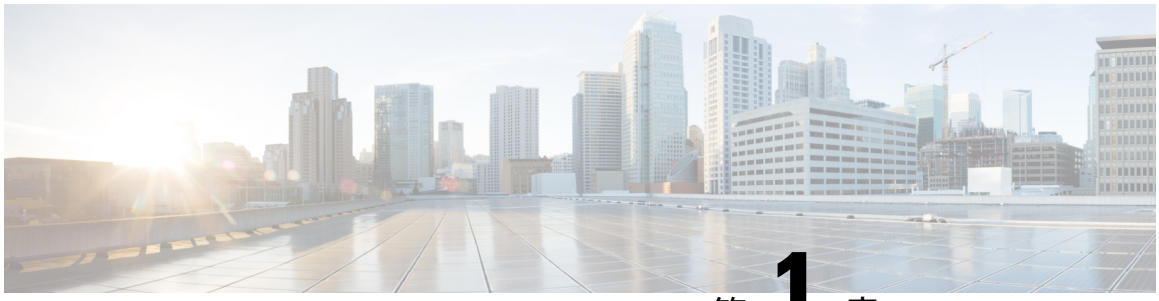
OpenStack Platform 16.2 Director を使用した Red Hat OpenStack の Cisco ACI Installation Guide

初版：2022 年 5 月 20 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

目 次



第 1 章

新機能と更新情報

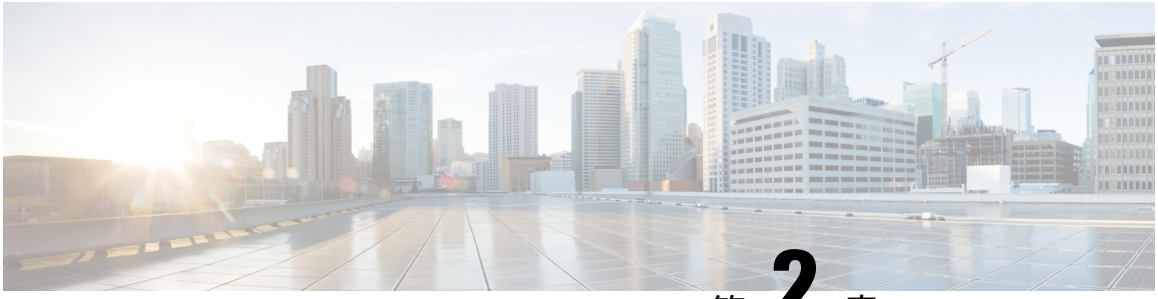
新機能と更新情報

次の表は、この最新リリースに関するマニュアルでの主な変更点の概要を示したものです。ただし、今リリースまでのガイドにおけるすべての変更点や新機能の一部は表に記載されていません。

表 *1*：新機能と変更された動作

OpenStack ACI Unified プラグイン	特長	説明	参照先
5.2(3)	OpenStack 16.2 の Cisco Application Centric Infrastructure（ACI）サポート。	このガイドは、Cisco ACI OpenStack プラグインを使用して Red Hat OpenStack Platform 16.2 をインストールする方法を説明するためにリリースされました。	本書です。

新機能と更新情報



第 2 章

インストール

OpenStack Platform 16.2 Director を使用した OpenStack 対応 Cisco ACI

Cisco Application Centric Infrastructure（ACI）は、インテリジェントなコントローラベースのネットワーク スイッチング ファブリックを提供する包括的なポリシーベースのアーキテクチャです。このファブリックは、OpenStack を含む複数のオーケストレーション ツール、自動化ツール、および管理ツールに直接統合可能な API インターフェイスを通じてプログラムすることにより管理する設計となっています。Cisco ACI を OpenStack と統合することによって、ネットワーキング構造体の動的な作成を OpenStack 要件に従って直接駆動するだけでなく、Cisco Application Policy Infrastructure Controller （APIC）内のさらなる可視性を個別の仮想マシン（VM）インスタンスのレベルに至るまで実現できます。

OpenStack は、クラウドコンピューティング環境を構築するための柔軟なソフトウェア アーキテクチャを明確にします。OpenStack のリファレンス ソフトウェア ベースの実装では、VLAN、GRE、VXLAN を含む複数のレイヤ 2 トランスポートが考慮されています。OpenStack 内の Neutron プロジェクトは、ソフトウェアベースのレイヤ 3 転送も提供できます。Cisco ACI と ACI OpenStack Unified ML2 プラグインを組み合わせて使用すると、統合されたレイヤ 2 およびレイヤ 3 VXLAN ベースのオーバーレイ ネットワーキング機能が提供されます。このアーキテクチャは、ハードウェアベースのネットワーキングのパフォーマンスと運用上の利点に加えて、ソフトウェア オーバーレイ ネットワーキングの柔軟性を提供します。

Cisco ACI OpenStack プラグインは、ML2 モードまたは GBP モードで使用できます。モジュラレイヤ 2（ML2）モードでは、ネットワークの作成に標準 Neutron API が使用されます。これは OpenStack に VM およびサービスを導入するための従来の方法です。グループ ベース ポリ

シー（GBP）モードでは、アプリケーションをポリシーグループとして説明、作成、展開する新しいAPIが提供されます。これにより、ネットワーク固有の詳細を気にする必要がなくなります。単一のOpenStackプロジェクトでGBPとNeutron APIを混在させることはサポートされていないことに注意してください。詳細については、次のURLにある『*OpenStack Group-Based Policy User Guide*』を参照してください。

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

OSP Director を使用した OpenStack での Cisco ACI の要件および前提条件

- 対象者：Linux、Red Hat OpenStack ディストリビューション、Cisco Application Centric Infrastructure（ACI）ポリシーモデル、およびCisco Application Policy Infrastructure Controller（APIC）構成に関する実践的な知識が必要です。また、OpenStackのアーキテクチャと展開に精通している必要があります。
- Cisco ACI ファブリック：『Cisco ACI Virtualization Compatibility Matrix』に記載されている最小サポートバージョンでインストールおよび初期化されたCisco ACIファブリックが必要です。

（注） 複数のリーフペア間の通信の場合、ファブリックには、OpenStack外部ネットワークを使用できるように有効になっているBGPルートリフレクタが必要です。

- 仮想ポートチャネル（vPC）でボンディングされたファブリックインターフェイスを使用する場合、ファブリック インターフェイスの `ovs_bond` の追加はサポートされていません。これは、Open vSwitch（OVS）ブリッジに単一のインターフェイスとして追加する必要があるためです。ファブリックインターフェイスを集約するには、タイプを`linux_bond`に設定する必要があります。`nic-config` テンプレートでファブリック インターフェイスを作成する方法の簡単な例を示します。

```
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
```

- ボンディングを使用する際にサポートされるのは 802.3ad のみです。
- UCS-B シリーズを使用して展開する場合、冗長性を確保するために、ボンディングを備えたデュアル vNIC のみがファブリック インターフェイスでサポートされます。

（注） ハードウェア フェールオーバーで単一の vNIC を使用しないでください。

- Cisco APIC GUI で、ファブリックの OpFlex 認証を無効にします。**[システム（System）]** > **[システム設定（System Settings）]** > **[ファブリック幅設定（Fabric Wide Setting）]** > **[ファブリック幅設定ポリシー（Fabric Wide Setting Policy）]** ウィンドウで、[GOLF と Linux に対して Opflex クライアント証明書の認証を施行する（To enforce Opflex client certificate authentication for GOLF and Linux）] がオフになっていることを確認します。
- オーバークラウド Heat スタックを削除すると、オーバークラウドノードは解放されますが、仮想マシンマネージャ（VMM）ドメインは Cisco APICに残ります。VMM ドメインを手動で削除しない限り、VMM はテナントとともに古い VMM ドメインとして Cisco APIC に表示されます。

  VMM ドメインを削除する前に、スタックがアンダークラウドから削除されていることを確認し、VMM ドメインの下に表示されるハイパーバイザが接続状態でなくなっていることを確認します。これらの条件の両方が満たされたら、Cisco APIC から VMM ドメインを安全に削除できます。

関連資料

詳細については、Red Hat Web サイトにある『*Director Installation and Usage Red Hat OpenStack Platform 16.2*』ドキュメントを参照してください。

OpFlex の展開

このセクションでは、Red Hat OpenStack ディストリビューションに Cisco Application Centric Infrastructure（ACI）OpenStack プラグインをインストールして構成する方法について説明します。

これらの手順の例は、Red Hat OpenStack の OpenStack Platform 16.2 リリースで検証されています。OpenStack システムは、インストール方法によって大きく異なる可能性があります。したがって、ここで示した例は、特定のインストール状態に適応するための基本として使用してください。

Red Hat OpenStack Platform Director のインストール マニュアルに従って、OpenStack Platform Director を用意し、適切な導入ファイルとリソース ファイルを作成してください。

詳細については、このガイドの「関連資料 （5 ページ）」を参照してください。

OpenStack 向け Cisco ACI のインストール準備

Cisco APIC とネットワークの設定

ここでは、Cisco Application Policy Infrastructure Controller（APIC）およびネットワークのセットアップ方法について説明します。

次の図に示すようなネットワーク レイアウトについては、OpenStack Platform Director ドキュメントの「Network Planning」の項を参照してください。詳細については、Red Hat Web サイトにある『*Director Installation and Usage Red Hat OpenStack Platform*』ドキュメントを参照してください。

図 *1 :* 一般的な *OpenStack Platform* トポロジ

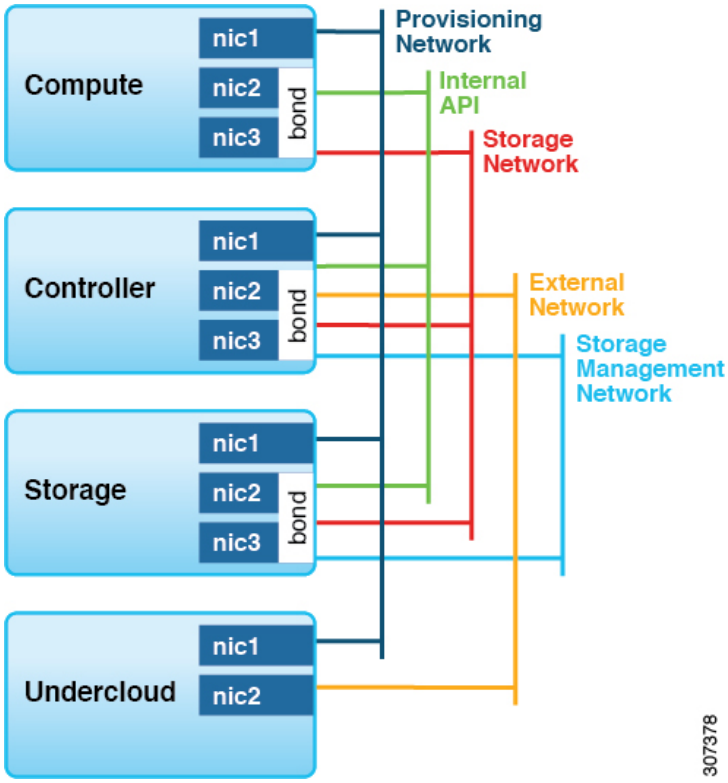
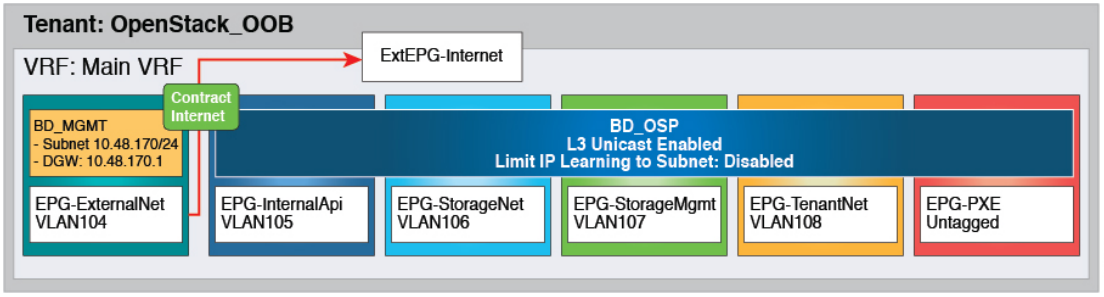


図 *2 : Cisco ACI* プラグインを使用した *Red Hat OpenStack Platform 16.2* のインストールの一般的なトポロジ



- PXE ネットワークはネイティブ VLAN を使用する必要があります。ネイティブ VLAN は通常、OpenStack ノードの専用 NIC として定義されるため、PXE ネットワーク インターフェイスを Cisco Application Centric Infrastructure（ACI）ファブリックまたは別のスイッチング ファブリックに接続できます。
- PXE を除くすべての OpenStack Platform（OSP）ネットワークは、Cisco ACI を介するインバンド（IB）です。次の VLAN はその例です。
  - API：VLAN 10
  - ストレージ ：VLAN 11
  - StorageMgmt：VLAN 12
  - テナント：VLAN 13
  - 外部：VLAN 14
  - Cisco ACI インフラ：VLAN 4093
- この例で使用される ExtEPG-Internet は、OpenStack 外部ネットワークのインターネットへの接続を可能にする L3Out 外部 EPG です。また、要件に応じて、内部 API OpenStack ネットワークに外部接続を提供する必要がある場合もあります。

インバンド構成の Cisco ACI を準備するには、物理ドメインと、これらのネットワーク用に作成された EPG への静的バインドを使用できます。これにより、必要な物理ドメインと接続可能アクセス エンティティ プロファイル（AEP）が作成されます。AEP でインフラ VLAN を有効にする必要があることに注意してください。詳細については、ナレッジベースの記事「特定のポートに *EPG* を導入するためのドメイン、接続エンティティ プロファイル、および *VLAN* の作成」を参照してください。

手順

ステップ 1 Cisco APIC GUI にログインし、OpenStack プラットフォームのインストールに必要な VLAN の VLAN プールを作成します。

a) メニューバーで **[ファブリック（Fabric）]** > **[アクセス ポリシー（Access Policies）]** > **[プール（Pools）]** を選択し、**[VLAN]** を右クリックして VLAN プールを作成します。

b) **[名前（Name）]**フィールドに、ポリシー名を入力します。（たとえば、OSP16.2-infra）。
c) （任意）[Description] フィールドに、VLAN 範囲の名前空間ポリシーの説明を入力します。
d) [Encap Blocks] セクションで [+] アイコンをクリックし、 encap ブロックの範囲を入力します。
e) **[送信（Submit）]** をクリックします。

ステップ 2 接続可能エンティティ プロファイルを作成して上記の物理ドメインを割り当てます。また、[Enable Infra VLAN] が選択されていることを確認します。

a) メニュー バーで、**[ファブリック（Fabric）]** > **[アクセス ポリシー（Access Policies）]** > **[グローバル ポリシー（Global Policyies）]** を選択し、**[接続可能なアクセス エンティティ プロファイル（Attachable Access Entity Profiles）]** を右クリックして、接続可能なアクセス エンティティ プロファイルを作成します。
b) **[名前（Name）]** フィールドに、接続可能アクセス エンティティ プロファイルの名前を入力します。（たとえば、OSP16.2-AEP）。
c) （任意）[Description] フィールドに、接続可能アクセス エンティティ プロファイルの説明を入力します。
d) [Enable Infrastructure VLAN] チェックボックスをオンにしてインフラストラクチャ VLAN を有効にします。
e) [Domains (VMM, Physical or External) To Be Associated To Interfaces:] セクションで [+] アイコンをクリックし、ドロップダウンリストからドメインプロファイルを選択して [Update] をクリックします。
f) [次へ（Next）] をクリックします。
g) **[完了（Finish）]** をクリックします。

ステップ 3 物理ドメインを作成して VLAN プールを割り当てます。

a) メニュー バーで、**[ファブリック（Fabric）]** > **[アクセスポリシー（Access Policies）]** > **[物理ドメインと外部ドメイン（Physical and External Domains）]**を選択し、**[物理ドメイン（Physical Domains）]** を右クリックして、物理ドメインを作成します。
b) **[名前（Name）]** フィールドに、物理ドメインの名前を入力します。（OSP16.2-phys）。
c) [Associated Attachable Entity Profile] フィールドで、関連付けられた接続可能エンティティ プロファイルを選択します。
d) [VLAN Pool] フィールドで VLAN プール（[OSP16.2-infra-dynamic]）を選択します。

   OpenStack ノードと Cisco ACI リーフ スイッチ間のカプセル化方式として VLAN を使用する場合は、OpenStack Neutron ネットワークから使用されるプールに応じて VLAN プール範囲を選択する必要があります。

e) **[送信（Submit）]** をクリックします。

ステップ 4 別のテナントで、[Common] を使用してアプリケーション プロファイルを作成することもできます。（OSP-16.2 など）。OSP ネットワークの EPG、ブリッジ ドメイン、および VRF を作成します。PXE ネットワークも Cisco ACI にアクセスする場合は、PXE の EPG と BD も作成します（この例では省略します）。

ステップ 5 必要な VLAN のスタティック バインディング（パス）を追加します。[:Static Binding Paths] を表示するには、EPG を展開する必要があります。

a) 作成した物理ドメインがこの EPG に接続されていることを確認します。**[アプリケーション プロファイル（Application Profiles）]** > **[EPG]** > **[EPG_name]** > **[Domains]** を使用して物理ドメインを追加できます。

b) メニューバーで、**[テナント（Tenants）]** > **[テナント共通（Tenant common）]** > **[アプリケーションプロファイル（Application Profiles）]** > ***ACI-OSP16.2*** > **Application EPGs）]** > ***EPG API*** > **[静的バインド パス（Static Binding Paths）]** を選択します。

ステップ 6 EPG に物理ドメインが接続されていることを確認します。

（注） Cisco ACI を前述のネットワーク用にプロビジョニングする必要があります（テナント、外部、およびフローティング IP ネットワークを除く）。これには、必要な物理ドメインと接続エンティティプロファイルの作成が含まれます。接続エンティティ プロファイルでインフラ VLAN を有効にする必要があることに注意してください。

Cisco ACI の OpenStack 導入に対する準備はこれで完了です。

オーバークラウドの設定

『*Director Installation and Usage, Red Hat OpenStack Platform 16.2*』ドキュメントに従って、OpenStack Platform 16.2 Director を準備し、正しい展開ファイルとリソース ファイルを作成する必要があります。

詳細については、Red Hat の Web サイトにあるドキュメントを参照してください。第 5 章「コンテナ イメージ ソースの構成」に従う場合は、レジストリ アドレスを書き留めてください。Red Hat のマニュアルに従って、必要に応じてカスタム NIC テンプレートを準備する必要があります。

OpenStack Platform Director を設定したら、展開を続行する前に Cisco Application Centric Infrastructure（ACI）TripleO オーケストレーションをインストールする必要があります。

OpFlex オーケストレーションを備えた Cisco ACI のアンダークラウドの準備

このセクションでは、Cisco Application Centric Infrastructure（ACI）と OpFlex オーケストレーションの統合パッケージをインストールする方法について説明します。

手順

ステップ 1 ユーザ `stack` としてアンダークラウドにログインします。

ステップ 2 Cisco ACI OSP（tripleo-ciscoaci-16）RPM 5.1.3 以降および対応するプラグイン tarball（openstack-ciscorpms-repo-16）を Cisco.com からダウンロードし、OpenStack Platform Director に配置します。

ステップ 3 RPM をインストールします。これによって依存関係がインストールされます。

`rpm` コマンドを使用して RPM をインストールする場合、一部の依存関係は手動によるインストールが必要な場合があります。

例：

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

ステップ 4 次の手順を実行して、Cisco ACI コンテナを作成します。

a) 次のコマンドを実行します。**sudo podman loginregistry.connect.redhat.com**

b) プロンプトが表示されたら、Red Hat のログイン情報を使用して、redhat のユーザー名とパスワードを入力します。

c) ログインしたら、ルートとして次のスクリプトを実行して Cisco ACI コンテナを作成します。

/opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py

d) ダウンロードしたプラグイン tarball をスクリプトで指定します。

例：

```
/opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py -z
/home/stack/openstack-ciscorpms-repo-16-778.tar.gz
```

コマンドは、アップストリームの Red Hat 認定 ACI コンテナ イメージをプルし、ローカル コンテナ リポジトリにプッシュしま
す。/home/stack/templates/ciscoaci_containers.yaml という名前の環境ファイルを作成します。これは、オーバークラウドの展開時にテンプレートとして含める必要があります。-o オプションを使用すると、出力ファイル名を上書きできます。指定したとおりに出力ファイルが作成されたことを確認します。

注

- コンテナをローカルに構築するには、「-p」オプションを省略できます。ただし、これらのコンテナは Red Hat 認定ではない場合があります。コンテナを構築する前に、**registry.redhat.io** にログインする必要があります。
- ローカル コンテナ作成コマンドの実行中に、コマンド **/bin/gbp-db-manage** によって生成されたエラーが表示される場合があります。このエラーは無視しても問題ありません。スクリプトの実行が失敗することはありません。
- OpenStack Director 16.2 の展開では、Docker レジストリの設定がサポートされています。レジストリについては、次の選択肢があります。
  - アップストリーム レジストリ（ローカル サテライト サーバー（現在は Red Hat レジストリ）を使用できます）
  - ダウンストリーム レジストリ アドレス/ポート/URI（現在はアンダーレイ コントローラ、8787、/rhosp16.2）

Docker レジストリは、build_openstack_aci_containers.py スクリプトを使用して構成されます。

```
usage: build_openstack_aci_containers.py [-h] [-u UCLOUD_IP] [-o OUTPUT_FILE]
                                         [-c CONTAINERS_TB]
                                         [-s UPSTREAM_REGISTRY]
                                         [-d DESTINATION_REGISTRY]
                                         [-r REGSEPARATOR] [-i RELEASE_TAG]
                                         [-t TAG]
```

```
                                              [-a [ADDITIONAL_REPOS [ADDITIONAL_REPOS
 ...]]]
                                               [--force] [-p] (-f FILE | -z FILE)

Build containers for ACI Plugin

optional arguments:
  -h, --help            show this help message and exit
  -u UCLOUD_IP, --ucloud_ip UCLOUD_IP
                        Undercloud ip address
  -o OUTPUT_FILE, --output_file OUTPUT_FILE
                        Environment file to create, default is
                        /home/stack/templates/ciscoaci_containers.yaml
  -c CONTAINERS_TB, --container CONTAINERS_TB
                        Containers to build, comma separated, default is all
  -s UPSTREAM_REGISTRY, --upstream UPSTREAM_REGISTRY
                        Upstream registry to pull base images from, eg.
                        registry.access.redhat.com/rhosp13, defaults to
                        registry.access.redhat.com/rhosp13
  -d DESTINATION_REGISTRY, --destregistry DESTINATION_REGISTRY
                        Destination registry to push to, eg:
                        1.100.1.1:8787/rhosp13
  -r REGSEPARATOR, --regseparator REGSEPARATOR
                        Upstream registry separator for images, eg. '/' for
                        normal upstream registrys (default). Will be added
                        between upstream registry name and container name. Use
                        '_' for satellite based registries.
  -i RELEASE_TAG, --image-tag RELEASE_TAG
                        Upstream release tag for images, defaults to 16.2
  -t TAG, --tag TAG     tag for images, defaults to current timestamp
  -a [ADDITIONAL_REPOS [ADDITIONAL_REPOS ...]], --additional-repos
[ADDITIONAL_REPOS [ADDITIONAL_REPOS ...]]
                        Additional repos to use when building containers
                        (defaults to empty list). Use with
                        'rhel-8-for-x86_64-baseos-eus-rpms
                        rhel-8-for-x86_64-appstream-eus-rpms' when using
                        satellite.
  --force               Override check for md5sum mismatch
  -p, --pull            Pull upstream containers instead of building locally
  -f FILE, --aci_repo_file FILE
                        Path to yum repoistory file, which describes the
                        repository which provides ACI plugin rpm files. If you
                        want this script to create a repository on undercloud,
                        please use the -z option to provide path to openstack-
                        aci-rpms-repo tar file downloaded from cisco website
  -z FILE, --aci_rpm_repo_tar_file FILE
                        Path to openstack-aci-rpms-repo tar file. This will be
                        use to create a local yum repository on undercloud
```

オーバークラウドのインストール

ここでは、オーバークラウドのインストール方法について説明します。

手順

ステップ 1 /usr/share/openstack-tripleo-heat-templates/roles_data.yaml ファイルをプライベートな場所にコピーします。

例：

```
cp /usr/share/openstack-tripleo-heat-templates/roles_data.yaml
/home/stack/templates/aci_roles_data.yaml
```

ステップ 2 roles_data.yaml（aci_roles_data.yaml）のローカル コピーを編集して、CiscoAciAIM と CiscoAciLldp サービスをコントローラ ロールに、CiscoAciLldp サービスをコンピューティング ロールに追加します。

a) コントローラ ロールの下に、次の行を追加します。

```
- OS::TripleO::Services::CiscoAciAIM
- OS::TripleO::Services::CiscoAciLldp
- OS::TripleO::Services::CiscoAciOpflexAgent
```

b) コンピューティング ロールの下に、次の行を追加します。

```
- OS::TripleO::Services::CiscoAciLldp
- OS::TripleO::Services::CiscoAciOpflexAgent
```

アップストリームの /usr/share/openstack-tripleo-heat-templates/roles_data.yaml ファイルを変更してこれらのロールを追加する ansible プレイブックが提供されています。上記の手順をスキップし、代わりに ansible-playbook -i ~/inventory.yaml /opt/ciscoaci-tripleo-heat-templates/tools/generate_ciscoaci_role_data.yaml コマンドを使用してプレイブックを実行できます。

ステップ 3 Cisco Application Centric Infrastructure（ACI）環境のリソースを宣言します。

展開に含める .yaml テンプレートファイルで Cisco ACI リソースを定義します。たとえば、/home/stack/templates/aci_cs.yaml などです。この手順では、OpFlex エージェントの使用例のリソース宣言について説明します。

（注）
- 完全なリソース宣言の例については、このガイドの付録の「リソース宣言の例」の項を参照してください。
- OpFlex ではないユース ケース（neutron-openvswitch-agent）のリソース宣言の例については、このガイドの付録の「Neutron OVS エージェントを使用する場合のリソース宣言の例」のセクションを参照してください。
- Cisco ACI 環境に必要なパラメータのリストについては、このガイドの付録の「Cisco ACI 環境のパラメータ」の項を参照してください。

例：

次に、opflex を使用して OSP を展開するためのリソースの例を示します。

```
resource_registry:
```

```
#controller
OS::TripleO::ControllerExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml

  OS::TripleO::Services::NeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Docker::NeutronMl2PluginBase:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron/neutron-ml2-ciscoaci.yaml
  OS::TripleO::Services::CiscoAciAIM:
/opt/ciscoaci-tripleo-heat-templates/deployment/aciaim/cisco-aciaim-container-puppet.yaml

  OS::TripleO::Services::NeutronMetadataAgent:
/usr/share/openstack-tripleo-heat-templates/deployment/neutron/neutron-metadata-container-puppet.yaml

  OS::TripleO::Services::NeutronDhcpAgent:
/usr/share/openstack-tripleo-heat-templates/deployment/neutron/neutron-dhcp-container-puppet.yaml

  #compute
 OS::TripleO::ComputeExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml

  OS::TripleO::Services::ComputeNeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Services::ComputeNeutronMetadataAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/compute_neutron_metadata/compute-neutron-metadata.yaml


  OS::TripleO::Services::CiscoAciOpflexAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/opflex/opflex-agent-container-puppet.yaml

  OS::TripleO::Services::CiscoAciLldp:
/opt/ciscoaci-tripleo-heat-templates/deployment/lldp/cisco_lldp.yaml

  OS::TripleO::Services::OVNDBs: OS::Heat::None
  OS::TripleO::Services::OVNController: OS::Heat::None
  OS::TripleO::Services::OVNMetadataAgent: OS::Heat::None
  OS::TripleO::Services::ComputeNeutronL3Agent: OS::Heat::None
  OS::TripleO::Services::NeutronL3Agent: OS::Heat::None
```

ステップ 4 Cisco ACI 証明書ベースの認証を使用するには、X.509 証明書を持つローカル ユーザーを作成し、パラメータ ACIApicPrivateKey および ACIApicCertName を使用して、Cisco ACI リソース ファイルで証明書とキーを指定します。

『*Cisco APIC Security Configuration Guide, Release 5.1(x)*』の「Creating a Local User and Adding a User Certificate」の項を参照してください。

（注） 証明書ベースの認証を使用する場合は、`ACIApicPassword`パラメータを指定しないでください。

ステップ 5 オーバークラウドを展開します。

オーバークラウドを展開する場合は、`-r`オプションを使用して作成したカスタム ロールのデータ ファイルを含めます。サイト固有の環境ファイルに加えて、Cisco ACI環境ファイルとCisco ACI コンテナ YAML ファイルも環境リストに含めます。

例：

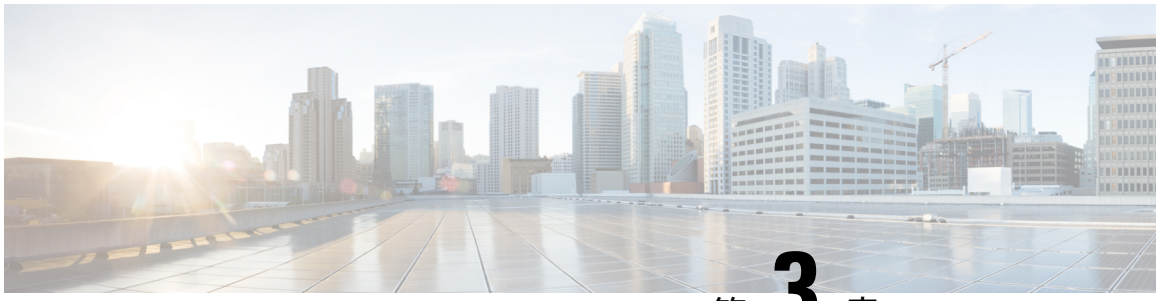
```
openstack overcloud deploy --templates /home/stack/tripleo-heat-templates -r
/home/stack/templates/aci_roles_data.yaml  -e
/home/stack/tripleo-heat-templates/environments/network-isolation.yaml -e
```

```
/home/stack/templates/overcloud_images.yaml -e
/home/stack/templates/network-environment.yaml -e
/home/stack/templates/ciscoaci_containers.yaml -e /home/stack/templates/ciscoaci-env.yaml
 -e /home/stack/templates/rhel-registration-resource-registry.yaml -e
/home/stack/templates/environment-rhel-registration
```

上記の例は、Cisco ACI テンプレートとロールの使用方法を示しています。他のテンプレートは、インストール構成によって異なる場合があります。カスタムテンプレートの作成とネットワーク環境テンプレートの自動生成については、Red Hat のガイドラインに従ってください。

参照

- Red Hat Web サイトでの Red Hat OpenStack プラットフォームの Director のインストールと使用方法。
- Cisco.com の特定のポートに *EPG* を展開するためのドメイン、接続エンティティプロファイル、および *VLAN* の作成のナレッジベースの記事。
- 「Cisco ACI ファブリックの初期化の例」も参照してください。



第 3 章

Cisco ACI および OSP のアップグレード

Cisco APIC および OSP のアップグレードに関するガイドライン

OpenStack プラグインは Cisco Application Policy Infrastructure Controller APIC リリースでリリースされるため、 Cisco APIC と同じセマンティック バージョンを使用します。たとえば、5.1(1) プラグインは Cisco APIC 5.1(1) リリースで提供されます。通常、OpenStack プラグインの Cisco APIC リリースは、一致するリリースおよび以前の長期サポート（LTS）Cisco APIC リリースに対してテストされます。ただし、特定のプラグインリリースは、追加の Cisco APIC リリースと互換性がある場合があります。使用されているプラグインのバージョンが Cisco APIC のバージョンと互換性があることを確認するには、『*Cisco ACI Virtualization Compatibility Matrix*』を参照してください。

互換性のある Cisco APIC と Red Hat OSP リリースについては、『*Cisco ACI Virtualization Compatibility Matrix*』を参照してください。

アップグレード前のガイドライン

Cisco Application Centric Infrastructure（ACI）プラグインをアップグレードします。

さまざまな OpenStack バージョンを使用したプラグインの互換性に関する詳細は、「*Cisco ACI 仮想化互換性マトリックス*」 を参照してください。

アップグレードのガイドライン

Cisco Application Centric Infrastructure（ACI）ファブリックは、『*Cisco APIC Installation, Upgrade, and Downgrade Guide*』の情報に従ってアップグレードできます。

オプションで、Cisco ACIプラグインとCisco ACIファブリック リリースの組み合わせがサポートされている限り、プラグインをアップグレードせずにCisco ACIファブリックをアップグレードできます。詳細については、次の URL にある『*Cisco ACI Virtualization Compatibility Matrix*』を参照してください。

アップグレード後のガイドライン

Cisco Application Centric Infrastructure（ACI）ファブリックをアップグレードした後、必要に応じて、OpenStack Cisco ACI パッケージをアップグレードした Cisco ACI ファブリック コード以下のバージョンにアップグレードできます。特定の情報については、Cisco.com の OpenStack Cisco ACI プラグインのリリースノートも参照してください。

OpenStack Cisco ACI プラグインをアップグレードする方法の詳細については、このガイドの Cisco ACI パッケージのアップグレード （16 ページ） を参照してください。

Cisco ACI パッケージのアップグレード

次の手順では、Cisco Application Centric Infrastructure（ACI）プラグインの新しいバージョンで完全に展開されたオーバークラウドを更新します。アップグレードはライブで実行できます。

（注） Red Hat Director のドキュメントに従って、ステップ 4 のプラグインをアップグレードします。

手順

ステップ 1 更新されたバージョンの `tripeo-ciscoaci-version` RPMと対応するプラグイン tarball（`openstack-ciscorpms-repo`）を Cisco.com から OSP Director にコピーします。

ステップ 2 次のとおり、`yum`を使用して、`tripeo-ciscoaci-version` パッケージを更新します。**`yum update tripleo-ciscoaci-`**

ステップ 3 最初に **sudo podman login registr.connect.redhat.com** を使用して**registry.connect.redhat.com** にログインし、次に `/opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py` を実行して、 コマンドを実行して Cisco ACI コンテナを作成します。ダウンロードしたプラグイン tarball を指すようにします。次に例を示します。

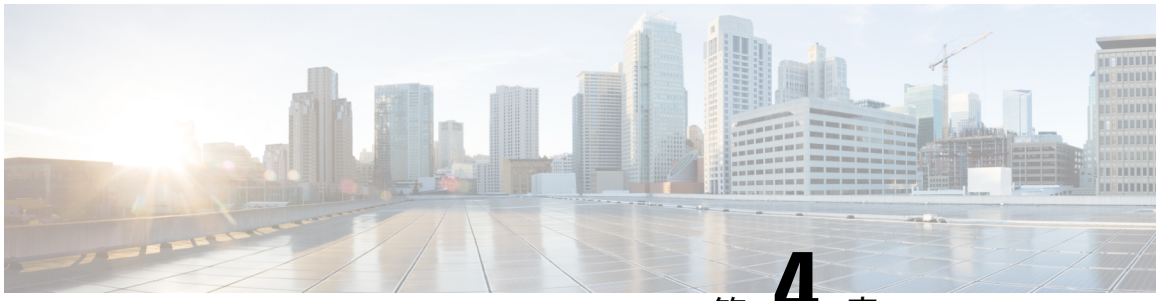
```
opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py -p
-z/home/stack/openstack-ciscorpms-repo-163.0-848.tar.gz
```

コマンドは、Docker イメージを更新
し、`/home/stack/templates/cisco_containers.yaml` ファイルを更新します。

（注） コンテナをローカルに構築するには、「-p」オプションを省略できます。これらのローカルで構築されたコンテナは、Red Hat 認定ではない場合があります。コンテナを構築する前に、**registry.redhat.io** にログインします。

ステップ 4 Red Hat カスタマー ポータルの Red Hat アップグレード手順に従います。

『*Keeping Red Hat OpenStack Platform Updated*』の第 4 章「Updating the Overcloud」を参照してください。**製品およびサービスの** > **製品ドキュメント** > **Red Hat OpenStack Platform** > **16.2** に移動します。



第 4 章

OpenStack 外部ネットワークの追加

OpenStack 外部ネットワークの追加

このセクションでは、OpenStack 外部ネットワークを追加する方法について説明します。

（注） ネットワーク構成とインスタンスを作成するプロジェクトの keystone ファイルを取得して、この手順のコマンドを実行します。

始める前に

OpenStack 外部ネットワークを追加する前に、以下を実行しておく必要があります。

- Cisco Application Centric Infrastructure（ACI）でレイヤ 3 外部接続（L3Out）を作成しました。

  L3Out は、OpenStack で作成されたテナント（OpenStack テナントの専用 L3out）または Common テナント（複数の OpenStack テナントにまたがる共有 L3out）で構成することができます。この手順では、*l3out1* という専用の L3out が OpenStack テナントで構成されていることを前提としています。

- L3Out で次を指定します。
  - インターフェイスとその IP アドレス情報。
  - 使用されたダイナミック ルーティング。
  - 外部エンドポイント グループ（EPG）。

    この手順では、*extEpg* という名前の外部 EPG を使用します。

（注） コントラクトを追加しないでください。プラグインによって自動的に追加されます。

重要 送信元ネットワーク アドレス変換（SNAT）またはフローティング IP（FIP）アドレスが必要な場合は、OpenStack によって作成された VRF とは異なる VRF で L3Out を定義する必要があります。

手順

ステップ 1 Neutron 外部ネットワークを作成し、L3Out の識別名を指定します。

例：

```
neutron net-create network_name --router:external --apic:distinguished_names type=dict
ExternalNetwork=uni/tn-ACI_tenant_name/out-ACI_L3out_name/instP-ACI_externalEPG_name
(--apic:nat_type "")
```

`--apic:nat_type ""` はオプションです。特定の外部 Neutron ネットワークに NAT を使用しない場合にのみ使用します。

次に、NAT を有効にした外部ネットワークの作成例を示します。

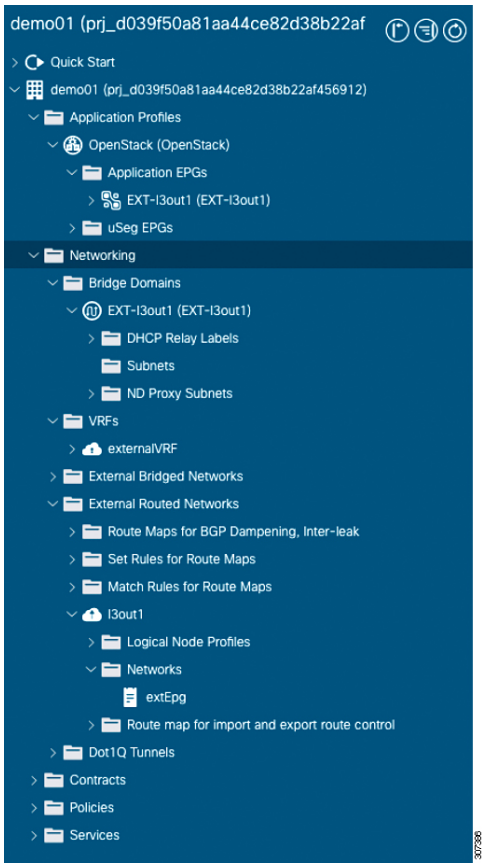
```
neutron net-create external-net-dedicated --router:external --apic:distinguished_names
type=dict ExternalNetwork=uni/tn-prj_$demo01/out-l3out1/instP-extEpg
Created a new network:
+-----------------------------------+-----------------------------------------------------+
| Field                             | Value
      |
+-----------------------------------+-----------------------------------------------------+
| admin_state_up                    | True
      |
| apic:bgp_asn                      | 0
      |
| apic:bgp_enable                   | False
      |
| apic:bgp_type                     | default_export
      |
| apic:distinguished_names          | {"EndpointGroup":
"uni/tn-prj_cdeda9c674a94394a09e86a2fea498c2/ap-OpenStack/epg-EXT-l3out1",
"ExternalNetwork": "uni/tn-prj_cdeda9c674a94394a09e86a2fea498c2/out-l3out1/instP-extEpg",

"VRF": "uni/tn-prj_cdeda9c674a94394a09e86a2fea498c2/ctx-externalVRF", "BridgeDomain":
"uni/tn-prj_cdeda9c674a94394a09e86a2fea498c2/BD-EXT-l3out1"}
      |
| apic:external_cidrs               | 0.0.0.0/0
      |
| apic:nat_type                     | distributed
      |
| apic:nested_domain_allowed_vlans  |
      |
| apic:nested_domain_infra_vlan     |
      |
| apic:nested_domain_name           |
      |
| apic:nested_domain_node_network_vlan |
```

OpenStack 外部ネットワークの追加

```
        |
| apic:nested_domain_service_vlan      |
        |
| apic:nested_domain_type              |
        |
| apic:svi                             | False
        |
| apic:synchronization_state           | build
        |
| availability_zone_hints              |
        |
| availability_zones                   |
        |
| created_at                           | 2019-05-22T13:38:32Z
        |
| description                          |
        |
| id                                   | 635623ed-5dba-42ec-b3f8-3cff18f925c6
        |
| ipv4_address_scope                   |
        |
| ipv6_address_scope                   |
        |
| is_default                           | False
        |
| mtu                                  | 9000
        |
| name                                 | external-net-dedicated
        |
| port_security_enabled                | True
        |
| project_id                           | cdeda9c674a94394a09e86a2fea498c2
        |
| provider:network_type                | opflex
        |
| provider:physical_network            | physnet1
        |
| provider:segmentation_id             |
        |
| revision_number                      | 6
        |
| router:external                      | True
        |
| shared                               | False
        |
| status                               | ACTIVE
        |
| subnets                              |
        |
| tags                                 |
        |
| tenant_id                            | cdeda9c674a94394a09e86a2fea498c2
        |
| updated_at                           | 2019-05-22T13:38:33Z
        |
+--------------------------------------+------------------------------------------------------+
```

Cisco ACI では、Cisco Application Policy Infrastructure Controller（APIC）GUI の次のスクリーンキャプチャに示すように、コマンドは新しい EPG（*EXT-l3out1*）と新しいブリッジ ドメイン（*EXT-l3out1*）を作成します。

OpenStack 外部ネットワークの追加



ステップ 2 SNAT とフローティング IP アドレスに使用される Neutron サブネットを作成します。

この手順は、Neutron 外部ネットワークの作成時に `--apic:nat_type ""` を使用した場合は必要ありません（NAT が無効になっているため）。

例：

```
neutron subnet-create net_name subnet/mask --name subnet_name --disable-dhcp --gateway
gateway_ip --apic:snat_host_pool True
```

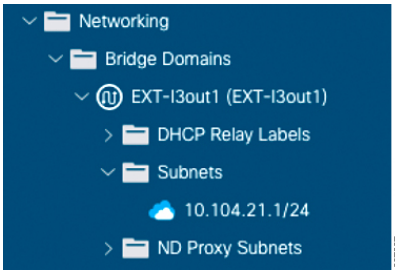
OpFlex エージェントは、サブネットからコンピューティングノードごとに 1 つの IP アドレスを自動的に割り当てます。外部に接続する仮想マシン（VM）は、フローティング IP アドレスが割り当てられていない限り、この IP アドレス（1 対多 NAT）を使用します。

次に、NAT が有効になっている外部ネットワークの作成例を示します。

```
neutron subnet-create external-net-dedicated 10.104.21.0/24 --name ext-subnet
--disable-dhcp --gateway 10.104.21.1 --apic:snat_host_pool True
Created a new subnet:
+---------------------------+-------------------------------------------------+
| Field                     | Value                                           |
+---------------------------+-------------------------------------------------+
| allocation_pools          | {"start": "10.104.21.2", "end": "10.104.21.254"} |
| apic:distinguished_names  | {}                                              |
| apic:snat_host_pool       | True                                            |
| apic:synchronization_state | N/A                                             |
| cidr                      | 10.104.21.0/24                                  |
| created_at                | 2019-05-22T13:38:35Z                            |
| description               |                                                 |
```

```
| dns_nameservers           |                                                    |
| enable_dhcp               | False                                              |
| gateway_ip                | 10.104.21.1                                        |
| host_routes               |                                                    |
| id                        | 238aa55d-1537-4f01-86c9-5f6fc4bde625               |
| ip_version                | 4                                                  |
| ipv6_address_mode         |                                                    |
| ipv6_ra_mode              |                                                    |
| name                      | ext-subnet                                         |
| network_id                | 635623ed-5dba-42ec-b3f8-3cff18f925c6               |
| project_id                | cdeda9c674a94394a09e86a2fea498c2                   |
| revision_number           | 0                                                  |
| service_types             |                                                    |
| subnetpool_id             |                                                    |
| tags                      |                                                    |
| tenant_id                 | cdeda9c674a94394a09e86a2fea498c2                   |
| updated_at                | 2019-05-22T13:38:35Z                               |
+---------------------------+----------------------------------------------------+
```

SNAT サブネットを作成すると、次のCisco APIC GUI のスクリーン キャプチャに示すように、ブリッジ ドメインの下に新しいサブネットが生成されます。



ステップ 3 （オプション）1 つ以上のフローティング サブネットを外部 Neutron ネットワークに割り当てます。

例：

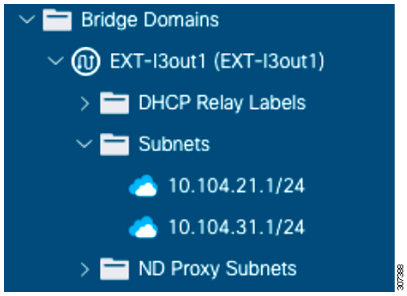
```
neutron subnet-create net_name fip_subnet/mask --name subnet_name --allocation-pool
start=start_ip,end=end_ip --disable-dhcp --gateway gateway_ip
```

次の Cisco APIC GUI の出力とスクリーン キャプチャは、フローティング IP サブネットの作成例を示しています。

```
neutron subnet-create external-net-dedicated 10.104.31.0/24 --name ext-subnet-FIP
--allocation-pool start=10.104.31.10,end=10.104.31.100 --disable-dhcp --gateway 10.104.31.1
Created a new subnet:
+---------------------------+----------------------------------------------------+
| Field                     | Value                                              |
+---------------------------+----------------------------------------------------+
| allocation_pools          | {"start": "10.104.31.10", "end": "10.104.31.100"}  |
| apic:distinguished_names  | {}                                                 |
| apic:snat_host_pool       | False                                              |
| apic:synchronization_state| N/A                                                |
| cidr                      | 10.104.31.0/24                                     |
| created_at                | 2019-05-22T13:38:38Z                               |
| description               |                                                    |
| dns_nameservers           |                                                    |
| enable_dhcp               | False                                              |
| gateway_ip                | 10.104.31.1                                        |
| host_routes               |                                                    |
| id                        | 107c2714-2ace-44a7-9cb0-1a7f40ba2833               |
| ip_version                | 4                                                  |
| ipv6_address_mode         |                                                    |
```

OpenStack 外部ネットワークの追加

```
| ipv6_ra_mode              |                                                      |
| name                      | ext-subnet-FIP                                       |
| network_id                | 635623ed-5dba-42ec-b3f8-3cff18f925c6                 |
| project_id                | cdeda9c674a94394a09e86a2fea498c2                     |
| revision_number           | 0                                                    |
| service_types             |                                                      |
| subnetpool_id             |                                                      |
| tags                      |                                                      |
| tenant_id                 | cdeda9c674a94394a09e86a2fea498c2                     |
| updated_at                | 2019-05-22T13:38:38Z                                 |
+---------------------------+------------------------------------------------------+
```

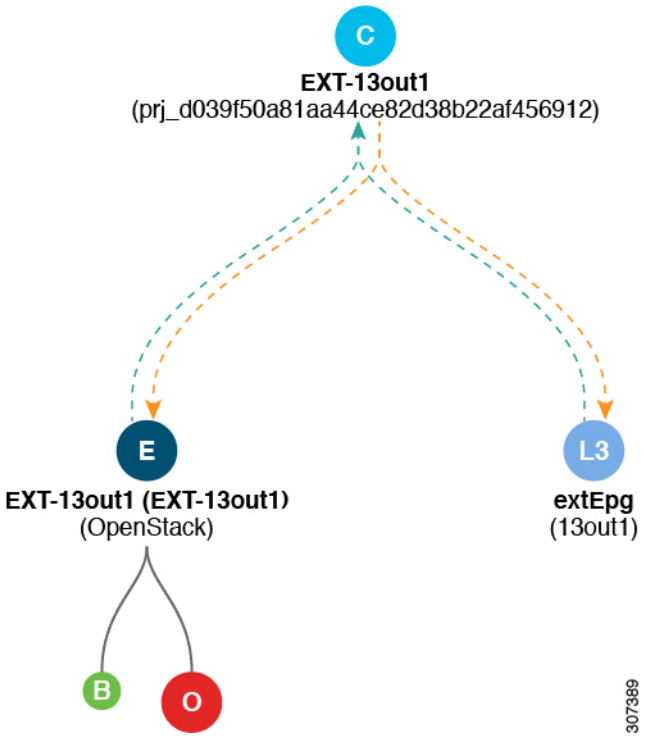


ステップ 4 Neutron 外部ネットワークを 1 つの OpenStack ルータにゲートウェイとして接続します。

例：

```
openstack router set --external-gateway external_net_name router_name
```

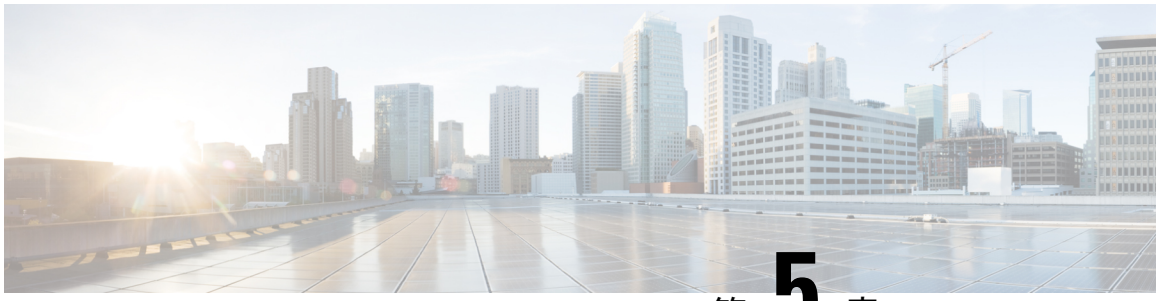
コマンドは、次の図に示すように、外部 Neutron ネットワークの OpenStack ルータに接続されたテナント ネットワークの外部接続を許可するコントラクトを作成します。



OpenStack 外部ネットワークの追加

次のタスク

OpenStack 外部ネットワークの追加



第 5 章

インプレース アップグレード

Red Hat OSP13 から OSP16 へのインプレース アップグレードは、Cisco ACI OpenStack Plug-in 5.2(1) 以降でサポートされています。アップグレードプロセスは、『*FRAMEWORK FOR UPGRADES (13 TO 16.2)* Red Hat』ガイドで説明されている手順に関連しており、主にこの手順に基づいています。

この章で説明する各手順は必須であり、アップグレードに必要です。『*FRAMEWORK FOR UPGRADES (13 TO 16.2)*』ガイドの関連手順が示されています。

カスタム ACI リポジトリの削除

カスタム ACI リポジトリを削除するには、次の手順を実行します。

始める前に

『*FRAMEWORK FOR UPGRADES (13 TO 16.2)* Red Hat』ガイドの「構成可能なサービスとパラメータの更新」セクションまでの手順に従います。

手順

ステップ 1 次の Ansible プレイブックを作成して、オーバークラウドノードの現在の yum リポジトリから ACI リポジトリを削除します。

```
---
- name: Remove ACI Repo
  hosts: overcloud
  become: yes
  tasks:
```

```
    - name: remove_acirepo
      ansible.builtin.file:
        path: /etc/yum.repos.d/ciscoaci.repo
        state: absent
```

ステップ 2 アンダークラウドからプレイブックを実行します。

```
ansible-playbook -i ~/inventory.yaml <name of playbook file>
```

ロールのカスタマイズ

ロールをカスタマイズするには、次の手順を使用します。

『*FRAMEWORK FOR UPGRADES (13 TO 16.2)*』ガイドの「*Updating composable services in custom routes_data files*」セクションで、カスタム ロール テンプレートを使用してシスコの構成可能サービスを追加します。アップストリームの `/usr/share/openstack-tripleo-heat-templates/roles_data.yaml` ファイルを変更してこれらのロールを追加する ansible プレイブックが提供されています。

手順

次のコマンドを使用してプレイブックを実行します。

```
ansible-playbook -i ~/inventory.yaml
/opt/ciscoaci-tripleo-heat-templates/tools/generate_ciscoaci_role_data.yaml
```

これにより、`/home/stack/templates` ディレクトリに新しい custom_roles_data.yaml ファイルが作成されます。

カスタム ロールファイルを使用している場合は、上記の手順の代わりに、コントローラおよびコンピューティング ロールにサービスを追加する必要があります。

コントローラ ロールに次のサービスを追加します。

```
OS::TripleO::Services::CiscoAciAIM
OS::TripleO::Services::CiscoAciLldp
OS::TripleO::Services::CiscoAciOpflexAgent
```

コンピューティング ロールに次のサービスを追加します。

```
OS::TripleO::Services::CiscoAciLldp
OS::TripleO::Services::CiscoAciOpflexAgent
```

Open vSwitch の互換性

『*FRAMEWORK FOR UPGRADES (13 TO 16.2)* Red Hat』ガイドの「アップグレード中の *Open vSwitch* 互換性の維持」セクションをスキップしてください。次に、シスコ固有の設定環境 yaml ファイル（`ciscoaci-config.yaml`）の例を示します。

```
# A Heat environment file which can be used to enable a
# a Neutron Cisco Aci backend on the controller, configured via puppet
resource_registry:

  #controller
  OS::TripleO::ControllerExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml

  OS::TripleO::Services::NeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Docker::NeutronMl2PluginBase:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron/neutron-ml2-ciscoaci.yaml
  OS::TripleO::Services::CiscoAciAIM:
/opt/ciscoaci-tripleo-heat-templates/deployment/aciaim/cisco-aciaim-container-puppet.yaml

  OS::TripleO::Services::NeutronMetadataAgent:
/usr/share/openstack-tripleo-heat-templates/deployment/neutron/neutron-metadata-container-puppet.yaml

  OS::TripleO::Services::NeutronDhcpAgent:
/usr/share/openstack-tripleo-heat-templates/deployment/neutron/neutron-dhcp-container-puppet.yaml

  #compute
  OS::TripleO::ComputeExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml

  OS::TripleO::Services::ComputeNeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Services::ComputeNeutronMetadataAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/compute_neutron_metadata/compute-neutron-metadata.yaml


  OS::TripleO::Services::CiscoAciOpflexAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/opflex/opflex-agent-container-puppet.yaml

  OS::TripleO::Services::CiscoAciLldp:
/opt/ciscoaci-tripleo-heat-templates/deployment/lldp/cisco_lldp.yaml

  OS::TripleO::Services::OVNDBs: OS::Heat::None
  OS::TripleO::Services::OVNController: OS::Heat::None
  OS::TripleO::Services::OVNMetadataAgent: OS::Heat::None
  OS::TripleO::Services::ComputeNeutronL3Agent: OS::Heat::None
  OS::TripleO::Services::NeutronL3Agent: OS::Heat::None

parameter_defaults:

  EC2MetadataIp: 1.100.1.1
  ControlPlaneDefaultRoute: 1.100.1.1
  OvercloudControllerFlavor: control
  OvercloudComputeFlavor: compute

  DockerInsecureRegistryAddress: ["ostack-pt-1-s1-ucloud-13.ctlplane.localdomain:8787",
 "1.100.1.1:8787"]

  NeutronCorePlugin: 'ml2plus'
```

```
  NeutronServicePlugins: 'group_policy,ncp,apic_aim_l3'
  NeutronEnableIsolatedMetadata: true
  NeutronEnableForceMetadata: true
  NeutronPhysicalDevMappings: physnet1:eth1,physnet2:eth2
  EnablePackageInstall: true
  ACIScopeNames: true
  ACIApicHosts: 10.30.120.190
  ACIApicUsername: admin
  ACIApicPassword: noir0123
  ACIApicSystemId: ostack-pt-1-s1
  ACIMechanismDrivers: 'apic_aim'
  ACIApicEntityProfile: sauto_ostack-pt-1-s1_aep
  ACIApicInfraVlan: 3701
  ACIApicInfraSubnetGateway: 10.0.0.30
  ACIApicInfraAnycastAddr: 10.0.0.32
  ACIOpflexUplinkInterface: bond1
  ACIYumRepo: http://1.100.1.1:8787/v2/__acirepo

  ACIOpflexEncapMode: vxlan
  NeutronNetworkVLANRanges: physnet1:1751:1800
  ACIOpflexVlanRange: 751:800
  HeatEnginePluginDirs:
/usr/lib64/heat,/usr/lib/heat,/usr/local/lib/heat,/usr/local/lib64/heat,/usr/lib/python2.7/site-packages/gbpautomation/heat

  ACIVpcPairs: 101:102
  NeutronPluginMl2PuppetTags: 'neutron_plugin_ml2,neutron_plugin_cisco_aci'
  AciVmmMcastRanges: 225.2.1.1:225.2.255.255
  AciVmmMulticastAddress: 225.2.10.3

#Below parameters are only needed when installing Openshift on Openstack
  ACIOpflexInterfaceType: 'ovs'
  ACIOpflexInterfaceMTU: 8000
```

Cisco コンテナの構築

Cisco コンテナを構築するには、次の手順を使用します。

『*FRAMEWORK FOR UPGRADES (13 TO 16.2)* Red Hat』ガイドの「標準オーバークラウドのアップグレード」セクションに進む前に、シスコ固有のコンテナを構築する必要があります。

手順

ステップ 1 アンダークラウドから OSP13 Cisco Tripleo パッケージを削除し、新しい OSP16 RPM をインストールします。

例：`tripleo-ciscoaci-16.2-1054.noarch.rpm` をインストールするには、次のコマンドを使用します。

```
sudo yum remove tripleo-ciscoaci
sudo yum install ./tripleo-ciscoaci-16.2-1054.noarch.rpm
```

ステップ 2 アップストリーム コンテナ レジストリにログインします。たとえば、Red Hat のアップストリーム コンテナ レジストリを使用している場合は、次のコマンドを実行します。

```
sudo podman login registry.redhat.io
```

ステップ 3 アップストリーム コンテナ レジストリにログインした後、次を使用して OSP16 の ACI コンテナ ビルド スクリプトを実行します。

```
sudo /opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py -z
openstack-ciscorpms-repo-16.2-1006.tar.gz
```

このスクリプトは、`/home/stack/templates/ciscoaci_containers.yaml` ファイルを作成します。このファイルは、シスコ固有のアップストリーム サービスまたは変更されたアップストリーム サービスのコンテナ イメージへのマッピングを提供します。

ステップ 4 OSP16 のコンテナを構築した後、次のコマンドを使用して移行（Stein）コンテナを構築します。

```
sudo /opt/ciscoaci-tripleo-heat-templates/tools/build_transitional_aci_containers.py -z
 openstack-ciscorpms-repo-15.0-995.tar.gz --force
```

このスクリプトは、OSP15 用の ACI 移行コンテナを構築
し、`/home/stack/templates/ciscoaci_containers_stein.yaml` ファイル内のコンテナイメージへのシスコ固有のサービスのマッピングを提供します。

アップグレードの準備

『*FRAMEWORK FOR UPGRADES (13 TO 16.2)* Red Hat』ガイドの「*Running the overcloud upgrade preparation*」セクションに、Cisco ACI 統合に固有のファイルが含まれていることを確認します。その内容は次のとおりです。

- カスタム ロール ファイル
- Cisco ACI OSP16 コンテナ マッピング Heat 環境ファイル
- Cisco ACI OSP15 コンテナ マッピング Heat 環境ファイル
- Cisco ACI 固有の構成環境ファイル

次に、シスコ固有のテンプレートを使用した **upgrade prepare** コマンドの例を示します。

```
source ~/stackrc
openstack overcloud upgrade prepare \
            --templates /home/stack/tripleo-heat-templates \
            -r /home/stack/templates/custom_roles_data.yaml \
            -e /home/stack/tripleo-heat-templates/environments/network-isolation.yaml \
            -e /home/stack/templates/containers-prepare-parameter.yaml \
            -e /home/stack/templates/network-environment.yaml \
            -e /home/stack/templates/ciscoaci_containers.yaml \
            -e /home/stack/templates/ciscoaci_containers_stein.yaml \
            -e /home/stack/templates/ciscoaci-config.yaml \
            -e /home/stack/templates/rhsm.yaml \
            -e /home/stack/templates/upgrades-environment.yaml -y
```

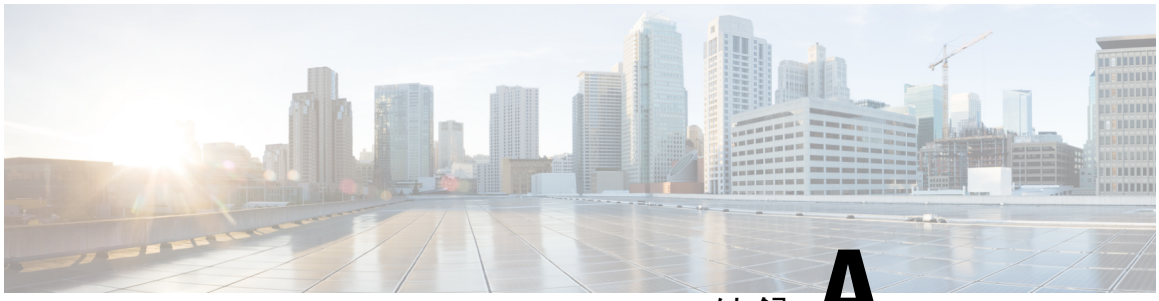
アップグレードの収束

『*FRAMEWORK FOR UPGRADES*（*13 TO 16.2*）Red Hat』ガイドの「*Synchronizing the overcloud stack*」セクションに、Cisco ACI 統合に固有のファイルが含まれていることを確認します。その内容は次のとおりです。

- カスタム ロール ファイル
- Cisco ACI OSP16 コンテナ マッピング Heat 環境ファイル
- Cisco ACI OSP15 コンテナ マッピング Heat 環境ファイル
- Cisco ACI 固有の構成環境ファイル

次に、Cisco 固有の yaml ファイルを使用した **upgrade converge** コマンドの例を示します。

```
source ~/stackrc
openstack overcloud upgrade converge \
            --templates /home/stack/tripleo-heat-templates \
            -r /home/stack/templates/custom_roles_data.yaml \
            -e /home/stack/tripleo-heat-templates/environments/network-isolation.yaml \
            -e /home/stack/templates/containers-prepare-parameter.yaml \
            -e /home/stack/templates/network-environment.yaml \
            -e /home/stack/templates/ciscoaci_containers.yaml \
            -e /home/stack/templates/ciscoaci_containers_stein.yaml \
            -e /home/stack/templates/ciscoaci-config.yaml \
            -e /home/stack/templates/rhsm.yaml \
            -e /home/stack/templates/upgrades-environment.yaml -y
```



付録 A

UCS B シリーズの構成

Cisco ACI および OpenStack オーケストレーション用の UCS B シリーズの構成

Cisco Unified Computing System（UCS）B シリーズを Cisco Application Centric Infrastructure（ACI）および OpenStack オーケストレーションと連携させるには、3 つのレベルの設定が必要です。最初のレイヤは Cisco UCS 上にあり、2 番目はホスト上にあり、3 番目はリーフ スイッチ上にあります。

（注） このドキュメントは、UCS モードのファブリック インターコネクトに接続された Cisco UCS B シリーズおよび C シリーズ サーバーに適用され、Cisco UCS に OpenStack をインストールするために必要な追加設定について説明します。

Linux ホストでの構成

Linux ホストでの設定には、Active Backup モードでの NIC のバインド、BondWatch サービスの実行、NIC 最大伝送ユニット（MTU）の設定が含まれます。

NIC のバインド

Active Backup モードで NIC をバインドします。これを行うには、OSP ネットワーク環境の NIC テンプレートで適切な構成を行います。

手順

適切な構成を行います。

例：

```
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
```

ボンド ウォッチ サービスの実行

ボンドウォッチサービス（apic-bond-watch）は、ボンド内のNICの障害を検出し、Gratuitous ARP 要求を送信して、現在アクティブな NIC をファブリックに通知します。アンダークラウドでボンドウォッチサービスを実行することをお勧めします。

使用する Cisco Application Policy Infrastructure Controller（APIC）のバージョンに応じて、apic-bond-watch サービスを実行する方法は 2 つあります。

- **Cisco APIC リリース 4.1(x) 以前**：短い一連の手順を実行します。
- **Cisco APIC リリース 4.2(1) 以降**：単一のパラメータを設定すると、apic-bond-watch が有効になり、開始されます。apic-bond-watch サービスを設定、有効化、または開始するために必要な手動の手順はありません。

以下は、ボンド ウォッチ サービスを実行するためのガイドラインと推奨事項のリストです。

- /usr/bin/apic-bond-watch がインストールされていることを確認します。

  ファイルは apicapi パッケージの一部です。

- OpFlex アップリンク デバイスを /etc/environments（opflex_bondif=bond1）に追加します

  インターフェイスがデフォルト（bond0）以外の場合は、この手順を実行する必要があります。

- 次のボンドウォッチサービスを有効にします。systemctl enable apic-bond-watch。
- 次のボンドウォッチサービスを開始します。systemctl start apic-bond-watch。

UCS B シリーズの構成

（注） Cisco Cisco Application Policy Infrastructure Controller（APIC）4.1(2) より前のリリースでは、サービスファイルが欠落している可能性があるため、手動で apic-bond-watch を実行する必要がある場合があります。バイナリを手動で開始するには、nohup /usr/bin/apic-bond-watch <interface name>& をルートユーザーとして使用します。デフォルトのインターフェイス名は bond0 です。次に例を示します。

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

手順

ステップ 1 Cisco Application Policy Infrastructure Controller（APIC）のバージョンに応じて、次のいずれかのアクションを実行します。

- **Cisco APIC4.2(1)** 以降：パラメータ ACIEnableBondWatchService を True に設定します。『*Cisco ACI Installation Guide for Red Hat OpenStack Using OpenStack Platform 10 Director*』の「Installing Overcloud」セクションまたは『*Cisco ACI Installation Guide for Red Hat OpenStack Using the OpenStack Platform 13 Director*』の「Parameters for the Cisco ACI Environment」セクションを参照してください。この手順の残りのステップは実行しないでください。
- **Cisco APIC4.1(x) 以前**：この手順のステップ 2 ～ 4 を完了します。

ステップ 2 すべてのコンピューティングノードの IP アドレスのインベントリを作成します。

例：

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

必要に応じて、すべてのコントローラのインベントリを作成できます。

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

ステップ 3 サービスをインストールし 有効にします

例：

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

ステップ 4 ボンド ウォッチ サービスを開始します。

例：

```
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"
```

ボンド ウォッチが定義されていないバージョンでは、サービスを手動で開始できます。

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

ボンドでアクティブな NIC の特定

/proc/net/bonding ディレクトリの bond0 ファイルは、2 つの NIC のどちらがアクティブであるかを示します。

手順

bond0 を調べて、アクティブな NIC を確認します。

例：

```
[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]#
```

NIC MTU の設定

NIC の最大伝送ユニット（MTU）を設定して確認します。MTU は、Cisco Unified Computing System（UCS）Manager で指定した設定に基づきます。

手順

ステップ 1 NIC の MTU を 1600 または 9000 に設定します。

ステップ 2 UCS B シリーズ サーバーに移動して NIC を選択し、[MTU] フィールドの値を確認して、MTU 設定を確認します。

NIC の MTU 設定の確認

NIC の 最大伝送ユニット（MTU）を確認します。

手順

次のコマンドを入力します。

```
ip link
```

以下のような出力が表示されます。

```
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
```

Cisco UCS の構成

Cisco Unified Computing System（UCS）B シリーズを適切に構成して、Cisco Application Centric Infrastructure（ACI）および OpenStack と統合します。サポート可能な構成には、次のものが含まれている必要があります。

- 2 つの NIC を使用した Cisco UCS サービス プロファイルの構成。これらの NIC は、OpFlex 通信と VM データ パスに使用されます。
- これらの NIC でファブリック フェールオーバーを無効にします。
- 仮想 NIC（vNIC）を異なるファブリック インターコネクトに接続します。
- vNIC の 最大伝送ユニット（MTU）を設定します。
- vNIC が目的の VLAN を許可していることを確認します。
- ファブリック インターコネクトのマルチキャストをオンにします。
- ファブリック インターコネクトでのポート チャネル インターフェイス ポリシーの構成。

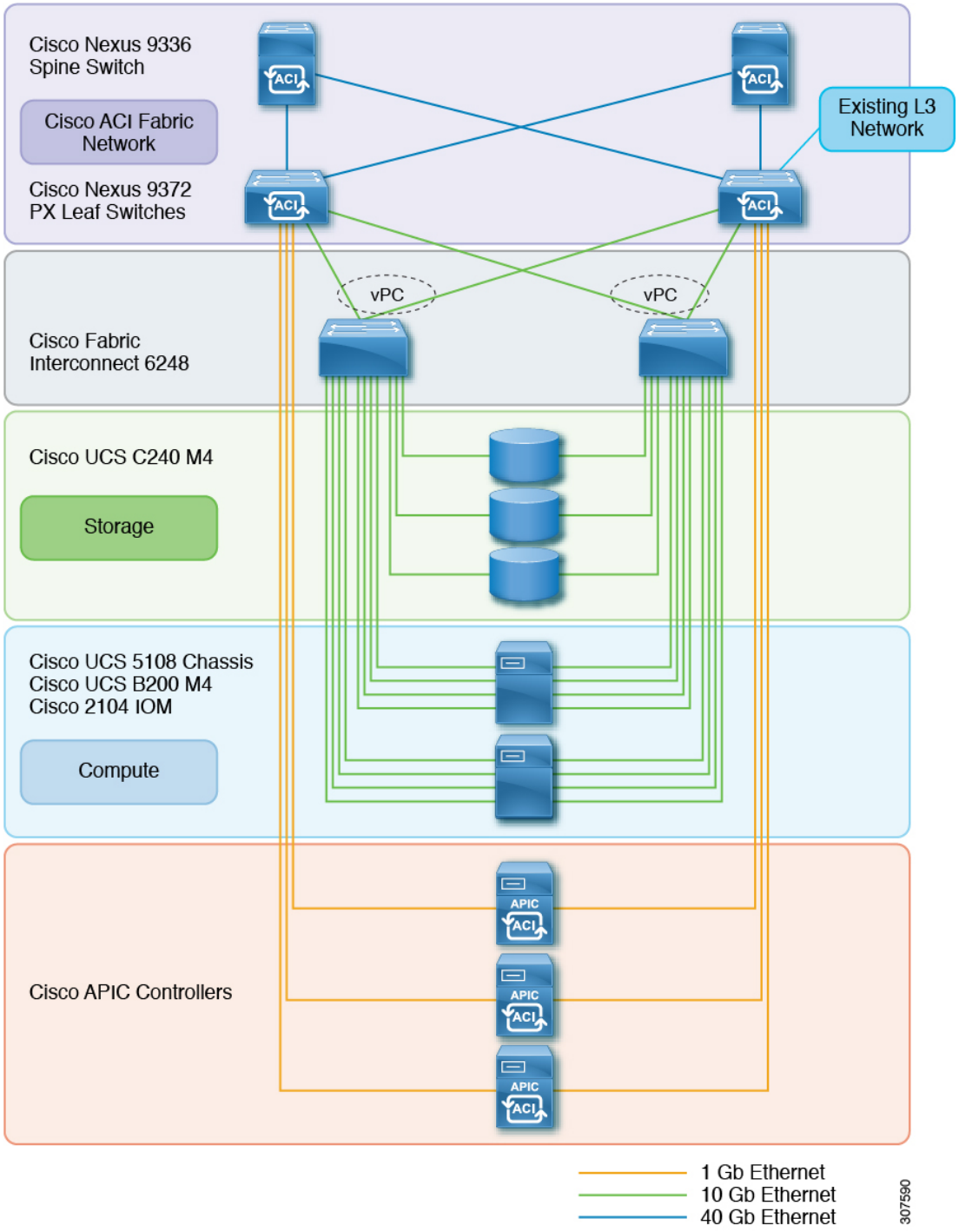
リーフ スイッチの構成

パスの冗長性を確保するには、2 つのリーフ スイッチ間で仮想ポート チャネル（vPC）インターフェイス ポリシーを構成します。仮想ポートチャネル（vPC）を構成するには、さまざまな方法があります。詳細については、『*Cisco APIC Layer 2 Networking Configuration Guide*』を参照してください。

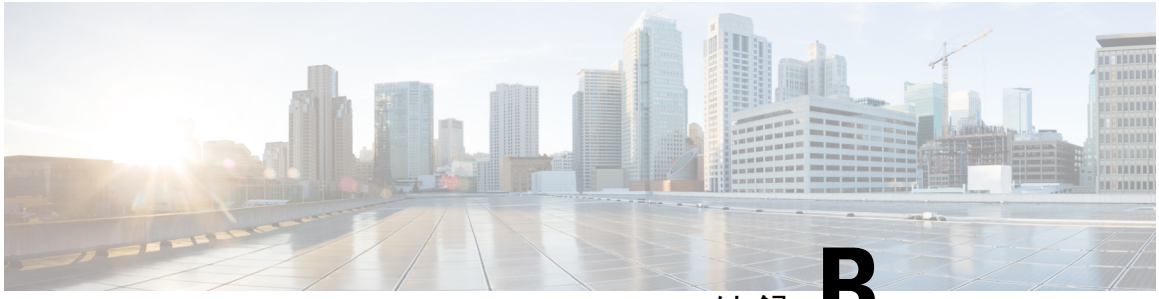
UCS と vPC の設定方法に関係なく、構成は次の図のようになります。

UCS B シリーズの構成

図 *3 :* リーフ スイッチの構成



UCS B シリーズの構成



付録 B

OpenStack の Ironic の構成

Cisco ACI による OpenStack の Ironic

Cisco Application Policy Infrastructure Controller（APIC）の 5.0(1) OpenStack プラグイン リリース以降、ベア メタル サーバーの Ironic プロビジョニングの使用がサポートされています。

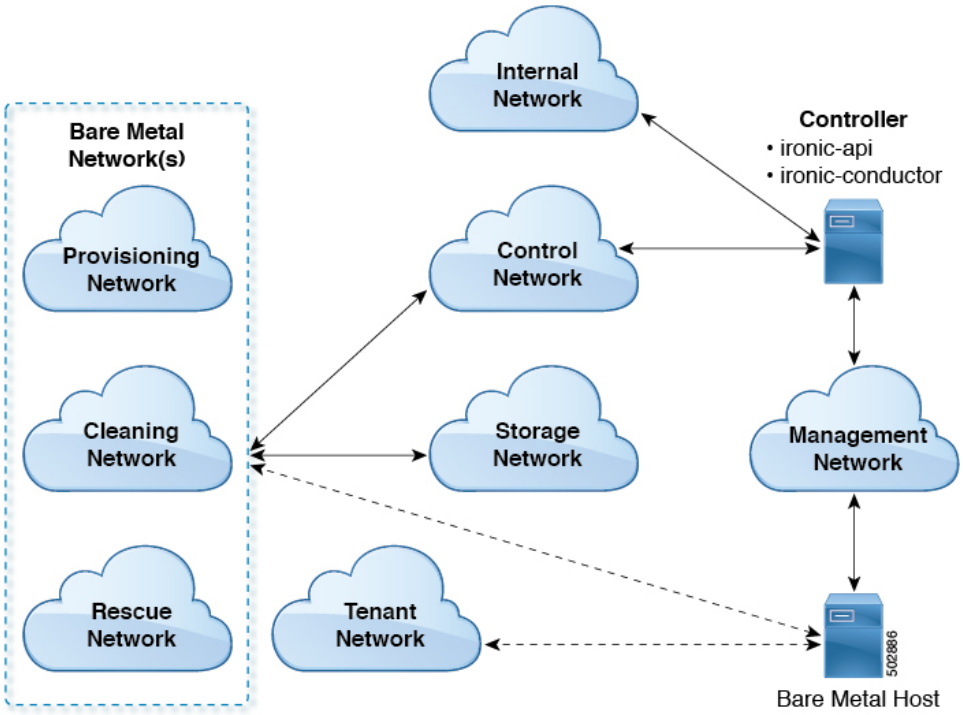
OpenStack オーバークラウドに展開される Ironic は、Compute（Nova）や Network（Neutron）などの OpenStack サービスと統合されます。Ironic は単独で使用することも、OpenStack クラウドの一部として使用することもできます。Ironic を OpenStack で使用すると、同じ OpenStack クラスタ上で仮想マシンとベア メタル サーバーの両方をプロビジョニングできます。

この付録では、OpenStack の Cisco Application Centric Infrastructure（ACI）プラグインを使用する場合の 5.0(1) リリースでの Ironic の変更について説明します。

ネットワークの Ironic

Neutron は、ベア メタル ホストとスイッチ間の接続を管理します。Ironic は、ベア メタル ホストのインベントリと、ホストの NIC とスイッチ上のポート間の接続を維持します。ベア メタル ホスト ポートを Neutron ネットワークに接続する必要がある場合、Ironic はこのインベントリ情報を Neutron に提供します。メカニズム ドライバは、この情報を使用して、Neutron ネットワークに適切な VLAN を使用してスイッチ ポートを設定します。

次の図は、Ironic を使用する場合のネットワークを示しています。



Ironic は 3 つの論理ネットワークを定義します。これらはまとめてベア メタル ネットワークと呼ばれ、ベア メタル インスタンスのライフサイクルにおける目的に基づいて命名されています。

- プロビジョニング：起動用のインスタンスの設定
- クリーニング：ハード ディスクの消去とその他のセキュリティの問題
- レスキュー：リカバリ用のインスタンスのアタッチ

ベア メタル ネットワークは、管理者によって作成された Neutron の通常のネットワークです。ただし、OpenStack の管理プロジェクトにのみ表示されるように作成する必要があります。（**`--no-share`** オプションを使用して作成します）。3 つの論理ネットワークはすべて、同じ Neutron ネットワーク、個別の Neutron ネットワーク、またはネットワークの任意の組み合わせに実装できます。プロビジョニング ネットワークを使用すると、プロビジョニングとテナン

トネットワーク接続の両方に単一のプロバイダーネットワークを使用するよりもセキュリティが向上します。

Ironic の構成と展開のワークフロー

このセクションでは、OpenStack Ironic ベア メタル インスタンスを構成および展開するために実行する必要があるタスクの高度な概要を示します。

1. すべての前提条件を満たしてください。

   このガイドのIronic をサポートするための OpenStack の展開の前提条件 （44 ページ）セクションを参照してください。

2. オーバークラウドに Ironic サービスを展開するために必要なテンプレートを作成します。

   このガイドの展開前の構成 （45 ページ）セクションを参照してください。

3. Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従って、オーバークラウドを展開します。

4. Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従って、イメージを作成し、オーバークラウドにアップロードします。

5. ベア メタル ネットワークと総称される Ironic のクリーニング、プロビジョニング、およびレスキュー ネットワークを作成します。

   このガイドのベア メタル ネットワークの作成 （54 ページ）セクションを参照してください。

6. ベア メタル ネットワークを Ironic サービスに接続します。この接続は、コントラクトを適用し、OpenStack サービスの保護をさらに強化する Cisco Application Centric Infrastructure（ACI）ポリシーを追加するために必要です。

   このガイドのIronic サービスへのベア メタル ネットワークの接続 （56 ページ）セクションを参照してください。

7. Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従って、ベア メタル ノードを登録します。

8. ベア メタル ポートおよびポート グループを作成します。手順は Red Hat OpenStack Platform 13 のドキュメントとほぼ同じですが、追加のトポロジ情報を指定する必要があります。

   このガイドのベア メタル ポートおよびポート グループの作成 （57 ページ）セクションを参照してください。

9. Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従って、ベア メタル フレーバーとアベイラビリティ ゾーンを作成します。

10. Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従って、ベア メタル インスタンスを起動します。

Ironic をサポートするための OpenStack の展開の前提条件

Ironic をサポートするために OpenStack を展開する前に、次のタスクを完了する必要があります。

- Red Hat OpenStack Platform（OSP）13 が Cisco Application Centric Infrastructure（ACI）ML2 プラグインバージョン 5.0(1) 以降でインストールされていることを確認します。

（注） Ironic ベア メタル プロビジョニングは、Red Hat OSP16.2 でのみサポートされます。

- すべての OpenStack サーバーが Nexus 9000 EX、FX、または GX スイッチにのみ接続されていることを確認します。
- Ironic サービスの場所を決定します。

  Ironic は、展開に次のサービスを追加します。

  - **Ironic-api**：Ironic 展開用のアプリケーション プログラミング インターフェイスです。
  - **Ironic- Conductor**：ベア メタル リソースですべてのアクションを実行するコンダクタ マネージャを作成します。
  - **Preboot Execution（PXE）サーバー**：オペレーティング システムがまだロードされていないネットワーク コンピュータを、管理者がリモートで設定および起動できるようにします。HTTP、TFTP、またはその両方の場合。

  推奨されるアプローチは、カスタムの構成可能なネットワークを使用して Ironic サービスをホストすることです。カスタム構成可能ネットワークの作成に関する OSP16.2 ドキュメントのガイドラインと、このネットワークで Ironic サービスをターゲットにする方法を説明する Ironic セクションに従ってください。
- コントローラのインターフェイスとそれに関連付けられたブリッジマッピングを定義します。

  OpenStack Platform（OSP）は、各コントローラ ホストにアンダークラウド制御ネットワーク上のインターフェイスがあることを保証します。つまり、デフォルト設定が使用されている場合、Ironic サービスはネットワーク インターフェイスにそれを使用します。ただし、ServiceNetMap パラメータを使用して Ironic サービスに別のネットワークを指定する場合は、追加のネットワーク インターフェイスを設定する必要があります。Red Hat の Web サイトにある OSP16.2 ドキュメントの「OpenStack Provisioning」ページを参照してください。

Ironic サービスは通常、コントローラノードで実行するように設定されます。他のサービスと同様に、IP アドレス、OpenStack Identity（Keystone）でのサービス認証、および OpenStack の

メッセージバス、データベース、およびその他のサービスに到達する方法に関する情報を含む構成ファイルが必要です。

展開前の構成

Red Hat OpenStack Platform13 Director のドキュメントでは、オーバークラウドに Ironic サービスを展開するために必要な追加のテンプレート構成について説明しています。Ironic サービスで使用する場合、OpenStack の Cisco Application Centric Infrastructure（ACI）ML2 プラグイン統合に固有の新しいテンプレート設定はありません。

仮想ポート チャネル（vPC）を使用してベア メタル サーバーを接続する場合、リーフ インターフェイス ポリシー グループに割り当てられたポート チャネル ポリシーには、次の制御パラメータのみが含まれている必要があります。

- ホットスタンバイポートの高速セレクト
- グレースフルコンバージェンス

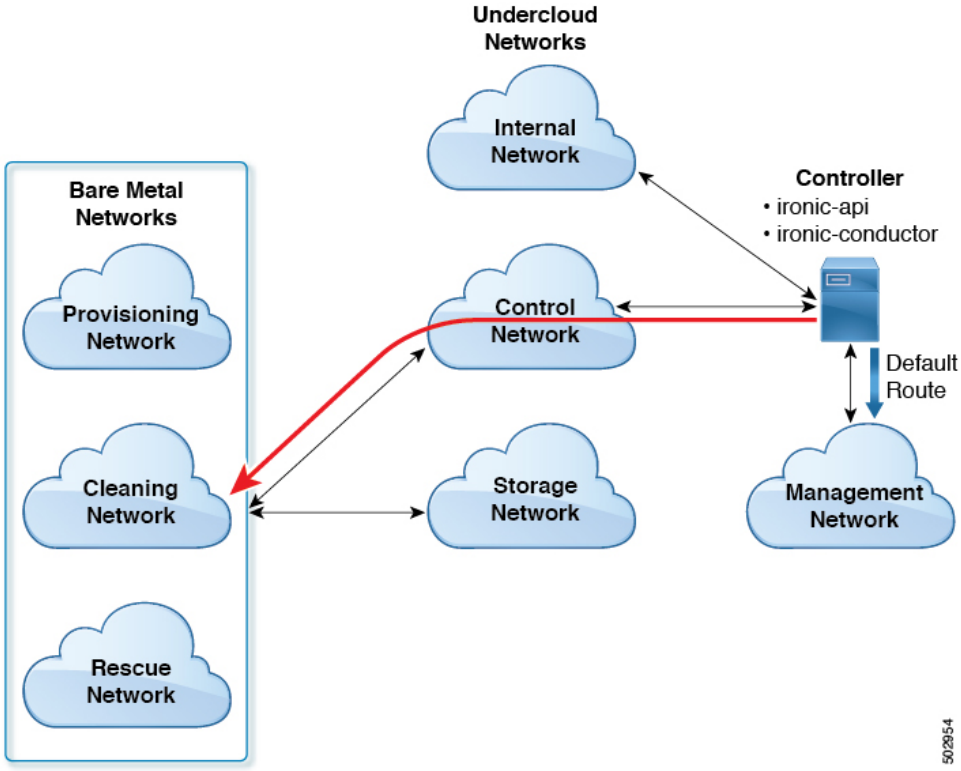
ポート チャネル ポリシーでは、モードが「LACP Active」に設定されている必要があります。この設定が必要なのは、Unified Extensible Firmware Interface（UEFI）が起動してベア メタル ネットワークに接続するときに、ベア メタル インスタンスが単一の NIC のみを使用するためです。両方のポートは、テナント ネットワークに接続するときに使用されます。また、[個別ポートの一時停止（Suspend Individual Port）] オプションを無効にする必要があります。

カスタムの構成可能ネットワークが Ironic サービスに使用されている場合は、Cisco ACI でエンドポイントグループ（EPG）とブリッジドメインを作成して、そのネットワークを実装する必要があります。これは、『Cisco ACI Installation Guide for Red Hat OpenStack Using the OpenStack Platform 13 Director』の「Setting Up the Cisco APIC and the Network」セクションで説明されているように、 Cisco ACI に実装されているアンダークラウド ネットワークでも同じです。

Ironic サービスが作成される場所（アンダークラウド制御ネットワークやカスタム構成可能ネットワークなど）に関係なく、Ironic サービスをホストするアンダークラウド ネットワークを実装するブリッジ ドメインにサブネットを設定する必要があります。このサブネットには、アンダークラウド ネットワークに使用されるサブネットと一致する CIDR IP アドレスが必要です。

たとえば、コントロール プレーン ネットワークがサービスに使用されている場合（デフォルト）、サブネットのデフォルト値は 1.100.1.0/24 です。したがって、このサブネット上の一部の IP アドレスは、このブリッジ ドメインのサブネットで構成された CIDR として使用する必要があります。

次の図は、コントローラがベア メタル ネットワークに到達するために使用するさまざまなパスを示しています。コントローラからベア メタル ネットワークに向かう赤い矢印は、（デフォルト ルートとは対照的に）確立する必要があるパスを示しています。両方向の黒い矢印は接続を示しています。コントローラは内部ネットワーク、制御ネットワーク、および管理ネットワークに接続され、制御ネットワークとストレージ ネットワークはベア メタル ネットワークに接続されます。



（注） Ironic サービスに制御ネットワークを使用する場合、OpenStack プラットフォームがアンダークラウド仮想マシン（VM）にこの IP アドレスを割り当てるため、IP アドレスは 1.100.1.1 以外にする必要があります。

ブリッジ ドメインで構成されたこのサブネットは、Ironic サービスがベア メタル ネットワークに到達できるように、展開後に追加されるルートのネクストホップ IP アドレスとして使用されます。

オーバークラウドの展開

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） の項のタスクを実行します。

手順

Red Hat OpenStack Platform オーバークラウドを展開します。

Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従います。

オーバークラウドへのベアメタルイメージの作成とアップロード

始める前に

- Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） の項のタスクを実行します。
- オーバークラウドを展開します。

手順

Red Hat の Web サイトにある Red Hat OpenStack Platform 16.2 ドキュメントの手順に従って、イメージを作成し、オーバークラウドにアップロードします。

EPG 間のトラフィックを許可するコントラクトの作成

Ironicサービスをホストするアンダークラウドネットワークのエンドポイントグループ（EPG）とオーバークラウドベアメタルネットワークのEPG間のトラフィックを許可するには、ポリシーが必要です。Ironic サービスは他のサービスと同じネットワーク上にある可能性があるため、ポリシーは Ironic サービスとベア メタル インスタンス間で必要なトラフィックのみに制限する必要があります。これらのポリシーのコントラクト、サブジェクト、フィルタ、およびフィルタ エントリは、このセクションのプロトコルをカバーする必要があります。

必要なコントラクト

- ベア メタル ネットワークによって消費され、Ironic サービスによって提供される必要があるコントラクト：
  - PXE-TFTP（UDP 接続先ポート 69）
  - PXE-HTTP（TCP、デフォルトの接続先ポート 8088）

    OSP16.2 インストール用のコントローラで次のスクリプトを実行して、使用中のポートを確認します。

```
# egrep ^http_url
/var/lib/config-data/puppet-generated/ironic/etc/ironic/ironic.conf  \
   | awk -F":" '{print $3}'
```

OpenStack の Ironic の構成

- 直接展開インターフェイス用の Swift API（TCP、デフォルトの接続先ポート 8080）

  使用されているポートを確認し、OSP16.2 インストール用のコントローラで次のスクリプトを実行します。

```
# egrep -A 1 'listen swift' \
   /var/lib/config-data/puppet-generated/haproxy/etc/haproxy/haproxy.cfg \
   | awk -F":" '{print $2}' | awk '{print $1}'
```

- Ironic API（TCP、デフォルトの接続先ポート 6385）

  使用されているポートを確認し、OSP16.2 インストール用のコントローラで次のスクリプトを実行します。

```
# egrep -A 1 'listen ironic' \
   /var/lib/config-data/puppet-generated/haproxy/etc/haproxy/haproxy.cfg \
   | awk -F":" '{print $2}' | awk '{print $1}'
```

- ベアメタルネットワークによって提供される必要があり、Ironic サービスによって消費されるコントラクト：
  - RAMDISK サーバー（TCP 接続先ポート 9999）
  - iSCSI 展開インターフェイス用の iSCSI（TCP 接続先ポート 3260）

（注） オンネットワーク トラフィックは EPG 内トラフィックであるため、DHCP や ARP などのオンネットワーク トラフィックのコントラクトは必要ありません。

コントラクトの作成のためのコマンド

コントラクトは、**aimctl** コマンドを使用するか、Cisco Application Centric Infrastructure（APIC）GUI または CLI を使用して直接作成できます。Cisco Application Centric Infrastructure（ACI）でコントラクトを作成するために OpenStack オーバークラウド コントローラ ノードで実行できる XML コマンドの次の例を参照してください。

```
<polUni>
  <fvTenant name="common">

      <!-- Here are the filters ironic-to-instances -->

      <vzFilter childAction="" descr="" dn="uni/tn-common/flt-ironic-to-instances"
extMngdBy="" fwdId="51" id="implicit" lcOwn="local" modTs="2020-05-14T16:47:26.264+00:00"
 monPolDn="uni/tn-common/monepg-default" name="ironic-to-instances" nameAlias=""
ownerKey="" ownerTag="" revId="52" status="" txId="0" uid="15374" unsupportedEntries="no"
 unsupportedMgmtEntries="no" userdom="" usesIds="yes">
        <vzRsFwdRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.264+00:00" rType="mo" rn="rsFwdRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-51" tType="mo"/>

        <vzRsRevRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.264+00:00" rType="mo" rn="rsRevRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-52" tType="mo"/>

         <vzRtSubjFiltAtt childAction="" lcOwn="local"
modTs="2020-05-14T16:47:26.138+00:00"
```

```
rn="rtsubjFiltAtt-[uni/tn-common/brc-ironic-services/subj-ironic-services]" status=""
tCl="vzSubj" tDn="uni/tn-common/brc-ironic-services/subj-ironic-services"/>
        <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="8088"
 dToPort="8088" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified"
icmpv6T="unspecified" lcOwn="local" matchDscp="unspecified"
modTs="2020-05-14T16:47:26.221+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-pxe-http" nameAlias="" prot="tcp" rn="e-ironic-pxe-http"
sFromPort="unspecified" sToPort="unspecified" stateful="yes" status="" tcpRules=""
uid="15374" userdom=""/>
        <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="6385"
 dToPort="6385" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified"
icmpv6T="unspecified" lcOwn="local" matchDscp="unspecified"
modTs="2020-05-14T16:47:26.179+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-api" nameAlias="" prot="tcp" rn="e-ironic-api" sFromPort="unspecified"
sToPort="unspecified" stateful="yes" status="" tcpRules="" uid="15374" userdom=""/>
         <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="69"
 dToPort="69" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified" icmpv6T="unspecified"
 lcOwn="local" matchDscp="unspecified" modTs="2020-05-14T16:47:26.264+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-tftp" nameAlias="" prot="udp"
rn="e-ironic-tftp" sFromPort="unspecified" sToPort="unspecified" stateful="yes" status=""
 tcpRules="" uid="15374" userdom=""/>
     </vzFilter>


     <!-- Here are the filters instances-to-ironic -->

     <vzFilter  childAction="" descr="" dn="uni/tn-common/flt-instances-to-ironic"
extMngdBy="" fwdId="35" id="implicit" lcOwn="local" modTs="2020-05-14T16:47:25.921+00:00"
 monPolDn="uni/tn-common/monepg-default" name="instances-to-ironic" nameAlias=""
ownerKey="" ownerTag="" revId="46" status="" txId="0" uid="15374" unsupportedEntries="no"
 unsupportedMgmtEntries="no" userdom="" usesIds="yes">
        <vzRsFwdRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:25.921+00:00" rType="mo" rn="rsFwdRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-35" tType="mo"/>

        <vzRsRevRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:25.921+00:00" rType="mo" rn="rsRevRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-46" tType="mo"/>

         <vzRtSubjFiltAtt childAction="" lcOwn="local"
modTs="2020-05-14T16:47:25.762+00:00"
rn="rtsubjFiltAtt-[uni/tn-common/brc-ironic-agents/subj-ironic-agents]" status=""
tCl="vzSubj" tDn="uni/tn-common/brc-ironic-agents/subj-ironic-agents"/>
        <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="3260"
 dToPort="3260" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified"
icmpv6T="unspecified" lcOwn="local" matchDscp="unspecified"
modTs="2020-05-14T16:47:25.809+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-iscsi" nameAlias="" prot="tcp" rn="e-ironic-iscsi" sFromPort="unspecified"
 sToPort="unspecified" stateful="yes" status="" tcpRules="" uid="15374" userdom=""/>
        <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="9999"
 dToPort="9999" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified"
icmpv6T="unspecified" lcOwn="local" matchDscp="unspecified"
modTs="2020-05-14T16:47:25.921+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-ramdisk" nameAlias="" prot="tcp" rn="e-ironic-ramdisk" sFromPort="unspecified"
 sToPort="unspecified" stateful="yes" status="" tcpRules="" uid="15374" userdom=""/>
     </vzFilter>

     <!-- Here are the filters for ironic-tftp-client -->

     <vzFilter  childAction="" descr="" dn="uni/tn-common/flt-ironic-tftp-client"
extMngdBy="" fwdId="29" id="implicit" lcOwn="local" modTs="2020-05-14T16:47:26.018+00:00"
 monPolDn="uni/tn-common/monepg-default" name="ironic-tftp-client" nameAlias="" ownerKey=""
 ownerTag="" revId="29" status="" txId="0" uid="15374" unsupportedEntries="no"
unsupportedMgmtEntries="no" userdom="" usesIds="yes">
```

OpenStack の Ironic の構成

```
        <vzRsFwdRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.018+00:00" rType="mo" rn="rsFwdRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-29" tType="mo"/>

        <vzRsRevRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.018+00:00" rType="mo" rn="rsRevRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-29" tType="mo"/>

         <vzRtSubjFiltAtt childAction="" lcOwn="local"
modTs="2020-05-14T16:47:25.973+00:00"
rn="rtsubjFiltAtt-[uni/tn-common/brc-ironic-tftp-client/subj-ironic-tftp-client]" status=""
 tCl="vzSubj" tDn="uni/tn-common/brc-ironic-tftp-client/subj-ironic-tftp-client"/>
         <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction=""
dFromPort="unspecified" dToPort="unspecified" descr="" etherT="ip" extMngdBy=""
icmpv4T="unspecified" icmpv6T="unspecified" lcOwn="local" matchDscp="unspecified"
modTs="2020-05-14T16:47:26.018+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-tftp-client" nameAlias="" prot="udp" rn="e-ironic-tftp-client"
sFromPort="unspecified" sToPort="unspecified" stateful="no" status="" tcpRules=""
uid="15374" userdom=""/>
      </vzFilter>

      <!-- Here are the filters for ironic-tftp-server -->

      <vzFilter  childAction="" descr="" dn="uni/tn-common/flt-ironic-tftp-server"
extMngdBy="" fwdId="47" id="implicit" lcOwn="local" modTs="2020-05-14T16:47:26.096+00:00"
 monPolDn="uni/tn-common/monepg-default" name="ironic-tftp-server" nameAlias="" ownerKey=""
 ownerTag="" revId="50" status="" txId="0" uid="15374" unsupportedEntries="no"
unsupportedMgmtEntries="no" userdom="" usesIds="yes">
        <vzRsRevRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.096+00:00" rType="mo" rn="rsRevRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-50" tType="mo"/>

        <vzRsFwdRFltPAtt childAction="deleteNonPresent" forceResolve="yes" lcOwn="local"
 modTs="2020-05-14T16:47:26.096+00:00" rType="mo" rn="rsFwdRFltPAtt" state="formed"
stateQual="none" status="" tCl="vzAFilterableUnit" tDn="uni/tn-common/fp-47" tType="mo"/>

        <vzRtFiltAtt childAction="" lcOwn="local" modTs="2020-05-14T16:47:26.057+00:00"
 rn="rtfiltAtt-[uni/tn-common/brc-ironic-tftp-server/subj-ironic-tftp-server/intmnl]"
status="" tCl="vzInTerm"
tDn="uni/tn-common/brc-ironic-tftp-server/subj-ironic-tftp-server/intmnl"/>
         <vzEntry  applyToFrag="no" arpOpc="unspecified" childAction="" dFromPort="69"
 dToPort="69" descr="" etherT="ip" extMngdBy="" icmpv4T="unspecified" icmpv6T="unspecified"
 lcOwn="local" matchDscp="unspecified" modTs="2020-05-14T16:47:26.096+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-tftp-server" nameAlias="" prot="udp"
 rn="e-ironic-tftp-server" sFromPort="unspecified" sToPort="unspecified" stateful="no"
status="" tcpRules="" uid="15374" userdom=""/>
      </vzFilter>

      <!-- Here are the contracts for ironic-agents -->

     <vzBrCP  childAction="" configIssues="" descr="" dn="uni/tn-common/brc-ironic-agents"
 extMngdBy="" intent="install" lcOwn="local" modTs="2020-05-14T16:47:25.921+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-agents" nameAlias="" ownerKey=""
ownerTag="" prio="unspecified" reevaluateAll="no" scope="context" status=""
targetDscp="unspecified" uid="15374" userdom="">
         <vzDirAssDef childAction="deleteNonPresent" descr="" lcOwn="local"
modTs="2020-05-14T16:47:24.933+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" ownerKey="" ownerTag="" rn="dirass" status="">
           <vzConsDef anyDn=""
bdDefDn="uni/bd-[uni/tn-common/BD-sauto-osd-bd-vlan105]-isSvc-no" bdDefStQual="none"
childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105" exceptionTag=""
```

```
intent="install" isAny="no" isCifLblIntersEmpty="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" modTs="2020-05-14T16:47:24.933+00:00"
monPolDn="uni/tn-common/monepg-default" name="sauto-osd-epg-vlan105" nameAlias=""
ownerKey="" ownerTag="" pcEnfPref="enforced" pcTag="49155" prefGrMemb="exclude"
prio="unspecified"
rn="cons-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]-any-no"
scopeId="2555905" status="" targetDscp="unspecified" txId="1729382256910271363"
useAnyDef="no"/>
              <vzProvDef anyDn=""
bdDefDn="uni/bd-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/BD-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-isSvc-no"
 bdDefStQual="none" childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 exceptionTag="" hasTriggeredAlloc="no" intent="install" isAny="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" matchT="AtleastOne"
modTs="2020-05-14T16:47:24.956+00:00" monPolDn="uni/tn-common/monepg-default"
name="net_1005e02b-2ed2-44eb-8ac5-6d989176ebef" nameAlias="" ownerKey="" ownerTag=""
pcEnfPref="enforced" pcTag="49159" prefGrMemb="exclude" prio="unspecified"
rn="prov-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-any-no"
 scopeId="2555905" status="" targetDscp="unspecified" txId="7493989779947781613"
useAnyDef="no"/>
          </vzDirAssDef>
          <vzSubj  childAction="" configIssues="" consMatchT="AtleastOne" descr=""
extMngdBy="" lcOwn="local" modTs="2020-05-14T16:47:25.010+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-agents" nameAlias=""
prio="unspecified" provMatchT="AtleastOne" revFltPorts="yes" rn="subj-ironic-agents"
status="" targetDscp="unspecified" uid="15374" userdom="">
             <vzRsSubjFiltAtt action="permit"  childAction="" directives="" extMngdBy=""
 forceResolve="yes" lcOwn="local" modTs="2020-05-14T16:47:25.762+00:00"
monPolDn="uni/tn-common/monepg-default" priorityOverride="default" rType="mo"
rn="rssubjFiltAtt-instances-to-ironic" state="formed" stateQual="none" status=""
tCl="vzFilter" tContextDn="" tDn="uni/tn-common/flt-instances-to-ironic"
tRn="flt-instances-to-ironic" tType="name" tnVzFilterName="instances-to-ironic" uid="15374"
 userdom=""/>
          </vzSubj>
          <vzRtProv childAction="" lcOwn="local" modTs="2020-05-14T16:47:24.956+00:00"
rn="rtfvProv-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]"
 status="" tCl="fvAEPg"
tDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"/>

          <vzRtCons childAction="" lcOwn="local" modTs="2020-05-14T16:47:24.933+00:00"
rn="rtfvCons-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]" status=""
 tCl="fvAEPg" tDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105"/>
      </vzBrCP>

      <!-- Here are the contracts for ironic-services -->

      <vzBrCP  childAction="" configIssues="" descr=""
dn="uni/tn-common/brc-ironic-services" extMngdBy="" intent="install" lcOwn="local"
modTs="2020-05-14T16:47:26.264+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-services" nameAlias="" ownerKey="" ownerTag="" prio="unspecified"
reevaluateAll="no" scope="context" status="" targetDscp="unspecified" uid="15374"
userdom="">
          <vzDirAssDef childAction="deleteNonPresent" descr="" lcOwn="local"
modTs="2020-05-14T16:47:25.067+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" ownerKey="" ownerTag="" rn="dirass" status="">
              <vzConsDef anyDn=""
bdDefDn="uni/bd-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/BD-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-isSvc-no"
 bdDefStQual="none" childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 exceptionTag="" intent="install" isAny="no" isCifLblIntersEmpty="no" isMsiteEPg="no"
```

OpenStack の Ironic の構成

```
l3CtxEncap="vxlan-2555905" lcOwn="local" modTs="2020-05-14T16:47:25.081+00:00"
monPolDn="uni/tn-common/monepg-default" name="net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 nameAlias="" ownerKey="" ownerTag="" pcEnfPref="enforced" pcTag="49159"
prefGrMemb="exclude" prio="unspecified"
rn="cons-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-any-no"
 scopeId="2555905" status="" targetDscp="unspecified" txId="7493989779947781613"
useAnyDef="no"/>
              <vzProvDef anyDn=""
bdDefDn="uni/bd-[uni/tn-common/BD-sauto-osd-bd-vlan105]-isSvc-no" bdDefStQual="none"
childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105" exceptionTag=""
hasTriggeredAlloc="no" intent="install" isAny="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" matchT="AtleastOne"
modTs="2020-05-14T16:47:25.067+00:00" monPolDn="uni/tn-common/monepg-default"
name="sauto-osd-epg-vlan105" nameAlias="" ownerKey="" ownerTag="" pcEnfPref="enforced"
pcTag="49155" prefGrMemb="exclude" prio="unspecified"
rn="prov-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]-any-no"
scopeId="2555905" status="" targetDscp="unspecified" txId="1729382256910271363"
useAnyDef="no"/>
          </vzDirAssDef>
          <vzSubj  childAction="" configIssues="" consMatchT="AtleastOne" descr=""
extMngdBy="" lcOwn="local" modTs="2020-05-14T16:47:25.117+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-services" nameAlias=""
prio="unspecified" provMatchT="AtleastOne" revFltPorts="yes" rn="subj-ironic-services"
status="" targetDscp="unspecified" uid="15374" userdom="">
            <vzRsSubjFiltAtt action="permit"  childAction="" directives="" extMngdBy=""
 forceResolve="yes" lcOwn="local" modTs="2020-05-14T16:47:26.138+00:00"
monPolDn="uni/tn-common/monepg-default" priorityOverride="default" rType="mo"
rn="rssubjFiltAtt-ironic-to-instances" state="formed" stateQual="none" status=""
tCl="vzFilter" tContextDn="" tDn="uni/tn-common/flt-ironic-to-instances"
tRn="flt-ironic-to-instances" tType="name" tnVzFilterName="ironic-to-instances" uid="15374"
 userdom=""/>
          </vzSubj>
          <vzRtProv childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.067+00:00"
rn="rtfvProv-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]" status=""
 tCl="fvAEPg" tDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105"/>
          <vzRtCons childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.081+00:00"
rn="rtfvCons-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]"
 status="" tCl="fvAEPg"
tDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"/>

      </vzBrCP>

      <!-- Here are the contracts for ironic-tftp-client -->

      <vzBrCP  childAction="" configIssues="" descr=""
dn="uni/tn-common/brc-ironic-tftp-client" extMngdBy="" intent="install" lcOwn="local"
modTs="2020-05-14T16:47:26.018+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-tftp-client" nameAlias="" ownerKey="" ownerTag="" prio="unspecified"
reevaluateAll="no" scope="context" status="" targetDscp="unspecified" uid="15374"
userdom="">
          <vzDirAssDef childAction="deleteNonPresent" descr="" lcOwn="local"
modTs="2020-05-14T16:47:25.168+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" ownerKey="" ownerTag="" rn="dirass" status="">
              <vzConsDef anyDn=""
bdDefDn="uni/bd-[uni/tn-common/BD-sauto-osd-bd-vlan105]-isSvc-no" bdDefStQual="none"
childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105" exceptionTag=""
intent="install" isAny="no" isCifLblIntersEmpty="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" modTs="2020-05-14T16:47:25.168+00:00"
```

```
monPolDn="uni/tn-common/monepg-default" name="sauto-osd-epg-vlan105" nameAlias=""
ownerKey="" ownerTag="" pcEnfPref="enforced" pcTag="49155" prefGrMemb="exclude"
prio="unspecified"
rn="cons-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]-any-no"
scopeId="2555905" status="" targetDscp="unspecified" txId="1729382256910271363"
useAnyDef="no"/>
              <vzProvDef anyDn=""
bdDefDn="uni/bd-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/BD-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-isSvc-no"
 bdDefStQual="none" childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 exceptionTag="" hasTriggeredAlloc="no" intent="install" isAny="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" matchT="AtleastOne"
modTs="2020-05-14T16:47:25.189+00:00" monPolDn="uni/tn-common/monepg-default"
name="net_1005e02b-2ed2-44eb-8ac5-6d989176ebef" nameAlias="" ownerKey="" ownerTag=""
pcEnfPref="enforced" pcTag="49159" prefGrMemb="exclude" prio="unspecified"
rn="prov-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-any-no"
 scopeId="2555905" status="" targetDscp="unspecified" txId="7493989779947781613"
useAnyDef="no"/>
          </vzDirAssDef>
          <vzSubj  childAction="" configIssues="" consMatchT="AtleastOne" descr=""
extMngdBy="" lcOwn="local" modTs="2020-05-14T16:47:25.252+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-tftp-client" nameAlias=""
prio="unspecified" provMatchT="AtleastOne" revFltPorts="yes" rn="subj-ironic-tftp-client"
 status="" targetDscp="unspecified" uid="15374" userdom="">
            <vzRsSubjFiltAtt action="permit"  childAction="" directives="" extMngdBy=""
 forceResolve="yes" lcOwn="local" modTs="2020-05-14T16:47:25.973+00:00"
monPolDn="uni/tn-common/monepg-default" priorityOverride="default" rType="mo"
rn="rssubjFiltAtt-ironic-tftp-client" state="formed" stateQual="none" status=""
tCl="vzFilter" tContextDn="" tDn="uni/tn-common/flt-ironic-tftp-client"
tRn="flt-ironic-tftp-client" tType="name" tnVzFilterName="ironic-tftp-client" uid="15374"
 userdom=""/>
          </vzSubj>
          <vzRtProv childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.189+00:00"
rn="rtfvProv-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]"
 status="" tCl="fvAEPg"
tDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"/>

          <vzRtCons childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.168+00:00"
rn="rtfvCons-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]" status=""
 tCl="fvAEPg" tDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105"/>
      </vzBrCP>

      <!-- Here are the contracts for ironic-tftp-server -->
      <vzBrCP  childAction="" configIssues="" descr=""
dn="uni/tn-common/brc-ironic-tftp-server" extMngdBy="" intent="install" lcOwn="local"
modTs="2020-05-14T16:47:26.096+00:00" monPolDn="uni/tn-common/monepg-default"
name="ironic-tftp-server" nameAlias="" ownerKey="" ownerTag="" prio="unspecified"
reevaluateAll="no" scope="context" status="" targetDscp="unspecified" uid="15374"
userdom="">
          <vzDirAssDef childAction="deleteNonPresent" descr="" lcOwn="local"
modTs="2020-05-14T16:47:25.310+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" ownerKey="" ownerTag="" rn="dirass" status="">
              <vzConsDef anyDn=""
bdDefDn="uni/bd-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/BD-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-isSvc-no"
 bdDefStQual="none" childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 exceptionTag="" intent="install" isAny="no" isCifLblIntersEmpty="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" modTs="2020-05-14T16:47:25.324+00:00"
monPolDn="uni/tn-common/monepg-default" name="net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"
 nameAlias="" ownerKey="" ownerTag="" pcEnfPref="enforced" pcTag="49159"
```

```
prefGrMemb="exclude" prio="unspecified"
rn="cons-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]-any-no"
 scopeId="2555905" status="" targetDscp="unspecified" txId="7493989779947781613"
useAnyDef="no"/>
            <vzProvDef anyDn=""
bdDefDn="uni/bd-[uni/tn-common/BD-sauto-osd-bd-vlan105]-isSvc-no" bdDefStQual="none"
childAction="deleteNonPresent" ctrctUpd="ctrct"
ctxDefDn="uni/ctx-[uni/tn-common/ctx-sauto_l3out-1_vrf]" ctxDefStQual="none"
ctxPcTag="49153" ctxSeg="2555905" descr=""
epgDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105" exceptionTag=""
hasTriggeredAlloc="no" intent="install" isAny="no" isMsiteEPg="no"
l3CtxEncap="vxlan-2555905" lcOwn="local" matchT="AtleastOne"
modTs="2020-05-14T16:47:25.310+00:00" monPolDn="uni/tn-common/monepg-default"
name="sauto-osd-epg-vlan105" nameAlias="" ownerKey="" ownerTag="" pcEnfPref="enforced"
pcTag="49155" prefGrMemb="exclude" prio="unspecified"
rn="prov-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]-any-no"
scopeId="2555905" status="" targetDscp="unspecified" txId="1729382256910271363"
useAnyDef="no"/>
          </vzDirAssDef>
          <vzSubj  childAction="" configIssues="" consMatchT="AtleastOne" descr=""
extMngdBy="" lcOwn="local" modTs="2020-05-14T16:47:25.350+00:00"
monPolDn="uni/tn-common/monepg-default" name="ironic-tftp-server" nameAlias=""
prio="unspecified" provMatchT="AtleastOne" revFltPorts="yes" rn="subj-ironic-tftp-server"
 status="" targetDscp="unspecified" uid="15374" userdom="">
            <vzInTerm  childAction="" descr="" extMngdBy="" lcOwn="local"
modTs="2020-05-14T16:47:25.350+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" prio="unspecified" rn="intmnl" status="" targetDscp="unspecified" uid="15374"
 userdom="">
            <vzInTerm  childAction="" descr="" extMngdBy="" lcOwn="local"
modTs="2020-05-14T16:47:25.350+00:00" monPolDn="uni/tn-common/monepg-default" name=""
nameAlias="" prio="unspecified" rn="intmnl" status="" targetDscp="unspecified" uid="15374"
 userdom="">
              <vzRsFiltAtt action="permit"  childAction="" directives="" extMngdBy=""
 forceResolve="yes" lcOwn="local" modTs="2020-05-14T16:47:26.057+00:00"
monPolDn="uni/tn-common/monepg-default" priorityOverride="default" rType="mo"
rn="rsfiltAtt-ironic-tftp-server" state="formed" stateQual="none" status="" tCl="vzFilter"
 tContextDn="" tDn="uni/tn-common/flt-ironic-tftp-server" tRn="flt-ironic-tftp-server"
tType="name" tnVzFilterName="ironic-tftp-server" uid="15374" userdom=""/>
            </vzInTerm>
          </vzSubj>
          <vzRtProv childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.310+00:00"
rn="rtfvProv-[uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105]" status=""
 tCl="fvAEPg" tDn="uni/tn-common/ap-sauto_osd_infra_app/epg-sauto-osd-epg-vlan105"/>
          <vzRtCons childAction="" lcOwn="local" modTs="2020-05-14T16:47:25.324+00:00"
rn="rtfvCons-[uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef]"
 status="" tCl="fvAEPg"
tDn="uni/tn-prj_4660546efda44d6e9d9dba6670920894/ap-OpenStack/epg-net_1005e02b-2ed2-44eb-8ac5-6d989176ebef"/>

      </vzBrCP>
```

新しく作成されたコントラクトは、Ironic サービスをホストするアンダークラウド ネットワークに使用される EPG に適用する必要があります。これは、Cisco ACI GUI またはその他のサポートされている方法で実行できます。

ベア メタル ネットワークの作成

Ironic のクリーニング、プロビジョニング、およびレスキュー ネットワークは、総称してベア メタル ネットワークと呼ばれます。3 つの目的すべてに 1 つの Neutron ネットワークを使用するか、それぞれに個別のネットワークを使用することができます。Ironic サービスはインスタ

ンスと直接通信する必要があるため、これらのベア メタル ネットワークを作成するときに追加の手順を実行する必要があります。

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） およびこのセクションに先行する他のセクションのタスクを完了している必要があります。

手順

ステップ 1 **apic:extra_consumed_contracts** および **apic:extra_provided_contracts** 拡張機能を使用してベア メタル ネットワークを作成します。

これらの拡張機能は、Neutron ネットワーク上のサブネットが Neutron ルータに接続されると適用される追加の提供または消費されたコントラクトのリストを作成します。次の例は、Cisco Application Policy Infrastructure Controller（APIC）の common テナントで以前に作成されたベア メタル コントラクトを使用して、ネットワークを作成するときに拡張機能がどのように使用されるかを示しています。

例：

```
# neutron net-create baremetal_network --shared \
--apic:extra_consumed_contracts list=true ironic-services \
--apic:extra_provided_contracts list=true ironic-agents
```

コントラクトは、ベア メタル ホストと Ironic サービス間で必要なトラフィック タイプのみを許可する必要があります。また、Ironic サービスをホストするアンダークラウド ネットワークに使用されるエンドポイント グループ（EPG）によって、コントラクトもすでに提供および使用されている必要があります。

ステップ 2 ネットワークが、Ironic サービスで使用される EPG と Cisco Application Centric Infrastructure（ACI）の同じ VRF に属していることを確認します。

たとえば、Ironic サービスがアンダークラウド制御ネットワークに配置されている場合（デフォルト）、アンダークラウド制御ネットワークのブリッジ ドメインに使用される VRF は、オーバークラウドのベア メタル ネットワークにも使用する必要があります。

OpenStack のアドレス範囲を使用して、ネットワークが属する VRF を制御できます。`apic:distinguished_names` 拡張機能を使用すると、Cisco ACI の既存の VRF にマッピングするアドレス範囲を作成できます。次の例は、Cisco ACI の common テナントの下の制御 VRF にマッピングするアドレス範囲を OpenStack で作成する方法を示しています。

例：

```
# neutron address-scope-create baremetal_as_v4 4 \
    --apic:distinguished_names type=dict VRF=uni/tn-common/ctx-control
```

ステップ 3 Cisco ACI でサブネット プールを VRF に関連付け、そのサブネット プールを使用してベア メタル ネットワークにサブネットを割り当てます。

例：

```
# neutron subnetpool-create --pool-prefix 40.40.40.0/24 \
    --address-scope baremetal_as_v4 baremetal_v4_sp

# openstack subnet create --network baremetal_network \
    --subnet-pool baremetal_v4_sp --dhcp  baremetal_subnet \
    --prefix-length 24 baremetal-subnet
```

ステップ 4 次のコマンドを使用して、Ironic構成ファイルを更新し、新しく作成されたベア メタル ネットワークを使用します。

```
# crudini --set /var/lib/config-data/puppet-generated/ironic/etc/ironic/ironic.conf
neutron provisioning_network <provisioning network UUID>
# crudini --set /var/lib/config-data/puppet-generated/ironic/etc/ironic/ironic.conf
neutron cleaning_network <cleaning network UUID>
# crudini --set /var/lib/config-data/puppet-generated/ironic/etc/ironic/ironic.conf
neutron rescuing_network <rescuing network UUID>
```

ステップ 5 Ironic コンテナを再起動します。

例：

```
# docker restart ironic_pxe_http \
                 ironic_pxe_tftp \
                 ironic_conductor \
                 ironic_api
```

Ironic サービスへのベア メタル ネットワークの接続

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） およびこのセクションに先行する他のセクションのタスクを完了している必要があります。

手順

ステップ 1 Neutron ネットワークのサブネットを Neutron ルータに接続します。

ベア メタル ネットワーク拡張で指定されたコントラクトは、添付ファイルを作成した後に適用されます。

例：

```
# openstack router create baremetal-router
# openstack router add subnet baremetal-router baremetal-subnet
```

ステップ 2 Ironic サービスを実行しているホストにルーティング テーブル エントリを追加します。

通常、ホストのデフォルト ルートは、Ironic サービスに使用されるインターフェイスを経由しません。サブネット ルートを追加して、ベア メタル ネットワークに到達するために Ironic サービス インターフェイスが使用されるようにします。Cisco Application Policy Infrastructure Controller（APIC）がサービスのサブネット ゲートウェイを提供できるように、Ironic サービスをホスト

するアンダークラウド ネットワークに使用されるブリッジ ドメインにサブネットを設定する必要があります。（コントロールプレーンネットワークがIronicサービスに使用されている場合、.1 アドレスはCIDR に使用できません。これは、アンダークラウドVM で使用するために割り当てられているためです）。

次に、サブネット ゲートウェイ 1.100.1.254（アンダークラウド コントロール プレーン ネットワークに使用される CIDR）を介して、40.40.40.0/24 サブネットを持つベア メタル ネットワークに到達するスタティック ルートの例を示します。

```
# route add -net 40.40.40.0 netmask 255.255.255.0 \
   gateway 1.100.1.254 dev br-baremetal
```

リブート後もルートが保持されるようにするには、ルートをネットワーク初期化スクリプトに追加します。たとえば、/etc/sysconfig/network-scripts/route-br-baremetal などです。

ベア メタル ノードを登録する

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） およびこのセクションに先行する他のセクションのタスクを完了している必要があります。

手順

ベア メタル ノードを登録します。

Red Hat Web サイトの 『*Red Hat OpenStack Platform 13 Bare Metal Provisioning*』 の「Adding Physical Machines as Bare Metal Nodes」セクションの手順に従います。

ベア メタル ポートおよびポート グループの作成

この手順は、OpenStack Platform（OSP）13 ドキュメントの手順とほぼ同じです。ただし、ベアメタルインスタンスがNeutronネットワークに接続されるたびにファブリックを正しく再設定できるように、Ironic ポートの追加のトポロジ情報を指定する必要があります。

Ironic ポートの local-link-connection 部分で `switch_id` パラメータを使用してトポロジ情報を指定します。このパラメータは、カンマで区切られた一連のキーと値のペアで構成される文字列です。現在、次の 3 つのパラメータがサポートされています。

- `apic_dn`：

このパラメータは、静的パス バインドに使用されるCisco Application Policy Infrastructure Controller（APIC）内の識別名（DN）を指定します。単一のインターフェイスを使用する場合の DN の形式は次のとおりです。

```
topology/pod-ポッド番号/パス-ノード/pathep-ポートまたはポリシー グループ名
```

次に例を示します。

```
topology/pod-1/paths-101/pathep-[eth1/2]
```

次に、仮想ポート チャネル（vPC）を使用する場合の DN の形式を示します。

```
topology/pod-pod number/protpaths-nodes/pathep-vpc
```

- `physical_network`：

  このパラメータは、VLANを静的パスに割り当てる必要がある場合に使用される物理ネットワークです。

- `physical_domain`：

  このパラメータは、静的ポートが作成されるエンドポイントグループ（EPG）に関連付けられる PhysDom の名前です。このパラメータはオプションです。

このセクションでは、ベア メタル ポートの local_link_connection フィールドにトポロジ情報を設定する方法を示します。ベア メタル ノード ID とそれに関連付けられたベア メタル ポート ID は、それぞれの環境変数で定義されていることを前提としています。

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） およびこのセクションに先行する他のセクションのタスクを完了している必要があります。

手順

ステップ 1 トポロジ情報を使用してベア メタル ポートを更新します。

例：

```
# export VPC="topology/pod-1/protpaths-501-502/pathep-[ironic-vpc]"
# openstack baremetal port set --node ${NODE_ID} \
    --local-link-connection port_id="Eth1/1" \
    --local-link-connection switch_id="00:be:75:8f:25:a1"  \
    --local-link-connection \
      switch_info="apic_dn:${VPC},physical_network:physnet1 \
    ${BAREMETAL_PORT1_ID}
# openstack baremetal port set --node ${NODE_ID} \
    --local-link-connection port_id="Eth1/1" \
    --local-link-connection switch_id="00:be:75:8f:25:a2"  \
    --local-link-connection \
      switch_info="apic_dn:${VPC},physical_network:physnet1 \
${BAREMETAL_PORT2_ID}
```

port_id と switch_id は現在プラグインでは使用されていませんが、入力する必要があることに注意してください。

ステップ 2 ベア メタル ポート グループを作成します。

例：

```
# PORT_GROUP_ID=$(openstack baremetal port group create
    --node ${NODE_ID} --name bond1 --address 00:fc:ba:e8:86:4c \
    --mode 802.3ad --property miimon=100 \
    --property xmit_hash_policy='layer2' --support-standalone-ports
```

ステップ 3 Ironic ポートを Ironic ポート グループに追加します。

例：

```
# openstack baremetal port set --node ${NODE_ID} \
     --port-group ${PORT_GROUP_ID} ${BAREMETAL_PORT1_ID}

# openstack baremetal port set --node ${NODE_ID} \
     --port-group ${PORT_GROUP_ID} ${BAREMETAL_PORT2_ID}
```

ベア メタル フレーバーとアベイラビリティ ゾーンの作成

始める前に

Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） およびこのセクションに先行する他のセクションのタスクを完了している必要があります。

手順

ベア メタル ポートとポート グループを作成します。

Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従います。

ベアメタル インスタンスを開始する

始める前に

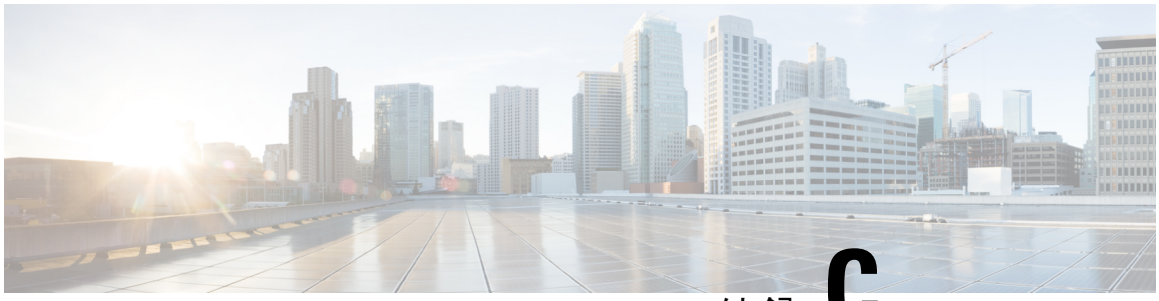
Ironic をサポートするための OpenStack の展開の前提条件 （44 ページ） の項のタスクを実行します。

手順

ベアメタル インスタンスを開始します。

OpenStack の Ironic の構成

Red Hat の Web サイトにある Red Hat OpenStack Platform 13 ドキュメントの手順に従います。



付録 C

詳細設定

詳細設定

Virtual Machine Manager（VMM）ドメインに VXLAN カプセル化を使用する大規模なインストールでは、追加の設定が必要になる場合があります。

ホストのエンドポイント グループ（EPG）の最大数と一致するように、マルチキャスト グループの数を設定できます。ソケットの最大補助メモリを増やし、コンピューティング ノードまたはコントローラ ノードにカーネル ブート パラメータを追加することもできます。

マルチキャスト グループの構成とソケットのメモリの増加

手順

マルチキャスト グループを設定し、ソケットのメモリを増やし、展開テンプレートの parameter_defaults に次のパラメータを追加します。

例：

```
parameter_defaults:
  ControllerParameters:
      ExtraSysctlSettings:
         net.ipv4.igmp_max_memberships:
             value: 4096
         net.core.optmem_max:
             value: 1310720
   ComputeParameters:
      ExtraSysctlSettings:
```

```
    net.ipv4.igmp_max_memberships:
        value: 1024
```

IGMP メンバーシップの最大値は、ホストの Neutron ポートがオンになっている Neutron ネットワークの数以上である必要があります。たとえば、コンピューティング ホストに 100 個のインスタンスがあり、各インスタンスが異なる Neutron ネットワーク上にある場合、この数を 100 以上に設定する必要があります。neutron-dhcp-agent を実行しているコントローラ ホストは、そのエージェントによって管理される Neutron ネットワークの数と一致するようにこの値を設定する必要があります。つまり、この数は、おそらくコンピューティング ホストよりもコントローラ ホストで大きくする必要があります。

追加のカーネル ブート パラメータの追加

resource_registry および parameter_defaults ファイルを変更して、コンピューティング ノードまたはコントローラ ノードにカーネル ブート パラメータを追加します。

手順

ステップ 1 resource_registry ファイルを変更します。

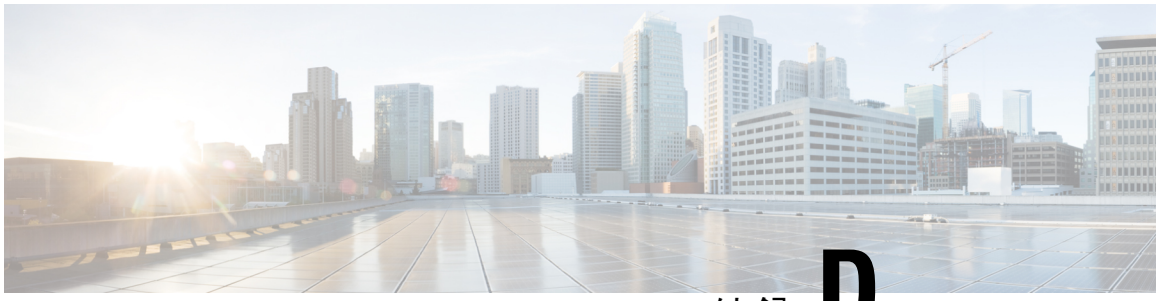
例：

```
resource_registry:
     OS::TripleO::Compute::PreNetworkConfig:
/usr/share/openstack-tripleo-heat-templates/extraconfig/pre_network/host_config_and_reboot.yaml
```

ステップ 2 parameter_defaults ファイルを変更します。

例：

```
parameter_defaults:
    ComputeParameters:
        KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G
hugepages=60"
```



付録 D

参考情報

階層型ポート バインディングの構成

このセクションでは、OpFlexプラグインと連携するようにシングルルートI/O仮想化（SR-IOV）およびその他の VLAN ベースの ml2 メカニズムエージェントを設定する方法について説明します。構成は階層型ポート バインディング（HPB）を使用して実行され、構成を特別に変更しなくても機能します。OpFlex に SR-IOV を構成するために必要な基本手順は次のとおりです。

HPB を使用する場合、Cisco Application Centric Infrastructure（ACI）でのデータ パス接続は、OpenStack によって作成されたネットワークの EPG へのスタティック VLAN バインディングを作成することによって実現されます。SR-IOV NIC での VLAN の設定や OVS（または LBaaS の場合はロードバランサ）の設定など、データ パスに他の設定が必要な場合があります。これは、サードパーティのエージェントまたはメカニズム ドライバ（sriovnicswitch など）によって実行します。

これらの資産を作成する方法：

始める前に

スタティック VLAN バインディングを使用してデータ パスを設定するには、プラグインに次のアセットが必要であることを確認します。

- 物理ドメイン（physdom）と適切な VLAN プール。
- host-link 情報（どのコンピューティング ノードのファブリック イーサネット インターフェイスがどのリーフ スイッチ ポートに接続されているかを示す情報）

- host-link-network-label 情報（コンピューティング ノードのどのファブリック イーサネット インターフェイスが、どの物理ネットワークの提供に使用されているかを示す情報）
- この情報は、展開で複数の physnet を使用する場合にのみ必要です。

手順

OpenStack Platform オーバークラウドを導入する前に、必要な物理ネットワークごとに 1 つの物理ドメインが作成済みであることを確認します。作成した物理ドメインの名前に pdom_ プレフィックスを追加します。たとえば、physnet1 の場合は、pdom_physnet1 を作成し、適切な VLAN プールを接続します。

また、NeutronNetworkVLANRanges も設定し、ACIMechanismDrivers パラメータを使用してサードパーティのメカニズム ドライバを有効にして、apic_aim をリスト内の最後のメカニズムにする必要があります。

例：

```
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"}]}'
```

Cisco ACI 環境のパラメータ

次の表に、Cisco Application Centric Infrastructure（ACI）環境の構成に必要なパラメータに関する情報を示します。

パラメータ	詳細
NeutronCorePlugin	• **値**： 'ml2plus' • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし
NeutronServicePlugins	• **値**： 'group_policy,ncp,apic_aim_l3' • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし

パラメータ	詳細
NeutronEnableIsolatedMetadata	• **値**：true • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：true に設定
NeutronEnableForceMetadata	• **値**：true • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：true に設定
ACIYumRepo	• **値**：http://undercloud pxe network ipaddress:8787/v2/__acirepo • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし
ACIApicHosts	• **値**： Cisco Application Policy Infrastructure Controller（APIC）名前とアドレス • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし
ACIApicUsername	• **値**：管理者権限を持つユーザー名 • **デフォルト**：admin • **必須またはオプション**：オプション • **コメント**：なし
ACIApicPassword	• **値**：パスワード • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし （注） 証明書ベースの認証を使用する場合は、このパラメータを指定しないでください。

参考情報

パラメータ	詳細
ACIMechanismDrivers	• **値**：「apic_aim」 • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：追加のドライバを加えます。たとえば、neutron ovs エージェントを使用する場合はOpen vSwitch 用、sriov を使用する場合はsriovnicswitch 用です
ACIApicEntityProfile	• **値**：Cisco ACI で事前にプロビジョニングされたCisco ACIエンティティ プロファイル • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし
ACIApicInfraVlan	• **値**：Cisco ACIファブリック インフラ VLAN • **デフォルト**：4093 • **必須またはオプション**：オプション • **コメント**：正しい値については、Cisco ACI 管理者にお問い合わせください。
ACIApicInfraSubnetGateway	• **値**：Cisco ACIインフラ サブネット ゲートウェイ • **デフォルト**：10.0.0.30 • **必須またはオプション**：オプション • **コメント**：正しい値については、Cisco ACI 管理者にお問い合わせください。
ACIApicInfraAnycastAddr	• **値**：Cisco ACIエニーキャスト アドレス • **デフォルト**：10.0.0.32 • **必須またはオプション**：オプション • **コメント**：正しい値については、Cisco ACI 管理者にお問い合わせください。

パラメータ	詳細
ACIUseLldp	• **値**：true または false • **デフォルト**：true • **必須またはオプション**：オプション • **コメント**：false に設定する場合は、CiscoAciLldp サービスを `OS::Hat:None` に設定します。
ACIOpflexUplinkInterface	• **値**：Cisco ACI リーフ スイッチに接続されているインターフェイス名 • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：実際のインターフェイス名（例：enp8s0）
ACIOpflexEncapMode	• **値**：vxlan または vlan • **デフォルト**：vxlan • **必須またはオプション**：オプション • **コメント**：なし
ACIOpflexVlanRange	• **値**： *starting_vlan*:*ending_vlan* • **デフォルト**：なし • **必須またはオプション**：ACIOpflexEncapMode が vlan に設定されている場合は必須 • **コメント**：なし
ACIOpflexInterfaceType	• **値**：linux または ovs • **デフォルト値**：linux • **必須またはオプション**：オプション • **コメント**：「OpenShift on OpenStack」のネストされたインストールを展開する場合は、この値を「ovs」に設定します。この設定により、Ovs スイッチで OpFlex インターフェイスが作成されます。

パラメータ	詳細
ACIOpflexInterfaceMTU	• **値**：意図された MTU サイズ • **デフォルト**：1500 • **必須またはオプション**：オプション • **備考**： このパラメータを使用して、OpFlex インターフェイスの MTU を設定します。OpenStack に OpenShift をインストールするには、これを 8000 に設定する必要があります。
NeutronPluginMl2PuppetTags	• **値**：'neutron_plugin_ml2,neutron_plugin_cisco_aci,neutron_sfc_service_config' • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：なし
NeutronNetworkVLANRanges	• **値**： *physnet*:*starting vlan*:*end vlan*（例：physnet1:1100:1150,physnet2:1201:1211） • **デフォルト**：なし • **必須またはオプション**：neutron ovs エージェントを使用する場合は必須 • **コメント**：なし
AciPhysDomMappings	• **値**：例：'physnet0:my_pdom0, physnet1:my_pdom1' • **デフォルト**：なし • **必須またはオプション**：オプション • **コメント**：*physical_network* のリスト：*ACI Physdom*。デフォルトでは、各 physnet は **pdom**_*physnet_name* を使用して事前に作成された ACI physdom にマッピングされます。たとえば、**physnet0** は **pdom_phynet0** という名前の physdom にマッピングされます。このパラメータを使用すると、ユーザーはマッピングを上書きできます。

パラメータ	詳細
NeutronBridgeMappings	• **値**：例：'physnet1:br-ex,physnet2:br-ex' • **デフォルト**：なし • **必須またはオプション**：neutron ovs エージェントを使用する場合は必須 • **コメント**：Physnets は、NeutronBridgeMappings で指定されたとおりに一致する必要があります。
AciTenantNetworkType	• **値**：vlan • **デフォルト**：なし • **必須またはオプション**：neutron ovs エージェントを使用する場合は必須 • **コメント**：なし
AciOpenvswitch	• **値**：true または false • **デフォルト**：false • **必須またはオプション**：neutron ovs エージェントを使用する場合は true に設定します。 • **コメント**：なし
NeutronOVSFirewallDriver	• **値**：'neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver' • **デフォルト**：なし • **必須またはオプション**：必須 • **コメント**：neutron ovs エージェントを使用する場合に表示される値に設定します。
ACIHostLinks	• **値**：例：'{"101":{"ha.dom":"1/1", "hb.dom":"1/2"}, "102":{"hc.dom":"1/1"} }' • **デフォルト**：なし • **必須またはオプション**：neutron ovs エージェントを使用し、lldp エージェントを使用しない場合は必須です。 • **コメント**：スイッチへのホスト接続を JSON 形式で記述します。 この例では、ha.dom ホストはスイッチ ID 101 のポート 1/1 に接続され、hb.dom ホストはスイッチ ID 101 のポート 1/2 に接続され、hc.dom はスイッチ ID 102 のポート 1/1 に接続されます。

参考情報

パラメータ	詳細
NeutronPhysicalDevMappings	• **値**： • **デフォルト**：なし • **必須またはオプション**：オプション • **備考**：
NeutronPhysicalDevMappings	• **値**：例：physnet1:eth1,physnet2:eth2 • **デフォルト**：なし • **必須またはオプション**：オプション • **コメント**：特定のインターフェイスを特定の physnet にマッピングする場合は、このパラメータを設定する必要があります。
ACIApicCertName	• **値**：Cisco APIC 証明書ユーザーの名前（証明書ベースの認証に使用） • **型**：文字列 • **デフォルト**：なし • **必須またはオプション**：オプション
ACIApicPrivateKey	• **値**：cert User の秘密キー • **型**：文字列 • **デフォルト**：なし • **必須またはオプション**：オプション
AciKeystoneNotificationPurge	• **値**：True または False • **型**：ブール型 • **デフォルト**：False • **コメント**：OpenStack でプロジェクトが削除されたときの Cisco APIC テナントの自動消去を有効にします。

パラメータ	詳細
NeutronPluginExtensions	• **値**：有効な拡張プラグインのカンマ区切りリスト。 • **デフォルト**：apic_aim、port_security • **必須またはオプション**：オプション • **コメント**：パラメータが明示的に設定されている場合の推奨値は、apic_aim、port_security、qos です。
NeutronMechanismDrivers	• **値**：'apic_aim' • **デフォルト**：ovn • **必須またはオプション**：必須（Octavia が展開されている場合） • **コメント**：Octavia を展開すると、デフォルトで **ovn** プロバイダーが展開されます。Octavia の展開では、プロバイダーを定義するために NeutronMechanismDrivers が考慮されます。Amhora 展開では、**ovn** を明示的に設定する必要があります。

リソース宣言の例

次に、Cisco Application Centric Infrastructure（ACI）リソース宣言（`ciscoaci-env.yaml`）の完全な例を示します。

次に、Cisco Application Centric Infrastructure（ACI）のリソース宣言（`ciscoaci-env.yaml`）の完全な例を示します。提供されている例は、Cisco ACI リリース 5.2(1) 以降用です。オーバークラウドをインストールするときに、Cisco ACI 環境のリソースを宣言します。このガイドの手順オーバークラウドのインストール （11 ページ）を参照してください。

```
# A Heat environment file which can be used to enable a
# a Neutron Cisco Aci backend on the controller, configured via puppet
resource_registry:

  #controller
 OS::TripleO::ControllerExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml

  OS::TripleO::Services::NeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Docker::NeutronMl2PluginBase:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron/neutron-ml2-ciscoaci.yaml
  OS::TripleO::Services::CiscoAciAIM:
/opt/ciscoaci-tripleo-heat-templates/deployment/aciaim/cisco-aciaim-container-puppet.yaml

  OS::TripleO::Services::NeutronMetadataAgent:
/usr/share/openstack-tripleo-heat-templates/deployment/neutron/neutron-metadata-container-puppet.yaml

  #compute
 OS::TripleO::ComputeExtraConfigPre: /opt/ciscoaci-tripleo-heat-templates//nodepre.yaml
```

参考情報

```
  OS::TripleO::Services::ComputeNeutronOvsAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/neutron_opflex/neutron-opflex-agent-container-puppet.yaml

  OS::TripleO::Services::ComputeNeutronMetadataAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/compute_neutron_metadata/compute-neutron-metadata.yaml


 OS::TripleO::Services::CiscoAciLldp:
/opt/ciscoaci-tripleo-heat-templates/deployment/lldp/cisco_lldp.yaml
  OS::TripleO::Services::CiscoAciOpflexAgent:
/opt/ciscoaci-tripleo-heat-templates/deployment/opflex/opflex-agent-container-puppet.yaml



  OS::TripleO::Services::OVNDBs: OS::Heat::None
  OS::TripleO::Services::OVNController: OS::Heat::None
  OS::TripleO::Services::OVNMetadataAgent: OS::Heat::None
  OS::TripleO::Services::ComputeNeutronL3Agent: OS::Heat::None
  OS::TripleO::Services::NeutronL3Agent: OS::Heat::None


parameter_defaults:
  NeutronSfcDriver: 'aim'
  NeutronFcDriver: 'aim'
  NeutronCorePlugin: 'ml2plus'
  NeutronServicePlugins: 'group_policy,ncp,apic_aim_l3'
  NeutronPluginMl2PuppetTags: 'neutron_plugin_ml2,neutron_plugin_cisco_aci'
  NeutronEnableIsolatedMetadata: true
  EnablePackageInstall: true
  ACIYumRepo: http://10.10.250.67:8787/v2/__acirepo
  ACIApicHosts: 10.105.1.10
  ACIApicUsername: admin
  ACIApicPassword: password
  ACIApicSystemId: osp16.2
  ACIUseLLDPDiscovery: 'true'
  ACIApicEntityProfile: OSP16.2
  ACIApicInfraVlan: 4093
  ACIApicInfraSubnetGateway: 10.0.0.30
  ACIApicInfraAnycastAddr: 10.0.0.32
  ACIOpflexUplinkInterface: ens8
  ACIOpflexEncapMode: vxlan
  ACIOpflexVlanRange: 1200:1300
  ACIYumRepoMetadataExpiry: 90
  DockerInsecureRegistryAddress: ["director16.2.ctlplane.localdomain:8787",
"10.10.250.67:8787"]
```

（注） Cisco ACI リリース 5.2(1) より前のリリースを展開する場合は、上記の例に次の変更を加える必要があります。

- `OS::TripleO::Services::CiscoAciOpflexAgent` の定義を削除します。
- `/opt/ciscoaci-tripleo-heat-templates/deployment/opflex/opflex-agent-container-puppet.yaml` テンプレートを参照するように、`OS::TripleO::Services::NeutronOvsAgent` と `OS::TripleO::Services::ComputeNeutronOvsAgent` を変更します。

ホスト レポートの作成例

Judith の以前のバージョンから作成されたトピック。~ catortiz 2020 年 11 月 22 日

トラブルシューティング中に、OpenStack クラスタからホストレポートを収集することが必要な場合があります。これは、提供されているプレイブック `/opt/ciscoaci-tripleo-heat-templates/tools/report.yaml` を使用して行います。このセクションでは、ホスト レポート プレイブックの使用例を示します。

- `ansible-playbook /opt/ciscoaci-tripleo-heat-templates/tools/report.yaml`

  この例では、すべてのノードからデータを収集し、ファイル `/home/stack/overcloud_aci_report.tgz` を作成します。

- `ansible-playbook /opt/ciscoaci-tripleo-heat-templates/tools/report.yaml -e '{"limit_fraors":['control'], "dest_file":/tmp/abc}`

  この例では、レポートをコントローラに制限し、デフォルトの出力ファイルを変更します。

- `ansible-playbook /opt/ciscoaci-tripleo-heat-templates/tools/report.yaml -e '{"limit_hosts":[overcloud-controller-0, overcloud-controller-2]}'`

  この例では、レポート収集を指定されたホストに制限します。「`limit_fraors`」と「`limit_hosts`」をクラブして、データを収集するノードをさらにフィルタリングできます。

TLS を使用した展開

Transport Layer Security（TLS）を使用した Red Hat OpenStack 16.2 の展開は、サポートされている構成です。OpenStack エンドポイントで TLS を有効にするには、Red Hat Web サイトの「*Advanced Overcloud Customization*」の手順に従ってください。

AIM とCisco Application Policy Infrastructure Controller（APIC）間の TLS を有効にするには、このガイドの オーバークラウドのインストール （11 ページ） のステップ 4 で説明されている証明書ベース認証手順に従います。

Cisco ACI コンテナ イメージのクリーンアップ

cisco コンテナ イメージ生成

`/opt/ciscoaci-tripleo-heat-templates/tools/build_openstack_aci_containers.py` スクリプトを複数回実行すると（マイナーアップデートの場合でも）、古いCisco ACI コンテナイメージがリポジトリに残ります。古いコンテナイメージをクリアするには、次の手順を使用します。

手順

ステップ 1 最新のビルド イメージの タグを見つけます。

`ciscoaci_containers.yaml` ファイルで最新のイメージ タグを確認できます。次の例では、タグは 1614292118 です。

```
[stack@director16 ~]$ cat templates/ciscoaci_containers.yaml
parameter_defaults:
ContainerHorizonImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-horizon-ciscoaci:1614292118
ContainerHeatEngineImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-heat-engine-ciscoaci:1614292118
ContainerNeutronApiImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-neutron-server-ciscoaci:1614292118
ContainerNeutronConfigImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-neutron-server-ciscoaci:1614292118
ContainerCiscoLldpImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-ciscoaci-lldp:1614292118
ContainerCiscoAciAimImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-ciscoaci-aim:1614292118
ContainerCiscoAciAimConfigImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-ciscoaci-aim:1614292118
ContainerOpflexAgentImage:
director16.ctlplane.localdomain:8787/ciscoaci/openstack-ciscoaci-opflex:1614292118
[stack@director16 ~]$
```

ステップ 2 リポジトリ内の古いイメージを特定するには、**sudo openstack Tripleo container image list** コマンドを使用します。

次の例は、新しいイメージと古いイメージを示しています。ステップ 1 の例を参照すると、新しいイメージは 1614292118 で示され、その他は古いイメージです。

```
[stack@director16 ~]$ sudo openstack tripleo container image list|grep cisco
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-horizon-ciscoaci:1613593371
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-horizon-ciscoaci:1613614575
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-horizon-ciscoaci:1614292118
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-heat-engine-ciscoaci:1613593371
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-heat-engine-ciscoaci:1613614575
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-heat-engine-ciscoaci:1614292118
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-neutron-server-ciscoaci:1613593371
 |
|
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-neutron-server-ciscoaci:1613614575
 |
|
```

```
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-neutron-server-ciscoaci:1614292118
```

ステップ 1 の例を参照すると、新しいイメージは 1614292118 で示され、その他は古いイメージです。

ステップ 3 リポジトリから古いイメージを削除するには、**sudo openstack Tripleo container image delete** コマンドを使用します。

```
[stack@director16 ~]$ sudo openstack tripleo container image delete \
docker://director16.ctlplane.localdomain:8787/ciscoaci/openstack-horizon-ciscoaci:1613593371
```

参考情報

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。