ribbon

Your Path to
Autonomonus Networks

Segment Routing Traffic Engineering for IP Transport Networks

Introduction

As we begin the new decade, we start to see the new generation of communication services slowly coming into reality. User traffic will become more and more dynamic and nomadic, bringing new challenges and setting new goals in IP transport networks. New mobility services, which boost productivity and enhance the quality of life, will require strict, if not better network performance. The network will have to support dynamic traffic behaviours as well as delivering deterministic services. This requires new advanced traffic engineering techniques to be able to support these differentiated services while optimising network resources.

When traffic enters the network, it is routed towards its destination, traditionally using shortest path algorithms on the basis of having the least number of hops. This works if network resources on the shortest path match the traffic demand. In the real world, network resources are limited. Shortest path algorithms usually lead to traffic converging on certain favourable paths making other links in the network underutilized while congesting others. Over the years, traffic engineering (TE) techniques were developed to solve this problem by diverting traffic away from the congested paths and in the process, optimising the network resources.

Early TE mechanisms involved assessing the current state of the network, pre-planning the network according to known traffic patterns, as well as dynamic routing based on link metrics. However, during heavy network loads, congestion still occurs on certain parts of the network, with other parts remaining underutilized. As networks get denser and bigger, these network inefficiencies became more problematic. This led to the development of a better forwarding paradigm called Multiprotocol Label Switching (MPLS), which was originally intended to reduce the complex routing table look-ups by using labels. Being multi-protocol in nature, MPLS was versatile and it didn't take long to incorporate TE mechanisms into it.

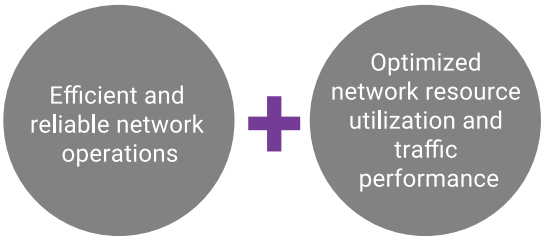
One popular TE implementation on MPLS is using Resource Reservation Protocol - Traffic Engineering (RSVP-TE) and Constraint-based routing (CBR). This involves reoptimizing a path during congestion, by running a constraint-based routing algorithm at the ingress router to find feasible paths that satisfy a specified set of constraints. The calculation is based on the resource requirements of the traffic flow and RSVP-TE is used to reserve the resources along the new path, before the new path is established for the traffic flow. With this mechanism, the network became responsive to network demands while being resource-aware. For relatively static traffic patterns and small networks this is a good solution. However, as networks begin to scale and traffic behaviour became more dynamic, it was apparent that this type of traffic engineering was becoming very complex to control and maintain, leading to network instability and scalability problems.

The motivation for Segment Routing – Traffic Engineering (SR-TE) is to address the complexity and scalability problems of the older traffic engineering techniques, while adapting to the dynamic nature of the current and future traffic patterns, which require better flexibility and programmability in IP transport networks.

This whitepaper is a supplement to Ribbon's Segment Routing in Transport Networks whitepaper and is focusing on the traffic engineering implementation on SR-MPLS. SR-MPLS (RFC 8660) is the instantiation of Segment Routing (SR) over the MPLS data plane. The why and the how are explored in this document by stating the advantages of SR-TE and explaining the basic mechanisms of steering the traffic within an SR enabled IP transport network. Lastly, we look how an SR-TE enabled IP transport network enables new advance services.

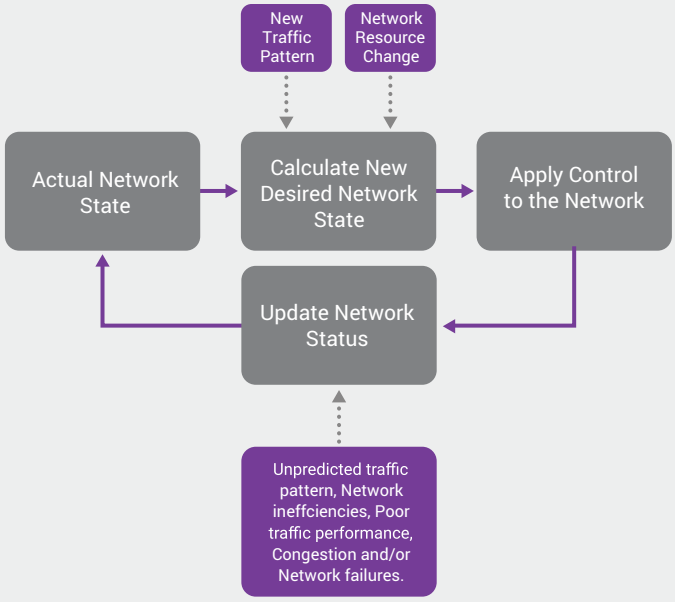
What is Traffic Engineering?

By definition, traffic engineering aims to facilitate efficient and reliable network operations while simultaneously optimizing network resource utilization and traffic performance. If we translate this in business language, the ultimate results of a successful TE implementation will reduce costs and reduce complexity while providing reliable, competitive and advanced services to end users. How is this achieved?



Traffic Engineering

A network operator normally has performance objectives. These can be influenced or dictated by an industry regulator, SLAs with end-users, market competitiveness, or end application requirements. The performance objectives correlate to a desired network state which serves as a goal for TE. To arrive at a desired network state some information is needed. These can be traffic patterns, existing and planned network resources, and the actual network state. This information is processed to calculate the related network changes needed to achieve the desired network state. This is the very heart of the TE process and can be carried out by a Network Management, a centralized controller, a distributed controller or in some small networks it can be a Network Traffic Engineer manually calculating desired network changes. Control is then applied to the network which will bring the network to a new network status. Traditionally, applying of control is done manually but the trend is towards automation. If actual network status does not satisfy the performance objectives, this will trigger a new process cycle and a new desired network state is calculated again, forming an adaptive feedback system.



Traffic Engineering Process in IP Transport Networks

The TE process can either be reactive or pro-active. Reactive means applying control to the network in response to an actual network state. Pro-active means applying control to the network to anticipate a change in traffic pattern or to prevent an undesirable network state – a network failure, for example. In either case, the application of control can be effective if the network infrastructure is flexible, dynamic and programmable, especially today when traffic patterns tend to be dynamic and changes occur frequently. The emergence of new modern communications services makes the traditional IP Transport network no longer efficient and reliable, especially when the new services require the network infrastructure to satisfy specific strict requirements, while coexisting with best-effort services. These new network challenges led to the introduction of segment routing as a new forwarding scheme, which is more suitable for traffic engineering for this new service paradigm.

Segment Routing

SR steers a packet by using an ordered list of instructions appended as packet headers at the ingress node. Each instruction in the ordered list is called a segment, represented by its Segment Identifier (SID). On an MPLS data plane, a SID is encoded as an MPLS label on the packet. A sequence of instructions, or label stack, describes how the packet should be forwarded and processed as it traverses the network. The instruction can be "go to a node", "pass through a specific interface" or "execute an operation". Once a segment is executed by a node, it is deleted from the stack of label to expose the next instruction, then it is forwarded to the next node. This process is repeated until all instructions in the stack are executed and the packet has reached the egress node. This manner of forwarding enables explicit paths as well as dynamic paths to steer traffic using network resources, which are not part of the Interior Gateway Protocol (IGP) shortest path. This allows the operator to divert traffic to avoid congestion or choose an alternative path, when the shortest path is not the best path for the given service.

SR Characteristics and Advantages

SR has the following main characteristics and advantages, which make it suitable for advanced TE in packet networks.

SR supports Equal-cost multipath (ECMP) routing natively. ECMP helps in spreading the load in the network by using multiple paths of equal cost for traffic having the same source and destination. ECMP is desired for even load distribution in the network, thus maximizing the use of resources.

There is no more hop-by-hop signaling in the network and SR simplifies label distribution by incorporating it in the IGP. This significantly improves scalability.

The path can be calculated either by a central entity or by the ingress node itself. The path is then encoded on the packet and maintained only at the ingress nodes, not at the transit nodes and not at the egress nodes. This reduces the number of states transit nodes and egress nodes have to maintain.

Path protection mechanisms like Fast Reroute (FRR) are supported.

Paths are only changed at the ingress node. The transit nodes and egress node along the path simply carry out the instructions on the label stack. This makes the network highly responsive to changes, thus enabling a dynamic network infrastructure.

SR Policy

Implementing TE with SR involves steering a packet flow along a desired path defined by an SR Policy. The headend, which is the start of the SR path, evaluates the incoming packet and matches it to the corresponding SR Policy and then forwards the packet to the next hop. ***The SR Policy contains the segment list of the desired path intended for packets traversing the network from a defined headend towards a defined endpoint.***

An SR policy contains one candidate path or a set of candidate paths, which can be defined as primary path and back-up path/s. These are called ***candidate paths*** because first, they need to be processed and evaluated according to constraints, optimization objectives, preferences and validity. Only after the evaluation and selection processes, the chosen candidate path becomes active and its associated segment list(s) is installed in the forwarding table (FIB) / routing table (RIB) to be used as the packet header.

By definition, an SR Policy is identified through the **tuple** <headend, color, endpoint>.

- The **headend** node, specified as an IPv4 or IPv6 address, executes the SR Policy and encodes the segment list on the packet.
- The **color**, represented by a 32-bit numerical value, is the intent of the SR Policy. The value can represent the "color" or how traffic should be treated as it traverses the network. For example, a color value X can represent low latency path and a color value Y can represent a high bandwidth path.
- The **endpoint**, specified as an IPv4 or IPv6 address, is the destination of the SR Policy.

It is possible to assign a symbolic name using ASCII characters to an SR Policy and its candidate paths to ease troubleshooting and network programming. In our example below, SR Policy ***RedPolicy*** has two candidate paths – ***Path1*** and ***Path2***.

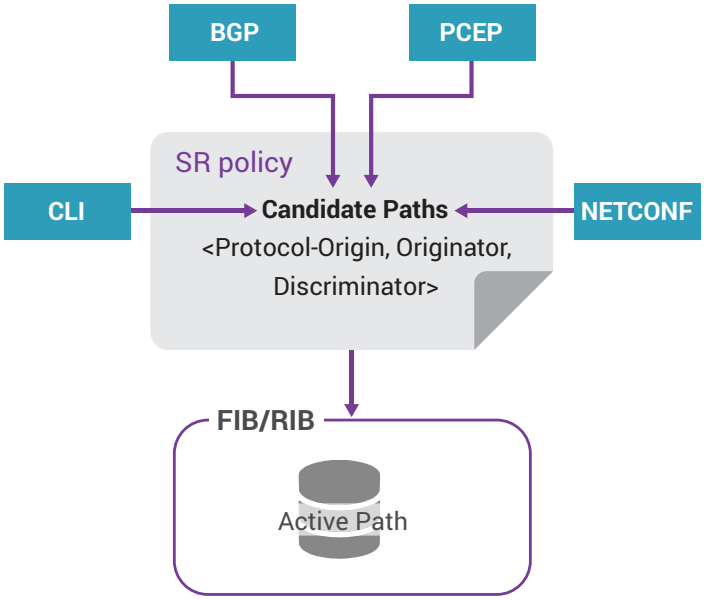
SR policy ***RedPolicy*** < headend=***192.168.0.51*** , color=***IntentValue*** , endpoint=***192.168.0.57*** >
Candidate-path ***Path1***,
Preference $Value_1$,
SID List < SID_1, SID_2, SID_3 >
Candidate-path ***Path2***,
Preference $Value_2$,
SID List < SID_4, SID_5, SID_6 >

Anatomy of an SR Policy

Path Validation and Selection Processes

A candidate path is represented by a segment list or a set of segment lists. In our example above, candidate paths Path1 and Path2 contain only one segment list (SID List) for simplicity. Each candidate path has a preference value. The highest **preference** value is considered the primary path and will be selected if the path is **valid**. If the primary path is invalid, then the secondary path having the next highest preference value, will be selected. This validity verification is performed by the headend to check if the candidate path is useable. For example, a common criterion for validity is checking the reachability SIDs on the SID List. If one of the SIDs, which is required to be validated, is unreachable, or not present in the **SR Database**, then the candidate path becomes invalid. An invalid candidate path cannot be selected even if the preference value is the highest.

SR-DB (SR Database). A database holding basic information about the network's topology, IGP metric, TE metric, latency extended TE metric and SR information like SRGB, SRLB, and SIDs. It may also contain other network related information. The info may come from the IGP, BGP-LS or NETCONF. This collection of data is used to validate and compute SR Policies together with the candidate paths.



In most implementations, the preference value is sufficient to evaluate which ones are the primary and secondary paths within an SR policy. ***Preference values are recommended to be configured with different values, so selection can be made simple.*** In complex scenarios, where the network implementation requires equal preference values, a candidate path has attributes as part of its identification to break the tie. These attributes relate to the source of the candidate path. These are **protocol-origin**, **originator** and **discriminator** values. The tie breaking process is evaluated in this order until a candidate path is chosen by the headend. The protocol-origin relates to the protocol source of the candidate path, which can be learned via CLI, NETCONF, PCEP or BGP. The operator can set which source is preferred than others. If that doesn't break the tie, then the ***originator*** is evaluated. This relates to the address of the source. These are the ASN and IP addresses of the protocol-origin. This is useful if there are multiple controllers in the network. If the above criteria are not sufficient to reach a selection, a ***discriminator*** value is configured by the operator as the ultimate tie breaker.

The selected valid candidate path becomes the **active** path of the SR Policy and the corresponding SID list is installed in the FIB / RIB of the headend. In the case where the current active path is no longer valid or deleted, the selection process will be re-executed or revalidated. This is usually the case if there is a change in the topology and the active path is affected.

Types of Candidate Paths

There are two types of paths which are being used in traffic engineering.

Explicit Paths. These are predefined paths, which can be provisioned manually by the operator or via a controller. Operators can have more control on the logic in calculating these paths. The algorithm can be based on known operational practices or it can be ad hoc, fulfilling a specific purpose. The segment list of an explicit path is not computed by the headend. However, the headend is responsible for checking the validity of an explicit path.

Dynamic Paths. These are paths that are calculated by the headend based on metrics and constraints using the SR-DB. If local computation is not possible or not desired, the headend can request a Path Computation Element (PCE) to compute the path. Metrics can be based on IGP metric, TE metric, xor latency extended TE metric. Constraints can also be used in addition to a metric. Thus, it is possible to calculate a path to have a minimum cumulative metric while applying specific constraints. Examples of constraints are:

- Inclusion and/or exclusion of TE affinity, IP address, Node, Link and/or SRLG;
- Maximum accumulated metric (IGP, TE and latency);
- Maximum number of SIDs in the SID List; and
- Must be disjoint from another path.

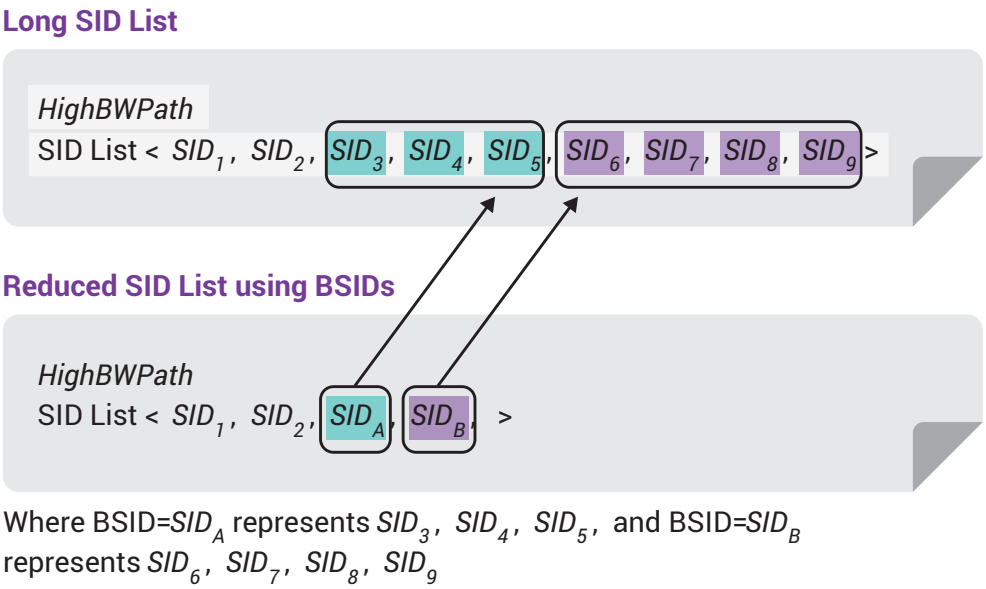
For instance, we can configure the headend to dynamically calculate a path having the lowest latency while considering only the nodes in the network configured with a specific TE-affinity.

TE-Affinity – These are link attributes that can be set to create areas in the network that can be included or excluded in path computation.

Shared Risk Link Groups (SRLG) – Links with the same risk of failure, usually sharing the same network resource.

Computations of dynamic paths ideally follow an **SR native algorithm**, which is finding a SID List that minimizes the number of SIDs and taking advantage of ECMP routing along the path. However, this algorithm may not be preferred for paths mimicking TDM circuit behavior, where load balancing is not ideal. In this case, path computation is done using a classic algorithm without ECMP.

Binding Segment

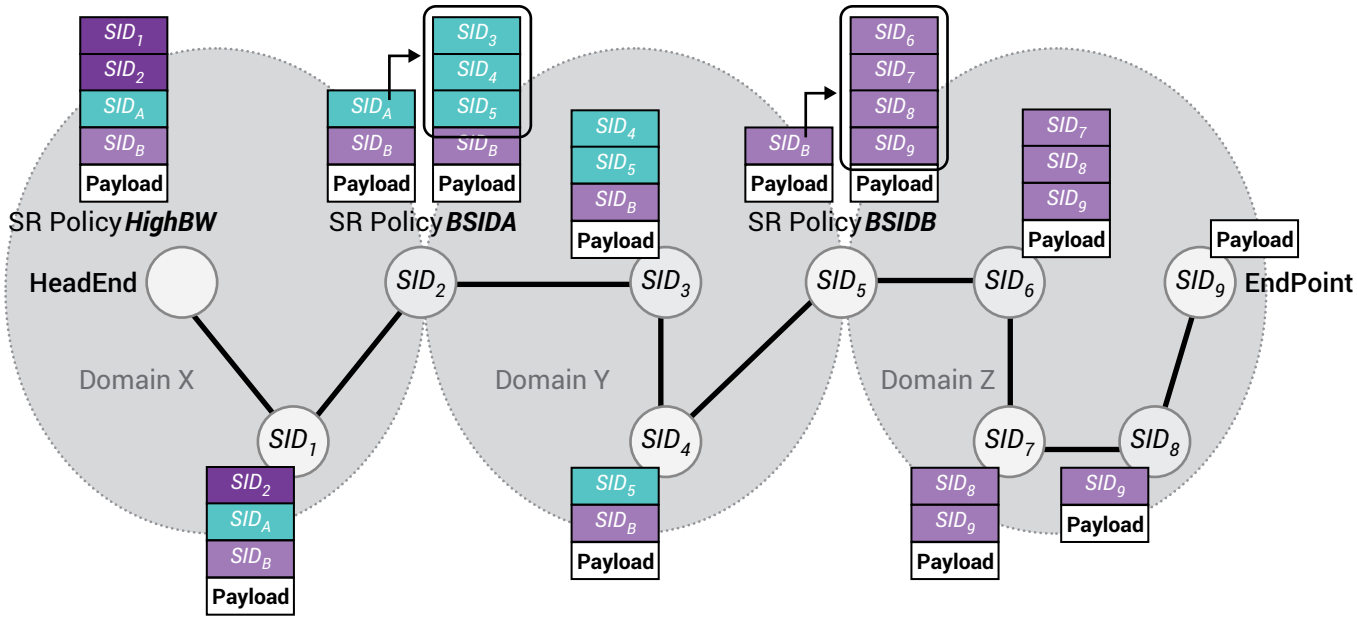


Where BSID=SID_A represents SID_3, SID_4, SID_5, and BSID=SID_B represents SID_6, SID_7, SID_8, SID_9

Binding Segment is a special type of segment represented by BSID, which enables scaling, network opacity and service independence in SR networks. In cases where the SID list is very long, the headend's hardware may not be capable of processing it, or the headend will invalidate the path because the SID List had exceeded the Maximum SID Depth (MSD) set for the network.

The SR native algorithm will always try to compute a minimized number of SIDs in the SID list. However, in complex network topologies, a long SID List cannot be avoided. In such cases, a binding segment will reduce the SID list by representing some of the SIDs using a BSID, which can be allocated locally by the node from the SRLB or globally from the SRGB.

In a simple illustration, let's say we want to instantiate an explicit path called ***HighBWPath*** in a network with MSD of eight SIDs. However, ***HighBWPath*** has nine SIDs in the SID list. To shorten the list while still traversing the same exact path in the network, we use two BSIDs. ***SID_A*** represents ***SID_3*** , ***SID_4*** , and ***SID_5*** , while ***SID_B*** represents ***SID_6*** , ***SID_7*** , ***SID_8*** and ***SID_9***. This will result in an explicit path having only four SIDs in the SID List instead of nine. In SR-MPLS, this refers to ***label stack compression***.



Explicit Path: HighBWPath

Instantiation of BSID requires an SR Policy configured at the node replacing the BSID with the corresponding set of SIDs. In our illustration, we then have three SR Policies:

- SR Policy ***HighBW*** at the headend having ***HighBWPath*** as active path with SID List < SID_1 , SID_2 , SID_A , SID_B >;
- SR Policy ***BSIDA*** at the node with SID_2 having BSID=SID_A associated to SID List < SID_3 , SID_4 , SID_5 >; and
- SR Policy ***BSIDB*** at the node with SID_5 having BSID=SID_B associated to SID List < SID_6 , SID_7 , SID_8 , SID_9 >.

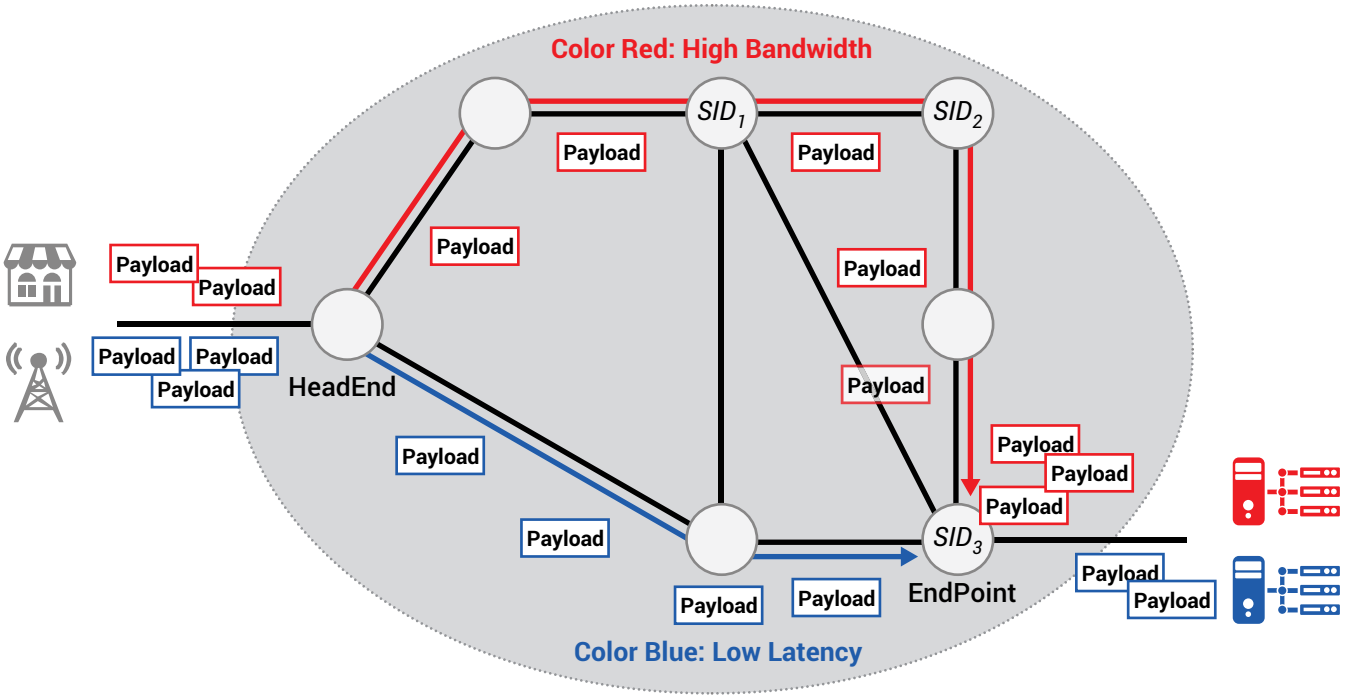
This will result in a packet steered into the SR Policy ***HighBW*** to have no more than four encoded instructions in the packet as it travels through the explicit path ***HighBWPath***.

SR Policy ***HighBW*** also demonstrates the benefits of network opacity and service independence, where the active path goes through different IGP-SR domains X, Y and Z. In this multi-domain topology, the BSIDs can remain the same, even if the path is re-routed within domain Y or domain Z. For example, $BSID_A$ in domain Y could change to SID List <SID_4 , SID_5> bypassing SID_3 due to a topology change. The headend in domain X maintains the same SID list < SID_1 , SID_2 , SID_A , SID_B > and doesn't have to know how, that the path is routed within domain Y. This is useful when the administrative domains do not want to share network information with other domains, as long as they can maintain the agreed SLA.

Color

In real network scenarios, there are many types of data traffic between a headend and endpoint pair. These are data traffic having the same set of source and destination addresses but having different purposes. One type of data traffic can be for low latency, another for best-effort, another for high bandwidth and so on. Each type requires different treatment as it traverses across the network. The color attribute of the SR Policy will make sure that each type of traffic or service is differentiated in the network. It also makes sure that each SR Policy is unique, even if it has the same set of headend and endpoint with other SR Policies.

Color is a numerical value that represents the intent of the SR Policy. We illustrate this with a fixed and mobile convergence (FMC) network carrying two types of traffic: a premium fixed network application requiring high bandwidth; and a premium mobile application requiring low latency.



SR policy ***ReadPolicy***
< headend=***192.168.0.51*** , color=***100***, endpoint=***192.168.0.57*** >
Candidate-path ***RedHighBW*** ,
Preference ***Value*** $_1$,
SID List < SID_1 , SID_2 , SID_3 >

SR policy ***BluePolicy***
< headend=***192.168.0.51*** , color=***200***, endpoint=***192.168.0.57*** >
Candidate-path ***BlueLowLatency*** ,
Preference ***Value*** $_1$,
SID List < SID_3 >

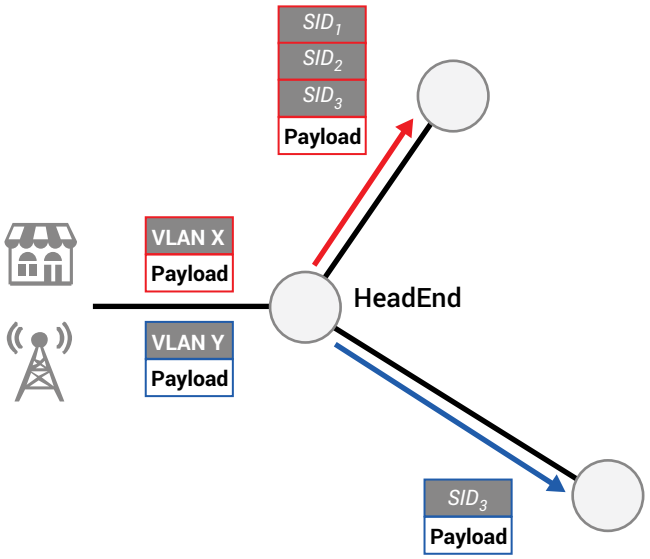
Fixed and Mobile Convergence (FMC) Packet Network

In this network, two SR Policies are configured. SR Policy ***RedPolicy*** is configured to have the attribute color=**100** to represent the color **red**. SR Policy ***BluePolicy*** is configured to have the attribute color=**200** to represent the **color blue**. The **RedHighBW** path is an explicit path <SID_1 , SID_2 and SID_3 >, which takes the longer route but guarantees high bandwidth SLA. The ***BlueLowLatency*** path takes the normal shortest path IGP route, which guaranties the lowest latencies. Thus, it has only one SID < SID_3 > in the SID List.

Steering traffic into the SR Policy

As the packet enters the SR domain, it is evaluated by the headend, which decides how to steer the traffic using an SR Policy. There are many ways to do this and different logics can be applied. The common way is to evaluate any packet header attribute of the incoming packet. These attributes can be destination/source MAC Address, VLAN, TOS, destination/source IP Address, DSCP or transport port number. These attributes are then matched to a desired SR Policy based on the color and endpoint values. This is referred to as ***per-flow Steering***. Packets that have the same matched header attribute belong to the same flow and will be steered with the same SR Policy instantiated in the FIB/RIB.

In our simple FMC illustration, we can configure the traffic coming from the fixed network application with ***VLAN X*** and the traffic from the mobile network application with ***VLAN Y***. After the headend inspects the packets, it determines ***VLAN X*** should be steered to an SR Policy having color=***100***, which is SR Policy ***RedPolicy***, and ***VLAN Y*** should be steered to an SR Policy having color=***200***, which is SR Policy ***BluePolicy***. The headend then appends the corresponding SID List on the packet and forwards the packets to the corresponding next hops.



In the case of L2VPN and L3VPN, a preference policy can be defined at the headend to direct the service to use corresponding SR Policies. In this case, the headend will first append the service label, pseudowire label or BGP VPN label on the packet. After that, the SID List of the SR Policy will be appended on top of the service label of the packet.

In a multi-domain topology scenario, it is also possible that the incoming packet, related to a service is already carrying a label, which matches to a BSID associated with an SR Policy. In this case, the BSID is assessed and the packet is forwarded replacing the BSID with its corresponding SID List of the SR Policy.

A local routing policy can also be defined at the headend to override the IGP path, directing the incoming packet to use a specific SR Policy instead. For other incoming packets that does not have any SR Policy association, the normal IGP forwarding can be applied.

In summary, we illustrated the fundamental process of traffic engineering using SR, which describes how a packet is delivered from source to destination within an SR domain. It starts with defining an SR Policy for a given headend, endpoint and color. Candidate paths for an SR Policy are then either defined explicitly or computed dynamically using metrics and constraints. Afterwards, validity and selection processes are executed to determine the active path, which is installed in the FIB/RIB. Finally, an incoming packet is evaluated, and the corresponding SR Policy is applied to the packet to deliver the service according to the intent.

Conclusion

Data is the new gold and we have seen a huge increase in the amount data traffic over the recent years. New types of data services have also appeared, driven by new applications. Having the capacity is no longer enough. There is now a need to guarantee the SLA associated to the type of service, especially for applications which necessitate not just bandwidth, but also other requirements such as latency and high availability. Services also tend to be more event-driven and intent-driven. This means, paths in the network will be dynamic, which can change based on events or intents. This requires the IP transport network to have a high degree of flexibility and programmability.

SR-TE provides different ways to steer a packet and is intent-aware. Using the color attribute, it enables the network operator to define different paths to correspond to the different types services it carries. Paths can be calculated dynamically using various network variables and algorithms, or paths can also be explicitly defined, giving the network operator the freedom to use a chosen logic, non-network variables and business applications to influence the calculation of the path. Depending which strategy will yield better efficiency for the network infrastructure, paths can be computed by the headend using SR-DB, or computation can be delegated to a centralized controller, therefore reducing the processing load for the headend.

SR-TE does not only enable the network operator to offer new services, it also solves complexity and scalability problems of the older TE techniques. SR-TE complies to the very definition of TE. It facilitates efficient and reliable network operations by supporting advanced protection schemes. It is ECMP-aware, utilizing the network resources efficiently by distributing the load evenly. It further optimizes network resource and traffic performance by supporting explicit paths and source routing, which enables programmability and flexibility. Many developments in the industry are looking at advanced capabilities offered by SR-TE and the services it enables, thereby ensuring network longevity, efficiency and profitability.

Contact Us Contact us for more information on the 5G revenue generating opportunity at rbbn.com

Abbreviations

ASN	BGP Autonomous System Number
BGP	Border Gateway Protocol
BGP-LS	Border Gateway Protocol Link-State
BSID	Binding SID
CLI	Command-Line Interface
ECMP	Equal-cost multipath
FIB	Forwarding Information Base
FMC	Fixed–mobile convergence
FRR	Fast Reroute
IGP	Interior Gateway Protocol
IP	Internet Protocol
IPv4	IP version 4
IPv6	IP version 6
MPLS	Multiprotocol Label Switching
MSD	Maximum SID Depth
NETCONF	Network Configuration Protocol
PCE	Path Computation Element
PCEP	Path Computation Element Communication Protocol
RFC	Request for Comments
RIB	Routing Information Base
RSVP-TE	Resource Reservation Protocol - Traffic Engineering
SID	Segment Identifier
SR	Segment Routing
SR-DB	Segment Routing-Database
SR-MPLS	SR over MPLS
SRGB	SR Global Block
SRLB	SR Local Block
TDM	Time Division Multiplexing
TE	Traffic Engineering
L2VPN	Layer 2 Virtual Private Network
L3VPN	Layer 3 Virtual Private Network

About Ribbon

Ribbon Communications (Nasdaq: RBBN) is a global provider of voice communications software, IP routing, and optical networking to mobile and wireline service providers, enterprises, critical infrastructure and defense sectors. We support our customers' Path to Autonomous Networks by leveraging the latest AIOps automation platforms and Agentic AI technologies, helping them deliver better customer experiences, reduce operational costs, and achieve sustainable growth. To learn more about Ribbon visit rbbn.com.