# Cisco ACI Installation Guide for Red Hat OpenStack Using OpenStack Platform 10 Director

**First Published:** 2017-09-15

**Last Modified:** 2019-10-01

**Americas Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

# CONTENTS

# Preface

This preface includes the following sections:



## Audience

This guide is intended primarily for data center administrators with responsibilities and expertise in one or more of the following:

- Virtual machine installation and administration
- Server administration
- Switch and network administration
- Cloud administration

## Document Conventions

Command descriptions use the following conventions:

| Convention | Description |
|---|---|
| **bold** | Bold text indicates the commands and keywords that you enter literally as shown. |
| *Italic* | Italic text indicates arguments for which the user supplies the values. |
| [x] | Square brackets enclose an optional element (keyword or argument). |
| [x \| y] | Square brackets enclosing keywords or arguments separated by a vertical bar indicate an optional choice. |

| Convention | Description |
|---|---|
| {x \| y} | Braces enclosing keywords or arguments separated by a vertical bar indicate a required choice. |
| [x {y \| z}] | Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element. |
| `variable` | Indicates a variable for which you supply values, in context where italics cannot be used. |
| string | A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks. |

Examples use the following conventions:

| Convention | Description |
|---|---|
| `screen font` | Terminal sessions and information the switch displays are in screen font. |
| **`boldface screen font`** | Information you must enter is in boldface screen font. |
| *italic screen font* | Arguments for which you supply values are in italic screen font. |
| < > | Nonprinting characters, such as passwords, are in angle brackets. |
| [ ] | Default responses to system prompts are in square brackets. |
| !, # | An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line. |

This document uses the following conventions:

**Note** Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

**Caution** Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

**Warning** IMPORTANT SAFETY INSTRUCTIONS

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

# Related Documentation

### Cisco Cloud APIC Documentation

The Cisco Cloud APIC documentation is available at the following URL: https://www.cisco.com/c/en/us/support/cloud-systems-management/cloud-application-policy-infrastructure-controller/tsd-products-support-series-home.html

### Cisco Application Policy Infrastructure Controller (APIC) Documentation

The following companion guides provide documentation for Cisco APIC:

- *Cisco APIC Getting Started Guide*
- *Cisco APIC Basic Configuration Guide*
- *Cisco ACI Fundamentals*
- *Cisco APIC Layer 2 Networking Configuration Guide*
- *Cisco APIC Layer 3 Networking Configuration Guide*
- *Cisco APIC NX-OS Style Command-Line Interface Configuration Guide*
- *Cisco APIC REST API Configuration Guide*
- *Cisco APIC Layer 4 to Layer 7 Services Deployment Guide*
- *Cisco ACI Virtualization Guide*
- *Cisco Application Centric Infrastructure Best Practices Guide*

All these documents are available at the following URL: http://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html

### Cisco Application Centric Infrastructure (ACI) Documentation

The broader Cisco ACI documentation is available at the following URL: http://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html.

### Cisco Application Centric Infrastructure (ACI) Simulator Documentation

The Cisco ACI Simulator documentation is available at http://www.cisco.com/c/en/us/support/cloud-systems-management/application-centric-infrastructure-simulator/tsd-products-support-series-home.html.

### Cisco Nexus 9000 Series Switches Documentation

The Cisco Nexus 9000 Series Switches documentation is available at http://www.cisco.com/c/en/us/support/switches/nexus-9000-series-switches/tsd-products-support-series-home.html.

### Cisco Application Virtual Switch Documentation

The Cisco Application Virtual Switch (AVS) documentation is available at http://www.cisco.com/c/en/us/support/switches/application-virtual-switch/tsd-products-support-series-home.html.

### Cisco ACI Virtual Edge Documentation

The Cisco Application Virtual Edge documentation is available at https://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html.

### Cisco ACI Virtual Pod Documentation

The Cisco Application Virtual Pod (vPod) documentation is available at https://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html.

### Cisco Application Centric Infrastructure (ACI) Integration with OpenStack Documentation

Cisco ACI integration with OpenStack documentation is available at http://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html.

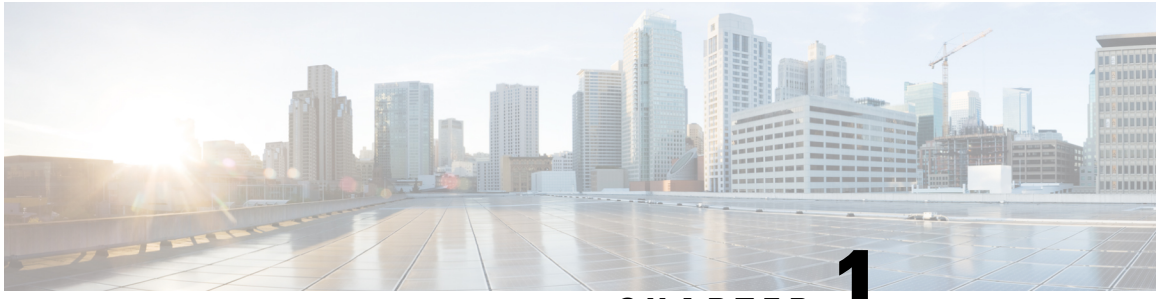
# Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, please send your comments to apic-docfeedback@cisco.com. We appreciate your feedback.

# Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see *What's New in Cisco Product Documentation* at: http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html

Subscribe to *What's New in Cisco Product Documentation*, which lists all new and revised Cisco technical documentation as an RSS feed and delivers content directly to your desktop using a reader application. The RSS feeds are a free service.



CHAPTER 1

# New and Changed Information

This chapter contains the following sections:



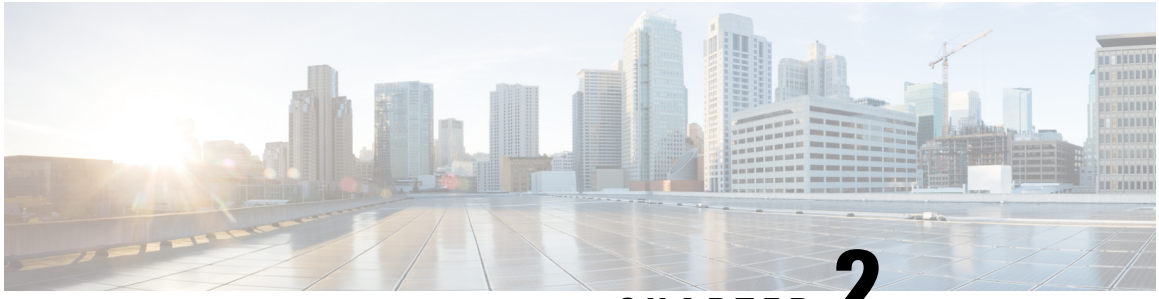
## New and Changed Information

The following table provides an overview of the significant changes to this guide up to this current release. The table does not provide an exhaustive list of all changes made to the guide or of the new features up to this release.

***Table 1: New Features and Changed Behavior***

| Cisco APIC Release Version | Feature | Description | Where Documented |
|---|---|---|---|
| 4.2(1) | Run Bond Watch Service from Heat template | When integrating Cisco UCS B-Series to work with Cisco ACI and OpenStack orchestration, you can run the bond watch service by enabling a parameter to the Heat template.<br><br>The parameter ACIEnableBondWatchService enables the `apic-bond-watch service` when it is added to the Cisco ACI resources template and set to true. | Run the Bond Watch Service, on page 32 |
| 4.2(1) | OpenStack Director deployments can now use certificate-based authentication with Cisco ACI. | Add the ACIApicCertName and ACIApicPrivateKey parameters to the Heat template to enable certificate-based authentication for Cisco ACI. | Installing Overcloud, on page 11 |

| Cisco APIC Release Version | Feature | Description | Where Documented |
|---|---|---|---|
| 4.2(1) | Automatic purge of the tenant from Cisco ACI when a project is deleted from OpenStack | Setting the parameter AciKeystoneNotificationPurge to True enables Keystone notification purge. | Installing Overcloud, on page 11TBD |
| N/A | Configuration of UCS B-Series for Cisco ACI and OpenStack Orchestration | Added appendix about configuring UCS B-Series for Cisco ACI and OpenStack orchestration. | Configuring UCS B-Series, on page 31 |
| 2.3(1) | -- | Added how to add an OpenStack external network. | For more information, see Adding an OpenStack External Network, on page 21. |
| 2.3(1) | -- | Added support for multiple physnets. | For more information, see Installing Overcloud, on page 11. |
| 2.3(1) | -- | Added upgrade procedures. | For more information, see:<br>• Upgrading the Cisco ACI Packages, on page 18<br>• Performing an OSP Major Release Upgrade, on page 19<br>• Release Specific Changes, on page 20 |
| 2.3(1) | -- | Added information about bonded fabric interface with vPC. | For more information, see Requirements and Prerequisites for Cisco ACI with OpenStack Using OSP Director, on page 4. |
| 2.3(1) | Uses tripleo composable services that are introduced in the Newton release. | This guide was released. | -- |



CHAPTER 2

# Cisco ACI Installation Guide for Red Hat OpenStack Using OSP Director

This chapter contains the following sections:



## ACI with OpenStack Using OSP Director Overview

This document uses tripleo composable services that are introduced in the Newton release. This *Cisco ACI Installation Guide for Red Hat OpenStack Using OSP Director, Release 2.3(x) or later* document replaces the *Cisco ACI Installation Guide for Red Hat OpenStack Using OSP Director, Release 2.2(x)* and should be used for Releases 2.3 or later. For more information about composable services, see the OpenStack Composable Services Tutorial at:

https://docs.openstack.org/tripleo-docs/latest/install/developer/tht_walkthrough/tht_walkthrough.html

Cisco Application Centric Infrastructure (ACI) is a comprehensive policy-based architecture that provides an intelligent, controller-based network switching fabric. This fabric is designed to be programmatically managed through an API interface that can be directly integrated into multiple orchestration, automation, and management tools, including OpenStack. Integrating ACI with OpenStack allows dynamic creation of networking constructs to be driven directly from OpenStack requirements, while providing additional visibility within the ACI Application Policy Infrastructure Controller (APIC) down to the level of the individual virtual machine (VM) instance.

OpenStack defines a flexible software architecture for creating cloud-computing environments. The reference software-based implementation of OpenStack allows for multiple Layer 2 transports including VLAN, GRE, and VXLAN. The Neutron project within OpenStack can also provide software-based Layer-3 forwarding. When utilized with ACI, the ACI fabric provides an integrated Layer 2 and Layer 3 VXLAN-based overlay networking capability that can offload network encapsulation processing from the compute nodes onto the top-of-rack or ACI leaf switches. This architecture provides the flexibility of software overlay networking in conjunction with the performance and operational benefits of hardware-based networking.

The Cisco ACI OpenStack plugin can be used in either ML2 or GBP mode. In Modular Layer 2 (ML2) mode, a standard Neutron API is used to create networks. This is the traditional way of deploying VMs and services in OpenStack. In Group Based Policy (GBP) mode, a new API is provided to describe, create, and deploy applications as policy groups without worrying about network-specific details. Keep in mind that mixing GBP and Neutron APIs in a single OpenStack project is not supported. For more information, see the *OpenStack Group-Based Policy User Guide* at:

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/1-x/openstack/b_OpenStack_Group-Based_Policy_User_Guide.html

In previous OpFlex plugin versions (referred to as Classical mode), it was necessary to decide at the time of deployment if the mode of the plugin will be Neutron/ML2 or GBP, and it was not possible to use both GBP and Neutron/ML2 APIs at the same time. Starting from OpFlex plugin version 2.2.1. It is possible to deploy the plugin in "Unified" mode. In unified mode it is possible to create application topologies using either Neutron or GBP API. Unified plugin mode also requires OpenStack release Mitaka or later and ACI release 2.2(1) or later.

This guide covers deployment of the OpFlex plugins in Unified installation mode.

**Note** While creating GBP groups in Unified mode, an (auto-ptg) group will also appear, these groups are for internal use and user interaction (attaching VM, adding members) is not supported.

# Requirements and Prerequisites for Cisco ACI with OpenStack Using OSP Director

- Target audience: You must have working knowledge of Linux, Red Hat OpenStack distribution, Cisco ACI policy model and GUI-based APIC configuration. Also familiarity with OpenStack architecture and deployment.
- Cisco ACI fabric: Cisco ACI fabric is installed and initialized with the minimum supported version that is documented in the Cisco ACI Virtualization Compatibility Matrix. For basic guidelines on initializing a new Cisco ACI fabric, see the ACI Fabric Initialization Example, on page 15, on page.

  **Note** For communication between multiple leaf pairs, the fabric must have a BGP route reflector enabled to use an OpenStack external network.

- When using bonded fabric interface with vPC, adding the `ovs_bond` for the fabric interface is not supported because it must be added as a single interface to the ovs bridge. You must set the `type` to `linux_bond` for aggregating the fabric interfaces. Here is a rough example of how the fabric interface needs to be created in the `nic-config` templates:

```
type: ovs_bridge
              name: {get_input: bridge_name}
              mtu: 1500
              members:
                -
                  type: linux_bond
                  name: bond1
```

```
ovs_options: {get_param: BondInterfaceOvsOptions}
mtu: 1600
members:
  -
    type: interface
    name: nic1
    primary: true
    mtu: 1600
  -
    type: interface
    name: nic2
    mtu: 1600
```

- When using bonding, only 802.3ad is supported.
- When deploying with UCS-B series, only dual vNICs with bonding is supported for fabric interface for redundancy.

**Note** Do not use a single vNIC with hardware failover.

- In the Cisco APIC GUI, disable the OpFlex authentication in the fabric. Make sure "To enforce OpFlex client certificate authentication for GOLF and Linux." is not checked in **System** > **System Settings** > **Fabric Wide Setting** > **Fabric Wide Setting Policy** pane.
- When you delete the Overcloud Heat stack, the Overcloud nodes are freed, but the Virtual Machine Manager (VMM) domain remains present in Cisco APIC. The VMM appears in Cisco APIC as a stale VMM domain along with the tenant unless you delete the VMM domain manually.

  Before you delete the VMM domain, verify that the stack has been deleted from the undercloud. Also check that any hypervisors appearing under the VMM domain are no longer in the connected state. Once both of these conditions are met, then you can safely delete the VMM domain from Cisco APIC.

## Related Documentation

For more information, see the *Director Installation and Usage Red Hat OpenStack Platform 10* documentation at:

https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/10/html/director_installation_and_usage/

# Deploying OpFlex Using Unified Mode

This section describes how to install and configure the Cisco ACI OpenStack Plug-in on a Red Hat OpenStack distribution.

These example steps were validated on OpenStack Platform 10 releases of Red Hat OpenStack. OpenStack systems can vary widely in how they are installed. Therefore, you can adapt the example steps according to the specifics of your installation.

Follow the Red Hat OpenStack Platform Director installation document to prepare the OpenStack Platform Director and create the correct deployment and resource files.

For more information, see the Related Documentation, on page 5.

# Preparing ACI for OpenStack Installation

## Setting Up the APIC and the Network

This section describes how to set up the Cisco APIC and the network.

Refer to the Network Planning section of the OpenStack Platform Director documentation for network layout such as the one shown in the figure below.

For more information, see Related Documentation, on page 5.

*Figure 1: A typical OpenStack Platform topology*

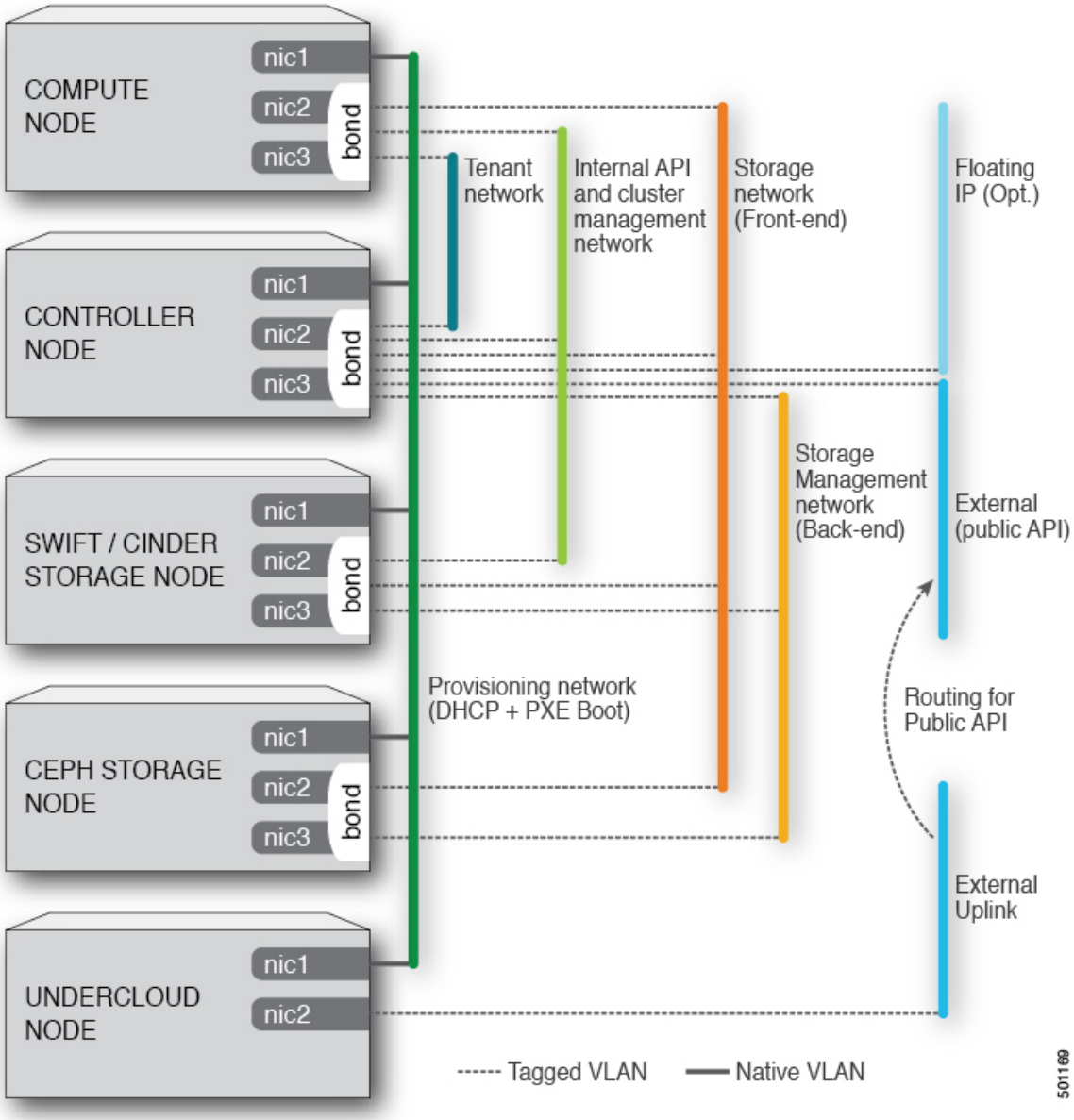
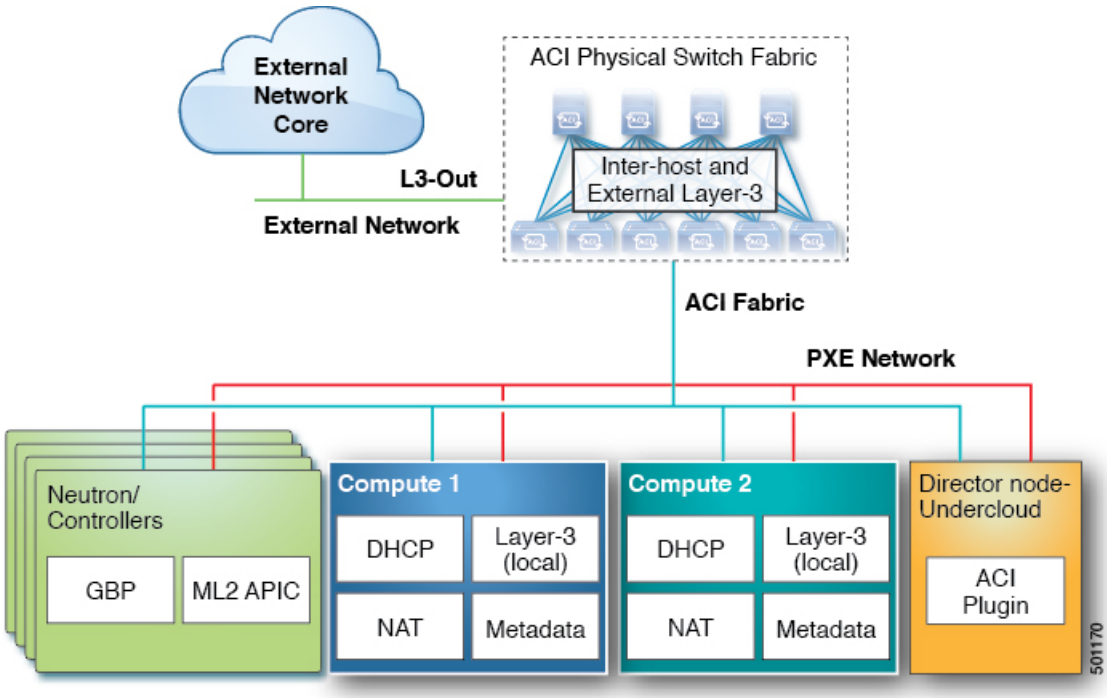


*Figure 2: A typical topology for installation of Red Hat OpenStack Platform 10 with the ACI plug-in*




- PXE Network is out-of-band (OOB) and uses a dedicated interface.
- All OpenStack Platform (OSP) networks except for PXE are in-band (IB) through ACI.
  - API - VLAN 10
  - Storage - VLAN 11
  - StorageMgmt - VLAN 12
  - Tenant - VLAN 13
  - External - VLAN 14
  - ACI Infra - VLAN 4093
- L3Out is pre-configured (In this example it is called L3-Out and EPG is L3-Out-EPG).

To prepare Cisco ACI for in-band configuration you can use the physical domain and the static binding to the endpoint groups (EPGs) created for these networks. This involves creating the required physical domain and attachable access entity profile (AEP). Important thing to note is that Infra VLAN should be enabled for the AEP. For more details, see:

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/kb/b_KB_Creating_AEP_Physical_Domains_VLANS_to_Deploy_an_EPG_on_a_Specific_Port.html

### Procedure

**Step 1** Log in to the Cisco APIC GUI and create a VLAN Pool for VLANs required for OpenStack Platform installation.

a) On the menu bar, choose **Fabric** > **Access Policies** > **Pools** and right-click **VLAN** to create a VLAN Pool.
b) In the **Name** field, enter the VLAN range namespace policy name (OSP8-infra).
c) (Optional) In the **Description** field, enter the description of the VLAN range namespace policy.
d) In the **Encap Blocks** section, click on the + icon to enter the encap block range.
e) Click **SUBMIT**.

**Step 2** Create an attachable entity profile and assign the above PhysDom to it. Also make sure **Enable Infra VLAN** is selected:

a) On the menu bar, choose **Fabric** > **Access Policies** > **Global Policies** and right-click **Attachable Access Entity Profile** to create an attachable access entity profile.
b) In the **Name** field, enter the name of the attachable access entity profile (OSP8-AEP).
c) (Optional) In the **Description** field, enter the description of the attachable access entity profile.
d) Check the **Enable Infrastructure VLAN** check box to enable the infrastructure VLAN.
e) In the **Domains (VMM, Physical or External) To Be Associated To Interfaces:** section, click on the + icon, from the drop-down list, choose the domain profile and click **Update**.
f) Click **Next**.
g) Click **Finish**.

**Step 3** Create a Physical Domain (PhysDom) and assign the VLAN pool to it.

a) On the menu bar, choose **Fabric** > **Access Policies** > **Physical and External Domains** and right-click **Physical Domains** to create a Physical Domain.
b) In the **Name** field, enter the name of the physical domain (OSP8-Phys).
c) In the **Associated Attachable Entity Profile** field, choose an associated attachable entity profile.
d) In the **VLAN Pool** field, choose a VLAN pool ([OSP8-infra-dynamic]).
e) Click **SUBMIT**.

**Step 4** In a separate tenant, you can also use Common to create an application profile (For example: OSP-8). Create the EPGs, bridge domains, and a VRF for the OSP Networks. If the PXE network is also going through ACI then also create EPG and BD for PXE (This is not shown in this example).

**Step 5** Add static bindings (Paths) for the required VLANs. You have to expand the EPG to see the ":Static Binding Paths".

a) Make sure the physical domain you created is attached to this EPG. You can add the physical domain using **Application Profiles** > **EPG** > **EPG_name** > **Domains**.
b) On the menu bar, choose **Tenants** > **Tenant common** > **Application Profiles** > ***ACI-OSP8*** > **Application EPGs** > ***EPG API*** > **Static Binding Paths**.

**Step 6** Make sure the PhysDom is attached to the EPG.

**Note** Cisco ACI needs to be provisioned for networks mentioned above except for Tenant, External and Floating IP network. This involves creating the required phys-doms and attached entity profile. Important thing to note is that Infra VLAN should be enabled for the attached entity profile.

Cisco ACI should now be ready for OpenStack deployment.

# Setting Up Overcloud

Follow the Red Hat OpenStack Platform Director installation document to prepare the OpenStack Platform Director and create the correct deployment and resource files.

**Note** At the time of writing, the overcloud nodes try to resolve ipv6 DNS entries for localdomain, this can cause significant slowdown if the DNS server actually tries to resolve the name instead of sending the NXDomain. If you notice a significant slowdown, make sure that your DNS server is configured correctly.

Once the OpenStack Platform (OSP) Director is setup, you need to install the ACI TripleO orchestration before proceeding with deployment.

## Preparing for ACI with OpFlex Orchestration

To install and enable ACI OpFlex on Overcloud, the following is a summary of steps that are required.

- Modify the undercloud to include the necessary software packages.
- Add to the Neutron puppet manifests, which are part of Overcloud image.
- Add the OpFlex puppet manifests.
- Modify some files on the undercloud tripleO infrastructure.
- Create a HEAT environment file to provide ACI-related parameter values.
- After the above modifications, Overcloud can be provisioned using the **openstack overcloud deploy** command and add the new environment file to the **openstack overcloud deploy** command.

## Preparing Undercloud for Cisco ACI with OpFlex Orchestration

This section describes how to install integration package for Cisco ACI with OpFlex Orchestration.

**Note** The following steps automatically create a local RPM repository on the undercloud, which will host Cisco ACI OpFlex RPM packages.

**Procedure**

**Step 1** Log in to undercloud as user `stack`.

**Step 2** Source the `stackrc` file.

**Example:**

```
$ source stackrc
```

**Step 3** Download the latest ACI OSP (tripleo-ciscoaci) rpm from cisco.com.

For more information, see the APIC OpenStack Plugins.

**Step 4** Install the rpm. This installs the dependencies. If the rpm is installed using the `rpm` command, some dependency may need to be manually installed.

**Example:**

```
$ sudo yum --nogpgcheck localinstall <rpm file>
```

# Installing Overcloud

This section describes how to install Overcloud.

**Procedure**

**Step 1** To use Cisco ACI certificate-based authentication, create a local user with a X.509 certificate.

Follow the procedure "Creating a Local User and Adding a User Certificate" in the *Cisco APIC Security Configuration Guide, Release 4.2(x)* .

**Note** When you use certificate-based authentication, make sure that the parameter ACIApicPassword is not specified.

**Step 2** Copy the `/opt/tripleo-ciscoaci/example_ciscoaci.yaml` file to the `/home/stack/templates/apic_gbp_config.yaml` file. Edit the `apic_gbp_config.yaml` file and change the `parameter_defaults` to reflect the setup details.

```
parameter_defaults:
NeutronCorePlugin: 'ml2plus'
NeutronServicePlugins: 'group_policy,ncp,apic_aim_l3'
NeutronEnableIsolatedMetadata: true
EnablePackageInstall: true
ACIYumRepo: http://10.10.250.10/acirepo
ACIApicHosts: 172.31.218.136,172.31.218.137,172.31.218.138
ACIApicUsername: admin
ACIApicPassword: cisco123
ACIApicSystemId: osp10_cs
ACIApicEntityProfile: f-aep
ACIApicInfraVlan: 4093
ACIApicInfraSubnetGateway: 10.0.0.30
ACIApicInfraAnycastAddr: 10.0.0.32
ACIOpflexUplinkInterface: nic2
ACIOpflexEncapMode: vxlan
ACIOpflexVlanRange: 1200:1300
NeutronEnableForceMetadata: true
ACIOpflexBridgeToPatch: br-custom
```

| Parameter | Description |
|---|---|
| `ACIYumRepo: http://10.10.250.10/acirepo` | Where the IP address is in the URL should be replaced with IP address of the director. This is where the OpFlex RPM will be installed from. The Repo is automatically created when `tripleo_ciscoaci` package is installed. |

| Parameter | Description |
|---|---|
| `ACIApicHosts: 172.31.218.136,172.31.218.137,172.31.218.138` | This lists the IP addresses or hostnames for the APICs. |
| `ACIApicUsername: admin` | This is the APIC username. |
| `ACIApicPassword: cisco123` | This is the APIC password. |
| `ACIApicSystemId: osp10_cs` | This should be a unique string to identify this particular OpenStack instance. |
| `ACIApicEntityProfile: f-aep` | This is the name of the AEP to attach the VMM domain in the ACI. This AEP needs to be created manually and should pre-exist before installing the Overcloud. |
| `ACIApicInfraVlan: 4093` | The ACI Infra VLAN is the OpFlex infra VLAN. It is picked during the ACI fabric initialization. |
| `ACIApicInfraSubnetGateway: 10.0.0.30` | This is the anycast IP address assigned to the SVI of the infra VLAN. |
| `ACIApicInfraAnycastAddr: 10.0.0.32` | This IP address matches the anycast IP address assigned to interface Loopback 1023 on the leaf switches. |
| `ACIOpflexUplinkInterface: nic2` | The interface is used for OpFlex. This is the fabric interface, it can be a individual or bonded interface. Follow the OSP director template guidelines for determining the interface name. |
| `ACIOpflexEncapMode: vxlan` | The encapsulation to be used between compute nodes and leaf switches is `vxlan` or `vlan`. |
| `ACIOpflexVlanRange: 1200:1300` | This is the VLAN range for encapsulation. Only needed when using the `vlan` encapsulation. |
| `NeutronEnableForceMetadata: true` | This is required to enable OpFlex optimized metadata. |
| `ACIOpflexBridgeToPatch: br-custom` | This parameter is only needed when using the VLAN encapsulation and customized templates. This parameter should be set to name of the bridge which is attached to the fabric uplink interface (or bond). The default bridge in Red Hat template is ‘br-ex’. If the default ‘br-ex’ is used for the deployment, this parameter is not needed. Otherwise set the value to the bridge name and a patch will be created between this bridge and integration bridge ‘br-int’. |

| Parameter | Description |
|---|---|
| `ACIApicCertName` | • **Value:** Name of the Cisco APIC cert User (used for certificate-based authentication<br>• **Type:** String<br>• **Default:** None<br>• **Mandatory:** No |
| `ACIApicPrivateKey` | • **Value:** Private key for the cert User<br>• **Type:** String<br>• **Default:** None<br>• **Mandatory:** No |
| `ACIEnableBondWatchService` | • **Value:** True or False<br>• **Type:** Boolean<br>• **Default:** False<br>• **Comment** Set this parameter to True if you use Cisco Unified Computing System (UCS) blade servers for OpenStack nodes. |
| `AciKeystoneNotificationPurge` | • **Value:** True or False<br>• **Type:** Boolean<br>• **Default:** False<br>• **Comment** Enables automatic purge of Cisco APIC tenants when the project is deleted in OpenStack. |

**Step 3** Edit the `/home/stack/templates/network-environment.yaml` file and set the following values:

```
NeutronEnableTunnelling: False
NeutronTunnelTypes: "''"
NeutronEnableL3Agent: False
NeutronEnableOVSAgent: False
```

**Note** These settings are necessary in order to make sure the default agents that are replaced by OpFlex agents are disabled. We recommend to keep these setting as indicated.

**Step 4** Deploy Overcloud as described in the installation document.

For more information, see the *Director Installation and Usage Red Hat OpenStack Platform*, chapter 7.

a) Deploy the new environment file created.

**Example:**

```
openstack overcloud deploy --templates -e
/usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml -e
```

```
~/templates/network-environment.yaml  -e
~/templates/apic_gbp_config.yaml
```

**Step 5** On successful deployment, the appropriate VMM domain is created on Cisco APIC. Make sure to add this VMM domain to the right attached entity profile before creating OpenStack networks. If the attached entity profile is specified in the Cisco ACI yaml file using the parameter "ACIApicEntityProfile," this step is not required.

**Step 6** Configuring multiple physnets and Hierarchical Port Binding (HPB).

It is now possible to deploy OSP with ACI, automatically configured for HPB and multiple physnets. The plug-in supports specification of multiple mechanism drives for HPB. The HPB requires pre-creation of physical domains in ACI per HPB physnet with VLAN pool for the NetworkVLANRanges for that particular physnet.

To specify physnet to physical interface or bond relationship:

You need to add the following parameters to the apic_gbp_config.yaml file as in step 1:

```
NeutronPhysicalDevMappings: physnet1:ens11,physnet2:ens7,physnet3:bond1
NeutronNetworkVLANRanges: physnet1:1200:1250,physnet2:1251:1300,physnet3:1301:1350
ACIMechanismDrivers: 'sriovnicswitch,apic_aim'
```

| Parameter | Description |
|---|---|
| `NeutronPhysicalDevMappings` | This parameter specifies which interface belongs to which physnet. For this to work correctly APIC physdoms need to be pre-created with names prefixed with pdom_.<br>For example:<br>For physnet2, create a physdom with name pdom_physnet2. |
| `NeutronNetworkVLANRanges` | This lists the IP addresses or hostnames for the APICs. |
| `ACIMechanismDrivers` | The mechanism drivers to configure in ml2 configuration. For HPB to work correctly, the apic_aim should be the last one in the list. |

# Reference Links

- *Director Installation and Usage Red Hat OpenStack Platform 8* documentation:

  https://access.redhat.com/documentation/en/red-hat-openstack-platform/version-8/director-installation-and-usage/#chap-Introduction

- *VLAN static binding to ports* knowledge base article:

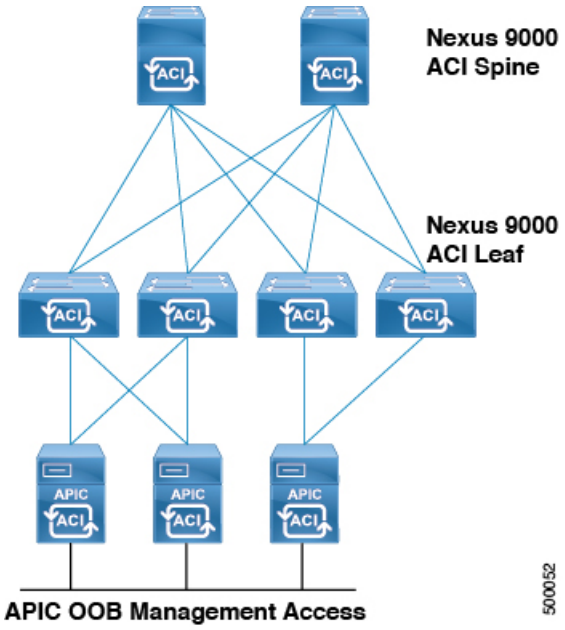
  http://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/kb/b_KB_Creating_AEP_Physical_Domains_VLANS_to_Deploy_an_EPG_on_a_Specific_Port.html

# ACI Fabric Initialization Example

This example solution is based on a basic Spine/Leaf switching fabric, installed with all defaults on the APIC configuration other than Fabric Name and controller IP addressing. Three APICs are used to form a highly available cluster. Each APIC is connected to one or more of the leaf switches in the fabric, it is best to use diverse leaf switches for connecting multiple APICs to provide higher availability of the controller services.

The switching system continues to forward traffic regardless of the presence of the APIC cluster. All configuration of the fabric is driven through the cluster so no configuration could be added, changed, or deleted without APIC connectivity in place. To ensure that administrative control of the fabric is not dependent on the fabric itself, an Out-Of-Band (OOB) network connection is needed on each of the APICs as shown in the Figure below:

***Figure 3: APIC Cluster Connectivity***




### Procedure

**Step 1** A good practice for setup of the ACI fabric is to make a note of the serial number of each of the switches in the fabric prior to discovering the fabric. Ideally, the console port of each of the switches is also connected to a terminal server so there is always administrative control regardless of the state of the ACI fabric. To recover the serial number when logged into a switch running an ACI software image, enter the **show inventory** command at the ACI switch CLI, noting the primary system serial number. This is the number that displays in the APIC during fabric discovery, allowing you to assign the correct name and node numbering in your scheme to the devices.

**Step 2** To allow the APIC to discover and register the switches in the fabric, log in to the APIC GUI (Advanced mode).

a) On the menu bar, choose **Fabric** > **Inventory**.

b) In the **Navigation** pane, choose **FabricMembership**.

c) In the **Work** pane, you should see an entry for the first switch discovered by the APIC.
d) Verify this is the expected first switch for the first APIC in the cluster based on serial number.
e) In the **Work** pane, choose the switch, right-click and choose **Register Switch**.

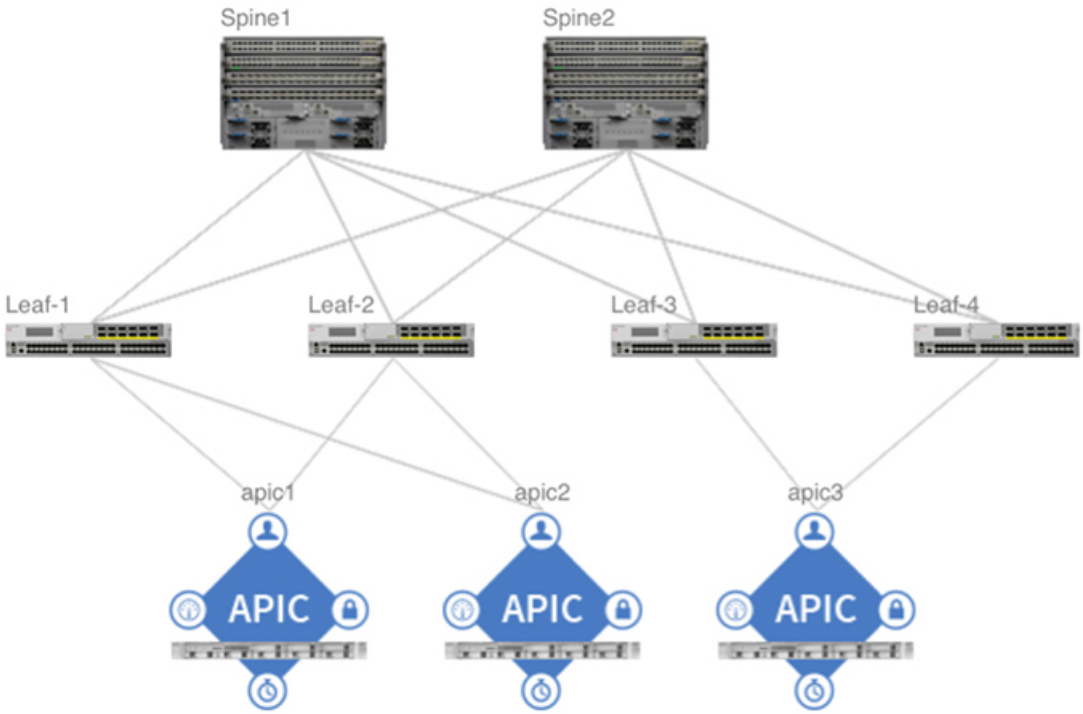
**Note** Assign logical numeric node IDs and node names that make sense for future troubleshooting, and Virtual Port Channel (vPC) pairing plans. For example, Node IDs 101/102 for the first two leaf switches, to be named leaf1/leaf2.

**Step 3** Once the first leaf is discovered, the system will pass through that leaf to discover the spine switches, and then use the spine switches to discover remaining leaf switches. Register the additional nodes assigning logical node ID numbers and names according to the spine/leaf fabric layout.

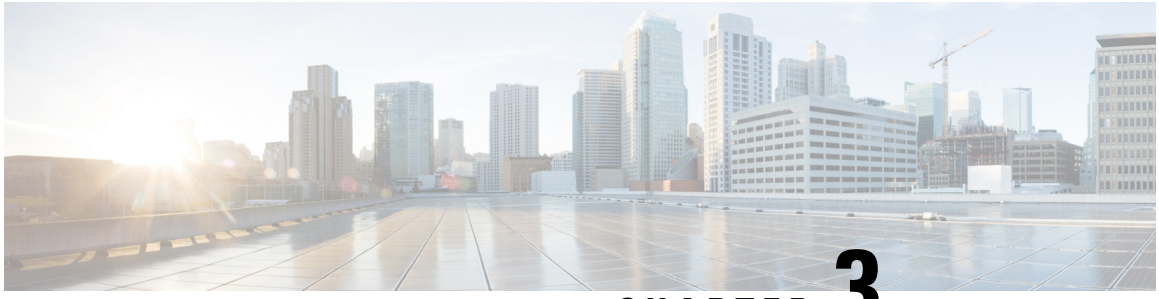
**Step 4** Confirm visually that the topology is discovered and physically connected as expected, perform the following actions:

a) On the menu bar, choose **Fabric** > **Inventory**.
b) In the **Navigation** pane, choose **Topology**.

***Figure 4: Discovered Spine/Leaf Topology***




c) Once the fabric is discovered, choose **Admin** > **Firmware** and validate the firmware versions running on all APICs, and fabric nodes (switches). If needed, upgrade to current or consistent versions before beginning initial configuration.



CHAPTER 3

# Upgrade ACI and OSP

This chapter contains the following sections:



# Guidelines for Upgrading Cisco APIC and OSP

The OpenStack plug-in is released with Cisco Application Policy Infrastructure ControllerAPIC releases, and therefore uses the same semantic version as Cisco APIC. For example, the 4.2(1) plug-in is provided with the Cisco APIC 4.2(1) release. Generally, the OpenStack plug-in releases are tested against the matching Cisco APIC release, as well as the previous Long Term Support (LTS) Cisco APIC release. However, a given plug-in release may be compatible with additional Cisco APIC releases. See the *Cisco ACI Virtualization Compatibility Matrix* to verify that the version of the plug-in used is compatible with the version of Cisco APIC.

See the *Cisco ACI Virtualization Compatibility Matrix* for information about compatible Cisco APIC and Red Hat OSP releases.

## Pre-upgrade Guidelines

You need to first upgrade the Cisco Application Centric Infrastructure (ACI) plug-in.

For more information about the compatibility of the plug-in with various OpenStack versions, see the *Cisco ACI Virtualization Compatibility Matrix* at:

https://www.cisco.com/c/dam/en/us/td/docs/Website/datacenter/aci/virtualization/matrix/virtmatrix.html

## Upgrade Guidelines

The Cisco ACI fabric can be upgraded following the information on Cisco Application Policy Infrastructure Controller (APIC) Management, Installation, Upgrade, and Downgrade Guide at:

https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/2-x/managing_ACI_fabric_upgrades_and_downgrades/b_Managing_ACI_Fabric_Upgrades_and_Downgrades/b_Managing_ACI_Fabric_Upgrades_and_Downgrades_chapter_01.html

Optionally, you can upgrade the Cisco ACI fabric without upgrading the plug-in, as long as the Cisco ACI plug-in and Cisco ACI fabric release combination is supported. For more information, see the *Cisco ACI Virtualization Compatibility Matrix* at:

https://www.cisco.com/c/dam/en/us/td/docs/Website/datacenter/aci/virtualization/matrix/virtmatrix.html

## Post-upgrade Guidelines

After you upgrade the Cisco ACI fabric, you can optionally upgrade the OpenStack ACI packages to a version which is equal or lower than the ACI fabric code you have upgraded to. You should also refer to the OpenStack ACI Plugin Release Notes for specific information:

https://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html#Cisco_APIC_OpenStack_Release_Notes

For more information on how to upgrade the OpenStack ACI plugin, see Upgrading the Cisco ACI Packages, on page 18

# Upgrading the Cisco ACI Packages

The Cisco Application Centric Infrastructure (ACI) release upgrade can be live; follow the Red Hat director documentation. See the "Director-Based Environments: Performing Updates to Minor Versions" chapter of *Upgrading Red Hat OpenStack Platform* for Platform 10 on the Red Hat website.

**Note** The update of ACI packages from pre-2.3 releases to 2.3 or later is not supported. In other words, only composable services deployments are upgradable.

Here are example steps to perform an upgrade in a fully deployed Overcloud:

### Procedure

**Step 1** Copy the updated version of the `tripeo-ciscoaci` RPM from CCO to the OSP Director.

**Step 2** Remove the existing RPM package by entering the following command:

```
sudo rpm -ev tripleo-ciscoaci
```

**Step 3** Install the `tripleo-ciscoaci` package using `yum`:

```
sudo yum install tripleo-ciscoaci-<version number>.rpm
```

**Step 4** Update the current plan using your original **openstack overcloud deploy** command but including the **--update-plan-only** option.

**Example:**

```
openstack overcloud deploy --update-plan-only --templates -e \
         #/home/stack/templates/scheduler_hints_env.yaml -e \
         #/usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml
\
         #-e ~/templates/network-environment.yaml -e /home/stack/templates/aci_cs.yaml \
```

```
          #-e ~/templates/rhel-registration-resource-registry.yaml -e \
          #~/templates/environment-rhel-registration.yaml  --control-flavor control
--compute-flavor compute
```

**Step 5** Update the stack to push the new packages to the Overcloud:

```
openstack overcloud update stack -i overcloud
```

Upon completion of the preceding command, the Overcloud updates with the new version of the plug-in.

# Performing an OSP Major Release Upgrade

For a major release upgrade, you must follow the Red Hat director documentation for a major release upgrade. For more information, see the "Director-Based Environments: Performing a Major Version Upgrade" section of *Upgrading Red Hat OpenStack Platform* on the Red Hat website.

**Note** In release 2.3 or later, live upgrade is only supported with composable services as described in this document. You cannot upgrade from previous versions.

You may also need to update some templates and update the `tripleo-ciscoaci` package. Using the right procedure, it is possible to perform a live upgrade of the Overcloud. Here is the summary of the steps:

**Procedure**

**Step 1** Remove the `tripleo-ciscoaci package` from the existing OpenStack version.

**Step 2** Stop and upgrade the undercloud by following the Red Hat director documentation to the end of section "Upgrading the Overcloud Images" in *Upgrading Red Hat OpenStack Platform* on the Red Hat website.

**Step 3** Stop the aim processes on all overcloud controller nodes. This can be accomplished by using the following commands on the director:

**Example:**

```
source stackrc
for IP in $(nova list | grep ACTIVE | sed 's/.*ctlplane=//' | sed
's/ |//'); do ssh -o UserKnownHostsFile=/dev/null -o
StrictHostKeyChecking=no heat-admin@$IP "sudo systemctl stop
aim-event-service-rpc; sudo systemctl stop aim-aid; sudo systemctl
stop aim-event-service-polling"; done
```

**Step 4** Install the `tripleo-ciscoaci` package for the new major Red Hat release.

**Step 5** Update the templates required for a specific release.

This step may be release-specific, and the Cisco ACI plug-in upgrade may require changes. For more information, see the Release Specific Changes, on page 20.

**Step 6** Upgrade the Overcloud by following the rest of Red Hat the document.

# Release Specific Changes

This section describe release specific changes to the templates that maybe require to upgrade between releases. Follow the section that correspond to your release version.

In OSP 11 and Ocata release of OpFlex plugin, the package "`neutron-ml2-cisco`" is deprecated and needs to be removed during the upgrade to OSP 11. These changes in the `openstack-tripleo-templates` are required for the upgrade from OSP 10 to OSP 11.

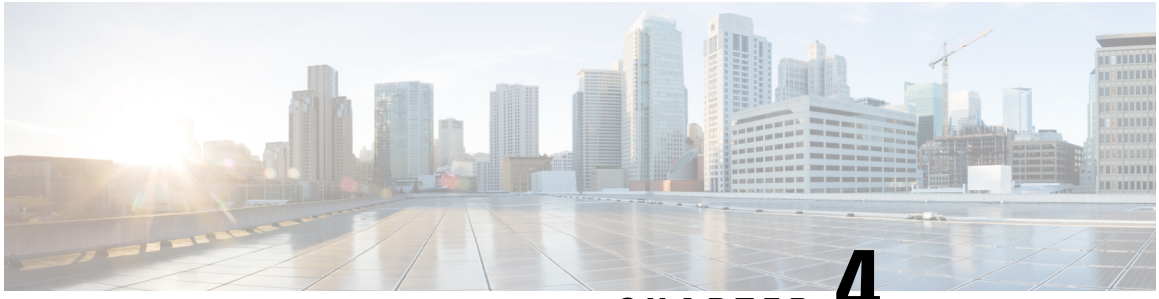
### Procedure

**Step 1** Edit the `/usr/share/openstack-tripleo-heat-templates/puppet/services/tripleo-packages.yaml` fiel and add the highlighted lines:

```
- name: Fail when rpm-python wasn't present
  fail: msg="rpm-python package was not present before this run! Check environment before
re-running"
  when: rpm_python_check.changed != false
  tags: step0
- name: clean yum cache
  tags: step0
  command: yum clean all
  ignore_errors: True
- name: remove neutron-ml2-cisco
  tags: step3
  yum: name=neutron-ml2-driver-apic state=removed
- name: Update all packages
  tags: step3
  yum: name=* state=latest
```

**Step 2** Edit the `/usr/share/openstack-tripleo-heat-templates/extraconfig/tasks/tripleo_upgrade_node.sh` file and add the highlighted line:

```
set -eu
yum -y remove neutron-ml2-driver-apic || true
NOVA_COMPUTE=""
```



CHAPTER 4

# Add an OpenStack External Network

This chapter contains the following sections:



## Adding an OpenStack External Network

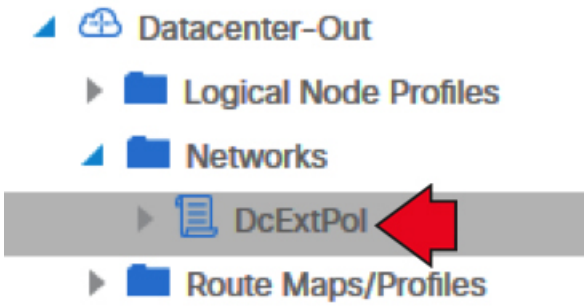
This section describes how to add an OpenStack External Network.

**Note** A preexistng L3Out and corresponding endpoint group (EPG) is required for external connectivity.

There is a change in the command for configuring external-network (new extensions have been added) in the OpenStack to correspond to the above pre-existing L3Out and its EPG. The configuration requires the DN of the EPG for the specific the L3Out in APIC. It can be done in two ways:

**Procedure**

**Step 1** Go to the APIC and get the DN of the external network using the API-Inspector:




**Step 2** Use the **`aimctl manager`** command on the Openstack-Controller to get the **`dn`** as shown below:

**Note** See the example for L3Out: Datacenter-Out.

**Example:**

```
aimctl manager external-network-find
+---------------+----------------+------------+
| tenant_name   | l3out_name     | name       |
|---------------+----------------+------------|
| common        | Datacenter-Out | DcExtPol   |
| common        | Management-Out | MgmtExtPol |
| common        | default        | MgmtNet    |
+---------------+----------------+------------+

 aimctl manager external-network-get common Datacenter-Out DcExtPol
+-------------------------+-----------------------------------------------+
|Property                 |Value                                          |
|-------------------------+-----------------------------------------------
| tenant_name             | common                                        |
| l3out_name              | Datacenter-Out                                |
| name                    | DcExtPol                                      |
| display_name            |                                               |
| nat_epg_dn              |                                               |
| provided_contract_names | []                                            |
| consumed_contract_names | []                                            |
| monitored               | True                                          |
| dn                      | uni/tn-common/out-Datacenter-Out/instP-DcExtPol |
+-------------------------------------------------------------------------+
```

Now the **dn** from above can be used to create the external network.

For Distributed NAT:

**Example:**

```
neutron net-create Datacenter-Out --router:external True --shared --apic:distinguished_names
 \
type=dict ExternalNetwork=uni/tn-common/out-Datacenter-Out/instP-DcExtPol
```

For No NAT:

**Example:**

```
neutron net-create Datacenter-Out --router:external True --shared --apic:distinguished_names
 \
type=dict ExternalNetwork=uni/tn-common/out-Datacenter-Out/instP-DcExtPol --apic:nat_type
""
```

For Floating IPs, add a subnet to external network with the desired floating IP pool cidr.

For SNAT, add a subnet to the external network as below:

**Example:**

```
neutron subnet-create Datacenter-Out 10.104.21.0/24 --name ext-subnet --disable-dhcp --gateway
 \
10.104.21.1 --apic:snat_host_pool True
```

For GBP external connectivity, first create the external segment using the SNAT subnet:

**Example:**

```
gbp external-segment-create Datacenter-Out --subnet-id <SNAT Subnet>
```

Now this segment can be used as usual to create the network service policy.

**Example:**

```
gbp nat-pool-create nat-pool-0 --ip-pool 10.104.31.0/24 --external-segment Datacenter-Out

gbp network-service-policy-create --network-service-params
type=ip_pool,name=nat-pool-0,value=nat_pool \
net-svc-nat-0
```

APPENDIX A

# Reference Information

This chapter contains the following sections:



# Sample Deployment Templates

This appendix lists the sample deployment files. These are for reference only and are not intended for a production system. Create your deployment files to reflect your topology and deployment scenario.

## controller.yaml

```
heat_template_version: 2015-04-30

description: >
  Software Config to drive os-net-config to configure VLANs for the
  controller role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
```

```
    type: string
  ManagementIpSubnet: # Only populated when including environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  ExternalNetworkVlanID:
    default: 10
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: 20
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
  StorageMgmtNetworkVlanID:
    default: 40
    description: Vlan ID for the storage mgmt network traffic.
    type: number
  TenantNetworkVlanID:
    default: 50
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 60
    description: Vlan ID for the management network traffic.
    type: number
  ExternalInterfaceDefaultRoute:
    default: '10.0.0.1'
    description: default route for the external network
    type: string
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations) that will be added
 to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::StructuredConfig
    properties:
      group: os-apply-config
      config:
        os_net_config:
          network_config:
            -
              type: interface
              name: nic1
              dns_servers: {get_param: DnsServers}
              addresses:
                -
                  ip_netmask:
                    list_join:
                      - '/'
                      - - {get_param: ControlPlaneIp}
```

```
                    - {get_param: ControlPlaneSubnetCidr}
            routes:
              -
                ip_netmask: 169.254.169.254/32
                next_hop: {get_param: EC2MetadataIp}
          -
            type: ovs_bridge
            name: {get_input: bridge_name}
            members:
              -
                type: interface
                name: nic2
                # force the MAC address of the bridge to this interface
                primary: true
              -
                type: vlan
                vlan_id: {get_param: ExternalNetworkVlanID}
                addresses:
                -
                  ip_netmask: {get_param: ExternalIpSubnet}
                routes:
                  -
                    default: true
                    next_hop: {get_param: ExternalInterfaceDefaultRoute}
              -
                type: vlan
                vlan_id: {get_param: InternalApiNetworkVlanID}
                addresses:
                  -
                    ip_netmask: {get_param: InternalApiIpSubnet}
              -
                type: vlan
                vlan_id: {get_param: StorageNetworkVlanID}
                addresses:
                  -
                    ip_netmask: {get_param: StorageIpSubnet}
              -
                type: vlan
                vlan_id: {get_param: StorageMgmtNetworkVlanID}
                addresses:
                  -
                    ip_netmask: {get_param: StorageMgmtIpSubnet}
              -
                type: vlan
                vlan_id: {get_param: TenantNetworkVlanID}
                addresses:
                  -
                    ip_netmask: {get_param: TenantIpSubnet}
              #-  # Uncomment when including environments/network-management.yaml
              #  type: vlan
              #  vlan_id: {get_param: ManagementNetworkVlanID}
              #  addresses:
              #    -
              #      ip_netmask: {get_param: ManagementIpSubnet}

outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

# compute.yaml

```
heat_template_version: 2015-04-30

description: >
  Software Config to drive os-net-config to configure VLANs for the
  compute role.

parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  InternalApiNetworkVlanID:
    default: 20
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 30
    description: Vlan ID for the storage network traffic.
    type: number
  TenantNetworkVlanID:
    default: 50
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 60
    description: Vlan ID for the management network traffic.
    type: number
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations) that will be added
 to resolv.conf.
```

```
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string

resources:
  OsNetConfigImpl:
    type: OS::Heat::StructuredConfig
    properties:
      group: os-apply-config
      config:
        os_net_config:
          network_config:
            -
              type: interface
              name: nic1
              dns_servers: {get_param: DnsServers}
              addresses:
                -
                  ip_netmask:
                    list_join:
                      - '/'
                      - - {get_param: ControlPlaneIp}
                        - {get_param: ControlPlaneSubnetCidr}
              routes:
                -
                  ip_netmask: 169.254.169.254/32
                  next_hop: {get_param: EC2MetadataIp}
            -
              type: ovs_bridge
              name: {get_input: bridge_name}
              members:
                -
                  type: interface
                  name: nic2
                  # force the MAC address of the bridge to this interface
                  primary: true
                -
                  type: vlan
                  vlan_id: {get_param: InternalApiNetworkVlanID}
                  addresses:
                    -
                      ip_netmask: {get_param: InternalApiIpSubnet}
                -
                  type: vlan
                  vlan_id: {get_param: StorageNetworkVlanID}
                  addresses:
                    -
                      ip_netmask: {get_param: StorageIpSubnet}
                -
                  type: vlan
                  vlan_id: {get_param: TenantNetworkVlanID}
                  addresses:
                    -
                      ip_netmask: {get_param: TenantIpSubnet}
                # Uncomment when including environments/network-management.yaml
                #-
                #  type: vlan
                #  vlan_id: {get_param: ManagementNetworkVlanID}
                #  addresses:
                #    -
                #      ip_netmask: {get_param: ManagementIpSubnet}

outputs:
```

```
OS::stack_id:
  description: The OsNetConfigImpl resource.
  value: {get_resource: OsNetConfigImpl}
```

# Configuring Hierarchical Port Binding

This section describes configuring the Single Root I/O Virtualization (SR-IOV) and other VLAN-based ml2 mechanism agents to work with opflex plug-in. This is accomplished by using Hierarchical Port Binding (HPB) and should work without any special modification to the configuration. Here are the basic steps needed to configure opflex with SR-IOV.

When using HPB, datapath connectivity in Cisco Application Centric Infrastructure (ACI) is accomplished by creating static VLAN bindings to the EPGs for networks created by OpenStack. There maybe other configuration required for datapath, for example, setting up VLAN on SR-IOV NIC or configuring OVS (or a load balancer in case of LBaaS). This is done by the third party agent or mechanism driver (e.g. sriovnicswitch).

How to create these assets:

### Before you begin

In order to configure the datapath using static VLAJN bindings. The plug-in requires following assets:

- A physical domain (physdom) with the correct VLAN pool.
- Host-link information (which compute node fabric ethernet interface is connected to which leaf switch port)
- Host-link-network-label information (describing which fabric ethernet interface on compute node is used to serve which physnet)

  This information is only needed if the deployment uses multiple physnets.

### Procedure

**Step 1** Before deploying OpenStack Platform Overcloud, make sure you have one Physical Domain (physdom) created per each physnet required. Add pdom_ prefix to the name of physical domain created. For example for physnet1 create pdom_physnet1, and attach the right VLAN pool.

**Step 2** The host-link information usually comes from LLDP auto discovery. It can also be provided statically using ACIHostLinks parameter in the Cisco ACI deployment template as shown in example below.

You also need to set NeutronNetworkVLANRanges and enable the third party mechanism drivers using ACIMechanismDrivers parameter, make sure that the apic_aim is the last mechanism in the list.

**Example:**

```
NeutronPhysicalDevMappings: physnet1:ens11,physnet2:ens7,physnet3:ens9
NeutronNetworkVLANRanges:physnet1:1200:1250,physnet2:1251:1300,physnet3:1301:1350
ACIMechanismDrivers: 'sriovnicswitch,apic_aim'
ACIHostLinks: '{"101": [{"host01|ens11": "1/14"}], "102": [{"host02|ens9": "1/14"}]}'
```

APPENDIX B

# Configuring UCS B-Series



## Configuring UCS B-Series for Cisco ACI and OpenStack Orchestration

You need three levels on configuration for Cisco Unified Computing System (UCS) B-Series to work with Cisco Application Centric Infrastructure (ACI) and OpenStack orchestration. The first layer is on Cisco UCS, the second on the host, and the third on the leaf switches.

**Note** This document applies to the Cisco UCS B-Series and C-Series servers connected to Fabric Interconnects in UCS mode and provides additional configuration required to install OpenStack on Cisco UCS.

## Configuration on Linux Hosts

Configuration on Linux hosts includes binding the NICs in Active Backup mode, running the BondWatch service, and setting the NIC maximum transmission unit (MTU).

### Bind the NICs

Bind the NICs in Active Backup mode, which you can do by setting the appropriate configuration in your OSP network environment NIC templates.

**Procedure**

Set the appropriate configuration.

**Example:**

```
type: linux_bond
bonding_options: "mode=active-backup miimon=10"
name: bond0
mtu: 1600
members:
 -
  type: interface
  name: nic2
  mtu: 1600
 -
  type: interface
  name: nic3
  primary: true
  mtu: 1600
```

# Run the Bond Watch Service

The bond watch service (`apic-bond-watch`) detects failure of a NIC in the bond and sends gratuitous ARP requests to inform the fabric of the currently active NIC. We recommend that you run the bond watch service on the undercloud.

There are two ways you run the `apic-bond-watch` service, depending on which version of Cisco Application Policy Infrastructure Controller (APIC) that you use:

- **Cisco APIC Release 4.1(x) and earlier:** You perform a short series of steps.
- **Cisco APIC Release 4.2(1) and later:** You set a single parameter, and the `apic-bond-watch` is enabled and started. There are no manual steps that are required to set up, enable, or start the `apic-bond-watch` service.

The following is list of guidelines and recommendations for running the bond watch service:

- Verify that you have installed `/usr/bin/apic-bond-watch`.

  The file is part of the `apicapi` package.
- Add the OpFlex uplink device to `/etc/environments` (opflex_bondif=bond1)

  You must perform this step if the interface is other than default (bond0).
- Enable the bond watch service: `systemctl enable apic-bond-watch`.
- Start the bond watch service: `systemctl start apic-bond-watch`.

**Note** In releases earlier than Cisco Cisco Application Policy Infrastructure Controller (APIC) 4.1(2), you may need to manually run `apic-bond-watch` because the service file may be missing. To manually start the binary, you can use `nohup /usr/bin/apic-bond-watch <interface name>&` as the root user. The default interface name is `bond0`. For example:

```
nohup /usr/bin/apic-bond-watch &          //To use bond0
nohup /usr/bin/apic-bond-watch bond1 &    //To use bond1
```

**Procedure**

**Step 1** Complete one of the following actions, depending on your version of Cisco Application Policy Infrastructure Controller (APIC).

- **Cisco APIC 4.2(1) or later:** Set the parameter ACIEnableBondWatchService to True. See the section "Installing Overcloud" in *Cisco ACI Installation Guide for Red Hat OpenStack Using OpenStack Platform 10 Director* or the section "Parameters for the Cisco ACI Environment" in *Cisco ACI Installation Guide for Red Hat OpenStack Using the OpenStack Platform 13 Director*. Do not complete the remaining steps of this procedure.
- **Cisco APIC 4.1(x) or earlier:** Complete step 2 through step 4 in this procedure.

**Step 2** Create an inventory of all compute node IP addresses.

**Example:**

```
source ~/stackrc
openstack server list --flavor compute -f value -c Networks|cut -d=  -f2 >~/compute-nodes
```

If necessary, you can create an inventory of all controllers:

```
openstack server list --flavor control -f value -c Networks|cut -d= -f2  >>~/compute-nodes
```

**Step 3** Install and enable the service.

**Example:**

```
ansible --become --inventory=compute-nodes all -m shell -u heat-admin -a "yum -y install
apicapi"
```

**Step 4** Start the bond watch service.

**Example:**

```
ansible --become --inventory=compute-nodes all -m shell -u heat-admin -a "systemctl start
apic-bond-watch"
ansible --become --inventory=compute-nodes all -m shell -u heat-admin -a "systemctl enable
 apic-bond-watch"
```

For versions that do not define bond watch, you can start the service manually:

```
ansible --become --inventory=compute-nodes all -m shell -u heat-admin -a 'nohup
/usr/bin/apic-bond-watch&'
```

# Identify Which NIC Is Active in the Bond

The `bond0` file in the `/proc/net/bonding` directory shows which of the two NICs is active.

**Procedure**

Examine the `bond0` to see which NIC is active.

**Example:**

```
[root@overcloud-compute-0 heat-admin]# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)
```

```
Bonding Mode: fault-tolerance (active-backup)
Primary Slave: None
Currently Active Slave: enp13s0
MII Status: up
MII Polling Interval (ms): 1
Up Delay (ms): 0
Down Delay (ms): 0

Slave Interface: enp13s0
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 3
Permanent HW addr: 00:25:b5:00:00:0f
Slave queue ID: 0

Slave Interface: enp14s0
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 3
Permanent HW addr: 00:25:b5:00:00:10
Slave queue ID: 0
[root@overcloud-compute-0 heat-admin]#
```

# Set the NIC MTU

Set and verify the maximum transmission unit (MTU) of the NICs. The MTU is based on settings that you specify in the Cisco Unified Computing System (UCS) Manager.

### Procedure

**Step 1** Set the MTU of the NICs to either 1600 or 9000.

**Step 2** Verify the MTU setting by navigating to the UCS B-Series server, choosing the NIC, and then checking the value in the MTU field.

# Verify MTU Settings for the NICs

Check the maximum transmission unit (MTU) setting on a NIC.

### Procedure

Enter the following command:

```
ip link
```

You should see output similar to the following:

```
5: enp13s0: <BROADCAST,MULTICAST> mtu 9000 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether 00:25:b5:00:00:03 brd ff:ff:ff:ff:ff:ff
```

```
6: enp14s0: <BROADCAST,MULTICAST> mtu 9000 qdisc noop state DOWN mode DEFAULT qlen 1000
    link/ether 00:25:b5:00:00:04 brd ff:ff:ff:ff:ff:ff
```

# Configuration on Cisco UCS

Configure Cisco Unified Computing System (UCS) B-Series properly to integrate it with the Cisco Application Centric Infrastructure (ACI) and OpenStack. A supportable configuration must include the following:

- Configuring the Cisco UCS service profile with two NICs. These NICs are used for OpFlex communication and VM data path.
- Disabling fabric failover on these NICs.
- Connecting the virtual NICs (vNICs) to different fabric interconnects.
- Setting the maximum transmission unit (MTU) on the vNICs.
- Ensuring that the VNICs allow the desired VLANs.
- Turning multicast on for the fabric interconnects.
- Configuring a port channel interface policy on the fabric interconnects.

# Configuration on Leaf Switches

For path redundancy, configure a virtual port channel (vPC) interface policy across the two leaf switches. There are different ways to configure a virtual port channel (vPC), see the *Cisco APIC Layer 2 Networking Configuration Guide* for details.

Regardless of the method you choose to configure the UCS and vPCs, the configuration should resemble the following illustration:

*Figure 5: Configuration on Leaf Switches*