ASR 9000でのXR Embedded Packet Tracerの設定

内容

はじめに

このドキュメントでは、XR組み込みパケットトレーサについて説明します。カスタムパケットフローをトレースして、サービスの検証やトラブルシューティングに役立てることができます。

前提条件

要件

XR Embedded Packet Tracerは、Cisco IOS® XRバージョン7.1.2以降で最初に使用可能になり、ASR 9000シリーズでサポートされます。 追加のXR製品ファミリは、将来のアップデートでサポートされる予定です。

使用するコンポーネント

XR Embedded Packet Tracerは特定のプロトコルに依存せず、すべてのタイプのユニキャストおよびマルチキャストパケットと互換性があります。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認して

ください。

バックグラウンド情報.

XR Embedded Packet Tracerフレームワークにより、サービスフロー検証とパケット転送問題のトラブルシューティングが大幅に簡素化されました。

パケットトレースがインターフェイス上でアクティブになると、ネットワークプロセッサ(NP)は着信パケットを評価して、定義された基準を満たしているかどうかを判断します。パケットが指定の条件を満たす場合、識別子が内部ヘッダーに追加されます。このIDにより、ルータ内のデータパスとパントパスに関係するすべてのコンポーネントでパケットを容易に追跡できます。

条件とは、ルータを通過するときにトレースできるパケットを定義する一連の条件またはルールのことです。これらの条件は、トラブルシューティングまたはサービス検証のために、システムが特定のパケットフローを識別および監視するのに役立ちます。

条件は、次のコンポーネントで構成されます。

1. 物理インターフェイス:

    - パケットの到着先となるネットワークインターフェイスを指定します。
    - 例： packet-trace condition interface Gi0/0/0/1

2. オフセット/値/マスクの3要素:

    - パケットヘッダー（またはペイロード）の特定の部分に一致する基準を定義します。
    - これらの3つの組み合わせにより、プロトコルヘッダーの任意の部分を表すことができるため、フレームワークをプロトコルに依存しないようにすることができます。

      例：

        - オフセット：パケット内の位置（バイトオフセット）。
        - 値：その位置の期待値。
        - マスク：精度を照合するビット。

XR Embedded Packet Tracerの制限事項

XRリリース7.1.2

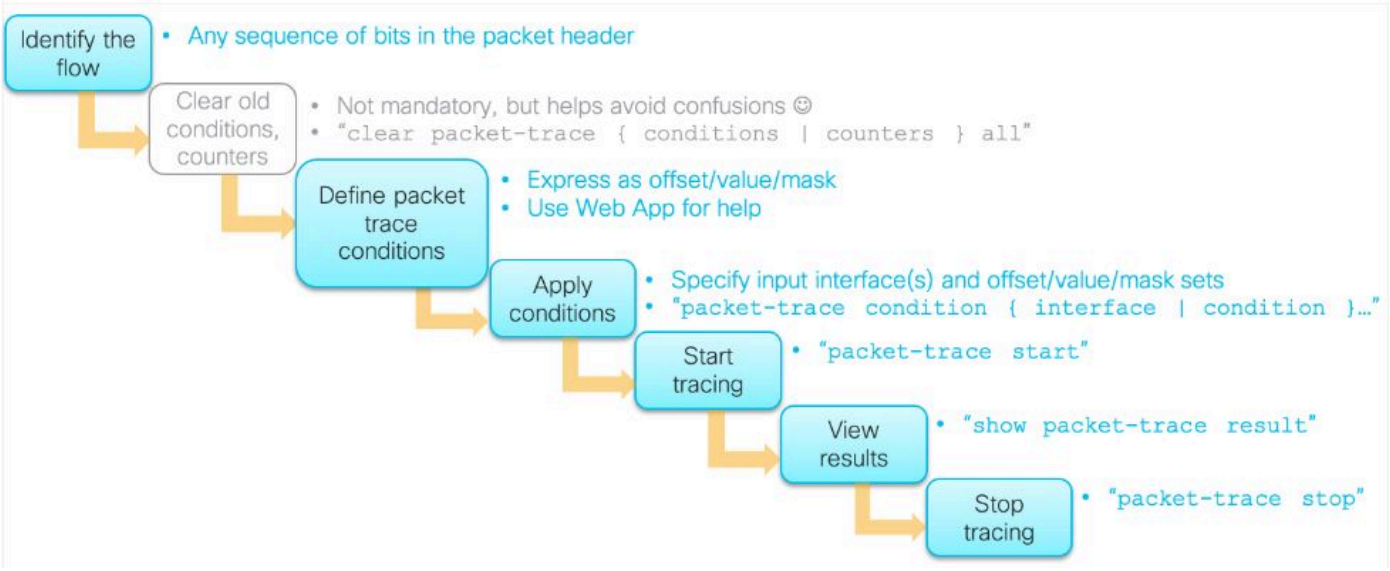
パケットマーキングは、Lightspeed Plus、LightweightPeed、およびTomahawkラインカードでサポートされています。
パケットトレースは、前述のタイプのラインカードでサポートされています。
最大3つの4オクテットのオフセット/値/マスクセットを指定できます。

XRリリース7.5.2

Packet Tracerは、条件が設定されたときにバンドルメンバーを自動的に解決します
spp、NetIO、UDP、TCPのパントパスでパケットをトレースできるようになりました

Packet Tracerのワークフロー

次の図は、Packet Tracerワークフローの動作を示しています。



設定

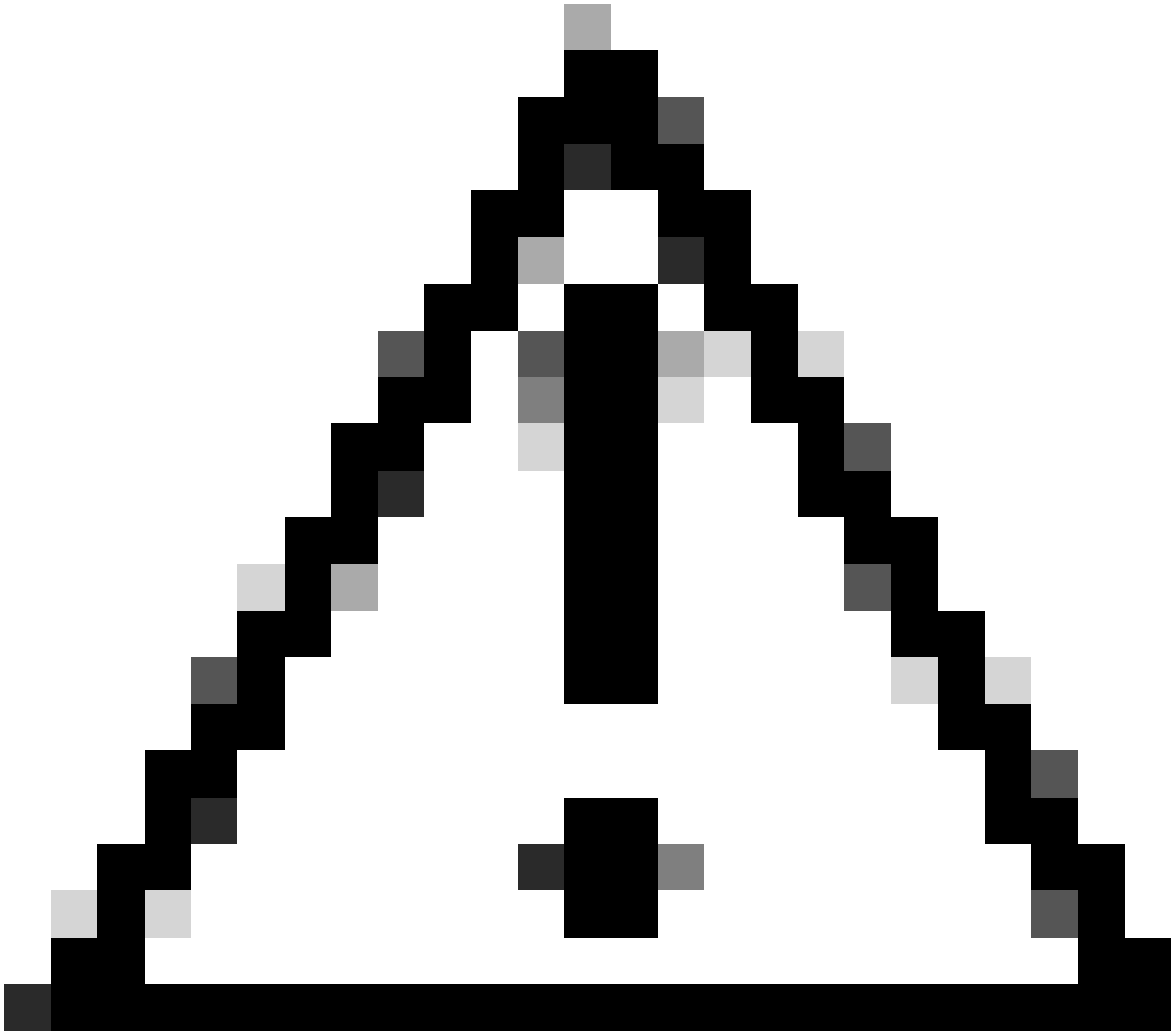
Packet Tracerのカウンタと状態をクリアする

パケットトレーサカウンタをリセットするコマンド。パケットトレーサカウンタは、必要に応じてリセットできます。

```
clear packet-trace counters all
```

すべてのパケットトレーサの状態を削除するには、次のコマンドを使用します。

```
clear packet-trace conditions all
```



注意：設計上、パケットトレーサの状態をクリアできるのは、パケットトレースが非アクティブである間だけです。

パケットトレースの開始/停止

パケットトレースの開始と終了を手動で指定する必要があります。

```
RP/0/RP0/CPU0:Device#  packet-trace start
RP/0/RP0/CPU0:Device#  packet-trace stop
```

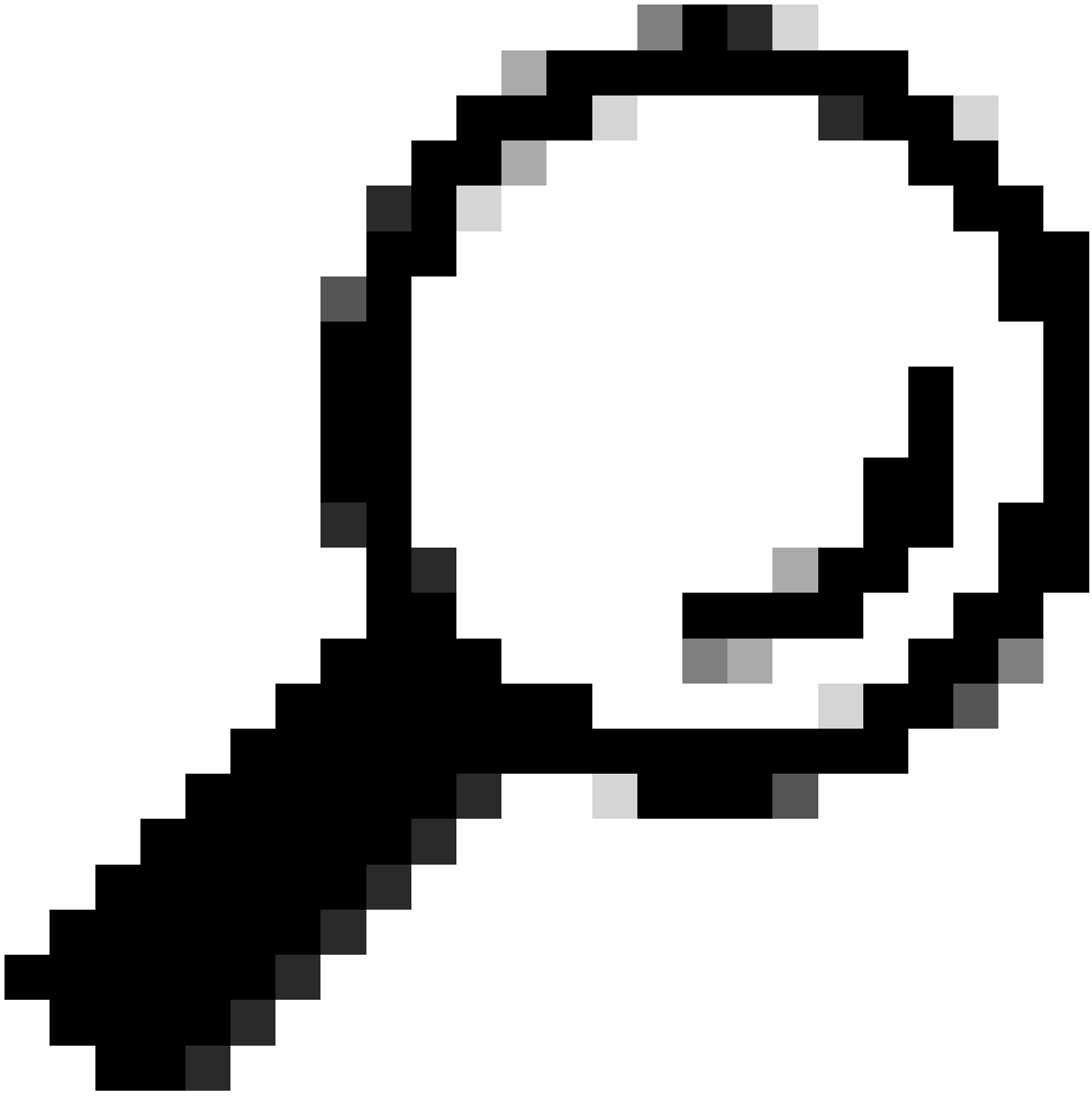
Packet Tracerの状態

条件は次のとおりです。

1. 物理インターフェイス：パケットを受信すると予想される物理インターフェイスを示します。
2. [オフセット]、[値]、[マスク] [3要素]を選択します。関心の流れを定義するのに役立ちます。

Packet Tracerの状態 – インターフェイス

```
RP/0/RP0/CPU0:Device#packet-trace condition interface GigE0/0/0/0
RP/0/RP0/CPU0:Device#packet-trace condition interface GigE0/0/0/1
```



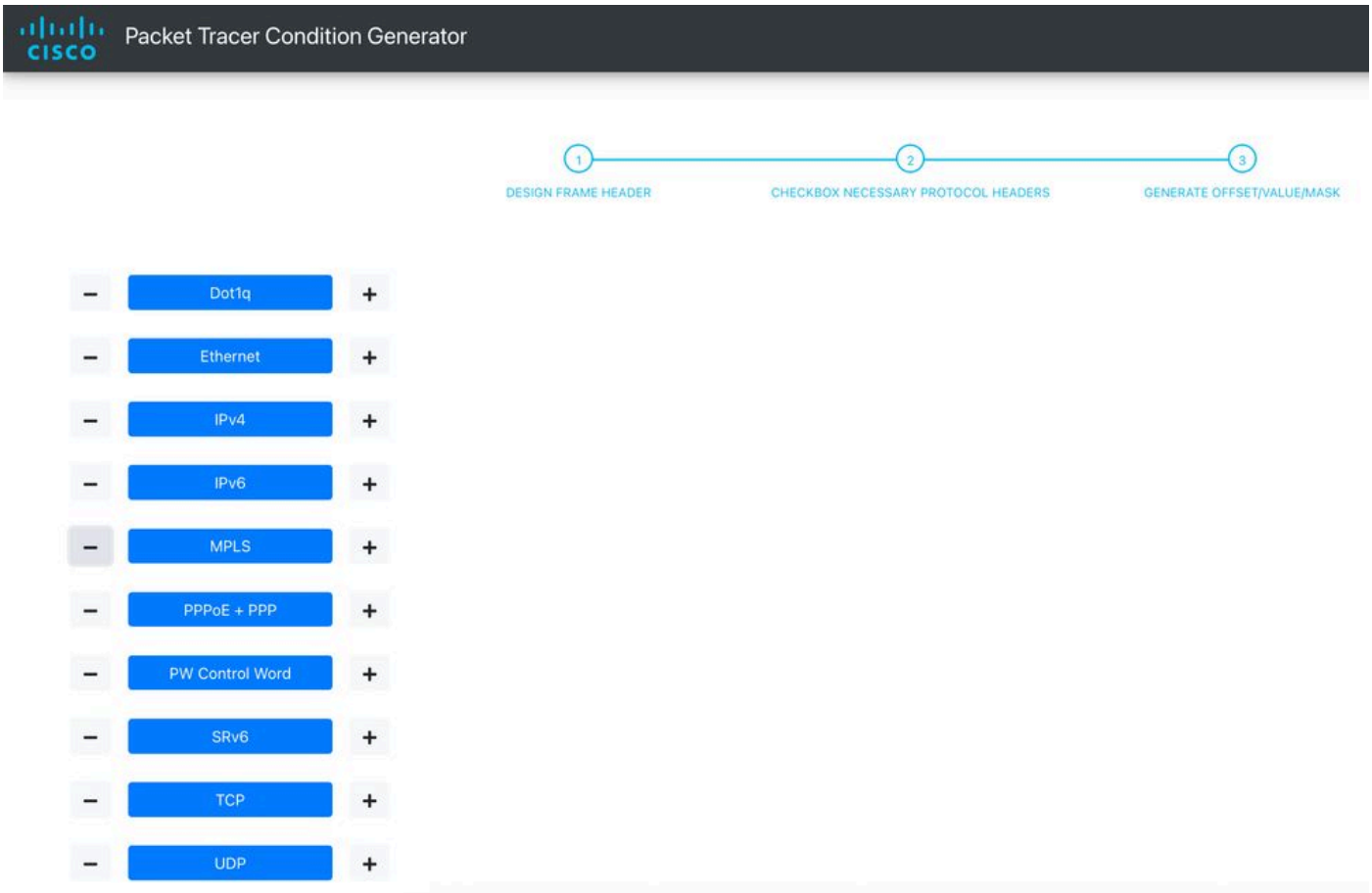
ヒント：サブインターフェイスでトレースする場合、オフセット/値/マスクの指定では、dot1qまたはQinQカプセル化を考慮する必要があります。

Packet Tracerの状態 – オフセット/値/マスク

「XR Packet Tracer Condition Generator Web App」は、パケットトレーサ条件を作成するためのツールを提供します。

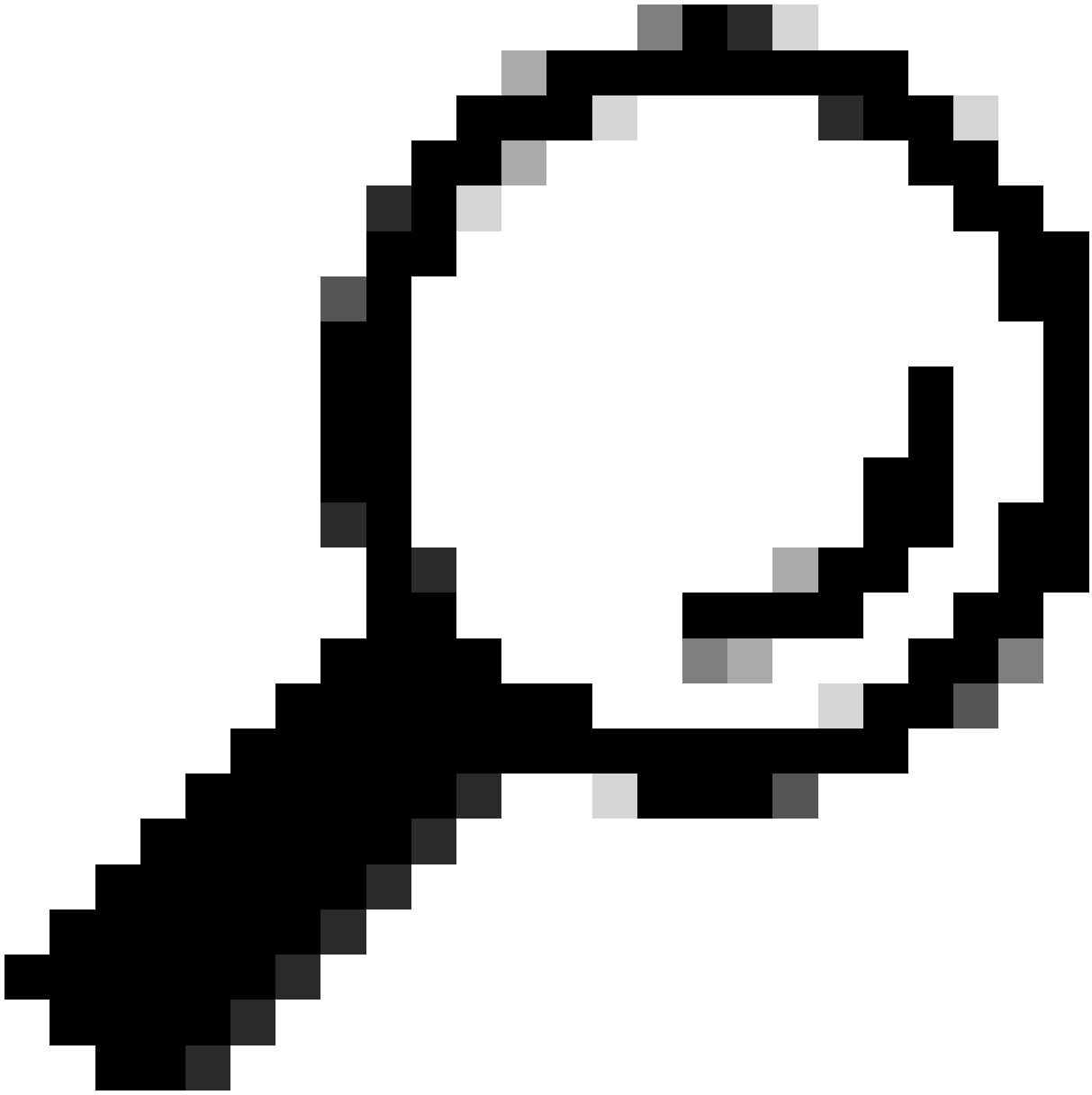
ソースコードとインストールのガイドラインは、GitHubのXR Embedded Packet Tracer - Condition Generatorという名前でアクセスできます。

このアプリケーションを使用すると、目的のパケットフローのプロトコルスタックを視覚的に構築し、条件を定義するための関連レイヤを選択し、トレースする特定のフローを記述する値（オプションのマスクを使用）を入力できます。



Web Appのランディングページには、設定がサポートされているプロトコルヘッダーのリストが表示されます。

オフセットの計算はヘッダーの順序に依存するため、トラフィックを照合するヘッダーの前に必要なすべてのヘッダーを含めてください。



ヒント:PW制御ワードヘッダーを使用する場合は、必ずこれを含めてください。

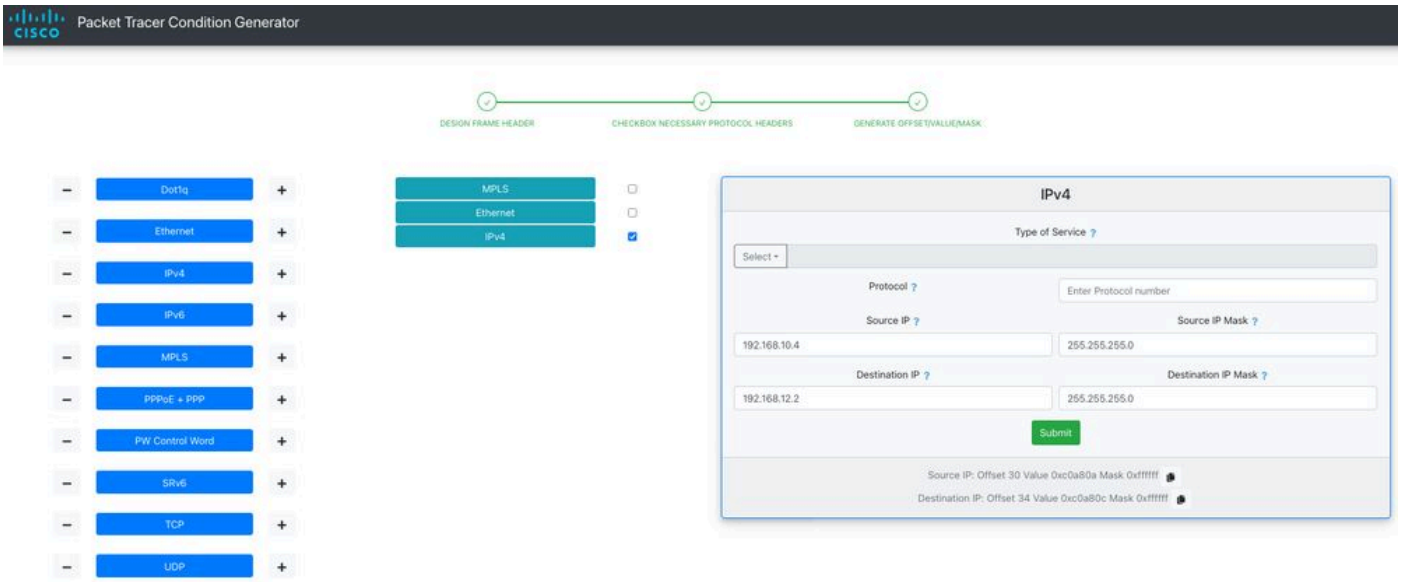
設定例：

次にトポロジの例を示します。このドキュメントの目的は、パケットがXRV1デバイスを介して適切に送受信されているかどうかを確認することです。

1. – 監視する特定のインターフェイスのパケットトレース条件を設定します。

```
RP/0/RP0/CPU0:xrv-1#packet-trace condition interface Bundle-Ether1
RP/0/RP0/CPU0:xrv-1#packet-trace condition interface Bundle-Ether2
```

2. – オフセット/値/マスクを生成します。一致させるヘッダーの横にあるチェックボックスを選択します。必要に応じて、複数のヘッダーを選択できます。選択したヘッダーごとに、対応するフレームが右側に表示されます。必要な値とマスクをフレームに入力し、Submitボタンをクリックして設定を完了します。



3. – オフセット/値/マスクがクリップボードにコピーされたら、これを使用して条件を定義します。

```
RP/0/RP0/CPU0:xrv-1#packet-trace condition 1 Offset 30 Value 0xc0a80a Mask 0xffffff
RP/0/RP0/CPU0:xrv-1#packet-trace condition 5 Offset 34 Value 0xc0a80c Mask 0xffffff
```

4. – パケットトレースステータスを確認します。

```
RP/0/RP0/CPU0:xrv-1#show packet-trace status
------------------------------------------------------------
Packet Trace Master Process:
  Buffered Conditions:
    Interface Bundle-Ether1
      Member GigE0/0/0/0
    Interface Bundle-Ether2
      Member GigE0/0/0/1
    1 Offset 30 Value 0xc0a80a Mask 0xffffff
    5 Offset 34 Value 0xc0a80c Mask 0xffffff
  Status: Inactive
RP/0/RP0/CPU0:xrv-1#
RP/0/RP0/CPU0:xrv-1#show packet-trace status detail
------------------------------------------------------------
Location: 0/0/CPU0
Available Counting Modules: 4

  #1 SPP
    Last errors:
  #2 npu_server_lsp
   Last errors:
  #3 NETIO
    Last errors:
  #4 UDP
    Last errors:

Available Marking Modules: 1

  #1 npu_server_lsp
    Interfaces: 0
    Conditions: 0
    Last errors:
------------------------------------------------------------
Packet Trace Master Process:

  Buffered Conditions:
    Interface Bundle-Ether1
      Member GigE0/0/0/0
    Interface Bundle-Ether2
      Member GigE0/0/0/1
    1 Offset 30 Value 0xc0a80a Mask 0xffffff
    5 Offset 34 Value 0xc0a80c Mask 0xffffff
  Status: Inactive
------------------------------------------------------------
Location: 0/RP0/CPU0
Available Counting Modules: 3

  #1 SPP
    Last errors:
  #2 NETIO
    Last errors:
  #3 UDP
    Last errors:

Available Marking Modules: 0
RP/0/RP0/CPU0:xrv-1#
```

5.- Packet Tracerを起動します。

```
RP/0/RP0/CPU0:xrv-1# packet-trace start
RP/0/RP0/CPU0:xrv-1#
RP/0/RP0/CPU0:xrv-1# show packet-trace status
---------------------------------------------------------
Packet Trace Master Process:
  Buffered Conditions:
    Interface Bundle-Ether1
      Member GigE0/0/0/0
    Interface Bundle-Ether2
      Member GigE0/0/0/1
    1 Offset 30 Value 0xc0a80a Mask 0xffffff
    5 Offset 34 Value 0xc0a80c Mask 0xffffff
  Status: Active
RP/0/RP0/CPU0:xrv-1#
```

6. – トラフィックをキャプチャするために、数分待ちます。

7. – 結果を確認します。

```
RP/0/RP0/CPU0:xrv-1#show packet-trace result
T: D - Drop counter; P - Pass counter
Location     | Source       | Counter                   | T | Last-Attribute
-----------   ------------   -------------------------   -   ----------------------------------------
0/0/CPU0       NP0            PACKET_MARKED               P      GigE0_0_0_0
0/0/CPU0       NP0            PACKET_TO_FABRIC            P
0/0/CPU0       NP0            PACKET_TO_PUNT              P
0/0/CPU0       NP0            PACKET_FROM_FABRIC          P
0/0/CPU0       NP0            PACKET_TO_INTERFACE         P      GigE0_0_0_1
RP/0/RP0/CPU0:xrv-1#
```

8.- show packet-trace descriptionコマンドを使用して、パケットトレーサフレームワークに登録されているすべてのカウンタとその説明を確認できます。

```
RP/0/RP0/CPU0:xrv-1#show packet-trace descriptions
NP0                PACKET_MARKED                M   Marked from ingress interface
NP0                PACKET_FROM_INJECT           P   Injected from linecard CPU
NP0                PACKET_FROM_FAB_INJECT       P   Injected from fabric
NP0                PACKET_ING_DROP              D   Dropped on ingress
NP0                PACKET_TO_FABRIC             P   Sent to router fabric
NP0                PACKET_TO_PUNT               P   Punted to linecard for CPU handling
NP0                PACKET_FROM_FABRIC           P   From router fabric
NP0                PACKET_EGR_DROP              D   Dropped on egress
NP0                PACKET_TO_INTERFACE          P   Packet sent to network interface
```

```
RP/0/RP0/CPU0:xrv-1#
```

9. – パケットトレースを停止します。

```
RP/0/RP0/CPU0:xrv-1#packet-trace stop
```

関連情報

XR組み込みパケットトレーサ

シスコのテクニカルサポートとダウンロード

ASR 9000シリーズラインカードのタイプの理解

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。