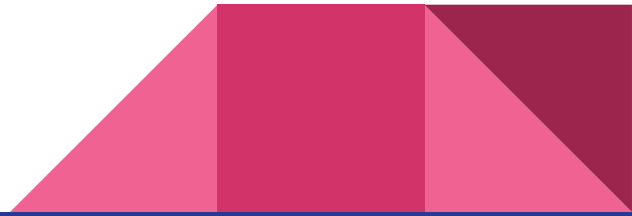
Auto-bandwidth with SR-TE

Dmytro Shypovalov
dmytro@vegvisir.ie

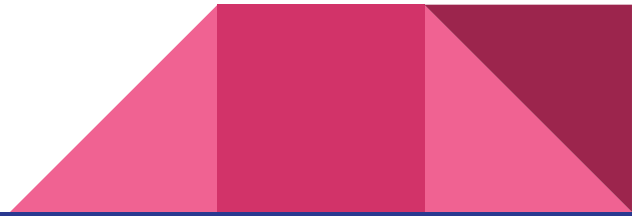
RSVP-TE auto-bandwidth

- RSVP-TE can signal bandwidth reservations per LSP
- A router can also measure the actual bandwidth utilization of a given LSP and adjust the signaling as required
- For example, adjust every few hours or once per day, to keep track on daily peak usage
- There is also overflow and underflow, to quickly adjust bandwidth upon sudden changes



Problems with RSVP-TE

- High operational complexity, poor multi-vendor interop, poor scalability
- Each router signals bandwidth independently -> lack of deterministic routing, each reoptimization can yield different results
- Auto-bandwidth can’t adequately react on changing traffic destinations (e.g. when BGP best path changes from PE1 to PE2)
- RSVP has no mechanism to map different services on the same PE to different LSP
- Adding new routers requires configuring new LSP on all existing routers

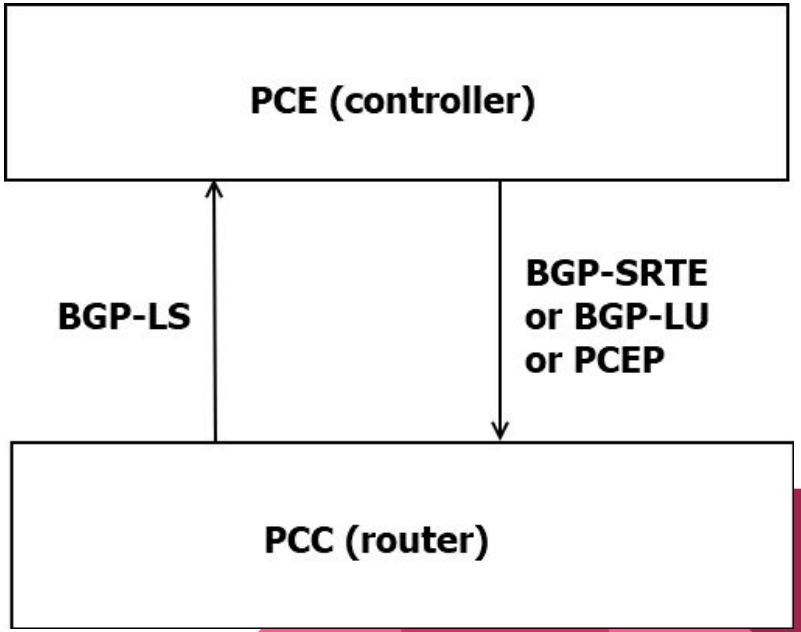


Traffic engineering with Segment Routing

- SR is simpler than RSVP, there are a lot of vendor implementations and they interwork reasonably well
- SR is also more scalable than RSVP
- SR-TE color extended community allows to map traffic from different services on the same router to different LSP
- One caveat - bandwidth reservations in SR require a controller (PCE)
    - This is because SR is stateless so an intermediate router is not aware of SR-TE policies transiting through it
- By using a PCE with a centralized BW database, it’s possible to assign static bandwidth constraint to SR LSP
- What about auto-bandwidth?

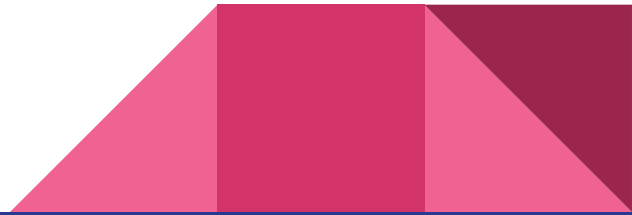
SR-TE PCE basics

- PCE receives network topology via BGP-LS
- PCE computes policies
- PCE sends policies via either of the following:
  - BGP-LU (simplest, works with a lot of open source and whitebox vendors, but limited feature support)
  - BGP-SRTE (best option: simple, scalable, supports readvertisement via RR)
  - PCEP (extremely overengineered protocol, big vendors push for it to get lock-in)
  - Also possible to use vendor-specific API to push policies (GNMI, Netconf etc)



SR-TE auto-bandwidth: IETF approach

- **RFC8773:** PCEP extensions for MPLS-TE LSP Auto-Bandwidth adjustment
- Similar to old RSVP-TE auto-bandwidth, but with a PCE
  - Router measures traffic rates on LSP, advises controller on how much bandwidth it has to reserve
- Implementation status is not clear, but generally speaking, PCEP doesn't interop well due to poorly written standards, so likely this will be limited to a vendor-specific solution
- Also, a PCEP session is required between PCE and each PCC

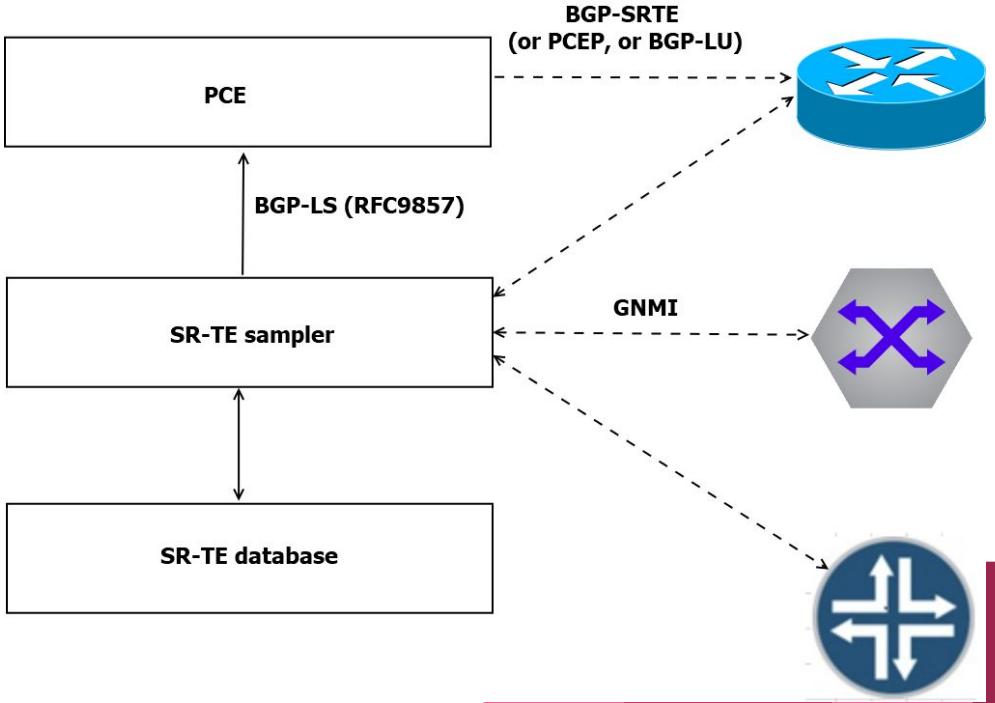


SR-TE auto-bandwidth: practical approach

- **RFC9857:** Advertising Segment Routing policies using BGP-LS
- SR-TE with must use BGP-LS anyway, this new RFC improves SR policy visibility
- Some implementations already support this RFC
- Policies can be advertised by either PCC or PCE
  - SR Bandwidth constraint TLV (section 5.6.3) can be used to advertise current bandwidth on each LSP
  - RFC doesn't mention auto-bandwidth, but also doesn't prohibit using it
- Most SR implementations support per-LSP counters
- In practice, counters + RFC9857 is all we need

SR-TE bandwidth sampler design

- Sampler gets SR LSP counters via GNMI
- Based on adjust interval and threshold, the sampler updates bandwidth rate for each SR LSP and advertises it via BGP-LS
- Currently works with Cisco, Arista, Juniper
- If a router can advertise LSP bandwidth per RFC9857, sampler is not needed, but having a centralized bandwidth database is still useful



Sampler config and outputs

```
router bgp 65001
   router-id 100.2.2.2
   !
   neighbor 10.10.10.202
      remote-as 65001
   !
   neighbor 192.168.102.102
      remote-as 65002
      ebgp-multihop 10
!
sampling options
   sampling interval 10
   adjust interval 60
   adjust threshold 10
!
telemetry profiles
   !
   profile EOS_PROFILE
      os eos
      port 6030
      auth password
      username admin
      password admin
!
telemetry clients
   !
   group EOS_CLIENTS
      profile EOS_PROFILE
      client 192.168.102.107
      client 192.168.102.108
```

```
lmk-vm103-dev-bw-sampler#show sampling policies
Sampling policies information
Number of policies: 12, active 12, stale 0
Status codes: ~ stale

    Policy                                          Rate                 Last updated
    [2.2.2.2][1.1.1.1][101]                         40.564 Gbps                0:00:08
    [2.2.2.2][7.7.7.7][101]                         40.408 Gbps                0:00:08
    [2.2.2.2][8.8.8.8][101]                         40.345 Gbps                0:00:08
    [1.1.1.1][2.2.2.2][101]                         39.924 Gbps                0:00:08
    [1.1.1.1][7.7.7.7][101]                         39.297 Gbps                0:00:08
    [1.1.1.1][8.8.8.8][101]                         39.191 Gbps                0:00:08
    [7.7.7.7][1.1.1.1][101]                         40.391 Gbps                0:00:08
    [7.7.7.7][2.2.2.2][101]                         39.368 Gbps                0:00:08
    [7.7.7.7][8.8.8.8][101]                         40.281 Gbps                0:00:08
    [8.8.8.8][1.1.1.1][101]                         39.950 Gbps                0:00:08
    [8.8.8.8][2.2.2.2][101]                         39.947 Gbps                0:00:08
    [8.8.8.8][7.7.7.7][101]                         39.841 Gbps                0:00:08

lmk-vm103-dev-bw-sampler#show sampling policies [1.1.1.1][2.2.2.2][101]
Detailed sampling policies information
Number of policies: 1, active 1, stale 0

Sampled policy entry for [1.1.1.1][2.2.2.2][101]
  Router-id: 1.1.1.1
  Endpoint: 2.2.2.2
  Color: 101
  Rate 39.924 Gbps, calculated from 6 samples within 50.4 seconds
  Last updated: 0:00:49 ago
```

Calculating auto-bandwidth policies on PCE

- Once PCE supports RFC9857, it can map received BGP-LS routes to local SR-TE LSP, using the tuple ***(headend router-id, endpoint, color)*** and assign bandwidth constraint
- For this demo, I implemented RFC9857 support in Traffic Dictator
- In the current (experimental) implementation, adjust interval and threshold are controlled by the sampler, however it's possible to add extra constraints on the PCE
- This way, auto-bandwidth now works with SR-TE
  - But it's not good enough
  - We just replicated the RSVP-TE behaviour from 25 years ago (with better scalability and multi-vendor support)
  - However, this is just a stepping stone to greatness

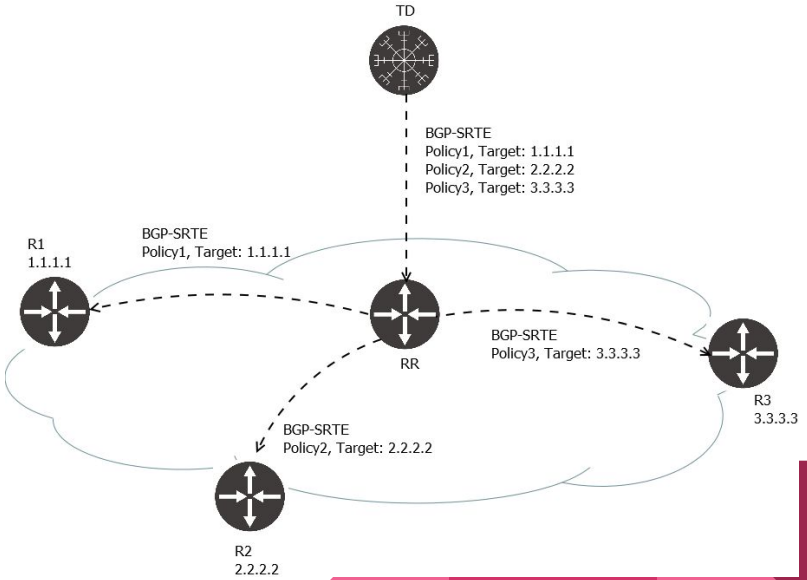
SR-TE mesh templates - foundation for scalable TE

- While SR-TE **control plane** scales better than RSVP-TE, the **management** remains the same
  - You still have to configure a lot of LSP
  - When adding a new router, configure new LSP on every router
  - If using a controller (PCE) to provision LSP with PCEP, a PCEP session is required with each router
- Mesh templates implementation in Traffic Dictator let the operator generate a lot of similar LSP within the topology using a very minimalistic config
- Templates are fully dynamic - i.e. they automatically add/remove LSP when routers are added/removed
- BGP-SRTE is used to advertise the LSP generated from template - only a session with RR is required; route-target designates the headend router

SR-TE mesh templates - example

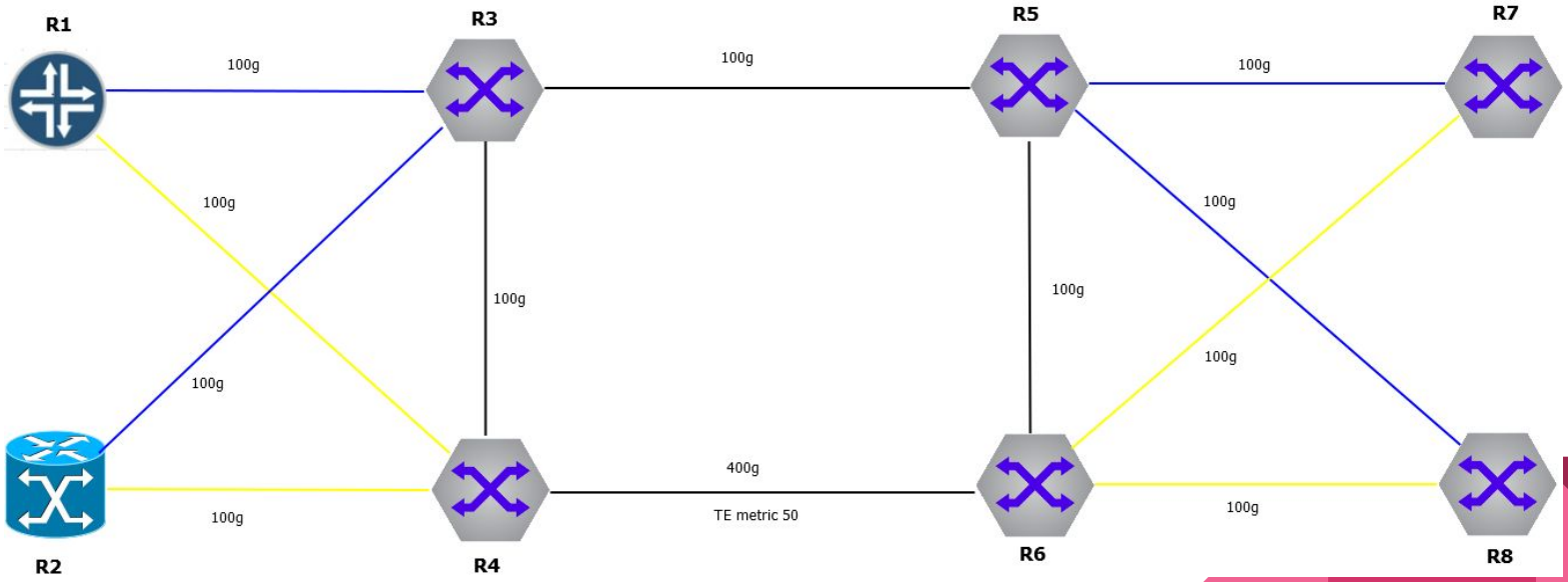
- This config generates a full mesh of SR-TE LSP in topology with BGP-LS id 101
- Every LSP will have color 11101
- Constraint is affinity-set CORE_LINKS (relevant admin group)
- All LSP are advertised to RR using BGP-SRTE
- RR advertises the LSP to the relevant routers

```
traffic-eng mesh-templates
   !
   template TOPO_101_BLUE
      topology-id 101
      color 11101
      install indirect srte peer-group TOPO_101_RR
      !
      candidate-path preference 100
         affinity-set CORE_LINKS
```

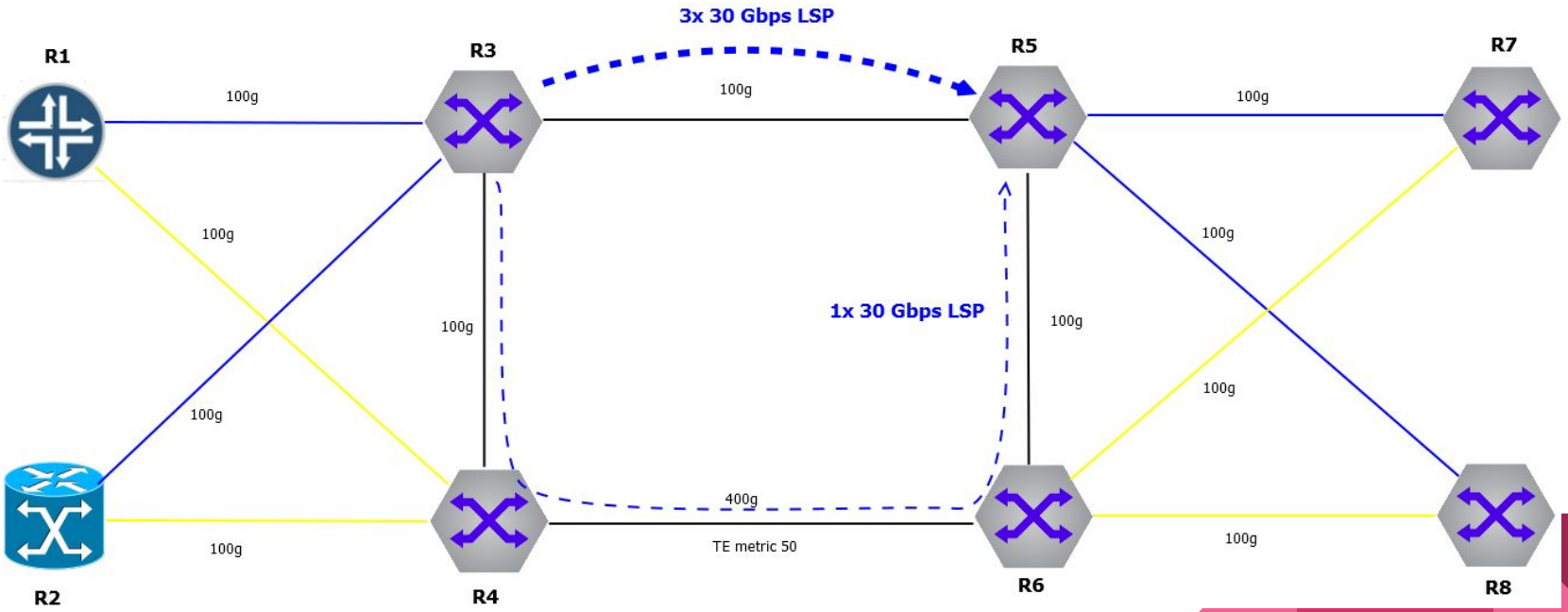


SR-TE mesh templates + auto-bandwidth

- We need an LSP mesh between PE (R1, R2, R7, R8)
- Core link R3-R5 has lower latency than R4-R6, however it has limited capacity



Scenario 1: one mesh, each LSP has ~30 Gbps of traffic



Scenario 1: config and outputs

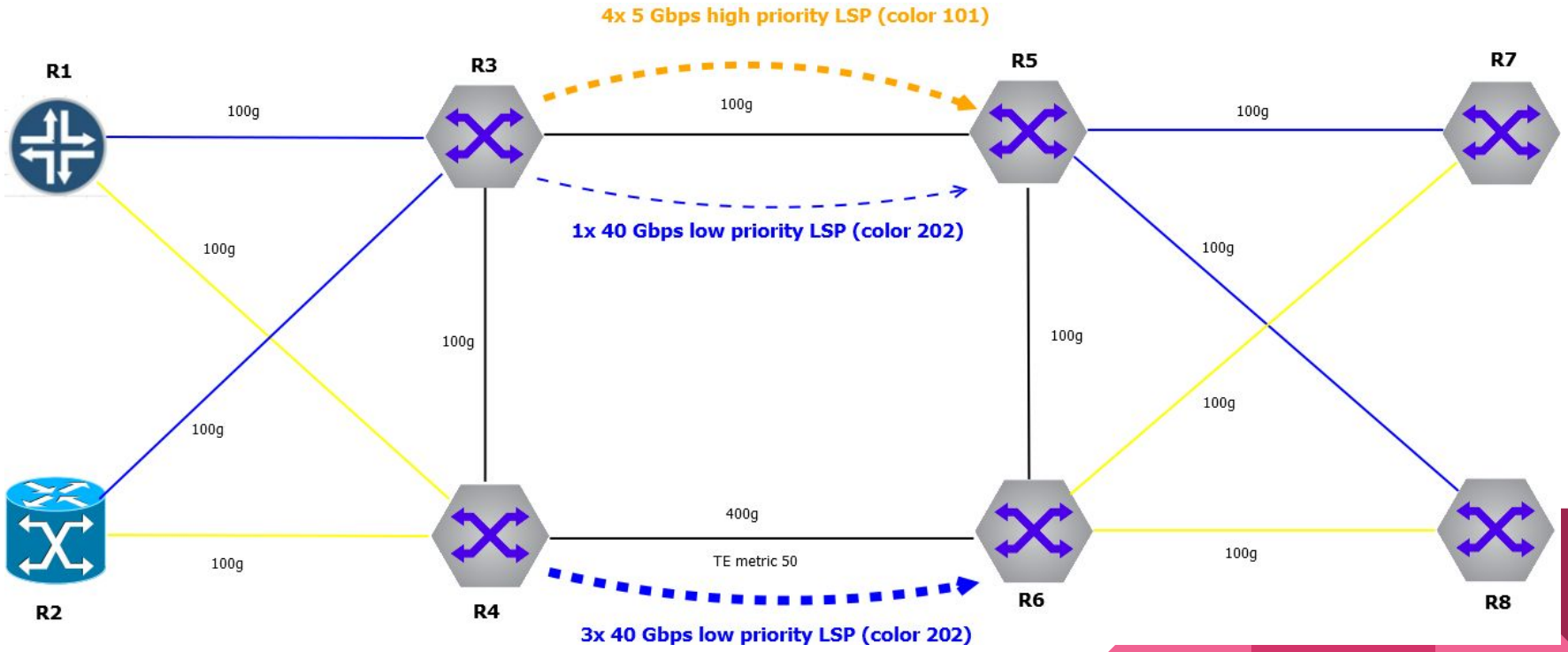
```
traffic-eng capacity-profiles
   !
   profile AUTO_BW
      bandwidth auto
!
traffic-eng mesh-templates
   !
   template TOPO101_BLUE_AUTOBW
      topology-id 101
      color 101
      access-list ipv4 ALL_PE_IPV4
      install indirect srte peer-group RR
      !
      candidate-path preference 100
         metric te
         affinity-set BLUE_ONLY
         capacity-profile AUTO_BW
```

```
lmk-vm102-dev-td1#show traffic-eng mesh-template policies mesh_h2.2.2.2_e8.8.8.8_c101 detail
Detailed traffic-eng policy information:

Traffic engineering policy "mesh_h2.2.2.2_e8.8.8.8_c101"

    Generated from mesh-template TOPO101_BLUE_AUTOBW
--------------snip------------
    Bandwidth type: auto
    Reserved bandwidth: 29.783 Gbps
--------------snip------------
    Candidate paths:
        Candidate-path preference 100
--------------snip------------
            This path is currently active

    Calculation results:
        Aggregate metric: 90
        Topologies: ['101']
        Segment lists:
            [900003, 900004, 900006, 900005, 900008]
```

```
BGP routing table entry for
[SP][SR][I0][N[c65001][b0][q2.2.2.2][2.2.2.2]][C[po2][f0][e8.8.8.8][c101][as650
01][oa2.2.2.2][di100]]
NLRI Type: sr_policy
Protocol: None
Identifier: 0
Local Node Descriptor:
      AS Number: 65001
      BGP Identifier: 0.0.0.0
      BGP Router Identifier: 2.2.2.2
      TE Router Identifier: 2.2.2.2
SRTE Policy CP Descriptor:
      Protocol origin: SR Policy
      Flags: 0
      Endpoint: 8.8.8.8
      Color: 101
      AS Number: 65001
      Originator Address: 2.2.2.2
      Discriminator: 100
Paths: 2 available, best #1
  Last modified: January 30, 2026 18:25:02
  Local
    10.10.10.203 from 10.10.10.203 (100.2.2.2)
      Origin igp, metric 0, localpref 100, weight 0, valid, internal, best
      Link-state: SRTE Bandwidth rate bps 29783246848
```

```
lmk-vm102-dev-td1#show traffic-eng mesh-template policies
Traffic-eng policy information per mesh-template
-----
Mesh-template: TOPO101_BLUE_AUTOBW
Status codes: * valid, > active, s - admin down
       Policy name                              Headend          Endpoint         Color            Protocol          Reserved bandwidth      Priority   Status/Reason
    *>  mesh_h8.8.8.8_e7.7.7.7_c101             8.8.8.8          7.7.7.7          101              SR-TE/indirect    30.167 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h7.7.7.7_e8.8.8.8_c101             7.7.7.7          8.8.8.8          101              SR-TE/indirect    30.161 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h1.1.1.1_e8.8.8.8_c101             1.1.1.1          8.8.8.8          101              SR-TE/indirect    31.015 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h8.8.8.8_e2.2.2.2_c101             8.8.8.8          2.2.2.2          101              SR-TE/indirect    30.311 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h1.1.1.1_e7.7.7.7_c101             1.1.1.1          7.7.7.7          101              SR-TE/indirect    29.849 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h7.7.7.7_e2.2.2.2_c101             7.7.7.7          2.2.2.2          101              SR-TE/indirect    30.860 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h2.2.2.2_e8.8.8.8_c101             2.2.2.2          8.8.8.8          101              SR-TE/indirect    29.783 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h8.8.8.8_e1.1.1.1_c101             8.8.8.8          1.1.1.1          101              SR-TE/indirect    29.886 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h1.1.1.1_e2.2.2.2_c101             1.1.1.1          2.2.2.2          101              SR-TE/indirect    30.326 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h7.7.7.7_e1.1.1.1_c101             7.7.7.7          1.1.1.1          101              SR-TE/indirect    29.838 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h2.2.2.2_e7.7.7.7_c101             2.2.2.2          7.7.7.7          101              SR-TE/indirect    30.085 Gbps(auto)       7/7        Active/Installed
    *>  mesh_h2.2.2.2_e1.1.1.1_c101             2.2.2.2          1.1.1.1          101              SR-TE/indirect    30.009 Gbps(auto)       7/7        Active/Installed
```

Scenario 2: two meshes with different priority



Scenario 2: config and outputs

```
traffic-eng mesh-templates
   !
   template TOPO101_AUTO_BW_HIGH_PRIORITY
      topology-id 101
      color 101
      priority 5 5
      access-list ipv4 ALL_PE_IPV4
      access-list ipv6 DENY_IPV6
      install indirect srte peer-group RR
      !
      candidate-path preference 100
         metric te
         affinity-set BLUE_ONLY
         capacity-profile AUTO_BW
   !
   template TOPO101_AUTO_BW_LOW_PRIORITY
      topology-id 101
      color 202
      access-list ipv4 ALL_PE_IPV4
      access-list ipv6 DENY_IPV6
      install indirect srte peer-group RR
      !
      candidate-path preference 100
         metric te
         capacity-profile AUTO_BW
```

- High-priority template for latency-sensitive traffic
- The other template has default priority 7 (lowest)
- SR-TE color is used to map data traffic to the relevant policies, so that different services on the same PE can have different treatment

Scenario 2: config and outputs (cont.)

```
lmk-vm102-dev-td1#show traffic-eng mesh-template policies
Traffic-eng policy information per mesh-template
-----
Mesh-template: TOPO101_AUTO_BW_HIGH_PRIORITY
Status codes: * valid, > active, s - admin down
       Policy name                          Headend          Endpoint         Color            Protocol          Reserved bandwidth       Priority   Status/Reason
    *>  mesh_h7.7.7.7_e8.8.8.8_c101         7.7.7.7          8.8.8.8          101              SR-TE/indirect    5.086 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h2.2.2.2_e8.8.8.8_c101         2.2.2.2          8.8.8.8          101              SR-TE/indirect    4.989 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h2.2.2.2_e7.7.7.7_c101         2.2.2.2          7.7.7.7          101              SR-TE/indirect    5.038 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h7.7.7.7_e1.1.1.1_c101         7.7.7.7          1.1.1.1          101              SR-TE/indirect    4.999 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h1.1.1.1_e2.2.2.2_c101         1.1.1.1          2.2.2.2          101              SR-TE/indirect    5.002 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h7.7.7.7_e2.2.2.2_c101         7.7.7.7          2.2.2.2          101              SR-TE/indirect    4.894 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h8.8.8.8_e1.1.1.1_c101         8.8.8.8          1.1.1.1          101              SR-TE/indirect    5.081 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h8.8.8.8_e7.7.7.7_c101         8.8.8.8          7.7.7.7          101              SR-TE/indirect    5.083 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h8.8.8.8_e2.2.2.2_c101         8.8.8.8          2.2.2.2          101              SR-TE/indirect    4.949 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h1.1.1.1_e8.8.8.8_c101         1.1.1.1          8.8.8.8          101              SR-TE/indirect    5.056 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h2.2.2.2_e1.1.1.1_c101         2.2.2.2          1.1.1.1          101              SR-TE/indirect    5.101 Gbps(auto)         5/5        Active/Installed
    *>  mesh_h1.1.1.1_e7.7.7.7_c101         1.1.1.1          7.7.7.7          101              SR-TE/indirect    5.003 Gbps(auto)         5/5        Active/Installed
Mesh-template: TOPO101_AUTO_BW_LOW_PRIORITY
Status codes: * valid, > active, s - admin down
       Policy name                          Headend          Endpoint         Color            Protocol          Reserved bandwidth       Priority   Status/Reason
    *>  mesh_h1.1.1.1_e8.8.8.8_c202         1.1.1.1          8.8.8.8          202              SR-TE/indirect    40.954 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h2.2.2.2_e8.8.8.8_c202         2.2.2.2          8.8.8.8          202              SR-TE/indirect    40.049 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h8.8.8.8_e2.2.2.2_c202         8.8.8.8          2.2.2.2          202              SR-TE/indirect    39.991 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h1.1.1.1_e7.7.7.7_c202         1.1.1.1          7.7.7.7          202              SR-TE/indirect    40.873 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h1.1.1.1_e2.2.2.2_c202         1.1.1.1          2.2.2.2          202              SR-TE/indirect    40.403 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h8.8.8.8_e7.7.7.7_c202         8.8.8.8          7.7.7.7          202              SR-TE/indirect    39.728 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h7.7.7.7_e8.8.8.8_c202         7.7.7.7          8.8.8.8          202              SR-TE/indirect    40.380 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h7.7.7.7_e1.1.1.1_c202         7.7.7.7          1.1.1.1          202              SR-TE/indirect    40.239 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h2.2.2.2_e1.1.1.1_c202         2.2.2.2          1.1.1.1          202              SR-TE/indirect    39.433 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h2.2.2.2_e7.7.7.7_c202         2.2.2.2          7.7.7.7          202              SR-TE/indirect    39.860 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h7.7.7.7_e2.2.2.2_c202         7.7.7.7          2.2.2.2          202              SR-TE/indirect    39.433 Gbps(auto)        7/7        Active/Installed
    *>  mesh_h8.8.8.8_e1.1.1.1_c202         8.8.8.8          1.1.1.1          202              SR-TE/indirect    38.796 Gbps(auto)        7/7        Active/Installed
```

Redundancy and failover

- The solution is based on open-standards (BGP-LS, BGP-SRTE), hence it's robust, easy to understand, and fails in predictable ways
- SR-TE sampler doesn't need to be redundant (running multiple samplers writing to the same database leads to unnecessary complexity)
- If sampler fails, just restart it on another machine. Worst case - there won't be traffic rate updates for a few minutes (hours? days? who cares?)
- PCE - deploy as many as you want, no proprietary state-sync is needed (think of route reflector)
- Each PCE is configured with different SRTE distinguisher, so each router receives multiple copies of the same SR-TE policy NLRI from different PCE
    - If one is withdrawn, router just uses another one
- Catastrophic failure of all PCE: traffic just follows the shortest path

SR-TE auto bandwidth - conclusion

- Simple, standard-based solution without vendor lock-in
- Minimalistic, self-updating config, turn on once and forget
  - Intent-based networking done the right way
- Routing state in the network is guaranteed to be deterministic (unlike RSVP-TE)
- We can define different TE requirements for different services (e.g. some services have priority)
- BGP-SRTE scales extremely well, deploy as many LSP as you want
- SR-TE sampler is open source, <github link TBD>
- Traffic Dictator is not open source but a free version is available for evaluation and studying

SR-TE auto bandwidth - next steps

- This is a prototype for now, I want to gauge interest and what features are in demand
- It is possible to implement LSP splitting/merging (TE++/container LSP)
  - Caveat - LSP will use multiple segment lists, this can hit hardware limitations
- Thanks to the centralized view, we can detect not only traffic changes per LSP but also correlate it to the traffic in the entire network (e.g. react on shifting BGP destinations)
- It is possible freeze/rollback BW state for troubleshooting purposes
- It is possible to run various analytics
  - Spot long-term patterns, show overall network utilization and recommend to upgrade certain links
  - Give recommendations to add a new link, e.g. there is a lot of traffic between routers A and B but they are not directly connected

Questions?