

How to Disassociate Sites from Cisco Nexus Dashboard Orchestrator

Contents

[Introduction](#)

[Background](#)

[Abbreviations:](#)

[Objective](#)

[Topology](#)

[Disassociate Sites](#)

[Step 1. Disassociate Sites in Templates](#)

[Step 2. Confirm the Objects are not Managed by NDO on each APIC](#)

[Step 3. Remove Empty Templates](#)

[Step 4. Remove Empty Schemas](#)

[Step 5. Disassociate Sites from Tenant](#)

[Step 6. Remove Empty Tenant in NDO](#)

[Step 7. Remove NDO Application in ND](#)

[Step 8. Remove the NDO App in the ND](#)

Introduction

This document describes the procedure to disassociate sites from the Cisco Nexus Dashboard Orchestrator (NDO) and keep them managed locally in APICs.

Background

The goal is to eliminate both ND and NDO.

This procedure is useful when customers are pursuing the decommission of a site and want to keep the configuration that was initially stretched, as local, in the site that continues up.



Warning: Please be advised that this document outlines the steps to disassociate sites from the Cisco Nexus Dashboard Orchestrator (NDO) and maintain local management in APICs. Proceeding with this procedure without proper understanding and caution may result in potential risks or complications. It is recommended to exercise caution and seek expert guidance before making any changes to your network configuration.

Abbreviations:

APIC: Application Policy Infrastructure Controller

ND: Nexus Dashboard

NDO: Nexus Dashboard

VRF: Virtual routing and forwarding

BD: Bridge Domain

EPG: EndPoint Group

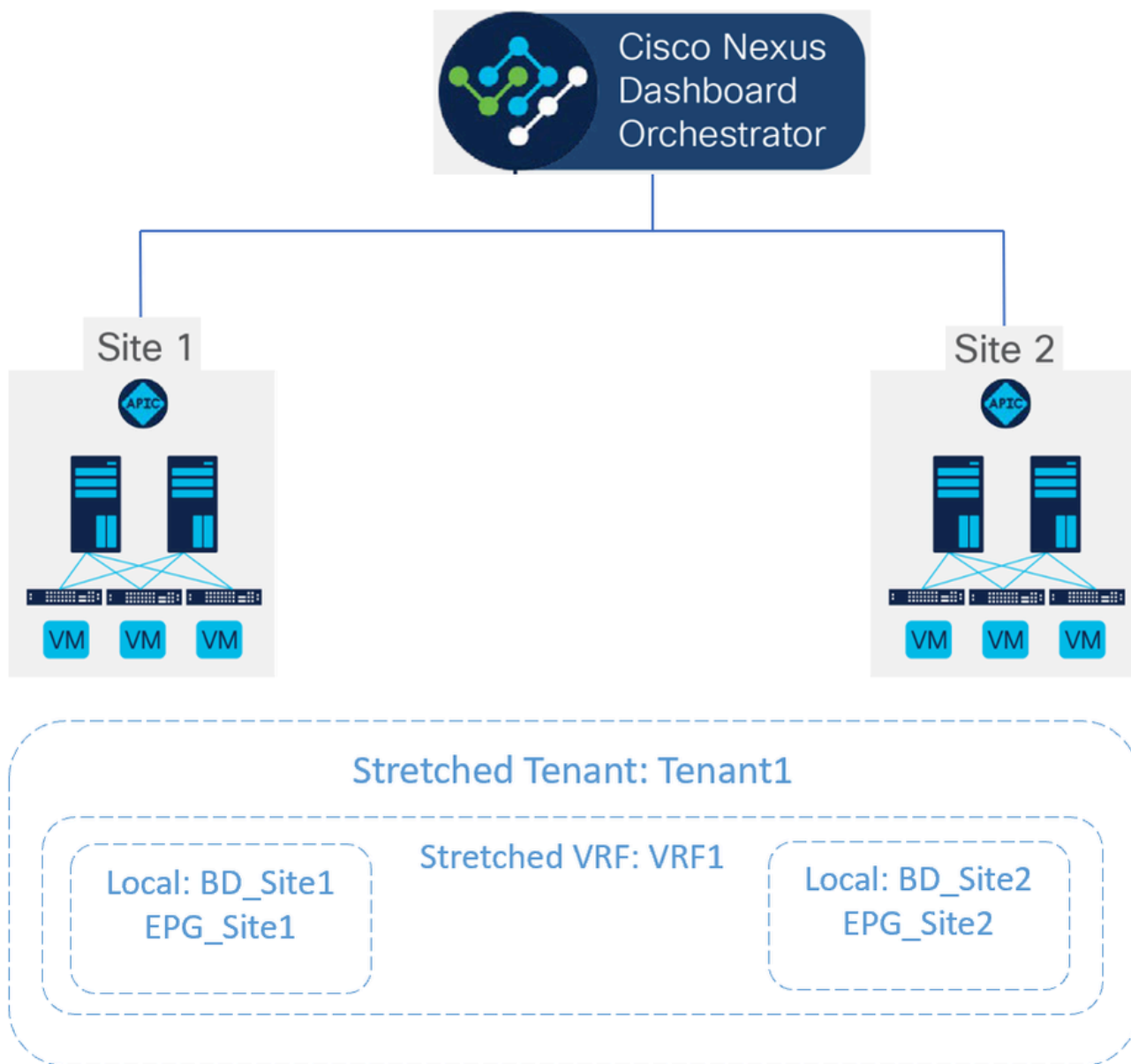
AP: Application Profile

Objective

The purpose of this process is to fully unlink objects managed from NDO and manage them individually from each APIC cluster on every fabric.

Topology

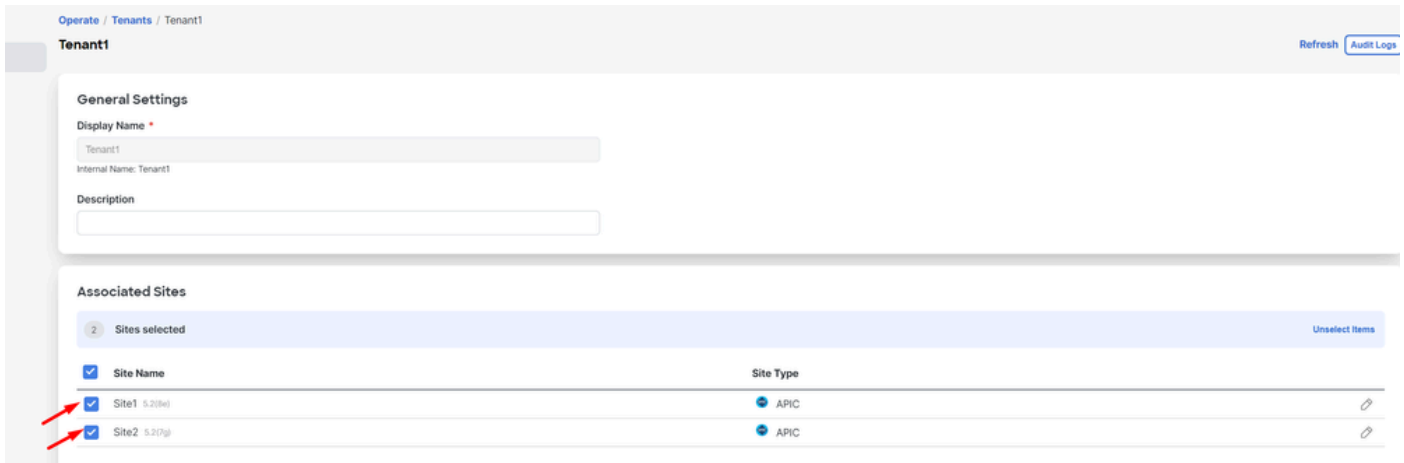
For demonstration purposes, this topology is deployed:



Topology proposed

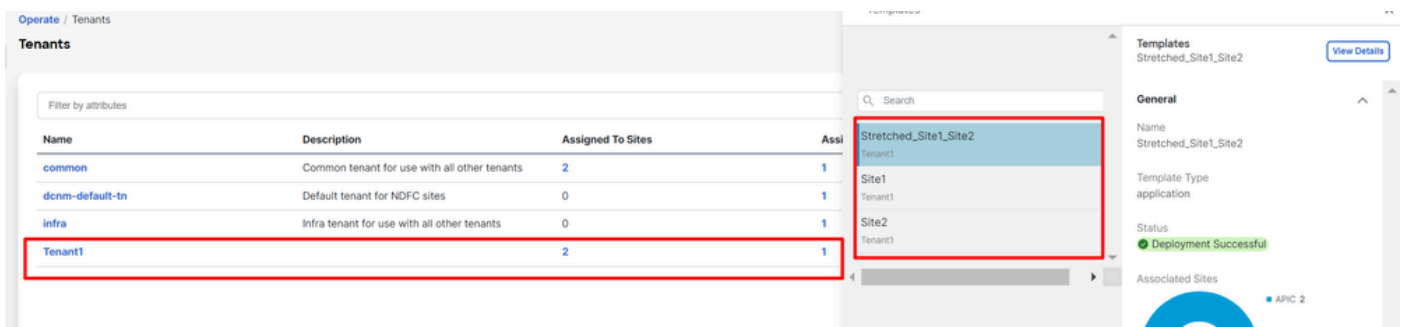
In NDO, the deployment looks like this:

- **Tenant level:** The tenant called Tenant1 is created from NDO, and is associated with both sites, named Site1 and Site2:



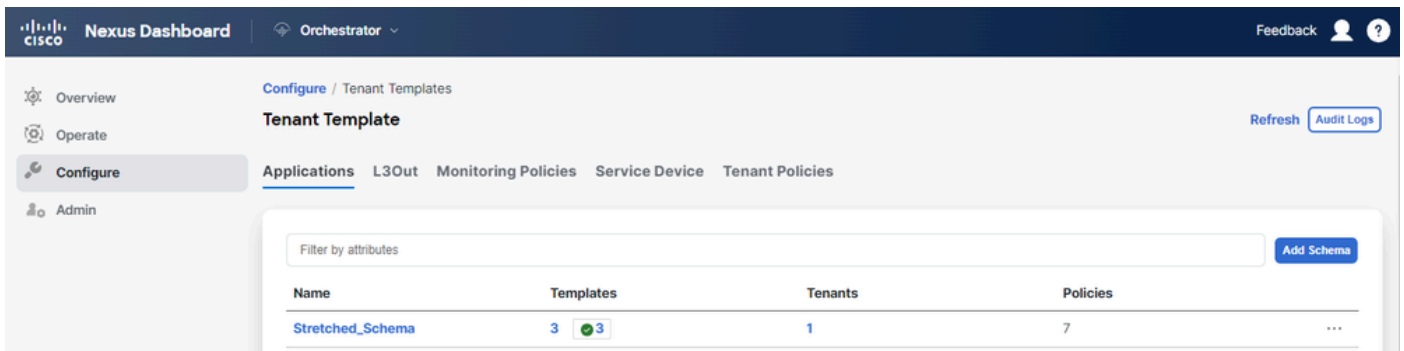
Validation of tenant association with 2 sites

It has been associated with 3 templates:



Validation of template association to a tenant

- **Schema level:** The schema called Schema1 contains the 3 templates:



Validation of templates contained in Stretched_Schema

- **Template level:**
 - Stretched_Site1_Site2 is the stretched template, where the stretched VRF, called VRF1, is defined and is associated with both sites:

Stretched_Schema Refresh Audit Logs Create New Template Save Schema

View **Stretched_Site1_Site2** ▾

Template Properties ● Site1 ● Site2

Template Summary Edit Template Deploy Template Actions ▾

Type	Tenant	Template Status	Associated Sites	Last Action
Application	Tenant1	In Sync	● In Sync 2 ● Out of Sync 0	● Deployment Successful Last Deployed: Oct 11, 2023 02:56 pm

Filter IMPORT ▾ SELECT Create Object ▾

VRFs ▾ Create VRF

Validation that template *Stretched_Site1_Site2* is stretched in 2 sites

- In the template called Site1, associated with only Site1, the local BD_Site1 is defined, and is associated with the stretched VRF1; also, AP_Site1 and EPG_Site1 are locally defined in this template:

Stretched_Schema Refresh Audit Logs Create New Template Save Schema

View **Site1** ▾

Template Properties ● Site1

Template Summary Edit Template Deploy Template Actions ▾

Type	Tenant	Template Status	Associated Sites	Last Action
Application	Tenant1	In Sync	● In Sync 1 ● Out of Sync 0	● Deployment Successful Last Deployed: Oct 11, 2023 08:05 pm

Filter IMPORT ▾ SELECT Create Object ▾

Application Profile AP_Site1 Create Application Profile 🗑️

EPGs ▾ Create EPG

Bridge Domains ▾ Create Bridge Domain

Validation that template *Site1* is local to a single site

BD_Site1

Common Properties

Display Name *

BD_Site1

Deployed Name: BD_Site1

Description

Annotations

Key

Value

[+ Create Annotations](#)

Properties

On-Premise

Reference
Schema - Stretched_Schema
Template - Stretched_Site1_Site2

Virtual Routing & Forwarding  *

VRF1

Validation that VRF for the local BD is the stretched one

- In the template called Site2, associated with only Site2, the local BD_Site2 is defined, and is associated with the stretched VRF1; also, AP_Site2 and EPG_Site2 are locally defined in this template:

Stretched_Schema Refresh Audit Logs Create New Template Save Schema

View Site2 ▼

Template Properties • Site2

Template Summary Edit Template Deploy Template Actions ▼

Type	Tenant	Template Status	Associated Sites	Last Action
Application	Tenant1	In Sync	1 In Sync 1 0 Out of Sync	Deployment Successful Last Deployed: Oct 11, 2023 06:04 pm

Filter IMPORT SELECT Create Object ▼

Application Profile AP_Site2 Create Application Profile 🗑️

EPGs ▼ Create EPG

Bridge Domains ▼ Create Bridge Domain

Validation of template Site 2 to confirm is local

BD_Site2

Common Properties

Display Name *

Deployed Name: BD_Site2

Description

Annotations

Key

Value

[+ Create Annotations](#)

Properties

On-Premise

Reference
Schema - Stretched_Schema
Template - Stretched_Site1_Site2

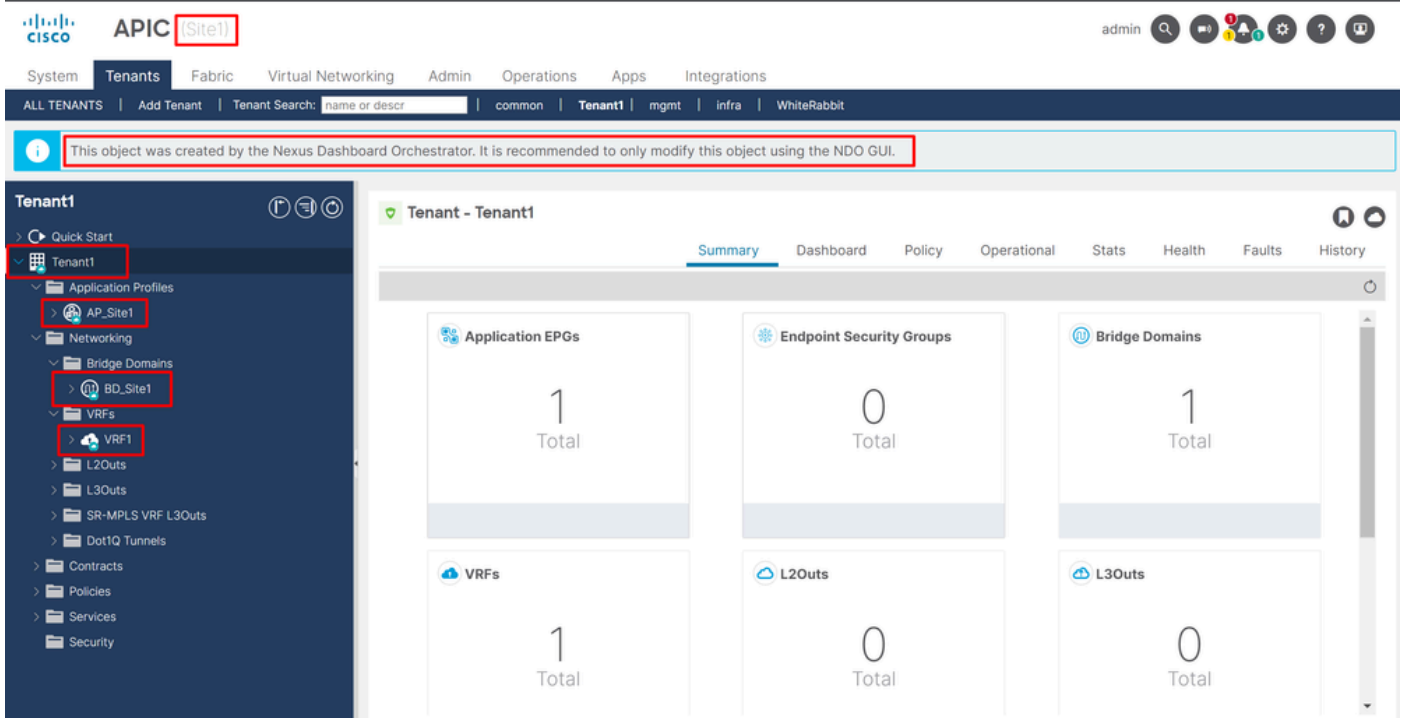
Virtual Routing & Forwarding *

Validation that VRF for the local BD is the stretched one

To confirm the objects are correctly deployed:

- **In Site1:**

Tenant1 is deployed and managed by NDO, as well as the VRF, AP, BD and EPG:



Stretchment validation in GUI

It is possible to confirm as well that all the MIT objects have the annotation set to "orchestrator:misc", meaning are managed from NDO:

Tenant:

```
{
  "totalCount": "1",
  "imdata":
  [
    {
      "fvTenant":
      {
        "attributes":
        {
          "annotation": "orchestrator:misc",
          "descr": "",
          "dn": "uni/tn-Tenant1",
          "name": "Tenant1",
          "nameAlias": "",
          "ownerKey": "",
          "ownerTag": "",
          "userdom": ":all:"
        }
      }
    }
  ]
}
```

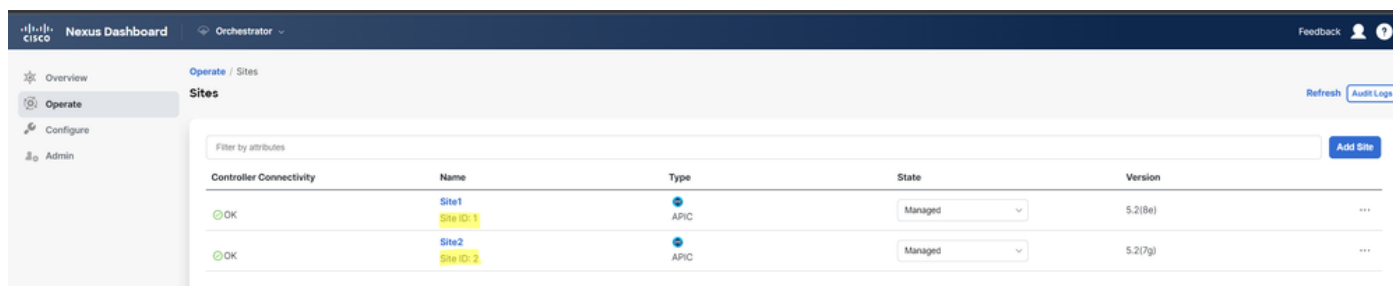
VRF:

"fvCtx":

```
{
  "attributes":
  {
    "annotation": "orchestrator:msc-shadow:no",
    "bdEnforcedEnable": "no",
    "descr": "",
    "ipDataPlaneLearning": "enabled",
    "knwMcastAct": "permit",
    "name": "VRF1",
    "nameAlias": "",
    "ownerKey": "",
    "ownerTag": "",
    "pcEnfDir": "ingress",
    "pcEnfPref": "enforced",
    "userdom": ":all:",
    "vrfIndex": "0"
  },
  "children":
  [
    {
      "fvSiteAssociated":
      {
        "attributes":
        {
          "annotation": "",
          "descr": "",
          "name": "",
          "nameAlias": "",
          "ownerKey": "",
          "ownerTag": "",
          "siteId": "1",
          "userdom": ":all:"
        },
        "children":
        [
          {
            "fvRemoteId":
            {
              "attributes":
              {
                "annotation": "",
                "descr": "",
                "name": "2",
                "nameAlias": "",
                "ownerKey": "",
                "ownerTag": "",
                "remoteCtxPcTag": "32770",
                "remotePcTag": "2686983",
                "siteId": "2",
                "userdom": ":all:"
              }
            }
          }
        ]
      }
    },
  ]
}
```

For the VRF, it can be seen that besides the "orchestrator:misc" annotation, some children properties are also seen.

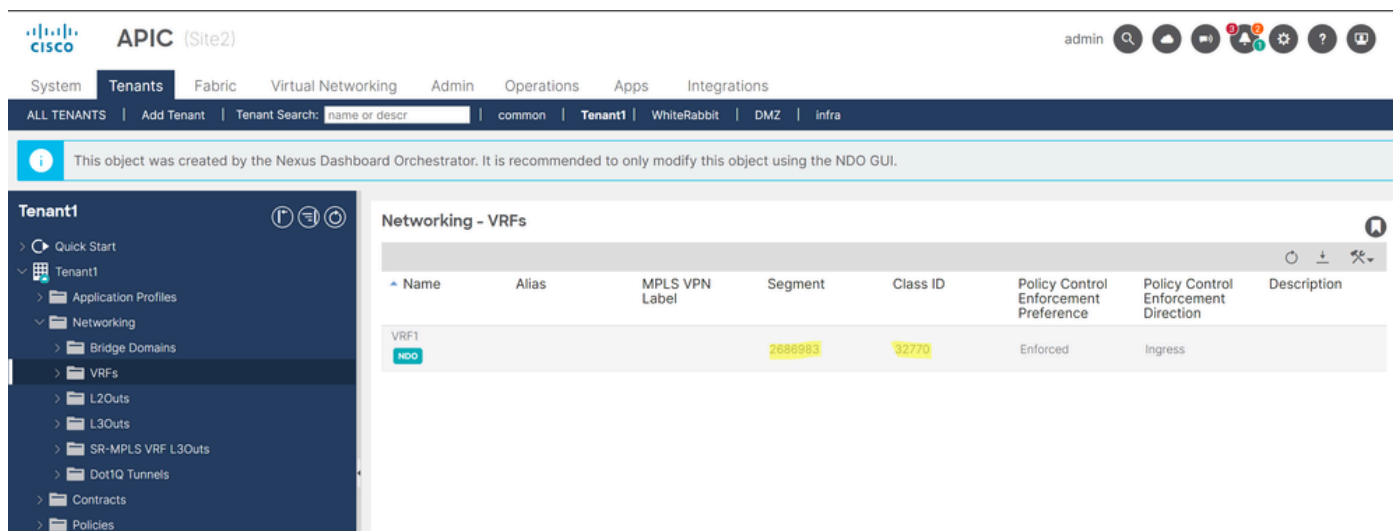
To understand better these children objects, it is important to notice that in NDO, besides the Site name, a unique site ID is associated with each site in NDO. To query the IDs, in NDO, navigate to Operate > Sites :



Validation of SiteID per site in NDO

Once this information is explained, the children objects are:

1. fvSiteAssociated: Shows the Site-ID of the local Site.
2. fvRemoteID: The remote Site IDs where the object is stretched too. This object is also useful to know the translation of objects across sites; in the case of this VRF, the segment, and the ClassID can be seen, corresponding to Site 2. To confirm, a comparison can be done from Site 2:



Validation of Segment and ClassID of remote objects

As can be seen, the Segment and ClassID from Site 2, are contained in the fvRemoteID inside the VRF object in Site 1.

BD:

```
"fvBD":
{
  "attributes":
  {
    "OptimizeWanBandwidth": "yes",
    "annotation": "orchestrator:misc-shadow:no",
    "name": "BD_Site1",
    ...
  }
}
```

```

},
"children":
[
  ...
  {
    "fvSiteAssociated":
    {
      "attributes":
      {
        "annotation": "",
        "descr": "",
        "name": "msc-local",
        "nameAlias": "",
        "ownerKey": "",
        "ownerTag": "",
        "siteId": "1",
        "userdom": ":all:"
      }
    }
  },
]
}

```

AP and EPG:

"fvAp":

```

{
  "attributes":
  {
    "annotation": "orchestrator:msc-shadow:no",
    "descr": "",
    "name": "APP_Site1",
    "nameAlias": "",
    "ownerKey": "",
    "ownerTag": "",
    "prio": "unspecified",
    "userdom": ":all:"
  },
  "children":
  [
    {
      "fvAEPg":
      {
        "attributes":
        {
          "annotation": "orchestrator:msc-shadow:no",
          "descr": "",
          "exceptionTag": "",
          "floodOnEncap": "disabled",
          "fwdCtrl": "",
          "hasMcastSource": "no",
          "isAttrBasedEPg": "no",
          "matchT": "None",
          "name": "EPG_Site1",
          "nameAlias": "",
          "pcEnfPref": "unenforced",
          "prefGrMemb": "exclude",

```

```

        "prio": "unspecified",
        "shutdown": "no",
        "userdom": ":all:"
    },
    "children":
    [
        {
            "fvSiteAssociated":
            {
                "attributes":
                {
                    "annotation": "",
                    "descr": "",
                    "name": "msc-local",
                    "nameAlias": "",
                    "ownerKey": "",
                    "ownerTag": "",
                    "siteId": "1",
                    "userdom": ":all:"
                }
            }
        },
    ]
}

```

In the BD, AP, and EPG objects, there are no fvRemoteId children objects, since these objects are locally significant, and are not stretched.

- **In Site 2:**

Site 2 has pretty similar outputs, only changing the corresponding remote objects, so this information is omitted.

Disassociate Sites

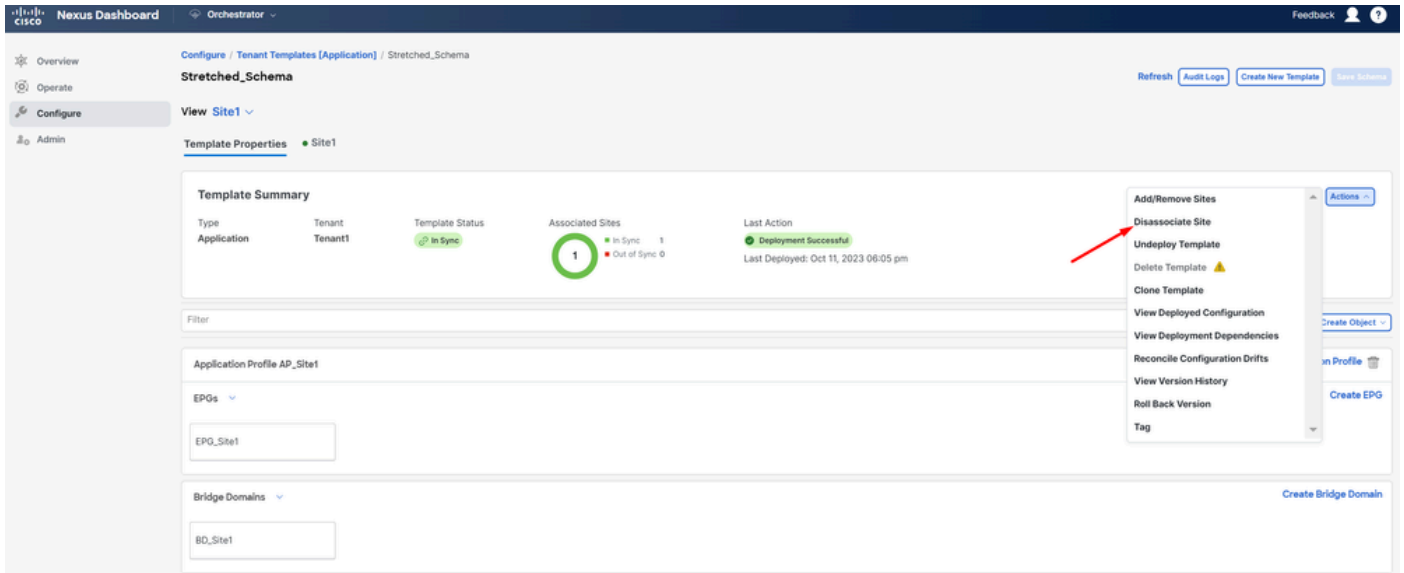
It is recommended to take a backup in NDO, as well as a snapshot in the APIC before doing this procedure, in case it is desired to roll this back later.

Step 1. Disassociate Sites in Templates

This step needs to be run on each template. Similarly to the logic behind the circle dependencies, it is needed to start first on templates that have dependencies on other templates, and finally, disassociate the templates that do not have any cross reference.

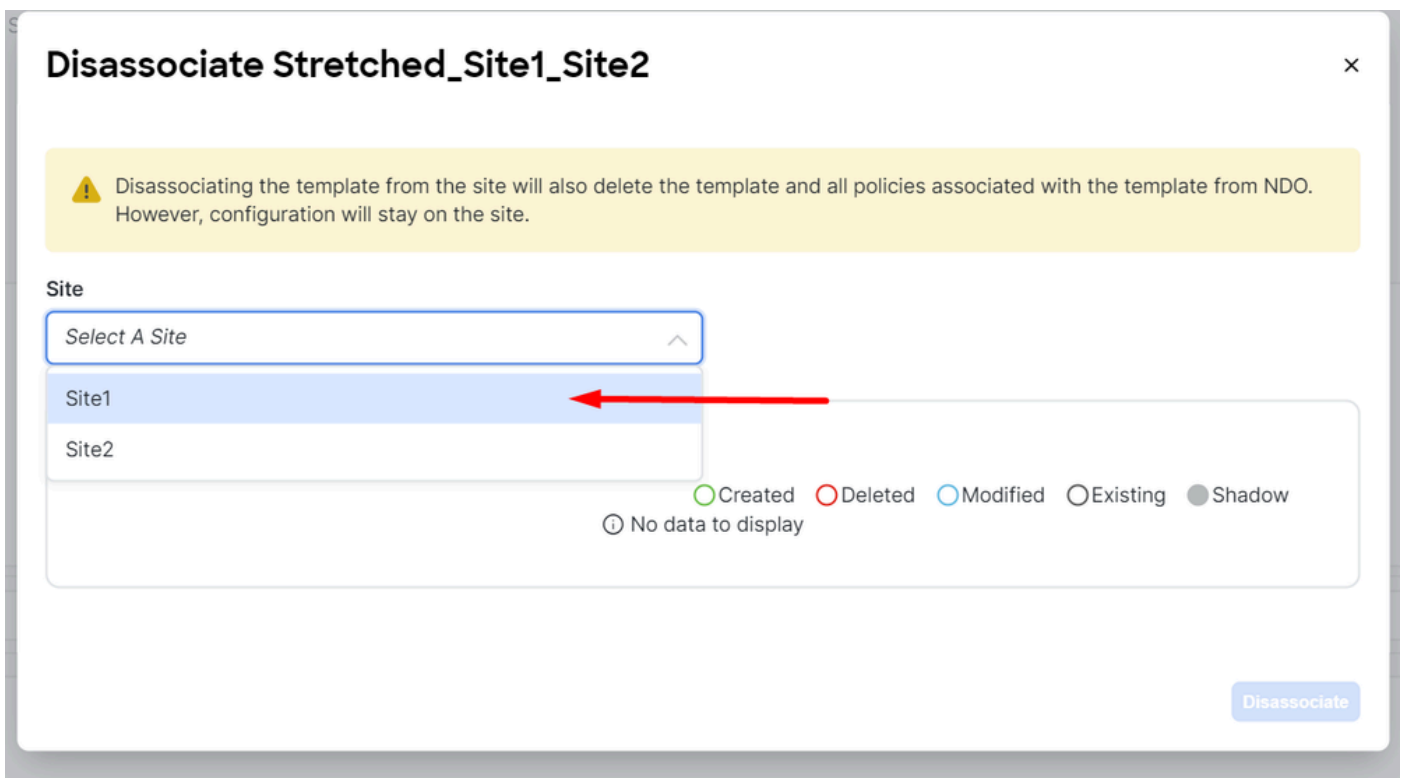
In the topology used in this document, the last template to be disassociated must be the Stretched_Site1_Site2, this is because templates Site1 and Site2 have a reference to it.

Navigate to the template inside the schema, click on **Actions** , and navigate to **Disassociate Site:**



How to disassociate template

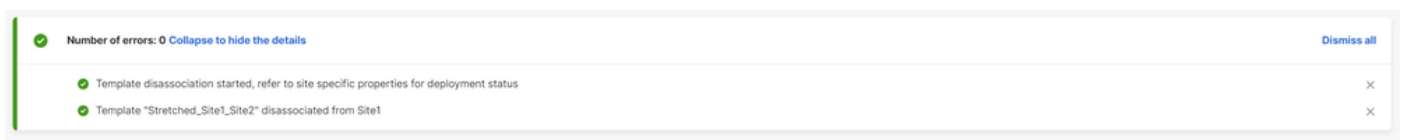
In the next window, choose from the drop-down menu site by site, since the disassociation is done one by one (in case the template has more than 2 sites):

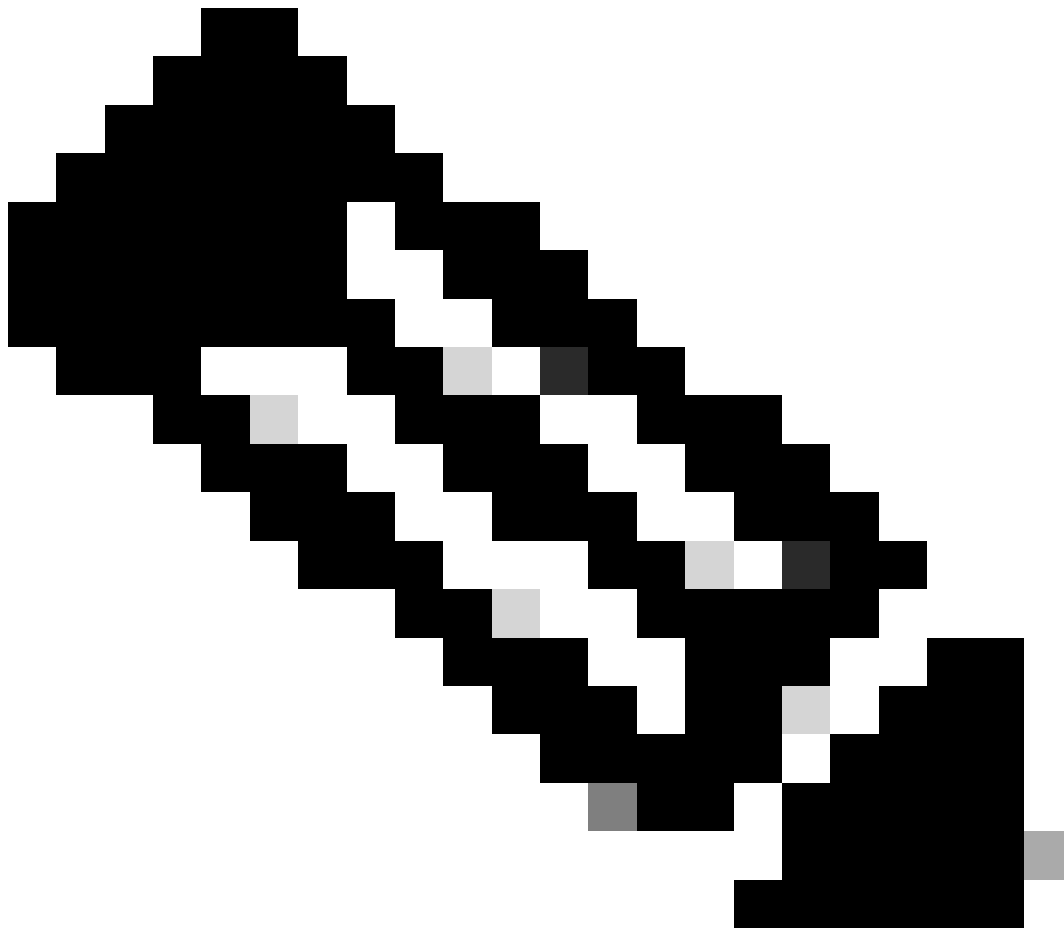


Selection of site from where to disassociate template

Then click on **Disassociate**.

A message with the confirmation is displayed once it finishes:



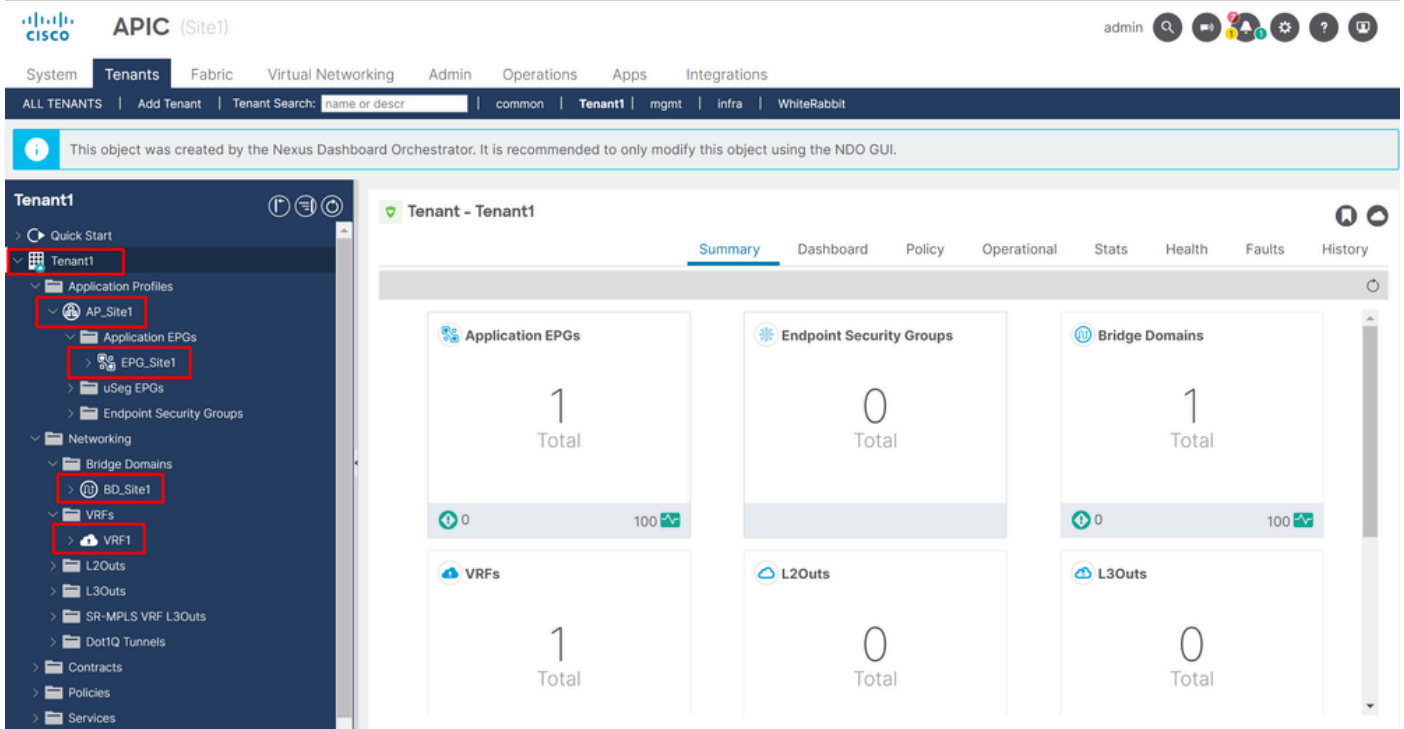


Note: As mentioned before, repeat this procedure for all templates on the schema.

Step 2. Confirm the Objects are not Managed by NDO on each APIC

To confirm the objects are still present in the APICs, now with different properties:

In APIC (example in Site 1):



GUI validation that configuration persists.

Objects do not show the Cloud NDO icon next to it anymore, only the Tenant is still managed by NDO.

In JSON:

```
"fvTenant":
  {
    "attributes":
      {
        "annotation": "orchestrator:msc",
        "descr": "",
        "dn": "uni/tn-Tenant1",
        "name": "Tenant1",
        "nameAlias": "",
        "ownerKey": "",
        "ownerTag": "",
        "userdom": ":all:"
      },
    "children":
      [
        {
          "fvCtx":
            {
              "attributes":
                {
                  "annotation": "",
                  "bdEnforcedEnable": "no",
                  "descr": "",
                  "ipDataPlaneLearning": "enabled",
                  "knwMcastAct": "permit",
                  "name": "VRF1",
                  "nameAlias": "",
                  "ownerKey": "",
                  "ownerTag": "",
                  "pcEnfDir": "ingress",
```



```

        "pcEnfPref": "enforced",
        "userdom": ":all:",
        "vrfIndex": "0"
    },
    "fvBD":
    {
        "attributes":
        {
            "OptimizeWanBandwidth": "yes",
            "annotation": "",
            "arpFlood": "yes",
            "descr": "",
            "epClear": "no",
            "epMoveDetectMode": "",
            "hostBasedRouting": "no",
            "intersiteBumTrafficAllow": "yes",
            "intersiteL2Stretch": "yes",
            "ipLearning": "yes",
            "ipv6McastAllow": "no",
            "limitIpLearnToSubnets": "yes",
            "llAddr": "::",
            "mac": "00:22:BD:F8:19:FF",
            "mcastARPDrop": "yes",
            "mcastAllow": "no",
            "multiDstPktAct": "bd-flood",
            "name": "BD_Site1",
            "nameAlias": "",
            "ownerKey": "",
            "ownerTag": "",
            "type": "regular",
            "unicastRoute": "yes",
            "unkMacUcastAct": "proxy",
            "unkMcastAct": "flood",
            "userdom": ":all:",
            "v6unkMcastAct": "flood",
            "vmac": "not-applicable"
        }
    }

```

...

```

    "fvAp":
    {
        "attributes":
        {
            "annotation": "",
            "descr": "",
            "name": "APP_Site1",
            "nameAlias": "",
            "ownerKey": "",
            "ownerTag": "",
            "prio": "unspecified",
            "userdom": ":all:"
        },
        "children":
        [
            {
                "fvAEPg":
                {
                    "attributes":
                    {
                        "annotation": "",

```


Stretched_Schema

View **Stretched_Site1_Site2** ^

Temp Overview

Stretched_Site1_Site2 Unassociated ✓

Site1 Unassociated

Site2 Unassociated

Validation of templates in an unassociated state

These templates can be safely removed. To remove them, click on **Actions** and select **Delete Template** as shown in the image:

The screenshot shows the 'Stretched_Schema' interface. At the top right, there are buttons for 'Refresh', 'Audit Logs', 'Create New Template', and 'Save Schema'. Below this, the 'View Stretched_Site1_Site2' dropdown is visible. The 'Template Properties' section is expanded to show 'Template Summary'. This summary includes: Type: Application, Tenant: Tenant1, Template Status: Unassociated, Associated Sites: 0 (0 In Sync, 0 Out of Sync), and Last Action: Updated. On the right side, an 'Actions' menu is open, listing options such as 'Add/Remove Sites', 'Disassociate Site', 'Undeploy Template', 'Delete Template' (highlighted with a red arrow), 'Clone Template', 'View Deployed Configuration', 'View Deployment Dependencies', 'Reconcile Configuration Drifts', 'View Version History', 'Roll Back Version', and 'Tag'. There are also 'Create Object' and 'Create VRF' buttons on the right.

Deletion of template

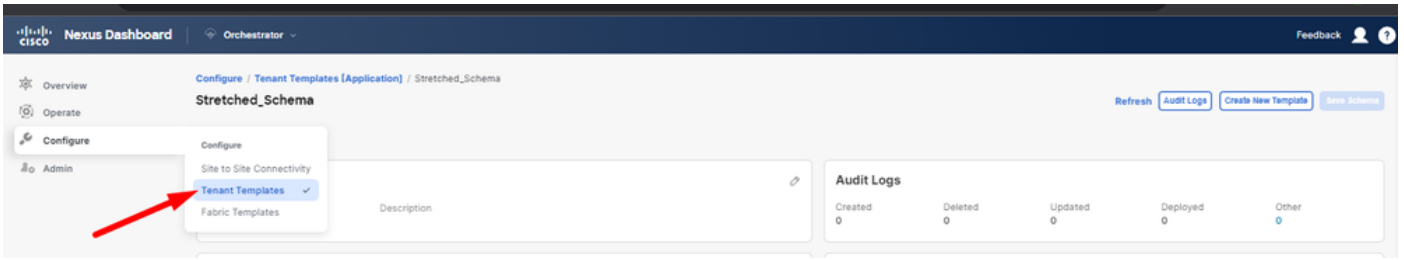
Once the schema is empty, save the changes:

The screenshot shows the 'Stretched_Schema' interface after the template has been removed. The breadcrumb path is 'Configure / Tenant Templates [Application] / Stretched_Schema'. The 'View Overview' dropdown is selected. The 'General' tab is active, showing the Name 'Stretched_Schema'. The 'Audit Logs' table is visible, with columns for Created, Deleted, Updated, Deployed, and Other. The 'Deleted' column shows 0. At the top right, the 'Save Schema' button is highlighted with a red arrow.

Save changes on the empty schema

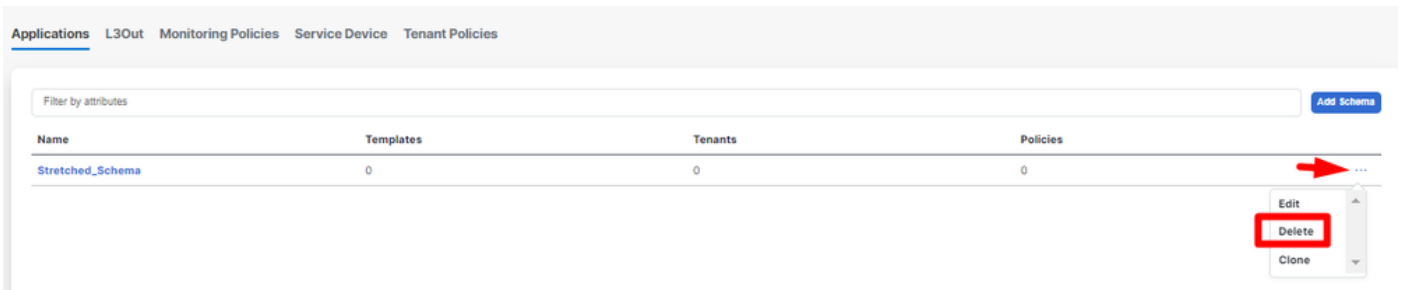
Step 4. Remove Empty Schemas

It is time to remove the empty schema. Navigate to `Configure > Tenant Templates` as shown in the image:



Steps to move to tenant menu

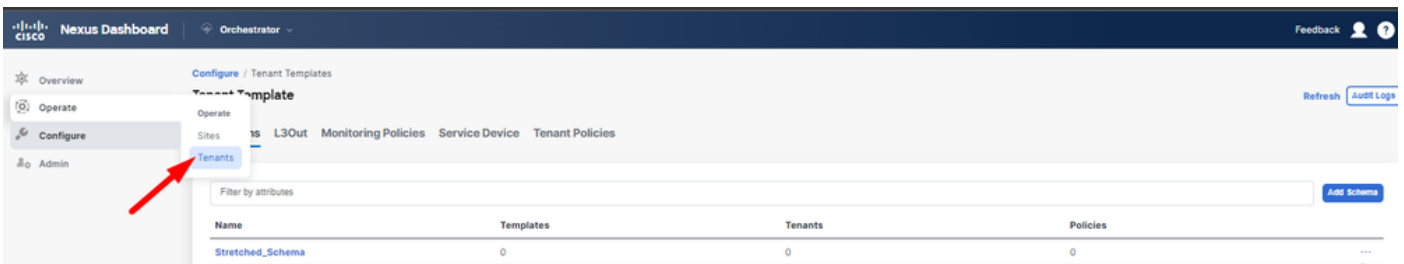
And click on the 3 dots next to the schema, and click on `Delete` as shown in the image:



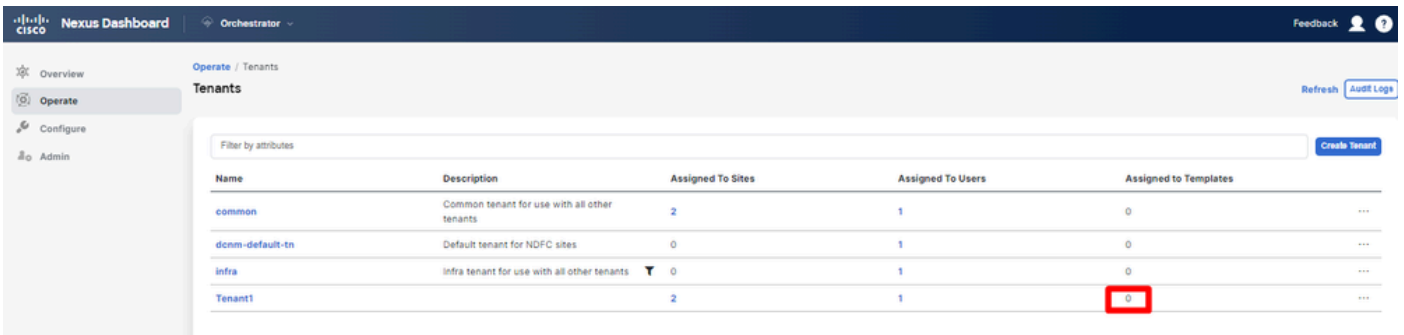
Delete empty schema associated with the template

Step 5. Disassociate Sites from Tenant

Once there are no more schemas, the tenant must show it is no longer associated with any template. To confirm, navigate to `Operate > Tenants`:



Disassociate sites from Tenant



Confirming the Tenant has no templates associated

As can be seen, the number of templates associated with Tenant1 is 0. Click on the 3 dots, and select **Edit**:

Name	Description	Assigned To Sites	Assigned To Users	Assigned to Templates	
common	Common tenant for use with all other tenants	2	1	0	...
dcnm-default-tn	Default tenant for NDFC sites	0	1	0	...
infra	Infra tenant for use with all other tenants	0	1	0	...
Tenant1		2	1	0	...

Edit tenant properties to remove sites

Now, it is needed to unselect the sites. Click on **Unselect items** at the top of the table of sites:

General Settings

Display Name: Tenant1
Internal Name: Tenant1

Description:

Associated Sites

2 Sites selected

Site Name	Site Type	
Site1 5.218e	APIC	
Site2 5.217g	APIC	

Unselect Sites associated with the tenant

Ensure the option to delete the tenant is unchecked before confirming:



Warning

Are you sure you want to disassociate all sites

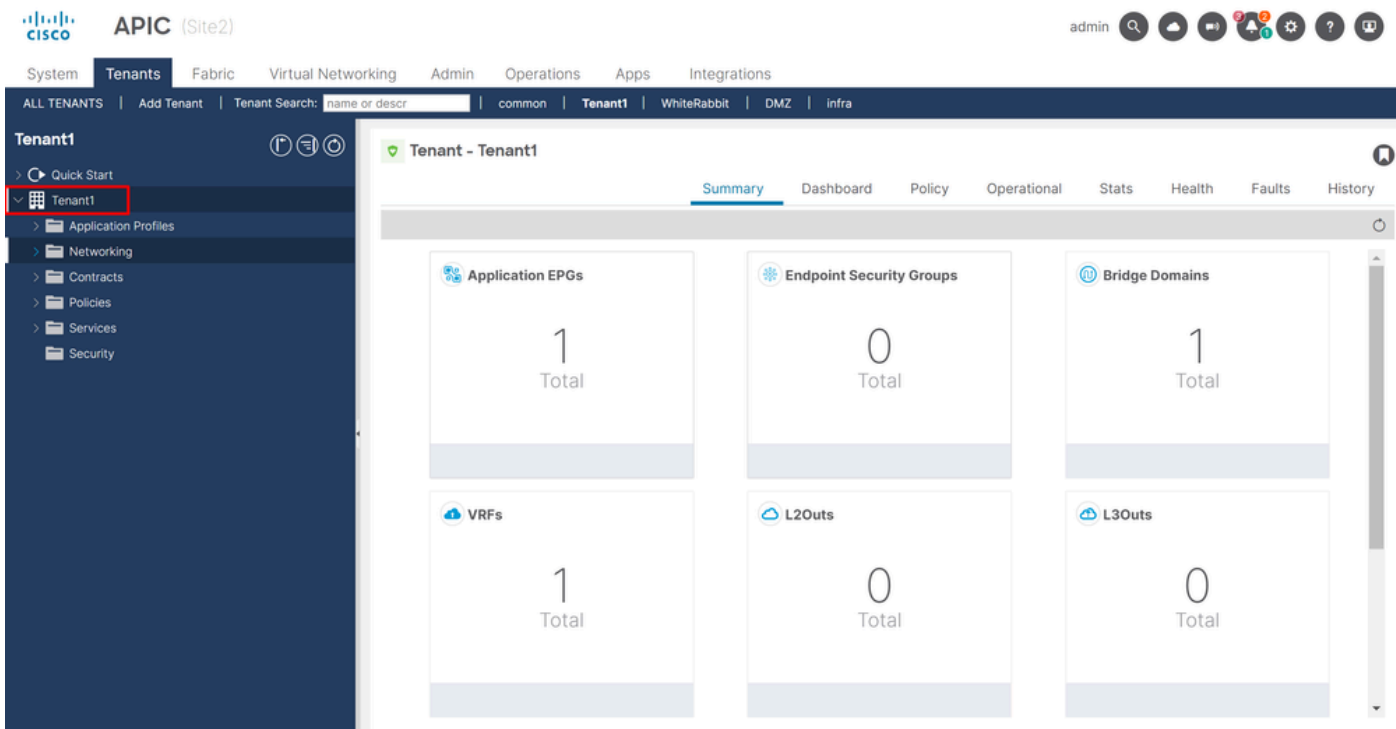
Also delete Tenant1 from all sites

Cancel

Continue

Confirm the operation without the check

When both sites are unchecked, save the changes. Once this is done, confirm the Tenant in each APIC stays in there:



IGUI validation that the tenant is still configured, but not managed from NDO

As expected, now the annotation is empty:

```

"fvTenant":
  {
    "attributes":
      {
        "annotation": "",
        "descr": "",
        "dn": "uni/tn-Tenant1",
        "name": "Tenant1",
        "nameAlias": "",
        "ownerKey": "",
        "ownerTag": "",
        "userdom": ":all:"
      }
  }

```

Step 6. Remove Empty Tenant in NDO

It is time to remove the Tenant. To do so, navigate to Operate > Tenants , click on the 3 dots, and click on Delete as shown in the image:

Operate / Tenants

Tenants Refresh Audit Logs

Filter by attributes Create Tenant

Name	Description	Assigned To Sites	Assigned To Users	Assigned to Templates	
common	Common tenant for use with all other tenants	2	1	0	...
dcnm-default-tn	Default tenant for NDFC sites	0	1	0	...
infra	Infra tenant for use with all other tenants	0	1	0	...
Tenant1		0	1	0	...

Edit
Delete

Delete empty tenant

Confirm, and verify the Tenant Object stays in the APICs.

Step 7. Remove NDO Application in ND

To remove NDO, the app needs to be disabled first.

in ND, navigate to Admin Console > Services. The NDO application is displayed there. Click on the 3 dots and select Disable:

Disable NDO application

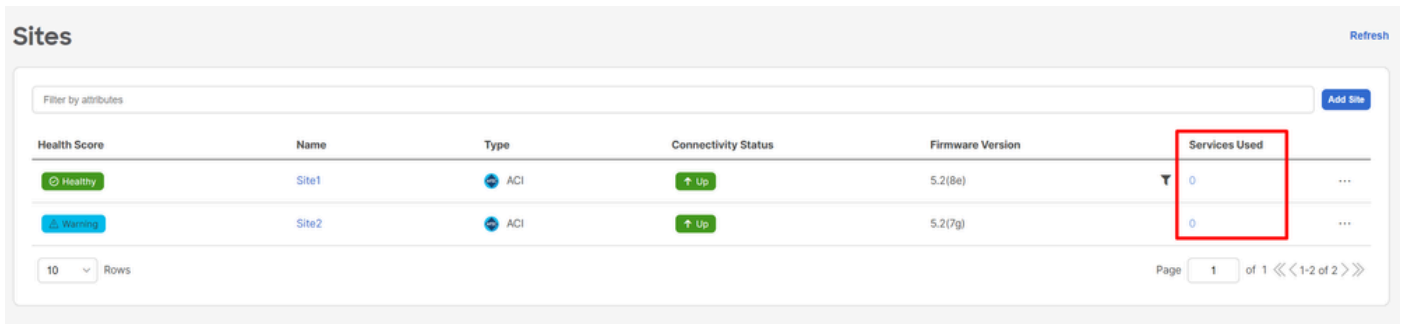
It can take a couple of minutes to be completely disabled.

Then, click on the 3 dots again, and this time click on the option Delete.

Step 8. Remove the NDO App in the ND

Finally, from ND, remove the Sites. To be able to remove the sites, they must not be consuming any

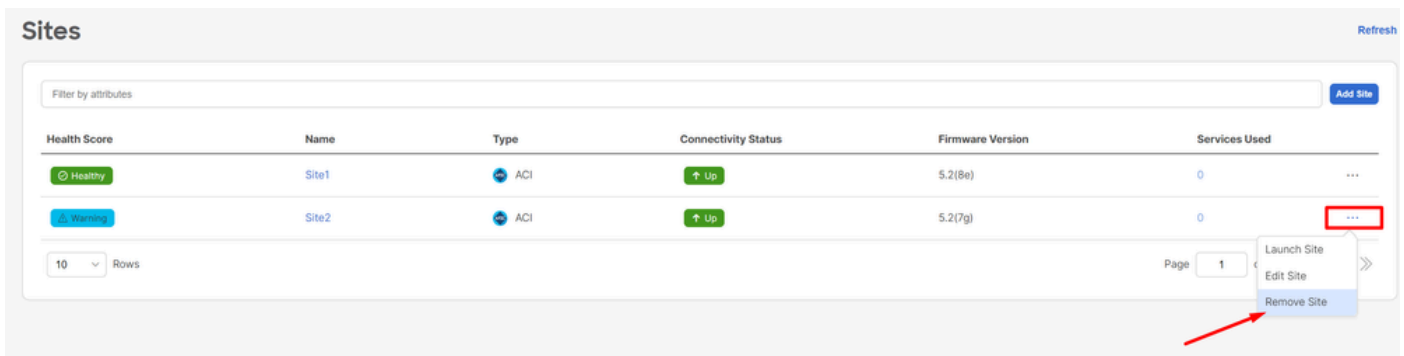
services, so, if any other application is installed, it needs to be removed too:



Health Score	Name	Type	Connectivity Status	Firmware Version	Services Used
Healthy	Site1	ACI	Up	5.2(8e)	0
Warning	Site2	ACI	Up	5.2(7g)	0

Validation that sites do not use the NDO service

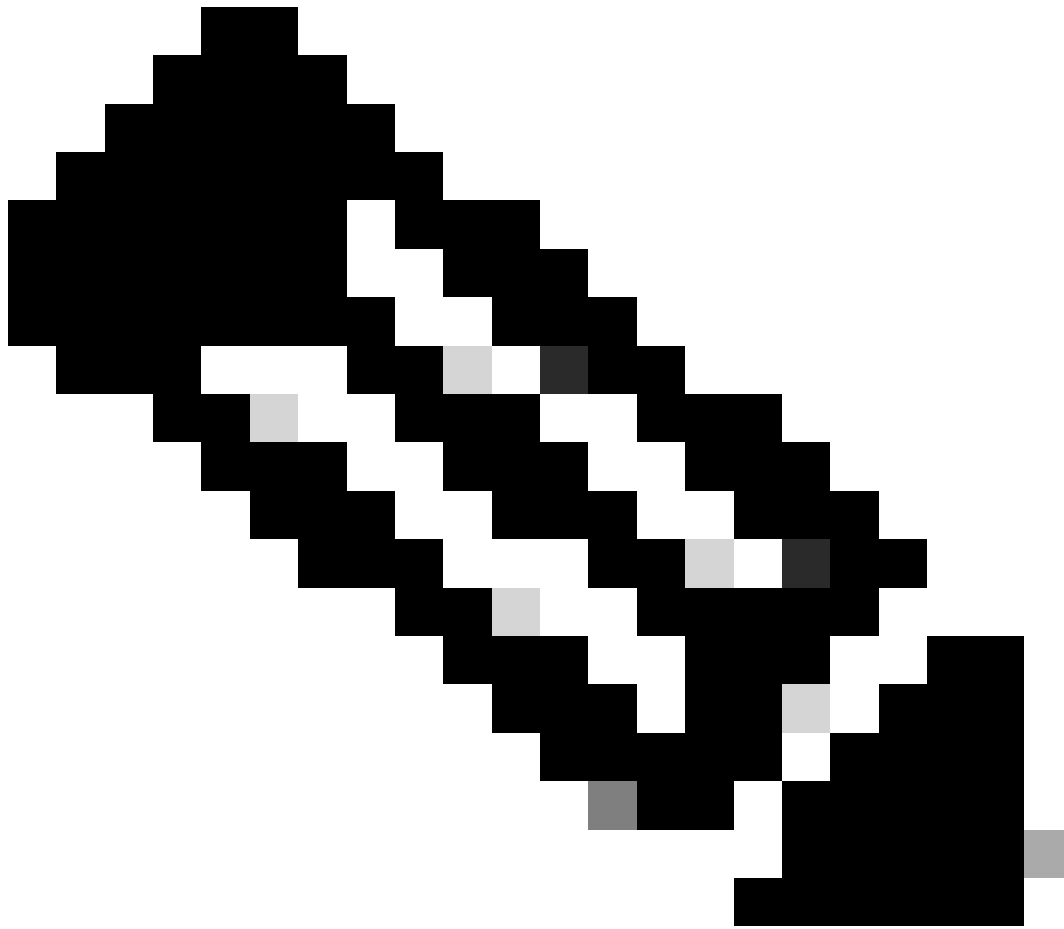
To remove it, click on the 3 dots, and choose Remove Site as shown in the image:



Health Score	Name	Type	Connectivity Status	Firmware Version	Services Used
Healthy	Site1	ACI	Up	5.2(8e)	0
Warning	Site2	ACI	Up	5.2(7g)	0

Remove Sites in ND

Once the sites are completely removed, each fabric is independent now and ND can also be retired.



Note: Once the sites are independent, the L3out for intersite in the infra tenant is still present. It can be manually removed, make sure is only for intersite connectivity.
