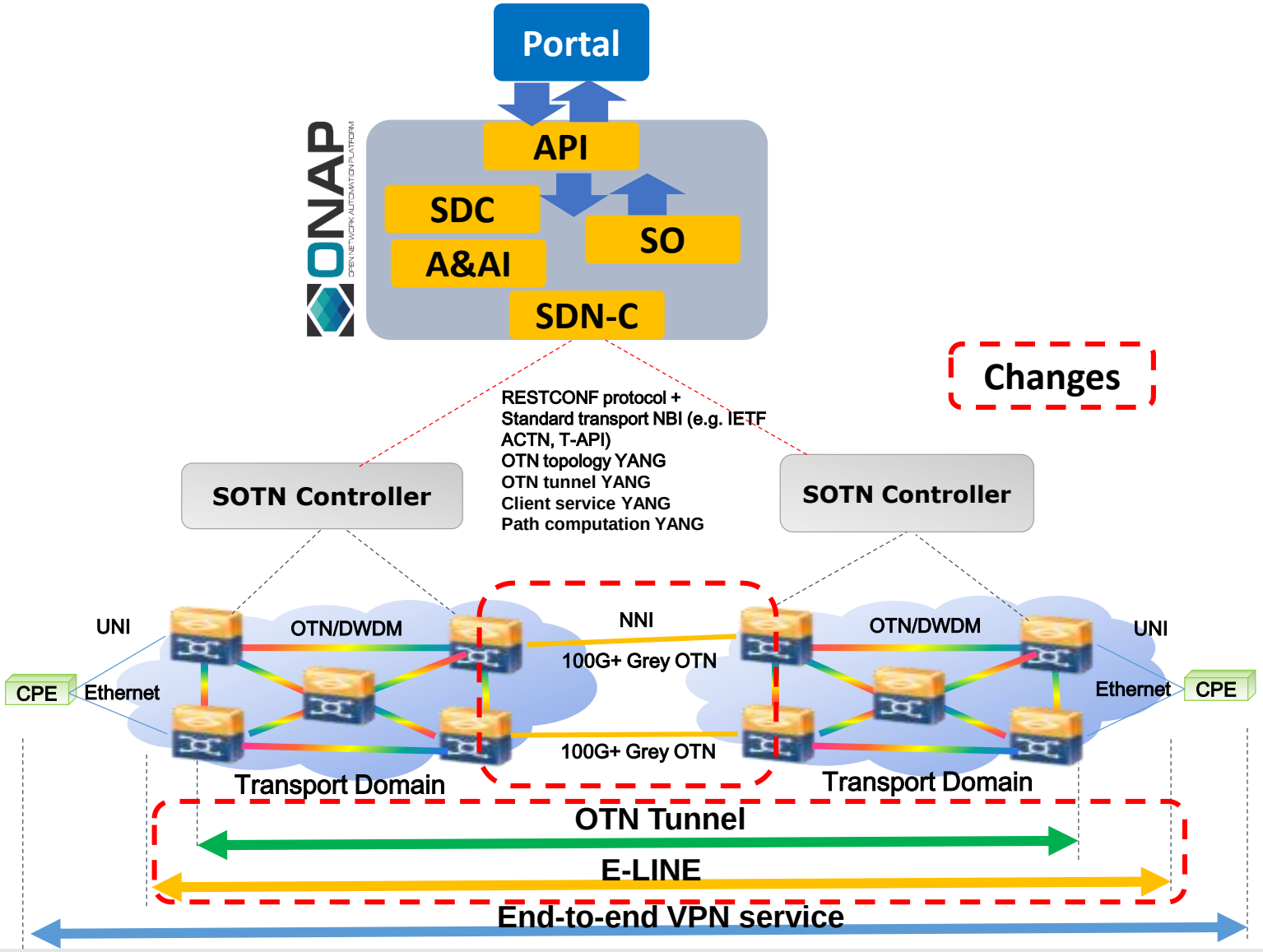
CCVPN: SDNC Changes for NNI-SOTN Support Low Level Design

Use Case Description



Changes Description:

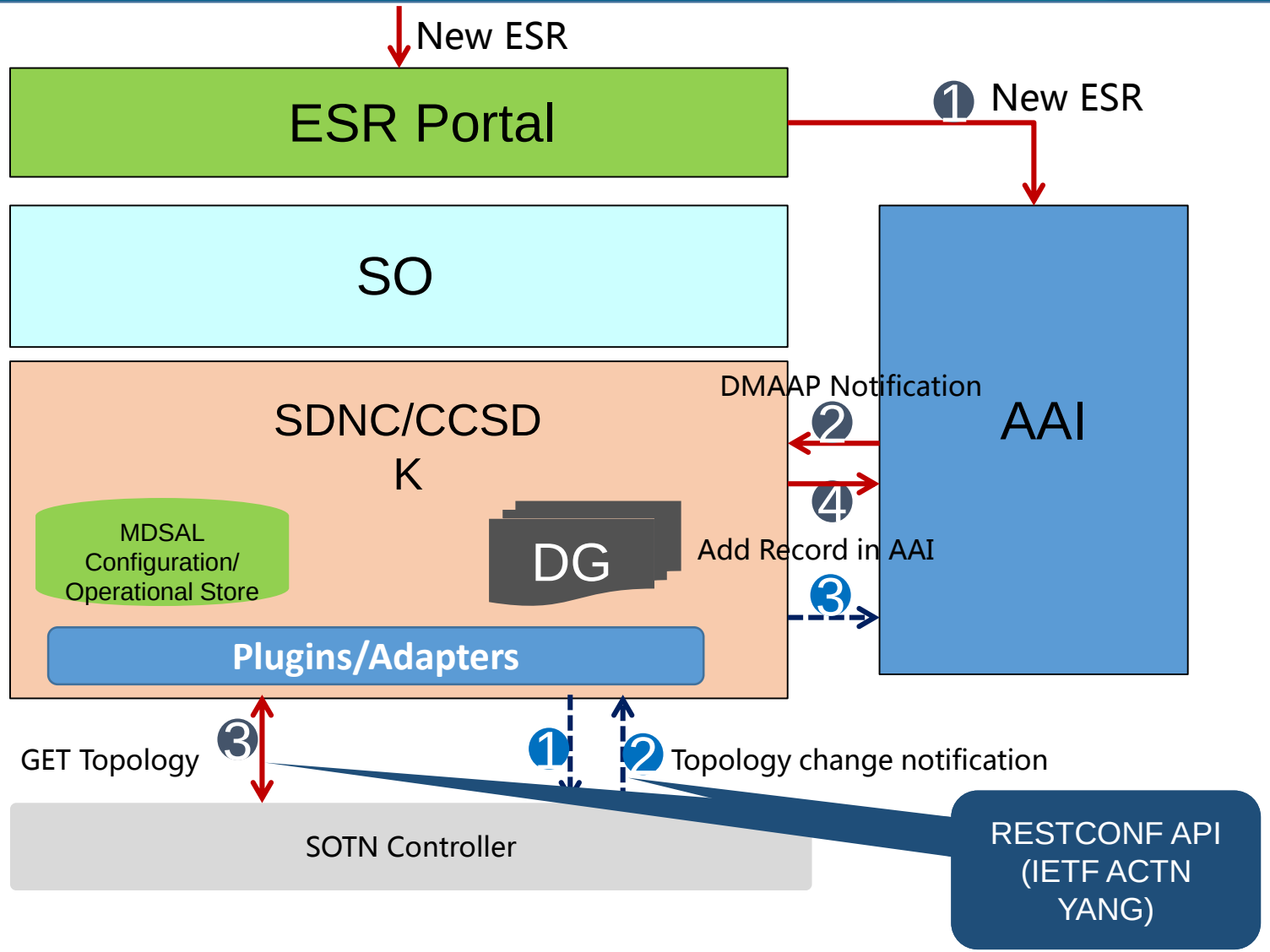
- NNI handover between 2 OTN domain.
- E2E OTN tunnel across multiple domains
- E2E E-LINE service

Scope and Structure of the LLD

The NNI-SOTN support is an enhancement/extension of the existing SOTN use case. As a result, this LLD covers the design changes and enhancements of the SDNC, in addition to the existing functionality. For a more complete architectural illustration of the design, one should read this document in conjunction with the previous SDNC LLD.

This LLD is divided into 2 parts. The first part focuses on the design changes needed for topology discovery and topology change synchronization. The second part discusses the design changes required for resource configuration.

Required Changes for Topology Discovery and Synchronization



Initial Topology Synchronization

1) New 3rd party controller is added to AAI
2) AAI sends notification to SDNC for new controller addition.
3) SDNC get topology from SOTN controller.
4) SDNC add topology information to AAI.

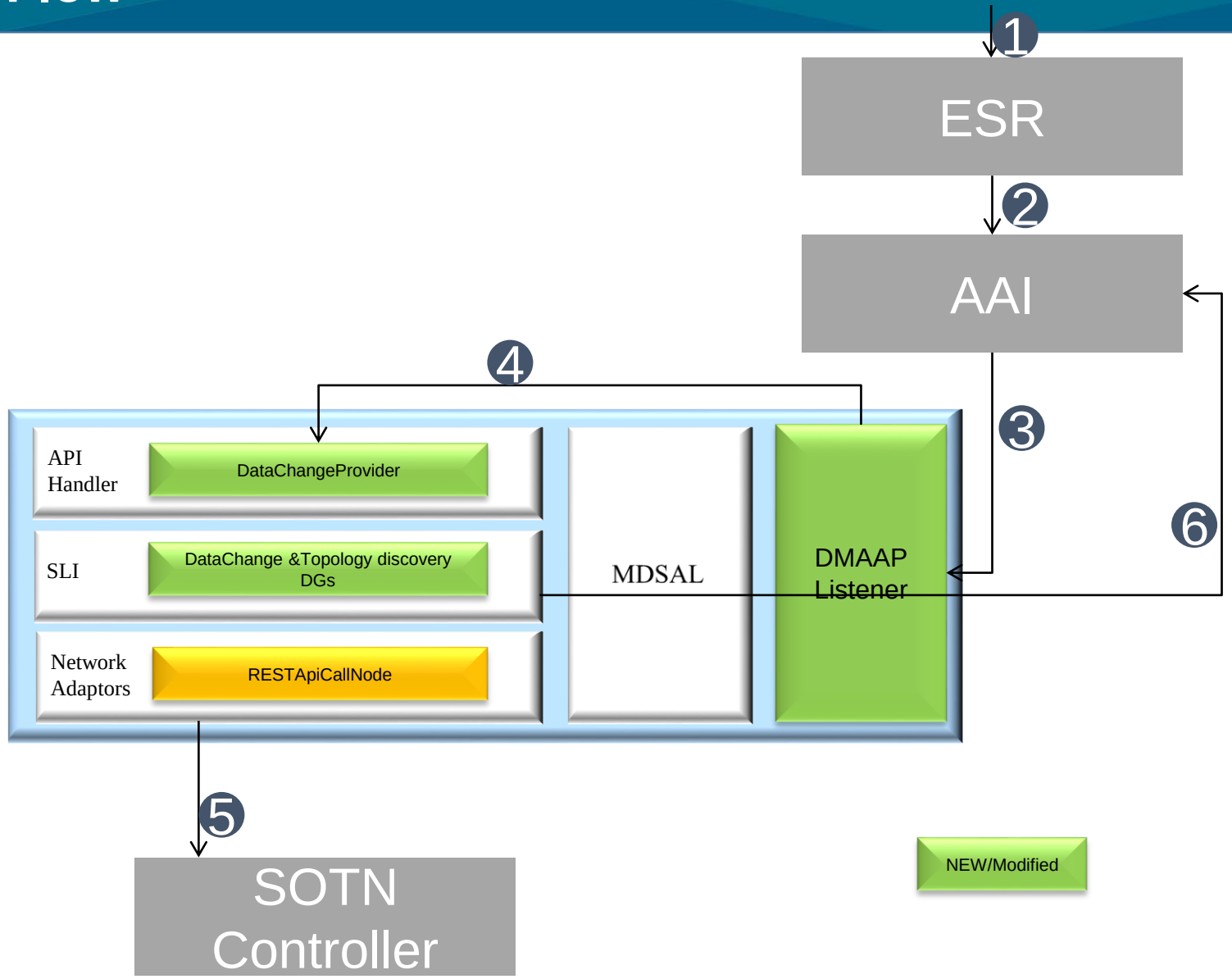
Continuous Synchronization

1) SDNC subscribes for topology change to SOTN controller.
2) SOTN notifies Topology change to SDNC.
3) SDNC updates topology information to AAI

Required Changes:

- Topology discovery framework: **Need to discover multi-layer and native topologies.**
- Topology discovery DGs: **Changes are needed to support the above.**
- Notification handling: **Ethernet links are dynamic links. They need to be created in AAI after SDNC receives link creation notification from NCE-t.**

SDNC/CCSDK Design: Physical Network Topology Onboarding Framework & Flow



1) User registers 3rd party controller information in ESR.

2) ESR adds 3rd party controller information in AAI.

3) AAI notify addition of 3rd party controller information to DMAAP Listener in SDNC.

4) DMAAP Listener in SDNC sends Data change request.

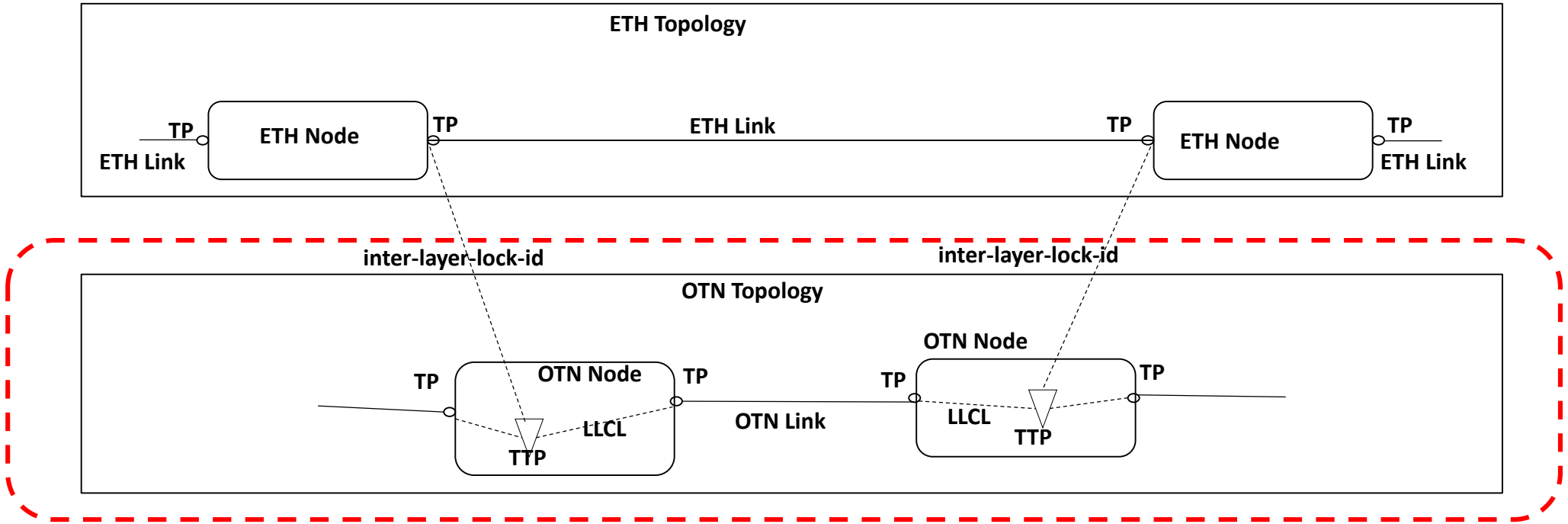
5) SDNC query topology from 3rd party Controller

6) SDNC adds topology to AAI

Development Changes:

- Listener to New 3rd party controller events: **Handle Ethernet link creation notification.**
- DataChangeProvider DGs: **Create Ethernet links in AAI**
- Topology Discovery DGs: **Discover and create multi-layer native topologies in AAI.**

Multi-layer Topology structure in AAI

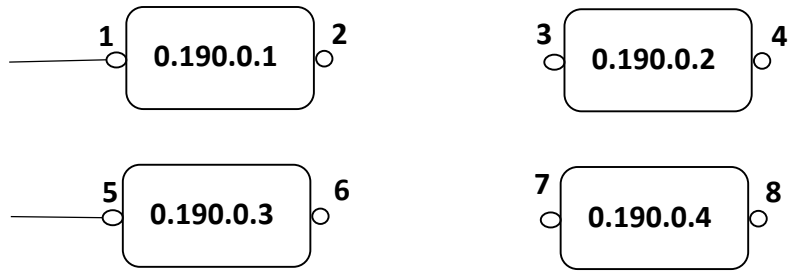


- AAI needs to have two layers (Eth and OTN) of topology.
- Need to add Tunnel Termination Point (TTP) attribute to Topology Node.
- TP in upper layer node (ETH Node) are associated with TTP in lower layer node (OTN Node) by inter-layer lock ID

Topology Discovery: Single Domain Topology Creation in AAI

SDNC discovers the 2-layer domain topology from NCE-T and save it into AAI. The following example shows the logical content of AAI. (Note that there are 2 domain topologies. For simplicity, only one is shown below.)

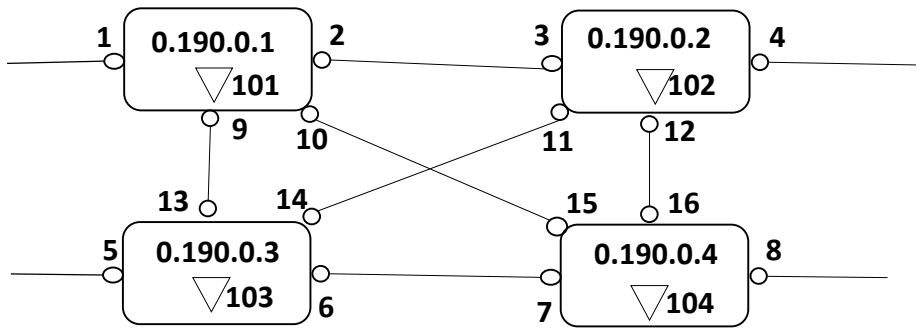
ETH Topology



TP	type	Inter-layer-Lock-id
1	UNI	
2	NNI	10100
3	NNI	10200
4	NNI	

TP	type	Inter-layer-Lock-id
5	UNI	
6	NNI	10300
7	NNI	10400
8	NNI	

OTN Topology



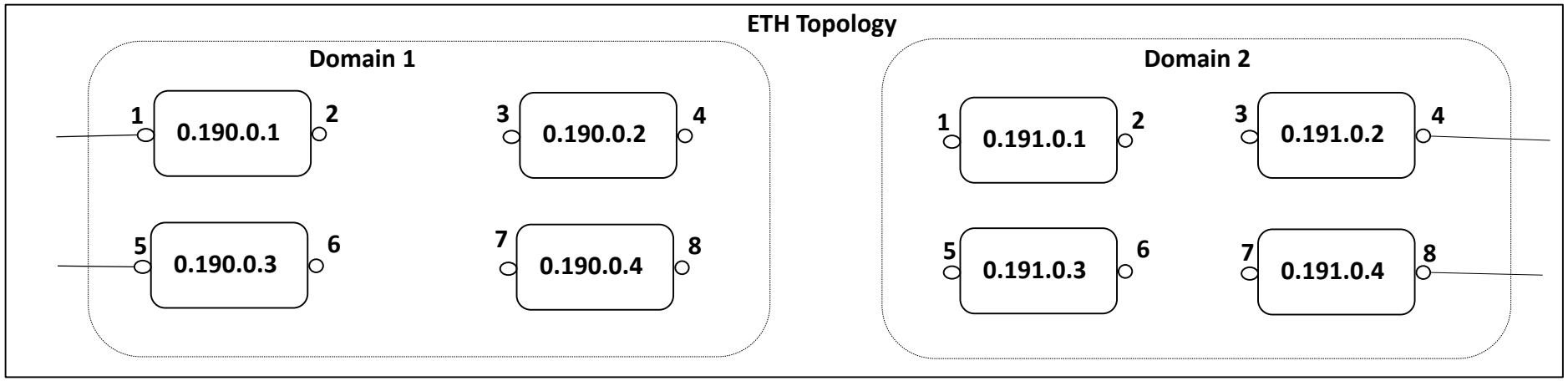
TP	type	Inter-layer-Lock-id	LLCL	Plug-id
1	UNI			
2	NNI			
3	NNI			
4	NNI			50001
101	TTP	10100	2	
102	TTP	10200	4	

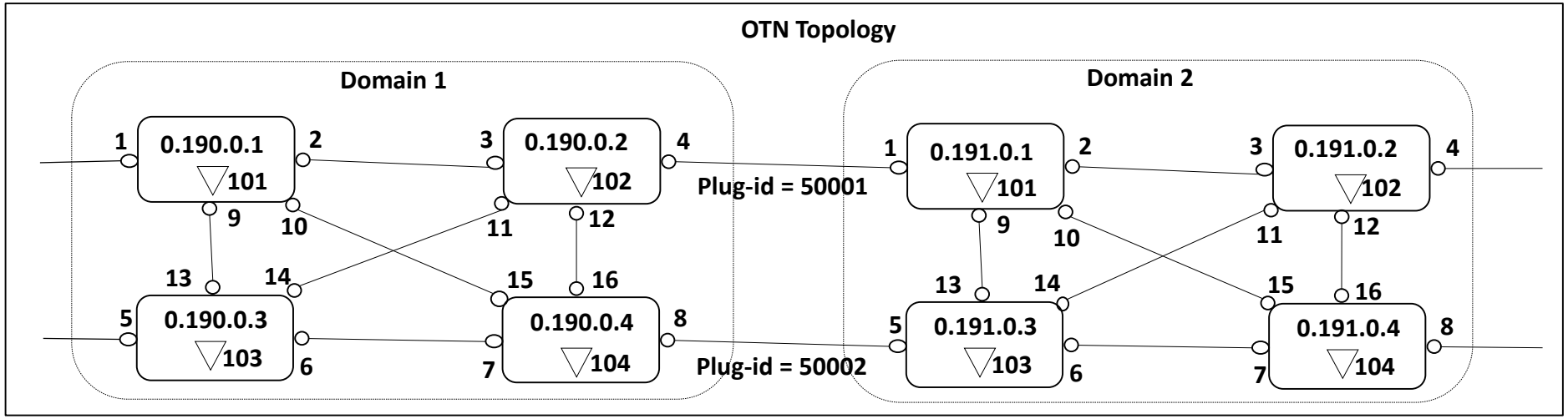
TP	type	Inter-layer-Lock-id	LLCL	Plug-id
5	UNI			
6	NNI			
7	NNI			
8	NNI			50002
103	TTP	10300	2	
104	TTP	10400	4	

NOTE: TPs from 9 to 16 are NNI ports which are omitted from the above tables due to limited space..

Topology Discovery: Merge Two OTN Domain Topologies in AAI

After the domain topologies are discovered and saved into AAI, SDNC merges the OTN topologies by connecting the inter-domain links by inter-domain-plug-id matching.

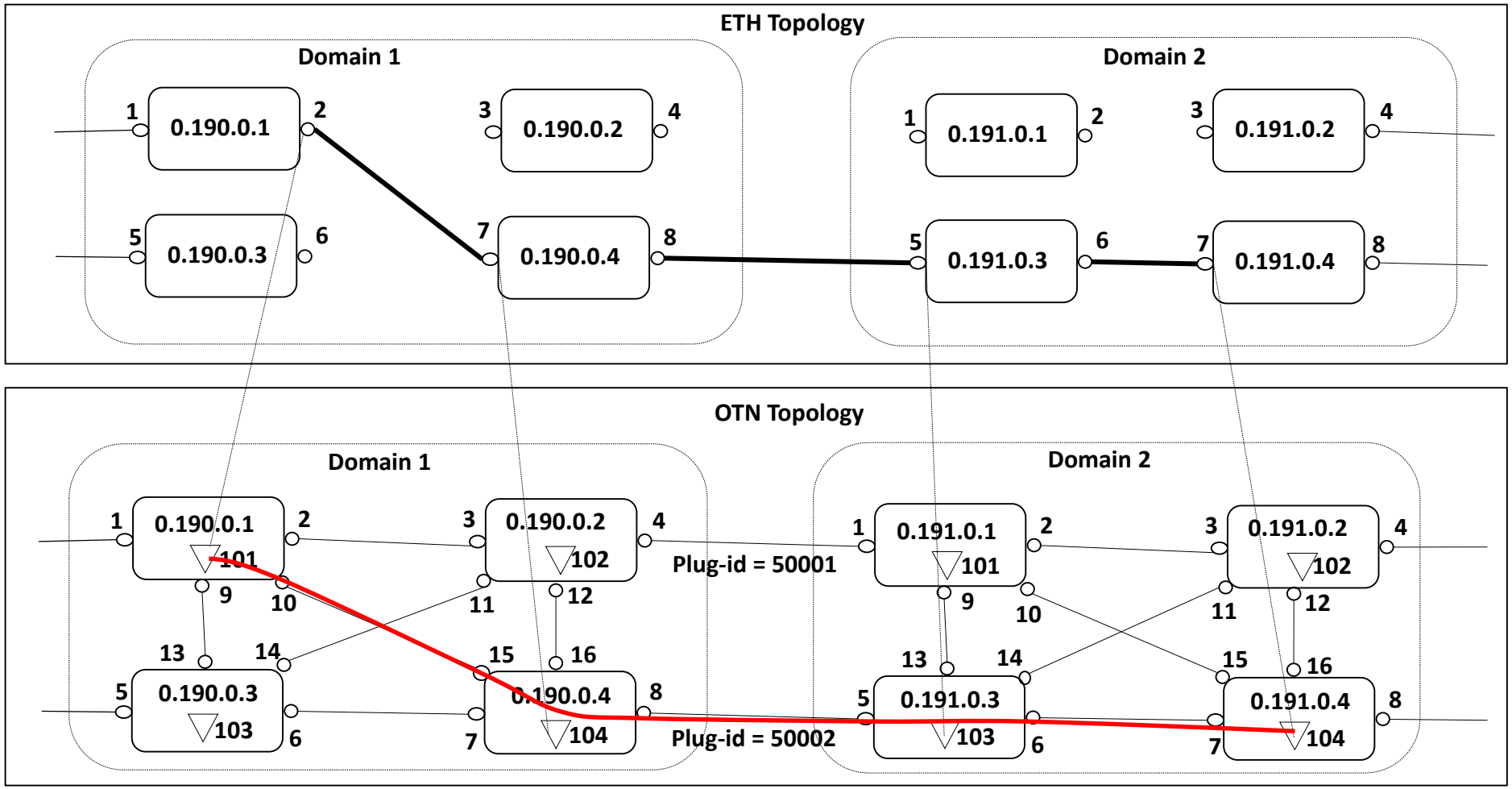




NOTE: Ethernet links are dynamic links, which means that they are not created until E-LINE services are provisioned.

Topology Discovery: Update Ethernet Links after E-LINE service creation

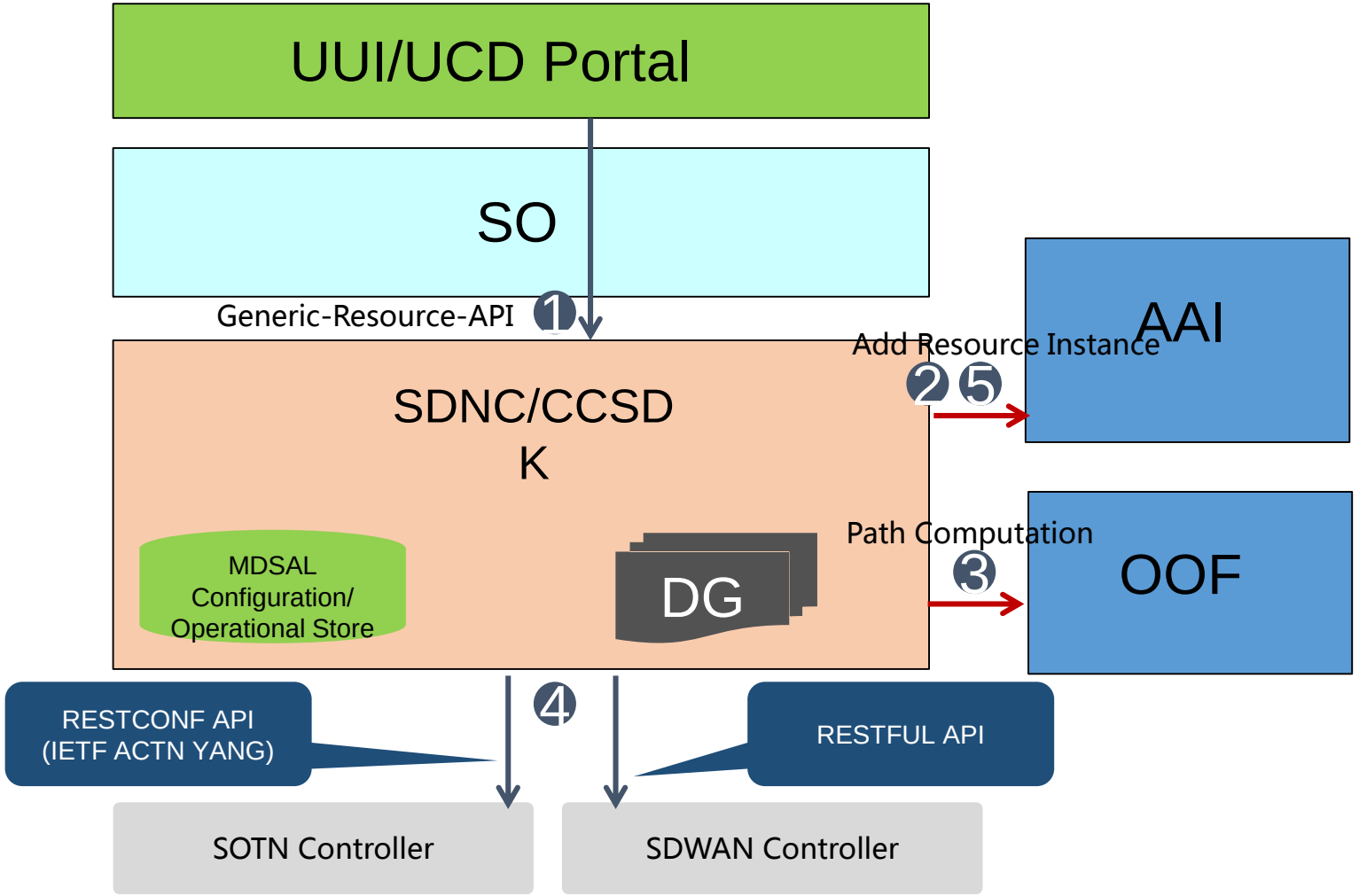
After an E-LINE service is created, SDNC will receive Ethernet Link creation notifications from NCE-T. SDNC will create the Ethernet links in AAI accordingly. For example, suppose an E-LINE from 0.190.0.1 port 1 to 0.191.0.4 port 8 is provisioned. Then the Ethernet links in the following figure will be created.



NOTE: The inter-domain Ethernet link is not reported by NCE as part of the notifications. SDNC needs to create the link when it receives the intra-domain link creation notifications in both domains.

NOTE: The red link represents the underlay OTN tunnel which supports the E-LINE. Note this this tunnel itself does not exist in AAI.

Required Changes for Resource Configuration

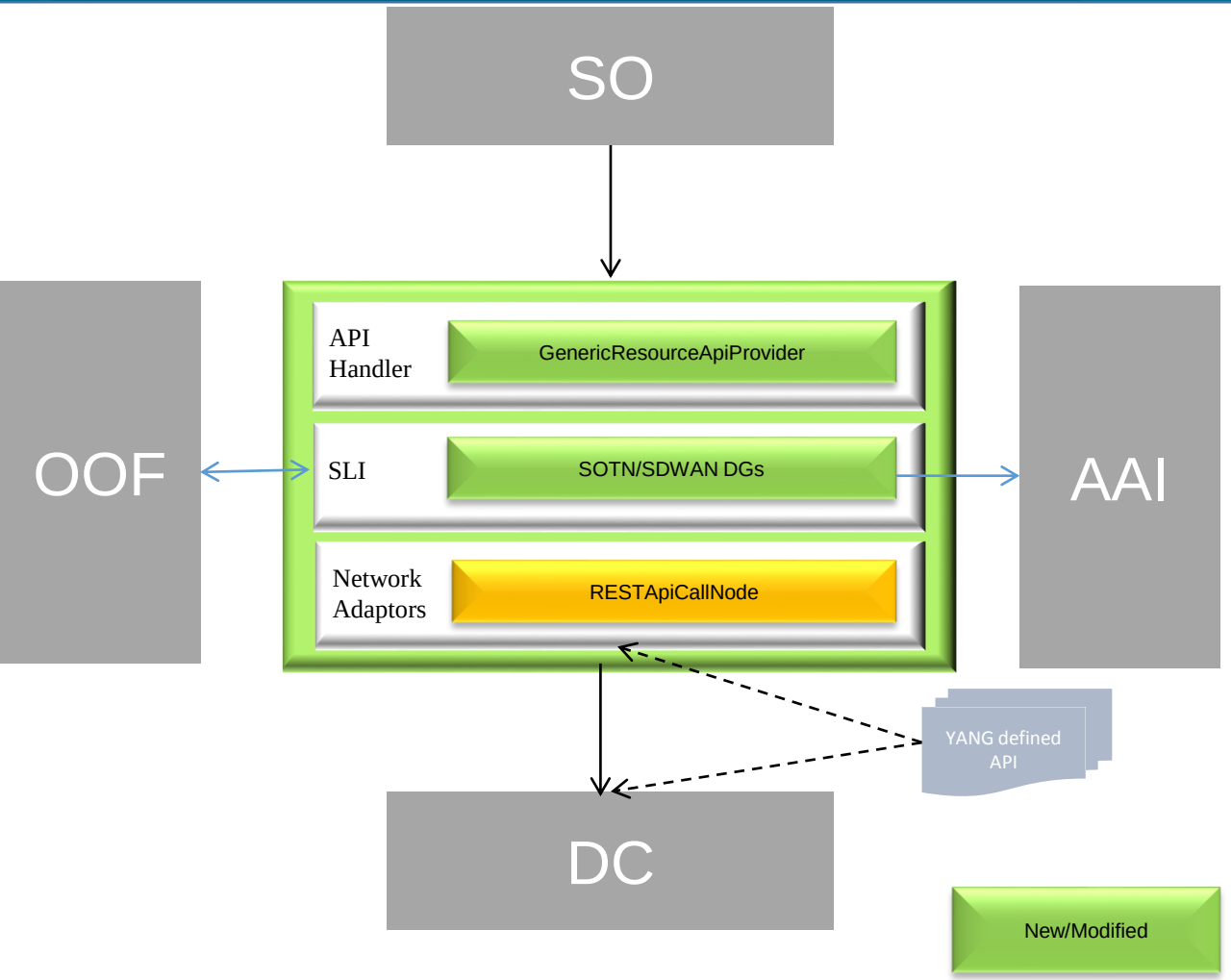


1) SO call SDNC to create/activate resources using generic-resource-api
2) SDNC store data in AAI for create requests.
3) SDNC call OOF for path computation (only applicable for SOTN)
4) SDNC call 3rd SOTN controller to create resource
5) SDNC stores created resource to AAI.

Required Changes:

- DGs for SOTN resources: **Changes to support new service resource definition**
- OOF Plugin in DG: **Changes to support new input & output parameters of OTN tunnel path computation**
- RESTCONF based CRUD operation: **ACTN L1 Service API support**

SDNC/CCSDK Design: Resource Configuration



1) SO Calls Generic Resource APIs corresponding to resource type.

2) SDNC calls OOF for path computation if required

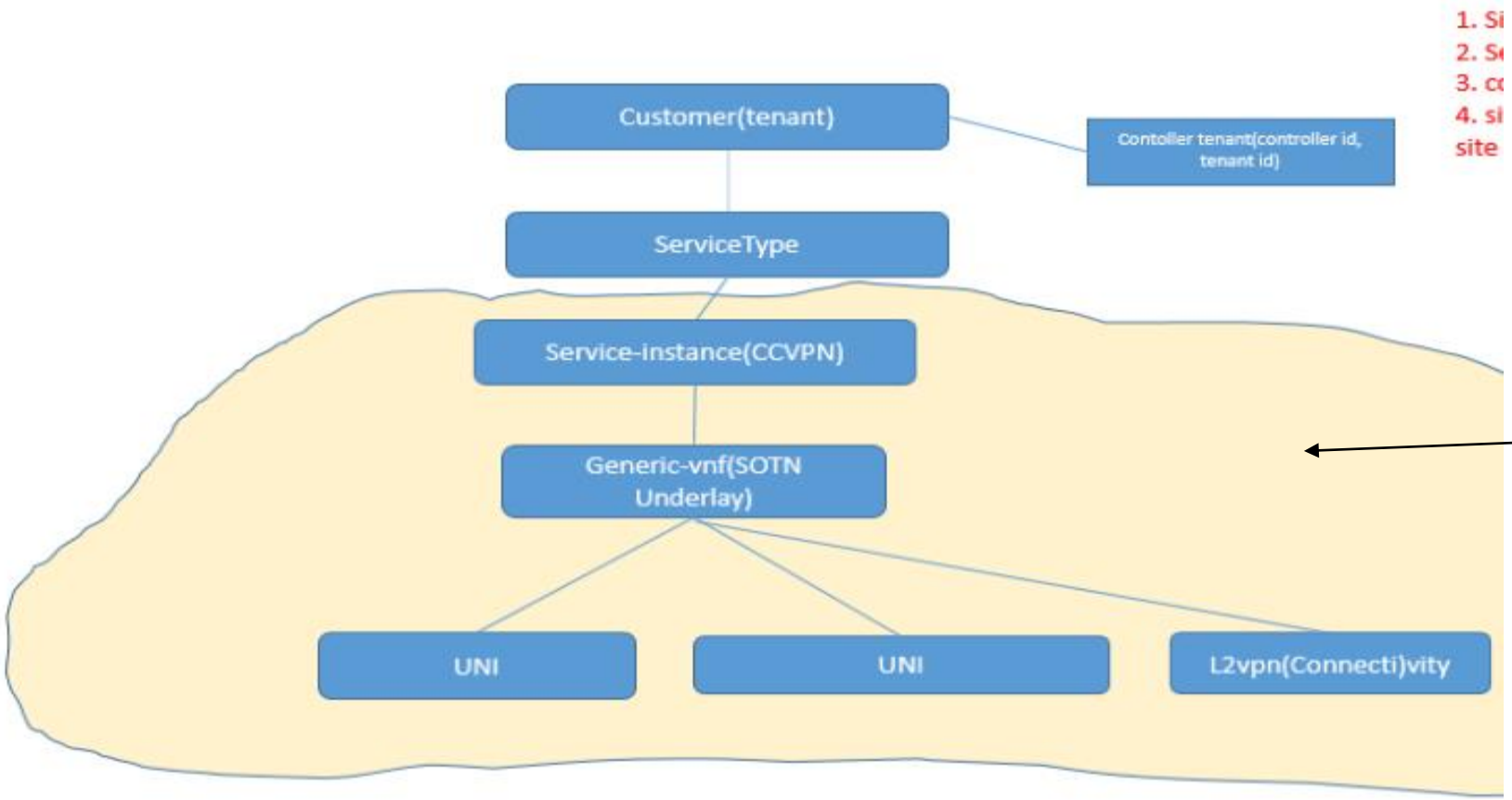
3) SDNC create & activate resource

4) SDNC adds resource to AAI

Development Changes:

- Changes to generic-resource-api.yang for ELINE Service creation
- Changes to GenericResourceApiProvider.java
- Changes to DG for SOTN

SDNC/CCSDK Design: A&AI Service Creation Data Model



Enclosed in the yellow zone are new data modules need to be added to A&AI

SDNC/CCSDK Design: Generic-Resource-API Details (New changes are marked in Red)

https://wiki.onap.org/display/DW/SDNC+APIs?preview=/13599902/25439118/sdnc-Generic-Resource-API-AID-1806-latest.docx

Request (CURD)	URL	request-action
Service	http://x.x.x.x: 8282/restconf/operations/GENERIC-RESOURCE-API:service-topology-operation	NA
Network	http://x.x.x.x: 8282/ restconf/operations/GENERIC-RESOURCE-API:network-topology-operation	<Oper Type>SOTNInfraInstance <Oper Type>SDWANInfraInstance
VNF	http://x.x.x.x:8282/restconf/operations/GENERIC-RESOURCE-API:vnf-topology-operation	<Oper Type>SiteInstance <Oper Type>LANInstance <Oper Type>WANInstance <Oper Type>SDWANDeviceInstance **<Oper Type>SotnUnderlayInstance**
VF-Module	http://x.x.x.x:8282/restconf/operations/GENERIC-RESOURCE-API:vf-module-topology-operation	**<Oper Type>SotnUniInstance** **<Oper Type>SotnL2VpnInstance**
Connection Attachment Allotted Resource	http://x.x.x.x:8282/restconf/operations/GENERIC-RESOURCE-API:connection-attachment-topology-operation	<Oper Type>SOTNAttachmentInstance

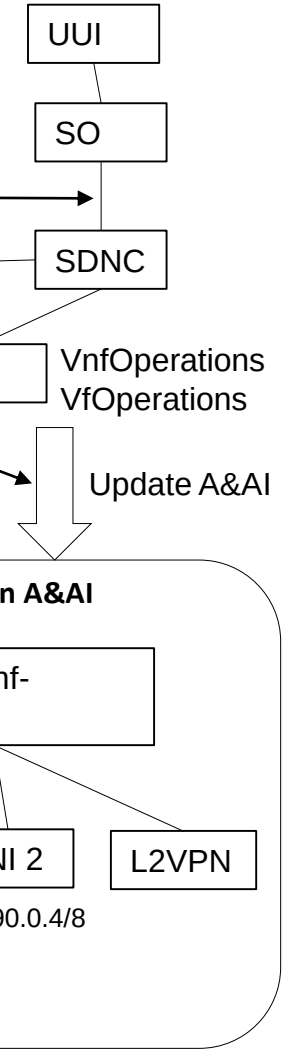
Resource Configuration: Input parameters from the SO

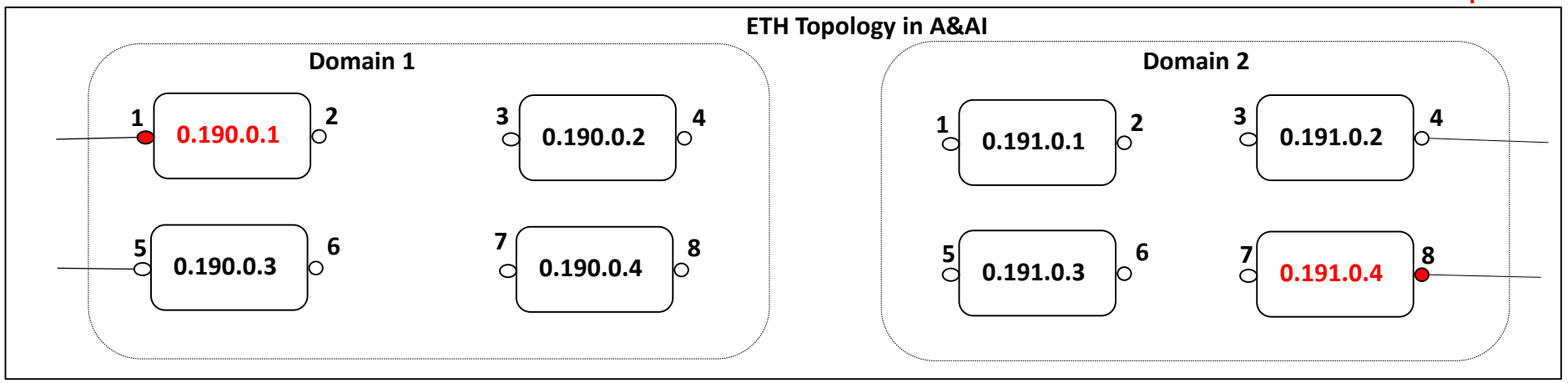
Suppose, for example, the E-LINE to be created is from 0.190.0.1 port 1 to 0.191.0.4 port 8. SO will need to invoke the following API with the input parameters of UNI and L2VpnInstance.

VF-Module	http://x.x.x.x:8282/restconf/operations/GENERIC-RESOURCE-API:vf-module-topology-operation	<Oper Type>SotnUniInstance <Oper Type>SotnL2VpnInstance

API

New enums need to be defined

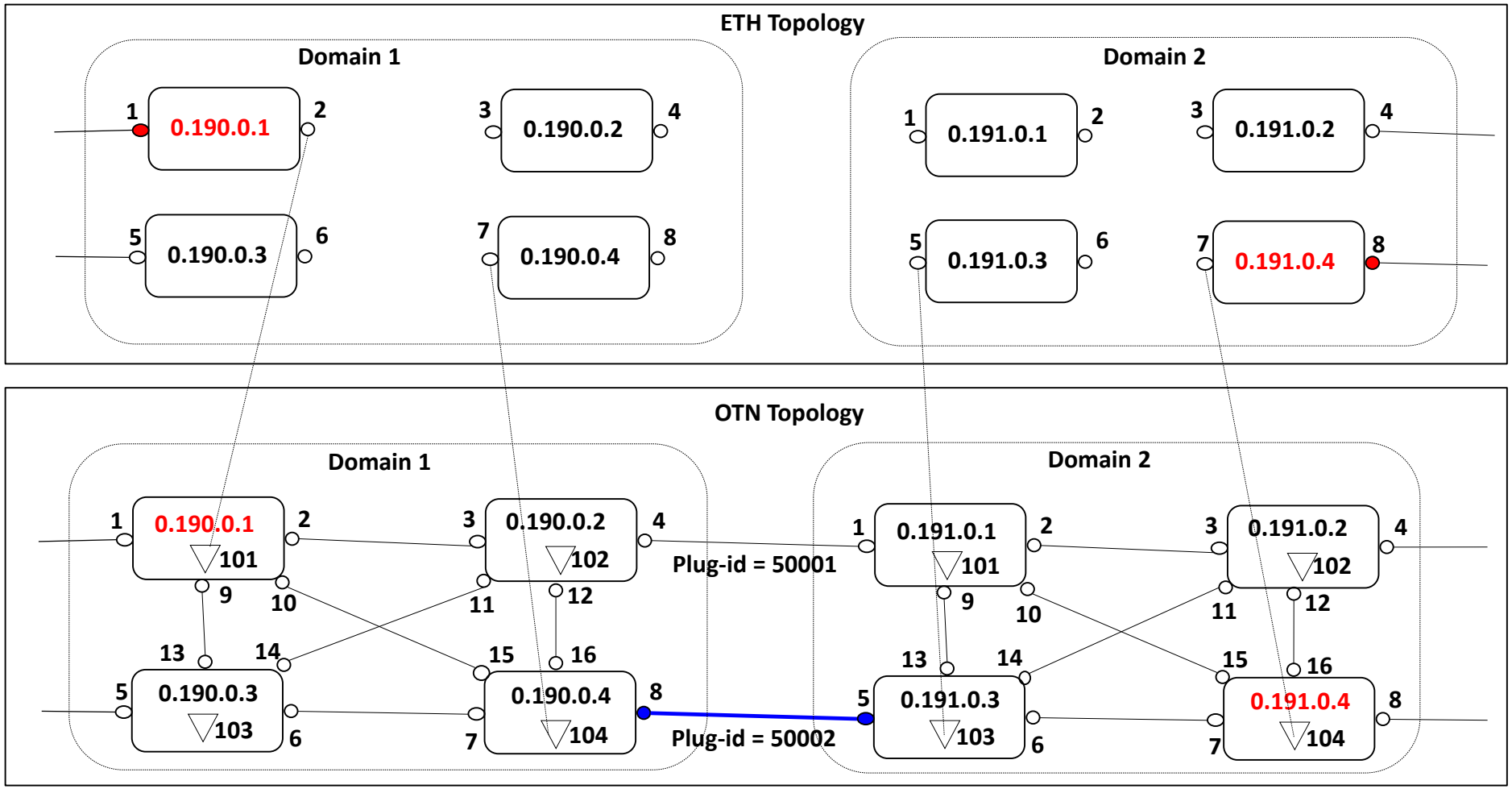


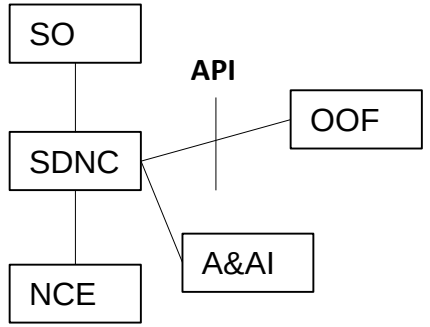


Resource Configuration: OTN tunnel path computation from OOF

Suppose the E-LINE to be created is from 0.190.0.1 port 1 to 0.191.0.4 port 8.

TODO: OOF API ? Input parameters and output parameters.

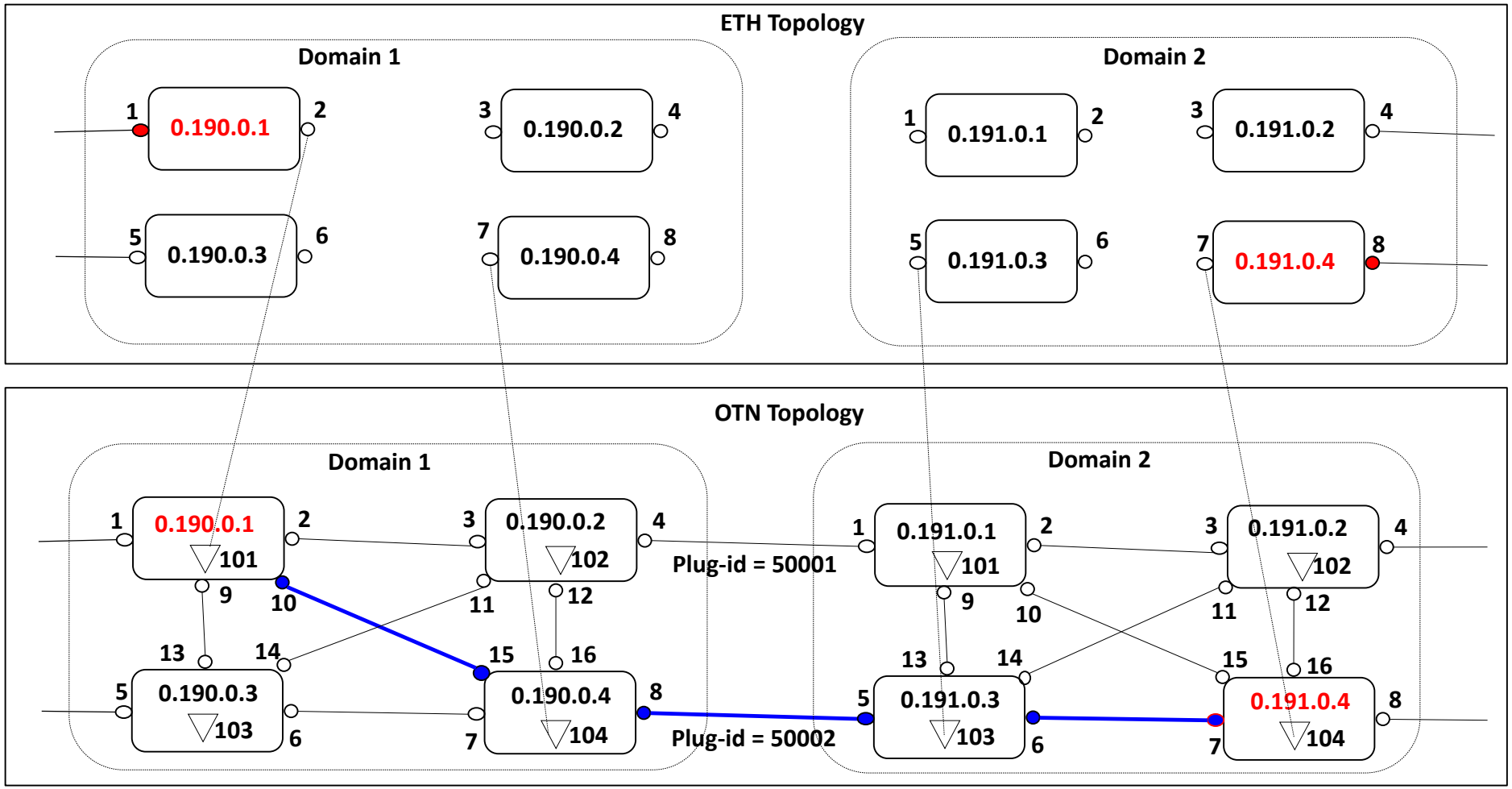




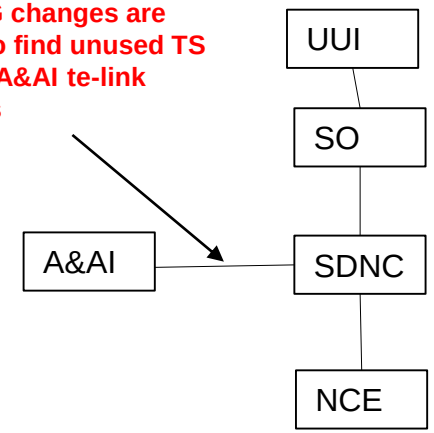
1. SDNC receives resource creation request: source = 0.190.0.1/2; dest = 0.191.0.4/8
2. From source node 0.190.0.1, find the corresponding underlay OTN node (0.190.0.1) using inter-layer-lock-id
3. Similarly, find the destination OTN node (0.191.0.4)
4. Invoke OOF for OTN path computation, with the source OTN node (0.190.0.1) and the dest OTN node (0.191.0.4) as input parameters.
5. OOF returns the inter-domain link (marked in blue in the figure), through which the OTN tunnel is to be established.

Resource Configuration: Inter-domain Tributary Port Setup

Suppose the E-LINE to be created is from 0.190.0.1 port 1 to 0.191.0.4 port 8.



SDNC DG changes are needed to find unused TS from the A&AI te-link attributes

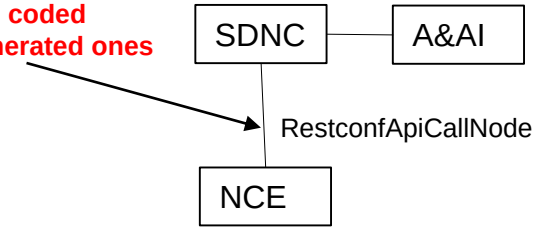


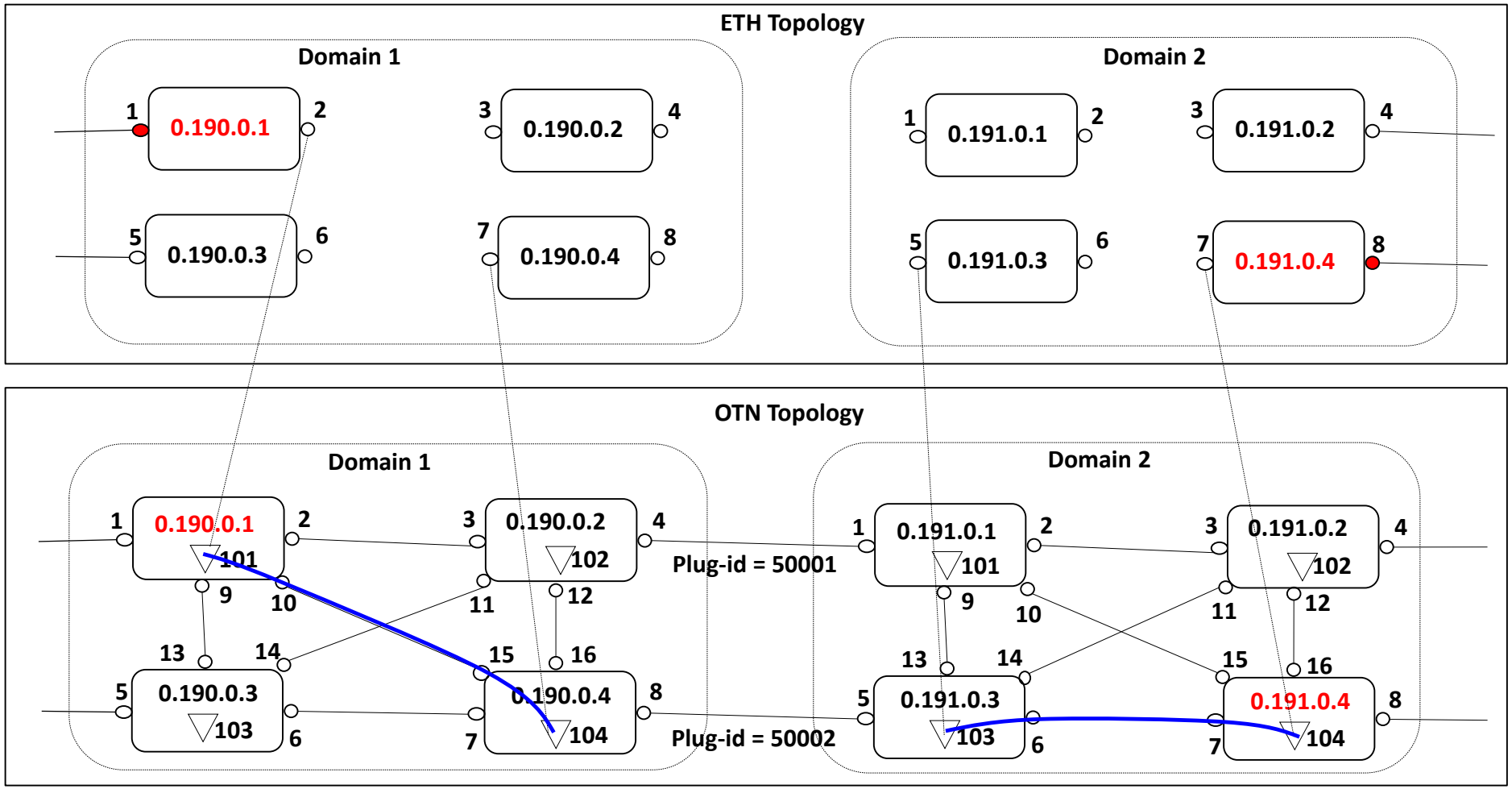
1. From computed OTN tunnel path (marked in blue in the figure), find the inter-domain link, which is 0.190.0.4/8 - 0.191.0.3/5.
2. From the inter-domain link, find the unused ODU resource, i.e., trib port, from the link's label-restriction.

Resource Configuration: OTN Tunnel Parameters Setup

Suppose the E-LINE to be created is from 0.190.0.1 port 1 to 0.191.0.4 port 8.

Parameters are needed for NCE RESTCONF API calls. Some need to be hard coded (template in next page). The generated ones are specified below.





1. OTN tunnel parameters needed by domain 1 controller:

Name	Value
source	0.190.0.1
destination	0.190.0.4
dest-ttp-id	104
dst-tpn	An unused slot # from bitmap

2. OTN tunnel parameters needed by domain 2 controller:

Name	Value
source	0.190.0.3
destination	0.190.0.4
src-ttp-id	103
src-tpn	Same value as dst-tpn in table 1

Service Configuration: Sample RESTCONF SB API

```
{
  "ietf-te:te": {
    "tunnels": {
      "tunnel": [
        {
          "source": "0.191.0.3",
          "destination": "0.191.0.1",
          "dst-ttp-id": "MTc1",
          "ietf-otn-tunnel:dst-tpn": "1",
          "te-bandwidth": {
            "ietf-otn-tunnel:odu-type": "ietf-otn-types:prot-ODUFlex-gfp"
          },
          "encoding": "ietf-te-types:lsp-encoding-oduk",
          "name": "otntunnel-22",
          "restoration": {
            "enable": "false",
            "hold-off-time": "0",
            "restoration-reversion-disable": "true",
            "restoration-type": "ietf-te-types:lsp-restoration-restore-any",
            "wait-to-revert": "0"
          },
          "switching-type": "ietf-te-types:switching-otn",
          "te-topology-identifier": {
            "client-id": "6666",
            "provider-id": "5555",
            "topology-id": "11"
          },
          "provisioning-state": "ietf-te-types:tunnel-state-down"
        },
        {
          "source": "0.191.0.3",
          "destination": "0.191.0.1",
          "encoding": "ietf-te-types:lsp-encoding-packet",
          "name": "tptunnel-22",
          "dependency-tunnels": {
            "dependency-tunnel": [
              {
                "name": "otntunnel-22",
                "encoding": "ietf-te-types:lsp-encoding-oduk",
                "switching-type": "ietf-te-types:switching-otn"
              }
            ]
          },
          "switching-type": "ietf-te-types:switching-psc1",
          "te-topology-identifier": {
            "client-id": "6666",
            "provider-id": "5555",
            "topology-id": "33"
          },
          "provisioning-state": "ietf-te-types:tunnel-state-down"
        }
      ]
    }
  }
}
```

```
{
    "ietf-eth-tran-service:etht-svc":{
        "etht-svc-instances":[
            {
                "etht-svc-name":"Unprotection-Eth-08",
                "etht-svc-descr":"desc-Eth-service",
                "etht-svc-type":"ietf-eth-tran-types:p2p-svc",
                "te-topology-identifier":{
                    "provider-id":"5555",
                    "client-id":"6666",
                    "topology-id":"33"
                },
                "underlay":{
                    "pw":{
                        "pw-name":"pwName",
                        "pw-id":"123",
                        "pw-paths":[
                            {
                                "path-id":"123",
                                "tp-tunnels":[
                                    {
                                        "name":"tptunnel-22"
                                    }
                                ]
                            }
                        ]
                    }
                },
                "resilience":{
                    "protection":{
                        "enable":"true",
                        "hold-off-time":"0",
                        "protection-reversion-disable":"true",
                        "protection-type":"ietf-te-types:lsp-protection-unprotected",
                        "wait-to-revert":"0"
                    }
                },
                "admin-status":"ietf-te-types:tunnel-admin-state-up",
                "etht-svc-end-points":[
                    {
                        "etht-svc-end-point-name":"source-point-name",
                        "etht-svc-access-points":[
                            {
                                "access-point-id":"0",
                                "access-node-id":"0.191.0.3",
                                "access-ltp-id":"16777219"
                            }
                        ],
                        "outer-tag":{
                            "tag-type":"ietf-eth-tran-types:classify-c-vlan",
                            "vlan-value":1
                        },
                        "service-classification-type":"ietf-eth-tran-types:vlan-classification",
                        "ingress-egress-bandwidth-profile":{
                            "bandwidth-profile-type":"ietf-eth-tran-types:mef-10-bwp",
                            "CIR":5000,
                            "EIR":5000
```

Thanks