Ciso Transport Slice Automaiton Design Guide V1.0

First Published: 2023-12-17
Last Modified: 2023-2-06

Author : John Mullooly- Cisco Distinguished Architect

1 Introduction

Many of our Service Provider (SP) customers have been asking Cisco for our vision, strategy, and plans for Transport Network Slicing and specifically the automation of transport slices. Typically, this question is asked in the context of end-to-end (E2E) slicing for Mobility based 5G use-cases, which include much more than just the transport domain. Customers want to understand how Cisco's Transport Slicing Solution would fit into an overall end-to-end 5G slicing business architecture (which would also include 5GCore, DC and RAN domains). This design document describes Cisco's Transport Slicing Automation Solution which is planned to be delivered with the Crosswork Network Controller (CNC) 6.0 release, which also includes enabling functionality delivered in Cisco's Network Services Orchestrator (NSO) product as part of the NSO T-SDN Core Function Pack (CFP). While 5G mobility transport services are a critical use-case (fronthaul, midhaul, backhaul), Cisco's Transport Slicing Solution is generic enough to satisfy any transport service requiring "intent based" services delivered via an SDN controller provisioning experience for an intelligent IP/MPLS/SR/PCE network, provisioning services from Provider Edge (PE) device to Provider Edge device.

This document outlines the steps and logic that a CNC user would perform to both design new "slice types" (build the slice-type catalog (i.e., slice templates)) and also how to request new "slice instances" using the intent-based slice UI, CLI or API interface. This document goes into detail on both the northbound abstracted IETF slicing YANG model and the underlying machinery Cisco recommends being available in the network in order to realize transport slicing services. For the "Slice Designer", it will be critical they have a network engineering background (including a good understanding of their network's own capabilities) and understand the recommended technology underpinnings used to realize the slicing services in order to build the slice catalog. While for the "Slice Requestor", only a simple, declarative, "intent based" understanding is required to deploy a slice instance.

We also try to give the reader an understanding of how we made our design decisions and also what new capabilities we hope to provide in future releases.

2 Background

2.1 What is 5G Network Slicing and how does Transport Slicing fit in?

5G network slicing is a network architecture that enables the multiplexing of virtualized and independent logical networks on the same physical network infrastructure. Each network slice is an isolated end-to-end network tailored to fulfil diverse requirements requested by a particular application.

For this reason, this technology assumes a central role to support 5G mobile networks that are designed to efficiently embrace a plethora of new services with very different service level requirements. The realization of this service-oriented view of the network leverages on the concepts of software-defined networking (SDN) that allow the implementation of flexible and scalable network slices on top of a common network infrastructure.

In commercial terms, network slicing allows a mobile operator to create specific virtual networks that cater to particular clients and use cases. Certain applications - such as mobile broadband, machine-to-machine communications (e.g., in manufacturing or logistics), or smart cars - will benefit from leveraging different aspects of 5G technology. One might require higher speeds, another low latency, and yet another access to edge computing resources. By creating separate slices that prioritize specific resources, a 5G operator can offer tailored solutions to particular industries. Some sources insist this will revolutionize industries like marketing, augmented reality, business services or mobile gaming.

The benefits are self-evident, since a single network can be divided to cover diverse use cases based on customer demand and segmentation. Operators can then allocate resources to each slice, utilizing the necessary speed, throughput and latency to cover the breadth of network slicing in 5G. The ability to offer a network slice as a service minimizes operating expenses (OPEX) as well as capital expenditures (CAPEX). It can allow critical public entities, such as first responders and medical emergency teams, to be prioritized with respect to coverage, capacity and connectivity.

As described in the Introduction, a 5G Network slice is an end-to-end solution crossing multiple sub-domains. Cisco's Transport Slicing solution only addresses the transport domain given our strong set of products and solutions in this space. ***It is out of scope of this document to explain the E2E solution or how other domains will be sliced*** (which is unique to the vendors and technology options within those domains). 3GPP calls the E2E 5G Network Slice the "Network Slice Instance" (NSI) which is orchestrated by an E2E orchestrator called the Network Slice Management Function (NSFM). This slice "instance" is a unique slice provisioned in the network but with a set of service level requirements chosen from a set of pre-created Network Slice Templates (NST). The NSMF in turn communicates with each sub-domain controller, called a Network Slice Subnet Management Function (NSSMF) which in turn provisions the corresponding domain specific slice instance across its own sub-domain boundaries (called a Network Slice Subnet Instance (NSSI)) using a similar set of domain specific Network Slice Subnet Templates (NSST). For Cisco, we are developing the NSSMF (CNC R6.0) function for the transport domain and aligning to IETF slicing standards in defining the data-model interface between the NSMF and transport NSSMF dictating how a transport slice is requested and managed. The IETF defines the transport slice as a "IETF Network Slice" in order to differentiate the transport slice from other non-transport slice domains. The remainder of this document will focus only on the transport domain solution.

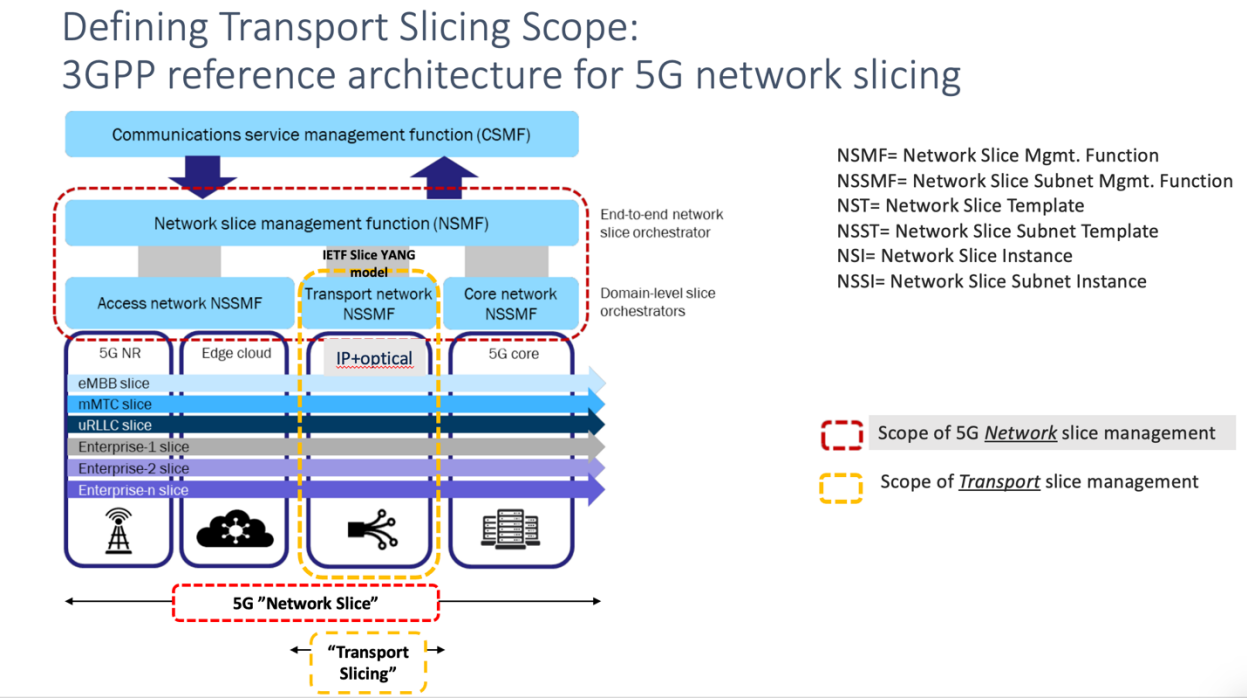

Figure 1: Scope of Transport Slicing

2.2 Intent based Service abstraction with SLOs and SLEs

Some lessons we could all learn from the cloud operators are that simplicity, ease of use and "on demand" are now expected behaviors for any new service offering. The cloud operators built their services with modular principles and well abstracted service interfaces using common "black box" software programming fundamentals, which allow their capabilities to snap together very seamlessly, while hiding unnecessary complexity. For many of us in the SP telecom industry those basic principles still need to be realized in how transport domain service offerings are requested from the transport orchestration layer. Work is being done by Cisco with Transport Slicing to exactly define this layer of service abstraction into a simplified, yet extensible, transport services model allowing for powerful network automation. The slice instance user simply wants to request an "intent" (or call it an "outcome") and they expect the network service to then be built (and monitored) to honor that intent, based on quantifiable Service Level Objectives (SLOs) and promised Service Level Expectations (SLEs). The network service requestor wants to avoid exposure to the plethora of configuration parameters required to actually deploy that service at the device layer. Embracing such a basic principle would not only reduce the cost of operations but also enable new "as-a-Service" business models which could monetize the network for the operator. But realizing the vision requires the creation of intent-based modularity for the value-added transport services via well-abstracted and declarative service layer APIs. These service APIs would be exposed by an intelligent transport slice orchestration "controller" that can act in a declarative and "outcome" based way.

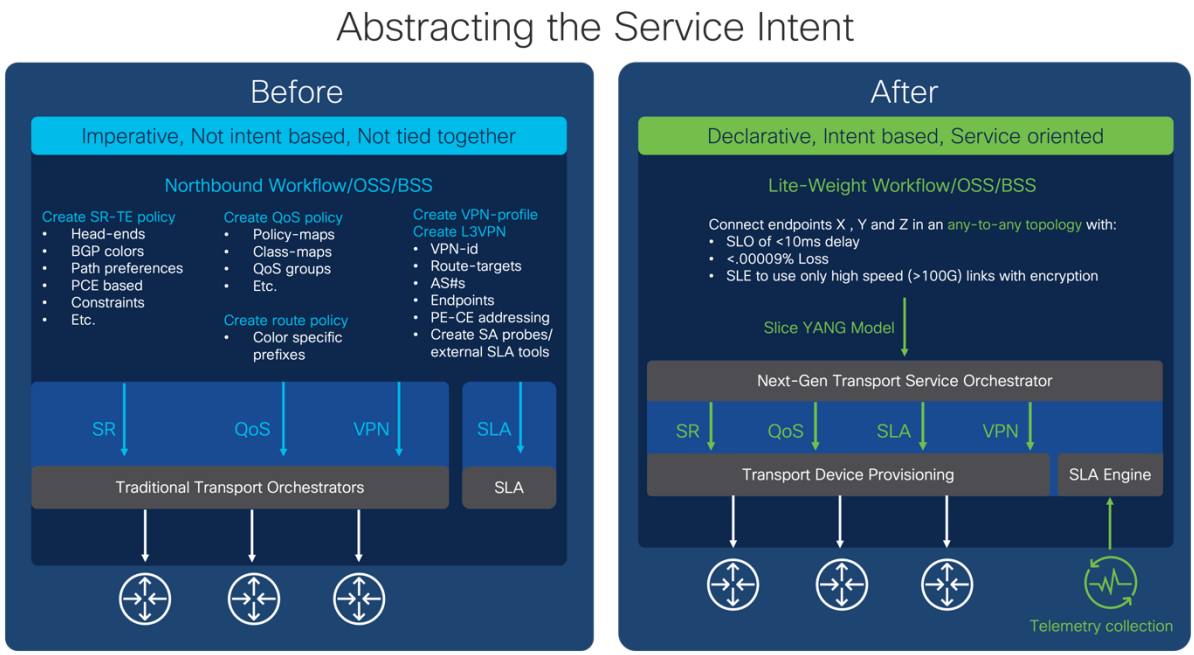

Figure 2: Abstracting Service Intent

A ***Service Level Indicator*** (SLI) is a quantifiable measure of an aspect of the performance of a network. For example, it may be a measure of throughput in bits per second, or it may be a measure of latency in milliseconds.

A ***Service Level Objective*** (SLO) is a target value or range for the measurements returned by observation of an SLI. For example, an SLO may be expressed as "SLI <= target", or "lower bound <= SLI <= upper bound".

A ***Service Level Expectation*** (SLE) is an expression of an unmeasurable service-related request that a customer makes of the provider. An SLE is distinct from an SLO because the customer may have little or no way of determining whether the SLE is being met, but they still contract with the provider for a service that meets the expectation (see the below table of sample SLEs).

A ***Service Level Agreement*** (SLA) is an explicit or implicit contract between the customer of an IETF Network Slice service and the provider of the slice. The SLA is expressed in terms of a set of SLOs and SLEs that are to be applied for a given connectivity construct between a sending endpoint and the set of receiving endpoints and may describe the extent to which divergence from individual SLOs and SLEs can be tolerated, and commercial terms as well as any consequences for violating these SLOs and SLEs.

Encrypted Link Services	Only transit encrypted links
Disjoint Path Services	Network has multiple forwarding planes with no common nodes/links
High speed links only	Only transit high speed links (say >=100Gbs, typically for elephant flows)
Lowest Latency	Always take lowest latency path (no SLO given)
Regional Avoidance	Do not use nodes/link in specific regions
Trusted Nodes	Only use trusted nodes (verified and not in common carrier space)
L4-L7 Services	Redirect to "in-line" L4-L7 service on traffic (typically security services)
Reliable Links	Only transit links that have optical protection and L1 diversity
"Circuit-Style" Services	Provide L1 "circuit" like connectivity
Gaming Services	Network is optimized for network gamers (latency, BW)
Connected Car	Network is optimized for network connected cars (latency, proximity)
Cloud Provider XYZ	Connect me to secure "walled garden" for cloud provider (AWS, Azure...)

Figure 3: Sample SLE offerings

3 Cisco Transport Slicing Explained

3.1 What makes up a Cisco Transport Slice

As defined above, the actual slice service is based on "intents" and is highly abstracted for the slice user. Yet, these intents need to be translated by the transport slice controller (CNC) into actual device configurations which can provide the actual transport slice services. Neither the IETF nor 3GPP define the actual transport network machinery (device level features and protocols) required to deliver on the slice intent. That decision is left to the vendors and the network providers. In Cisco's case we have a complete toolkit we can leverage which was built on many years of innovation.

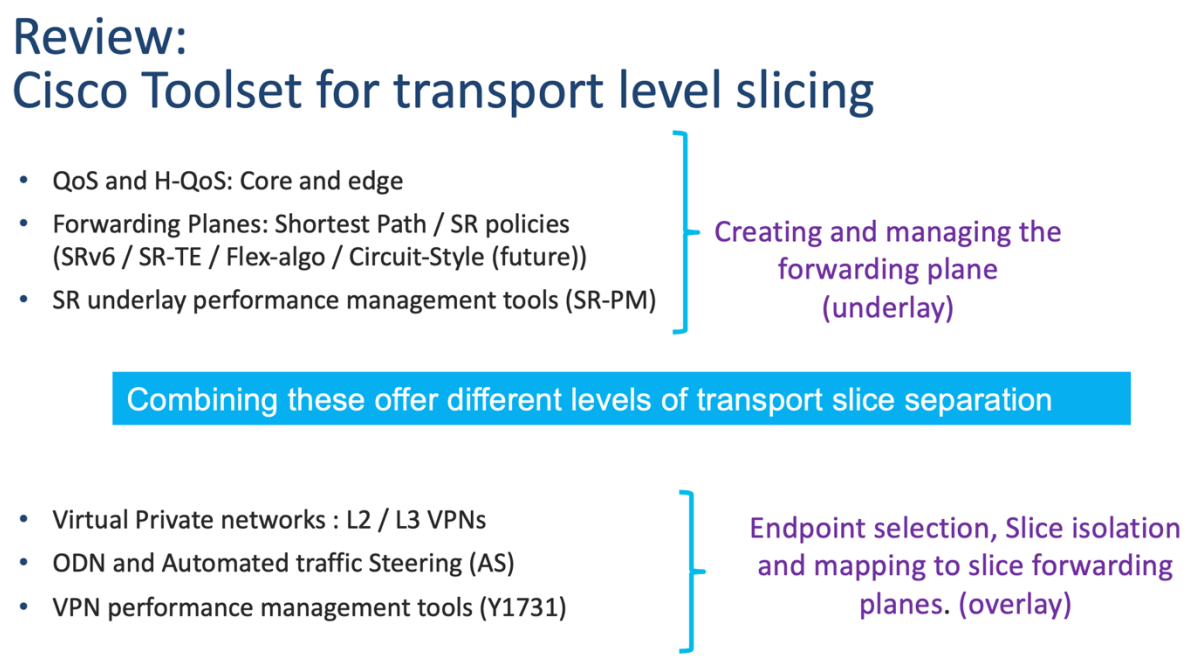

Figure 4: Cisco Networking Technology Toolkit

For a Cisco Transport Slice Instance, we combine the above features into four categories (see figure below), of which three will be defined in the slice template catalog (i.e., the NSST). The slice catalog contains various slice templates (also called slice types or blueprints) and is built once by an administrator who is the overall Slice Designer. These templates can be referenced many times for different slice instances. It is important to understand the difference between slice templates and slice instances. Slice templates are just blueprints and are not instantiated in the network in any way, slice instances are deployed transport slices in the network. The end-user really doesn't need to know the details of what is inside the templates, just what the overall service agreement is for that template. Thus, the catalog is considered a catalog of slice intents. The fourth category that makes up a Cisco Transport Slice instance is the actual customer-facing endpoint selections, which are defined when deploying the slice instance itself and utilize BGP-based VPN technology under-pinning.

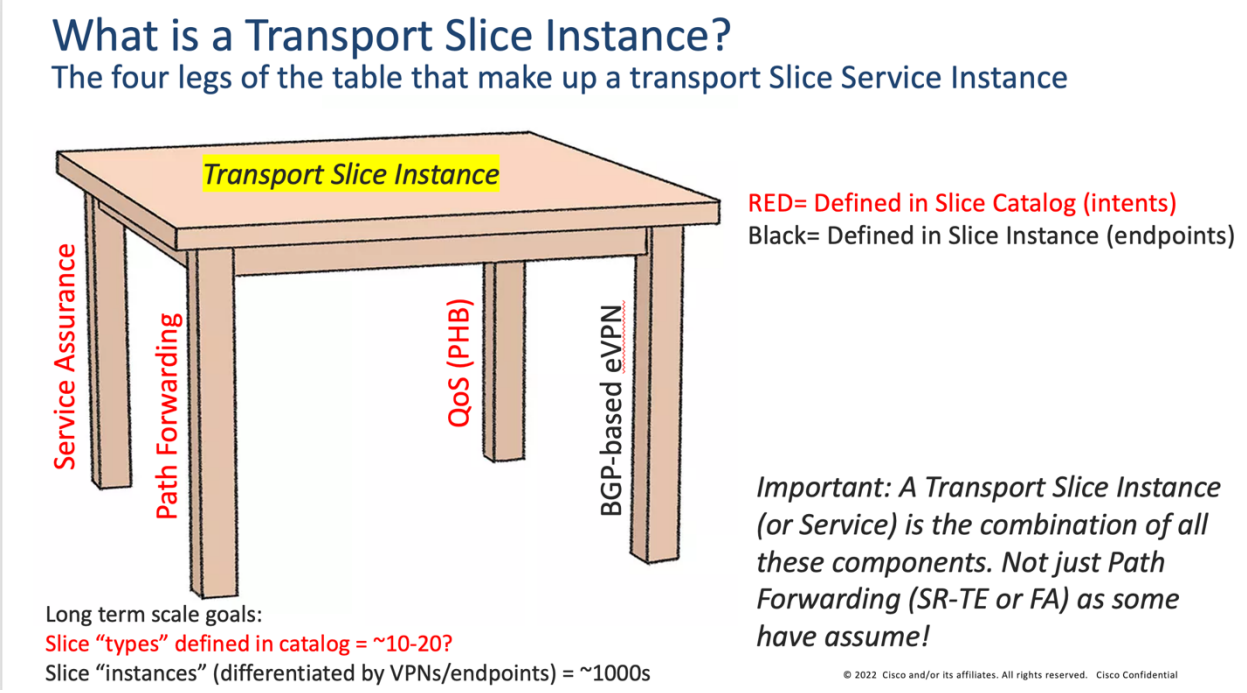

Figure 5: Cisco Slicing Components.

The benefit of our approach is to fully abstract the underlying configuration details of the various machinery components required to realize a Transport Slice Instance (IETF Network Slice or also called a Network Subnet Slice Instance (NSSI) by 3GPPP). To deploy a new slice the following steps are executed:

1. The user facing data model (customer-facing) requests an "intent" and "outcome" by referencing the pre-created template catalog.
2. The user selects the desired slice endpoints and connectivity details (any-to-any, point-to-point) and commits the request.

   The slice template referenced in step 1 abstracts the details of the:
   - The forwarding plane policy which drive the Segment Routing Traffic Engineering (SR-TE) configurations and BGP prefix coloring for On-Demand Next-Hop with Automated Steering (ODN/AS)[1].
   - The QoS profile details drive the ingress marking (for PHB treatment) and the egress scheduling.
   - The SLA details will drive the needed Service Assurance configurations.

[1] A Cisco SR Innovation

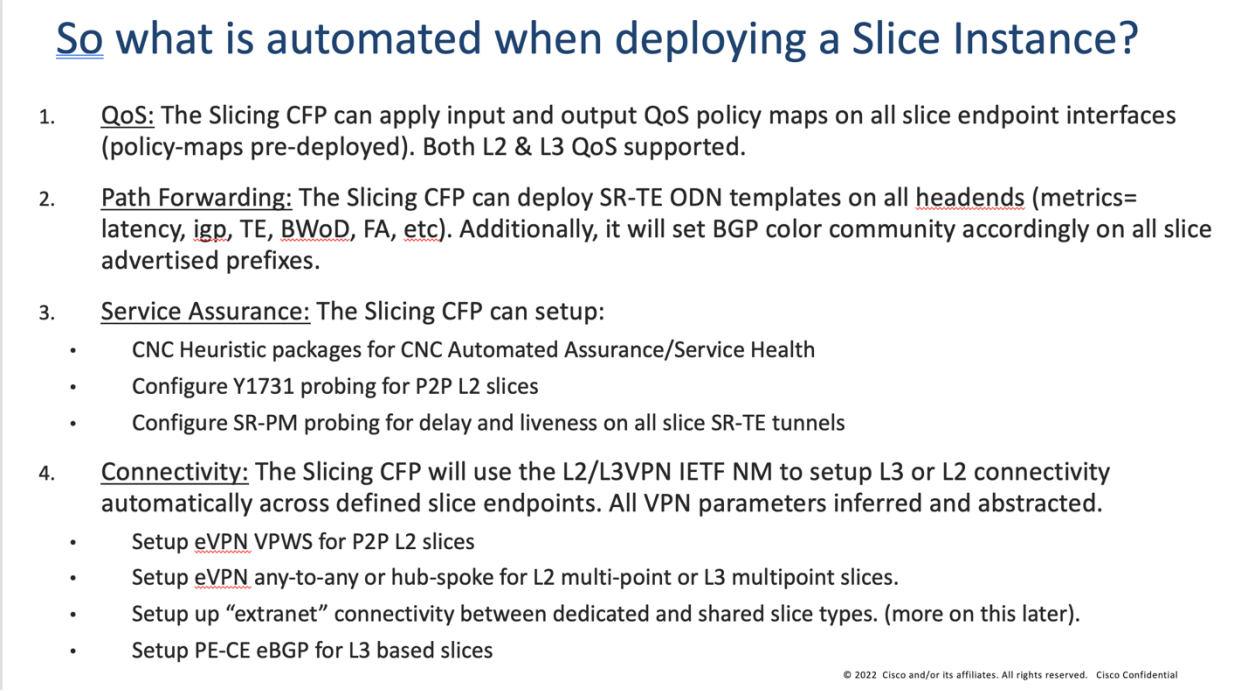

Figure 6: Cisco Slicing Automation Workflow

3.2 Slice User Personas

We will describe the solution from the view of two slice user "personas" in CNC:

- The "Slice Designer" persona (also can be considered the slice administrator) who understands the transport provider's business slice service requirements desired to be offered by the controller and who is very familiar with the network's underlying capabilities. This person has authorization to do one-time setup operations within the network and has a networking engineering background. They will setup the needed network pre-requisites and then build the slice template catalog, which offers a listing of available slice service offerings for slice users that can be monetized by the provider.
- The "Slice Requestor" persona are users that request new slice instances via an intent-based and simplified slice user interface. They pick their desired slice-type/template from the pre-built slice catalog, select their endpoints and click submit.

Our objective in the Cisco Transport Slice Solution is to make the user experience as simple as possible for the Slice Requestors as this is the actual slice deployment operation driving the network service provisioning and it's done constantly for a large SP network. On the other hand, the slice template catalog creation is done once, by skilled personnel (the Slice Designer). The catalog creation requires a good understanding of the network capabilities and requires various pre-requisite configuration that will be described below. Automation of the slice catalog creation is not a primary objective at this time.

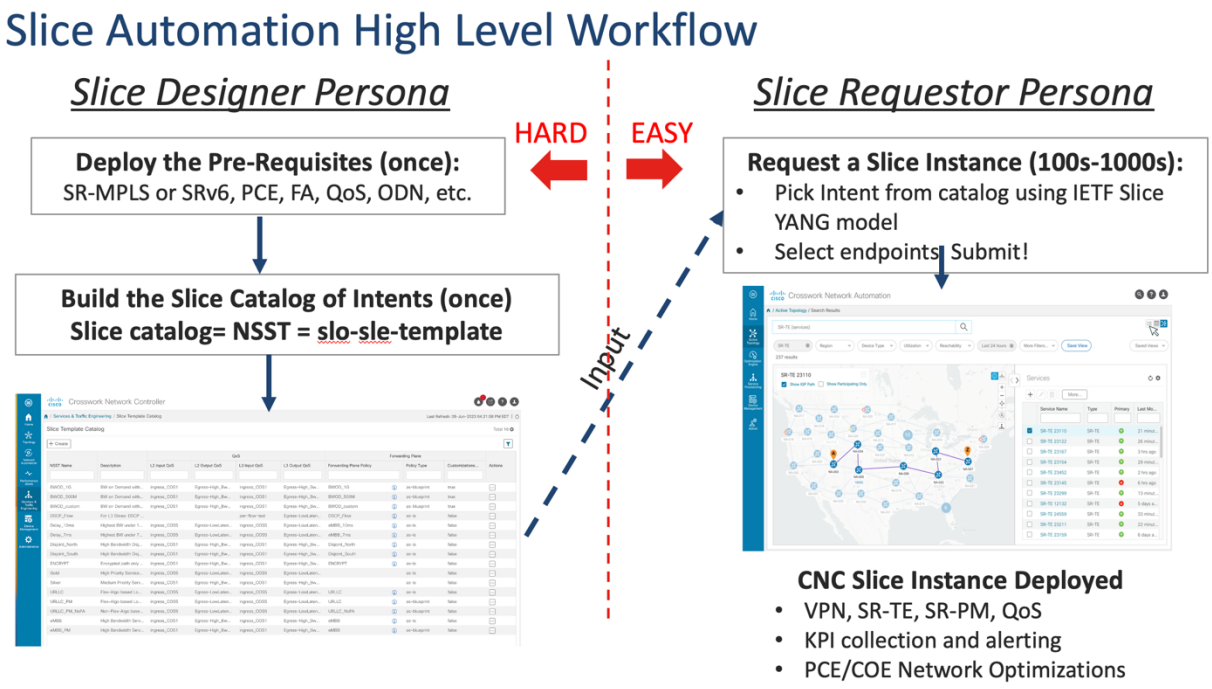

Figure 7: Slice User Personas

3.3 Cisco Transport Slice Design Details

3.3.1 Template "Catalog" for slice intents

The architecture is focused on a catalog-based solution where the operator/designer pre-builds a static catalog of different slice "types". The 3GPP community has a similar solution that calls this a Network Subnet Slice Template (NSST), while the IETF calls it an "slo-sle-template". The operator will pre-build these slice types to meet specific SLOs and SLEs that their network supports. Then, when the slice requester (user) desires a specific slice service be instantiated, they would pick a specific slice type from the catalog to match their needs, along with adding the endpoints (Service Demarcation Points or SDPs) that they wish to connect. The slice catalog will contain entries defining the QOS policy, forwarding plane policy, performance-measurement policies and whether to allow for per-slice customizations (i.e., BW settings, etc.). These details can be hidden by the operator.

3.3.2 Leverage SR-TE machinery at an abstracted level

This is a topic that needs to be well understood by the Slice Designer but not necessarily the Slice Requestor. A key piece to our solution will be the re-use of much of the Cisco invented Segment Routing Traffic Engineering (SR-TE) machinery to drive the forwarding plane policy to meet the desired SLOs/SLEs. While slices can be built without SR-TE, the overall value to the user is minimized as the surgical path forwarding behavior of SR-TE is a key piece of driving the SLO/SLE. For the initial release we will focus on the use of Cisco's SR On-Demand-Nexthop (ODN) feature with Automated Steering (AS) to map the slice traffic into the desired SR-TE tunnel. (See Figure below).

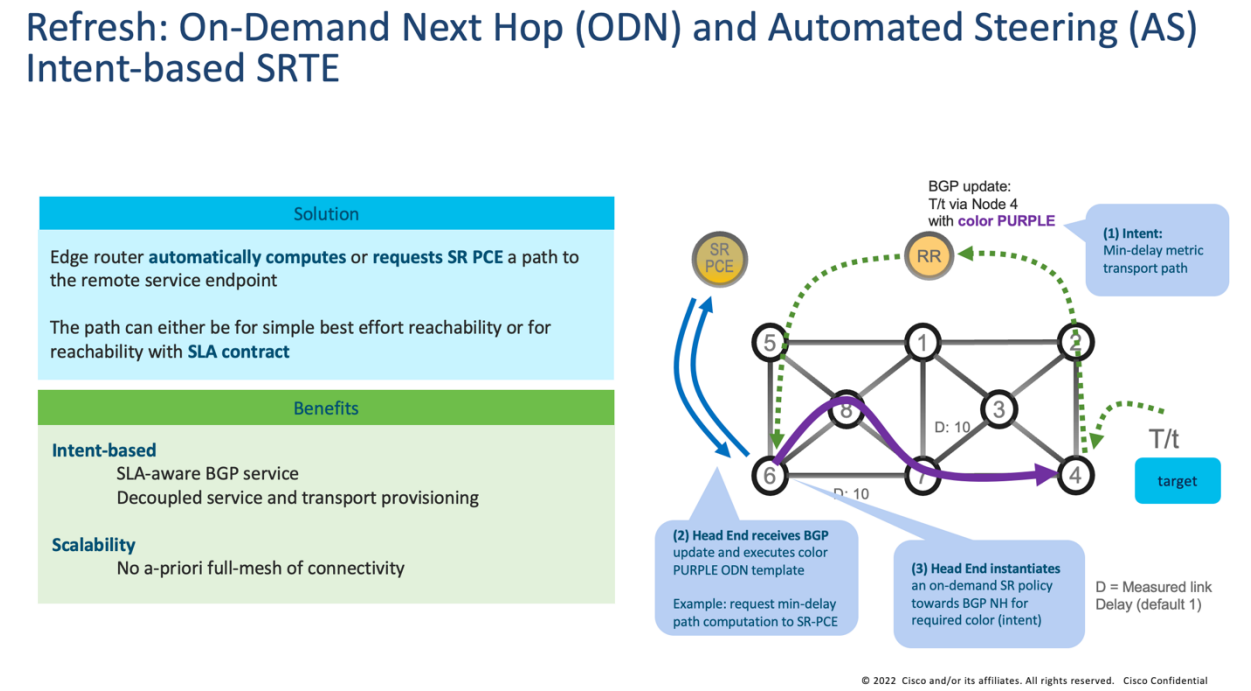

Figure 8: SR-TE ODN/AS Refresh- A Cisco Innovation

Much of the "pre-work" by the designer is focused at creating the right ODN templates to meet the SLO/SLE forwarding objectives and then mapping those into the catalog entries. The slicing system will use the catalog referenced ODN templates accordingly, with the expectation that they have been designed properly by the slice designer/operator to meet a specific SLO/SLE. ***Any feature or capability supported in Cisco's ODN template machinery and supported by the NSO T-SDN sr-te-odn CFP service should be available by the CNC slicing service package since creating the ODN template is done independently from the slicing service code.*** This document cannot adequately cover all the innovate features being developed by the Cisco SR-TE team. But the below figures show a sample of possibilities offering different SLO/SLE slice types.

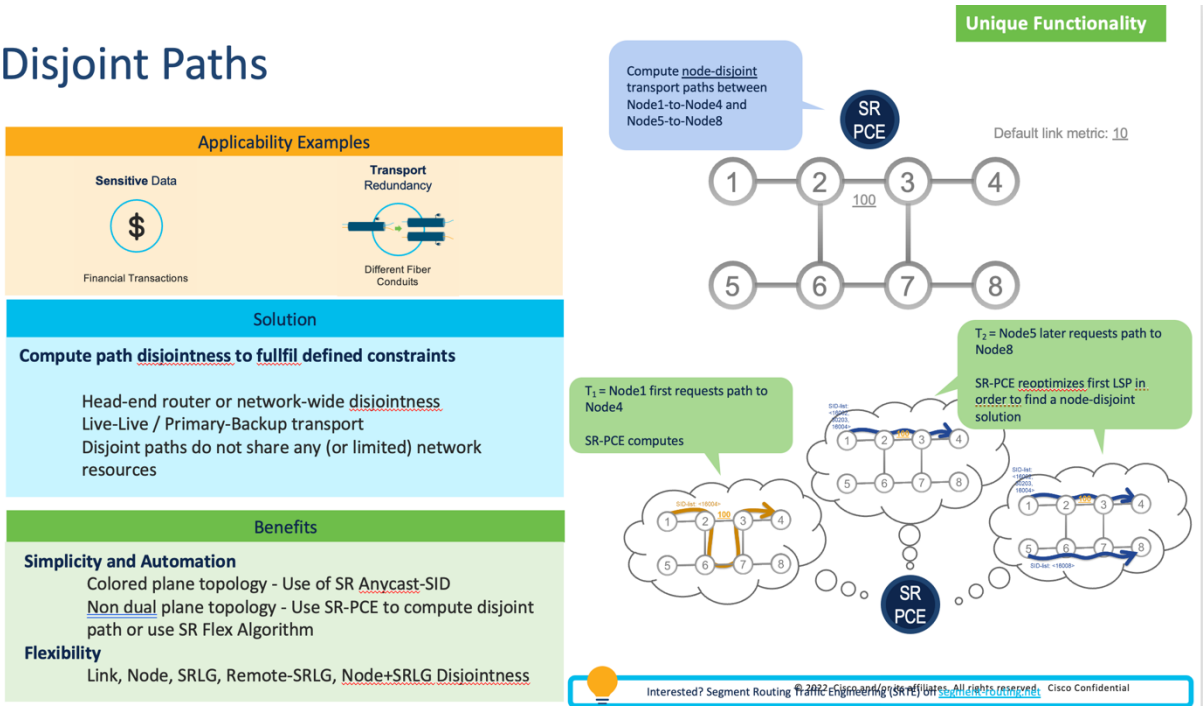

Figure 9: SR-TE Disjoint Paths Intent

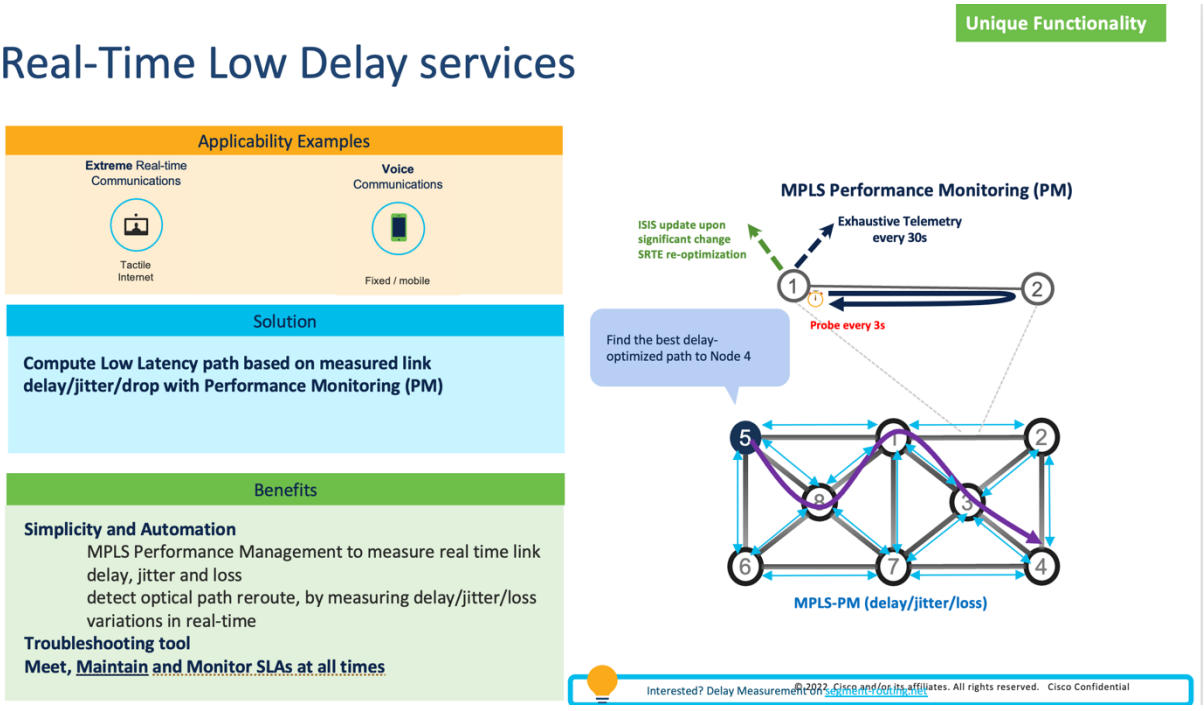

Figure 10: SR-TE Min-Delay Intent

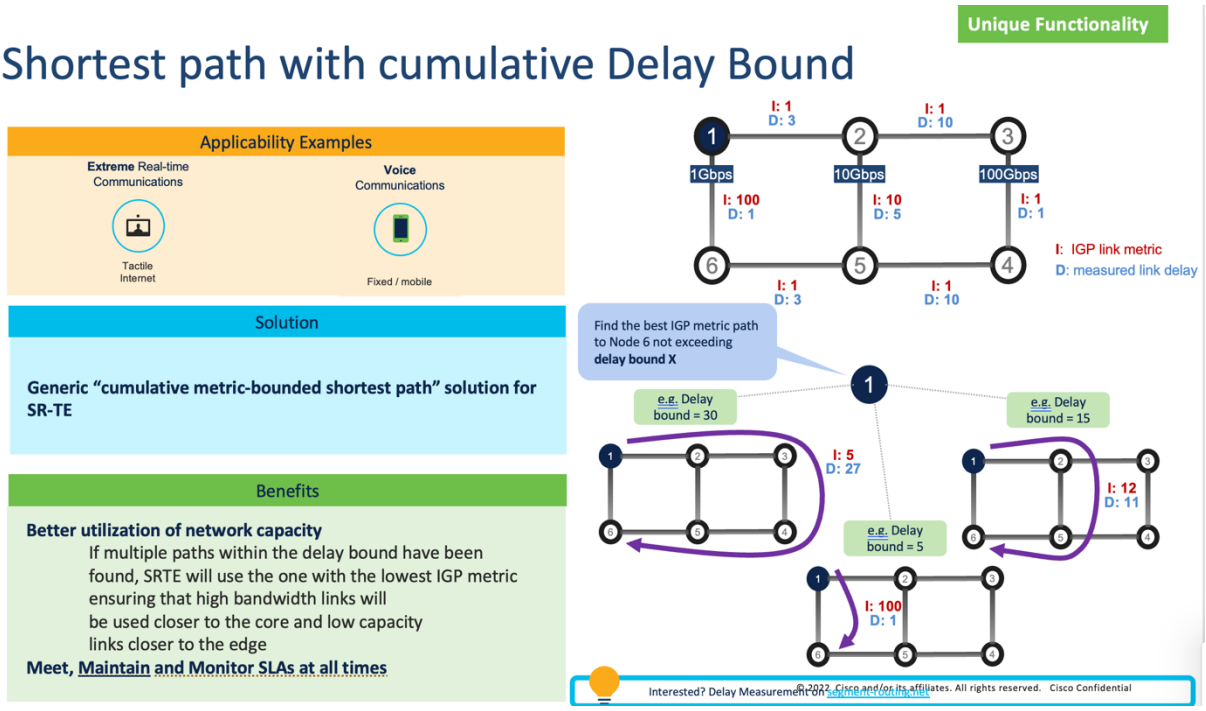

Figure 11: SR-TE Cumulative Constraints Intent

3.3.2.1 SR-TE Scalability and Flexibility with "as-is", "as-blueprint" and "customizations" Template Settings

The scalability desire is for slice service instances that share the same catalog slice entry/intent (without customization enabled) to share the same forwarding plane policies and thus use the same SR-TE tunnels built within the network underlay. This provides a N:1 scalability factor for slice services to underlay SR forwarding behaviors. This is key scalability value proposition for Cisco. An example would be the slice instance Coke and the slice instance Pepsi, where each have unique endpoints on the same two PE devices and they also share the same slice intent which is focused on "minimizing delay". In this case the same SR-TE tunnel will be used for both Coke and Pepsi traffic between the same two PEs which again is a big scalability benefit.

In some use-cases the slice service itself may need to build a new ODN template "on-the-fly" for that slice type or even for a specific slice instance. For example, if any dynamic modifications to the pre-created ODN template are required. Since multiple catalog entries (slice types) can reference the same ODN template via the forwarding plane policy configuration, a dynamic change to the ODN template by one entry could mistakenly affect all entries. To allow for this flexibility, we have introduced a configurable knob during slice template creation that can be set by the Slice Designer to define if the system should use the referenced ODN template for path forwarding "as-is" or "as-blueprint".

- If "as-is" is set (which is the default) then this exact ODN template name and color referenced in the forwarding plane policy entry in the catalog will be used for that slice

type. This includes that exact color specified. All slices service instances referencing this slice type will share that ODN-template/color and thus share any underlying SR-TE tunnels. The user is free to build the ODN template with any sr-te ODN features available with no restrictions as long as they are supported by the sr-te odn CFP. In fact, multiple slice template entries can reference this same "as-is" ODN policy if the same forwarding behavior is required across different templates (slice types).

- If "as-blueprint" is set, then the slice designer is saying use the referenced ODN template as a blueprint. Basically, make a carbon-copy and assign it a unique new color (auto-selected from a pool). This new ODN template will now be available for all slice instances that reference this catalog entry and the benefit is that the slicing service package can make modifications to that ODN template without affecting other catalog entries. For example, if performance measurement is enabled for this catalog entry, the slice service would need to supplement the ODN template with the required SR-PM configurations. Additionally, there are scenarios where a specific slice service instance needs *its own dedicated SR-TE tunnel* due to customizations (i.e., dedicated bandwidth). In that case, not only do we need the "as-blueprint" mode enabled but also have the "customization allowed" flag set.

A bit more on this "customizations" topic as there may be per-slice instance requirements for parameters such as dedicated bandwidth (BW) or special performance measurement thresholds per slice instance. The catalog will allow for the operator to enable a flag to allow for customizations in the slice type. This knob is considered under operator control (via the catalog) to prevent users from trying to customize slice types that are considered fixed or static. The result of setting the custom flag (which first requires the "as-blueprint" to be enabled) is:

- The slicing package will create a new dedicated ODN template (with a new color) based on the blueprint referenced but for each slice instance created (not just one ODN template for the whole slice type). Thus, each slice service instance will have a dedicated SR-TE tunnel. For BWoD slice use cases this is an important requirement.

- Apply any custom attributes that are specified in the slice service instance such as BW or different SR-PM settings (future). The slice service will merge these changes into the new ODN template with these per-service settings.

Why? To allow flexibility for different path-forwarding behaviors across multiple deployed Slice Instances.

Slice Catalog Forwarding Plane Policy type	Behavior and Pros/Cons
As-is	Forwarding-Plane specified ODN template will be used unmodified (same color). Allows for multiple slice catalog entries to share the same ODN template to promote scale of underlying SR-TE infra. All Slice instances across All catalog entries with same "as-is" Forwarding-Plane policy will use same SR-TE infra.
As-blueprint	A new ODN template will be dynamically created (new color) by the slicing package using the the specified ODN template as a baseline. All Slice instances within this catalog entries will use this ODN template (color) and share the same SR-TE infra. This option allows for per catalog entry modifications (typically different SR-PM requirements added by slicing package) but still allowing for scale of infra.
As-blueprint with customization	A new ODN template will be dynamically created (new color) by the slicing package using the the specified ODN template as a baseline FOR EACH Slice instance. All Slice instances within this catalog entries will each have a dedicated ODN template and thus NOT share SR-TE infra. This option is the least scalable but allows for per slice service customizations (typically per-slice BWoD requirements since dedicated SR-TE infra).

Figure 12: Slice Catalog Path Forwarding Policy Types

3.3.2.2 BWoD Slices

It's a very important use-case to support bandwidth-based slices. Our underlying ODN machinery along with PCE and COE do support a bandwidth-based routing use-case by adding the "bandwidth" parameter (with a value) in the ODN template. Nothing precludes the user from adding the BW parameter to the ODN template as part of the ODN template creation, which would then be used in the catalog but for all slices that point to that catalog entry and share the same head-end- and tail-end will share the same SR-TE policies/tunnel, including the BW available.

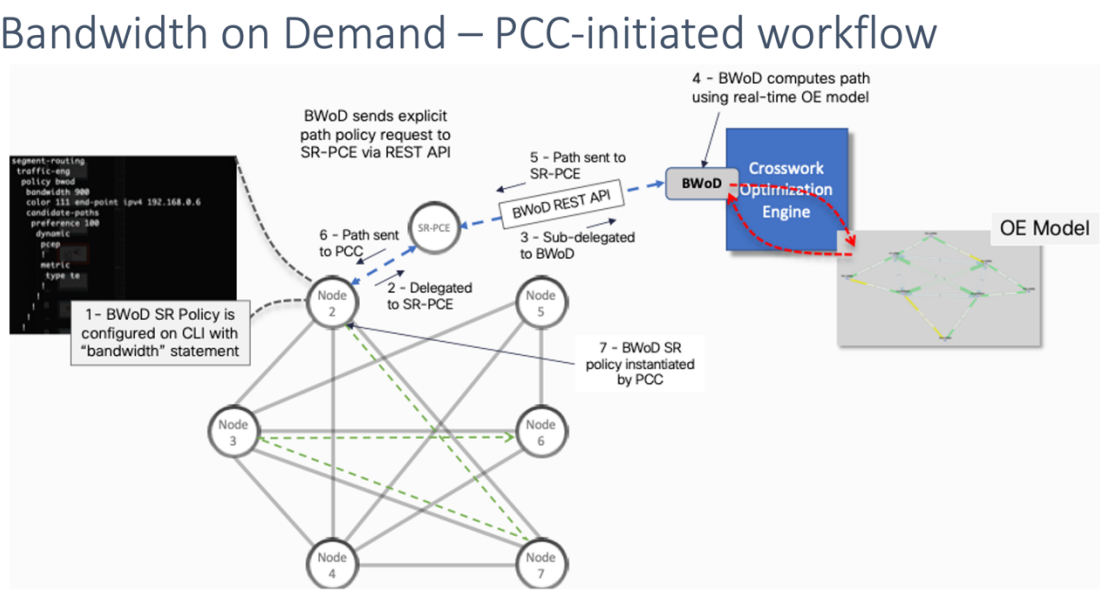

Figure 13: BWoD Intent- Under the Hood.

In the case where a slice instance needs its own dedicated BW between head-end and tail-end, we would need to map only that VPN to that dedicated SR-TE ODN policy. Thus, the team has come up with an approach to signal dedicated SR-TE tunnel creation with the customization parameter.

In the case of the catalog's customization flag being set to enable, the slicing service package will take this as a signal to create a dedicated SR-TE ODN policy and color for this slice. The slice service will use the ODN template specified in the catalog's forwarding plane policy as a blueprint to be used as the basis for the dedicated ODN template policy, such that it will produce a carbon-copy template but change the color to a unique value. The service will pull the unique color from a pre-created resource pool. Additionally, when a user is creating a new slice service and selects a catalog entry with the customizations flag enabled, the slice service code will allow the user (via UI, CLI or API) to set custom attributes which would be a Cisco augmentation to the IETF YANG model. These attributes will include a bandwidth option to be entered when provisioning the slice instance. If the user does not specify a value for these custom attributes, the dedicated SR-TE tunnel will still be created exactly as defined in the corresponding sample ODN template, but with a custom color. If a BW statement is already in the sample ODN template that BW value will be used unless overwritten (i.e., If a new BW is entered in the custom attributes of the slice instance, that BW will overwrite any BW value specified in the sample ODN template.)

3.3.3 Keeping QoS Automation Simplified

QoS policy machinery is a very complex subject and varies for each network, including vendor to vendor and even device type by device type (depending on device capabilities). From an automation perspective, we need to leave the detailed QoS configurations in the hands of the network designers and a slicing design decision has been made for slice automation that the QOS policy maps (both ingress and egress) on the customer-facing PE interfaces will be pre-built and pre-positioned on the devices (not yet on the service interfaces). The CNC slicing service will automatically attach the policies accordingly to the slice endpoint interfaces.

Additionally, since SPs have different QoS policies for L2 versus L3 services, the catalog will allow entry for separate L2 and L3 policies on a per slice template basis (i.e. DSCP vs CoS settings). When the slice template is referenced by a new slice service instance, the CNC slice service code will assign the proper QoS policies depending on the type of service being configured. This eliminates the need for having separate catalog entries for L2 and L3 slice services when the only difference is the QoS required.

3.3.4 Service Assurance

It's important to leverage all the underlying work already done (i.e., CNC Service Health, Heuristic packages, Y1731, SR-PM) that is being delivered in the CNC6.0 Automated Assurance program. So, the CNC slice service will be able to feed the needed SA configuration parameters into these underlying services. As such we will expand the catalog with service assurance needs for the slice type. These include:

- Defining the Service Health heuristic package(s) needs for the slice instance. Catalog Heuristic packages are defined for L3, L2 multi-point (MP) or L2 point-to-point (P2P) and the CNC slicing service will pick the proper heuristic package from the catalog depending on slice instance service type and connectivity.
- Ethernet Service OAM: i.e., Y1731 for L2 P2P slice types (if the slice type is not L2P2P then this catalog entry will be ignored).
- Performance Measurement: SR-PM for either L2 or L3 slice instance service types (any topology) since the implementation will be based on SR-TE based SR-PM which is agnostic to the overlay VPN technology.

Similar to the QoS discussion above, depending on the service type (L2 or L3) and the topology (any-to-any, P2P, etc.) the CNC slice service code will select only the appropriate SA parameters needed from the slice type catalog.

The NSO slice service package will use the referenced catalog entry and depending on the service type (L2 or L3) will configure:

- The proper Heuristic packages for Service Health in the L2/L3 VPN NM YANG model.
- Insert any needed Ethernet Service OAM (Y1731) into the L2NM P2P service model.
- If "as-blueprint" is selected for forwarding policy, insert the referenced SR Performance Measurement (PM) profiles into the proper ODN templates via the sr-te odn-policy CFP. Additionally use the PM CFP to push the needed head-end PM profiles to the desired devices. Optionally, the operator can choose not to configure SR-PM in the catalog but

pre-configure the ODN templates with the desired PM profiles, along with pre-deploying the PM profiles to the head-ends.

For slicing in CNC 6.0, the Service Assurance parameters can be added to the catalog via the slicing API or NSO CLI. The CNC UI will not visualize nor allow for Service Assurance parameter configuration in the slice catalog. The CNC UI will ignore these fields when set by the API and display only the supported fields.

In this approach, we can begin to see how the slicing package can deploy the underlying SA machinery with the slice instance to do the proper measurements (delay, jitter, loss) and then by using system or customized heuristic packages, these results will be retrieved by Service Health (via CDG) and displayed in the Assurance Graph for the service. Any thresholds exceeded (delay, etc) will alarm accordingly and set the service health to degraded. This health state is represented by the underlying VPNs in CNC6.0 and will be represented in the slice health in a future release.

Lastly, since Service Assurance parameters are set at the catalog level for all slice instances sharing that slice type, all slice instances for that slice type will share the same SA settings, including monitoring state. Meaning in CNC6.0 Service Health monitoring cannot be individually turned ON or OFF at the individual slice level (which is allowed when configuring individual VPN services without slicing). Monitoring is set once at the slice catalog level and is a fixed setting for all slices sharing that entry. If monitoring state is changed at the component VPN level for a VPN that was created by the slice service, the monitoring state will be automatically reset to the catalog configured value immediately.

3.3.5 Selecting L2 or L3 Service Type and Auto Building the Connectivity

When requesting a new slice instance, a required step is to indicate the service type (L2 or L3) and the connectivity requirements (i.e., how slice endpoints will communicate together). Cisco has developed an “auto-connectivity” feature for Transport Slicing that allows the user to select from three basic topology options (any-to-any, point-to-point or hub & spoke) and the slice service will automatically setup the connectivity requirements by using intelligent BGP based VPN Route-Target (RT) settings. The CNC UI even simplifies this further by using default settings and pull-down menus.

First when building the slice instance, you will need to select the “service-type” (either L2 or L3 based forwarding). Then, if using the API/CLI, the connectivity-type selected in the connectivity-group portion of the YANG data model will drive the configuration for the various supported topology types (any-to-any, hub-spoke, point-2-point). If using hub/spoke, the endpoints connectivity-role (hub or spoke) is set in match criteria. These settings will help the slice service package build the needed topology and assign the required BGP-VPN based Route Target configurations.

- RT selection can either be set to either “auto” (which is the default) or “manual”. This can be set at the slice instance level.

- If RT is set to “auto” the slice service package will automatically assign the RTs to the endpoints from a pre-created pool that the operator configures as a pre-requisite. The operator can pick the RT range for the pool.
- If “manual”, then the user must assign the RTs.
    - Only one RT is required for any-to-any topologies.
    - Two RTs are required for hub/spoke (one for the hub and one for the spoke). The import/export policy will be such that all hubs have any-to-any connectivity between each other and can communicate to the spokes. Spokes can only communicate to the hubs, not to other spokes.
- Additionally, on a per endpoint basis, the user may specify to import/export other “additional” RTs. This allows for the operator to create custom connectivity configurations to endpoints external to the slice, though these endpoints are not considered part of the slice service and not visualized.
- If using hub-spoke connectivity, you cannot have both a hub and a spoke endpoint on the same PE, as the endpoint roles need to be the same. This is a limitation in the L2/L3 YANG Network Model and an area for future enhancements.

The slice service package will build the any-to-any or hub/spoke topologies for L3 service types (point-to-point is not an option for L3 service type) or the any-to-any, hub/spoke or point-to-point topologies for L2 service types.

The above will be done when the slice instance’s isolation parameter is set to dedicated, for isolation set for shared only the any-to-any connectivity is allowed. This means the shared slice endpoints will always be in an any-to-any topology with other shared slice endpoints. The dedicated slices attaching to the shared slice can be any-to-any or hub-spoke connectivity-type. Layer-2 point-to-point dedicated slices are not permitted to join a shared slice. More on dedicated and shared slices below. This auto-connectivity function is used for both L2 and L3 slice instance service types.

3.4 Dedicated and Shared Slice Isolation Types

Cisco has innovated beyond the IETF standard, given our visibility into upcoming customer needs and use-cases, to define two isolation types for slices. ***Dedicated*** slices and ***Shared*** slices**,** where shared slices can be accessed from one or many dedicated slice endpoints (i.e., a typical extranet connectivity model). In this case, the dedicated slices still cannot communicate with each other, but it does allow them to share common resource endpoints in the shared slice. Our thinking here was simplification and to focus on defining two well-known connectivity models based on intent, and thus the slice service model can significantly abstract the connectivity settings. Below are the definitions for **Dedicated slices** and for **Shared slices**.

3.4.1 Dedicated Slice Isolation Rules

- Both L3 and L2 service types supported.
- Dedicated transport slice instances will have a unique slice id/name to identify the transport slice instance.
- All endpoints in the dedicated slice will use the same transport behaviors specified in the referenced catalog entry. (forwarding, qos, Service assurance, etc.). We may allow some level of override at the SDP level in the future.

- Desired endpoints (Service Demarcation Points (SDPs)), which are based on PE and interface level identifiers, are assigned to the dedicated slices.
- Dedicated slice endpoint connectivity follows the connectivity type setting (any-any, hub/spoke, P2P). Note: P2P not available for L3 Slice types as this can be accomplished with A2A with only two endpoints.
- Endpoints in different dedicated slices can NOT communicate with each other (unless additional manually RTs are set on the SDP, but this is not the default behavior).
- A dedicated slice endpoint (SDP) can belong to only one slice service at a time.
- Dedicated slices can be updated dynamically to change their transport "behavior" (just point to a different catalog/NSST entry). This is an important value-proposition utilizing Cisco's declarative based NSO system.
- Dedicated slices (other than L2 point-to-point) can optionally connect to Shared slices.

3.4.2 Shared Isolation Slices

Shared slices support many of the same principles as dedicated slices, but the assumption is they are less fluid, as they are setup to support shared infrastructure endpoints and are not changed that often. The main reason for defining a shared slice is to again simplify the service model abstraction for common extranet use-cases.

- Both L3 and non-P2P L2 service types are supported for shared slices.
- When attaching a dedicated slice to a shared slice, they must be the same service type (L2 or L3)
- All endpoints of the dedicated slice (regardless of connectivity type except for L2 P2P which is not supported) will have direct connectivity to the shared slice endpoints.
- Shared slices are typically static and usually built once as "infrastructure". The sites are shared, in that they can belong to multiple dedicated slices simultaneously (extranet model). While typically static, they can still be updated as needed, including changing the catalog entry.
    - 5G Example: A 5G RAN gNodeB backhaul endpoint
    - Non-5G Example: A shared AAA application server at a central site
- Shared Slice instances will have a unique slice id/name to identify the transport slice (just like dedicated transport slices).
- All sites in a shared slice can communicate with each other (any-to-any connectivity only)
- Shared slices can support connectivity to multiple dedicated slices simultaneously (using extranet VPN machinery).
- Different dedicated slices connecting to the same shared slice do NOT have connectivity to each other. The shared slice resources are available to each dedicated slice, but do not provide transit between different dedicated slices.
- Only non-point-2-point dedicated slices can communicate to shared slices.
- A single dedicated slice can connect to multiple shared slices.
- When a shared slice is connected to the dedicated slice, all dedicated slice endpoints can communicate directly to the shared slice, this includes spokes sites in a hub/spoke connectivity topology.
- For L3 service types, IP addressing for dedicated slices accessing shared slice endpoints cannot overlap with the shared slice subnets or other dedicated slice subnets that are also accessing the same shared slice services.
- Shared slices also reference their own catalog/NSST value, and this behavior is used between sites in the shared slice. If this value is different than the dedicated slice that attach to it, then "asymmetric" forwarding behavior for bi-directional traffic will occur (see single-sided control point feature below).
- 3GPP provides for a slice instance (NSSI) sharing model: "A NSSI may be shared by two or more NSSI(s), this is also called a shared constituent of NSSI."

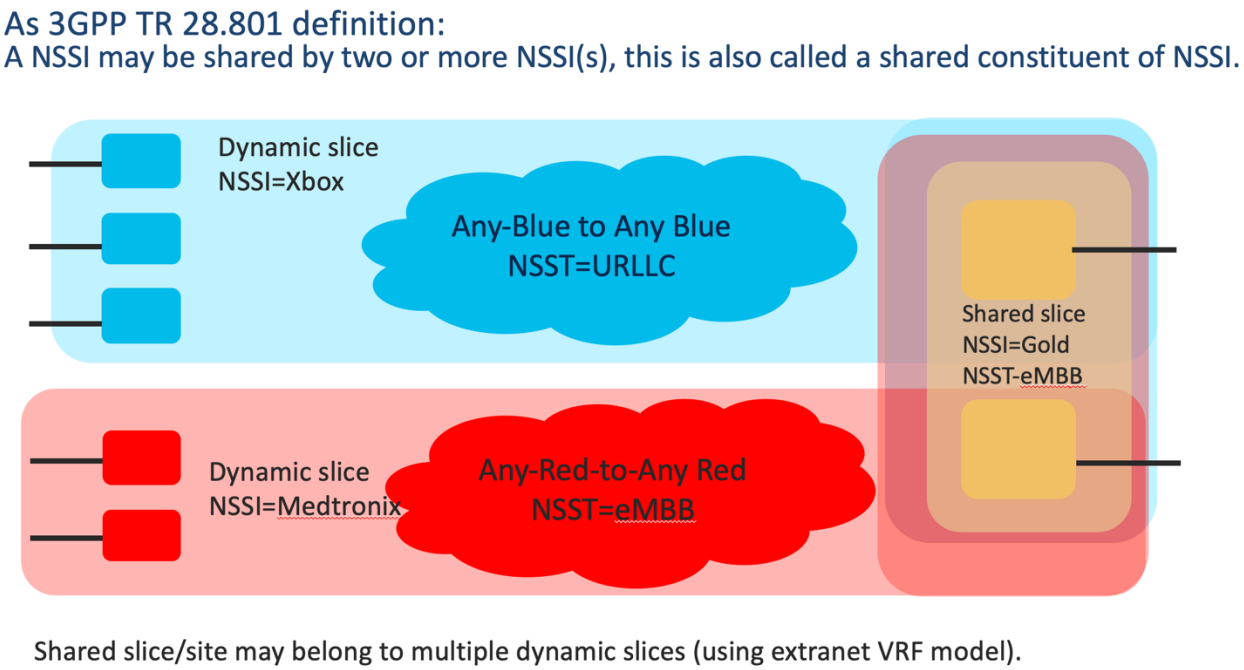

Figure 14: Shared Slice Example

3.4.3 Why do we need shared slices?

In the case of 5G RAN, the existing industry technology thinking is to map transport "behavior" with a dedicated VLAN interface from the gNodeB device (mobility domain). Assuming that the gNodeB vendors will classify which slice to use, and then map the corresponding GTP traffic onto the proper VLAN for transmission across the transport network to the UPF (in the case of N3 backhaul). This basically means separate VLANs per slice configuration at a common resource such as the gNodeB. While simple, this approach will eventually suffer from scalability challenges, and it also makes assumptions on gNodeB vendor capabilities. In contrast, when using a shared slice interface, this is a single connection to the gNodeB into a shared transport slice instance. This shared slice can be connected to multiple dedicated slices for unique transport behaviors per UPF. This use-case is commonly called an extranet or shared services VPN and it is implemented by careful settings of the import/export Route Target (RT) machinery in the BGP control plane. It also allows for BGP prefix "coloring" to signal different forwarding behaviors to different destinations (i.e., UPFs). It provides superior scale and simple automation flexibility and it is a Cisco differentiator.

3.4.4 The need for Single-Sided Slice Control Point

In the case where multiple dedicated slices (each with a unique catalog/NSST intent) need to connect to a shared slice endpoint (gNodeB RAN device for example), which may also have a different NSST, then the question is what forwarding behavior do we use for the underlay data processing? With no changes to the above logic, we will get asymmetric path forwarding behavior. The slice endpoint will ***receive*** traffic matching it's specified NSST path forwarding behavior (this is because the NSST defines the BGP route coloring for advertised prefixes

(exported) for those endpoints). Cisco has resolved these uni-directional limitations when using BGP coloring and ODN/AS by allowing for recoloring on VRF import that will allow the right classification of traffic bi-directionally. This feature will be used to enhance forwarding path behavior in shared slice scenarios and can be configured from just one end of the connection. Thus, the dedicated side of the slice can be configured to impose the dedicated slice forwarding path selection in each direction when connecting to a shared endpoint. This behavior can be enabled in the dedicated slice service by setting the "single-sided-control" flag. It will be enabled by default.

This feature is delivered as a control plane feature so it is supported across all XR device types.

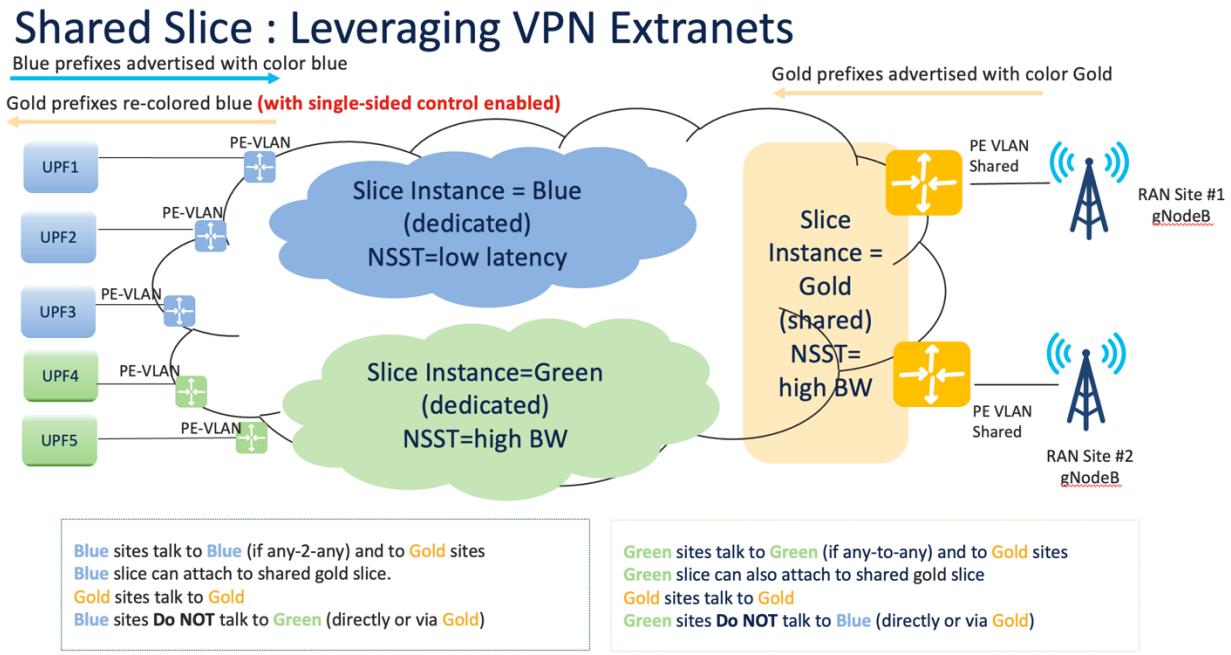

Figure 15: Shared Slices implemented with Single Sided Control Path Control

4 IETF Yang model

The initial slicing CNC 6.0 release will be developed from the 7/12/22 posted IETF Slice YANG model (plus Cisco augmentations). This is defined with the YANG model from this TEAS working group document: https://datatracker.ietf.org/doc/html/draft-ietf-teas-ietf-network-slice-nbi-yang-02

It is expected that this model will have continued modifications as IETF debate occurs in the TEAS Working Group and Cisco will continue to try and keep in good alignment until it is finally standardized. For interfacing with the CNC 6.0 Slicing solution, the Cisco Slicing YANG model and APIs are available here: https://developer.cisco.com/docs/cw-sandbox/#!ietf-network-slice-api

The latest IETF slice draft (as of 11/01/23- IETF 118) is still being modified but close to going WGLC and being finalized into an RFC. https://datatracker.ietf.org/doc/html/draft-ietf-teas-ietf-network-slice-nbi-yang-08

The IETF Slice YANG model is based on the following slice framework draft which is also still being reviewed: https://datatracker.ietf.org/doc/draft-ietf-teas-ietf-network-slices/25/

The latest version IETF slice Yang tree and Yang model are located here:
https://github.com/lana-wu/ietf-ns-nbi/blob/main/ietf-network-slice-service.tree
https://github.com/lana-wu/ietf-ns-nbi/blob/main/ietf-network-slice-service.yang

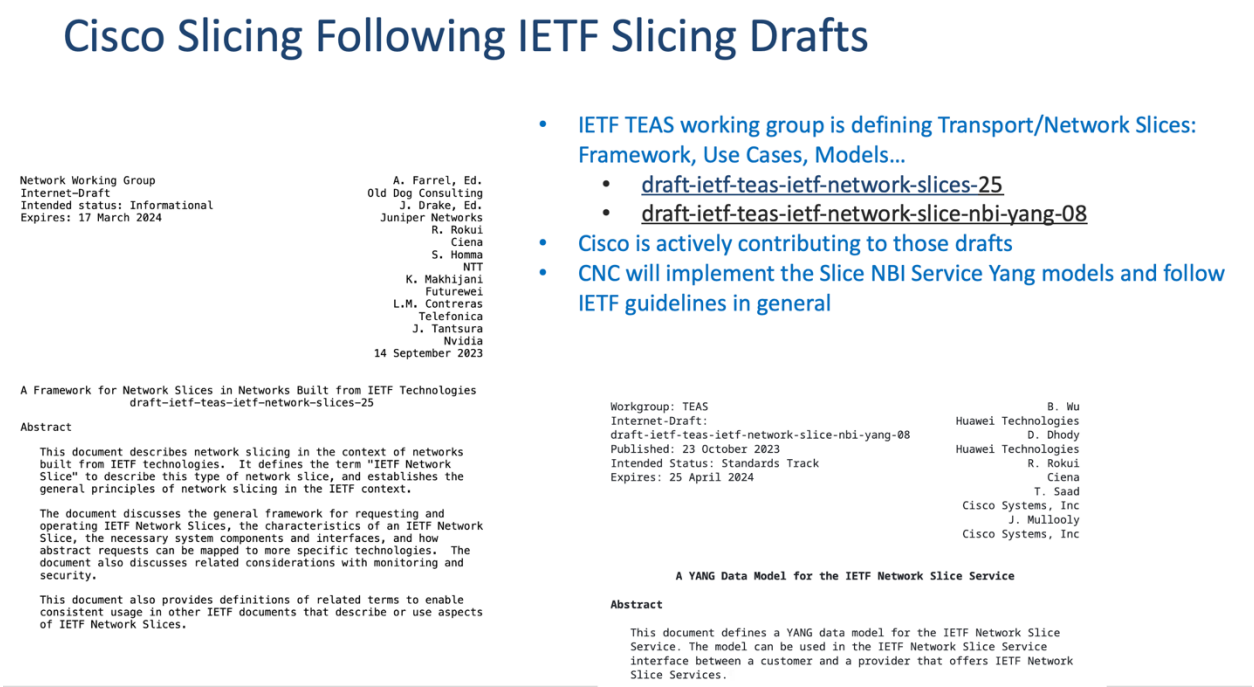

Figure 16: IETF Slice Drafts

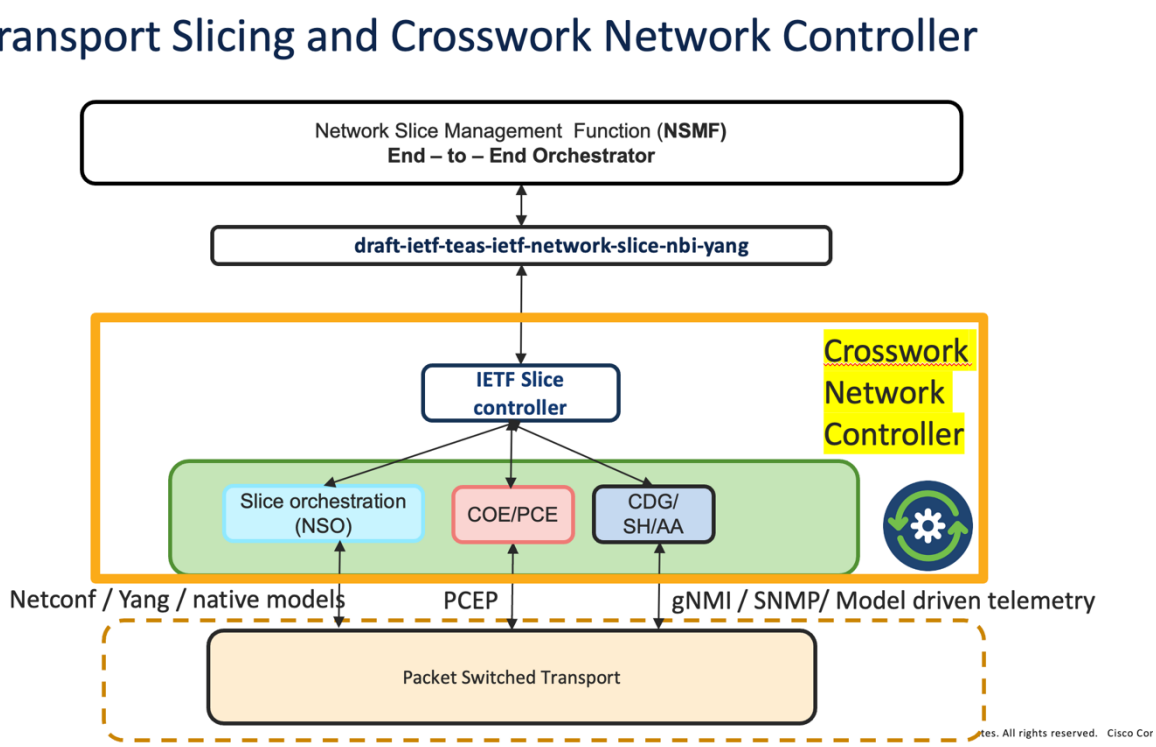

Figure 17: Integration with E2E Slice Controller

5 Transport Slice Deployment Pre-Requisites

As shown in the below workflow figure, the first step in enabling the network for Transport Slicing Services will be for the Slice Designer (or network administrator) to configure (once) certain pre-requisite configurations to allow the underlying machinery discussed above to be used by the slicing service.

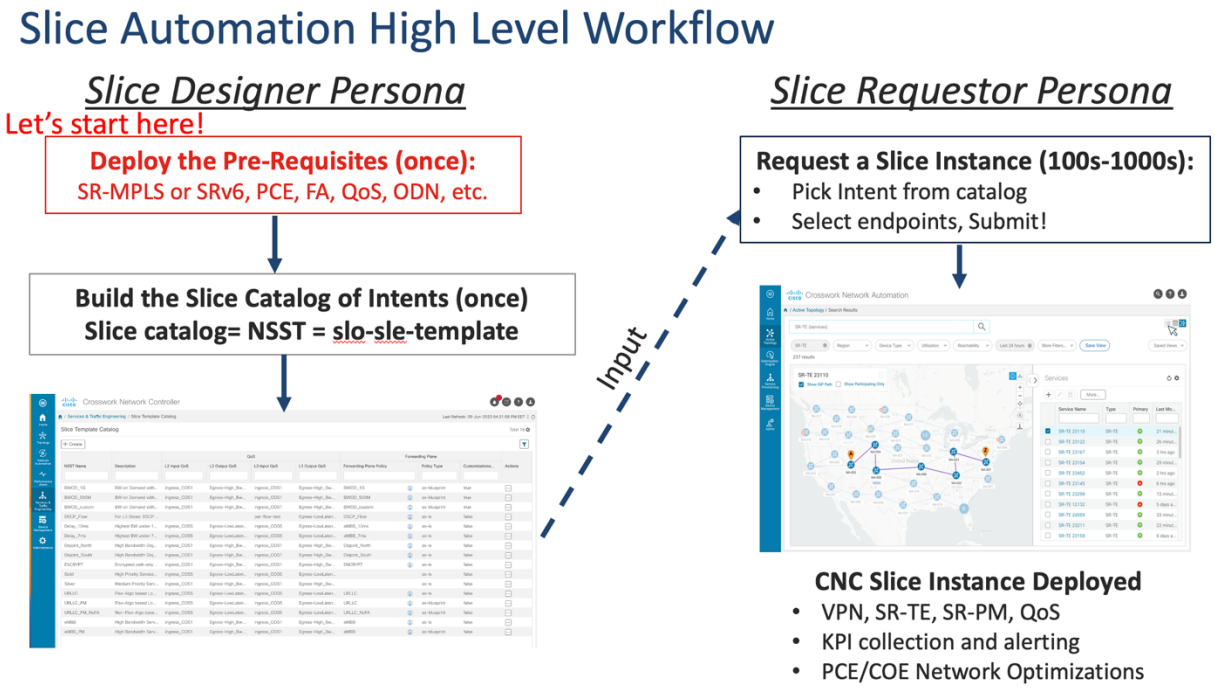

Figure 18: Step 1- Deploy the Pre-Requisites

5.1 A "Modern" IP/MPLS/SR Network

It is beyond the scope of this document to review all the pieces required for a modern packet network to be in place, but some of the important capabilities leveraged by the Cisco Transport Slicing Solution include:

- Support for L2 and/or L3 BGP based VPNs.
- If path forwarding is desired (a key capability of slicing) then the network should be enabled for SR-MPLS or SRv6 (with SR-TE policy support)
- Advanced IGP features such as:
  - SR Flex-Algorithm
  - Link affinity maps
  - Link Delay Performance Measurements propagated in the IGP (i.e., ISIS).
- Y1731 and SR-PM functionality and device support for granular Service Assurance
- Feature rich device level QoS capabilities for classification, scheduling, policing and shaping.
- Deployment of Cisco's Crosswork Network Controller (CNC) solution which includes:
  - Network Services Orchestrator (NSO) for declarative-based service provisioning
  - Path Computation Element (PCE)
  - Crosswork Optimization Engine (COE) for advanced path selection (BWoD, LCM).

5.2 QoS Pre-Requisites

As reviewed earlier, the Slice Designer should be familiar with the network's QoS capabilities and these settings have been pre-staged at the PE device with pre-built ingress and egress policy-

maps. The Transport Slicing Solution can automatically apply these policy maps to any identified slice endpoint interfaces. Additionally, core (P) link QoS is in place and assigned to the core interfaces as the Transport Slice Solution does not configure these links. When building the Slice template catalog, the Slice Designer can pick from these pre-staged policy-maps for use at both ingress and egress for both L2 and L3 slice services. These policy-maps would dictate the classification requirements (ingress) and thus the per-hop-behavior (PHB) scheduling treatment across the network.

In the Below figure we show an example of policy-map "ingress_TRUST" (again, built by the user given the capabilities of their PE device) which is used to match on incoming DSCP and set various device level qos settings (which in this case can be used to mark MPLS EXP bits).

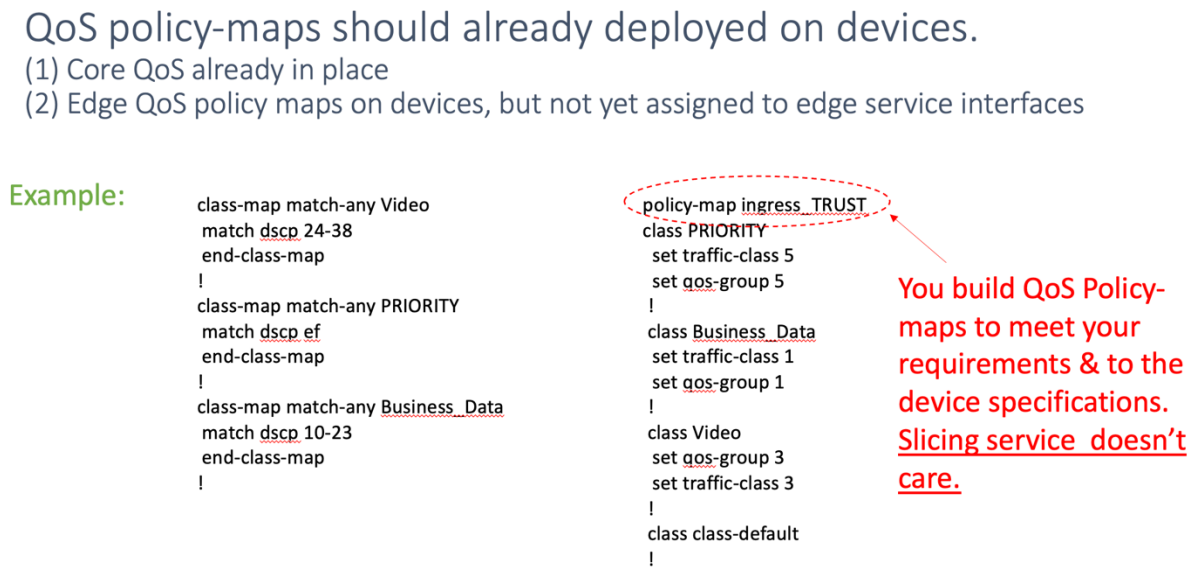

Figure 19: QoS Pre-Requisites

Note that this is not a new step or requirement for a modern IP/MPLS/SR network. If the network already has a well-defined QoS plan, then just use those pre-deployed maps. Once the policy-maps are pre-staged on the devices, the Slice Designer needs to identify to the Transport Slice Solution the names of the relevant policy-maps that are available to be used in the Slice template catalog. Further, the user needs to identify if these policy maps are an input or an output policy and if available for either L2 or L3 service types. If the same policy-map can be used for both L2 and L3, then it can be entered multiple times. This step allows the Slice Designer to provide a curated list of available policy maps for input, output, L2 and L3 that are available to be used in the Slice Template catalog. It prevents the accidental (or malicious) use of device level policy-maps that are not allowed for Slice services. Since this mapping step is done by the Slice Designer as a pre-requisite, the settings are done either via the NSO CLI or CNC/NSO API.

Add which QoS policies will be available for the Slicing Catalog

Simple NSO table of allowable QoS policies-maps for slicing services.

Names should match what is already available in the devices

Notice slice QoS policies can be further qualified as type L2 or L3

```
network-slice-services slo-sle-templates qos-catalog L2 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_TRUST_L2
 description "Trust COS markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_UNTRUST
 description "Do Not Trust COS  markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-High_Bw_Apps
 description "High BW egress"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_TRUST
 description "Trust DSCP markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_UNTRUST
 description "Do Not Trust DSCP markings from CE"
```

Figure 20: Identifying allowed QoS policies

5.3 ODN Templates for SR-TE policies Pre-Requisites

A key component of the Cisco Transport Slicing Solution is using SR-TE for surgical slice path forwarding behaviors. The Slicing service uses the SR-TE ODN template feature with Automated Steering (AS) to automatically classify traffic into the right SR-TE tunnel by building SR policies for a given “color”. The destination prefixes announce their intent by advertising via BGP an extended color communities which is then matched on ingress for received traffic from the PE interface. The Slice Designer needs to be familiar with these mechanisms and pre-build the available SR-TE ODN templates in CNC/NSO such that they are available for the Slicing Template catalog. The Slice service will do the actual deployment of all ODN templates to the head-end routers and coloring of all destination prefixes during Slice Instance provisioning. The ODN templates will define the slice forwarding intents, including reference to any flex-algos that have been pre-defined. The slice service leverages these pre-created ODN templates, and thus any supported ODN SR-TE CFP policy features are automatically supported by the Slice service.

These ODN templates are called “headless” because a head-end device is not required to be entered. The slicing package will automatically add the head-ends as slice instance SDP endpoints are selected. If the user does add the head-ends, these ODN policies will be pre-positioned on the devices and the Slicing service will just use them automatically. These pre-built ODN templates can also be used as the “blueprint” reference if “as-blueprint” is selected in the Slice Template catalog when adding a new Slice catalog entry.

Some example intents are shown below. They include metric options of lowest latency, high BW (igp) or preferring a set of specific links that have affinity markings (in the below cases tied to a specific flex-algo). In the affinity link scenario, the network administrator can have specific links

that meet a specific criterion included in the flex-algo, such as dis-jointness, encryption or using only high speed (100Gbps) interfaces.

Example 1: Lowest Latency Intent- Using PCE and link level delay metrics, pick the lowest latency path

```
cisco-sr-te-cfp:sr-te odn odn-template URLLC_NoFA
 color 111
 dynamic pce
 dynamic metric-type latency
!
```

Example 2: Highest BW path Intent- Using PCE and link igp cost metrics, pick the highest BW path

```
cisco-sr-te-cfp:sr-te odn odn-template eMBB
 color 100
 dynamic pce
 dynamic metric-type igp
!
```

Example 3: Reserve 1Gbps BW Intent- Using PCE & COE provide BW on Demand reservation of 1Gbps over high BW path (also, BWoD setting is enabled on COE)

```
cisco-sr-te-cfp:sr-te odn odn-template BWOD_1G
 color     121
 bandwidth 1000000
 dynamic pce
 dynamic metric-type igp
!
```

Example 4: Only transit Encrypted Links Intent- Using PCE and flex-algo 129 to pick path (flex-algo 129 is configured for link affinities of only Encrypted links)

```
cisco-sr-te-cfp:sr-te odn odn-template ENCRYPT
 color 115
 dynamic pce
 dynamic flex-alg 129
 dynamic metric-type igp
!
```

Example 5: Disjoint Paths Intent- Using PCE and flex-algo 131 and 132 to pick path (flex-algo 131 and 132 are configured for link affinities for disjoint paths)

```
cisco-sr-te-cfp:sr-te odn odn-template Disjoint_North
 color 116
 dynamic pce
 dynamic flex-alg 131
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template Disjoint_South
 color 117
 dynamic pce
 dynamic flex-alg 132
 dynamic metric-type igp
```

Sample Affinity based Flex-Algo device configs for the above examples:

```
router isis SPNAC
 is-type level-2-only
 net 47.0000.0000.0020.00
 distribute link-state
```

```
 nsf ietf
 affinity-map North bit-position 1
 affinity-map South bit-position 2
 affinity-map ENCRYPT bit-position 0
 flex-algo 128
  metric-type delay
  advertise-definition
 !
 flex-algo 129
  advertise-definition
  affinity include-all ENCRYPT
 !
 flex-algo 131
  advertise-definition
  affinity include-all North
 !
 flex-algo 132
  advertise-definition
  affinity include-all South
 !

interface Loopback0
  …
  address-family ipv4 unicast
   prefix-sid index 104
   prefix-sid algorithm 128 absolute 16804
   prefix-sid algorithm 129 absolute 16904
   prefix-sid algorithm 131 absolute 17104
   prefix-sid algorithm 132 absolute 17204
  !
 !
!
 interface TenGigE0/0/0/1
  …
  affinity flex-algo South ENCRYPT
  address-family ipv4 unicast
   metric 10000
  !
 !
```

5.4 Service Assurance Pre-Requisites

5.4.1 Y.1731 support for L2 Point-to-Point Slices

For point-to-point L2 services, we may need to monitor latency, delay, loss bandwidth with Y.1731. The slice service package will enable Ethernet Y1731 based probes for delay, loss and synthetic loss if configured in the catalog and the service is a L2P2P. Additionally, the catalog heuristic packages will allow monitoring to be enabled from CNC Service Health application.

But first the Y.1731 profiles that will be referenced in the Slice Template catalog need to be pre-built with the NSO l2vpn-ntw package. Here is an example:

```
l2vpn-ntw y-1731-profile Profile-Delay-1
 schedule interval 5
 schedule duration 5
 type delay
 probe measurement-interval 60
 delay-params statistic delay-two-way
 !
 delay-params statistic jitter-two-way
 !
!
l2vpn-ntw y-1731-profile Profile-Loss-1
 type loss
 probe measurement-interval 60
 probe priority       2
 loss-params statistic loss-sd
 !
!
```

5.4.2 Performance Measurement (SR-TE) Pre-Requisites

The Slicing solution can automate the configurations to enable SR-TE based Performance Measurement (SR-PM) probing for liveness or delay measurements as part of the slice instance deployment. Note, this is not the same Performance Measurement probing used for link level monitoring and integrated with ISIS for delay-based metrics. Whether to use PM for SR-TE monitoring is decided in the Slice Template Catalog, but before that can be done, the required pm-profiles and headless svc-profiles need to be pre-created in NSO. Here is an example:

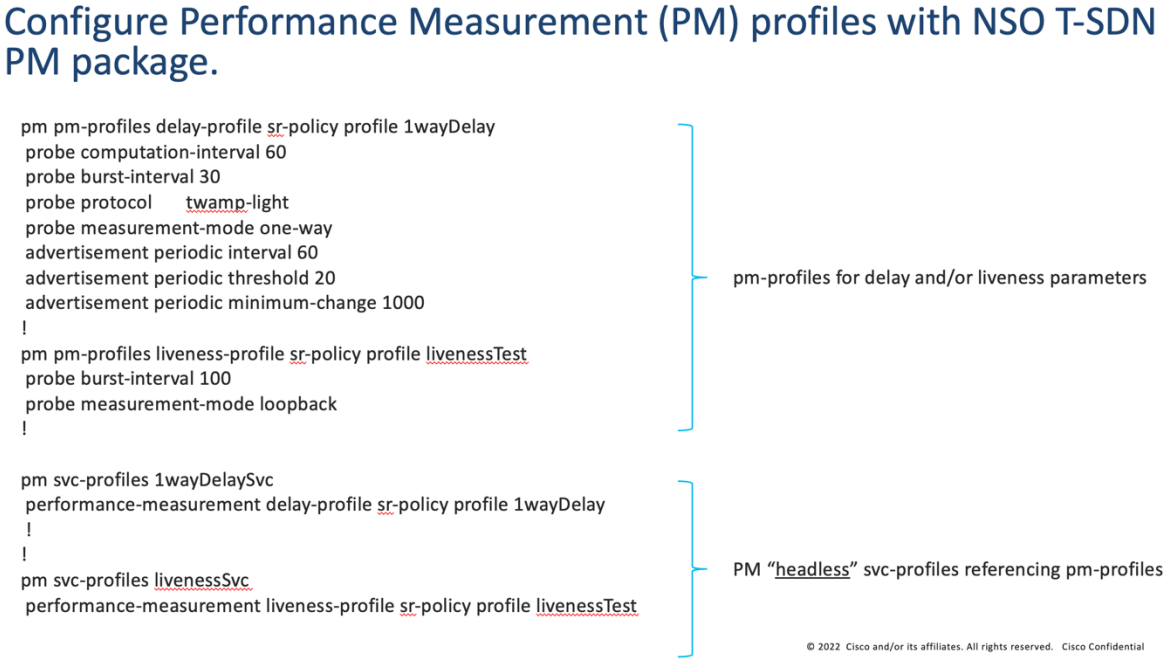

Figure 21: Building Performance Measurement Pre-Requisites

Also, enable SR-PM collection via the CNC UI under Administration→ Settings

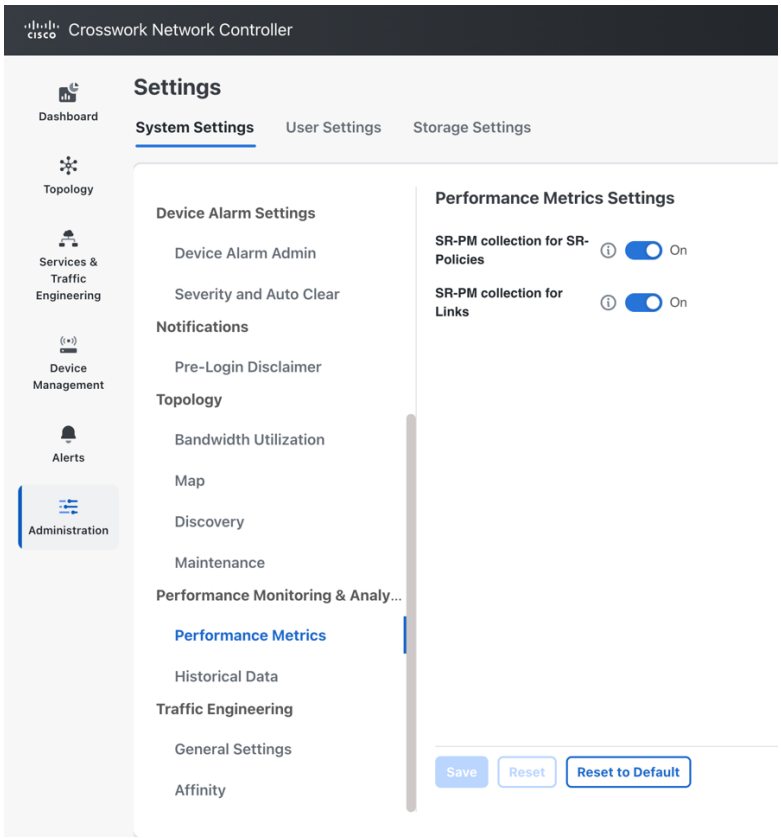

Figure 22 Enable CNC SR-PM Analysis

5.4.3 CNC Custom Heuristic Package Pre-Requisites

When CNC provisions a new slice instance, CNC Service Health can also be included. When using Service Health, a set of Heuristic Packages are used (either system provided or custom) to set up the device telemetry collection (via CDG) and trigger unhealthy thresholding. The user will be able to assign the desired heuristic packages in the Slice template catalog. If any custom thresholds or monitoring is required (say for SR-PM delay thresholds), the user should pre-create these custom Heuristic packages so they are made available to the slice catalog.

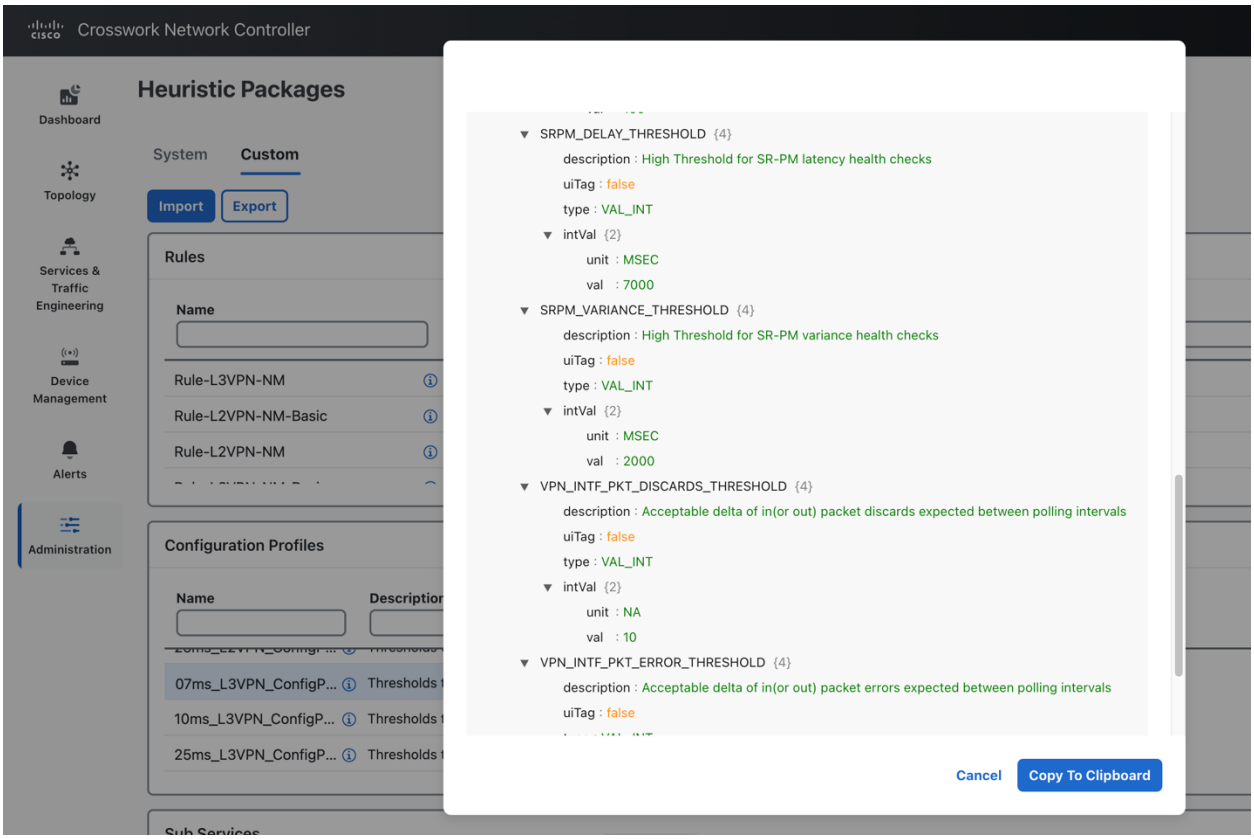

Figure 23: Custom Heuristic package with 7ms SRPM_Delay_Threshhold

CNC product documentation should be referenced on the procedure to create custom Service Health Heuristic packages.

5.5 NSO Resource Pool Pre-Requisites

There is only one resource pool used specifically for slicing and that is only if using “as-blueprint” based Slice templates where the system will create a new color for the cloned ODN template from the pool. The other resource pools referenced in the below example are required as part of the T-SDN, L2 and L3 VPN CFPs and also include Y1731 resources for L2 Service Assurance.

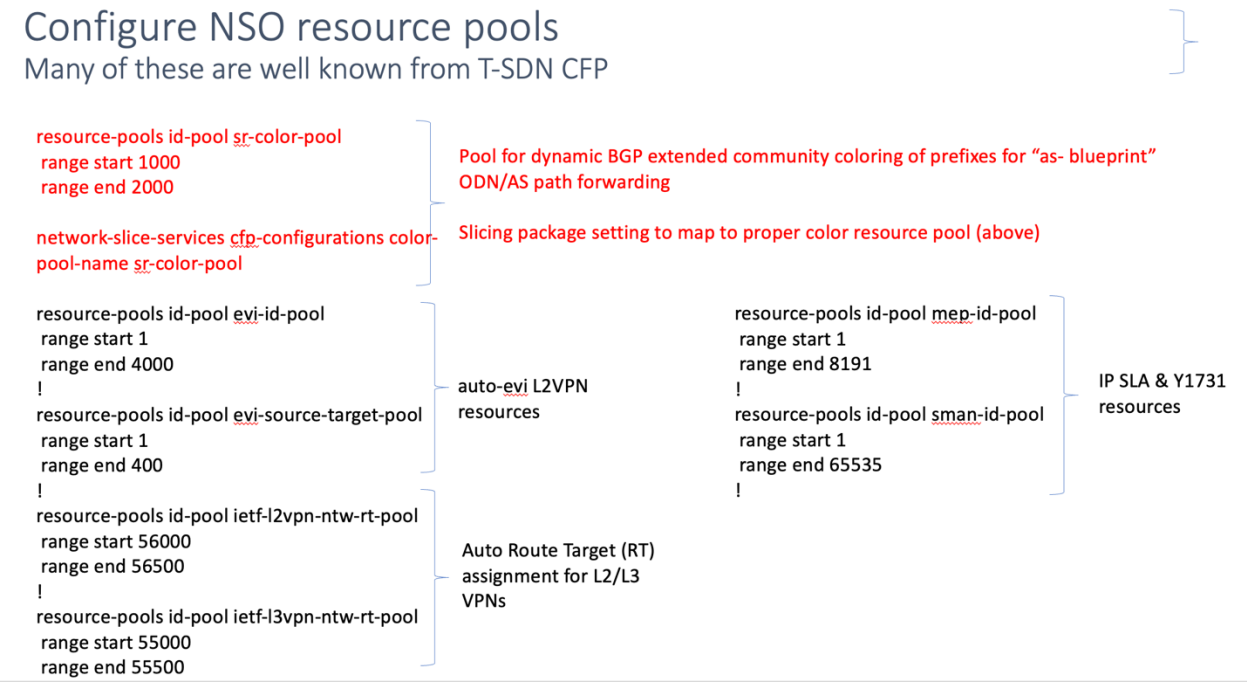

Figure 24: NSO Pre-Requisites- Resource Pools

5.6 Miscellaneous Pre-Requisites

There are several additional miscellaneous pre-requisites listed below in order for certain use cases to work properly. Again, these are not specific to slicing but are required as part of the underlying feature machinery. For example, for L2 point-to-point slicing, the VPWS machinery is leveraged, and in this case, a single route-policy map is used by the device to advertise BGP updates to the Route Reflector (RR). These updates will contain the color community for ODN/AS SR-TE selection. The pre-requisite here is that the policy-map name must be identified to the slicing package in the global slice settings configuration along with the policy-map and BGP configurations being pre-configured on the device. In the below example, the BGP RR is at 192.168.255.8.

```
route-policy SET_COLOR_EVPN_VPWS
  pass
end-policy
!
router bgp 64001
 bgp router-id 192.168.255.20
 address-family vpnv4 unicast
 !
 address-family l2vpn evpn
 !
 neighbor 192.168.255.8
  remote-as 64001
  update-source Loopback0
  address-family vpnv4 unicast
  !
  address-family l2vpn evpn
   route-policy SET_COLOR_EVPN_VPWS out
  !
 !
```

Figure 25 Example: Bootstrap RR route-policy map on PE device for VPWS slices

For L2 EVPN single-homed multi-point services (non-vpws), an ESI is required at the port level. Cisco's SR-TE autosteering (AS) capabilities do not support steering colored evpn type-2 routes (mac addresses). An ESI is required to be advertised which is a type-1 route. The assumption is that the ESI has been pre-configured on all physical ports that will support evpn L2 multi-point services. A single non-zero and unique value must be configured on the main port (not at the sub-interface level). Here is an example configuration for setting the ESI on an interface that is assumed to be in-place before any slice services are created:

```
evpn
 interface TenGigE0/0/0/2
  ethernet-segment
   identifier type 0 33.00.00.00.00.00.00.33.02
  !
!
```

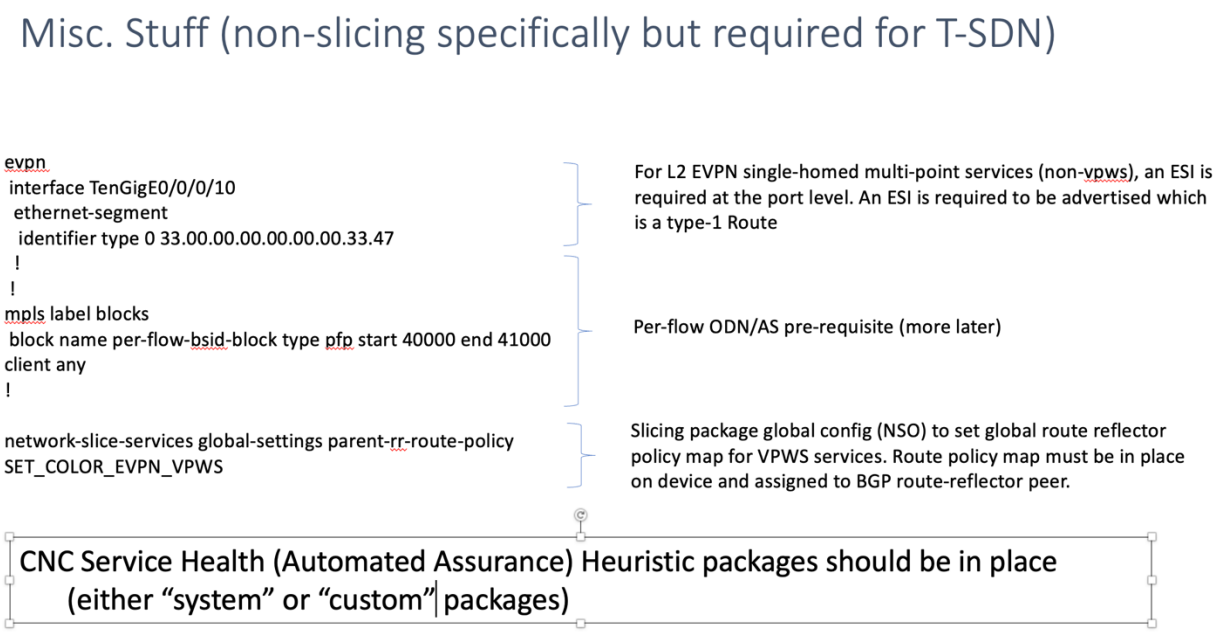

Figure 26: Miscellaneous Pre-Requisites

Enable Bandwidth on Demand (BWoD) processing via the CNC UI if this capability will be used.

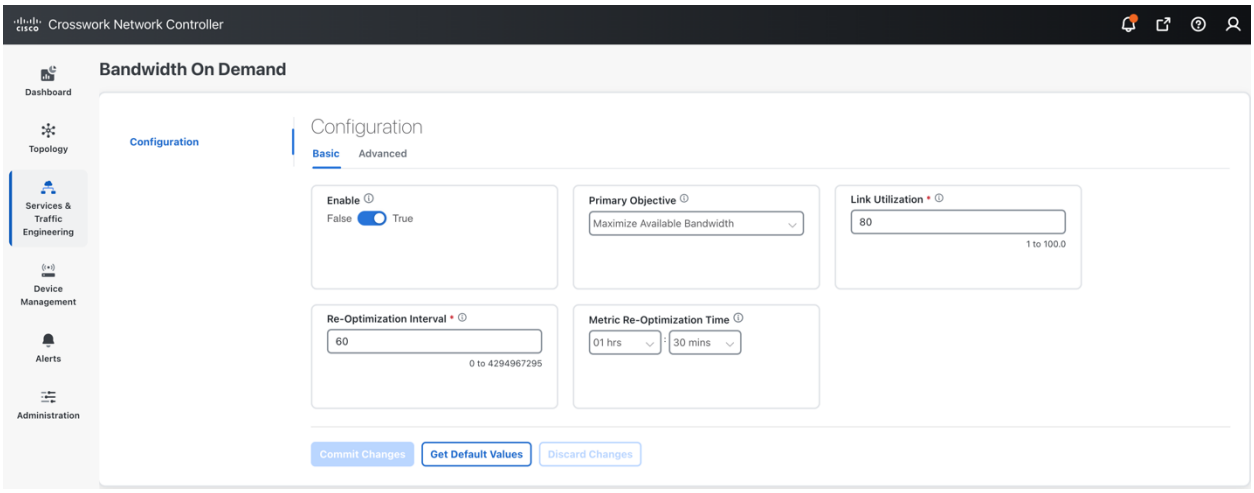

Figure 27 Enabling BWoD Functionality

6 Transport Slice Template Catalog Creation

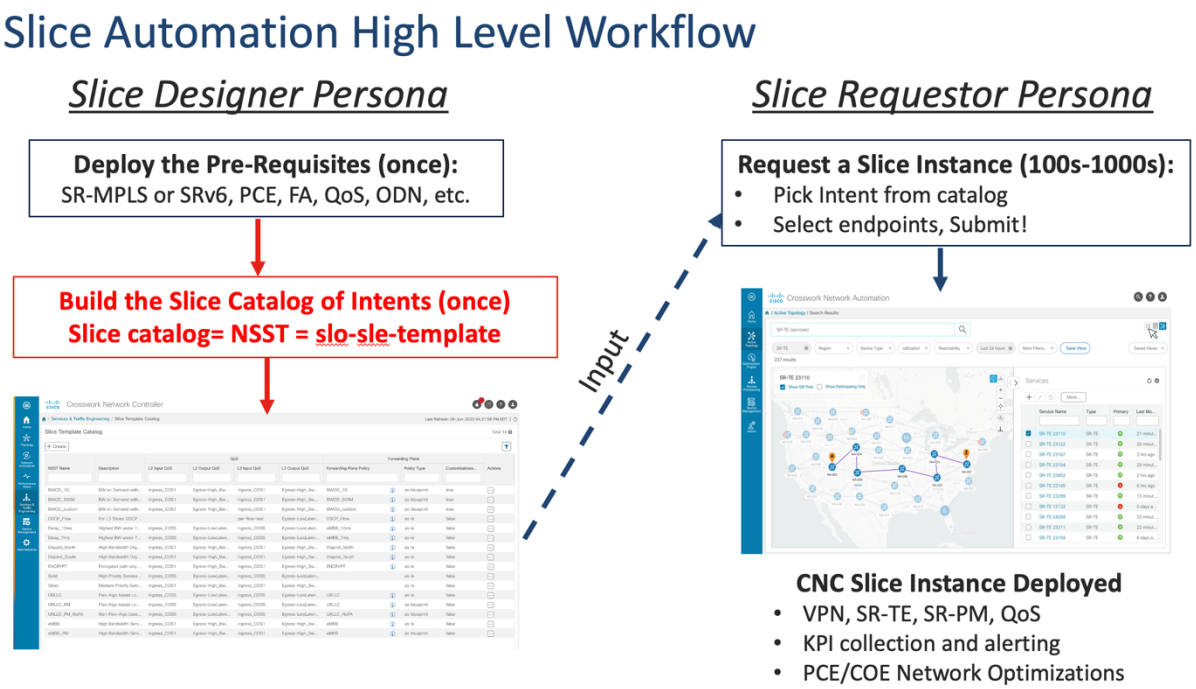

Figure 28: Step 2- Building the Slice Catalog of Intents

The Transport Slice Template Catalog provides a list of Slice Designer built “slice types” which can be referenced when creating the actual slice instance. This catalog (along with the slice instances themselves) can be built in multiple ways:

1. Via CNC UI, though the Service Assurance parameters are not yet supported in CNC 6.0 via the UI
2. Via CNC/NSO Slicing API
3. Via NSO CLI including “load merge” from a text file

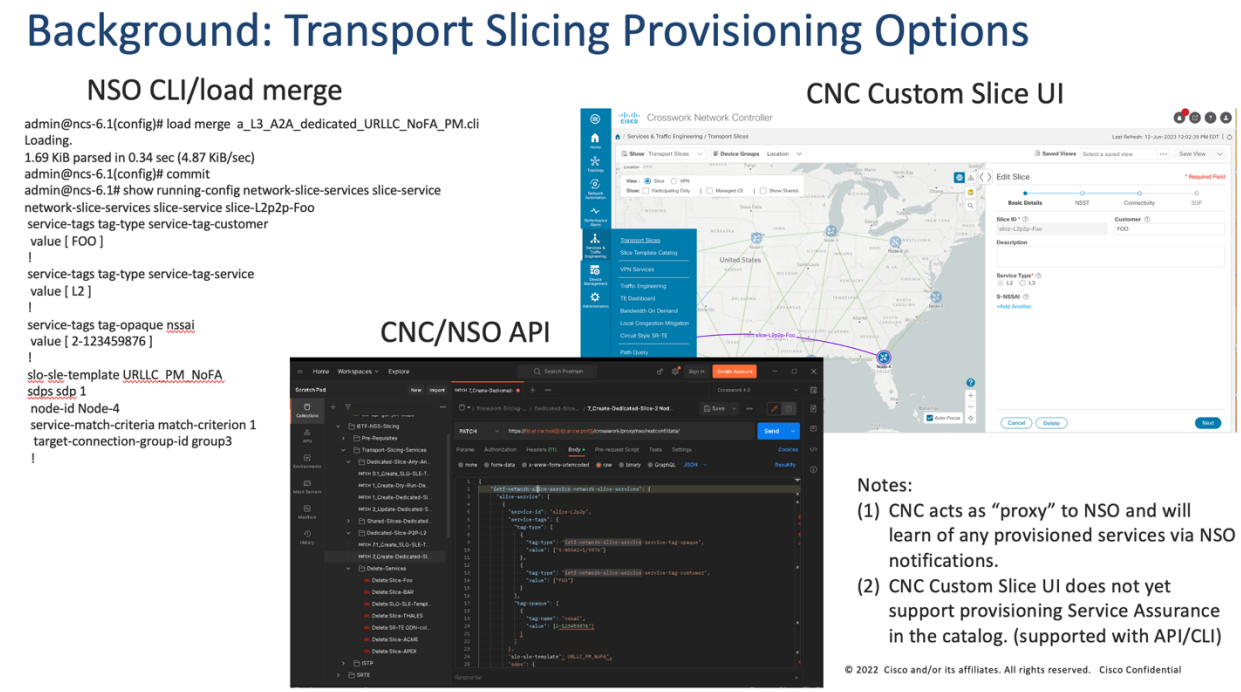

Figure 29: Different Provisioning Options

When using the Slice Catalog UI, the user can build the new Slice Templates and:

1) Configure a template name and a description.
2) Assign QOS ingress and egress policy maps (for both L2 and L3 service types)
    a. Referencing the pre-created table built under the QoS pre-requisite section.
3) Select a Forwarding Plane Policy
    a. Referencing the pre-created ODN templates with forwarding intent
    b. Determine if the template should be used "as-is" or "as-blueprint". As-is forwarding templates increase overall scalability as the SR-TE tunnels can be shared across multiple slice templates and instances but these ODN templates will not be dynamically modified by the slice package with additional functionality, including dynamic support for Performance Measurement or BWoD reservations. A template can be set "as-blueprint" in order to support dynamic PM, but with customization still disabled.
    c. If using "as-blueprint", determine if customization is need, which delivers dedicated per-slice instance SR-TE tunnels. The usual use-case here is to have dedicated BW reservations for a slice instance.

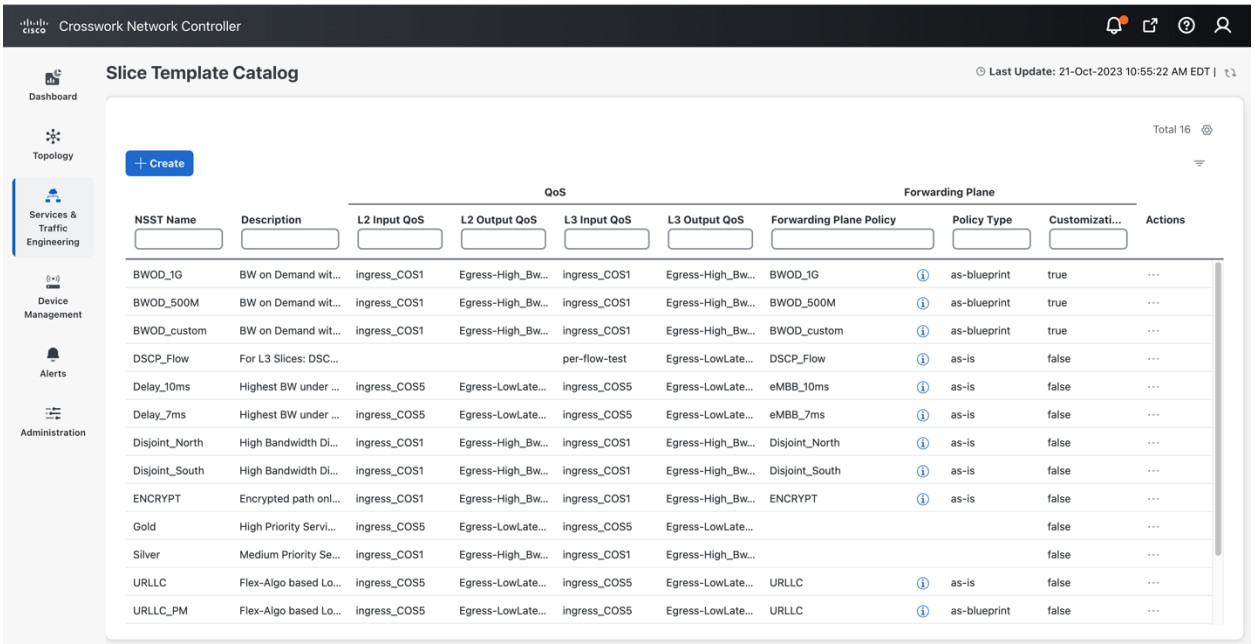

Figure 30: Sample Slice Template Catalog

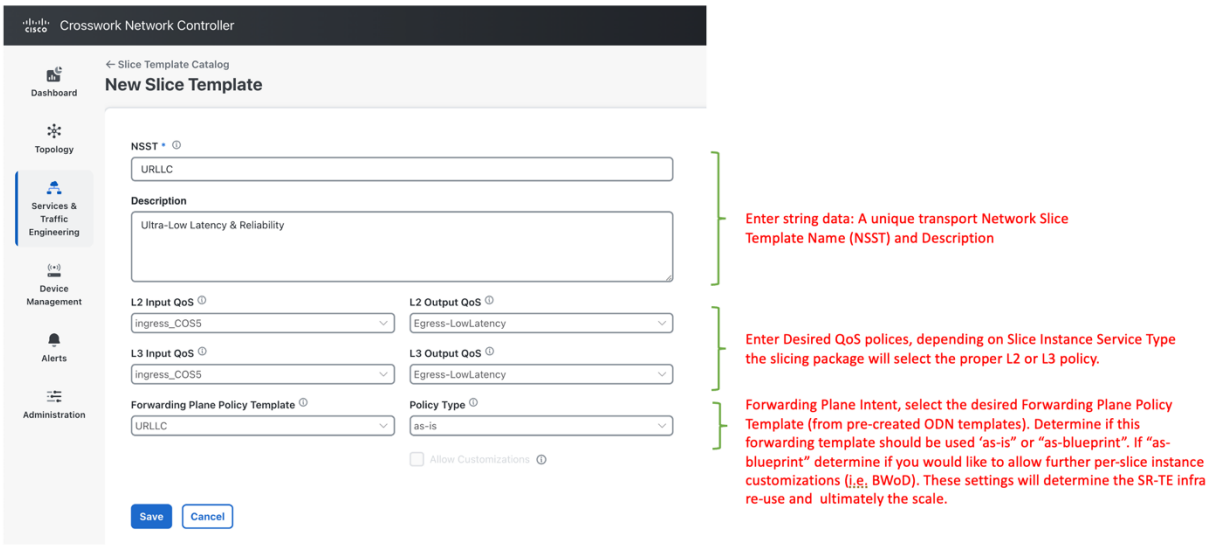

Figure 31: Creating Slice Template Catalog Entry via CNC GUI

If Service Assurance is required for the Slice instance, then the Slice Designer can create the new template with necessary SA functionality via NSO CLI or API. Below is the same template in NSO CLI with Service Assurance parameters.

There are three Service Assurance sub-sections to the template setting:

1) Reference pointers to CNC Service Health Heuristic packages to be used and monitoring state. Since different connectivity-types (pt-2-pt or multi-point) can be selected when provisioning a slice instance and different instance service types (L2 or L3), multiple heuristic package options are available and the system will select the proper package

depending on the slice instance requirements. Also, this monitoring state cannot be changed individually per slice instance or VPN level, it is set universally for all slice instances referencing this Slice template.

2) If the Slice Instance is a L2 service type with pt-2-pt connectivity, then Y1731 probe monitoring can be enabled. The settings required are shown in the below example. Slice SLA alarming and alerting can be configured if the proper settings are selected in the L2 Heuristic package for Service Health.
3) Lastly, for all slice instance types and connectivity models, SR-PM can be dynamically enabled on the SR-TE tunnel if the Slice Forwarding policy-type is set for "as-blueprint". Slice SLA alarming and alerting can be configured if the proper settings are selected in the L2 Heuristic package for Service Health.

```
admin@ncs# show running-config network-slice-services slo-sle-templates slo-sle-template URLLC_PM
network-slice-services slo-sle-templates slo-sle-template URLLC_PM
 template-description "Flex-Algo based Low latency Service with Performance Measurement Delay Probing"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  URLLC
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "25ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 2wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 2wayDelay
!
```

See previous explanations

CNC Heuristic packages to be used for Service health. Slicing package will pick proper package depending on Slice instance service type (L2 or L3) and L2 connectivity model (point-to-point or multi-point). User can also associate custom packages.

Y1731 probing specifications: For L2 point-to-point slice service types only. Ignored for other slice types

SR-TE based SR-PM probing. Only allowed if "as-blueprint" selected since these SR-PM configs will need to be merged into ODN template. Note: static SR-PM policies can be configured in "as-is" referenced ODN template if desired with pre-deployed headend PM profiles.

Figure 32: Creating Slice Catalog Template via NSO CLI with Service Assurance Intents

Once created Slice templates cannot be modified/edited. They can be deleted and recreated with the same name, but not changed. This is because changing existing template parameters would then require an update to any existing slice instances using those templates.

SA parameters can only be set vi the NSO CLI or CNC/NSO API in CNC 6.0, as there is no UI support for these parameters. What this also means is that these parameters will not be visible when viewing the slice template in the CNC UI. The other Slice template entries created via the API are still visible in the CNC UI minus the SA fields.

7 Transport Slice Instance Creation

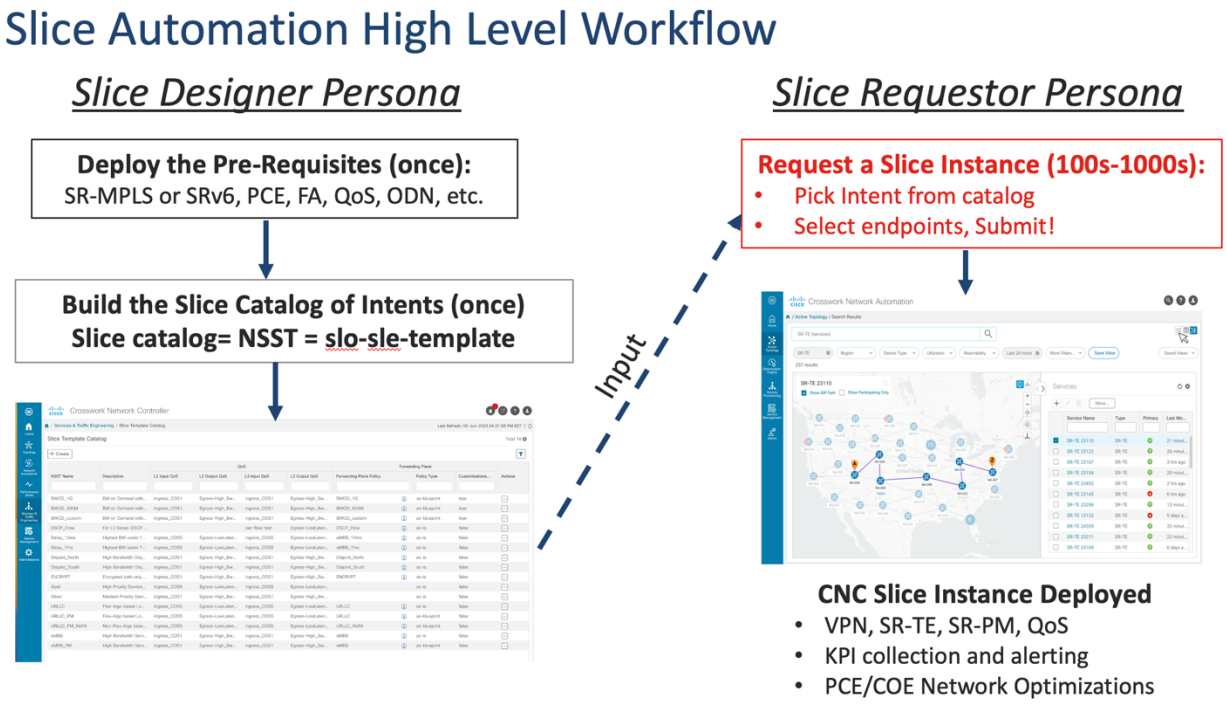

Figure 33: Step 3- Slice Instance Provisioning

The Slice Instance can be created via CNC UI, CNC/NSO API or NSO CLI. The below examples will show the UI steps and explain the various fields (both required and optional). The design goal was to make this slice instance creation as automated and as simple as possible, but given customer input, we had to add some optional capabilities in order to meet customer requirements. These capabilities add some complexities and such are maintained under the “advanced settings” tabs and are completely optional.

From the CNC UI, select “Transport Slices” under “Services & Traffic Engineering” on the left side tool bar. Existing slices are shown in the list on the right and new slice instances can be created by selecting “Create”.

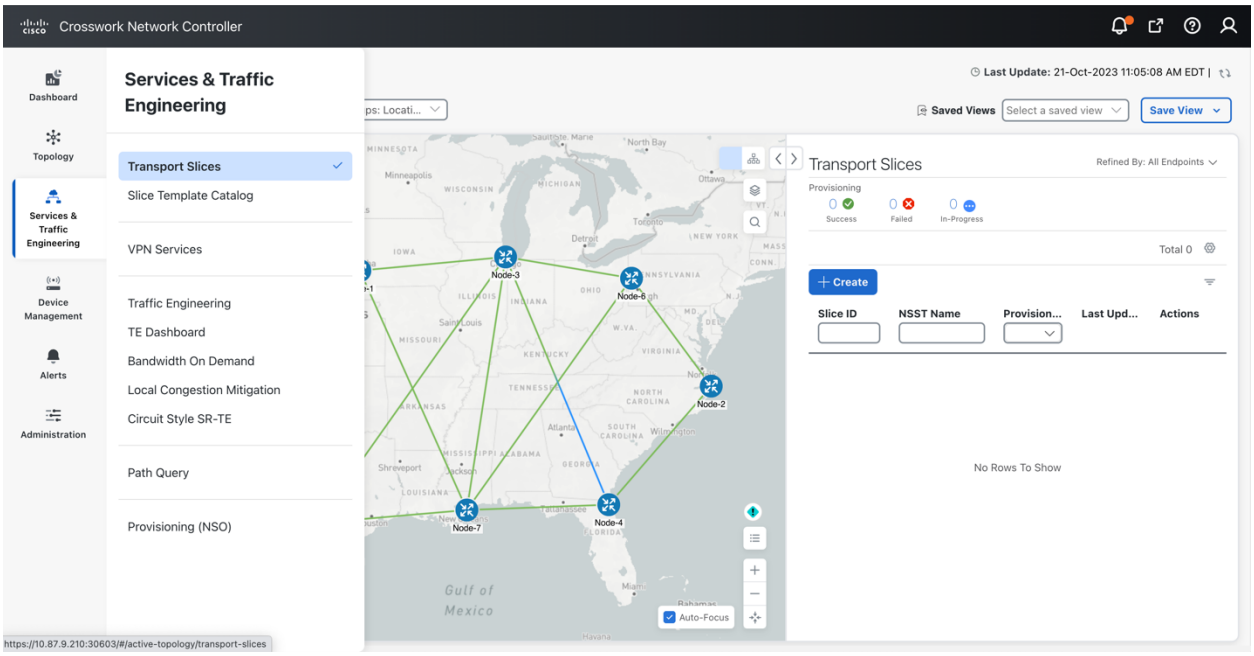

Figure 34: Slice Instance Creation via CNC GUI

7.1 Slice Instance Creation- Step 1 Basic Details

The first screen asks some basic questions such as Slice ID (i.e., the slice name), Description and Customer. All of these fields are strings that can take in any user input, with only the Slice ID being mandatory. The "S-NSSAI" (Single Network Slice Selection Information) field is also optional and it is string formatted. It allows for the Slice Requester to provide some meta-data tagging to associate this Transport Slice in the Transport Domain with a Mobility Slice built in the Mobility domains where S-NNSAI is the common identifier (5GCore, RAN, etc.). It is not used in any transport device level provisioning; it is simply a mechanism that can be used to help the administrator or top-level orchestrator make proper transport to mobility slice relationship associations. Multiple S- NSSAIs can be added to a single Transport slice providing an N:1 relationship. As an example of its usefulness, the top-level orchestrator can do an API request to retrieve all Transport Slice Instances that have a matching NSSAI value.

Lastly, the Slice "Service Type" is selected (either L2 or L3) based forwarding. Future Service Types may include "circuit-style" or Private Line Emulation (PLE).

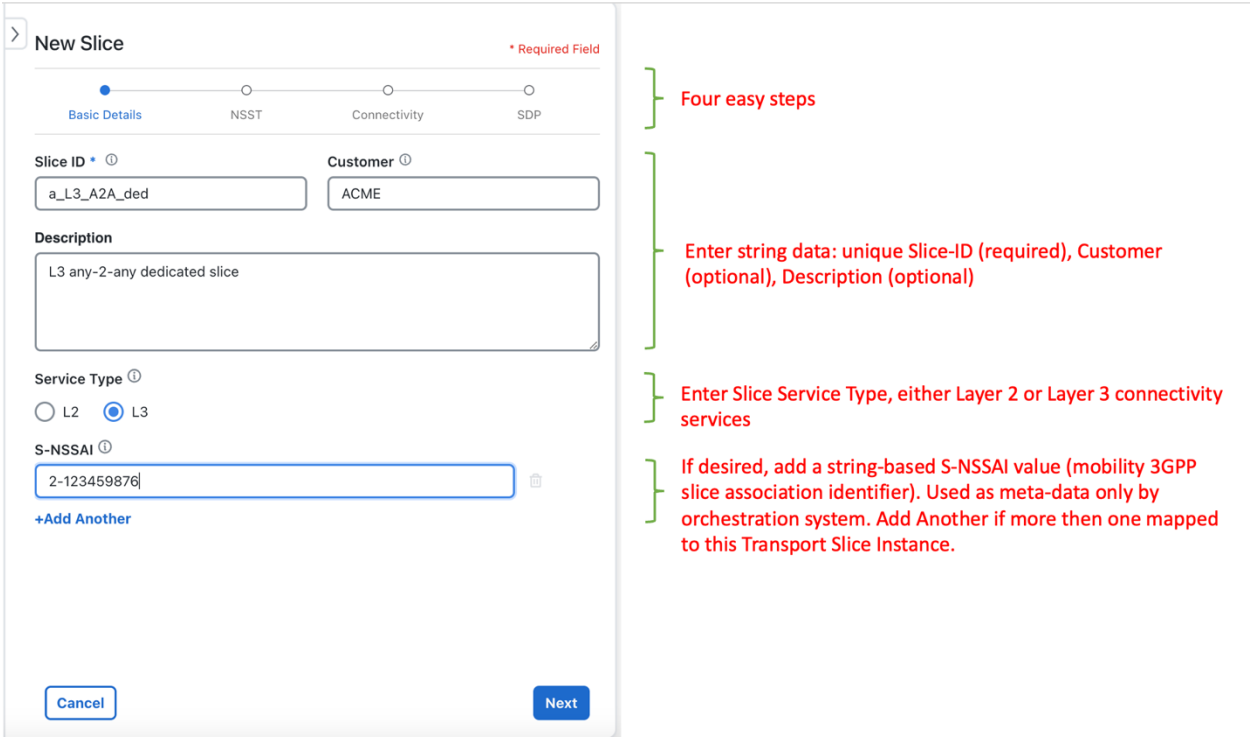

Figure 35: Slice Instance Provisioning- Basic Details

7.2 Slice Instance Creation- Step 2 Pick a Slice Template (NSST)

The next step is to pick a Slice Template to use from the pre-created Template catalog. To match 3GPP 5G naming convention this is called the Network Slice Subnet Template (NSST). The description on these templates will describe the intent specified by the Slice Designer. Depending on the Slice Service Type selected in the previous step, the system will pull the appropriate L2 or L3 based functionality referenced in the template (for example QoS settings).

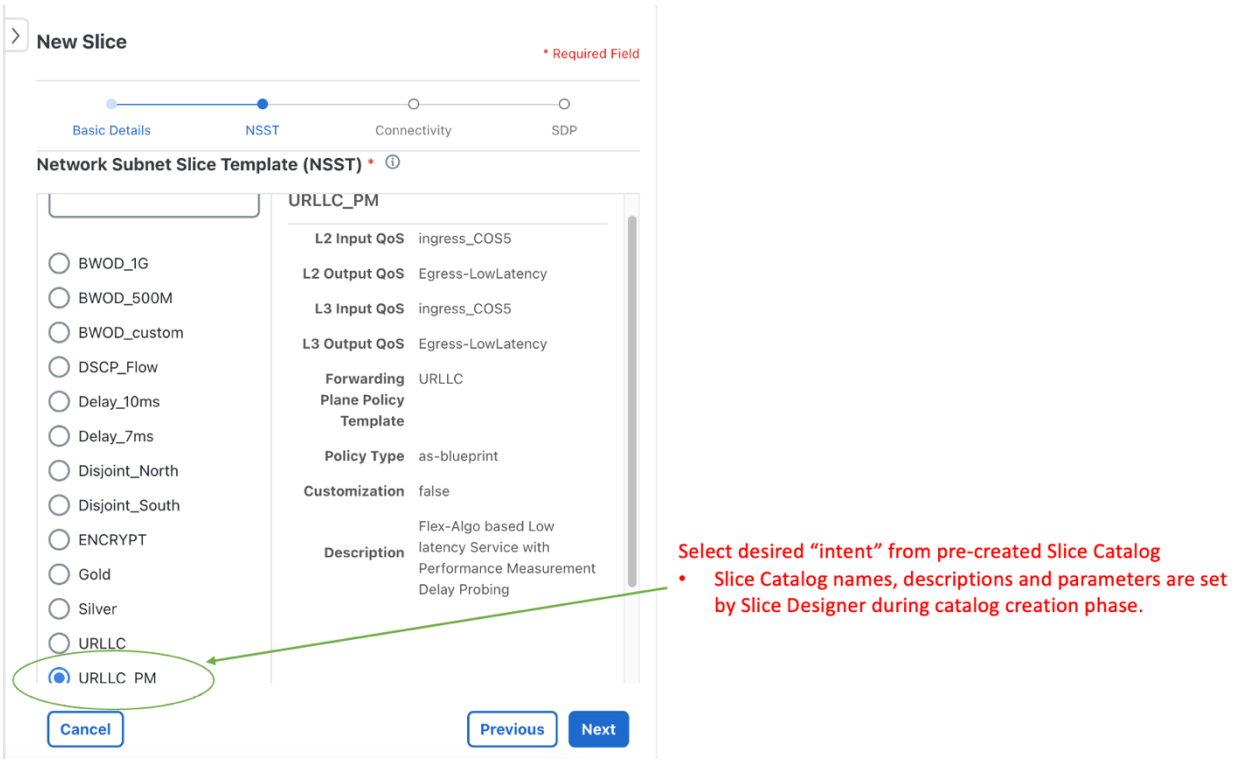

Figure 36: Slice Instance Provisioning- Catalog Template Selection

7.3 Slice Instance Creation- Step 3 Build the Connectivity Model

Looking at the example below, in this step we build the connectivity details for the slice. The IETF Slice YANG model has the concept of Connectivity Groups with the idea that multiple Connectivity Groups can be built under a single Slice ID. Cisco only supports one Connectivity Group in CNC6.0 thus this field is for future use. We may (for example) use Connectivity Group to allow for multiple VPNs in the future.

On this page is where we select if this slice is a dedicated or shared slice. If this is a dedicated slice, it can optionally connect to pre-created shared slices (if it is not a L2 P2P slice). Single Sided Control will allow for uniform bi-directional policies when connecting to the shared slice (i.e., the dedicated slice polices are used when connecting to shared slice endpoints).

Regarding the connectivity-type selection box, there are certain rules:

- If it's a shared slice then connectivity type should be set to any-to-any and is non-editable.
- If it's a L3 slice type (previous step) and a dedicated slice, then only any-to-any and hub-spoke are allowed (point-to-point is not an option for L3 slice types as this connectivity can be built as any-to-any with only two endpoints).
- If it's L2 and dedicated slice and point-to-point, then there should be no option to connect to shared slices.

For Connectivity to Shared Slices: None is the default. Multiple selections are possible referencing multiple pre-created shared slice instances. Also, if this new slice instance is a shared slice, you cannot connect to other shared slices so this section would be grayed out.

Only if "use-as-blueprint" with "customizations allowed" is enabled in the selected Slice Template can we configure a BW value. (Else grayed out).

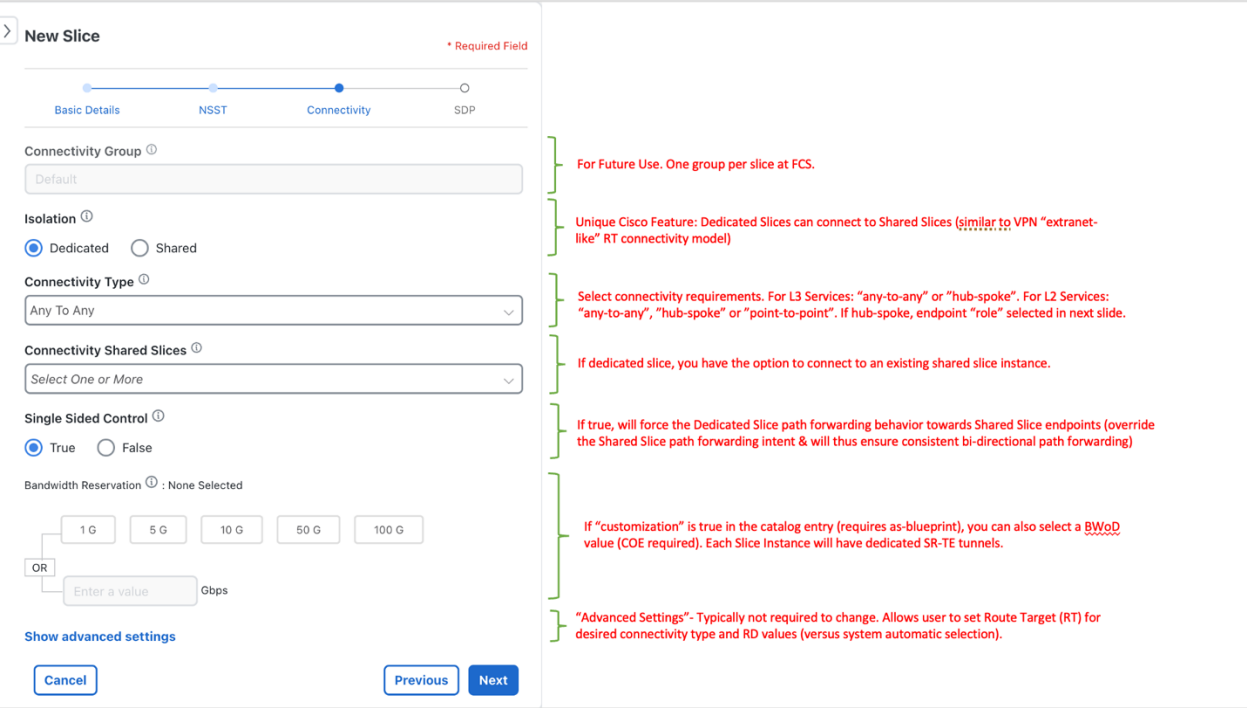

Figure 37: Slice Instance Provisioning- Connectivity Details

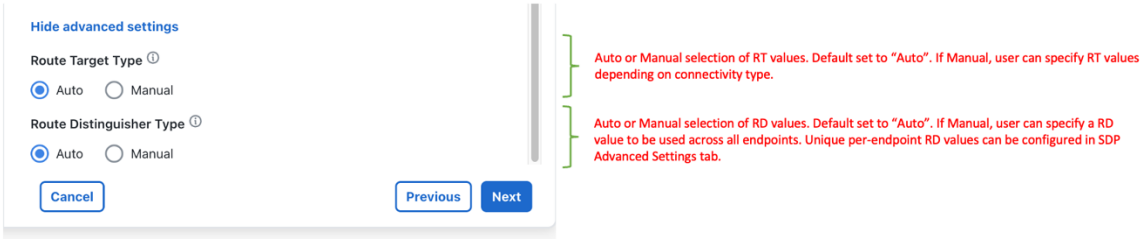

Figure 38: Slice Instance Provisioning- Connectivity Advanced Details

When Advanced Settings box is selected, the user can edit optional parameters. This is only available for non-L2 P2P based slices. This will allow for custom Route Target (RT) and Route Distinguisher (RD) settings. By default, these are set to auto and thus not required to configure. If connectivity type is any-to-any and if manual RT is selected then a box allowing for manual entry of the RT is presented, with this value being used uniformly across all sites to import/export. If the connectivity type is hub-spoke, then there will be two RTs required, one for hub use and one for spoke use.

7.4 Slice Instance Creation- Step 4 Select Endpoints

In this step, the Slice Requester provides endpoint details. These endpoints are called Service Demarcation Points (SDPs) per IETF Slicing standards. The below figure goes through the required steps for a L3 based Slice service. First an SDP ID and an Attachment Circuit ID (AC ID) are configured. This is string data and only needs to be unique within the slice service instance. Next, we provide the Node and Interface details for the PE device for the CE facing interface (i.e., the PE-CE interface). If a VLAN is required, that is specified as "VLAN ID". Since this is a L3 Service Type, we also need the interface IP address and subnet mask and we have the option to configure a peering routing protocol to the CE or not. If peering protocol BGP is selected, we require the neighbor (CEs) IP address and Remote-AS Number (ASN).

If this was a hub and spoke based connectivity service, the endpoint role can also be configured (hub or spoke). Note that endpoints on the same Node must be the same role (hub or spoke).

At this time, you can add additional SDPs by selecting "+ Add Another".

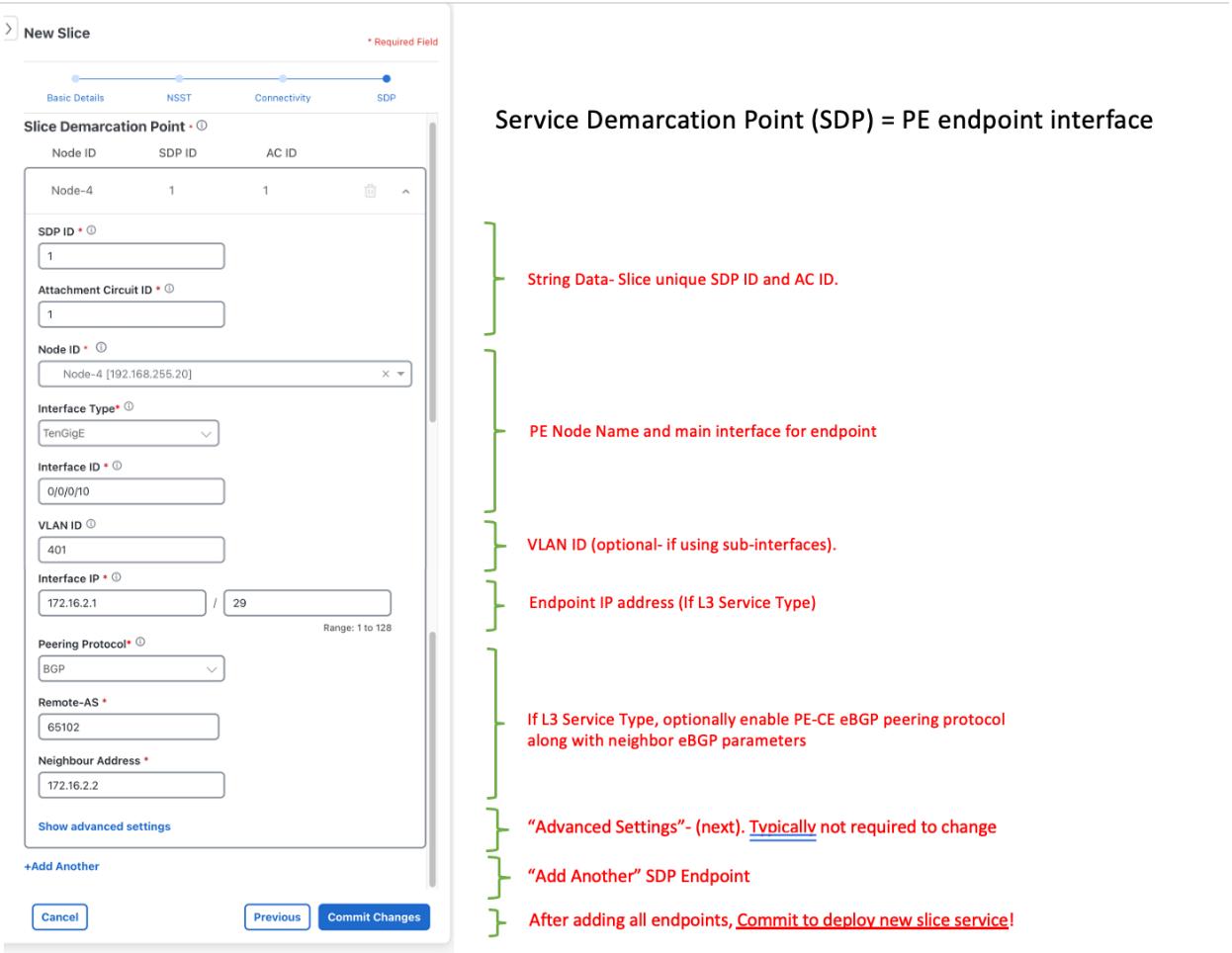

Figure 39: Slice Instance Provisioning- Endpoint Selection

If selected, the below figure shows the SDP Advanced Settings. The need to go into Advanced Settings to change anything is not required, as all the values will be set to default behaviors. The advanced settings tab allows for a unique RD to be set for this endpoint along with additional RTs to be added. If a manual RD was set endpoint-wide in the previous step (Advanced Settings Connectivity), it can be overridden to an SDP specific value here.

The user can add to a list of additional RTs to be included on the SDP. There is an RT-type that is also required (either import, export or both). So, the user can add the RT (string but there is a specific format shown in the "i" field) and select it as import or export type. By default, no additional RTs will be added. Adding these RTs allows for very custom connectivity mappings to other non-slice VPN endpoints (for example an ops center for Network Management).

Match Criteria is for future use and by default this is set to "any-match" and there should be a corresponding connectivity group box (which by default is set to "default"). This maps traffic from this SDP to the default connectivity-group.

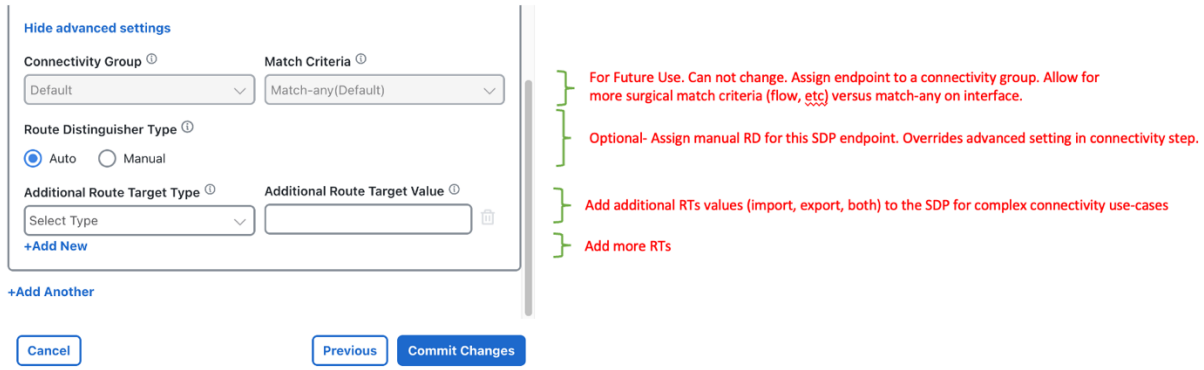

Figure 40: Slice Instance Provisioning- Endpoint Advanced Settings

8 Deployment Results after committing a New Slice Instance

8.1.1 Device level configuration provisioning

The below example shows the specific commands sent to a device when provisioning a L3 any-to-any slice with a Slice Template enabled for SR-PM service assurance.

Figure 41: Detailed Device Provisioning Results

8.1.2 CNC UI results after Slice Instance Provisioning

By clicking on Transport Slices on the left had tool bar, all created slices will be shown on the right-hand list. As of CNC6.0 there is no slice status field showing the health of the slice, yet the underlying VPN and TE components do have a health indicator. The topology view of the selected slice will be shown on the map.

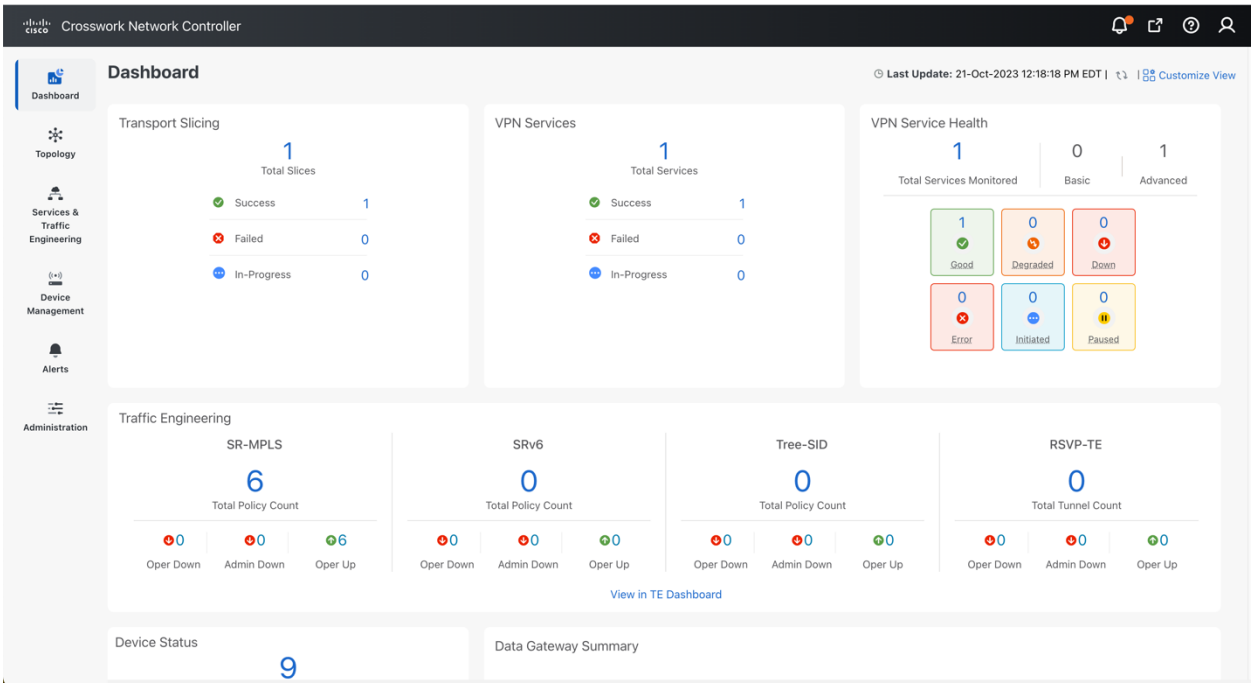

Figure 42: CNC UI- Top Level Dashboard

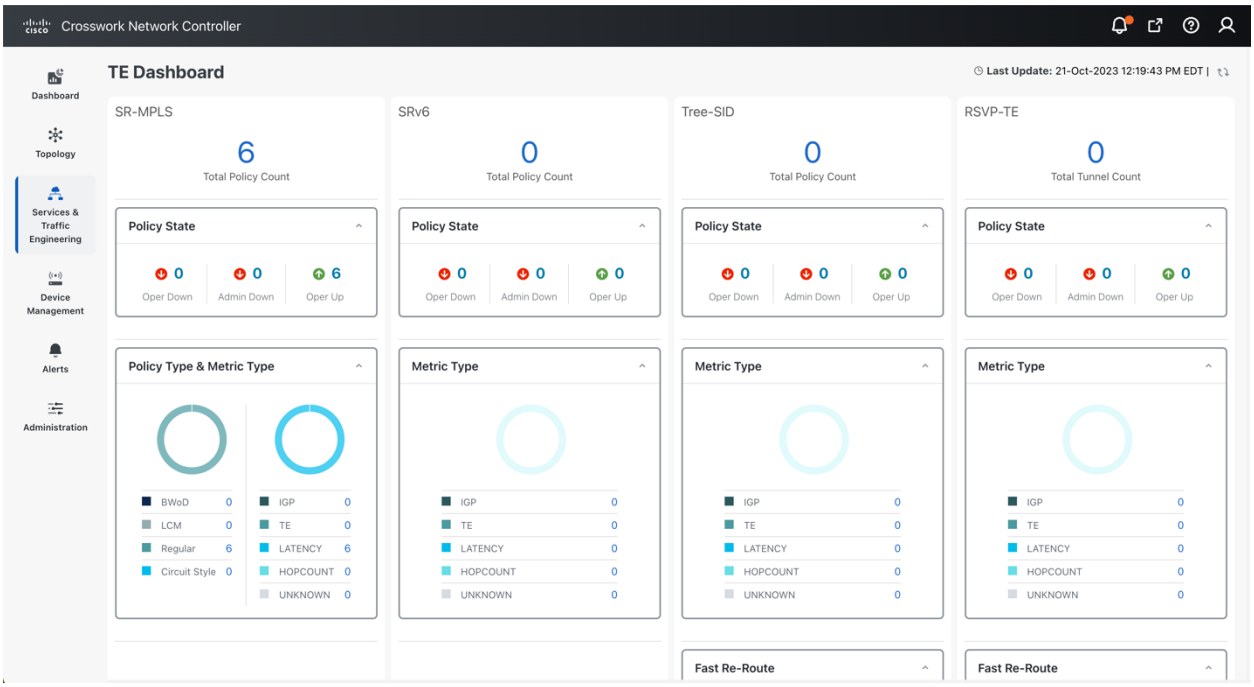

Figure 43: CNC UI- TE Dashboard

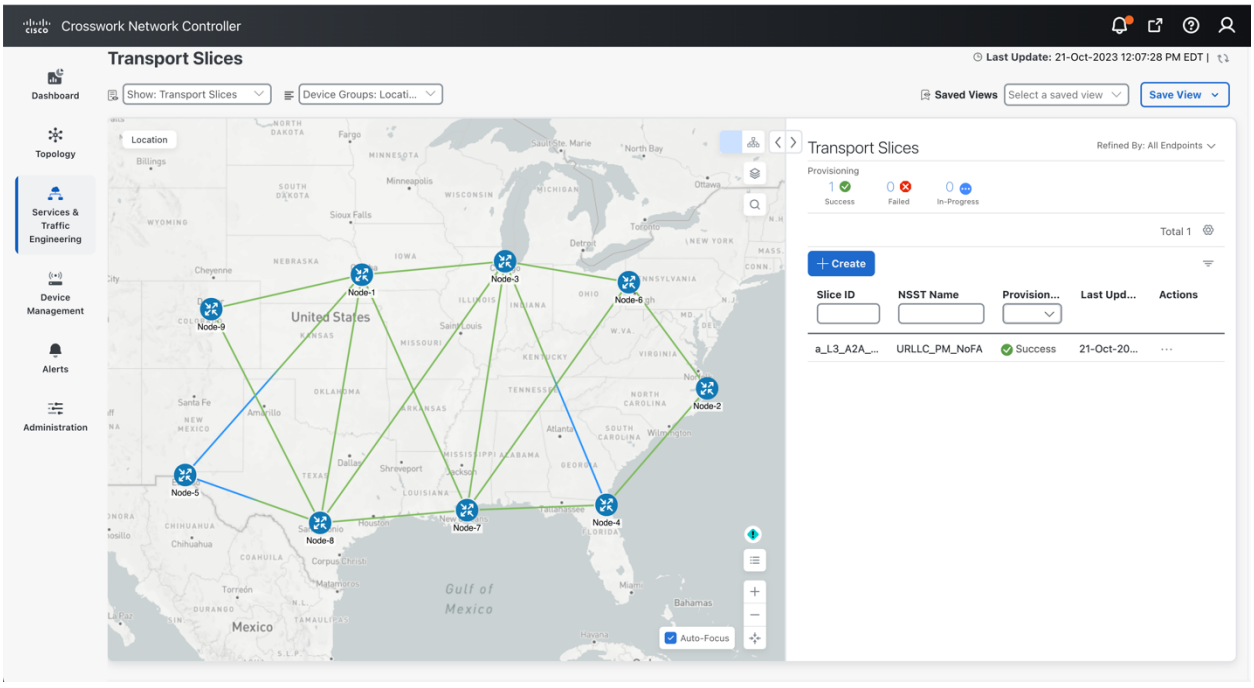

Figure 44: CNC UI- Transport Slice Dashboard

If the view details is selected (via three dots to right of slice selected, then additional information on the various sub-components that make up the slice are shown. Including a summary of the Slice details:

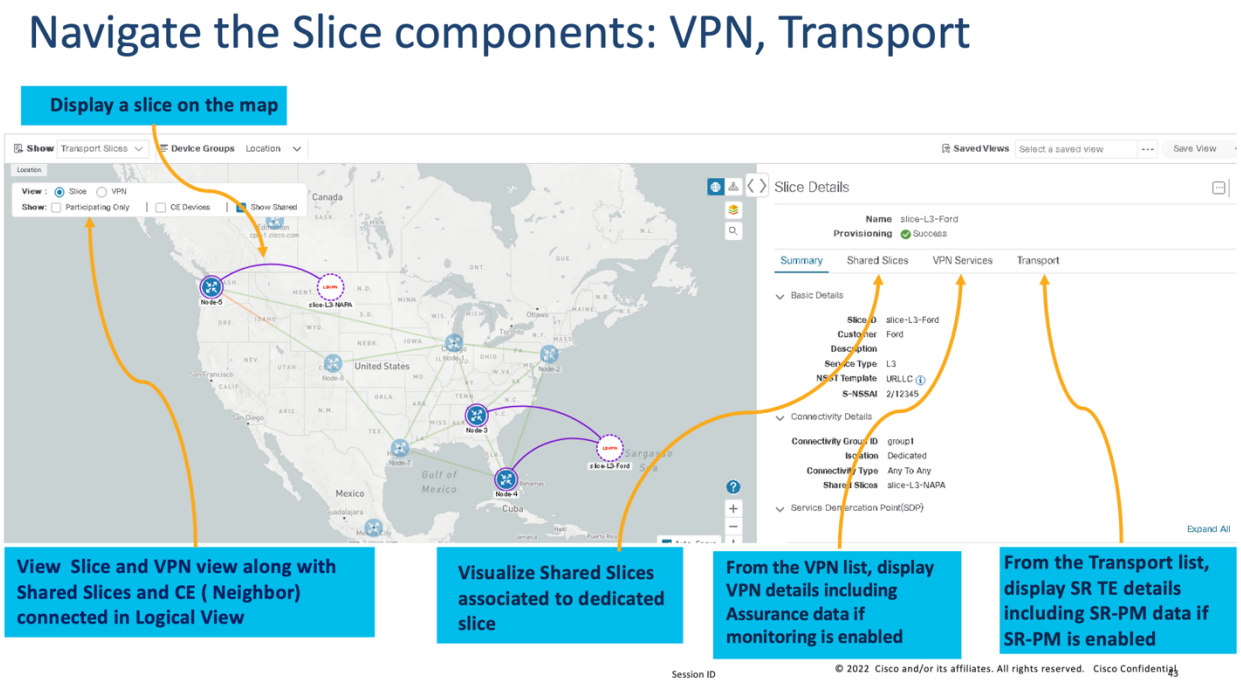

Figure 45 Slice Visualization

If VPN Service tab is selected, the VPN details can be shown. At this stage this is using the standard CNC VPN UI capabilities, including the Assurance graph for Service Health.

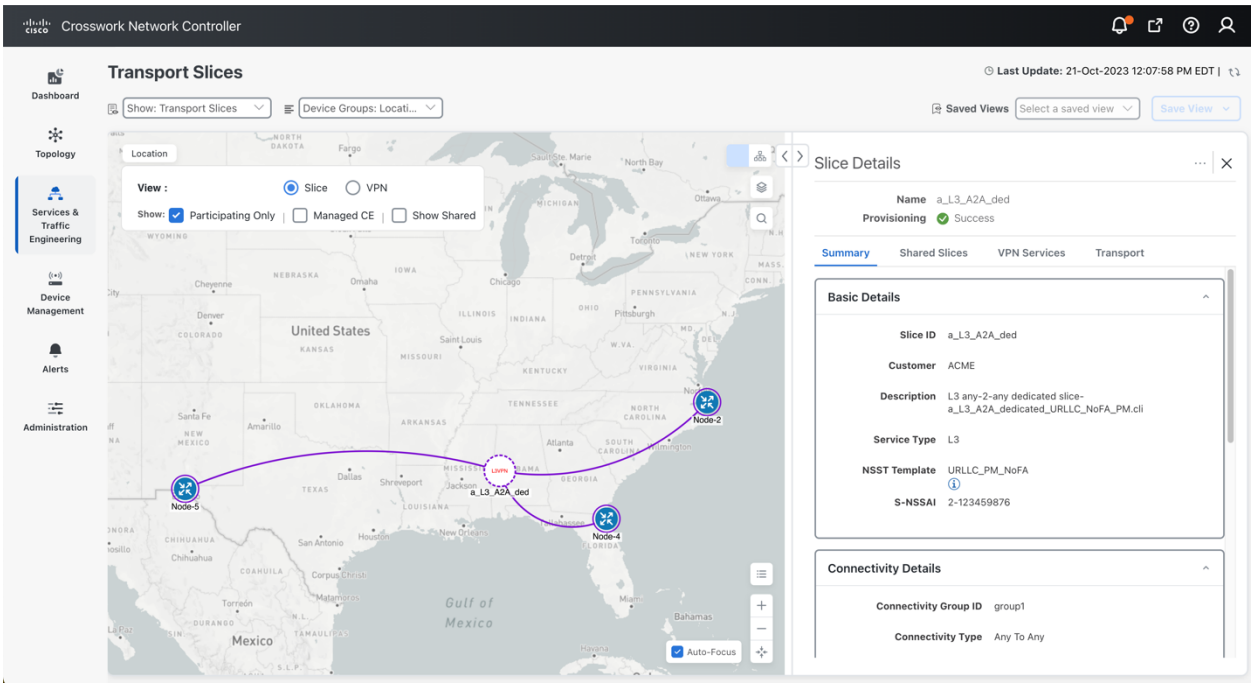

Figure 46: CNC UI- Transport Slice Instance Details

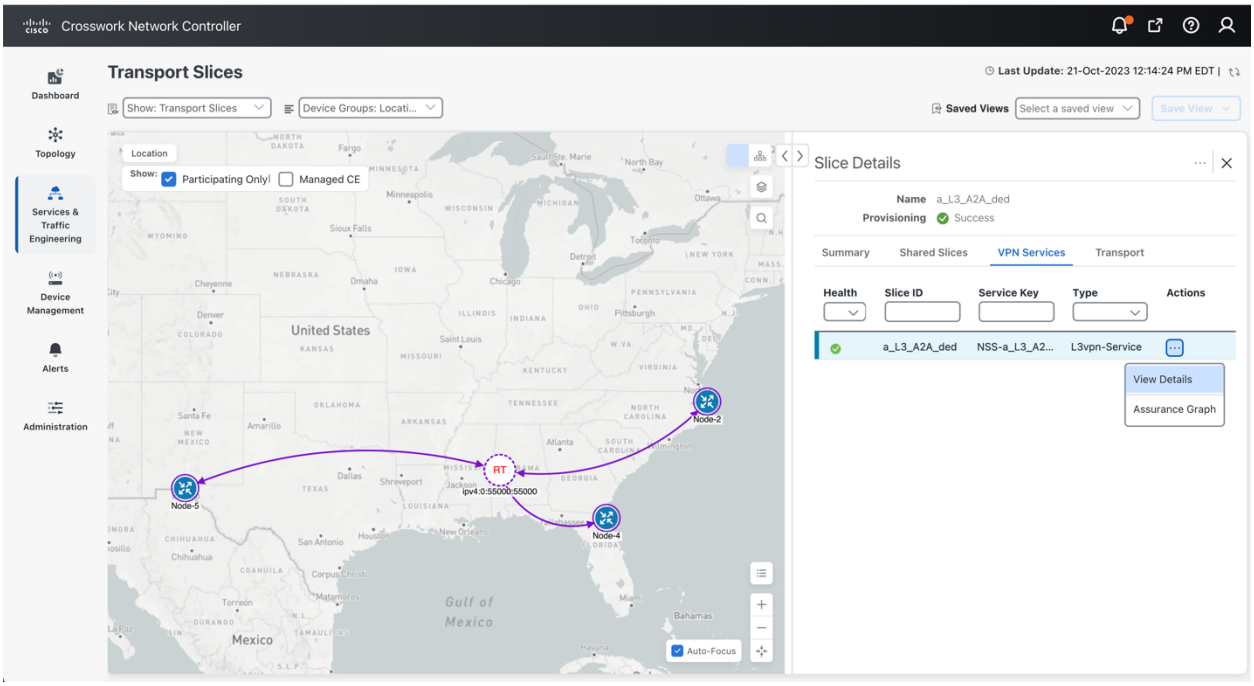

Figure 47: CNC UI- Transport Slice VPN Services

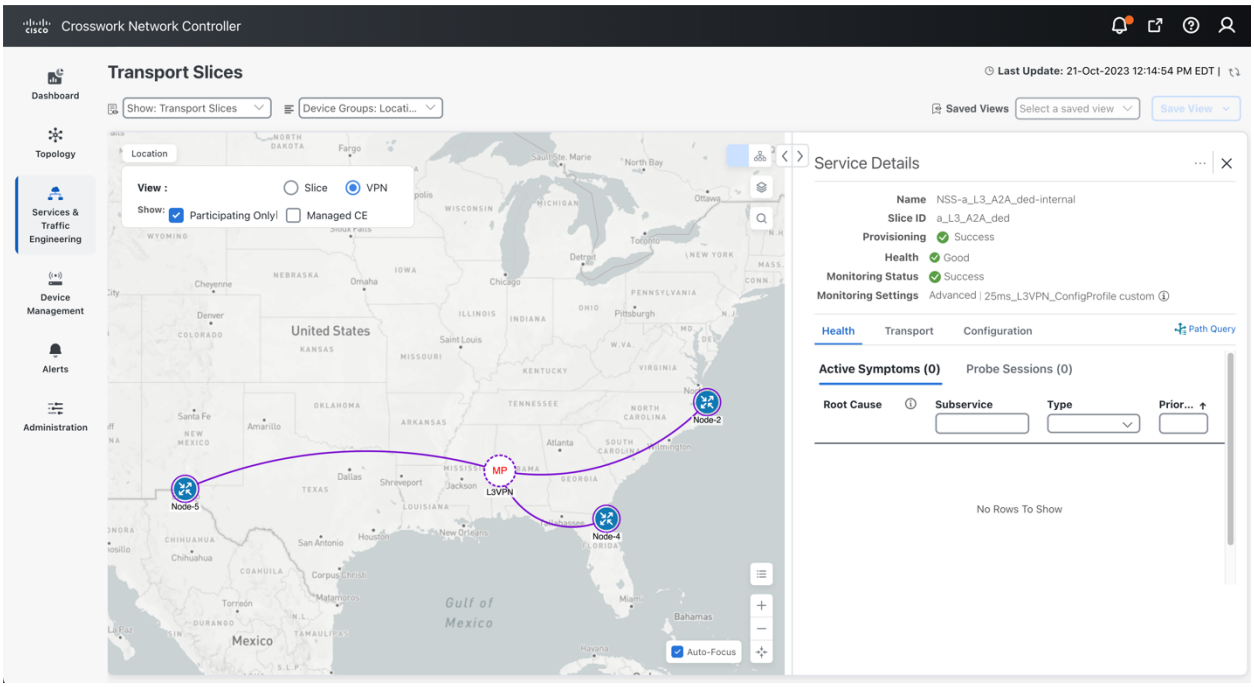

Figure 48: CNC UI- Transport Slice VPN Service Health Details

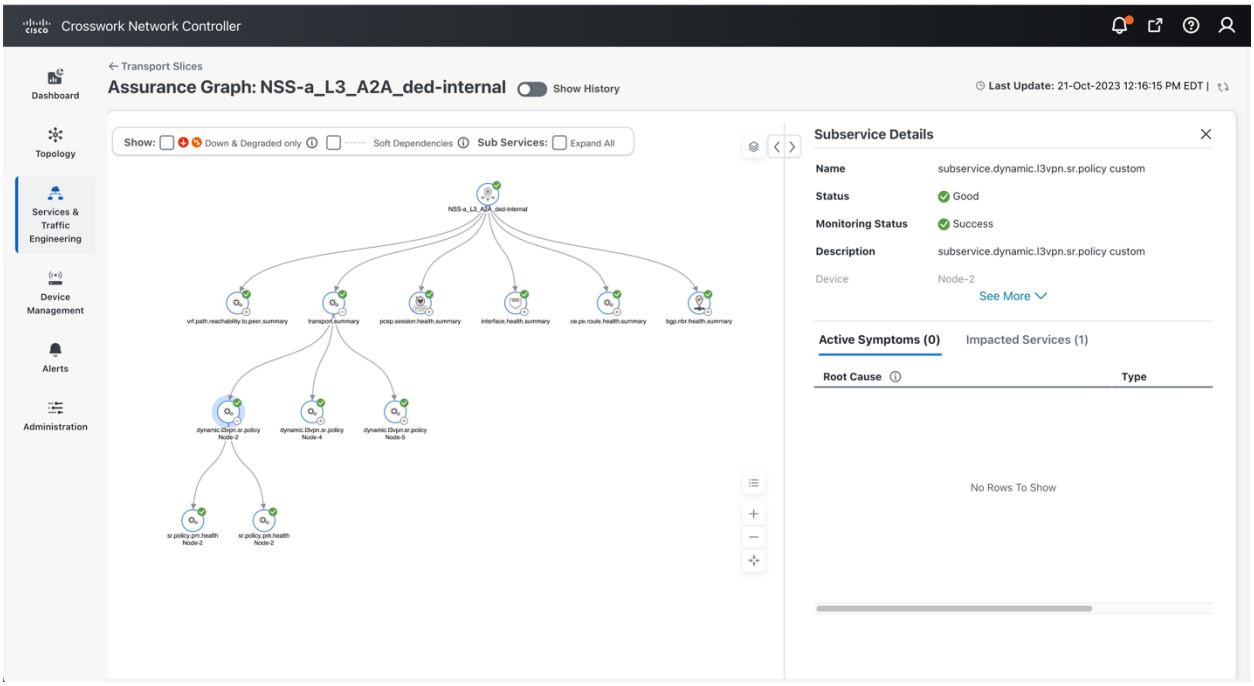

Figure 49: CNC UI- Transport Slice Assurance Graph

If Transport is selected, the SR-TE details are shown, including SR-PM results if enabled.

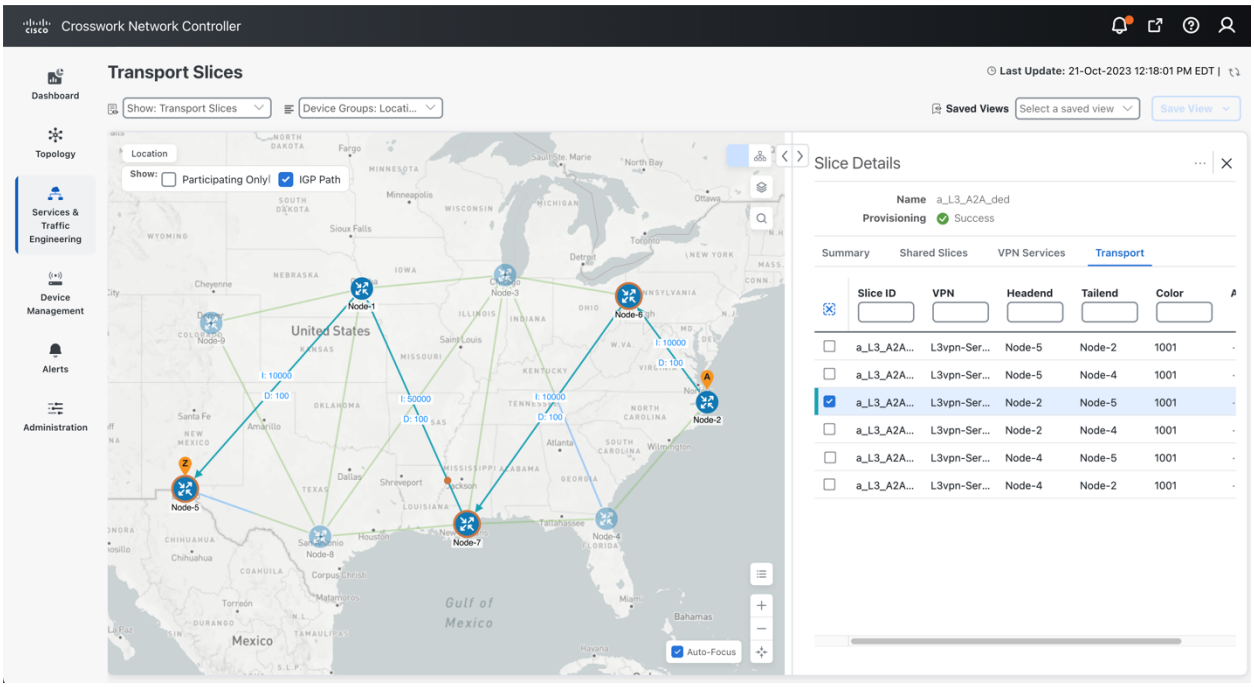

Figure 50: CNC UI- Transport Slice SR-TE Path Forwarding Overview

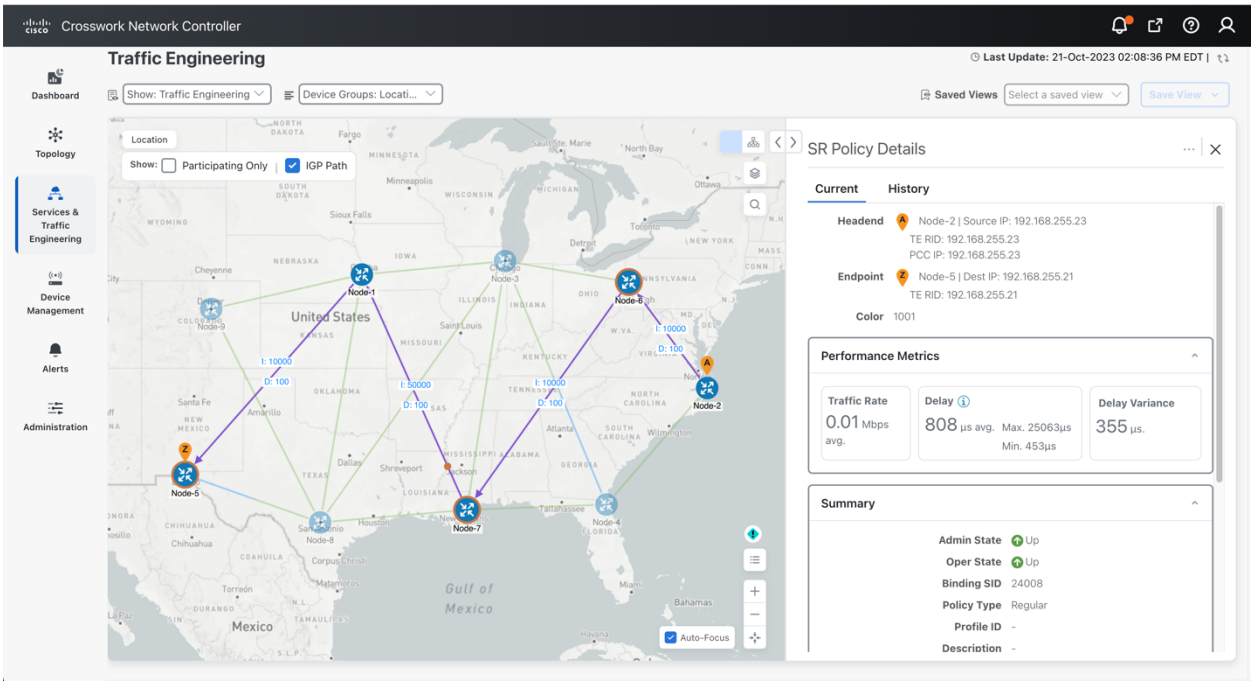

Figure 51: CNC UI- Transport Slice Path Forwarding Details with Performance Metrics

9 Scenario Examples

The following scenarios will use the below reference topology and we will start with some simple transport slice intents and then expand to more complex use-cases. For brevity we will show most slicing configuration objects as NSO CLI payload. All slice scenario payloads (both CLI and JSON) can be pulled from: https://github.com/jmullool/Cisco_Transport_Slicing.

9.1 Reference Topology

We will use the below reference topology with associated link metrics. Notice that all link metrics have both a IGP BW and a Delay component. All link metrics except those noted in Red are set to the default parameters of IGP=10k and Delay=500us. Notice that the link from Node-1 to Node-7 has an extremely high IGP metric (50k), which is considered to be a lower capacity BW when compared to our default link settings (10k) by the routing protocols. While the link from Node-8 to Node-3 has a low IGP metric (5k), which is considered a high-capacity BW setting, it also has a very high Delay (10000us) metric, thus this link will be preferred for most BW based intents, yet non-preferred for Delay based intents. All links are 1Gbps, which will be an important parameter to consider for the BWoD capacity-based intent use -cases.

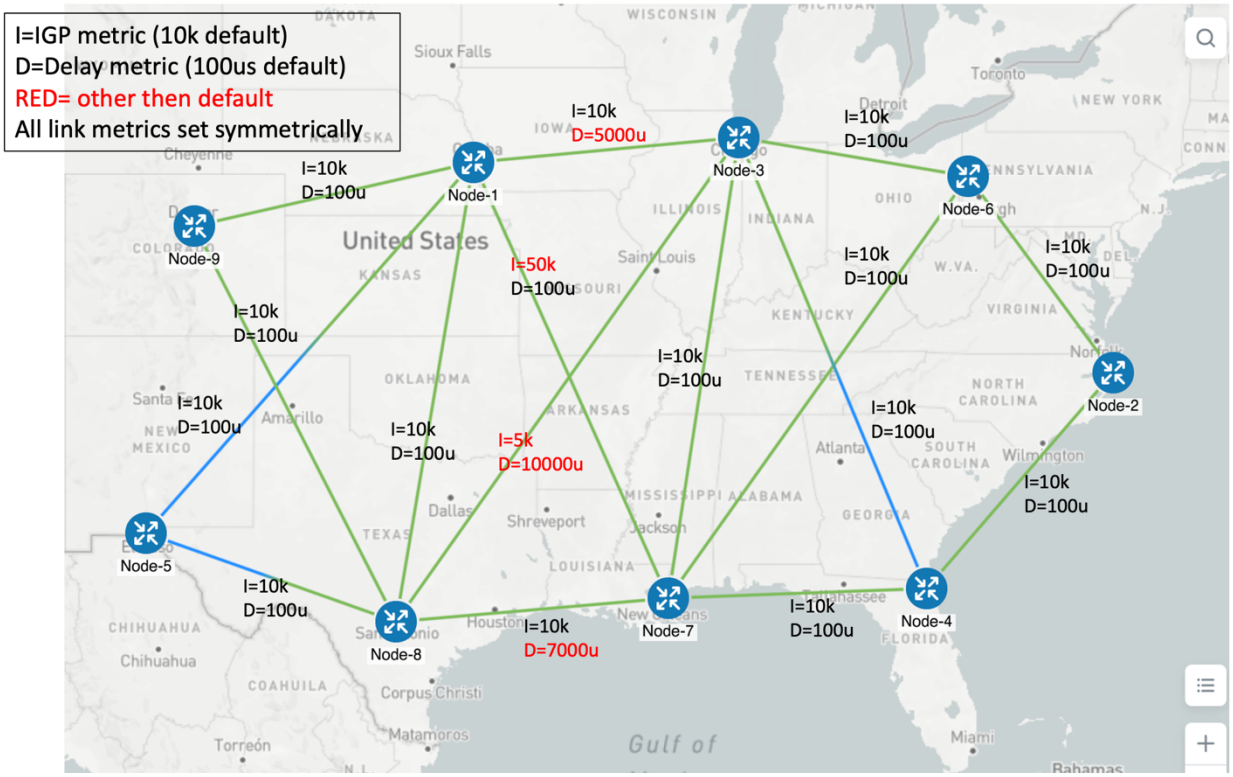

Figure 52: Reference Topology Metrics

Additionally, we have several Flex-Algo topologies built.

9.1.1 Latency based Flex-Algo 128

All nodes and links are included in Flex-Algo 128, which is based on minimum delay link metric. Thus, all links will advertise their point-to-point Delay Performance Measurements (PM) into ISIS for Flex-Algo 128 calculations.

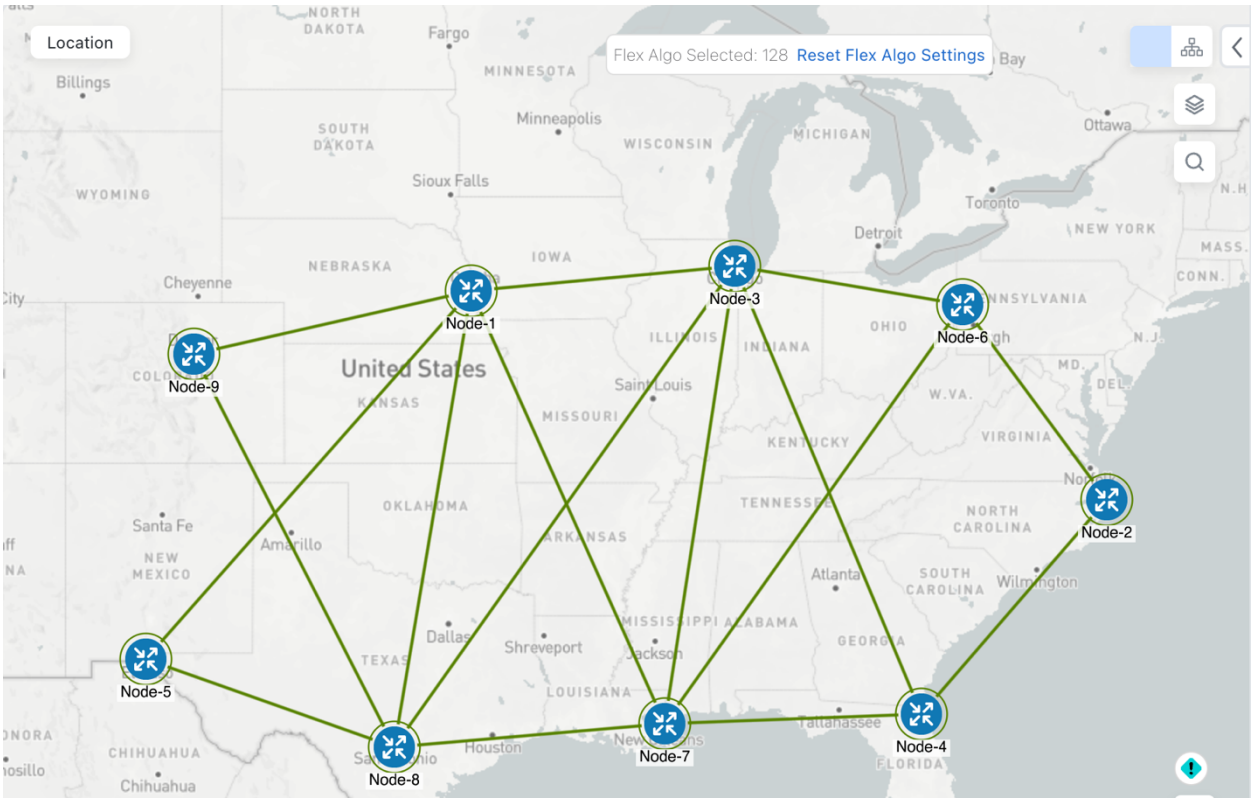

Figure 53 Flex-Algo 129 Forwarding Plane

9.1.2 Encrypted Links based Flex-Algo 129

The administrator has configured all links around the perimeter of the topology with encryption based on MacSec on a point-to-point link basis. Thus, all traffic over these links will be encrypted. Flex-Algo 129 is using link affinity-based coloring. This topology will serve as the basis for any slices requesting encryption intent.

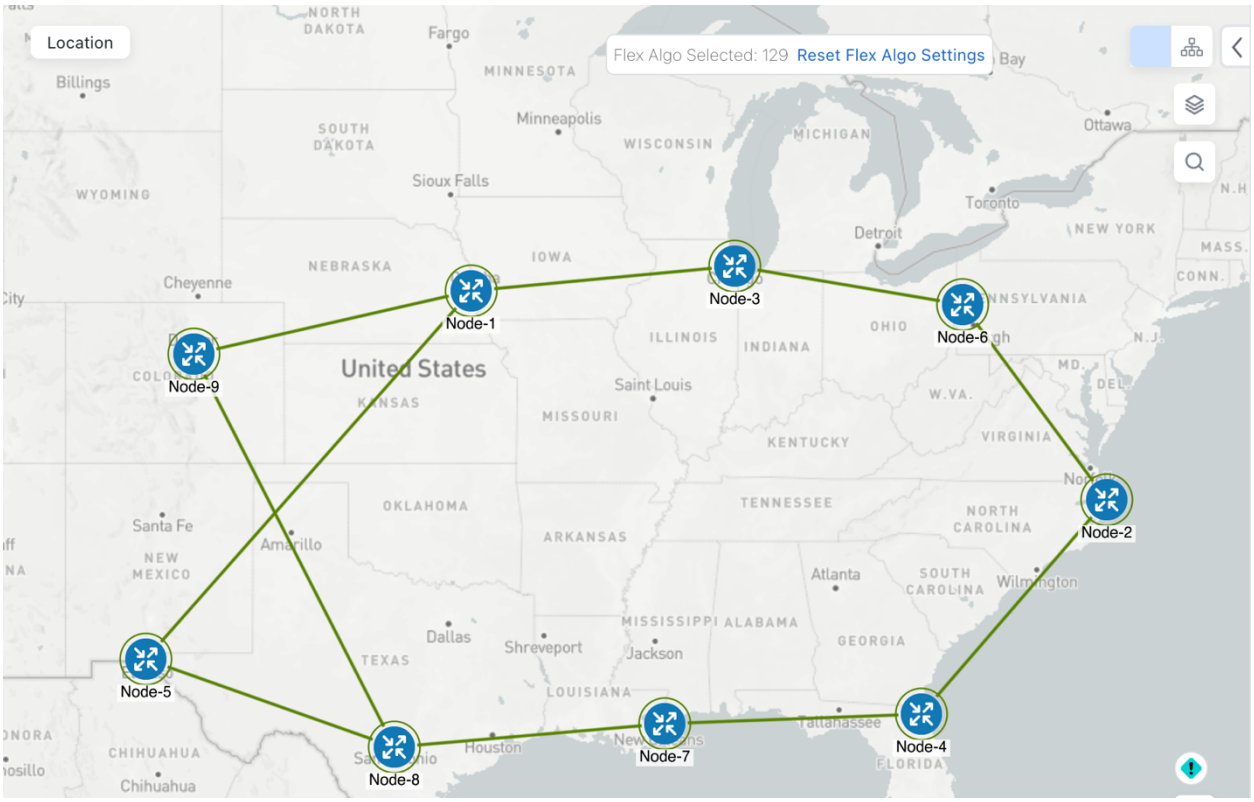

Figure 54 Flex-Algo 129 Encrypted Link Forwarding Planes

9.1.3 Disjoint Path based Flex-Algos 131 and 132

The administrator has also set up a Disjoint-Path topology, again with link-affinities and Flex-Algo. Flex-Algo 131 will be used for “Disjoint-North” path and Flex-Algo 132 will be used for “Disjoint-South”.

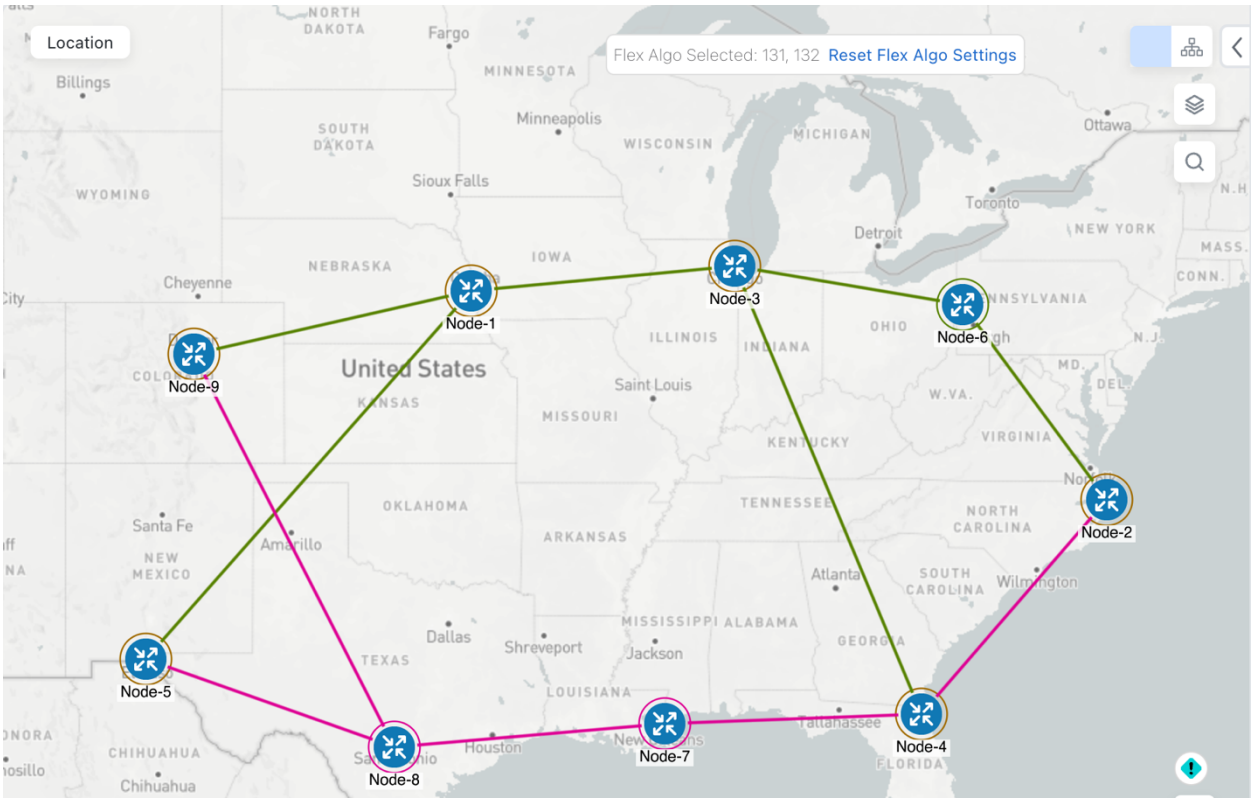

Figure 55 Flex-Algo 131 and 132 Disjoint Path Forwarding Planes

9.2 Basic Scenario Use-cases

9.2.1 Scenario #1- Implement an any-to-any L3 eMBB slice

User Expectation:
In this scenario, the user desires a transport slice which has Layer3 any-to-any connectivity across three endpoints, using the enhanced Mobile Broadband (eMBB) catalog intent.

The eMBB intent will provide the highest BW available path (including proper QoS marking/scheduling treatment), along with some basic service assurance capabilities such as endpoint interface status and PE-CE route health. The user also desires to specify eBGP peer routing connectivity details to their CE devices.

The Slice Designer has already built out the required network capabilities for this intent, we will review them briefly here for this scenario. For more detailed explanations please see the above design section.

9.2.1.1 Slice Service Package Pre-Requisites

Let us first highlight a few optional pre-requisites used by the NSO slice services package that we need to bootstrap into NSO. The need for these pre-requisites are dependent on the types of slices and intents required. First, if we plan on using any "as-blueprint" forwarding-plane-policy-types in our template catalog, then we need to create an NSO sr-color resource pool so that the

slicing service can assign colors for dynamically created ODN policies. This pool should then be referenced by the slicing service (see below). Secondly, if creating point-to-point L2 slice service types, the route-policy map assigned to the PE's BGP session for the route reflector peer is required to be identified to the slicing package. This policy map will be modified by the slicing package with new policies as needed for L2 services which is a standard approach for VPWS.

In this first scenario, these items are not necessarily required since we're using neither of these functions.

```
resource-pools id-pool sr-color-pool
 range start 1000
 range end 2000
!
network-slice-services cfp-configurations color-pool-name sr-color-pool
network-slice-services global-settings parent-rr-route-policy
SET_COLOR_EVPN_VPWS
!
```

9.2.1.2 Path Forwarding Pre-Requisites

The following settings have been pre-configured with the NSO T-SDN SR-TE CFP for the eMBB ODN path-forwarding intent which is based on the igp metric. The slice designer is assigning color 100 to this intent and requesting pce support for selecting the path, along with requesting the path should use the best igp metric.

```
admin@ncs# show running-config cisco-sr-te-cfp:sr-te odn odn-template eMBB
cisco-sr-te-cfp:sr-te odn odn-template eMBB
 color 100
 dynamic pce
 dynamic metric-type igp
!
```

9.2.1.3 QoS Pre-Requisites

As described above the Slice Designer persona should have a good understanding of the network's settings and device capabilities. They should have a well designed and implemented QoS design throughout their network. In the case of QoS treatment for their high BW business services and for example illustration purposes only, the designer has chosen to deploy these services with their network's existing "Class of Service 1" traffic policy (called "ingress_COS1"). The details of this policy are again provider specific, but in this example the policy will not examine or modify the ingress traffic's IP DSCP setting, but simply mark all the traffic with an MPLS experimental bit (EXP) of 1 so that downstream core scheduling can provide the proper BW treatment. On egress from the provider network the designer has chosen a policy called "Egress-High_BW-Apps" which will assign 50% of the bandwidth to Class of Service 1 (COS1) marked traffic.

It is assumed these qos policies are already deployed on all edge PE devices (but not yet on the customer facing interfaces but have been built out and ready for use). Yet, we still need to

identify that these QoS policies are available to the Slice catalog for use for Slice services. The Slice Designer will need to provide that mapping and will need to identify which policies are available for either Layer 2 or Layer 3 slice services, or both. Since QoS policies can be tailored specifically to Layer 3 or Layer 2 traffic (for example matching on L3 DSCP vs L2 ToS bits) the system allows the user to specify the usage. In the case of scenario #1, since all traffic is being marked with EXP=1 regardless of DSCP or ToS, these QoS policies can be applicable to both L2 or L3 services.

```
admin@ncs# show running-config network-slice-services slo-sle-templates qos-catalog
network-slice-services slo-sle-templates qos-catalog L2 output-qos-policy Egress-High_Bw_Apps
 description "High BW egress"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_COS1
 description "Treat all as Business_Data"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-High_Bw_Apps
 description "High BW egress"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_COS1
 description "Treat all as Business Data"
!
```

9.2.1.4 Slice Service Assurance Settings

In this scenario we only have basic service assurance requirements which are based on passive state monitoring (no active probing). We will be using CNC's Service Health (SH) capability and the CNC system's pre-built heuristic packages (ConfigProfiles and Rules) which define the objects to be monitored. In our scenario we want to monitor basic device health and PE-CE route health which are included in the basic system package. When we build our slice catalog, we can define which packages to use for L2 point-to-point, L2 multi-point and/or L3 services. The scenario above is requiring L3 services, but when we create our catalog for the eMBB intent (next step) we also need to consider future slices instances for eMBB services that are L2 service types, thus we can include all these pre-built system heuristic packages in our catalog entry for eMBB now. Since these are all pre-built system packages, no pre-requisite configurations are required. The system heuristic packages can be viewed via the CNC UI by selecting Administration→Heuristic Packages (see below figure).

System Heuristic Profile and Rule names used for eMBB	Usage:
"Silver_L3VPN_ConfigProfile system" "Rule-L3VPN-NM-Basic system	L3 profile-name L3 rule-name
"Silver_L2VPN_ConfigProfile system" "Rule-L2VPN-MP-Basic system"	L2 multipoint profile-name L2 multipoint rule-name
"Silver_L2VPN_ConfigProfile system" "Rule-L2VPN-NM-Basic system"	L2 point-to-point profile-name L2 point-to-point rule-name

Figure 56 System Heuristic Packages

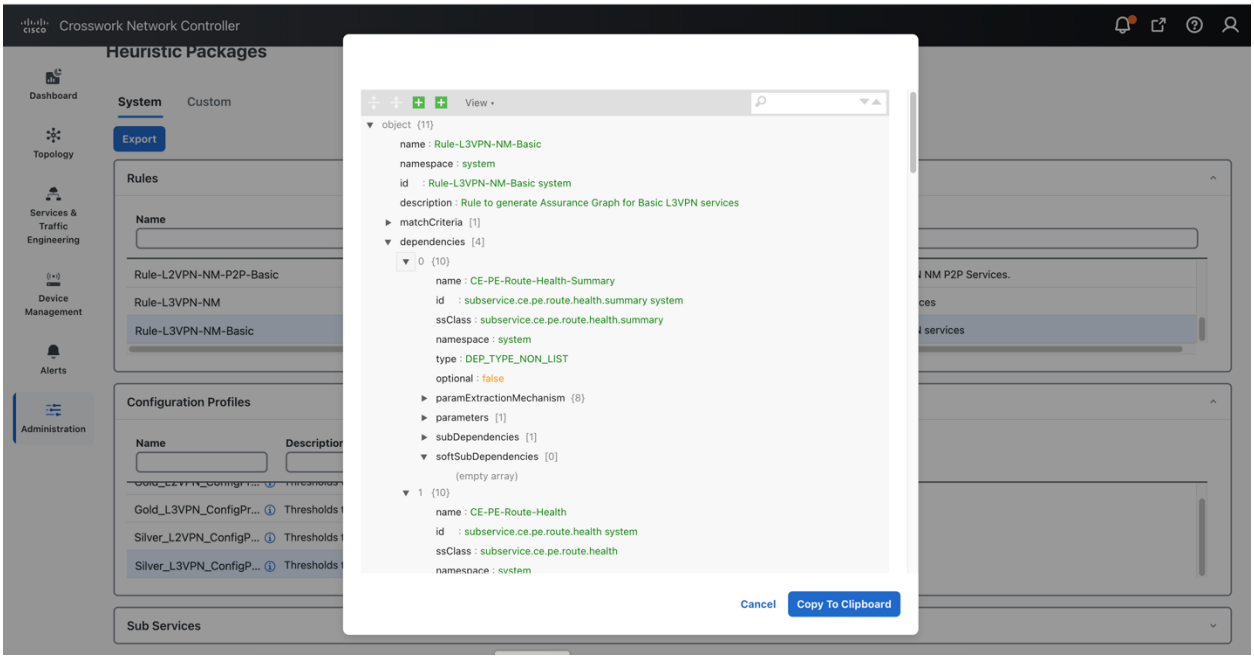

Figure 57 Viewing Heuristic packages via CNC UI

9.2.1.5 eMBB Slice Template Catalog Creation

The Slice Designer will now build out the slice catalog entry for eMBB intent based slice services. This operation is done once and will use the above components as input and can cover both L2 and L3 future slice instance requests. Once complete, we can then move to deploying our slice service instance for the scenario #1 and this template entry will now be available for future slice instances requiring eMBB intent services. (i.e. all of the above pre-requisite steps will not be required again).

Since this catalog entry has Service Assurance requirements, it can only be created via the NSO CLI or the CNC API in the initial CNC release. It will be displayed below via NSO CLI.

```
admin@ncs# show running-config network-slice-services slo-sle-templates slo-sle-template eMBB
network-slice-services slo-sle-templates slo-sle-template eMBB
 template-description "High Bandwidth Service with basic SLA monitoring"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy eMBB
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
!
```

9.2.1.6 Deploying the Scenario #1 Slice Instance

Once the slice type catalog has been created, we can now deploy our scenario #1 slice. Optional and advanced settings exist and are covered in the above design sections. The below table outlines the user data required to deploy this slice. The mandatory data consists of a series of string-data names (user defined), selection of the service type (L2 or L3), catalog intent selection and then defining the Service Demarcation Points (SDPs) which are the PE endpoints facing the customer. These PE endpoints will require IP information since this is a L3 slice and optionally since eBGP was desired for the PE-CE peering protocol, the CE eBGP information.

User Required Data:

Parameter	User Value	Mandatory	Notes
slice-service-name	a_L3_A2A_ded	Y	String-must be unique
description	"any string data"	N	
customer	ACME	N	String meta data- user defined
service-tag	L3	Y	L2 or L3 forwarding
nssai	123459876	N	String meta data- could match 5G nssai assignment if provider desires
slo-sle-template	eMBB	Y	Selection from pre-built slice catalog
isolation	dedicated	N	The default is dedicated- the other option is shared
First SDP endpoint name	1	Y	String- unique within slice instance- At least one SDP must be created, the rest optional.
Node-Name	Node-4	Y	PE Node-Name as defined in CNC topology
Attachment-circuit name	ac1	Y	String- unique within slice instance
Interface-ID	TenGigE0/0/0/10	Y	Customer facing PE Interface
VLAN ID	401	N	VLAN ID if using vlan sub-interfaces
Interface IP	172.16.2.1	Y	PE Interface IP address (since L3 service)
Interface prefix length	29	Y	Interface prefix length (i.e. /29)
Peering protocol	bgp	N	PE-CE peering protocol (bgp or none)
BGP Neighbor Address	172.16.2.2	Y	Since bgp was selected, peer IP address
BGP Neighbor ASN	65102	Y	Since bgp selected, peer ASN
Second SDP endpoint name	2	N	Additional SDPs are optional, but in this scenario we have three endpoints
Node-Name	Node-5		PE Node-Name as defined in CNC topology

Attachment-circuit name	Ac2		String- unique within slice instance
Interface-ID	TenGigE0/0/0/2		Customer facing PE Interface
VLAN	301		VLAN ID if using vlan sub-interfaces
Interface IP	172.16.1.1		PE Interface IP address (since L3 service)
Interface prefix length	29		Interface prefix length (i.e. /29)
Peering protocol	bgp		PE-CE peering protocol (bgp or none)
BGP Neighbor Address	172.16.1.2		Since bgp was selected, peer IP address
BGP Neighbor ASN	65101		Since bgp selected, peer ASN
Third SDP endpoint name	3	N	Additional SDPs are optional, but in this scenario we have three endpoints
Node-Name	Node-2		PE Node-Name as defined in CNC topology
Attachment-circuit name	Ac3		String- unique within slice instance
Interface-ID	TenGigE0/0/0/2		Customer facing PE Interface
VLAN	601		VLAN ID if using vlan sub-interfaces
Interface IP	172.16.3.1		PE Interface IP address (since L3 service)
Interface prefix length	29		Interface prefix length (i.e. /29)
Peering protocol	bgp		PE-CE peering protocol (bgp or none)
BGP Neighbor Address	172.16.3.2		Since bgp was selected, peer IP address
BGP Neighbor ASN	65103		Since bgp selected, peer ASN

Figure 58 Scenario #1 User Required Inputs

9.2.1.6.1 Deploying Slice using CNC UI

Select “Service & Traffic Engineering → Transport Slices” from top-level left-hand menu and then select the “+Create” button on the right. Start with the basic slice details and enter the desired values from the above user input and then select “Next”.

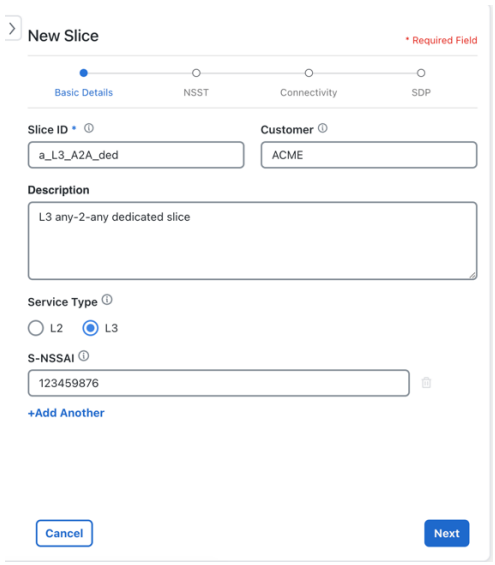

Figure 59 Scenario #1 Provisioning Basic Details

On the Network Slice Template screen, scroll down and select the eMBB catalog entry. (note in the below figure other user created catalog entries are also shown) and then select “Next”.

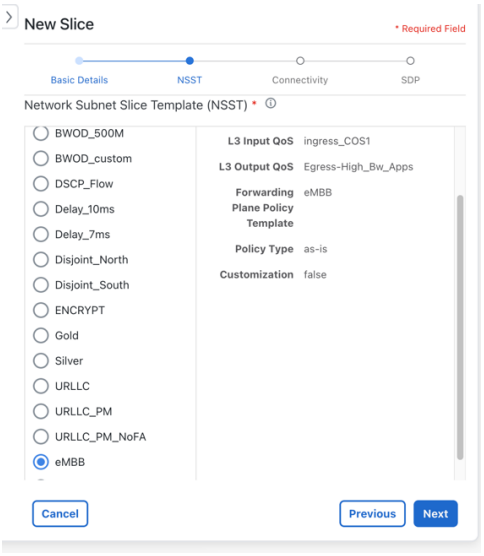

Figure 60 Scenario #1 Provisioning Slice Template Selection

On the Connectivity screen, we are deploying a dedicated slice with any-to-any connectivity. Since there is no requirement to attach to a shared slice (advanced topic), we can leave that selection blank and can ignore the single-sided control setting. Since this eMBB slice is not requiring Bandwidth Reservations (defined in the Slice catalog with customization=true), we are not permitted to select a BW reservation value. Select “Next”.

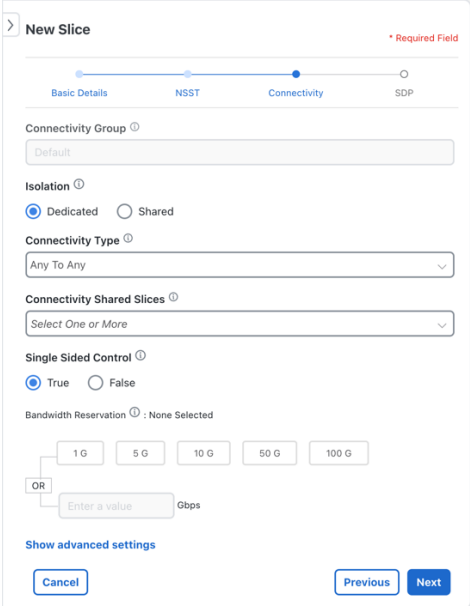

Figure 61 Scenario #1 Provisioning Connectivity Details

On the SDP Screen, enter the endpoint details for the first SDP. When complete, select “+Add Another” for the next two SDPs. When complete, select “Commit Changes”.

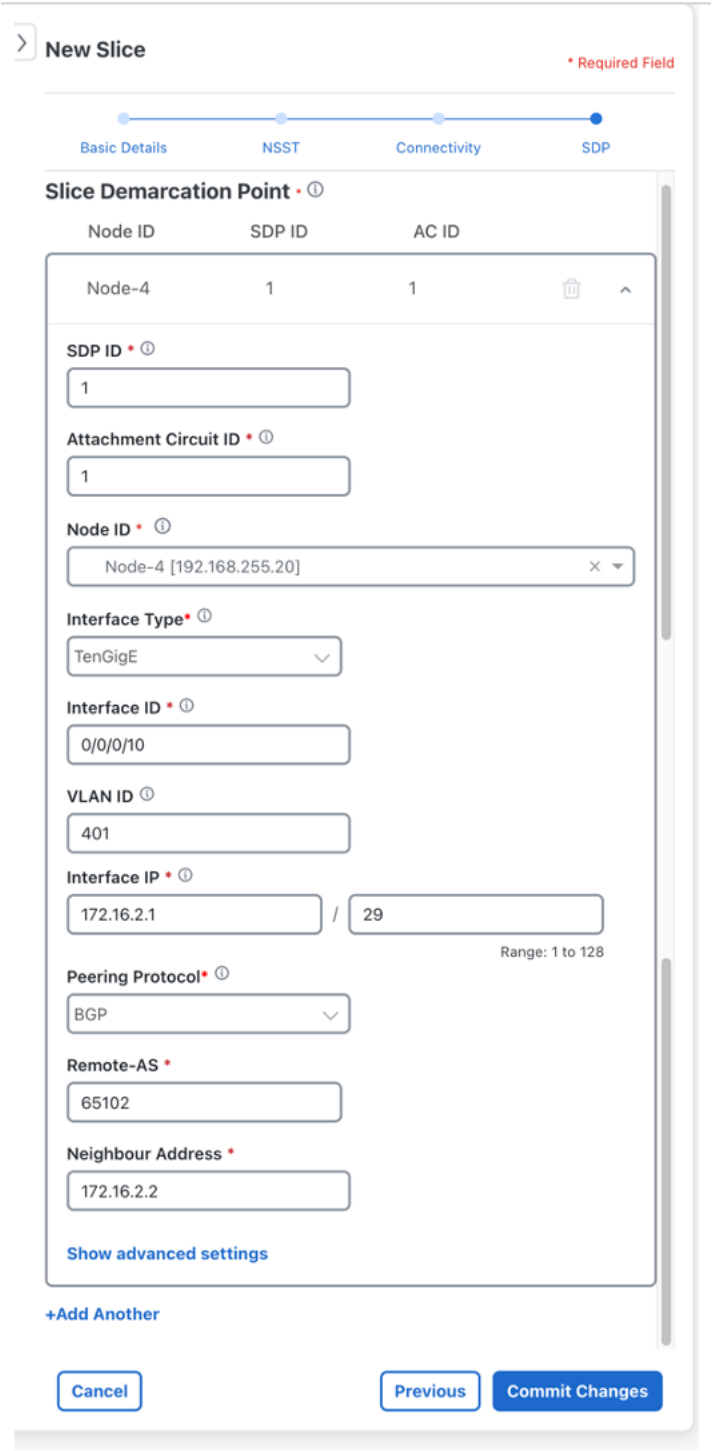

Figure 62 Scenario #1 Provisioning SDP Details

The below NSO CLI payload is another approach to deploy the slice (using “load merge”). Note that defaults are not displayed.

9.2.1.6.2 Deploying Slice using NSO CLI

```
network-slice-services slice-service a_L3_A2A_ded
 service-description "L3 any-2-any dedicated slice- a_L3_A2A_dedicated_eMBB.cli"
 service-tags tag-type service-tag-customer
  value [ ACME ]
 !
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
 service-tags tag-opaque nssai
  value [ 123459876 ]
 !
 slo-sle-template    eMBB
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 sdps sdp 2
  node-id Node-5
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac2
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.1.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 301 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.1.2 ]
    bgp-attributes remote-as 65101
   !
  !
 !
 sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.3.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 601 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.3.2 ]
    bgp-attributes remote-as 65103
   !
  !
 !
 connection-groups connection-group group1
  connectivity-type any-to-any
 !
!
```

Figure 63 Scenario #1 Slice Instance Payload

9.2.1.7 Slice Instance Post-Deployment Checks and Visualization

In the CNC UI, select “Service & Traffic Engineering → Transport Slices” from the top-level left-hand menu and the new slice should be displayed on the right and if successfully deployed the “Provisioning” state should indicate “Success”.

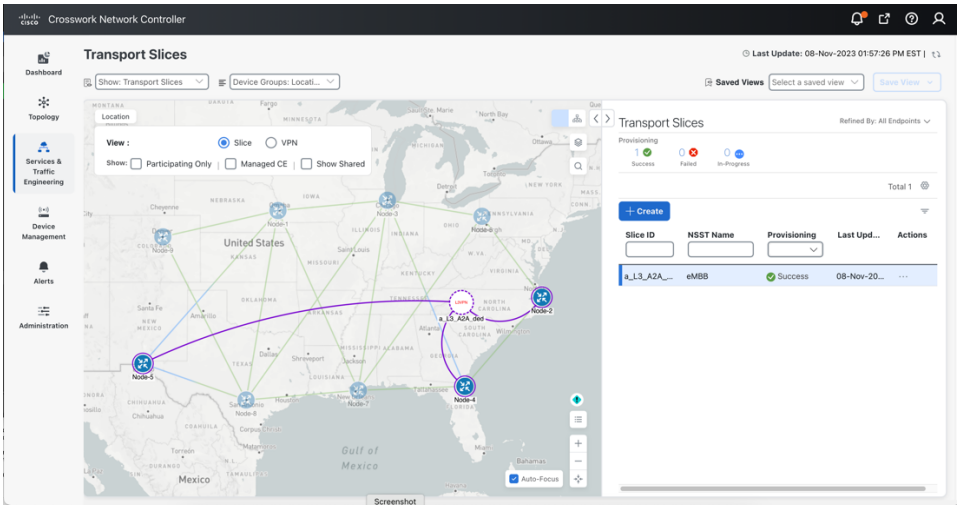

Figure 64 Slice Instance Visualization

Optionally, from the NSO CLI, the slice service state can be verified that all stages were successfully provisioned with all plan states “reached”.

```
admin@ncs# show network-slice-services slice-service-plan a_L3_A2A_ded
                                                                                                  POST
            BACK                STATUS                                                            ACTION
TYPE  NAME  TRACK  GOAL  CODE    NODE    STATE                          STATUS   WHEN                 ref  STATUS
-------------------------------------------------------------------------------------------------------------
self  self  false  -     -       -       init                           reached  2023-10-21T16:06:21  -    -
                                         ready                          reached  2023-10-21T20:51:31  -    -
sdp   1     false  -     -       Node-4  init                           reached  2023-10-21T16:06:21  -    -
                                         ietf-nss-nano:config-apply     reached  2023-10-21T16:06:21  -    -
                                         ready                          reached  2023-10-21T20:51:31  -    -
sdp   2     false  -     -       Node-5  init                           reached  2023-10-21T16:06:21  -    -
                                         ietf-nss-nano:config-apply     reached  2023-10-21T16:06:21  -    -
                                         ready                          reached  2023-10-21T20:51:31  -    -
sdp   3     false  -     -       Node-2  init                           reached  2023-10-21T16:06:21  -    -
                                         ietf-nss-nano:config-apply     reached  2023-10-21T16:06:21  -    -
                                         ready                          reached  2023-10-21T20:51:31  -    -

 plan status color-allocation-data color 100
 plan status service-tag-service [ L3 ]
 plan status forwarding-plane-policy eMBB
 plan status rt-allocation-data hub-rt 0:55000:55000
```

Figure 65 NSO Slice Instance Plan Status

In the below Slice VPN view, we can see that the Slice was provisioned with auto-RTs of 55000:55000 and the service is healthy.

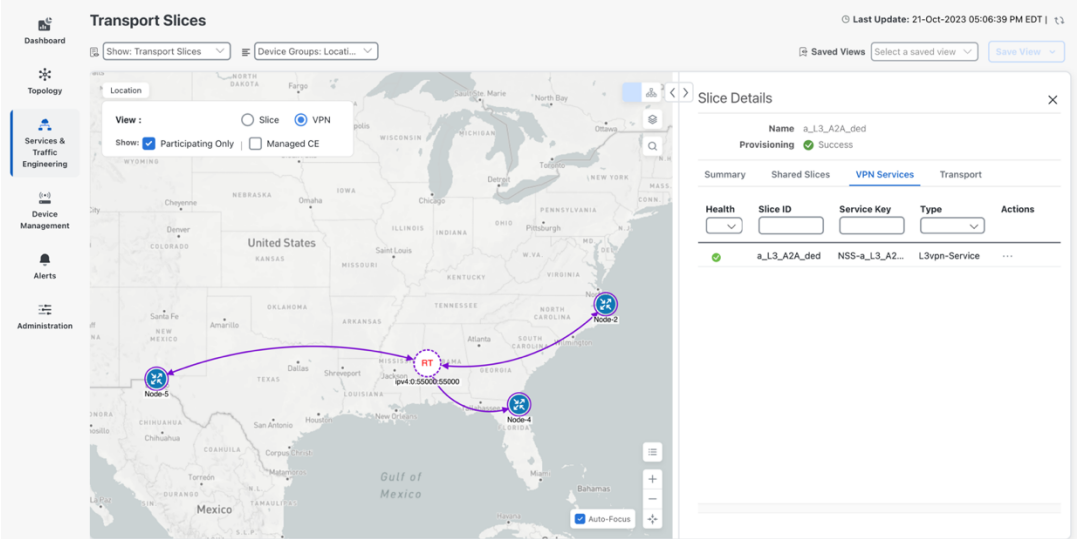

Figure 66 Slice VPN View

By clicking on the three dots in the “Actions” column to the right of the desired Slice ID, you can now select “View Details”:

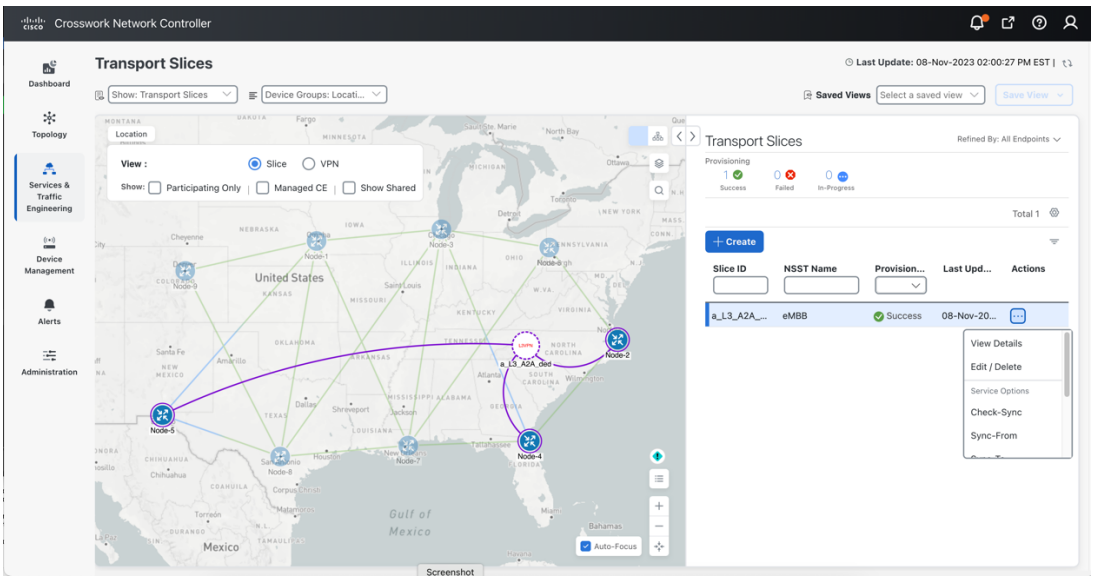

Figure 67 Selecting Slice Instance Details View

From the Summary view, you can see all the Slice details or view the VPN and Transport components of the slice. If this slice is connected to any Shared Slices, that information can also be seen.

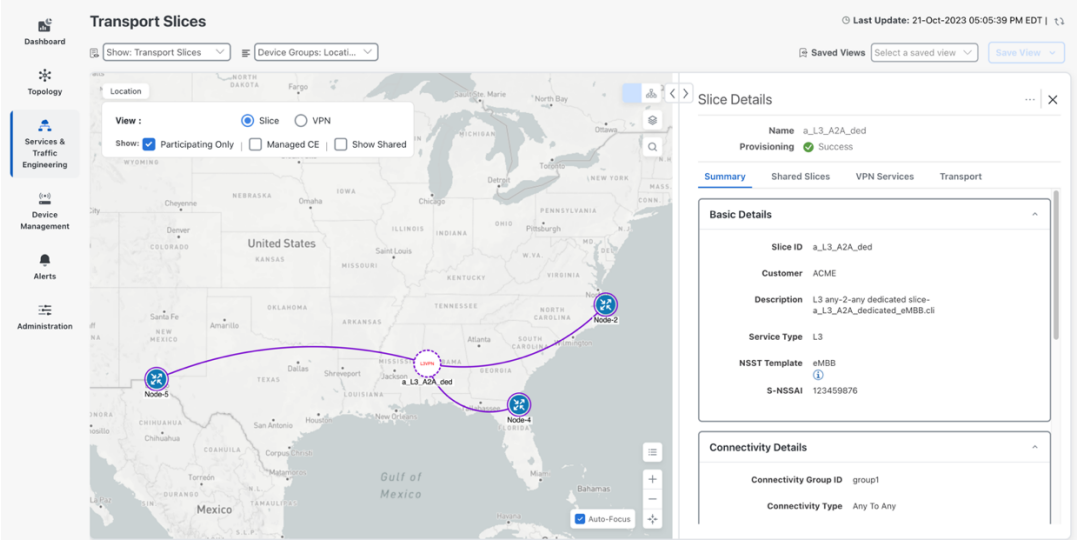

Figure 68 Scenario #1 Slice Summary Details

From the Slice VPN Service view, if we click on the three dots under “Actions” we can select “Assurance Graph” to view the Monitoring Status and the status of the Service Health components defined in the Service Assurance Heuristics Package that was included in the Slice catalog eMBB intent.

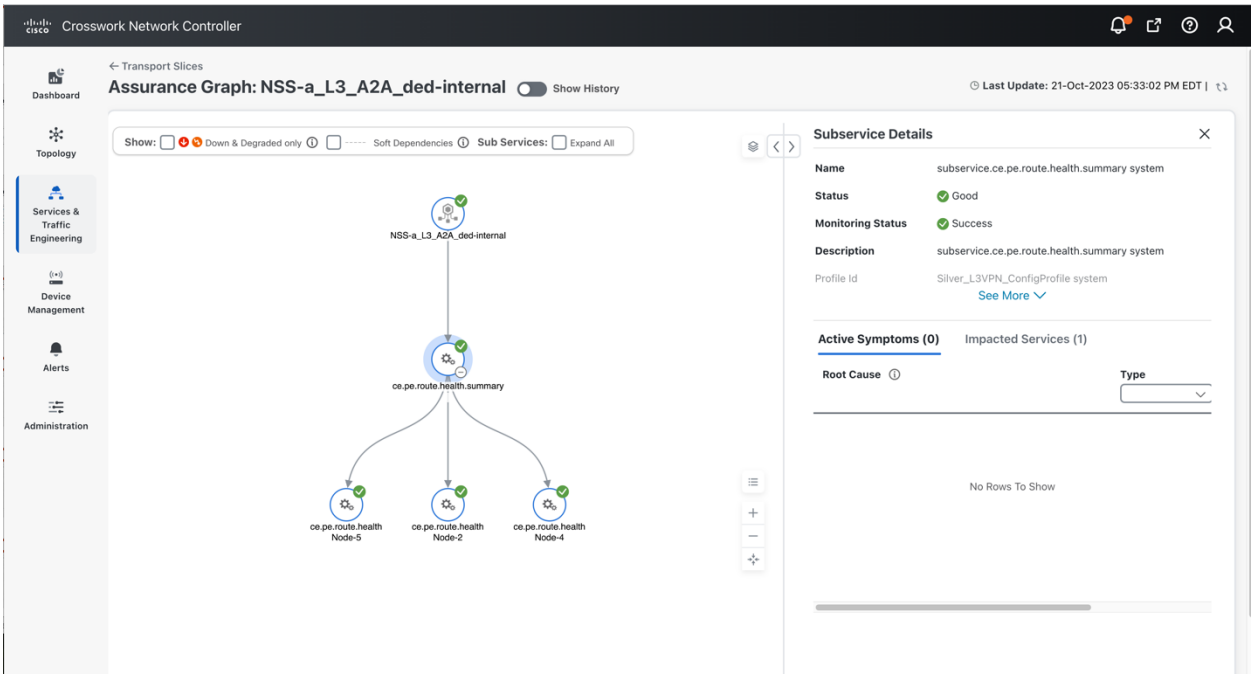

Figure 69 Scenario #1 Slice Service Health

In the below Slice Transport view, we look at a sample forwarding path for slice traffic from Node-2 to Node-5. A few observations: The high BW link between Node-3 and Node-8 (IGP=5k) is used (which was the desired intent), but this link also has a delay of high delay of 10ms (in each direction). Since latency was not an intent objective, this is fine. Also notice that Equal Cost Multi-Pathing is used when available (multiple ECMP paths between Node-2 and Node-3).

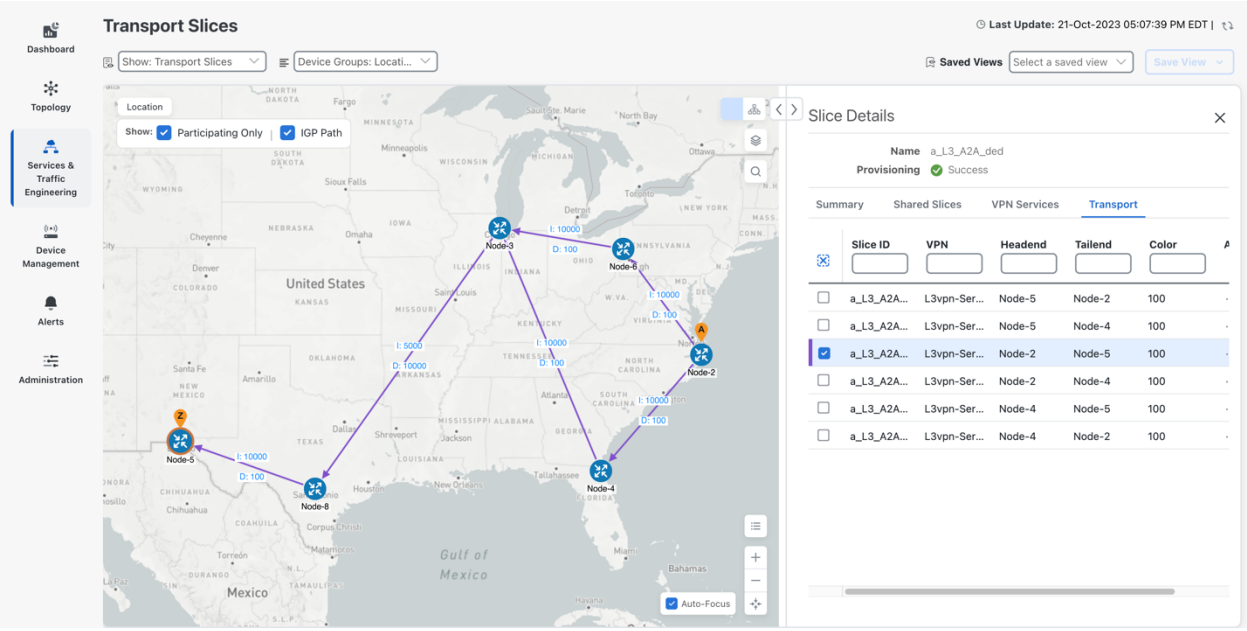

Figure 70 Scenario #1 Slice Transport View for Node-2 to Node-5 Forwarding Path

We also have installed external Accedian probes at the CPE sites connected to the Slice endpoints at Node-2 and Node-5. We can see an ~11ms latency (each way) between the two sites which is accurate as we know that the link between Node-3 and Node-8 has a latency of ~10ms in each direction.

Figure 71 Accedain Skylight Analytics for External Probes between Node-2 and Node-5

9.2.2 Scenario #2- Change the Previous Slice Intent to URLLC_PM

User Expectation:

In this scenario, the same user from above now desires to change his existing transport slice intent which has Layer3 any-to-any connectivity across three endpoints, to an Ultra Reliable Low Latency (URLLC) with Performance Monitoring catalog intent.

The URLLC_PM intent will provide the lowest latency available path (including proper QoS marking/scheduling treatment), along with advanced Performance Monitoring service assurance capabilities which include SR-PM probing.

Before we deploy the change, we need the Slice Designer to add the URLLC_PM intent into the slice template catalog. As previously discussed, these are one-time operations and can be leveraged for all future slice instances using this intent.

9.2.2.1 Path Forwarding Pre-Requisites

The following settings have been pre-configured with the NSO T-SDN SR-TE CFP for the URLLC ODN path-forwarding intent which is based on Flex-Algo 128 which is using the latency metric. The slice designer is assigning color 110 to this intent and requesting pce support for selecting the path.

```
admin@ncs# show running-config cisco-sr-te-cfp:sr-te odn odn-template URLLC
cisco-sr-te-cfp:sr-te odn odn-template URLLC
 color 110
 dynamic pce
 dynamic flex-alg 128
!
```

9.2.2.2 QoS Pre-Requisites

As previously discussed, the Slice Designer has pre-built various QoS policies in the network devices as a pre-requisite. For this use case they have created QoS ingress policy "ingress_COS5" which sets the MPLS EXP to 5, which is then used throughout the core network to give this traffic priority scheduling treatment. They have also defined egress QoS policy "Egress-LowLatency" which also provides priority scheduling this traffic.

The Slice Designer still needs to make these policies available to the Slicing Service package (Note, only the new qos-catalog entries are displayed below).

```
admin@ncs# show running-config network-slice-services slo-sle-templates qos-catalog
network-slice-services slo-sle-templates qos-catalog L2 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_COS5
 description "Treat all as Priority Data"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_COS5
 description "High Priority/Low Latency"
!
```

9.2.2.3 Slice Service Assurance Settings

In this scenario we will have advanced service assurance requirements which include both passive state monitoring along with SR-PM (active probing). We will be using CNC's Service Health (SH) capability and we will customize the CNC system's pre-built heuristic packages (ConfigProfiles and Rules) which define the objects to be monitored and the various delay alarming thresholds. Our objective is to always take the lowest latency path, but we can also set the delay threshold that will trigger an alarm, for our example we will set it at 25ms. (Note, customizing the Heuristic packages is outside the scope of this example).

Custom Heuristic Profile and Rule names used for URLLC_PM	Usage:
"25ms_L3VPN_ConfigProfile custom" "Rule-L3VPN-NM custom	L3 profile-name L3 rule-name
"25ms_L2VPN_ConfigProfile custom" "Rule-L2VPN-MP custom"	L2 multipoint profile-name L2 multipoint rule-name
"25ms_L2VPN_ConfigProfile custom" "Rule-L2VPN-NM custom"	L2 point-to-point profile-name L2 point-to-point rule-name

Figure 72 Custom Heuristic Packages

In addition to the above Heuristic packages, we also will need to create Performance Measurement profiles which will be used on the SR-TE tunnels and while we're at it, we can create a Y.1731 profile for L2 point-to-point slices using the URLLC_PM intent.

The below PM profiles (used for SR-PM) and y-1731 profiles (used for L2 P2P) have been pre-created into the NSO PM service package.

```
admin@ncs# show running-config cisco-pm-fp:pm

pm pm-profiles delay-profile sr-policy profile 2wayDelay
 probe computation-interval 60
 probe tx-interval    30000
 probe protocol       twamp-light
 probe measurement-mode one-way
 advertisement periodic interval 60
 advertisement periodic threshold 20
 advertisement periodic minimum-change 1000
!
pm svc-profiles 2wayDelaySvc
 performance-measurement delay-profile sr-policy profile 2wayDelay
 !
!
```

```
admin@ncs# show running-config l2vpn-ntw
y-1731-profile

l2vpn-ntw y-1731-profile Profile-Delay-1
 schedule interval 5
 schedule duration 5
 type delay
 probe measurement-interval 60
 delay-params statistic delay-two-way
 !
 delay-params statistic jitter-two-way
 !
!
```

9.2.2.4 Scenario #2 Slice Template Catalog Creation

The Slice Designer will now build out the slice catalog entry for the URLLC_PM intent-based slice service. Just like before, this operation is done once, and will use the above components as input and can cover both L2 and L3 slice instance requests. In this scenario, we have decided to use SR-TE Performance Measurement (SR-PM), which will require that the ODN template specified for path forwarding to be modified with the SR-PM commands. As we discussed in the design session, this will require us to use the "as-blueprint" setting to tell the slice package to make a copy of the referenced ODN template (URLLC_PM) and assign a new color (automatically) so the slice automation code can then add the new SR-PM configurations.

Since this catalog entry has Service Assurance requirements, it can only be created via the NSO CLI or the CNC API in the initial release. It will be displayed below via NSO CLI.

```
network-slice-services slo-sle-templates slo-sle-template URLLC_PM
 template-description "Flex-Algo based Low latency Service with Performance Measurement Delay Probing"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  URLLC
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "25ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 2wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 2wayDelay
!!
```

9.2.2.5 Deploying the new Intent to the existing Slice Instance

Once the URLLC_PM slice type catalog has been created, we can now update our scenario #1 slice to scenario #2. In this scenario, all user data previously defined is still in place. The only new data is to select the new template intent.

User Required Data:

Parameter	User Value	Mandatory	Notes
slice-service-name	a_L3_A2A_ded	Y	String-must be unique
slo-sle-template	URLLC_PM	Y	Selection from pre-built slice catalog

Figure 73 Scenario #2 User Required Data

One of the value propositions of the Cisco Slicing solution is the use of NSO's declarative provisioning capabilities, where the system will compare the proposed new configuration to the existing configuration and only make the changes needed to reconcile the differences. This is called the "minimum-diff" and it's an important capability for declarative systems as it's not a "rip and replace". This capability is utilized in this this scenario where we will change the Slice Template of the previously implemented eMBB L3 any-to-any Slice to the URLLC_PM slice intent.

In this case, the system will determine the required changes and push them into the devices automatically. These changes will include re-coloring the VPN prefixes for automated steering over the low-latency based Flex-algo 128 path, applying new SR-TE policies, along with enabling delay-based Performance Measurements probing, telemetry collection and monitoring the delay thresholds (with new Advanced Heuristic package thresholding).

This upgrade can be done either via NSO CLI, CNC API or CNC UI. Below are examples of the NSO CLI and CNC UI approaches.

9.2.2.5.1 Editing Slice using CNC UI

Select "Service & Traffic Engineering → Transport Slices" from top-level left-hand menu and then select the three dots under "Actions" and select "Edit". Move to the Network Subnet Slice Template (NSST) selection screen and select the "URRLC_PM" intent. Continue to hit "Next" until you come to the last screen to "Commit Changes".

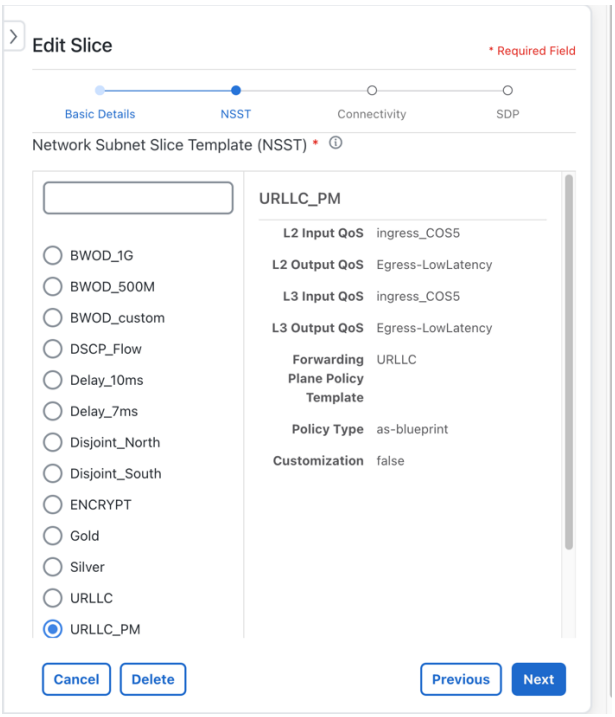

Figure 74 Dynamically changing Slice Intent via UI

9.2.2.5.2 Editing Slice using NSO CLI

```
network-slice-services slice-service a_L3_A2A_ded
 service-description "L3 any-2-any dedicated slice- upgrade to URLLC with SRPM"
 slo-sle-template    URLLC_PM
!
```

Figure 75 Dynamically changing Slice Intent via NSO CLI

After committing the change, the slice service will be updated to the new Intent. If we now examine the SR-TE path forwarding between Node-2 and Node-5, we will see the path changed to use the low delay link (yet high IGP metric) between Node-1 and Node 7 to meet the low delay intent. We can also see that SR-PM measurements have been enabled and we are getting both Delay (517us one-way) and Delay Variance measurements that meet our objective.

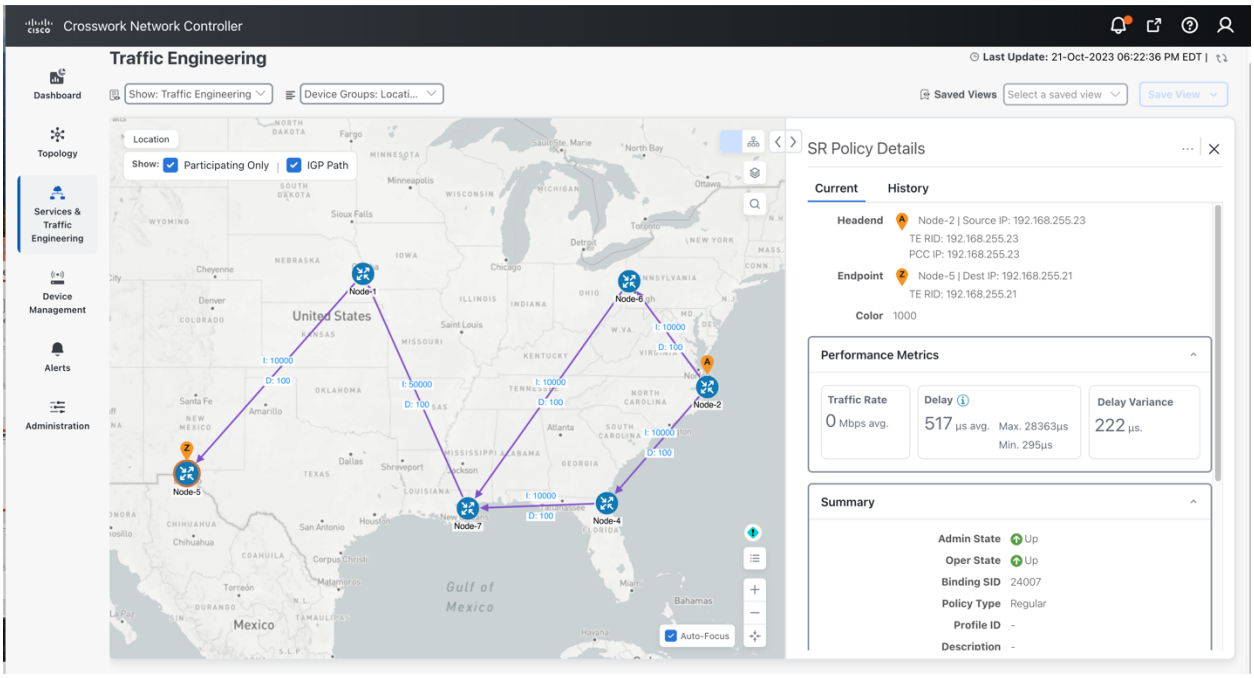

Figure 76 Low Delay SR-TE path with SR-PM

We can also see with the external Accedian probes the one-way delay has dropped from 11ms to 1ms.

Figure 77 Accedian External probe Measurements

9.3 Additional Scenario Use-cases

The above two basic scenarios walked through in great detail both the pre-requisite steps and the slice instance creation details, including CNC UI screenshots. For the following scenarios we will use the following consolidated pre-requisite material and slice catalog. We will also only show NSO CLI payloads for slice instance creation.

9.3.1 SR-TE ODN Settings

The following settings have been pre-configured with the NSO T-SDN SR-TE CFP for various ODN path-forwarding intents.

```
admin@ncs# show running-config cisco-sr-te-cfp:sr-te odn odn-template
```

```
cisco-sr-te-cfp:sr-te odn odn-template BWOD_1G
 color     121
 bandwidth 1000000
 dynamic pce
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template BWOD_500M
 color     122
 bandwidth 500000
 dynamic pce
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template BWOD_custom
 color 120
 dynamic pce
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template DSCP_Flow
 color 500
!
cisco-sr-te-cfp:sr-te odn odn-template Disjoint_North
 color 116
 dynamic pce
 dynamic flex-alg 131
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template Disjoint_South
 color 117
 dynamic pce
 dynamic flex-alg 132
 dynamic metric-type igp
!
```

```
cisco-sr-te-cfp:sr-te odn odn-template ENCRYPT
 color 115
 dynamic pce
 dynamic flex-alg 129
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template URLLC
 color 110
 dynamic pce
 dynamic flex-alg 128
!
cisco-sr-te-cfp:sr-te odn odn-template URLLC_NoFA
 color 111
 dynamic pce
 dynamic metric-type latency
!
cisco-sr-te-cfp:sr-te odn odn-template eMBB
 color 100
 dynamic pce
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template eMBB_10ms
 color 130
 dynamic metric-type igp
!
cisco-sr-te-cfp:sr-te odn odn-template eMBB_7ms
 color 131
 dynamic metric-type igp
```

9.3.2 QoS catalog settings

The following QoS policy names have been pre-placed into the network edge devices with administrator defined settings for ingress and egress QoS treatment. They need to be included as permitted qos policies options for the Slicing Template qos-catalog.

```
admin@ncs# show running-config network-slice-services slo-sle-templates qos-catalog
```

```
network-slice-services slo-sle-templates qos-catalog L2 output-qos-policy Egress-High_Bw_Apps
 description "High BW egress"
!
network-slice-services slo-sle-templates qos-catalog L2 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_COS1
 description "Treat all as Business_Data"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_COS3
 description "Treat all as Video Data"
!
```

```
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_COS5
 description "Treat all as Priority Data"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_TRUST_L2
 description "Trust COS markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L2 input-qos-policy ingress_UNTRUST
 description "Do Not Trust COS  markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-High_Bw_Apps
 description "High BW egress"
!
network-slice-services slo-sle-templates qos-catalog L3 output-qos-policy Egress-LowLatency
 description "Low Latency priority egress"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_COS1
 description "Business Data"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_COS3
 description Video
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_COS5
 description "High Priority/Low Latency"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_TRUST
 description "Trust DSCP markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy ingress_UNTRUST
 description "Do Not Trust DSCP markings from CE"
!
network-slice-services slo-sle-templates qos-catalog L3 input-qos-policy per-flow-test
 description "DSCP based per-flow path selection testing"
!
```

9.3.3 Service Assurance settings

The below performance measurement (PM) profiles (used for SR-PM) and y-1731 profiles (used for L2 P2P) have been pre-created into the NSO PM service package.

```
admin@ncs# show running-config cisco-pm-fp:pm

pm pm-profiles delay-profile sr-policy profile 1wayDelay
 probe computation-interval 60
 probe tx-interval    30000
 probe protocol       twamp-light
 probe measurement-mode one-way
 advertisement periodic interval 60
 advertisement periodic threshold 20
 advertisement periodic minimum-change 1000
!
pm pm-profiles liveness-profile sr-policy profile livenessTest
 probe tx-interval 100000
!
pm svc-profiles 1wayDelaySvc
 performance-measurement delay-profile sr-policy profile 1wayDelay
  !
!
pm svc-profiles livenessSvc
 performance-measurement liveness-profile sr-policy profile livenessTest
  !
!
```

```
admin@ncs# show running-config l2vpn-ntw
y-1731-profile

l2vpn-ntw y-1731-profile Profile-Delay-1
 schedule interval 5
 schedule duration 5
 type delay
 probe measurement-interval 60
 delay-params statistic delay-two-way
 !
 delay-params statistic jitter-two-way
 !
!
l2vpn-ntw y-1731-profile Profile-Loss-1
 type loss
 probe measurement-interval 60
 probe priority       2
 loss-params statistic loss-sd
 !
!
```

9.3.4 Slice Catalog

The following sample slice catalog has been created providing for various scenario slice intents. We will use these in several of our use-cases. These entries are referencing the above pre-requisite settings and are displayed with details included (showing all defaults). Any interesting notations about the intent are included on the right. Also please recall:

- In order to support SR-PM based service assurance, the “as-blueprint” flag must be set as a clone of the referenced ODN template is required due to modification requirements.

- In order to support BWoD customization (where a BW value can be inserted dynamically by the user at slice instance creation), both “as-blueprint” and “customization” flags must be set.
- In order to get a dedicated SR-TE tunnel per-slice instance, both “as-blueprint” and “customization” flags must be set.
- Lastly, note that all of these catalog entries are optional. We do not necessarily need to set QoS, Path forwarding or Service Assurance.

```
admin@ncs#show running-config network-slice-services slo-sle-templates slo-sle-template | details
```

```
network-slice-services slo-sle-templates slo-sle-template eMBB
 template-description "High Bandwidth Service with basic SLA monitoring"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  eMBB
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
```

NOTES:
A very simple BW (igp) based policy using system based heuristic package Service Assurance (not SR-PM).

```
network-slice-services slo-sle-templates slo-sle-template eMBB_PM
 template-description "High Bandwidth Service with Advanced Performance Measurement liveness
Probing"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  eMBB
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Gold_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM system"
 service-assurance heuristics L2 multipoint profile-name "Gold_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP system"
 service-assurance heuristics L3 profile-name "Gold_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile livenessSvc
 service-assurance performance-measurement sr-te liveness-detection profile livenessTest
!
```

NOTES:
Similar to eMBB but in this case we have added SR-PM liveness-based probing. Notice that “as-blueprint” is required. We have also upgraded the “gold-based system Heuristic packages.

```
network-slice-services slo-sle-templates slo-sle-template URLLC
 template-description "Flex-Algo based Low latency Service"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  URLLC
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
```

NOTES:
A low latency-based intent (delay) policy. The referenced URLLC forwarding-plane is using Flex-Algo 128.

```
network-slice-services slo-sle-templates slo-sle-template URLLC_PM
 template-description "Flex-Algo based Low latency Service with Performance Measurement Delay
Probing"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  URLLC
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "25ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 2wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 2wayDelay
!
network-slice-services slo-sle-templates slo-sle-template URLLC_PM_NoFA
 template-description "Non-Flex-Algo based Low latency Service with Performance Measurement
Delay Probing"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  URLLC_NoFA
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "25ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "25ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 2wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 2wayDelay
!
network-slice-services slo-sle-templates slo-sle-template BWOD_1G
 template-description "BW on Demand with 1Gbps BW selection"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  BWOD_1G
 odn forwarding-plane-policy-type as-blueprint
 custom
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template BWOD_500M
 template-description "BW on Demand with 500Mbps BW selection"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  BWOD_500M
 odn forwarding-plane-policy-type as-blueprint
 custom
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
```

NOTES:
Similar to URRLC but we have added SR-PM probing and custom heuristic packages with modified delay thresholds (25ms).

NOTES:
The same URLLC_PM intent but not using Flex-Algo based forwarding.

NOTES:
A BWoD 1Gbps (pre-set) use case. The “customization” flag is set so each slice instance gets a dedicated BWoD reservation. If customization was not set, then all slice instances using this entry would share the BW.

NOTES:
Similar to above but for 500M (pre-set).

```
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template BWOD_custom
 template-description "BW on Demand with Custom BW selection"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  BWOD_custom
 odn forwarding-plane-policy-type as-blueprint
 custom
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template DSCP_Flow
 template-description "For L3 Slices: DSCP Flow Based Path Forwarding Selection DSCP EF -->
URLLC, other-->eMBB"
 qos-policy L3 input-policy per-flow-test
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  DSCP_Flow
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L3 profile-name "Gold_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM system"
!
network-slice-services slo-sle-templates slo-sle-template Delay_10ms
 template-description "Highest BW under 10ms delay"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  eMBB_10ms
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "10ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "10ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "10ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template Delay_7ms
 template-description "Highest BW under 7ms delay"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy  eMBB_7ms
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
```

NOTES:
A custom BWoD use case. Where the user could specify the requested BW when provisioning the slice instance.

NOTES:
This is an advanced use case and not yet supported. The catalog entry would be used for DSCP Flow based slices, which require other pre-requisites be set such as flow-based parameters in the ODN templates.

NOTES:
This is an advanced use case. The catalog entry would be used for “max-delay” based slice types and require other pre-requisites be set for cumulative constraints in the ODN template. Custom heuristic packages are used to set proper delay thresholds.

```
!
network-slice-services slo-sle-templates slo-sle-template Disjoint_North
 template-description "High Bandwidth Disjoint North Service with basic SLA monitoring"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  Disjoint_North
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template Disjoint_South
 template-description "High Bandwidth Disjoint South Service with basic SLA monitoring"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  Disjoint_South
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template ENCRYPT
 template-description "Encrypted path only High Bandwidth Service"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 odn forwarding-plane-policy  ENCRYPT
 odn forwarding-plane-policy-type as-is
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
network-slice-services slo-sle-templates slo-sle-template Gold
 template-description "High Priority Services with SLAs"
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Gold_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM system"
 service-assurance heuristics L2 multipoint profile-name "Gold_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP system"
 service-assurance heuristics L3 profile-name "Gold_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
```

NOTES:
Disjoint path slice for path “north” links. Using flex-algo 131

NOTES:
Disjoint path slice for path “south” links. Using flex-algo 132

NOTES:
Encrypted slice for path encrypted links. Using flex-algo 129.

NOTES:
Simple Slice with high-priority QoS but no SR-TE path forwarding settings (use the igp path). This would be applicable in non-SR networks.

```
network-slice-services slo-sle-templates slo-sle-template Silver
 template-description "Medium Priority Services with limited SLAs"
 qos-policy L2 input-policy ingress_COS1
 qos-policy L2 output-policy Egress-High_Bw_Apps
 qos-policy L3 input-policy ingress_COS1
 qos-policy L3 output-policy Egress-High_Bw_Apps
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics preservation remove
 service-assurance heuristics L2 point-to-point profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM-Basic system"
 service-assurance heuristics L2 multipoint profile-name "Silver_L2VPN_ConfigProfile system"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP-Basic system"
 service-assurance heuristics L3 profile-name "Silver_L3VPN_ConfigProfile system"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM-Basic system"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
!
```

9.3.5 Scenario #3- Implementing a L2 Point-2-Point eMBB PM Slice

User Expectation:
In this scenario, the user requires a L2 P2P eMBB slice.

In this case the system will configure a BGP based VPN using VPWS and use the eMBB_PM intent which requires high BW path forwarding with PM. While this particular template does have SR-PM enabled for liveness, it technically is not required since for L2 P2P slices we can also enable Y.1731 based Service Assurance (which we will also do as part of the Slice template).

```
network-slice-services slice-service c_L2_P2P_ded
 service-description "L2 pt-2-pt dedicated slice- c_L2_P2P_eMBB.cli"
 service-tags tag-type service-tag-customer
  value [ FOO ]
 !
 service-tags tag-type service-tag-service
  value [ L2 ]
 !
 service-tags tag-opaque nssai
  value [ 2-123459876 ]
 !
 slo-sle-template   eMBB_PM
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group3
  !
  attachment-circuits attachment-circuit AC-1
   ac-tp-id TenGigE0/0/0/10
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 422 ]
   !
  !
 !
 sdps sdp 2
  node-id Node-5
  service-match-criteria match-criterion 2
   target-connection-group-id group3
  !
  attachment-circuits attachment-circuit AC-2
   ac-tp-id TenGigE0/0/0/2
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 322 ]
   !
  !
 !
 connection-groups connection-group group3
  connectivity-type point-to-point
 !
!
```

Figure 78 Scenario #3 Slice Instance Payload

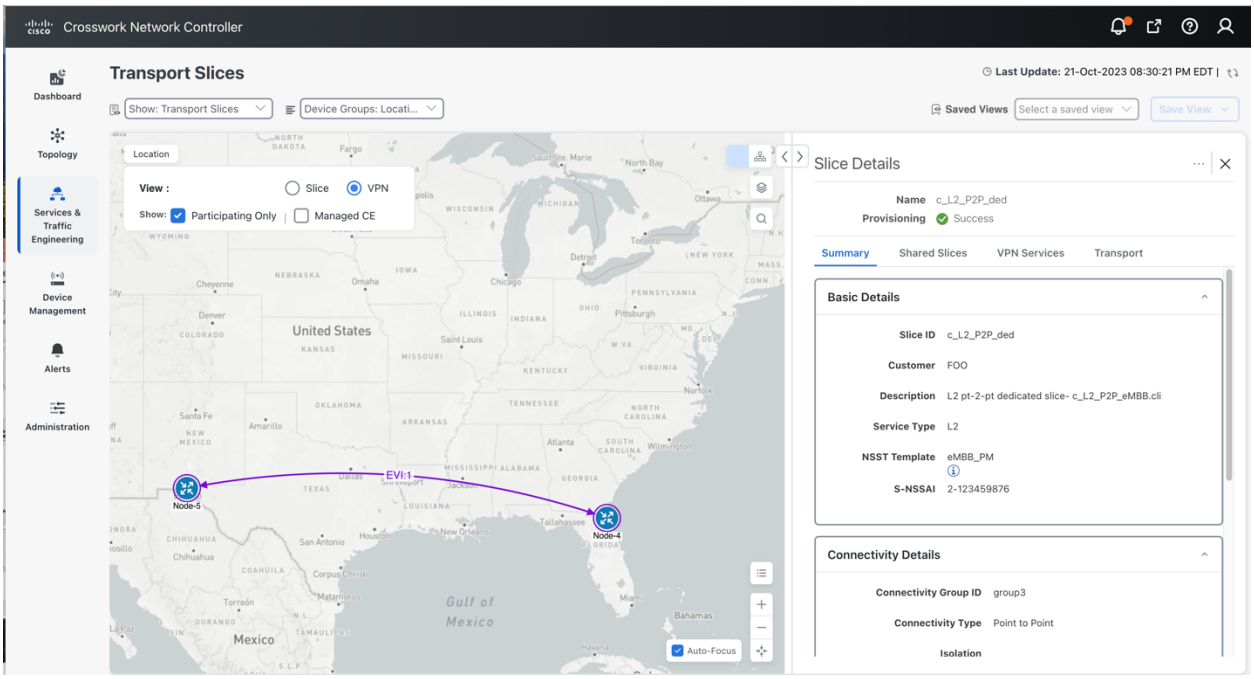

Figure 79 Scenario #3 Slice Summary View

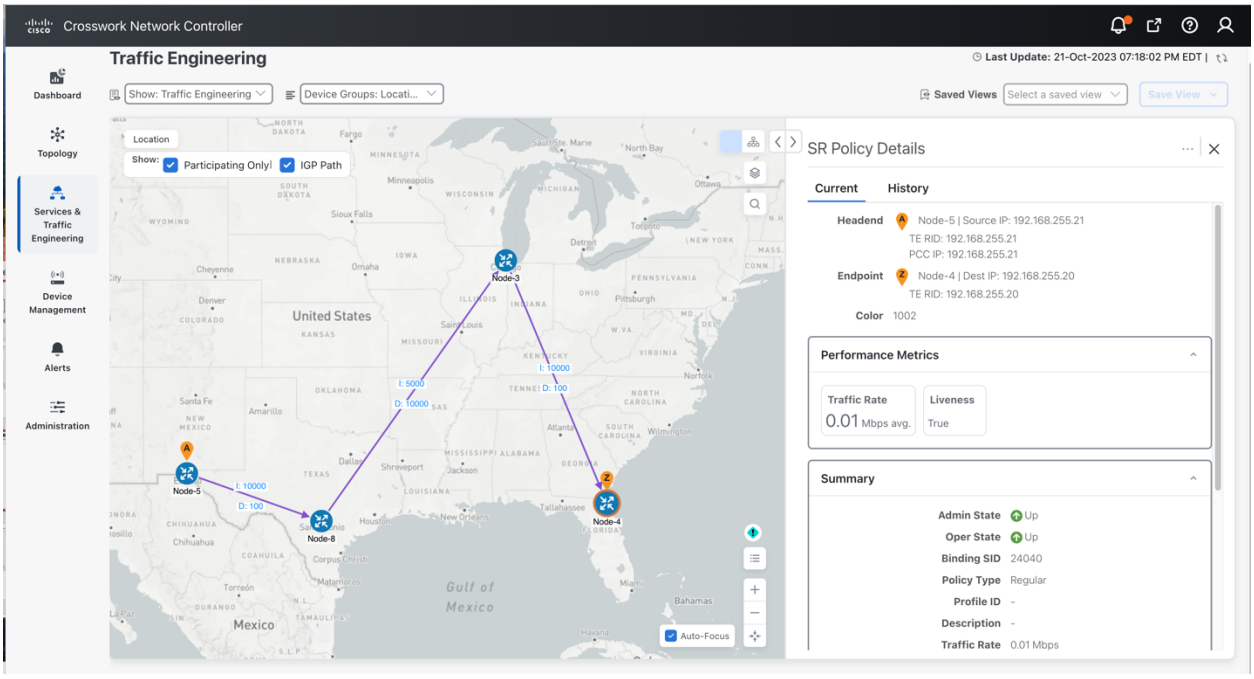

Figure 80 Scenario #3 Slice SR-TE Transport View

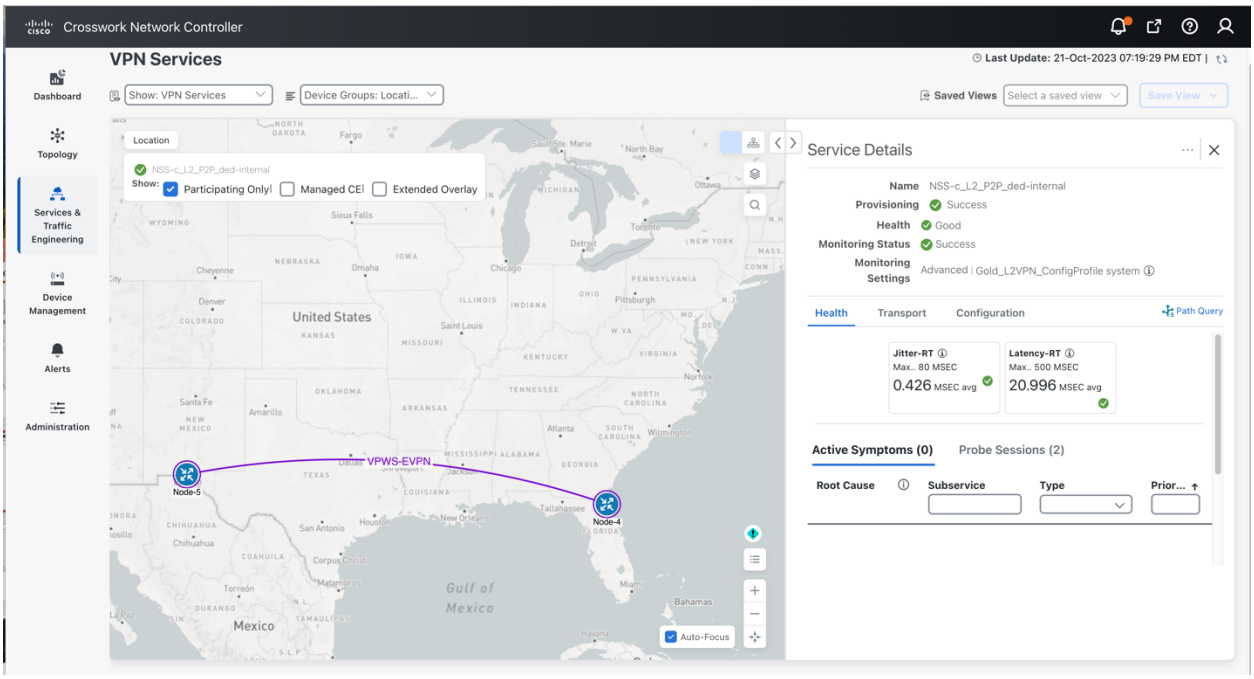

Figure 81 Scenario #3 Slice VPN Health View Showing Y1731 Measurements

9.3.6 Scenario #4- Implementing a L2 Hub and Spoke URLLC Slice

User Expectation:
In this scenario, the user requires a L2 hub-and-spoke URLLC slice with the intent to minimize latency.

This example shows the slice connectivity model for a hub & spoke scenario where spokes cannot communicate with each other but only with the hub. This is accomplished by the spoke sites importing only the hub RT and not importing the spoke RT, while the hub imports both. In this case the system will configure a BGP based VPN using eVPN and use the URLLC intent which requires low latency path forwarding.

```
network-slice-services slice-service i_L2_HubSpoke_ded
 service-description "L2 hub/spoke dedicated slice- i_L2_HubSpoke_dedicated_URLLC.cli"
 service-tags tag-type service-tag-service
  value [ L2 ]
 !
 service-tags tag-opaque nssai
 value [ 2-123459876 ]
 !
 service-tags tag-type service-tag-customer
  value [ Foo ]
 !
!
!
 connection-groups connection-group group1
   connectivity-type hub-spoke
 !
!
 slo-sle-template URLLC
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
```

```
   connection-group-sdp-role  spoke-role
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 411 ]
   !
  !
 !
 sdps sdp 2
  node-id Node-5
  service-match-criteria match-criterion 1
   target-connection-group-id group1
   connection-group-sdp-role  hub-role
  !
  attachment-circuits attachment-circuit ac2
   ac-tp-id            TenGigE0/0/0/2
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 311 ]
   !
  !
 !
 sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group1
   connection-group-sdp-role  spoke-role
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 611 ]
   !
  !
 !
!
```

Figure 82 Scenario #4 Slice Instance Payload

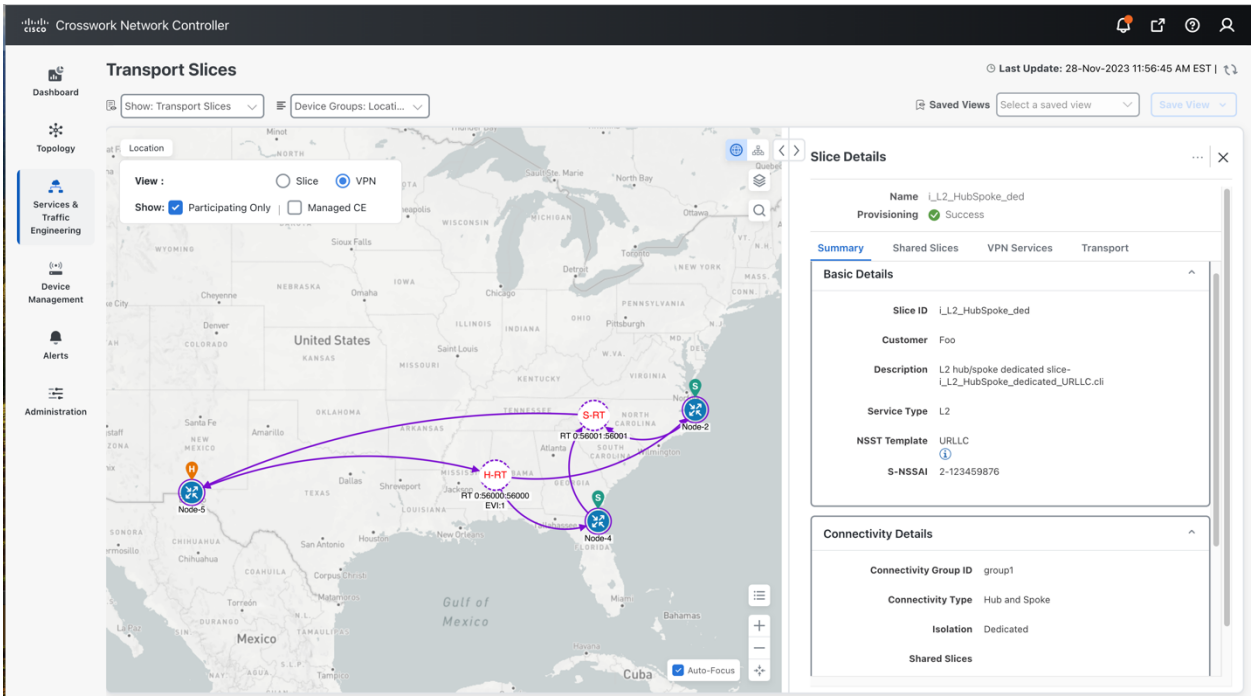

Figure 83 Scenario #4 Slice Summary View

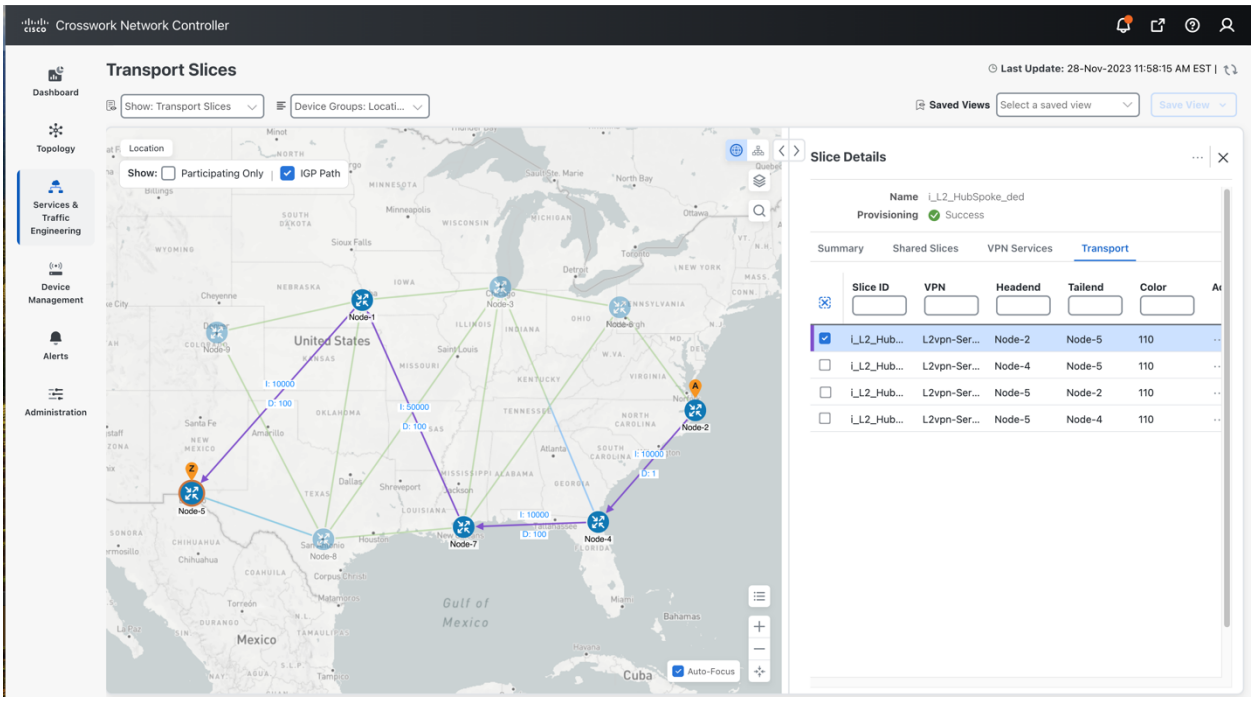

Figure 84 Scenario #4 Slice SR-TE Transport View showing low latency path

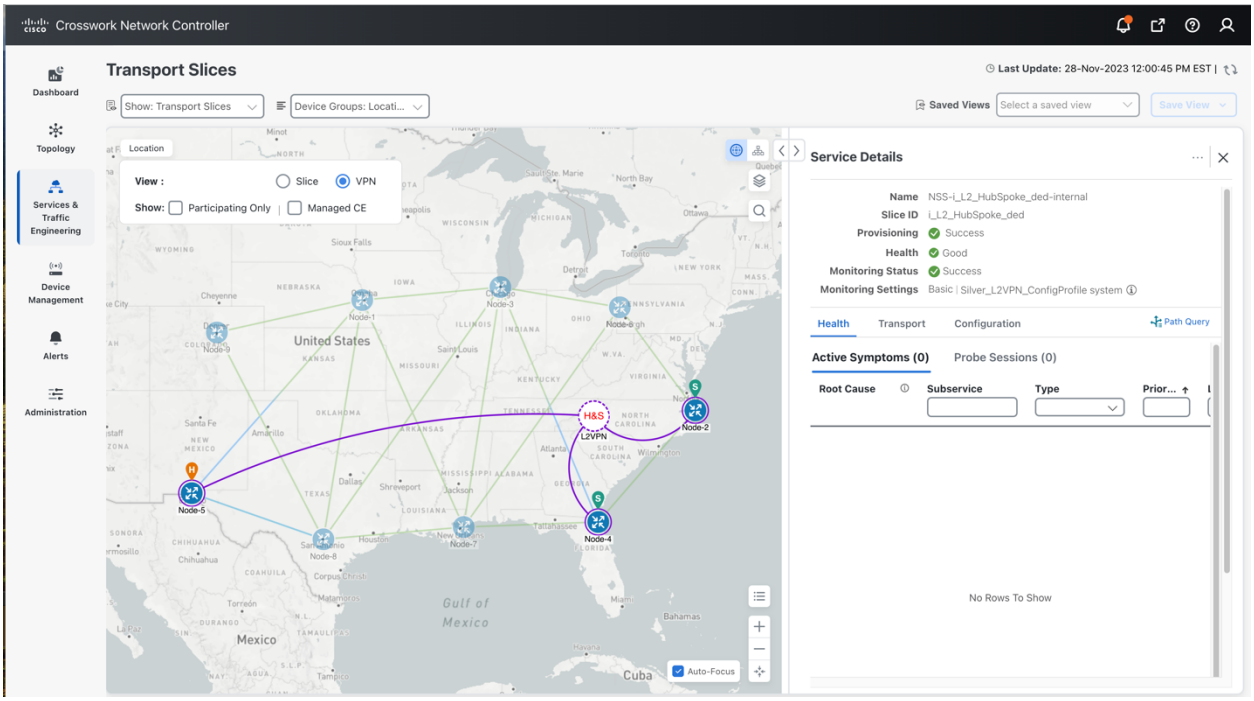

Figure 85 Scenario #4 Slice VPN Health View

9.3.7 Scenario #5- Implementing Shared Slices

User Expectation:
In this scenario, the user requires three L3 Slice Instances, where two are dedicated slice instances and one is a shared slice instance. The two dedicated slices are required to communicate to the shared slice endpoints.

The dedicated slice endpoints can communicate with the shared slice endpoint and (if configured) other SDP endpoints within its own dedicated slice service instance. But the two dedicated slices cannot communicate between each other. Additionally, one dedicated slice is using the URLLC (low latency) intent, while the other slice instance is using the eMBB (high BW) intent. While the shared slice is also setup for eMBB. In this scenario, we desire the communication between the URLLC slice instance and the shared slice instance to use the URLLC behavior for all traffic (both being sent to and received from the shared slice). Thus, we will enable Single-Sided Control on the dedicated slice instance.

```
network-slice-services slice-service f_L3_A2A_ded-1
 service-description "L3 A2A dedicated slice1 connected to shared slice- f_L3_A2A_multi-dedicated_shared.cli"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
 service-tags tag-opaque nssai
 value [ 2-123459876 ]
 !
 service-tags tag-type service-tag-customer
  value [ Ford ]
 !
!
 slo-sle-template URLLC
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 connection-groups connection-group group1
   connectivity-type any-to-any
 !
 isolation        service-isolation-dedicated
 !
 shared slice f_L3_A2A_shared
 !
 shared single-sided-control true
!
network-slice-services slice-service f_L3_A2A_ded-2
 service-description "L3 A2A dedicated slice2 connected to shared slice- f_L3_A2A_multi-dedicated_shared.cli"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
 service-tags tag-opaque nssai
 value [ 1-789 ]
 !
 service-tags tag-type service-tag-customer
  value [ Chevy ]
 !
!
 slo-sle-template eMBB
 sdps sdp 3
```

```
   node-id Node-2
   service-match-criteria match-criterion 1
    target-connection-group-id group1
   !
   attachment-circuits attachment-circuit ac3
    ac-tp-id            TenGigE0/0/0/2
    ac-ip-address       172.16.3.1
    ac-ip-prefix-length 29
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 601 ]
    !
    sdp-peering protocol peering-protocol-bgp
     bgp-attributes neighbor [ 172.16.3.2 ]
     bgp-attributes remote-as 65103
    !
   !
  !
  connection-groups connection-group group1
    connectivity-type any-to-any
  !
  isolation        service-isolation-dedicated
  !
  shared slice f_L3_A2A_shared
  !
 !
 network-slice-services slice-service f_L3_A2A_shared
  service-description "L3 A2A shared slice- f_L3_A2A_multi-dedicated_shared.cli"
  service-tags tag-type service-tag-service
   value [ L3 ]
  !
  service-tags tag-opaque nssai
  value [ 1-12345 1-789 2-123459876 ]
  !
  service-tags tag-type service-tag-customer
   value [ NAPA ]
  !
 !
  slo-sle-template eMBB
   sdps sdp 1
   node-id Node-5
   service-match-criteria match-criterion 1
    target-connection-group-id group1
   !
   attachment-circuits attachment-circuit ac2
    ac-tp-id            TenGigE0/0/0/2
    ac-ip-address       172.16.1.1
    ac-ip-prefix-length 29
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 301 ]
    !
    sdp-peering protocol peering-protocol-bgp
     bgp-attributes neighbor [ 172.16.1.2 ]
     bgp-attributes remote-as 65101
    !
   !
  !
  connection-groups connection-group group1
    connectivity-type any-to-any
  !
  isolation        service-isolation-shared
  !
 !
```

Figure 86 Scenario #5 Slice Instance Payload for all Slices

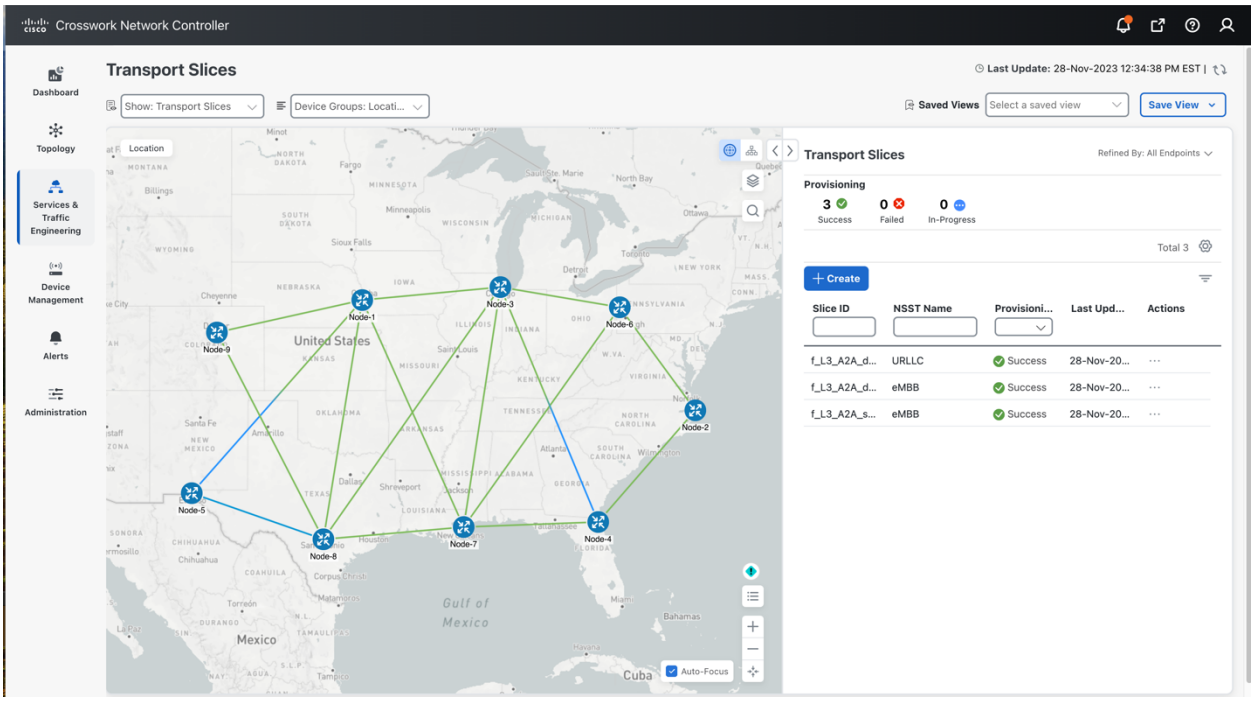

Figure 87 Scenario #5 Transport Slices View

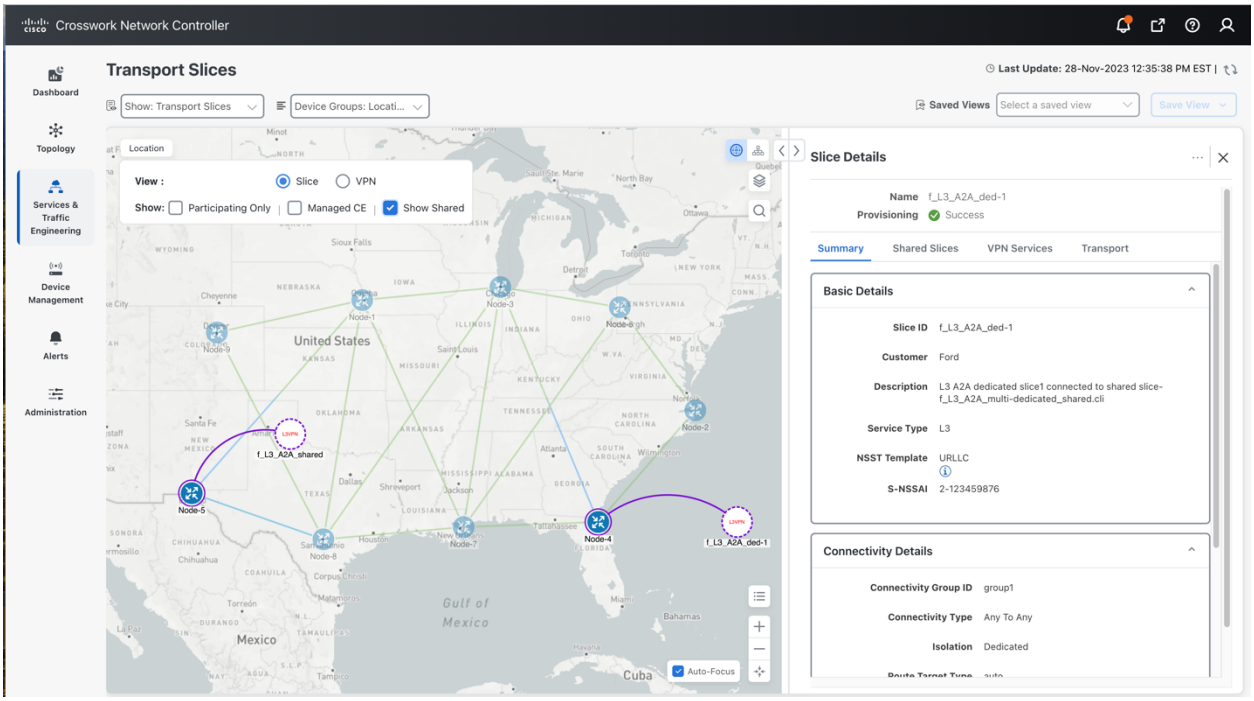

Figure 88 Scenario #5 Dedicated Slice #1 Summary with Shared Slice view

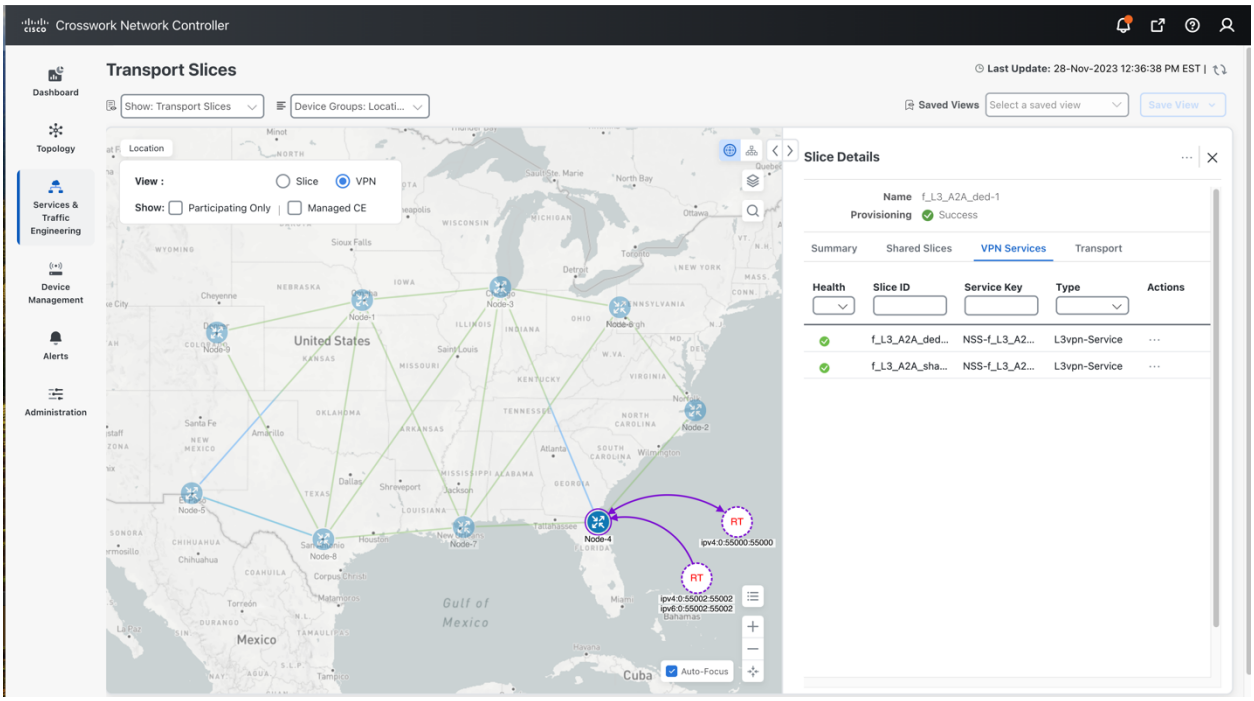

Figure 89 Scenario #5 Dedicated Slice #1 VPN Service View

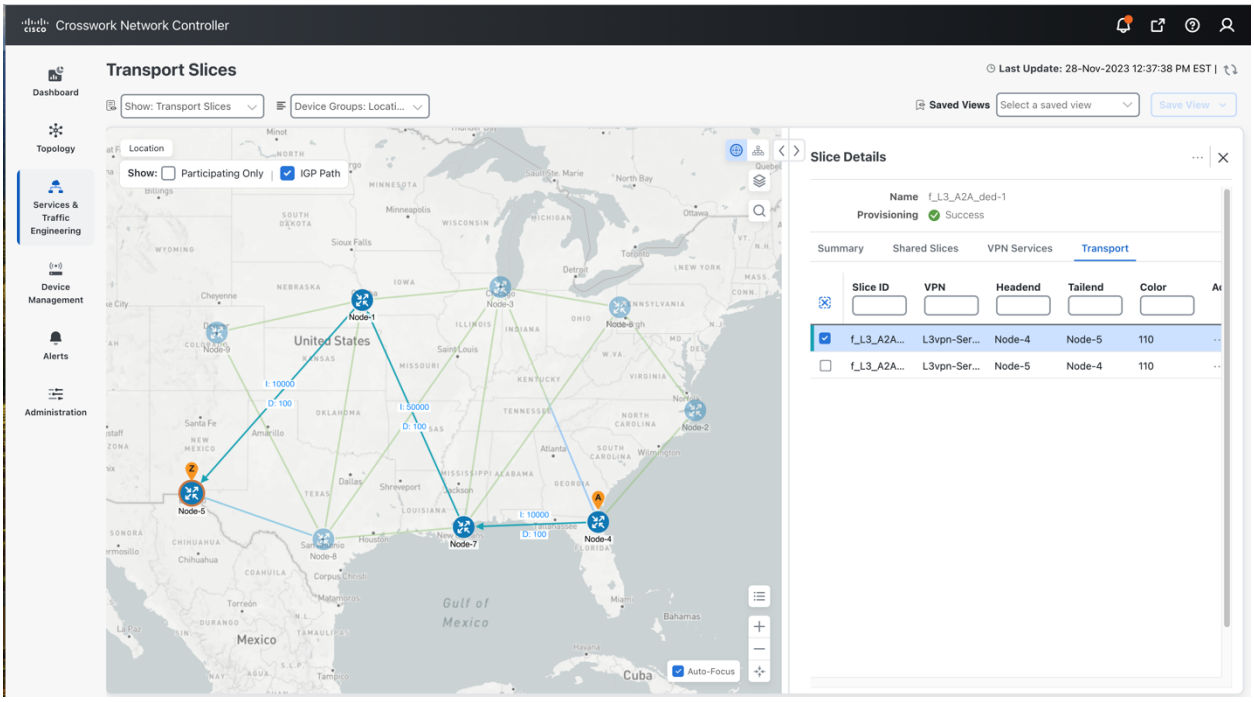

Figure 90 Scenario #5 Dedicated Slice #1 SR-TE Transport View showing Low Latency Path between Node-4 and Node-5 for Dedicated Slice #1

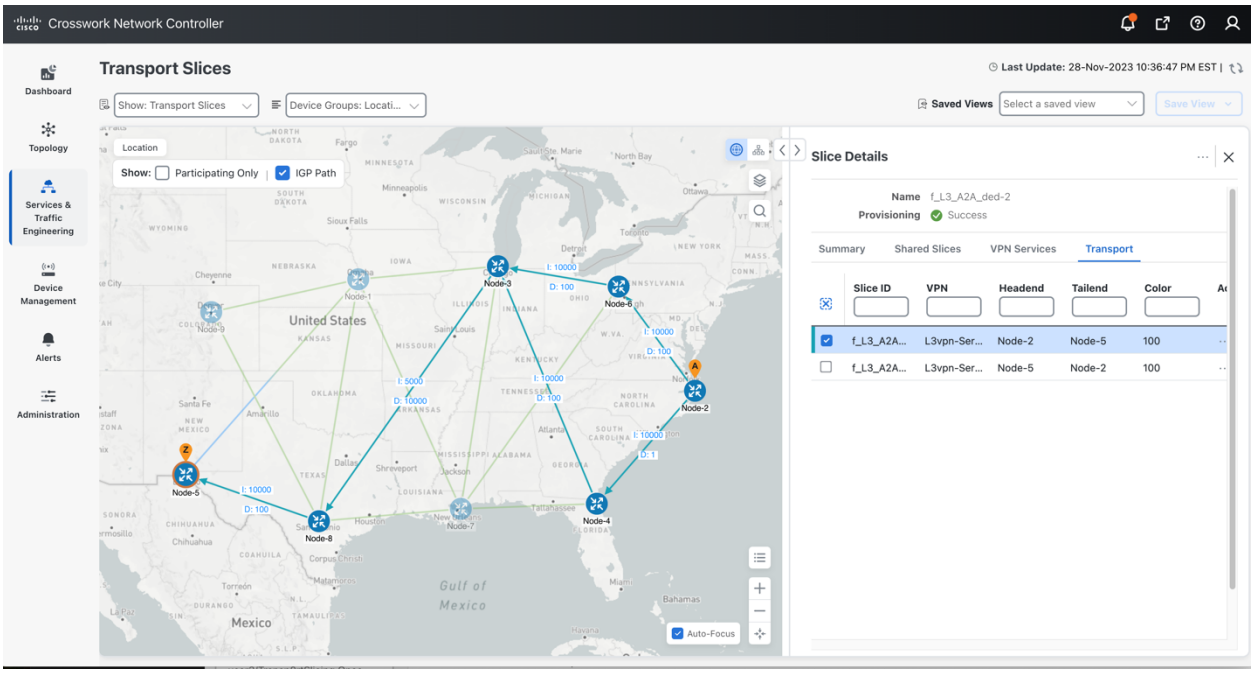

Figure 91 Scenario #5 Dedicated Slice #2 SR-TE Transport View showing High BW Path between Node-2 and Node-5 for Dedicated Slice #2

9.3.8 Scenario #6- Implementing BWoD Slice Instances

User Expectation:
In this scenario, the user requires a point-to-point L2 slice instance with a Bandwidth on Demand (BWoD) reservation of 900Gbps.

In the below slice instance payload, the "BWOD_custom" template has been selected. This template is focused on the IGP BW metric, but also is set up with "as-blueprint" and with "customization" enabled. This allows the user to specify a BW value in the slice instance payload which will then be configured into a new ODN template that will be dedicated for only this slice instance. When a BW value is configured inside an ODN template, the PCE will request BW reservation services from the Optimization Engine if BWoD is enabled in the CNC dashboard (see Figure 27 and Figure 13).

```
network-slice-services slice-service n_L2_P2P_ded_BWoD
 service-description "dedicated slice for L2 P2P slice with customized BwoD setting- n_L2_P2P_dedicated_BWOD_custom.cli"
 service-tags tag-type service-tag-service
  value [ L2 ]
 !
 service-tags tag-opaque nssai
  value [ 1-123459876 ]
 !
 service-tags tag-type service-tag-customer
  value [ FOO ]
 !
 slo-sle-template BWOD_custom
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group3
  !
```

```
   attachment-circuits attachment-circuit AC-1
    ac-tp-id TenGigE0/0/0/10
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 422 ]
    !
   !
  !
  sdps sdp 2
   node-id Node-5
   service-match-criteria match-criterion 2
    target-connection-group-id group3
   !
   attachment-circuits attachment-circuit AC-2
    ac-tp-id TenGigE0/0/0/2
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 322 ]
    !
   !
  !
  connection-groups connection-group group3
   connectivity-type point-to-point
   service-slo-sle-policy bandwidth 900000
  !
 !
```

Figure 92 Scenario #6 Slice Instance Payload

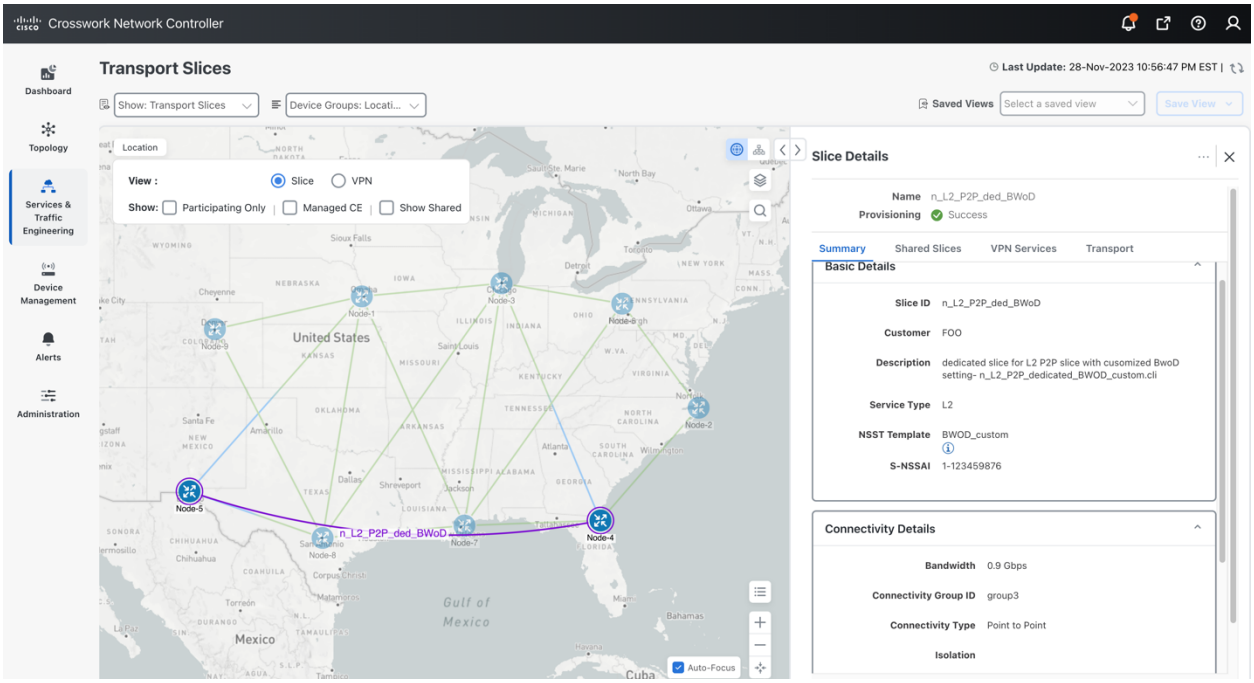

Figure 93 Scenario #6 BWoD Slice Summary view

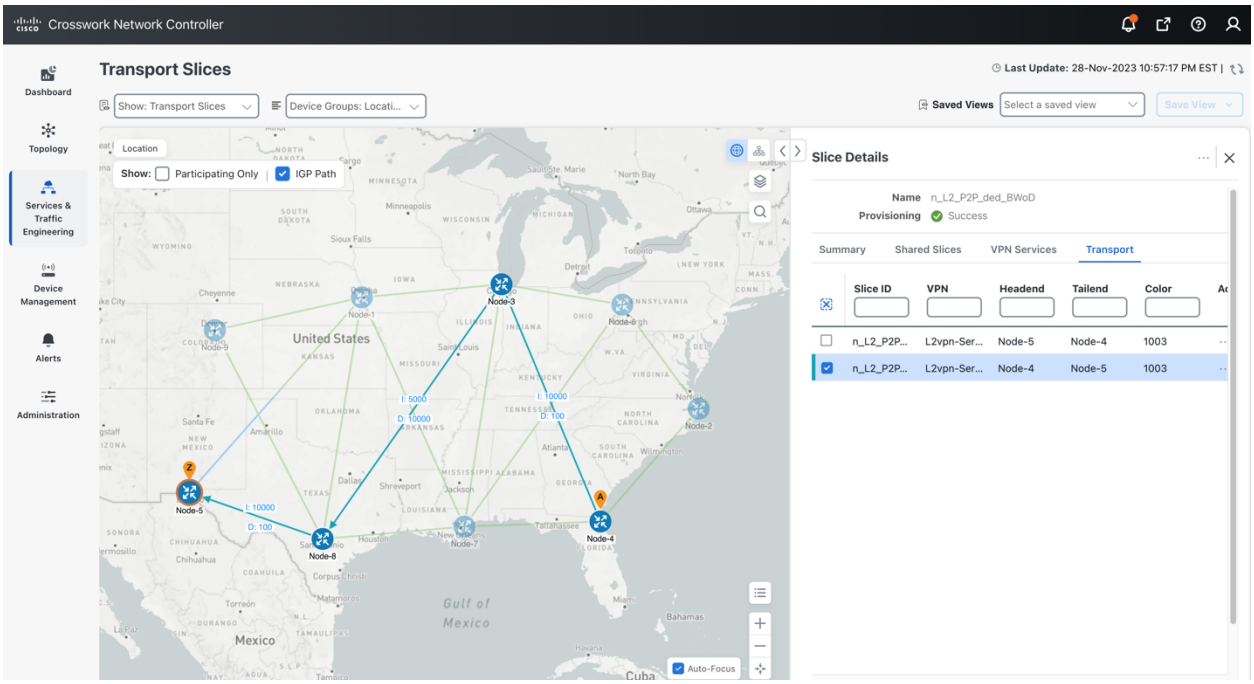

Figure 94 Scenario #6 Slice SR-TE Transport View showing IGP path.

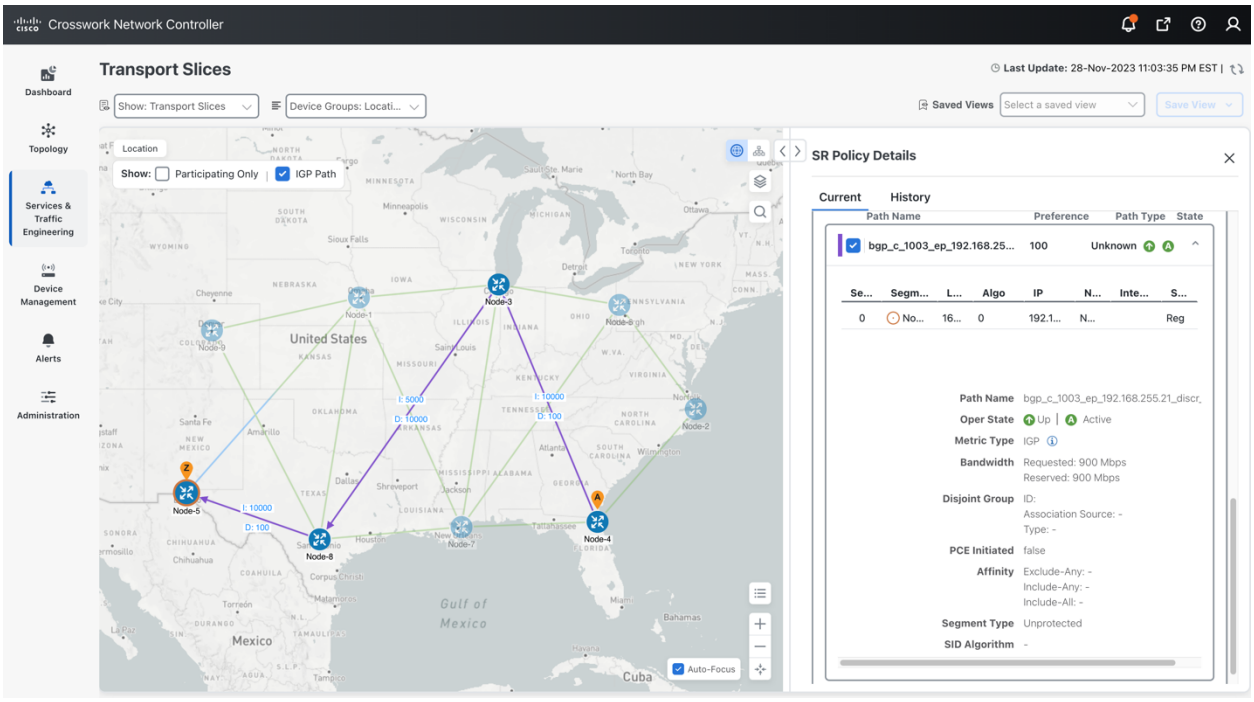

Figure 95 Scenario #6 Slice SR Policy Details showing BW requested and reservation.

9.3.9 Scenario #7- Implementing Encrypted Path Slice Instances

User Expectation:
In this scenario, the user requires a L3 any-to-any slice instance which only traverses encrypted links for added security.

In the below slice instance payload, the “ENCRYPT” template has been selected. This template is focused on using Flex-Algo 129, which is an affinity based forwarding plane. The Slice Designer has colored specific links that have MacSec enabled and then has mapped those links to Flex-Algo 129, thus creating a new topology with a forwarding plane that meets the intent. (see Figure 54).

```
network-slice-services slice-service o_L3_A2A_ded
 service-description "L3 A2A slice use Encrypted Links- o_L3_A2A_dedicated_ENCRYPT.cli"
 service-tags tag-type service-tag-customer
  value [ ACME ]
 !
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
 service-tags tag-opaque nssai
  value [ 123459876 ]
 !
 slo-sle-template ENCRYPT
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 sdps sdp 2
  node-id Node-5
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac2
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.1.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 301 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.1.2 ]
    bgp-attributes remote-as 65101
   !
  !
 !
 sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.3.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 601 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.3.2 ]
    bgp-attributes remote-as 65103
   !
  !
 !
 connection-groups connection-group group1
  connectivity-type any-to-any
 !
```

Figure 96 Scenario #7 Slice Instance Payload

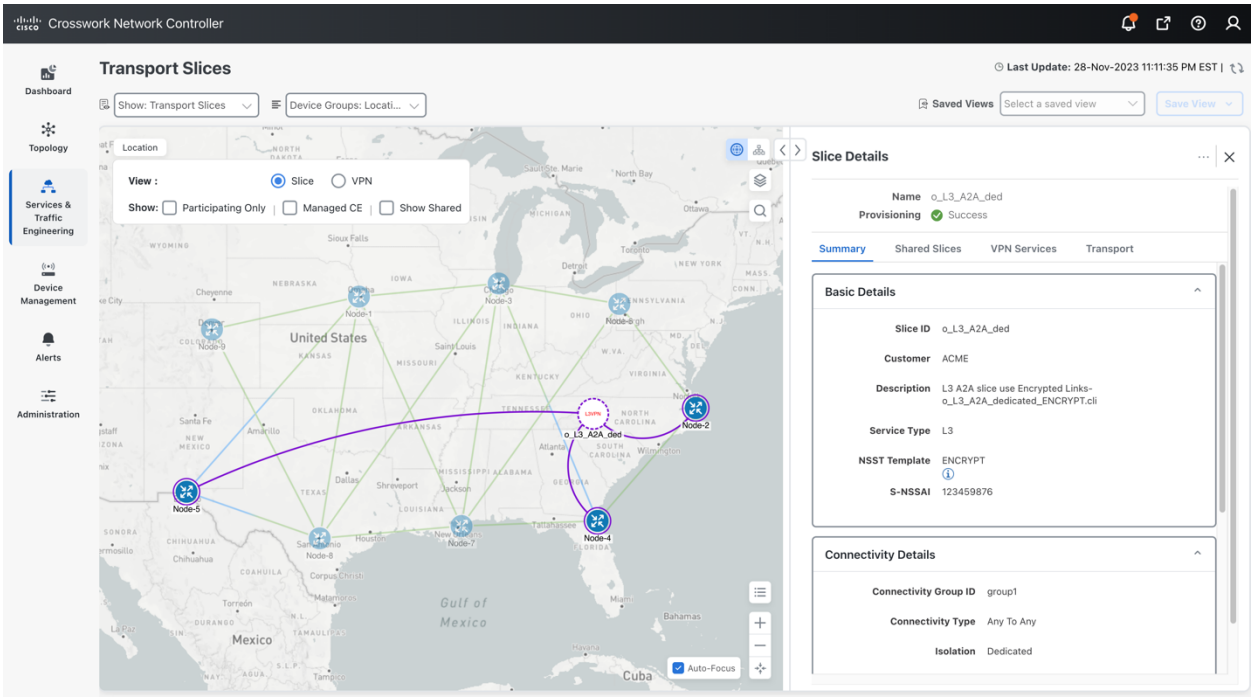

Figure 97 Scenario #7 Slice Summary view

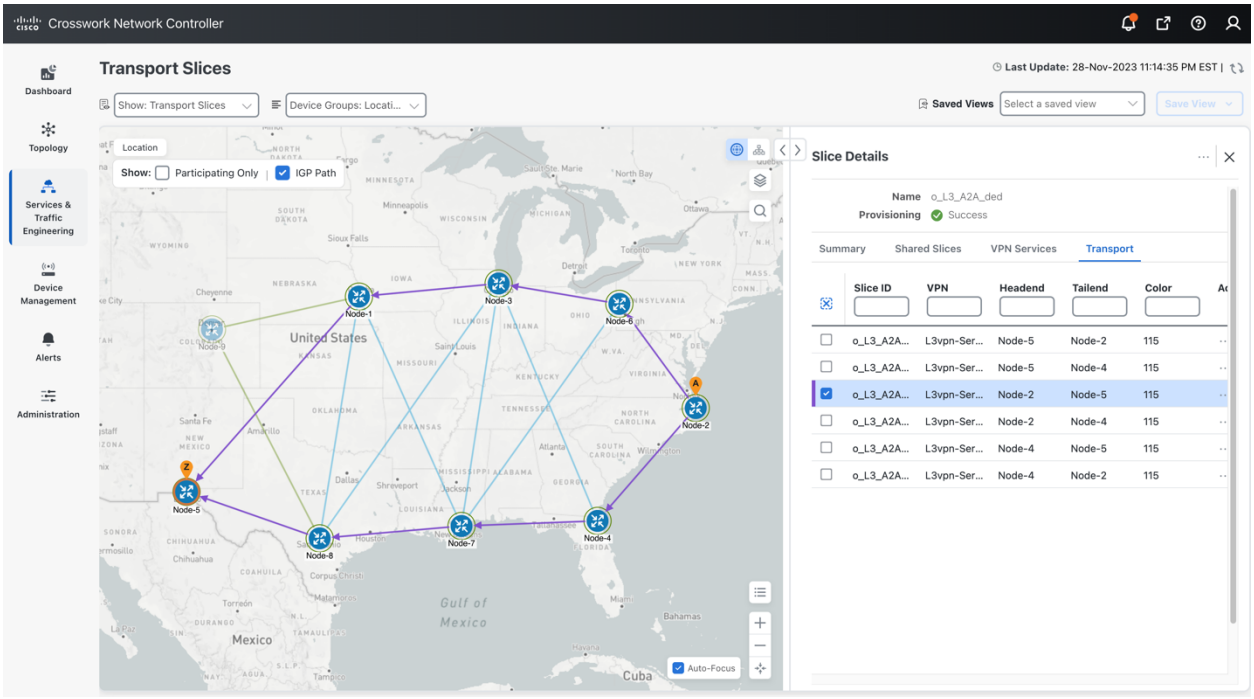

Figure 98 Scenario #7 Slice SR-TE Transport View showing path Node-2 to Node-5

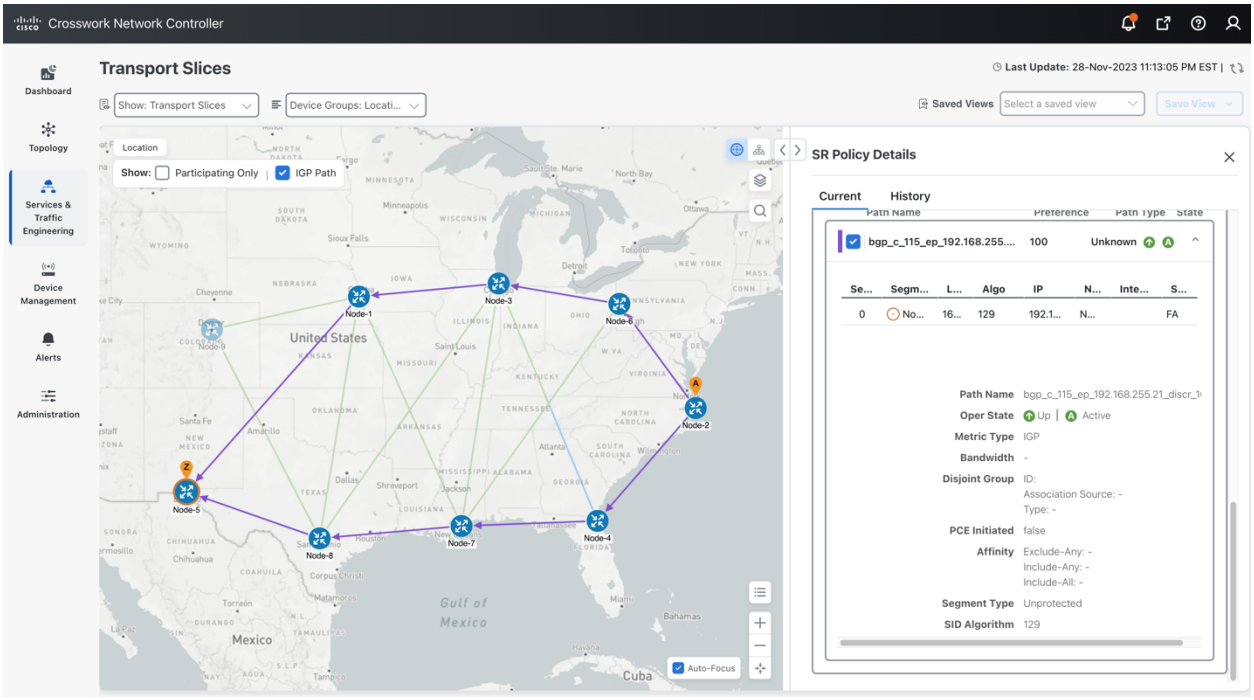

Figure 99 Scenario #7 Slice SR Policy Details showing Flex-Algo 129

9.3.10 Scenario #8- Implementing Cumulative Constraints Slice Instances- 7ms Max Delay

User Expectation:

In this scenario, the user requires a L3 any-to-any slice instance with an end-to-end maximum delay of less than 7ms, with service assurance.

This is an advanced slicing scenario as described in section 10.1.1. In this scenario, the lowest latency path is not necessarily required, but we do have an upper bound maximum delay value intent. The best IGP path available that meets this delay constraint will be selected. Service assurance via SR-PM path probing will also be enabled along with Service Health with custom heuristic packages that have a delay threshold of 7ms configured. This is all setup in the Delay_7m Slice Template. Lastly, as discussed in 10.1.1, for this scenario, the ODN templates must be configured with “custom-template” functionality in order to support the appropriate cumulative bound commands. Additionally, the path is computed via the head-end PCC, as the PCE does not support the cumulative constraint functionality at this time. It may aid the reader to review Figure 52 to understand the overall topology and all per-link delay values.

```
devices template ct-color-131-7ms
 ned-id cisco-iosxr-cli-7.52
  config
   segment-routing traffic-eng on-demand color 131
    dynamic bounds cumulative type latency
     value 7000
    !
   !
  !
 !
cisco-sr-te-cfp:sr-te odn odn-template eMBB_7ms
 custom-template ct-color-131-7ms
 color 131
 dynamic metric-type igp
!
```

Figure 100 Required ODN template prerequisites for scenario #8

```
network-slice-services slice-service s_L3_A2A_ded_Del
 service-description "L3 A2A with max delay 07ms- s_L3_A2A_dedicated_Delay_07ms.cli"
 service-tags tag-type service-tag-customer
  value [ ACME ]
 !
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
 service-tags tag-opaque nssai
  value [ 1-123459876 ]
 !
 slo-sle-template Delay_7ms
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 sdps sdp 2
  node-id Node-5
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac2
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.1.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 301 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.1.2 ]
    bgp-attributes remote-as 65101
   !
  !
 !
 sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group1
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.3.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 601 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.3.2 ]
```

```
    bgp-attributes remote-as 65103
   !
  !
 !
 connection-groups connection-group group1
  connectivity-type any-to-any
 !
!
```

Figure 101 Scenario #8 Slice Instance Payload

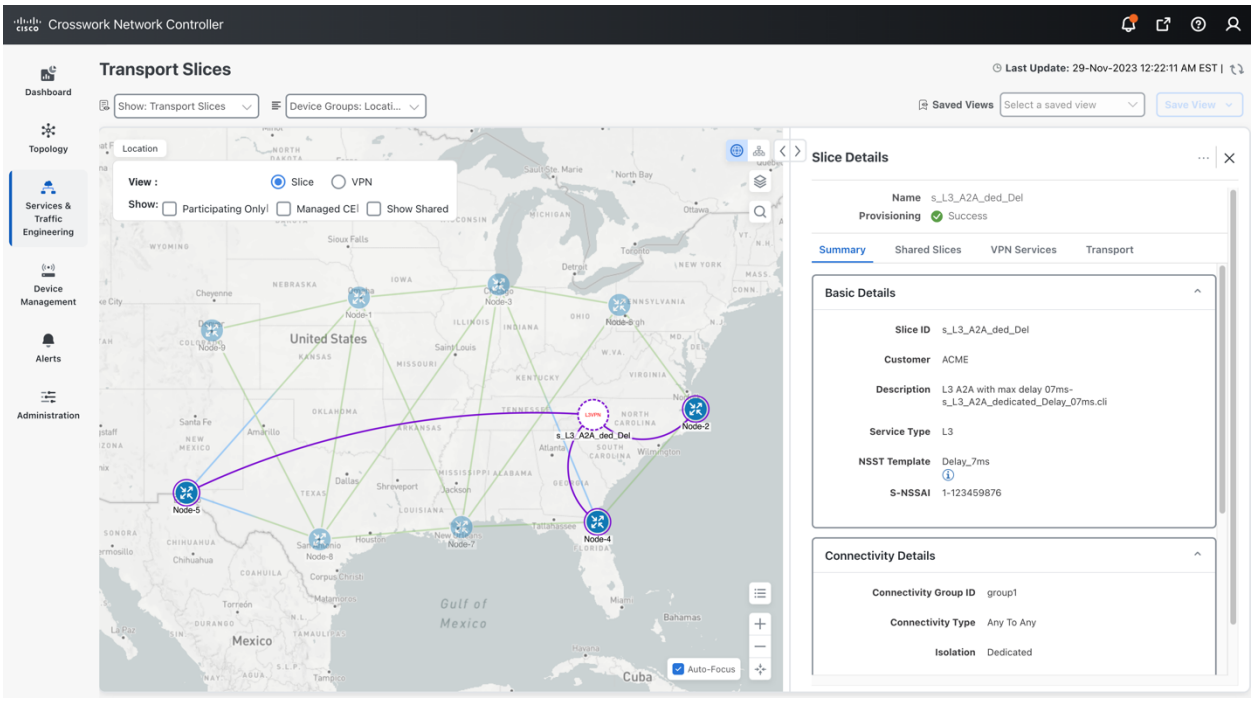

Figure 102 Scenario #8 Slice Summary view

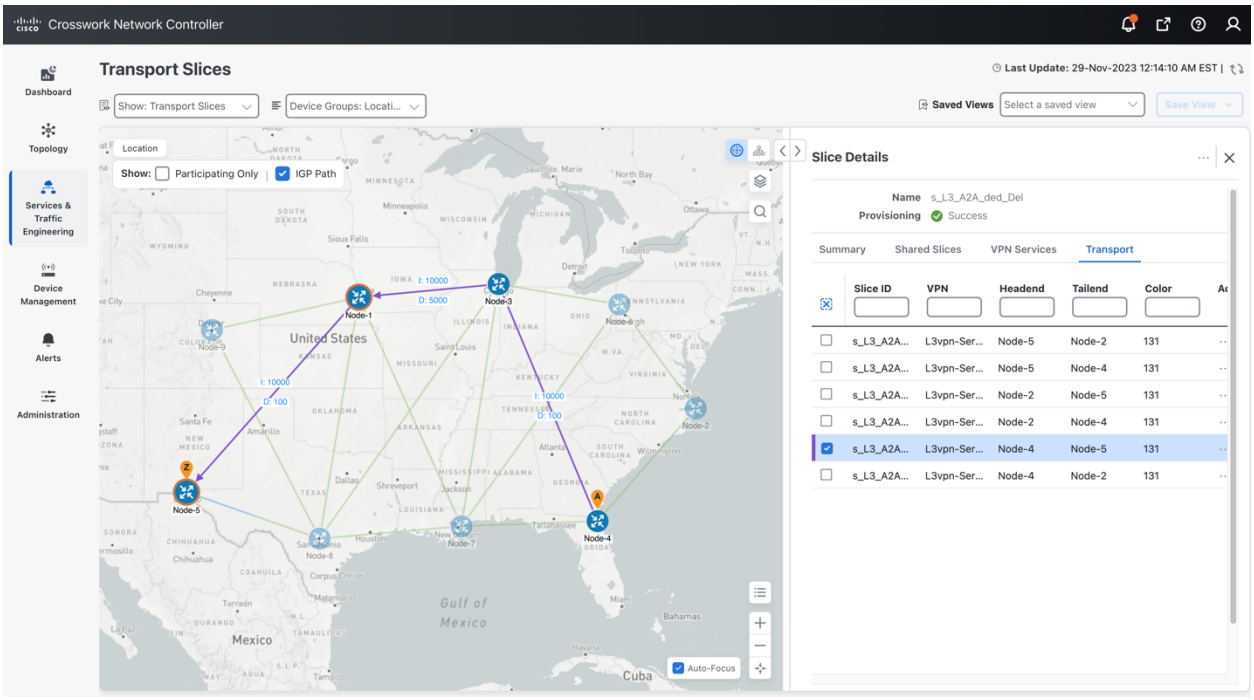

Figure 103 Scenario #8 Slice SR-TE Transport View showing path Node-4 to Node-5 with Delay < 7ms

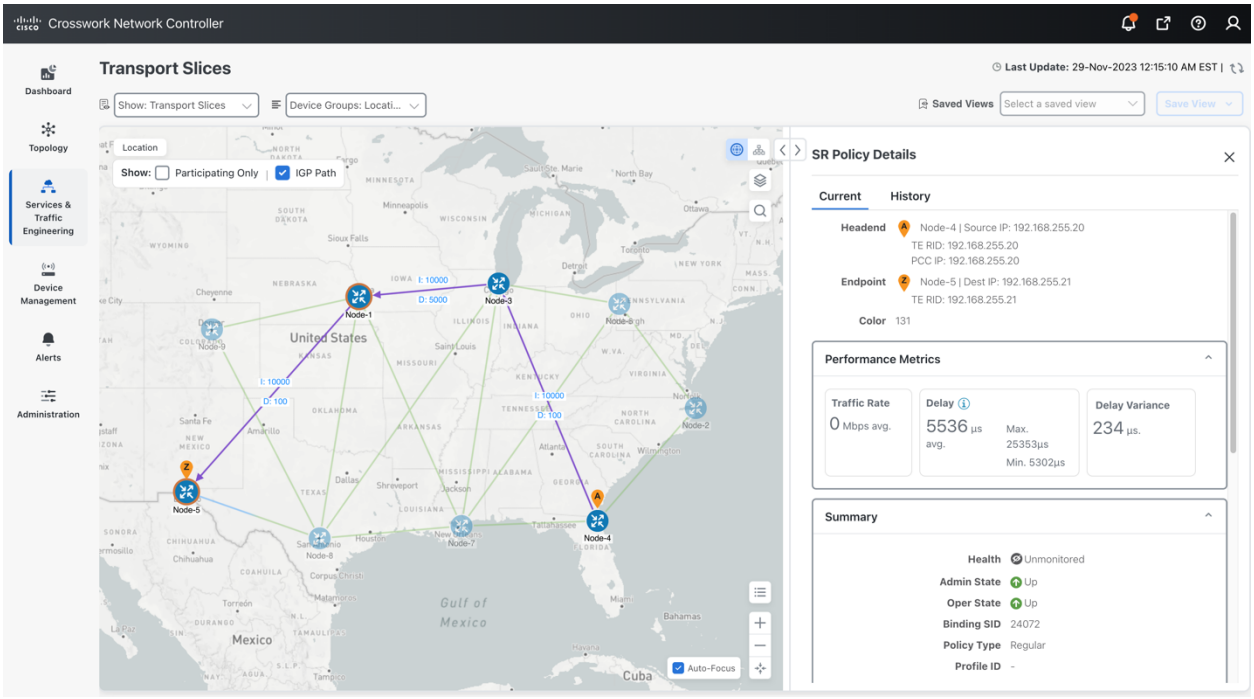

Figure 104 Scenario #8 Slice SR Policy Details with Performance Metrics

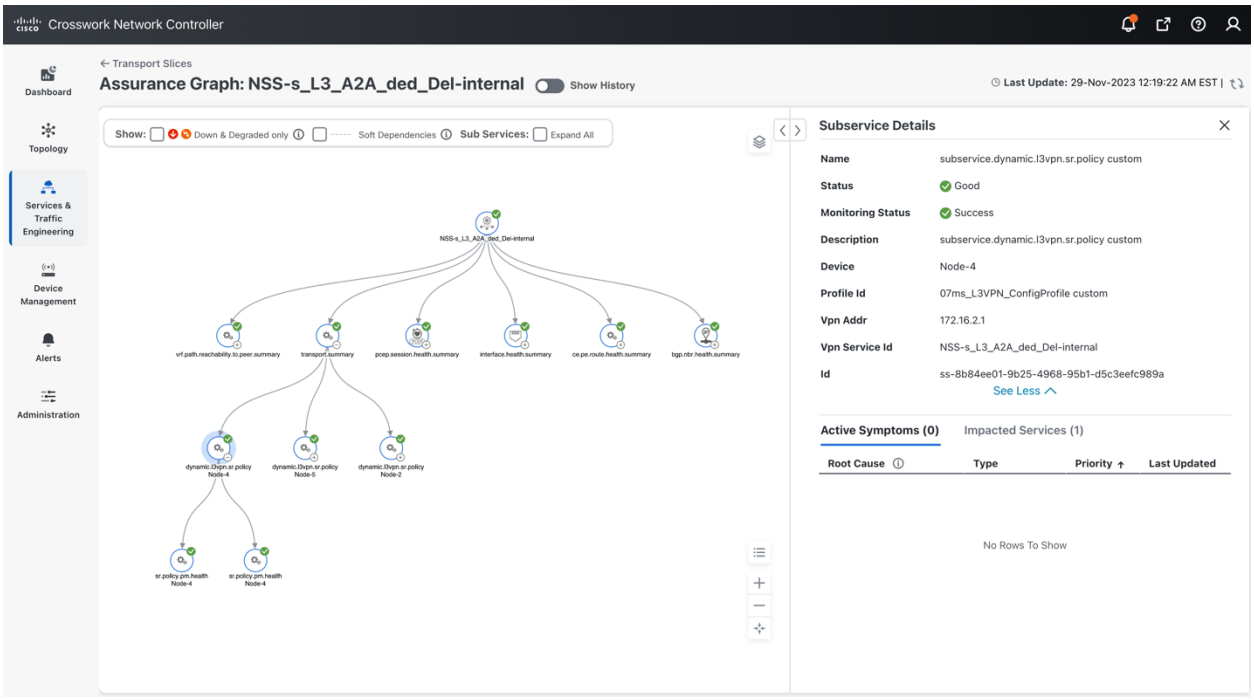

Figure 105 Scenario #8 Slice Service Assurance Graph with Custom Heuristic Package

9.3.11 Scenario #9- Re-Work Non-Slicing Scenario with Slicing

In the CNC Solution Workflow Guide, there is a scenario titled "Implement and Maintain SLA for an L3VPN Service for SR-MPLS (using ODN)". It would be an interesting experiment to redo this scenario with the new CNC slicing solution as it is a fairly complex use-case that we can simplify. This can help set a foundation on how slicing services could be used to abstract complexity.

User Expectation:
In this example, the lowest latency path is the SLA objective as the customer requires a low latency path for high priority traffic. The customer also wants to use disjoint paths, i.e., two unique paths that steer traffic from the same source (Node-5) but to two unique destinations (Node-4 and Node-2), avoiding common links so that there is no single point of failure. What was not well defined in the original scenario was the user's connectivity requirements, so we will assume a hub-spoke connectivity model where Node-5 is the hub and Node-4 and Node-2 are spokes that do not need connectivity to each other.

Slicing Scenario Value-Added Additions:
While the original use-case is interesting, let's add QoS policies for high priority traffic treatment into the scenario. Also, let's use a Flex-Algo based policy based on latency to reduce complexity. Let's also add a PE-CE eBGP peering session. Lastly, let's add CNC Service Health with SR-TE Performance Measurement based Service Assurance with a SLO of 7ms maximum latency alarming threshold.

We will use the same topology diagram and metrics from the previous examples, shown below:

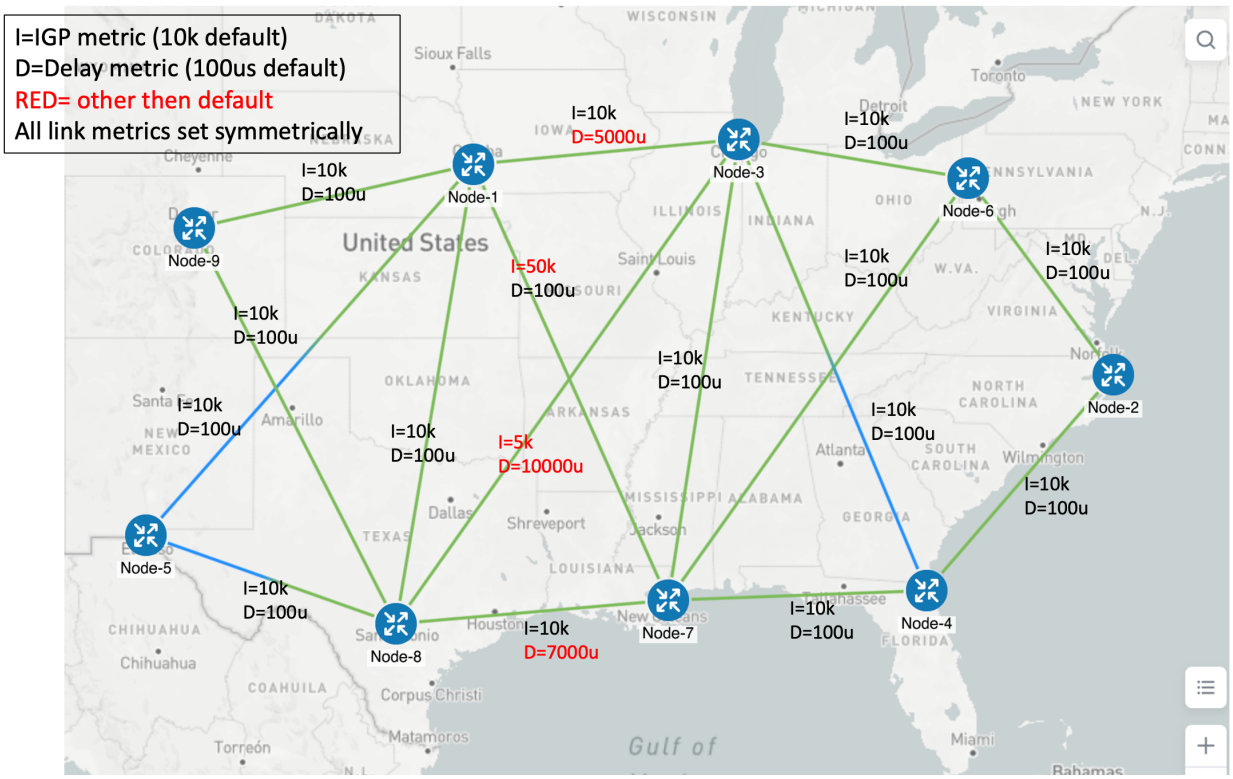

Figure 106- Scenario Topology

Design Consideration:

The Transport Slicing Solution's goal is to simplify service provisioning into intent-based outcomes whenever possible by making educated assumptions. When these assumptions are wrong, and more curated and custom configurations are required, the user can always revert to a non-slicing approach as outlined in the original example. It should be highlighted that the Transport Slicing Solution assumes uniformed forwarding policies across all endpoints in a Slice Service Instance and in this scenario the requirements are for a different policy on Node-5 (which requires both low latency and disjoint paths to Node-4 and Node-2), while the policies from Node-4 and Node-2 to Node-5 only have a low latency requirement. There are two approaches to solve this issue with the slicing solution and we will work through each of them.

- **Solution #1:** Use a dedicated slice for Node-4 and Node-2 endpoints and connect them to a shared slice at Node-5. Build a forwarding policy requesting disjoint services and with Single-Sided Control being disabled, this will provide asymmetric forwarding policies between dedicated slices and shared slices.
- **Solution #2:** Use two dedicated slices (one for the endpoint on Node-4 and one for the endpoint on Node-2) and connect both to a shared slice at Node-5. For each of the dedicated slices, use separate disjoint Flex-Algo based policies using link affinities (if you recall we have a "Disjoint-North" and a "Disjoint-South" Forwarding plane already configured). This approach not only keeps traffic from Node-5 disjoint, but also the return traffic.

9.3.11.1 Scenario #9 Solution #1 Walk-Through

Step 1 Create headless ODN templates to map SLA objective and constraints to intents
Step 2 Create the Slice Catalog Entry
Step 3 Create and provision the Shared and Dedicated Slicing Service
Step 4 Visualize the New VPN Service on the Map to See the Traffic Path

9.3.11.1.1 Step 1 Create ODN templates and PM profiles

Dis-jointness constraints work by associating a disjoint group ID with the ODN template, and all tunnels with the same disjoint group ID will be disjoint, i.e., they will use different links, nodes and shared risk link groups depending on how the disjoint groups are configured.

We will create the following ODN templates:

color 72, latency (flex-algo 128), disjoint path (link), group ID 16 - Delay_Disjoint_72
color 70, latency (flex-algo 128) - Delay_70

Before you begin
In this step, we will create a single ODN template (color 72) similar to the original example, but without the head-ends specified. The ODN template specifies the color and the intent; in this case, latency and dis-jointness. This ODN template will be used to dynamically create tunnels (on-demand) when prefixes with matching colors are received via BGP. Traffic to these prefixes will be automatically steered into the newly created tunnels, thereby meeting the SLA objective and constraints intended for these prefixes and signaled using colors in the BGP routes.

The same procedure is followed from the original scenario to create the ODN templates. Below is the CLI output from NSO with no head-ends defined:

```
cisco-sr-te-cfp:sr-te odn odn-template Delay_Disjoint_72
 color 72
 dynamic pce
 dynamic flex-alg 128
 dynamic disjoint-path type link
 dynamic disjoint-path group-id 16
!
cisco-sr-te-cfp:sr-te odn odn-template Delay_70
 color 70
 dynamic pce
 dynamic flex-alg 128
!
```

We are also going to provide Service Health which will include SR-PM probing on the SR-TE tunnels. So, we need to build the prerequisite headless PM profiles and also customize the heuristic package with a 7ms latency threshold (not shown).

```
pm pm-profiles delay-profile sr-policy profile 1wayDelay
 probe computation-interval 60
 probe tx-interval    30000
 probe protocol       twamp-light
 probe measurement-mode one-way
 advertisement periodic interval 60
 advertisement periodic threshold 20
 advertisement periodic minimum-change 1000
!
pm svc-profiles 1wayDelaySvc
 performance-measurement delay-profile sr-policy profile 1wayDelay
```

9.3.11.1.2 Step 2 Create Slicing Catalog Template

Below are two catalog entries for the scenario, each referencing one of the forwarding plane policies, as highlighted, we will add some additional functionality into the original scenario such as QoS, Service Health and Performance Measurement. We will also be proactive and allow this template to support future L2 based services including L2 point-to-point with Y1731 Service Assurance. L2 is not used in this scenario but the template can support either L3 or L2.

```
network-slice-services slo-sle-templates slo-sle-template URLLC_Disjoint
 template-description         "Low latency service with Disjoint Paths and 7ms SLO with SR-PM "
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy Delay_Disjoint_72
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 1wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 1wayDelay
!
network-slice-services slo-sle-templates slo-sle-template URLLC_nonDisjoint
 template-description         "Low latency service with 7ms SLO with SR-PM "
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy Delay_70
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 1wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 1wayDelay
```

9.3.11.1.3 Step 3 Create Slicing Service Instance

In this step we will create the needed slice instances, starting with the Shared Slice Instance by first giving it a Slice ID (HQ-slice) and then referencing the catalog template just created (URLLC_nonDisjoint) identifying the slice intent. This intent specifies the treatment of traffic sent to the hub site (Node-5). We will identify one endpoint (called SDPs) and identify the Node-5 interface.

```
network-slice-services slice-service HQ_slice
 service-description "L3 A2A shared slice"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
!
 slo-sle-template URLLC_nonDisjoint
  sdps sdp 1
  node-id Node-5
  service-match-criteria match-criterion 1
   target-connection-group-id group1
```

```
   !
   attachment-circuits attachment-circuit ac2
    ac-tp-id            TenGigE0/0/0/2
    ac-ip-address       172.16.1.1
    ac-ip-prefix-length 29
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 301 ]
    !
    sdp-peering protocol peering-protocol-bgp
     bgp-attributes neighbor [ 172.16.1.2 ]
     bgp-attributes remote-as 65101
    !
   !
  !
  connection-groups connection-group group1
    connectivity-type any-to-any
  !
  isolation        service-isolation-shared
  !
 !
```

Now let's setup the dedicated slice instance. Notice this slice is being setup as hub-spoke (with no hub endpoint defined), to connect to a shared slice instance and to not use single-sided-control. Recall that with single-sided control enabled, the dedicated slice is also controlling the forwarding traffic towards the shared slice endpoint. In essence overriding the shared slice policy. In this case we do not want that behavior. The dedicated slice is also referencing the catalog template (URLLC_Disjoint) identifying the slice intent. This intent specifies the treatment of traffic sent from the shared site (Node-5) to the dedicated slice endpoint.

```
network-slice-services slice-service branch_slice
 service-description "L3 A2A dedicated slice connected to shared slice with disjointness"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
!
 connection-groups connection-group group1
   connectivity-type hub-spoke
 !
 slo-sle-template URLLC_Disjoint
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group1
   connection-group-sdp-role  spoke-role
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
 !
 service-tags tag-opaque nssai
  value [ 2-123459876 ]
 !
 service-tags tag-type service-tag-customer
  value [ FOO ]
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group1
   connection-group-sdp-role  spoke-role
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.3.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 601 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.3.2 ]
```

```
    bgp-attributes remote-as 65103
   !
  !
 !
 isolation        service-isolation-dedicated
 !
 shared slice HQ_slice
 !
 shared single-sided-control false
!
```

9.3.11.1.4 Resultant Configurations

Given the above scenario, let's review the configurations automatically pushed down to the three devices. Compare this with the ala-carte approach from <Insert Link>. This includes:

- Configuring the needed route-policy maps with BGP color community
- Configuring VRFs with associated RTs for the connectivity requested
- Adding export route policy maps to the VRF for proper BGP color community advertisements
- Creating the interface (i.e. sub-interfaced VLANs)
- Associating the interface IP and VRF information to the interface
- Adding the appropriate BGP information for the vrf, including any PE-CE eBGP peers
- Deploying the SR-TE ODN policy maps with proper coloring.
- Deploying SR-PM configurations to the ODN policy
- Deploying the Service Health with the associated heuristic packages to enable monitoring of specific telemetry sensor points with monitoring

9.3.11.1.5 Step 4- Visualize the New Slice to See the Traffic Path

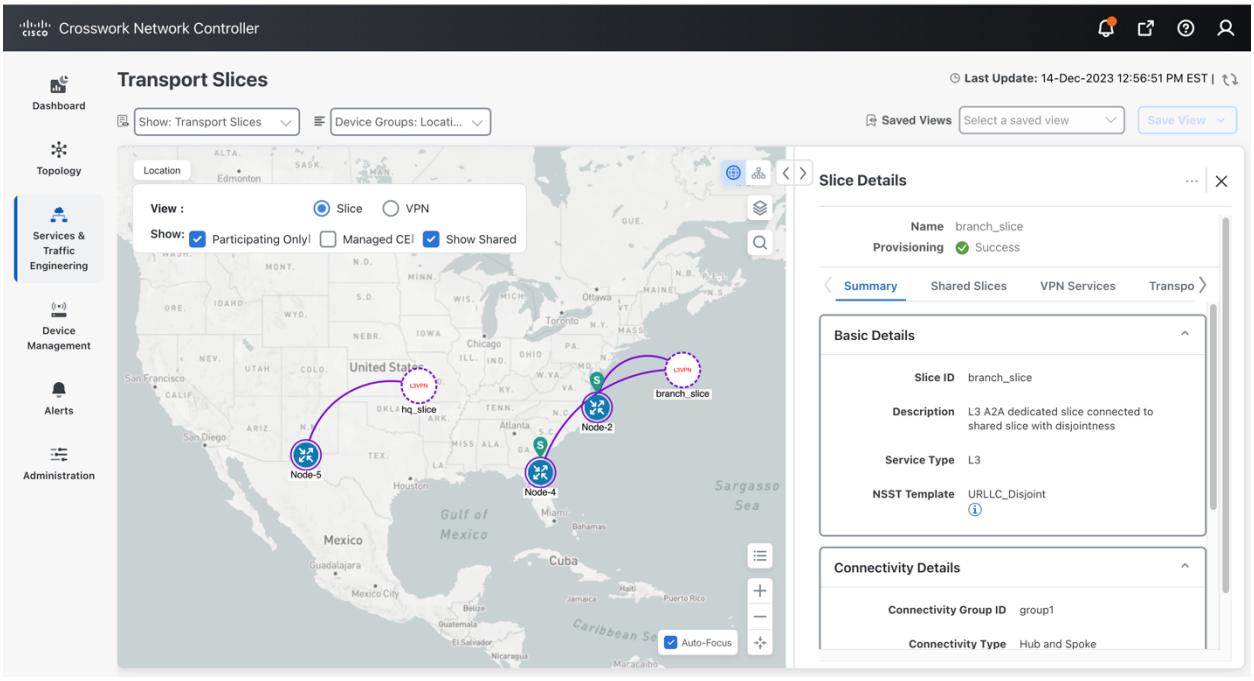

Figure 107- Scenario #9 Solution #1 Slice Summary View

Notice in the below figure that the two paths (Mode-5 to Node-4) and (Node-5 to Node-2) take diverse paths and prefer the low latency path (i.e. it avoids the link between Node-8 and Node-3 has a high latency metric, but also a high BW). The user could have also specified “node” diversity instead of “link” diversity in the intent, which would have given a much different set of paths.

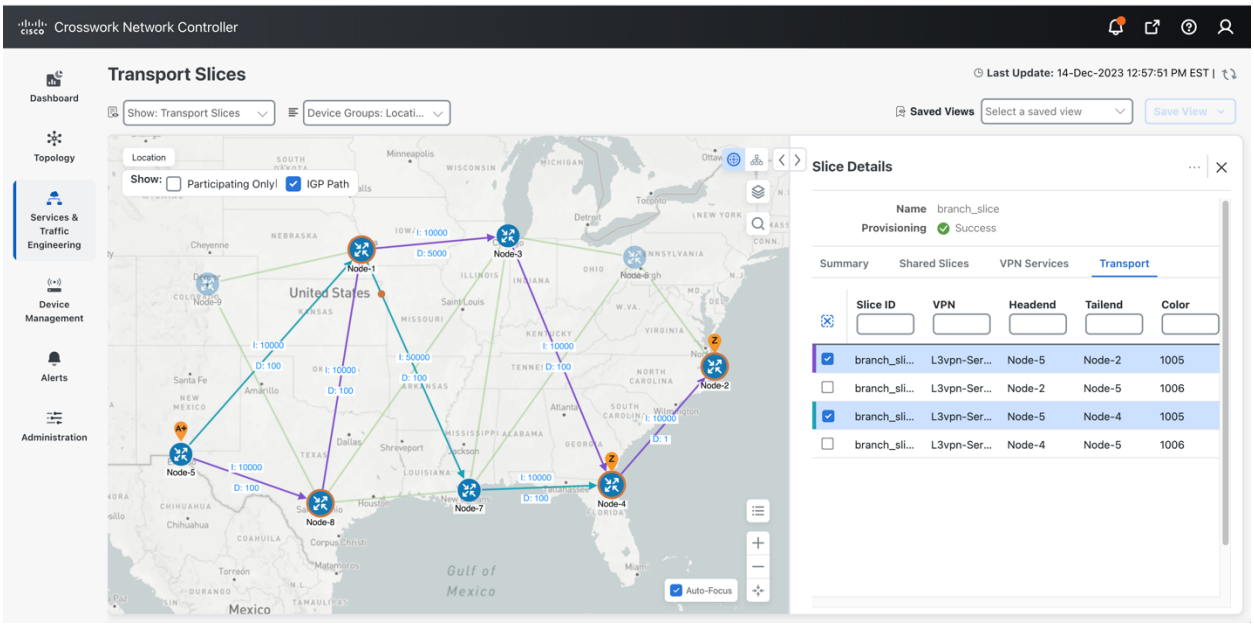

Figure 108: Disjoint Link View Solution #1

Further examination of the path from Node-5 to Node-2 shows the delay and jitter performance measurements. These are used by Service Health to verify the 7ms latency objective.

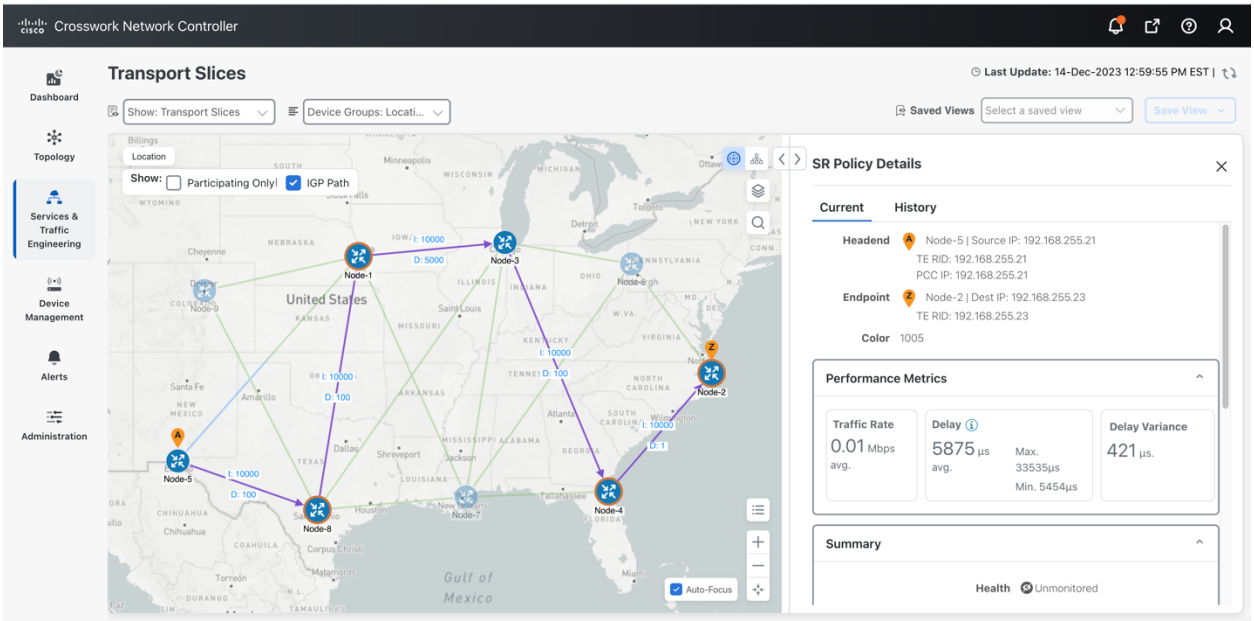

Figure 109: Performance Measurement View

9.3.11.1.6 Step 6 Inspect a degraded service using Service Health to determine symptoms

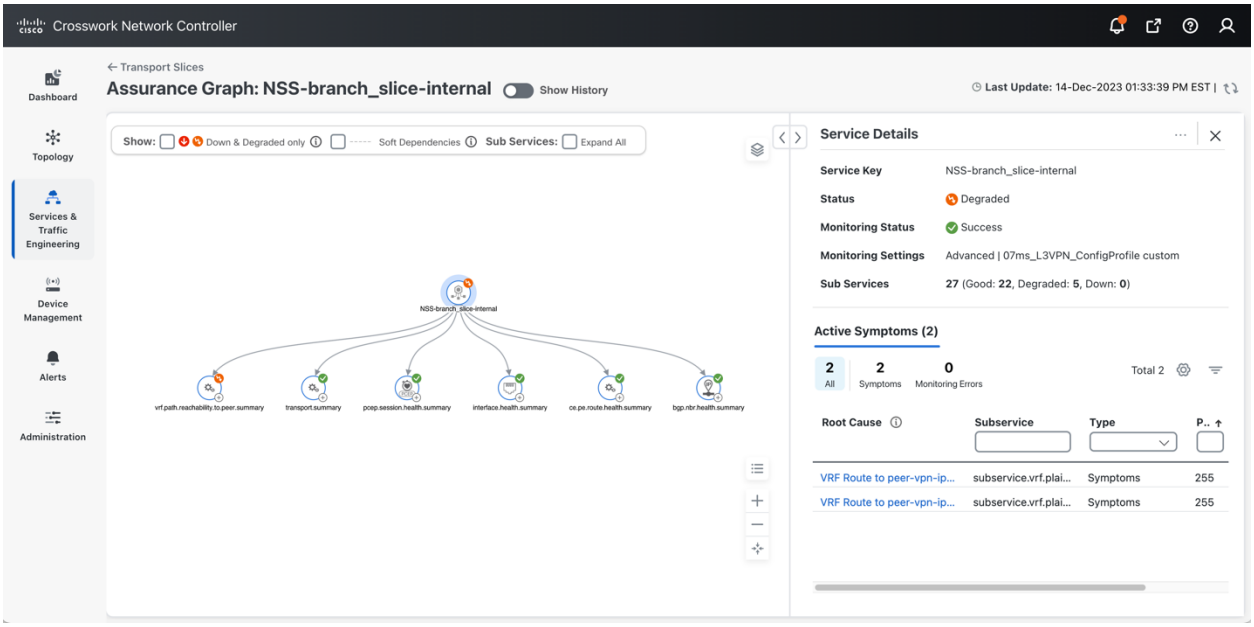

Figure 110: Service Health View

9.3.11.2 Scenario #9 Solution #2 Walk-Through- Using Flex-Algo for disjointness

Use two dedicated slices (one for the endpoint on Node-4 and one for the endpoint on Node-2) and connect both to a shared slice at Node-5. For each of the dedicated slices, use separate disjoint Flex-Algo based policies using link affinities (if you recall we have a “Disjoint-North” and a “Disjoint-South” flex-algo forwarding plane already configured). This approach not only keeps traffic from Node-5 disjoint, but also the return traffic is disjoint.

Step 1 Create headless ODN templates to map SLA objective and constraints to intents
Step 2 Create the Slice Catalog Entries
Step 3 Create and provision the Shared and Dedicated Slicing Services
Step 4 Visualize the New VPN Service on the Map to See the Traffic Path

9.3.11.2.1 Step 1 Create ODN templates and PM profiles

The administrator has also set up a Disjoint-Path topology, again with link-affinities and Flex-Algo. Flex-Algo 131 will be used for “Disjoint-North” path and Flex-Algo 132 will be used for “Disjoint-South”. See Figure **55**. We will also continue to use the previously deployed policy Delay_70 for the shared slice.

We will create the following ODN templates:

color 73, latency, Disjoint-North (flex-algo 131) - Delay_North_73
color 74, latency, Disjoint-South (flex-algo 132) - Delay_ South_74
color 70, latency (flex-algo 128) - Delay_70

Before you begin
In this step, we will create a single ODN template (color 73) similar to the original example, but without the head-ends specified. The ODN template specifies the color and the intent; in this case, latency and dis-jointness is implied via the flex-algo being referenced. This ODN template will be used to dynamically create tunnels (on-demand) when prefixes with matching colors are received via BGP. Traffic to these prefixes will be automatically steered into the newly created tunnels, thereby meeting the SLA objective and constraints intended for these prefixes and signaled using colors in the BGP routes.

The same procedure is followed from the original scenario to create the ODN templates. Below is the CLI output from NSO with no head-ends defined:

```
cisco-sr-te-cfp:sr-te odn odn-template Delay_North_73
 color 73
 dynamic pce
 dynamic metric-type latency
 dynamic flex-alg 131
!
cisco-sr-te-cfp:sr-te odn odn-template Delay_South_74
 color 74
 dynamic pce
 dynamic metric-type latency
 dynamic flex-alg 132
!
cisco-sr-te-cfp:sr-te odn odn-template Delay_70
 color 70
 dynamic pce
 dynamic flex-alg 128
!
```

We are also going to provide Service Health which will include SR-PM probing on the SR-TE tunnels. So, we need to build the prerequisite headless PM profiles and also customize the heuristic package with a 7ms latency threshold (not shown). This is the same step as in Solution #1 above.

```
pm pm-profiles delay-profile sr-policy profile 1wayDelay
 probe computation-interval 60
 probe tx-interval    30000
 probe protocol       twamp-light
 probe measurement-mode one-way
 advertisement periodic interval 60
 advertisement periodic threshold 20
 advertisement periodic minimum-change 1000
!
pm svc-profiles 1wayDelaySvc
 performance-measurement delay-profile sr-policy profile 1wayDelay
```

9.3.11.2.2 Step 2 Create Slicing Catalog Template

Below are three catalog entries for the scenario, each referencing one of the forwarding plane policies. As highlighted, we will add some additional functionality into the original scenario such as QoS, Service Health and Performance Measurement. We will also be proactive and allow this template to support future L2 based services including L2 point-to-point with Y1731 Service Assurance. L2 is not used in this scenario but the template can support either L3 or L2.

```
network-slice-services slo-sle-templates slo-sle-template URLLC_Disjoint_North
 template-description         "Low latency service with Disjoint Paths and 7ms SLO with SR-PM "
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy Delay_North_73
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 1wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 1wayDelay
!
network-slice-services slo-sle-templates slo-sle-template URLLC_Disjoint_South
 template-description         "Low latency service with 7ms SLO with SR-PM "
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy Delay_South_74
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 1wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 1wayDelay
!
network-slice-services slo-sle-templates slo-sle-template URLLC_nonDisjoint
 template-description         "Low latency service with 7ms SLO with SR-PM "
 qos-policy L2 input-policy ingress_COS5
 qos-policy L2 output-policy Egress-LowLatency
 qos-policy L3 input-policy ingress_COS5
 qos-policy L3 output-policy Egress-LowLatency
 odn forwarding-plane-policy Delay_70
 odn forwarding-plane-policy-type as-blueprint
 service-assurance heuristics monitoring-state enable
 service-assurance heuristics L2 point-to-point profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 point-to-point rule-name "Rule-L2VPN-NM custom"
 service-assurance heuristics L2 multipoint profile-name "07ms_L2VPN_ConfigProfile custom"
 service-assurance heuristics L2 multipoint rule-name "Rule-L2VPN-MP custom"
 service-assurance heuristics L3 profile-name "07ms_L3VPN_ConfigProfile custom"
 service-assurance heuristics L3 rule-name "Rule-L3VPN-NM custom"
 service-assurance ethernet-service-oam md-name foo
 service-assurance ethernet-service-oam md-level 4
 service-assurance ethernet-service-oam y-1731 id-type icc-based
 service-assurance ethernet-service-oam y-1731 message-period 1s
 service-assurance ethernet-service-oam y-1731 profile-delay Profile-Delay-1
 service-assurance performance-measurement sr-te pm-svc-profile 1wayDelaySvc
 service-assurance performance-measurement sr-te delay-measurement profile 1wayDelay
```

9.3.11.2.3 Step 3 Create Slicing Service Instance

In this step we will create the needed slice instances, starting with the Shared Slice Instance by first giving it a Slice ID (HQ-slice) and then referencing the catalog template just created (URLLC_nonDisjoint) identifying the slice intent. This intent specifies the treatment of traffic sent to the hub site (Node-5). We will identify one endpoint (called SDPs) and identify the Node-5 interface.

```
network-slice-services slice-service HQ_slice
 service-description "L3 A2A shared slice"
```

```
  service-tags tag-type service-tag-service
   value [ L3 ]
  !
 !
  slo-sle-template URLLC_nonDisjoint
   sdps sdp 1
   node-id Node-5
   service-match-criteria match-criterion 1
    target-connection-group-id group1
   !
   attachment-circuits attachment-circuit ac2
    ac-tp-id            TenGigE0/0/0/2
    ac-ip-address       172.16.1.1
    ac-ip-prefix-length 29
    ac-tags ac-tags attachment-circuit-tag-vlan-id
     value [ 301 ]
    !
    sdp-peering protocol peering-protocol-bgp
     bgp-attributes neighbor [ 172.16.1.2 ]
     bgp-attributes remote-as 65101
    !
   !
  !
  connection-groups connection-group group1
    connectivity-type any-to-any
  !
  isolation        service-isolation-shared
  !
 !
```

Now let's setup the dedicated slice instances (branch_slice1 and branch_slice2). Notice this slice is being setup to connect to a shared slice instance and will use single-sided-control. Recall that with single-sided control enabled, the dedicated slice is also controlling the forwarding traffic towards the shared slice endpoint. In essence overriding the shared slice policy. In this case we do want the traffic to bi-directionally use the same disjoint forwarding plane. The dedicated slices are referencing their respective catalog template (North_URLLC_Disjoint and South_URLLC_Disjoint) identifying the slice intent. This intent specifies the treatment of traffic sent from the shared site (Node-5) to the dedicated slice endpoint and thus will follow the disjoint path flex-algo specified. Since single-sided control is enabled, traffic from the dedicated slice to the shared slice will also follow the disjoint path.

```
network-slice-services slice-service branch_slice2
 service-description "L3 A2A dedicated slice connected to shared slice with disjointness"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
!
 connection-groups connection-group group2
   connectivity-type any-to-any
 !
 slo-sle-template URLLC_Disjoint_South
 sdps sdp 1
  node-id Node-4
  service-match-criteria match-criterion 1
   target-connection-group-id group2
  !
  attachment-circuits attachment-circuit ac1
   ac-tp-id            TenGigE0/0/0/10
   ac-ip-address       172.16.2.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 401 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.2.2 ]
    bgp-attributes remote-as 65102
   !
  !
 !
 isolation        service-isolation-dedicated
 !
 shared slice hq_slice
 !
 shared single-sided-control true
!
```

```
network-slice-services slice-service branch_slice1
 service-description "L3 A2A dedicated slice connected to shared slice with disjointness"
 service-tags tag-type service-tag-service
  value [ L3 ]
 !
!
 connection-groups connection-group group3
   connectivity-type any-to-any
 !
 slo-sle-template URLLC_Disjoint_North
  sdps sdp 3
  node-id Node-2
  service-match-criteria match-criterion 1
   target-connection-group-id group3
  !
  attachment-circuits attachment-circuit ac3
   ac-tp-id            TenGigE0/0/0/2
   ac-ip-address       172.16.3.1
   ac-ip-prefix-length 29
   ac-tags ac-tags attachment-circuit-tag-vlan-id
    value [ 601 ]
   !
   sdp-peering protocol peering-protocol-bgp
    bgp-attributes neighbor [ 172.16.3.2 ]
    bgp-attributes remote-as 65103
   !
  !
 !
 isolation        service-isolation-dedicated
 !
 shared slice hq_slice
 !
 shared single-sided-control true
!
```

9.3.11.2.4 Step 4- Visualize the New Slice to See the Traffic Path

In the below Traffic Engineering view, we can see the path dis-jointness for traffic sent from Node-5 to both Node-4 and Node-2.

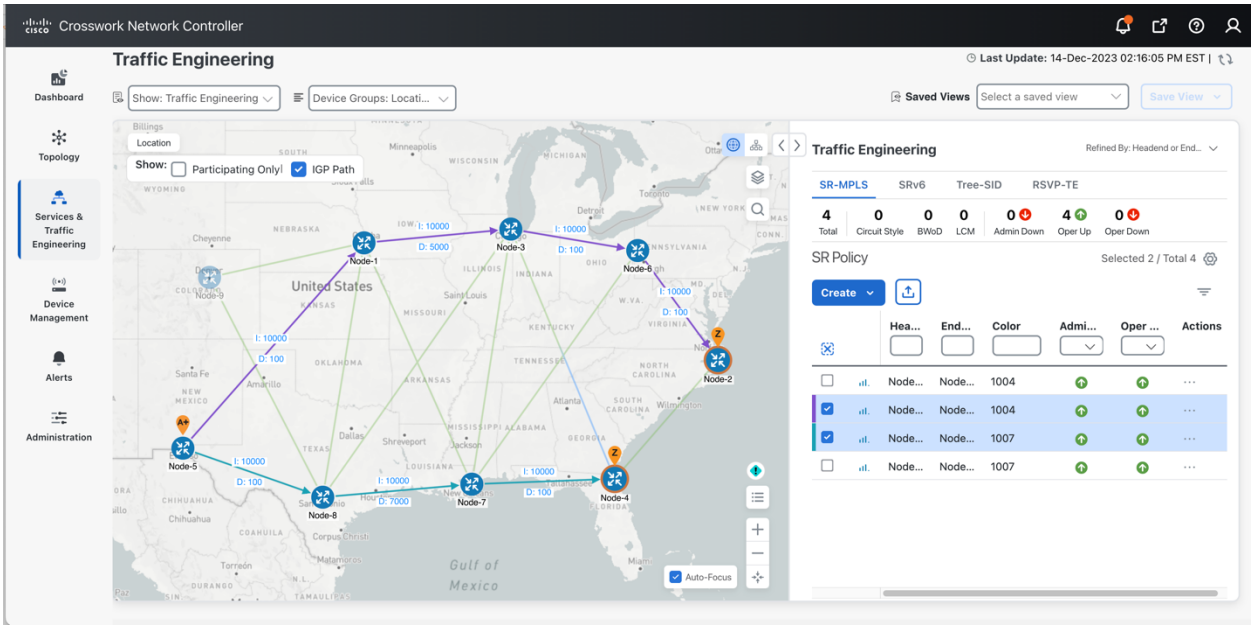

Figure 111: Disjoint Path View Solution #2

10 Appendix

10.1 Advanced Intents

The CNC 6.0 slicing service supports ODN template-based slice intents that are supported by the NSO T-SDN CFP. Some advanced configurations are not yet available in the T-SDN CFP, for example cumulative constraints and flow-based intents. For demo and test purposes we can still use these advanced capabilities for slicing but the ODN templates must be built manually and pre-positioned onto the devices or we can use the custom templates feature. Thus, the slicing service can use any pre-built ODN template and make use of any advanced SR-TE functionality supported by ODN. These advanced features are supported by IOS XR CLI but may not yet be supported by the NSO SR-TE CFP, so a workaround exists. First, create a simple ODN template (without the advanced features) for that intent inside the T-SDN CFP, with a color and include all headend nodes. This will pre-position the ODN templates on all nodes. Next, modify the ODN template directly on each headend device via IOS-XR CLI with any missing advanced commands and then do a NSO sync-from. This procedure will allow unsupported NSO SR-TE CFP commands to work with slicing.

10.1.1 Cumulative Constraint based ODN Templates

Cumulative constraint-based templates allow for multiple metrics to be specified. The example most relevant for slicing services would be to specify an upper delay bound to not exceed. In this case the lowest latency is not required, but the user does not want to exceed an upper bound. While this cumulative constraints feature has been supported in IOS-XR CLI and NSO device NED, it is not yet available in the NSO SR-TE CFP. Below is an example enabling the workaround for this capability.

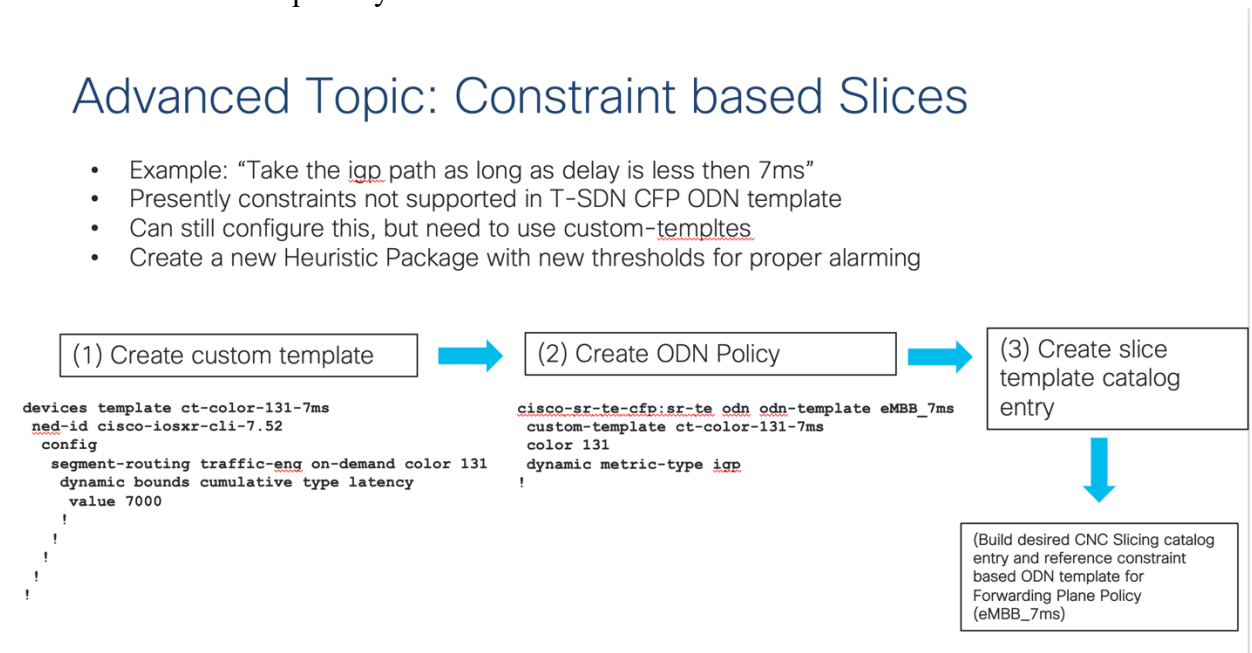

Figure 112 Example: Cumulative Constraint Based Intents.

10.1.2 Flow-based Intents (for demo only)

IOS-XR supports an advanced capability to match on specific “flow criteria” (usually DSCP) on ingress and to use this as classification for the SR-TE path forwarding mechanism. Presently these advanced ODN template commands are not supported by CNC or the NSO SR-TE CFP. We can use a workaround to demo this capability by first creating the needed QoS policy maps to set the devices Forwarding Class (FC) bits on ingress (this is device type dependent), then create a custom-template with the CFP unsupported commands and then attach it to a simple ODN template for Flow processing (color 500 below). The custom template must contain all the subordinate (flow level) ODN policies.

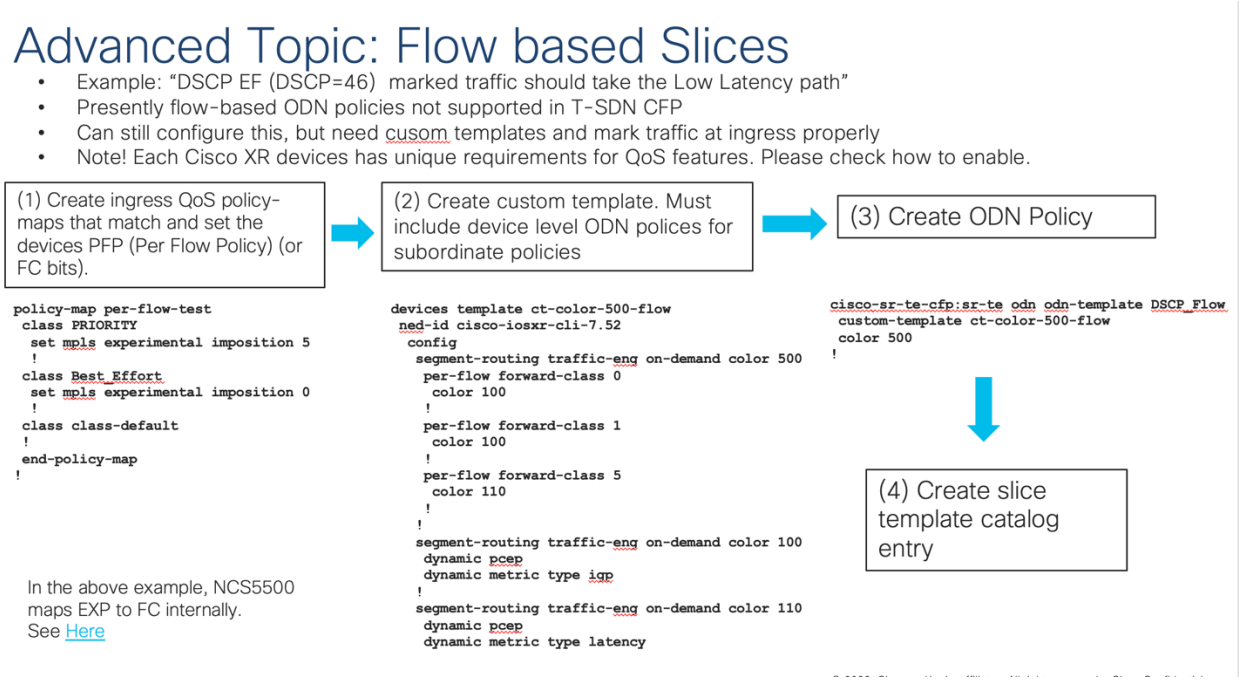

Figure 113 Example: Flow-based Intent