Software Defined Networks, FS2022

Luzia Kündig

July 5, 2022

1 Introduction and Concepts

Traditional Networking Architecture is divided into planes, depending on the layer

.	Control Plane	Data Plane	Management Plane
Layer 2	Spanning Tree Overlays (VLANs)	Forward *Ethernet Frames*	
Layer 3	Routing Protocols Overlays (MPLS)	Forward *IP Packets*	

This results in some drawbacks such as

- – Limited decision making “intelligence”
- – Difficult administration
- – Missing overall analytics

1.1 Vision of SDN

- Hardware: cheaper
- Software: features frequent releases, decoupled from hardware
- Functionality: driven by software and controller. Aiming for a programmable network
- Simplicity: from manual to automated, from box centric to network wide, from provisioning in months to provisioning in hours
- Innovations: from closed systems to open and programmable

Virtualization of Computing needs virtualization of Network!

1.2 SDN Devices

All Information from FIB to Config can be updated via API calls (support depending: RESTCONF, NETCONF, . . .)

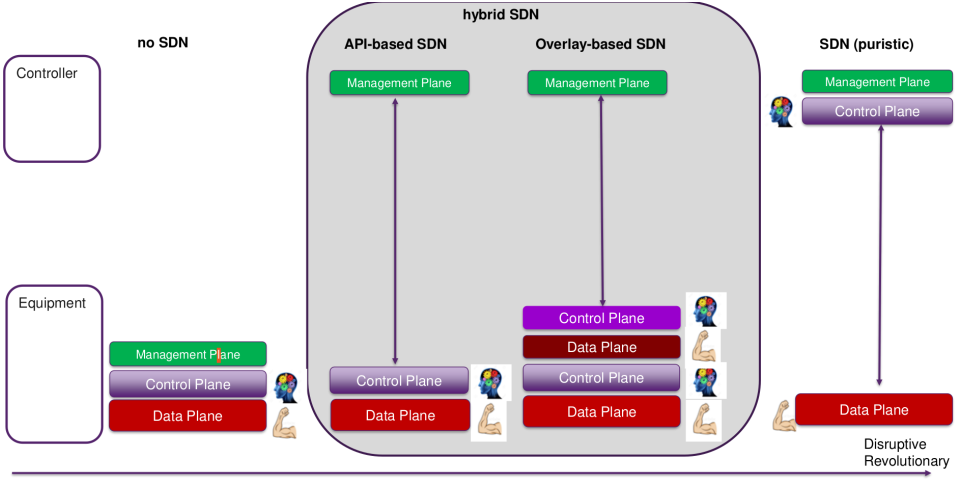
White box switches

- support OpenFlow 1.3
- Third Party OS Support

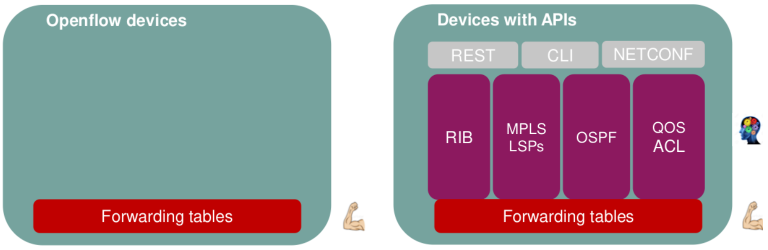
White box OS

- Open Compute Project OCP

Figure 1: Different abstraction levels



- Pica8
- Nvidia Cumulus Linuz
- opennetlinux.org
- fboss

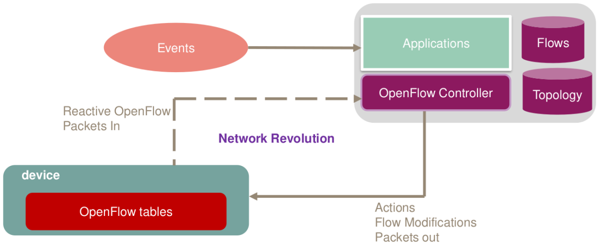


1.3 SDN Approaches

Different levels of abstraction can be applied to the topology, resulting in mainly three different SDN approaches.

1.3.1 Pure SDN

Academic approach. Only Data Plane on each device. Management and Control Plane centralized, resulting in full decoupling. OpenFlow Protocol distributes information, unknown Packets are sent to controller. Flow table is used for forwarding decisions.



Positive

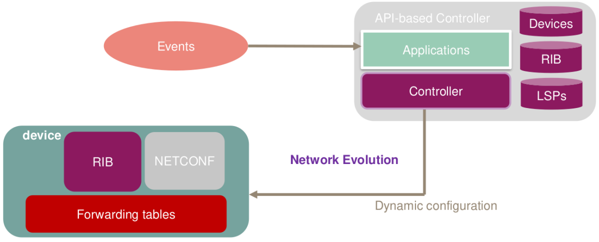
- Independent evolution and development
  - software control of the network can evolve independently from hardware
- control from high-level software program
  - debug/check behaviour more easily
  - testing/troubleshooting

Negative

- controller could be single point of failure
- no topology change without controller
- migration
- high risk

1.3.2 Hybrid SDN, API based

Data and Control Plane on each device, Management Plane centralized. Similar to snmp, ssh scripts.



Positive

- Faster provisioning of new customers and services
- Low impact in case of controller loss:
  - Provisioning delayed

Figure 2: Pure SDN schema example

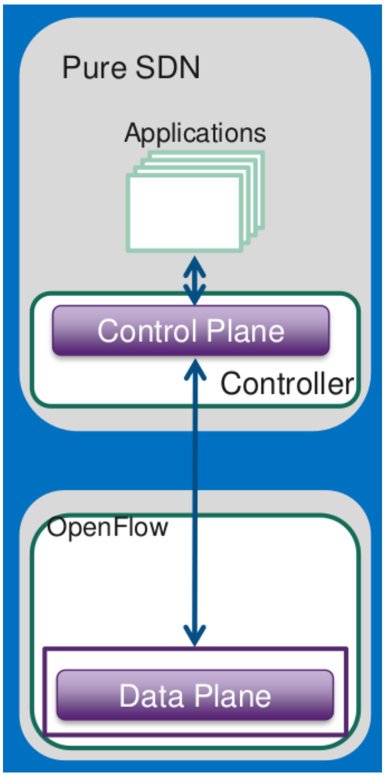
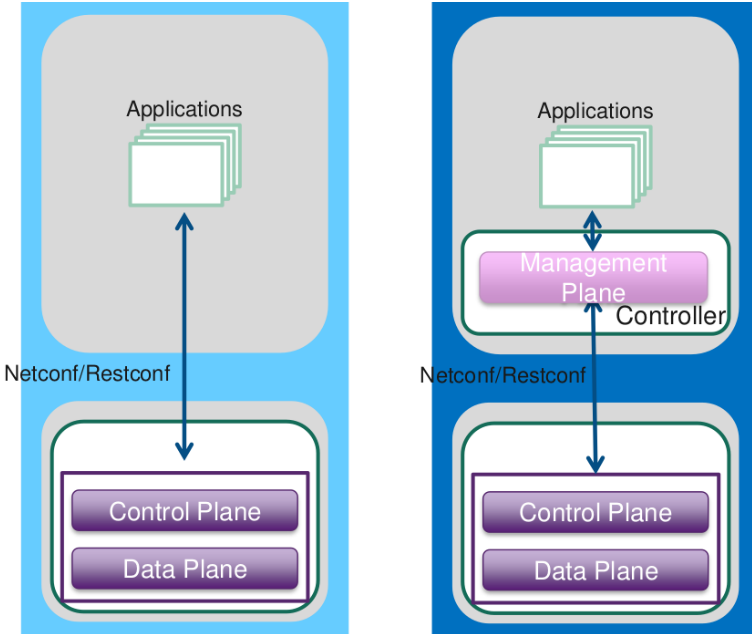


Figure 3: NETCONF and RESTCONF schema example



  - Visibility loss
  - Equivalent to any orchestration system failure

- Network partitioning: low impact
- Increased flexibility and speed

Neutral

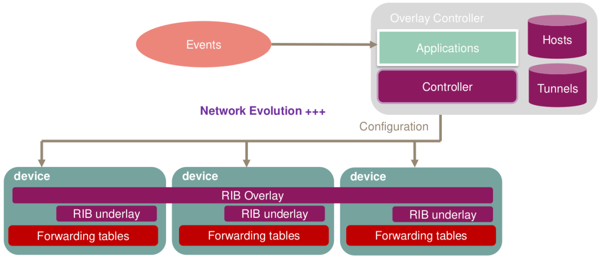
- Normal hardware cost
- No control plane change
- Transactional consistency important (all or nothing commands on devices)

Negative

- Static Management
- Not suited for multivendor environments
- software dependencies

1.3.3 Hybrid SDN, Overlay based

- Underlay Network: optimized, traditional Architecture
- Overlay Network: flexible, virtual network, centralized Management Plane
- Encapsulation necessary (VXLAN, NVGRE, IPSEC, ...)



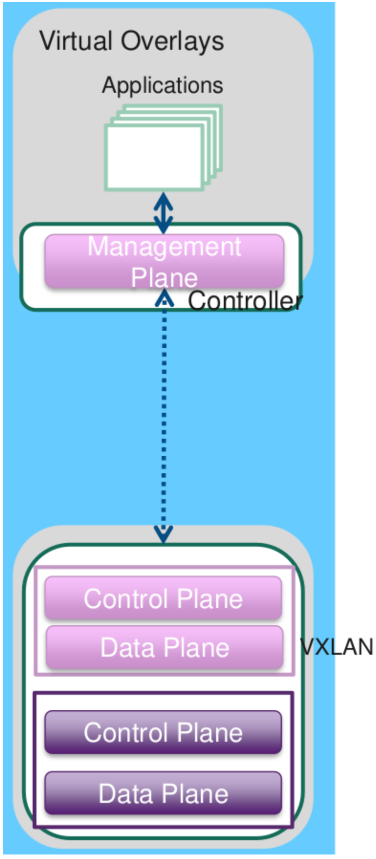
Positive

- Decoupling of services and network
  - Service provisioning in the edge elements
- No impact on the transport core

Negative

- overhead in
  - Encapsulation
  - Processing power
  - Complexity (additional control plane)
- Overlay-to-physical gateways
- End-to-end monitoring and troubleshooting

Figure 4: Overlay based schema example



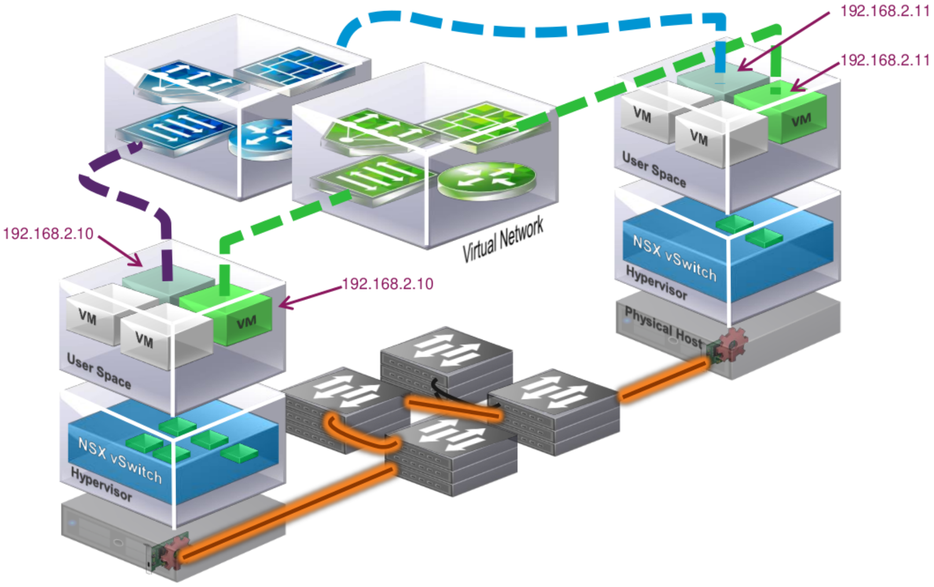


Figure 5: Datacenter example of an overlay solution

2 Segment Routing

RFC 8402: `https://datatracker.ietf.org/doc/html/rfc8402`

"Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through an ordered list of instructions, called segments. A segment can represent any instruction, topological or service based. A segment can have a semantic local to an SR node or global within an SR domain. SR provides a mechanism that allows a flow to be restricted to a specific topological path, while maintaining per-flow state only at the ingress node(s) to the SR domain."

- Prefix-SIDs are
- Adjacency-SIDs are labels with the format 24NXY for the N-th adjacency from x → y
- LDP/RSVP/BGP labels are in the range [90000 - 99999]

Source Routing: The entire path is calculated as a *Segment List* by the source router, or received by a PCE (Path Computation Element). The rest of the network only executes these encoded instructions, there is no per-flow state information.

Segment: An instruction to the processing device on how to forward the packet. The Segment-ID can be encoded as an MPLS label or an IPv6 and is usually associated either with a destination prefix, a local interface or a local service. In combination, a list of segments specifies the entire path a packet is supposed to take.

Local Segment: Only the node that originates this segment understands the associated instruction.

Global Segment: Each node in the SR Domain understands the associated instruction and installs it in its forwarding table.

Default label range [16000- 23999] is called SRGB / Segment Routing Global Block.

Instruction: can be one of the following three.
PUSH – insert segment(s) at the packet head and set first as active
CONTINUE – active segment is not completed and remains active
NEXT – active segment is completed, make next item in SID list active

2.1 IGP Extensions

See also: Segment Routing IGP Control Plane on segment-routing.net

The following segment types are based on on IGP routing information. The usual topology updates with added SID information are distributed by the IGP protocol within the SR-Domain. Prefix-to-SID mapping server

SR for IS-IS supports TLV extensions of the routing protocol – additional Information transmitted via Link State Packets (LSP)

SR for OSPF is implemented by adding new Types of LSA (Link State Advertisements).

Metric-style wide must be applied for the routing protocol configuration in order to support SR capabilities.

Some sub-TLVs supported are

- SR Capability: IS-IS router Capability
- Prefix-SID: Extended IP Reachability
- Prefix-SID: IPv6 IP Reachability
- Prefix-SID: SID/Label Binding
- Adjacency-SID: Extended IS Reachability
- LAN-Adjacecny-SID: Extended IS Reachability

2.1.1 IGP Prefix Segment *(global)*

Shortest path to any known IGP network prefix. ECMP-aware. Global Segment, Label identified as 16000 + Index. Distributed by ISIS/OSPF. Prefix SID is domain-wide unique, assigned manually to the loopback address of each node. **Algorithm ID** specifies the method of choosing a path. Default is 0, shortest path.

To ensure uniqueness of Prefix-SIDs, only one can be associated with each Prefix/Algorithm combination.

Example:

```
1.1.1.1/32 prefix-SID 1001 for algorithm 0
1.1.1.1/32 prefix-SID 2001 for algorithm 1
```

A Prefix Segment can be of two different types

- Node Segment
  Associated with a /32 prefix which is a node address.
  Sets **N-Flag** in Segment ID.

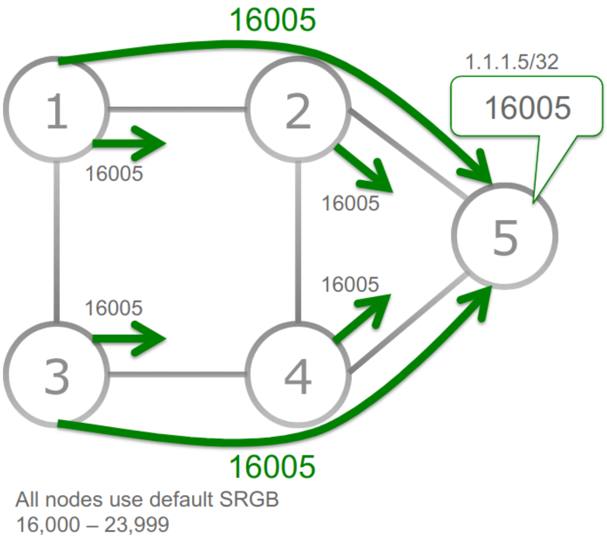


Figure 6: IGP Prefix Segment

- Anycast Segment
  Associated with an anycast prefix, which routes to the geographically closest out of a group of hosts.
  N-Flag is unset!
  Macro-Engineering: can be used to steer traffic via specific region, or make it pass some router performing special network functions.
  Offers ECMP load balancing and high availability.

2.1.2 Adjacency Segment *(local)*

Unidirectional Adjacency, traffic is steered explicitly over an interface / link. Overrides shortest path routing decisions. SID list contains node prefix first, then Adjacency-ID. Distributed by ISIS/OSPF.

- Layer-2 Adjacency can address one specific link inside a Link Aggregation Group (LAG).

- Group Adjacency

2.2 BGP Segments

- BGP Prefix Segment
  Global segment, associated with a BGP Prefix
  "steer traffic along the ECMP-aware BGP multi-path to the prefix associated with this segment"

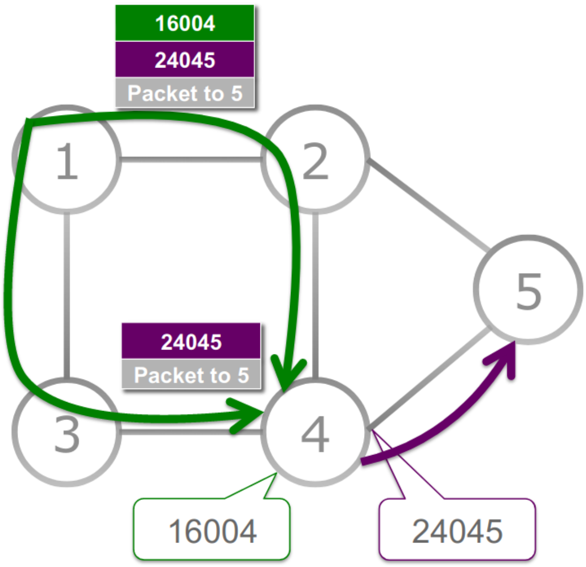


Figure 7: Combining IGP PRefix and Adjacency Segments

- BGP Anycast Segment
  Traffic steering capabilities such as *"steer traffic via spine nodes in group A"*
- BGP Peer Segment
  Associated with BGP Peering sessions to specific neighbor
  Local segments that are signaled via BGP link-state address-family
  "steer traffic to the specific BGP peer node via ECMP multi-path towards that peer router"
  Overrides the traditional BGP mechanism
- BGP Peer Adjacency Segment
  "steer traffic to the specific BGP peer node via the specified interface towards that peer router"

Combining segments can create any kind of end-to-end path.
Traffic steering only happens on source nodes to enable per-flow load balancing.
For **traffic engineering** (see 4) a policy defines the path (SID-List) to be used.

2.3 Labelling defaults

Label space of Segment Routing capable software is usually reserved, even if Segment Routing is not enabled:

- Label range [0-15] reserved for special-purposes
- Label range [16-15,999] reserved for static MPLS labels
- Label range [16,000-23,999] preserved for Segment Routing (Global Block)
- Label range [24,000-max] used for dynamic label allocation (SR Local labels)

2.4 MPLS Data Plane

MPLS data plane allows for direct mapping of key functionalities:

- *Segment* → *Label*
- *SegmentList* → *LabelStack*

Penultimate Hop Popping is enabled by default, Explicit-Null can be enabled if needed.

Prefix-SID label is imposed on a packet if

- Destination matches on a FEC (Forwarding Equivalence Class) with a Prefix-SID
- Downstream Neighbor is SR-Enabled
- Node is configured to prefer SR label imposition
- The matching FEC does not have an associated LDP label

MPLS services can be transported over SR-MPLS, removing the need for LDP as an additional protocol to operate.

Verification commands include

- `show cef 10.0.0.1/32`
- `show cef vrf RED 10.0.0.0/30`
- `show mpls forwarding`
- `show mpls forwarding labels 16004`

3 SRv6

IPv6 Segment Routing header: Next header field: 43 = Routing

- *Segment* $\rightarrow$ *IPv6 address*
- *Segment list* $\rightarrow$ *Address list in the SRH*

A pointer in the SRH points to the Active Segment in the list of segments encoded in the header. No segments are removed while forwarding the packet, only pointers manipulated. Active Segment is copied to the destination address field of the IP header.

- Last segment index is 0
- First segment index is *first segment*
- Active segment index is *segments left*

3.1 The SR Procedure

If source node is SR capable, the following steps are applied to a packet:

1. SR Header is created with the segment list in reversed order of the path.
2. Segment list [0] is the *last* segment
3. Segment list $[n-1]$ is the *first* segment
4. Segments left is set to $n-1$
5. First segment is set to $n-1$

In case a node in transit is not SR-enabled, plain IPv6 forwarding based on the Destination Address header field can be used. No inspection or update of the SR-header is performed.

This results in ***full interoperability*** between SRv6 and IPv6 nodes.

SR Segment Endpoints perform the following steps
IF (segments left > 0), THEN

1. Decrement Segments left by 1
2. Update DA with Segment List [Segments left]
3. Forward according to the new DA

ELSE (segments left $= 0$)

1. Remove the IP and SR header
2. Process the payload
   - Inner IP: Lookup DA and forward
   - TCP/UDP: send to socket

The final destination does not have to be SR-capable.

3.2 Segment format

SRV6 SID is a 128-bit address. Locator part routes to the node performing any possible function defined in the second part.

Optional: the function part can be split into function bits and argument bits.

Locator
Function

3.2.1 SRv6 uSID

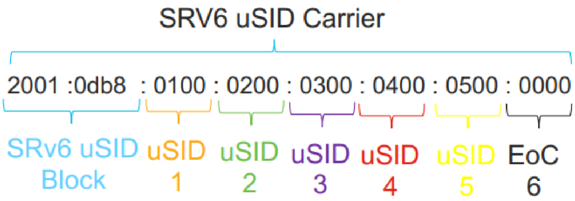
Combines several router IDs into one SRv6 SID. Completely compatible with default SRv6 SIDs.

SRv6 Locator is configured by device:

```
segment-Routing
  srv6
    locators
      locator MAIN
        micro-segment behavior unode psp-usd
        prefix fcbb:bb00:100::/48
```

ISIS configuration for SRv6:

```
router isis 1
  address-family ipv6 unicast
```



‘’

```
    segment-routing srv6
      locator MAIN
```

BGP Control Plane: per-VRF oder per-CE modes possible

```
router bgp 1
  vrf 1
    address-family ipv4 unicast
      segment-routing srv6
        locator MAIN
        alloc mode per-vrf
```

4 Traffic Engineering

Why? Simple, automated and scalable:

- no core state
- no tunnel interface
- no head-end a-priori configuration
- no head-end a-priori steering

Multi-Domain:

- SR-PCE: Path Computation Element
- Binding-SID for scale

What? SR Policies

Tuple of `(head-end, color, end-point)` uniquely identifies an SR Policy. Head-end: Router originating the SR policy (source). Color: a numerical value to differentiate multiple policies between the same pair of nodes. (Green: low-cost policy, red: low-delay policy)

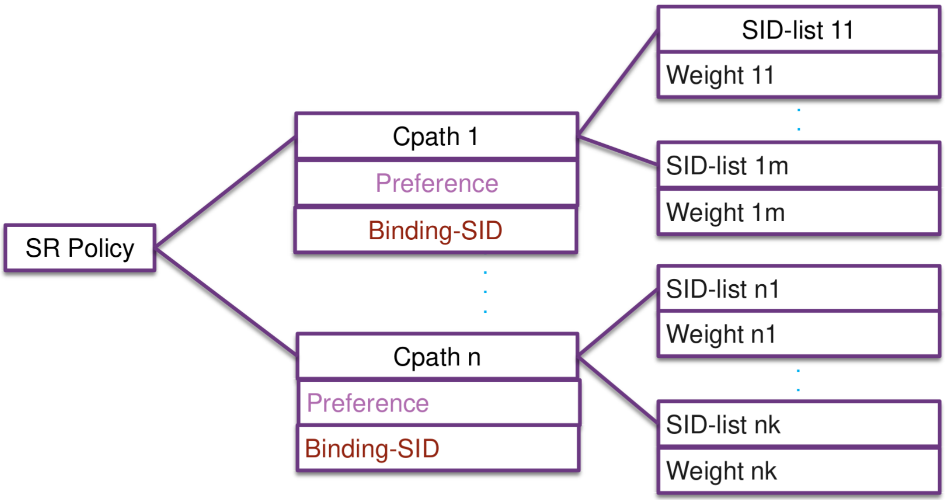


Figure 8: Structure of an SR Policy

- An SR Policy consists of 1-n candidate paths
- An SR Policy instantiates one single candidate path in RIB/FIB
- An SR Policy and all its candidate paths are associated with a single Binding-SID

- Binding SID may change at some point In time, true ID of SR Policy is its tuple
- *Binding SID installs an entry in the forwarding table to steer packets to use this policy*

A candidate path

- is either:
  - dynamic, so that it contains an optimization objective and constraints
  - explicit, so that it contains a single or a set of weighted SID lists.
- has a preference
- is valid if it is usable:
  - not empty
  - first SID is resolvable (to account for multi-domain)

Candidate Path selection happens if

- it is valid
- preference is highest

Validity of Policies is updated upon any network topology change. Traffic steered into an SR Policy path is load-shared over all SID-lists of the path $\rightarrow$ weighted ECMP based on SID List Weight

4.1 Traffic Engineering Controller

Usually, any head-end is able to compute SR-TE paths with certain optimization requirements. Still, a central view of the whole segment routing domain is necessary for several special use cases.

- Disjoint paths: two head-ends explicitly request calculation of disjoint paths from the PCE. The controller can keep track of this requirement also in case of topology changes.
- Inter-domain routing: The SR-TE database is natively multi-domain capable. Information about other SR domains is available via BGP Peering links (BGP-LS).
- Bandwidth brokering: The centralized bandwidth broker receives the bandwidth-related request from the individual routers, keeps track of the available bandwidth in the network and routes the requests accordingly.

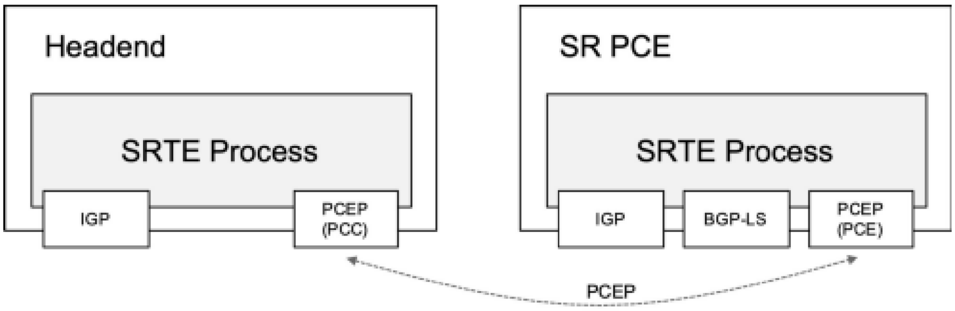


Figure 9: SRTE Headend to PCE communication

PCE – Path Computation Element
PCC – Path Computation Client (Headend)

A PCE provides a path computation service to other devices (PCCs) in the network, based on provided optimization objectives and constraints. Path calculation can be initiated by the headend via stateless request/reply protocol exchange. Once the "delegation" bit is set by the headend, control of the path is then taken over by the PCE.

Path calculation can be initiated by the PCE or by an application via API.

PCE functionality can be enabled on any IOS XR device. However it is recommended to deploy separate nodes for PCE functionality to avoid the mixing of different functionality and enable better scalability.

Receives all topology information from different protocols (IS IS, OSPF, BGP LS) and combines them into the SR TE Database.

Link-delay metric is activated by default and available for policy computation.

`distribute link-state` command enables feeding SRTE DB by the routing protocol (OSPF and IS IS).

Redundancy/High Availability in PCE deployment is achieved with the following concepts:

- Primary/Secondary PCE configuration: PCE configuration on any path calculation client is either designated as primary or secondary, enabling instant failover.

- Topology learning: Using IGP / BGP information, all PCEs in the same network receive the same information about the topology present.

- SR Policy Reporting: When an SR Policy is instanciated, updated or deleted, the headend sends an **SR Policy State Report** to all its connected PCEs.

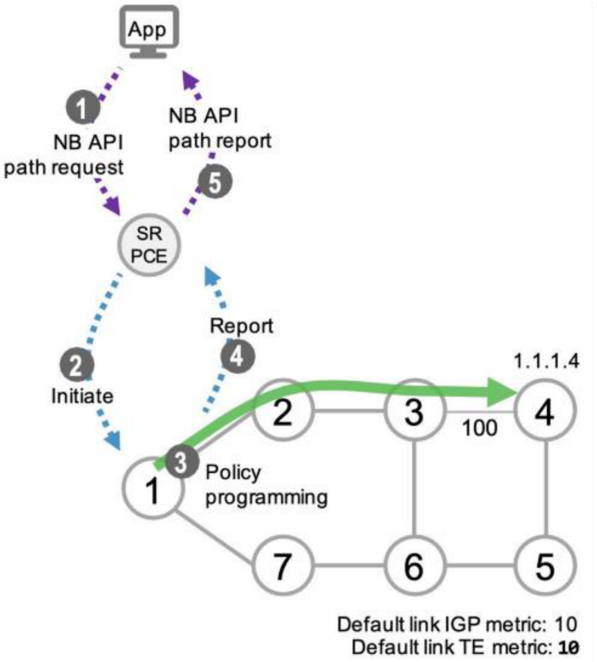


Figure 10: Basic SR TE Architecture

- Re-delegation behaviour: In case of failure of the primary PCE, all paths will be re-delegated to another PCE.
- PCEP Keepalive/Dead Timer: PCEP Messages (keepalive or other) are sent at least every 30 seconds.
- Reachability of the PCE is tracked in every PCCs' forwarding table (no need to wait on any timers).
- Inter-PCE State-Sync PCEP Session: An SR PCE can maintain PCEP sessions to other SR PCEs to indirectly distribute information in case some headend loses connection to one PCE.

Split-brain situation when calculating disjoint-paths: path calculation can be impossible if one path is delegated to PCE1, and it would have to be changed to calculate a disjoint path on PCE2. Creating a master/slave relationship between two PCEs solves this problem by only letting one of two PCEs calculate any disjoint path.

Northbound Interface: communicates with external applications, provides a structured endpoint to access and update topology and path information.

Southbound Interface: communicates with PCC devices, exchanging link state information and SR Paths.
Candidate paths can be distributed via

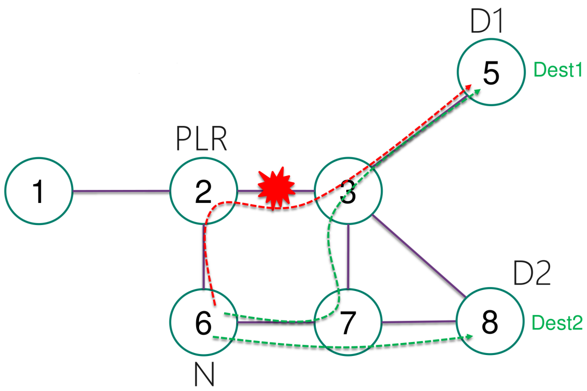


Figure 11: LFA non-ideal repair path

- **PCEP**
- CLI
- NETCONF
- BGP

4.2 LFA – Loop Free Alternate

"a directly connected neighbor that offers a repair path whose shortest path to the destination D does not traverse the protected component."

A suitable LFA does not always exist: it depends on the network topology, metrics and the component to be protected.

This is the reason LFA is usually topology dependent.

The LFA basic Loop-free condition:

$$\text{“}Dist(N, D) < Dist(N, PLR) + Dist(PLR, D)\text{”}$$

compares the length of the new path from the LFA (neighbor) to destination with the path from the LFA crossing the protected (failed) link.

Point of Local Repair **(PLR)**: The Node that registered a link-down event and now has to adjust its path.

N: Node that is/has designated LFA path.

4.2.1 TI-LFA: Topology Independent Loop-Free Alternate

See also TI-LFA on segment-routing.net

Offers node, link and Shared Risk Link Group SRLG protection with sub-50ms downtime. 100% Coverage in any topology. Simple to operate and understand. Automatically computed by the IGP, no other protocol required. No state outside the protecting state at the PLR, local mechanism. Optimum: Backup path follows the post-convergence path. Can be incrementally deployed Applies also to IP and LDP traffic (besides segment routing).

Path Computation: Calculate the shortest path with the outgoing link along the primary path pruned from the topology. Encode this path as a segment list to avoids microloops.

In a network with symmetric metrics, maximum two additional segments are required to form a valid repair path.

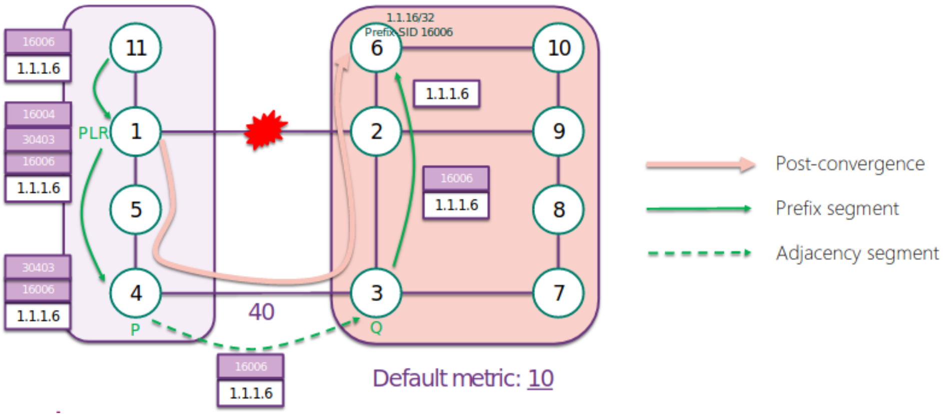


Figure 12: Repair Path using TI-LFA

Sample configuration for link protection:

```
router isis 1
  address-family ipv4 unicast
    segment-routing mpls

interface GigabitEthernet0/0/0/0
  point-to-point
  address-family ipv4 unicast
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa

router ospf 1
  segment-routing mpls
    fast-reroute per-prefix
    fast-reroute per-prefix ti-lfa enable
```

5 Software Defined Access

6 AWS Cloud Networking

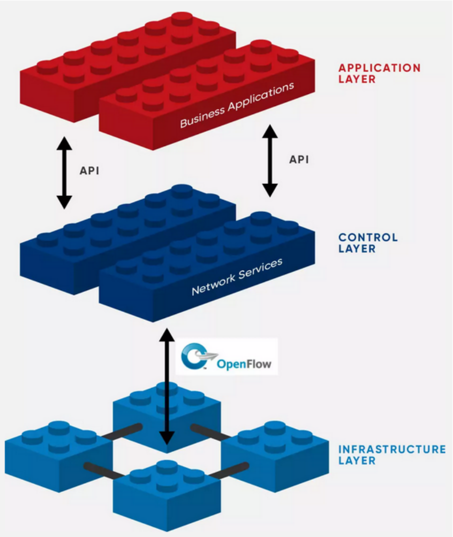
7 Software Defined WAN

HELLO	Sent by the switch, reply by the controller
FEATURE_REQUEST	Sent by controller, as supported OF capabilities
FEATURE_REPLY	Sent by switch to advertise

8 OpenFlow

Managed by the Open Networking Foundation.

Standard Southbound Protocol used between the SDN controller and the switch - *management only!*



OpenFlow operates as TCP Protocol (6644 / 6653) and can be secured by TLS using certificates.

Components of an OpenFlow Switch

- Flow Table(s)
- Group Table
- OpenFlow channel(s) to external controller

8.1 Controller

OpenFlow messages - for OF Channel setup between switch and controller Controller manages *Flow Entries* in every switches flow tables (add, update, delete).

8.2 Flow Tables

Flow entry consists of

- Match fields

- Counter
- Instructions

Replaces traditional MAC/CAM table that stores hosts' hardware addresses. A flow entry is selected by IP packet matching fields, first matching entry is used ordered by priority.

- 39 fields possible to match on in OpenFlow 1.3, BUT must be supported by the hardware used
- usually in routing: most specific match

Instructions can be actions or modify pipeline processing. Possible actions are

- Forward on port
- Drop
- Flood
- Send to controller

If no match in any flow table is found: TABLE_MISS rule configuration: send to controller or drop.