Cisco ACI OpenStack プラグイン ユーザ ガイド

初版：2018 年 12 月 21 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

はじめに

この前書きは、次の項で構成されています。

対象読者

このガイドは、アプリケーション セントリック インフラストラクチャ ファブリック の設定および維持に携わるネットワーク管理者およびシステム管理者を対象としています。

表記法

コマンドの説明には、次のような表記法が使用されます。

表記法	説明
bold	太字の文字は、表示どおりにユーザが入力するコマンドおよびキーワードです。
italic	イタリック体の文字は、ユーザが値を入力する引数です。
[x]	省略可能な要素（キーワードまたは引数）は、角カッコで囲んで示しています。
[x | y]	いずれか1つを選択できる省略可能なキーワードや引数は、角カッコで囲み、縦棒で区切って示しています。
{x | y}	必ずいずれか1つを選択しなければならない必須キーワードや引数は、波カッコで囲み、縦棒で区切って示しています。

表記法	説明
[x {y | z}]	角カッコまたは波カッコが入れ子になっている箇所は、任意または必須の要素内の任意または必須の選択肢であることを表します。角カッコ内の波カッコと縦棒は、省略可能な要素内で選択すべき必須の要素を示しています。
`variable`	ユーザが値を入力する変数であることを表します。イタリック体が使用できない場合に使用されます。
string	引用符を付けない一組の文字。string の前後には引用符を使用しません。引用符を使用すると、その引用符も含めて string とみなされます。

例では、次の表記法を使用しています。

表記法	説明
`screen フォント`	スイッチが表示する端末セッションおよび情報は、screen フォントで示しています。
太字の screen フォント	ユーザが入力しなければならない情報は、太字の screen フォントで示しています。
イタリック体の *screen* フォント	ユーザが値を指定する引数は、イタリック体の screen フォントで示しています。
< >	パスワードのように出力されない文字は、山カッコ（< >）で囲んで示しています。
[ ]	システム プロンプトに対するデフォルトの応答は、角カッコで囲んで示しています。
!、#	コードの先頭に感嘆符（!）またはポンド記号（#）がある場合には、コメント行であることを示します。

このマニュアルでは、次の表記法を使用しています。

（注） 「注釈」です。役立つ情報やこのマニュアルに記載されていない参照資料を紹介しています。

注意 「要注意」の意味です。機器の損傷またはデータ損失を予防するための注意事項が記述されています。

警告 安全上の重要な注意事項

この警告記号は危険を意味します。人身事故を予防するための注意事項が記述されています。機器の取り扱い作業を行うときは、電気回路の危険性に注意し、一般的な事故防止対策に留意してください。各警告の最後に記載されているステートメント番号を基に、装置に付属の安全についての警告を参照してください。

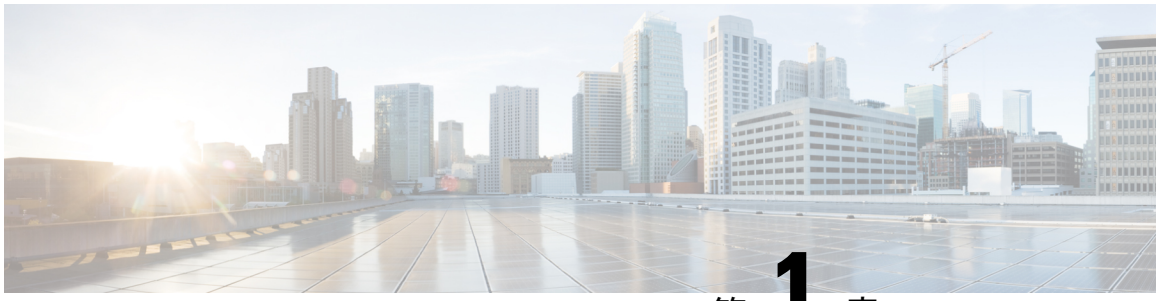
これらの注意事項を保存しておいてください。

関連資料

APIC の機能と運用については、『*Cisco Application Centric Infrastructure Fundamentals*』を参照してください。

マニュアルに関するフィードバック

このマニュアルに関する技術的なフィードバック、または誤りや記載もれなどお気づきの点がございましたら、apic-docfeedback@cisco.com までご連絡ください。ご協力をよろしくお願いいたします。



第 1 章

新機能および変更された機能に関する情報

この章の内容は、次のとおりです。

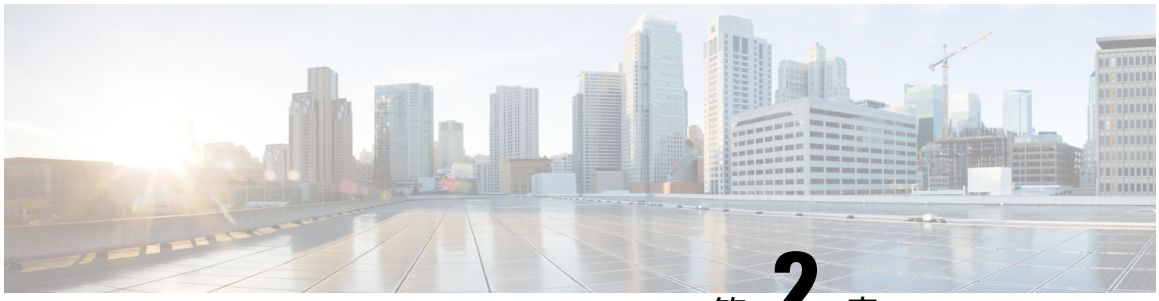
新機能および変更された機能に関する情報

次の表は、この最新リリースまでのガイドでの主な変更点の概要を示したものです。ただし、このリリースまでのこのガイドの変更点や新機能の中には一部、この表に記載されていないものもあります。

表 *1*: 新機能と変更された動作

Cisco APIC のリリース バージョン	機能	説明	参照先
3.0(1)	--	このガイドがリリースされました。	--

新機能および変更された機能に関する情報



第 2 章

グループベース ポリシー

この章の内容は、次のとおりです。

グループベース ポリシーの概要

グループベース ポリシー（GBP）はOpenStack用のAPIフレームワークです。基盤となるインフラストラクチャに依存しない方法でアプリケーション要件を説明することを目的としたインテント主導型モデルを提供します。GBPはレイヤ2ドメインなどのネットワーク中心の構成要素を提供するのではなく、グループ間の接続、セキュリティ、およびネットワーク サービスを説明するポリシー モデルとともに、汎用の「グループ」プリミティブを導入します。ネットワーク ドメインに重点が置かれているGBPですが、ネットワーキングを越えた一般的なフレームワークとして機能できます。

GBP は Neutron プロセス空間内のサービス プラグインとして実行されます。

グループベース ポリシーの使用例

グループベース ポリシー（GBP）は、OpenStack クラウドでアプリケーション要件をキャプチャして複雑なアプリケーションを導入するための強力かつシンプルな言語の提供を目的に設計されました。GBPは、アプリケーション要件を理解しているアプリケーション開発者とさまざまなインフラストラクチャ機能を理解しているインフラストラクチャ チーム間の差異を解消します。

GBP は OpenStack に備わっていない次の機能を提供します。

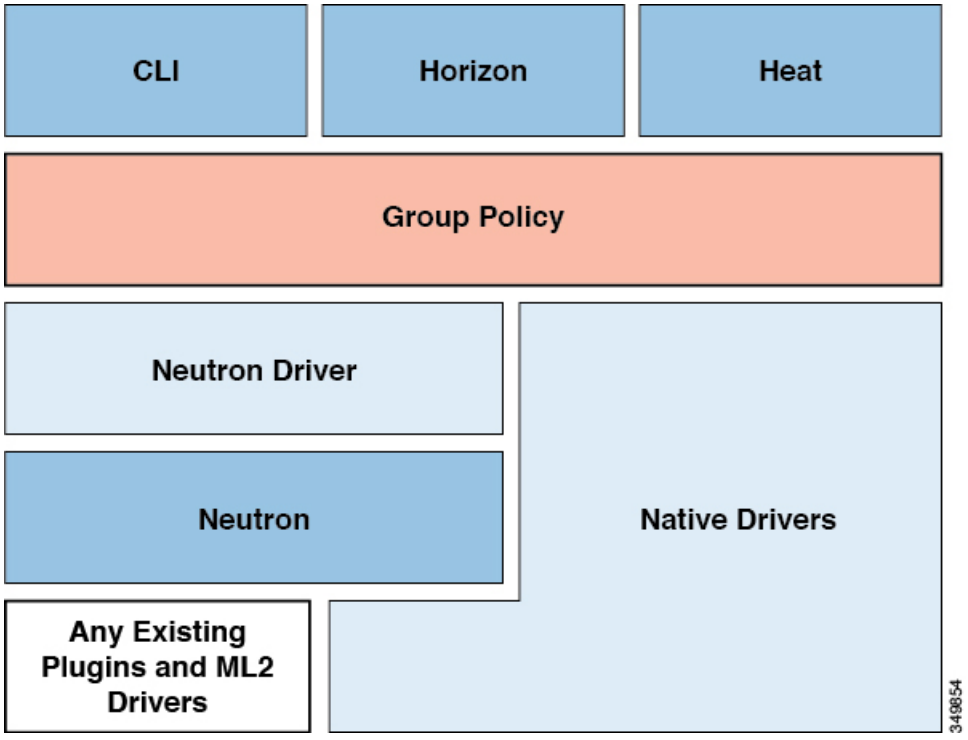
依存関係マッピング	ユーザは GBP によって、階層が異なるアプリケーション間の関係を指定できます。この依存関係マップはアプリケーションのセキュリティ要件ドキュ

	メントとして機能し、異なる階層のアプリケーションの個別の進化を可能にします。依存関係マップにより、インフラストラクチャのスケーリングおよび自動化も非常に簡単に行えます。
考慮事項の分離	GBP は、アプリケーションのセキュリティ要件（通信できる相手など）をネットワーク固有の要件（使用する IP アドレス範囲、ネットワーク境界の位置、仮想 IP アドレスの割り当て方法など）と分離するために設計されました。これにより、アプリケーション チーム、セキュリティ チーム、および運用チームがそれぞれ独立しながら協調的に業務を行えます。

グループベース ポリシーのしくみ

グループベース ポリシー（GBP）は、Horizon 拡張機能（openstack-dashboard-gbp）、Heat（openstack-heat-gbp）、cli（openstack-neutron-gbp）などの複数の OpenStack インターフェイスを介してポリシー API を提供します。GBP は Neutron 上のレイヤとして機能するように設計されています。

GBP では、基盤となるインフラストラクチャへのマッピング形式として次の 2 つがサポートされます。



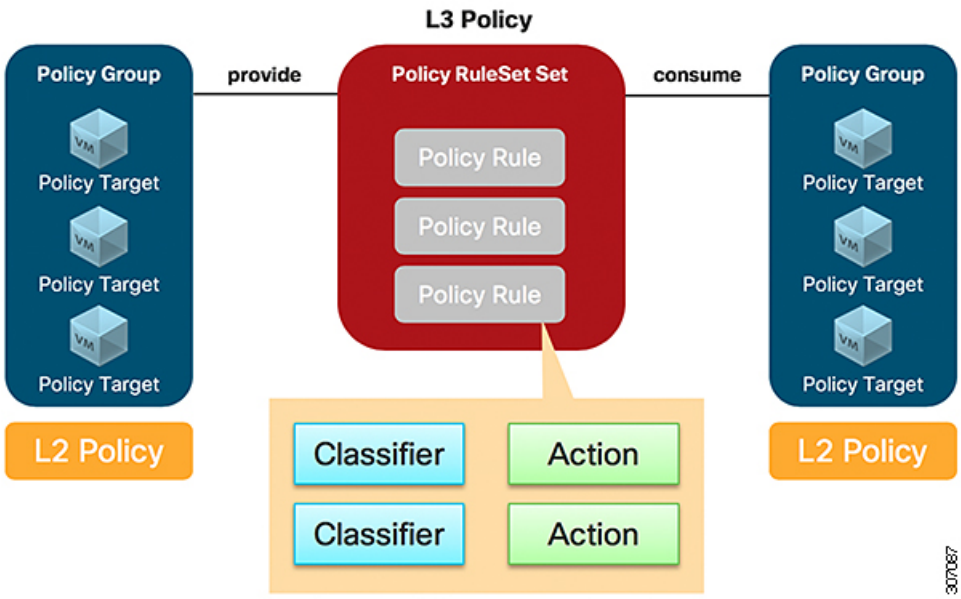
Neutron マッピング ドライバ	Neutron マッピング ドライバは、GBP リソースを既存の Neutron API コールに変換します。これにより、Neutron で既存のオープン ソースやベンダー プラグイン（ML2 など）を実行できます。また、OpenStack 環境で GBP を使用できるようにします。Neutron API または GBP API を使用して OpenStack プロジェクトを定義する必要があります。同じ OpenStack インストールの

	別のプロジェクトでは、異なる API が使用される可能性があります（例：Neutron または GBP）。
ネイティブ ドライバ	ポリシーの構成要素を最初に Neutron API に変換せずに、別の SDN コントローラまたは外部エンティティ経由で直接レンダリングするドライバを作成できます。これにより、コントローラが L2 または L3 の動作に関係なく、より柔軟にポリシーを解釈および適用できるようになるため重要です。現在は、Cisco APIC、VMware NSX ポリシー、および One Convergence NSP を含む 4 つのネイティブ ドライバがあります。

グループベース ポリシー モデルの概要

グループベース ポリシー（GBP）は、さまざまな論理グループやアプリケーション層の関係を説明するポリシーモデルを導入します。基盤となるインフラストラクチャ機能からセマンティクスを分離するようにプリミティブが選択されています。リソースは、特定のテナントに対してパブリックの場合もローカルの場合もあります。

主なプリミティブは次のとおりです。



リソース	説明
ポリシー ターゲット	個々のネットワーク エンドポイント（通常は NIC）です。ポリシー ターゲットは、アーキテクチャ内のアドレス指定可能な基本ユニットです。
ポリシー グループ	同じプロパティを持つポリシー ターゲットは、GBP の基本プリミティブであるポリシー グループに編成されます。ポリシー グループによって、ネットワーク セマンティクスを指定しなくても、ブロードキャストと同じ方法でインフラストラクチャに依存せずにグルー

リソース	説明
	プを構築できます。各グループは、消費するルール セットだけでなくグループに提供するルール セットも宣言することで、依存関係をモデル化します。
ポリシー分類子	プロトコル、ポート範囲、方向（イン、アウト、または双方向）などの、ネットワーク トラフィックのフィルタリング手段です。
ポリシー アクション	特定のルールが適用されるときに実行するアクションです。サポートされているタイプには、「allow」などがあります。
ポリシー ルール	アクションとペアリングされた分類子です。
ポリシー ルール セット	ポリシー ルール セットには、複数のポリシー ルールが含まれます。ルール セットは親子関係を使ってネストできます。

ネットワーク ポリシーの概要

グループベース ポリシー（GBP）の目的は、ネットワーク ポリシーの説明を一元化して、グループやルール セットなどのアプリケーション レベル ポリシーと分けることです。これにより、アプリケーション所有者の考慮事項とクラウド/インフラストラクチャ管理者の考慮事項を分離できます。

リソース	説明
レイヤ 2 ポリシー	同じスイッチング ドメイン内のグループセットを指定します。レイヤ 2 ポリシーは特定のレイヤ 3 ポリシーを参照する必要があります。
レイヤ 3 ポリシー	多数のレイヤ 2 ポリシーを含む、重複の可能性がある IP アドレス空間を指定します。
ネットワーク サービス ポリシー	VIP 割り当てなどのネットワーク サービス チェーンに必要なネットワーク固有パラメータを指定します。

共有ポリシーの概要

グループベース ポリシー（GBP）モデル内のリソースは、テナント間でリソースを可視化するように設定できる `shared` 属性をサポートします。ポリシー リソースを共有して、関連するチームがポリシーを作成して他のユーザがそのポリシーを消費できるようにすると、考慮事項の分離が促進されます。共有リソースにはグローバル可視性があるため、すべてのテナントが共有リソースを表示できます。`shared` 属性を設定または更新できるのは、管理者ロールを持つユーザのみです。

次の例では、CLI を使用して共有レイヤ 3 ポリシーを作成します。

```
# gbp l3policy-create --shared True my-shared-l3p
```

同様に、GUI にはリソースの共有または非共有を有効にするチェックボックスがあります。

リソースを共有するタイミングと方法には特定の制約があります。

- 共有リソースは、他の共有リソースにのみ関連付けることができます。たとえば、共有レイヤ 2 ポリシーが存在できるのは共有レイヤ 3 ポリシー上のみです。
- 現在使用されている共有リソースは共有解除できません。

（注） ポリシー ターゲット リソースは、`shared` 属性をサポートしません。

Neutron マッピング ドライバの概要

グループベース ポリシー（GBP）モデルとその実装の最も便利な側面の 1 つは、ポリシーを Neutron API に直接マッピングできるので既存の Neutron プラグインを使用できることです。デフォルト マッピングは次のとおりです。

GBP リソース	Neutron
ポリシー ターゲット	ポート
ポリシー ターゲット グループ	サブネット
レイヤ 2 ポリシー	ネットワーク
レイヤ 3 ポリシー	ルータ

（注） カスタム マッピングを設計して「リソース マッピング」ポリシー ドライバに実装できます。

外部接続

グループベース ポリシー（GBP）モデルでは、ユーザの意図をキャプチャしてポリシー ターゲットに外部との通信を許可する API がサポートされます。これを実現するために次のプリミティブが使用されます。

リソース	説明
外部セグメント	Classless Inter-Domain Routing（CIDR）によって定義された外部ネットワークと、指定のネクストホップを使用してこのセグメント経由で到達可能なその他のネットワークをモデル化します。

リソース	説明
外部ポリシー	外部セグメントのポリシー ターゲット グループをモデル化します。これはポリシー ルール セットを提供および消費できる外部ポリシー ターゲット グループのようなものですが、外部ポリシーではポリシー ターゲットを作成できない点が異なります。
Neutron マッピング外部セグメント	Neutron 外部ネットワークで作成された Neutron サブネットです。

次の例では、「Datacenter-Out」という名前の外部ネットワークの設定を示します。同じ名前で外部セグメントを作成すると、関連する CIDR とその他の属性を自動的に設定できます。

次のコマンドは、「Datacenter-Out」という名前の外部セグメントを作成します。

```
# export EXT_SEG_ID=$(gbp external-segment-create Datacenter-Out --shared
  True --external-route destination=0.0.0.0/0,nexthop= | awk "/ id / {print \$4}")
```

次のコマンドは、外部を表すグループをモデル化する外部ポリシーを作成します。

```
# gbp external-policy-create my-external-policy --external-segments $EXT_SEG_ID
```

次のコマンドは、TCP アクセスを許可するポリシー ルール セットを作成します。

```
# gbp policy-classifier-create all-tcp-traffic --protocol tcp --direction in
# gbp policy-rule-create tcp-policy-rule --classifier all-tcp-traffic --actions allow
# gbp policy-rule-set-create   tcp-ruleset --policy-rules tcp-policy-rule
```

次のコマンドは、Web ポリシー ターゲット グループが外部にアクセスするためのすべての TCP トラフィックを許可するように、提供/消費関係を設定します。

```
# gbp external-policy-update my-external-policy
  --provided-policy-rule-sets "tcp-ruleset=true"
```

この例では、レイヤ 3 ポリシーが「default」と呼ばれています。

次のコマンドは、Web ポリシー ターゲット グループ用に暗黙的に作成されたレイヤ 3 ポリシーを作成済みの外部セグメントに関連付けます。

```
# gbp l3policy-update default -external-segment Datacenter-Out
```

ネットワーク アドレス変換プールとフローティング IP アドレス

各外部セグメントには、フローティング IP アドレスの割り当てに使用できる IP アドレスのプールを 0 個以上設定できます。この場合のフローティング IP アドレスは、Neutron フローティング IP アドレス リソースに固有です。

リソース	説明
NAT プール	仮想マシンのフローティング IP アドレスの割り当てに使用される IP アドレスのプールです。

この例の **1.105.2.128/25** 範囲には、ルーティング可能な任意の範囲を指定できます。また、設定内の CIDR と相関する必要はありません。

次のコマンドは、外部サブネットに使用されているサブネットから NAT プールを作成します。

```
# gbp nat-pool-create --ip-version 4 --ip-pool 1.105.2.128/25
--external-segment Datacenter-Out my-nat-pool
```

次のコマンドは、上記のコマンドで作成した NAT プールを使用するネットワーク サービス ポリシー（NSP）を作成します。

```
# gbp network-service-policy-create --network-service-params
type=ip_pool,name=nat_fip,value=nat_pool my-nat-pool-nsp
```

次のコマンドは、上記のコマンドで作成した NSP を Web ポリシー ターゲット グループと関連付けます。

```
# gbp group-update web --network-service-policy my-nat-pool-nsp
```

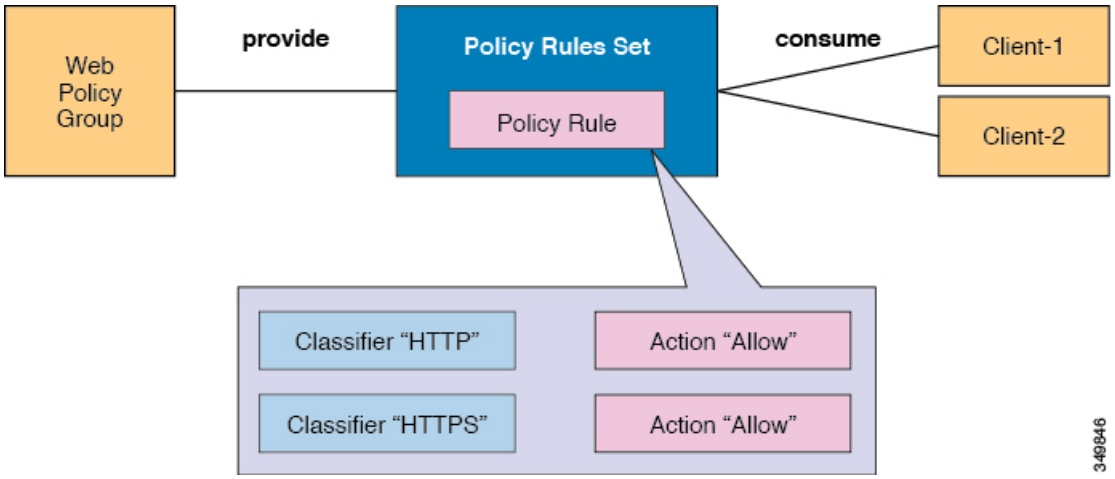
この時点で、Web ポリシー ターゲット グループに作成されたすべての新しいポリシー ターゲットに、NAT プールからフローティング IP アドレスが自動的に割り当てられます。

NAT プールは、設定ファイルで `cidr_exposed` および `host_pool_cidr` オプションによって指定されている Application Policy Infrastructure Controller（APIC）の外部ネットワーク設定と重複しない限り、どのアドレス範囲であっても構いません。

ユーザがポリシー ターゲット グループの特定のメンバーにのみ明示的にフローティング IP アドレスを割り当てる必要がある場合は、上記のワークフローを使用しないでください。この場合は NSP を作成せずに、フローティング IP アドレスを Neutron ポートに関連付ける明示的なワークフローを実行します。作成した NAT プールに対応するサブネットからフローティング IP アドレスを選択します。

グループベース ポリシー CLI を使用した多層アプリケーションの作成例

次の例では、グループベース ポリシー（GBP）を使用してシンプルなポリシーを作成します。このポリシーは、2 つのグループとそれらの間のポリシー ルール セットを作成します。



手順

ステップ 1 ルールおよびルールセット（一連の Web サーバに対してポリシーを説明する）を設定します。

ルールには、一部のトラフィックとそのトラフィックを処理するアクションが一致するように設計された分類子が含まれています。一般的なアクションには、トラフィックを許可したりネットワーク サービスにリダイレクトしたりするアクションがあります。

a) `allow` アクションを作成します。

```
# gbp policy-action-create allow --action-type allow
```

b) HTTP ルールを作成します。

```
# gbp policy-classifier-create web-traffic --protocol tcp --port-range 80 \
--direction in
# gbp policy-rule-create web-policy-rule --classifier web-traffic --actions allow
```

c) HTTPS ルールを作成します。

```
# gbp policy-classifier-create secure-web-traffic --protocol tcp --port-range 443 \
--direction in
# gbp policy-rule-create secure-web-policy-rule --classifier secure-web-traffic \
--actions allow
```

d) Web ルール セットを作成します。

```
# gbp policy-rule-set-create web-ruleset --policy-rules "web-policy-rule
secure-web-policy-rule"
```

ステップ 2 グループを作成してルール セットを関連付けます。

ルール セットは、双方向の一連のルールを説明します。ただし API は、グループが自身の動作を説明するルール セットを「提供」し、他のグループがそのルール セットを「消費」して接続できるように設計されています。このモデルの意図は、グループが自身の動作を説明するルール セットを提供して他のグループがアクセスを選択できるようにすることです。

a) グループを作成します。

```
# gbp group-create web
# gbp group-create client-1
# gbp group-create client-2
```

b) ルール セットを関連付けます。

```
# gbp group-update client-1 --consumed-policy-rule-sets "web-ruleset=scope"
# gbp group-update client-2 --consumed-policy-rule-sets "web-ruleset=scope"
# gbp group-update web --provided-policy-rule-sets "web-ruleset=scope"
```

ステップ 3 前に作成した各グループのメンバーを作成します。

各メンバーは、グループの接続とセキュリティの要件を指定するプロパティをすべて継承します。

a) Web グループのポリシー ターゲットを作成し、そのポリシー ターゲットに関連付けられている Neutron ポートを抽出します。

```
# export WEB_PORT=$(gbp policy-target-create web-pt-1 --policy-target-group web \
| awk "/port_id/ {print \$4}”)
```

b) 抽出した Neutron ポートを使用して Web グループ メンバー（仮想マシン インスタンス）を作成します。

```
# nova boot --flavor m1.tiny --image image_name --nic port-id=$WEB_PORT web-vm-1
```

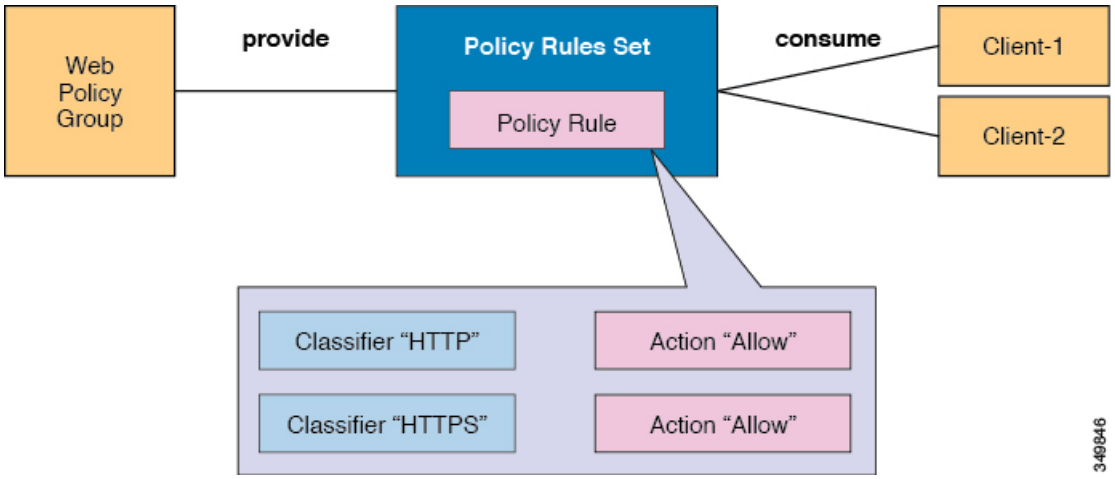
c) `client-1` および `client-2` グループのメンバーを作成します。

```
# export CLIENT1_PORT=$(gbp policy-target-create client-pt-1 \
--policy-target-groupclient-1 | awk "/port_id/ {print \$4}") nova boot \
--flavor m1.tiny --image image_name --nic port-id=$CLIENT1_PORT \
client-vm-1

# export CLIENT2_PORT=$(gbp policy-target-create client-pt-2 \
--policy-target-group client-2 | awk "/port_id/ {print \$4}") nova boot \
--flavor m1.tiny --image image_name --nic port-id=$CLIENT2_PORT \
client-vm-2
```

グループベース ポリシー GUI を使用した多層アプリケーションの作成例

次の例では、グループベース ポリシー（GBP）を使用して多層アプリケーションのシンプルなポリシーを作成します。このポリシーは、2 つのグループとそれらの間のポリシー ルール セットを作成します。



手順

ステップ 1 管理者として OpenStack プラットフォーム GUI にログインします。

ステップ 2 メニュー バーで **[Project]** > **[Policy]** > **[Application Policy]** > **[Policy Rule Set]** を選択します。

ステップ 3 [Policy Rule Set] ペインで [Create Policy Rule Set] をクリックします。

ステップ 4 [Create Policy Rule Set] ダイアログボックスで、次の操作を実行します。

a) [Name] フィールドに、新しいポリシー ルール セットの名前（web-ruleset）を入力します。
b) [Description] フィールドに、新しいポリシー ルール セット（web-ruleset）の説明を入力します。
c) [Next] をクリックします。
d) [Policy Rules] フィールドで [+] アイコンをクリックします。

ステップ 5 [Create Policy-Rule] ダイアログボックスで、次の操作を実行して HTTP のルールを作成します。

a) [Name] フィールドに、新しいポリシールールの名前（http-rule）を入力します。
b) [Description] フィールドに、新しいポリシールール（http-rule）の説明を入力します。
c) [Next] をクリックします。
d) [Policy Classifier] フィールドで [+] アイコンをクリックします。

ステップ 6 [Create Policy Classifier] ダイアログボックスで、次の操作を実行します。

a) [Name] フィールドに、新しいポリシー分類子の名前（http-classifier）を入力します。
b) [Port/Range(min:max)] フィールドにポート範囲（80）を入力します。
c) [Direction] ドロップダウンリストから方向（[BI]）を選択します。
d) [Create Policy Classifier] をクリックします。
e) [Next] をクリックします。
f) [Policy Action] フィールドで、ポリシールールのアクション（[allow]）を選択します。
g) **[Create]** をクリックします。

ステップ 7 [Create Policy Rule Set] ダイアログボックスで、HTTPS 用に次の操作を実行します。

a) [Policy Rules] フィールドで [+] アイコンをクリックします。

b) [Name] フィールドに、新しいポリシールールの名前（https-rule）を入力します。
c) [Description] フィールドに、新しいポリシールール（https-rule）の説明を入力します。
d) [Next] をクリックします。
e) [Policy Classifier] フィールドで [+] アイコンをクリックします。

ステップ 8 [Create Policy Classifier] ダイアログボックスで、次の操作を実行します。
a) [Name] フィールドに、新しいポリシー分類子の名前（https-classifier）を入力します。
b) [Port/Range(min:max)] フィールドにポート範囲（443）を入力します。
c) [Direction] ドロップダウンリストから方向（[BI]）を選択します。
d) [Create Policy Classifier] をクリックします。

ステップ 9 [Create Policy-Rule] ダイアログボックスで、次の操作を実行します。
a) [Next] をクリックします。
b) [Policy Action] フィールドで、ポリシールールのアクション（[allow]）を選択します。
c) **[Create]** をクリックします。
d) [Policy Rules] フィールドで、ポリシー ルール セットのポリシー ルール（[http-rule] と [https-rule]）を選択します。
e) **[Create]** をクリックします。

ステップ 10 メニュー バーで **[Project]** > **[Policy]** > **[Application Policy]** > **[Policy Rule Set]** を選択します。

ステップ 11 [Policy Rule Set] ペインで、ポリシー ルールを含むルール セットの存在を確認できます。

ステップ 12 [Policy Rules] をクリックします。

ステップ 13 [Policy Rules] ペインで、ポリシー ルールの存在を確認できます。

ステップ 14 [Policy Classifier] をクリックします。

ステップ 15 [Policy Classifier] ペインで、ポリシー分類子の存在を確認できます。

ステップ 16 [Policy Actions] をクリックします。

ステップ 17 [Policy Actions] ペインで、前に作成したポリシー アクションの存在を確認できます。

ステップ 18 ポリシー ターゲット グループを作成します。メニュー バーで **[Project]** > **[Policy]** > **[Groups]** を選択します。

ステップ 19 [Create Group] をクリックします。

ステップ 20 [Create Group] ダイアログボックスで、次の操作を実行します。
a) [Name] フィールドに、新しいグループの名前（web-group）を入力します。
b) [Description] フィールドに、新しいグループ（web-group）の説明を入力します。
c) [Next] をクリックします。
d) [Provided Policy Rule Set] フィールドで、グループのポリシー ルール セット（web-ruleset）を選択します。
e) [Next] をクリックします。
f) [Network Policy] ドロップダウンリストで、グループのネットワーク ポリシー（[Default]）を選択します。[Default] を選択すると、グループと同じ名前のレイヤ 2 ポリシーが自動的に作成されます。
g) **[Create]** をクリックします。

ステップ 21 クライアント用に 2 つ目のグループを作成します。[Create Group] をクリックします。

ステップ 22 [Create Group] ダイアログボックスで、次の操作を実行します。

a) [Name] フィールドに、クライアントの新しいグループの名前（client-group）を入力します。

b) [Description] フィールドに、クライアントの新しいグループ（client-group）の説明を入力します。

c) [Next] をクリックします。

d) [Consumed Policy Rule Set] フィールドで、グループで消費されるポリシー ルール セット（web-ruleset）を選択します。

e) [Next] をクリックします。

f) [Network Policy] フィールドで、グループのネットワーク ポリシー（[Default]）を選択します。[Default] を選択すると、ステップ 20 （13 ページ）で暗黙的に作成されたデフォルトのレイヤ 2 ポリシーにグループが追加されます。

g) **[Create]** をクリックします。

ステップ 23 メニュー バーで **[Project]** > **[Policy]** > **[Network and Services' Policy]** を選択します。

ステップ 24 [L3 Policy] ペインで、デフォルトのレイヤ 3 ポリシーが作成されたことを確認できます。

ステップ 25 ネットワーク ポリシーの [default] をクリックして、レイヤ 2 ポリシーが作成されていることを確認します。

ステップ 26 仮想マシンを作成します。メニュー バーで **[Project]** > **[Policy]** > **[Groups]** を選択します。

ステップ 27 [Groups] ペインでグループ（web-group）をクリックします。

ステップ 28 [Members] ペインで [Create Member] をクリックします。

ステップ 29 [Create Member] ダイアログボックスで、次の操作を実行します。

a) [Instance Name] フィールドにインスタンス名（web1）を入力します。

b) [Instance Boot Source] ドロップダウンリストで、インスタンス ブート ソース（[Boot from image]）を選択します。

c) [Image Name] ドロップダウンリストでイメージ（cirros-web）を選択します。

d) [Launch] をクリックします

e) [Members] ペインで、新しく作成されたグループ（web-group）を確認できます。

ステップ 30 クライアント仮想マシンを作成します。メニュー バーで **[Project]** > **[Policy]** > **[Groups]** を選択します。

ステップ 31 [Groups] ペインでグループ（client-group）をクリックします。

ステップ 32 [Members] ペインで [Create Member] をクリックします。

ステップ 33 [Create Member] ダイアログボックスで、[Details] を選択して次の操作を実行します。

a) [Instance Name] フィールドにインスタンス名（client1）を入力します。

b) [Instance Boot Source] ドロップダウンリストで、インスタンス ブート ソース（[Boot from image]）を選択します。

c) [Image Name] ドロップダウンリストでイメージ（cirros-web）を選択します。

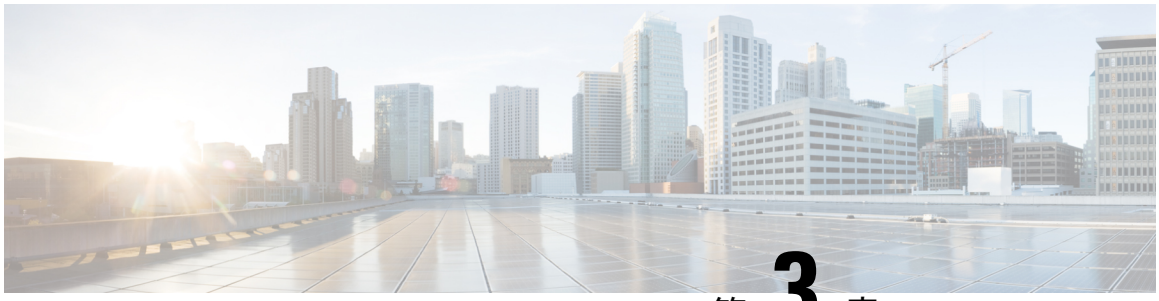
d) [Launch] をクリックします

e) [Members] ペインで、新しく作成されたグループ（client-group）を確認できます。

ステップ 34 すべてのインスタンスを表示するには、メニュー バーで **[Admin]** > **[System]** > **[Instances]** を選択します。

グループベース ポリシーの導入

グループベース ポリシー（GBP）の導入については、次を参照してください。
https://wiki.openstack.org/wiki/GroupBasedPolicy#Try_Group-based_Policy



第 3 章

Neutron SVI 統合

この章の内容は、次のとおりです。

Neutron SVI 統合の概要

スイッチ仮想インターフェイス（SVI）はスイッチ ポートの仮想ローカル エリア ネットワーク（VLAN）であり、ルーティング システムまたはブリッジング システムへの 1 つのインターフェイスに相当します。この VLAN に物理インターフェイスは存在せず、VLAN に関連付けらたすべてのスイッチ ポートからのパケットには、SVI によってレイヤ 3 処理が行われます。VLAN と SVI の間には 1 対 1 のマッピングが存在するため、1 つの VLAN にマッピングできるのは 1 つの SVI のみです。デフォルトで、SVI はデフォルト VLAN（VLAN1）用に作成され、リモート スイッチの管理を可能にします。SVI は、物理ポートに関連付けられていないとアクティブ化できません。

Neutron SVI 機能は、OpFlex エージェントが使用されていない OpenStack コンピューティング ノード上の VM でのみ有効にできます。この機能は、AIM ベースのプラグインによって統合モードでのみ使用可能です。

SVI は、一般に次の理由で VLAN 用に設定されます。

- 仮想マシン（VM）と L3-out とのピアリングを可能にする
- アップストリーム OpenStack API を使用して l3-out ノード プロファイルを制御する
- APIC で l3out にマッピングする neutron ネットワークの VLAN タイプの作成を可能にする（この neutron ネットワークで作成された DHCP および VM エンドポイントでは、状況に応じて対応する l3out SVI インターフェイスが作成されます。この neutron ネットワークの VLAN ID と、この VM がスピンアップされたホスト リンクを使用して、対応する SVI インターフェイスを作成します）。

SVI の利点は次のとおりです。

- ファブリック スイッチと仮想ネットワーク機能（VNF）間でのダイナミック ルーティング プロトコルの設定
- 複数の ACI ポッドにまたがるダイナミックおよび分散 VNF のサポート
- VNF 間の等コスト マルチパス（ECMP）トラフィック分散
- VNF での最適なパフォーマンス
- OpenStack 向け Cisco ACI プラグインにより、スイッチと OpenStack VNF 間の分散型ルート ピアリングが可能になります。VNF の作成または破棄に基づいて、Neutron SVI 機能が動的かつ自動的にアンダーレイ上で SVI を作成および破棄し、ライン レート ルーティング機能と最大 64 方向の ECMP を VNF に対して有効にします。現在、Nuetron SVI では同じ L3out の最大 6 組のスイッチ ペアがサポートされています。分散サイト（マルチポッド）全体での VNF、および高速で VM 障害を検出する Bidirectional Forwarding Detection（BFD）による VPC とファブリックの□ボンディングがサポートされます。

SVI の設定

ここでは、スイッチ仮想インターフェイス（SVI）の設定方法について説明します。

手順

「--apic:svi True」を使用して neutron ネットワークを作成します。

例：

```
#######
#creates the LB SVI network and its subnet which will be used for BGP peering between
ACI leaf and LB --no-dhcp is required initially not to
#assign a random IP to the SVI

neutron net-create LBSVI --provider:network_type vlan --provider:physical_network physnet1
 --apic:svi True --apic:bgp_enable True --apic:bgp_asn 2010
openstack subnet create --ip-version 4 --subnet-range 172.168.0.0/24  --gateway 172.168.0.1
 --network LBSVI LBSUBNET --no-dhcp

#defines the static leaf IP address for the SVI (this is optional but nice to have so
the LB knows the neighbor to peer with)
openstack port create apic-svi-port:node-101 --network LBSVI --device-owner apic:svi
--fixed-ip subnet=LBSUBNET,ip-address=172.168.0.11
openstack port create apic-svi-port:node-102 --network LBSVI --device-owner apic:svi
--fixed-ip subnet=LBSUBNET,ip-address=172.168.0.12

#now that static ports are set dhcp can be enabled
openstack subnet set LBSUBNET --dhcp

#create 2 LB VMs with static IP 172.168.0.21 and 172.168.0.22
openstack port create LB1PORT --network LBSVI --fixed-ip
subnet=LBSUBNET,ip-address=172.168.0.21
openstack port create LB2PORT --network LBSVI --fixed-ip
subnet=LBSUBNET,ip-address=172.168.0.22
```

```
LB1=$(openstack port list | awk '/LB1/ {print $2}')
LB2=$(openstack port list | awk '/LB2/ {print $2}')

nova boot --flavor m1.tiny --image LB1 --nic port-id=$LB1 vLB1
nova boot --flavor m1.tiny --image LB2 --nic port-id=$LB2 vLB2
```

SVI ネットワークのトラブルシューティング

ここでは、SVI ネットワークのトラブルシューティング方法について説明します。

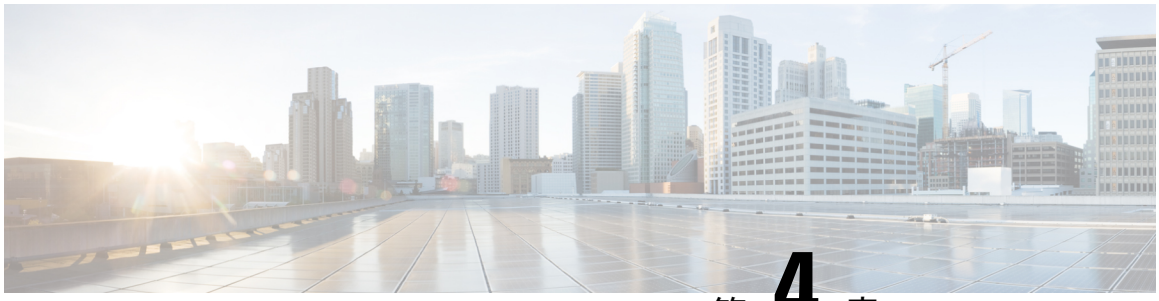
- l3 ドメイン DN が正しく設定されていることを確認します。neutron 設定ファイルで l3 ドメイン DN が APIC の有効な外部ルーテッド ドメインを指しており、すべてのコントローラ ノード上で neutron-server が再起動されています。
- 既存の l3-out、およびメカニズム ドライバによって作成された自動 l3-out のどちらにも障害が発生していないことを確認します。
- DHCP または VM エンドポイントが作成されるときに、正しいパスと SVI ネットワークの VLAN ID で SVI インターフェイスが適切に作成されていることを確認します。
  - VPC セットアップでは、メカニズム ドライバによって SVI サブネットからサイト A とサイト B のプライマリ IP が割り当てられ、SVI インターフェイス全体で VPC ペアごとの一貫性が保たれます。セカンダリ IP は、常に SVI サブネットのゲートウェイ IP です。
- l3-out のノード プロファイルでノードが適切に作成されていることを確認します。
  - SVI インターフェイスの作成時に、mechanism_driver によって対応するノードも作成されます。ノード情報は SVI パス自体に含まれています。VPC セットアップで、各 SVI インターフェイスのパスに 2 つのノードが含まれます。
- ネットワークの作成時または更新時に指定する `Bgp_asn` パラメータは、bgp ピアとして動作するゲストマシンがピアリングに使用するものと同じで、ACI による内部ファブリック BGP ピアリングに使用される AS 番号とは異なる必要があります。また、ピア間でルートを再配布するために、プロバイダーとコンシューマの BGP ピアで AS 番号が異なることを確認します。

  （注） 直接接続されているサブネットは、一般的に eBGP を使用するため再配布されません。必要に応じて、APIC にポストすることにより、明示的な permit ルートマップを使用してこれらのサブネットを openstack の外部にエクスポートできます。

- BGP セッションを確立し、他のピアへのピアリングによって学習されるプレフィックスをインポート/エクスポートするための最小限の設定が、ルート マップ/コミュニティ/パス

ワードのすべての高度な設定用に Openstack 統合で公開され、APIC API を直接使用します。

- l3-out で作成されたはずのものが APIC に表示されない場合（またはその逆の場合）は、「**aimctl manager**」コマンドを使用してデバッグします。
  - 「**aimctl manager | grep out**」および「**aimctl manager | grep external**」と入力するだけで、利用可能な l3-out および外部ネットワークに関連するすべての CLI コマンドがリストされます。
  - `sync_status` に障害が表示された場合は、`/var/log/aim-aid.log` ファイルで詳細を確認します。



第 4 章

Neutron SFC 統合

この章の内容は、次のとおりです。

Neutron SFC 統合の概要

CLI および GUI のサポートについては、OpenStack プロバイダーにお問い合わせください。

Neutron SFC 統合を使用すべき理由：

- マルチノード PBR が容易になる
- アップストリーム Openstack API を使用してマルチノード PBR でサービス グラフを導入できる
- ACI の手動設定が不要である
- Neutron API を使用してサービス チェーンを作成するフレームワークである

CLI を使用した Neutron SFC の設定

ここでは、CLI を使用して Neutron Service Function Chaining（SFC）を設定する方法について説明します。この方法はテストおよびサポートの対象外であるため、情報提供のみが目的です。CLI リファレンスについては、下記のリンクを参照してください。

https://docs.openstack.org/neutron/queens/admin/config-sfc.html

例外は次のとおりです。

1. シスコでは、REST API を使用した Neutron SFC の設定のみがサポートされます。
2. アップストリーム プロジェクトからのトラフィック分類子のうち、ドライバでサポートされるのは下記のみです。
    - `source_ip_prefix` - 送信元 IP アドレスまたはプレフィックス

- `destination_ip_prefix` - 宛先 IP アドレスまたはプレフィックス

始める前に

CLI および GUI のサポートについて OpenStack プロバイダーに問い合わせます。

手順

ステップ 1 左右のネットワーク（BD）を作成します。

例：

```
neutron net-create SRC-NET

openstack subnet create --ip-version 4 --gateway 1.1.0.1 --network SRC-NET_ID \
--subnet-range 1.1.0.0/24 --host-route destination=10.0.0.0/16,gateway=1.1.0.1 ''

neutron net-create DST-NET

openstack subnet create --ip-version 4 --gateway 2.2.0.1 --network DST-NET_ID \
--subnet-range 2.2.0.0/24 --host-route destination=0.0.0.0/0,gateway=2.2.0.1 ''
```

a) フロー分類子を作成します。

例：

```
neutron flow-classifier-create --destination-ip-prefix 0.0.0.0/0 --source-ip-prefix
 \
10.0.1.0/24 --l7-parameters logical_source_network=\
SRC-NET_ID,logical_destination_network=DST-NET_ID CLASSIFIER1
```

ステップ 2 送信元と宛先の neutron ポートを作成します。

例：

```
openstack port create SERVICE1-INGRESS --network SRC-NET_ID --no-security-group \
--disable-port-security --fixed-ip subnet=SRC-SUBNET_ID,ip-address=1.1.0.11

openstack port create SERVICE1-EGRESS --network DST-NET_ID --no-security-group \
--disable-port-security --fixed-ip subnet=DST-SUBNET_ID,ip-address=2.2.0.11
```

ステップ 3 ポート ペアを作成します。

例：

```
neutron port-pair-create --ingress SERVICE1-INGRESS-PORT_ID --egress \
SERVICE1-EGRESS-PORT_ID PORTPAIR1
```

ステップ 4 ポート ペア グループを作成します。

例：

```
neutron port-pair-group-create --port-pair PORTPAIR1_ID CLUSTER1
```

ステップ 5 サービス チェーンを作成します。

例：

```
neutron -port-chain-create --flow-classifier CLASSIFIER1_ID --port-pair-group CLUSER1_ID
 \
SERVICE-CHAIN1
```

（注） サービス チェーンが確立された後は、サービス VM（VNF）の出力および入力インターフェイスはリダイレクトされたトラフィック以外に使用できません。たとえば、これらのインターフェイスの DHCP は機能しなくなります。出力および入力インターフェイスの IP 設定を含む VNF 設定の管理には、別の管理インターフェイスを使用することをお勧めします。

ステップ 6 サービス VM を作成します。

例：

```
nova boot --flavor medium --image ServiceImage1 --nic \
port-id=SERVICE1-INGRESS-PORT_ID --nic port-id=SERVICE1-INGRESS-_PORT_ID SERVICE-VM-1
```

ステップ 7 有線でのバンピングを追加します。

a) 左右のネットワーク（BD）を追加作成します。

例：

```
neutron net-create SRC-NET2

openstack subnet create --ip-version 4 --gateway 1.1.0.1 --network SRC-NET2_ID \
--subnet-range 1.1.0.0/24 --host-route destination=10.0.0.0/16,gateway=1.1.0.1 ''

neutron net-create DST-NET2

openstack subnet create --ip-version 4 --gateway 2.2.0.1 --network DST-NET2_ID \
--subnet-range 2.2.0.0/24 --host-route destination=0.0.0.0/0,gateway=2.2.0.1 ''
```

b) サービス 2 の送信元と宛先の neutron ポートを作成します。

例：

```
openstack port create SERVICE2-INGRESS --network SRC-NET2_ID --no-security-group \
--disable-port-security --fixed-ip subnet=SRC-SUBNET2_ID,ip-address=3.3.0.11

openstack port create SERVICE2-EGRESS --network DST-NET2_ID --no-security-group \
--disable-port-security --fixed-ip subnet=DST-SUBNET2_ID,ip-address=4.4.0.11
```

ステップ 8 有線でのバンピングを追加するには、次の手順を実行します。

a) サービス 2 のポート ペアを作成します。

例：

```
neutron port-pair-create --ingress SERVICE2-INGRESS_PORT_ID --egress \
SERVICE2-EGRESS_PORT_ID PORTPAIR2
```

b) サービス 2 のポート ペア グループを作成します。

例：

```
neutron port-pair-group-create –port-pair PORTPAIR1_ID CLUSTER2
```

c) サービス チェーンを更新します（新しいポート ペア グループを追加します）。

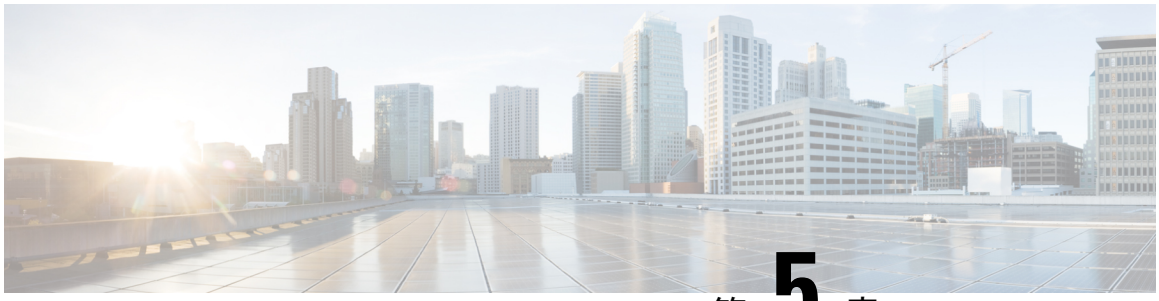
例：

```
neutron port-chain-update SERVICE-CHAIN1_ID --flow-classifier CLASSIFIER1_ID \
--port-pair-group CLUSTEER1 --port-pair-group CLUSTER2
```

d) service2 VM を作成します。

例：

```
nova boot --flavor medium --image ServiceImage1 --nic \
port-id=SERVICE2-INGRESS-PORT_ID --nic port-id=SERVICE2-EGRESS-PORT_ID SERVICE_VM-2
```



第 5 章

Day 2 オペレーション

この章の内容は、次のとおりです。

データ プレーン検証

データ プレーン検証の概要

この章では、OpenStack 向け Cisco ACI プラグインが正しくインストールされていることを検証する方法と、OpenStack インスタンスとそのデフォルト ゲートウェイおよび外部で作成されたネットワークとの基本的な接続をテストする方法について説明します。

前提条件

開始する前に、次の前提条件を満たしていることを確認してください。

- コマンドを実行して、ネットワーク構造とインスタンスを作成するプロジェクト用の keystone ファイルを読み取ることを確認します。
- external-network-shared という 1 つの外部ネットワークが OpenStack プロジェクトの共有リソースとしてすでに作成されていることを確認します。
- インスタンスを作成するために、Nova でフレーバーが事前に定義されていることを確認します。
- インスタンスをブートするイメージが Glance に定義済みであることを確認します。

Neutron ネットワークの作成

ここでは、Neutron ネットワークを作成する方法について説明します。

手順

ステップ 1 次のコマンドを入力して Neutron ネットワークを作成します。

```
$ openstack network create test_net
```

サンプル出力：

```
+---------------------------+-------------------------------------------------+
|Field                      |Value                                            |
|---------------------------+-------------------------------------------------
| admin_state_up            | UP                                              |
| availability_zone_hints   |                                                 |
| availability_zone         |                                                 |
| created_at                | 2018-07-24T20:32:27z                            |
| description               |                                                 |
| dns_domain                | None                                            |
| id                        | 016b9885-c8ac-4a2d-be7e-e5203c945ba4            |
| ipv4_address_scope        | None                                            |
| ipv6_address_scope        | None                                            |
| is_default                | None                                            |
| mtu                       | 1500                                            |
| name                      | test_net                                        |
| port_security_enabled     | True                                            |
| project_id                | 7d0be879a12c47ae9c0a26d3fd4407d1                |
| provider:network_type     | opflex                                          |
| provider:physical_type    | physnet1                                        |
| provider:segmentation_id  | None                                            |
| qos_policy_id             | None                                            |
| revision_number           | 3                                               |
| router:external           | Internal                                        |
| segments                  | None                                            |
| shared                    | False                                           |
| status                    | ACTIVE                                          |
| subnets                   |                                                 |
| updated_at                | 2018-07-24T20:32:27Z                            |
+---------------------------+-------------------------------------------------+
```

同様に ACI ファブリックでも、対応するテナントに Neutron ネットワーク名が付いた EPG と BD が定義されていることを確認できます。

ステップ 2 Neutron ネットワーク名が付いた EPG と BD が定義されていることを確認します。

a) APIC GUI のメニュー バーで、**[Tenants]** > ***[ternant_name]*** > **[Application Profiles]** > **[OpenStack]** > **[Application EPGs]** > ***[EPG_name]*** を選択します。EPG が定義されているかどうかを確認します。

b) APIC GUI のメニュー バーで、**[Tenants]** > ***[tenant_name]*** > **[Networking]** > **[Bridge Domains]** > ***[BD_name]***を選択します。BD が定義されているかどうかを確認します。

Neutron サブネットの作成

ここでは、Neutron サブネットを作成する方法について説明します。

手順

次の CLI コマンドを入力して Neutron サブネットを作成し、以前に作成したネットワークにバインドします。

例：

```
openstack subnet create --network test_net --gateway 192.168.1.254 \
--subnet-range 192.168.1.0/24 subnet01
```

サンプル出力：

```
+---------------------------+-------------------------------------------------+
|Field                      |Value                                            |
|---------------------------+-------------------------------------------------
| allocation_pools          | 192.168.1.1-192.168.1.253                       |
| cidr                      | 192.168.1.0/24                                  |
| created_at                | 2018-07-24T20:37:03Z                            |
| description               |                                                 |
| dns_nameservers           |                                                 |
| enable_dhcp               | True                                            |
| gateway_ip                | 192.168.1.254                                   |
| host_routes               |                                                 |
| id                        | d3341f6d-5fbe-476e-a0b7-d0e1b546eba4            |
| ip_version                | 4                                               |
| ipv6_address_mode         | None                                            |
| ipv6_ra_mode              | None                                            |
| name                      | subnet01                                        |
| network_id                | 016b9885-c8ac-4a2d-be7e-e5203c945ba4            |
| project_id                | 7d0be879a12c47ae9c0a26d3fd4407d1                |
| revision_number           | 2                                               |
| segment_id                | None                                            |
| service_types             |                                                 |
| subnetpool_id             | None                                            |
| updated_at                | 2018-07-24T20:37:03Z                            |
+---------------------------+-------------------------------------------------+
```

このコマンドによってACIに変更が加えられることはありません。サブネットはどのルータにもまだ接続されておらず、L3 ルーティングでは有効になっていません。したがって、ACI ブリッジ ドメインにはまだサブネットが関連付けられていません。

Neutron ルータの作成

ここでは、Neutron ルータを作成する方法について説明します。

手順

ステップ 1 次の CLI コマンドを入力して Neutron ルータを作成します。

例：

```
openstack router create router01
```

サンプル出力：

```
+---------------------------+-------------------------------------------------+
|Field                      |Value                                            |
|---------------------------+-------------------------------------------------
| admin_state_up            | UP                                              |
| availability_zone_hints   | None                                            |
| availability_zone         | None                                            |
| created_at                | 2018-07-24T20:44:11Z                            |
| description               |                                                 |
| distributed               | False                                           |
| external_gateway_info     | None                                            |
| flavor_id                 | None                                            |
| ha                        | False                                           |
| id                        | 236734ab-c39e-4ad7-a9ab-c0d1fb03691a            |
| name                      | router01                                        |
| project_id                | 7d0be879a12c47ae9c0a26d3fd4407d1                |
| revision_number           | None                                            |
| routes                    | None                                            |
| status                    | ACTIVE                                          |
| updated_at                | 2018-07-24T20:41:11Z                            |
+---------------------------+-------------------------------------------------+
```

このコマンドは、ACI 共通テナントに ACI 契約を作成します。実際には、OpenStack ルータは ACI で契約の「permit IP any any」 タイプとしてレンダリングされます。契約は常に共通テナントに配置され、コンシューマおよびプロバイダーとして、サブネットがそのルータにバインドされている EPG で作成されたすべての Neutron ネットワークに適用されます。

ステップ 2 共通 ACI テナントに契約が作成されていることを確認します。

APIC GUI のメニュー バーで、**[Tenants]** > **[common]** > **[Tenant Common]** > **[Contracts]** > **[Standards]** > ***[router_name]*** を選択します。ルータが定義されているかどうかを確認します。

ルータへのサブネットのバインディング

ここでは、サブネットをルータにバインドする方法について説明します。

手順

ステップ 1 次の CLI コマンドを入力して、作成した neutron ネットワーク上でルーティングを有効にします。

例：

```
openstack router add subnet router01 subnet01
```

この結果、APIC で DefaultRouterVRF という VRF が作成されます。この VRF に BD がバインドされ、BD サブネットとして Neutron サブネットも作成されます。

ステップ 2 DefaultRouterVRF という VRF が作成されて BD が VRF にバインドされ、BD サブネットとして Neutron サブネットも作成されていることを確認します。

a) APIC GUI のメニュー バーで、**[Tenants]** > **[tenant_name]** > **[Networking]** > **[Bridge Domains]** > ***BD* 名** > **[Subnets]** > ***[subnet]*** を選択します。サブネットが定義されているかどうかを確認します。

b) APIC GUI のメニュー バーで、**[Tenants]** > **[tenant_name]** > **[VRFs]** > **[DefaultRoutedVRF (DefaultVRF)]** を選択します。DefaultRouterVRF が定義されているかどうかを確認します。

ルータへのゲートウェイの設定

ここでは、ルータにゲートウェイを設定する方法について説明します。

手順

ステップ 1 OpenStack ドメインから外部ルータへの外部接続を実現するには、以前に作成した OpenStack ルータにゲートウェイを設定する必要があります。次のコマンドでは、external-net-shared として定義された外部ネットワークがすでに存在し、OpenStack プロジェクトで消費可能であることを前提としています。

例：

```
openstack router set --external-gateway external-net-shared router01
```

ステップ 2 L3out が作成されていることを確認します。

APIC GUI のメニュー バーで、**[Tenants]** > **[tenant_name]** > **[Networking]** > **[External Routed Networks]** > **[l3out1-DefaultVRF (l3out1-DefaultVRF)]** を選択します。l3out1-DefaultVRF が定義されているかどうかを確認します。

インスタンスを作成して作成済みのネットワークに NIC を接続する

ここでは、インスタンスを作成し、以前に作成したネットワークにその NIC を接続する方法について説明します。

手順

ステップ 1 ネットワークを作成して外部ルータにルーティングできるように設定したので、OpenStack インスタンスを作成して Neutron ネットワークに接続し、接続を確認できるようになりました。次の CLI コマンドを入力して Nova VM を作成します。

例：

```
NET1=$(openstack network list | awk '/test_net/ {print $2}')
nova boot --flavor m1.tiny --image cirros --nic net-id=$NET1 vm1
```

ステップ 2 VM `vm1` が [EPG `test_net` Operational] タブに表示されることを確認します。

APIC GUI のメニュー バーで、**[Tenants]** > ***[ternant_name]*** > **[Application Profiles]** > ***EPG*** 名 > **[Application EPGs]** > ***EPG*** を選択します。ペインにある [Operational] タブをクリックします。

VM が表示されているかどうかを確認します。APIC によって IP アドレスが正しく検知される必要があります。

ICMP を使用して VM がインフラストラクチャの他の部分に正しく接続していることを確認する

ここでは、ICMP を使用して VM がインフラストラクチャの他の部分に正しく接続されていることを確認する方法について説明します。

手順

次の CLI コマンドを入力して、VM からそのデフォルト ゲートウェイおよび外部 IP への ICMP 接続に L3out 経由で到達可能であることを確認します。

例：

```
$ ifconfig eth0
$ ping 192.168.1.254
```