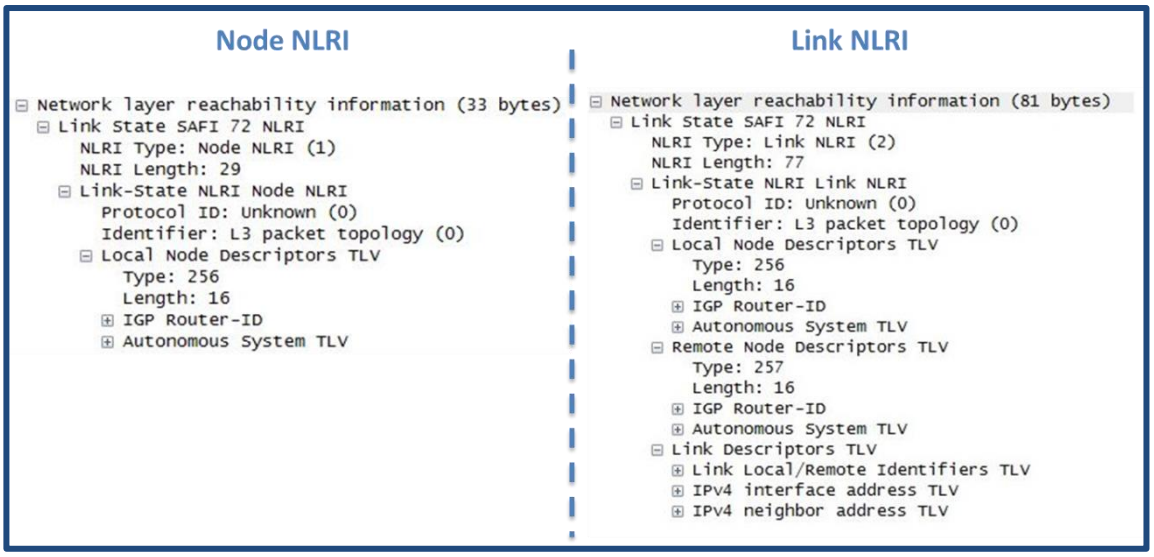


Figure 30: Test-bed BGPLS NLRI format

In Figure 30 the NLRI format used for this demonstration is shown. For didactical purposes, both NLRI types are shown although we are going to base our explanation in the Link NLRI as it contains all the possible descriptors an NLRI can carry. Accompanying an NLRI a Link State attribute (node or link) could be present characterizing the node or link being announced through it.

As it has already been mentioned in this document, an NLRI contains node and link descriptors. In case of a Link NLRI, two nodes descriptors TLVs (local and remote) and a link descritors TLV are carried. For the purpose of this study the node descriptors used are the IGP Router ID TLV and the Autonomous Number TLV. As link descriptors, both IPv4 addresses TLVs (local and remote) and the Link Local/Remote Identifiers TLV (unnumbered interfaces) are used.

4.4.2.1.1 Node Descriptors TLV

In Figure 31 the Node Descriptors TLV (local and remote) format used is displayed. In it, the node's IPv4 address as well as the domain ID to which it belongs are carried.

In this case, as our Wireshark capture belongs to a Link NLRI both Local and Remote Node Descriptors TLVs are present. They announce the end-points to a link between two intra-domain nodes (see Figure 27).

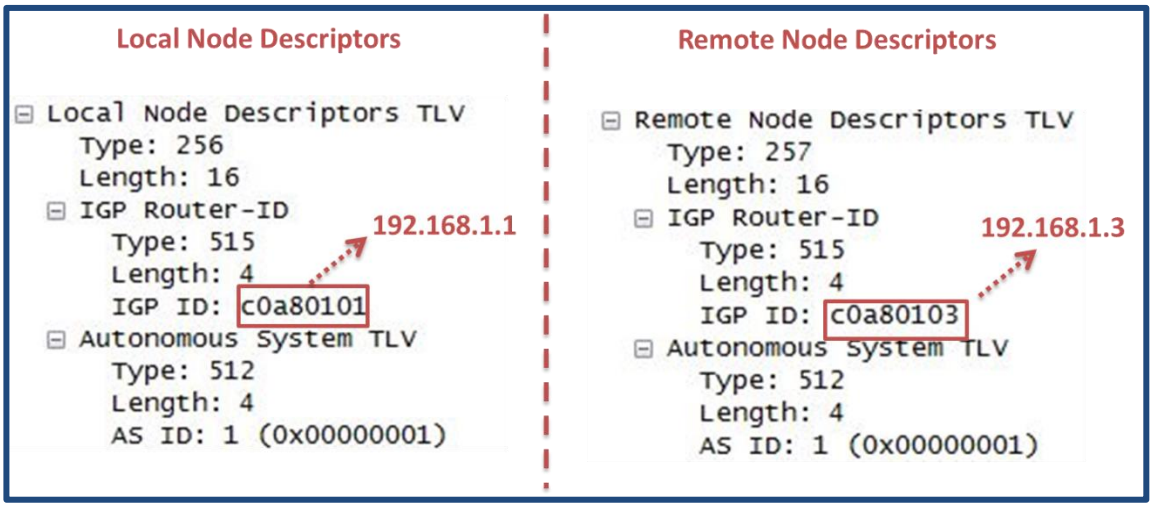


Figure 31: Test-bed Node Descriptors TLV format

The IGP Router ID of both nodes is the IPv4 of each router, 192.168.1.1 (0xc0a80101) and 192.168.1.3 (0xc0a80103) respectively. It must be noted that this Router ID could be presented in several formats depending on whether it is representing a pseudo-node or not, and the IGP protocol used (IS-IS or OSPF). In our case we are dealing with non-pseudo nodes and OSPF so the IGP Router ID is the IPv4 of the node. In addition the Autonomous System TLV is carried, identifying the domain to which each router belongs. As expected, both domains are the same as we are dealing with intra-domain nodes in this example.

4.4.2.1.2 Link Descriptors TLV

In Figure 32, the Link Descriptors TLV format used in our proof of concept is shown. In it, the IPs for both local and remote node interfaces are carried as well as support to unnumered interfaces with the Link Local/Remote Identifiers TLV.

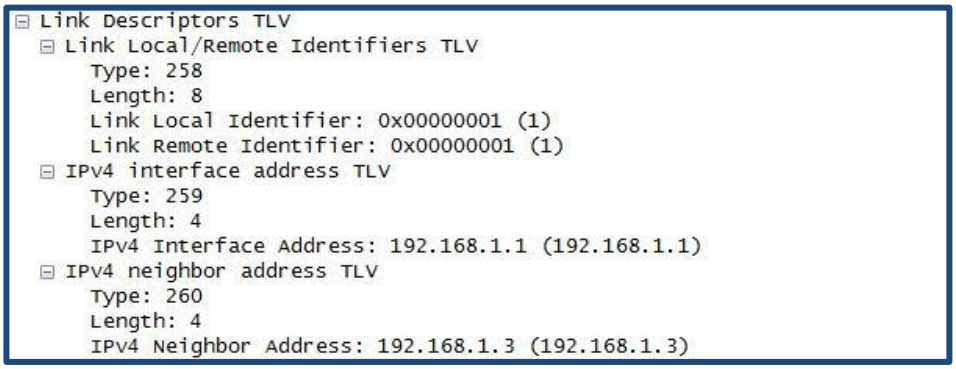


Figure 32: Test-bed Link Descriptors TLV format

It should be pointed out that in our case, the IGP Router ID and the Interface Address TLV values are the same as we are identifying the nodes with their IPv4 address. This is where unnumbered interfaces[3] are useful as they allow you to identify the egress interface of a packet just by a 4 byte identifier without needing to ask for an independent IP address, thus conserving address space. In such a way, if Router A (.1) needs to send a packet to B (.3) it would just need to specify the interface identifier (e.g. packet A to 192.168.1.3 via 0x00000001 (1)).

With both Node and Link Descriptors TLV our links are fully identified and we can proceed to add Traffic Engineering Information through the LS Attribute.

4.4.2.2 *Link State attribute*

As stated in the ietf-idr-ls-distribution draft:

'The BGP-LS attribute is an optional, non-transitive BGP attribute that is used to carry link, node and prefix parameters and attributes. This attribute SHOULD only be included with Link- State NLRIs'.

In Figure 33 the Link State attribute carried together with the previously described Link NLRI is presented. The chosen TLVs to characterize such link are the Maximum Link Bandwidth TLV and the Unreserved Bandwidth TLV (see section 4.2.2.2 for the full definition). They are many others but we believe these two are sufficient in order to illustrate the test-bed scenario in Figure 27.

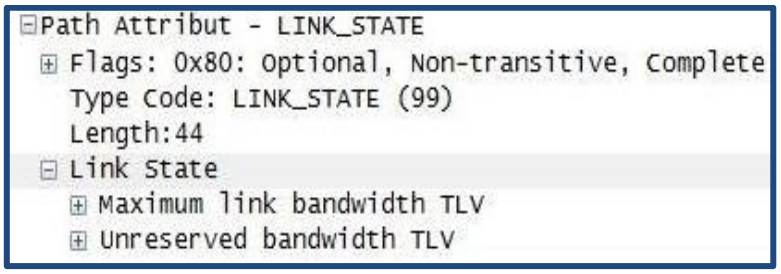


Figure 33: Test-bed Link State attribute format

As explained in section 4.4.1, when the external BGP speaker receives an update message it stores the TE information carried in it in the TED. This TED is a network graph where the vertices represent the routers in the domain and the edges are the links. Every time an MP_REACH (NLRI) attribute is received either a vertex (node NLRI) or an edge (link NLRI) is inserted into the graph.

[3] When unnumbered interfaces are configured, routes learned through the IP unnumbered interface have the interface as the next hop instead of the source address of the routing update. If we use IP unnumbered on each serial interface, we save address space; IP unnumbered only makes sense for point-to-point links.

The edges (links) have a TE table attached to them with their attributes (Figure 33) learnt through BGP-LS. Note that in this implementation we do not use Node attributes as they were not necessary for our proof of concept. Figure 34 shows how the graph from the proposed test-bed scenario would look like.

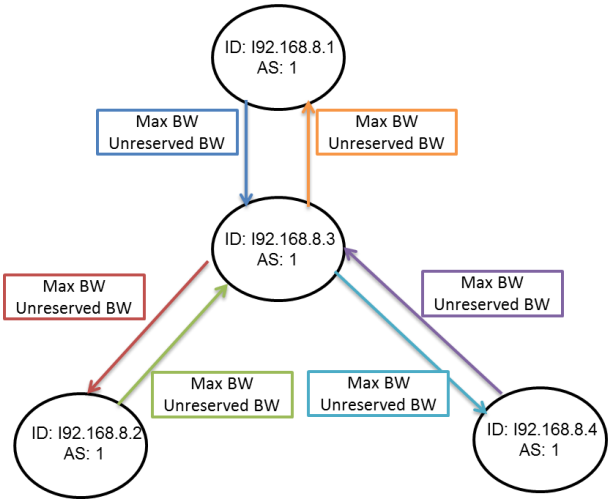


Figure 34: Traffic Engineering Database format

In such a way, a PCE feeding from such a TED could easily determine the appropriate path for a certain flow based on the parameters each local BGP peer has exchanged. Policies can be applied to limit or expand the exchanged information. In our case, we have used the bandwidth attributes as they are the minimum requirements for any PCE running a computation algorithm on an IP network.

4.4.3 OSPF to BGP-LS translation

As stated in section 4.3 of this document, a mapping from OSPF to BGP-LS must be performed in order to export the information flooded internally through IGP into the external controller, the PCE.

In Figure 27 this procedure is illustrated in the form of an IGP peer connected to a Traffic Engineering Database (TED) where all the information interchanged by the nodes is stored. In other words, the nodes multicast their OSPF messages and, a TED with an integrated OSPF speaker, dissects such messages and stores the information carried inside them appropriately.

4.4.3.1 *OSPF Link State Advertisement*

To explain this mode of operation, we must explain how Link State Advertisement works in an OSPF domain. The LSDB is a database of all OSPF router LSAs, summary LSAs, and external route LSAs. The LSDB is compiled by an ongoing exchange of LSAs between adjacent routers so that each router is synchronized with its neighbour. When the AS has converged, all routers have the appropriate entries in their LSDB.

To create the LSDB, each OSPF router must receive a valid LSA from each other router in the AS. This is performed through a procedure called flooding. Each router initially sends out an LSA which contains its own configuration. As it receives LSAs from other routers, it propagates those LSAs to its neighbour routers.

In a broadcast network, this flooding is performed by sending each LSA packet to IANA's designated multicast address for Link State Advertisement in OSPF, address 224.0.0.5. Every OSPF peer is listening on this address and completing its LSDB with the broadcasted links and TE parameters. In this way, an LSA from a given router is flooded across the AS so that each other router contains that router's LSA.

This is precisely how the TED is filled in, in our scenario. The OSPF peer attached to the TED is constantly listening for broadcasted LSAs from the nodes in the domain. Consequently, every message sent to broadcast address 224.0.0.65 is collected and introduced into the database. In this manner, every link is included with its TE parameters and the model shown in Figure 34 is created.

In Figure 35, one can appreciate how this broadcast is done (traces in red). Every unidirectional link in the scenario is advertised through address 224.0.0.5 and collected by the OSPF speaker connected to the TED.

9	3.573209	192.168.1.1	224.0.0.5	OSPF	154 LS Update
10	4.177689	192.168.1.2	224.0.0.5	OSPF	154 LS Update
13	4.459467	192.168.1.3	224.0.0.5	OSPF	154 LS Update
14	4.459493	192.168.1.3	224.0.0.5	OSPF	154 LS Update
15	4.459499	192.168.1.3	224.0.0.5	OSPF	154 LS Update
21	5.212587	192.168.1.4	224.0.0.5	OSPF	154 LS Update
126	12.103217	192.168.1.201	192.168.1.200	BGP	85 KEEPALIVE Message
127	12.121229	192.168.1.200	192.168.1.201	BGP	85 KEEPALIVE Message
145	14.284840	192.168.1.200	192.168.1.201	BGP	224 UPDATE Message
147	14.290868	192.168.1.200	192.168.1.201	BGP	224 UPDATE Message
150	14.317678	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
152	14.335302	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
154	14.341417	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
156	14.352826	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
158	14.364391	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
160	14.379729	192.168.1.200	192.168.1.201	BGP	268 UPDATE Message
162	14.387148	192.168.1.200	192.168.1.201	BGP	141 UPDATE Message
164	14.390619	192.168.1.200	192.168.1.201	BGP	141 UPDATE Message
166	14.396535	192.168.1.200	192.168.1.201	BGP	141 UPDATE Message
168	14.405269	192.168.1.200	192.168.1.201	BGP	141 UPDATE Message

Figure 35: OSPF to BGP-LS message trace

4.4.3.2 *Translating Descriptors and TE parameters from OSPF to BGP-LS*

As one can see in the Wireshark screen-shot (Figure 35), the network elements work as described in section 4.4.1. As soon as the TED receives an LS Update through OSPF it makes the translation to BGP-LS and exports the learnt parameters to the external BGP speaker (192.168.1.201).

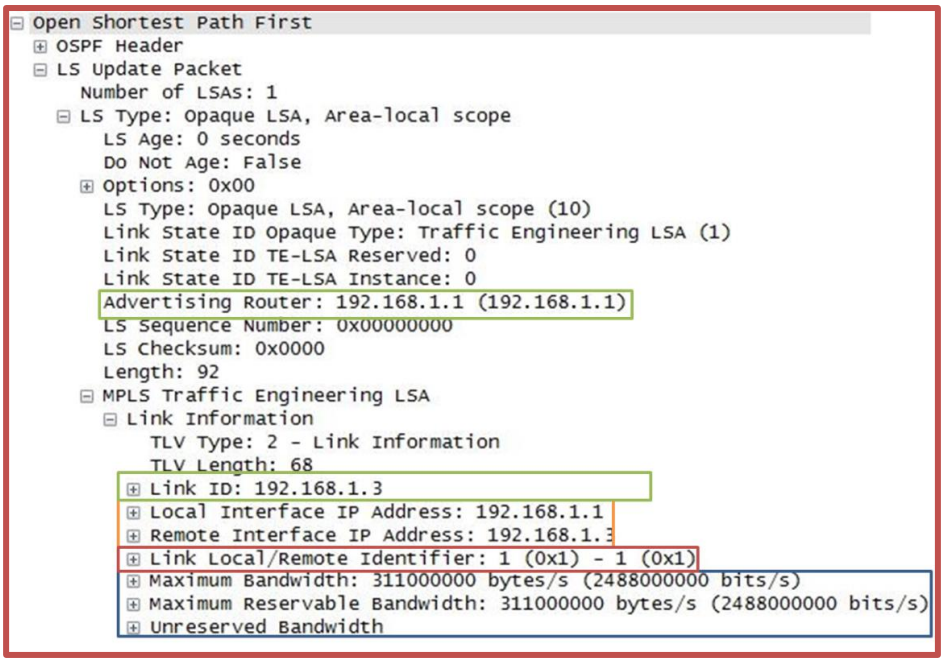


Figure 36: Test-bed OSPF LS Update message

In Figure 36 an example of an exchanged LS Update message is shown. In it router with ID 192.168.1.1 advertises a link to router with ID 192.168.1.3. These two instances would be the IGP identifier field in both the Local Node Descriptors and Remote Node Descriptors TLVs.

Furthermore, coloured in orange, the IPv4 addresses of the Local and Remote Interfaces are carried. As one can appreciate, their value is the same as the IGP Identifier of the node to which it belongs. This creates a situation where a certain node having multiple interfaces connecting to other nodes would have multiple interfaces having the same IP address violating the main principle of IP addressing; "an IP address must be globally unique".

For this reason, support for unnumbered interfaces (shown in red) is given through the Link Local/Remote Identifier. In such a way an interface is given a 4 byte identifier distinguishing it from other local interfaces. In the trace above, the Link Local/Remote Identifier field specifies that the link joining 192.168.1.1 and 192.168.1.3 has local interface with local unnumbered identifier 0x1 and remote interface with remote unnumbered identifier 0x1 as end-points. In other words, a packet going from .1 to .3 should exit through interface 0x1.

The two fields, just described, are translated into the Link Descriptors TLV shown in Figure 32 for our scenario.

Highlighted in blue in the Wireshark capture, the TE parameters associated to the described link are shown. As already mentioned throughout the document we have used BW parameters to describe the link's capacity. In

this, we can appreciate a new TLV, not included in the BGP LS attribute shown in Figure 33, the Maximum Reservable link bandwidth. It describes the maximum allowed bandwidth a certain flow can reserve. As the value is the same as the Maximum Bandwidth of the link, this means that no limitation is imposed. Thus, in this case, it is not exported through BGP-LS as it does not present any additional value.

4.4.3.2.1 Available Labels Sub-TLV

Some data plane technologies that wish to make use of a GMPLS control plane contain additional constraints on switching capability and label assignment. In addition, some of these technologies must perform non-local label assignment based on the nature of the technology, e.g., wavelength continuity constraint in WSON. Such constraints can lead to the requirement for link by link label availability in path computation and label assignment. [10]

As already explained in 4.3.2, TE parameters in OSPF are directly mapped into the Link State attribute of BGP (see Figure 33). To illustrate such statement, we chose to include the Available Labels TLV used to describe optical links in OSPF into the BGP-LS protocol.

The Available Labels sub-TLV (1200) [10] indicates the available labels for use in a certain link. Its value field is illustrated in Figure 37.

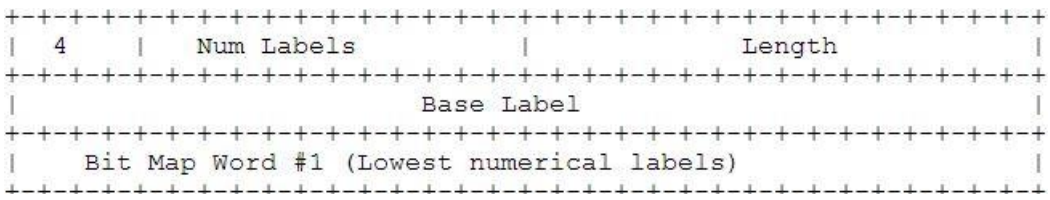


Figure 37: OSPF Available Labels sub-TLV

NumLabels in this case tells us the number of labels represented by the bit map. Each bit in the bit map represents a particular label with a value of 1/0 indicating whether the label is in the set of available labels or not. In other words, a bit set to one indicates that the label is free and a bit set to 0, the contrary.

Bit position zero represents the lowest label and corresponds to the base label, while each succeeding bit position represents the next label logically above the previous.

The size of the bit map is NumLabels bits long, but the bit map is padded out to a full multiple of 32 bits so that the TLV is a multiple of four bytes. Bits that do not represent labels are set to zero and must be ignored.

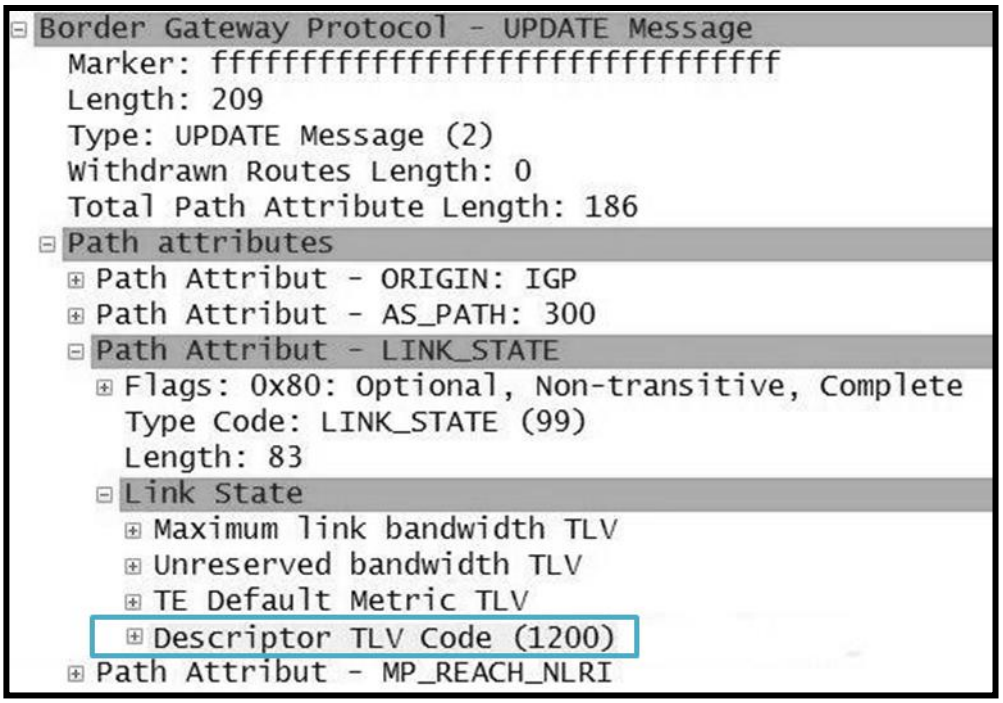


Figure 38: Link State Attribute with Available Labels

A future use of this TLV as presented in Figure 38 would open the possibility of using BGP-LS to export optical Link State information with all the advantages that could present. For example, an external entity (e.g. PCE) could be used to compute end-to-end optical paths leveraging the use of dedicated path computation elements in optical domain scenarios.

5 Segment Routing Extensions

This chapter explains the extensions to support PCEP and BGP-LS for Segment Routing scenarios.

5.1 PCEP Extensions for Segment Routing support

In SR scenarios, the edge router of the SR path incorporates in all packets an SR header containing all the hops that the packet is going to traverse. These hops are codified as a set of Segment Identifiers (SIDs) which contains all the information to guide the packet from the ingress node to the target node, making no use of any signalling protocol.

In a PCEP message exchange, path information is codified in the Explicit Route Object (ERO). As one can imagine, these objects need to carry SIDs making it possible for a PCE and a PCC to interact in an SR context.

When a PCEP session between different PCEP entities is brought up, both parties exchange information that announces their ability to support SR-based technics. From here on, both peers will use SR-specific PCEP elements to exchange session information.

A PCEP message consists of a common header followed by a variable length body made up of mandatory or optional elements [11]. Many of these optional elements represent additional TLVs that have been defined to make PCEP useful as new technologies are implemented such as Segment Routing.

5.1.1 The Segment Routing PCE Capability TLV

During the initialization stage both peers exchange information about their ability to use certain characteristics. These characteristics are sent as capabilities included in the Open message exchange. One of them is the Segment Routing capability sent as the SR-PCE-Capability TLV. Its format is shown in Figure 39.

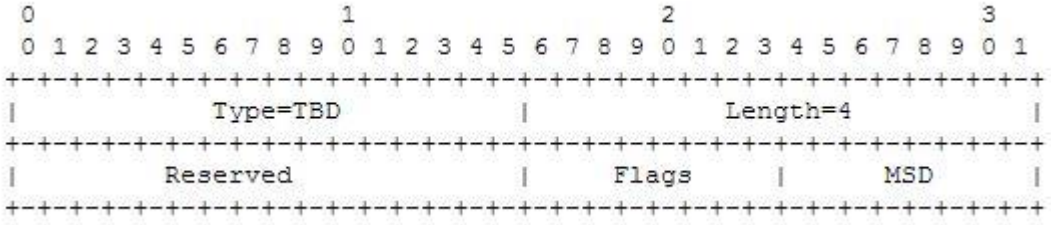


Figure 39: SR-PCE-Capability TLV [11]

The value length of this TLV is 4 bytes although the MSD (Maximum SID Depth) is the only field in use at the moment. This field carries the maximum number of tags (e.g. SIDs) that a PCC can impose on a packet. The other two fields must be dismissed upon reception.

The negotiation of this capability is pretty simple. Any element including this TLV in its Open message is telling the other peer that it has support for SR-TE paths. The SR Capability TLV is meaningful only in the Open message sent from a PCC to a PCE; the PCE does not have to set the MSD field in the reply message.

The default value of the MSD field is 0 meaning that the PCC does not set any limitation on the number of SIDs that can be included in the SR header. In case a PCEP session is set-up with a not-null MSD value, the corresponding PCE cannot announce SR paths that exceed this MSD value. If a PCC receives a path containing more labels than agreed, it must send back an error message (PCErr) detailing this issue.

5.1.2 The Path Setup Type TLV

Included in the RP (Request Parameters) object, this TLV specifies the LSP setup method for the PCEP session. In order for the PCE to choose the appropriate encoding for the LSP being calculated it must first be made aware of the desired setup format. In other words, the PCC must specify whether it wants a set of tags (e.g. SIDs) or it wants to use the traditional RSVP-TE to setup the LSP. The format of this TLV is shown in Figure 40.

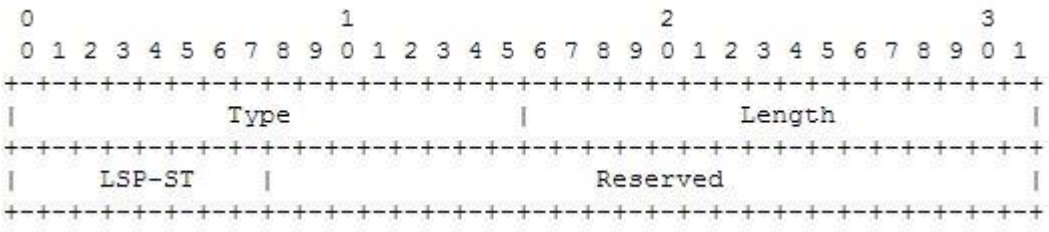


Figure 40: LSP-Setup-Type TLV [12]

The only value field being used at the moment is the LSP-ST field. It can be either set to 1 to indicate that the PCC is going to setup the LSP using Segment Routing or to 0 if RSVP-TE is going to be used.

The RP object must be present in all requests (PCReq) as well as in all responses (PCRep). Apart from the optional TLV detailed above, it includes the priority of the request and the Request-ID number that uniquely identifies the communication context between the PCC and the PCE.

5.1.3 The SR-ERO Object

Every SR path consists on a number of SIDs determining a per-hop path from the ingress router to the chosen destination. Each SID represents a node or an adjacency identifier (NAI). This NAI can have multiple formats but for the sake of this document we will suppose it is an IPv4 frame.

The ERO object can be found in the reply (PCRep) from a PCE to a PCC. It is a container made up of the necessary number of SR-ERO sub-objects to fully describe a SR-path. The format of an SR-ERO sub-object is shown in Figure 41.

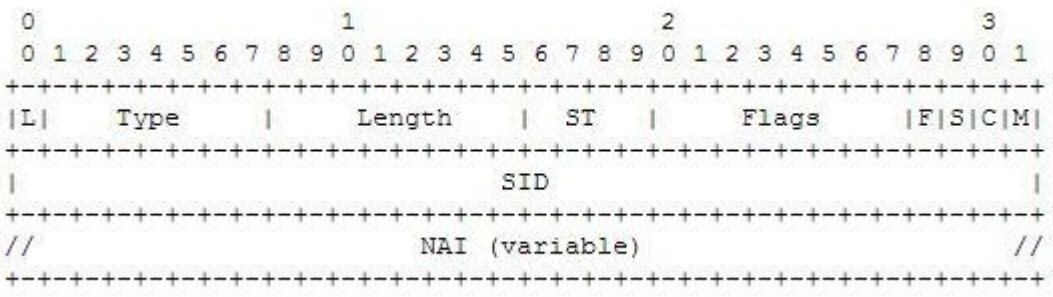


Figure 41: SR-ERO Sub-Object [11]

The SR-ERO sub-object contains two possible fields of information. The SID field contains the 4 byte identifier associated with node or adjacency identifier (NAI). The NAI field is optional and it is only used in troubleshooting activities. As we are dealing with Segment Routing the NAI is not relevant for this document so we will consider it as inexistent.

The amount of SR-ERO sub-objects must not exceed the maximum number of labels a PCC can impose on a packet agreed via the MSD field in the Open message. In case this happens, the PCC must notify this issue via the PCErr message.

5.2 PCEP extensions for SR validation scenario

The following section will present the interoperability tests that have been performed in order to validate the Segment Routing (SR) architecture between Telefonica´s SR capable PCE and Cisco's SR capable PCCs.

In Figure 42 the proposed scenario is shown. In it we can see how the different nodes are identified with their corresponding node SIDs (90000x) and each of their local rules that are represented by adjacency SIDs (1600x). These local rules determine how the packet is treated locally. In other words, rule 16001 in CISCO_1 is not the same as rule 16001 in CISCO_2 and so on.

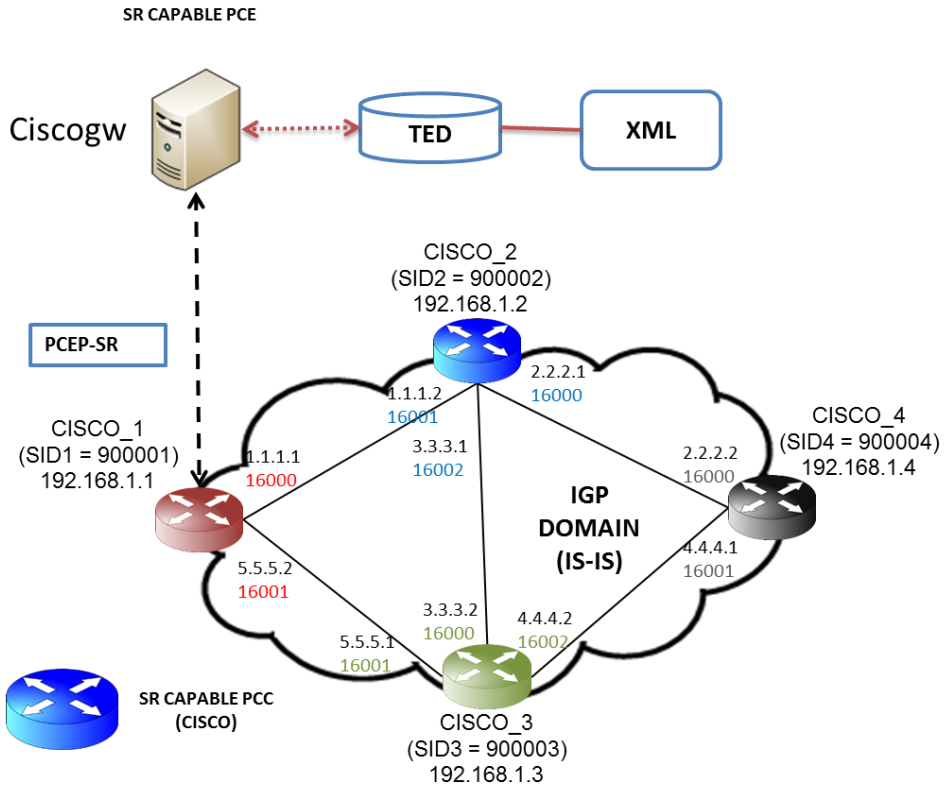


Figure 42: Test-Bed Scenario

The scenario consists of four SR routers acting as Path Computation Clients that are interconnected through 100 Mbps bidirectional links. The nodes interchange their reachability through the IS-IS Interior Gateway Protocol (ISP) creating a map with the topology of the test-bed scenario.

Each router is a virtual machine mounted with the latest Cisco OS image (IOS-XR v5.1.1). They are running on a server equipped with two processors Intel Xeon E5-2630 2.30GHz, 6 cores each, and 192 GB RAM. The virtualization tasks to create the test-bed scenario have been run under VMware's VSphere software.

The routing tasks are delegated to an external controller, in this case, a SR capable PCE implemented by Telefonica´s Core Network Evolution group. Communication between the PCCs and the PCE is done using the Path Computation Element Protocol (PCEP) described in RFC 5440 with the Segment Routing extensions presented in the sivabalan-pce-segment-routing-02 draft.

The PCE is an active stateful PCE with full knowledge of the SR-Paths created in the domain and capable of instantiating SR-paths itself. In addition, it learns the topology in a static manner through an XML document where all the nodes and links are detailed.

The following table relates the draft section to be tested with the actual test to be performed.

Table 2: Segment Routing Validation Tests with Cisco OS

Draft Section	PCEP object at test	Test Summary	Figure	Capture
5.1.1	Open object	SR capability TLV negotiation	Figure 43	Figure 44
5.2	RP object	PST = 1 in Path Setup TLV	Figure 46	Figure 48
5.3	SR-ERO object	SID being an adjacency	Figure 51	Figure 57
5.3	SR-ERO object	Path described by node SIDs: Two SIDs in SR-ERO	Figure 49	Figure 46
5.3	SR-ERO object	Path described by node & adjacency SIDs: One SID being a node and another being an adjacency	Figure 52	NO
4.3	LSP object	PCInitiate message containing Symbolic Path Name TLV.	Figure 53	Figure 54
4.3	PCInitiate Message	PCInitiate message with an SR-ERO to create an SR-path	Figure 53	Figure 54
5.3.3	PCError Message	PCE sends an invalid value and PCC reports this error back	Figure 59	NO
5.3.3	PCError Message	PCE sends an invalid number of SIDs and PCC reports this error back	Figure 60	NO

By performing these tests every section of the sivabalan-pce-segment-routing-02 draft will be validated except for the error handling messages which we were not able to implement on time as they were not a priority for the task in hand.

5.2.1 SR Capability Negotiation

In Figure 43 the capability negotiation is illustrated. Tests will verify that this negotiation takes place and that a PCEP session with SR capability is successfully opened. Furthermore, the maximum number of Segment Identifiers allowed by the PCC must be announced through the MSD parameter.

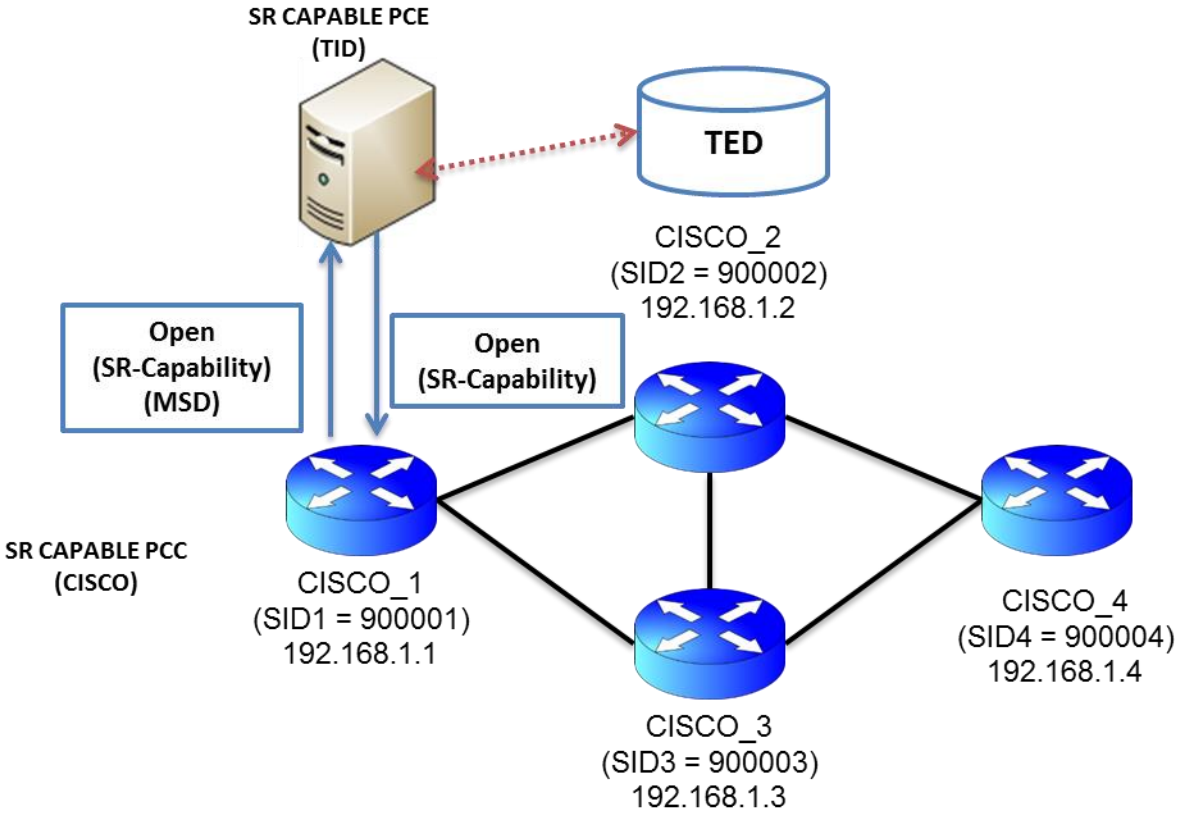


Figure 43: Capability Negotiation

In order to proof this session establishment a Wireshark capture is presented in Figure 44.

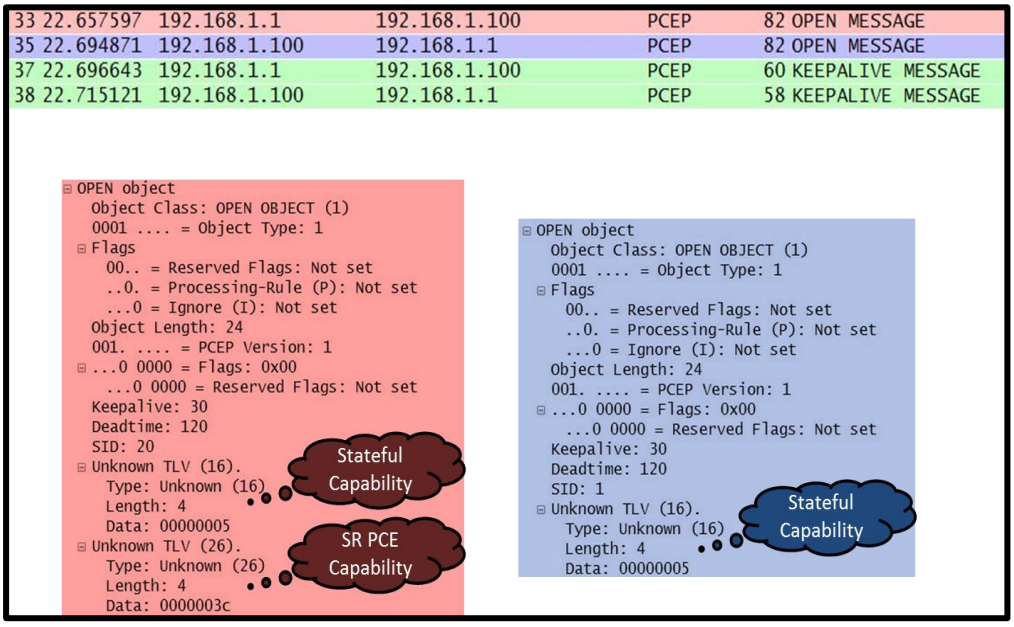


Figure 44: Capability Negotiation with SR-capable Cisco Router

As stated before, the PCC sends an Open message with the SR Capability TLV (26) included in the Open object. This TLV, see section 5.1.1, includes the

MSD field containing the maximum number of SR tags supported by the PCC. In this case the maximum number of SIDs is 60 (0x3c).

In addition, as we are dealing with a stateful PCE with PCE Instantiation capabilities, the Stateful Capability TLV (16) must be sent by the PCC to the PCE to notify that such characteristics are tolerated. The TLV format is shown in Figure 45.

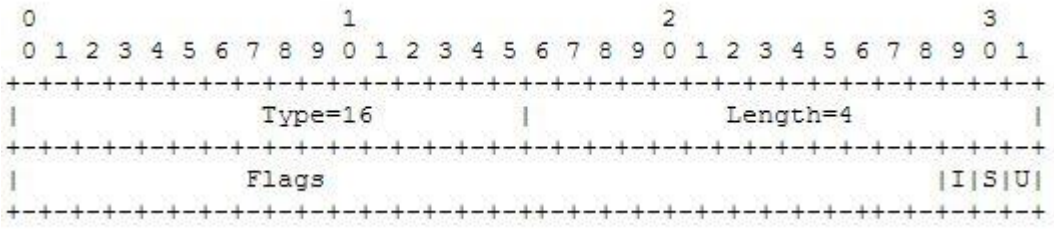


Figure 45: Stateful Capability TLV format

By setting flag I, the PCC indicates that it supports remote path instantiation (PCInitiate message). Flag U indicates that the PCC allows modification of LSP parameters (PCUpdate message). The PCE must acknowledge these capabilities by returning the TLV in its Open message to the PCC.

As it can be seen in the capture, these flags (0x5) are set in both Stateful Capability TLVs carried in the Open messages in both directions.

5.2.2 Path Computation Request-Reply Message Exchange

In Figure 46 the general Request/Reply message interchange sequence is presented. The main test to be performed under this situation is the PCE's ability to reply to a PCReq message using a set of SR tags. It must be noticed that the request from the PCC is done setting the PST parameter which means that it expects a SR path as a response.

The request will be performed from the edge router as a SR proof of concept to validate the idea that a packet can be source-routed from the ingress node to the destination node without any intra-node computation. In addition, the PCE uses Dijkstra's algorithm to provide with the shortest path to destination. It sends the stack of SIDs to the PCC and the SR-path is created.

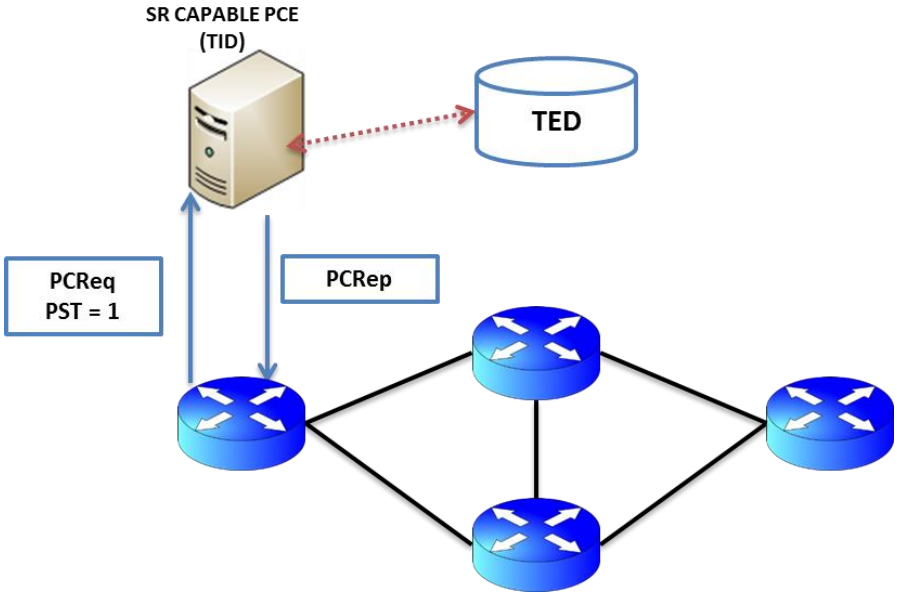


Figure 46: General PCReq-PCRep message exchange

As this use case (PCReq) is not supported by the current Cisco implementation, an equivalent scenario has been simulated using the available infrastructure in Telefonica R&D's network laboratory. The emulation has been carried out without the data-plane layer as it is not necessary to make our proof of concept.

The nodes have been emulated in separate mini PCs. Each of them is running a PCC client, just as Cisco routers do, that generates requests to the PCE. The topology is configured statically through an XML document as shown in Figure 47.

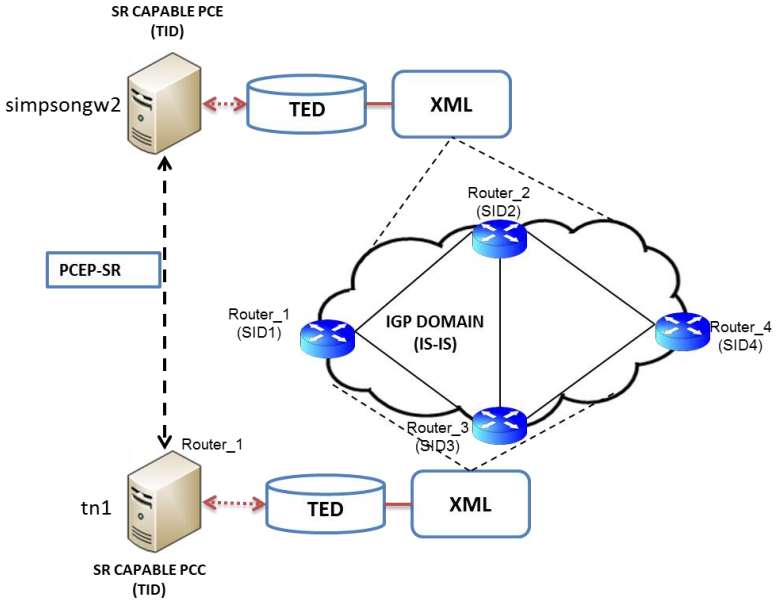


Figure 47: Emulated SR scenario

As an example, a Path Computation Request message from node 1 to node 4 is shown in Figure 48. In it, a path is established from the ingress node (node 1) to the egress node (node 4) through node 3. The routing algorithm running in the PCE is responsible for determining the optimum path to destination depending on the situation.

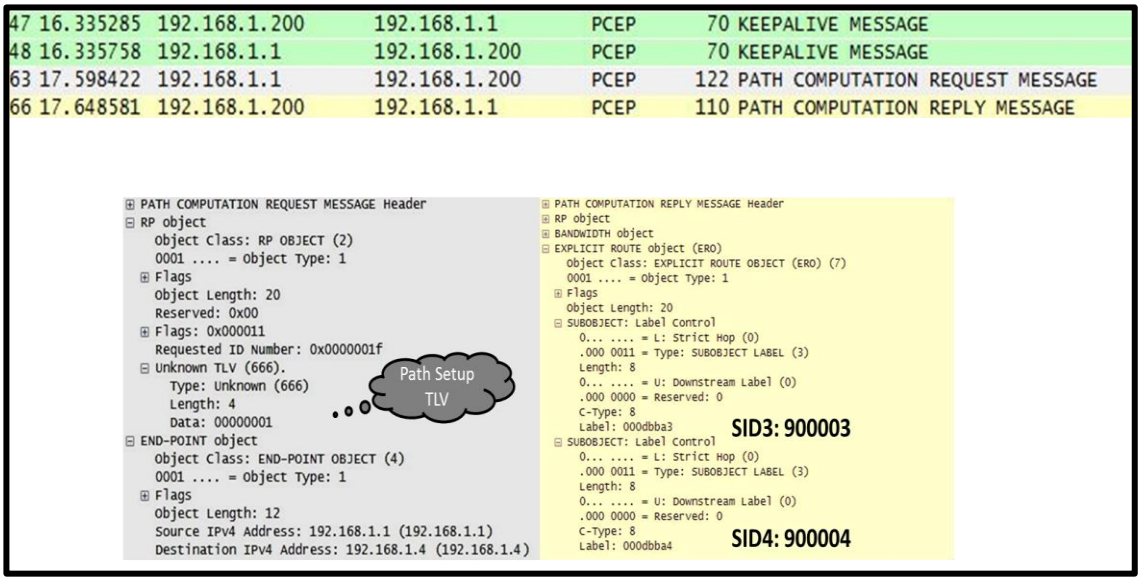


Figure 48: Request-Response message trace

As mentioned at the beginning of the section, the PCEP messages follow the format specified in rfc5440 while the SR-extensions are implemented as specified in the sivabalan-pce-segment-routing-02 draft.

Following the draft, the path setup TLV (666) is included in the RP object of the PCReq message to notify the PCE that a SR path is expected. No further changes are introduced into the PCReq message.

On the other hand, the new SR-ERO (3) sub-object is included in the Explicit Route Object (7) of the PCRep message. This sub-object[4] carries the SIDs for each hop in the SR path. In this case 900003 is the SID for node 3 and 900004 is the SID for node 4.

We have emulated a SR path using node SIDs as the SR tags. If the test was performed in a scenario including the data-plane, the resulting path would look as shown in Figure 49.

[4] SR-ERO sub-object appears as Label Control in the Wireshark capture as no dissector is implemented yet for such sub-object.

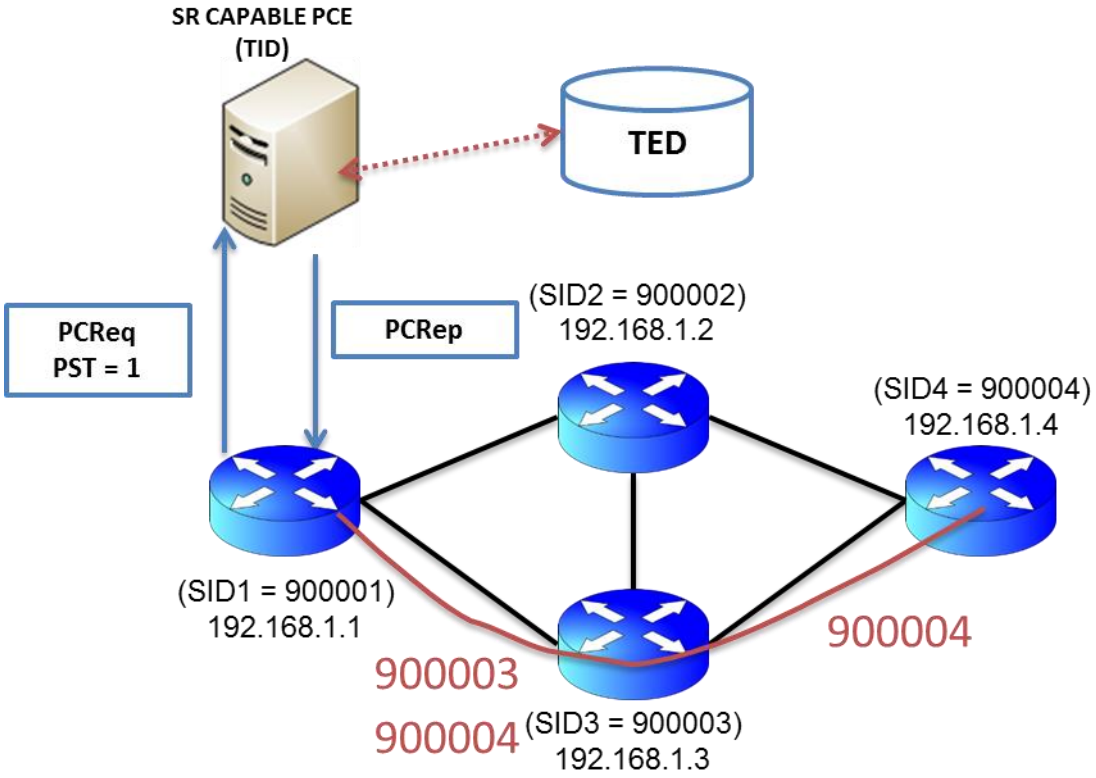


Figure 49: Resulting SR path using node SIDs

In Figure 50 we show a possible situation that could be addressed by making use of node SIDs.

Let us imagine that the link that joins nodes 3 and 4 is congested. This technique would allow us to leverage the use of Traffic Engineering by choosing a specific path for each request not necessarily being the Shortest Path to destination. In this example, a path consisting on three hops to node 4 is chosen when the Shortest Path would be to go through 3 directly to 4.

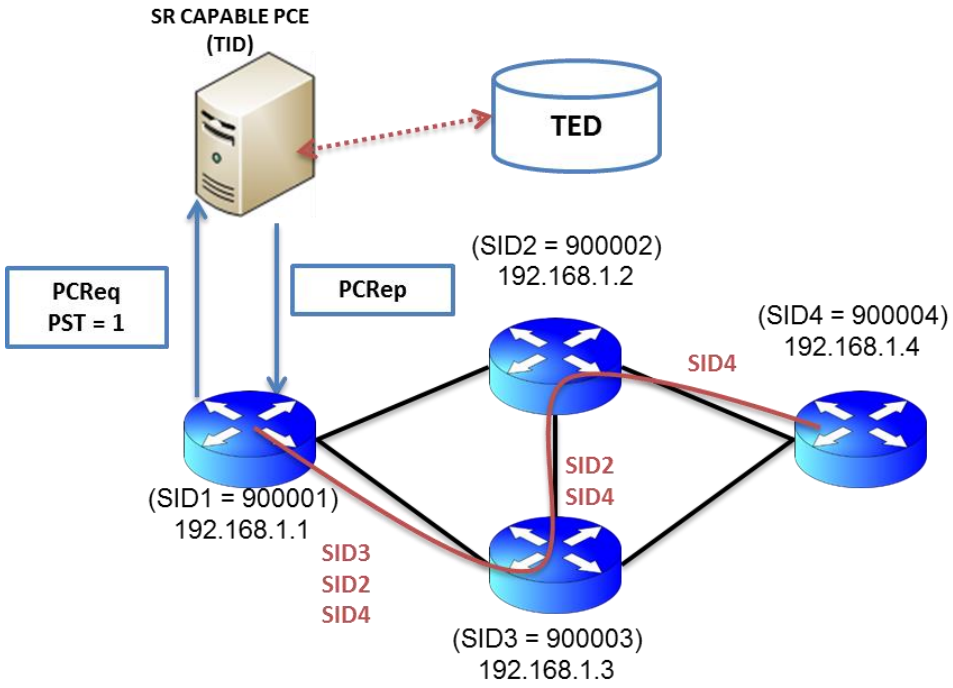


Figure 50: Source Routing using Node SIDs

The following sub-sections will show possible use cases that were taken into consideration for the Request-Reply scenario but where not actually validated.

5.2.2.1 *Possible use case 1: Explicit Route Provisioning using adjacency SIDs*

In this first possible use case, the reply message would consist in an explicit route provided in the form of a hop by hop, end-to-end path from the ingress node to the egress node seldom using AdjSIDs.

As shown in Figure 51, if we want to create a path to 4 using Adjacency SIDs we would need to include a SR tag of type adjacency (ST = 3) to the routed packet. This adjacency SID would represent a local service provided at node 2 that would send the packet through the interface that connects the latter to node 4.

It must be noted that adjacency SIDs are local to each node so their numeric value may be duplicated within the scenario although they each represent a different rule within their corresponding node.

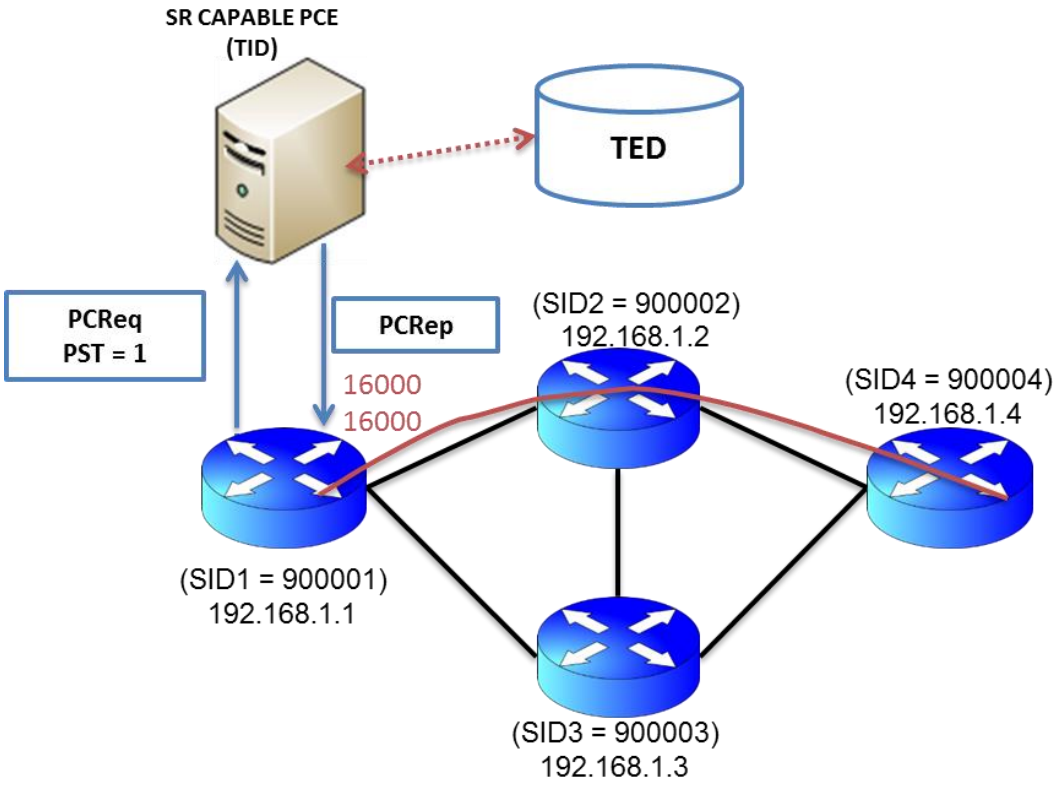


Figure 51: Route provisioning using adjacency SIDs

The use of adjacency SIDs will be validated in section 5.2.3 through the PCInitiate message.

5.2.2.2 *Possible use case 2: Explicit Route provisioning using a combination of Node and Adjacency SIDs*

This scenario uses a combination of node and adjacency SIDs to proof the local significance of the latter.

An adjacency SID is a local service provided by a certain node. It is local to that node; in other words, it is only meaningful when interpreted by the node issuing that tag.

Here, the packet will reach node 3 with tag 16000 which represents a local rule for 3 to send the packet to 2. Once at 2, local rule 16001 is applied sending the packet through the interface to 4.

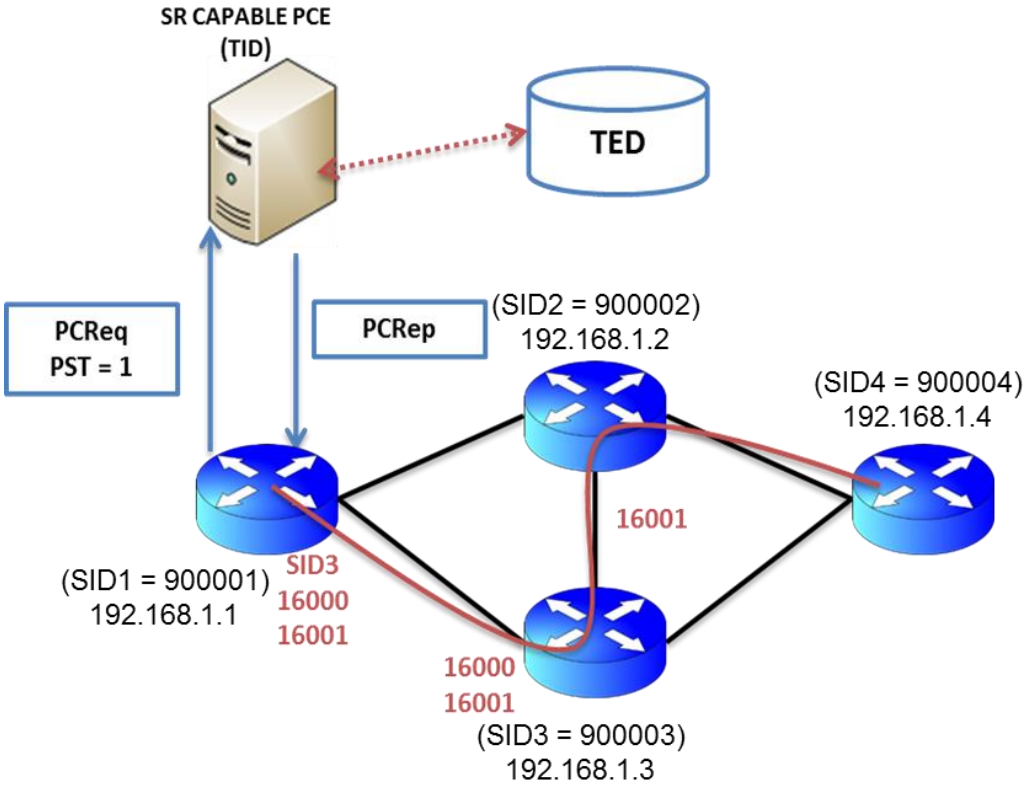


Figure 52: Path Provisioning using Node and Adjacency SIDs combined

5.2.3 Initiating a SR-Path from the PCE

In the following sub-sections the instantiation of SR paths using PCEP will be validated. It must be noted that as the validation was made against SR-capable Cisco routers, we had to adapt our implementation to meet their requirements.

The following table relates the message formats used by Cisco with the drafts where they are defined:

Table 3: Cisco's PCEP message format

PCEP Message	IETF draft
PCInitiate Message	draft-crabbe-pce-pce-initiated-lsp-00
PCRpt Message	draft-ietf-pce-stateful-pce-02

5.2.3.1 *Using node SIDs*

Figure 53 represents our active stateful PCE instantiating an end-to-end SR path along the domain. The chosen path was CISCO_1 to CISCO_4 but any existing combination of SIDs that form a valid path along the domain would also be valid.

In this case we have a path from router 1 to router 4 as in 5.2.2. It is identified by SID 900004. Together with the instantiation of the path we want to

validate that the format used in the PCInitiate message includes the Symbolic Path Name TLV as specified in the sivabalan-pce-segment-routing-02 draft. Furthermore, we want to proof that an SR-path instantiated by an incomplete stack of SIDs is routed through the shortest path to destination

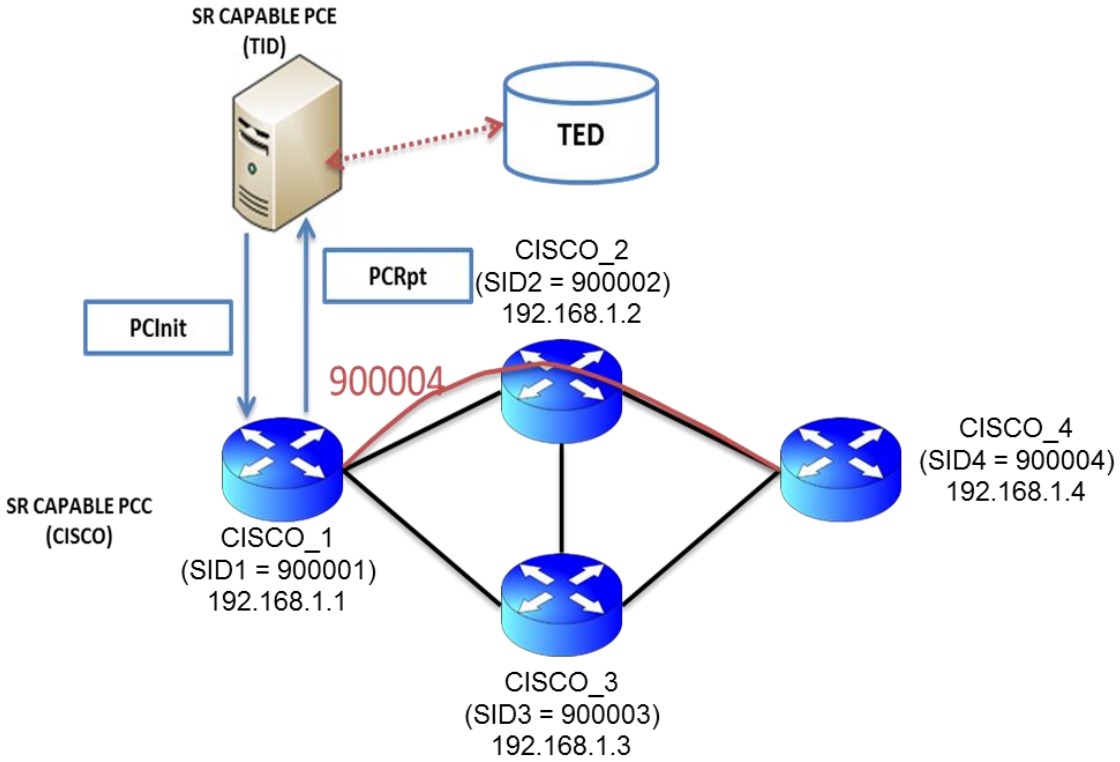


Figure 53: PCE Initiated SR-Path using Node SIDs

The exchanged messages between the PCE and the PCC are shown in Figure 54.

On the one hand, the PCInit (12[5]) carries the Path Setup TLV (666) set to 1 that indicates that SR is going to be used. In addition, it also carries the Symbolic Path Name TLV (17) that serves as a unique identifier for the path being instantiated. The SID carried in the SR-ERO sub-object (see footnote 4) is 900004 which is the node SID for CISCO_4 (see Figure 42).

On the other, the PCRpt (10[6]) message is sent back carrying the path that has been created using the Symbolic Path Name TLV. This is the way in which the PCC notifies the PCE that path *'foo1' (ASCII 0x666f6f31)* has been successfully created.

[5] The PCInitiate message is not decoded yet by the latest Wireshark developer version.
[6] The PCRpt message is not decoded yet by the latest Wireshark developer version.

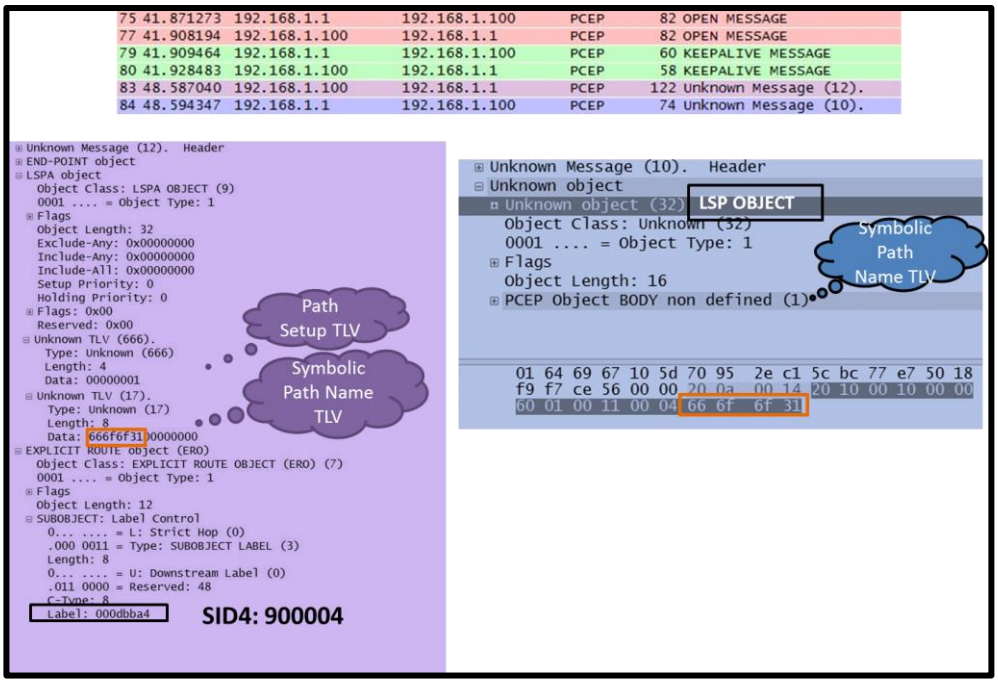


Figure 54: PCE Initiated SR path using node SIDs

If we recall Figure 5 where the Shortest Path in SR is illustrated we can see that this is the exactly the same case. CISCO_1 receives an incomplete SR to CISCO_4 and, consequently, routes the packet through the shortest path.

A capture of the router's user interface is shown in Figure 55 containing the created path, '*foo1*'.

```
Name: tunnel-te4  Destination: 10.0.0.4 (auto-tunnel pcc)
  Signalled-Name: autopcc_cisco1_t4
  Status:
    Admin:    up Oper: down   Path:  valid   Signalling: Segment Routed Tunnel

    path option 62664,  type
    Last PCALC Error: Thu Jul 10 10:13:22 2014
      Info: No path to destination, 10.0.0.4 (unknown)
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Thu Jul 10 10:13:22 2014 (00:00:04 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  0  0 Affinity: 0x0/0xffff
    Metric Type: TE (default)
    Hop-limit: disabled
    AutoRoute: disabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (default)
    Forwarding-Adjacency: disabled
    Loadshare:          0 equal loadshares
    Auto-bw: disabled
    Fast Reroute: Disabled, Protection Desired: None
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled

  SR path info:
      SID1: 900004

  Auto PCC:
    Symbolic name: foo1
    PLSP ID: 5
    Created by: 192.168.1.100
```

Figure 55: CISCO_1 to CISCO4 shortest path example

5.2.3.2 *Using Adjacency SIDs*

As a validation test of Figure 51, a SR path has been instantiated using Adjacency SIDs from CISCO_1 to CISCO_4. Moreover, it shows another way of creating a path to 4, different to that in 5.2.3.1.

In this case the PCE will send a PCInit message to the PCC containing each of the local services that need to be provided at the respective nodes in order to obtain the desired path.

In such a way CISCO_1 applies local rule 16000 setting 1.1.1.2 as next hop. In a similar way, CISCO_2 sets 2.2.2.2 as next hop following its local rule represented by Adjacency SID 16000.

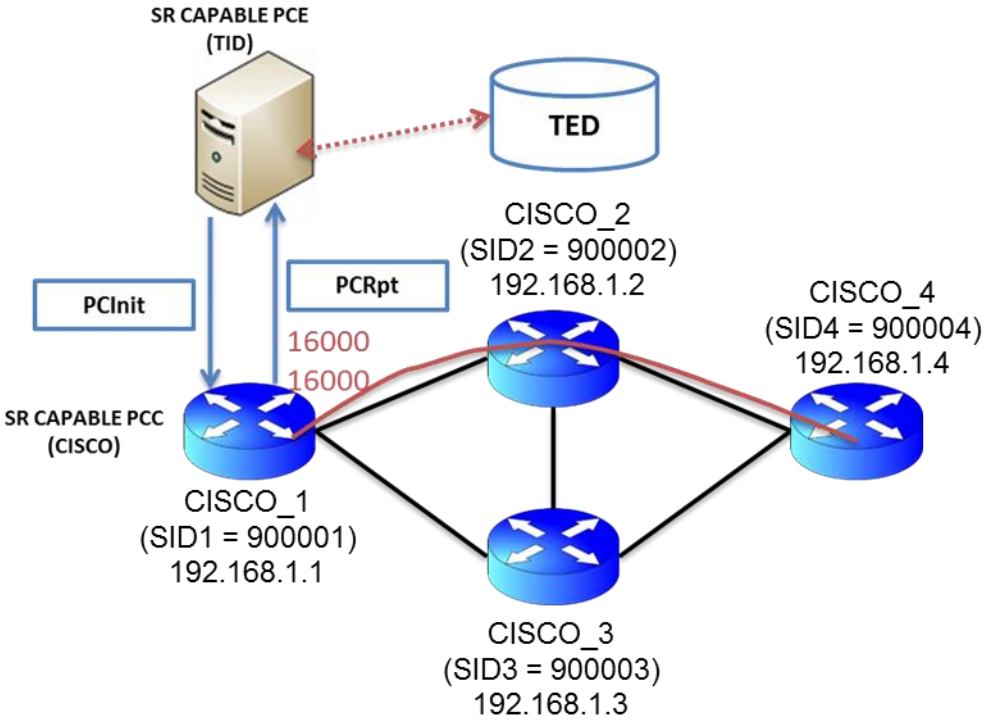


Figure 56: PCE Initiated SR-Path using AdjSIDs

Figure 57 shows the message trace. As it has already been mentioned, both the Path Setup TLV and the Symbolic Path Name TLV are present in the PCInit message. Highlighted in orange, the identifier for this new path is shown, *'foo' (ASCII 0x666f6f).*

The main difference in this case is that the ERO object is composed of two SR-ERO sub-objects representing each of the AdjSIDs. Consequently a packet entering node 1 will follow the explicit path signalled by these two tags.

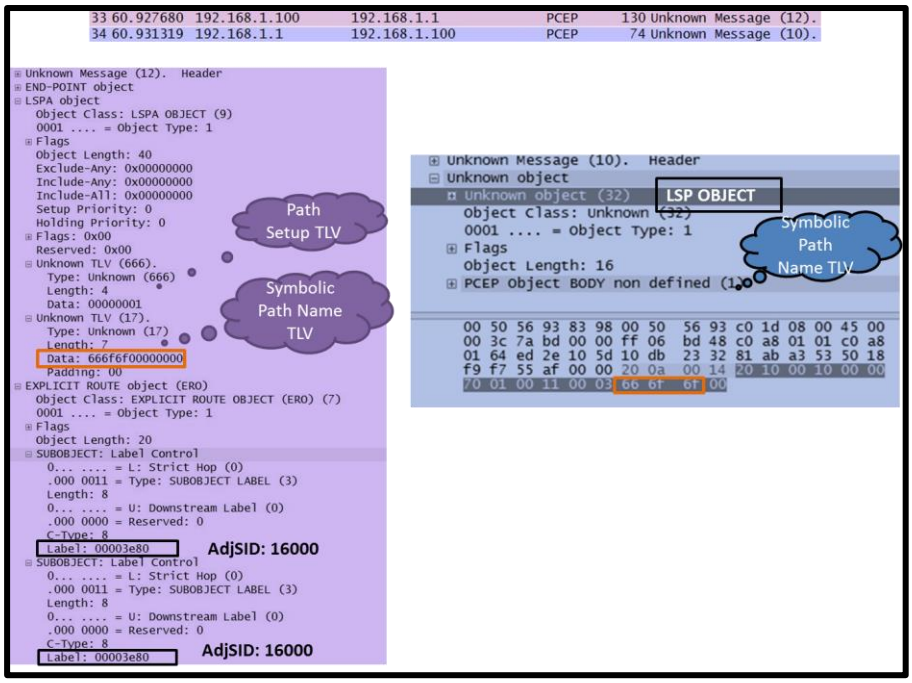


Figure 57: PCE Initiated SR path using Adjacency SIDs

As in the previous section, a picture of the router's interface is shown in Figure 58 proving this use case.

```
Name: tunnel-te5  Destination: 10.0.0.4 (auto-tunnel pcc)
  Signalled-Name: autopcc_cisco1_t5
  Status:
    Admin:    up Oper: down   Path:  valid   Signalling: Segment Routed Tunnel

    path option 42668,  type
    Last PCALC Error: Thu Jul 10 10:15:09 2014
      Info: No path to destination, 10.0.0.4 (unknown)
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Thu Jul 10 10:15:08 2014 (00:00:04 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  0  0 Affinity: 0x0/0xffff
    Metric Type: TE (default)
    Hop-limit: disabled
    AutoRoute: disabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (default)
    Forwarding-Adjacency: disabled
    Loadshare:          0 equal loadshares
    Auto-bw: disabled
    Fast Reroute: Disabled, Protection Desired: None
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled

  SR path info:
      SID1: 16000
      SID2: 16000

  Auto PCC:
    Symbolic name: foo
    PLSP ID: 6
    Created by: 192.168.1.100
```

Figure 58: CISCO_1 SR-Path to 4 using Adjacency SIDs

5.2.4 Error Reporting

These scenarios could not be tested as neither Cisco's PCC nor Telefonica's PCC supported SR error handling. However, this document contains the explanation about the error types to complete the SR explanation.

5.2.4.1 *Bad Label Value Error*

In Figure 59 the Bad Label Value PCErr message is illustrated. If a PCC receives a SID value that does not comply with the numbering rules of MPLS labelling, it must answer back with a Bad Label Value PCErr message.

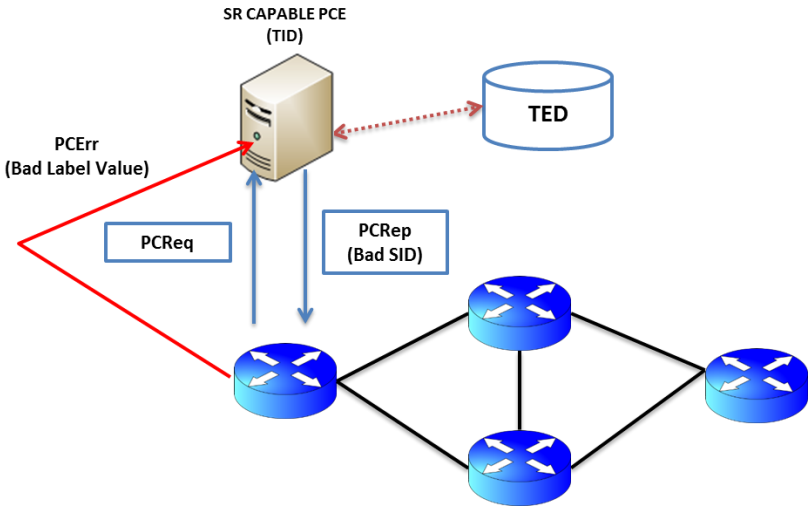


Figure 59: Bad Label Value Error

5.2.4.2 *Unsupported Number of SIDs Error*

If a PCC receives a stack of SR tags that exceed the maximum allowed, it must answer back with the Unsupported Number of SIDs PCErr. Figure 60 illustrates this procedure.

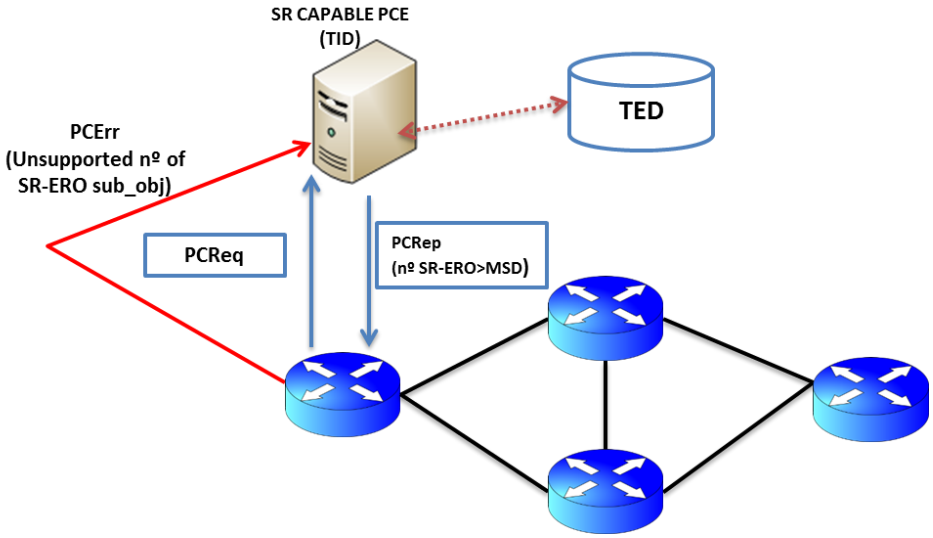


Figure 60: Unsupported Number of SIDs Error

5.3 BGP-LS Extensions for Segment Routing support

This section contains a theoretical explanation of the extensions to support SR in BGP-LS, but they have not been implemented due to the recent release of the draft.

As stated in section 2.2, Segment Routing defines end-to-end paths by concatenating together sequences of hops called "*segments*" that joined together form a complete route to destination. These segments are appended to the header of the data packet and will define the actions that need be taken by the following nodes to route the packet to the egress point.

Segment routing is enabled in a network by flooding the segments (SIDs) to the nodes in the network using the corresponding IGP protocol (OSPF/IS-IS) and their Segment Routing defined extensions. Segment Routing extensions for IGP protocols are defined through new TLVs that can be found in the following drafts:

- IS-IS: http://tools.ietf.org/html/draft-ietf-isis-segment-routing-extensions-01
- OSPFv2: draft-psenak-ospf-segment-routing-extensions-04
- OSPFv3: draft-psenak-ospf-segment-routing-ospfv3-extension-01

The area of application of these IGP extensions, as one can imagine, is IGP-wide meaning SIDs can only be flooded within the networks domain. This creates a limitation, preventing the use of Segment Routing in multi-domain scenarios. This is where Segment Routing extensions for BGP-LS come into action.

In Figure 13, BGP speakers export their domain topology to a "northbound" controller (PCE) so the latter can construct a path from the ingress node to the egress node in a multi-domain scenario. The same can be applied when using Segment Routing. By using Segment Routing extensions for BGP-LS, a network controller (PCE) can collect the segment information from all the nodes in the network and construct the SR stack that would be needed to route a packet successfully across the corresponding IGP areas.

These extensions that encode SR information are included as optional TLVs in the BGP-LS attribute. As in 4.2.2 we will only consider Node and Link attributes. Prefix attributes will not be used in this document as they are of no use for our approach.

5.3.1 SR Node Attribute TLVs

In Figure 61 the current Node attributes for Segment Routing are displayed.

TLV Code Point	Description	Length	IS-IS SR TLV/sub-TLV
1033	SID/Label Binding	variable	149
1034	SR Capabilities	variable	2
1035	SR Algorithm	variable	15

Figure 61: SR Node Attributes [13]

These TLVs will be included in the BGP-LS attribute field associated with the node NLRI that announced the information. They are all optional so therefore their use is up to the developer. Let us describe briefly the information carried by these TLVs:

- **SID/Label Binding TLV**: Advertises SID/label bindings and their associated primary and backup paths [13]. E.g. advertising paths from other protocols.
- **SR Capabilities TLV:** It announces that the node being described is SR capable. It also informs of the range of SID values it supports.
- **SR Algorithm TLV:** It allows the router to advertise the algorithm it is using to calculate the reachability to other nodes (e.g. SPF).

All three of these TLVs contain the SID sub-TLV uniquely identifying the router making the announcement.

5.3.2 SR Link Attribute TLVs

In Figure 62 the currently defined Link attributes for Segment Routing are shown.

TLV Code Point	Description	Length	IS-IS SR TLV/sub-TLV
1099	Adjacency Segment Identifier (Adj-SID) TLV	variable	31
1100	LAN Adjacency Segment Identifier (Adj-SID) TLV	variable	32

Figure 62: SR Link Attributes [14]

These TLVs will be included in the BGP-LS attribute field associated with the link whose local node flooded the information. For the purpose of this

document TLV 1100 will not be taken into consideration so we will focus solely in the Adjacency SID TLV.

The Adjacency SID TLV will carry the SID sub-TLV uniquely identifying the link between two SR-capable routers. Multiple AdjSIDs may be used per neighbour allowing the local node to announce several paths to the controller for the same destination. This opens the possibility of making use of load balancing techniques on behalf of the controller.

6 Conclusions

The first thing that comes to mind after finishing this project is that the main objective that we were seeking when we decided to move on with it has been utterly fulfilled. What started as a mere study about the state-of-art of current SDN techniques, derived in a throughout research process to proof the use of Segment Routing in centralized traffic-engineering scenarios.

On the one hand, we carried out a deep study of the available extensions to the Path Computation Element Protocol, analysing the possibility of adding new features to what already was one of the most advanced PCEs available. By making use of such extensions and our PCE, we discovered a whole new universe of possibilities that ultimately emanated in one the first validations of the SR architecture.

Especial mention must be given to the tests ran with the Cisco team. They were the first interoperability tests carried-out between an operator and a vendor of the Segment Routing technology. They not only provided feedback of the use of Segment Routing in centralized traffic-engineering scenarios but they also validated, for the first time, the use of an active stateful PCE with Segment Routing capabilities. Moreover, let us point out that some of the features that are not covered by Cisco's implementation have been tested with Telefonica's implementation.

The second contribution of this work is to upgrade Telefonica R&D's BGP-LS implementation to the latest version of the draft. Furthermore, it has been tested with external entities like CTTC, CNIT and Telecom Italia. This implementation not only supports IP/MPLS layer, but also optical layer extensions which are new contributions added by Telefonica

As it has been mentioned, our primary goal was to formalise the theory behind the Segment Routing architecture but as this project developed, new ideas emerged. One of them was the possibility of extending the novel BGP-LS protocol to make it compatible with Segment Routing. Consequently, a full description and validation of such protocol had to be carried out as there are not many implementations of it worldwide, and BGP is not known for carrying LS information outside the most specialized communities.

It must be noted that the extensions to make BGP-LS compatible with SR were presented from a theoretical point of view due to the recent appearance of the draft describing them.

Experimental proof of every element used during the project was provided. In addition, every section was backed up with working code including annexes to give any interested researcher the possibility of testing or further developing the existing architecture.

Once the general ideas of the project have been covered, please let us review the objectives of this project:

1. Define the Segment Routing technology.
2. Analyse the applicability of SR in different scenarios.
3. Analyse the different use cases defined for the Path Computation Element in order to apply them to SR.
4. Implement the latest BGP-LS draft.
5. Validate the exportation of topology using BGP-LS.
6. Extend the PCE Protocol in order to carry Segment Routing.
7. Study how such extensions carry Segment Routing information.
8. Test Segment Routing with a Path Computation Element using PCEP.
9. Present extensions to the BGP-LS protocol to make it compatible with SR.

Looking back at the goals in hand, we can claim that this work not only covers the original expectations, but also goes beyond thanks to the interoperability tests done with external entities (Cisco, CTTC, CNIT and Telecom Italia).

Further Lines of Research

We strongly believe that by completing this project we have made huge strides in the world of centralized networking setting the table for future developments of the paradigms addressed in this document. One of them would be to extend the use of Segment Routing to the use cases defined for BGP-LS.

As we have already seen, BGP-LS can carry SR information seldom adding a couple of TLV triplets to its Link-State attribute. Hence, we think that including SR in those use cases is not such a long shot.

A possibility would be to make use of Segment Routing in multi-domain scenarios with a hierarchical PCE. This hierarchical scenario would be constructed by different inter-connected domains, each of them having its own PCE.

Each PCE would have the possibility of exporting the SR topology and TE information through BGP-LS to the parent PCE creating a SR-TE database for the whole scenario. This use case would allow for optimum end-to-end path computations across multiple domains using SR technology.

Of course, this constitutes just one of the many applications the near futures of this technology presents but with our state of implementation and the bases that we have established in this project we firmly believe this could become a reality in the years to come.

References

1] M. Cuaresma, «Experimental Demonstration of H-PCE with BGP-LS in elastic optical networks, in European Conference on Optical Communication (ECOC),» 2013.

2] D. Isemberg, «Rise of the Stupid Network».

3] Kurzweil,«Scalometer,»Available:http://scalometer.wikispaces.com/singularity.

4] IETF, "Segment Routing with IS-IS Routing Protocol," 20 March 2013. [Online]. Available: http://tools.ietf.org/html/draft-previdi-filsfils-isis-segment-routing-02.

5] IETF, «Segment Routing Architecture draft v01,» 31 October 2013. [En línea].

6] I. R. 4655, «A PCE based architecture,» Octubre 2006. [En línea].

7] B. Draft, «IETF BGP Draft,» [En línea]. Available: http://tools.ietf.org/html/rfc4271#page-14.

8] IETF, «IDR-LS distribution,» November 2013. [En línea].

9] IETF, «IS-IS Extensions for Traffic Engineering - Proposed Standard,» 2008. [En línea].

10] IETF, «draft-ietf-ccamp-general-constraint-encode-05#section-3.2,» 2011. [En línea]. Available: http://tools.ietf.org/html/draft-ietf-ccamp-general-constraint-encode-05#section-3.2.

11] IETF, «PCE Segement Routing Extensions,» october 2013. [En línea].

12] IETF, «Setup Type TLV for PCE,» October 2013. [En línea].

13] IETF, «draft-previdi-isis-segment-routing-extensions-05,» February 2014. [En línea].

14] IETF, «idr-ls segment routing extensions,» November 2013. [En línea].

Acronyms

AS	Autonomous System
BGP-LS	Border Gateway Protocol – Link State
CoS	Class of Service
DDOS	Distributed Denial of Service
ERO	Explicit Route Object
FRR	Fast Re-Route
FSM	Finite State Machine
GMPLS	Generalized MPLS
IGP	Interior Gateway Protocol
IS-IS	Intermediate System – Intermediate System
LDP	Label Distribution Protocol
LER	Label Edge Router
LSP	Label Switched Path
LSR	Label Switch Router
MPLS	Multiprotocol Label Switching
NAI	Node or Adjacency Identifier
NLRI	Network Layer Reachability Information
OSPF	Open Shortest Path First
PCC	Path Computation Client
PCE	Path Computation Element
PCEP	Path Computation Element Protocol
RSVP-TE	Resource Reservation Protocol – Traffic Engineering
SDN	Software Defined Networking
SID	Segment Identifier
SPT	Shortest Path Tree
SR	Segment Routing
TE	Traffic Engineering
TED	Traffic Engineering Database
TLV	Type Length Value
VPN	Virtual Private Network

Annex A: Steps to test Segment Routing using Telefonica R&D's PCE

This annex will detail the steps that need to be taken in order to test the Segment Routing technology with Telefonica R&D's Path Computation Element.

The first step in the process is to prepare the configuration file in the PCE. The name of the file is PCEServerConfiguration.xml and the relevant parameters for this task are:

```
<isStateful>true</isStateful>
    <isSRCapable>true</isSRCapable>
    <MSD>0</MSD>
    <isActive>true</isActive>
    <statefulDFlag>false</statefulDFlag>
    <statefulSFlag>false</statefulSFlag>
    <statefulTFlag>false</statefulTFlag>
    <networkDescriptionFile>topologia_pce1.xml</networkDescriptionFile>
```

These parameters will configure the PCE to be active and stateful with Segment Router Capabilities. Parameter 'networkDescriptionFile' contains the topology for the considered scenario.

As explained in section 5.2 the tests were run in a scenario composed of 4 SR-capable cisco routers. Following, the steps needed to instantiate a SR-path are explained.

1. Connect to ciscogw: This is the virtual machine were the PCE is running. It is virtually connected to each of the four cisco routers (PCCs).
    - *ssh ciscogw*
2. Connect to one of the routers (PCC)
    - *telnet cisco1*
3. Open the session between the router and the PCE
    - In the exec menu of the cisco router run: *mpls traffic-eng pce activate-pcep all*

Information displayed in the PCE when opening the PCEP session should look like the following:

```
Information: open object saved: Ver: 1 Flags: Parent PCE Indication Bit: falseParent PCE Request Bit: false Keepalive30 Deadtimer: 60 SID: 2
jun 12, 2014 12:47:22 PM tid.pce.pcepsession.GenericPCEPSession initializePCEPSession
Information: KeepAlive Message Received
jun 12, 2014 12:47:22 PM tid.pce.pcepsession.GenericPCEPSession initializePCEPSession
Information: Entering STATE_SESSION_UP
jun 12, 2014 12:47:22 PM tid.pce.server.DomainPCESession run
Information: PCE Session succesfully established!!
jun 12, 2014 12:47:22 PM tid.pce.server.DomainPCESession run
Information: Received Report message
```

4. Enter in the PCE console
   - *telnet 192.168.1.100 6666*
   - Select 8) send initiate
5. Create a path from the router:

```
PCE:>8
Choose origin IP:
PCE:>10.0.0.1
Choose dest IP:
PCE:>10.0.0.3
Enter SID:
PCE:>16001
Enter Sybolic Name for new path:
PCE:>NewPath
Enter ID:
PCE:>4
```

Information that should be displayed in the PCE is the following:

```
Information: Starting Management session
jun 12, 2014 12:48:30 PM tid.pce.pcep.messages.PCEPInitiate encode
Information: Empezando enconde
jun 12, 2014 12:48:30 PM tid.pce.pcep.objects.EndPointsIPv4 encode
Information: Encoding EndPointsIPv4
jun 12, 2014 12:48:30 PM tid.pce.pcep.objects.tlvs.SymbolicPathNameTLV encode
Information: Encoding SymbolicPathName TLV
jun 12, 2014 12:48:30 PM tid.pce.pcep.objects.PCEPIntiatedLSP encode
Information: Leeeength:::60
jun 12, 2014 12:48:30 PM tid.pce.pcep.objects.PCEPIntiatedLSP encode
Information: SRERO leength:: 12
jun 12, 2014 12:48:30 PM tid.pce.pcep.messages.PCEPInitiate encode
Information: CCCC
jun 12, 2014 12:48:30 PM tid.pce.pcep.messages.PCEPInitiate encode
```

```
Information: DDDD 64
jun 12, 2014 12:48:30 PM tid.pce.pcep.messages.PCEPInitiate encode
Información: Vamos a por el encodeHeaer
jun 12, 2014 12:48:30 PM tid.pce.pcep.messages.PCEPInitiate encode
Information: a por array copy
jun 12, 2014 12:48:30 PM tid.pce.server.DomainPCESession run
Information: Received Report message
jun 12, 2014 12:48:31 PM tid.pce.server.DomainPCESession run
Information: Received Report message
```

To see the path status run "show mpls traffic-eng tunnels" in the router:

```
RP/0/0/CPU0:cisco1#show mpls traffic-eng tunnels
Thu Jun 12 10:49:59.851 UTC


Name: tunnel-te4  Destination: 10.0.0.3 (auto-tunnel pcc)
  Signalled-Name: autopcc_cisco1_t4
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 45280,  type   (Basis for Setup, path weight 10)
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Thu Jun 12 10:48:30 2014 (00:01:29 ago)
  Config Parameters:
    Bandwidth:       0 kbps (CT0) Priority:  0  0 Affinity: 0x0/0xffff
    Metric Type: TE (default)
    Hop-limit: disabled
    AutoRoute:  enabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (default)
    Forwarding-Adjacency: disabled
    Loadshare:         0 equal loadshares
    Auto-bw: disabled
    Fast Reroute: Disabled, Protection Desired: None
    Path Protection: Not Enabled
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
  Auto PCC:
    Symbolic name: NewPath
    PLSP ID: 5
    Created by: 192.168.1.100
  History:
    Tunnel has been up for: 00:01:28 (since Thu Jun 12 10:48:31 UTC 2014)
    Current LSP:
      Uptime: 00:01:28 (since Thu Jun 12 10:48:31 UTC 2014)
```

```
  Path info (IS-IS 1 level-1):
  Node hop count: 1
  Hop0: 5.5.5.1
  Hop1: 10.0.0.3
Displayed 1 (of 1) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

Annex B: Steps to tests Segment Routing with TID-PCE & TID-PCC

This annex will provide the necessary steps to configure a SR path using Telefonica R&D's PCE.

The first step in the process is to prepare the configuration file in the PCE for this scenario. The name of the file is PCEServerConfiguration.xml and the procedure is exactly the same as in Annex A.

The PCE is running in server Simpsonsgw1 which is connected to 4 mini PCs acting as the nodes in the network.

1. Once in Simpsonsgw1 access to the execution file and run the scenario.
   - cd execution
   - ./restart_emulation.sh Simple4IPSR
2. Access to one of the nodes and enter the LSP Management menu.
   - telnet 192.168.1.1 6666

```
ROADM Main Menu:

    1) Configure ROADM (WSON)
    2) Configure ROADM (Flexigrid)
    3) Turn off the ROADM
    4) LSPs Management NODE
    5) Show Topology NODE

    ENTER) quit

Please, choose an option
ROADM:>4
```

3. Choose option 2 to set an LSP.

```
Node Management Main Menu:

Available commands:

1)show LSPs
2)set LSP
3)teardown LSP
4)help
```

```
5)set traces on
6)set traces off
7)back
8)print eros
9)quit

NODE:>2
```

4. Insert the destination IP address.

```
You chose ADD LSP
Insert the Destination Node ID Please: 192.168.1.4

Insert the bandwidth Please: 100

Insert Bidirectionality Please (yes/no): yes

LSP being established
```

5. Check that the path has been established.

```
NODE:>show LSPs

LSP id: 1  ---->  Source: /192.168.1.1 - Destination: /192.168.1.4
```