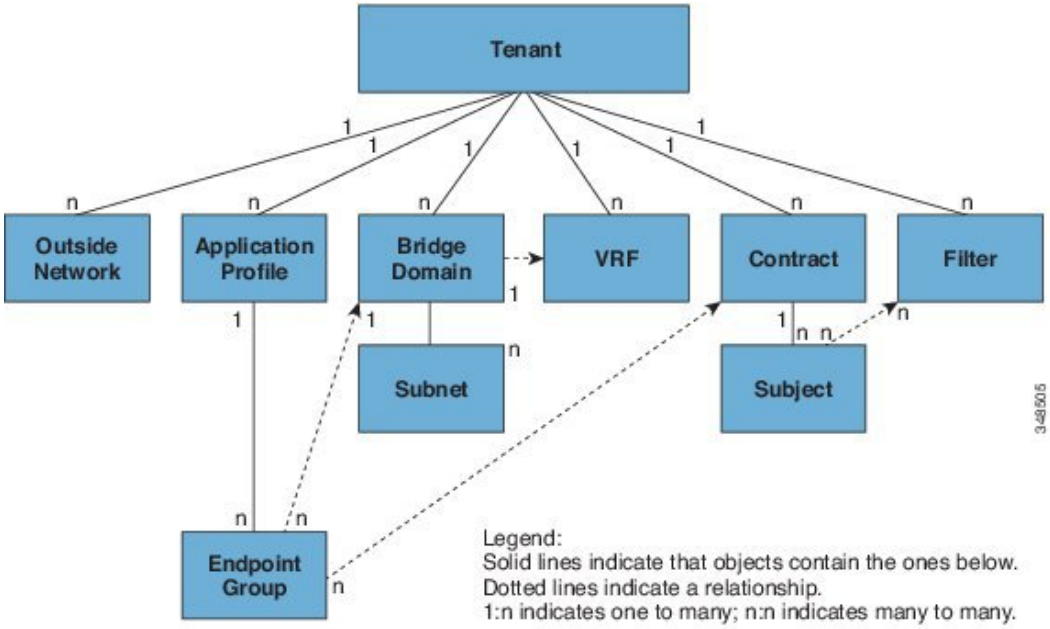
基本ユーザ テナント設定

この章の内容は、次のとおりです。

テナント

テナント(`fvTenant`)は、アプリケーション ポリシーの論理コンテナで、管理者はドメインベースのアクセス コントロールを実行できます。テナントはポリシーの観点から分離の単位を表しますが、プライベート ネットワークは表しません。テナントは、サービス プロバイダーの環境ではお客様を、企業の環境では組織またはドメインを、または単にポリシーの便利なグループ化を表すことができます。次の図は、管理情報ツリー(MIT)のテナント部分の概要を示します。

図 *1:* テナント



テナントは相互に分離することも、リソースを共有することもできます。テナントに含まれる主要な要素は、フィルタ、コントラクト、外部ネットワーク、ブリッジ ドメイン、仮想ルーティングおよび転送 (VRF) インスタンス、エンドポイント グループ (EPG) を含むアプリケーション プロファイルです。テナントのエンティティはそのポリシーを継承します。VRF はコンテキストとも呼ばれ、それぞれを複数のブリッジ ドメインに関連付けることができます。

（注） APIC GUI のテナント ナビゲーション パスでは、VRF (コンテキスト) はプライベート ネットワークと呼ばれます。

テナントはアプリケーションポリシーの論理コンテナです。ファブリックには複数のテナントを含めることができます。レイヤ 4 ～ 7 のサービスを展開する前に、テナントを設定する必要があります。ACI ファブリックは、テナント ネットワークに対して IPv4、IPv6、およびデュアル スタック構成をサポートします。

テナント内のルーティング

アプリケーションセントリック インフラストラクチャ（ACI）のファブリックでは、テナントのデフォルト ゲートウェイ機能が提供され、ファブリックの Virtual Extensible Local Area（VXLAN）ネットワーク間のルーティングが行えます。各テナントについて、APIC でサブネットが作成されるたびに、ファブリックは仮想デフォルト ゲートウェイまたはスイッチ仮想インターフェイス（SVI）を提供します。これは、そのテナント サブネットの接続エンドポイントがあるすべてのスイッチにわたります。各入力インターフェイスはデフォルトのゲート

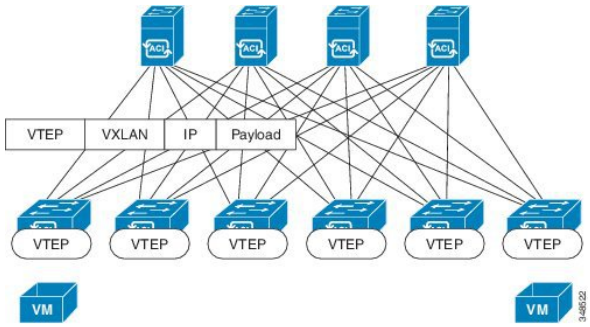
ウェイ インターフェイスをサポートし、ファブリック全体のすべての入力インターフェイスは任意のテナント サブネットに対する同一のルータの IP アドレスと MAC アドレスを共有します。

サブネット間のテナント トラフィックの転送を促進するレイヤ3VNID

ACI ファブリックは、ACI ファブリック VXLAN ネットワーク間のルーティングを実行するテナントのデフォルト ゲートウェイ機能を備えています。各テナントに対して、ファブリックはテナントに割り当てられたすべてのリーフ スイッチにまたがる仮想デフォルト ゲートウェイを提供します。これは、エンドポイントに接続された最初のリーフ スイッチの入力インターフェイスで提供されます。各入力インターフェイスはデフォルト ゲートウェイ インターフェイスをサポートします。ファブリック全体のすべての入力インターフェイスは、特定のテナント サブネットに対して同一のルータの IP アドレスと MAC アドレスを共有します。

ACI ファブリックは、エンドポイントのロケータまたは VXLAN トンネル エンドポイント (VTEP) アドレスで定義された場所から、テナント エンドポイント アドレスとその識別子を切り離します。ファブリック内の転送は VTEP 間で行われます。次の図は、ACI で切り離された ID と場所を示します。

図 *2 : ACI* によって切り離された *ID* と場所



VXLAN は VTEP デバイスを使用してテナントのエンド デバイスを VXLAN セグメントにマッピングし、VXLAN のカプセル化およびカプセル化解除を実行します。各 VTEP 機能には、次の 2 つのインターフェイスがあります。

- ブリッジングを介したローカル エンドポイント通信をサポートするローカル LAN セグメントのスイッチ インターフェイス
- 転送 IP ネットワークへの IP インターフェイス

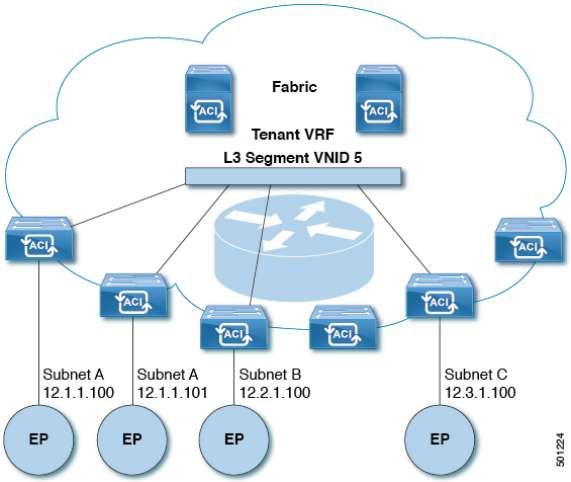
IP インターフェイスには一意の IP アドレスがあります。これは、インフラストラクチャ VLAN として知られる、転送 IP ネットワーク上の VTEP を識別します。VTEP デバイスはこの IP アドレスを使用してイーサネット フレームをカプセル化し、カプセル化されたパケットを、IP インターフェイスを介して転送ネットワークへ送信します。また、VTEP デバイスはリモート VTEP で VXLAN セグメントを検出し、IP インターフェイスを介してリモートの MAC Address-to-VTEP マッピングについて学習します。

ACI の VTEP は分散マッピング データベースを使用して、内部テナントの MAC アドレスまたはIP アドレスを特定の場所にマッピングします。VTEP はルックアップの完了後に、宛先リーフ スイッチ上の VTEP を宛先アドレスとして、VXLAN 内でカプセル化された元のデータ パケットを送信します。宛先リーフ スイッチはパケットをカプセル化解除して受信ホストに送信します。このモデルにより、ACI はスパニングツリー プロトコルを使用することなく、フル メッシュでシングル ホップのループフリー トポロジを使用してループを回避します。

VXLAN セグメントは基盤となるネットワーク トポロジに依存しません。逆に、VTEP 間の基盤となる IP ネットワークは、VXLAN オーバーレイに依存しません。これは発信元 IP アドレスとして開始 VTEP を持ち、宛先 IP アドレスとして終端 VTEP を持っており、外部 IP アドレス ヘッダーに基づいてパケットをカプセル化します。

次の図は、テナント内のルーティングがどのように行われるかを示します。

図 *3 : ACI* のサブネット間のテナント トラフィックを転送するレイヤ *3 VNID*



ACI はファブリックの各テナント VRF に単一の L3 VNID を割り当てます。ACI は、L3 VNID に従ってファブリック全体にトラフィックを転送します。出力リーフ スイッチでは、ACI によって L3 VNID からのパケットが出力サブネットの VNID にルーティングされます。

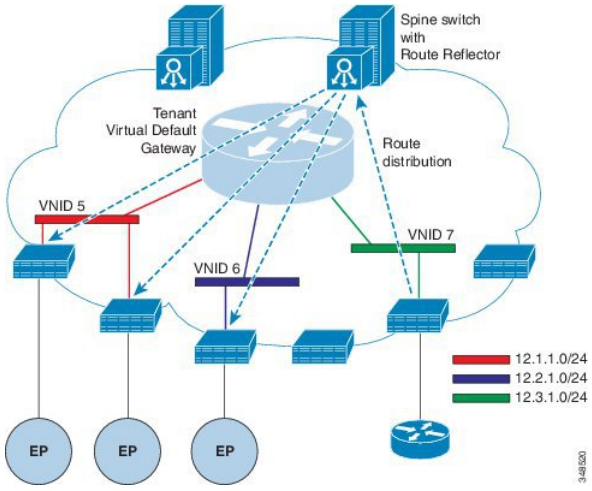
ACI のファブリック デフォルト ゲートウェイに送信されてファブリック入力に到達したトラフィックは、レイヤ 3 VNID にルーティングされます。これにより、テナント内でルーティングされるトラフィックはファブリックで非常に効率的に転送されます。このモデルを使用すると、たとえば同じ物理ホスト上の同じテナントに属し、サブネットが異なる 2 つの VM 間では、トラフィックが (最小パス コストを使用して) 正しい宛先にルーティングされる際に経由する必要があるは入力スイッチ インターフェイスのみです。

ACI ルート リフレクタは、ファブリック内での外部ルートの配布にマルチプロトコル BGP（MP-BGP）を使用します。ファブリック管理者は自律システム (AS) 番号を提供し、ルート リフレクタにするスパイン スイッチを指定します。

ルータ ピアリングおよびルート配布

次の図に示すように、ルーティング ピア モデルを使用すると、リーフ スイッチ インターフェイスが外部ルータのルーティング プロトコルとピアリングするように静的に設定されます。

図 *4*: ルータのピアリング

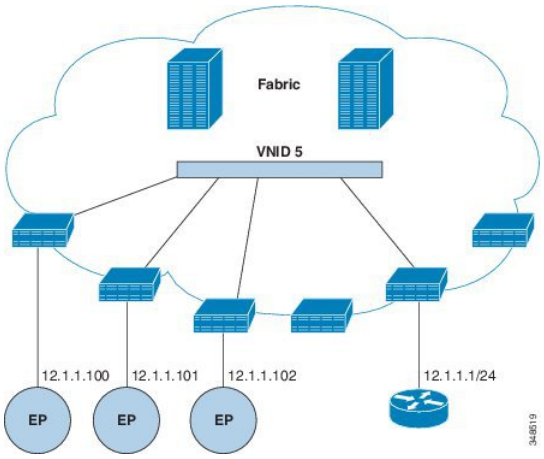


ピアリングによって学習されるルートは、スパイン スイッチに送信されます。スパイン スイッチはルート リフレクタとして動作し、外部ルートを同じテナントに属するインターフェイスを持つすべてのリーフ スイッチに配布します。これらのルートは、最長プレフィクス照合(LPM)により集約されたアドレスで、外部ルータが接続されているリモートのリーフ スイッチのVTEP IP アドレスが含まれるリーフ スイッチの転送テーブルに配置されます。WAN ルートには転送プロキシはありません。WAN ルートがリーフ スイッチの転送テーブルに適合しない場合、トラフィックはドロップされます。外部ルータがデフォルト ゲートウェイではないため、テナントのエンドポイント (EP) からのパケットは ACI ファブリックのデフォルト ゲートウェイに送信されます。

外部ルータへのブリッジド インターフェイス

次の図に示すように、リーフ スイッチのインターフェイスがブリッジド インターフェイスとして設定されている場合、テナント VNID のデフォルト ゲートウェイが外部ルータとなります。

図 *5:* ブリッジド外部ルータ



ACI ファブリックは、外部ルータの存在を認識せず、APIC はリーフ スイッチのインターフェイスを EPG に静的に割り当てます。

ルート リフレクタの設定

ACIファブリックのルートリフレクタは、マルチプロトコルBGP（MP-BGP）を使用してファブリック内に外部ルートを配布します。ACIファブリックでルートリフレクタをイネーブルにするには、ファブリックの管理者がルート リフレクタになるスパイン スイッチを選択して、自律システム（AS）番号を提供する必要があります。ルート リフレクタが ACI ファブリックでイネーブルになると、管理者は次の項で説明するように、外部ネットワークへの接続を設定できます。

ACIファブリックに外部ルータを接続するには、ファブリック インフラストラクチャの管理者がボーダー ゲートウェイ プロトコル（BGP）のルート リフレクタとしてスパイン ノードを設定します。冗長性のために、複数のスパインがルータ リフレクタ ノードとして設定されます（1 台のプライマリ リフレクタと 1 台のセカンダリ リフレクタ）。

テナントが ACI ファブリックに WAN ルータを接続する必要がある場合は、インフラストラクチャの管理者が WAN ルータが WAN のトップ オブ ラック（ToR）として接続されるリーフ ノードを（以下の通りに）設定し、この WAN ToR を BGP ピアとしてルート リフレクタ ノードの 1 つと組み合わせます。ルート リフレクタが WAN ToR に設定されていると、ファブリックにテナント ルートをアドバタイズできます。

各リーフ ノードには最大 4000 のルートを保存できます。WAN ルータが 4000 を超えるルートをアドバタイズしなければならない場合、複数のリーフ ノードとピアリングする必要があります。インフラストラクチャの管理者は、ペアになったリーフ ノードそれぞれをアドバタイズできるルート（またはルート プレフィクス）で設定します。

インフラストラクチャの管理者は、次のようにファブリックに接続されている外部 WAN ルータを設定する必要があります。

1. ルート リフレクタとして最大 2 つのスパイン ノードを設定します。冗長性のために、プライマリおよびセカンダリ ルート リフレクタを設定します。

2. WAN ToR で、プライマリおよびセカンダリ ルート リフレクタのノードを設定します。
3. WAN ToR で、ToR がアドバタイズを担当するルートを設定します。これは任意で、テナント ルータが 4000 を超えるルートをアドバタイズすることがわかっている場合にのみ行う必要があります。

テナントの外部接続の設定

アプリケーション セントリック インフラストラクチャ（ACI）ファブリック上の他のリーフ スイッチにスタティック ルートを配布する前に、Multiprotocol BGP （MP-BGP）プロセスを最初に実行し、スパイン スイッチは BGP ルート リフレクタとして設定する必要があります。

ACI ファブリックを外部ルーテッド ネットワークに統合するために、管理テナントのレイヤ 3 接続に対し Open Shortest Path First（OSPF）を設定できます。

GUI を使用した MP-BGP ルート リフレクタの設定

ステップ 1 メニュー バーで、**[System]** > **[System Settings]** の順に選択します。

ステップ 2 **Navigation** ウィンドウで、**BGP Route Reflector** を右クリックして、**Create Route Reflector Node Policy EP** をクリックします。

ステップ 3 [Create Route Reflector Node Policy EP] ダイアログボックスで、[Spine Node] ドロップダウン リストから、適切なスパイン ノードを選択します。**Submit** をクリックします。

（注） 必要に応じてスパイン ノードを追加するには、上記の手順を繰り返してください。

スパイン スイッチがルート リフレクタ ノードとしてマークされます。

ステップ 4 **BGP Route Reflector** プロパティ エリアの **Autonomous System Number** フィールドで、適切な番号を選択します。**Submit** をクリックします。

（注） 自律システム番号は、Border Gateway Protocol（BGP）がルータに設定されている場合は、リーフが接続されたルータ設定に一致する必要があります。スタティックまたは Open Shortest Path First（OSPF）を使用して学習されたルートを使用している場合は、自律システム番号値を任意の有効な値にできます。

ステップ 5 メニュー バーで、**Fabric** > **Fabric Policies** > **POD Policies** をクリックします。

ステップ 6 [Navigation] ペインで、[Policy Groups] を展開して右クリックし、[Create POD Policy Group] をクリックします。

ステップ 7 [Create POD Policy Group] ダイアログボックスで、[Name] フィールドに、ポッド ポリシー グループの名前を入力します。

ステップ 8 [BGP Route Reflector Policy] ドロップダウン リストで、適切なポリシー（デフォルト）を選択します。[Submit] をクリックします。`
BGP ルート リフレクタのポリシーは、ルート リフレクタのポッド ポリシー グループに関連付けられ、BGP プロセスはリーフ スイッチでイネーブルになります。

ステップ 9 [Navigation] ペインで、**[Pod Policies]** > **[Profiles]** > **[default]** の順に選択します。[Work] ペインで、[Fabric Policy Group] ドロップダウン リストから、前に作成されたポッド ポリシーを選択します。[Submit] をクリックします。`
ポッド ポリシー グループが、ファブリック ポリシー グループに適用されました。

ACI ファブリックの MP-BGP ルート リフレクタの設定

ACI ファブリック内のルートを配布するために、MP-BGP プロセスを最初に実行し、スパイン スイッチを BGP ルート リフレクタとして設定する必要があります。

次に、MP-BGP ルート リフレクタの設定例を示します。

（注） この例では、BGP ファブリック ASN は 100 です。スパイン スイッチ 104 と 105 が MP-BGP ルート リフレクタとして選択されます。

```
apic1(config)# bgp-fabric
apic1(config-bgp-fabric)# asn 100
apic1(config-bgp-fabric)# route-reflector spine 104,105
```

REST API を使用した MP-BGP ルート リフレクタの設定

ステップ 1 スパイン スイッチをルート リフレクタとしてマークします。

例：

```
POST https://apic-ip-address/api/policymgr/mo/uni/fabric.xml

<bgpInstPol name="default">
  <bgpAsP asn="1" />
  <bgpRRP>
     <bgpRRNodePEp id="<spine_id1>"/>
     <bgpRRNodePEp id="<spine_id2>"/>
  </bgpRRP>
</bgpInstPol>
```

ステップ 2 次のポストを使用してポッド セレクタをセットアップします。

例：

FuncP セットアップの場合：

```
POST https://apic-ip-address/api/policymgr/mo/uni.xml

<fabricFuncP>
  <fabricPodPGrp name="bgpRRPodGrp">
    <fabricRsPodPGrpBGPRRP tnBgpInstPolName="default" />
  </fabricPodPGrp>
</fabricFuncP>
```

例：

PodP セットアップの場合：

```
POST https://apic-ip-address/api/policymgr/mo/uni.xml

<fabricPodP name="default">
  <fabricPodS name="default" type="ALL">
    <fabricRsPodPGrp tDn="uni/fabric/funcprof/podpgrp-bgpRRPodGrp"/>
  </fabricPodS>
</fabricPodP>
```

MP-BGP ルート リフレクタ設定の確認

ステップ 1 次の操作を実行して、設定を確認します。

a) セキュア シェル（SSH）を使用して、必要に応じて各リーフ スイッチへの管理者としてログインします。

b) show processes | grep bgp コマンドを入力して、状態が S であることを確認します。

状態が NR（実行していない）である場合は、設定が正常に行われませんでした。

ステップ 2 次の操作を実行して、自律システム番号がスパイン スイッチで設定されていることを確認します。

a) SSH を使用して、必要に応じて各スパイン スイッチへの管理者としてログインします。

b) シェル ウィンドウから次のコマンドを実行します。

例：

cd /mit/sys/bgp/inst

例：

grep asn summary

設定した自律システム番号が表示される必要があります。自律システム番号の値が 0 と表示される場合は、設定が正常に行われませんでした。

GUI を使用した管理テナントの OSPF 外部ルーテッド ネットワークの作成

- ルータ ID と論理インターフェイス プロファイルの IP アドレスが異なっていて重複していないことを確認します。
- 次の手順は、管理テナントの OSPF 外部ルーテッド ネットワークを作成するためのものです。テナントの OSPF 外部ルーテッド ネットワークを作成するには、テナントを選択し、テナント用の VRF を作成する必要があります。
- 詳細については、『*Cisco APIC and Transit Routing*』を参照してください。

ステップ 1 メニュー バーで、**[TENANTS]** > **[mgmt]** を選択します。

ステップ 2 [Navigation] ペインで、**[Networking]** > **[External Routed Networks]** を展開します。

ステップ 3 [External Routed Networks] を右クリックし、[Create Routed Outside] をクリックします。

ステップ 4 [Create Routed Outside] 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、名前（RtdOut）を入力します。
b) [OSPF] チェックボックスをオンにします。
c) [OSPF Area ID] フィールドに、エリア ID を入力します。
d) [OSPF Area Control] フィールドで、適切なチェックボックスをオンにします。
e) [OSPF Area Type] フィールドで、適切なエリア タイプを選択します。
f) [OSPF Area Cost] フィールドで、適切な値を選択します。
g) [VRF] フィールドのドロップダウン リストから、VRF（inb）を選択します。

（注） このステップでは、ルーテッド Outside をインバンド VRF に関連付けます。

h) [External Routed Domain] ドロップダウン リストから、適切なドメインを選択します。
i) [Nodes and Interfaces Protocol Profiles] 領域の [+] アイコンをクリックします。

ステップ 5 [Create Node Profile] 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、ノード プロファイルの名前を入力します（borderLeaf）。
b) [Nodes] フィールドで、[+] アイコンをクリックして [Select Node] ダイアログボックスを表示します。
c) [Node ID] フィールドで、ドロップダウン リストから、最初のノードを選択します（leaf1）。
d) [Router ID] フィールドに、一意のルータ ID を入力します。
e) [Use Router ID as Loopback Address] フィールドをオフにします。

（注） デフォルトでは、ルータ ID がループバック アドレスとして使用されます。これらが異なるようにする場合は、[Use Router ID as Loopback Address] チェックボックスをオフにします。

f) [Loopback Addresses] を展開し、[IP] フィールドに IP アドレスを入力します。[Update] をクリックし、[OK] をクリックします。

希望する IPv4 または IPv6 の IP アドレスを入力します。

g) [Nodes] フィールドで、[+] アイコンを展開して [Select Node] ダイアログボックスを表示します。

（注） 2 つ目のノード ID を追加します。

h) [Node ID] フィールドで、ドロップダウン リストから、次のノードを選択します（leaf2）。
i) [Router ID] フィールドに、一意のルータ ID を入力します。
j) [Use Router ID as Loopback Address] フィールドをオフにします。

（注） デフォルトでは、ルータ ID がループバック アドレスとして使用されます。これらが異なるようにする場合は、[Use Router ID as Loopback Address] チェックボックスをオフにします。

k) [Loopback Addresses] を展開し、[IP] フィールドに IP アドレスを入力します。[Update] をクリックし、[OK] をクリックします。[OK] をクリックします。

希望する IPv4 または IPv6 の IP アドレスを入力します。

ステップ 6 [Create Node Profile] ダイアログボックスで、[OSPF Interface Profiles] 領域の [+] アイコンをクリックします。

ステップ 7 [Create Interface Profile] ダイアログボックスで、次のタスクを実行します。

a) [Name] フィールドに、プロファイルの名前（portProf）を入力します。
b) [Interfaces] 領域で、[Routed Interfaces] タブをクリックし、[+] アイコンをクリックします。
c) [Select Routed Interfaces] ダイアログボックスの [Path] フィールドで、ドロップダウン リストから、最初のポート（leaf1、ポート 1/40）を選択します。
d) [IP Address] フィールドに、IP アドレスとマスクを入力します。[OK] をクリックします。
e) [Interfaces] 領域で、[Routed Interfaces] タブをクリックし、[+] アイコンをクリックします。
f) [Select Routed Interfaces] ダイアログボックスの [Path] フィールドで、ドロップダウン リストから、2 つ目のポート（leaf2、ポート 1/40）を選択します。
g) [IP Address] フィールドに、IP アドレスとマスクを入力します。[OK] をクリックします。

（注） この IP アドレスは、前に leaf1 に入力した IP アドレスと異なっている必要があります。

h) [Create Interface Profile] ダイアログボックスで、[OK] をクリックします。

インターフェイスが OSPF インターフェイスとともに設定されます。

ステップ 8 [Create Node Profile] ダイアログボックスで、[OK] をクリックします。

ステップ 9 [Create Routed Outside] ダイアログボックスで、[Next] をクリックします。
[Step 2 External EPG Networks] 領域が表示されます。

ステップ 10 [External EPG Networks] 領域で、[+] アイコンをクリックします。

ステップ 11 [Create External Network] 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、外部ネットワークの名前（extMgmt）を入力します。
b) [Subnet] を展開し、[Create Subnet] ダイアログボックスの [IP address] フィールドに、サブネットの IP アドレスとマスクを入力します。
c) [Scope] フィールドで、目的のチェックボックスをオンにします。[OK] をクリックします。
d) **[Create External Network]** ダイアログボックスで、**[OK]** をクリックします。
e) **[Create Routed Outside]** ダイアログボックスで、**[Finish]** をクリックします。

（注） [Work] ペインで、[External Routed Networks] 領域に、外部ルーテッド ネットワークのアイコン（RtdOut）が表示されるようになりました。

NX-OS CLI を使用したテナントの OSPF 外部ルーテッド ネットワークの作成

外部ルーテッド ネットワーク接続の設定には、次のステップがあります。

1. テナントの下に VRF を作成します。
2. 外部ルーテッド ネットワークに接続された境界リーフ スイッチの VRF の L3 ネットワーキング構成を設定します。この設定には、インターフェイス、ルーティング プロトコル (BGP、OSPF、EIGRP) 、プロトコル パラメータ、ルートマップが含まれています。

3. テナントの下に外部 L3 EPG を作成してポリシーを設定し、これらの EPG を境界リーフ スイッチに導入します。ACI ファブリック内で同じポリシーを共有する VRF の外部ルーテッド サブネットが、1 つの「外部 L3 EPG」または 1 つの「プレフィクス EPG」を形成します。

設定は、2 つのモードで実現されます。

- テナント モード：VRF の作成および外部 L3 EPG 設定
- リーフ モード：L3 ネットワーキング構成と外部 L3 EPG の導入

次の手順は、テナントの OSPF 外部ルーテッド ネットワークを作成するためのものです。テナントの OSPF 外部ルーテッド ネットワークを作成するには、テナントを選択してからテナント用の VRF を作成する必要があります。

（注） この項の例では、テナント「exampleCorp」の「OnlineStore」アプリケーションの「web」epg に外部ルーテッド接続を提供する方法について説明します。

ステップ 1 VLAN ドメインを設定します。

例：

```
apic1(config)# vlan-domain dom_exampleCorp
apic1(config-vlan)# vlan 5-1000
apic1(config-vlan)# exit
```

ステップ 2 テナント VRF を設定し、VRF のポリシーの適用を有効にします。

例：

```
apic1(config)# tenant exampleCorp
apic1(config-tenant)# vrf context
 exampleCorp_v1
apic1(config-tenant-vrf)# contract enforce
apic1(config-tenant-vrf)# exit
```

ステップ 3 テナント BD を設定し、ゲートウェイ IP を「public」としてマークします。エントリ「scope public」は、このゲートウェイ アドレスを外部 L3 ネットワークのルーティング プロトコルによるアドバタイズに使用できるようにします。

例：

```
apic1(config-tenant)# bridge-domain exampleCorp_b1
apic1(config-tenant-bd)# vrf member exampleCorp_v1
apic1(config-tenant-bd)# exit
apic1(config-tenant)# interface bridge-domain exampleCorp_b1
apic1(config-tenant-interface)# ip address 172.1.1.1/24 scope public
apic1(config-tenant-interface)# exit
```

ステップ 4 リーフの VRF を設定します。

例：

```
apic1(config)# leaf 101
apic1(config-leaf)# vrf context tenant exampleCorp vrf exampleCorp_v1
```

ステップ 5 OSPF エリアを設定し、ルート マップを追加します。

例：

```
apic1(config-leaf)# router ospf default
apic1(config-leaf-ospf)# vrf member tenant exampleCorp vrf exampleCorp_v1
apic1(config-leaf-ospf-vrf)# area 0.0.0.1 route-map map100 out
apic1(config-leaf-ospf-vrf)# exit
apic1(config-leaf-ospf)# exit
```

ステップ 6 VRF をインターフェイス (この例ではサブインターフェイス) に割り当て、OSPF エリアを有効にします。

例：

（注） サブインターフェイスの構成では、メインインターフェイス (この例では、ethernet 1/11) は、「no switchport」によって L3 ポートに変換し、サブインターフェイスが使用するカプセル化 VLAN を含む vlan ドメイン (この例では dom_exampleCorp) を割り当てる必要があります。サブインターフェイス ethernet1/11.500 で、500 はカプセル化 VLAN です。

```
apic1(config-leaf)# interface ethernet 1/11
apic1(config-leaf-if)# no switchport
apic1(config-leaf-if)# vlan-domain member dom_exampleCorp
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 1/11.500
apic1(config-leaf-if)# vrf member tenant exampleCorp vrf exampleCorp_v1
apic1(config-leaf-if)# ip address 157.10.1.1/24
apic1(config-leaf-if)# ip router ospf default area 0.0.0.1
```

ステップ 7 外部 L3 EPG ポリシーを設定します。これは、外部サブネットを特定し、epg 「web」と接続する契約を消費するために一致させるサブネットが含まれます。

例：

```
apic1(config)# tenant t100
apic1(config-tenant)# external-l3 epg  l3epg100
apic1(config-tenant-l3ext-epg)# vrf member v100
apic1(config-tenant-l3ext-epg)# match ip 145.10.1.0/24
apic1(config-tenant-l3ext-epg)# contract consumer web
apic1(config-tenant-l3ext-epg)# exit
apic1(config-tenant)#exit
```

ステップ 8 リーフ スイッチの外部 L3 EPG を導入します。

例：

```
apic1(config)# leaf 101
apic1(config-leaf)# vrf context tenant t100 vrf v100
apic1(config-leaf-vrf)# external-l3 epg l3epg100
```

テナント、VRF、およびブリッジ ドメインの作成

テナントの概要

- テナントには、承認されたユーザのドメインベースのアクセス コントロールをイネーブルにするポリシーが含まれます。承認されたユーザは、テナント管理やネットワーキング管理などの権限にアクセスできます。
- ユーザは、ドメイン内のポリシーにアクセスしたりポリシーを設定するには読み取り/書き込み権限が必要です。テナント ユーザは、1 つ以上のドメインに特定の権限を持つことができます。
- マルチテナント環境では、リソースがそれぞれ分離されるように、テナントによりグループ ユーザのアクセス権限が提供されます(エンドポイント グループやネットワーキングなどのため)。これらの権限では、異なるユーザが異なるテナントを管理することもできます。

テナントの作成

テナントには、最初にテナントを作成した後に作成できるフィルタ、契約、ブリッジ ドメイン、およびアプリケーション プロファイルなどのプライマリ要素が含まれます。

VRF およびブリッジ ドメイン

テナントの VRF およびブリッジ ドメインを作成および指定できます。定義されたブリッジ ドメイン要素のサブネットは、対応するレイヤ 3 コンテキストを参照します。

IPv6 ネイバー探索を有効にする方法については、『*Cisco APIC Layer 3 Networking Guide*』の「*IPv6 and Neighbor Discovery*」を参照してください。

GUI を使用したテナント、VRF およびブリッジ ドメインの作成

外部ルーテッドを設定するときにパブリック サブネットがある場合は、ブリッジ ドメインを外部設定と関連付ける必要があります。

手順の概要

1. メニュー バーで **Tenants** > **Add Tenant** の順に選択します。
2. [Create Tenant] ダイアログボックスで、次のタスクを実行します。
3. [Navigation] ペインで、**[Tenant-name]** > **[Networking]** の順に展開し、[Work] ペインで、[VRF] アイコンをキャンバスにドラッグして [Create VRF] ダイアログボックスを開き、次のタスクを実行します。

4. [Networking] ペインで、[BD] アイコンを [VRF] アイコンにつなげながらキャンバスにドラッグします。[Create Bridge Domain] ダイアログボックスが表示されたら、次のタスクを実行します。
5. [Networks] ペインで、[L3] アイコンを [VRF] アイコンにつなげながらキャンバスにドラッグします。[Create Routed Outside] ダイアログボックスが表示されたら、次のタスクを実行します。

手順の詳細

ステップ 1 メニュー バーで **Tenants** > **Add Tenant** の順に選択します。

ステップ 2 [Create Tenant] ダイアログボックスで、次のタスクを実行します。

a) [Name] フィールドに、名前を入力します。
b) [Security Domains +] アイコンをクリックして [Create Security Domain] ダイアログボックスを開きます。
c) [Name] フィールドに、セキュリティ ドメインの名前を入力します。**Submit** をクリックします。
d) [Create Tenant] ダイアログボックスで、作成したセキュリティ ドメインのチェックボックスをオンにし、[Submit] をクリックします。

ステップ 3 [Navigation] ペインで、**[Tenant-name]** > **[Networking]** の順に展開し、[Work] ペインで、[VRF] アイコンをキャンバスにドラッグして [Create VRF] ダイアログボックスを開き、次のタスクを実行します。

a) [Name] フィールドに、名前を入力します。
b) [Submit] をクリックして VRF の設定を完了します。

ステップ 4 [Networking] ペインで、[BD] アイコンを [VRF] アイコンにつなげながらキャンバスにドラッグします。[Create Bridge Domain] ダイアログボックスが表示されたら、次のタスクを実行します。

a) [Name] フィールドに、名前を入力します。
b) [L3 Configurations] タブをクリックします。
c) [Subnets] を展開して [Create Subnet] ダイアログボックスを開き、[Gateway IP] フィールドにサブネットマスクを入力し、[OK] をクリックします。
d) [Submit] をクリックしてブリッジ ドメインの設定を完了します。

ステップ 5 [Networks] ペインで、[L3] アイコンを [VRF] アイコンにつなげながらキャンバスにドラッグします。[Create Routed Outside] ダイアログボックスが表示されたら、次のタスクを実行します。

a) [Name] フィールドに、名前を入力します。
b) [Nodes And Interfaces Protocol Profiles] を展開して [Create Node Profile] ダイアログボックスを開きます。
c) [Name] フィールドに、名前を入力します。
d) [Nodes] を展開して [Select Node] ダイアログボックスを開きます。
e) [Node ID] フィールドで、ドロップダウン リストからノードを選択します。
f) [Router ID] フィールドに、ルータ ID を入力します。
g) [Static Routes] を展開して [Create Static Route] ダイアログボックスを開きます。
h) [Prefix] フィールドに、IPv4 アドレスまたは IPv6 アドレスを入力します。
i) [Next Hop Addresses] を展開し、[Next Hop IP] フィールドに IPv4 アドレスまたは IPv6 アドレスを入力します。

j) [Preference] フィールドに数値を入力し、[UPDATE] をクリックしてから [OK] をクリックします。
k) [Select Node] ダイアログボックスで、[OK] をクリックします。
l) [Create Node Profile] ダイアログボックスで、[OK] をクリックします。
m) 必要に応じてチェックボックス [BGP]、[OSPF]、または [EIGRP] をオンにし、[NEXT] をクリックします。**OK**をクリックしてレイヤ 3 の設定を完了します。

L3 設定を確認するには、[Navigation] ペインで、**[Networking]** > **[VRFs]** の順に展開します。

EPG の導入

特定のポートへの EPG の静的な導入

このトピックでは、Cisco APIC を使用しているときに特定のポートに EPG を静的に導入する一般的な方法の例を示します。

GUI を使用して特定のノードまたはポートへ EPG を導入する

始める前に

EPG を導入するテナントがすでに作成されていること。

特定のノードまたはノードの特定のポートで、EPG を作成することができます。

ステップ 1 Cisco APIC にログインします。
ステップ 2 **Tenants** > ***tenant*** を選択します。
ステップ 3 左側のナビゲーション ウィンドウで、 ***tenant***、 **Application Profiles**、および ***application profile*** を展開します。
ステップ 4 **Application EPGs** を右クリックし、**Create Application EPG** を選択します。
ステップ 5 **Create Application EPG STEP 1 > Identity** 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、EPG の名前を入力します。
b) **Bridge Domain** ドロップダウンリストから、ブリッジ ドメインを選択します。
c) **Statically Link with Leaves/Paths** チェック ボックスをオンにします

このチェック ボックスを使用して、どのポートに EPG を導入するかを指定できます。

d) **Next** をクリックします。
e) **Path** ドロップダウンリストから。宛先 EPG への静的パスを選択します。

ステップ 6 **Create Application EPG STEP 2 > Leaves/Paths** ダイアログボックスで、**Physical Domain** ドロップダウンリストから物理ドメインを選択します。
ステップ 7 次のいずれかの手順を実行します:

オプション	説明
次のものに EPG を展開する場合、	次を実行します。
ノード	1. **Leaves** エリアを展開します。 2. **Node** ドロップダウンリストからノードを選択します。 3. **Encap** フィールドで、適切な VLAN を入力します。 4. (オプション) **Deployment Immediacy** ドロップダウンリストで、デフォルトの **On Demand** のままにするか、**Immediate**を選択します。 5. (オプション) [Mode] ドロップダウンリストで、デフォルトの **Trunk** ままにするか、別のノードを選択します。
ノード上のポート	1. **Paths** エリアを展開します。 2. **Path** ドロップダウンリストから、適切なノードおよびポートを選択します。 3. (オプション) **Deployment Immediacy** フィールドのドロップダウンリストで、デフォルトの **On Demand** のままにするか、**Immediate** を選択します。 4. (オプション) [Mode] ドロップダウンリストで、デフォルトの **Trunk** ままにするか、別のノードを選択します。 5. **Port Encap** フィールドに、導入するセカンダリ VLAN を入力します。 6. (オプション) **Primary Encap** フィールドで、展開するプライマリ VLAN を入力します。

ステップ 8 **Update** をクリックし、**Finish** をクリックします。

ステップ 9 左側のナビゲーション ウィンドウで、作成した EPG を展開します。

ステップ 10 次のいずれかの操作を実行します:

- ノードで EPG を作成した場合は、**Static Leafs** をクリックし、作業ウィンドウで、静的バインド パスの詳細を表示します。
- ノードのポートで EPG を作成した場合は、**Static Ports** をクリックし、作業ウィンドウで、静的バインド パスの詳細を表示します。

NX-OS スタイルの CLI を使用した APIC の特定のポートへの EPG の導入

ステップ 1 VLAN ドメインを設定します。

例：

```
apic1(config)# vlan-domain dom1
apic1(config-vlan)# vlan 10-100
```

ステップ 2 テナントを作成します。

例：

```
apic1# configure
apic1(config)# tenant t1
```

ステップ 3 プライベート ネットワーク/VRF を作成します。

例：

```
apic1(config-tenant)# vrf context ctx1
apic1(config-tenant-vrf)# exit
```

ステップ 4 ブリッジ ドメインを作成します。

例：

```
apic1(config-tenant)# bridge-domain bd1
apic1(config-tenant-bd)# vrf member ctx1
apic1(config-tenant-bd)# exit
```

ステップ 5 アプリケーション プロファイルおよびアプリケーション EPG を作成します。

例：

```
apic1(config-tenant)# application AP1
apic1(config-tenant-app)# epg EPG1
apic1(config-tenant-app-epg)# bridge-domain member bd1
apic1(config-tenant-app-epg)# exit
apic1(config-tenant-app)# exit
apic1(config-tenant)# exit
```

ステップ 6 EPG を特定のポートに関連付けます。

例：

```
apic1(config)# leaf 1017
apic1(config-leaf)# interface ethernet 1/13
apic1(config-leaf-if)# vlan-domain member dom1
apic1(config-leaf-if)# switchport trunk allowed vlan 20 tenant t1 application AP1 epg EPG1
```

（注） 上の例に示した vlan-domain コマンドと vlan-domain member コマンドは、ポートに EPG を導入するための前提条件です。

REST API を使用した APIC の特定のポートへの EPG の導入

始める前に

EPG を導入するテナントが作成されていること。

特定のポート上に EPG を導入します。

例：

```
<fvTenant name="<tenant_name>" dn="uni/tn-test1" >
    <fvCtx name="<network_name>" pcEnfPref="enforced" knwMcastAct="permit"/>
    <fvBD name="<bridge_domain_name>" unkMcastAct="flood" >
        <fvRsCtx tnFvCtxName="<network_name>"/>
    </fvBD>
    <fvAp name="<application_profile>" >
        <fvAEPg name="<epg_name>" >
            <fvRsPathAtt tDn="topology/pod-1/paths-1017/pathep-[eth1/13]" mode="regular"
instrImedcy="immediate" encap="vlan-20"/>
        </fvAEPg>
    </fvAp>
</fvTenant>
```

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

このトピックでは、特定のポートに EPG を導入する場合に必須である物理ドメイン、接続エンティティ プロファイル（AEP）、および VLAN を作成する方法の典型的な例を示します。

（注） すべてのエンドポイント グループ（EPG）にドメインが必要です。また、インターフェイス ポリシー グループを接続エンティティ プロファイル（AEP）に関連付ける必要があり、AEP と EPG が同じドメインに存在する必要がある場合は、AEP をドメインに関連付ける必要があります。EPG とドメイン、およびインターフェイス ポリシー グループとドメインの関連付けに基づいて、EPG が使用するポートと VLAN が検証されます。以下のドメイン タイプが EPG に関連付けられます。

- アプリケーション EPG
- レイヤ 3 Outside 外部ネットワーク インスタンス EPG
- レイヤ 2 Outside 外部ネットワーク インスタンス EPG
- アウトオブバンドおよびインバンド アクセスの管理 EPG

APIC は、これらのドメイン タイプのうち 1 つまたは複数に EPG が関連付けられているかどうかを確認します。EPG が関連付けられていない場合、システムは設定を受け入れますが、エラーが発生します。ドメインの関連付けが有効でない場合、導入された設定が正しく機能しない可能性があります。たとえば、VLAN のカプセル化を EPG で使用することが有効でない場合、導入された設定が正しく機能しない可能性があります。

GUI を使用した、EPG を特定のポートに導入するためのドメインおよび VLAN の作成

始める前に

- EPG を導入するテナントがすでに作成されていること。
- EPG は特定のポートに静的に導入されます。

ステップ 1 メニュー バーで、**Fabric** > **External Access Policies** を選択します。

ステップ 2 **Navigation** ウィンドウで、**Quick Start** をクリックします。

ステップ 3 **Work** ウィンドウで、**Configure an Interface, PC, and VPC** をクリックします。

ステップ 4 **Configure an Interface, PC, and VPC** ダイアログボックスで、+アイコンをクリックしてスイッチを選択し、次の操作を実行します:

a) [Switches] ドロップダウン リストで、目的のスイッチのチェックボックスをオンにします。
b) [Switch Profile Name] フィールドに、スイッチ名が自動的に入力されます。

（注） 任意で、変更した名前を入力することができます。

c) スイッチ インターフェイスを設定するために [+] アイコンをクリックします。
d) [Interface Type] フィールドで、[Individual] オプション ボタンをクリックします。
e) [Interfaces] フィールドに、目的のインターフェイスの範囲を入力します。
f) [Interface Selector Name] フィールドに、インターフェイス名が自動的に入力されます。

（注） 任意で、変更した名前を入力することができます。

g) [Interface Policy Group] フィールドで、[Create One] オプション ボタンを選択します。
h) [Link Level Policy] ドロップダウン リストで、適切なリンク レベル ポリシーを選択します。

（注） 必要に応じて追加のポリシーを作成します。または、デフォルトのポリシー設定を使用できます。

i) [Attached Device Type] フィールドから、適切なデバイス タイプを選択します。
j) [Domain] フィールドで、[Create One] オプション ボタンをクリックします。
k) [Domain Name] フィールドに、ドメイン名を入力します。
l) [VLAN] フィールドで、[Create One] オプション ボタンをクリックします。
m) [VLAN Range] フィールドに、目的の VLAN 範囲を入力します。**[Save]** をクリックし、**[Save]** をもう一度クリックします。
n) **Submit** をクリックします。

ステップ 5 メニューバーで、**[テナント]** をクリックします。**Navigation** ウィンドウで、適切な ***Tenant_name*** > **Application Profiles** > **Application EPGs** > ***EPG_name*** を展開し、次の操作を実行します:

a) [Domains (VMs and Bare-Metals)] を右クリックし、[Add Physical Domain Association] をクリックします。
b) [Add Physical Domain Association] ダイアログボックスで、[Physical Domain Profile] ドロップダウン リストから、適切なドメインを選択します。

c) **Submit** をクリックします。
AEP は、ノード上の特定のポート、およびドメインに関連付けられます。物理ドメインは VLAN プールに関連付けられ、テナントはこの物理ドメインに関連付けられます。

スイッチ プロファイルとインターフェイス プロファイルが作成されます。インターフェイス プロファイルのポート ブロックにポリシー グループが作成されます。AEP が自動的に作成され、ポート ブロックおよびドメインに関連付けられます。ドメインは VLAN プールに関連付けられ、テナントはドメインに関連付けられます。

NX-OS スタイルの CLI を使用した、EPG を特定のポートに導入するための AEP、ドメイン、および VLAN の作成

始める前に

- EPG を導入するテナントがすでに作成されていること。
- EPG は特定のポートに静的に導入されます。

ステップ 1 VLAN ドメインを作成し、VLAN 範囲を割り当てます。

例：

```
apic1(config)# vlan-domain domP
apic1(config-vlan)# vlan 10
apic1(config-vlan)# vlan 25
apic1(config-vlan)# vlan 50-60
apic1(config-vlan)# exit
```

ステップ 2 インターフェイス ポリシー グループを作成し、そのポリシー グループに VLAN ドメインを割り当てます。

例：

```
apic1(config)# template policy-group PortGroup
apic1(config-pol-grp-if)# vlan-domain member domP
```

ステップ 3 リーフ インターフェイス プロファイルを作成し、そのプロファイルにインターフェイス ポリシー グループを割り当てて、そのプロファイルを適用するインターフェイス ID を割り当てます。

例：

```
apic1(config)# leaf-interface-profile InterfaceProfile1
apic1(config-leaf-if-profile)# leaf-interface-group range
apic1(config-leaf-if-group)# policy-group PortGroup
apic1(config-leaf-if-group)# interface ethernet 1/11-13
apic1(config-leaf-if-profile)# exit
```

ステップ 4 リーフ プロファイルを作成し、そのリーフ プロファイルにリーフ インターフェイス プロファイルを割り当てて、そのプロファイルを適用するリーフ ID を割り当てます。

例：

```
apic1(config)# leaf-profile SwitchProfile-1019
apic1(config-leaf-profile)# leaf-interface-profile InterfaceProfile1
apic1(config-leaf-profile)# leaf-group range
apic1(config-leaf-group)# leaf 1019
apic1(config-leaf-group)#
```

REST API を使用した、EPG を特定のポートに導入するための AEP、ドメイン、および VLAN の作成

始める前に

- EPG を導入するテナントがすでに作成されていること。
- EPG は特定のポートに静的に導入されます。

ステップ 1 インターフェイスプロファイル、スイッチプロファイル、および接続エンティティプロファイル（AEP）を作成します。

例：

```
   <infraInfra>

      <infraNodeP  name="<switch_profile_name>" dn="uni/infra/nprof-<switch_profile_name>" >
           <infraLeafS  name="SwitchSeletor" descr="" type="range">
                <infraNodeBlk name="nodeBlk1" descr="" to_="1019" from_="1019"/>
           </infraLeafS>
          <infraRsAccPortP tDn="uni/infra/accportprof-<interface_profile_name>"/>
      </infraNodeP>

      <infraAccPortP name="<interface_profile_name>"
dn="uni/infra/accportprof-<interface_profile_name>" >
           <infraHPortS  name="portSelector"  type="range">
              <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-<port_group_name>" fexId="101"/>

               <infraPortBlk name="block2"  toPort="13" toCard="1" fromPort="11" fromCard="1"/>
         </infraHPortS>
     </infraAccPortP>

   <infraAccPortGrp  name="<port_group_name>" dn="uni/infra/funcprof/accportgrp-<port_group_name>"
 >
          <infraRsAttEntP tDn="uni/infra/attentp-<attach_entity_profile_name>"/>
          <infraRsHIfPol tnFabricHIfPolName="1GHifPol"/>
    </infraAccPortGrp>

    <infraAttEntityP  name="<attach_entity_profile_name>"
dn="uni/infra/attentp-<attach_entity_profile_name>" >
         <infraRsDomP tDn="uni/phys-<physical_domain_name>"/>
    </infraAttEntityP>

<infraInfra>
```

ステップ 2 ドメインを作成する。

例：

```
<physDomP  name="<physical_domain_name>" dn="uni/phys-<physical_domain_name>">
    <infraRsVlanNs tDn="uni/infra/vlanns-[<vlan_pool_name>]-static"/>
</physDomP>
```

ステップ 3 VLAN 範囲を作成します。

例：

```
<fvnsVlanInstP  name="<vlan_pool_name>" dn="uni/infra/vlanns-[<vlan_pool_name>]-static"
allocMode="static">
    <fvnsEncapBlk name="" descr="" to="vlan-25" from="vlan-10"/>
</fvnsVlanInstP>
```

ステップ 4 ドメインに EPG を関連付けます。

例：

```
<fvTenant  name="<tenant_name>" dn="uni/tn-" >
    <fvAEPg prio="unspecified" name="<epg_name>" matchT="AtleastOne"
dn="uni/tn-test1/ap-AP1/epg-<epg_name>" descr="">
        <fvRsDomAtt tDn="uni/phys-<physical_domain_name>" instrImedcy="immediate"
resImedcy="immediate"/>
    </fvAEPg>
</fvTenant>
```

AEP またはインターフェイス ポリシー グループを使用したアプリケーション EPG の複数のポートへの導入

APIC の拡張 GUI と REST API を使用して、接続エンティティ プロファイルをアプリケーション EPG に直接関連付けることができます。これにより、単一の構成の接続エンティティ プロファイルに関連付けられたすべてのポートに、関連付けられたアプリケーション EPG を導入します。

APIC REST API または NX-OS スタイルの CLI を使用し、インターフェイス ポリシー グループを介して複数のポートにアプリケーション EPG を導入できます。

APIC GUI を使用した AEP による複数のインターフェイスへの EPG の導入

短時間でアプリケーションを接続エンティティプロファイルに関連付けて、その接続エンティティプロファイルに関連付けられたすべてのポートにEPGを迅速に導入することができます。

始める前に

- ターゲット アプリケーション EPG が作成されている。
- AEP での EPG 導入に使用する VLAN の範囲が含まれている VLAN プールが作成されている。
- 物理ドメインが作成され、VLAN プールと AEP にリンクされている。
- ターゲットの接続エンティティ プロファイルが作成され、アプリケーション EPG を導入するポートに関連付けられている。

ステップ 1 ターゲットの接続エンティティ プロファイルに移動します。

a) 使用する接続エンティティ プロファイルのページを開きます。拡張 GUI で、 **Fabric** > **External Access Policies** > **Policies** > **Global** > **Attachable Access Entity Profiles** をクリックします。

b) ターゲットの接続エンティティ プロファイルをクリックして、[Attachable Access Entity Profile] ウィンドウを開きます。

ステップ 2 [Show Usage] ボタンをクリックして、この接続エンティティ プロファイルに関連付けられたリーフ スイッチとインターフェイスを表示します。

この接続エンティティ プロファイルに関連付けられたアプリケーション EPG が、この接続エンティティ プロファイルに関連付けられたすべてのスイッチ上のすべてのポートに導入されます。

ステップ 3 [Application EPGs] テーブルを使用して、この接続エンティティ プロファイルにターゲット アプリケーション EPG を関連付けます。アプリケーション EPG エントリを追加するには、[+] をクリックします。各エントリに次のフィールドがあります。

フィールド	アクション
Application EPG	ドロップダウンを使用して、関連付けられたテナント、アプリケーション プロファイル、およびターゲット アプリケーション EPG を選択します。
Encap	ターゲット アプリケーション EPG の通信に使用される VLAN の名前を入力します。
Primary Encap	アプリケーション EPG にプライマリ VLAN が必要な場合は、プライマリ VLAN の名前を入力します。
Mode	ドロップダウンを使用して、データを送信するモードを指定します。 • [Trunk]：ホストからのトラフィックに VLAN ID がタグ付けされている場合に選択します。 • [Access]：ホストからのトラフィックに 802.1p タグがタグ付けされている場合に選択します。 • [Access Untagged] ：ホストからのトラフィックがタグ付けされていない場合に選択します。

ステップ 4 **Submit** をクリックします。

この接続エンティティ プロファイルに関連付けられたアプリケーション EPG が、この接続エンティティ プロファイルに関連付けられたすべてのスイッチ上のすべてのポートに導入されます。

NX-OS スタイルの CLI を使用したインターフェイス ポリシー グループによる複数のインターフェイスへの EPG の導入

NX-OS CLI では、接続エンティティ プロファイルを EPG に関連付けることによる迅速な導入が明示的に定義されていません。代わりにインターフェイス ポリシー グループが定義されてドメインが割り当てられます。このポリシー グループは、VLAN に関連付けられたすべてのポートに適用され、その VLAN を介して導入されるアプリケーション EPG を含むように設定されます。

始める前に

- ターゲット アプリケーション EPG が作成されている。
- AEP での EPG 導入に使用する VLAN の範囲が含まれている VLAN プールが作成されている。
- 物理ドメインが作成され、VLAN プールと AEP にリンクされている。
- ターゲットの接続エンティティ プロファイルが作成され、アプリケーション EPG を導入するポートに関連付けられている。

ステップ 1 ターゲット EPG をインターフェイス ポリシー グループに関連付けます。

このコマンドシーケンスの例では、VLAN ドメイン **domain1** と VLAN **1261** に関連付けられたインターフェイス ポリシー グループ **pg3** を指定します。このポリシー グループに関連付けられたすべてのインターフェイスに、アプリケーション EPG **epg47** が導入されます。

例：

```
 apic1# configure terminal
apic1(config)# template policy-group pg3
apic1(config-pol-grp-if)# vlan-domain member domain1
apic1(config-pol-grp-if)# switchport trunk allowed vlan 1261 tenant tn10 application pod1-AP
      epg epg47
```

ステップ 2 ターゲット ポートで、アプリケーション EPG に関連付けられたインターフェイス ポリシー グループのポリシーが導入されたことを確認します。

次の **show** コマンドシーケンスの出力例は、ポリシー グループ **pg3** がリーフ スイッチ **1017** 上のイーサネット ポート **1/20** に導入されていることを示しています。

例：

```
 apic1# show run leaf 1017 int eth 1/20
# Command: show running-config leaf 1017 int eth 1/20
# Time: Mon Jun 27 22:12:10 2016
  leaf 1017
    interface ethernet 1/20
      policy-group pg3
      exit
    exit
```

```
ifav28-ifc1#
```

REST API を使用した AEP による複数のインターフェイスへの EPG の導入

AEP のインターフェイス セレクタを使用して、AEPg の複数のパスを設定できます。以下を選択できます。

1. ノードまたはノード グループ
2. インターフェイスまたはインターフェイス グループ

   インターフェイスは、インターフェイス ポリシー グループ（および `infra:AttEntityP`）を使用します。
3. infra:AttEntityP を AEPg に関連付けることで、使用する VLAN を指定する。

   infra:AttEntityP は、VLAN が異なる複数の AEPg に関連付けることができます。

3 のように infra:AttEntityP を AEPg に関連付けた場合、1 で選択したノード上の 2 のインターフェイスに、3 で指定した VLAN を使用して AEPg が導入されます。

この例では、AEPg `uni/tn-Coke/ap-AP/epg-EPG1` が、ノード 101 および 102 のインターフェイス 1/10、1/11、および 1/12 に vlan-102 で導入されます。

始める前に

- ターゲット アプリケーション EPG（`AEPg`）を作成する。
- 接続エンティティ プロファイル（AEP）による EPG 導入に使用する VLAN の範囲が含まれている VLAN プールを作成する。
- 物理ドメインを作成して VLAN プールおよび AEP にリンクさせる。

選択したノードとインターフェイスに AEPg を導入するには、次の例のような XML を POST 送信します。

例：

```
<infraInfra dn="uni/infra">
  <infraNodeP name="NodeProfile">
      <infraLeafS name="NodeSelector" type="range">
         <infraNodeBlk name="NodeBlok" from_="101" to_="102"/>
         <infraRsAccPortP tDn="uni/infra/accportprof-InterfaceProfile"/>
       </infraLeafS>
  </<infraNodeP>

  <infraAccPortP name="InterfaceProfile">
     <infraHPortS name="InterfaceSelector" type="range">
         <infraPortBlk name=" InterfaceBlock" fromCard="1" toCard="1" fromPort="10" toPort="12"/>
         <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-PortGrp" />
     </infraHPortS>
  </infraAccPortP>

  <infraFuncP>
```

```
        <infraAccPortGrp name="PortGrp”>
            <infraRsAttEntP  tDn="uni/infra/attentp-AttEntityProfile"/>
        </infraAccPortGrp>
    </infraFuncP>

    <infraAttEntityP name=“AttEntityProfile” >
        <infraGeneric name=“default” >
           <infraRsFuncToEpg tDn=“uni/tn-Coke/ap-AP/epg-EPG1” encap=“vlan-102"/>
        </infraGeneric>
    </infraAttEntityP>
</infraInfra>
```

マイクロセグメント EPG

ベア メタルでのネットワークベースの属性によるマイクロセグメンテーションの使用

Cisco APIC を使用して Cisco ACI でのマイクロセグメンテーションを設定し、ネットワークベースの属性、MAC アドレス、または 1 つ以上の IP アドレスを使用した新しい属性ベースの EPG を作成できます。ネットワークベースの属性を使用して Cisco ACI でのマイクロセグメンテーションを設定し、単一のベース EPG または複数の EPG 内で VM または物理エンドポイントを分離できます。

IP ベースの属性の使用

IP ベースのフィルタを使用して、単一のマイクロセグメントで単一 IP アドレス、サブネット、または多様な非連続 IP アドレスを分離できます。ファイアウォールの使用と同様に、セキュリティ ゾーンを作成するための迅速かつ簡単な方法として、IP アドレスに基づいて物理エンドポイントを分離できます。

MAC ベースの属性の使用

MAC ベースのフィルタを使用して、単一 MAC アドレスまたは複数の MAC アドレスを分離できます。不適切なトラフィックをネットワークに送信するサーバがある場合はこの方法を推奨します。MAC ベースのフィルタを使用してマイクロセグメントを作成することで、このサーバを分離できます。

GUI を使用したベアメタル環境でのネットワークベースのマイクロセグメント EPG の設定

Cisco APIC を使用してマイクロセグメンテーションを設定し、異なる複数のベース EPG または同一の EPG に属する物理エンドポイント デバイスを新しい属性ベースの EPG に配置できます。

ステップ 1 Cisco APIC にログインします。

ステップ 2 [Tenants] を選択し、マイクロセグメントを作成するテナントを選択します。

ステップ 3 テナントのナビゲーション ウィンドウで、テナント フォルダ、[Application Profiles] フォルダ、[Profile] フォルダ、および [Application EPGs] フォルダを展開します。

ステップ 4 次のいずれかを実行します。

- 同じベース EPG の物理エンドポイント デバイスを新しい属性ベースの EPG に配置するには、物理エンドポイント デバイスを含むベース EPG をクリックします。
- 異なる複数のベース EPG の物理エンドポイント デバイスを新しい属性ベースの EPG に配置するには、物理エンドポイント デバイスを含むベース EPG の 1 つをクリックします。

ベース EPG のプロパティが作業ウィンドウに表示されます。

ステップ 5 作業ウィンドウで、画面の右上にある [Operational] タブをクリックします。

ステップ 6 [Operational] タブの下の [Client End-Points] タブがアクティブになっていることを確認します。
作業ウィンドウに、ベース EPG に属するすべての物理エンドポイントが表示されます。

ステップ 7 新しいマイクロセグメントに配置するエンドポイント デバイス（複数可）の IP アドレスまたは MAC アドレスを書き留めます。

ステップ 8 異なる複数のベース EPG のエンドポイント デバイスを新しい属性ベースの EPG に配置する場合は、各ベース EPG に対してステップ 4 ～ 7 を繰り返します。

ステップ 9 テナントのナビゲーション ウィンドウで、[uSeg EPGs] フォルダを右クリックし、[Create uSeg EPG] を選択します。

ステップ 10 以下の一連の手順を実行し、エンドポイント デバイス グループの 1 つに対して属性ベースの EPG の作成を開始します。

a) [Create uSeg EPG] ダイアログボックスで、[Name] フィールドに名前を入力します。

新しい属性ベースの EPG はマイクロセグメントであることを示す名前を選択することを推奨します。

b) [intra-EPG isolation] フィールドで [enforced] または [unenforced] を選択します。

[enforced] を選択した場合は、ACI によってこの uSeg EPG 内のエンドポイント デバイス間の通信がすべて阻止されます。

c) [Bridge Domain] エリアで、ドロップダウン リストからブリッジ ドメインを選択します。

d) [uSeg Attributes] 領域で、ダイアログボックスの右側にある [+] ドロップダウン リストから [IP Address Filter] または [MAC Address Filter] を選択します。

ステップ 11 フィルタを設定するには、次のいずれかの一連の手順を実行します。

項目	結果
IP ベースの属性	**1.** [Create IP Attribute] ダイアログボックスで、[Name] フィールドに名前を入力します。 名前については、フィルタ機能を反映したものを選択するよう推奨します。 **2.** [IP Address] フィールドに、適切なサブネット マスクの IP アドレスまたはサブネットを入力します。

項目	結果
	3. [OK] をクリックします。 4. （オプション）ステップ 10 c ～ 11 c を繰り返して、2 番目の IP アドレス フィルタを作成します。 この手順で、マイクロセグメントに不連続の IP アドレスを含めることができます。 5. [Create uSeg EPG] ダイアログボックスで、[Submit] をクリックします。
MAC ベースの属性	1. [Create MAC Attribute] ダイアログボックスで、[Name] フィールドに名前を入力します。 名前については、フィルタ機能を反映したものを選択するよう推奨します。 2. [MAC Address] フィールドに、MAC アドレスを入力します。 3. [OK] をクリックします。 4. [Create uSeg EPG] ダイアログボックスで、[Submit] をクリックします。

ステップ 12 次の手順を実行して uSeg EPG を物理ドメインに関連付けます。

a) [Navigation] ペインで、uSeg EPG フォルダが開いていることを確認し、作成したマイクロセグメントのコンテナを開きます。
b) [Domains (VMs and Bare-Metals)] フォルダをクリックします。
c) 作業ウィンドウの右側にある [Actions] をクリックし、ドロップダウンリストから [Add Physical Domain Association] を選択します。
d) [Add Physical Domain Association] ダイアログボックスで、[Physical Domain Profile] ドロップダウン リストからプロファイルを選択します。
e) [Deploy Immediacy] エリアで、デフォルトの [On Demand] を受け入れます。
f) [Resolution Immediacy] エリアで、デフォルトの [Immediate] を受け入れます。
g) [Submit] をクリックします。`

ステップ 13 uSeg EPG を適切なリーフ スイッチに関連付けます。

a) ナビゲーション ウィンドウで、uSeg EPG フォルダが開いていることを確認して [Static Leafs] をクリックします。
b) [Static Leafs] ウィンドウで、**[Actions]** > **[Statically Link with Node]** をクリックします
c) [Statically Link With Node] ダイアログで、リーフ ノードとモードを選択します。
d) **Submit** をクリックします。

ステップ 14 作成するその他のネットワーク属性ベースの EPG すべてに対してステップ 9 ～ 13 を繰り返します。

次のタスク

属性ベースの EPG が正しく作成されたことを確認します。

IP ベースまたは MAC ベースの属性を設定する場合は、新しいマイクロセグメントに配置したエンドポイント デバイスでトラフィックが動作していることを確認します。

NX-OS スタイルの CLI を使用したベアメタル環境でのネットワークベースのマイクロセグメント EPG の設定

ここでは、ベアメタル環境のベース EPG 内で、ネットワークベースの属性（IP アドレスまたは MAC アドレス）を使用して Cisco ACI でマイクロセグメンテーションを設定する方法について説明します。

手順の概要

1. CLI で、コンフィギュレーション モードに入ります。
2. マイクロセグメントを作成します。
3. EPG を導入します。
4. マイクロセグメントの作成を確認します。

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	CLI で、コンフィギュレーション モードに入ります。
	例：
	`apic1# configure`
	`apic1(config)#`	

ステップ 2	マイクロセグメントを作成します。
	例：
	この例では、IP アドレスに基づいてフィルタを使用します。
	`apic1(config)# tenant cli-ten1`
	`apic1(config-tenant)# application cli-a1`
	`apic1(config-tenant-app)# epg cli-uepg1 type micro-segmented`
	`apic1(config-tenant-app-uepg)# bridge-domain member cli-bd1`
	`apic1(config-tenant-app-uepg)# attribute cli-upg-att match ip <X.X.X.X>`
	`#Schemes to express the ip`
	`A.B.C.D IP Address`
	`A.B.C.D/LEN IP Address and mask`
	例：
	この例では、MAC アドレスに基づいてフィルタを使用します。
	`apic1(config)# tenant cli-ten1`
	`apic1(config-tenant)# application cli-a1`
	`apic1(config-tenant-app)# epg cli-uepg1 type`	

	コマンドまたはアクション	目的
	micro-segmented apic1(config-tenant-app-uepg)# bridge-domain member cli-bd1 apic1(config-tenant-app-uepg)# attribute cli-upg-att match mac <FF-FF-FF-FF-FF-FF> #Schemes to express the mac E.E.E MAC address (Option 1) EE-EE-EE-EE-EE-EE MAC address (Option 2) EE:EE:EE:EE:EE:EE MAC address (Option 3) EEEE.EEEE.EEEE MAC address (Option 4) **例**： この例では、MAC アドレスに基づいてフィルタを使用し、この uSeg EPG のすべてのメンバー間に EPG 間分離を適用します。 apic1(config)# tenant cli-ten1 apic1(config-tenant)# application cli-a1 apic1(config-tenant-app)# epg cli-uepg1 type micro-segmented apic1(config-tenant-app-uepg)# isolation enforced apic1(config-tenant-app-uepg)# bridge-domain member cli-bd1 apic1(config-tenant-app-uepg)# attribute cli-upg-att match mac <FF-FF-FF-FF-FF-FF> #Schemes to express the mac E.E.E MAC address (Option 1) EE-EE-EE-EE-EE-EE MAC address (Option 2) EE:EE:EE:EE:EE:EE MAC address (Option 3) EEEE.EEEE.EEEE MAC address (Option 4)	
ステップ 3	EPG を導入します。 **例**： この例では、EPG を導入してリーフを指定します。 apic1(config)# leaf 101 apic1(config-leaf)# deploy-epg tenant cli-ten1 application cli-a1 epg cli-uepg1 type micro-segmented	
ステップ 4	マイクロセグメントの作成を確認します。 **例**： apic1(config-tenant-app-uepg)# show running-config # Command: show running-config tenant cli-ten1 application cli-app1 epg cli-uepg1 type micro-segmented # Time: Thu Oct 8 11:54:32 2015 tenant cli-ten1 application cli-app1 epg cli-esx1bu type micro-segmented bridge-domain cli-bd1 attribute cli-uepg-att match mac 00:11:22:33:44:55 exit	

	コマンドまたはアクション	目的
	exit exit	

REST API を使用したベアメタル環境でのネットワークベースのマイクロセグメント EPG の設定

ここでは、REST API を使用してベアメタル環境の Cisco ACI でネットワーク属性のマイクロセグメンテーションを設定する方法について説明します。

手順の概要

1. Cisco APIC にログインします。
2. https://*apic-ip-address*/api/node/mo/.xml にポリシーをポストします。

手順の詳細

ステップ 1 Cisco APIC にログインします。

ステップ 2 https://*apic-ip-address*/api/node/mo/.xml にポリシーをポストします。

例：

A：次の例では、IP ベースの属性を使用して 41-subnet という名前のマイクロセグメントを設定します。

```
<polUni>
  <fvTenant dn="uni/tn-User-T1" name="User-T1">
      <fvAp dn="uni/tn-User-T1/ap-Base-EPG" name="Base-EPG">
          <fvAEPg dn="uni/tn-User-T1/ap-Base-EPG/epg-41-subnet" name="41-subnet" pcEnfPref="enforced”
 isAttrBasedEPg="yes" >
                <fvRsBd tnFvBDName="BD1" />
                <fvCrtrn name="Security1">
                   <fvIpAttr name="41-filter" ip="12.41.0.0/16"/>
                </fvCrtrn>
            </fvAEPg>
        </fvAp>
   </fvTenant>
</polUni>
```

例：

次の例は、例 A のベース EPG です。

```
<polUni>
  <fvTenant dn="uni/tn-User-T1" name="User-T1">
      <fvAp dn="uni/tn-User-T1/ap-Base-EPG" name="Base-EPG">
          <fvAEPg dn="uni/tn-User-T1/ap-Base-EPG/baseEPG” name=“baseEPG” pcEnfPref="enforced” >
               <fvRsBd tnFvBDName="BD1" />
            </fvAEPg>
        </fvAp>
   </fvTenant>
</polUni>
```

例：

B：次の例では、MAC ベースの属性を使用して useg-epg という名前のマイクロセグメントを設定します。

```
<polUni>
  <fvTenant name="User-T1">
        <fvAp name="customer">
            <fvAEPg name="useg-epg" isAttrBasedEPg="true">
              <fvRsBd tnFvBDName="BD1"/>
              <fvRsDomAtt  instrImedcy="immediate" resImedcy="immediate" tDn="uni/phys-phys" />
            <fvRsNodeAtt tDn="topology/pod-1/node-101" instrImedcy="immediate" />
            <fvCrtrn name="default">
                <fvMacAttr name="mac" mac="00:11:22:33:44:55" />
              </fvCrtrn>
            </fvAEPg>
        </fvAp>
  </fvTenant>
</polUni>
```

共有リソースとしての IP アドレスベースのマイクロセグメント EPG

IP アドレスベースのマイクロセグメント EPG を VRF（この EPG が配置されている）の内外からアクセスできるリソースとして設定できます。この場合は、既存の IP アドレスベースのマイクロセグメント EPG にサブネット（ユニキャスト IP アドレスが割り当てられている）を設定し、そのサブネットをこの EPG が属する VRF 以外の VRF にあるデバイスでアドバタイズおよび共有できるようにします。次に、EPG を共有サブネットの IP アドレスに関連付けるオプションを有効にした状態で IP 属性を定義します。

GUI を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定

VRF および現在のファブリック外のクライアントがアクセス可能な共有サービスとして、32 ビット マスクの IP アドレスを持つマイクロセグメント EPG を設定できます。

始める前に

設定に関する次の GUI の説明では、サブネット マスクが /32 に設定された IP アドレスベースのマイクロセグメント EPG が事前設定されていることを前提としています。

（注）

- 物理環境で IP アドレス ベースの EPG を設定する手順については、次を参照してください。 ベア メタルでのネットワークベースの属性によるマイクロセグメンテーションの使用 （27 ページ）
- 仮想環境で IP アドレス ベースの EPG を設定する手順については、『*Cisco ACI Virtualization Guide*』の「*Configuring Microsegmentation with Cisco ACI*」を参照してください。

ステップ 1 ターゲットとなる IP アドレスベースの EPG に移動します。

a) APIC GUI で、**[Tenant]** > **[tenant_name]** > **[uSeg EPGs]** > **[uSeg_epg_name]** をクリックして EPG の [Properties] ダイアログを表示します。

ステップ 2 ターゲット EPG では、EPG のサブネット アドレスに一致するように IP 属性を設定します。

a) [Properties] ダイアログで、[uSeg Attributes] テーブルを見つけて [+] をクリックします。プロンプトが表示されたら、**[IP Address Filter]** を選択して **[Create IP Attribute]** ダイアログを表示します。
b) [Name] フィールドに名前を入力します。
c) **[Use FV Subnet]** のチェックボックスをオンにします。

このオプションを有効にすることで、IP 属性値が共有サブネットの IP アドレスに一致することを示します。

d) **[Submit]** をクリックします。

ステップ 3 ターゲット EPG の共有サブネットを作成します。

a) ターゲットとなる IP アドレスベースの uSeg EPG のフォルダを APIC のナビゲーション ウィンドウで開いたまま、**[Subnets]** フォルダを右クリックして **[Create EPG Subnets]** を選択します。
b) **[Default Gateway]** フィールドに、IP アドレスベースのマイクロセグメント EPG の IP アドレスまたはマスクを入力します。

（注）
- いずれの場合もサブネット マスクは /32 である必要があります。
- IP アドレスベースの EPG に関しては、実際にゲートウェイのデフォルトアドレスを入力するのではなく、共有 EPG サブネットの IP アドレスを入力します。

c) **[Treat as a virtual IP address]** を選択します。
d) [Scope] で [Advertised Externally] と [Shared between VRFs] を選択します。
e) [Submit] をクリックします。`

NX-OS CLI を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定

始める前に

設定に関する次の GUI の説明では、サブネット マスクが /32 に設定された IP アドレスベースのマイクロセグメント EPG が事前設定されていることを前提としています。

手順

	コマンドまたはアクション	目的

ステップ 1	EPG をサブネットの IP アドレスに関連付けることで、共有サービスに対して IP アドレスのマイクロセグメント EPG を有効にします。
	例：
	`apic-1(config)# tenant t0`
	`apic-1(config-tenant-app)# epg cli-epg type micro-segmented`
	`apic-1(config-tenant-app-uepg)# bridge-domain member b0`
	`apic-1(config-tenant-app-uepg)# attribute ip match ip-use-epg-subnet`
		この例では、マイクロセグメント EPG（cli-epg）に **ip-use-epg-subnet** オプション（useFvSubnet）が設定され、その結果、EPG はサブネットの IP アドレスに関連付けられます。次に APIC がそのサブネット アドレスをアドバタイズし、EPG が属する VRF 以外にあるデバイスがサービスとして EPG にアクセスできるようになります。

	コマンドまたはアクション	目的
	apic-1(config-tenant-app-uepg)# show run # Command: show running-config tenant t0 application a0 epg cli-epg type micro-segmented # Time: Thu Sep 22 00:17:07 2016 tenant t0 application a0 epg cli-epg type micro-segmented bridge-domain member b0 **attribute ip match ip-use-epg-subnet** exit exit Exit	
ステップ 2	EPG をリーフに導入します。	この例では、マイクロセグメント EPG（cli epg）をリーフ 102 に導入します。 apic-1(config)# leaf 102 apic-1(config-leaf)# deploy-epg tenant t0 application a0 epg cli-epg type micro-segmented apic-1(config-leaf)# show run # Command: show running-config leaf 102 # Time: Thu Sep 22 00:18:46 2016 leaf 102 deploy-epg tenant t0 application a0 epg cli-epg type micro-segmented

REST API を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定

VRF および現在のファブリック外のクライアントがアクセス可能な共有サービスとして、32 ビット マスクの IP アドレスを持つマイクロセグメント EPG を設定できます。

手順の概要

1. 共有サブネットを持つ IP アドレス属性のマイクロセグメント EPG（epg3）を設定するには、IP アドレスと 32 ビット マスクを使用して、次の例のような XML を POST 送信します。IP 属性の **usefvSubnet** は「yes」に設定します。

手順の詳細

共有サブネットを持つIPアドレス属性のマイクロセグメントEPG（epg3）を設定するには、IPアドレスと32 ビット マスクを使用して、次の例のような XML を POST 送信します。IP 属性の **usefvSubnet** は「yes」に設定します。

例：

```
<fvAEPg descr="" dn="uni/tn-t0/ap-a0/epg-epg3" fwdCtrl=""
          isAttrBasedEPg="yes" matchT="AtleastOne" name="epg3" pcEnfPref="unenforced"
prefGrMemb="exclude"prio="unspecified">
        <fvRsCons prio="unspecified" tnVzBrCPName="ip-epg"/>
        <fvRsNodeAtt descr="" encap="unknown" instrImedcy="immediate" mode="regular"
tDn="topology/pod-2/node-106"/>
        <fvSubnet ctrl="" descr="" ip="56.4.0.2/32" name="" preferred="no"
```

```
            scope="public,shared" virtual="no"/>
        <fvRsDomAtt classPref="encap" delimiter="" encap="unknown" encapMode="auto"
instrImedcy="immediate"
            primaryEncap="unknown" resImedcy="immediate" tDn="uni/phys-vpc"/>
        <fvRsCustQosPol tnQosCustomPolName=""/>
        <fvRsBd tnFvBDName="b2"/>
        <fvCrtrn descr="" match="any" name="default" ownerKey="" ownerTag="" prec="0">
            <fvIpAttr descr="" ip="1.1.1.3" name="ipv4" ownerKey="" ownerTag="" usefvSubnet="yes"/>
        </fvCrtrn>
        <fvRsProv matchT="AtleastOne" prio="unspecified" tnVzBrCPName="ip-epg"/>
        <fvRsProv matchT="AtleastOne" prio="unspecified" tnVzBrCPName="shared-svc"/>
</fvAEPg>
```

GUI を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定解除

共有サービスとして設定された IP アドレスベースのマイクロセグメント EPG を設定解除するには、共有サブネットを削除し、さらにそのサブネットを共有リソースとして使用するオプションを無効にする必要があります。

始める前に

共有サービスとして設定された IP アドレスベースのマイクロセグメント EPG を設定解除するには、次の情報を確認しておく必要があります。

- IP アドレスベースのマイクロセグメント EPG の共有サービス アドレスとして設定されているサブネット。
- **Use FV Subnet** オプションが有効な状態で設定されている IP 属性。

ステップ 1 IP アドレスベースのマイクロセグメント EPG からサブネットを削除します。

a) APIC GUI で、**[Tenant]** > **[tenant_name]** > **[Application Profiles]** > **[epg_name]** > **[uSeg EPGs]** > **[uSeg EPGs]** > **[uSeg_epg_name]** をクリックします。

b) ターゲットとなる IP アドレスベースの uSeg EPG のフォルダを APIC のナビゲーション ウィンドウで開いたまま、[Subnets] フォルダをクリックします。

c) **Subnets** ウィンドウで、アドバタイズされて他の VRF と共有されるサブネットを選択し、**Actions** > **Delete** をクリックします。

d) [Yes] をクリックして削除を確定します。

ステップ 2 [Use FV Subnet] オプションを無効にします。

a) ターゲットとなる IP アドレスベースの uSeg EPG のフォルダを APIC のナビゲーション ウィンドウで開いたまま、マイクロセグメント EPG の名前をクリックして EPG の [Properties] ダイアログを表示します。

b) [Properties] ダイアログで、[uSeg Attributes] テーブルから [Use FV Subnet] オプションが有効になっている IP 属性の項目を見つけます。

c) その項目をダブルクリックして **Edit IP Attribute** ダイアログを表示します。

d) [Edit IP Attribute] ダイアログで、[Use FV Subnet] オプションを選択解除します。

e) [IP Address] フィールドに別の IP アドレス属性を指定します。

（注） このアドレスは、32 ビット マスクのユニキャスト アドレスである必要があります（例：124.124.124.123/32）。

f) [Submit] をクリックします。`

NX-OS スタイルの CLI を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定解除

共有サービスとして設定された IP アドレスベースのマイクロセグメント EPG を設定解除するには、その EPG の ip-use-epg-subnet オプションを無効にします。

始める前に

手順

	コマンドまたはアクション	目的

ステップ 1	ip-use-epg-subnet オプションを無効にします。
	例：
	`apic-1(config)# tenant t0`
	`apic-1(config-tenant-app)# epg cli-epg type micro-segmented`
	`apic-1(config-tenant-app-uepg)# no attribute ip match ip-use-epg-subnet`
	`apic-1(config-tenant-app-uepg)# exit`
	`apic-1(config-tenant-app)# exit`
		このコード例では、マイクロセグメント EPG「cli epg」の ip-use-epg-subnet オプションを無効にします。

REST API を使用した共有リソースとしての IP ベースのマイクロセグメント EPG の設定解除

usefvSubnet プロパティを「no」に設定することで、IP アドレスベースのマイクロセグメント EPG を無効にすることができます。

共有サービスとして現在設定されているマイクロセグメント EPG の API 構造で、usefvSubnet プロパティの値を「yes」から「no」に変更します。

この例では、IP アドレスベースのマイクロセグメント EPG「epg3」が共有サービスとして無効になります。

例：

```
<fvAEPg descr="" dn="uni/tn-t0/ap-a0/epg-epg3" fwdCtrl="" isAttrBasedEPg="yes" matchT="AtleastOne"
 name="epg3" pcEnfPref="unenforced" prefGrMemb="exclude"prio="unspecified">
   <fvRsCons prio="unspecified" tnVzBrCPName="ip-epg"/>
   <fvRsNodeAtt descr="" encap="unknown" instrImedcy="immediate" mode="regular"
tDn="topology/pod-2/node-106"/>
   <fvSubnet ctrl="" descr="" ip="56.4.0.2/32" name="" preferred="no" scope="public,shared"
virtual="no"/>
```

```
  <fvRsDomAtt classPref="encap" delimiter="" encap="unknown" encapMode="auto" instrImedcy="immediate"
 primaryEncap="unknown" resImedcy="immediate" tDn="uni/phys-vpc"/>
   <fvRsCustQosPol tnQosCustomPolName=""/>
   <fvRsBd tnFvBDName="b2"/>
   <fvCrtrn descr="" match="any" name="default" ownerKey="" ownerTag="" prec="0">
      <fvIpAttr descr="" ip="1.1.1.3" name="ipv4" ownerKey="" ownerTag="" usefvSubnet="no"/>
   </fvCrtrn>
   <fvRsProv matchT="AtleastOne" prio="unspecified" tnVzBrCPName="ip-epg"/>
   <fvRsProv matchT="AtleastOne" prio="unspecified" tnVzBrCPName="shared-svc"/>
</fvAEPg>
```

アプリケーション プロファイルと契約の導入

セキュリティ ポリシーの適用

トラフィックは前面パネルのインターフェイスからリーフ スイッチに入り、パケットは送信元 EPG の EPG でマーキングされます。リーフ スイッチはその後、テナント エリア内のパケットの宛先 IP アドレスでフォワーディング ルックアップを実行します。ヒットすると、次のシナリオのいずれかが発生する可能性があります。

1. ユニキャスト（/32）ヒットでは、宛先エンドポイントの EPG と宛先エンドポイントが存在するローカル インターフェイスまたはリモート リーフ スイッチの VTEP IP アドレスが提供されます。
2. サブネット プレフィクス（/32 以外）のユニキャスト ヒットでは、宛先サブネット プレフィクスの EPG と宛先サブネット プレフィクスが存在するローカル インターフェイスまたはリモート リーフ スイッチの VTEP IP アドレスが提供されます。
3. マルチキャスト ヒットでは、ファブリック全体の VXLAN カプセル化とマルチキャスト グループの EPG で使用するローカル レシーバのローカル インターフェイスと外側の宛先 IP アドレスが提供されます。

（注） マルチキャストと外部ルータのサブネットは、入力リーフ スイッチでのヒットを常にもたらします。セキュリティ ポリシーの適用は、宛先 EPG が入力リーフ スイッチによって認識されるとすぐに発生します。

転送テーブルの誤りにより、パケットがスパイン スイッチの転送プロキシに送信されます。転送プロキシはその後、転送テーブル検索を実行します。これが誤りである場合、パケットはドロップされます。これがヒットの場合、パケットは宛先エンドポイントを含む出力リーフ スイッチに送信されます。出力リーフ スイッチが宛先の EPG を認識するため、セキュリティ ポリシーの適用が実行されます。出力リーフ スイッチは、パケット送信元の EPG を認識する必要があります。ファブリック ヘッダーは、入力リーフ スイッチから出力リーフ スイッチに

EPG を伝送するため、このプロセスをイネーブルにします。スパイン スイッチは、転送プロキシ機能を実行するときに、パケット内の元の EPG を保存します。

出力リーフ スイッチでは、送信元 IP アドレス、送信元 VTEP、および送信元 EPG 情報は、学習によってローカルの転送テーブルに保存されます。ほとんどのフローが双方向であるため、応答パケットがフローの両側で転送テーブルに入力し、トラフィックが両方向で入力フィルタリングされます。

セキュリティ ポリシー仕様を含むコントラクト

ACI セキュリティ モデルでは、コントラクトに EPG 間の通信を管理するポリシーが含まれます。コントラクトは通信内容を指定し、EPG は通信の送信元と宛先を指定します。コントラクトは次のように EPG をリンクします。

EPG 1 ---------- コントラクト ---------- EPG 2

コントラクトで許可されていれば、EPG 1 のエンドポイントは EPG 2 のエンドポイントと通信でき、またその逆も可能です。このポリシーの構造には非常に柔軟性があります。たとえば、EPG 1 と EPG 2 間には多くのコントラクトが存在でき、1 つのコントラクトを使用する EPG が 3 つ以上存在でき、コントラクトは複数の EPG のセットで再利用できます。

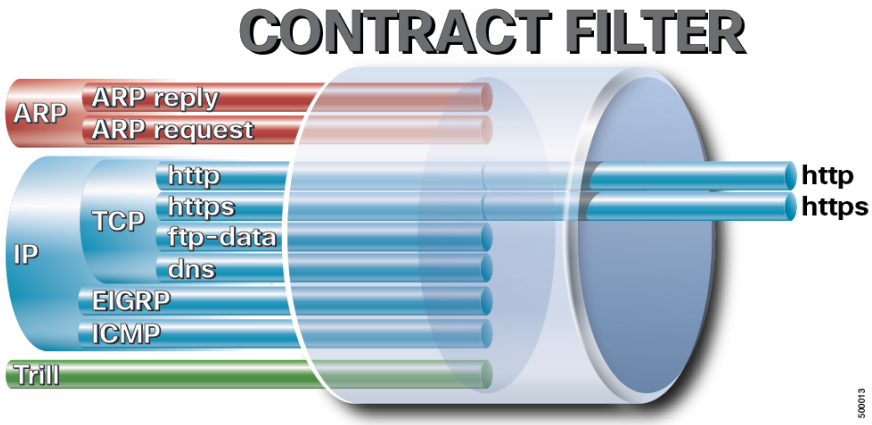
また EPG とコントラクトの関係には方向性があります。EPG はコントラクトを提供または消費できます。コントラクトを提供する EPG は通常、一連のクライアント デバイスにサービスを提供する一連のエンドポイントです。そのサービスによって使用されるプロトコルはコントラクトで定義されます。コントラクトを消費する EPG は通常、そのサービスのクライアントである一連のエンドポイントです。クライアント エンドポイント (コンシューマ) がサーバ エンドポイント (プロバイダー) に接続しようとすると、コントラクトはその接続が許可されるかどうかを確認します。特に指定のない限り、そのコントラクトは、サーバがクライアントへの接続を開始することを許可しません。ただし、EPG 間の別のコントラクトが、その方向の接続を簡単に許可する場合があります。

この提供/消費の関係は通常、EPG とコントラクト間を矢印を使って図で表されます。次に示す矢印の方向に注目してください。

EPG 1 <------- 消費 -------- コントラクト <------- 提供 -------- EPG 2

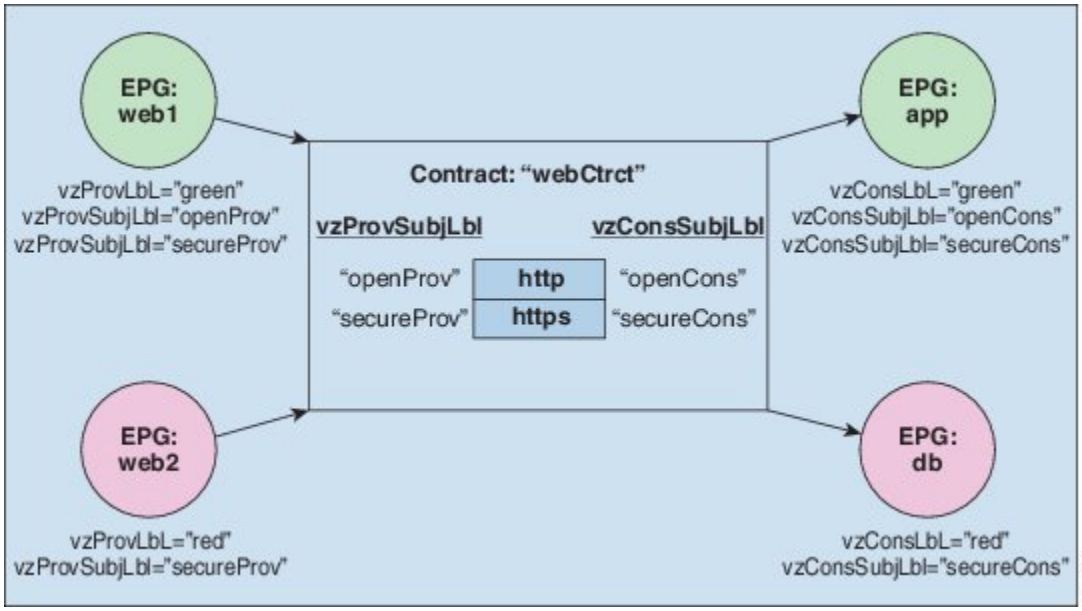
コントラクトは階層的に構築されます。1 つ以上のサブジェクトで構成され、各サブジェクトには 1 つ以上のフィルタが含まれ、各フィルタは 1 つ以上のプロトコルを定義できます。

図 *6:* コントラクト フィルタ



次の図は、コントラクトが EPG の通信をどのように管理するかを示します。

図 *7: EPG/EPG* 通信を決定するコントラクト



たとえば、TCP ポート 80 とポート 8080 を指定する HTTP と呼ばれるフィルタと、TCP ポート 443 を指定する HTTPS と呼ばれる別のフィルタを定義できます。その後、2 セットのサブジェクトを持つ webCtrct と呼ばれるコントラクトを作成できます。openProv と openCons are は HTTP フィルタが含まれるサブジェクトです。secureProv と secureCons は HTTPS フィルタが含まれるサブジェクトです。この webCtrct コントラクトは、Web サービスを提供する EPG とそのサービスを消費するエンドポイントを含む EPG 間のセキュアな Web トラフィックと非セキュアな Web トラフィックの両方を可能にするために使用できます。

これらの同じ構造は、仮想マシンのハイパーバイザを管理するポリシーにも適用されます。EPG が Virtual Machine Manager（VMM）のドメイン内に配置されると、APIC は EPG に関連付けられたすべてのポリシーを VMM ドメインに接続するインターフェイスを持つリーフ スイッ

チにダウンロードします。VMM ドメインの完全な説明については、『*Application Centric Infrastructure Fundamentals*』の「*Virtual Machine Manager Domains*」の章を参照してください。このポリシーが作成されると、APIC は EPG のエンドポイントへの接続を可能にするスイッチを指定する VMM ドメインにそれをプッシュ（あらかじめ入力）します。VMM ドメインは、EPG内のエンドポイントが接続できるスイッチとポートのセットを定義します。エンドポイントがオンラインになると、適切な EPG に関連付けられます。パケットが送信されると、送信元 EPG および宛先 EPG がパケットから取得され、対応するコントラクトで定義されたポリシーでパケットが許可されたかどうかが確認されます。許可された場合は、パケットが転送されます。許可されない場合は、パケットはドロップされます。

コントラクトは1つ以上のサブジェクトで構成されます。各サブジェクトには1つ以上のフィルタが含まれます。各フィルタには1つ以上のエントリが含まれます。各エントリは、アクセス コントロール リスト (ACL) の 1 行に相当し、エンドポイント グループ内のエンドポイントが接続されているリーフ スイッチで適用されます。

詳細には、コントラクトは次の項目で構成されます。

- 名前：テナントによって消費されるすべてのコントラクト (common テナントまたはテナント自体で作成されたコントラクトを含む) にそれぞれ異なる名前が必要です。
- サブジェクト：特定のアプリケーションまたはサービス用のフィルタのグループ。
- フィルタ：レイヤ 2 ～ レイヤ 4 の属性 (イーサネット タイプ、プロトコル タイプ、TCP フラグ、ポートなど) に基づいてトラフィックを分類するために使用します。
- アクション：フィルタリングされたトラフィックで実行されるアクション。次のアクションがサポートされます。
  - トラフィックの許可 (通常のコントラクトのみ)
  - トラフィックのマーク (DSCP/CoS) (通常のコントラクトのみ)
  - トラフィックのリダイレクト (サービス グラフによる通常のコントラクトのみ)
  - トラフィックのコピー (サービス グラフまたは SPAN による通常のコントラクトのみ)
  - トラフィックのブロック（禁止コントラクトのみ）

    Cisco APIC リリース 3.2(x) および名前が EX または FX で終わるスイッチでは、標準コントラクトで代わりに件名 [拒否] アクションまたは [コントラクトまたは件名の除外] を使用して、指定のパターンを持つトラフィックをブロックできます。
  - トラフィックのログ（禁止コントラクトと通常のコントラクト）
- エイリアス：(任意) 変更可能なオブジェクト名。オブジェクト名は作成後に変更できませんが、エイリアスは変更できるプロパティです。

このように、コントラクトによって許可や拒否よりも複雑なアクションが可能になります。コントラクトは、所定のサブジェクトに一致するトラフィックをサービスにリダイレクトしたり、コピーしたり、その QoS レベルを変更したりできることを指定可能です。具象モデルでアクセス ポリシーをあらかじめ入力すると、APIC がオフラインまたはアクセスできない場合でも、エンドポイントは移動でき、新しいエンドポイントをオンラインにでき、通信を行うこ

とができます。APIC は、ネットワークの単一の障害発生時点から除外されます。ACI ファブリックにパケットが入力されると同時に、セキュリティ ポリシーがスイッチで実行している具象モデルによって適用されます。

Three-Tier アプリケーションの展開

フィルタは、フィルタを含むコントラクトにより許可または拒否されるデータ プロトコルを指定します。コントラクトには、複数のサブジェクトを含めることができます。サブジェクトは、単方向または双方向のフィルタを実現するために使用できます。単方向フィルタは、コンシューマからプロバイダー（IN）のフィルタまたはプロバイダーからコンシューマ（OUT）のフィルタのどちらか一方向に使用されるフィルタです。双方向フィルタは、両方の方向で使用される同一フィルタです。これは、再帰的ではありません。

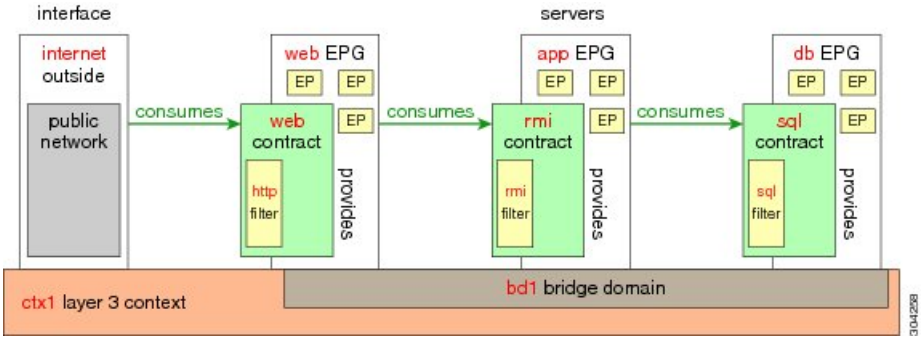
コントラクトは、エンドポイント グループ間 (EPG 間) の通信をイネーブルにするポリシーです。これらのポリシーは、アプリケーション層間の通信を指定するルールです。コントラクトが EPG に付属していない場合、EPG 間の通信はデフォルトでディセーブルになります。EPG 内の通信は常に許可されているので、EPG 内の通信には契約は必要ありません。

アプリケーション プロファイルでは、APIC がその後ネットワークおよびデータセンターのインフラストラクチャで自動的にレンダリングするアプリケーション要件をモデル化することができます。アプリケーションプロファイルでは、管理者がインフラストラクチャの構成要素ではなくアプリケーションの観点から、リソースプールにアプローチすることができます。アプリケーション プロファイルは、互いに論理的に関連する EPG を保持するコンテナです。EPG は同じアプリケーション プロファイル内の他の EPG および他のアプリケーション プロファイル内の EPG と通信できます。

アプリケーション ポリシーを展開するには、必要なアプリケーション プロファイル、フィルタ、および契約を作成する必要があります。通常、APIC ファブリックは、テナント ネットワーク内の Three-Tier アプリケーションをホストします。この例では、アプリケーションは 3 台のサーバ（Web サーバ、アプリケーション サーバ、およびデータベース サーバ）を使用して実行されます。Three-Tier アプリケーションの例については、次の図を参照してください。

Web サーバには HTTP フィルタがあり、アプリケーション サーバには Remote Method Invocation（RMI）フィルタがあり、データベース サーバには Structured Query Language（SQL）フィルタがあります。アプリケーション サーバは、SQL コントラクトを消費してデータベース サーバと通信します。Web サーバは、RMI コントラクトを消費して、アプリケーション サーバと通信します。トラフィックは Web サーバから入り、アプリケーション サーバと通信します。アプリケーション サーバはその後、データベース サーバと通信し、トラフィックは外部に通信することもできます。

図 *8 : Three-Tier* アプリケーションの図



http 用のフィルタを作成するパラメータ

この例での http 用のフィルタを作成するパラメータは次のとおりです。

パラメータ名	http のフィルタ
名前	http
エントリの数	2
エントリ名	Dport-80 Dport-443
Ethertype	IP
プロトコル	tcp tcp
宛先ポート	http https

rmi および sql 用のフィルタを作成するパラメータ

この例での rmi および sql 用のフィルタを作成するパラメータは次のとおりです。

パラメータ名	rmi のフィルタ	sql のフィルタ
名前	rmi	sql
エントリの数	1	1
エントリ名	Dport-1099	Dport-1521
Ethertype	IP	IP
プロトコル	tcp	tcp

パラメータ名	rmi のフィルタ	sql のフィルタ
宛先ポート	1099	1521

アプリケーション プロファイル データベースの例

この例のアプリケーション プロファイル データベースは次のとおりです。

EPG	提供される契約	消費される契約
web	web	rmi
app	rmi	sql
db	sql	--

GUI を使用したアプリケーション プロファイルの作成

手順の概要

1. メニュー バーで、**[TENANTS]** を選択します。**[Navigation]** ペインで、テナントを展開し、**[Application Profiles]** を右クリックし、**[Create Application Profile]** をクリックします。
2. **[Create Application Profile]** ダイアログボックスで、**[Name]** フィールドに、アプリケーション プロファイル名 (OnlineStore) を追加します。

手順の詳細

ステップ 1 メニュー バーで、**[TENANTS]** を選択します。**[Navigation]** ペインで、テナントを展開し、**[Application Profiles]** を右クリックし、**[Create Application Profile]** をクリックします。

ステップ 2 **[Create Application Profile]** ダイアログボックスで、**[Name]** フィールドに、アプリケーション プロファイル名 (OnlineStore) を追加します。

GUI を使用した EPG の作成

EPG が使用するポートは、VM マネージャ (VMM) ドメインまたは EPG に関連付けられた物理ドメインのいずれか 1 つに属している必要があります。

ステップ 1 メニュー バーで、[Tenants]、EPG を作成するテナントの順に選択します。

ステップ 2 ナビゲーション ペインで、テナントのフォルダ、[Application Profiles] フォルダ、アプリケーション プロファイルのフォルダの順に展開します。

ステップ 3 [Application EPG] フォルダを右クリックし、[Create Application EPG] 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、EPG の名前（db）を追加します。

b) [Bridge Domain] フィールドで、ドロップダウン リストからブリッジ ドメイン（bd1）を選択します。

c) [Associate to VM Domain Profiles] チェックボックスをオンにします。[Next] をクリックします。

d) [STEP 2 > Domains] エリアで、[Associate VM Domain Profiles] を展開し、ドロップダウンリストから対象の VMM ドメインを選択します。

e) [Deployment Immediacy] ドロップダウンリストで、デフォルト値を受け入れるか、いつポリシーが Cisco APIC から物理リーフ スイッチに展開されるかを選択します。

f) [Resolution Immediacy] ドロップダウンリストで、いつポリシーが物理リーフ スイッチから仮想リーフに展開されるかを選択します。

Cisco AVS がある場合には、**Immediate** または **On Demand** を選択します。Cisco ACI Virtual Edge または VMware VDS がある場合には、**Immediate**、 **On Demand**、または **Pre-provision** を選択します。

g) （オプション）[Delimiter] フィールドに、|、~、!、@、^、+、または= のいずれかの記号を入力します。

記号を入力しなかった場合、システムは VMware ポートグループ名のデリミタとしてデフォルトの | を使用します。

h) Cisco ACI Virtual Edge または Cisco AVS を利用している場合は、[Encap Mode] ドロップダウンリストからカプセル化モードを選択します。

次のいずれかのカプセル化モードを選択できます。

- [VXLAN]： これはドメインの VLAN 設定をオーバーライドし、EPG は VXLAN カプセル化を使用します。ただし、ドメインでマルチキャスト プールが設定されていない場合は、EPG に対してエラーが発生します。
- [VLAN]：これはドメインの VXLAN 設定より優先され、EPG は VLAN のカプセル化を使用することになります。ただし、ドメインで VLAN プールが設定されていない場合は、EPG に対してエラーがトリガーされます。
- **Auto** — EPG は、VMM ドメインと同じカプセル化モードを使用します。これはデフォルトの設定です。

i) Cisco ACI Virtual Edge がある場合、**Switching Mode** ドロップダウンリストで、 **native** または **AVE** を選択します。

native を選択した場合、EPGはVMware VDS を通して切り替えられます。**AVE** を選択した場合、EPGはCisco ACI Virtual Edge を通して切り替えられます。デフォルトは**native** す。

j) **Update** をクリックし、**Finish** をクリックします。

ステップ 4 **Create Application Profile** ダイアログボックスで、EPG をさらに 2 つ作成します。同じブリッジ ドメイン、同じデータセンター内に、3 つの EPG を作成します。これらは、db、app、および web です。

APIC GUI を使用したコントラクトの設定

GUI を使用したフィルタの作成

3つの個別のフィルタを作成します。この例では、HTTP、RMI、SQL です。このタスクでは、HTTP フィルタを作成する方法を示します。このタスクは、他のフィルタを作成するタスクと同じです。

始める前に

テナント、ネットワーク、およびブリッジ ドメインが作成されていることを確認します。

手順の概要

1. メニュー バーで、[テナント] を選択します。**Navigation** ウィンドウで、***tenant-name*** > **Contracts** を選択し、**Filters** を選択し、**Create Filter** をクリックします。
2. [Create Filter] ダイアログボックスで、次の操作を実行します。
3. [Name] フィールドの [Entries] を展開します。同じプロセスを実行して、別のエントリを**宛先**ポートとして HTTPS で追加し、[Update] をクリックします。
4. さらに 2 つのフィルタ（rmi および sql）を作成し、rmi および sql 用のフィルタを作成するパラメータ （43 ページ）に示すパラメータを使用するには、上記手順の同じプロセスを実行します。

手順の詳細

ステップ 1 メニューバーで、[テナント] を選択します。**Navigation** ウィンドウで、***tenant-name*** > **Contracts** を選択し、**Filters** を選択し、**Create Filter** をクリックします。

（注） [Navigation] ペインで、フィルタを追加するテナントを展開します。

ステップ 2 [Create Filter] 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、フィルタ名（http）を入力します。
b) [Entries] を展開し、[Name] フィールドに、名前（Dport-80）を入力します。
c) [EtherType] ドロップダウン リストから、EtherType（IP）を選択します。
d) [IP Protocol] ドロップダウン リストから、プロトコル（tcp）を選択します。
e) [Destination Port/Range] ドロップダウン リストから、[From] フィールドと [To] フィールドで、[http] を選択します。（http）
f) [Update] をクリックし、[Submit] をクリックします。
新しく追加されたフィルタが、[Navigation] ペインと [Work] ペインに表示されます。

ステップ 3 [Name] フィールドの [Entries] を展開します。同じプロセスを実行して、別のエントリを**宛先**ポートとして HTTPS で追加し、[Update] をクリックします。

この新しいフィルタ ルールが追加されます。

ステップ 4 さらに 2 つのフィルタ（rmi および sql）を作成し、rmi および sql 用のフィルタを作成するパラメータ （43 ページ）に示すパラメータを使用するには、上記手順の同じプロセスを実行します。

GUI を使用した契約の作成

手順の概要

1. メニュー バーで **Tenants** を選択し、実行するテナント名を選択します。**Navigation** ウィンドウで、***tenant-name*** > **Contracts** を展開します。
2. **Standard** > **Create Contract** を右クリックします。
3. [Create Contract] ダイアログボックスで、次のタスクを実行します。
4. [Create Contract Subject] ダイアログボックスで、[OK] をクリックします。
5. この手順と同じステップに従って、rmi と sql 用の契約をさらに 2 つ作成します。rmi 契約の場合は rmi サブジェクトを選択し、sql の場合は sql サブジェクトを選択します。

手順の詳細

ステップ 1 メニュー バーで **Tenants** を選択し、実行するテナント名を選択します。**Navigation** ウィンドウで、***tenant-name*** > **Contracts** を展開します。

ステップ 2 **Standard** > **Create Contract** を右クリックします。

ステップ 3 [Create Contract] ダイアログボックスで、次のタスクを実行します。

a) [Name] フィールドに、契約名（web）を入力します。

b) [Subjects] の横の [+] 記号をクリックし、新しいサブジェクトを追加します。

c) [Create Contract Subject] ダイアログボックスで、[Name] フィールドにサブジェクト名を入力します。（web）

d) （注） この手順では、契約のサブジェクトで前に作成されたフィルタを関連付けます。

[Filter Chain] 領域で、[Filters] の横の [+] 記号をクリックします。

e) ダイアログボックスで、ドロップダウンメニューから、フィルタ名（http）を選択し、[Update] をクリックします。

ステップ 4 [Create Contract Subject] ダイアログボックスで、[OK] をクリックします。

ステップ 5 この手順と同じステップに従って、rmi と sql 用の契約をさらに 2 つ作成します。rmi 契約の場合は rmi サブジェクトを選択し、sql の場合は sql サブジェクトを選択します。

GUI を使用した契約の消費と提供

EPG 間のポリシー関係を作成するために、前に作成した契約を関連付けることができます。

提供するコントラクトと使用するコントラクトに名前を付けるときは、提供するコントラクトと使用するコントラクトの両方に同じ名前を付けてください。

手順の概要

1. APIC GUI ウィンドウをクリックして db EPG から app EPG にドラッグします。
2. [Name] フィールドで、ドロップダウン リストから、**sql** 契約を選択します。[OK] をクリックします。
3. APIC GUI 画面 をクリックして、app ePG から web EPG にドラッグします。
4. [Name] フィールドで、ドロップダウン リストから、**rmi** 契約を選択します。[OK] をクリックします。
5. web EPG のアイコンをクリックし、[Provided Contracts] 領域の [+] 記号をクリックします。
6. [Name] フィールドで、ドロップダウン リストから、**web** 契約を選択します。[OK] をクリックします。[Submit] をクリックします。`
7. 確認するには、[Navigation] ペインで、[Application Profiles] 下の [OnlineStore] に移動してクリックします。
8. [Work] ペインで、**[Operational]** > **[Contracts]** を選択します。

手順の詳細

ステップ 1 （注） db、app、および web EPG は、アイコンで表示されます。

APIC GUI ウィンドウをクリックして db EPG から app EPG にドラッグします。
[Add Consumed Contract] ダイアログボックスが表示されます。

ステップ 2 [Name] フィールドで、ドロップダウン リストから、**sql** 契約を選択します。[OK] をクリックします。
この手順により、db EPG は sql 契約を提供でき、app EPG は sql 契約を消費することができます。

ステップ 3 APIC GUI 画面 をクリックして、app ePG から web EPG にドラッグします。
[Add Consumed Contract] ダイアログボックスが表示されます。

ステップ 4 [Name] フィールドで、ドロップダウン リストから、**rmi** 契約を選択します。[OK] をクリックします。
この手順により、app EPG は rmi 契約を提供でき、web EPG は rmi 契約を消費することができます。

ステップ 5 web EPG のアイコンをクリックし、[Provided Contracts] 領域の [+] 記号をクリックします。
[Add Provided Contract] ダイアログボックスが表示されます。

ステップ 6 [Name] フィールドで、ドロップダウン リストから、**web** 契約を選択します。[OK] をクリックします。
[Submit] をクリックします。`
OnlineStore と呼ばれる 3 層アプリケーション プロファイルが作成されました。

ステップ 7 確認するには、[Navigation] ペインで、[Application Profiles] 下の [OnlineStore] に移動してクリックします。
[Work] ペインで、3 つの EPG app、db および web が表示されていることを確認できます。

ステップ 8 [Work] ペインで、**[Operational]** > **[Contracts]** を選択します。
消費/提供される順番で表示された EPG と契約を確認できます。

NX-OS スタイルの CLI を使用したコントラクトの設定

コントラクトの設定

コントラクトは次のタスクでテナントの下に設定します。

- アクセス リストとしてフィルタを定義します
- コントラクトおよびサブジェクトを定義します
- EPG にコントラクトをリンクします

タスクは、この順序に従う必要はありません。たとえば、コントラクトを定義する前に、EPG にコントラクト名をリンクすることができます。

（注） APIC のフィルタ（ACL）では、従来の NX-OS ACL の **permit | deny** の代わりに **match** が使用されます。フィルタ エントリの目的は、特定のトラフィック フローを一致させることだけです。トラフィックは、ACL にコントラクトまたはタブー コントラクトが適用されると、許可または拒否されます。

手順の概要

1. **configure**
2. **tenant** *tenant-name*
3. **access-list** *acl-name*
4. （任意） **match** {**arp** | **icmp** | **ip**}
5. （任意） **match** {**tcp** | **udp**} [**src** *from*[-*to*]] [**dest** *from*[-*to*]]
6. （任意） **match raw** *options*
7. **exit**
8. **contract** *contract-name*
9. **subject** *subject-name*
10. （任意） **[no] access-group** *acl-name* [**in** | **out** | **both**]
11. （任意） **[no] label name** *label-name* {**provider** | **consumer**}
12. （任意） **[no] label match** {**provider** | **consumer**} [**any** | **one** | **all** | **none**]
13. **exit**
14. **exit**
15. **application** *app-name*
16. **epg** *epg-name*
17. **bridge-domain member** *bd-name*
18. **contract provider** *provider-contract-name*
19. **contract consumer** *consumer-contract-name*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `apic1# configure`	コンフィギュレーション モードに入ります。
ステップ 2	**tenant** *tenant-name* 例： `tenant exampleCorp`	テナント (存在しない場合) を作成し、テナント コンフィギュレーション モードを開始します。
ステップ 3	**access-list** *acl-name* 例： `apic1(config-tenant)# access-list http_acl`	コントラクトで利用できるアクセス リスト(フィルタ) を作成します。
ステップ 4	（任意） **match** {**arp** | **icmp** | **ip**} 例： `apic1(config-tenant-acl)# match arp`	選択したプロトコルのトラフィックに一致するルールを作成します。

ステップ 5	（任意） **match** {**tcp** | **udp**} [**src** *from*[-*to*]] [**dest** *from*[-*to*]]
	例：
	`apic1(config-tenant-acl)# match tcp dest 80`
	`apic1(config-tenant-acl)# match tcp dest 443`
		TCP または UDP トラフィックに一致するルールを作成します。

ステップ 6	（任意） **match raw** *options* 例： `apic1(config-tenant-acl)#`	Raw vzEntry に一致するルールを作成します。
ステップ 7	**exit** 例： `apic1(config-tenant-acl)# exit`	テナント コンフィギュレーション モードに戻ります。
ステップ 8	**contract** *contract-name* 例： `apic1(config-tenant)# contract web80`	コントラクトを作成し、コントラクト コンフィギュレーション モードを開始します。
ステップ 9	**subject** *subject-name* 例： `apic1(config-tenant-contract)# subject web80`	コントラクト サブジェクトを作成し、サブジェクト コンフィギュレーション モードを開始します。
ステップ 10	（任意） **[no] access-group** *acl-name* [**in** | **out** | **both**] 例：	一致するトラフィックの方向を指定し、コントラクトからアクセス リストを追加 (削除) します。

	コマンドまたはアクション	目的
	`apic1(config-tenant-contract-subj)# access-group http_acl both`	
ステップ 11	（任意） **[no] label name** *label-name* {**provider** | **consumer**} 例： `apic1(config-tenant-contract-subj)#`	サブジェクトにプロバイダーまたはコンシューマのラベルを追加 (削除) します。
ステップ 12	（任意） **[no] label match** {**provider** | **consumer**} [**any** | **one** | **all** | **none**] 例： `apic1(config-tenant-contract-subj)#`	次のプロバイダーまたはコンシューマのラベルの一致タイプを指定します。 • **any**：任意のラベルにコントラクト関係がある場合の一致のこと。 • **one**：1 つのラベルにコントラクト関係がある場合の一致のこと。 • **all**：すべてのラベルにコントラクト関係がある場合の一致のこと。 • **none**：ラベルにコントラクト関係がない場合の一致のこと。
ステップ 13	**exit** 例： `apic1(config-tenant-contract-subj)# exit`	コントラクト コンフィギュレーション モードに戻ります。
ステップ 14	**exit** 例： `apic1(config-tenant-contract)# exit`	テナント コンフィギュレーション モードに戻ります。
ステップ 15	**application** *app-name* 例： `apic1(config-tenant)# application OnlineStore`	アプリケーション コンフィギュレーション モードを開始します。
ステップ 16	**epg** *epg-name* 例： `apic1(config-tenant-app)# epg exampleCorp_webepg1`	コントラクトにリンクするEPGのコンフィギュレーション モードを開始します。
ステップ 17	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member exampleCorp_bd1`	この EPG のブリッジ ドメインを指定します。
ステップ 18	**contract provider** *provider-contract-name* 例：	この EPG のプロバイダー コントラクトを指定します。この EPG との通信は、このプロバイダー コン

	コマンドまたはアクション	目的
	`apic1(config-tenant-app-epg)# contract provider web80`	トラクトに従う通信である限り、その他の EPG から開始することができます。
ステップ 19	**contract consumer** *consumer-contract-name* 例： `apic1(config-tenant-app-epg)# contract consumer rmi99`	この EPG のコンシューマ コントラクトを指定します。この EPG のエンドポイントは、このコントラクトを提供する EPG の任意のエンドポイントとの通信を開始することができます。

例

この例では、EPG にコントラクトを作成し適用する方法を示します。

```
apic1# configure
apic1(config)# tenant exampleCorp

    # CREATE FILTERS
apic1(config-tenant)# access-list http_acl
apic1(config-tenant-acl)# match tcp dest 80
apic1(config-tenant-acl)# match tcp dest 443
apic1(config-tenant-acl)# exit

    # CREATE CONTRACT WITH FILTERS
apic1(config-tenant)# contract web80
apic1(config-tenant-contract)# subject web80
apic1(config-tenant-contract-subj)# access-group http_acl both
apic1(config-tenant-contract-subj)# exit
apic1(config-tenant-contract)# exit

    # ASSOCIATE CONTRACTS TO EPG
apic1(config-tenant)# application OnlineStore
apic1(config-tenant-app)# epg exampleCorp_webepg1
apic1(config-tenant-app-epg)# bridge-domain member exampleCorp_bd1
apic1(config-tenant-app-epg)# contract consumer rmi99
apic1(config-tenant-app-epg)# contract provider web80
apic1(config-tenant-app-epg)# exit
apic1(config-tenant-app)#exit
apic1(config-tenant)#exit

    # ASSOCIATE PORT AND VLAN TO EPG
apic1(config)#leaf 101
apic1(config-leaf)# interface ethernet 1/4
apic1(config-leaf-if)# switchport trunk allowed vlan 102 tenant exampleCorp application
 OnlineStore epg exampleCorp_webepg1
```

この例では、コントラクト自体のフィルタインラインを宣言してコントラクトを定義するためのシンプルな方法を示します。

```
apic1# configure
apic1(config)# tenant exampleCorp
apic1(config-tenant)# contract web80
apic1(config-tenant-contract)# match tcp 80
apic1(config-tenant-contract)# match tcp 443
```

他のテナントへのコントラクトのエクスポート

1 つのテナントからコントラクトをエクスポートし、別のテナントにインポートできます。コントラクトをインポートするテナントでは、コントラクトはコンシューマコントラクトとしてのみ適用できます。コントラクトはエクスポート時に名前を変更できます。

手順の概要

1. **configure**
2. **tenant** *tenant-name*
3. **contract** *contract-name*
4. **scope** {**application** | **exportable** | **tenant** | **vrf**}
5. **export to tenant** *other-tenant-name* **as** *new-contract-name*
6. **exit**
7. **exit**
8. **tenant** *tenant-name*
9. **application** *app-name*
10. **epg** *epg-name*
11. **contract consumer** *consumer-contract-name* **imported**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `apic1# configure`	コンフィギュレーション モードに入ります。
ステップ 2	**tenant** *tenant-name* 例： `apic1(config)# tenant RedCorp`	エクスポートするテナントのテナント コンフィギュレーション モードを開始します。
ステップ 3	**contract** *contract-name* 例： `apic1(config-tenant)# contract web80`	エクスポートするコントラクトのコントラクト コンフィギュレーション モードを開始します。
ステップ 4	**scope** {**application**	**exportable**	**tenant**	**vrf**} 例： `apic1(config-tenant-contract)# scope exportable`	コントラクトの共有方法を設定します。スコープは次のようになります。 • **application**—同じアプリケーションの EPG で共有可能 • **exportable**—テナントで共有可能 • **tenant**—同じテナントの EPG で共有可能 • **vrf**—同じ VRF の EPG で共有可能

	コマンドまたはアクション	目的
ステップ 5	**export to tenant** *other-tenant-name* **as** *new-contract-name* 例： `apic1(config-tenant-contract)# export to tenant BlueCorp as webContract1`	他のテナントにコントラクトをエクスポートします。同じコントラクト名を使用することも、名前を変更することもできます。
ステップ 6	**exit** 例： `apic1(config-tenant-contract)# exit`	テナント コンフィギュレーション モードに戻ります。
ステップ 7	**exit** 例： `apic1(config-tenant)# exit`	グローバル コンフィギュレーション モードに戻ります。
ステップ 8	**tenant** *tenant-name* 例： `tenant BlueCorp`	インポートするテナントのテナント コンフィギュレーション モードを開始します。
ステップ 9	**application** *app-name* 例： `apic1(config-tenant)# application BlueStore`	アプリケーション コンフィギュレーション モードを開始します。
ステップ 10	**epg** *epg-name* 例： `apic1(config-tenant-app)# epg BlueWeb`	コントラクトにリンクするEPGのコンフィギュレーション モードを開始します。
ステップ 11	**contract consumer** *consumer-contract-name* **imported** 例： `apic1(config-tenant-app-epg)# contract consumer webContract1 imported`	このEPGにインポートされたコンシューマ コントラクトを指定します。このEPGのエンドポイントは、このコントラクトを提供するEPGの任意のエンドポイントとの通信を開始することができます。

例

次に、テナント RedCorp から、コンシューマ コントラクトになるテナント BlueCorp にコントラクトをエクスポートする例を示します。

```
apic# configure
apic1(config)# tenant RedCorp
apic1(config-tenant)# contract web80
apic1(config-tenant-contract)# scope exportable
apic1(config-tenant-contract)# export to tenant BlueCorp as webContract1
apic1(config-tenant-contract)# exit
apic1(config-tenant)# exit
apic1(config)# tenant BlueCorp
apic1(config-tenant)# application BlueStore
```

```
apic1(config-tenant-application)# epg BlueWeb
apic1(config-tenant-application-epg)# contract consumer webContract1 imported
```

REST API を使用したコントラクトの設定

REST API を使用したコントラクトの設定

手順の概要

1. 次の例のように、XML POST 要求を使用してコントラクトを設定します。

手順の詳細

次の例のように、XML POST 要求を使用してコントラクトを設定します。

例：

```
<vzBrCP name="webCtrct">
         <vzSubj name="http" revFltPorts="true" provmatchT="All">
             <vzRsSubjFiltAtt tnVzFilterName="Http"/>
             <vzRsSubjGraphAtt graphName="G1" termNodeName="TProv"/>
             <vzProvSubjLbl name="openProv"/>
             <vzConsSubjLbl name="openCons"/>
         </vzSubj>
         <vzSubj name="https" revFltPorts="true" provmatchT="All">
             <vzProvSubjLbl name="secureProv"/>
             <vzConsSubjLbl name="secureCons"/>
             < vzRsSubjFiltAtt tnVzFilterName="Https"/>
             <vzRsOutTermGraphAtt graphName="G2" termNodeName="TProv"/>
         </vzSubj>
     </vzBrCP>
```

REST API を使用した禁止コントラクトの設定

始める前に

次のオブジェクトを作成する必要があります。

- これで関連付けられるテナント **Taboo 契約**
- テナントのアプリケーション プロファイル
- テナントの最低 1 個の EPG

REST API を使用してタブー契約を作成するには、次の例ではよう XML を使用します。

例：

```
<vzTaboo ownerTag="" ownerKey="" name="VRF64_Taboo_Contract"
dn="uni/tn-Tenant64/taboo-VRF64_Taboo_Contract" descr=""><vzTSubj
name="EPG_subject" descr=""><vzRsDenyRule tnVzFilterName="default"
directives="log"/>
</vzTSubj>
</vzTaboo>
```

契約、タブー契約は、REST API を使用してフィルタの確認

このトピックでは、契約、タブー契約は、およびフィルタを確認する REST API XML を提供します。

ステップ 1 プロバイダーの EPG または XML で、次の例などの外部ネットワークには、契約を確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/fvRsProv.xml
```

ステップ 2 消費者の次の例など、EPG と XML の契約を確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/fvRsCons.xml
```

ステップ 3 次の例など XML を使用してエクスポートされた契約を確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/vzCPif.xml
```

ステップ 4 次の例などと XML の VRF の契約を確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/vzBrCP.xml
```

ステップ 5 次の例などと XML タブー契約を確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/vzTaboo.xml
```

EPG のタブー契約は、Epg の契約と同じクエリを使用します。

ステップ 6 次の例など XML を使用してフィルタを確認します。

例：

```
QUERY https://apic-ip-address/api/node/class/vzFilter.xml
```

コントラクト パフォーマンスの最適化

契約のパフォーマンスの最適化

Cisco APIC、リリース 3.2 で始まるより効率的なハードウェア契約データの TCAM ストレージをサポートしている双方向契約を設定できます。最適化を有効になっている、両方向の統計情報を契約は統合します。

TCAM 最適化は名前で終わるとラック (TOR) スイッチの Cisco Nexus 9000 シリーズの上部でサポート EX と FX、以降 (たとえば、N9K-C93180LC-EX または N9K-C93180YC-FX)。

TCAM 契約の効率的なデータ ストレージを設定するには、次のオプションが有効にします。

- コンシューマとプロバイダー間を両方向で適用する契約をマークします。
- リバース ポート オプションを有効に、IP TCP または UDP プロトコルによるフィルタ
- 契約件名を設定するときに有効にする、 **no 統計** directive。

制限事項

No_stats オプションを有効にして、ルールごとの統計情報は失われます。ただし両方向の統計情報を連結ルールにはハードウェアの統計情報があります。

追加の Cisco APIC 3.2(1) へのアップグレード後、 `no 統計` オプション (フィルタとフィルタ エントリ) のアップグレード前の契約 subject, には必要があります件名を削除し、再設定すると、 `no 統計` オプション。そうしないと、圧縮は行われません。

2 ルール、1 つのルールでの双方向サブジェクト フィルタと各の契約では、Cisco NX-OS を作成、 `sPcTag` と `dPcTag` マークされた `方向 = 双 dir` 、ハードウェアにプログラムされますが、別のルールが付いている `方向uni dir 無視 =` が設定されていません。

次の設定とルールは圧縮されません。

- ルールの優先順位を持つ `fully_qual`
- ルールの反対側 (`双 dir` および `uni dir 無視` マーク) と同一ではないプロパティは、次のように **アクション** を含む **統制** 、 **prio、qos** または **markDscp**
- ルール `暗黙的` または `implarp` フィルタ
- ルール アクションで `Deny` 、 `Redir` 、 `コピー` 、または `Deny ログ`

次の月クエリ出力は、圧縮のと見なされる、契約の 2 つのルールを示します。

```
# actrl.Rule
scopeId             : 2588677
sPcTag              : 16388
dPcTag              : 49156
fltId               : 67
action              : no_stats,permit
```

```
actrlCfgFailedBmp  :
actrlCfgFailedTs   : 00:00:00:00.000
actrlCfgState      : 0
childAction        :
descr              :
direction          : bi-dir
dn                 : sys/actrl/scope-2588677/rule-2588677-s-16388-d-49156-f-67
id                 : 4112
lcOwn              : implicit
markDscp           : unspecified
modTs              : 2018-04-27T09:01:33.152-07:00
monPolDn           : uni/tn-common/monepg-default
name               :
nameAlias          :
operSt             : enabled
operStQual         :
prio               : fully_qual
qosGrp             : unspecified
rn                 : rule-2588677-s-16388-d-49156-f-67
status             :
type               : tenant


# actrl.Rule
scopeId            : 2588677
sPcTag             : 49156
dPcTag             : 16388
fltId              : 64
action             : no_stats,permit
actrlCfgFailedBmp  :
actrlCfgFailedTs   : 00:00:00:00.000
actrlCfgState      : 0
childAction        :
descr              :
direction          : uni-dir-ignore
dn                 : sys/actrl/scope-2588677/rule-2588677-s-49156-d-16388-f-64
id                 : 4126
lcOwn              : implicit
markDscp           : unspecified
modTs              : 2018-04-27T09:01:33.152-07:00
monPolDn           : uni/tn-common/monepg-default
name               :
nameAlias          :
operSt             : enabled
operStQual         :
prio               : fully_qual
qosGrp             : unspecified
rn                 : rule-2588677-s-49156-d-16388-f-64
status             :
type               : tenant
```

表 *1:* 圧縮マトリクス

リバース フィルタ ポートが有効	TCP または UDP 発信元ポート	TCP または UCP 宛先ポート	圧縮
Yes	ポート A	ポート B	Yes
Yes	未指定	ポート B	Yes
Yes	ポート A	未指定	Yes

リバース フィルタ ポートが有効	TCP または UDP 発信元ポート	TCP または UCP 宛先ポート	圧縮
Yes	未指定	未指定	Yes
No	ポート A	ポート B	No
No	未指定	ポート B	No
No	ポート A	未指定	No
No	未指定	未指定	Yes

GUI を使用して TCAM の使用が最適化された契約を設定する

この手順はで、ハーウェア上の TCAM による契約データの保存を最適化する契約を設定する方法について説明します。

始める前に

- 契約を提供および利用するテナント、VRF、および EPG を作成します。
- この契約で許可または拒否されるトラフィックを定義する、1 つ以上のフィルタを作成します。

ステップ 1 メニューバーで **Tenants** を選択し、実行するテナント名を選択します。**Navigation** ウィンドウで、*tenant-name* および **Contracts** を展開します。

ステップ 2 **Standard** > **Create Contract** を右クリックします。

ステップ 3 **Create Contract** ダイアログボックスで、次のタスクを実行します:

a) **Name** フィールドに、契約 名を入力します。

b) + アイコン (**Subjects** の隣にあるもの) をクリックして、新しい情報カテゴリを追加します。

c) [Create Contract Subject] ダイアログボックスで、[Name] フィールドにサブジェクト名を入力します。

（注） この手順では、フィルタを契約の情報カテゴリに関連付けます。

d) TCAM の契約し状況強最適化機能を有効にするには、**Apply Both Directions** および **Reverse Filter Ports** が有効になっていることを確認します。

e) + アイコンをクリックして **Filters** を展開します。

f) ダイアログボックスで、ドロップダウン メニューから、デフォルトのフィルタを指定します。すでに設定したフィルタを選択するか、**Create Filter** で新しいフィルタを作成します。

g) **Directives** フィールドで、**no stats** を選択します。

h) **Action** フィールドで、**Permit** または **Deny** を選択します。

（注） 現在のところ、**Deny** アクションはサポートされていません。最適化は **Permit** アクションに対してのみ行われます。

i) （任意） **Priority** フィールドで、優先度レベルを選択します。
j) **Update** をクリックします。

ステップ 4 **Create Contract Subject** ダイアログボックスで、**OK** をクリックします。

ステップ 5 **Create Contract** ダイアログボックスで、**Submit** をクリックします。

最適化された TCAM の使用率が REST API を使用すると、契約を設定します。

始める前に

テナント、VRF、およびを提供し、契約を消費する Epg を作成します。

フィルタとハードウェアの契約データの TCAM ストレージを最適化する契約を設定するには、例を次のような XML で post を送信します。

例：

```
<vzFilter dn="uni/tn-Tenant64/flt-webFilter" name="webFilter>
    <vzEntry applyToFrag="no" dFromPort="https" dToPort="https"
       dn="uni/tn-Tenant64/flt-webFilter/e-https" etherT="ip" name="https" prot="tcp" stateful="no"/>

    </vzFilter>
<vzBrCP dn="uni/tn-Tenant64/brc-OptimizedContract" name="OptimizedContract" provMatchT="AtleastOne"
 revFltPorts="yes">
    <vzSubj consMatchT="AtleastOne" dn="uni/tn-Tenant64/brc-OptimizedContract/subj-WebSubj"
lcOwn="local" name="WebSubj"
          provMatchT="AtleastOne" revFltPorts="yes">
         <vzRsSubjFiltAtt action="permit" directives="no_stats" forceResolve="yes" lcOwn="local"
tCl="vzFilter"
          tDn="uni/tn-Tenant64/flt-webFilter" tRn="flt-webFilter" tType="name"
tnVzFilterName="webFilter"/>
    </vzSubj>
</vzBrCP>
```

ポリシー圧縮

2 つの EPG 間で契約を作成する際にはフィルタ ルールを設定します。各フィルタ ルールは、契約対象の 2 つの EPG（プロバイダー EPG とコンシューマ EPG）、フィルタするトラフィック プロトコル、および送信元ポートと宛先ポートで構成されます。たとえば、TCP ポート 443 を指定する HTTPS というフィルタを使用して、2 つの EPG 間で契約を定義できます。この契約を使用して、Web サービスを提供する EPG とそのサービスを消費するエンドポイントを含む EPG 間のセキュアな Web トラフィックを許可できます。

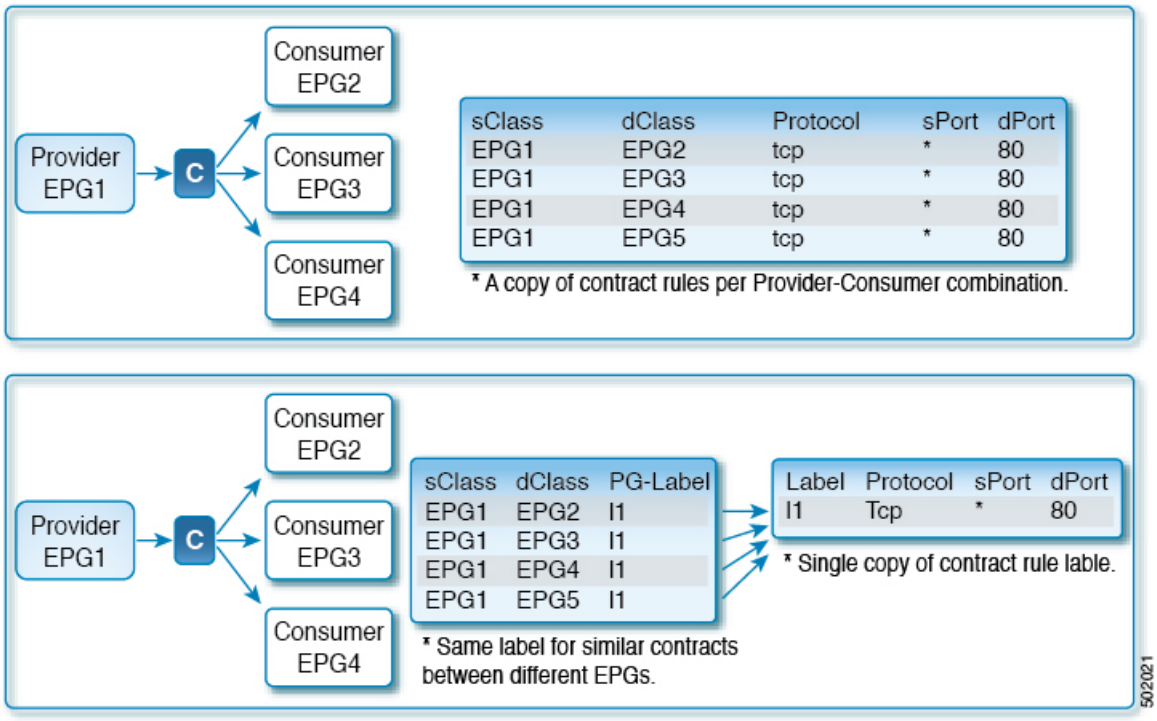
一般的な導入では、通常、多くの異なる EPG ペアで同じ契約が作成されます。たとえば多対 1 のシナリオでは、複数のコンシューマ EPG が同じ契約を使用して 1 つのプロバイダー EPG

にアクセスします。ただし、これらの契約にフィルタ ルールがそれぞれ指定されていると、APIC は該当するスイッチに、プロバイダー EPG、コンシューマ EPG、トラフィック プロトコル、およびスイッチ ポートの組み合わせごとに一意のフィルタ ルールを設定します。この結果、スイッチの TCAM テーブル エントリ数が増加し、スイッチの TCAM テーブルのサイズに基づいてファブリック内で使用できるフィルタ ルールの最大合計数が物理的に制限されます。

この物理的な制限を軽減するために APIC リリース 4.0(1) で追加されたポリシー圧縮機能により、プロバイダーおよびコンシューマ EPG の複数のペアに同一のフィルタ ルールが適用されている場合も 1 つの TCAM エントリを共有できます。ポリシー圧縮は、ポリシーと TCAM エントリの間でポリシー グループ（PG）ラベル経由の間接的な関連付けを作成することで実現されます。

次の図は、1 対多で PG ラベル経由の間接的な関連付けによって、TCAM エントリの合計数を減らし、ファブリック内の最大フィルタ ルール数を増やすシナリオ例を示しています。

図 *9*: ポリシー圧縮



（注） ポリシー リダイレクトを有効化すると、複数の契約で同じ TCAM テーブル エントリが共有されるようになるため、個々の契約統計情報の追跡が無効になることに注意してください。

ポリシー圧縮の注意点

ファブリック内の 1 つ以上の契約でポリシー圧縮を有効にする場合は、次の点を考慮してください。

- ポリシー圧縮を有効にすると、個々のフィルタ ルールの統計情報が無効になります。

  異なる契約の同じフィルタ ルールによってTCAMスペースが共有されることになるため、個々のフィルタ ルールの統計情報は保持されません。契約でポリシー圧縮を有効にすると、既存の統計情報は破棄されます。特定の契約の統計情報を保持する必要がある場合は、該当する契約でポリシー圧縮機能を無効にしておく必要があります。

- ポリシー圧縮は、Nexus 9300 FX 以降のプラットフォームでサポートされます。

- ポリシー圧縮は、ユーザ定義のルールに対してのみ有効にできます。

  Cisco APICによって作成された暗黙のルールでポリシー圧縮を有効にすることはできません。

- ポリシー圧縮は、`permit` および `permit-log` ルールでのみ機能します。

  `copy`、`redir`、`deny`、`deny-log` などの他のフィルタ ルールでは、設定が有効になっていても無視されます。

- ラベルおよびサブジェクト例外が関連付けられている契約では、ポリシー圧縮を有効にできません。

- ポリシー圧縮は、明示的に有効化されている契約にのみ適用されます。その他の契約で同じフィルタを使用する場合にこの機能を利用するには、その契約でポリシー圧縮を有効にする必要があります。

- ポリシー圧縮は、EPG ペアあたり 1 つの契約でのみ有効にできます。

- 送信元と宛先のクラスをラベルにマッピングするテーブルのサイズは、ポリシープロファイルによって異なります。具体的なスケールの情報については、『Verified Scalability Guide』を参照してください。

  このテーブルがいっぱいになると、追加された契約は非圧縮形式で保存されます。Cisco APIC GUI の **[Operations]** > **[Capacity Dashboard]** > **[Leaf Capacity]** 画面で現在の使用状況を表示できます。

- ポリシー圧縮を有効にしない限り、デフォルト TCAM スペースは減少しません。

  機能が有効になっている場合にのみ、TCAMスペースが動的に分割されます。機能を無効にする（ポリシー圧縮が有効になっているすべてのルールを削除する）と、TCAMスペースが戻されます。

- ポリシー圧縮機能と双方向契約機能の両方が有効な場合、圧縮ロジックにより、最大圧縮を提供する契約が選択されます。

  ポリシー圧縮は双方向契約より優先されますが、ラベルテーブルの枯渇またはルックアップの競合が発生した場合は、ロジックによって双方向契約にフォールバックされます。

- 多数のルールを使用するスイッチがリロードされた場合、スイッチが end-of-bootstrap 通知を受信するまでルールは圧縮されません。

- リリース 4.0(1) にアップグレードしてポリシー圧縮を有効化する場合：

- リリース 3.2(1) 以降からアップグレードする場合は、契約を削除して再設定する必要があります。
- 3.2(1) より前のリリースからアップグレードする場合は、契約およびフィルを削除して再設定する必要があります。

APIC GUI を使用したポリシー圧縮の有効化

Cisco ACI GUI を使用して、契約のポリシー圧縮を有効化できます。

ステップ 1 APIC にログインします。

ステップ 2 **[Tenant]** > **<テナント名>** > **[Contracts]** > **[Standard]** > **<コントラクト名>** > **<サブジェクト名>** に移動します。

ステップ 3 [Filter] テーブルで、ポリシー圧縮を有効にするフィルタをダブルクリックします。

ステップ 4 表示された [Filter] ダイアログ ウィンドウで、[Enable Policy Compression] チェックボックスをオンにします。

NX-OS スタイル CLI を使用したポリシー圧縮の有効化

NX-OS スタイル CLI を使用して、契約のポリシー圧縮を有効化できます。

ステップ 1 APIC コンフィギュレーション モードを開始します。

例：

```
apic1# config
```

ステップ 2 ポリシー圧縮オプションを設定します。

次の設定の *Tn17_comp*、*ctr4k_udp_tcp*、*sub_udp_4001-4100*、*udp4001-4100_14001-14100* をポリシー圧縮を設定するテナント、契約、サブジェクト、およびアクセス グループに置き換えます。

例：

```
apic1(config)# tenant Tn17_comp
apic1(config-tenant)# contract ctr4k_udp_tcp
apic1(config-tenant-contract)# subject sub_udp_4001-4100
apic1(config-tenant-contract)# access-group udp4001-4100_14001-14100 both log no-stats
apic1(config-tenant-contract)# exit
apic1(config-tenant)# exit
apic1(config)# exit
```

REST API を使用したポリシー圧縮の有効化

REST API を使用して、契約のポリシー圧縮を有効化できます。

ポリシー圧縮オプションを設定します。

次の設定の *ctr4k_udp_tcp*、*sub_udp_4001-4100*、*udp4001-4100_14001-14100* をポリシー圧縮を設定するテナント、契約、サブジェクト、およびアクセス グループに置き換えます。

POST URL：https://<*apic-ip*>/api/node/mo/uni.xml

例：

```
<vzBrCP name="ctr4k_udp_tcp" scope="global" targetDscp="unspecified">
    <vzSubj consMatchT="AtleastOne" name="sub_udp_4001-4100" provMatchT="AtleastOne"
            revFltPorts="yes" targetDscp="unspecified">
        <vzRsSubjFiltAtt directives="log,no_stats"  action="permit"
                         tnVzFilterName="udp4001-4100_14001-14100"/>
    </vzSubj>
</vzBrCP>
```

契約とサブジェクトの例外

コントラクトまたはコントラクトの件名の例外の設定

Cisco APIC リリース 3.2(1) では、EPG 間のコントラクトが拡張され、コントラクトに参加しているコントラクトプロバイダまたはコンシューマのサブネットを拒否できます。インターEPG コントラクトおよびい内部 EPG コントラクトは、この機能でサポートされます。

プロバイダ EPG の件名を有効にして、件名またがコントラクトの例外で一致基準が設定されているものを除くすべてのコンシューマ EPG との通信が可能になります。たとえば、サブセットを除く、テナントのすべての EPG にサービスを提供するために EPG を有効にする場合、これら EPG を除外できます。これを設定するには、コントラクトまたはそのコントラクトの件名のいずれかで例外を作成します。サブセットがコントラクトの提供または消費のアクセスを拒否します。

ラベル、カウンタ、許可および拒否ログは、コントラクトおよび件名の例外でサポートされています。

コントラクトのすべての件名に例外を適用するには、コントラクトに例外を追加します。コントラクトの単一の件名にのみ例外を適用する場合、件名に例外を追加します。

件名にフィルタを追加する場合、フィルタのアクションを設定できます（フィルタ条件に一致するオブジェクトを許可または拒否する）。また、**[拒否]**フィルタについては、フィルタの優先順位を設定することができます。**[許可]**フィルタは常にデフォルトの優先順位があります。自動拒否の件名-フィルタ関係をマーキングすると、件名に一致している場合、各 EPG のペア

に適用されます。コントラクトと件名には、複数の件名-フィルタ関係を含むことができます。これは、フィルタに一致するオブジェクトを許可または拒否するように独自に設定できます。

例外タイプ

コントラクトと件名の例外は次のタイプに基づき、* ワイルドカードなどの正規表現を含むことができます。

例外の条件は、[コンシューマ正規表現] および [プロバイダ正規表現]のフィールドで定義されているように、これらのオブジェクトを除外します。	例	説明
テナント	<vzException consRegex= “common” field= “Tenant” name= “excep03” provRegex= “t1” />	この例では、common テナントを使用して、 EPG が t1 テナントにより提供されるコントラクトを消費しないように除外します。
VRF	<vzException consRegex= “ctx1” field= “Ctx” name= “excep05” provRegex= “ctx1” />	この例では、ctx1 のメンバーが同じ VRF から提供されるサービスを使用しないように除外します。
EPG	<vzException consRegex= “EPgPa*” field= “EPg” name= “excep03” provRegex= “EPg03” />	この例では、名前が EPGPa から始まる複数の EPG が存在すると仮定し、EPg03 により提供されているコントラクトのコンシューマとしてすべて拒否される必要があります。
Dn	<vzException consRegex= “uni/tn-t36/ap-customer/epg-epg193” field= “Dn” name=“excep04” provRegex= “uni/tn-t36/ap-customer/epg-epg200” />	この例では、epg193 が epg200 により提供されたコントラクトを消費しないように除外します。
タグ	<vzException consRegex= “red” field= “Tag” name= “excep01” provRegex= “green” />	例では、red タグでマークされているオブジェクトが消費することと、green タグでマークされているオブジェクトがコントラクトに参加しないように除外します。

GUI を使用したコントラクトまたはサブジェクトの例外の設定

このタスクでは、EPGのほとんどに対して通信を許可するものの、その一部のアクセスは拒否するコントラクトを設定します。

始める前に

コントラクトを提供し、利用するために、テナント、VRF、アプリケーションプロファイルとEPG を設定します。

ステップ 1 メニュー バーで **[テナント]** > **[すべてテナント]** をクリックします。

ステップ 2 コントラクトを作成しているテナントをダブルクリックします。

ステップ 3 ナビゲーション バーで、**[コントラクト]** を展開し、**[フィルタ]** を右クリックして、**[フィルタの作成]** を選択します。

フィルタでは、コントラクト経由のアクセスを許可または拒否するトラフィックを定義するアクセス制御リスト（ACL）に重要です。許可または拒否できるオブジェクトを定義する複数のフィルタを作成することができます。

ステップ 4 フィルタ名を入力し、許可または拒否するトラフィックを定義する条件を追加して、**[送信]** をクリックします。

ステップ 5 **[コントラクト]** を右クリックし、**[コントラクトの作成]** を選択します。

ステップ 6 コントラクト名を入力し、範囲を設定して、[+] アイコンをクリックし件名を追加します。

ステップ 7 繰り返して別の件名を追加します。

ステップ 8 [Submit] をクリックします。

ステップ 9 コントラクトのすべての件名の例外を追加する手順は、次のとおりです。

a) コントラクトをクリックし、**[コントラクトの例外]** をクリックします。
b) 件名を追加し、許可または拒否するように設定します。
c) [+] アイコンをクリックしてコントラクトを追加します。
d) 例外の名前とタイプを入力します。
e) 正規表現を **[コンシューマ Regex]** および **[プロバイダ Regex]** フィールドに追加し、コントラクトのすべての件名から除外する EPG を定義します。

ステップ 10 コントラクトの 1 つの件名の例外を追加する手順は、次のとおりです。

a) 件名をクリックし、**[件名の例外]** をクリックします。
b) [+] アイコンをクリックしてコントラクトを追加します。
c) 例外の名前とタイプを入力します。
d) 正規表現を **[コンシューマ Regex]** および **[プロバイダ Regex]** に追加し、コントラクトのすべての件名から除外する EPG を定義します。

NX-OS スタイルの CLI を使用したコントラクトまたはコントラクトの件名除外の設定

このタスクでは、ほとんどの EPG の通信を許可するコントラクトを設定しますが、それらのサブネットへのアクセスを拒否します。コントラクトまたは件名には、複数の例外を追加できます。

始める前に

テナント、VRF、アプリケーション プロファイル、EPG を設定して、コントラクトを提供し消費します。

ステップ 1 次の例のようにコマンドを使用して、HTTP および HTTPS のフィルタを設定します。

例：

```
apic1(config)# tenant t2
apic1(config-tenant)# access-list ac1
apic1(config-tenant-acl)# match ip
apic1(config-tenant-acl)# match tcp dest 80
apic1(config-tenant-acl)# exit
apic1(config-tenant)# access-list ac2
apic1(config-tenant-acl)# match ip
apic1(config-tenant-acl)# match tcp dest 443
```

ステップ 2 `EPg01` の消費と `EPg03` の提供を除外するコントラクトを設定します。

例：

```
apic1(config-tenant)# contract webCtrct
apic1(config-tenant-contract)# subject https-subject
apic1(config-tenant-contract-subj)# exception name EPG consumer-regexp EPg01 field EPg provider-regexp
 EPg03
apic1(config-tenant-contract-subj)# access-group ac1 in blacklist
apic1(config-tenant-contract-subj)# access-group ac2 in whitelist
```

REST API を使用した契約またはサブジェクトの例外の設定

このタスクでは、ほとんどの EPG の通信を許可するコントラクトを設定しますが、それらのサブネットへのアクセスを拒否します。契約またはサブジェクトには、複数の例外を追加することができます。

始める前に

テナント、VRF、アプリケーション プロファイル、EPG を設定して、コントラクトを提供し消費します。

ステップ 1 次の例のような XML を POST 送信することによりフィルタを作成します:

例：

```
<vzFilter name='http-filter'>
    <vzEntry name='http-e' etherT='ip' prot='tcp'/>
    <vzEntry name='https-e' etherT='ip' prot='tcp'/>
</vzFilter>
```

ステップ 2 次の例のような XML を POST 送信することにより、サブジェクトを利用できないように `EPg01` を例外とし、提供できないように `EPg03` を例外とします:

vzException MO は、vzBrCP または vzSubj MO に含めることができます。

例：

```
<vzBrCP name="httpCtrct" scope="context">
    <vzSubj name="subj1"
        <vzException consRegex="EPg01" field="EPg" name="excep01" provRegex=EPg03"/>
    <vzSubj/>
         <vzRsSubjFiltAtt tnVzFilterName="http-filter" Action="deny"/>
         <vzRsSubjFiltAtt tnVzFilterName="https-filter" Action="permit"/>
     </vzSubj>
</vzBrCP>
```

EPG 内契約

EPG 内契約

EPG 間の通信を制御するには、契約を設定します。Cisco APIC リリース 3.0(1) 以降では、EPG 内の契約を設定できます。

EPG 内契約がない場合、EPG のエンドポイント間の通信は、完全に可能か不可能かになります。通信はデフォルトでは無制限ですが、エンドポイント間の通信を禁止するために、EPG内分離を設定することができます。

ただし、EPG 内契約を使用すれば、同じ EPG のエンドポイント間の通信を制御して、いくつかのトラフィックを許可し、残りの部分を禁止することができます。たとえば、Webトラフィックを許可し、残りの部分をブロックすることが必要な場合があるでしょう。または、すべての ICMP トラフィックと TCP ポート 22 のトラフィックを許可し、他のすべてのトラフィックをブロック中することができます。

EPG 内契約の注意事項と制約事項

EPG 内契約を計画する場合は、次の注意事項と制約事項に従ってください。

- EPG 内契約は、VMware VDS、Open vSwitch (OVS)、およびベアメタル サーバ上のアプリケーション EPG とマイクロセグメント EPG (uSeg) で設定できます。

（注） OVS は、Kubernetes integration with Cisco ACI で利用可能です。これは、シスコ アプリケーション セントリック インフラストラクチャ (ACI) とKubernetes の統合により使用できる機能です。Kubernetes では、EPG を作成し、それらに名前空間を割り当てることがことができます。VMware VDS またはベアメタルと同様、Cisco APIC では、EPG 内ポリシーを EPG に適用することができます。

- EPG 内契約では、リーフ スイッチがプロキシによる Address Resolution Protocol（ARP）をサポートしていることが必要です。これらは、モデル名の末尾に EX または FX が付いている Cisco Nexus 9000 シリーズ スイッチおよびさらに新しいモデルでサポートされています。
- EPG内契約は、AVS、AVE、およびMicrosoftドメインではサポートされていません。EPG内契約を設定してこれらのドメインに適用しようとすると、ポートがブロック状態になる可能性があります。
- サービス グラフでの EPG 内契約：
  - サービス グラフを拒否のアクションを含む EPG 内契約のサブジェクトと関連付けることはできません。
  - サービス グラフで EPG 内契約がサポートされるのは、シングル ノード ワンアーム モードのポリシーベース リダイレクトおよびコピー サービスに限られます。

GUI を使用したアプリケーション EPG への EPG 内契約の追加

コントラクトを設定した後、EPG内コントラクトとしてEPGにコントラクトを追加できます。この手順は、VMware VDS、OVS、およびベアメタル サーバと同じです。

始める前に

- アプリケーション EPG が設定済みである必要があります。
- このアプリケーション用のフィルタが設定された契約が必要です。「GUIを使用した契約の作成 （47 ページ）」を参照してください。

ステップ 1 APIC GUI にログインします。

ステップ 2 **[Tenants]** > **テナント**に移動します。

ステップ 3 EPG のタイプに応じて、次の一連の手順のいずれかを実行します。

EPG 内コントラクトに適用する場合：	結果
アプリケーション EPG	結果 1. 左のナビゲーション ペインで、**[アプリケーション プロファイル]** > ***application profile*** > **[アプリケーション EPG]** > ***epg*** を展開します。 2. **[コントラクト]** フォルダを右クリックして、**[EPG 内コントラクトの追加]** を選択します。 3. **[EPG 内コントラクトの追加]** ダイアログ ボックスで **[コントラクト]** ドロップダウン リストからコントラクトを選択します。 4. [Submit] をクリックします。`
USeg EPG	1. 左のナビゲーション ウィンドウで、**[アプリケーション プロファイル]** > ***application profile*** > **[uSeg EPG]** > ***epg***. を展開します。 2. **[コントラクト]** フォルダを右クリックして、**[EPG 内コントラクトの追加]** を選択します。 3. **[EPG 内コントラクトの追加]** ダイアログ ボックスで **[コントラクト]** ドロップダウン リストからコントラクトを選択します。 4. [Submit] をクリックします。`

NX-OS スタイルの CLI を使用した EPG 内契約の設定

契約を設定した後、内通EPG契約として、契約を設定できます。手順は、VMware VDS、OVS、およびベアメタル サーバの同じです。

始める前に

- 設定されている、EPG は必須です。
- フィルタを持つ契約を設定している必要があります。

ステップ 1 NX-OS CLI で、コンフィギュレーション モードで開始します。

例：

```
apic #
apic # configure
```

ステップ 2 テナントを選択します。

例：

```
apic (config) # tenant t001
```

ステップ 3 アプリケーション プロファイルを選択します。

例：

```
apic (config-tenant) application ap3
```

ステップ 4 EPG を選択します。

例：

```
apic (config-tenant-app) epg ep3
```

ステップ 5 EPG の内通 EPG 契約を設定します。

例：

```
apic (config-tenant-app-epg) contract intra-epg ct1
```

REST API を使用した EPG 内契約の設定

契約を設定した後、内通EPG契約として、契約を設定できます。手順はVMwareVDS、OVS、およびベアメタル サーバで同じです。

始める前に

- 設定されている、EPG は必須です。
- フィルタを持つ契約を設定している必要があります。

XML POST 要求を使用して EPG 内契約を設定する方法は、次の例と似ています:

例：

```
<fvTenant name="t001">
  <fvAp name="ap3">
    <fvAEPg name="ep3">
      <fvRsIntraEpg tnVzBrCPName="ct1"/>
    </fvAEPg>
  </fvAp>
</fvTenant>
```

EPG のコントラクト継承

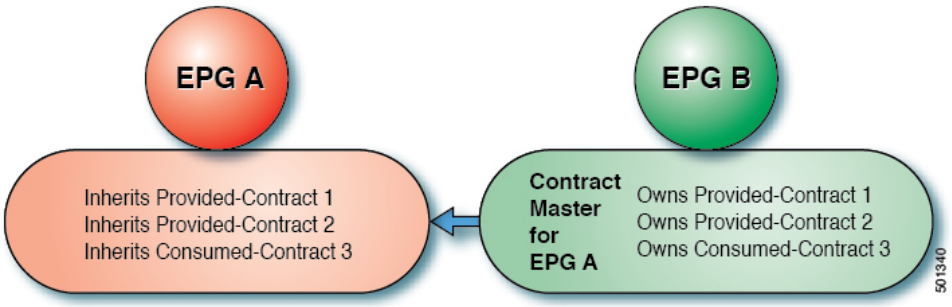
コントラクト継承について

関連する契約を新しい EPG に統合するため、EPG を有効にして同じテナントの別の EPG に直接関連する契約すべて（提供済み/消費済み）を継承できます。コントラクトの継承は、アプリケーション EPG、マイクロセグメント EPG、L2Out EPG、および L3Out EPG に設定できます。

リリース 3.x では、EPG 間の提供済み/消費済みの両方の契約に、契約を継承する設定も可能です。EPG 間契約が、モデル名や後発のモデルの最後に EX または EX が付く、Cisco Nexus 9000 シリーズ スイッチでサポートされています。

EPG を有効にし、APIC GUI、NX-OS スタイル CLI、REST API を使用して、別の EPG に直接関連する契約すべてを継承できます。

図 *10 :* コントラクトの継承



上の図で、EPG A は EPG B から（EPG A の契約マスター）提供済みの契約 1 および 2、消費済みの契約 3 を継承するように設定されています。

コントラクト継承を設定する際は、次のガイドラインに従ってください。

- コントラクト継承は、アプリケーション EPG、マイクロセグメント（uSeg）EPG、外部 L2Out EPG、および外部 L3Out EPG 用に設定できます。コントラクト関係は同じタイプの EPG 間で確立する必要があります。
- 関係が確立されると、提供するコントラクトと消費するコントラクトの両方がコントラクト マスターから継承されます。
- コントラクト マスターとコントラクトを継承する EPG は同じテナント内にある必要があります。
- マスター契約への変更は、すべての継承に伝播されます。新しい契約がマスターに追加される場合、継承先にも追加されます。
- EPG は、複数のコントラクト マスターからコントラクトを継承することができます。

- コントラクト継承は単一のレベルでのみサポートされ（連結できない）、コントラクトマスターがコントラクトを継承することはできません。
- コントラクト サブジェクト ラベルおよび EPG ラベルの継承がサポートされています。EPG A が EPG B から契約を継承する場合、EPG A と EPG B で異なるサブジェクト ラベルが設定されていると、APIC は EPG B で設定されているサブジェクト ラベルのみを使用し、どちらの EPG からもラベルを収集しません。
- EPGが契約に直接関連付けられている、または契約を継承しているかどうかに関わらず、TCAM 内のエントリが消費されます。したがって契約スケール ガイドラインが引き続き適用されます。詳細については、お使いのリリースの「検証されたスケーラビリティ ガイド」を参照してください。
- vzAny セキュリティ コントラクトとタブー コントラクはサポートされません。

契約の継承設定および継承済みおよびスタンドアロン契約を表示することに関する詳細は、「*Cisco APIC* の基本設定ガイドを参照してください。

GUI を使用した EPG のコントラクト継承の設定

GUI を使用したアプリケーション EPG のコントラクト継承の設定

アプリケーション EPG のコントラクト継承を設定するには、APIC の基本または拡張モード GUI で次の手順を使用します。

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

オプション。コントラクトを継承する EPG が使用するブリッジ ドメインを設定します。

EPG コントラクト マスターとして機能するように、少なくとも 1 つのアプリケーション EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

ステップ 1 **[Tenants]** > ***[tenant-name]*** > **[Application Profiles]** に移動して、[AP-name] を展開します。

ステップ 2 [Application EPGs] を右クリックし、[Create Application EPG] を選択します。

ステップ 3 **EPG コントラクト マスター**からコントラクトを継承する EPG の名前を入力します。

ステップ 4 [Bridge Domain] フィールドで、共通/デフォルトのブリッジ ドメインまたは以前に作成したブリッジ ドメインを選択するか、この EPG のブリッジ ドメインを作成します。

ステップ 5 [EPG Contract Master] フィールドで、+ 記号をクリックして事前に設定したアプリケーション プロファイルと EPG を選択し、[Update] をクリックします。

ステップ 6 [Finish] をクリックします。

ステップ 7 EPGに関する情報（コントラクトマスターなど）を表示するには、**[Tenants]** > ***[tenant-name]*** > **[Application Profiles]** > ***[AP-name]*** > **[Application EPGs]** > ***[EPG-name]*** に移動します。**EPG コントラクト マスター**を表示するには、[General] をクリックします。

ステップ 8 継承されるコントラクトに関する情報を表示するには、[EPG-name] を展開して [Contracts] をクリックします。

GUI を使用した uSeg EPG のコントラクト継承の設定

uSeg EPG のコントラクト継承を設定するには、APIC の基本または拡張モード GUI で次の手順を使用します。

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

オプション。コントラクトを継承する EPG が使用するブリッジ ドメインを設定します。

EPG コントラクト マスターとして機能するように uSeg EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

ステップ 1 **[Tenants]** > ***[tenant-name]*** > **[Application Profiles]** に移動して、[AP-name] を展開します。

ステップ 2 [uSeg EPGs] を右クリックし、[Create uSeg EPG] を選択します。

ステップ 3 コントラクト マスターからコントラクトを継承する EPG の名前を入力します。

ステップ 4 [Bridge Domain] フィールドで、共通/デフォルトのブリッジ ドメインまたは以前に作成したブリッジ ドメインを選択するか、この EPG のブリッジ ドメインを作成します。

ステップ 5 [uSeg-EPG-name] をクリックします。[EPG Contract Master] フィールドで、+ 記号をクリックしてアプリケーション プロファイルと EPG（コントラクト マスターとして機能する）を選択し、[Update] をクリックします。

ステップ 6 [Finish] をクリックします。

ステップ 7 契約に関する情報を表示するには、**[Tenants]** > **テナント名** > **[Application Profiles]** > ***AP* 名** > **[uSeg EPGs]** > に移動し、EPG 名を展開して [Contracts] をクリックします。.

GUI を使用した L2Out EPG のコントラクト継承の設定

外部 L2Out EPG のコントラクト継承を設定するには、APIC の拡張モード GUI で次の手順を使用します。

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

L2Out コントラクト マスターとして機能する外部ブリッジ型ネットワーク（L2Out）および外部ネットワーク インスタンス プロファイル（L2extInstP）を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

ステップ 1 外部 L2Out EPG のコントラクト継承を設定するには、**[Tenants]** > ***[tenant-name]*** > **[Networking]** > **[External Bridged Networks]** に移動し、次の手順を実行します。

ステップ 2 [L2Out-name] を展開します。

ステップ 3 [Networks] を右クリックして、[Create External Network] を選択します。

ステップ 4 外部ネットワークの名前を入力し、必要に応じてその他の属性を追加します。

ステップ 5 **Submit** をクリックします。

ステップ 6 [Networks] を展開します。

ステップ 7 [network-name] をクリックします。

ステップ 8 [External Network Instance Profile] パネルで、[L2Out Contract Masters] フィールドの + 記号をクリックします。

ステップ 9 この外部 L2Out EPG の L2Out および L2Out コントラクト マスターを選択します。

ステップ 10 [Update] をクリックします。

ステップ 11 この外部 L2Out EPG が継承するコントラクトを表示するには、外部ネットワーク インスタンス プロファイル名をクリックし、**[Contracts]** > **[Inherited Contracts]** をクリックします。

拡張 GUI を使用した外部 L3Out EPG のコントラクト継承の設定

外部 L3Out EPG のコントラクト継承を設定するには、APIC の拡張モード GUI で次の手順を使用します。

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

L3Out コントラクト マスターとして機能する外部ルーテッド ネットワーク（L3Out）および外部ネットワーク インスタンス プロファイル（L3extInstP）を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

ステップ 1 外部 L3Out EPG のコントラクト継承を設定するには、**[Tenants]** > ***[tenant-name]*** > **[Networking]** > **[External Routed Networks]** に移動し、次の手順を実行します。

ステップ 2 外部 L3Out EPG につながる [L3Out-name] を展開します。

ステップ 3 [Networks] を右クリックして、[Create External Network] を選択します。

ステップ 4 外部ネットワークの名前を入力し、必要に応じてサブネットとその他の属性を追加します。

ステップ 5 **Submit** をクリックします。

ステップ 6 [Networks] を展開します。

ステップ 7 [network-name] をクリックします。

ステップ 8 [External Network Instance Profile] パネルで、[L3Out Contract Masters] フィールドの + 記号をクリックします。

ステップ 9 この外部 L3Out EPG の L3Out コントラクト マスターとして機能する L3Out およびインターフェイス プロファイルを選択します。

ステップ 10 [Update] をクリックします。

ステップ 11 この外部 L3Out EPG が継承するコントラクトを表示するには、外部ネットワーク インスタンス プロファイル名をクリックし、**[Contracts]** > **[Inherited Contracts]** をクリックします。

NX-OS スタイルの CLI を使用したコントラクト継承の設定

NX-OS スタイルの CLI を使用したアプリケーションまたは uSeg EPG のコントラクト継承の設定

アプリケーション EPG または uSeg EPG のコントラクト継承を設定するには、次のコマンドを使用します。

始める前に

EPG が使用するテナント、アプリケーション プロファイル、およびブリッジ ドメインを設定します。

VRF レベルで EPG が共有するコントラクトを設定します。

手順

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `apic1# configure`	コンフィギュレーション モードに入ります。
ステップ 2	**tenant** *tenant-name* 例： `apic1# (config) tenant Tn1`	設定するテナントを作成または指定し、テナント コンフィギュレーション モードを開始します。
ステップ 3	**application** *application-name* 例： `apic1(config-tenant)# application AP1`	アプリケーションを作成または指定し、アプリケーション モードを開始します。
ステップ 4	**epg** *epg-name* [**type micro-segmented**] 例： `apic1(config-tenant-app)# epg AEPg403`	設定するアプリケーション EPG または uSeg EPG を作成または指定し、EPG コンフィギュレーション モードを開始します。uSeg EPG の場合はタイプを追加します。

	コマンドまたはアクション	目的
		この例では、アプリケーション EPG のコントラクト マスターです。
ステップ 5	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに EPG を関連付けます。
ステップ 6	**contract consumer** *contract-name* 例： `apic1(config-tenant-app-epg)# contract consumer cctr5`	この EPG が消費するコントラクトを追加します。
ステップ 7	**contract provider** [**label** *label*] 例： `apic1(config-tenant-app-epg)# contract provider T1ctrl_cif`	サブジェクトまたは EPG ラベルのオプション リストなど（事前に設定済みである必要があります）、この EPG が提供するコントラクトを追加します。
ステップ 8	**exit** 例： `apic1(config-tenant-app-epg)# exit`	コンフィギュレーション モードを終了します。
ステップ 9	**epg** *epg-name* [**type micro-segmented**] 例： `apic1(config-tenant-app)# epg AEPg404`	設定するアプリケーション EPG または uSeg EPG を作成または指定し、EPG コンフィギュレーション モードを開始します。uSeg EPG の場合はタイプを追加します。 この例では、コントラクトを継承する EPG です。
ステップ 10	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに EPG を関連付けます。
ステップ 11	**inherit-from-epg application** *application-name***epg** *EPG-contract-master-name*] 例： `apic1(config-tenant-app-epg)# inherit-from-epg application AP1 epg AEPg403`	この EPG が EPG コントラクト マスターからコントラクトを継承するように設定します。
ステップ 12	**exit** 例： `apic1(config-tenant-app-epg)# exit`	コンフィギュレーション モードを終了します。

	コマンドまたはアクション	目的
ステップ 13	**epg** *epg-name* [**type micro-segmented**] 例： `apic1(config-tenant-app)# epg uSeg1_403_10 type micro-segmented`	設定するアプリケーション EPG または uSeg EPG を作成または指定し、EPG コンフィギュレーション モードを開始します。 この例では、uSeg EPG のコントラクト マスターです。
ステップ 14	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに EPG を関連付けます。
ステップ 15	**contract provider** [**label** *label*] 例： `apic1(config-tenant-app-epg)# contract provider T1ctrl_uSeg_l3out`	サブジェクトまたは EPG ラベルのオプション リストなど（事前に設定済みである必要があります）、この EPG が提供するコントラクトを追加します。
ステップ 16	**attribute-logical-expression** *logical-expression* 例： `apic1(config-tenant-app-epg)# attribute-logical-expression 'ip equals 192.168.103.10 force'`	一致基準として論理式を uSeg EPG に追加します。
ステップ 17	**exit** 例： `apic1(config-tenant-app-epg)# exit`	コンフィギュレーション モードを終了します。
ステップ 18	**epg** *epg-name* [**type micro-segmented**] 例： `apic1(config-tenant-app)# epg uSeg1_403_30 type micro-segmented`	設定するアプリケーション EPG または uSeg EPG を作成または指定し、EPG コンフィギュレーション モードを開始します。 この例では、EPG コントラクト マスターからコントラクトを継承する uSeg EPG です。
ステップ 19	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに EPG を関連付けます。
ステップ 20	**attribute-logical-expression** *logical-expression* 例： `apic1(config-tenant-app-epg)# attribute-logical-expression 'ip equals 192.168.103.30 force'`	基準として論理式を uSeg EPG に追加します。

	コマンドまたはアクション	目的
ステップ 21	**inherit-from-epg application** *application-name* **epg** *EPG-contract-master-name* 例： `apic1(config-tenant-app-epg)# inherit-from-epg application AP1 epg uSeg1_403_10`	この EPG が EPG コントラクト マスターからコントラクトを継承するように設定します。
ステップ 22	**exit** 例： `apic1(config-tenant-app-epg)# exit`	コンフィギュレーション モードを終了します。
ステップ 23	**exit** 例： `apic1(config-tenant-app)# exit`	コンフィギュレーション モードを終了します。
ステップ 24	**exit** 例： `apic1(config-tenant)# exit`	コンフィギュレーション モードを終了します。
ステップ 25	**exit** 例： `apic1(config)# exit`	コンフィギュレーション モードを終了します。

例

```
ifav90-ifc1# show running-config tenant Tn1 application AP1
# Command: show running-config tenant Tn1 application AP1
# Time: Fri Apr 28 17:28:32 2017
  tenant Tn1
    application AP1
      epg AEPg403
        bridge-domain member T1BD1
        contract consumer cctr5 imported
        contract provider T1ctr1_cif
        exit
      epg AEPg404
        bridge-domain member T1BD1
        inherit-from-epg application AP1 epg AEPg403
        exit
      epg uSeg1_403_10 type micro-segmented
        bridge-domain member T1BD1
        contract provider T1Ctr1_uSeg_l3out
        attribute-logical-expression 'ip equals 192.168.103.10 force'
        exit
      epg uSeg1_403_30 type micro-segmented
        bridge-domain member T1BD1
        attribute-logical-expression 'ip equals 192.168.103.30 force'
        inherit-from-epg application AP1 epg uSeg1_403_10
        exit
      exit
    exit
```

NX-OS スタイルの CLI を使用した L2Out EPG のコントラクト継承の設定

外部 L2Out EPG のコントラクト継承を設定するには、次のコマンドを使用します。

始める前に

EPG が使用するテナント、VRF、およびブリッジ ドメインを設定します。

EPG が使用するレイヤ 2 外部ネットワーク（L2Out）を設定します。

VRF レベルで EPG が共有するコントラクトを設定します。

手順の概要

1. **configure**
2. **tenant** *tenant-name*
3. **external-l2 epg** *external-l2-epg-name*
4. **bridge-domain member** *bd-name*
5. **contract provider** *contract-name* [**label** *label*]
6. **exit**
7. **external-l2 epg** *external-l2-epg-name*
8. **bridge-domain member** *bd-name*
9. **inherit-from-epg** *L2Out-contract-master-name*
10. **exit**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `apic1# configure`	コンフィギュレーション モードに入ります。
ステップ 2	**tenant** *tenant-name* 例： `apic1(config)# tenant Tn1`	設定するテナントを作成または指定し、テナント コンフィギュレーション モードを開始します。
ステップ 3	**external-l2 epg** *external-l2-epg-name* 例： `apic1(config-tenant)# external-l2 epg l2out1:l2Ext1`	外部 L2Out EPG を設定または指定します。この例では、L2out コントラクト マスターです。
ステップ 4	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-l2ext-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに L2Out EPG を関連付けます。

	コマンドまたはアクション	目的
ステップ 5	**contract provider** *contract-name* [**label** *label*] 例： `apic1(config-tenant-l2ext-epg)# contract provider T1ctr_tcp`	この EPG が提供するコントラクトを追加します。
ステップ 6	**exit** 例： `apic1(config-tenant-l2ext-epg)# exit`	コンフィギュレーション モードを終了します。
ステップ 7	**external-l2 epg** *external-l2-epg-name* 例： `apic1(config-tenant)# external-l2 epg L2out12:l2Ext12`	外部 L2Out EPG を設定します。この例では、L2out コントラクト マスターからコントラクトを継承する EPG です。
ステップ 8	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-l2ext-epg)# bridge-domain member T1BD1`	ブリッジ ドメインに L2out EPG を関連付けます。
ステップ 9	**inherit-from-epg** *L2Out-contract-master-name* 例： `apic1(config-tenant-l2ext-epg)# inherit-from-epg epg l2out1:l2Ext1`	この EPG が L2Out コントラクト マスターからコントラクトを継承するように設定します。
ステップ 10	**exit** 例： `apic1(config-tenant-l2ext-epg)# exit`	コンフィギュレーション モードを終了します。

例

上記の手順は次の例からの抜粋です。

```
apic1# show running-config tenant Tn1 external-l2
# Command: show running-config tenant Tn1 external-l2
# Time: Thu May 11 13:10:14 2017
  tenant Tn1
    external-l2 epg l2out1:l2Ext1
      bridge-domain member T1BD1
      contract provider T1ctr_tcp
      exit
    external-l2 epg l2out10:l2Ext10
      bridge-domain member T1BD10
      contract provider T1ctr_tcp
      exit
    external-l2 epg l2out11:l2Ext11
      bridge-domain member T1BD11
      contract provider T1ctr_udp
      exit
    external-l2 epg l2out12:l2Ext12
```

```
  bridge-domain member T1BD12
  inherit-from-epg epg l2out1:l2Ext1
  inherit-from-epg epg l2out10:l2Ext10
  inherit-from-epg epg l2out11:l2Ext11
  inherit-from-epg epg l2out2:l2Ext2
  inherit-from-epg epg l2out3:l2Ext3
  inherit-from-epg epg l2out4:l2Ext4
  inherit-from-epg epg l2out5:l2Ext5
  inherit-from-epg epg l2out6:l2Ext6
  inherit-from-epg epg l2out7:l2Ext7
  inherit-from-epg epg l2out8:l2Ext8
  inherit-from-epg epg l2out9:l2Ext9
  exit
external-l2 epg l2out2:l2Ext2
  bridge-domain member T1BD2
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out3:l2Ext3
  bridge-domain member T1BD3
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out4:l2Ext4
  bridge-domain member T1BD4
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out5:l2Ext5
  bridge-domain member T1BD5
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out6:l2Ext6
  bridge-domain member T1BD6
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out7:l2Ext7
  bridge-domain member T1BD7
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out8:l2Ext8
  bridge-domain member T1BD8
  contract provider T1ctr_tcp
  exit
external-l2 epg l2out9:l2Ext9
  bridge-domain member T1BD9
  contract provider T1ctr_tcp
  exit
exit
```

NX-OS スタイルの CLI を使用した外部 L3Out EPG のコントラクト継承の設定

外部 L3Out EPG のコントラクト継承を設定するには、次のコマンドを使用します。

始める前に

EPG が使用するテナント、VRF、およびブリッジ ドメインを設定します。

EPG が使用するレイヤ 3 外部ネットワーク（L3Out）を設定します。

VRF レベルで EPG が共有するコントラクトを設定します。

手順

	コマンドまたはアクション	目的
ステップ 1	**configure** 例： `apic1# configure`	コンフィギュレーション モードに入ります。
ステップ 2	**tenant** *tenant-name* 例： `apic1(config)# tenant Tn1`	設定するテナントを作成または指定し、テナント コンフィギュレーション モードを開始します。
ステップ 3	**external-l3 epg** *external-l3-epg-name* **l3out** *l3out-name* 例： `apic1(config-tenant-app)# external-l3 epg l3Ext108 l3out T1L3out1`	外部 L3Out EPG を設定します。この例では、L3out コントラクト マスターです。
ステップ 4	**vrf member** *vrf-name* 例： `apic1(tenant-l3out)# vrf member T1ctx1`	L3out を VRF に関連付けます。
ステップ 5	**match ip** *ip-address-and-mask* 例： `apic1(config-tenant-l3ext-epg)# match ip 192.168.110.0/24 shared`	EPG の一部としてホストを識別するサブネットを追加し、そのサブネットのオプションの共有範囲を追加します。
ステップ 6	**contract provider** *contract-name* [**label** *label*] 例： `apic1(config-tenant-l3ext-epg)# contract provider T1ctrl-L3out`	この EPG が提供するコントラクトを追加します。
ステップ 7	**exit** 例： `apic1(config-tenant-l3ext-epg)# exit`	コンフィギュレーション モードを終了します。
ステップ 8	**external-l3 epg** *external-l3-epg-name* **l3out** *l3out-name* 例： `apic1(config-tenant-app)# external-l3 epg l3Ext110 l3out T1L3out1`	外部 L3Out EPG を設定します。この例では、L3out コントラクト マスターからコントラクトを継承する EPG です。
ステップ 9	**vrf member** *vrf-name* 例： `apic1(tenant-l3out)# vrf member T1ctx1`	L3out を VRF に関連付けます。

	コマンドまたはアクション	目的
ステップ 10	**match ip** *ip-address-and-mask* 例： `apic1(config-tenant-l3ext-epg)# match ip 192.168.112.0/24 shared`	EPG の一部としてホストを識別するサブネットを追加し、そのサブネットのオプションの共有範囲を追加します。
ステップ 11	**inherit-from-epg** *L3Out-contract-master-name* 例： `apic1(config-tenant-l3ext-epg)# inherit-from-epg l3Ext108`	この EPG が L3Out コントラクト マスターからコントラクトを継承するように設定します。
ステップ 12	**exit** 例： `apic1(config-tenant-l3ext-epg)# exit`	コンフィギュレーション モードを終了します。

例

```
ifav90-ifc1# show running-config tenant Tn1 external-l3 epg l3Ext110
# Command: show running-config tenant Tn1 external-l3 epg l3Ext110
# Time: Fri Apr 28 17:36:15 2017
  tenant Tn1
    external-l3 epg l3Ext108 l3out T1L3out1
      vrf member T1ctx1
      match ip 192.168.110.0/24 shared
      contract provider T1ctrl-L3out
      exit
    external-l3 epg l3Ext110 l3out T1L3out1
      vrf member T1ctx1
      match ip 192.168.112.0/24 shared
      inherit-from-epg epg l3Ext108
      exit
    exit
```

REST API を使用した EPG のコントラクト継承の設定

REST API を使用したアプリケーション EPG のコントラクト継承の設定

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

EPG コントラクト マスターとして機能するようにアプリケーション EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

REST API を使用してコントラクト継承を設定するには、コントラクトを継承する EPG に転送される URL を指定して、次の XML および JSON の例のような XML を POST 送信します。

例：

XML の例

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- /api/node/mo/uni/tn-coke/ap-AP/epg-EPg_B.xml -->
<polUni>

  <fvEPg>

    <fvRsSecInherited tDn="uni/tn-coke/ap-AP/epg-EPg_B/>
  </fvEPg>
</polUni>
```

JSON の例

```
https://192.168.200.10/api/node/mo/uni/tn-coke/ap-AP/epg-EPg_B.json
fvAEPg":{"attributes":{
   "dn":"uni/tn-coke/ap-AP/epg-EPg_B","name":"EPg_C",
   "rn":"epg-EPg_C",
   "status":"created"},
   "children":[{
     "fvRsBd":{"attributes":{
        "tnFvBDName":"default",
        "status":"created,modified"},
        "children":[]}},{
     "fvRsSecInherited":{"attributes":{
        "tDn":"uni/tn-coke/ap-AP/epg-EPg_B",
        "status":"created"},
        "children":[]}}]}}
```

REST API を使用した uSeg EPG のコントラクト継承の設定

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

EPG コントラクト マスターとして機能するようにアプリケーション EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

REST API を使用して uSeg のコントラクト継承を設定するには、次の例のような XML を POST 送信します。

例：

```
<polUni>
    <fvTenant name="Tn1" >
        <fvAEPg descr="" dn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_120" fwdCtrl="" isAttrBasedEPg="yes"
matchT="AtleastOne" name="uSeg1_301_120" pcEnfPref="unenforced" prefGrMemb="exclude"
prio="unspecified">
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_100" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_110" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_50" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_60" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_30" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_10" />
```

```
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_40" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_70" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_90" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_20" />
            <fvRsSecInherited tDn="uni/tn-Tn1/ap-AP1/epg-uSeg1_301_80" />
            <fvRsNodeAtt descr="" encap="unknown" instrImedcy="immediate" mode="regular"
tDn="topology/pod-1/node-108" />
            <fvRsNodeAtt descr="" encap="unknown" instrImedcy="immediate" mode="regular"
tDn="topology/pod-1/node-109" />
            <fvRsDomAtt classPref="encap" delimiter="" encap="vlan-301" encapMode="auto"
instrImedcy="immediate" netflowPref="disabled" primaryEncap="unknown" resImedcy="immediate"
tDn="uni/phys-PhysDom1" />
            <fvRsCustQosPol tnQosCustomPolName="" />
            <fvRsBd tnFvBDName="T1BD21" />
            <fvCrtrn descr="" match="any" name="default" nameAlias="" ownerKey="" ownerTag=""
prec="0">
                <fvIpAttr descr="" ip="192.14.1.120" name="0" nameAlias="" ownerKey="" ownerTag=""
 usefvSubnet="no" />
            </fvCrtrn>
        </fvAEPg>
    </fvTenant>
</polUni>
```

次のタスク

REST API を使用した L2Out EPG のコントラクト継承の設定

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

L2Out コントラクト マスターとして機能するように L2Out EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

REST API を使用して L2Out EPG のコントラクト継承を設定するには、次の例のような XML を POST 送信します。

例：

```
<polUni>
  <fvTenant name="Tn1" >
     <l2extOut name="l2out1">
        <l2extRsEBd encap="vlan-51" tnFvBDName="T1BD1" />
        <l2extRsL2DomAtt tDn="uni/l2dom-l2Dom1" />
        <l2extLNodeP name="default" >
           <l2extLIfP name="default" >
              <l2extRsPathL2OutAtt tDn="topology/pod-1/protpaths-108-109/pathep-[VPC83]" />
           </l2extLIfP>
        </l2extLNodeP>
        <l2extInstP matchT="AtleastOne" name="l2Ext1">
          <fvSubnet ctrl="nd" ip="192.13.1.10/24" preferred="no" scope="public,shared" virtual="no"
 />
           <fvRsProv tnVzBrCPName="T1ctr_tcp" />
        </l2extInstP>
     </l2extOut>
```

```
    <l2extOut name="l2out2">
       <l2extRsEBd encap="vlan-53" tnFvBDName="T1BD3" />
       <l2extRsL2DomAtt tDn="uni/l2dom-l2Dom1" />
       <l2extLNodeP name="default" >
          <l2extLIfP name="default" >
             <l2extRsPathL2OutAtt tDn="topology/pod-1/protpaths-108-109/pathep-[VPC84]" />
          </l2extLIfP>
       </l2extLNodeP>
       <l2extInstP matchT="AtleastOne" name="l2Ext3" prefGrMemb="exclude">
         <fvSubnet ctrl="nd" ip="192.13.2.10/24" preferred="no" scope="public,shared" virtual="no"
 />
          <fvRsSecInherited tDn="uni/tn-Tn1/l2out-l2out1/instP-l2Ext1" />
       </l2extInstP>
    </l2extOut>

  </fvTenant>
</polUni>
```

REST API を使用した L3Out EPG のコントラクト継承の設定

始める前に

EPG が使用するテナントとアプリケーション プロファイルを設定します。

L3Out コントラクト マスターとして機能するように L3Out EPG を設定します。

共有するコントラクトを設定し、コントラクト マスターに関連付けます。

REST API を使用して L3Out EPG のコントラクト継承を設定するには、次の例のような XML を POST 送信します。

例：

```
<polUni>
  <fvTenant name="Tn6" >

  <!— L3out creation -->
     <ospfIfPol deadIntvl="40" helloIntvl="10" name="ospf1" pfxSuppress="inherit" prio="1"
rexmitIntvl="5" xmitDelay="1" />
     <l3extOut enforceRtctrl="export" name="T6L3out821">
        <ospfExtP areaCost="1" areaCtrl="redistribute,summary" areaId="0.0.0.1" areaType="regular"
 />
        <l3extRsL3DomAtt tDn="uni/l3dom-L3Dom1" />
        <l3extRsEctx tnFvCtxName="T6ctx21" />
        <l3extLNodeP name="l3out_vpc82_prof" >
           <l3extRsNodeL3OutAtt rtrId="1.1.1.8" rtrIdLoopBack="yes" tDn="topology/pod-1/node-108">
              <l3extInfraNodeP fabricExtCtrlPeering="no" />
           </l3extRsNodeL3OutAtt>
           <l3extRsNodeL3OutAtt rtrId="1.1.1.9" rtrIdLoopBack="yes" tDn="topology/pod-1/node-109">
              <l3extInfraNodeP fabricExtCtrlPeering="no" />
           </l3extRsNodeL3OutAtt>
           <l3extLIfP name="ospf1" >
              <ospfIfP authKeyId="1" authType="none" >
                 <ospfRsIfPol tnOspfIfPolName="ospf1" />
              </ospfIfP>
              <l3extRsPathL3OutAtt encap="vlan-551" ifInstT="ext-svi" mode="regular" mtu="1500"
tDn="topology/pod-1/protpaths-108-109/pathep-[VPC82]" >
                 <l3extMember addr="192.16.51.1/24" llAddr="0.0.0.0" side="B" />
```

```
                <l3extMember addr="192.16.51.2/24" llAddr="0.0.0.0" side="A" />
              </l3extRsPathL3OutAtt>
              <l3extRsNdIfPol tnNdIfPolName="" />
            </l3extLIfP>
          </l3extLNodeP>

          <l3extInstP matchT="AtleastOne" name="T6l3Ext821">
                <fvRsProv tnVzBrCPName="T6ctr_UDP_TCP2" />
                <fvRsCons tnVzBrCPName="T6ctr_UDP_TCP1" />
            <l3extSubnet ip="192.16.51.0/24" scope="import-security,shared-rtctrl,shared-security"
/>
              <l3extSubnet ip="192.16.61.0/24" scope="import-security,shared-rtctrl,shared-security"
 />
                <vzConsSubjLbl name="tcp" tag="green" />
            <vzProvSubjLbl name="tcp" tag="green" />
          </l3extInstP>

          <l3extInstP matchT="AtleastOne" name="T6l3Ext823">
            <fvRsSecInherited tDn="uni/tn-Tn6/out-T6L3out821/instP-T6l3Ext821" />
              <l3extSubnet ip="192.16.63.0/24" scope="import-security,shared-rtctrl,shared-security"
 />
          </l3extInstP>
      </l3extOut>

   </fvTenant>
</polUni>
```

優先グループ契約

契約優先グループについて

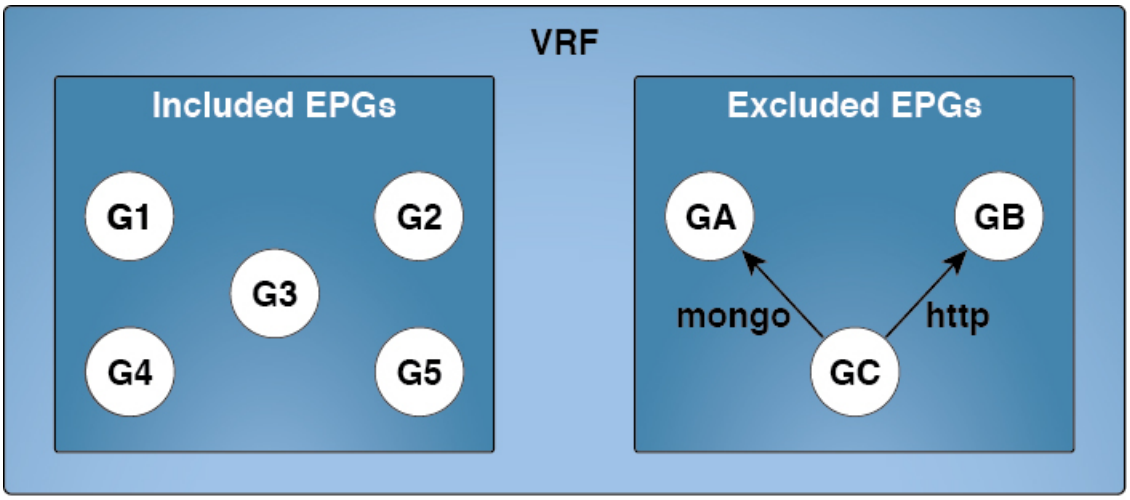
契約優先グループが設定されている VRF で、EPG に利用可能なポリシー適用には 2 種類あります。

- EPG を含む：EPG が契約優先グループのメンバーシップを持っている場合、EPG は契約をせずにお互いに自由に通信できます。これは、source-any-destination-any-permit デフォルト ルールに基づくものです。
- EPG を除外：優先グループのメンバーではない EPG は、相互に通信するために契約が必要です。そうしない場合、デフォルトの source-any-destination-any-deny ルールが適用されます。

契約優先グループ機能では、VRF で EPG 間のより高度な通信の制御が可能です。VRF の EPG のほとんどはオープン通信ですが、一部には他の EPG との制限がある場合、契約優先グループとフィルタ付きの契約の組み合わせを設定し、EPG 内の通信を正確に制御できます。

優先グループから除外されている EPG は、source-any-destination-any-deny デフォルト ルールを上書きする契約がある場合にのみ、他 EPG と通信できます。

図 *11* : 契約優先グループの概要



Source	Destination	Filter	Action
any	any	implicit	permit
GA	any	implicit	deny
any	GA	implicit	deny
GB	any	implicit	deny
any	GB	implicit	deny
GC	any	implicit	deny
any	GC	implicit	deny

Source	Destination	Filter	Action
GC	GA	mongo	permit
GA	GC	mongo	permit
GC	GB	http	permit
GB	GC	http	permit

サービス グラフ サポート

APIC リリース 4.0(1) 以降では、サービス グラフによって作成された EPG を優先契約グループに含めることができます。優先グループ メンバーシップのタイプ（include または exclude）を定義する新しいポリシー（サービス EPG ポリシー）が使用可能です。設定後は、デバイス選択ポリシーまたはサービス グラフ テンプレートのアプリケーションを通じて適用できます。

また、シャドウ EPG を優先グループに含めるか、優先グループから除外するかも設定できるようになりました。

制限事項

以下の制限が契約優先グループに適用されます。

- L3Out およびアプリケーション EPG が契約優先グループで設定されており、EPG が VPC でのみ展開されているトポロジで、VPC の 1 つのリーフ スイッチのみに L3Out のプレフィックス エントリがあることがわかります。この場合、VPC の他のリーフ スイッチにはエントリがなく、そのためトラフィックをドロップします。

  この問題を回避するには、次のいずれかを行います。

  - VRF の契約グループを無効および再度有効にします。

- L3Out EPG のプレフィックス エントリを削除し再度作成します。

- また、サービス グラフ契約のプロバイダまたはコンシューマ EPG が契約グループに含まれる場合、シャドウ EPG は契約グループから除外できません。シャドウ EPG は契約グループで許可されますが、シャドウ EPG が展開されているノードで契約グループ ポリシーの展開をトリガしません。ノードに契約グループ ポリシーをダウンロードするには、契約グループ内にダミー EPG を展開します。

契約優先グループの注意事項

契約優先グループを設定する際には、次の注意事項を参照してください:

- (s, g) エントリが境界リーフ スイッチにインストールされている場合、次の条件が満たされたときに、ファブリックからファブリック外のこの送信元に送信されるユニキャスト トラフィックがドロップされることがあります。
  - L3Out EPG で優先グループが使用されている
  - 送信元のユニキャスト ルーティング テーブルでデフォルト ルート 0.0.0.0/0 が使用されている

  これは予想された動作です。

- 契約優先グループに含まれる EPG は、外部 EPG (InstP) の 0/0 プレフィックスではサポートされていません。外部 EPG (InstP) からテナント EPG に対し、契約優先グループで使用するために 0/0 プレフィックスが必要な場合には、0/0 を 0/1 と 128/1 に分割することができます。

- 契約優先グループ EPG は、GOLF 機能ではサポートされていません。アプリケーション EPG と GOLF の L3Out EPG との間の通信は、明示的な契約によって制御する必要があります。

GUI を使用した契約優先グループの設定

始める前に

テナントと VRF、および契約優先グループを使用する EPG を作成します。

ステップ 1 メニュー バーで、**[Tenants]** > **テナント名**をクリックします。
ステップ 2 [Navigation] ペインで、テナント、[Networking]、[VRFs] の順に展開します。
ステップ 3 契約優先グループを設定する VRF 名を展開して、[EPG Collection for VRF] をクリックします。
ステップ 4 [Policy] および [General] タブを選択します。
ステップ 5 **Preferred Group Member** フィールドで、**Enabled** をクリックします。
ステップ 6 **Submit** をクリックします。

ステップ 7 **Navigation** ウィンドウで、**Application Profiles** を展開し、テナント VRF のアプリケーション プロファイルを作成するか、展開します。

ステップ 8 **Application EPGs** を展開し、契約優先グループを使用する EPG をクリックします。

ステップ 9 [Policy] および [General] タブを選択します。

ステップ 10 **Preferred Group Member** フィールドで、**Include** をクリックします。

ステップ 11 **Submit** をクリックします。

次のタスク

この EPG と無制限の通信を行う、他の EPG の優先グループのメンバーシップを有効にします。また、優先グループの EPG とメンバーではないかもしれない他の EPG の間の通信を制御する、適切な契約を設定することもできます。

（注） L4-L7 サービス グラフを介して優先グループ メンバーをサポートする場合は、L4-L7 サービス EPG ポリシーを作成する必要があります。L4-L7 サービス EPG ポリシーの作成に関する詳細については、GUI を使用した L4-L7 サービス EPG ポリシーの作成 （94 ページ）を参照してください。

NX-OS スタイル CLI を使用したコントラクト優先グループの設定

APIC NX-OS スタイル CLI を使用して、コントラクト優先グループを設定することができます。この例では、VRF のコントラクト優先グループが設定されています。VRF を使用する EPG のひとつは、優先グループに含まれます。

始める前に

コントラクト優先グループで消費されるテナント、VRF、EPG を作成します。

手順の概要

1. configure
2. **tenant***tenant-name*
3. **vrf context** *vrf-name*
4. **whitelist-blacklist-mix**
5. **bridge-domain** *bd-name*
6. **vrf member** *vrf-name*
7. **application** *app-name*
8. **epg** *epg-name*
9. **bridge-domain member** *bd-name*
10. **vrf-blacklist-mode**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	configure
	例：
	`apic1# configure`
	`apic1(config)#`
		設定モードを開始します。

ステップ 2	**tenant***tenant-name* 例： `apic1(config)# tenant tenant64`	テナントを作成するか、テナント設定モードを開始します
ステップ 3	**vrf context** *vrf-name* 例： `apic1(config-tenant)# vrf context vrf64`	VRF を作成するか、VRF 設定モードを開始します

ステップ 4	**whitelist-blacklist-mix**
	例：
	`apic1(config-tenant-vrf)# whitelist-blacklist-mix`
	`apic1(config-tenant-vrf)# exit`
		VRF のコントラクト優先グループを有効にし、テナント設定モードに戻ります。

ステップ 5	**bridge-domain** *bd-name* 例： `apic1(config-tenant)# bridge-domain  bd64`	VRF のブリッジ ドメインを作成するか、BD 設定モードを開始します。

ステップ 6	**vrf member** *vrf-name*
	例：
	`apic1(config-tenant-bd)# vrf member vrf64`
	`apic1(config-tenant-bd)# exit`
		ブリッジ ドメインと VRF を関連付け、テナント設定モードに戻ります。

ステップ 7	**application** *app-name* 例： `apic1(config-tenant)# application app-ldap`	アプリケーションを作成するか、アプリケーション設定モードを開始します。
ステップ 8	**epg** *epg-name* 例： `apic1(config-tenant-app)# epg epg-ldap`	EPG を作成するか、EPG テナント アプリケーション EPG 設定モードを開始します。
ステップ 9	**bridge-domain member** *bd-name* 例： `apic1(config-tenant-app-epg)# bridge-domain member bd64`	ブリッジ -ドメインに EPG を関連付けます。
ステップ 10	**vrf-blacklist-mode** 例：	コントラクト優先グループに含まれるこの EPG を設定します。

	コマンドまたはアクション	目的
	apic1(config-tenant-app-epg)# vrf-blacklist-mode	

例

次の例では、vrf64 のコントラクト優先グループを作成し、epg-ldap を含めます。

```
apic1# configure
apic1(config)# tenant tenant64
apic1(config-tenant)# vrf context vrf64
apic1(config-tenant-vrf)# whitelist-blacklist-mix
apic1(config-tenant-vrf)# exit

apic1(config-tenant)# bridge-domain bd64
apic1(config-tenant-bd)# vrf member vrf64
apic1(config-tenant-bd)# exit

apic1(config-tenant)# application app-ldap
apic1(config-tenant-app)# epg epg-ldap
apic1(config-tenant-app-epg)# bridge-domain member bd64
apic1(config-tenant-app-epg)# vrf-blacklist-mode
```

REST API を使用した契約優先グループの設定

次の例は、契約優先グループを作成 vrf64 、し、VRF で次の 3 つの Epg を作成します。

- epg ldap ：優先グループに含まれています
- メール ：優先グループに含まれています
- radius ：優先グループから除外

始める前に

VRF で、テナント、Vrf、および、Epg を作成します。

XML の例などと、post を送信することにより契約優先グループを作成します。

例：

```
<polUni>
  <fvTenant name="tenant64">
    <fvCtx name="vrf64"> <vzAny prefGrMemb="enabled"/> </fvCtx>
    <fvBD name="bd64"> <fvRsCtx tnFvCtxName="vrf64"/> </fvBD>
    <fvAp name="app-lldp">
      <fvAEPg name="epg-ldap" prefGrMemb="include">
        <fvRsBd tnFvBDName="bd64"/>
        <fvRsPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/3]" encap="vlan-113"
instrImedcy="immediate"/>
      </fvAEPg>
      <fvAEPg name="mail" prefGrMemb="include">
        <fvRsBd tnFvBDName="bd64"/>
        <fvRsPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/4]" encap="vlan-114"
```

```
instrImedcy="immediate"/>
      </fvAEPg>
      <fvAEPg name="radius" prefGrMemb="exclude">
        <fvRsBd tnFvBDName="bd64"/>
        <fvRsPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/5]" encap="vlan-115"
instrImedcy="immediate"/>
      </fvAEPg>
    </fvAp>
  </fvTenant>
</polUni>
```

次のタスク

通信を制御するには、契約の作成、 `radius` 他 Epg で EPG。

GUI を使用した L4-L7 サービス EPG ポリシーの作成

このタスクでは、EPGを優先グループに含めるか、優先グループから除外するかを定義するポリシーを作成します。優先グループ メンバーシップにより、エンドポイントは契約がなくても相互に通信できます。作成したポリシーは、EPG にサービス グラフ テンプレートを適用するときに選択できます。

始める前に

テナントを作成しておく必要があります。

ステップ 1 メニュー バーで、[Tenant] > テナント名を選択します。

ステップ 2 [Navigation] ペインで、[Policies] > [Protocol] > [L4-L7 Service EPG Policy] を選択します。

ステップ 3 [Navigation] ペインで、[L4-L7 Service EPG Policy] を右クリックして [Create L4-L7 Service EPG Policy] を選択します。

[Create L4-L7 Service EPG Policy] ダイアログボックスが表示されます。

ステップ 4 [Name] フィールドにポリシーの一意の名前を入力します。

ステップ 5 オプション。[Description] フィールドにポリシーの説明を入力します。

ステップ 6 [Preferred Group Member] フィールドで、EPG を除外するか優先メンバーとして含めるかを選択します。

ステップ 7 [Submit] をクリックします。`

新しく作成したポリシーが [L4-L7 Service EPG Policy] 作業ウィンドウ リストに表示されます。作業ウィンドウでポリシーを編集するには、ポリシーを含む行をダブルクリックします。

次のタスク

サービス グラフを EPG に適用するときに、サービス グラフ テンプレートで新しい L4-L7 サービス EPG ポリシーを選択できるようになりました。『*Cisco APIC Layer 4 to Layer 7 Services*

Deployment Guide』の「Using the GUI」の章で「Applying a Service Graph Template to Endpoint Groups Using the GUI」を参照してください。

許可ルールと拒否ルールを含む契約

許可ルールおよび拒否ルールを含む契約の概要

Cisco Application Policy Infrastructure Controller（Cisco APIC）リリース 3.2 以降では、許可だけではなく、許可と拒否の両方のアクションを含む契約を設定できます。さまざまな優先順位（デフォルト、高、中、低）の拒否アクションを設定できます。

ルールの競合は次のように解決されます。

- 暗黙の否定には、すべてのルールの中で最も低い優先順位が割り当てられます。
- VzAny 間の契約には暗黙の拒否より高い優先順位が割り当てられます。
- EPG 間の契約のルールは vzAny 間のルールより優先順位が高いため、特定の EPG ペア間の契約は vzAny の契約よりも優先されます。
- 特定の EPG ペア間の契約に含まれるデフォルト優先順位の拒否ルールは、その EPG ペアの許可ルールと優先順位レベルが同じです。同じ優先順位の許可ルールと拒否ルールの両方がトラフィックに一致する場合は、拒否ルールが優先されます。
- vzAny 間の契約に含まれるデフォルト優先順位の拒否ルールは、その vzAny ペアの許可ルールと優先順位レベルが同じです。同じ優先順位の許可ルールと拒否ルールの両方がトラフィックに一致する場合は、拒否ルールが優先されます。
- 優先順位が最も高い拒否ルールは、EPG 間の契約と同じレベルで処理されます。
- 優先順位が中の拒否ルールは、vzAny-EPG 間の契約と同じレベルで処理されます。
- 優先順位が最も低い拒否ルールは、vzAny 間の契約と同じレベルで処理されます。
- EPG 間の契約で拒否の優先順位を下げると、EPG 間の許可ルールの一致が拒否よりも優先されます。