ACIセキュリティポリシーのトラブルシューティング – 契約

内容

概要

このドキュメントでは、契約と呼ばれるACIセキュリティポリシーを理解してトラブルシューティングする手順について説明します。

背景説明

このドキュメントの内容は、『Troubleshooting Cisco Application Centric Infrastructure, Second Edition』マニュアルに記載されている、特に「Security Policies - Overview」、「Security Policies - Tools」、「Security Policies - EPG to EPG」、「Security Policies - Preferred group and Security Policies - vzAny to EPG 」の章から抜粋したものです。

概要

ACIソリューションの基本的なセキュリティアーキテクチャは、許可モデルに従います。VRFが**unenforced**モードに設定されていない限り、EPGからEPGへのすべてのトラフィックフローは暗黙的にドロップされます。初期状態の許可リストモデルに示されているように、デフォルトのVRF設定は**強制**モードです。スイッチノードにゾーン分割ルールを実装することで、トラフィックフローを許可または明示的に拒否できます。これらのゾーン分割ルールは、エンドポイントグループ(EPG)間の望ましい通信フローとそれらを定義するために使用する方法に応じて、さまざまな設定でプログラムできます。ゾーン分割ルールエントリはステートフルではなく、通常は、ルールがプログラムされた後に2つのEPGが与えられたポート/ソケットに基づいて許可/拒否する

ことに注意してください。

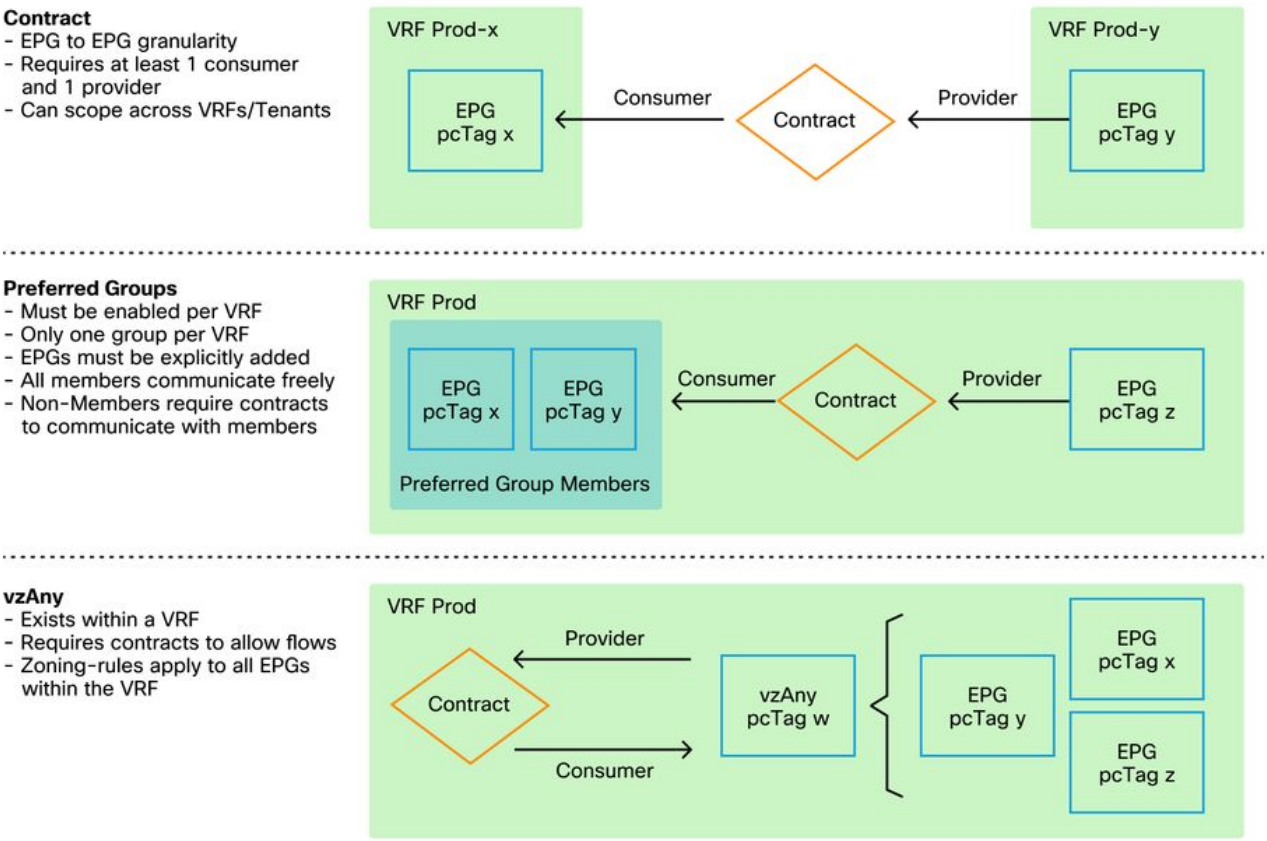
ゾーニング・ルールをプログラムする方法

ACI内でゾーン分割ルールをプログラムする主な方法は次のとおりです。

- **EPGとEPG間の契約：通常、2つ以上の異なるエンドポイントグループ間でゾーン分割ルールをプログラムするには**、少なくとも1つのコンシューマと1つのプロバイダーが必要です。
- **優先グループ**：VRFレベルでグループ化を有効にする必要があります。vrfごとに1つのグループしか存在できません。グループのすべてのメンバーが自由にコミュニケーションを取ることができます。非メンバーには、優先グループへのフローを許可するコントラクトが必要です。
- **vzAny:**特定のVRFで定義される「EPGコレクション」。vzAnyはVRF内のすべてのEPGを表します。vzAnyを使用すると、1つのEPGとVRF内のすべてのEPGの間で、1つの契約接続を介したフローが可能になります。

次の図を使用して、前述の各方法で制御できるゾーン分割ルールの粒度を参照できます。

ゾーニング・ルール方式の比較



ゾーニング・ルールをプログラミングするコントラクト方式を利用する一方で、コントラクトの範囲を定義するオプションがあります。ルート漏出/共有サービス設計が必要な場合は、このオプションを慎重に考慮する必要があります。ACIファブリック内で1つのVRFから別のVRFに到達する場合は、コントラクトを使用します。

スコープの値は次のとおりです。

- **アプリケーショ**ン：契約コンシューマ/プロバイダー関係は、同じアプリケーションプロファイル内で定義されたEPG間のルールのみをプログラムします。他のアプリケーションプロファイルEPG間で同じコントラクトを再利用すると、それらの間のクロストークは発生しません。
- **VRF（デフォルト）：契約コンシ**ューマ/プロバイダー関係は、同じVRF内で定義されているEPG間のルールをプログラムします。他のアプリケーションプロファイルEPG間で同じコントラクトを再利用すると、それらの間のクロストークが可能になります。望ましいフローのみが許可されるように注意してください。許可されない場合は、意図しないクロストークを防ぐために新しいコントラクトを定義する必要があります。
- **テナント**：契約のコンシューマ/プロバイダー関係は、同じテナント内で定義されているEPG間のルールをプログラムします。1つのテナント内の複数のVRFに関連付けられたEPGがあり、それらが同じコントラクトを消費または提供する場合、このスコープを使用してルート漏出を誘導し、VRF間通信を可能にすることができます。
- **グローバル**：契約のコンシューマ/プロバイダー関係は、ACIファブリック内の任意のテナント全体でEPG間のルールをプログラムします。これは定義の中で最も可能性の高い範囲であり、以前に定義した契約でこの機能を有効にすると、意図しないフローの漏れを防ぐように十分に注意する必要があります。

ゾーニング・ルール・エントリーの読み取り

ゾーン分割ルールがプログラムされると、リーフには次のように表示されます。

```
+---------+--------+--------+----------+----------------+---------+---------+-----------------+-
---------+----------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |      Dir       | operSt |  Scope  |       Name      |
Action  |       Priority       |
+---------+--------+--------+----------+----------------+---------+---------+-----------------+-
---------+----------------------+
```

- **[ルールID(Rule ID)]**：ルールエントリのID。一意の識別子として機能する以外に、真の意味はありません。
- **Src EPG：送信元エ**ンドポイントグループのVRF(pcTag)ごとの一意のID。
- **Dst EPG：宛先エ**ンドポイントグループのVRF(pcTag)ごとの一意のID。
- **FilterID：ル**ールが一致させようとしているフィルタのID。フィルタには、ルールが照合されるプロトコル情報が含まれています。
- **Dir：ゾー**ニングルールの方向性。
- **OperSt**：ルールの動作状態。
- **Scope：ル**ールが照合されるVRFの一意のID。
- **「名称」:**その入力がプログラムされた契約の名称。
- **Action：そのエ**ントリに一致した場合にリーフが行う処理。次が含まれます。[Drop、Permit、Log、Redirect]。
- **Priority：一致**するScope、SrcEPG、DstEPG、およびFilter Entriesが指定されたアクションに対してゾーン分割ルールが検証される順序。

Policy Content-Addressable Memory(CAM)

各ゾーニングルールがプログラムされると、フィルタエントリに対してマップされたゾーニングルールエントリのマトリックスがスイッチ上の**ポリシーCAM**を消費し始めます。ACIファブリックを通過する許可されたフローを設計する際は、最終設計に応じて新しいフローを作成するのではなく、契約を再利用する際に特別な注意を払う必要があります。結果として生じるゾーン分割

ルールを理解することなく、複数のEPG間で同じコントラクトを無計画に再利用すると、予期せず複数のフローにカスケードされる可能性があります。同時に、これらの意図しないフローは引き続きポリシーCAMを消費します。ポリシーCAMがいっぱいになると、ゾーン分割ルールのプログラミングが失敗し始め、設定やエンドポイントの動作によっては予期しない断続的な損失が発生する可能性があります。

VRFリーク、グローバルpcTag、および共有L3Outのポリシー適用方向

これは、契約を設定する必要がある共有サービスのユースケースの特殊なコールアウトです。Shared Servicesは通常、「テナント」または「グローバル」スコープのコントラクトの使用に依存するACIファブリック内のVRF間トラフィックを意味します。これを完全に理解するには、まず、EPGに割り当てられる一般的なpcTag値はグローバルに一意ではないという考えを強調する必要があります。pcTagはVRFにスコープされ、同じpcTagが別のVRF内で再利用される可能性があります。ルート漏出の説明が始まったら、サブネットやpcTagなど、グローバルに一意な値の必要性を含むACIファブリックの要件の適用を開始します。

この点を特別に考慮しているのは、EPGがコンシューマとプロバイダーの対比で方向性が重視されることです。共有サービスシナリオでは、通常、プロバイダーはグローバルpcTagを駆動してファブリック固有の値を取得することが期待されます。同時に、コンシューマはVRFスコープのpcTagを保持し、ポリシーを適用するためのグローバルpcTag値の使用をプログラムおよび理解できるように、特別な位置に置きます。

参考として、pcTagの割り当て範囲は次のとおりです。

- システム予約：1-15.
- グローバルスコープ：共有サービ16384プロバイダーEPGは16 ~ 100
- ローカルスコープ：VRFスコープEPGの場合は16385-65535。

VRFポリシー制御の強制方向

各VRFでは、強制方向設定を定義できます。

- エンフォースメント方向のデフォルト設定は[Ingress]です。
- エンフォースメント方向のもう1つのオプションは出力です。

ポリシーが適用される場所を理解するには、いくつかの異なる変数が必要です。

次の表は、セキュリティポリシーがリーフレベルで適用される場所を理解するのに役立ちます。

ポリシーはどこに適用されますか。

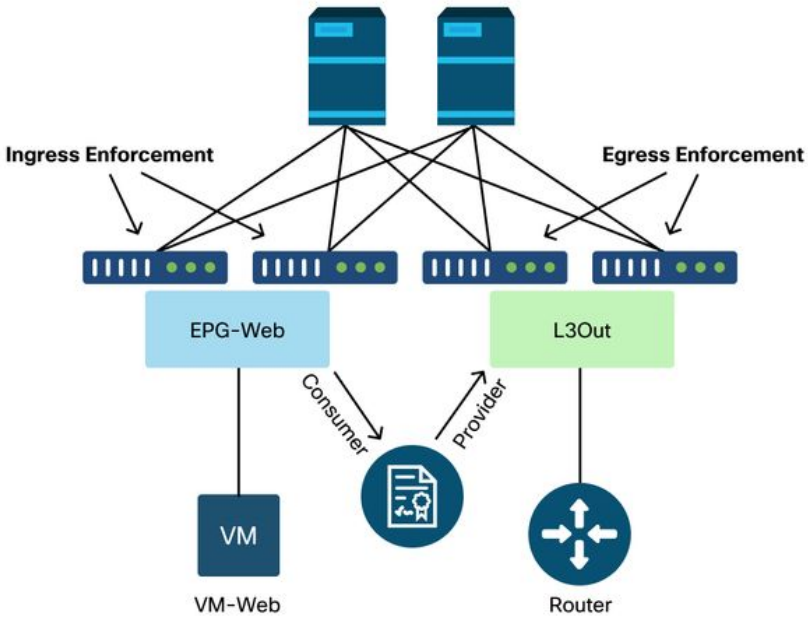
シナリオ	VRF強制モード	消費者	プロバイダー	ポリシーの適用
VRF内	入力/出力	EPG	EPG	・ 宛先エンドポイントが学習された場合：入力リーフ* ・ 宛先エンドポイントが学習されない場合：出力リーフ
	入力	EPG	L3Out EPG	コンシューマリーフ（非ボーダーリーフ）
	入力	L3Out EPG	EPG	プロバイダーリーフ（非ボーダーリーフ）
	出力	EPG	L3Out	Border leaf ->非Border Leafトラフィッ

			EPG	ク ・ 宛先エンドポイントが学習された場合：境界葉 ・ 宛先エンドポイントが学習されない場合：非境葉 非ボーダーリーフ – >ボーダーリーフ トラフィック
	出力	L3Out EPG	EPG	・ 境界リーフ
	入力/出力	L3Out EPG	L3Out EPG	入力リーフ*
	入力/出力	EPG	EPG	コンシューマリーフ
	入力/出力	EPG	L3Out EPG	コンシューマリーフ（非ボーダーリーフ）
VRF間	入力/出力	L3Out EPG	EPG	入力リーフ*
	入力/出力	L3Out EPG	L3Out EPG	入力リーフ*

*ポリシーの適用は、パケットによってヒットされた最初のリーフに適用されます。

次の図は、コンシューマとしてのEPG-Webとプロバイダーとしての L3Out EPGがVRF内契約を持つ契約適用の例を示しています。VRFが入力強制モードに設定されている場合、ポリシーはEPG-Webが存在するリーフノードによって適用されます。VRFが出力強制モードに設定されている場合、VM-Webエンドポイントが境界リーフで学習されると、ポリシーはL3Outが存在する境界リーフノードによって適用されます。

入力強制と出力強制



ツール

ポリシードロップの識別に役立つさまざまなツールとコマンドがあります。ポリシーのドロップ

は、契約設定またはその欠如によるパケットのドロップと定義できます。

ゾーニング・ルールの検証

次のツールとコマンドを使用して、完了した契約コンシューマ/プロバイダー関係の結果としてリーフスイッチにプログラムされているゾーン分割ルールを明示的に検証できます。

「show zoning-rules」

すべてのゾーニング・ルールを表示するスイッチ・レベルのコマンド。

```
leaf# show zoning-rule
+---------+--------+--------+----------+----------------+---------+----------+-----------------
+----------+----------------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |      Dir       |  operSt |  Scope   |       Name
|  Action  |          Priority          |
+---------+--------+--------+----------+----------------+---------+----------+-----------------
+----------+----------------------------+
|   4156  |   25   | 16410  |   425    | uni-dir-ignore | enabled | 2818048  | external_to_ntp
|  permit  |       fully_qual(7)        |
|   4131  | 16410  |   25   |   424    |     bi-dir     | enabled | 2818048  | external_to_ntp
|  permit  |       fully_qual(7)        |
+---------+--------+--------+----------+----------------+---------+----------+-----------------
+----------+----------------------------+
```

「show zoning-filter」

ゾーン分割ルールが実行されているスポーツ/ポート情報を含むフィルタ。フィルタプログラミングは、次のコマンドで確認できます。

```
leaf# show zoning-filter
+----------+----------+-------------+-------------+-------------+----------+-------------+------
-------+-------------+-------------+----------+
| FilterId |   Name   |    EtherT   |     Prot    | ApplyToFrag | Stateful |  SFromPort  |
SToPort   |  DFromPort  |   DToPort   |   Prio   |
+----------+----------+-------------+-------------+-------------+----------+-------------+------
-------+-------------+-------------+----------+
| implarp  | implarp  |     arp     | unspecified |      no     |    no    | unspecified |
unspecified | unspecified | unspecified |  dport   |
| implicit | implicit | unspecified | unspecified |      no     |    no    | unspecified |
unspecified | unspecified | unspecified | implicit |
|   425    |  425_0   |      ip     |     tcp     |      no     |    no    |     123     |
123     | unspecified | unspecified |  sport   |
|   424    |  424_0   |      ip     |     tcp     |      no     |    no    | unspecified |
unspecified |     123     |     123     |  dport   |
+----------+----------+-------------+-------------+-------------+----------+-------------+------
-------+-------------+-------------+----------+
```

'show system internal policy-mgr stats'

このコマンドを実行すると、ゾーン分割ルールごとのヒット数を確認できます。これは、より高い優先度を持つ可能性がある暗黙の廃棄ルールなど、予期されたルールが他のルールではなくヒットされているかどうかを判断するのに役立ちます。

```
leaf# show system internal policy-mgr stats
```

```
Requested Rule Statistics
Rule (4131) DN (sys/actrl/scope-2818048/rule-2818048-s-16410-d-25-f-424) Ingress: 0, Egress: 0,
Pkts: 0  RevPkts: 0
Rule (4156) DN (sys/actrl/scope-2818048/rule-2818048-s-25-d-16410-f-425) Ingress: 0, Egress: 0,
Pkts: 0  RevPkts: 0
```

「show logging ip access-list internal packet-log deny」

iBashレベルで実行できるスイッチレベルのコマンド。ACL（コントラクト）関連のドロップと、次のようなフロー関連情報を報告します。

- VRF
- VLAN-ID
- 送信元MAC/宛先MAC
- 送信元IP/宛先IP
- 送信元ポート/宛先ポート
- 送信元インターフェイス

```
leaf# show logging ip access-list internal packet-log deny
[ Tue Oct  1 10:34:37 2019 377572 usecs]: CName: Prod1:VRF1(VXLAN: 2654209), VlanType: Unknown,
Vlan-Id: 0, SMac: 0x000c0c0c0c0c, DMac:0x000c0c0c0c0c, SIP: 192.168.21.11, DIP: 192.168.22.11,
SPort: 0, DPort: 0, Src Intf: Tunnel7, Proto: 1, PktLen: 98
[ Tue Oct  1 10:34:36 2019 377731 usecs]: CName: Prod1:VRF1(VXLAN: 2654209), VlanType: Unknown,
Vlan-Id: 0, SMac: 0x000c0c0c0c0c, DMac:0x000c0c0c0c0c, SIP: 192.168.21.11, DIP: 192.168.22.11,
SPort: 0, DPort: 0, Src Intf: Tunnel7, Proto: 1, PktLen: 98
```

contract_parser

IDから名前検索を実行する際に、ゾーニングルール、フィルタ、ヒット統計を関連付ける出力を生成するデバイス上のPythonスクリプト。このスクリプトは、複数のステップからなるプロセスを単一のコマンドに変換し、特定のEPG/VRFまたは他のコントラクトに関連する値にフィルタリングできるという点で非常に便利です。

```
leaf# contract_parser.py
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]

[7:4131] [vrf:common:default] permit ip tcp tn-Prod1/ap-Services/epg-NTP(16410) tn-Prod1/l3out-
L3Out1/instP-extEpg(25) eq 123  [contract:uni/tn-Prod1/brc-external_to_ntp] [hit=0]
[7:4156] [vrf:common:default] permit ip tcp tn-Prod1/l3out-L3Out1/instP-extEpg(25) eq 123 tn-
Prod1/ap-Services/epg-NTP(16410)  [contract:uni/tn-Prod1/brc-external_to_ntp] [hit=0]
[12:4169] [vrf:common:default] deny,log any tn-Prod1/l3out-L3Out1/instP-extEpg(25) epg:any
[contract:implicit] [hit=0]
[16:4167] [vrf:common:default] permit any epg:any tn-Prod1/bd-Services(32789)
[contract:implicit] [hit=0]
```

パケット分類の検証

ELAM

転送の詳細を確認するために使用されるASICレベルのレポート。パケットがドロップされた場合は、ドロップの理由が示されます。このセクションに関連する理由は、SECURITY_GROUP_DENY（契約ポリシーのドロップ）です。

トリアージ

ELAMでエンドツーエンドのパケットフローを追跡できるAPIC上のPythonベースのユーティリティ。

ELAMアシスタントアプリケーション

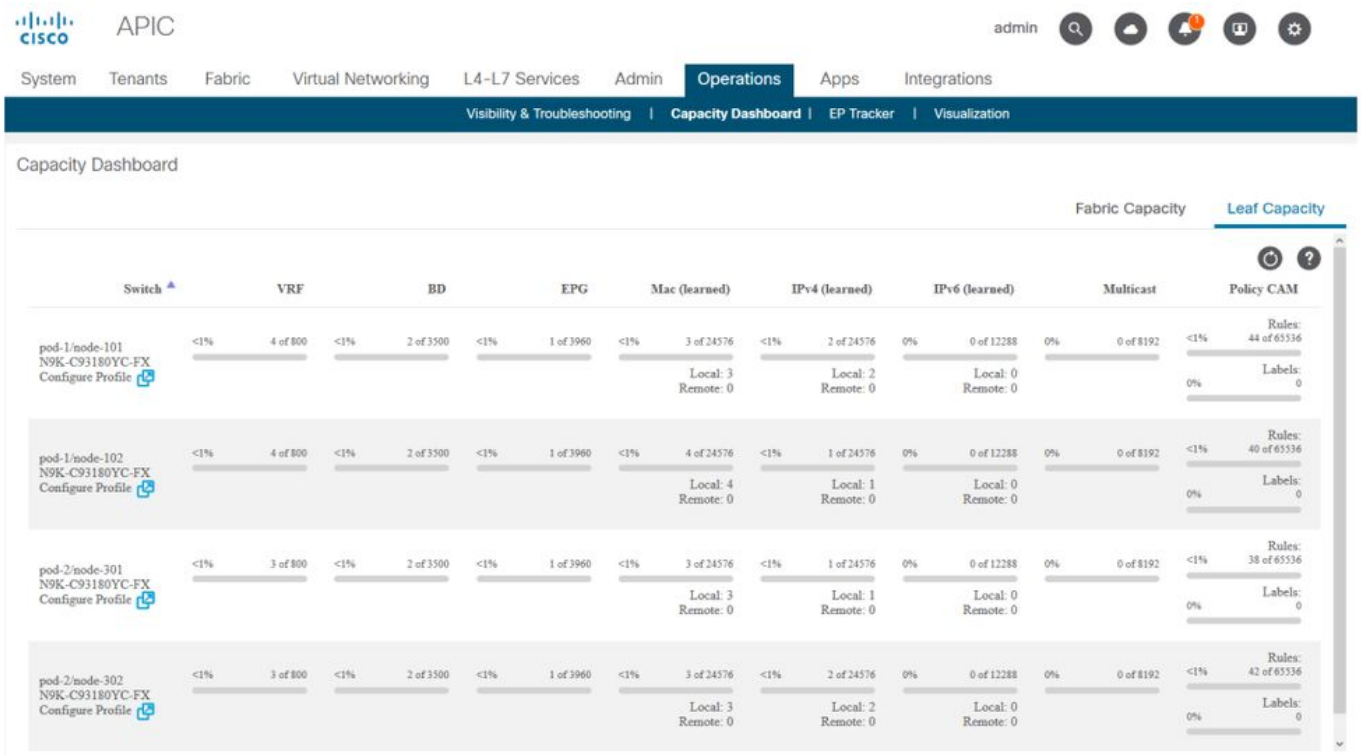
さまざまなASICの複雑さを抽象化し、転送決定検査をより便利で使いやすいものにするAPICアプリケーション。

ELAM、Triage、およびELAM Assistantツールの詳細については、「Intra-Fabric Forwarding」セクションを参照してください

ポリシーCAMの使用

リーフ単位のポリシーCAMの使用は、ファブリックが正常な状態であることを確認するために監視する重要なパラメータです。これを最も迅速に監視するには、GUI内で「Capacity Dashboard」を使用し、「Policy Cam」列を明示的にオンにします。

キャパシティダッシュボードの[Leaf Capacity]ビュー



'show platform internal hal health-stats'

このコマンドは、ポリシーCAMを含むさまざまなリソースの制限と使用状況を検証するのに役立ちます。このコマンドはvsh_lcでしか実行できないので、iBashから実行する場合は'-c'フラグを使用して渡してください。

```
leaf8# vsh_lc -c "show platform internal hal health-stats"
|Sandbox_ID: 0  Asic Bitmap: 0x0
|-----------------------------------
```

```
 ...
Policy stats:
=============
policy_count                  : 96
max_policy_count              : 65536
policy_otcam_count            : 175
max_policy_otcam_count            : 8192
policy_label_count                : 0
max_policy_label_count            : 0
=============
```

EPGからEPG

一般的なポリシードロップの考慮事項

2つのエンドポイント間の接続の問題をトラブルシューティングする方法は多数あります。次の方法は、接続の問題がポリシーのドロップ（契約によって引き起こされる）の結果であるかどうかを迅速かつ効果的に切り分ける**ための良**い出発点となります。

飛び込む前に尋ねるべき高度な質問：

- エンドポイントが同じEPGにあるか、異なるEPGにあるか。 異なるEPG（EPG間）に存在する2つのエンドポイント間のトラフィックは暗黙的に拒否され、通信を許可するにはコントラクトが必要です。同じEPG(intra-EPG)内の2つのエンドポイント間のトラフィックは、EPG内分離が使用されていない限り、暗黙的に許可されます。
- VRFは強制されますか、それとも非強制ですか。 VRFが**強制**モードの場合、VRF内では、2つの異なるEPG内のエンドポイントが通信するためにコントラクトが必要です。VRFが**非強制**モードの場合（VRF内）、適用されるACI契約に関係なく、VRF内のすべてのトラフィックは、非強制VRFに属する複数のEPGにわたってACIファブリックによって許可されます。

方法論

利用可能なさまざまなツールを使用して、影響を受けるフローに関してすでに認識されている情報のレベルに応じて、他のツールよりも適切で使いやすいツールがあります。

ACIファブリック内のパケットの完全なパス（入力リーフ、出力リーフなど）は既知ですか。

- 答えが「はい」の場合は、ELAM Assistantを使用して、送信元または宛先スイッチでのドロップの理由を特定する必要があります。
- 答えが「いいえ」の場合は、[Visibility & Troubleshooting]、[fTriage]、[contract_parser]、[Tenant]ビューの[Operational]タブ、およびiBashコマンドを使用して、パケットのパスを絞り込むか、ドロップの原因をより詳しく調べることができます。

fTriageツールについては、このセクションでは詳しく説明しません。このツールの使用の詳細については、「Intra-Fabric Forwarding」の章を参照してください。

Visibility & Troubleshootingは、2つのエンドポイント間でパケットがドロップされる場所をすばやく視覚化するのに役立ちますが、fTriageではさらに詳細なトラブルシューティング情報を表示できます。つまり、fTriageは、影響を受けるフローに関するインターフェイス、ドロップ理由、およびその他の低レベルの詳細を特定するのに役立ちます

.

このシナリオ例では、2つのエンドポイント間のポリシーのドロップをトラブルシューティングする方法を示します。192.168.21.11 および 192.168.23.11

これら2つのエンドポイント間でパケットのドロップが発生すると仮定し、次のトラブルシューティングワークフローを使用して問題の根本原因を特定します。

トラフィックフローに関係するsrc/dstリーフを特定します。

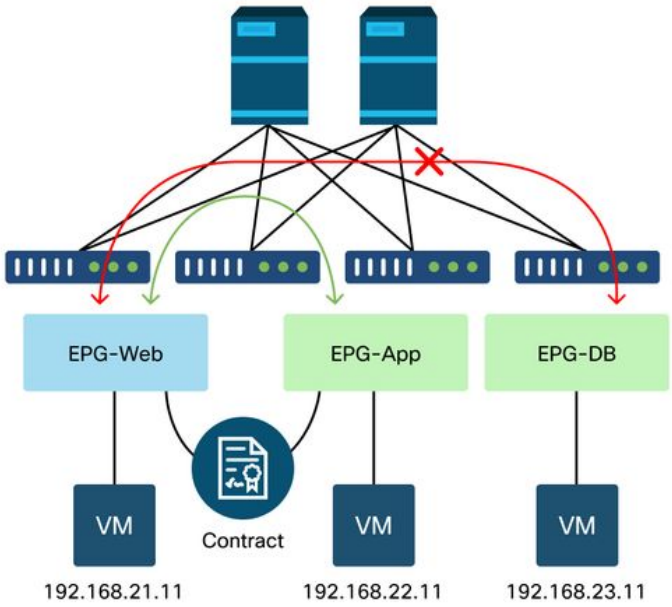
1. **Visibility & Troubleshooting**を使用してパケットフローをトレースし、パケットをドロップしているデバイスを特定します。
2. 選択したデバイスでコマンド「show logging ip access-list internal packet-log deny」を実行します。 対象のIPアドレスの1つを持つパケットが拒否され、ログに記録されている場合、**packet-log**はヒットごとに関連するエンドポイントとコントラクト名を出力します。
3. 送信元および宛先リーフでコマンド「contract_parser.py —vrf <tenant>:<VRF>」を使用して、設定されたコントラクトのヒットカウントを確認します。 パケットが送信元または宛先スイッチのコントラクトにヒットしている場合、関連するコントラクトのカウンタが増加しますこの方法は、多数のフローが同じルール（対象の2つのEPG間の多数のエンドポイント/フロー）にヒットする可能性がある状況でのIPアクセスリスト内部パケットログの方法よりも詳細ではありません。

上記の手順については、次の段落で詳しく説明します。

EPGからEPGへのトラブルシューティングシナリオの例

このシナリオ例では、2つのエンドポイント間のポリシーのドロップをトラブルシューティングする方法を示します。EPG-Webでは192.168.21.11、EPG-DBでは192.168.23.11です。

トポロジ

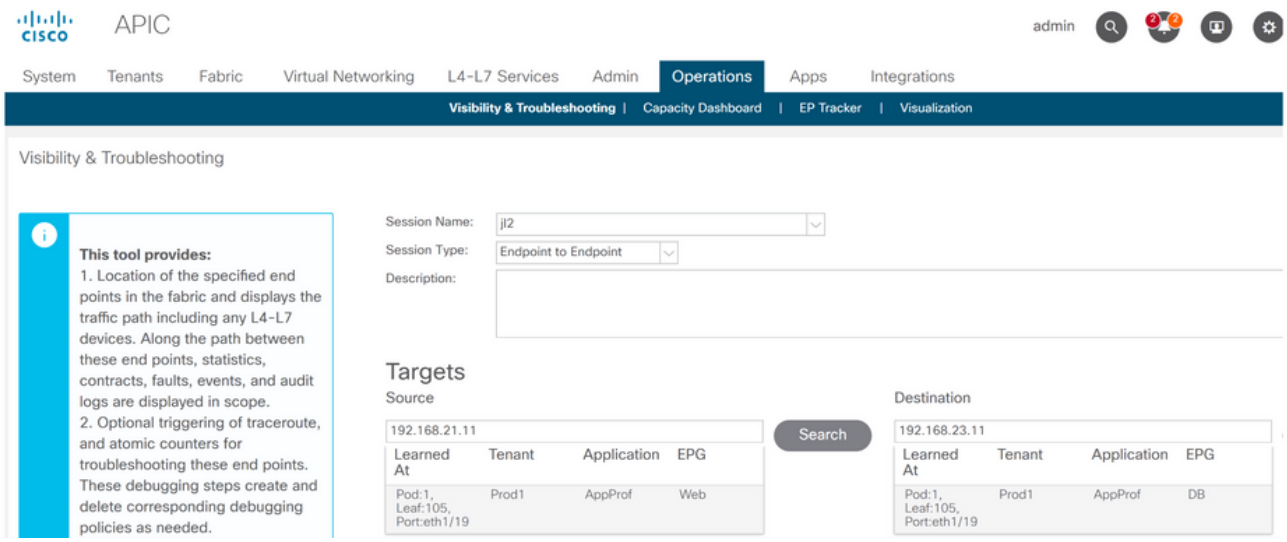


パケットドロップに関係する送信元および宛先リーフスイッチを特定する

可視性とトラブルシューティング

Visibility & Troubleshootingツールは、特定のEP-to-EPフローでパケットドロップが発生したスイッチを可視化し、パケットがドロップされた可能性のある場所を特定するのに役立ちます。

可視性とトラブルシューティングの設定

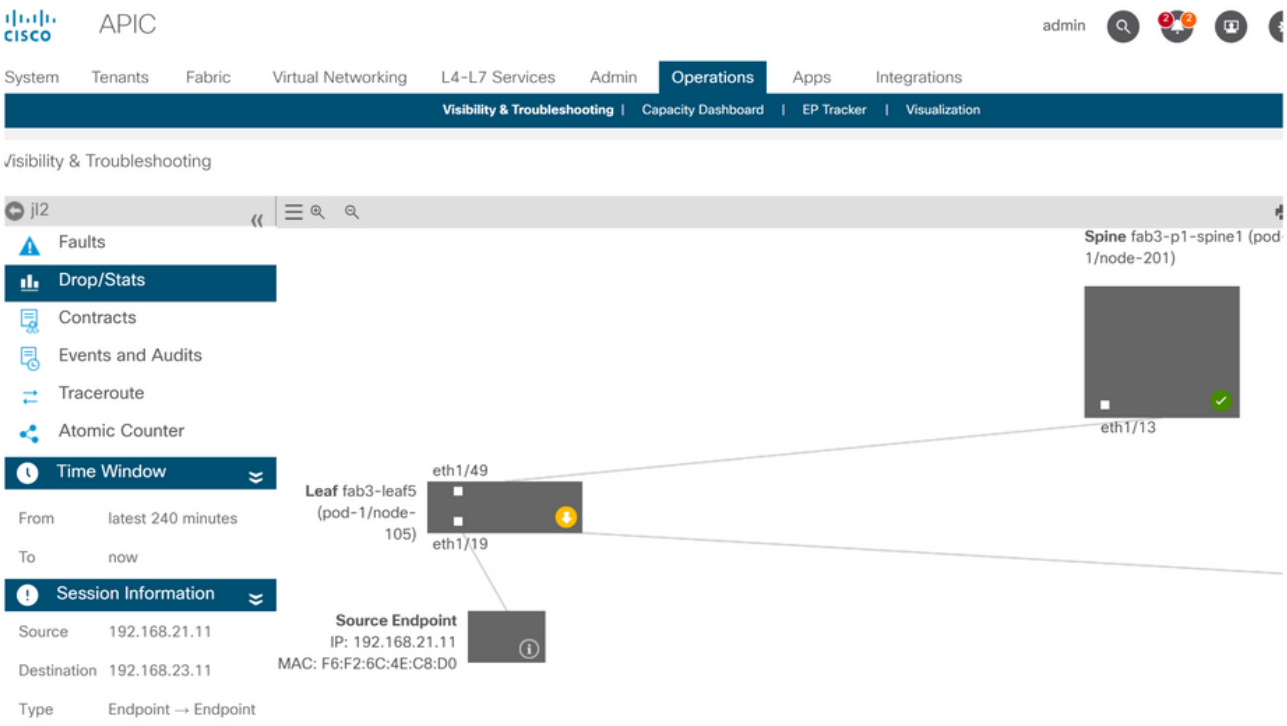


セッション名、送信元、および宛先エンドポイントを設定します。次に、[Submit]または[Generate Report]をクリックします。

このツールは、ファブリック内のエンドポイントを自動的に検出し、EPが属するテナント、アプリケーションプロファイル、およびEPGに関する情報を提供します。

この場合、EPがテナントProd1に属し、同じアプリケーションプロファイル「AppProf」に属し、異なるEPGに割り当てられていることが検出されます。「Web」と「DB」です。

ドロップID

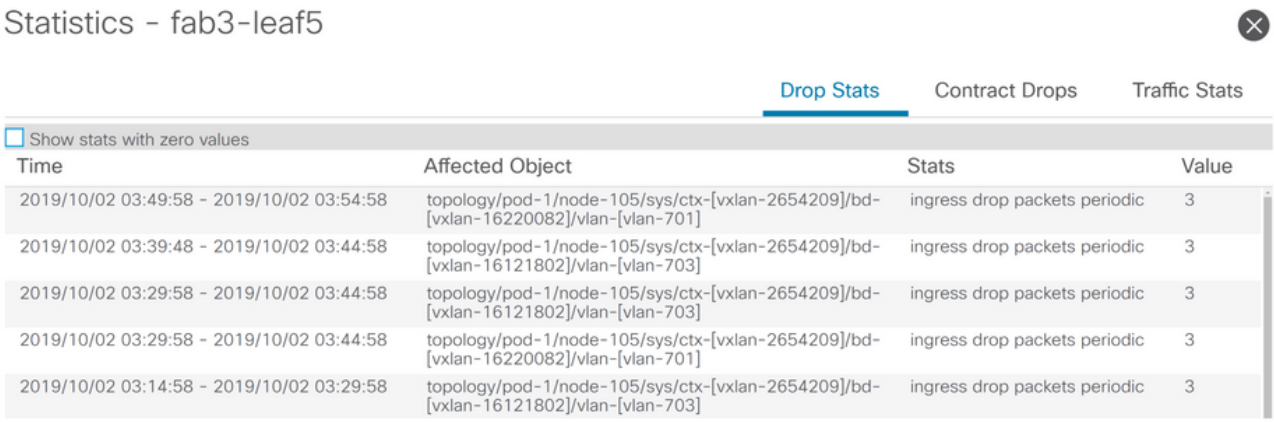


このツールは、トラブルシューティングシナリオのトポロジを自動的に視覚化します。この場合、2つのエンドポイントは同じリーフスイッチに接続されています。

[Drop/Stats]サブメニューに移動すると、対象のリーフまたはスパインの一般的なドロップを表示できます。関連するドロップについての詳細は、このマニュアルの「Intra-Fabric Forwarding」の章の「Interface Drops」の項を参照してください。

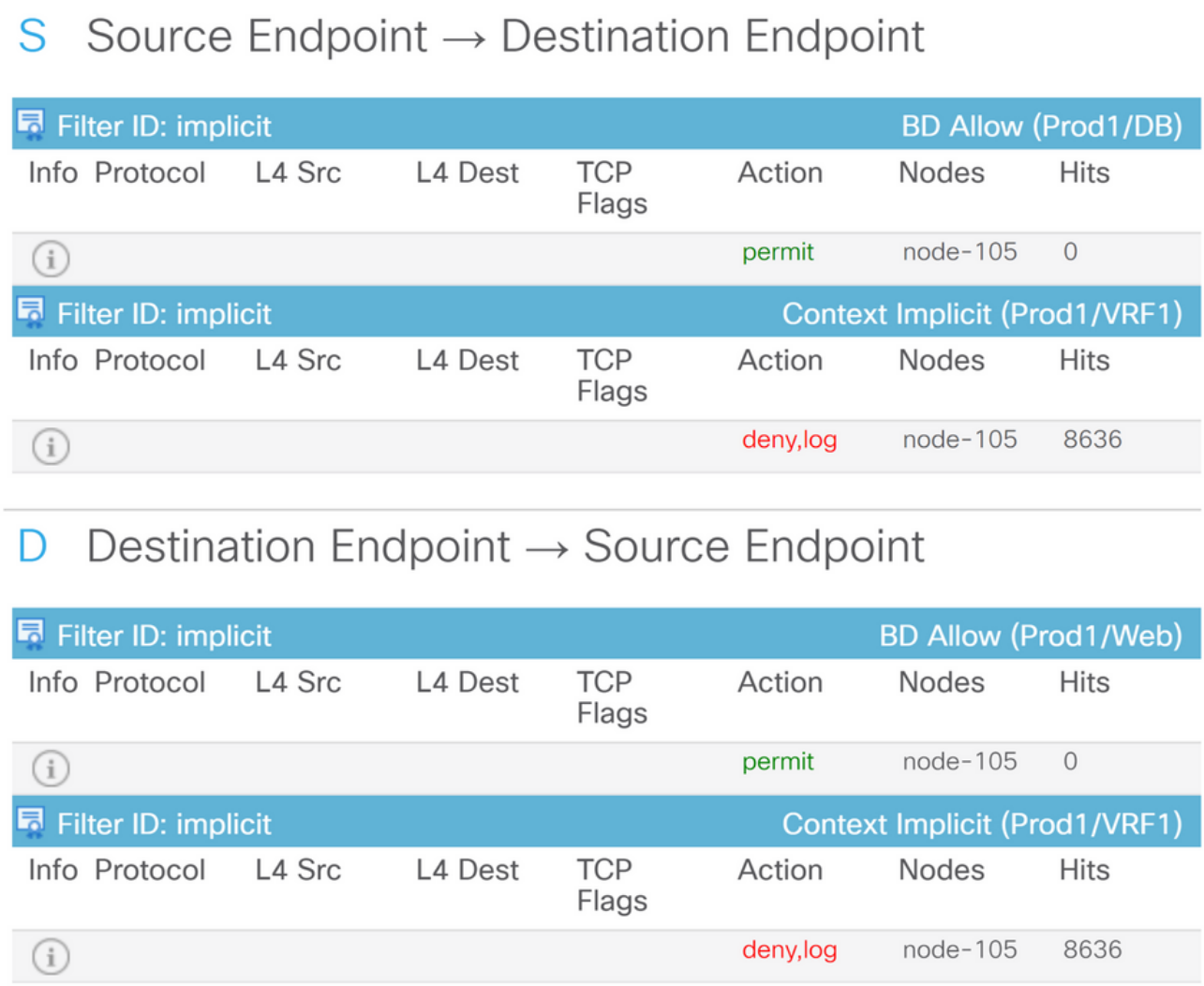
これらのドロップの多くは正常な動作であり、無視できます。

ドロップの詳細



スイッチダイアグラム上の黄色の[Packets dropped]ボタンを使用してドロップの詳細にドリルダウンすると、ドロップされたフローの詳細を表示できます。

『Contract Details（契約の詳細）』



[Contracts]サブメニューに移動すると、EPG間でポリシーのドロップを引き起こしている契約を特定できます。この例では、一部のヒットを示すProd1/VRF1を暗黙的に拒否しています。これは、必ずしも指定されたフロー（192.168.21.11および192.168.23.11）がこの暗黙のdenyにヒットすることを意味するわけではありません。Hits of Context Implicit denyルールが増加している場合は、Prod1/DBとProd1/Webの間に、どのコントラクトにもヒットしないトラフィックが存在することを意味するため、Implicit denyによってドロップされます。

[Tenant]の[Application Profile Topology]ビューで、左側の[Application Profile]名を選択し、[Topology]を選択すると、DB EPGに適用されている契約を確認できます。この場合、契約はEPGに割り当てられていません。

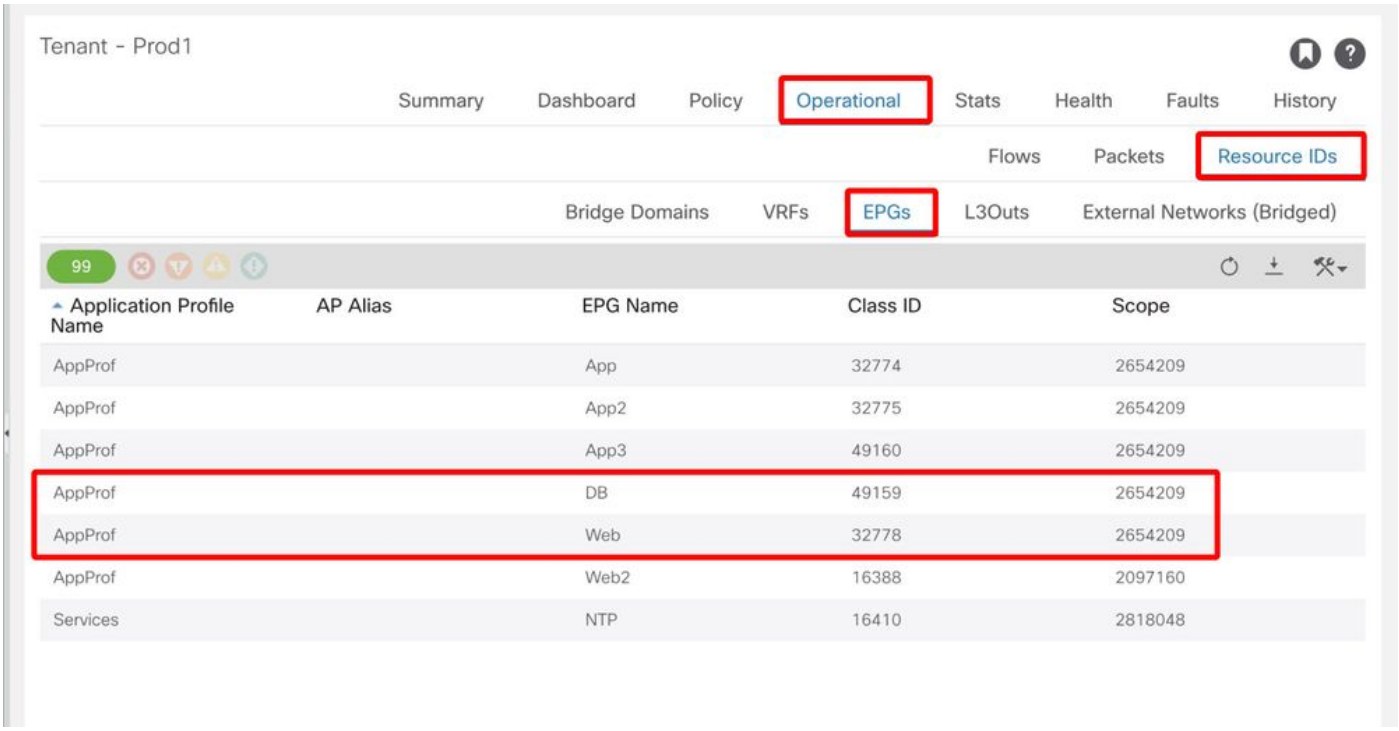
契約の可視化

これで、送信元と宛先のEPGがわかったので、次のような他の関連情報を特定することもできます。

- 該当するエンドポイントのsrc/dst **EPG pcTag**。pcTagは、ゾーン分割ルールを使用してEPGを識別するために使用されるクラスIDです。
- 該当するエンドポイントのsrc/dst **VRFVNID(スコープ**とも呼ばれる)。

クラスIDとスコープは、APIC GUIから簡単に取得できます。これを行うには、[Tenant]を開き、左側で[Tenant name]を選択し、[Operational] > [Resource IDs] > [EPGs]を選択します

EPG pcTagとスコープを検索するテナントリソースID



この場合、クラスIDとスコープは次のとおりです。

- Web EPG pcTag 32778
- Web EPGスコープ2654209
- DB EPG pcTag 49159
- DB EPGスコープ2654209

トラブルシューティング中のトラフィックフローに適用されるポリシーを確認する

iBash

ACIリーフでドロップされたパケットを確認する興味深いツールは、iBashコマンドラインです。'show logging ip access-list internal packet-log deny':

```
leaf5# show logging ip access-list internal packet-log deny | grep 192.168.21.11
[2019-10-01T14:25:44.746528000+09:00]: CName: Prod1:VRF1(VXLAN: 2654209), VlanType: FD_VLAN,
Vlan-Id: 114, SMac: 0xf6f26c4ec8d0, DMac:0x0022bdf819ff, SIP: 192.168.21.11, DIP: 192.168.23.11,
SPort: 0, DPort: 0, Src Intf: Ethernet1/19, Proto: 1, PktLen: 126
[2019-10-01T14:25:44.288653000+09:00]: CName: Prod1:VRF1(VXLAN: 2654209), VlanType: FD_VLAN,
Vlan-Id: 116, SMac: 0x3e2593f0eded, DMac:0x0022bdf819ff, SIP: 192.168.23.11, DIP: 192.168.21.11,
SPort: 0, DPort: 0, Src Intf: Ethernet1/19, Proto: 1, PktLen: 126
```

上記の出力からわかるように、リーフスイッチでは、EP 192.168.23.11から192.168.21.11に送信された多数のICMPパケットが廃棄されています。

contract_parserツールは、エンドポイントが関連付けられているVRFに適用される実際のポリシーを確認するのに役立ちます。

```
leaf5# contract_parser.py --vrf Prod1:VRF1
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]
```

```
[7:5159] [vrf:Prod1:VRF1] permit ip tcp tn-Prod1/ap-App1/epg-App(32771) eq 5000 tn-Prod1/ap-
App1/epg-Web(32772)  [contract:uni/tn-Prod1/brc-web_to_app] [hit=0]
[7:5156] [vrf:Prod1:VRF1] permit ip tcp tn-Prod1/ap-App1/epg-Web(32772) tn-Prod1/ap-App1/epg-
App(32771) eq 5000  [contract:uni/tn-Prod1/brc-web_to_app] [hit=0]
[16:5152] [vrf:Prod1:VRF1] permit any epg:any tn-Prod1/bd-Web(49154) [contract:implicit] [hit=0]
[16:5154] [vrf:Prod1:VRF1] permit arp epg:any epg:any [contract:implicit] [hit=0]
[21:5155] [vrf:Prod1:VRF1] deny,log any epg:any epg:any [contract:implicit] [hit=38,+10]
[22:5153] [vrf:Prod1:VRF1] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit] [hit=0]
```

これは、スイッチによって適用されるポリシーをリーフにプログラムされたゾーン分割ルールを使用して確認することもできます。

```
leaf5# show zoning-rule scope 2654209
+---------+--------+--------+----------+----------------+---------+----------+------------+-----
-----+-------------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |      Dir       |  operSt |  Scope   |    Name    |
Action  |         Priority        |
+---------+--------+--------+----------+----------------+---------+----------+------------+-----
-----+-------------------------+
|   5155  |   0    |   0    | implicit |    uni-dir     | enabled | 2654209  |            |
deny,log |     any_any_any(21)     |
|   5159  | 32771  | 32772  |   411    | uni-dir-ignore | enabled | 2654209  | web_to_app |
permit  |      fully_qual(7)      |
|   5156  | 32772  | 32771  |   410    |     bi-dir     | enabled | 2654209  | web_to_app |
permit  |      fully_qual(7)      |
+---------+--------+--------+----------+----------------+---------+----------+------------+-----
-----+-------------------------+
```

Visibility & Troubleshootingツール、contract_parserツール、およびゾーン分割ルールですでに確認したように、トラブルシューティングでは送信元EPGと宛先EPGの間に契約が存在しないことが出力から確認できます。ドロップされたパケットが暗黙のdenyルール5155と一致していると考えるのは簡単です。

ELAMキャプチャ

ELAMキャプチャは、転送の詳細を確認するために使用されるASICレベルのレポートを提供します。これは、ドロップされたパケットの場合に、ドロップの理由を示します。このシナリオのように、ドロップの理由がポリシーのドロップである場合、ELAMキャプチャの出力は次のようになります。

ELAMキャプチャの設定の詳細については、この章では説明しません。「Intra-Fabric Forwarding」の章を参照してください。

```
leaf5# vsh_lc
module-1# debug platform internal tah elam asic 0
module-1(DBG-elam)# trigger init in-select 6 out-select 0
module-1(DBG-elam)# trigger reset
module-1(DBG-elam-insel6)# set outer ipv4 src_ip 192.168.21.11 dst_ip 192.168.23.11
module-1(DBG-elam-insel6)# start
module-1(DBG-elam-insel6)# status

ELAM STATUS
===========
Asic 0 Slice 0 Status Triggered
Asic 0 Slice 1 Status Armed

module-1(DBG-elam-insel6)# ereport | grep reason
RW drop reason                          : SECURITY_GROUP_DENY
LU drop reason                          : SECURITY_GROUP_DENY
```

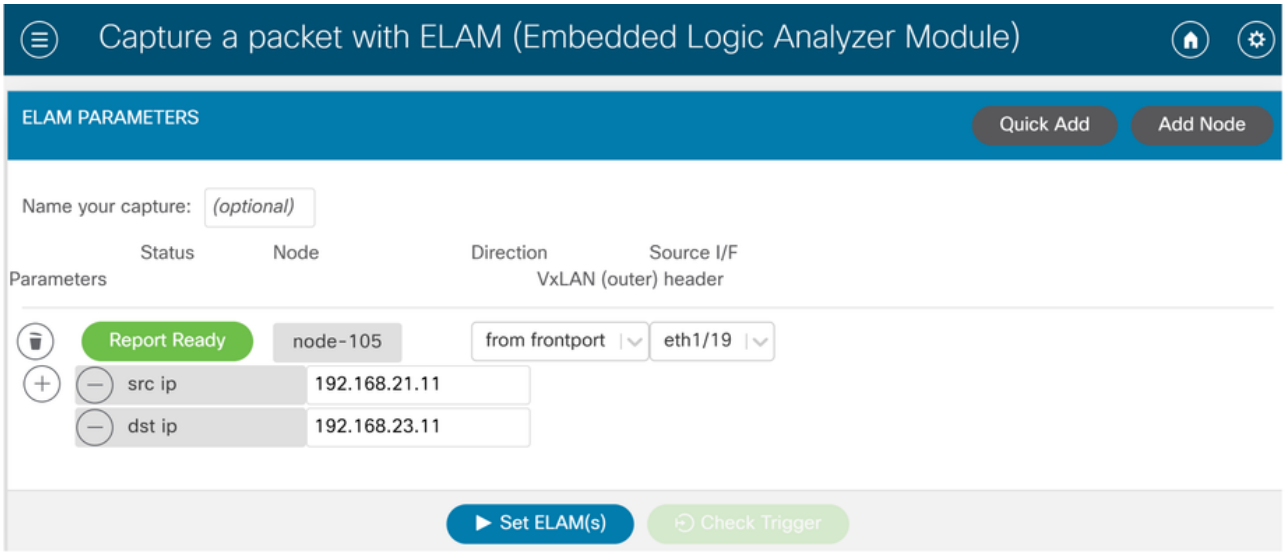
```
pkt.lu_drop_reason: 0x2D
```

上記のELAMレポートは、パケットがポリシーのドロップによってドロップされたことを明確に示しています。'SECURITY_GROUP_DENY'

ELAM Assistant:

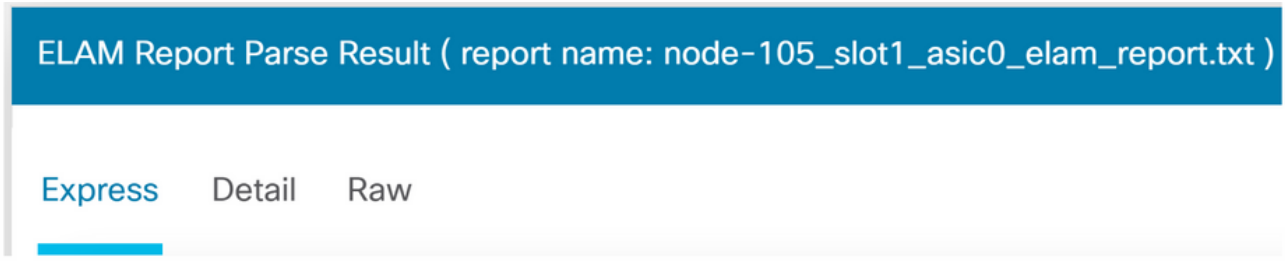
ELAMキャプチャの結果は、APIC GUIのELAM Assistant Appでも同様です。

コンフィギュレーション



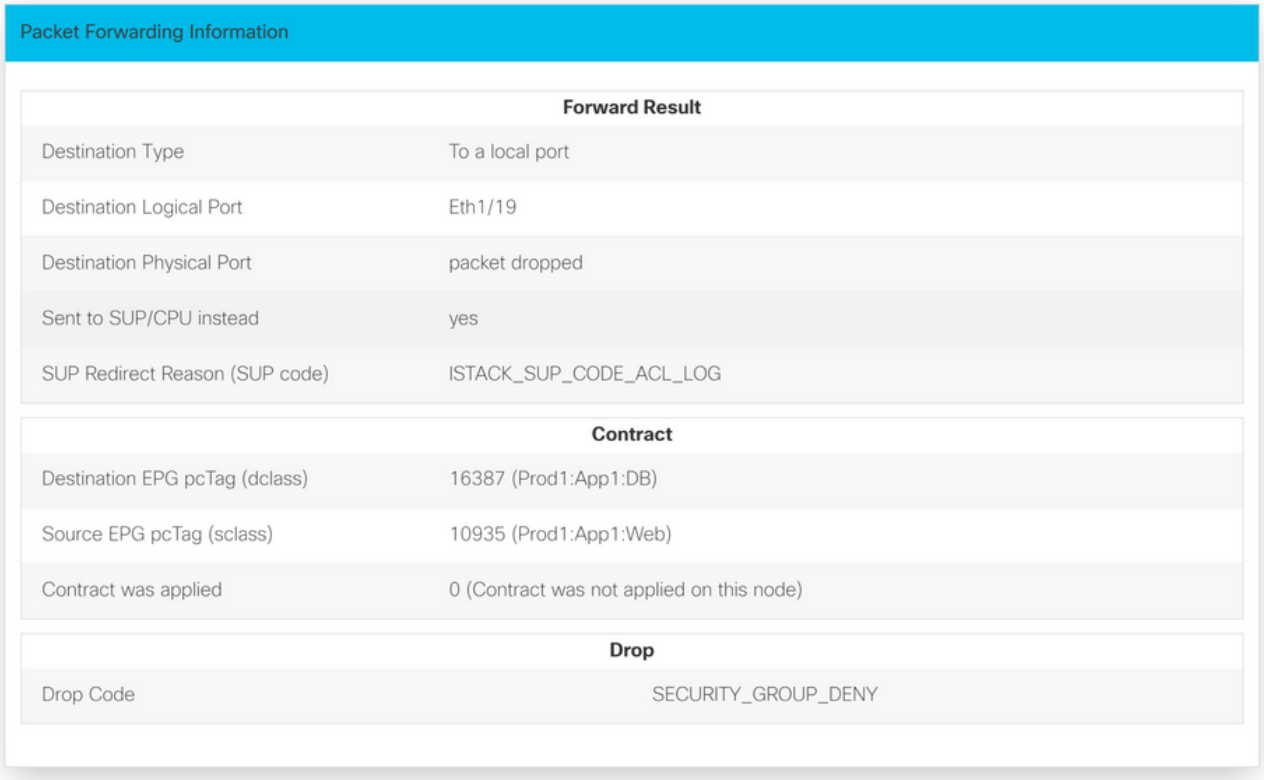
通常、ユーザは対象のフローの送信元と宛先の両方の詳細を設定します。この例では、送信元IPは、送信元EPGとの契約関係がない宛先EPGのエンドポイントへのトラフィックをキャプチャするために使用されます。

Elam Assistant Expressレポート



ELAM Assistantで表示できる出力レベルは3つあります。Express、Detail、Rawです。

Elam Assistant Expressレポート（続き）



Express Resultの下のDrop Code reason SECURITY_GROUP_DENYは、ドロップが契約ヒットの結果であったことを示します。

優先グループ

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たVRFのEPGでは、次の2種類のポリシー適用を使用できます。

- 含まれるEPG:EPGは、契約優先グループにメンバーシップがある場合、契約なしで自由に相互に通信できます。これは、source-any-destination-any-permitデフォルトルールに基づいています。
- 除外されるEPG:優先グループのメンバーではないEPGは、相互に通信する契約を必要とします。それ以外の場合は、除外されたEPGと任意のEPGの間の拒否ルールが適用されます。

Contract Preferred Group機能により、VRF内のEPG間の通信をより細かく制御できます。VRF内のほとんどのEPGがオープン通信を行う必要があるが、他のEPGとの通信が制限されている場合は、契約の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て、EPG間通信をより正確に制御します。

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

契約優先グループのプログラミング

基本的に、契約優先グループは通常の契約の逆です。通常の契約の場合、明示的な許可ゾーン分割ルールは、VRFスコープの暗黙のdenyゾーン分割ルールでプログラムされます。優先グループでは、暗黙的なPERMITゾーン分割ルールが最も高い数値のプライオリティ値を使用してプログラムされ、特定のDENYゾーン分割ルールが、優先グループメンバーではないEPGからのトラフィックを許可しないようにプログラムされます。その結果、拒否ルールが最初に評価され、フロ

ーがこれらのルールに一致しない場合、フローは暗黙的に許可されます。

優先グループ外のすべてのEPGに対して、常に明示的な拒否ゾーン分割ルールのペアがあります。

- 非優先グループメンバーから任意のpcTag（値0）への1つ。
- 任意のpcTag（値0）からPreferred Group以外のメンバへの別の値。

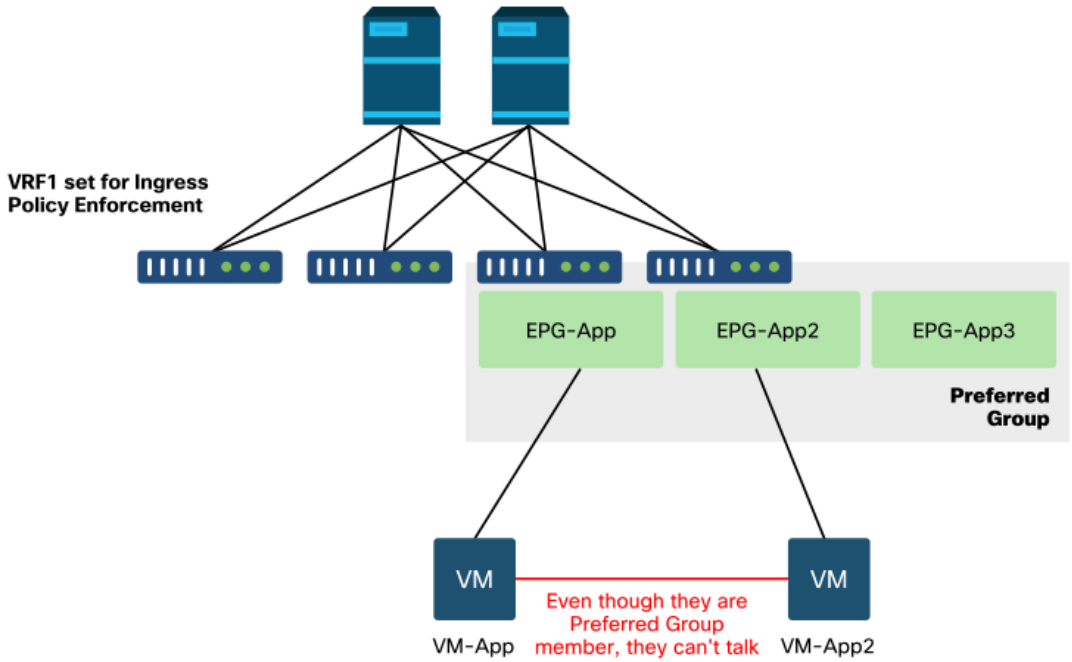
優先グループのトラブルシューティングシナリオ

次の図は、EPG App、App2、およびApp3がすべて優先グループメンバーとして設定されている論理トポロジを示しています。

VM-AppはEPG-Appの一部で、VM-App2はEPG-App2の一部です。AppとApp2の両方のEPGが優先される部分に含まれているので、自由に通信する必要があります。

VM-Appは、TCPポート6000からVM-App2へのトラフィックフローを開始します。EPG-AppとEPG-App2はどちらも、VRF1の一部として優先グループメンバーです。VM-App2はTCPポート6000でパケットを受信しません。

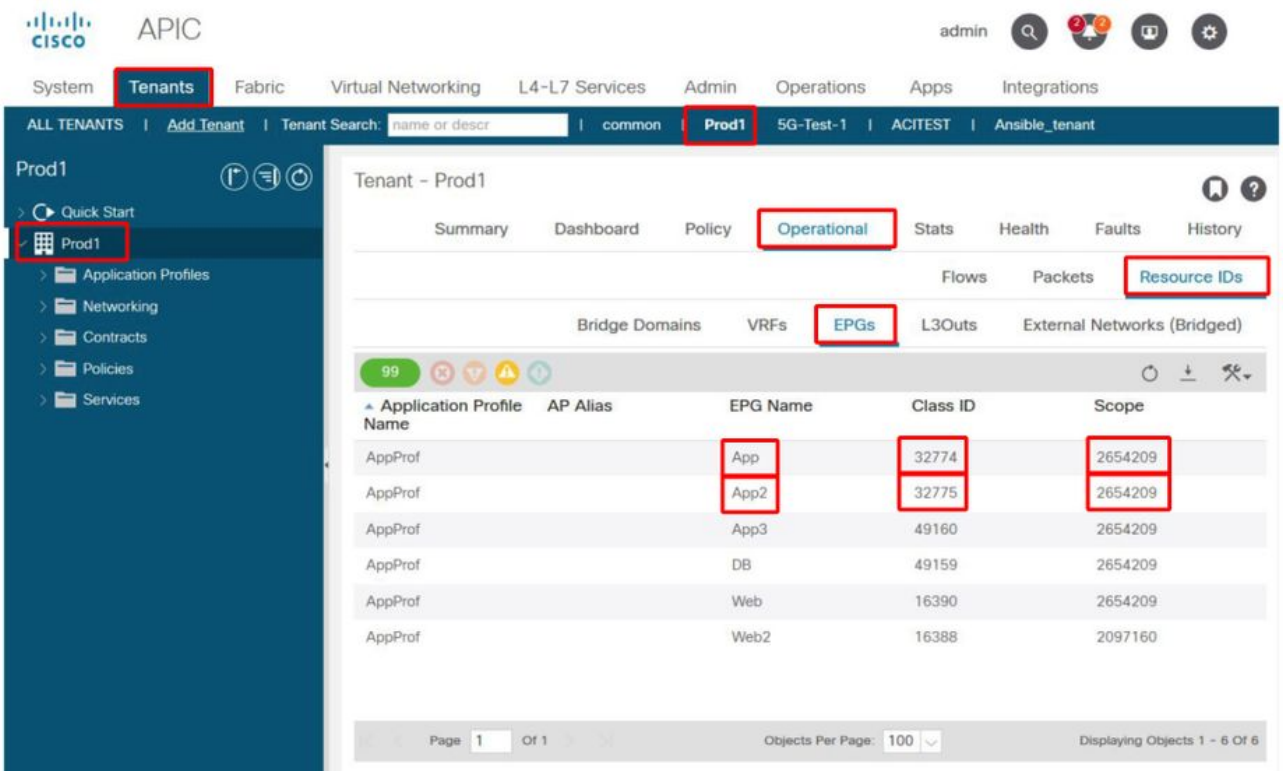
トポロジ



ワークフロー

1. EPG APPのpcTagとそのVRF VNID/Scopeを検索します。

EPGおよびVRF pcTag



2.入力リーフでcontract_parser.pyを使用してコントラクトのプログラミングを確認する

contract_parser.pyまたは「show zoning-rule」コマンドを使用して、VRFを指定します

```
fab3-leaf8# show zoning-rule scope 2654209
+---------+--------+--------+----------+---------+---------+---------+------+----------+--------
--------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |   Dir   |  operSt |  Scope  | Name |  Action
|          Priority          |
+---------+--------+--------+----------+---------+---------+---------+------+----------+--------
--------------------+
|   4165  |   0    |   0    | implicit | uni-dir | enabled | 2654209 |      |  permit  |
grp_any_any_any_permit(20) |
|   4160  |   0    |   0    | implarp  | uni-dir | enabled | 2654209 |      |  permit  |
any_any_filter(17)      |
|   4164  |   0    |   15   | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_any_dest_any_deny(19)  |
|   4176  |   0    | 16386  | implicit | uni-dir | enabled | 2654209 |      |  permit  |
any_dest_any(16)       |
|   4130  | 32770  |   0    | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_src_any_any_deny(18)   |
|   4175  | 49159  |   0    | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_src_any_any_deny(18)   |
|   4129  |   0    | 49159  | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_any_dest_any_deny(19)  |
|   4177  | 32778  |   0    | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_src_any_any_deny(18)   |
|   4128  |   0    | 32778  | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_any_dest_any_deny(19)  |
|   4178  | 32775  |   0    | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_src_any_any_deny(18)   |
|   4179  |   0    | 32775  | implicit | uni-dir | enabled | 2654209 |      | deny,log |
grp_any_dest_any_deny(19)  |
+---------+--------+--------+----------+---------+---------+---------+------+----------+--------
--------------------+

fab3-leaf8# contract_parser.py --vrf Prod1:VRF1
```

```
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]
[16:4176] [vrf:Prod1:VRF1] permit any epg:any tn-Prod1/bd-App(16386) [contract:implicit] [hit=0]
[16:4160] [vrf:Prod1:VRF1] permit arp epg:any epg:any [contract:implicit] [hit=0]
[18:4130] [vrf:Prod1:VRF1] deny,log any tn-Prod1/vrf-VRF1(32770) epg:any [contract:implicit]
[hit=?]
[18:4178] [vrf:Prod1:VRF1] deny,log any epg:32775 epg:any [contract:implicit] [hit=?]
[18:4177] [vrf:Prod1:VRF1] deny,log any epg:32778 epg:any [contract:implicit] [hit=?]
[18:4175] [vrf:Prod1:VRF1] deny,log any epg:49159 epg:any [contract:implicit] [hit=?]
[19:4164] [vrf:Prod1:VRF1] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit] [hit=0]
[19:4179] [vrf:Prod1:VRF1] deny,log any epg:any epg:32775 [contract:implicit] [hit=?]
[19:4128] [vrf:Prod1:VRF1] deny,log any epg:any epg:32778 [contract:implicit] [hit=?]
[19:4129] [vrf:Prod1:VRF1] deny,log any epg:any epg:49159 [contract:implicit] [hit=?]
[20:4165] [vrf:Prod1:VRF1] permit any epg:any epg:any [contract:implicit] [hit=65]
```

上記の出力を調べると、プライオリティが最も高い20の暗黙のpermitエントリ(ruleId 4165)が確認できます。この暗黙の許可ルールは、トラフィックフローを許可しない優先順位の低い明示的な拒否ルールがない限り、すべてのトラフィックフローを許可します。

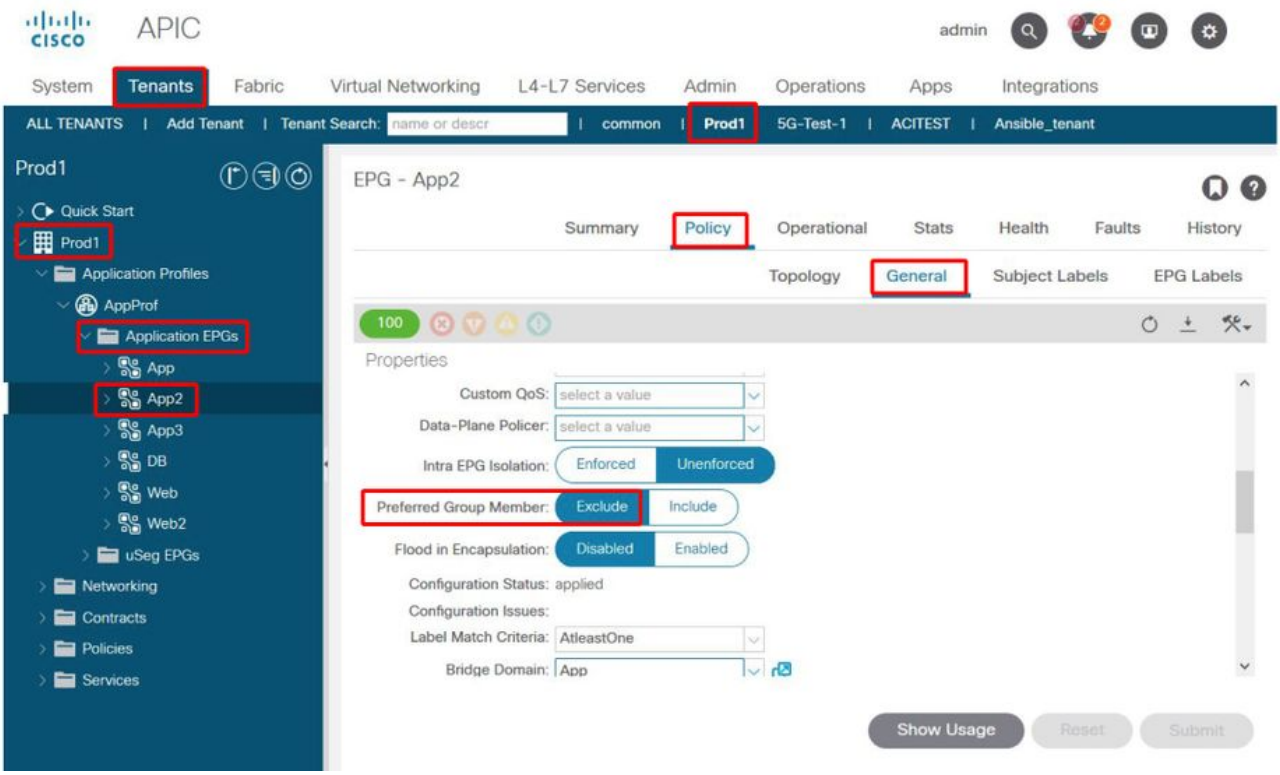
さらに、EPG App2のpcTagであるpcTag 32775に対して2つの明示的な拒否ルールが観察されています。これら2つの明示的な拒否ゾーン設定ルールは、EPGからEPG App2へのトラフィックを許可しません（逆も同様）。これらのルールの優先順位は18と19であるため、デフォルトの許可ルールよりも優先されます。

結論は、明示的な拒否ルールが観察されるため、EPG App2は優先グループメンバーではないということです。

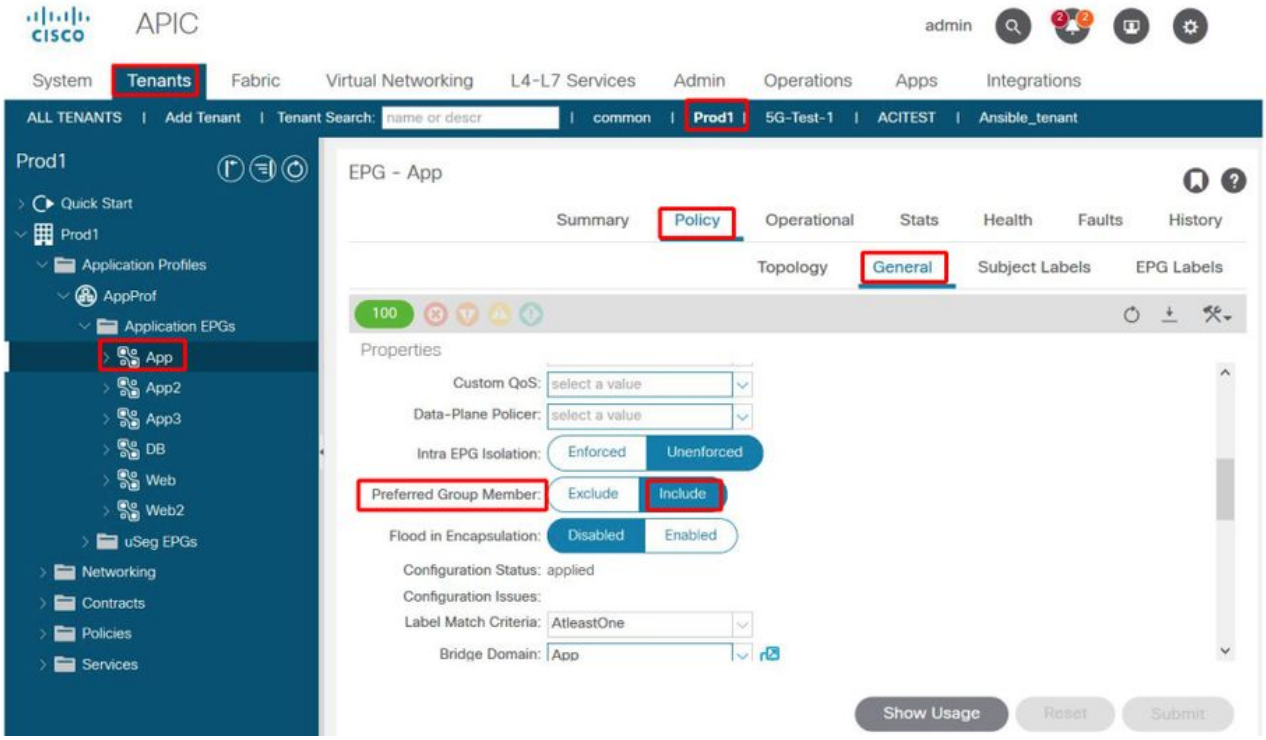
3. EPG優先グループメンバー設定の確認

APIC GUIに移動し、[EPG App2]および[EPG App Preferred Group Member Configuration]をオンにします。次の図で、「EPG App2 is not configured as a Preferred Group Member」を参照してください。

EPG App2：優先グループメンバー設定を除外



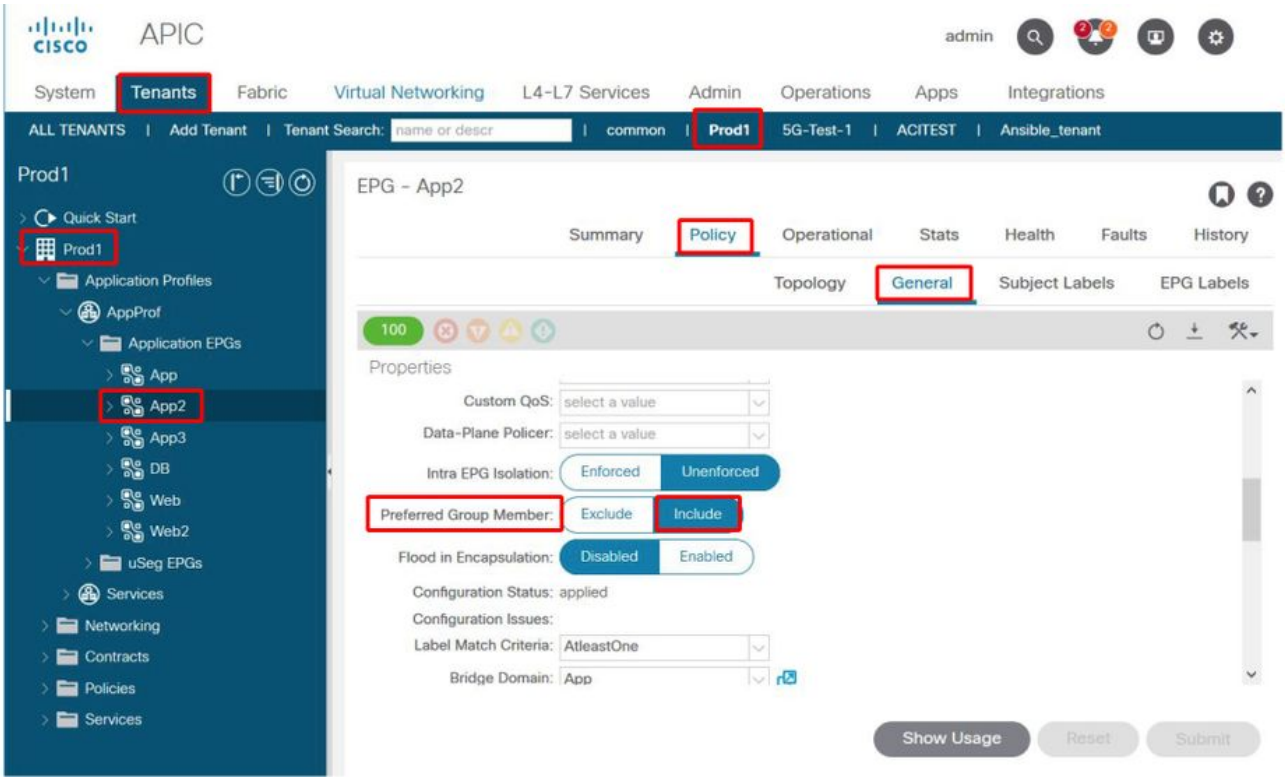
EPGアプリ – 優先グループメンバー設定が含まれています



4. EPG App2を優先グループメンバーとして設定します

App2 EPGの設定を変更すると、優先グループが優先グループの一部として自由に通信できるようになります。

EPG App2：優先グループメンバー設定を含む



5. src EPが存在するリーフ上でcontract_parser.pyを使用して契約プログラミングを再確認する

contract_parser.pyを再度使用し、VRF名を指定して、EPG App2の明示的な拒否ルールが削除さ

れたかどうかを確認します。

```
fab3-leaf8# contract_parser.py --vrf Prod1:VRF1
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]
[16:4176] [vrf:Prod1:VRF1] permit any epg:any tn-Prod1/bd-App(16386) [contract:implicit] [hit=0]
[16:4160] [vrf:Prod1:VRF1] permit arp epg:any epg:any [contract:implicit] [hit=0]
[18:4175] [vrf:Prod1:VRF1] deny,log any epg:16390 epg:any [contract:implicit] [hit=0]
[18:4167] [vrf:Prod1:VRF1] deny,log any epg:23 epg:any [contract:implicit] [hit=0]
[18:4156] [vrf:Prod1:VRF1] deny,log any tn-Prod1/vrf-VRF1(32770) epg:any [contract:implicit]
[hit=0]
[18:4168] [vrf:Prod1:VRF1] deny,log any epg:49159 epg:any [contract:implicit] [hit=0]
[19:4164] [vrf:Prod1:VRF1] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit] [hit=0]
[19:4169] [vrf:Prod1:VRF1] deny,log any epg:any epg:16390 [contract:implicit] [hit=0]
[19:4159] [vrf:Prod1:VRF1] deny,log any epg:any epg:23 [contract:implicit] [hit=0]
[19:4174] [vrf:Prod1:VRF1] deny,log any epg:any epg:49159 [contract:implicit] [hit=0]
[20:4165] [vrf:Prod1:VRF1] permit any epg:any epg:any [contract:implicit] [hit=65]
```

上記の出力では、EPG App2とそのpcTag 32775の明示的な拒否ルールは表示されなくなりました。つまり、EPGアプリケーションとEPGアプリケーション2のEP間のトラフィックは、最高の優先度20の暗黙の許可ルール(ruleId 4165)に一致します。

vzAnyからEPG

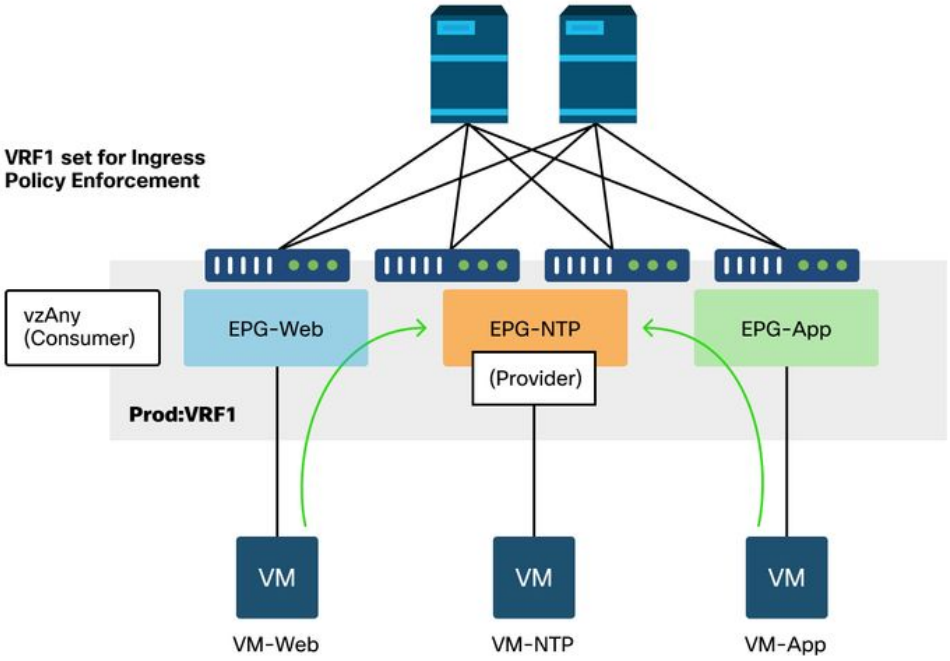
vzAnyについて

1つまたは複数のEPG間の契約を設定する場合、契約は消費または提供された関係として設定できます。EPGの数が増えると、EPG間の契約関係も増えます。一般的な使用例の中には、すべてのEPGが別の特定のEPGとトラフィックフローを交換する必要があるものがあります。このような使用例としては、同じVRF内の他のすべてのEPGで使用する必要があるサービスを提供するEPを含むEPGが考えられます（NTPやDNSなど）。vzAnyを使用すると、すべてのEPGと、他のすべてのEPGによって消費されるサービスを提供する特定のEPGとの間の契約関係を設定する際の運用オーバーヘッドを低減できます。さらに、vzAnyでは、各vzAny契約関係に対して2つのゾーン分割ルールのみが追加されるため、リーフスイッチでより効率的なセキュリティポリシーCAMを使用できます。

使用例

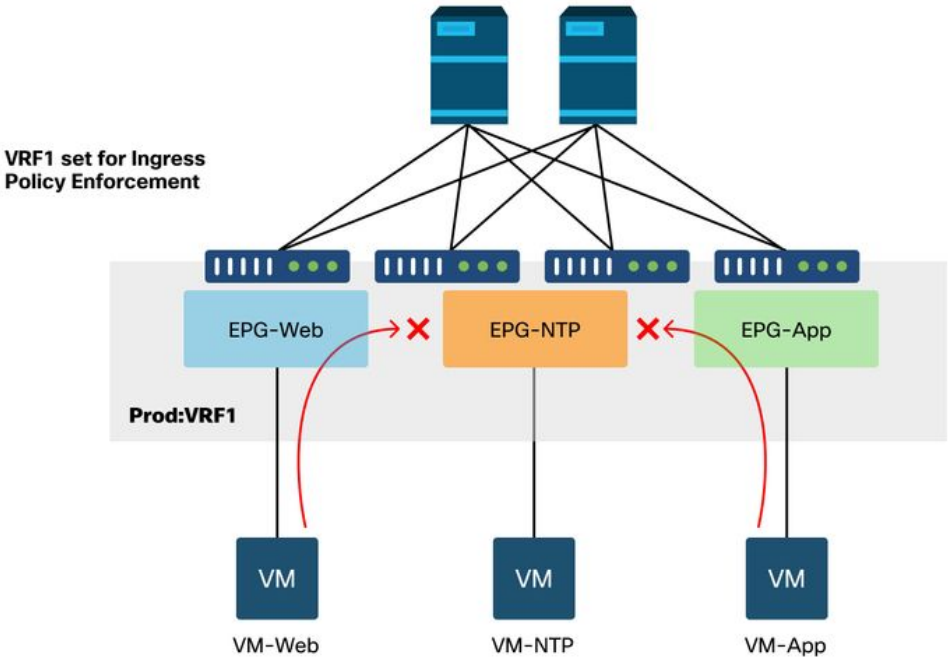
次の図は、EPGのWebとAppのVM-WebとVM-AppがそれぞれEPG-NTPのVM-NTPからNTPサービスを利用する必要がある使用例を示しています。vzAnyを使用すると、EPG NTPで指定されたコントラクトを設定し、その後EPGのWebおよびアプリケーションで使用されるコントラクトと同じコントラクトを持つのではなく、VRF Prod:VRF1の各EPGがEPG NTPからNTPサービスを使用できます。

vzAny:VRF Prod:VRF1内の任意のEPGは、EPG NTPからNTPサービスを使用できます



NTPサービスを消費するEPG間に契約がない場合にドロップが発生するシナリオを考えてみましょう。

トラブルシューティングシナリオ：契約がない場合のトラフィックドロップ

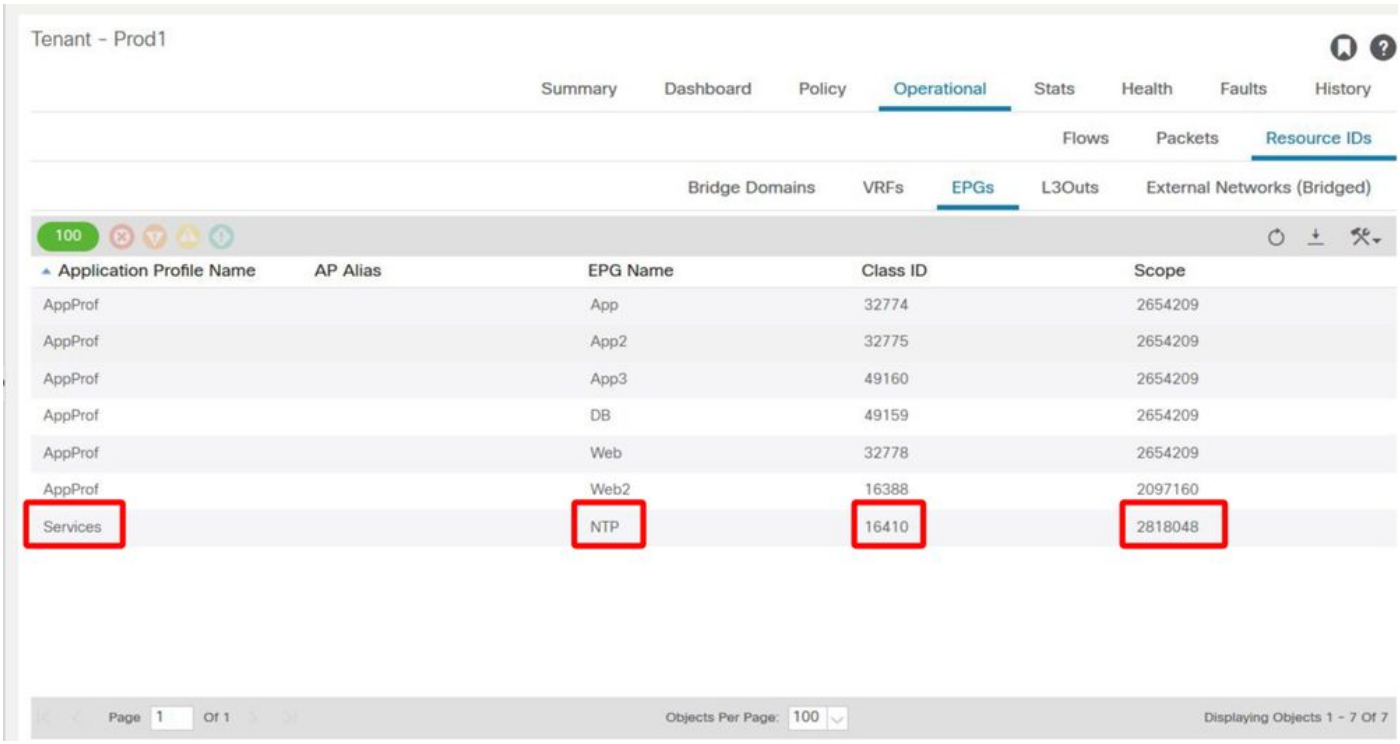


ワークフロー

1. EPG NTPのpcTagとそのVRF VNID/Scopeを検索します。

[Tenant] > [Operational] > [Resource IDs] > [EPGs]を選択すると、pcTagとスコープを検索できます

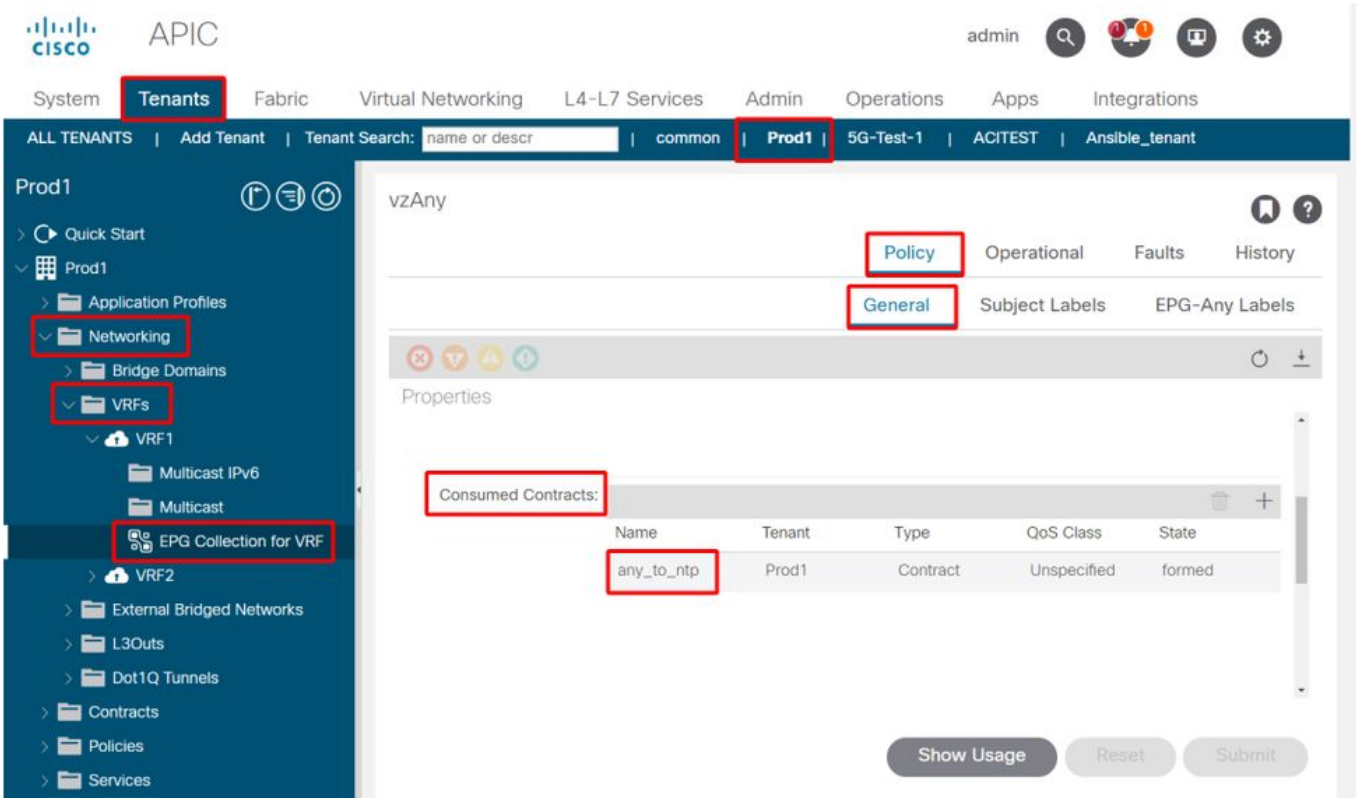
EPG NTP pcTagとそのVRF VNID/Scope



2.契約がvzとして設定されているかどうかを確認するVRFの一部として使用される契約

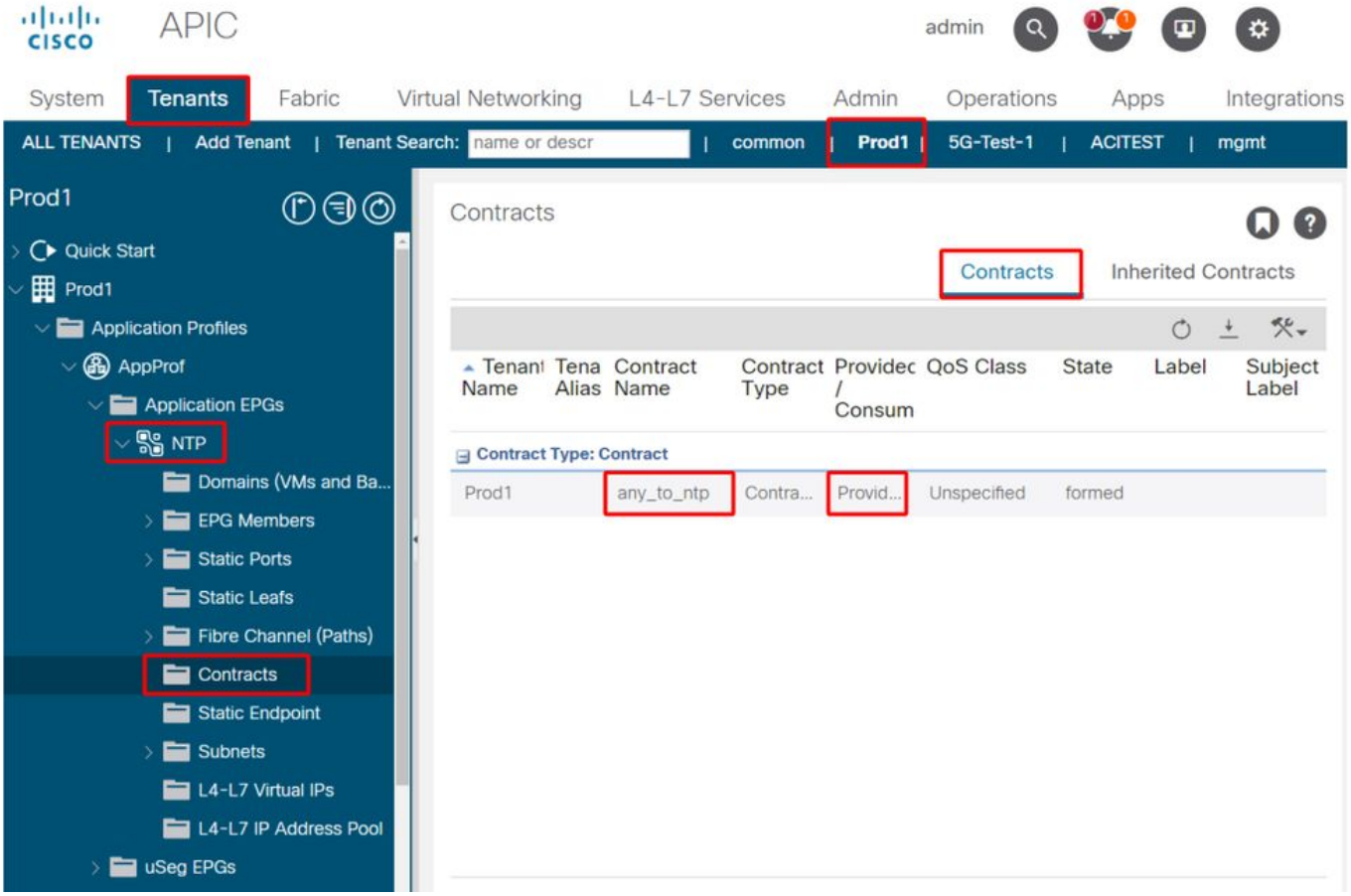
VRFに移動し、「VRFのEPGコレクション」の下にvzAnyとして設定されたコンシュームされたコントラクトがあるかどうかを確認します。

Contract configured as a consumed vzAny contract on the VRF



3. EPG NTPで指定されたコントラクトと同じコントラクトが適用されているかどうかを確認する

契約関係を確立するには、VRF内の他のEPGにNTPサービスを提供しているEPG NTP上の提供された契約と同じ契約を適用する必要があります。



4. contract_parser.pyまたは「show zoning-rule」を使用した入力リーフでのゾーン分割ルールの検証

入力リーフには、任意のEPGとEPG NTP間の双方向トラフィックフロー（コントラクトの対象が双方向に設定されている場合）を許可する2つのゾーン分割ルールが必要です。「任意のEPG」は、ゾーン分割ルールのプログラミングではpcTag 0と表記されます。

VRFを指定するときに入力リーフでcontract_parser.pyまたは「show zoning-rule」コマンドを使用すると、ゾーン分割ルールがプログラムされていることを確認できます。

VRF内の他のEPGからEPG NTPへのトラフィックを許可するゾーニングルール

contract_parser.pyおよび「show zoning-rule」を使用して、vzAnyベースのゾーン分割ルールの存在を確認します。

次の2種類のルールが明らかです。

1. ルール4156とルール4168は、AnyからNTPへの接続を許可し、その逆も許可します。プライオリティ13と14を持ちます。 任意のEPG(pcTag 0)からEPG NTP(pcTag 49161)へのトラフィックフローを許可するゾーニングルール。EPG NTP(pcTag 46161)から他のEPG(pcTag 0)へのトラフィックフローを許可するゾーン分割ルール。
2. 優先度21のany to any denyルール（デフォルト）であるルール4165。

優先順位が最も低い場合、VRFのすべてのEPGがNTP EPGにアクセスします。

```
fab3-leaf8# contract_parser.py --vrf Prod1:VRF
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]

[13:4156] [vrf:Prod1:VRF1] permit ip tcp tn-Prod1/ap-Services/epg-NTP(49161) eq 123 epg:any
[contract:uni/tn-Prod1/brc-any_to_ntp] [hit=0]
[14:4168] [vrf:Prod1:VRF1] permit ip tcp epg:any tn-Prod1/ap-Services/epg-NTP(49161) eq 123
[contract:uni/tn-Prod1/brc-any_to_ntp] [hit=0]
[16:4176] [vrf:Prod1:VRF1] permit any epg:any tn-Prod1/bd-App(16386) [contract:implicit] [hit=0]
[16:4174] [vrf:Prod1:VRF1] permit any epg:any tn-Prod1/bd-Services(32776) [contract:implicit]
[hit=0]
[16:4160] [vrf:Prod1:VRF1] permit arp epg:any epg:any [contract:implicit] [hit=0]
[21:4165] [vrf:Prod1:VRF1] deny,log any epg:any epg:any [contract:implicit] [hit=65]
[22:4164] [vrf:Prod1:VRF1] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit] [hit=0]

fab3-leaf8# show zoning-rule scope 2654209
+---------+--------+--------+----------+---------+---------+---------+------------+----------+--
--------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |   Dir   |  operSt |  Scope  |    Name    |  Action
|       Priority       |
+---------+--------+--------+----------+---------+---------+---------+------------+----------+--
--------------------+
|   4165  |   0    |   0    | implicit | uni-dir | enabled | 2654209 |            | deny,log |
any_any_any(21)    |
|   4160  |   0    |   0    | implarp  | uni-dir | enabled | 2654209 |            |  permit  |
any_any_filter(17)  |
|   4164  |   0    |   15   | implicit | uni-dir | enabled | 2654209 |            | deny,log |
any_vrf_any_deny(22) |
|   4176  |   0    | 16386  | implicit | uni-dir | enabled | 2654209 |            |  permit  |
any_dest_any(16)    |
|   4174  |   0    | 32776  | implicit | uni-dir | enabled | 2654209 |            |  permit  |
any_dest_any(16)    |
|   4168  |   0    | 49161  |   424    | uni-dir | enabled | 2654209 | any_to_ntp |  permit  |
any_dest_filter(14)  |
|   4156  | 49161  |   0    |   425    | uni-dir | enabled | 2654209 | any_to_ntp |  permit  |
```

```
src_any_filter(13)  |
+---------+--------+--------+---------+---------+---------+---------+-----------+----------+--
--------------------+
```

EPGへの共有L3Out

共有L3Outについて

共有レイヤ3アウトは、一部のサービス（外部アクセス）を提供する1つのVRFにL3Outを設定し、1つ以上の他のVRFがこのL3Outを消費できるようにする設定です。共有L3Outの詳細については、「外部ルーティング」の章を参照してください。

共有L3Outを実行する場合は、契約のプロバイダーを共有L3Outにして、EPGを契約のコンシューマにすることをお勧めします。このシナリオについては、このセクションで説明します。

逆の動作は推奨されません。つまり、L3OutはEPGによって提供されるサービスを消費します。共有サービスの場合、ゾーニングルールはコンシューマVRFにのみインストールされるため、この理由はスケーラビリティに関係します。消費と提供の原則は、トラフィックフローが開始される場所を示します。デフォルトの入力ポリシーの適用では、ポリシーの適用はコンシューマ側に適用され、より具体的には入力リーフ（非ボーダーリーフ）に適用されます。 入力リーフでポリシーを適用するには、宛先のpcTagが必要です。このシナリオでは、宛先は外部EPG pcTagです。入力リーフはポリシー適用を実行し、パケットをボーダーリーフに転送します。ボーダーリーフは、ルートルックアップ(LPM)を実行するファブリックリンクでパケットを受信し、宛先プレフィックスの隣接関係にパケットを転送します。

ただし、ボーダーリーフは、宛先EPにトラフィックを送信する際にはポリシーを適用せず、送信元EPに戻るリターントラフィックフローでも適用しません。

その結果、入力非BLリーフのポリシーCAMだけに（コンシューマVRFに）エントリがインストールされ、BLのポリシーCAMは影響を受けません。

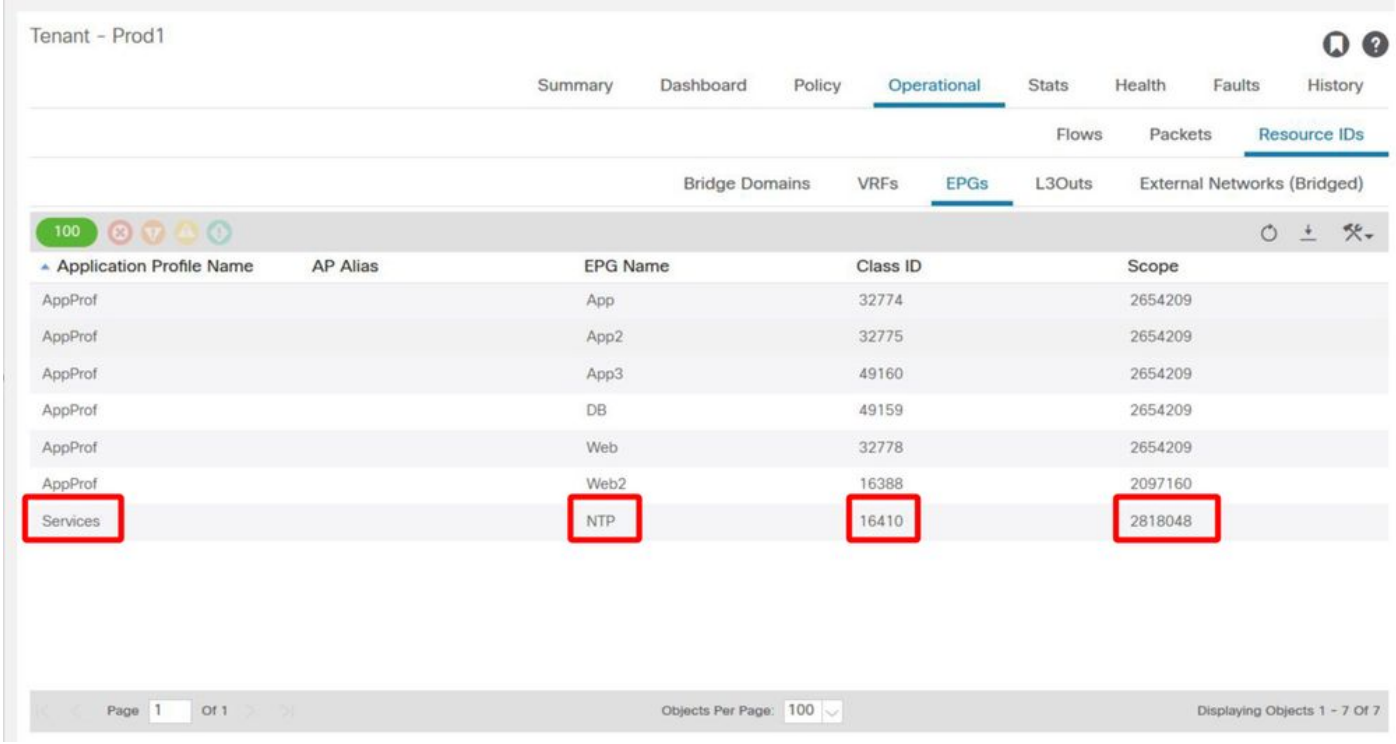
共有L3outのトラブルシューティング

ワークフロー

1.コンシューマEPGのEPG pcTagおよびVRF VNID/Scopeの確認

共有L3Outでは、ゾーン分割ルールはコンシューマVRFにのみインストールされます。プロバイダーは、このpcTagをすべてのコンシューマVRFで使用できるようにするグローバルpcTag（16k未満）を持っている必要があります。このシナリオでは、プロバイダーは外部EPGであり、グローバルpcTagを持ちます。コンシューマEPGは、通常どおりローカルpcTagを持ちます。

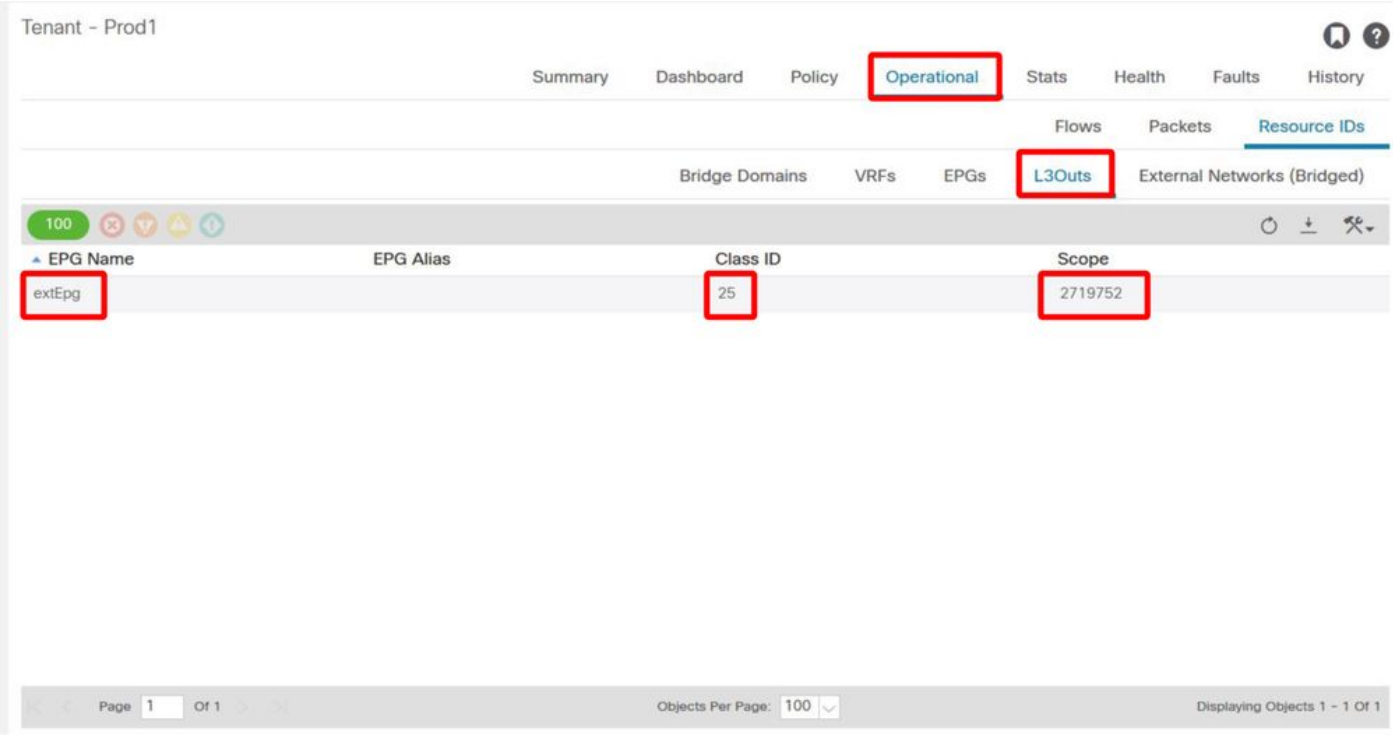
コンシューマEPGのpcTag



2.プロバイダーL3Out EPGのpcTagとVRF VNID/Scopeを確認します

ステップ1で説明したように、プロバイダーL3Out EPGには、コンシューマVRFに漏出されるL3Outからのプレフィクスとしてグローバル範囲pcTagがあります。その結果、L3Out EPGのpcTagは、コンシューマVRFのpcTagとオーバーラップしないように設定する必要があるため、グローバルpcTagの範囲内になります。

プロバイダー外部EPGのpcTag

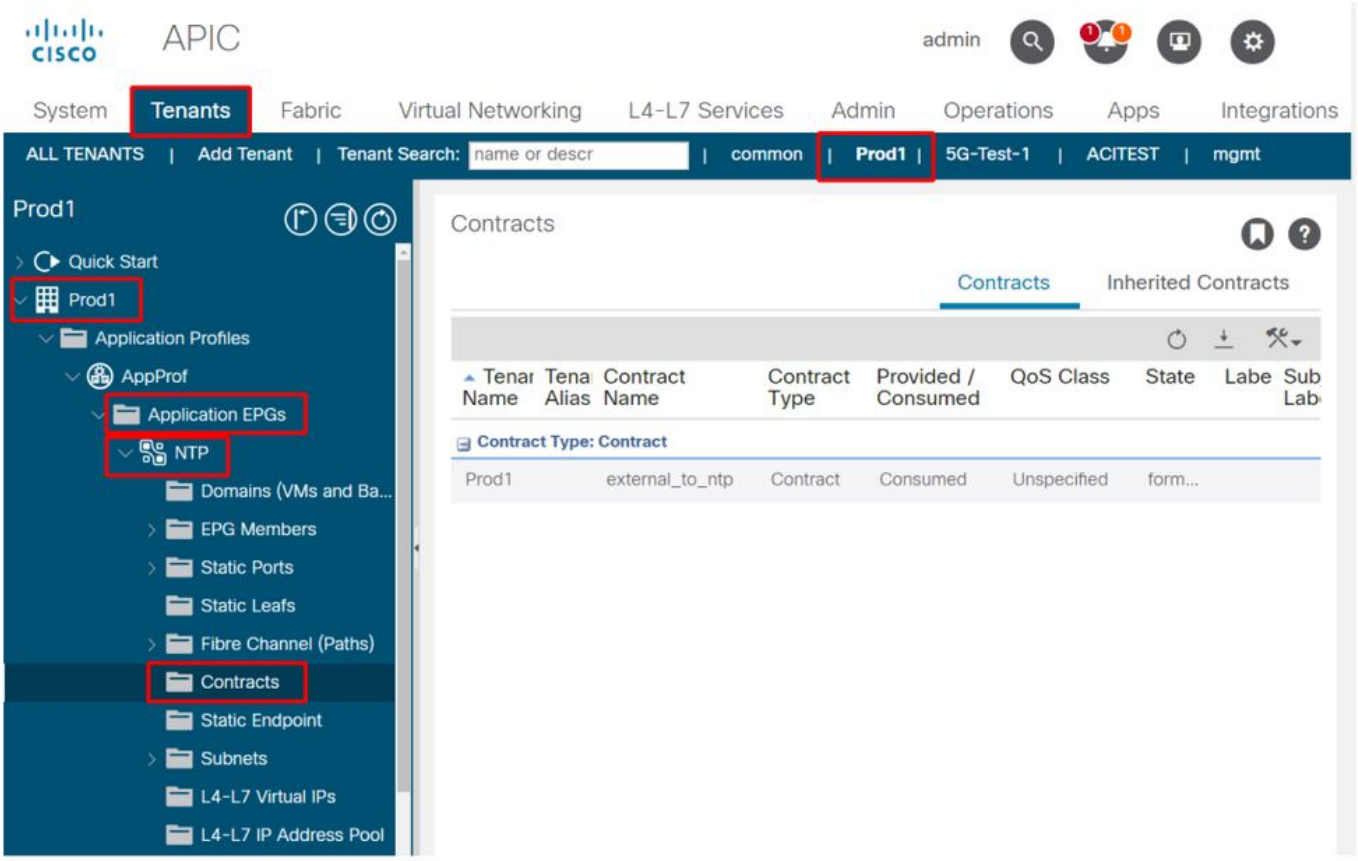


3.コンシューマEPGに、インポートされたテナント範囲コントラクトまたはグローバルコントラクトが構成されていることを確認します

EPG/BDで定義されたサブネットを持つコンシューマEPG NTPは、「テナント」または「グロー

バル」スコーピングされたコントラクトを消費しています

EPGで消費される契約



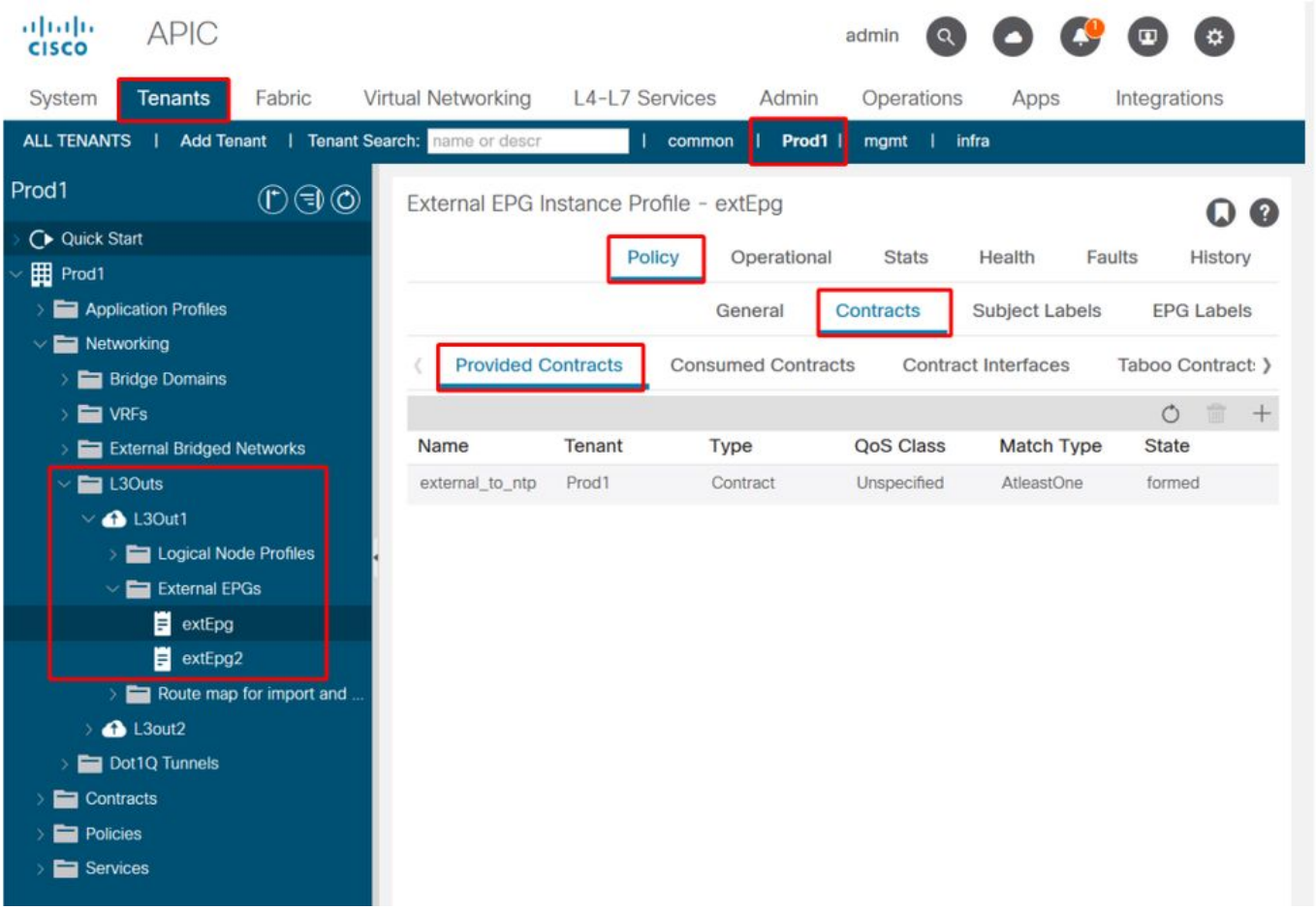
4.コンシューマEPGのBDに、スコープが「VRF間で共有」に設定されたサブネットがあるかどうかを確認します

EPGのサブネットはブリッジドメインの下で設定されますが、（ルーティングされた漏出を可能にするために）「VRF間で共有」フラグと（L3Outへのアドバタイズを可能にするために）「外部にアドバタイズ」フラグを持つ必要があります

5.プロバイダーL3Out EPGに、インポートされたテナント範囲コントラクトまたはグローバルコントラクトが設定されていることを確認します

L3Out EPGには、テナントスコープのコントラクト、または提供されたコントラクトとして設定されたグローバルコントラクトのいずれかが必要です。

プロバイダーL3Outのコントラクト



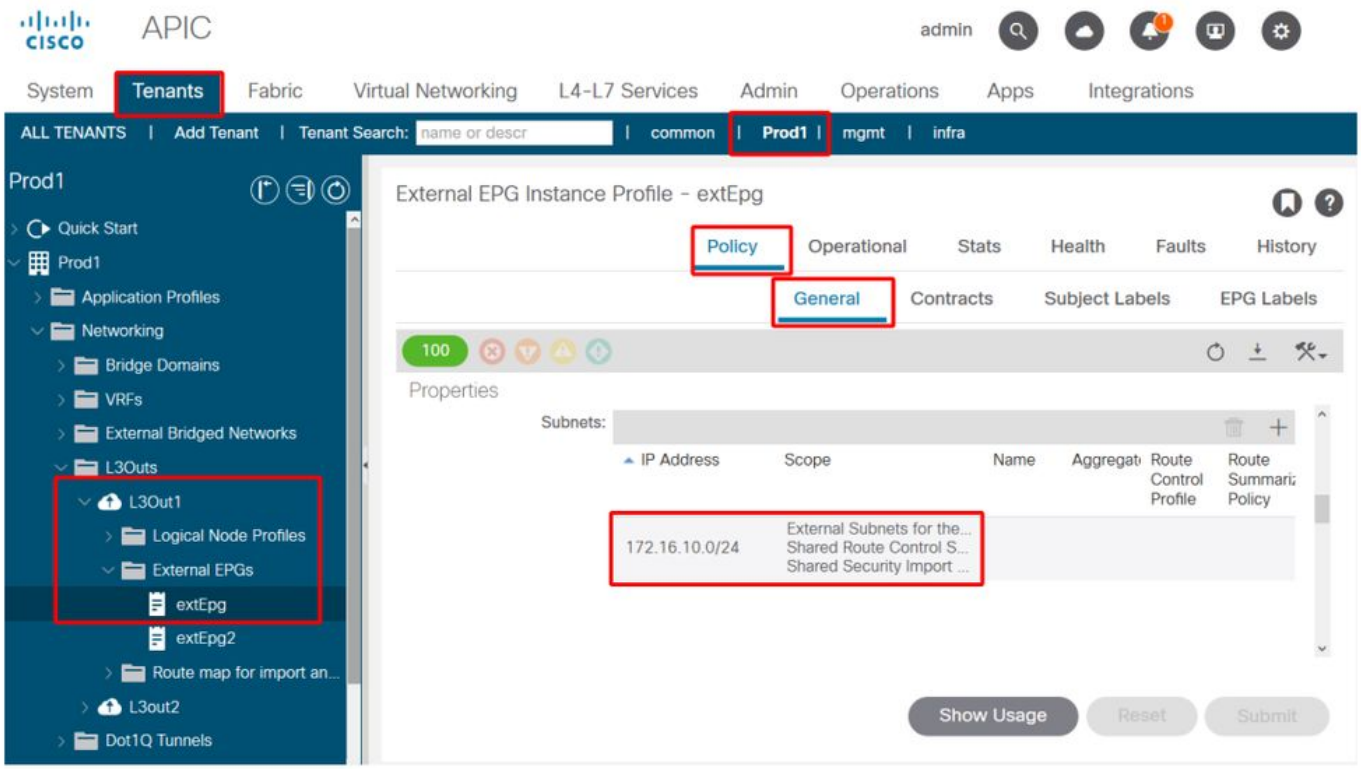
6.プロバイダーL3Out EPGに、必要なスコープがチェックされたサブネットが設定されているかどうかを確認します

プロバイダーL3Out EPGには、リークする必要のあるプレフィクスを次のスコープで設定する必要があります。

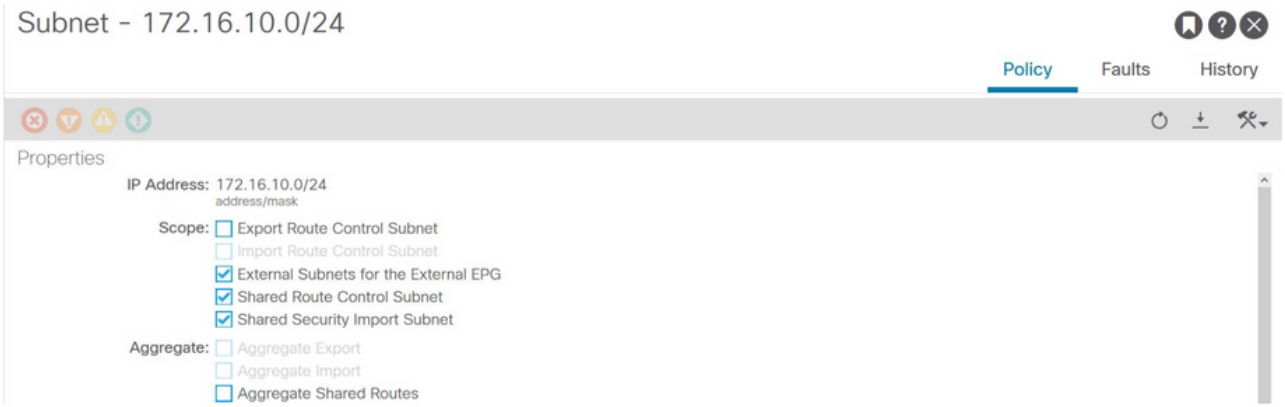
- 外部EPGの外部サブネット。
- 共有ルート制御サブネット。
- 共有セキュリティインポートサブネット。

L3Out EPGのサブネットフラグの詳細については、「外部転送」の章を参照してください。

外部EPGサブネットの設定



外部EPGサブネット設定を拡張



7.コンシューマVRFの非BL上のL3Out EPGサブネットのpcTagを確認します

外部EPGサブネット宛てのトラフィックが非BLに入ると、宛先プレフィクスに対してルックアップが実行され、pcTagが決定されます。これは、非BLで次のコマンドを使用して確認できます。

この出力は、コンシューマVRF VNIDであるVNI 2818048の範囲で取得されていることに注意してください。テーブルを見ると、同じVRF内になくても、コンシューマは宛先のpcTagを見つけることができます。

```
fab3-leaf8# vsh -c 'show system internal policy-mgr prefix' | egrep 'Vrf-Vni|==|common:default'
Vrf-Vni VRF-Id Table-Id Table-State  VRF-Name
Addr                                Class Shared Remote Complete
======= ======  ========== =======  ============================
================================= ====== ====== ====== ========
2818048 19     0x13          Up     common:default
0.0.0.0/0   15      False  False  False
2818048 19     0x80000013    Up     common:default
::/0   15      False  False  False
```

```
2818048 19     0x13          Up     common:default
172.16.10.0/24  25      True   True   False
```

上記の出力は、L3Out EPGサブネットとそのグローバルpcTag 25の組み合わせを示しています。

8.コンシューマVRFの非BLでプログラムされたゾーン分割ルールを確認します

'contract_parser.py'または'show zoning-rule'コマンドを使用して、VRFを指定します。

次のコマンド出力は、コンシューマEPGローカルpcTag 16410からL3Out EPGグローバルpcTag 25へのトラフィックを許可するために2つのゾーン分割ルールがインストールされていることを示しています。これは、コンシューマVRFのスコープであるスコープ2818048に含まれています。

```
fab3-leaf8# show zoning-rule scope 2818048
+---------+--------+--------+----------+----------------+---------+---------+-----------------+-
---------+-----------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |      Dir       |  operSt |  Scope  |       Name      |
Action  |        Priority       |
+---------+--------+--------+----------+----------------+---------+---------+-----------------+-
---------+-----------------------+
|   4174  |   0    |   0    | implarp  |    uni-dir     | enabled | 2818048 |                 |
permit  |   any_any_filter(17)  |
|   4168  |   0    |   15   | implicit |    uni-dir     | enabled | 2818048 |                 |
deny,log |  any_vrf_any_deny(22) |
|   4167  |   0    | 32789  | implicit |    uni-dir     | enabled | 2818048 |                 |
permit  |    any_dest_any(16)   |
|   4159  |   0    |   0    | implicit |    uni-dir     | enabled | 2818048 |                 |
deny,log |    any_any_any(21)    |
|   4169  |   25   |   0    | implicit |    uni-dir     | enabled | 2818048 |                 |
deny,log | shsrc_any_any_deny(12)|
|   4156  |   25   | 16410  |   425    | uni-dir-ignore | enabled | 2818048 | external_to_ntp |
permit  |     fully_qual(7)     |
|   4131  | 16410  |   25   |   424    |     bi-dir     | enabled | 2818048 | external_to_ntp |
permit  |     fully_qual(7)     |
+---------+--------+--------+----------+----------------+---------+---------+-----------------+-
---------+-----------------------+

fab3-leaf8# contract_parser.py  --vrf common:default
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]

[7:4131] [vrf:common:default] permit ip tcp tn-Prod1/ap-Services/epg-NTP(16410) tn-Prod1/l3out-
L3Out1/instP-extEpg(25) eq 123  [contract:uni/tn-Prod1/brc-external_to_ntp] [hit=0]
[7:4156] [vrf:common:default] permit ip tcp tn-Prod1/l3out-L3Out1/instP-extEpg(25) eq 123 tn-
Prod1/ap-Services/epg-NTP(16410)  [contract:uni/tn-Prod1/brc-external_to_ntp] [hit=0]
[12:4169] [vrf:common:default] deny,log any tn-Prod1/l3out-L3Out1/instP-extEpg(25) epg:any
[contract:implicit] [hit=0]
[16:4167] [vrf:common:default] permit any epg:any tn-Prod1/bd-Services(32789)
[contract:implicit] [hit=0]
[16:4174] [vrf:common:default] permit arp epg:any epg:any [contract:implicit] [hit=0]
[21:4159] [vrf:common:default] deny,log any epg:any epg:any [contract:implicit] [hit=0]
[22:4168] [vrf:common:default] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit]
[hit=0]
```

9.プロバイダーVRFのBLにプログラムされたゾーン分割ルールを確認します

'contract_parser.py'または'show zoning-rule'コマンドを使用して、VRFを指定します。次のコマンド出力は、以前に何度も説明したように、プロバイダーVRFに特定のゾーン分割ルールがNOであることを示しています。

これは、プロバイ2719752VRFのスコープです。

```
border-leaf# show zoning-rule scope 2719752
+---------+--------+--------+----------+----------------+---------+---------+-------------+----
------+----------------------+
| Rule ID | SrcEPG | DstEPG | FilterID |      Dir       |  operSt |  Scope  |     Name    |
Action  |       Priority       |
+---------+--------+--------+----------+----------------+---------+---------+-------------+----
------+----------------------+
|   4134  | 10937  |   24   | default  | uni-dir-ignore | enabled | 2719752 | vrf1_to_vrf2 |
permit  |    src_dst_any(9)    |
|   4135  |   24   | 10937  | default  |     bi-dir     | enabled | 2719752 | vrf1_to_vrf2 |
permit  |    src_dst_any(9)    |
|   4131  |   0    |   0    | implicit |    uni-dir     | enabled | 2719752 |              |
deny,log |   any_any_any(21)    |
|   4130  |   0    |   0    | implarp  |    uni-dir     | enabled | 2719752 |              |
permit  |  any_any_filter(17)  |
|   4132  |   0    |   15   | implicit |    uni-dir     | enabled | 2719752 |              |
deny,log | any_vrf_any_deny(22) |
+---------+--------+--------+----------+----------------+---------+---------+-------------+----
------+----------------------+

border-leaf# contract_parser.py --vrf Prod1:VRF3
Key:
[prio:RuleId] [vrf:{str}] action protocol src-epg [src-l4] dst-epg [dst-l4]
[flags][contract:{str}] [hit=count]

[9:4134] [vrf:Prod1:VRF3] permit any tn-Prod1/l3out-L3Out1/instP-extEpg2(10937) tn-Prod1/l3out-
L3Out2/instP-extEpg2(24) [contract:uni/tn-Prod1/brc-vrf1_to_vrf2] [hit=0]
[9:4135] [vrf:Prod1:VRF3] permit any tn-Prod1/l3out-L3Out2/instP-extEpg2(24) tn-Prod1/l3out-
L3Out1/instP-extEpg2(10937) [contract:uni/tn-Prod1/brc-vrf1_to_vrf2] [hit=0]
[16:4130] [vrf:Prod1:VRF3] permit arp epg:any epg:any [contract:implicit] [hit=0]
[21:4131] [vrf:Prod1:VRF3] deny,log any epg:any epg:any [contract:implicit] [hit=0]
[22:4132] [vrf:Prod1:VRF3] deny,log any epg:any pfx-0.0.0.0/0(15) [contract:implicit] [hit=0]
```

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。