POLITECNICO DI MILANO

Dipartimento di Elettronica, Informazione e Bioingegneria

Master of Science

Telecommunications Engineering

SEGMENT ROUTING PRINCIPLES AND APPLICATIONS FOR SDN

Author:
Ana Kos

Supervisor:
Prof. Maier Guido Alberto

816555
2013-2015

Abstract

Internet has been an essential part of modern society for a long time now. Some regulators are even trying to standardize it as a utility; meaning that it has become as vital for our lives as power and water at home. Internet is evolving constantly, improving to speeds never imagined in the past decades.

Businesses that use Internet are evolving as well. New business requirements for their Internet traffic are pushing industry and network service providers to come up with new solutions, to address these challenges. Software Defined Networks (SDN) and Segment Routing is one of these tools that help network service providers to tackle problems existing in the networking world. SDN is already well-established paradigm in the industry, making it easier to manage the network.

Segment Routing is the source-routing paradigm addressing existing drawbacks of Multiprotocol Label Switching (MPLS) networks in terms of simplicity, scalability and manageability. By reusing and extending existing protocols, Segment Routing enables network providers to transport traffic using specific or custom routes, which is different from normal shortest path.

Ability to engineer traffic flows with such granularity can give network service provider great business benefits. Segment Routing lays the foundation for application engineered routing. It enables one to easily address customer's dynamic traffic requirements from high-level application.

This thesis work presents a research and experimental analysis of Segment Routing principles and its application in Software Defined Networks. The main underlying technologies and protocols were analyzed within this work. The work summarize all technical requirements need for employing Segment Routing in the SDN network.

In the scope of this thesis I implemented three virtual networks for testing Segment Routing properties. In diverse network topologies Segment Routing was tested from different aspects. In this work I investigated Segment Routing performance capabilities, scalability properties and multi-domain application.

Contents

Chapter 1. Introduction ---- 1
1.1 Main Contributions ---- 3
Chapter 2. State of the Art ---- 5
Chapter 3. Technology Overview ---- 9
3.1 Source Routing ---- 9
3.2 Segment Routing ---- 10
3.2.1 Segment Routing Concepts ---- 11
Segment ---- 11
Segment Advertising ---- 11
Global and Local Segments ---- 12
IGP Segment Identifiers – IGP-SIDs ---- 12
Prefix-SID ---- 13
Adjacency-SID ---- 14
Routing Operations ---- 15
3.2.2 Segment Routing Use Cases ---- 16
Simplified transport of MPLS services ---- 16
Segment Routing and LDP coexistence ---- 17
Traffic Engineering ---- 17
Fast Reroute ---- 19
3.3 Intermediate System to Intermediate System ---- 19
3.3.1 IS-IS extensions for Segment Routing ---- 22
Prefix-SID sub-TLV ---- 22
Adjacency-SID sub-TLV ---- 23
SID-Label Binding TLV ---- 23
Capabilities Sub-TLV ---- 24
3.4 Border Gateway Protocol ---- 24
3.4.1 BGP in Segment Routing - concepts and extensions ---- 26
3.4.2 BGP-Link State ---- 28
3.4.3 BGP-LS extensions for Segment Routing ---- 31
3.5 Path Computation Element and Path Computation Element Protocol ---- 32
3.5.1 PCE Architecture ---- 33
PCEP session ---- 33
Stateful and Stateless PCE ---- 34
3.5.2 PCEP extensions for Segment Routing ---- 36
SR-PCE capability TLV ---- 36
The new ERO and RRO sub-object ---- 37
3.6 MultiProtocol Label Switching ---- 37
3.6.1 Segment Routing vs. MPLS ---- 38
Chapter 4. Operating Environment ---- 42
4.1 Software Defined Networks ---- 42
4.1.1 The SDN architecture ---- 43
4.1.2 Benefits of SDN ---- 45
4.2 Segment routing and SDN ---- 45

4.3 OpenDaylight ---- **46**
4.3.1 ODL Architecture ---- 47
Southbound Interface ---- 48
Control Layer ---- 48
Northbound Interface ---- 49
4.4 Virtual Network ---- **49**
4.4.1 VMware ---- 50
VMware networking features ---- 50
4.4.2 Cisco IOS XRv ---- 51
IOS XRv Features ---- 51
CISCO XRv Benefits ---- 52

Chapter 5. Implementation ---- **54**
5.1 Implementation Overview ---- **54**
5.2 Environment Setup ---- **56**
5.2.1 Network Build-Up ---- 56
5.2.2 OpenDaylight Setup ---- 58
5.2.3 Web Based REST API client ---- 59
5.3 Network Configuration ---- **60**
5.3.1 Network 1 - Segment Routing ECMP test-bed ---- 60
Router Configuration ---- 61
OpenDaylight Configuration ---- 67
Network try-out ---- 67
5.3.2 Network 2 - Multi-domain Segment Routing test-bed ---- 68
Router Configuration ---- 69
OpenDaylight Configuration ---- 73
Network try-out ---- 73
5.3.3 Network 3 - Performance Analysis test-bed ---- 75
Router Configuration ---- 76
OpenDaylight Configuration ---- 80
Network try-out ---- 80

Chapter 6. Results ---- **81**
6.1 ECMP Segment Routing ---- **81**
6.2 Performance Analysis ---- **84**
6.2.1 Test without Segment Routing ---- 85
6.2.2 Test with Segment Routing ---- 86
6.3 Multi-domain Segment Routing ---- **88**
6.4 Network Limitations ---- **93**

Chapter 7. Conclusion ---- **94**
7.1 Future work ---- **96**
7.2 Acknowledgements ---- **97**

References ---- **98**

List of figures

Figure 3.1: IGP-SID classification
Figure 3.2: Node-SID example
Figure 3.3: Adjacency-SID example
Figure 3.4: Segment Routing Operations
Figure 3.5: Simplified MPLS transport
Figure 3.6: Traffic Engineering - Path Avoidance
Figure 3.7: Using Anycast-SID in Dual-Plane networks
Figure 3.8: IS-IS hierarchical architecture
Figure 3.9: Prefix-SID sub-TLV format for IS-IS message
Figure 3.10: Adjacency-SID sub-TLV format for IS-IS message
Figure 3.11: Label-Index TLV in BGP message
Figure 3.12: Originator-SRGB TLV in BGP message
Figure 3.13: BGP-LS distribution - Direct / Route Reflector
Figure 3.14: TLV format
Figure 3.15: Node NLRI format
Figure 3.16: The Link NLRI format
Figure 3.17: The topology IP prefix NLRI format
Figure 3.18: PCEP session flow
Figure 3.19: SR-ERO Sub-object format
Figure 4.1: SDN architecture
Figure 4.2: OpenDaylight architecture
Figure 4.3: Cisco IOS XRv Router Virtual Form Factor
Figure 5.1: VMware settings
Figure 5.2: Edit VMware configuration file
Figure 5.3: PuTTY settings
Figure 5.4: Postman interface
Figure 5.5: Network 1 topology
Figure 5.6: Network 1 interfaces configuration
Figure 5.7: Network 1 IS-IS configuration
Figure 5.8: Network 1 MPLS configuration
Figure 5.9: Network 1 BGP configuration at node N4
Figure 5.10: Network 1 ODL configuration
Figure 5.11: Network 1 test - pinging from client to server 1
Figure 5.12: Network 1 - show MPLS forwarding table
Figure 5.13: Network 2 - Multi-domain Network
Figure 5.14: Network 2 - static route at N1
Figure 5.15: Network 2 - BGP configuration of router N4
Figure 5.16: Network 2 - additional IS-IS instance
Figure 5.17: Network 2 - ODL configuration for multi-domain network
Figure 5.18: Network 2 - traceroute from N1 to N8
Figure 5.19: Network 2 - MPLS forwarding table - router N4

Figure 5.20: Network 2 - N4 IP route
Figure 5.21: Network 3 - topology
Figure 5.22: Network 3 - interface configuration R1
Figure 5.23: Network 3 - IS-IS configuration R1
Figure 5.24: Network 3 - MPLS configuration R1
Figure 5.25: Network 3 - BGP configuration R4
Figure 5.26: Network 3 - traceroute from client to server 1
Figure 5.27: Network 3 - MPLS forwarding table
Figure 6.1: ECMP testing
Figure 6.2: ERO pushed from application to the network
Figure 6.3: Traceroute - ECMP testing
Figure 6.4: ECMP paths
Figure 6.5: Network scalability
Figure 6.6: MPLS vs Segment Routing states number
Figure 6.7: Performance analysis scenario
Figure 6.8: Performance analysis – test without Segment Routing
Figure 6.9: Results of performance analysis – test without Segment Routing
Figure 6.10: Results of performance analysis – test with Segment Routing
Figure 6.11: Network performance with Segment Routing
Figure 6.12: Multi-domain network
Figure 6.13: Multi-domain example 1
Figure 6.14: Traceroute for multi-domain example 1
Figure 6.15: Multi-domain example 2
Figure 6.16: Traceroute for multi-domain example 2
Figure 6.17a: Multi-domain example 3a
Figure 6.17b: Multi-domain example 3b
Figure 6.18: Router static

List of tables

Table 3.1: Comparison - Source Routing vs. MPLS
Table 5.1: Network 1 - interfaces and SIDs
Table 5.2: Network 2 - interfaces and SIDs
Table 5.3: Network 3 - interfaces and SIDs

Chapter 1. Introduction

Segment Routing (SR) is a new source routing paradigm. It is a network technology that wants to address several drawbacks of existing IP/MPLS networks in terms of scalability, simplicity, and ease of operation [1]. Segment Routing is a basis of application engineered routing. Application engineered routing is a new business model that can enable applications to direct behavior of network. It is a paradigm designed and built for SDN era. Segment Routing is being standardized by Internet Engineering Task Force under Source Packet Routing in Networking (SPRING) group [2].

Segment Routing enhances packet-forwarding behavior. It allows network to carry packets via specific forwarding path. This path can be different from natural shortest path that packet usually takes inside the network. Having control to set up custom forwarding paths opens up much use case scenarios that certain applications can benefit from.

Source-based routing is not a brand new idea in the world of networking, but it has not seen widespread adoption. A node (usually a router or a switch), which steers packets using list of ordered instructions is called segment. Segment Routing allows network to steer traffic via any topological flow.

One of the key advantages of Segment Routing is its simplicity. It only takes few lines of configuration to enable it on the router. Segment Routing is particularly useful in the era of SDN, where new business needs require new and finer granularity of traffic engineering and differentiation. Segment Routing can be deployed in the network step by step, there is no need to make massive upgrades inside the network. It can be integrated with existing MPLS network, since it is interoperable with existing MPLS control and data planes.

Today's traffic engineering solutions, such as Resource Reservation Protocol – Traffic Engineering (RSVP-TE) requires signaling for each path, and state of the each path needs to be present on each node that traffic traverses. Segment Routing can implement all these without the need of signaling protocol, making its architecture simpler and more scalable.

Segment Routing using MPLS data plane, does not require Label Distribution Protocol (LDP) or RSVP-TE. Labels are distributed using Interior Gateway Protocol either Intermediate System-to-Intermediate System (ISIS) or Open Shortest Path First (OSPF) and BGP. Running fewer protocols inside the network already makes network more stable and scalable. Segment Routing paths are protected with Fast Reroute (FRR) capability, that allows rerouting of traffic in under 50 milliseconds, in case of link or node failure.

Traditionally routers guide traffic inside the network primarily based on destination IP. Underlying Interior Gateway Protocol (IGP) was used to distribute network topology and compute shortest path from ingress to egress node. However, nowadays packet loss, jitter, delay, and available bandwidth have become major business differentiator when creating service-level agreements (SLAs). Therefore, this new business requirements are pushing networks to evolve towards more agility and flexibility.

Multiprotocol Label Switching (MPLS) introduced tunneling mechanism and traffic steering functions [3]. These were main reasons behind success of the MPLS technology. MPLS introduced MPLS based Virtual Private Networks (VPN).

However, Resource Reservation Protocol-Traffic Engineering (RSVP-TE) did not have same popularity as MPLS VPN. One of the main reasons of this was having poor load balancing characteristics. Another reason was that it was not very scalable. Final reason was that computation was distributed and this was causing some unpredictable traffic patterns and not optimal use of resources.

Nowadays network operators are demanding more flexible, agile, scalable, and simple network architectures. These architectures will help them to

implement application-centric networking and cloud-based services, which are increasingly demanded functionality on the market. Segment Routing was created to address this very issue and evolve network in the era of Software Defined Networking.

Target audience for Segment routing are mainly Internet Service Providers (ISP), content providers, over-the-top (OTT) providers, large enterprises, data centers, and others.

1.1 Main Contributions

MPLS-TE already exists as traffic engineering solution. However, it has drawbacks in terms of scalability, manageability, and uses heavy signaling protocols such as RSVP-TE and LDP. Segment Routing overcomes these drawbacks and enables network service providers to change network behavior dynamically.

However, in order to achieve this some tools and protocols needs to be present and enabled inside the network. In particular: SDN controller and protocols such as BGP, BGP-LS, IGP (IS-IS), PCEP, MPLS with Segment Routing.

SDN controller is needed to have a global view of the network, communicate messages and commands back and forth with network devices. It acts as a medium between high-level application and network devices.

BGP-LS and IGP protocols are needed to extract link state information from the network. This data includes link bandwidth, metric, delay, and more. This data is readily accessible by SDN controller, and therefore by high-level application.

Path Computation Element (PCEP) needs to be present in the network in order to calculate suitable paths and then push the path onto the network node using SDN controller.

Segment Routing can enable traffic engineering in three possible ways:

- By manually creating Segment Routing Label Switched Paths and explicitly defining route inside the network. This equivalent of MPLS-TE but without extra protocols
- By manually creating Segment Routing tunnels. Path is calculated by Path Computation Element (PCE) and later pushed by SDN Controller onto the network
- Dynamically create Segment Routing tunnels. Path is calculated by PCE using existing network information or SLA, such as delay, bandwidth, metric etc.

As technology and as a standard, Segment Routing is at very early stages of development. Only couple of demo implementations by major networking equipment manufacturers exists, and they are only available for very specific customers.

My thesis presents the research about Segment Routing and discusses: its underlying technologies (MPLS, SDN) and protocols (BGP, BGP-IS, IS-IS, PCEP). The listed protocols are essential for Segment Routing. In this work they are analyzed in detail together with defined protocol extensions for Segment Routing.

In my thesis I brought together all above mentioned tools to provide implementation of Segment Routing inside the virtual network of Cisco routers. For this thesis I implemented three different Segment Routing networks suitable for testing Segment Routing technology from different aspects.

This thesis has the aim to demonstrate and prove the major Segment Routing advantages. This thesis brings results on:

- Segment Routing performance benefits
- Segment Routing scalability
- Segment Routing multi-domain performance

Chapter 2. State of the Art

Nowadays the requirement for dynamic end-to-end service provisioning has been put into focus. The demand is driven by a need to interconnect geographically network elements and services. The end-to-end path can usually span over multiple domains. However, different domains can diverse from many aspects. Different domains can have different transmission and switching technologies but also different options in control plane. Heterogeneous control plane internetworking is the big challenge for ISPs that needs to be solved. [4] The solution must address the problems that come in multi-domain scenario such as limited topology visibility.

Software Defined Networking (SDN) is a new networking paradigm that promises that tends to solve control plane issues. SDN promotes the separation of control functions from the traditional forwarding elements. [5]. Control functions are centralized and placed onto dedicated entity called SDN controller. SDN architecture brings new network applications with and abstracted centralized view of network state. SDN employment can result in better network capacity utilization and improvement in terms of delay and traffic loss.

Software Defined Networking will change the way network operators work. Centralized network control paves the way for network programmability that opens a spectrum of new network applications. Some of emerging use cases are enhanced network management, monitoring and measurement, cloud orchestration, application traffic engineering, load balancing in data center. [6]. SDN is seen as the key technology for the next-generations networks.

SDN offers innovative networking platform [7]. The shift from legacy networks with distributed control plane to Software Defined Networks can be painful for ISPs. SDN interoperability with legacy devices attracted huge attention in networking world. However, existing employments of SDN are still limited and existing prototypes are still premature to offer confidence to real world deployment.

It is certain that SDN adoption will come incrementally in the close future. [8] is the one of the first scientific researches that investigates the network performance issue by migrating from traditional to SDN network. The work is focused on cooperation between SDN-capable forwarding elements and legacy equipment. The results have shown that improvements come even by employing a few strategically placed SDN forwarding elements into legacy network.

In SDN architecture SDN controllers are distinct from switches. The communication between switches and controller is enabled by a specific protocol. OpenFlow is the first standardized protocol that specifies the rules for communication between SDN controller and forwarding elements. It allows remote administration of a layer 3. OpenFlow enables programmability of forwarding behavior of different switches and routers [9] [10].

OpenFlow is being extended to support traditional traffic engineering services such as IP/MPLS. [11]. MPLS is today one of the most important technologies for ISPs. MPLS has introduced VPN and ability to support QoS. MPLS has been widely adopted among ISPs to establish dedicated paths for traffic engineering, load balancing, link protection, VPN and etc. [12]. However MPLS relies on distributed control plain. Protocols such as RSVP-TE and LDP are used to signal MPLS paths in the network and to fill up routing tables. Even though MPLS label based forwarding seems to be a good match with OpenFlow, their operability is very sensitive question. OpenFlow .1.0 does not support MPLS.

Recent researches put many effort to find the mechanism to integrate OpenFlow with MPLS. Transition to SDN requires simultaneous support development for legacy equipment and technologies. In [13] tunneling splicing mechanism is proposed for integrating OpenFlow with MPLS in the heterogeneous networks. The mechanism opens possibility to establish tunnels in hybrid networks. Hybrid networks are result of migrating from traditional to Software Defined Networks and they employ heterogeneous devices.

Recently IETF has proposed a traffic engineering technology that simplifies IP/MPLS operation and enables easier integration with centralized control architecture. Segment Routing is a new technology designed for SDN era [14]. Segment Routing does not depend on distributed signaling protocols and it easily fits the SDN concepts. However, even though is built with SDN in mind, Segment Routing is compatible with traditional IP/MPLS networks.

In scientific literature there is limited number of work that has been done regarding Segment Routing technology. The work [15], discusses SR implementation in both SDN and traditional IP/MPLS network. Firstly, it examines the Segment Routing tunnel setup in SDN environment where nodes are OpenFlow switches. They used SR-Controller that was developed from RYU framework by adding new modules such as request handler, network tracker, per flow monitor and SR engine. The second network is not software defined but it has centralized path computation entity (PCE) that creates the routes in traditional IP/MPLS network. Their scenario examines flow rerouting with aim of better resources utilization. Both implementations were successfully utilized to demonstrate dynamic packet rerouting enabled by enforcing different segment list at ingress node. Flow reroute is performed without any signaling protocol and with no packet loss.

In [16] SR technology is applied to multi-layer network – packet over optical network. Also in this work network control is centralized by a SDN controller. The controller controls the label stacking configuration and enables dynamic

traffic adaptations without requiring GMPLS operation. In the scope of this work the testbed network is built of packet capable switches (OpenvSwitch) and two ROADMS. To reach the destination a flow can be routed through the packet network, or it can be pushed to optical bypass. The path the flow will take depends on the predefined policy. The work exanimated how Segment Routing could be used to control the path that a flow will take through the network, depending on policy criteria. Additionally, this work presents scalability performance on time delay introduced by path computation. Tests have shown that if the label stack is deep the time increases approximately for 1ms due to stack computation at SDN controller. However this delay did not introduce any performance degradation.

In [17] investigates Software Defined Network using Carrier Ethernet and Segment Routing (CESR) in Tier-1 Service Provider. One of the work focuses is put on provisioning of the new SDN enabled services such bandwidth on demand and automated bandwidth sharing between enterprises and residential users. The results have shown that by employing SDN and CESR, the service provider can dynamically provision bandwidth to enterprise customers and remaining bandwidth is used for the best-effort service for residential users.

The research [18] has implemented algorithms for flow assignment that tries to minimize overall network crossing time. The heuristic this work proposes consists of two phase search. The first is Constrained Shortest Path First and the second is re-assignment phase that optimize existing paths with aim to decrease crossing time. SDN controller is equipped with modules capable to provide Segment Routing traffic engineering. The algorithm was tested on the large scale topology (over 150 nodes) and it showed improvement in terms of the path length. After applying the proposed algorithm the mean path length is decreases in terms of number of segment identifiers. That means the majority of total flows in the network gets the shortest path through the network. Results are valid also when network is saturated.

Chapter 3. Technology Overview

This chapter explains concepts and use cases of Segment Routing in more detailed manner. Moreover, one can find analysis of all protocols and technologies necessary for Segment Routing deployment.

3.1 Source Routing

Routing is the process of selecting the best path through the network for incoming data flows [19]. Usually, path is guided through the network according to its destination IP address. Such routing method is called destination based routing and it's commonly used in traditional IP networks. Once a router receives a packet, the router checks the packet's destination IP address and consults its routing information base (RIB). Once it finds a suitable IP address match, a router forwards a packet to a proper port and packet is forwarded towards the destination IP address.

In contrast of traditional destination based routing, one could implement source routing in a network. Source routing is a method where a sender of a packet specifies the route that a packet should take through the network [20]. When the packet with source routing travels through the network, a transit router routes a packet by inspecting the path information encoded in packet by source router. A source router must be aware of network layout in order to specify the route to the destination.

Source routing allows a router to determine a path partially or completely. When sender determines exact path that mechanism is called Strict Source and Record Routing (SSRR) [21]. This approach is rarely used. A common case of source routing is called Loose Source and Recorded Routing (LSRR),

where sender provide one or more intermediate hops that packet must visit on its path to the destination.

The main benefit of source routing is that intermediate nodes do not have to keep route information in RIB because the forwarding steps are specified in the data packet. Source routing enables easier network troubleshooting, enhances traceroutes and increases overall network performance. Software Defined Network can also be improved when source routing is used in the data plane. Source routing can minimize communication between SDN controller and switches when new flow is set up.

3.2 Segment Routing

Segment Routing is a new source routing paradigm designed to simplify existing routing techniques in traffic-engineered networks. It enables efficient packet steering through a specific network path, rather than natural shortest path that packet usually takes within a network. A source node directs incoming packet flow through the network by specifying a list of intermediate destinations that a packet must visit on its way to the final destination. In Segment Routing, labels called segments represent intermediate path points. The main benefit of this technology is its simplicity, easy implementation and scalability [1] [22].

Segment Routing is technique that is enabled by small number of extensions to routing protocols such as IGP, BGP and PCEP and it can be applied in MPLS and IPv6 architecture. Segment Routing does not require a lot of changes in MPLS forwarding plane. A segment is encoded as a MPLS label and list of segments are equivalent to MPLS label stack. In IPv6 architecture a segment is represented as an IPv6 address. This is enabled by introducing Segment Routing Extension Header that allows multiple IPv6 addresses to be encoded in source router, so multiple intermediate hops can be specified.

The Segment Routing concepts are same in both environments. However, some implementation details and protocol extensions differ between Segment Routing in MPLS and Segment Routing in IPv6 networks. This work puts

focus on Segment Routing over MPLS architecture and details on Segment Routing in IPv6 networks will be skipped.

3.2.1 Segment Routing Concepts

This subchapter discusses main Segment Routing concepts. Firstly, a concept of segment will be explained. After, the classification of segments will be represented with related examples.

Segment

According to the IETF, a segment is an instruction that node executes on the incoming packet. This instruction could be for instance, forward the packet to a specific network node according to shortest path, or forward packet through specific interface, or deliver the packet to a given application or service. Segment is identified with Segment Identifier (SID) and in MPLS environment it is encoded in 32 bits MPLS label [23].

Segment Advertising

Segments are advertised using IGP and BGP routing protocols. For both protocol types, Segment Routing extensions are defined to include Segment Routing information. In other words, routing protocols enable segments' signaling through the network. Let us now consider an autonomous system consisting of multiple IGP areas. Within each IGP area either IS-IS or OSPF is running. They are responsible to advertise segments within an IGP domain. However, in order to implement traffic engineering between an AS, segment exchanging between BGP peers must be enabled. BGP is extended to advertise the segments related to the BGP-prefix. The more details about protocol extensions will be provided subchapters 3.3 and 3.4.

Segment routing is constructed with SDN in mind. In software defined network it is assumed that SDN controller is in charge of determining end-to-end paths throughout a network. SDN has information on underlying network topology provided by BGP-LS protocol. In a software defined network that implements Segment Routing, BGP-LS is responsible to advertise SDN controller about

segment identifiers. The topological path calculated by a SDN controller is pushed down to the source node in a form of the list of segments. Calculated path is carried by PCEP protocol. In SDN environment both PCEP and BGP-LS extensions are necessary to support Segment Routing. More information about these extensions are presented in chapters 3.4 and 3.5.

Global and Local Segments

According to its significance in the network all the segments can be divided on global and local. For now, the term network will be related to an IGP area.

Global segment is related to the instruction that is supported by all nodes in an IGP domain. A global segment must be unique within a domain. Any node in an IGP domain must have all global segments in its Forwarding Information Base (FIB). The value of global segment identifiers is taken from the Segment Routing Global Block (SRGB). SRGB is a subspace of a 32bit SID space, and it takes values from 16000 up to 23999 [24].

Local segment is an instruction that is supported by the node originating it. Local segments take a value outside of SRGB range. Since it has only local significance, its value is related only to local router FIB. A router is not aware of local segments of the other routers in a domain. Moreover the local SID values could be reused within an IGP domain, since a local SID value has local meaning for each single router.

IGP Segment Identifiers – IGP-SIDs

Link state protocols have an important role in Segment Routing. Global and local segments are distributed throughout the domain using IGP [23]. Both Open Shortest Path First (OSPF) and Intermediate System to Intermediate System (IS-IS) support Segment Routing thanks to well-defined protocol extensions, which will be explained later in the paper. Segment Identifiers distributed by an IGP can be classified as it is shown in the following figure:

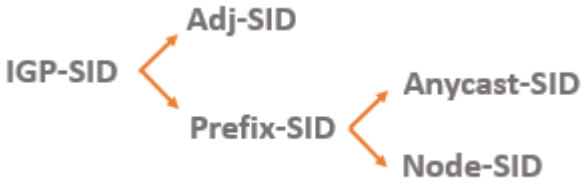


Figure 3.1: IGP-SID classification

Prefix-SID

In general, Prefix-SID is a segment that refers to a specific network prefix. Prefix-SID is always global within an IGP domain and it refers to the shortest path computed by IGP to the related prefix. A packet that enters an IGP area with an active Prefix-SID will be forwarded along the ECMP-aware shortest path to the prefix. Since a prefix could represent a node or a group of nodes within an IGP domain, Prefix-SIDs are further divided into Node-SIDs and Anycast-SIDs:

- Node-SID is a Prefix-SID and it refers to a specific node. A Node-SID has a global significance and it identifies exactly the prefix of the node's loopback interface. For instance, let's observe the figure 3.2. 16004 is a Node-SID of the R4. The R1 wants to send a packet to R4 and it pushes the Node-SID on top of the packet's header. Since the shortest path to the R4 is through R2 and R3, R4 forwards packet to R2. When the packet arrives to R2, the router checks its FIB and passes the packet towards R3. Since R3 is not the final destination of the label 16004, R2 does not remove the label. Since R3 is the last router towards destination it passes the packet to R4 and remove the label out of stack

- Anycast-SID identifies a set of routers. A packet with Anycast-SID will be forwarded towards the closest node of anycast set. The Anycast-SID is an interesting tool for traffic engineering because it makes it easy to express macro traffic-engineering policies

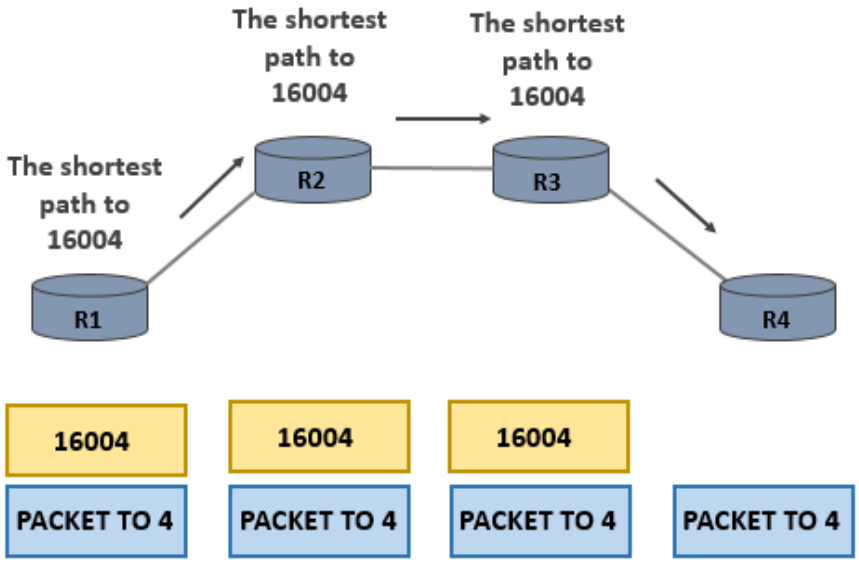


Figure 3.2: Node-SID example

Adjacency-SID

The Adj-SID is IGP-SID that points on a specific link that belongs to the same IGP domain. Adj-SID has local significance, which means that a router maintains Adj-SIDs only for its neighbors. Adjacency segments must take a value that is outside of SRGB range. Usually a router allocates them dynamically. Since Adjacency SID's has local significance they don't have to be unique in the SR domain. Adj-SID is very useful if u want to steer traffic flow through a specific interface. The figure 3.3 illustrates does how it work. Let's observe the R4. Adj-SIDs are assigned automatically for its three interfaces 24001, 24002 and 24003 (note that values are out of SRGB). If one wants to use a link between R4 and R5 it is enough just to push the local label (24002) and packet will be forwarded to the next hop.

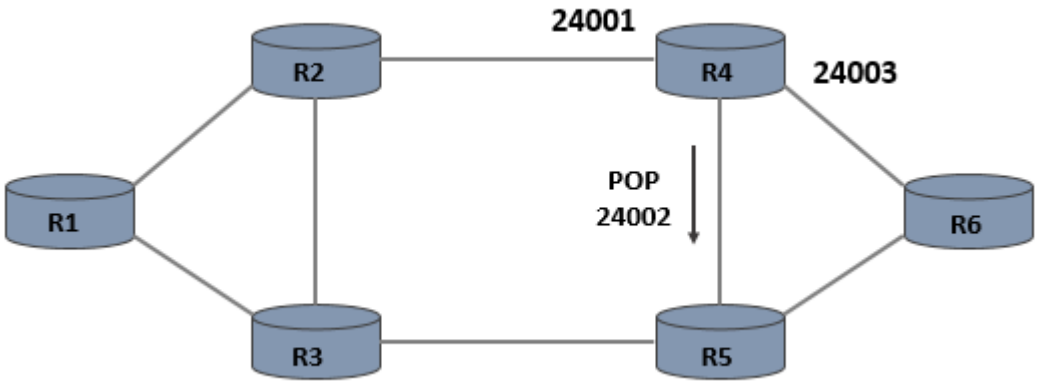


Figure 3.3: Adjacency-SID example

Routing Operations

Source node steers the incoming traffic flow by attaching an ordered list of SIDs to a packet header. The top segment is the first one that will be executed. Once the segment is executed (packet reaches an intermediate destination), next segment is going to be processed and so on. When last segment is executed, a flow either reaches its destination, or it just exits a SR domain and continues to be routed according to destination IP address.

There are three actions that could be performed on segments by SR-capable nodes [22]. However, they are closely related to operations performed on MPLS labels in MPLS networks. Segment Routing operations are:

1. PUSH (MPLS PUSH) – a segment is pushed on the top of segment stack
2. NEXT (MPLS POP) – an active segment is completed and it is removed from the stack
3. CONTINUE (MPLS SWAP) – active segment is not completed yet and it remains active. Naturally, this operation exists only for global segments, since their execution could include multi-hops. Local segments (adjacencies) are executed in a single hop

The figure 3.4 explains how a packet is forwarded through a SR domain.

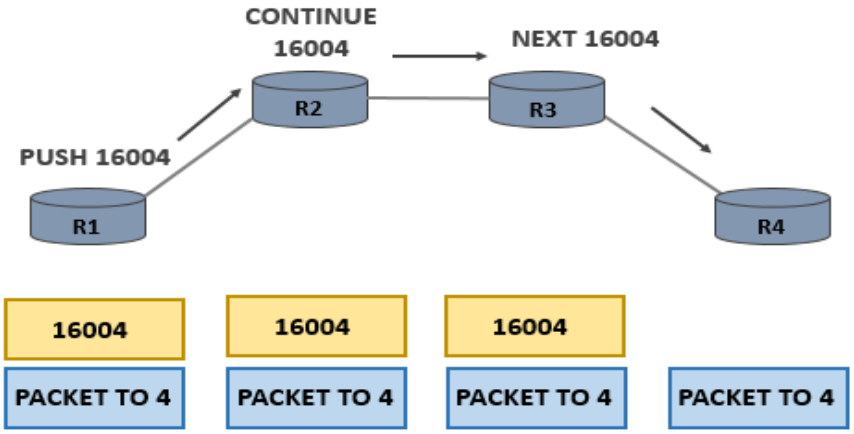


Figure 3.4: Segment Routing Operations

3.2.2 Segment Routing Use Cases

Simplified transport of MPLS services

Segment Routing can offer the same tunneling service as MPLS in simplified manner using just IS-IS or OSPF, figure 3.5. Service provider can easily enable services like L3VPN, VPLS and VPWS by setting up a Node-SID per network edge and ECMP tunnels will be created automatically from any ingress to any egress edge [1]. LDP and RSVP are no more required, and that leads to following benefits:

- Simpler operation – less signaling in the network meaning the gain in terms of bandwidth and operation complexity
- Scaling – only one label for each node, reducing number of LSDB entries

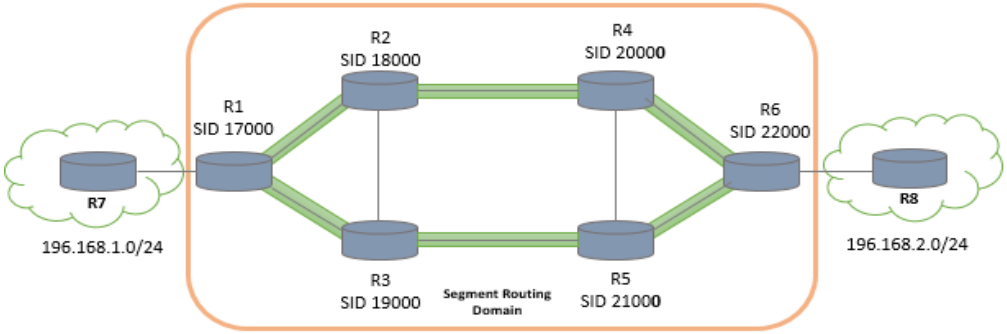


Figure 3.5: Simplified MPLS transport

Segment Routing and LDP coexistence

Inside MPLS Architecture, Segment Routing can coexist with LDP and RSVP-TE [1] [25]. Segment Routing Global Block assures that labels used for Segment Routing and LDP are allocated from different blocks of label. If both Segment Routing and LDP are enabled on the same router, LDP is given priority by default, but this can be changed using CLI configuration.

Traffic Engineering

Segment Routing can create tunnels according to customers' needs. SR enables traffic steering through any desired network path. By employing different SIDs, tunnels can be constructed in a smart way, which will result in increased network performance and throughput. Also, tunnels can be designed by taking into account customers' SLAs. The most significant TE use cases are presented below.

Deterministic path or path avoidance is for sure the most useful tool in traffic engineering [26]. By exploiting adjacency SIDs, one can specify a path as path which flow will take through the network. Typical use case is presented on the figure 3.6.

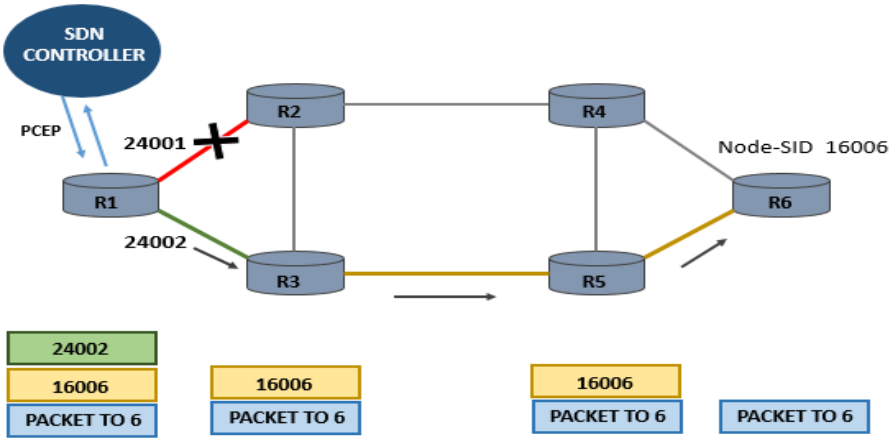


Figure 3.6: Traffic Engineering - Path Avoidance

One wants to send the traffic to R6 (Node-SID 16006). The easiest way to it is to push node segment on top of the packet and it will be forwarded according to the shortest path. However Node-SID represents the instruction for ECMP-aware shortest path to R6, meaning that flow will take either R1-R2-R4-R6 or R1-R3-R5-R6. In the case the link R1-R2 becomes overloaded and the QoS drops down, a controller can dynamically push the traffic to R3 and avoid a busy link. Traffic will arrive at R3 and then it will continue to R6 according to the shortest path.

By assigning anycast SIDs, one can define a group of routers which flow will take on its way to the destination. For service providers this is very interesting tool because it can express macro policies such as “go via plane one of dual-plane network” or “go via European Region” [27]. As an example let’s observe the figure 3.7. The network can be described as a dual plane. One can steer the traffic only through yellow or blue nodes to the final destination by assigning labels {16001, 16005} or {16002, 16005}. ECMP is supported within a plain, meaning if there are disjoint paths, the load will be balanced (per flow).

The main benefits are tunneling without RSVP and LDP signaling, ECMP-aware routing and zero per-flow state on transient routers. Only additional anycast SID have to be configured (one per network plane).

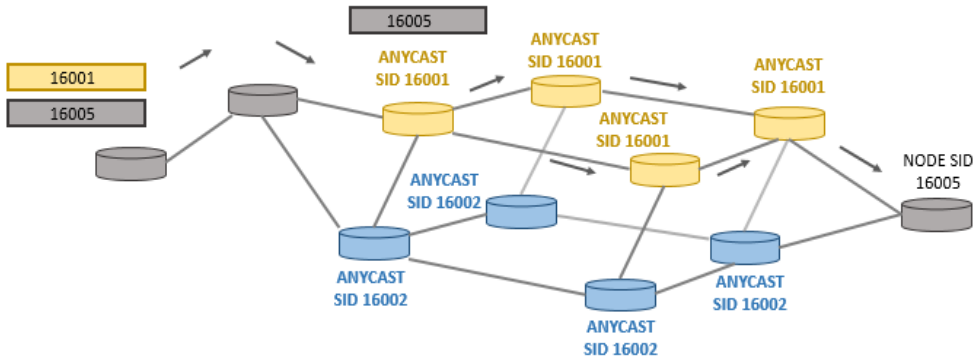


Figure 3.7: Using Anycast-SID in Dual-Plane networks

Fast Reroute

Segment Routing supports services with tight SLA. To do so, Topology Independent Loop-Free Alternate (TI-LFA) is used within Segment Routing network. TI-LFA guarantees 100-percent coverage in any IGP network and 50 msec of convergence time.

TI-LFA is very easy to implement because for the protection path it uses one that is automatically pre-computed by IGP. As a protection path, it uses post-convergence path, which is the optimum path in case of primary path failure [28]. Post-convergence path is typically planned by network architects to support traffic rerouting in the case of failure. If failure happens in SR network, the only node that keeps state is one that suffered from failure – it reroutes packets by attaching backup segments.

3.3 Intermediate System to Intermediate System

Intermediate system to Intermediate System (IS-IS) is an Interior Gateway Protocol, standardized by International Organization for Standardization in 1992, for communication between two network devices named Intermediate System (IS) [29]. Together with Open Shortest Path First (OSPF) belongs to the group of link-state protocols. Link state protocols enable very fast convergence as well as good scalability performance comparing to distance-vector protocols. The main features of IS-IS are:

- Classless behavior
- Rapid flooding
- Fast convergence
- Hierarchical routing
- Scalability
- Flexible timer tuning

IS-IS operation is based on flooding link state information inside within an administrative domain (routing domain). A router periodically floods a network with link state packets (LSP) that contain information about status of its links. According to collected information any network node can get a full picture of network topology. By applying certain algorithms a router can calculate the shortest path to other nodes in the domain. IS-IS uses Dijkstra algorithm. Upon calculation a router stores results in its FIB - new entry is added for each route containing destination node, next-hop address and output interface.

IS-IS routing uses two-level hierarchical routing [30]. A IS-IS router can be either Level 1, Level 2 or Level 12 router. Level 1 router is a non-backbone router that builds adjacency with neighbors in the same, Level 1 area. The only information that a level 1 router has regarding the other areas is a default route to the closest Level 1-2 router. Level 2 router is a backbone router and it builds adjacency with only Level 2 routers, which can be in the same area or in other areas. Level 2 routers are responsible for inter-domain routing and they have all information about topology. Level 1-2 router builds adjacency with both Level 1 and Level 2 routers. They behave as gateways to Level 1 routers that want to send the traffic outside of domain. Hierarchical IS-IS architecture is presented on the figure 3.8.

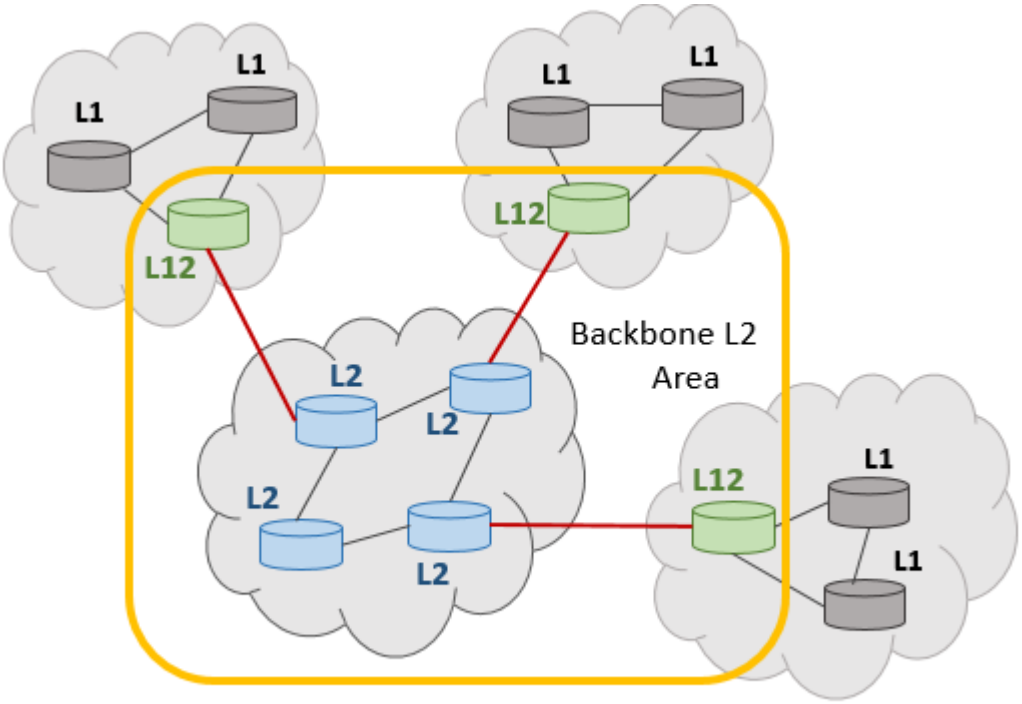


Figure 3.8: IS-IS hierarchical architecture

In general, an operation flow of an IS-IS router:

1. Sends hello messages and builds adjacencies
2. Creates link state packets and floods its neighbors
3. Receives all link state information from its neighbors
4. Runs shortest path first algorithm (Dijkstra) and partial route calculation
5. The procedure is repeated when new LSPs are received

IS-IS together with OSPF is essential for Segment Routing networks. IGP enables segments redistribution through the network, which eliminates needs for signaling protocols such as LDP. Comparing to OSPF, IS-IS became increasing popular among service providers because its performance in large topologies. That motivates us to implement it in our network as an IGP. In the following subchapter we will see what extensions to ordinary IS-IS are essential for Segment Routing.

3.3.1 IS-IS extensions for Segment Routing

IS-IS uses Type-Length-Value elements (TLV) inside protocol message [31]. TLVs make a protocol easy extensible. Once additional data should be encoded in message a new TLV is added and there is no need for redesigning a protocol. In order to support Segment Routing new IS-IS Sub-TLVs are defined. Sub-TLVs are placed inside of TLVs and they use the same concepts as ordinary TLVs.

Prefix-SID sub-TLV

Prefix-SID sub-TLV is defined to carry information on Prefix Segment Identifier. As mentioned before, that Prefix-SID must be unique within IGP domain. It has following format:

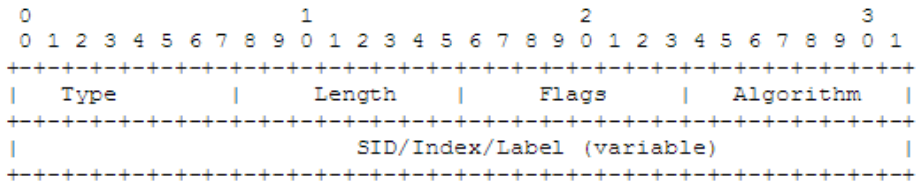


Figure 3.9: Prefix-SID sub-TLV format for IS-IS message

The first four fields are 8 bits length each. Six flags are defined for Prefix-SID sub-TLV: Re-advertisement flag, Node-SID flag, no-PHP flag, Explicit-Null flag, Value flag and Local flag. Depending on their value, these flags could influence where and how sub-TLV will be distributed within IGP domain. Algorithm field contains an identifier of the exact algorithm that a router should use to compute the shortest path (metric based SPF, constrained SPF and etc.). SID-Index-Label field carries information on SID. There are two ways how SID can be encoded into this field. SID value can be encoded directly, and in that case three octets out of four are used. Value and Local flags must be set, meaning that sub-TLV carries actual SID value. Otherwise, SID information could be carried under its index value. Index value is an offset in SID-Label range for a given router. Once it receives an index a router uses it to retrieve an actual SID value. This encoding uses all 32 bits, and it requires Local and Value flags to be unset.

Adjacency-SID sub-TLV

Adjacency-SID sub-TLV encodes and distributes Adjacency-SID. Adjacency sub-TLV can advertise adjacency of a single node or set of adjacencies referred to designated router. It is part of IS-Neighbor TLV and it has similar format as Prefix-SID sub-TLV:

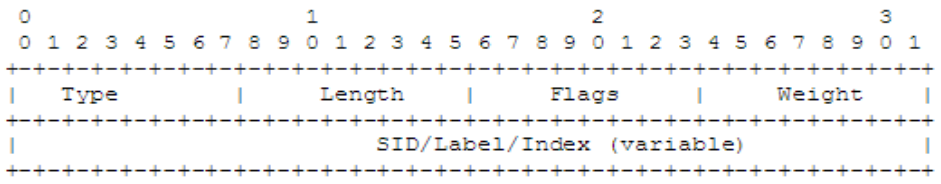


Figure 3.10: Adjacency-SID sub-TLV format for IS-IS message

Adjacency-SID sub-TLV has five flags defined: Address-family, Back-up, Value, Set and Local flag. Unlike Prefix-SID sub-TLV, here we have weight field that carries the weight of the link, for the purpose of load balancing.

SID-Index-Label field can carry one of those three values: Label value (3 octets), Index value (4 octets) or IPv6 address (in the case SR is implemented onto IPv6 data plane). Local and Value bits combination indicates which of those three values is inside of SID-Index-Label field.

In LAN sub-networks, a designated router is elected, and together with its adjacent nodes forms a pseudo-node that can advertise set of adjacencies to its 'new' neighbors. This kind of advertisement is supported by LAN-Adj-SID that has similar properties as an ordinary Adj-SID sub-TLV.

SID-Label Binding TLV

SID-Label Binding TLV could have multiple roles in SR architecture and could be originated by any router in a domain. It can be used for binding IP address to SID value. This could be important if in a domain not all of nodes are capable to advertise its SIDs. Another purpose of Binding TLV is to bind a primary path with a backup path. Lastly, the router may advertise a SID/Label binding to a FEC along with at least a single 'next hop style' anchor.

Capabilities Sub-TLV

Router features in SR architecture are advertised through SR-Capabilities Sub-TLV and SR-Algorithm sub-TLV. Each router in a domain must generate those sub-TLVs. The first sub-TLV advertises SR capability and label ranges that a certain router uses. The second sub-TLV allows the router to advertise the algorithm.

3.4 Border Gateway Protocol

Border Gateway Protocol (BGP) is the one of the most important Internet protocols today. It a standardized protocol used to exchange reachability and routing information between Autonomous Systems (AS) [32]. AS is referred to a network or a group of networks under common administration. Although BGP is referred to an exterior gateway protocol (eBGP), in complex networks it also might be used as an interior gateway protocol (iBGP). Routers on the boundary of one AS exchanging information with another AS are called border or edge routers or simply eBGP peers and are typically connected directly, while iBGP peers can be interconnected through other intermediate routers. The current version of BGP is BGP4 [33].

The router that implements BGP is called BGP speaker. BGP speakers peer among themselves exchanging network layer reachability information (NLRI) within UPDATE messages. NLRI contains the network number, path specific attributes and the list of autonomous system numbers that route must transit to reach a destination network. BGP prevents routing loops by rejecting any routing update that contains the local autonomous system number because this indicates that the route has already traveled through that autonomous system and therefore a loop would be created.

Any BGP speaker receives routing updates from other peers, processes the information for local use and then advertises selected routes to different peers based on predefined policies. In order for BGP to be able to perform its functions it stores this information in a special type of database called the

BGP Routing Information Base. BGP Routing Information Base consists of three parts:

a. Adjacency Routing Information Base Incoming (Adj-RIBs-In) is filled up with information received from incoming UPDATE messages

b. Local Routing Information Base (Loc-RIB) contains the local routing information the BGP speaker selected by applying its local policies to the Adj-RIBs-In

c. Adjacency Routing Information Base Outgoing (Adj-RIBs-Out) stores information the local BGP speaker selected for advertise to its neighbors

Policies about filtering routes from Loc-RIB-IN into Loc-RIB are configured manually by network operator. There are diverse types of criteria that can be applied in path decision such as: local preference (usually a path preferred by operator), shortest AS path, lowest origin type, smallest multiple discriminator. Also network operator can decide which NLRI will share with other networks. Moreover an operator can decide which routes will be advertised to other BGP domain.

BGP is a very robust and highly scalable routing protocol. BGP routing tables can carry thousands routes. BGP uses many route parameters, called attributes, to define routing policies and maintain a stable routing environment. BGP attributes can be mandatory or optional. Mandatory attributes are attributes that have to present for each entry. Some of them are ORIGIN, AS_PATH, NEXT_HOP, and LOCAL_PREF. On other hand, optional attributes do not have necessarily be present in RIB. BGP applies Classless Inter Domain Routing (CIDR) a mechanism to reduce the size of the Internet routing tables. When CIDR is applied block of network addresses is advertised and the concept of network class is eliminated (route aggregation).

3.4.1 BGP in Segment Routing - concepts and extensions

In the case of multi-domain traffic engineering, IGP segments are not enough to perform the task. IGP segments are configured for a single IGP domain and a router cannot pass the traffic to the next domain unless it doesn't know its segment identifier. Since there is no IGP running between routers in different domains, the only way to solve it is by using BGP protocol and BGP segments.

For the purpose of Segment Routing, BGP-Prefix-SID is defined to identify certain BGP prefix [34]. BGP-Prefix-SID is always global within BGP/SR area. A router that receives a packet with BGP-Prefix-SID will route a packet according to the shortest, ECMP-aware, path to specified BGP prefix.

Each BGP speaker can assign a label index to the prefixes it originates. Index is an offset value in label range. For BGP-Prefix-SIDs the values must be taken from SRGB. One must specify the range of global block that will be used in the network. This range must be between 16000 and 23999. When BGP speaker determines an index, the corresponding SID can be determined as SID = SR_start + index.

Regarding BGP-Prefix-SID, IETF has defined in a draft:

- BGP Prefix-SID attributes that uniquely represent BGP-Prefix-SID - explained below
- Rules for generating and processing the label index - label index is configured and advertised in BGP update message. With whom BGP will exchange those information is defined by policy. If a speaker receives unwanted message, it discards it
- Error processing for the label index attribute - if a speaker receives bad-formed attribute it ignores it. If it receives multiple attributes or if it receives unacceptable attribute it does not pass it further and it may log the error for further analysis

BGP Prefix-SID is encoded in TLV format. For MPLS data plane two TLVs represent the Prefix-SID: Label-Index TLV and Originator SRGB TLV. The first TLV is mandatory and carries label index value and it format is following:

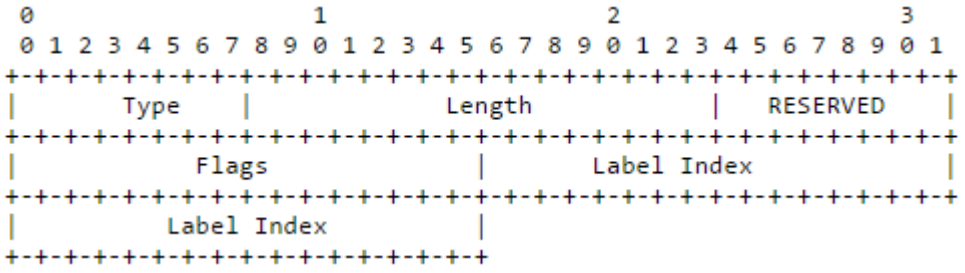


Figure 3.11: Label-Index TLV in BGP message

The type field is 1 and the length is 7. Eight bits are reserved and there are 16 bits for flags. Label index is encoded in 32 bits.

The second TLV is optional and describes SRGB of the router that originates BGP-Prefix-SID. SRGB is encoded in three octet field. SRGB field can appear multiple times meaning that SRGB consists of multiple ranges. Type value is 3, flag field is 2 octets and length field is variable and depends on number of SRGB ranges.

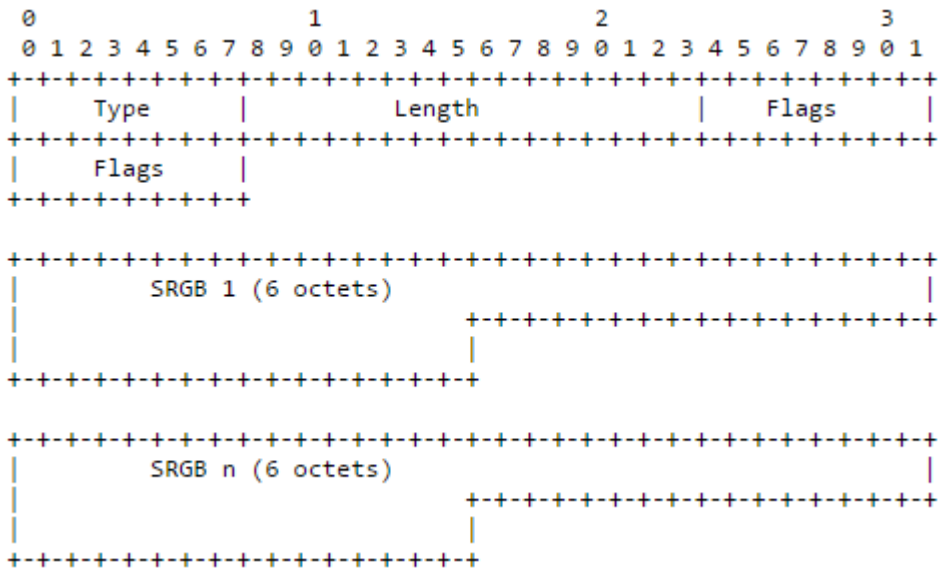


Figure 3.12: Originator-SRGB TLV in BGP message

3.4.2 BGP-Link State

To calculate an end-to-end path that spans over multiple IGP domains, one should exploit information from several LSDBs. For multi-domain path computation, central path computation entity is usually employed (PCE or SDN controller). However, in order to have full topology picture across domains, path computation entity must be able to retrieve link states information of domains it serves [35].

One way to solve this problem is to make a controller to be a passive IGP peer [36]. A controller receives IGP messages and accordingly constructs a virtual topology that can be used for path computation. In this case, a controller must be configured to support both OSPF and IS-IS. However, IGP could be very chatty protocol that can cause frequent updates on the controller side. Also, if the network consists of distant IGP domains it could be a challenging task to place a SDN controller in such a way that can peer all IGP domains

Another mechanism by which Link State information can be collected and shared with external entities is by using Border Gateway Protocol – Link State. BGP-LS is a protocol from BGP family designed to carry link state information collected by IGP and share them with an external entity. This is enabled by introducing new BGP Network Layer Reachability Information (NLRI) encoding format. BGP speaker can retrieve data from IGP LSDB and transfer it to a controller or to a centralized TED. BGP speaker can send data directly to a consumer, but typically is done via dedicated BGP node called Route Reflector, as shown on the figure 3.13.

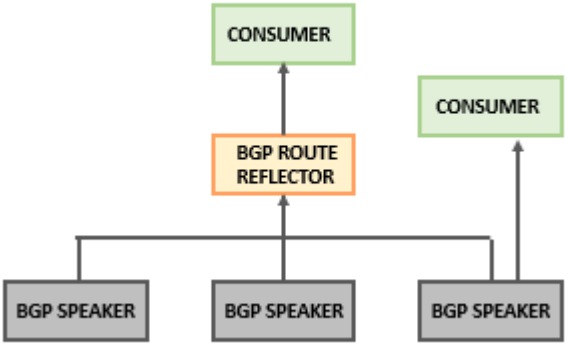


Figure 3.13: BGP-LS distribution - Direct and through the Route Reflector

Additionally, a BGP speaker can filter information that will be distributed by applying configurable policies. It can distribute a real physical topology retrieved from IGP LSDB, but also it may construct an abstracted topology that consists of virtual, aggregated nodes and virtual paths. Distribution of mix of those two is also possible. Furthermore, there are policies about how frequent BGP speaker advertises its consumers, so information flow and data processing can be reduced.

For BGP-LS a new BGP NLRI and new BGP path attributes are defined. NLRI describes three objects: nodes, links and IP prefixes. Having node and link object encoded in BGP-LS message, one can make a picture of a network topology. IP prefix object is used to provide IP reachability information. New BGP attributes (BGP-LS attributes) encode properties of the BGP objects such as node-name, IGP metric, TE metric, available bandwidth. Both link-state NLRI and attributes are encoded in Type-Length-Value triplets and its format is shown on a figure below.

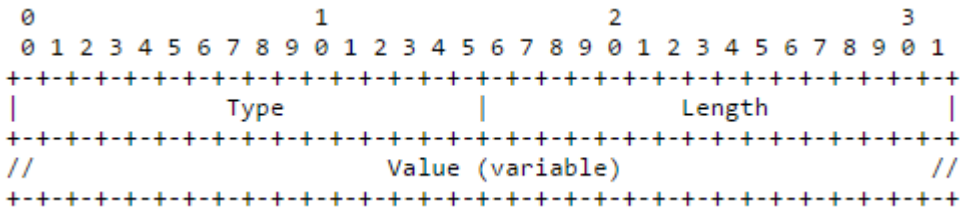


Figure 3.14: TLV format

- Type 1, Node NLRI (figure 3.15), contains information on a network node. It carries Router-ID that is used in IGP (48 bits for IS-IS, 32 bits for OSPF)
- Type 2, Link NLRI (figure 3.16), describes a link in a network. It carries information on the endpoints (local and remote node) and link descriptors that uniquely describes a unidirectional connection between two nodes
- Type 3 and 4, IP prefix NLRI (figure 3.17) carries prefix descriptor that identifies IPv4 and IPv6 prefixes originated by a node

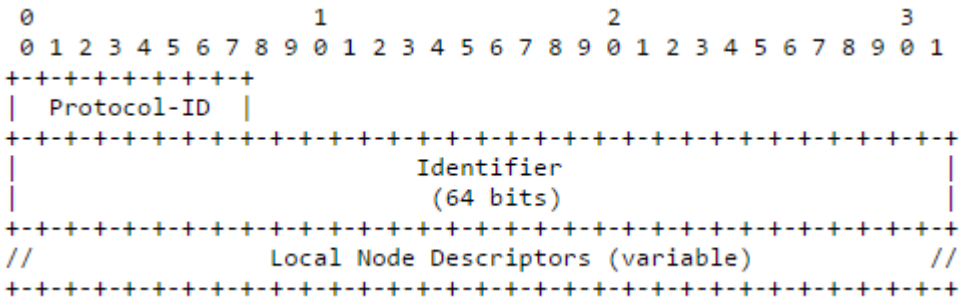


Figure 3.15: Node NLRI format

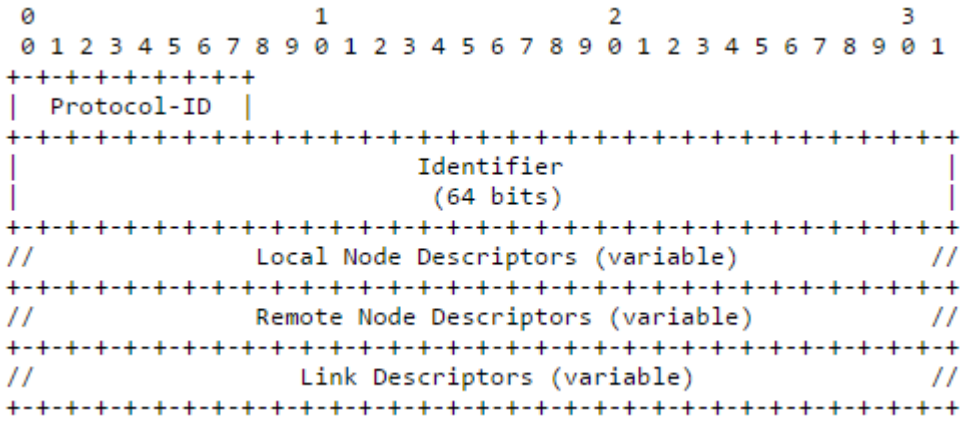


Figure 3.16: The Link NLRI format

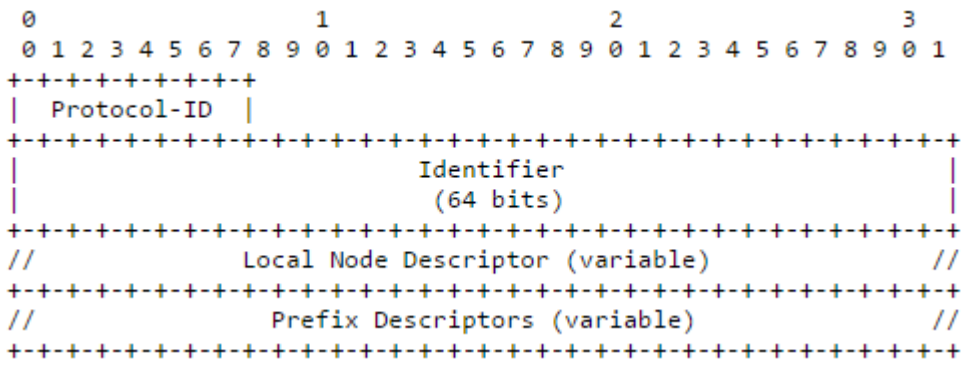


Figure 3.17: The topology IP prefix NLRI format

3.4.3 BGP-LS extensions for Segment Routing

BGP-LS is one of the protocols that enables Segment Routing in the network. It provides efficient sharing of segment information required for building end-to-end paths. IETF has defined new TLVs to be added to BGP-LS path attributes, so they can carry necessary information related to Segment Routing and share them with external entities [37].

As mentioned above, BGP-LS attribute could refer to a node, link or IP prefix. Information on SR are added to corresponding attribute:

- Node Attribute TLVs - three new TLVs are defined: SID/Label binding, SR capability and SR algorithm TLV. Biding labels could be used for different purposes, such as to bind primary and recovery path. SR capability must be advertised by a node to signal its capability to support SR. And algorithm TLVs carry the algorithm that a node used to calculate shortest path to other nodes
- Link Attribute TLVs – two TLVs are defined: Adj-SID and LAN Adj-SID TLVs. Adjacency TLV is added to the BGP-LS attribute of local node (Adj SID is related to the local endpoint). LAN Adj-SID TLV is used to advertise adjacency to other nodes attached to LAN
- Prefix Attribute TLVs – one TLV is defined: Prefix SID. TLV is referred to the local node

3.5 Path Computation Element and Path Computation Element Protocol

Constraint-based path computation is an essential function in traffic-engineered networks. In the era of high-demanding applications in terms of quality of service, determining an optimal end-to-end path is crucial task to do. Today, core transport networks evolve toward multi-domain networks and in such environment path computation becomes even more complex, and could require cooperation between elements in different domains. Distributed path computation can lead to non-optimal solutions, since there is no single entity that has the global visibility of network topology and available resources. Bad decisions in traffic engineering could cause traffic congestion and overloaded links as well as an extra delay for a customer.

Decoupling path computation function from the rest of control plane is beneficial for both efficient network utilization and providing better service to the customers [38]. Centralized path computation function that has view over multiple domains (or layers) can offer sophisticated path computation mechanisms according to established SLAs. Centralized path computation can provide end-to-end resource provisioning for customers in terms of bandwidth, latency and more. Moreover, in the context of Wavelength/Spectrum Switched Optical Networks it could be taken into account wavelength availability/continuity and optical impairments. For network operators centralized computation function means more flexibility in network management and control.

In 2006, IETF has defined Path Computation Element (PCE), a centralized path computation entity that is capable of computing a network path based on network topology, and by applying constraints and policies [39]. PCE retrieves a network topology from Traffic Engineering Database (TED), and the area that one PCE serves depends on the scope of TED it communicates with. Usually, PCE serves more than one IGP domain that also can belong to different ISPs.

3.5.1 PCE Architecture

A simple PCE architecture consist of: PCE and Path Computation Client (PCC). PCC is any network client that requests path computation for incoming traffic flows. Communication between PCC and PCE is enabled by Path Computation Element Communication Protocol (PCEP). PCEP defines messages and objects necessary to convey a PCEP session. The protocol is TCP based which is guarantee for a reliable session. PCEP uses TCP port number 4196.

PCE architecture supports diverse computational models such as multiple-coordinated PCE solutions and hierarchical PCE model. For example, a typical core network consists of IP/MPLS network over Wavelength-Switched Optical Network (WSON). In combined network infrastructure, end-to-end connection is set up by mapping IP/MPLS LSP over physical lambda-switched network. Here, joint path computation is essential and it requires coordinated path computation on logical and physical layer. It can be done in integrated manner, where a single PCE has access to TED that keeps information on both layers. But commonly, multi-layer infrastructure deploys multiple-coordinated PCEs. Communication between two PCEs is also enabled by PCEP.

PCEP session

Communication between PCC and PCE is client-server based. The following messages are defines for PCEP session:

1. Open message – initiate session
2. Keep alive – maintain session
3. Path computation request
4. Path computation reply
5. Notification – to notify a specific event
6. Error message
7. Close – close PCEP session

The initialization phase consists of two successive steps. Firstly a TCP connection (3-way handshake) is established between the PCC and the PCE. After TCP connection is setup, nodes initiate PCEP session. In this step various parameters are negotiated including keep alive and dead timer. Afterwards, a PCC sends path computation request, PCReq, that contains a variety of objects that specify the set of constraints and attributes for the path to be computed. Upon receiving a path computation request from a PCC, the PCE triggers a path computation. If PCE succeed to calculate a route that satisfies required constraints, it sends a positive reply with encoded path. Otherwise it sends a negative reply, so PCC may decide to resend a modified request or take any other appropriate action. The PCEP session flow is presented on the figure 3.18.

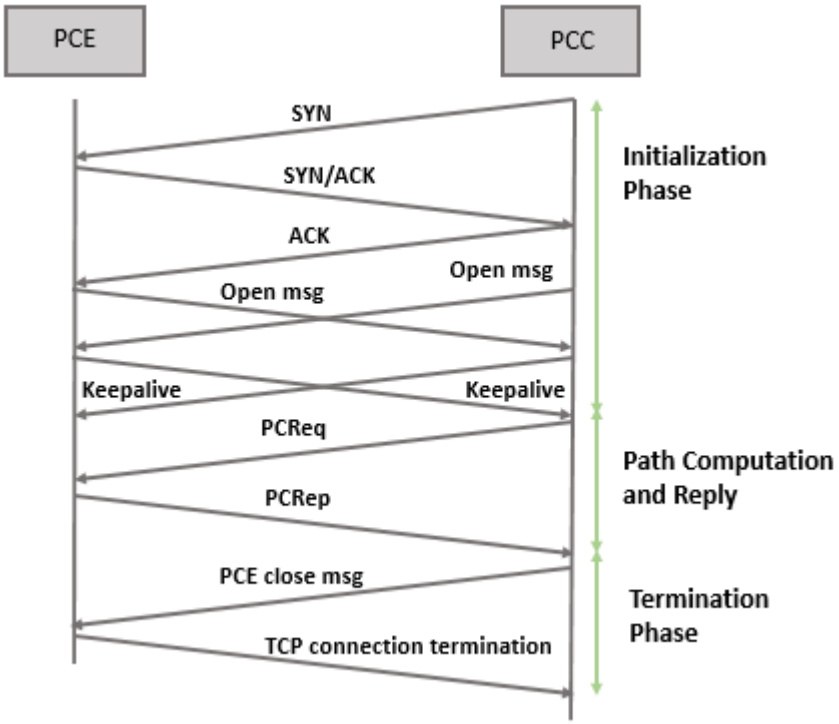


Figure 3.18: PCEP session flow

Stateful and Stateless PCE

The PCE synchronization with TED is essential for correct path computation. In PCE synchronization with database can be realized in different ways. PCE

might be passive IGP listener of single or multiple domains. PCE can act also on wider set of information. Sometimes in calculation might be included information about network state, for instance LSP database (LSPDB). Depending on synchronization habits and which information is used in path calculation we can distinguish two different PCE types: stateful and stateless.

A stateful PCE requires strict synchronization with network state (TED) and LSP state (LSPDB) [40]. This means that PCE takes into account not only topology and recourse information, but also previously computed paths and reserved resources in the network. A stateful PCE provides both efficient path computation and increased path computation success. A stateful PCE can be in active or passive mode:

- A passive stateful PCE synchronizes with TED and LSPDB in order to calculate new LSPs, but it does not have permission to modify existing LSPs in order to optimize overall network traffic flow. A passive PCE calculates a new route only when it is requested from PCC
- An active stateful PCE can change a network state while computing a new LSP. A network controller (GMPLS controller / SDN controller) temporarily delegates control on active LSPs to PCE and it can manage existing routes (e.g. reroute or change LSP attribute) while calculating a new route. This lead to overall network re-optimization whenever there is demand for a new traffic flow. Moreover, active stateful PCE can request from controller to modify existing LSPs, if necessary

A stateless PCE calculates a new path without having knowledge about previously calculated paths. A stateless PCE computes a path based on current TED, which might be out of synchronization with network state given the recent PCE paths changes. However, in this case it is possible that a PCC includes in PCReq the previously computed paths in order to take them into account.

3.5.2 PCEP extensions for Segment Routing

Segment Routing supports both distributed and centralized traffic engineering. Coupled with PCE, SR provides multiple benefits to network operator. Frist of all, path computation is optimized because a source node does not have end-to-end visibility in multi-area and multi-layer networks. Moreover, network resources can be effectively used by monitoring and optimizing existing LSPs. LSPs can be set up according to customers' SLAs.

In a PCEP session, information about LSP is carried in Explicit Route Object (ERO). ERO consists of ordered sequence of sub-objects that carry information on SID. Once an ingress node receives ERO from PCE it attaches SIDs to incoming packets. SR-TE LSP calculated by a PCE can be represented in one of following forms:

1. Ordered set of IP addresses which a flow must cross. In that case a PCC must translate them into SID by consulting a TED
2. Ordered set of SIDs
3. Ordered set of both IP addresses and SIDs. In that case also IP addresses must be translated into SIDs

An ordinary PCEP message consists of common header and variable length body that is made up of mandatory and optional objects. SR doesn't require any changes on PCEP message format. The following extensions are added to PCEP to support SR architecture: new ERO sub-object, RRO sub-object, PCEP capability extension and new PCEP error codes [41]. IETF also specified how two PCEP-SR speakers establish PCEP session, processing rules of ERO sub-object. However messages must include path-setup-type TLV to clearly identify that SR path is carried in massage.

SR-PCE capability TLV

The SR-PCE capability TLV is an optional TLV that enables exchanging SR capability of end points in PCEP session. If PCC includes this TLV in OPEN

message that means it supports SR-TE LSP. If PCE includes the TLV that means that it is capable of computing SR paths. During the session opening the Maximum SID Depth can be stated by PCC. If it is set to 0 by default and it means that the node supports any MSD. MSD is characteristic of data plane.

The new ERO and RRO sub-object

The ERO object within PCEP message is designed to carry SR-TE path information. A single ERO object can consists of one or more ERO sub-objects. In SR each sub-object carries one SID. Once it receives an ERO object, an ingress router builds a SID stack from sub-objects. The first sub-object will be the topmost label and the last one is on the bottom. SR-ERO sub-object has the following format:

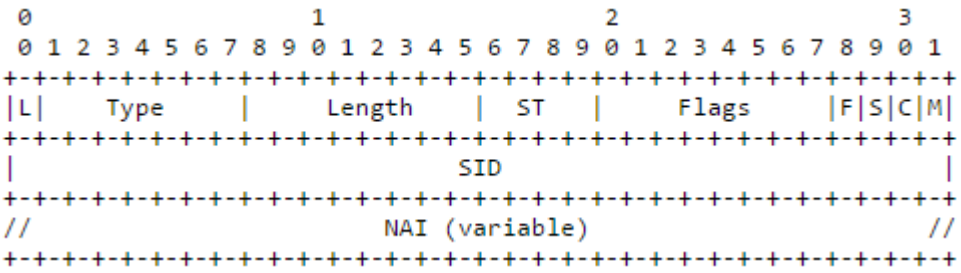


Figure 3.19: SR-ERO Sub-object format

SR-ERO contains common fields like Type, Length and Flags. Flags F, S, C, M are used to indicate the type of SID and NAI carried in sub-object. SID Type (ST) field indicates the type of information associated with SID. SID field contains SID and NAI contains IPv4 or IPv6 address related to the SID. Record Route Object for SR (SR-RRO) has the same format as SR-ERO without L, C, M flags.

3.6 Multi-Protocol Label Switching

MPLS is a technology for data tunneling service build for high-performance telecommunication networks. MPLS operates on the so called 2.5 Layer and it

is compatible with any network protocol of which IP is the most popular. MPLS has brought performance enhancements and new service creation capabilities in connectionless IP world. MPLS has introduced Virtual Private Network (VPN) services and QoS across the network [42].

In MPLS networks, packets are routed from one network node to the next based on 32-bit MPLS labels. In that way, a packet does not experience a delay caused by complex IP lookups in routing table, which can be especially crucial for high priority traffic such as voice, video and similar. MPLS tunnels are setup based on Forwarding Equivalence Criteria (FEC) [43]. When a tunnel is engineered by path calculation module, it is established using signaling protocols: Resource Reservation Protocol (RSVP) and/or Label Distribution Protocol (LDP) protocol. According to signaled information, each node on the route fills up MPLS routing table and reserve resources for a specific tunnel.

Once the packet comes to ingress of MPLS area, it is processed by Label Edge Router (LER). According to the predefined policies, edge router attaches a designated label to the packet and passes it inside the MPLS network. When packet is received, transit or Label Switch Router (LSR), checks the MPLS label and according to the MPLS table, swaps the old label with a new one, and passes the packet to the next router. When it comes to the end of the tunnel, egress LER, last label is removed and packet continuous with IP routing towards the final destination.

3.6.1 Segment Routing vs. MPLS

The following table presents the short comparison between Segment Routing and MPLS [44].

TECHNOLOGY FEATURES	SEGMENT ROUTING	MPLS
LABEL SIGNALING	IGP	LDP + RSVP-TE
IGP/LDP SYNC	NOT REQUIRED	REQUIRED
FAST REROUTE (FRR) 50ms	IGP	IGP+RSVP-TE
EXTRA STATES FOR FRR	NO	YES
OPTIMUM BACKUP	YES	NO
ECMP CAPABILITY	INBUILT	NO
TE STATE	ONLY HEAD-END	N^2 PROBLEM AT CORE NODE
SDN SUPPORT	YES	NO
ROUTING TYPE	SOURCE BASED	DESTINATION BASED
PATH CALCULATION	CONSTRAINED SPF OR PCE	IGP + RSVP-TE
SCALABILITY	HIGH	LOW
OPERATIONS & TROUBLESHOOTING	LOW	HIGH

Table 3.1: Comparison - Source Routing vs. MPLS

In MPLS network label signaling and resource reservation are done by implementing signaling protocols, LDP and RSVP-TE. As it was discussed before, in Segment Routing network it is enough to have an IGP protocol and once Segment Routing is configured, IGP will take labels and redistribute them within domain. There is no need to implement any other signaling protocol that is major benefit in terms of bandwidth and simplicity.

Moreover, LDP has lot of drawbacks regarding synchronization after a link failure. In fact, after a failure LDP must synchronize with IGP that calculates the new set of shortest paths within the network. Since there is a time gap while LSPs become stable again, in some cases it may cause the loss of packets because core routers do not know how to forward packets that are

addressed to external network. In Segment Routing only IGP is used and there is no need for synchronization with other protocols [45].

Having a good path protection is crucial for sensitive applications. In MPLS in some cases it is possible to have end-to-end path protection by calculating both primary and secondary path. For both paths resources must be reserved using RSVP-TE protocol [46]. All routers that are included in primary and secondary path must maintain the state of tunnels. This guarantees QoS and no traffic loss in case of failure - the path is fully protected and reroute can be perform in very fast manner <50ms (FRR). However, double resource reservation is not efficient in terms of network resource utilization, especially in busy core networks. On the other hand, Segment Routing uses post-convergent path that is automatically calculated by IGP upon link failure and guarantees optimal path in new situation. There are no extra states that should be maintained to protect the path. The FRR mechanism in Segment Routing is called Topology Independent Loop-Free Alternate (TI-LFA) and it guarantees <50ms convergence [47].

Equal Cost Multipath enables traffic balancing among equal cost paths between source and destination [48].In Segment Routing it is inbuilt - if there are two flows between the same source and destination (the same Prefix-SID) they will take different paths. This property supports network stability. In MPLS tunnels are determined strictly hop-by-hop meaning that ECMP is not supported.

In Segment Routing source routing paradigm is used, while in MPLS packets are tunneled by pushing the labels according their destination IP address. In Segment Routing network paths are determined by Constrained Shortest Path First (CSPF) algorithm. CSPF is extension of SPF algorithm. First, the shortest path algorithm is run, after which constrains are applied (available bandwidth, latency etc.) [49]. Path computation is usually done by an external entity such as SDN or PCEP. In MPLS paths are setup by combining IGP and RSVP-TE protocols.

Segment Routing was built for centralized data-plane network in mind. Even though, in theory, tunnels can be built manually, Segment Routing is fully

supported by SDN paradigm. MPLS has a different technology approach - control plane is distributed and paths can be setup and maintained by utilizing distributed protocols. In such distributed environment it is very difficult to apply centralized control.

Segment Routing is highly scalable compared to MPLS. First of all, it eliminates need for signaling protocols which simplifies overall architecture and leads to simplified and cheaper hardware. Moreover, number of FIB entries is highly reduced by applying Segment Routing - each node should have approximately N-1+number_of_interfaces. In MPLS each intermediate router has N^2 entries [50], which can create scalability problems in huge networks (e.g. Tier 1 core network).

At the end, Segment Routing simplifies overall operation and reduces need for network maintenance. Data plane is highly simplified since there are no signaling protocols. Furthermore, it enables easy operation by making labels constant over the network.

Chapter 4. Operating Environment

4.1 Software Defined Networks

IP networks recorded the rapid growth in past decades, which made them more complex and consequently difficult to manage. The main limitation comes from the fact that today's networks are built of switches, routers and other devices that became complex because they must implement a number of distributed protocols and they use closed and proprietary interfaces. In this environment it is difficult and sometimes almost impossible for network operators, third parties and vendors to innovate [51].

Traditional IP networks consider the control and data plane tightly coupled and embedded in the same network node. In other words, control function is distributed over network devices meaning that each device is responsible to make a forwarding decision autonomously. In early stage of IP networks development this was considered a good aspect because it guaranteed network resilience. However, any change on decentralized control plane requires changes on all network devices manually. Lack of automation in network managing makes today's networks static and unable to adapt for real time demands [52].

To overcome such limitations, new networking paradigm has been proposed – Software Defined Networking (SDN). In short, SDN can be defined as "*an emerging networking architecture where network control is decoupled and separated from forwarding mechanism and is directly programmable*". In SDN architecture brings logically centralized control named SDN controller, which has a global view on underlying network. Low-level devices become strictly forwarding elements without any control function. All the instructions they

receive from controller through specialized interface. The new protocols are defined for communication between controller and configurable switches. One of the most well-known protocols used by SDN controllers is OpenFlow.

The main pillars of Software Defined Networks are:

1. Separated data and control plane – control plane is removed from network devices and they became simplified forwarding elements

2. Control functionality is placed on a dedicated entity called SDN controller

3. Forwarding decisions are flow based. A flow could be defined as a packet stream between a source and destination that receive the same forwarding service.

4. The network is programmable through software applications running on the top of controller

4.1.1 The SDN architecture

The SDN architecture generally has three functional groups [53], as it's depicted below:

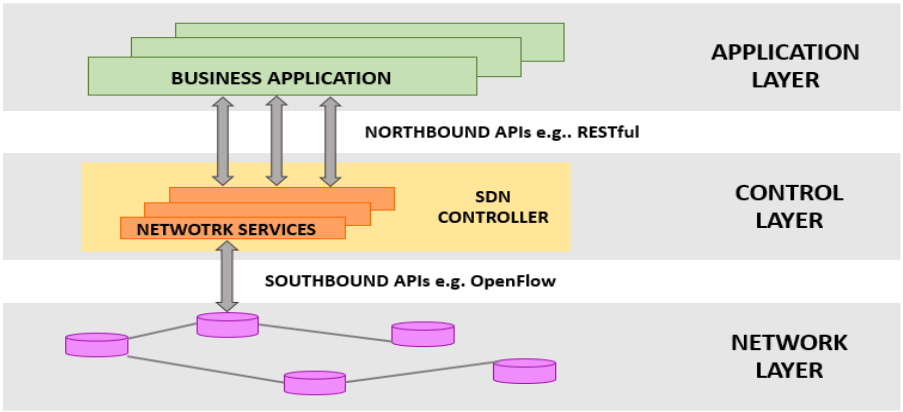


Figure 4.1: SDN architecture

Forwarding or data plane layer is placed on the bottom of SDN architecture. Data plane layer consists of switching devices connected in wired or wireless manner. Network devices perform set of elementary forwarding operations. They are programmable devices and they behave according to the instructions sent by controller.

The communication between SDN controller and programmable switches is enabled by southbound application program interfaces (southbound APIs). Southbound APIs facilitate efficient network control and enable SDN controller to dynamically make changes in forwarding plane in real time. For example, SDN controller can add or remove an entry in forwarding table through southbound interface. OpenFlow is the most common southbound interface, and will its functionalities will be explained later.

SDN controller is the “brain” of the network. It is centralized control point which manages flow control to the network devices below and the applications logic above. SDN controller makes abstraction view of the network, including statistics and the state of the network and sends it to the application level. Data plane could be controlled from the application level. Once instruction from the upper level is sent, controller takes it and forwards it to the lower level devices.

The northbound API presents a network abstraction interface to the applications that sit on the top of SDN stack. This interface enables network programmability from application level. The northbound API is certainly the most critical part of SDN architecture. SDN controller is valued by innovative applications it can support and northbound API must follow application requirements. Northbound APIs are used as well to connect SDN controller to automation stack and to orchestration platforms.

Application layer accommodate the set of applications that leverage functions offered by northbound API to implement operational logic and network control. From application level one can monitor physical network and control routing, firewalls, load balancers etc. All the commands coming from application layer are translated to southbound instructions that program behavior of forwarding devices.

4.1.2 Benefits of SDN

- Better network control - SDN promotes a central point of control to distribute provider's policies and configuration consistently throughout the network. SDN controllers provides complete visibility and control over network ensuring proper access control and traffic engineering

- Orchestration of multi-vendor environments – SDN controller can configure and manage any SDN capable device. A single protocol is used for communication between a controller and devices of any vendor

- Induces innovation – SDN gives possibilities to vendors, operators or a third party to develop applications, services and business models and trigger the revenue streams and more value from the network

- Reduces operational expenditures - network hardware is simplified by removing control function. Overall operation costs are reduced by easier network control and better network utilization

- Enhances network efficiency – centralized control and management increase automation and network orchestration. No need to configure individual network devices in forwarding plane to meet business policy changing. Network is directly programmable by a proprietary software or an open source automation tools

4.2 Segment routing and SDN

Segment Routing was designed for SDN era. Segment Routing and SDN (SDN-SR) is very powerful combination and present a winning proposal for service providers. A SDN controller with a global view of the network it is capable to process business requirements and policies and translate them in Segment Routing paths. This leaves to service providers a huge number of possibilities to provide differentiated services and optimize their network [1].

SDN-SR is the perfect platform for application engineered routing. It gives possibility to an application to require specific path (in terms of latency, bandwidth, SLA) parameters and to push the packets through that specific path, without having to inform the network about it. That has reciprocal benefit for both application and network operation. Application can directly specify its requirements and push the traffic on optimal path. On the other hand data layer is light-weighted because it doesn't have to maintain the traffic paths – they are directly specified from application [54].

In SDN-SR environment the network intelligence is combined [55]. Segments, as instructions, are designed in a smart and simple way to enable efficient traffic steering through the network. Segments give lot of possibilities to SDN controller how to express a wanted path. SDN controller intelligence is used to map the optimal path onto segments.

The key benefit of SDN-SR architecture is simplified control plane. Signaling protocols such as LDP and RSVP-TE are not necessary for SR functioning, which is direct benefit in terms of simplicity and bandwidth relaxation. State is maintained only at the head-end router. Intermediate nodes do not have to maintain tunnel information that leads to improved scalability. Explicit routing is possible with or without ECMP – a controller can decide but stating proper SIDs. Automated FFR is guaranteed for any topology.

In reality, SDN controller should support the protocols that are essential for SR, PCEP and BGP-LS. SDN controller behaves as stateful PCE and can compute path in terms of segments and push it back to the PCC. As a property of stateful PCE, SDN controller can initiate PCEP session and perform flow optimization, if necessary. Topology information is obtained by configuring BGP-LS peering with BGP speakers. Each IGP domain must have at least one BGP speaker that will redistribute LSDB to SDN controller.

4.3 OpenDaylight

OpenDaylight project (OpenDaylight controller, ODL) is an open source SDN project governed by Linux Foundation [56]. Open source SDN controllers enable easy network testing and support network virtualization. Architecture

of open source solutions is typically modular meaning that controller consists of pluggable modules that perform different network functions. Open source projects give possibility for development and customization. Today, there are many open source projects launched for further development such as ONOS, OpenContrail, Pox, Ryu etc.

OpenDaylight project was announced back in 2013 with an aim to accelerate SDN development and industry adoption. ODL is based on Java programming language and supports OpenFlow standard [57]. Some of the companies that contribute ODL development are Cisco, Juniper Networks, VMware, Microsoft, Ericsson etc. At the moment three four releases are available Hydrogen (Feb, 2014), Helium (Oct, 2014), Lithium (June 2015) and Beryllium that was released in February 2016.

4.3.1 ODL Architecture

Detailed OpenDaylight architecture diverse among releases [58]. Simplified ODL architecture presented below (figure 4.2) is common for all releases:

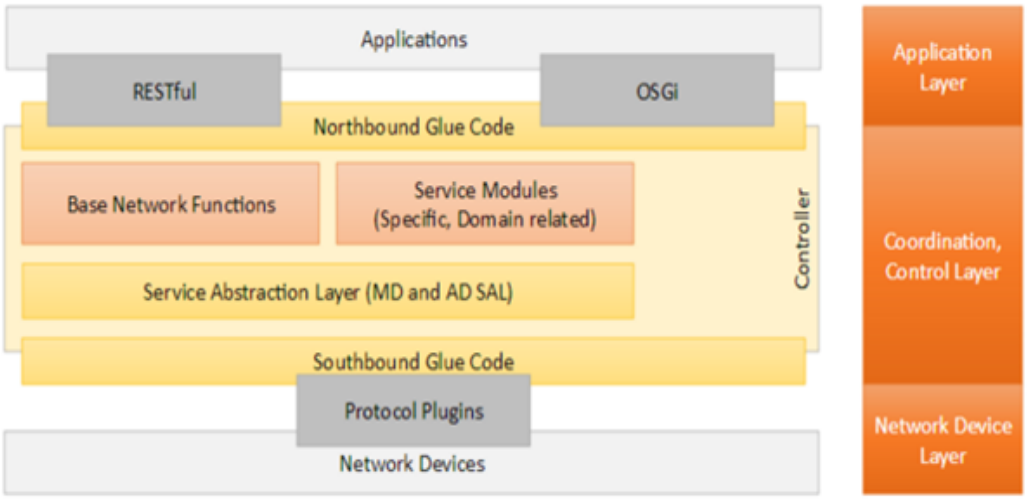


Figure 4.2: OpenDaylight architecture

As all SDN controllers, ODL consists of three main parts:

1. Southbound APIs
2. Control function layer
3. Northbound APIs

Southbound Interface

At southbound interface different protocol could be enabled as separate plugins. For instance ODL supports OpenFlow, BGP-LS, PCEP, LISP etc. Commonly, one plugin includes connection, session and state managers, error and packet handler mechanism and set of basic services. Supported protocols communicate to service abstraction layer (SAL).

Control Layer

The main components of ODL are service layer abstraction, service functions and pluggable modules.

Service Layer Abstraction (SAL) represents a key bundle between service producers and consumers. Modules that provide services have to register their APIs to the SAL registry. Whenever a request from service consumer comes, SAL binds them into 'contract'. There are two SAL architecture: application driven SAL and module driven SAL.

As was mentioned before, an open source project has pluggable module that enable particular function. However there are some basic network functions that come as preconfigured part of controller. Some base network functions that come shipped with ODL are:

- Topology functions – a service for discovering network layout by subscribing to processes of network-link discovery
- Statistics services – for managing state of counters across the nodes, flows and queues
- Switch manager – stores discovered nodes
- Forwarding services – manage network flow state and forwarding rules

Platform services modules or vendor components enhance SDN controller functionality. Some of platform oriented services are BGP-LS/PCEP that support traffic engineering, VTN (Virtual Tenant Network) component that enables network virtualization using OpenFlow, service function chaining that enables forming a ordered list of services, and etc.

Northbound Interface

ODL Controller exposes northbound APIs to the upper layer applications using OSGi framework or bidirectional REST APIs. REST APIs can be used by application that runs on the same computer as the controller or it can be totally different or remote machine. REST is based on popular technologies such as HTML, JSON and XML that enables straightforward combining with programming language as Python, Java, C. Interaction with top level application is done through HTTP basic operations GET, POST, PUT and DELETE. Data transmitted via REST API can be used on higher level to make higher-level business decisions, run algorithms, analytics etc. And results of this analytics can be channeled back to ODL Controller to for instance, create new rules in the network.

4.4 Virtual Network

A virtual network is a network that is made of virtual links. Virtual links represent non-physical connections between network devices implemented by using network virtualization method. There are two forms of virtual networks. One is a protocol-based virtual network such as VPN, VLAN, and VPLS. Second form of network virtualization is based on using virtual devices, such as Virtual Machines (VM) inside of hypervisor [59].

In this work virtual network is built of virtual routers which software run inside of virtual machines placed on same hypervisor. For building the network VMware virtual machine is used and Cisco IOS XRv virtual router.

4.4.1 VMware

VMware Player is desktop virtualization software that can run several guest operating systems at the same time on a single PC. User interface is quite easy to use and all functionalities are easy to find. VMware player supports hundreds of guest operating systems, from old operating systems to the latest ones [60].

VMware also supports guest operating system portability. Where one can take easily take already installed guest operating system and make clone of this virtual machine, that later can be taken to another location. This eliminates need to set up new virtual machine from the scratch.

VMware networking features

VMware Player gives several possibilities to configure a virtual machine for virtual networking [61]. Network adapters could be in one of following modes:

- Bridged mode – enables virtual machine to access Internet through host's Ethernet adapter. This is the easiest way to connect VM to internet

- Network Address Translation mode (NAT) – VM and host share single network identity, IP and MAC address, meaning that VM is not visible outside of network

- Host-only virtual adapter – creates a network that is completely within the host computer. This approach is useful for creating isolated virtual networks – virtual machines and the host virtual adapter are connected to a private network

- Custom networking – manual configuration of network connection. Manual configuration enables creation of sophisticated virtual networks

that can be connected to external networks or that can entirely run on host computer

4.4.2 Cisco IOS XRv

Cisco IOS XRv Router is a 32-bit Virtual Machine (VM) running on QNX microkernel [62]. XRv virtual machine contains a route processor (RP) together with control plane functionality, as it is shown on the figure 4.3. It also has network interfaces with their corresponding functionality. XRv represents Cisco IOS XR software and operating systems that are running on actual Cisco hardware. It gives user possibility to work on Cisco routers without having a hardware router. However, IOS XRv is not complete emulation of any physical cisco router or module.

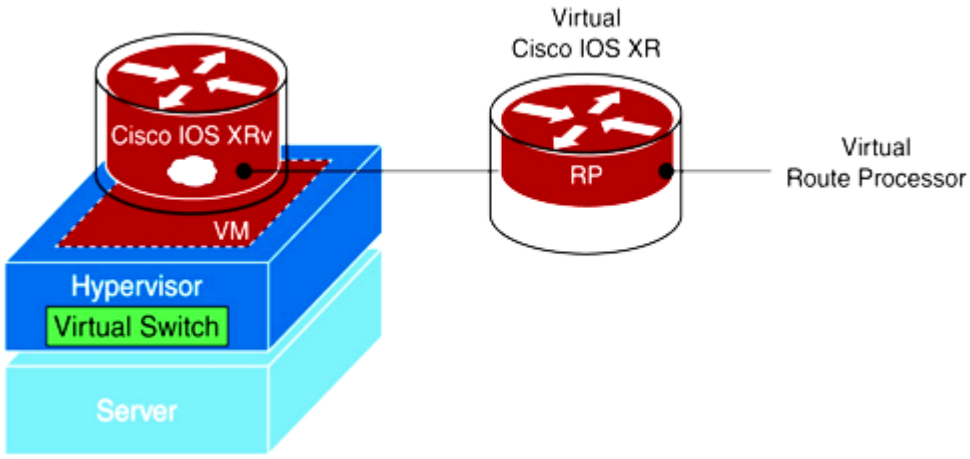


Figure 4.3: Cisco IOS XRv Router Virtual Form Factor

IOS XRv Features

Cisco IOS XRv support the range of IOS XR features such as manageability, control plane and switching and routing features. Cisco XRv supports E1000 and VirtIO network drivers supported by major supervisors. Virtual Machine can have up to 8 cores when running IOS XRv, and they can be directly configured in the hypervisor and IOS XRv detects them without extra configuration.

When running Cisco IOS XRv, the following features and services are available:

- IP features – Cisco XRv supports most of IPv4 and IPv6 services: unicast, multicast, ECMP, load balancing, addressing and ICMP
- Layer 3 protocols supported - BGPv4. OSPFv2, OSPFv3, IS-IS
- MPLS features – essential protocols supported LDP and RSVP as well as Diffserv Aware TE, MPLS control plane, MPLS forwarding and balancing
- Network Management features – enhanced CLI, XML interface and simple network management protocol support

Configuration of Cisco XRv can be done in two ways:

- By setting up a serial port in VM and connecting to Cisco XRv CLI commands through a serial console (e.g. PuTTY)
- Or by using remote Secure Shell or Telnet to connect to the management interface to access Cisco XRv router

CISCO XRv Benefits

The major benefits of using Cisco XRv:

- Hardware independence - XRv runs inside a virtual machine and can be run on any hardware with x86 architecture running virtualization software
- Resource sharing abilities - Resources that are used by XRv are managed by a hypervisor. Hypervisor is in charge of resource sharing among virtual machines. Hypervisor can allocate different virtual machines with specific and different hardware resource that could be required in different use cases
- Flexibility in deployment - XRv virtual machine can be easily moved from one place to another. For instance image of the XRv router virtual machine can be easily moved from one physical location to another without actually moving any hardware resources

IOS XRv comes in several packages: Demo Locked, Production, Simulation, and Demo Unlocked. Main difference between this packages are price (free or not free), and data transmission rate limit.

For this thesis project we used free Demo Locked version. This version has data transmission limit of 2 Mbps. It can be downloaded from Cisco website. Even though it has limited functionality, it is still respectable software for demonstration of basic use cases in control plane, and for training and familiarization with IOS XR software.

Chapter 5. Implementation

In this chapter implementation of this thesis work will be explained. Different network structures were created to test features of Segment Routing in SDN environment. The short summary of implementation is:

- Network 1 - Segment Routing ECMP test-bed
- Network 2 - Multi-domain Segment Routing test-bed
- Network 3 - Performance Analysis test-bed

All three implementations follow the same steps. In following subchapters implementation details will be presented: how to setup VMware for virtual network, how to import Cisco XRv disc image, how to set up controller and RESTful client. Afterwards, the configuration of each network will be presented.

5.1 Implementation Overview

The SDN-SR network is built is few stages. The summary of implementation is following:

Create a network in VMware:

- Import Cisco IOS XRv Disk Image in VMware virtual machine
- Clone a VM in order to create more routers (one VM - one virtual router)
- Customize the settings in VMware: network adapters, processors, memory

Setup controller

- Create an Ubuntu server and setup OpenDaylight Lithium SDN controller
- Setup controller IP address that will be used for communication between routers

Configure routers using Cisco CLI

- Configure interfaces:
    - Loopback interface
    - Management interface
    - GigabitEthernet interfaces for each network adapter
- Setup IS-IS instance on all network interfaces
- Configure MPLS
    - Configure PCEP peering between each node and controller
    - Configure Segment Routing
- For each IGP domain select a node that will be BGP speaker and configure BGP instance
    - For a single domain network one instance is enough to make a communication with a controller (BGP-LS)
    - For multi-domain network two (or more) BGP instances are needed. One for BGP-LS and one more for a single peering network

Create clients and servers

- Create a VM for each client/host
- Connect them to proper nodes by setting up IP address and netmask

5.2 Environment Setup

5.2.1 Network Build-Up

This chapter guides through all necessary steps for setting up the virtual network. As was mentioned before network is built of virtual switches that are run as separate VMs.

- The first step is to import Cisco IOS XRv VM. The VM can be downloaded from Cisco's web site and one can directly open it VMware by clicking 'Open a Virtual Machine'

- When VM is imported one should set up a number of network adapters. In SDN each router should have a management interface that will be used for communication with SDN controller. That adapter will be in Bridge mode. For connection to other routers Custom network adapter will be used. For each link VMnet should be chosen. The endpoints of one link should have the adapters with same VMnet. In the figure below, settings of R1 and R2 are presented. As could be noticed, they are both connected to VMnet 11, meaning that they share the link.

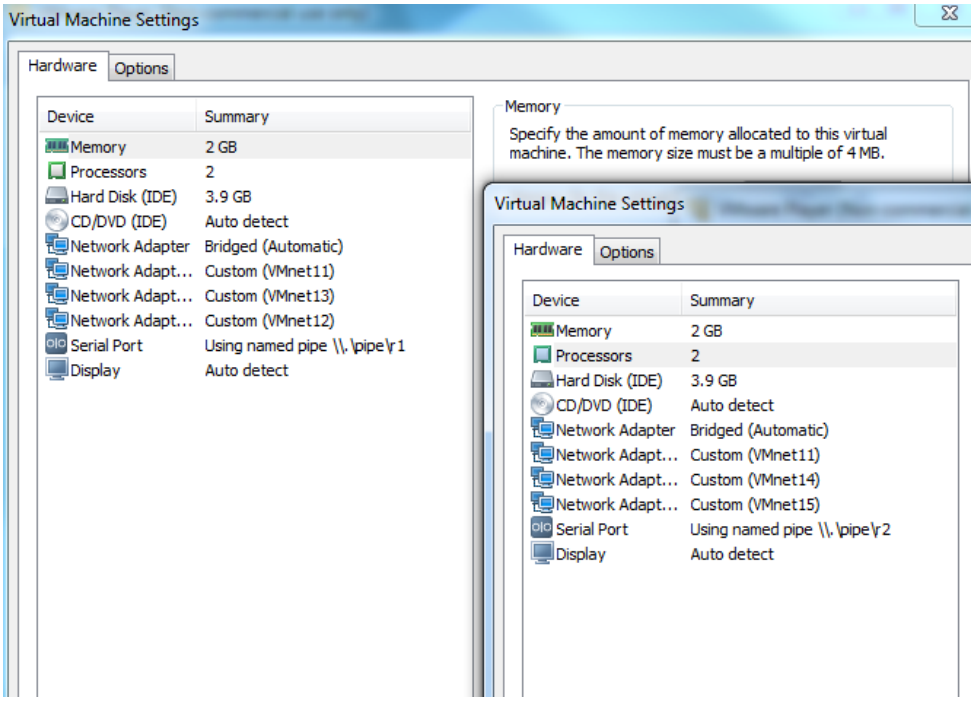


Figure 5.1: VMware settings

- For routers that have more than 2 network adapters, one must make changes in *vm* file. VMware does not recognize more than two adapters initially. To make them up, the following changes must be performed:

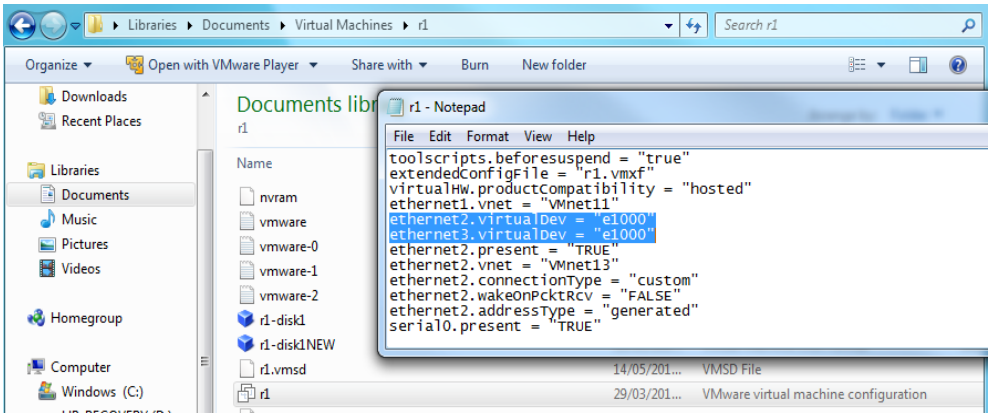


Figure 5.2: Edit VMware configuration file

- Other settings: 2GB memory, 2 CPU and 4 GB hard disk

- To enable access to CLI, serial port must be setup and pipe should be configured (figure VM settings)

- Router is accessed through serial console. We used PuTTY, free open-source terminal and serial console

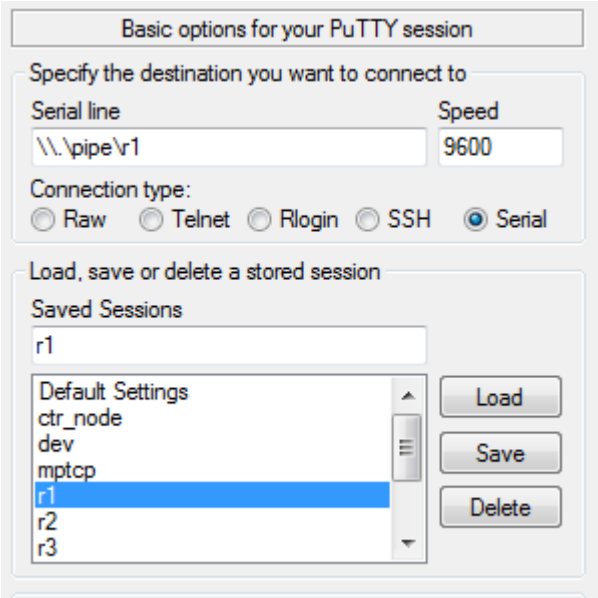


Figure 5.3: PuTTY settings

All routers in the network are setup in similar way. One can use the option clone the router and then just adjust network adapters. After this phase the routers can be boot up and configured.

5.2.2 OpenDaylight Setup

- Create VM where Linux Ubuntu will be installed. The basic settings of VM are 8GB RAM and 2 CPUs
- Download OpenDaylight from [63]. In this work we used Lithium distribution, which by default does not come with any feature. Upon installing ODL one can customize environment and add the features he/she needs

- Install ODL , one can follow this installation guide [65]
- Run karaf container bin/karaf
- Install necessary features by typing feature:install odl-bgpcep-bgp-all odl-bgpcep-pcep-all install odl-restconf-noauth [66]
- Upon feature installing a number of xml files will be generated in etc/opendaylight/karaf that will be reconfigured
- Setup OpenDaylight IP address through sudo vi /etc/network/interfaces. IP address that we used for ODL is 192.168.100.100

5.2.3 Web Based REST API client

OpenDaylight uses RESTful northbound interface to communicate with upper layers. To be able to receive an abstracted topology and to give commands to SDN controller one must use RESTful client. There are lot web-based RESTful clients available today, however we used Postmen [61]. Postmen can be installed as a browser extension and it has the following interface:

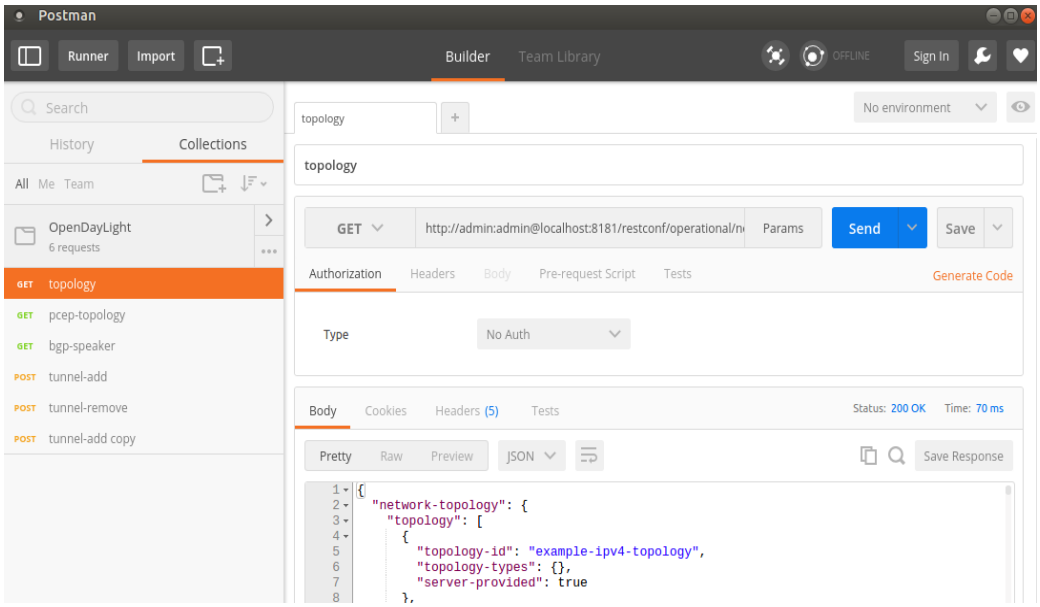


Figure 5.4: Postman interface

Postman has easy-to-use interface. On the left side one can find the collection of basic actions and history of requests. Central part is divided in request and response window. Request action can be any of classic HTTP commands GET, POST, PUT and DELETE. The data can be represented in JSON and XML format. To connect server to the client the one must specify server location address.

In this work Postmen was used to retrieve the network topology from controller. One can easily see information that controller actually receive from underlying network, such as links and nodes parameters (LSDB). We used Postmen to set up tunnels in the network. This is done by posting a XML request to the controller through Postmen's interface. The actual tunnel requests that we created will be presented in following chapters.

5.3 Network Configuration

Upon initial setup, network is configured through CLI. Firstly address management is done and interfaces are configured accordingly. Then protocol configuration is added - IS-IS, PCEP, MPLS and BGP. Also, for each network-case we added extra configuration to OpenDaylight modules (BGP and PCEP) in order to enable communication with underlying BGP speakers. You can find network configuration in following subchapters

5.3.1 Network 1 - Segment Routing ECMP test-bed

The first network we used to test ECMP. This is a simple network with two disjoint paths between source and destination. It consists of four routers as shown on the figure below:

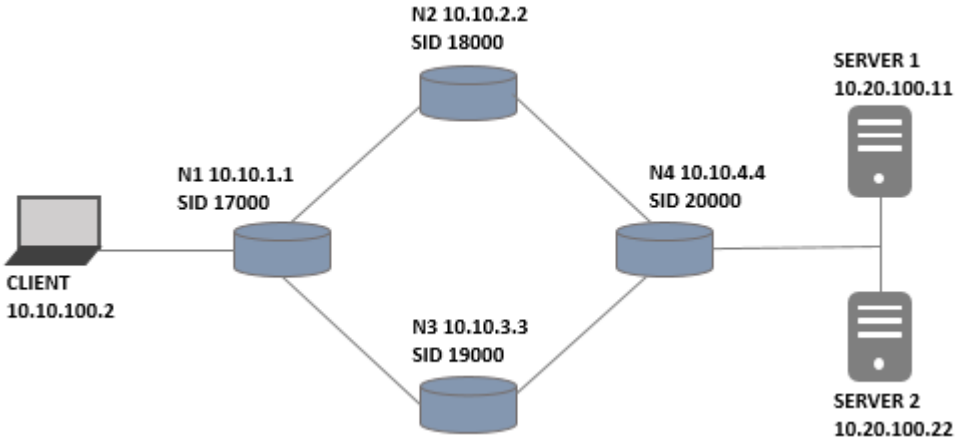


Figure 5.5: Network 1 topology

Network has one client and two servers. The client with IP address 10.10.100.2 is connected to N1. Two servers are in the same subnet attached to N4. Their IP addresses are 10.20.100.11 and 10.20.100.22 for server 1 and server 2 respectfully.

In following sections you can find configuration of the routers and SDN controller. Since router configurations are similar to each other, only the configuration of the first router will be present in the work. Other routers are configured accordingly. The difference is in terms of IP addresses and SIDs and they will be represented in table 5.1.

Router Configuration

Firstly, we configured interfaces as on the figure below. Each router has Loopback address, management interface and two GigabitEthernet interfaces. Management interface is used for communication with SDN controller. For routers interconnection we used GigabitEthernet interfaces.

INTERFACES configuration

```
1   !-----INTERFACES---------------------------------
2   hostname n1
3   ipv4 unnumbered mpls traffic-eng Loopback0
4   interface Loopback0
5    ipv4 address 10.10.1.1 255.255.255.255
6   !
7   interface MgmtEth0/0/CPU0/0
8    ipv4 address 192.168.100.1 255.255.255.0
9   !
10  interface GigabitEthernet0/0/0/0
11   ipv4 address 10.10.12.1 255.255.255.0
12  !
13  interface GigabitEthernet0/0/0/1
14   ipv4 address 10.10.13.1 255.255.255.0
15  !
16  interface GigabitEthernet0/0/0/2
17   ipv4 address 10.10.100.1 255.255.255.0
18  !
19  route-policy PASS
20    pass
21  end-policy
22  !
```

Figure 5.6: Network 1 interfaces configuration

Next, we configured IGP protocol in the network. As it was mentioned before we used IS-IS and we configured it as on the figure below. In the line 2 we initiated the IS- IS instance. This is the necessary command for initiating routing process on the router. One can put the area tag to identify the area to which instance was assigned. This is necessary if you build multi-domain network, otherwise it could be skipped. Even though in this case was not necessary, we clarify the area 10.

By using the command is-type we configured router as a level-2-only (line 3), meaning that router can only talk to other level-2 routers. All the routers in the domain are configured as level 2 routers. The next command net specifies the NET for the routing process. This is important to configure if you are building up multi area domain. To keep other routers updated if a neighbor router goes up or down, one should enable log-adjacency-changes (line 5).

Inside of address-family ipv4 unicast, firstly we configured metric-style wide, meaning that IS can receive only new style TLVs. The line 8 refers to Incremental Shortest Path First (ISPF). ISPF is the way a router computes shortest path tree (SPT) upon topology changes. Traditionally SPT is

completely recomputed if there is any change in the network. However, in most cases is completely unnecessary to compute the entire tree from the scratch. ISPF allows re-computation only of affected part of the tree providing faster convergence and saving CPU resources.

To enable a router to flood MPLS traffic engineering link information through configured IS-IS level, one must configure it inside IS-IS instance (lines 9-12). In our case, we configured distribution of MPLS links in level-2-only and we specified Loopback address as router ID. Here we also specified the preference of Segment Routing labels over LDP labels.

At the end, we configured interfaces that are advertised by IS-IS. Firstly, we enabled Loopback advertisement. This interface is in passive mode, meaning that we do not want packets to be sent on that interface, we just want its announcement in the network. Here we also specified Node-SID that will be binded to router's Loopback address. Other two interfaces are configured as GigabitEthernet, point-to-point links and address family unicast. Those interfaces are interfaces between routers N1-N2 and N1-N3.

IS-IS configuration

```
1   !-----ISIS-------------------------------------
2   router isis 10
3    is-type level-2-only
4    net 49.0000.0000.0000.0001.00
5    log adjacency changes
6    address-family ipv4 unicast
7     metric-style wide
8     ispf
9     mpls traffic-eng level-2-only
10    mpls traffic-eng router-id Loopback0
11    redistribute connected
12    segment-routing mpls sr-prefer
13   !
14   interface Loopback0
15    passive
16    address-family ipv4 unicast
17     prefix-sid absolute 17000
18    !
19   !
20   interface GigabitEthernet0/0/0/0
21    point-to-point
22    address-family ipv4 unicast
23    !
24   !
25   interface GigabitEthernet0/0/0/1
26    point-to-point
27    address-family ipv4 unicast
28    !
29   !
30  !
```

Figure 5.7: Network 1 IS-IS configuration

After IS-IS, we configured MPLS. To enter MPLS configuration mode one should type mpls oam (MPLS Operation and Management). Firstly we enabled two interfaces N1-N2 and N1-N3 for MPLS traffic engineering. Then we enter in PCE configuration mode and there we specified source and peer IP addresses. In our network SDN controller is PCE, so we defined its IP address as peer address (192.168.100.100). IP address of the source is router's management interface IP address. That interface is only used to communicate to controller. PCE must be configured on each router to enable path setup. Also, in OpenDaylight, in PCE module, we will configure peering address.

Here, it must be specified that we want path computation in SR mode (lines 13-16). Firstly, stateful client. This means that router will add stateful capabilities TLV when opening the new session. Moreover, we configured delegation of all active tunnels to PCE. In reality, this command allows SDN

controller to change existing LSPs while computing new paths. This is useful if PCE wants to re-optimize traffic-engineered tunnels.

Speaker ID is always router's Loopback ID. To configure the range of tunnel IDs to be used for stateful PCE instantiation requests we used command auto-tunnel-pcc and we specified the min and max tunnel ID. In line 23 re-optimization for the specified number of seconds is configured, meaning that installation of new LSPs with new labels after tunnel re-optimization will happen in specified number of seconds.

MPLS configuration

```
1   !-----MPLS----------------------------------------
2   mpls oam
3   !
4   mpls traffic-eng
5    interface GigabitEthernet0/0/0/0
6    !
7    interface GigabitEthernet0/0/0/1
8    !
9    pce
10    peer source ipv4 192.168.100.1
11    peer ipv4 192.168.100.100
12    !
13    segment-routing
14    stateful-client
15     instantiation
16     delegation
17    !
18    speaker-entity-id 10.10.1.1
19   !
20   auto-tunnel pcc
21    tunnel-id min 1 max 99
22   !
23   reoptimize timers delay installation 0
24  !
25  end
```

Figure 5.8: Network 1 MPLS configuration

The presented configuration is valid for all the routers in the network; one just should change IP addresses and SIDs that are router-specific. Here in the table one can find IP addresses and SIDs we used.

Router ID	Loopback Address	Management Interface	GigabitEternet Interfaces	Node SID
N1	10.10.1.1	192.168.100.1	10.10.12.1 10.10.13.1 10.10.100.1	17000
N2	10.10.2.2	192.168.100.2	10.10.12.1 10.10.24.1	18000
N3	10.10.3.3	192.168.100.3	10.10.13.1 10.10.34.1	19000
N4	10.10.4.4	192.168.100.4	10.10.24.1 10.10.34.1 10.20.100.1	20000

Table 5.1: Network 1 – interfaces and SIDs

In this network, we elected N4 to be a BGP speaker and to redistribute all IGP information to OpenDaylight. The BGP configuration is presented in the figure below. As in IS-IS, firstly BGP instance should be opened. Again we have area tag 10, which is the same as IS-IS tag. Then, we specified the router Loopback address as a router ID. BGP neighbor is SDN controller with is specified in line 10. SDN controller belongs to remote AS and that is configured in line 11. Management interface is used to update SDN controller about link-state information. In line 15 and 16 we enabled policy exchanging.

BGP configuration

```
1   !-----BGP---------------------------------------
2   router bgp 10
3    bgp router-id 10.10.4.4
4    bgp log neighbor changes detail
5    address-family ipv4 unicast
6     network 10.10.0.0/16
7    !
8    address-family link-state link-state
9    !
10   neighbor 192.168.100.100
11    remote-as 1
12    update-source MgmtEth0/0/CPU0/0
13    session-open-mode passive-only
14    address-family link-state link-state
15     route-policy PASS in
16     route-policy PASS out
17    !
18   !
19  !
```

Figure 5.9: BGP configuration at node N4

OpenDaylight Configuration

In ODL in BGP module which is placed '*/etc/opendaylight/karaf*' in file *41-bgp-example.xml* should be reconfigured. In the line 5 we specified the management interface of BGP speaker (N4).

ODL configuration

```
1  <module>
2     <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">
3                          prefix:bgp-peer</type>
4     <name>example-bgp-peer</name>
5     <host>192.168.100.4</host>
6     <holdtimer>180</holdtimer>
7     <peer-role>ibgp</peer-role>
8                  ......
9   </module>
```

Figure 5.10: Network - ODL configuration

Network try-out

To check network is working properly one should test it by pinging a server from the client, as on figure below. Connection with ODL can be checked by pinging the controller. Additionally, by using the show command one can check out the MPLS forwarding table, figure below. As could be noticed Node SIDs are configured correctly. Adj-SIDs are assigned automatically on two interfaces that lead to nodes N2 and N3.

traceroute 10.20.100.1

```
1  anakos@ubuntu:~$ traceroute 10.20.100.11
2  traceroute to 10.20.100.11 (10.20.100.11), 30 hops max, 60 byte packets
3   1  10.10.100.1 (10.10.100.1)  12.010 ms  11.482 ms  11.282 ms
4   2  10.10.12.2 (10.10.12.2)  40.752 ms  40.759 ms  40.703 ms
5   3  10.10.24.4 (10.10.24.4)  38.339 ms  38.298 ms  38.242 ms
6   4  10.20.100.11 (10.20.100.11)  40.357 ms  40.309 ms  40.253 ms
```

Figure 5.11: Network 1 test - pinging from client to server 1

MPLS forwarding table - node N1

```
1   RP/0/0/CPU0:n1#sho mpls forwarding
2   Fri Apr  1 15:08:33.715 UTC
3   Local  Outgoing    Prefix            Outgoing     Next Hop        Bytes
4   Label  Label       or ID             Interface                    Switched
5   ------ ----------- ----------------- ------------ --------------- ------------
6   18000  Pop         No ID             Gi0/0/0/0    10.10.12.2      0
7   19000  Pop         No ID             Gi0/0/0/1    10.10.13.3      0
8   20000  20000       No ID             Gi0/0/0/0    10.10.12.2      0
9          20000       No ID             Gi0/0/0/1    10.10.13.3      0
10  24000  Pop         SR Adj (idx 1)    Gi0/0/0/0    10.10.12.2      0
11  24002  Pop         SR Adj (idx 1)    Gi0/0/0/1    10.10.13.3      0
```

Figure 5.12: Network 2 - show MPLS forwarding table at N1

5.3.2 Network 2 - Multi-domain Segment Routing test-bed

The next thing we wanted to examine in this thesis is implementation of Segment Routing in multi-domain network. This task was not straightforward having in mind that Segment Routing is not standard yet and not lots of references were found regarding this topic. Moreover, we had limitation on IOS XRv that does not support all Segment Routing features that made us to improvise in order to enable Segment Routing traffic engineering in multi-domain network.

The multi-domain network we built consists of two IGP domains. Each IGP domain has structure as Network 1 explained in the previous chapter, as in following figure:

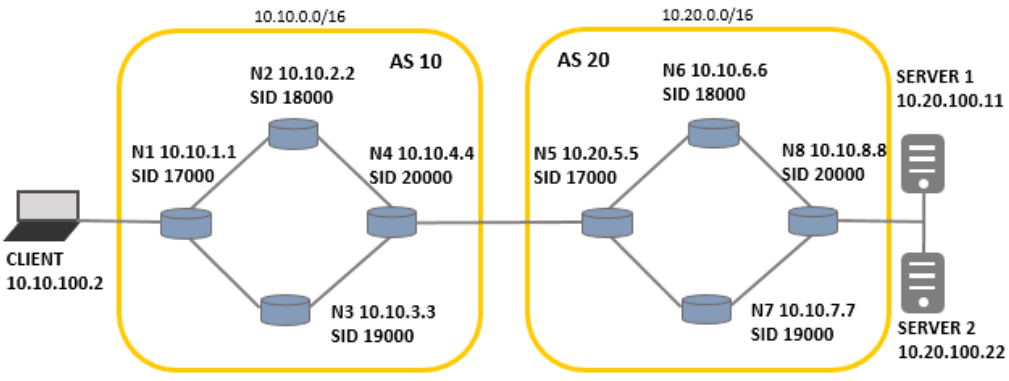


Figure 5.13: Network 2 - Multi-domain Network

The network consists of two autonomous systems AS 10 and AS 20. Each AS has 4 routers, N1-N4 form AS 10 and N5-N8 form AS 20. Inter-domain link is the link between N4 and N5. To N1 has a subnet 10.10.100.0/24 attached with a Client. Router N8 has a subnet 10.20.100.0/24 with two servers.

Border routers, N4 and N5, implement BGP in order to exchange reachability information. Also, they peer SDN controller and deliver IGP information (each one delivers link-state information of its own AS). In order to peer both BGP speakers one must add an extra BGP module in OpenDaylight configuration. Both routers' and OpenDaylight configuration could be find in following sections.

Router Configuration

The basic network configuration of interfaces, IS-IS and MPLS is identical as in the previous network. Both domains have identical structure that is derived from the previous network case, so we will skip the basic configuration and just consider the changes we made in order to enable this multi-domain network working.

To enable tunnel setup from N1 to N8 the route must be ping-able. This means that N1 must have the route to N8 in its forwarding table. Since N8 is in external domain the only way it could learn it is through the BGP protocol. BGP normally runs on the inter-network link N4-N5. N4 and N5 are configured to exchange the network reachability among themselves. However, the information about AS 20 that N4 has must be further redistributed within AS 10 if we want to make inter-domain routers pingable. In theory, this can be done by configuring IS-IS to redistribute BGP information in its message. This IS-IS configuration was not available in the Cisco XRv we used for this project. Instead, we used static route configuration at node N1, as it is shown below. In the line 3, we configured N4 to be the gateway for the default IP address. In the next line we added static route for all 10.20.0.0/16 network. The line 4 was necessary for the successful tunnel setup.

Router static configuration

```
1  router static
2   address-family ipv4 unicast
3    0.0.0.0/0 10.10.4.4
4    10.20.0.0/16 10.10.4.4
5   !
6  !
```

Figure 5.14: Network 2 -static route at N1

The additional configuration was implemented on the border routers. N4 and N5 must run both BGP and BGP-LS protocol and advertise both its neighbors and SDN controller about link-state information.

In BGP configuration we can see that a router now has two neighbors - N5 and OpenDaylight. N4 is configured to redistribute link state information of its network 10.10.0.0/16 (line 6) to its neighbors. The neighbors are defined below (line 10 and 18). The first neighbor (line 10-17) is defined by its interface - 10.30.0.5 is N5 interface towards N4. Furthermore, we specified external AS 20 and source link (GigabitEthernet0/0/0/2 is the N4 link towards N5). The second neighbor is OpenDaylight with IP address 192.168.100.100. N4 delivers link-state information through management interface (line 20).

BGP configuration on the N4

```
!-----BGP---------------------------------------
router bgp 10
 bgp router-id 10.10.4.4
 bgp log neighbor changes detail
 address-family ipv4 unicast
  network 10.10.0.0/16
 !
 address-family link-state link-state
 !
 neighbor 10.30.0.5
  remote-as 20
  update-source GigabitEthernet0/0/0/2
  address-family ipv4 unicast
   route-policy PASS in
   route-policy PASS out
  !
 !
 neighbor 192.168.100.100
  remote-as 1
  update-source MgmtEth0/0/CPU0/0
  session-open-mode passive-only
  address-family link-state link-state
   route-policy PASS in
   route-policy PASS out
  !
 !
!
```

Figure 5.15: Network 2 - BGP configuration of router N4

Let's now consider Segment Routing in multi-domain network. As it was mentioned, the basic Segment Routing commands in IS-IS configuration enables one to configure Node-SID. If everything is done properly, adjacent SIDs are then builded automatically by taking the values outside of SRGB range. However, in this case we have an external link that does not run any protocol except BGP. The IETF explained in the draft [draft bgp-prefix sid] that for multi-domain Segment Routing BGP-Prefix -SID must be assigned in order to be able to construct the SR tunnel. By specifying BGP-Prefix-SID in ERO, the packet will be routed to external domain according to the shortest path. However, this function was not supported by Cisco XRv, which means that we were not able to introduce Segment Routing capability through BGP protocol.

Since at this point only IS-IS supported Segment Routing configuration, the only way to build the segment towards external link is to enable IS-IS on that link. The new IS-IS instance runs just between N4 and N5 and form adjacent segment that is crucial for multi-domain traffic engineering. Having in mind

that this is multi-domain network; we disabled IS-IS message redistribution from domain AS10 to AS20 and vice versa. The new instance is presented below. Both N4 and N5 run this instance.

Additional IS-IS instance

```
router isis 100
 is-type level-2-only
 net 49.1111.0000.0000.0004.00
 log adjacency changes
 address-family ipv4 unicast
  metric-style wide
  ispf
  mpls traffic-eng level-2-only
  mpls traffic-eng router-id Loopback0
  segment-routing mpls sr-prefer
 !
 interface GigabitEthernet0/0/0/2
  address-family ipv4 unicast
  !
 !
!
```

Figure 5.16: Network 2 - additional IS-IS instance

Router ID	*Loopback Address*	*Management Interface*	*GigabitEternet Interfaces*	*Node SID*	*Autonomous System*
N1	**10.10.1.1**	**192.168.100.1**	**10.10.12.1 10.10.13.1 10.10.100.1**	**17000**	**10**
N2	**10.10.2.2**	**192.168.100.2**	**10.10.12.2 10.10.24.2**	**18000**	**10**
N3	**10.10.3.3**	**192.168.100.3**	**10.10.13.3 10.10.34.3**	**19000**	**10**
N4	**10.10.4.4**	**192.168.100.4**	**10.10.12.4 10.10.24.4 10.30.0.4**	**20000**	**10**
N5	**10.20.5.5**	**192.168.100.5**	**10.20.56.5 10.20.57.5 10.30.0.5**	**17000**	**20**
N6	**10.20.6.6**	**192.168.100.6**	**10.20.56.6 10.20.68.6**	**18000**	**20**
N7	**10.20.7.7**	**192.168.100.7**	**10.20.57.7 10.20.78.7**	**19000**	**20**
N8	**10.20.8.8**	**192.168.100.8**	**10.20.68.8 10.20.78.8 10.20.100.1**	**20000**	**20**

Table 5.2: Network 2 - interfaces and SIDs

OpenDaylight Configuration

OpenDaylight configuration must be changed to peer both domains. The module configuration file is xml file named *41-bgp-example.xml* placed in '*/etc/opendaylight/karaf'*. The first module peers N4 - management interface 192.168.100.4, AS 10. The new added module peers N5 - 192.168.100.5, AS 20. Peer role was changed to *ebgp* since SDN controller is placed in remote AS.

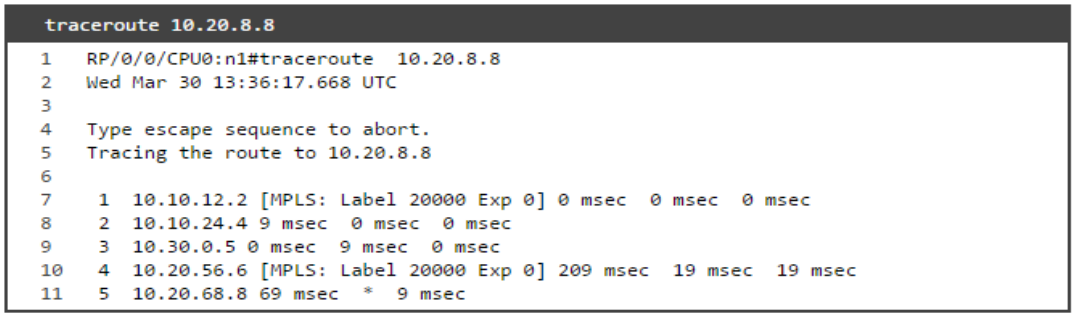
ODL configuration for multidomain network

```
1
2    <module>
3       <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">
4                           prefix:bgp-peer</type>
5       <name>example-bgp-peer</name>
6       <host>192.168.100.4</host>
7       <holdtimer>180</holdtimer>
8       <peer-role>ebgp</peer-role>
9       <remote-as>10</remote-as>
10                  ......
11    </module>
12
13    <module>
14      <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">
15                          prefix:bgp-peer</type>
16      <name>example-bgp-peer2</name>
17      <host>192.168.100.5</host>
18      <holdtimer>180</holdtimer>
19      <peer-role>ebgp</peer-role>
20      <remote-as>20</remote-as>
21                  ......
22    </module>
```

Figure 5.17: Network 2 -ODL configuration for multi-domain network

Network try-out

To make sure that multi-domain network is configured properly, firstly we did the traceroute from N1 to N8:

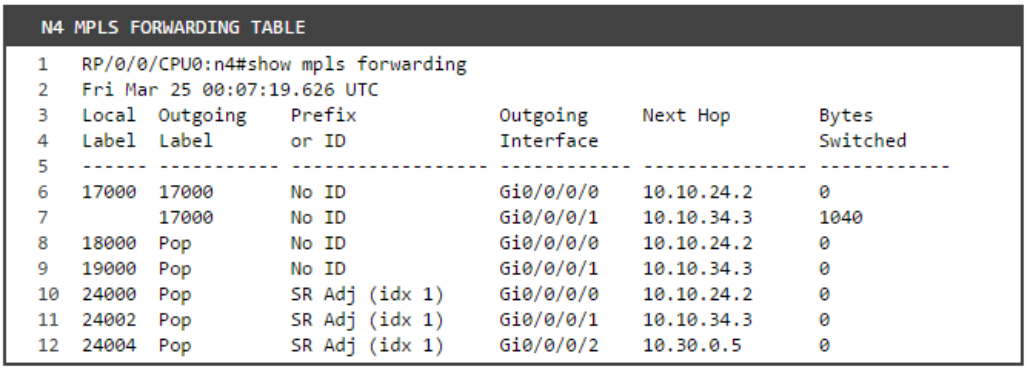


traceroute 10.20.8.8

```
1   RP/0/0/CPU0:n1#traceroute  10.20.8.8
2   Wed Mar 30 13:36:17.668 UTC
3
4   Type escape sequence to abort.
5   Tracing the route to 10.20.8.8
6
7    1  10.10.12.2 [MPLS: Label 20000 Exp 0] 0 msec  0 msec  0 msec
8    2  10.10.24.4 9 msec  0 msec  0 msec
9    3  10.30.0.5 0 msec  9 msec  0 msec
10   4  10.20.56.6 [MPLS: Label 20000 Exp 0] 209 msec  19 msec  19 msec
11   5  10.20.68.8 69 msec  *  9 msec
```

Figure 5.18: Network 2 - traceroute from N1 to N8

Then, we checked MPLS forwarding table at N4 to ensure that we have the label for external link (figure below). One can see that there is adjacency label 24004 towards external interface - 10.30.0.5.



N4 MPLS FORWARDING TABLE

```
1   RP/0/0/CPU0:n4#show mpls forwarding
2   Fri Mar 25 00:07:19.626 UTC
3   Local  Outgoing    Prefix            Outgoing     Next Hop        Bytes
4   Label  Label       or ID             Interface                    Switched
5   ------ ----------- ----------------- ------------ --------------- ------------
6   17000  17000       No ID             Gi0/0/0/0    10.10.24.2      0
7          17000       No ID             Gi0/0/0/1    10.10.34.3      1040
8   18000  Pop         No ID             Gi0/0/0/0    10.10.24.2      0
9   19000  Pop         No ID             Gi0/0/0/1    10.10.34.3      0
10  24000  Pop         SR Adj (idx 1)    Gi0/0/0/0    10.10.24.2      0
11  24002  Pop         SR Adj (idx 1)    Gi0/0/0/1    10.10.34.3      0
12  24004  Pop         SR Adj (idx 1)    Gi0/0/0/2    10.30.0.5       0
```

Figure 5.19: Network 2 - MPLS forwarding table - router N4

Lastly we checked IP routes at node N4. This show command prints out all the routes that a router has in its forwarding table.

N4 IP ROUTE

```
RP/0/0/CPU0:n4#show ip route
Fri Mar 25 00:08:08.553 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is not set

S    10.10.0.0/16 is directly connected, 3w0d, Null0
i L2 10.10.1.1/32 [115/20] via 10.10.24.2, 19:00:19, GigabitEthernet0/0/0/0
                  [115/20] via 10.10.34.3, 19:00:19, GigabitEthernet0/0/0/1
i L2 10.10.2.2/32 [115/10] via 10.10.24.2, 19:28:21, GigabitEthernet0/0/0/0
i L2 10.10.3.3/32 [115/10] via 10.10.34.3, 19:42:32, GigabitEthernet0/0/0/1
L    10.10.4.4/32 is directly connected, 3w0d, Loopback0
i L2 10.10.12.0/24 [115/20] via 10.10.24.2, 1w0d, GigabitEthernet0/0/0/0
i L2 10.10.13.0/24 [115/20] via 10.10.34.3, 3w0d, GigabitEthernet0/0/0/1
C    10.10.24.0/24 is directly connected, 3w0d, GigabitEthernet0/0/0/0
L    10.10.24.4/32 is directly connected, 3w0d, GigabitEthernet0/0/0/0
C    10.10.34.0/24 is directly connected, 3w0d, GigabitEthernet0/0/0/1
L    10.10.34.4/32 is directly connected, 3w0d, GigabitEthernet0/0/0/1
i L2 10.10.100.0/24 [115/20] via 10.10.24.2, 22:30:12, GigabitEthernet0/0/0/0
                    [115/20] via 10.10.34.3, 22:30:12, GigabitEthernet0/0/0/1
B    10.20.0.0/16 [20/0] via 10.30.0.5, 1w0d
B    10.20.8.8/32 [20/20] via 10.30.0.5, 1d01h
C    10.30.0.0/16 is directly connected, 3w0d, GigabitEthernet0/0/0/2
L    10.30.0.4/32 is directly connected, 3w0d, GigabitEthernet0/0/0/2
L    127.0.0.0/8 [0/0] via 0.0.0.0, 3w0d
C    192.168.100.0/24 is directly connected, 3w0d, MgmtEth0/0/CPU0/0
L    192.168.100.4/32 is directly connected, 3w0d, MgmtEth0/0/CPU0/0
```

Figure 5.20: Network 2 - N4 IP route

5.3.3 Network 3 - Performance Analysis test-bed

To test Traffic Engineering performance, we constructed a network that has multiple disjoint paths between server and client. This network, as could be seen from the figure below, consists of six routers R1-R6, one client and three servers.

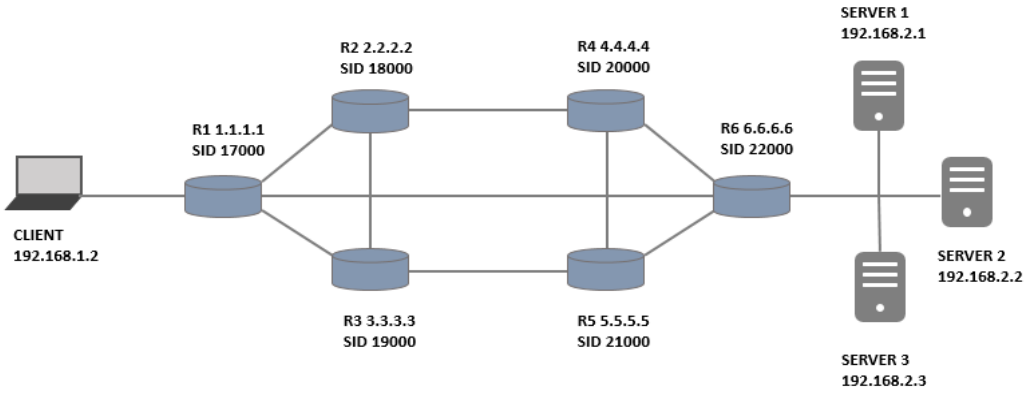


Figure 5.21: Network 3 - topology

This is single domain network and BGP speaker is R4. The router configuration will be presented below.

Router Configuration

In this section, we will observe configuration of router R1 - interface, IS-IS and MPLS configuration. Additionally BGP configuration of router R4 will be presented. Some details (commands meaning) will be skipped since they were provided in previous sections. Here we will put more focus on network construction.

In this network, each router is connected to three other routers. Router R1 is connected to the subnet 192.168.1.0/24 where we have the client with IP 192.168.1.2. On the router R6 we have subnet attached 192.168.2.0/24.

Configuration of R1 interfaces is presented below. R1 has Loopback, management and four GigabitEthernet interfaces. The Loopback address is 1.1.1.1 and management address is 192.168.100.1. Management interface is used for communication to OpenDaylight. Other three interfaces (to other routers) are 10.10.12.1, 10.10.13.1 and 10.10.16.1.

INTERFACES configuration

```
!-----INTERFACES---------------------------------
hostname r1
ipv4 unnumbered mpls traffic-eng Loopback0
interface Loopback0
 ipv4 address 1.1.1.1 255.255.255.255
!
interface MgmtEth0/0/CPU0/0
 ipv4 address 192.168.100.1 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 ipv4 address 10.10.12.1 255.255.255.0
!
interface GigabitEthernet0/0/0/1
 ipv4 address 10.10.13.1 255.255.255.0
!
interface GigabitEthernet0/0/0/2
 ipv4 address 10.10.16.1 255.255.255.0
!
interface GigabitEthernet0/0/0/3
 ipv4 address 192.168.1.1 255.255.255.0
!
route-policy PASS
  pass
end-policy
!
```

Figure 5.22: Network 3 - interface configuration R1

IS-IS configuration

```
!-----ISIS---------------------------------------
router isis 1
 is-type level-2-only
 net 49.0000.0000.0000.0001.00
 log adjacency changes
 address-family ipv4 unicast
  metric-style wide
  ispf
  mpls traffic-eng level-2-only
  mpls traffic-eng router-id Loopback0
  redistribute connected
  segment-routing mpls sr-prefer
 !
 interface Loopback0
 passive
  address-family ipv4 unicast
   prefix-sid absolute 17000
  !
 !
 interface GigabitEthernet0/0/0/0
  point-to-point
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/0/0/1
  point-to-point
  address-family ipv4 unicast
  !
 !
 interface GigabitEthernet0/0/0/2
  point-to-point
  address-family ipv4 unicast
  !
 !
!
```

Figure 5.23: Network - IS-IS configuration R1

In IS-IS section we specified the interfaces that will be announced in IS-IS messages. Loopback interface is announced, but passive (no packets are sent through it). IS-IS runs on GigabitEthernet interfaces toward the routers R2, R3 and R4. Segment Routing preference is included in MPLS traffic engineering section. The node SID is 17000.

MPLS configuration

```
!-----MPLS---------------------------------------
mpls oam
!
mpls traffic-eng
 interface GigabitEthernet0/0/0/0
 !
 interface GigabitEthernet0/0/0/1
 !
 interface GigabitEthernet0/0/0/2
 !
pce
 peer source ipv4 192.168.100.1
 peer ipv4 192.168.100.100
 !
 segment-routing
 stateful-client
  instantiation
  delegation
  !
 speaker-entity-id 1.1.1.1
 !
 auto-tunnel pcc
  tunnel-id min 1 max 99
  !
 reoptimize timers delay installation 0
 !
commit
end
```

Figure 5.24: Network 3 - MPLS configuration R1

In MPLS section we enabled traffic engineering on GigabitEthernet interfaces. Also, PCE source and peer IPv4 address are specified. Segment Routing capabilities are included, so OpenDaylight can push the Segment Routing tunnels to the router.

MPLS, IS-IS and interface sections are identical for each router in the network except the IP address and SIDs values. In the table below, one can find the

values we used for configuring each router: node name, Loopback address, Management interface IP address, GigabitEthernet interfaces and Node-SIDs

Router ID	Loopback Address	Management Interface	GigabitEternet Interfaces	Node SID
R1	1.1.1.1	192.168.100.1	10.10.12.1 10.10.13.1 10.10.16.1 192.168.1.1	17000
R2	2.2.2.2	192.168.100.2	10.10.12.2 10.10.23.2 10.10.24.2	18000
R3	3.3.3.3	192.168.100.3	10.10.13.3 10.10.23.3 10.10.35.3	19000
R4	4.4.4.4	192.168.100.4	10.10.24.4 10.10.46.4 10.10.45.4	20000
R5	5.5.5.5	192.168.100.5	10.10.35.5 10.10.45.5 10.10.56.5	21000
R6	6.6.6.6	192.168.100.6	10.10.16.6 10.10.46.6 10.10.56.6 192.168.2.1	22000

Table 5.3: Network 3 - interfaces and SIDs

Router R4 is BGP speaker and it peers OpenDaylight. They communicate by BGP-LS protocol. R4 periodically delivers link-state information to the controller. The BGP configuration is presented below.

BGP configuration

```
!-----BGP---------------------------------------
router bgp 1
 bgp router-id 4.4.4.4
 address-family link-state link-state
 bgp log neighbor changes detail
 !
 neighbor 192.168.100.100
  remote-as 1
  update-source MgmtEth0/0/CPU0/0
  session-open-mode passive-only
  address-family link-state link-state
   route-policy PASS in
   route-policy PASS out
  !
 !
!
```

Figure 5.25: Network 3 - BGP configuration R4

OpenDaylight Configuration

OpenDaylight configuration is the same as in the Network 1 (chapter 5.3.1).

Network try-out

To test the network configuration, again, we trace the route from the client to the server and we check MPLS forwarding table at node R1.

traceroute from client to server 1

```
1  RP/0/0/CPU0:r1#traceroute 192.168.2.11 source 192.168.1.1
2  Thu Mar  3 12:47:57.441 UTC
3
4  Type escape sequence to abort.
5  Tracing the route to 192.168.2.11
6
7   1  10.10.16.6 0 msec  0 msec  0 msec
8   2  192.168.2.11 0 msec  0 msec  0 msec
```

Figure 5.26: Network 3 - traceroute from client to server 1

MPLS forwarding table - router R1

```
1   RP/0/0/CPU0:r1#sho mpls forwarding
2   Thu Mar  3 12:49:38.635 UTC
3   Local  Outgoing    Prefix            Outgoing     Next Hop        Bytes
4   Label  Label       or ID             Interface                    Switched
5   ------ ----------- ----------------- ------------ --------------- ------------
6   18000  Pop         No ID             Gi0/0/0/0    10.10.12.2      0
7   19000  Pop         No ID             Gi0/0/0/1    10.10.13.3      0
8   20000  20000       No ID             Gi0/0/0/0    10.10.12.2      0
9          20000       No ID             Gi0/0/0/2    10.10.16.6      1232
10  21000  21000       No ID             Gi0/0/0/1    10.10.13.3      0
11         21000       No ID             Gi0/0/0/2    10.10.16.6      0
12  22000  Pop         No ID             Gi0/0/0/2    10.10.16.6      0
13  24000  Pop         SR Adj (idx 1)    Gi0/0/0/0    10.10.12.2      0
14  24002  Pop         SR Adj (idx 1)    Gi0/0/0/1    10.10.13.3      0
15  24004  Pop         SR Adj (idx 1)    Gi0/0/0/2    10.10.16.6      0
```

Figure 5.27: Network 3 - MPLS forwarding table

Chapter 6. Results

The implementation that was presented in previous chapter was used to run different test cases and to examine various aspects of Segment Routing in SDN environment. Firstly, we tested ECMP capabilities of Segment Routing tunnels. It will be shown how by implementing Segment Routing LSP database can be significantly decreased. We will prove resource utilization benefits of application engineered Segment Routing. Lastly, we can see how SDN-SR can enable efficient and easy traffic engineering in multi-domain network.

6.1 ECMP Segment Routing

Equal Cost Multipath is a strategy where the traffic can be forwarded from a source to the destination by using multiple best paths (equal cost). ECMP balances the overall traffic load and guarantees the efficient network utilization. Segment Routing adopts ECMP strategy by defining a Node-SID to be ECMP-aware shortest path to the destination. By constructing a tunnel just using a single label, one will have ECMP-aware tunnel through destination.

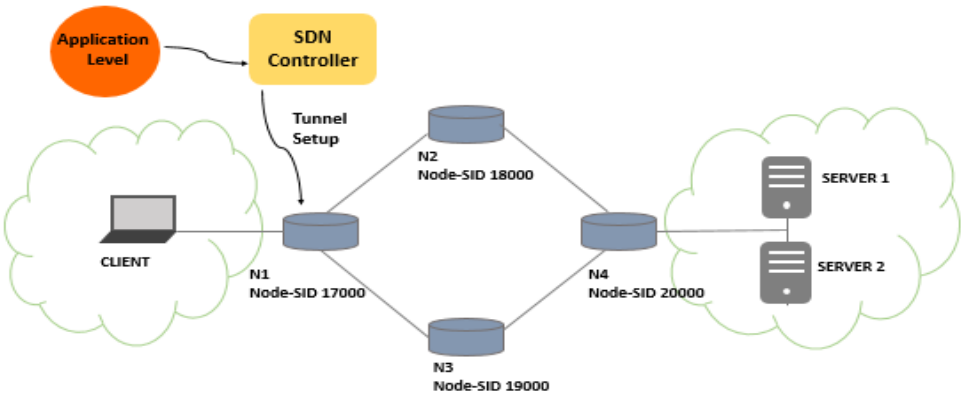


Figure 6.1: ECMP testing

In the figure a simple Segment Routing network is presented. One can create a Segment Routing tunnel from a network edge to another by simply pushing a single label on the packet header. The tunnel (label) is pushed from application level through controller to the network:

ERO pushed from application level

```
<ero>
   <subobject>
     <loose>false</loose>
     <sid-type xmlns="urn:opendaylight:params:xml:ns:yang:pcep:segment:routing">
                                        ipv4-node-id</sid-type>
     <m-flag xmlns="urn:opendaylight:params:xml:ns:yang:pcep:segment:routing">
                                        true</m-flag>
     <sid xmlns="urn:opendaylight:params:xml:ns:yang:pcep:segment:routing">20000</sid>
   </subobject>
</ero>
```

Figure 6.2: ERO pushed from application to the network

Simply tracing a route from a client to different destinations, one can see that traffic takes different paths to through the core network:

trace route from the client to server1 and server2

```
RP/0/0/CPU0:n1#traceroute 10.20.100.11 source 10.10.100.1
Wed Mar 30 16:40:54.118 UTC

Type escape sequence to abort.
Tracing the route to 10.20.100.11

 1  10.10.13.3 [MPLS: Label 20000 Exp 0] 0 msec  0 msec  9 msec
 2  10.10.34.4 0 msec  9 msec  0 msec
 3  10.20.100.11 9 msec  9 msec  19 msec

RP/0/0/CPU0:n1#traceroute 10.20.100.22 source 10.10.100.1
Wed Mar 30 16:41:05.048 UTC

Type escape sequence to abort.
Tracing the route to 10.20.100.22

 1  10.10.12.2 [MPLS: Label 20000 Exp 0] 0 msec  9 msec  0 msec
 2  10.10.24.4 0 msec  9 msec  9 msec
 3  10.20.100.22 0 msec  9 msec  0 msec
```

Figure 6.3: Traceroute - ECMP testing

Here we proved the ECMP property of Node-SID. Assigning just one label one can get an ECMP tunnel between two nodes e.g. network edges.

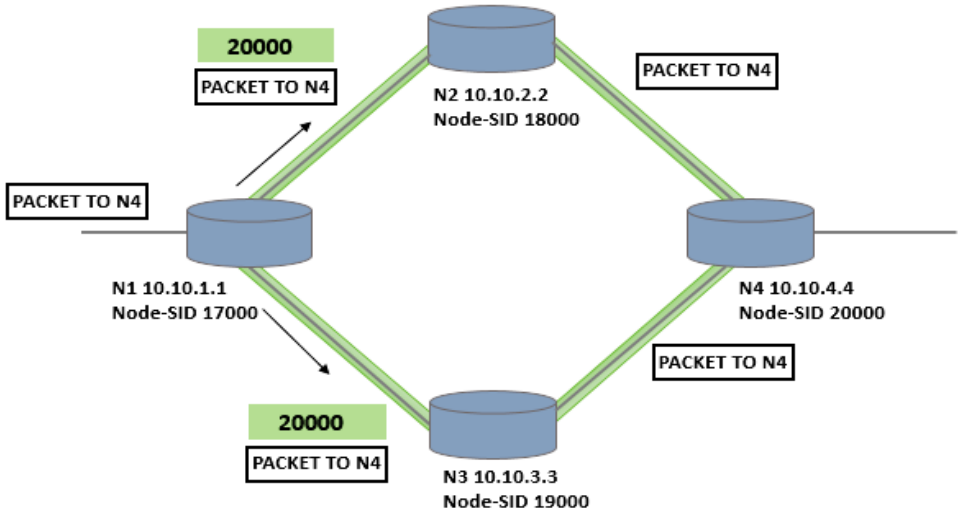
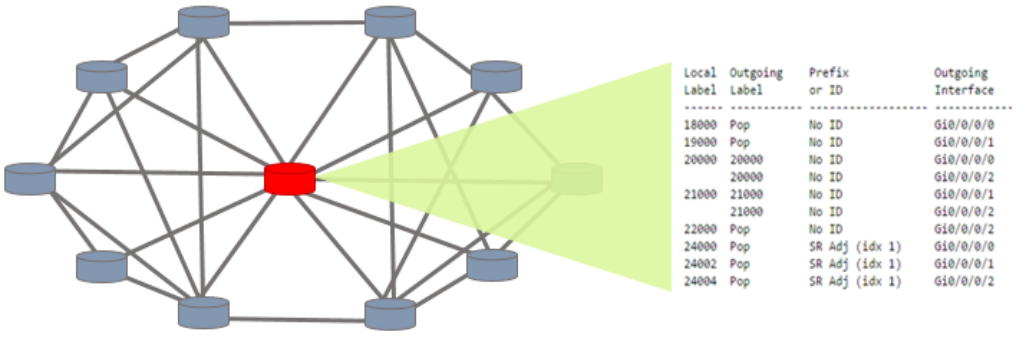


Figure 6.4: ECMP paths

This property can have a huge impact on LSP database size, especially in big backbone networks where a core node should keep track of thousands of tunnels. Due to the hardware limitation, we could not get practical result of database scalability - one should employ big number of nodes. In theory, one can compare number of state core MPLS node and in Segment Routing node. Let's assume a full mesh network as on the picture below. In MPLS full mesh a core node has approximately N^2 entries while in Segment Routing a number of entries tends to be equal to number of nodes in the network plus number of adjacencies. Moreover, the forwarding table in Segment Routing tends to be constant - once SIDs are setup there is no need for any further changes.



Local Label	Outgoing Label	Prefix or ID	Outgoing Interface
18000	Pop	No ID	Gi0/0/0/0
19000	Pop	No ID	Gi0/0/0/1
20000	20000	No ID	Gi0/0/0/0
	20000	No ID	Gi0/0/0/2
21000	21000	No ID	Gi0/0/0/1
	21000	No ID	Gi0/0/0/2
22000	Pop	No ID	Gi0/0/0/2
24000	Pop	SR Adj (idx 1)	Gi0/0/0/0
24002	Pop	SR Adj (idx 1)	Gi0/0/0/1
24004	Pop	SR Adj (idx 1)	Gi0/0/0/2

Figure 6.5: Network scalability

The following graph shows theoretical results of number of states in MPLS core node and Segment Routing node:

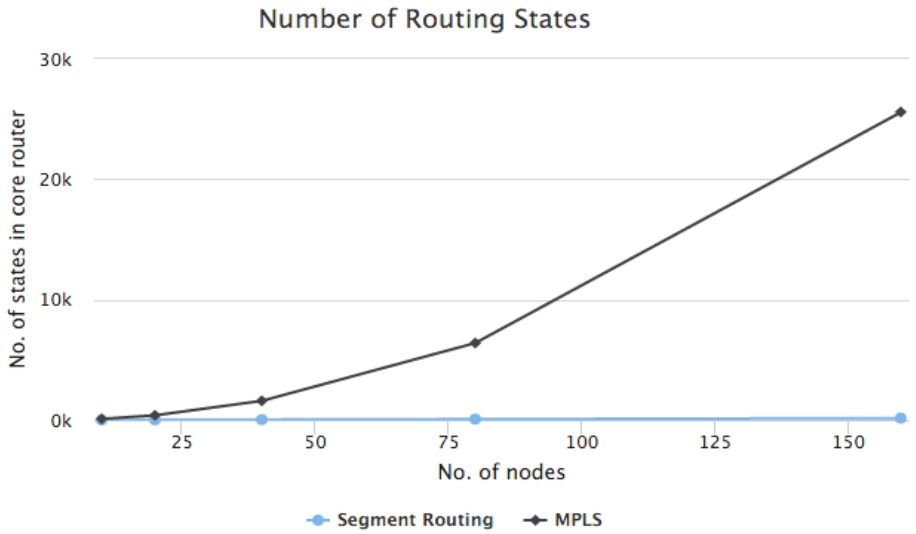


Figure 6.6: MPLS vs Segment Routing states number

The number of entries in a Segment Routing node is significantly lower than in MPLS. Segment Routing introduces the constant label in a network domain. Tunnels that are built within a network reuse the same set of labels so there is no need to add a new FIB entry when a new tunnel is added. While in MPLS for each tunnel a new FIB entry is necessary. This can cause huge scale problems in populated core networks, where some nodes have to keep a track of thousands of nodes.

6.2 Performance Analysis

This sub-chapter shows results of the tests we run to evaluate Segment Routing network performance. The Jperf network performance monitor tool was used to test and measure network utilization when multiple flows are present. Firstly we test network behavior without Segment Routing tunnels. After gathering results, we pushed tunnels from OpenDaylight and measure network performance again.

The scenario is following:

- The network consists of 6 routers, 1 client and 3 servers (figure below)
- There is a direct path between client and servers
- There are two disjoint (multi-hop) paths between client and server
- The client wants to send traffic to each server at the same time

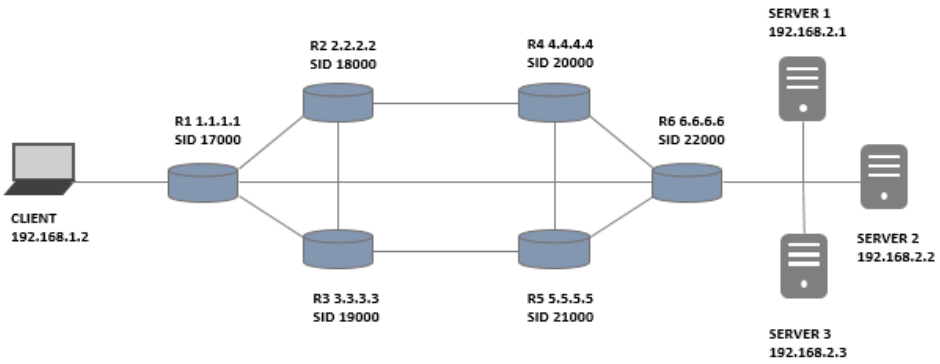


Figure 6.7: Performance analysis scenario

6.2.1 Test without Segment Routing

Without traffic engineering tunnels all three flows from the client to the server will naturally take the shortest path through the network (R1-R6). The capacity of a link will be shared among three flows which will have the direct impact on the flow rate and there on QoS.

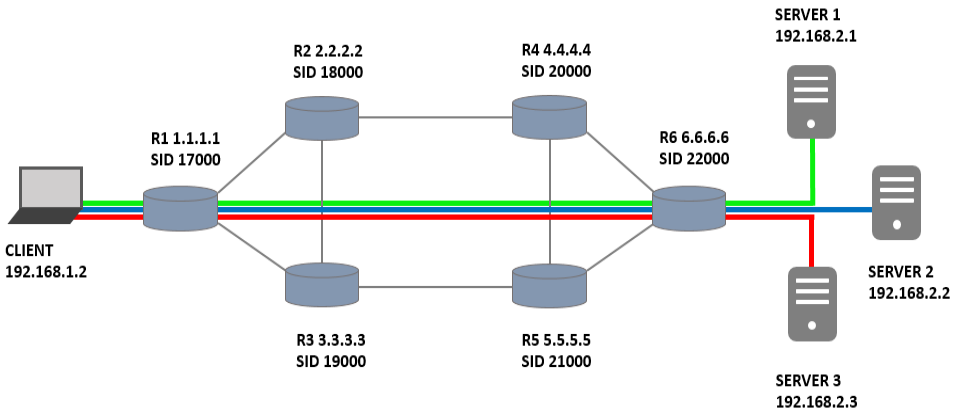


Figure 6.8: Performance analysis – test without Segment Routing

The test steps are:

- Firstly client connects to the server 1 (green line)
- After 50 sec the client connects to server 2 (blue line)
- After 50 sec the client connects to the server 3 (red line)

The links rate are limited to 500 Kbit/s due to Cisco XRv limitation.

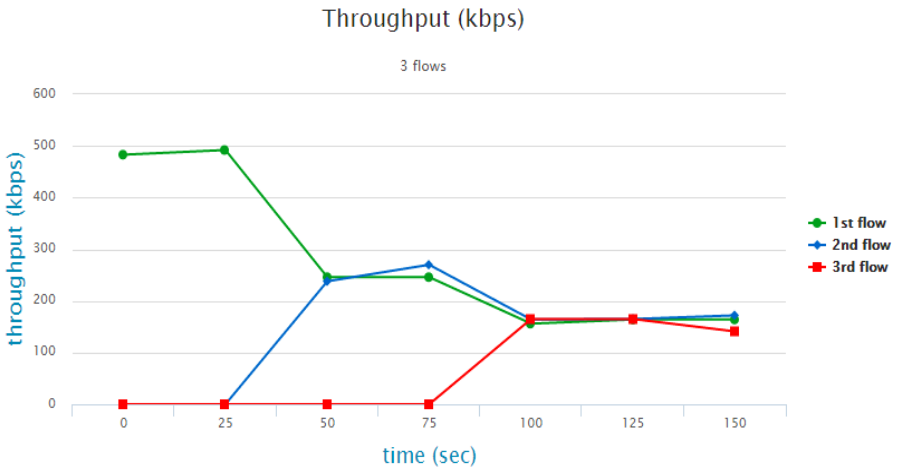


Figure 6.9: Results of performance analysis – test without Segment Routing

As can be seen from the graph, the first connection gets the full capacity initially. After 50s, the second connection is setup and two flows utilize half of the capacity each. Finally, when the third connection comes up the flows get one third of total capacity.

6.2.2 Test with Segment Routing

In the same scenario, Segment Routing can be used to improve overall throughput. Beside the shortest path between R1 and R6, there are two disjoint paths between those two nodes that can be used to convey client's traffic. To get the maximum capacity for all three flows we added two Segment Routing tunnels to the network:

- 1st tunnel: R1-R2-R4-R6 identified by segments 18000-22000
- 2nd tunnel: R1-R3-R5-R6 identified by segments 19000-22000

With two tunnels and with the natural shortest path between R1 and R6, all of three flows can exploit the full link bandwidth towards destination.

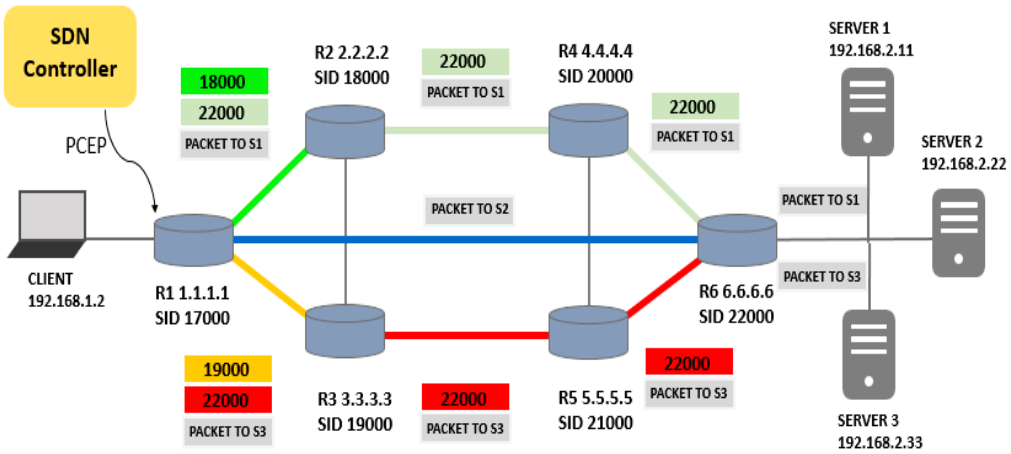


Figure 6.10: Performance analysis – test with Segment Routing

By measuring the traffic rate on the server side we got the following results:

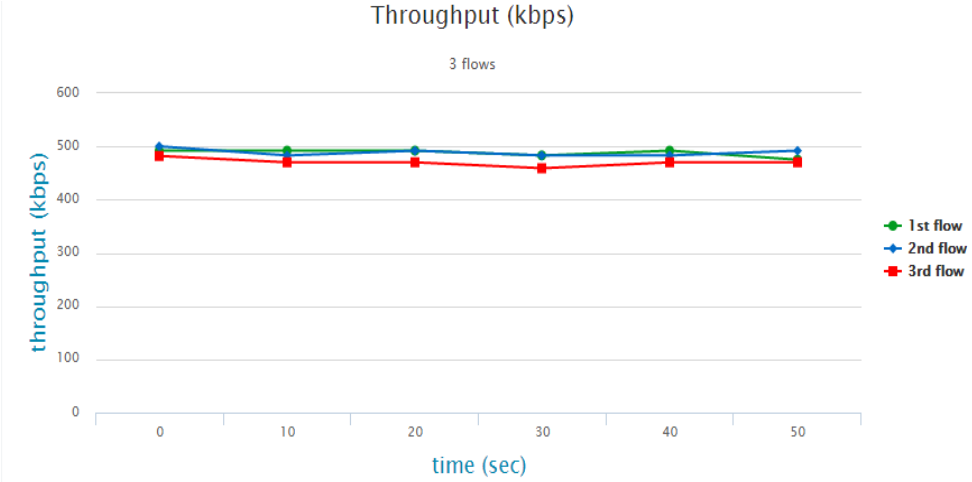


Figure 6.11: Results of performance analysis – test with Segment Routing

We can clearly see the benefit of Segment Routing in the above scenario. It utilizes network and satisfies all three flows with maximum link speed, instead of sharing the link on the shortest path. And all these can be initiated from

higher-level application with SDN Controller, which has a global view and control over the network.

Segment Routing paths can be created dynamically. It enables network providers to quickly address new business requirements in agile fashion. There is no need to modify configurations locally on routers. It can be simply done by sending a RESTful request from the application network through controller and tunnels. And when there are no more needs for these new paths, they can as easily removed as they were created.

6.3 Multi-domain Segment Routing

Lastly, we examined Segment Routing performance in multi-domain network. We observed a situation where the traffic crosses multiple domains toward destination. Domains belong to different ISPs. Multi-domain network has a single control entity – SDN controller that has global visibility of both domains and that is capable to monitor and control the overall network state.

In reality, an end-to-end path that crosses multiple domains might experience different traffic policies in different domains. A service provider's policy presents the rule that must be taking into account while provisioning the service to the end customers. In some cases policies can be relaxed and traffic can take the best path calculated by IGP. However, sometimes an ISP might set up strict policies and traffic must take a strict while crossing the domain.

This test case has the goal to demonstrate how one can exploit basic Segment Routing properties to design a path that respects ISPs' policies as well as traffic requirements. We will use ECMP and application engineering features to show how end-to-end path can be engineered in the environment where policies change in different domains.

The multi-domain network we implemented consists of two domains, 4 nodes each. An ODL controller has a full visibility of network topology and status. In each domain there is a BGP speaker that delivers link-state domains to the

controller. Also, BGP protocol runs on interconnection link and transfer network reachability information. A client that is on one side of the network communicates with two servers that are on the other part of the multi-domain network.

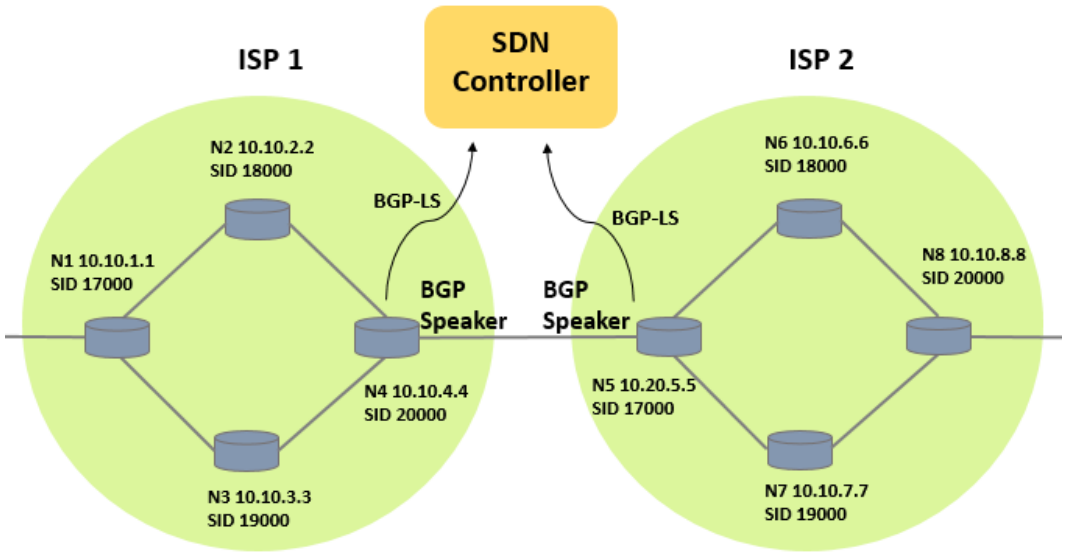


Figure 6.12: Multi-domain network

Let's observe the end to end path for highly sensitive traffic such as voice or video. For high priority traffic that crosses multiple domains it is crucial to ensure that in each domain the user SLA is satisfied. Sensitive traffic must take an optimum path in each domain otherwise QoS will degrade. In this case we show how SDN controller can set up deterministic path across multiple domains. By using Adjacency-SIDs an end-to-end path can be designed to satisfy user's criteria along all way from the source to the destination.

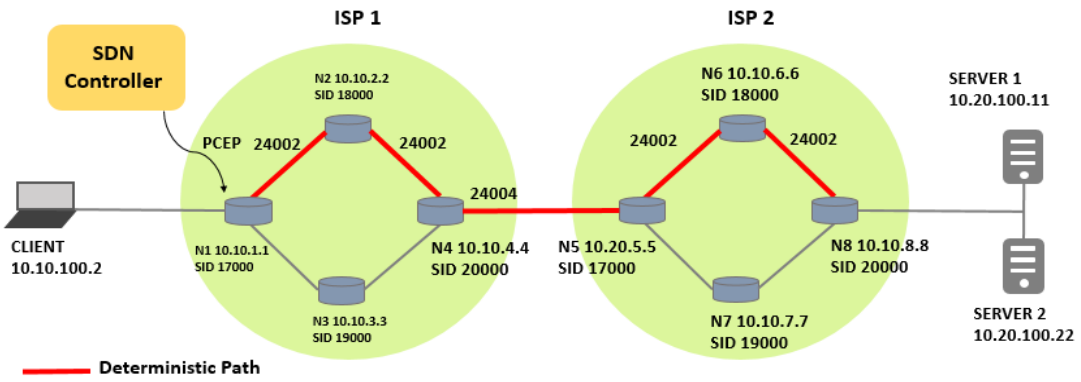


Figure 6.13: Multi-domain example 1

traceroute from client to server 1

```
1   RP/0/0/CPU0:n1#traceroute 10.20.100.11 source 10.10.100.2
2   Wed Mar 30 14:01:37.854 UTC
3
4   Type escape sequence to abort.
5   Tracing the route to 10.20.100.11
6
7    1  10.10.100.1 [MPLS: Labels 24002/24002/24004/24002/24002 Exp 0] 9 msec  19 msec  0 msec
8    2  10.10.12.2 [MPLS: Labels 24002/24004/24002/24002 Exp 0] 19 msec  19 msec  0 msec
9    3  10.10.24.4 [MPLS: Labels 24004/24002/24003 Exp 0] 9 msec  19 msec  0 msec
10   4  10.30.0.5 [MPLS: Labels 24002/24003 Exp 0] 9 msec  9 msec  0 msec
11   5  10.20.56.6 [MPLS: Label 24003 Exp 0] 0 msec  0 msec  49 msec
12   6  10.20.68.8 0 msec  0 msec  0 msec
13   7  10.20.100.11 9 msec  19 msec  19 msec
```

Figure 6.14: Traceroute for multi-domain example 1

However, it might happen there are multiple optimal paths across the domain. For example, the network is dense and the operator wants to use ECMP to distribute traffic through the network. In that case Prefix-SIDs can be used and traffic load will be equally distributed in each domain among the available paths.

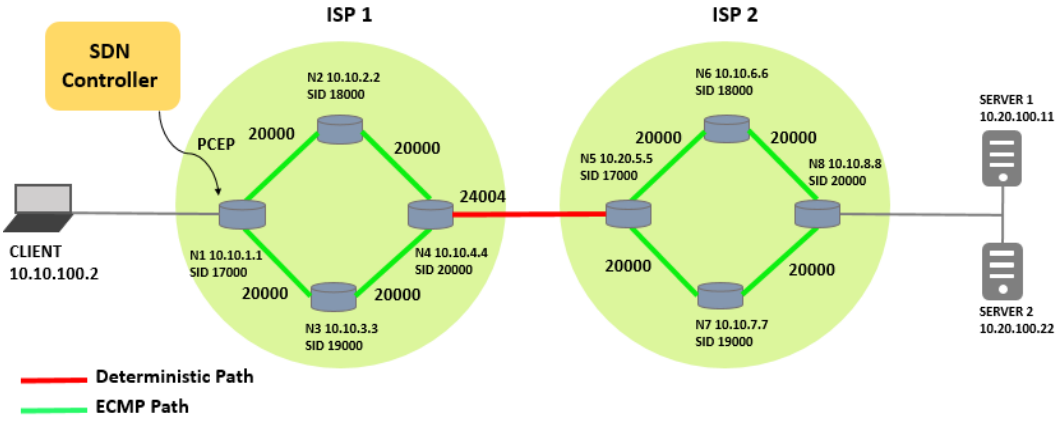


Figure 6.15: Multi-domain example 2

traceroute from client to server 1 and to server 2

```
RP/0/0/CPU0:n1#traceroute 10.20.100.11 source 10.10.100.1
Wed Mar 30 14:47:19.426 UTC

Type escape sequence to abort.
Tracing the route to 10.20.100.11

 1  10.10.13.3 [MPLS: Labels 20000/24004/20000 Exp 0] 0 msec  0 msec  19 msec
 2  10.10.34.4 [MPLS: Labels 24004/20000 Exp 0] 9 msec  9 msec  19 msec
 3  10.30.0.5 [MPLS: Label 20000 Exp 0] 19 msec  9 msec  9 msec
 4  10.20.57.7 [MPLS: Label 20000 Exp 0] 9 msec  9 msec  9 msec
 5  10.20.78.8 9 msec  9 msec  19 msec
 6  10.20.100.11 19 msec  9 msec  9 msec

RP/0/0/CPU0:n1#traceroute 10.20.100.22 source 10.10.100.1
Wed Mar 30 14:48:25.911 UTC

Type escape sequence to abort.
Tracing the route to 10.20.100.22

 1  10.10.12.2 [MPLS: Labels 20000/24004/20000 Exp 0] 9 msec  0 msec  29 msec
 2  10.10.24.4 [MPLS: Labels 24004/20000 Exp 0] 9 msec  19 msec  9 msec
 3  10.30.0.5 [MPLS: Label 20000 Exp 0] 29 msec  9 msec  29 msec
 4  10.20.56.6 [MPLS: Label 20000 Exp 0] 19 msec  9 msec  9 msec
 5  10.20.68.8 9 msec  0 msec  9 msec
 6  10.20.100.22 9 msec  9 msec  0 msec
```

Figure 6.16: Traceroute for multi-domain example 2

As we mentioned above, a single end-to-end path might experience different policies in different domains. Also, different applications coming from the same source might have different requirements and can be handle in different way across the multi-domain network.

The SDN controller can provision the path that matches different policies and requirements by combining adjacency and prefix SIDs. In the scenario the first ISP has relaxed policies and multiple disjoint paths towards destination. In the second domain, ISP2 has strict policies and specific paths for different kind of traffic.

The client wants to run different kind of applications on two servers. The first flow from the client to the server 1, crosses the first domain with ECMP path, then it takes deterministic path in the second:

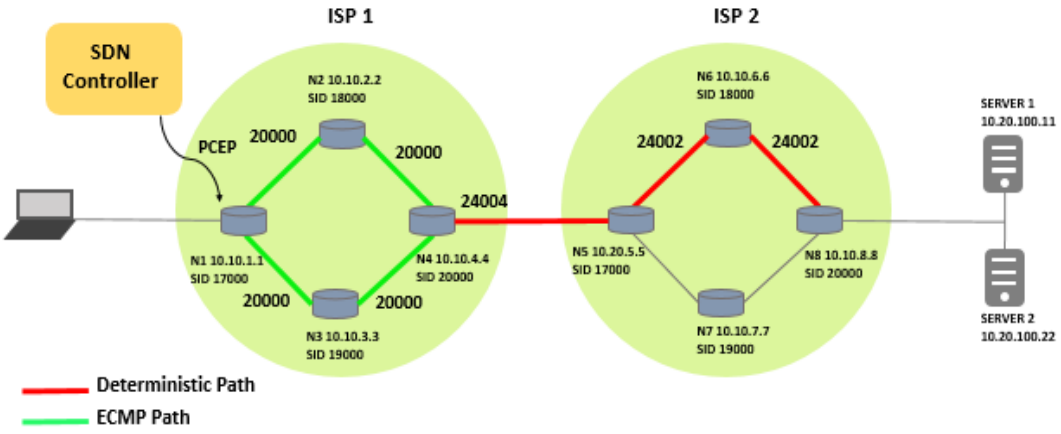


Figure 6.17a: Multi-domain example 3a

Similarly the other flow takes ECMP-aware route in the first domain and on the second it gets a deterministic paths:

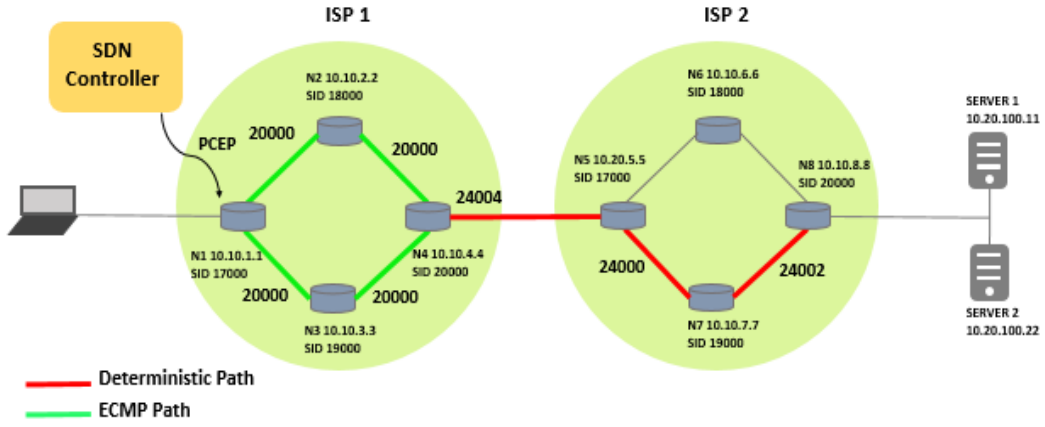


Figure 6.17b: Multi-domain example 3b

In the example above we wanted to demonstrate how different combination of SIDs can match different policies and requirements while forwarding end-to-end traffic. SDN controller that has global view and knowledge of network state and policies can provision a path that can satisfy both operator's and user's criteria. The benefit is mutual.

6.4 Network Limitations

During the testing phase we experienced some limitation on Cisco XRv and OpenDaylight controller that are worth to mention:

- The free version of Cisco XRv has limited throughput of 2 Mbit/s. As it was mentioned before, in order to test the network performance properly, in chapter 6.2, we limited all the links to the rate 500 Kbit/s
- ODL Controller does not support dynamic addition of tunnels into the routing table, because this functionality still is not part of the standard. This means that, even though we push the tunnels from RESTful client they are not dynamically setup - they are visible but not ready to use. To address this problem we had to add a static route and add the tunnel manually in the routing table. After this changes, the ingress node directs any traffic destined to towards specified address (in this case 10.20.100.11/32) into the Segment Routing path (tunnel-te8)

```
router static
1  router static
2   address-family ipv4 unicast
3    10.20.100.11/32 tunnel-te8
4   !
5  !
```

Figure 6.18: Router static

- Cisco XRv does not support configuration of BGP SIDs. As it was discussed previously, we were not able to configure BGP Prefix-SID on external link. Under IS-IS command there is possibility to set up Prefix-SID, and adjacency SIDs are configured automatically. In BGP configuration there is no such a command, so we had to the do the workaround as it was explained in 5.3.2

Chapter 7. Conclusion

This work was setup to investigate the new source routing paradigm called Segment Routing. Initially, we discussed Segment Routing: general overview, why it is emerging, its underlying technologies and its benefits in context of Software Defined Networking.

For analysis we implemented Segment Routing in three virtual networks. The implementation considered network creation in Virtual Machines, setup of SDN controller and its connection to the network, creation of Segment Routing tunnels inside the network using SDN controller.

We have tested characteristic of Segment Routing tunnels in different environments. Firstly we showed that with a single label an ECMP-aware tunnel could be set up. We demonstrated it in our network and we have shown that having the multiple flows in the network the overall load will be distributed among available resources. Moreover, we showed that Segment Routing reduces forwarding table that can lead to simplified hardware, reduced processing time and cost.

In the second test-case we tested network performance with and without Segment Routing. Test results showed that Segment Routing tunnels greatly increase network performance. Segment Routing enabled traffic flows to use network links on their full capacity. Which was a great improvement over the network performance without Segment Routing.

Lastly, we demonstrated Segment Routing capability to design multi-domain end-to-end paths. We have shown that by using SDN controller an end-to-end path can be constructed to satisfy both customer and providers criteria. The controller that has global view on multi-domain network can take all the policies into account and provide the customized path by combining different Segment Identifiers.

Segment Routing and SDN promote simplified control and traffic engineering. The SDN introduced centralized network control, global visibility and management. Segment Routing follows the SDN goals by further simplifying forwarding actions within a network.

Network providers and equipment manufacturers are embracing SDN paradigm. All these benefits of Segment Routing can be easily orchestrated from high-level application with help of SDN Controller. Segment Routing can become a powerful tool, which can be used by SDN Controller to improve performance of existing network infrastructure. Further, it enables network providers to add new value to their business. Being able to dynamically address new requirements from customers is a great advantage for network providers.

Segment Routing can highly enhance network performance. Segment Routing enables efficient and smart traffic tunneling which maximizes resource utilization and brings cost benefits to service providers. Employing Segment Routing in the network they can easily manage available resources and bring the network performance on higher level.

Segment Routing does not require massive upgrade in the network neither big investments for service providers. It can be employed step by step and it can cooperate with existing IP/MPLS technology. Segment Routing eliminates need for signaling protocols such as RSVP-TE and LDP that has the huge impact on network simplicity.

SDN and Segment Routing reduce network administration complexity. Global view and centralized control enable easy network troubleshooting, setup trials and maintenance. Pushing the configuration from central point decreases probability of misconfiguration and exhausting problems solving. This reduces time for service provisioning and cost savings.

Finally, SDN and Segment Routing improve services to the final users. SDN and Segment Routing are capable of provisioning the optimal path depending on certain network state. Moreover SDN controller keeps continuously monitoring on the network status and can optimize the path if it is necessary. Segment Routing brings innovative forwarding logic that can respond to efficient and intelligent traffic engineering in SDN environment.

7.1 Future work

Segment Routing is in very early stage of development. Related scientific work is scarce and documentation provided by standardization entity is keeps changing very often. Even though the technology is in early stages of standardization there is a big hop that industry will adopt in in the future.

In this work we aimed to analyses most of the benefits Segment Routing brings. However, due to technical limitation and support some of use cases are left to be demonstrated. One can continue the work on Fast Rerouting. According to the standard, Segment Routing should provide FRR inherently by using post convergence path as a secondary path. At this moment OpenDaylight does not support protected tunnel setup in dynamic manner. The tunnels pushed from ODL are not protected and that option cannot be changed from the router CLI since all the tunnels are delegated to the controller.

This research was done by using OpenDaylight controller. Being the one of the most popular open source solution, ODL is under constant development and improvement. Recent releases tend to be unstable and the older ones

lack of some functionalities. The work can be repeated using some other SDN controller available on the market such as ONOS, Ryu and etc.

Lastly, the work can be used in the future to be part of orchestrator. Integrating the network with an orchestrator one can easily set up Segment Routing tunnels through GUI by specifying all necessary parameters and requirements.

7.2 Acknowledgements

I would like to express my sincere appreciation to Prof. Guido Maier for his guidance and for giving me opportunity to realize this thesis work in Bonsai laboratory.

Furthermore, I would like to extend my thanks to Navin Kukreja for sharing his knowledge and providing helpful advices and support during this thesis work.

Also, I would like to thank David Giorgidze for his contribution into research process and for his collegial support during the final phase of this work.

Lastly, I would like to express my deepest gratitude to my family for supporting me not only during this thesis work, but also throughout the entire journey for achieving the Master Degree at Politecnico di Milano.

References

[1] Cisco, “Segment Routing: Prepare Your Network for New Business Models White Paper”, retrieved from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/application-engineered-routing/white-paper-c11-734250.html

[2] S. Previdi, C. Filsfils et al. SPRING Problem Statement and Requirements, IETF draft-ietf-spring-problem-statement-07, March 2016

[3] E. Rosen A. Viswanathan et al, Multiprotocol Label Switching Architecture, IETF RFC 3031, January 2001

[4] Lei, Liu. "SDN orchestration for dynamic end-to-end control of data center multi-domain optical networking." Communications, China 12.8 (2015): 10-21.

[5] Shin, Myung-Ki, Ki-Hyuk Nam, and Hyoung-Jun Kim. "Software-defined networking (SDN): A reference architecture and open APIs." ICT Convergence (ICTC), 2012 International Conference on. IEEE, 2012.

[6] Jarschel, Michael, et al. "Interfaces, attributes, and use cases: A compass for SDN." Communications Magazine, IEEE 52.6 (2014): 210-217.

[7] Nunes, Bruno AA, et al. "A survey of software-defined networking: Past, present, and future of programmable networks." Communications Surveys & Tutorials, IEEE 16.3 (2014): 1617-1634.

[8] Agarwal, Sankalp, Murali Kodialam, and T. V. Lakshman. "Traffic engineering in software defined networks." INFOCOM, 2013 Proceedings IEEE. IEEE, 2013.

[9] Lara, Adrian, Anisha Kolasani, and Byrav Ramamurthy. "Network innovation using openflow: A survey." Communications Surveys & Tutorials, IEEE 16.1 (2014): 493-512.

[10] McKeown, Nick, et al. "OpenFlow: enabling innovation in campus networks."ACM SIGCOMM Computer Communication Review 38.2 (2008): 69-74.

[11] Kempf, James, et al. "OpenFlow MPLS and the open source label switched router." Proceedings of the 23rd International Teletraffic Congress. International Teletraffic Congress, 2011.

[12] Lospoto, Gabriele, et al. "Rethinking virtual private networks in the software-defined era." Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on. IEEE, 2015.

[13] Tu, Xiaogang, et al. "Splicing MPLS and OpenFlow Tunnels Based on SDN Paradigm." Cloud Engineering (IC2E), 2014 IEEE International Conference on. IEEE, 2014.

[14] Bidkar, Sarvesh, et al. "Scalable Segment Routing—A New Paradigm for Efficient Service Provider Networking Using Carrier Ethernet Advances."Journal of Optical Communications and Networking 7.5 (2015): 445-460.

[15] Sgambelluri, A., et al. "SDN and PCE implementations for segment routing."Networks and Optical Communications-(NOC), 2015 20th European Conference on. IEEE, 2015.

[16] Sgambelluri, Andrea, et al. "First demonstration of sdn-based segment routing in multi-layer networks." Optical Fiber Communication Conference. Optical Society of America, 2015.

[17] Cai, Deng, Anna Wielosz, and Songbin Wei. "Evolve carrier Ethernet architecture with SDN and segment routing." World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a. IEEE, 2014.

[18] Davoli, Luca, et al. "Traffic Engineering with Segment Routing: SDN-based Architectural Design and Open Source Implementation." Software Defined Networks (EWSDN), 2015 Fourth European Workshop on. IEEE, 2015.

[19] Cisco, Configuring IP Routing, retrieved from: http://www.cisco.com/c/en/us/td/docs/security/asa/asa80/configuration/guide/conf_gd/ip.html

[20] Cisco, Configuring IP Services, retrieved from: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipapp/configuration/12-4t/iap-12-4t-book/iap-ipserv.pdf

[21] http://www.comptechdoc.org/independent/networking/terms/source-routing.html

[22] C. Filsfils, S. Previdi et al. “Segment Routing Architecture”, IETF draft-ietf-spring-segment-routing-07, December 2015

[23] C. Filsfils, S. Previdi et al. “Segment Routing with MPLS data plane”, IETF draft-ietf-spring-segment-routing-mpls-03, February 2016

[24] C. Filsfils, S. Previdi A. Bashandy et al. “Segment Routing Architecture”, IETF draft-filsfils-rtgwg-segment-routing-00, June 2013

[25] C. Filsfils, S. Previdi et al. “Segment Routing interoperability with LDP”, IETF draft-ietf-spring-segment-routing-ldp-interop-00, October 2015

[26] C. Filsfils, P. Francois, “Segment Routing Use Cases”, IETF draft-filsfils-rtgwg-segment-routing-use-cases-02, October 2013

[27] P. Sarkar, H. Gredler et al. “Anycast Segments in MPLS based Segment Routing”, IETF draft-psarkar-spring-mpls-anycast-segments-01, October 2015

[28] Pierre Francois, Clarence Filsfils et al.”Topology Independent Fast Reroute using Segment Routing”, IETF draft-francois-rtgwg-segment-routing-ti-lfa-00, August 2015

[29] Sharon, Oran. "Dissemination of routing information in broadcast networks: OSPF versus IS-IS." Network, IEEE 15.1 (2001): 56-65.

[30] Cisco, “IS-IS Overview and Basic Configuration”, retrieved from: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_isis/configuration/15-mt/irs-15-mt-book/is-is_overview_and_basic_configuration.html#GUID-F1498D99-0DD0-40C8-8978-16003E5B7E8F

[31] C. Filsfils, S. Previdi et al. “IS-IS Extensions for Segment Routing”, IETF draft-ietf-isis-segment-routing-extensions-06, December 2015

[32] Cisco, “BGP Overview”, retrieved from: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/15-mt/irg-15-mt-book/irg-overview.html

[33] Rekhter, Y., T. Li, and S. Hares. "RFC 4271." Internet Engineering Task Force, http://www. rfc-editor. org/rfc/rfc4271. Txt, access on 6 (2014).

[34] K. Patel, S. Previdi, “Segment Routing Prefix SID extensions for BGP”, IETF draft-keyupate-idr-bgp-prefix-sid-05, July 2015

[35] Medved, Jan, et al. "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP." (2016).

[36] http://packetpushers.net/yet-another-new-bgp-nlri-bgp-ls/

[37] H. Gredler, S. Ray et at. “BGP Link-State extensions for Segment Routing”, IETF draft-gredler-idr-bgp-ls-segment-routing-extension-02, October 2014

[38] Ramon Casellas, Raul Munoz et al. PCE Primer, PACE: Next Steps in PAth Computation Element (PCE) Architectures: From Software-Defined Concepts to Standards, Interoperability and Deployment, December 2013

[39] Farrel, Adrian, Jean-Philippe Vasseur, and Jerry Ash. A path computation element (PCE)-based architecture. RFC 4655, August, 2006.

[40] X. Zhang, I. Minei, ”Applicability of a Stateful Path Computation Element (PCE)”, IETF draft-ietf-pce-stateful-pce-app-05, October 2015

[41] S. Sivabalan, J. Medved et al. “PCEP Extensions for Segment Routing”, IETF draft-sivabalan-pce-segment-routing-03.txt , July 2014

[42] Xiao, Xipeng, et al. "Traffic Engineering with MPLS in the Internet." Network, IEEE 14.2 (2000): 28-33.

[43] Rosen, Eric, Arun Viswanathan, and Ross Callon. "Multiprotocol label switching architecture." (2001).

[44] http://www.mplsvpn.info/2015/07/segment-routing-based-mpls-vs-classic.html

[45] http://blog.ipspace.net/2011/11/ldp-igp-synchronization-in-mpls.html

[46] Cisco, ”MPLS Traffic Engineering (TE)--Fast Reroute (FRR) Link and Node Protection”, retrieved from: http://www.cisco.com/c/en/us/td/docs/ios/12_0s/feature/guide/gslnh29.html

[47] Filsfils, Clarence, et al. "Loop-free alternate (LFA) applicability in service provider (SP) networks." Internet Engineering Task Force, RFC 6571 (2012).

[48] Hopps, Christian E. "Analysis of an equal-cost multi-path algorithm." (2000).

[49] Manayya, K. B. "Constrained shortest path first." (2010).

[50]https://conference.apnic.net/data/37/apnic2014-segment routing_santanu_v5_1393404956.pdf

[51] Bakshi, Kapil. "Considerations for software defined networking (SDN): Approaches and use cases." Aerospace Conference, 2013 IEEE. IEEE, 2013.

[52] Diego Kreutz, Fernando M. V. Ramos,Paulo Verissimo,Christian Esteve Rothenberg,Siamak Azodolmolky and Steve Uhlig ,Software-Defined Networking: A Comprehensive Survey, IEEE

[53] https://www.sdxcentral.com/resources/sdn/what-the-definition-of-software-defined-networking-sdn/

[54] http://blogs.cisco.com/sp/segment-routing-impact-on-software-defined-networks

[55] Cisco, Segment Routing for Service Providers, http://www.cisco.com/c/m/en_us/training-events/events-webinars/webinars/segment-routing.html

[56] https://www.sdxcentral.com/resources/sdn/sdn-controllers/opendaylight-controller/

[57] Khattak, Zuhran Khan, Muhammad Awais, and Adnan Iqbal. "Performance evaluation of OpenDaylight SDN controller." Parallel and Distributed Systems (ICPADS), 2014 20th IEEE International Conference on. IEEE, 2014

[58] http://thenewstack.io/sdn-series-part-vi-opendaylight/

[59] Sundararaj, Ananth I., and Peter A. Dinda. "Towards Virtual Networks for Virtual Machine Grid Computing." Virtual machine research and technology symposium. 2004

[60] Ward, Brian. The book of VMware: the complete guide to VMware workstation. Vol. 1. San Francisco: No Starch Press, 2002.

[61] vmware.com/support/ws55/doc/ws_net_configurations_common.html

[62] Cisco, Cisco IOS XRv Router Overview. http://www.cisco.com/en/US/docs/ios_xr_sw/ios_xrv/install_config/b_xrvr_432_chapter_01.html

[63] http://www.opendaylight.org/downloads

[64] https://www.opendaylight.org/installing-opendaylight

[65] wiki.opendaylight.org/view/BGP_LS_PCEP:Lithium_Installation_Guide

[66] http://www.getpostman.com/