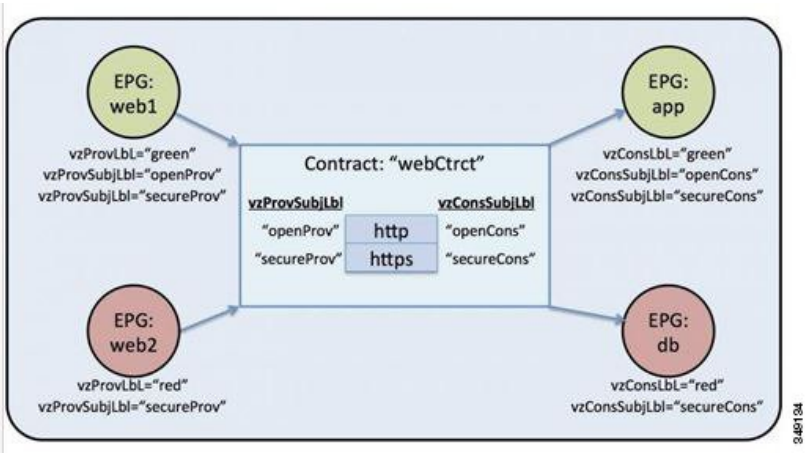
テナント ポリシーの例

この章の内容は、次のとおりです。

テナント ポリシー例の概要

この付録のテナント ポリシー例の説明では、XML 用語（http://en.wikipedia.org/wiki/XML#Key_terminology）を使用します。 この例では、基本的な APIC ポリシー モデル構造が XML コードにどのようにレンダリングされるかを示します。 次の図は、テナント ポリシー例の概要について説明します。

図 *1*：テナント *Solar* に含まれる *EPG* とコントラクト



この図では、webCtrct および EPG ラベルと呼ばれるコントラクトに従って、グリーンラベルの EPG:web1 が http と https の両方を使用してグリーンラベルの EPG:app と通信でき、レッドラベルの EPG:web2 は https のみを使用してレッドラベルの EPG:db と通信できます。

テナント ポリシー例の XML コード

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

      <vzFilter name="Http">
          <vzEntry name="e1" etherT="ipv4" prot="tcp" dFromPort="80" dToPort="80"/>
      </vzFilter>

      <vzFilter name="Https">
          <vzEntry name="e1" etherT="ipv4" prot="tcp" dFromPort="443" dToPort="443"/>
      </vzFilter>

      <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>

      <fvCtx name="solarctx1"/>

      <fvBD name="solarBD1">
          <fvRsCtx tnFvCtxName="solarctx1" />
          <fvSubnet ip="11.22.22.20/24">
            <fvRsBDSubnetToProfile  tnL3extOutName="rout1" tnRtctrlProfileName="profExport"
 />
          </fvSubnet>
          <fvSubnet ip="11.22.22.211/24">
              <fvRsBDSubnetToProfile  tnL3extOutName="rout1"
tnRtctrlProfileName="profExport"/>
          </fvSubnet>
      </fvBD>

      <fvAp name="sap">
          <fvAEPg name="web1">
              <fvRsBd tnFvBDName="solarBD1" />
              <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
              <fvRsProv tnVzBrCPName="webCtrct" matchT="All">
      <vzProvSubjLbl name="openProv"/>
      <vzProvSubjLbl name="secureProv"/>
                  <vzProvLbl name="green"/>
              </fvRsProv>
          </fvAEPg>
          <fvAEPg name="web2">
              <fvRsBd tnFvBDName="solarBD1" />
              <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
              <fvRsProv tnVzBrCPName="webCtrct" matchT="All">
                  <vzProvSubjLbl name="secureProv"/>
                  <vzProvLbl name="red"/>
              </fvRsProv>
          </fvAEPg>
          <fvAEPg name="app">
              <fvRsBd tnFvBDName="solarBD1" />
              <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
```

```
            <fvRsCons tnVzBrCPName="webCtrct">
    <vzConsSubjLbl name="openCons"/>
    <vzConsSubjLbl name="secureCons"/>
                <vzConsLbl name="green"/>
            </fvRsCons>
        </fvAEPg>
        <fvAEPg name="db">
            <fvRsBd tnFvBDName="solarBD1" />
            <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
            <fvRsCons tnVzBrCPName="webCtrct">
                <vzConsSubjLbl name="secureCons"/>
                <vzConsLbl name="red"/>
            </fvRsCons>
        </fvAEPg>
    </fvAp>
  </fvTenant>
</polUni>
```

テナント ポリシー例の説明

この項には、テナント ポリシー例の詳しい説明が含まれます。

ポリシー ユニバース

ポリシー ユニバースには、各テナントのポリシーが定義されているすべてのテナント管理対象オブジェクトが含まれます。

```
<polUni>
```

最初の行のこの開始タグ <polUni> は、ポリシー ユニバース要素の開始を示します。 このタグは、ポリシーの最後にある </polUni> と一致します。 間にあるのはすべて、ポリシー定義です。

テナント ポリシーの例

タグ <fvTenant> は、テナント要素の開始を識別します。

```
<fvTenant name="solar">
```

このテナントのポリシーはすべてこの要素で定義されます。 この例でのテナントの名前はsolarです。テナントの名前はシステム内で一意である必要があります。テナントが含む主要な要素は、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、およびEPGを含むアプリケーション プロファイルです。

フィルタ

フィルタ要素は、タグ <vzFilter> から始まり、タグ <vzEntry> で示される要素が含まれます。

次に、「HTTP」と「Https」フィルタを定義する例を示します。 フィルタの最初の属性が名前で、name 属性の値はテナントに一意の文字列です。 これらの名前は異なるテナントで再利用できます。 これらのフィルタは、この例の後でコントラクト内のサブジェクト要素で使用されます。

```
    <vzFilter name="Http">
```

```
        <vzEntry name="e1" etherT="ipv4" prot="tcp" dFromPort="80" dToPort="80"/>
    </vzFilter>

    <vzFilter name="Https">
        <vzEntry name="e1" etherT="ipv4" prot="tcp" dFromPort="443" dToPort="443"/>
    </vzFilter>
```

この例では、これら 2 つのフィルタ HTTP および Https を定義します。 フィルタの最初の属性が名前で、name 属性の値はテナントに一意の文字列です。つまり、これらの名前は異なるテナントで再利用できます。 これらのフィルタは、この例の後でコントラクト内のサブジェクト要素で使用されます。

各フィルタには、各エントリがレイヤ 4 TCP または UDP ポート番号のセットを説明する 1 つ以上のエントリを含めることができます。 `<vzEntry>` 要素の考えられる属性の一部は次のとおりです。

- `name`
- `prot`
- `dFromPort`
- `dToPort`
- `sFromPort`
- `sToPort`
- `etherT`
- `ipFlags`
- `arpOpc`
- `tcpRules`

この例では、各エントリの name 属性が指定されます。 名前はフィルタ内で一意でなければならない ASCII 文字列ですが、他のフィルタで再利用できます。 なぜなら、この例では、後で特定のエントリを参照せず、「e1」という単純な名前が与えられるためです。

EtherType 属性 `etherT` が次です。 ipv4 の値が割り当てられ、このフィルタが IPv4 パケット用であることを指定します。 この属性には考えられる他の多くの値があります。 一般的なものには、`ARP`、`RARP`、および今後のリリースでは `IPv6` があります。 デフォルトは `unspecified` なので、値を割り当てることが重要です。

EtherType 属性の後は、`prot` 属性です。この属性は `tcp` に設定され、このフィルタが TCP トラフィック用であることを示します。代替プロトコル属性には、`udp`、`icmp`、および `unspecified`（デフォルト）があります。

プロトコルの後、宛先の TCP ポート番号は 80 ～ 80 の範囲（正確には TCP port 80）になるように `dFromPort` および `dToPort` 属性で割り当てられます。 送信元と宛先が異なっている場合、それらはポート番号の範囲を指定します。

この例では、これらの宛先ポート番号は属性 `dFromPort` および `dToPort` で指定されます。 ただし、コントラクトで使用されている場合は、TCP クライアントからサーバへの宛先ポートのためにリターントラフィックの送信元ポートとして使用する必要があります。 詳細については、この例の後に出てくる属性 `revFltPorts` を参照してください。

2 番目のフィルタは基本的に同じ機能がありますが、ポート 443 に対するものです。

フィルタは、ターゲットの識別名 tDn によってコントラクト内のサブジェクトによって参照されます。 tDn 名は次のように構成されます。

```
uni/tn-<tenant name>/flt-<filter name>
```

たとえば、上記の最初のフィルタの tDn は uni/tn-coke/flt-Http です。 2 番目のフィルタには名前 uni/tn-coke/flt-Https があります。 いずれの場合も、solar がテナント名から取得されます。

コントラクト

コントラクト要素は、vzBrCP でタグ付けされ、name 属性があります。

```
<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"/>
       <vzRsFiltAtt tnVzFilterName="Https "/>
       <vzRsOutTermGraphAtt graphName="G2" termNodeName="TProv"/>
   </vzSubj>
</vzBrCP>
```

コントラクトはEPG間のポリシー要素です。コントラクトには、コントラクトを作成して消費する EPG 間で適用されるすべてのフィルタが含まれます。 コントラクト要素は、vzBrCP でタグ付けされ、name 属性があります。 コントラクト要素で使用できるその他の属性については、オブジェクトモデルの参照資料を参照してください。 この例では、webCtrct という名前のコントラクトが 1 つあります。

コントラクトには、各サブジェクトが一連のフィルタを含む複数のサブジェクト要素が含まれます。 この例では、2 つのサブジェクト、http と https があります。

コントラクトは、それを提供または消費する EPG によって後で参照されます。 EPG は、以下の方法で名前によってそのコントラクトを参照します。

```
uni/tn-[tenant-name]/brc-[contract-name]
```

tenant-name はテナントの名前で、この例では「solar」となります。contract-name はコントラクトの名前です。 この例では、コントラクトの tDn 名は uni/tn-solar/brc-webCtrct です。

サブジェクト

サブジェクト要素は、タグ vzSubj から始まり、3 つの属性、name、revFltPorts および matchT を持ちます。 name は、単にサブジェクトの ASCII 名です。

revFltPorts は、このサブジェクトのフィルタ内のレイヤ 4 送信元および宛先ポートをフィルタの説明に示すとおりに転送方向（つまり、コンシューマからプロデューサ EPG の方向）に使用する必要があり、逆方向には逆の方法で使用する必要があることを示すフラグです。 この例では、「http」サブジェクトには、TCP宛先ポート 80 を定義し、送信元ポートを指定していない「HTTP」フィルタが含まれます。 revFltPorts フラグが true に設定されているため、ポリシーは、TCP **宛**

先ポート80およびコンシューマからプロデューサへのトラフィック用の送信元ポートであり、また、TCP宛先ポートおよびプロデューサからコンシューマへのトラフィック用の**送信元ポート80**になります。コンシューマがプロデューサへのTCP接続を開始することを前提としています（コンシューマがクライアントで、プロデューサがサーバ）。

指定しない場合、`revFltPrts` 属性のデフォルト値は `false` です。

ラベル

一致タイプ属性、`provmatchT`（プロバイダー一致の場合）および `consmatchT`（コンシューマ一致の場合）は、サブジェクトが所定のコンシューマとプロデューサのペアに対し適用されるかを判断するためにサブジェクトラベルがどのように比較されるかを決定します。次の一致タイプの値が使用可能です。

- `All`
- `AtLeastOne`（デフォルト）
- `None`
- `ExactlyOne`

サブジェクトがプロデューサとコンシューマEPG間のトラフィックに適用されるかどうかを決定する場合、一致属性は、これらのEPGで定義されている（または定義されていない）サブジェクトラベルがサブジェクト内のラベルとどのように比較されるべきかを決定します。一致属性の値が `All` に設定されると、それはプロバイダー サブジェクト ラベル `vzProvSubjLbl` がサブジェクト内で定義されたすべての `vzProvSubjLbl` ラベルと一致するプロバイダーにのみ適用されます。2つのラベルが定義されている場合、両方ともプロバイダー内にある必要があります。プロバイダー EPG に 10 個のラベルがある場合、サブジェクト内のすべてのプロバイダー ラベルが存在する限り、一致が確認されます。同様の基準が `vzConsSubjLbl` を使用するコンシューマに使用されます。`matchT` 属性値が `AtLeastOne` の場合、ラベルの1つだけが一致する必要があります。`matchT` 属性が `None` の場合、サブジェクト内のプロバイダー ラベルがプロバイダー EPG のプロバイダーラベルと一致しない場合にのみ一致が発生します。コンシューマの場合も同様です。

プロデューサまたはコンシューマEPGにサブジェクトラベルがなく、サブジェクトがラベルを持たない場合、一致は `All`、`AtLeastOne`、および `None` の場合に発生します（ラベルを使用しない場合は、サブジェクトが使用され `matchT` 属性は問題になりません）。

この例には示されていないサブジェクトのオプション属性は `prio` で、フィルタに一致するトラフィックのプライオリティが指定されます。考えられる値は、`gold`、`silver`、`bronze`、または `unspecified`（デフォルト）です。

この例では、サブジェクト要素にフィルタ要素、サブジェクト ラベル要素およびグラフ要素への参照が含まれます。`<vzRsSubjFiltAtt tDn="uni/tn-coke/flt-Http"/>` は事前に定義されたフィルタへの参照です。この要素は `vzRsSubjFiltAtt` タグによって識別されます。

`<vzRsSubjGraphAtt graphName="G1" termNodeName="TProv"/>` は端末接続を定義します。

`<vzProvSubjLbl name="openProv"/>` は「openProv」という名前のプロバイダー ラベルを定義します。ラベルは、どのサブジェクトがどのEPGに適用されるかを認定したりフィルタリングするた

めに使用されます。 この特定のラベルがプロバイダー ラベルであり、対応するコンシューマ ラベルがタグ vzConsSubjLbl で識別されます。 これらのラベルは、現在のコントラクトに関連付けられたプロバイダーまたはコンシューマ EPG の対応するラベルと一致します。 前述の matchT 基準に従って一致が発生する場合は、特定のサブジェクトがEPGに適用されます。一致が発生しない場合、サブジェクトは無視されます。

複数のプロバイダーおよびコンシューマのサブジェクト ラベルをサブジェクトに追加して、より複雑な一致基準を可能にすることができます。 この例では、各サブジェクトに各タイプのラベルが 1 個だけあります。 ただし、最初のサブジェクトのラベルは 2 番目のサブジェクトのラベルとは異なり、これら 2 つのサブジェクトを対応する EPG のラベルに応じて、それぞれ処理できます。 サブジェクト要素内の要素の順序は重要ではありません。

コンテキスト

コンテキストは fvCtx タグによって識別され、name 属性が含まれます。

```
<fvCtx name="solarctx1"/>
```

テナントには、複数のコンテキストを含めることができます。 この例では、テナントは「solartx1」という名前のコンテキストを 1 個使用します。 名前は、テナント内で一意である必要があります。

コンテキストは、レイヤ 3 のアドレス ドメインを定義します。 レイヤ 3 ドメイン内のすべてのエンドポイントが一意の IPv4 または IPv6 アドレスを持っている必要があります。なぜなら、ポリシーで許可されている場合にこれらのデバイス間でパケットを直接転送できるためです。 コンテキストは、ネットワーキング ワールドの仮想ルーティングおよび転送（VRF）インスタンスに相当します。

コンテキストが一意のIPアドレス空間を定義する一方で、対応するサブネットがブリッジ ドメイン内で定義されます。 各ブリッジ ドメインはその後コンテキストに関連付けられます。

ブリッジ ドメイン

ブリッジ ドメインの要素は fvBD タグで識別され、name 属性があります。

```
        <fvBD name="solarBD1">
            <fvRsCtx tnFvCtxName="solarctx1" />
            <fvSubnet ip="11.22.22.20/24">
                <fvRsBDSubnetToProfile  tnL3extOutName="rout1"
tnRtctrlProfileName="profExport" />
            </fvSubnet>
            <fvSubnet ip="11.22.23.211/24">
                <fvRsBDSubnetToProfile  tnL3extOutName="rout1"
tnRtctrlProfileName="profExport"/>
            </fvSubnet>
        </fvBD>
```

ブリッジ ドメインの要素内では、サブネットが定義され、対応するレイヤ 3 コンテキストへの参照が行われます。 各ブリッジ ドメインは、コンテキストにリンクされ、少なくとも 1 個のサブネットを設定する必要があります。

この例では、「solarBD1」という名前のブリッジ ドメインを 1 個使用します。 この例では、「solarctx1」というコンテキストが、`fvRsCtx` とタグ付けされた要素を使用して参照され、`tnFvCtxName` 属性に値「solarctx1」が与えられます。 この名前は、上記で定義したコンテキストから取得されます。

サブネットがブリッジドメイン内に含まれ、ブリッジドメインは複数のサブネットを含むことができます。 この例では、2 個のサブネットを定義します。 ブリッジ ドメイン内で使用されるすべてのアドレスは、サブネットで定義されるアドレス範囲のいずれかに分類される必要があります。ただし、サブネットは、使用されることがないであろう多数のアドレスを含む大規模なサブネットであるスーパーネットにすることもできます。 現在および将来のアドレスすべてに対応する大規模なサブネットを 1 個指定すると、ブリッジ ドメインの仕様を簡素化できます。 ただし、異なるサブネットがブリッジ ドメイン内で重複したり、または同じコンテキストに関連付けられている他のブリッジドメインで定義されたサブネットと重複してはなりません。サブネットは、他のコンテキストに関連付けられている他のサブネットと重複できます。

前述のサブネットは、11.22.22.xx/24 と 11.22.23.xx/24 です。 ただし、24 だけが使用されることをマスクが示していても、アドレスの完全な 32 ビットが与えられます。それは、この IP 属性がそのサブネットに対するルータの完全なIPアドレスの役割も示しているためです。最初のケースでは、ルータの IP アドレス（デフォルト ゲートウェイ）は 11.22.22.20 で、2 番目のサブネットでは、11.22.23.211 です。

エントリ 11.22.22.20/24 は以下に相当しますが、コンパクト形式です。

- サブネット：11.22.22.00
- サブネット マスク：255.255.255.0
- デフォルト ゲートウェイ：11.22.22.20

アプリケーション プロファイル

アプリケーション プロファイルの開始はタグ `fvAp` で示され、name 属性があります。

```
<fvAp name="sap">
```

この例では、アプリケーション ネットワーク プロファイルが 1 つあり、「sap」という名前です。

アプリケーション プロファイルは、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 プロファイルは、互いに論理的に関連する複数の EPG を保持するために使用される簡易で便利なコンテナです。 それらは、「sap」などの提供するアプリケーション、「インフラストラクチャ」などの提供する機能、「DMZ」などのデータセンターの構造内のそれらが存在する場所、または管理者が使用することを選択した組織化の原則によって組織化できます。

アプリケーション プロファイルに含まれるプライマリ オブジェクトは、エンドポイント グループ（EPG）です。 この例では、「sap」アプリケーション プロファイルには 4 個の EPG、web1、web2、app および db が含まれます。

エンドポイントおよびエンドポイント グループ（EPG）

EPG は、タグ `fvAEPg` で始まり、name 属性があります。

```
<fvAEPg name="web1">
    <fvRsBd tnFvBDName="solarBD1" />
    <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
    <fvRsProv tnVzBrCPName="webCtrct" matchT ="All">
        <vzProvSubjLbl name="openProv"/>
        <vzProvSubjLbl name="secureProv"/>
        <vzProvLbl name="green"/>
    </fvRsProv>
</fvAEPg>
```

EPGは、ポリシーモデルの最も重要な基本オブジェクトです。これは、ポリシーの観点から同じ方法で処理されるエンドポイントの集合を表します。それらのエンドポイントは個別に設定および管理されるのではなく、EPG 内に配置され、集合またはグループとして管理されます。

EPG オブジェクトは、どのようなポリシーが適用されるのか、また他のどの EPG がこの EPG と通信できるかを規定するラベルが定義されている場所です。また、EPG 内のエンドポイントが関連付けられるブリッジ ドメイン、およびそれらが関連付けられる Virtual Machine Manager（VMM）のドメインへの参照が含まれています。VMM により、2 台の VM サーバ間の仮想マシンのモビリティがアプリケーションのダウンタイムなしで即座に可能になります。

この例の最初の EPG は「web1」という名前です。EPG 内の `fvRsBd` 要素は、関連付けられるブリッジ ドメインを定義します。ブリッジ ドメインは `tnFxBDName` 属性の値によって識別されます。この EPG は、前述の「ブリッジ ドメイン」の項で名前を付けられた「solarBD1」というブリッジ ドメインに関連付けられます。ブリッジ ドメインへのバインディングは、デフォルト ゲートウェイ アドレスがこの EPG のエンドポイントのためにどうあるべきかを理解するためにシステムによって使用されます。エンドポイントが同じサブネットにあるということや、ブリッジングを介してのみ通信できるという意味ではありません。エンドポイントのパケットがブリッジングまたはルーティングされるかどうかは、送信元エンドポイントがパケットをデフォルト ゲートウェイまたは要求される最後の宛先に送信するかどうかで決定されます。デフォルト ゲートウェイにパケットを送信すると、パケットはルーティングされます。

この EPG で使用される VMM ドメインは `fvRsDomAtt` タグによって識別されます。この要素は、他の場所で定義された VMM ドメイン オブジェクトを参照します。VMM ドメイン オブジェクトは、その `tDn` name 属性によって識別されます。この例では、「uni/vmmp-VMware/dom-mininet」と呼ばれる VMM ドメイン 1 個のみを示します。

「web1」EPG の次の要素は、この EPG が提供するコントラクトを定義し、`fvRsProv` タグによって識別されます。「web1」が複数のコントラクトを提供すると、`fvRsProv` 要素が複数あります。同様に、それが 1 つ以上のコントラクトを消費すると、`fvRsCons` 要素があります。

`fvRsProv` 要素には、提供されているコントラクトの名前である必須属性があります。「web1」は、`tDn="uni/tn-coke/brc-webCtrct"` と呼ばれる以前に定義されたコントラクト「webCtrct」を提供しています。

次の属性は、`matchT` 属性です。その属性には、それがサブジェクト ラベルのコントラクト内にあったので、プロバイダーまたはコンシューマのラベルと一致するための同じセマンティックがあります（`All`、`AtLeastOne` または `None` の値を取ることができます）。この条件は、対応するコ

ンシューマ ラベルと比較されるときにプロバイダーのラベルに適用されます。 ラベルの一致は、コンシューマとプロバイダーがコントラクトで許可された場合に通信できることを意味します。 つまり、コントラクトが通信を許可する必要があり、コンシューマとプロバイダーのラベルがプロバイダーで指定された一致基準を使用して一致する必要があります。

コンシューマには、対応する一致基準がありません。 使用される一致タイプはプロバイダーによって常に定められます。

プロバイダー要素 `fvRsProv` の中で、管理者は使用するラベルを指定する必要があります。 2 種類のラベル、プロバイダー ラベルとプロバイダー サブジェクト ラベルがあります。 プロバイダー ラベル `vzProvLbl` は、前述の `matchT` 基準を使用する他の EPG 内のコンシューマ ラベルに一致させるために使用されます。 プロバイダー サブジェクト ラベル `vzProvSubjLbl` は、コントラクトで指定されるサブジェクト ラベルに一致させるために使用されます。 ラベルの唯一の属性は name 属性です。

「web1」EPG では、2 つのプロバイダー サブジェクト ラベル `openProv` および `secureProv` が「webCtrct」コントラクトのサブジェクト「http」および「https」と一致するように指定されます。 1 つのプロバイダー ラベル「green」が、「App」EPG 内の同じラベルと一致する `All` の一致基準で指定されます。

この例の次の EPG「web2」は「web1」と似ていますが、`vzProvSubjLbl` が 1 つだけあり、ラベル自体は異なっています。

3 番目の EPG は「app」と呼ばれるもので、次のように定義されます。

```
<fvAEPg name="app">
    <fvRsBd tnFvBDName="solarBD1" />
    <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
    <fvRsCons tnVzBrCPName="webCtrct">
        <vzConsSubjLbl name="openCons"/>
        <vzConsSubjLbl name="secureCons"/>
        <vzConsLbl name="green"/>
     </fvRsCons>
</fvAEPg>
```

最初の部分は「web1」EPG とほぼ同じです。 主な違いは、この EPG は「webCtrct」のコンシューマで、対応するコンシューマ ラベルとコンシューマ サブジェクト ラベルがあることです。 構文はほぼ同じですが、タグで「Prov」が「Cons」に置き換えられます。 プロバイダーをコンシューマ ラベルに一致させるための一致タイプがプロバイダーで指定されるため、`FvRsCons` 要素に一致属性はありません。

最後の EPG では、純粋なコンシューマであるという点において「db」は「app」EPG と非常によく似ています。

この例では、EPG は単一コントラクトのコンシューマまたはプロデューサであり、EPG が即座に複数のコントラクトのプロデューサおよび複数のコントラクトのコンシューマになることが一般的です。

最後に

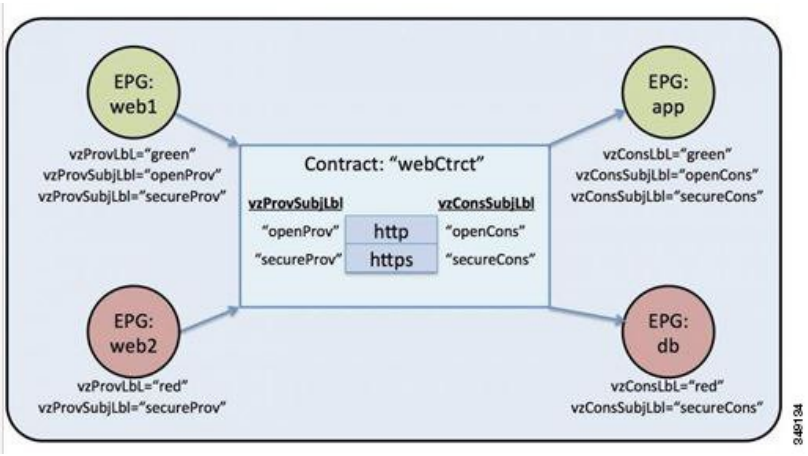
```
        </fvAp>
    </fvTenant>
</polUni>
```

最後の数行でポリシーが完了します。

この例のテナント ポリシーが行うこと

次の図は、コントラクトがエンドポイント グループ（EPG）の通信をどのように管理するかを示します。

図 *2*：*EPG/EPG* 通信を決定するラベルとコントラクト



4 つの EPG には、EPG:web1、EPG:web2、EPG:app および EPG:db という名前が付いています。 EPG:web1 および EPG:web2 は webCtrct と呼ばれるコントラクトを提供します。 EPG:app および EPG:db は、同じコントラクトを消費します。

EPG:web1 は EPG:app のみと通信でき、EPG:web2 は EPG:db のみと通信できます。 このインタラクションは、プロバイダーおよびコンシューマのラベル「green」と「red」によって制御されます。

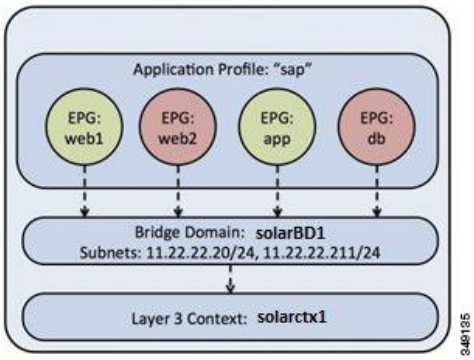
EPG:web1 が EPG:app と通信するとき、webCtrct コントラクトが使用されます。 EPG:app は、EPG:web1 が提供するコントラクトを消費するので、EPG:web1 への接続を開始できます。

EPG:web1 と EPG:app が通信を行うために使用できるサブジェクトは両方とも http および https です。なぜなら、EPG:web1 にはプロバイダー サブジェクト ラベル「openProv」があり、http サブジェクトにもそれがあるためです。 EPG:web1 には、プロバイダー サブジェクト ラベル「secureProv」があり、サブジェクト https も同様です。 同様に、EPG:app にはサブジェクト ラベル「openCons」および「secureCons」があり、サブジェクト http および https にもあります。

EPG:web2 が EPG:db と通信するとき、それらは https サブジェクトのみを使用できます。https サブジェクトのみがプロバイダーおよびコンシューマのサブジェクト ラベルを持っているためで

す。 EPG:db は EPG:web2 への TCP 接続を開始できます。なぜなら、EPG:db が EPG:web2 により提供されるコントラクトを消費するからです。

図 *3*：ブリッジ ドメイン、サブネット、およびレイヤ *3* コンテキスト



この例のポリシーは、EPG、アプリケーション プロファイル、ブリッジ ドメインおよびレイヤ 3 コンテキスト間の関係を次のように指定します。EPG の EPG:web1、EPG:web2、EPG:app および EPG:db は、「sap」と呼ばれるアプリケーション プロファイルのすべてのメンバーです。

これらの EPG は、ブリッジ ドメイン「solarBD1」にもリンクされます。 solarBD1 には、2 つのサブネット 11.22.22.XX/24 と 11.22.23.XX/24 があります。 4 つの EPG 内のエンドポイントは、これら 2 つのサブネット範囲内にある必要があります。 これら 2 つのサブネット内のデフォルト ゲートウェイの IP アドレスは 11.22.22.20 と 11.22.23.211 です。 solarBD1 ブリッジ ドメインは「solarctx1」レイヤ 3 コンテキストにリンクされます。

これらのポリシーの詳細はすべて、「solar」というテナントに含まれています。