

# モデル駆動型テレメトリ

## モデル駆動型テレメトリ

テレメトリは、自動の通信プロセスです。これにより、測定およびその他のデータがリモート ポイントで収集され、モニタリング用の受信装置に送信されます。モデル駆動型テレメトリは、YANG モデル化されたデータをデータコレクタに提供します。

アプリケーションは、NETCONF、RESTCONF、または gRPC ネットワーク管理インターフェイス（gNMI）の各プロトコルを介した標準ベースの YANG データ モデルを使用して、必要とする特定のデータ項目をサブスクライブします。サブスクリプションは、設定済みサブスクリプション用に CLI を使用して作成することもできます。

構造化データは、サブスクリプション基準とデータ タイプに基づき、定義されたパターンでまたは変更時にパブリッシュされます。

テレメトリを使用するシステムには、さまざまなロールがあります。このドキュメントでは、次のテレメトリロールについて説明します。

- パブリッシャ：テレメトリ データを送信するネットワーク要素。
- レシーバー：テレメトリデータを受信します。コレクタとも呼ばれます。
- コントローラ：サブスクリプションを作成するがテレメトリ データを受信しないネットワーク要素。作成したサブスクリプションに関連付けられたテレメトリ データが受信者に送信されます。管理エージェントまたは管理エンティティとも呼ばれます。
- サブスクライバ：サブスクリプションを作成するネットワーク要素。このドキュメントではレシーバーでもありません。

## モデル駆動型テレメトリの機能履歴

次の表に、このモジュールで説明する機能のリリースおよび関連情報を示します。

これらの機能は、特に明記されていない限り、導入されたリリース以降のすべてのリリースで使用できます。

| リリース                 | 機能          | 機能情報                                                                                                                                                                                                                              |
|----------------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Cisco IOS XE 17.18.1 | モデル駆動型テレメトリ | <p>モデル駆動型テレメトリでは、ネットワーク デバイスからサブスクライバに、リアルタイムの設定や運用状態の情報を継続的にストリームすることができます。</p> <p>この機能は、次のプラットフォームに実装されました。</p> <ul style="list-style-type: none"><li>• Cisco C9350 シリーズ スマートスイッチ</li><li>• Cisco C9610 シリーズ スマートスイッチ</li></ul> |

Cisco Feature Navigator を使用すると、プラットフォームおよびソフトウェアイメージのサポート情報を検索できます。Cisco Feature Navigator にアクセスするには、<https://cfng.cisco.com/> に進みます。

## モデル駆動型テレメトリの前提条件

これらの前提条件はモデル駆動型テレメトリに適用されます。

- テレメトリを使用する際に必要なデータを理解して定義するには、YANG に関する知識が必要です。
- XML、XML 名前空間、および XML XPath の知識。
- IETF テレメトリ仕様で定義されている標準および原則の知識。
- urn:ietf:params:netconf:capability:notification:1.1 機能は、hello メッセージでリストする必要があります。この機能は、IETF テレメトリをサポートするデバイスでのみアドバタイズされます。
- NETCONF-YANG がデバイス上で設定済みであり稼働している必要があります。



(注)

テレメトリを機能させるには、NETCONF-YANG または gNXI を設定する必要があります。プラットフォームが gNXI をサポートしていない場合は、NETCONF が使用されていない場合でも、NETCONF を設定する必要があります。NETCONF-YANG の設定の詳細については、「NETCONF プロトコル」モジュールを参照してください。gNXI の詳細については、「[gNMI Protocol](#)」[英語]を参照してください。

**show platform software yang-management process** コマンドを使用して、次のプロセスが実行中であることを確認します。

```
Device# show platform software yang-management process
```

```
confd      : Running
nesd       : Running
syncfd     : Running
ncsshd     : Running
dmiauthd   : Running
nginx      : Running
ndbmand    : Running
pubd       : Running
gnmib      : Running
```



(注)

プロセス pubd はモデル駆動型テレメトリ プロセスであり、これが実行していない場合にはモデル駆動型テレメトリは機能しません。

次の表に、各デバイス管理インターフェイス（DMI）プロセスの詳細を示します。

表 1: フィールドの説明

| DMI プロセス名 | 主要な役割                                     |
|-----------|-------------------------------------------|
| confd     | コンフィギュレーション デーモン                          |
| nesd      | ネットワーク要素シンクロナイザデーモン                       |
| syncfd    | 同期デーモン（実行状態と対応するモデル間の同期を維持）               |
| ncsshd    | NETCONF セキュアシェル（SSH）デーモン                  |
| dmiauthd  | DMI 認証デーモン。                               |
| nginx     | NGINX Web サーバ。RESTCONF の Web サーバとして機能します。 |
| ndbmand   | NETCONF データベースマネージャ                       |
| pubd      | モデル駆動型テレメトリに使用されるパブリケーションマネージャおよびパブリッシャ   |
| gnmib     | GNMI プロトコルサーバ。                            |

## モデル駆動型テレメトリの制約事項

次の制約事項がモデル駆動型テレメトリに適用されます。

- yang-push ストリームを使用している場合、選択における自動階層は、変更時サブスクリプション向けにサポートされません。つまり、リストを選択するときに、リストの子リストが自動的に含まれません。たとえば、サブスクライバでは、子リストごとにサブスクリプションを手動で作成する必要があります。

この制限は、次のリストの要素に登録している場合、定期的なサブスクリプションにも適用されます。

- openconfig-access-points
  - openconfig-ap-manager
  - openconfig-lacp
  - openconfig-platform-psu
- データアクセス許可のチェックはサポートされていません。サブスクライバによって要求されたすべてのデータが送信されます。
  - サブツリー フィルタはサポートされていません。サブツリー フィルタが指定された場合、サブスクリプションは無効としてマークされます。
  - サブスクリプション パラメータの中で複数の受信者を定義することはサポートされていません。最初の受信者の宛先だけが試行されます。他の定義済みの受信者は無視されます。

# サブスクリプションとは

サブスクリプションは、テレメトリロール間の関連付けを作成し、ロール間で送信されるデータを定義します。

具体的には、サブスクリプションは、テレメトリデータの一部として要求される一連のデータを定義するために使用されます。たとえば、データが必要なタイミング、データの書式設定の方法、データを受信するレシーバーを定義します。

テレメトリでは、動的サブスクリプションと設定済みサブスクリプションの2つのタイプを使用します。

- 動的サブスクリプションは、パブリッシャに接続するクライアントまたはサブスクライバによって作成されるため、ダイヤルインです。
- 設定済みサブスクリプションでは、パブリッシャがレシーバーへの接続を開始します。そのため、サブスクリプションはダイヤルアウトとなります。

サブスクリプションは、設定済みか動的のいずれかにすることができ、トランスポートプロトコルの任意の組み合わせを使用できます。定期的にトリガーされるサブスクリプション（最小値100センチ秒）と、変更時にトリガーされるサブスクリプションがサポートされています。

サポートされているサブスクリプションの最大数はプラットフォームによって異なりますが、すべてのプラットフォームで少なくとも100個のサブスクリプションがサポートされています。サブスクリプションの設定では、NETCONFやその他のノースバウンドプログラマブルインターフェイス（RESTCONF、gNMIなど）がサポートされています。同時に動作するサブスクリプションが多すぎて、有効な設定済みサブスクリプションがすべてアクティブにならない場合、アクティブなサブスクリプションを削除すると、非アクティブであるが有効な設定済みサブスクリプションがアクティブになることがあります。

## データソースの仕様

サブスクリプション内のテレメトリデータのソースは、ストリームとフィルタを使用して指定されます。ストリームとは、関連する一連のイベントを指します。RFC 5277ではイベントストリームを、いくつかの転送基準に一致する一連のイベント通知として定義しています。

通常、システムはストリームからイベントのセットをフィルタ処理し、ストリームのタイプによってさまざまなフィルタタイプを使用します。

Cisco IOS XE は、yang-push と yang-notif-native の2つのストリームをサポートしています。

## 更新の通知

サブスクリプションの一部として、データが必要になるタイミングを指定できます。ただし、これはストリームによって異なります。ストリームの中で変更またはイベントが発生した場合にのみデータを使用できるようにするストリームもあれば、変更発生時に、または定義済みの時間間隔でデータを使用できるようにするストリームもあります。

このタイミング指定の結果が、対象のテレメトリデータを送る一連の更新通知となります。データの送信方法は、パブリッシャとレシーバー間の接続に使用されるプロトコルによって異なります。

## サブスクリプション識別子

サブスクリプションは 32 ビットの正の整数値で識別されます。設定済みサブスクリプションの ID はコントローラによって設定され、動的サブスクリプションの場合はパブリッシャによって設定されます。

コントローラは、パブリッシャで作成された動的サブスクリプションとの競合を避けるために、設定済みサブスクリプションに使用する値を 0 ～ 2,147,483,647 の範囲に制限する必要があります。動的サブスクリプションの ID 空間はグローバルです。つまり、独立して作成された動的サブスクリプションのサブスクリプション ID は重複しません。

## ダイヤルインおよびダイヤルアウト サブスクリプション

モデル駆動型テレメトリには、ダイヤルインとダイヤルアウトの 2 種類があります。次の表では、両方のタイプのサブスクリプションを比較します。

表 2: ダイヤルインおよびダイヤルアウトサブスクリプション

| ダイヤルイン                                                                                             | ダイヤルアウト                                                                   |
|----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| テレメトリの更新は、イニシエータまたはサブスクリバに送信されます。                                                                  | テレメトリの更新は、指定された受信者またはコレクタに送信されます。                                         |
| サブスクリプションの存続期間は、そのサブスクリプションを作成した接続（セッション）に結び付けられ、その存続期間中テレメトリの更新が送信されます。実行コンフィギュレーションでは変更は観察されません。 | サブスクリプションは実行コンフィギュレーションの一部として作成されます。これは、コンフィギュレーションが削除されるまでデバイス設定として残ります。 |
| ダイヤルイン サブスクリプションはリロード後に再起動する必要があります。これは、確立された接続またはセッションがステートフルスイッチオーバー時に kill されるためです。             | ダイヤルアウト サブスクリプションはデバイス設定の一部として作成され、ステートフルスイッチオーバー後に自動的にレシーバーに再接続します。      |
| サブスクリプション ID は、サブスクリプションの確立が成功したときに動的に生成されます。                                                      | サブスクリプション ID は固定であり、設定の一部としてデバイス上で設定されます。                                 |

## ストリーム

ストリームは、サブスクライブ可能な一連のイベントを定義します。この一連のイベントは、ほぼどのような内容でも可能です。ただし、各ストリームの定義に従い、すべてのイベントの候補は何らかの形で関連しています。このセクションでは、サポートされているストリーム、yang-push および yang-notif-native について説明します。

サポートされているストリームのセットを表示するには、管理プロトコル操作を使用して、*mdt-streams* コンテナにある Cisco-IOS-XE-mdt-oper-v2 モジュール（YANG モデル Cisco-IOS-XE-mdt-oper-v2.yang からのもの）から *streams* テーブルを取得します。

次に、NETCONF を使用してサポートされているストリームを取得する例を示します。

```
<get>
<filter>
<mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
<mdt-streams/>
</mdt-oper-data>
```

```
</filter>
</get>
```

\* Enter a NETCONF operation, end with an empty line

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
  <data>
    <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
      <mdt-streams>
        <stream>native</stream>
        <stream>yang-notif-native</stream>
        <stream>yang-push</stream>
      </mdt-streams>
    </mdt-oper-data>
  </data>
</rpc-reply>
```

この例は、**native**、**yang-notif-native**、**yang-push** の3つのストリームがサポートされていることを示しています。ストリーム **native** は汎用としては使用できず、無視できます。



(注)

---

現在のところ、サポートされているストリームのリストを返す CLI はありません。

---

## yang-push ストリーム

**yang-push** ストリームは、YANG データストアからクライアントへの更新の自動継続的なストリーミングを可能にする標準です。**yang-push** ストリームは、サポートされている YANG モデルにより記述される、構成データベース内と運用データベース内のデータです。このストリームは、ストリームの中で対象とするデータを指定するための XPath フィルタをサポートしており、XPath 式は対象のデータを定義する YANG モデルに基づきます。

更新通知は、対象のサブスクリプションについて、データの変更時または固定間隔で送信される場合がありますが、両方に対応して送信されることはありません。現在存在しないデータのサブスクリプションは許可され、通常のサブスクリプションとして実行されます。サポートされている唯一のターゲットデータベースは「実行中」です。

**yang-push** ストリームを使用するテレメトリは、テレメトリの IETF NETCONF ワーキング グループの初期ドラフトに基づいています。これらを次に示します。

- [イベント通知のカスタム サブスクリプション、バージョン 03](#)
- [YANG データストア プッシュ更新のサブスクリプション、バージョン 07](#)

対応するドラフトに記載されている次の機能はサポートされていません。

- サブツリー フィルタ
- アウトオブバンドの通知
- サポート対象として明示的に記載されていないすべてのサブスクリプション パラメータ

## 変更時機能の決定

現在のところ、変更時サブスクリプションを使用し、サブスクリプション可能なデータのタイプについて YANG モデルの中で指定する手段はありません。変更時サブスクリプションを使用して、サブスクリプションできないデータにサブスクリプション

ライブしようとする、失敗（動的）となるか、無効なサブスクリプション（設定済み）となります。On-Change パブリケーションの詳細については、「*On-Change Publication for yang-push*」の項を参照してください。

## YANG-push の XPath フィルタ

XML Path Language (XPath) は、XML ドキュメントのノードを選択およびアドレス指定するように設計されたクエリおよびナビゲーション言語です。ファイルシステムのパスのような式を使用して、XML ドキュメントの階層構造内を移動し、要素、属性、テキスト、およびその他のノードタイプを識別します。

サブスクライブ先の *yang-push* ストリーム内のデータセットは、XPath フィルタを使用して指定する必要があります。XPath 式には次のガイドラインが適用されます。

- XPath 式では、リストまたはコンテナに 1 つのエントリを指定するためのキーを持たせることができます。サポートされているキー指定の構文は次のとおりです。

```
[{key name}={key value}]
```

次に示すのは、XPath 式のサンプルです。

```
filter xpath /rt:routing-state/routing-instance[name="default"]  
/ribs/rib[name="ipv4-default"]/routes/route # VALID!
```

複合キーを使用するには、複数のキー指定を使用します。キーの名前と値は正確である必要があります。範囲やワイルドカードによる値はサポートされていません。

- XPath 式では、複数のキーを順番に指定して選択します。XPath 式の例を次に示します。

```
/wireless-access-point-oper:access-point-oper-data/radio-oper-data/  
radio-slot-id[wtp-mac="00:11:22:33:44:55"][radio-slot-id="1"]
```

- XPath 式では、単一のサブスクリプションで複数のオブジェクトをサポートできるように、結合演算子 (|) を使用できます。ユニオン演算子は NETCONF トランスポートでのみ機能し、gRPC では機能しません。

## YANG-push の定期パブリケーション

YANG-push ストリームの定期的なパブリケーションとは、データが変更されたかどうかに関係なく、YANG データストアからの更新が設定された時間間隔で定期的に自動送信されるサブスクリプションモードを指します。これは、サブスクライバがデータスナップショットを、定義された時間間隔に基づいて定期的に受信することを意味します。

定期的なサブスクリプションでは、サブスクライブ対象情報による最初のプッシュ更新は即時に送信されます。ただしデバイスがビジー状態であったりネットワークが混雑していたりすると遅延することがあります。次に更新は、設定された定期タイマーの満了時に送信されます。たとえば、期間を 10 分と設定すると、サブスクリプションの作成直後に最初の更新が送信され、その後は 10 分おきに送信されます。

期間は、定期的なプッシュ更新間のセンチ秒 (1/100 秒) 単位の時間です。期間が 1000 であれば、サブスクライブ対象情報の更新は 10 秒ごとになります。設定できる最小の期間間隔は 100 (つまり 1 秒) です。デフォルト値はありません。この値は、動的サブスクリプションの場合は *establish-subscription* RPC で明示的に設定する必要があり、設定済みサブスクリプションの場合は設定で明示的に設定する必要があります。

定期的な更新には、サポートされているすべてのトランスポート プロトコルに関連するサブスクライブ対象のデータ要素またはテーブルのフルコピーが含まれています。

定期的なサブスクリプションを使用して空のデータをサブスクライブすると、要求された期間で空の更新通知が送信されます。データが存在するようになると、次の期間の値が通常の更新通知として送信されます。

## **YANG-push の変更時パブリケーション**

変更時パブリケーションでは、ターゲットの YANG データストア内のモニタリング対象データが変更された場合にのみ、更新がサブスクライバにプッシュされます。

変更時サブスクリプションを作成する場合は、ダンプニング期間がないことを示すためにダンプニング期間を0に設定する必要があります。その他の値はサポートされていません。

変更時サブスクリプションでは、最初のプッシュ更新は、サブスクライブされたデータのセット全体です（IETF の文書で定義されている初期同期）。これは制御できません。以降の更新は、データが変更され、変更後のデータのみで構成されている場合に送信されます。ただし、変更とみなされる最小のデータ分解能は行です。したがって、変更時サブスクリプションが行内のリーフに対するものである場合、その行のいずれかの項目が変更されると、更新通知が送信されます。更新通知の正確な内容はトランスポート プロトコルによって異なります。

また、変更時サブスクリプションは階層状ではありません。つまり、子コンテナを持つコンテナにサブスクライブしても、子コンテナ内の変更はサブスクリプションには認識されません。

現在存在しないデータのサブスクリプションは許可され、通常のサブスクリプションとして実行されます。初期同期更新通知は空であり、データが利用可能になるまでそれ以上更新されません。

XPath 式は単一のオブジェクトを指定する必要があります。このオブジェクトには、コンテナ、リーフ、リーフリスト、またはリストを使用できます。

## **yang-notif-native ストリーム**

yang-notif-native ストリームは、パブリッシャ内の任意の YANG 通知であり、通知の元のイベントソースで Cisco IOS XE のネイティブのテクノロジーが使用されています。このストリームは、対象となる通知を指定する XPath フィルタもサポートしています。このストリームの更新通知は、通知の対象になるイベントが発生した場合にのみ送信されます。

このストリームは変更時サブスクリプションのみをサポートしているため、ダンプニング間隔として値0を指定する必要があります。

## **yang-notif-native の XPath フィルタ**

サブスクライブ先の yang-notif-native ストリーム内のデータセットは、XPath フィルタを使用して指定します。XPath 式には次のガイドラインが適用されます。

- XPath 式は YANG 通知全体を指定する必要があります。属性のフィルタ処理はサポートされていません。
- ユニオン演算子 (|) はサポートされていません。

## **転送プロトコル**

データの送信方法は、パブリッシャとレシーバー間の接続に使用されるプロトコルによって決まります。このプロトコルはトランスポート プロトコルと呼ばれ、設定済みサブスクリプションの管理プロトコルからは独立しています。トランスポートプロトコルは、データのエンコーディング（XML、Google Protocol Buffers（GPB）など）と更新通知自体の形式に影響を与えます。gNMI、gRPC、および NETCONF はサポートされているトランスポートプロトコルです。



(注)

また、選択したストリームも更新通知の形式に影響を与える場合があります。

## NETCONF プロトコル

NETCONF プロトコルは、動的サブスクリプションのトランスポートにのみ使用でき、*yang-push* ストリームと *yang-notif-native* ストリームで使用できます。

NETCONF をトランスポート プロトコルとして使用する場合は、次の 3 つの更新通知形式が使用されます。

- サブスクリプションで *yang-push* ストリームが使用されていて、定期的な場合、または、初期同期更新通知が変更時サブスクリプションで送信される場合。
- サブスクリプションで *yang-push* ストリームが使用されていて、初期同期更新通知以外の変更時サブスクリプションの場合。
- サブスクリプションで *yang-notif-native* ストリームが使用されている場合。

### yang-push 形式

*yang-push* ソースストリームが NETCONF を介して XML エンコーディングのトランスポートとして送信される場合、2 つの更新通知形式が定義されます。これらの更新通知形式は、*draft-ietf-netconf-yang-push-07*に基づいています。詳細については、IETF ドラフトの 3.7 項を参照してください。

### yang-notif-native 形式

ソース ストリームが *yang-notif-native* の場合、NETCONF を介して XML でエンコードされときの更新通知の形式は RFC 7950 によって定義されています。詳細については、RFC の 7.16.2 項を参照してください。

*yang-push* ストリームの形式とは異なり、サブスクリプション ID は更新通知にはありません。

## gRPC プロトコル

gRPC プロトコルは、設定されたサブスクリプションのトランスポートにのみ使用でき、*yang-push* ストリームと *yang-notif-native* ストリームで使用できます。gRPC トランスポートプロトコルでは kvGPB エンコーディングのみがサポートされています。

gRPC プロトコルに基づく受信者の接続の再試行（指数バックオフ）がサポートされています。

proto ファイルで定義されたテレメトリメッセージについては、`mdt_grpc_dialout.proto` および <https://github.com/cisco-ie/cisco-proto/blob/9cc3967cb1cabbb3e9f92f2c46ed96edf8a0a78b/proto/xe/telemetry.prototelemetry.proto> を参照してください。

### gRPC テレメトリの相互認証

gRPC は、テレメトリデータの送信に使用できるダイヤルアウトプロトコルの 1 つです。ダイヤルアウトプロトコルでは、デバイスはクライアントと見なされ、コレクタはサーバーと見なされます。gRPC は、暗号化されていない TCP 接続と暗号化された TLS ベースの接続の両方をサポートします。

相互認証にクライアント ID 証明書を使用できるように、トラストポイントのペアを含む新しい gRPC-TLS プロファイルがテレメトリ構成に追加されました。プロファイルには 2 つのトラストポイントが含まれています。1 つはサーバー検証用の認証局（CA）証明書で、もう 1 つはクライアント検証用の ID 証明書です。

デバイスが受信者に初めて接続するとき、サーバーの設定に基づいて、クライアント認証または相互認証が必要になる場合があります。デバイスは受信者の ID 証明書を受信すると、その証明書が、受信者プロファイルで設定されたトラストポイントに関連付けられた証明書で識別された CA によって署名されているかどうかを検証します。次に、受信者がデバイスの証明書 ID を要求すると、デバイスは、プロファイルの ID トラストポイントフィールドに以前にインストールされたクライアント ID 証明書を送信します。

相互認証を要求するようにサーバーが設定されていて、プロファイルにクライアント ID トラストポイントがない場合、クライアント認証は行われず、接続も成功しません。

同じトラストポイントラベルを複数のプロファイルに設定でき、同じプロファイルを複数の受信者に設定できます。



TLS を使用した gRPC では相互認証が必要ではないため、プロファイル設定ではクライアント ID を持つトラストポイントが必須ではありません。プロファイルは、サーバー検証でのみ設定できます。

クライアント ID トラストポイントを追加するには、**telemetry protocol grpc profile <name>** コマンドを使用します。

gRPC テレメトリの相互認証は無効にできません。ただし、gRPC-TLS プロトコルを使用するようにレシーバーを設定しないか、レシーバーの設定でクライアント ID トラストポイントフィールドを削除するか設定しないことで、未使用のままにすることができます。

次に、相互認証用の gRPC-TLS プロファイルを設定する例を示します。

```
Device# configure terminal
Device(config)# telemetry receiver protocol grpc-mtls
Device(config-mdt-protocol-receiver)# host name collector.cisco.com 57500
Device(config-mdt-protocol-receiver)# protocol grpc-tls profile different-ca
Device(config-mdt-protocol-receiver)# exit
Device(config)# telemetry protocol grpc profile myprofile
Device(config-mdt-protocol-grpc-profile)# exit
Device(config)# telemetry ietf subscription 100
Device(config-mdt-subs)# encoding encode-kvgpb
Device(config-mdt-subs)# filter xpath /memory-ios-xe-oper:memory-statistics/memory-statistic
Device(config-mdt-subs)# receiver-type protocol
Device(config-mdt-subs)# stream yang-push
Device(config-mdt-subs)# update-policy periodic 5000
Device(config-mdt-subs)# receiver name grpc-mtls
Device(config-mdt-subs)# end
Device#
```

## サブスクリプション管理

管理操作の任意の形式を使用して、設定済みサブスクリプションの作成、削除、および変更を行うことができます。これには、CLI とネットワーク プロトコルの両方の管理操作が含まれます。

すべてのサブスクリプション（設定済みと動的）は、**show** コマンド、およびネットワーク プロトコル管理操作を使用して表示できます。

次の表で、サポートされているストリームとエンコーディング、およびサポートされている組み合わせについて説明します。

表 3: サポートされるプロトコルの組み合わせ

| トランスポート<br>プロトコル  | NETCONF |         | gRPC   |                                     | gNMI                                                                       |         |
|-------------------|---------|---------|--------|-------------------------------------|----------------------------------------------------------------------------|---------|
|                   | ダイヤルイン  | ダイヤルアウト | ダイヤルイン | ダイヤルアウト                             | ダイヤルイン                                                                     | ダイヤルアウト |
| ストリーム             |         |         |        |                                     |                                                                            |         |
| yang-push         | 対応      | 非対応     | 非対応    | 対応                                  | 対応                                                                         | 非対応     |
| yang-notif-native | 対応      | 非対応     | 非対応    | 対応                                  | 非対応                                                                        | 非対応     |
| エンコーディング          | XML     | 非対応     | 非対応    | キー値 Google Protocol Buffers (kvGPB) | <ul style="list-style-type: none"> <li>JSON_IETF</li> <li>PROTO</li> </ul> | 非対応     |

## NETCONF サブスクリプションの作成、変更、削除

確立された NETCONF セッションで YANG XML リモートプロシージャコール (RPC) を送受信し、サブスクリプションを作成、変更、および削除できます。

新しいサブスクリプションを確立するには、`establish-subscription` RPC を使用します。`establish-subscription` RPC が送信されると、パブリッシャからの RPC 応答には `rpc-reply` メッセージと、結果ストリングを含む `subscription-result` 要素が含まれます。

サブスクリプションを終了するには、`delete-subscription` RPC を使用して削除するか、`Kill-Subscription` RPC を使用して強制的に終了できます。`delete-subscription` RPC を使用すると、サブスクリバは、`establish-subscription` RPC を使用した同じサブスクリバによって以前に作成されたサブスクリプションを削除できます。

## 動的サブスクリプション

動的サブスクリプションでは、モニタリングするデータパスと更新を送信する頻度をクライアントが指定でき、現在の運用ニーズに合わせたリアルタイムのモニタリングが可能です。

サブスクリプションの存続期間は、サブスクリバとパブリッシャ間の接続の存続期間に制限され、テレメトリデータはそのサブスクリバにのみ送信されます。動的サブスクリプションは、パブリッシャまたはサブスクリバのいずれかが再起動された場合は存続しません。動的サブスクリプションはダイヤルイン サブスクリプションです。

動的サブスクリプションは、パブリッシャに接続するサブスクリバがその接続内部のメカニズム（通常はRPC）を使用することにより作成されます。

## 動的サブスクリプションの作成

動的サブスクリプションの作成にはインバンドの `establish-subscription` RPC を使用できます。この RPC は、IETF テレメトリのサブスクリバからネットワーク デバイスに送信されます。RPC では、`stream`、`XPath-filter`、および `period` の各フィールドが必須です。

NETCONF による動的サブスクリプションの作成および削除に使用する RPC は、[イベント通知のカスタムサブスクリプション draft-ietf-netconf-subscribed-notifications-03](#) および [YANG データストア プッシュ更新のサブスクリプション draft-ietf-netconf-yang-push-07](#) で定義されています。

## 定期的な動的サブスクリプション

定期動的サブスクリプションは、定期的に設定された時間間隔でネットワーク サーバからテレメトリ データの更新を要求する、クライアント開始型のサブスクリプションです。このサブスクリプションは、セッション(ダイヤルイン)中にサブスクライバまたはクライアントによって作成されるため、動的であり、サブスクライバとデバイスまたはパブリッシャとの接続がアクティブである間のみ有効です。

次に、NETCONF ダイヤルインの定期的なサブスクリプションの例を示します。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <establish-subscription
    xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
    xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <stream>yp:yang-push</stream>
    <yp:xpath-filter>/mdt-oper:mdt-oper-data/mdt-subscriptions</yp:xpath-filter>
    <yp:period>1000</yp:period>
  </establish-subscription>
</rpc>
```

## 変更時動的サブスクリプション

変更時動的サブスクリプションとは、一定の間隔ではなく、サブスクライブされたデータの値が変更された場合にのみ更新が送信されるテレメトリ サブスクリプションを指します。

次に、NETCONF を介した変更時動的サブスクリプションの例を示します。

```
<establish-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
  <stream>yp:yang-push</stream>
  <yp:xpath-filter>/cdp-ios-xe-oper:cdp-neighbor-details/cdp-neighbor-detail</yp:xpath-filter>
  <yp:dampening-period>0</yp:dampening-period>
</establish-subscription>
```

## gNMI ダイヤルイン サブスクリプション

次に、gNMI ダイヤルイン動的サブスクリプションの例を示します。

```
subscribe: <
  prefix: <>
  subscription: <
    path: <
      origin: "openconfig"
      elem: <name: "routing-policy">
    >
    mode: SAMPLE
    sample_interval: 10000000000
  >
  mode: STREAM
  encoding: JSON_IETF
```

```
>'
```

```
subscribe: <
  prefix: <>
  subscription: <
    path: <
      origin: "legacy"
      elem: <name: "oc-platform:components">
      elem: <
        name: "component"
        key: <
          key: "name"
          value: "PowerSupply8/A"
        >
      >
    elem: <name: "power-supply">
    elem: <name: "state">
  >
  mode: SAMPLE
  sample_interval: 10000000000
>
mode: STREAM
encoding: JSON_IETF
>'
```

## 応答メッセージの受信

サブスクリプションが正常に作成されると、デバイスはサブスクリプション結果 **notif-bis:ok** およびサブスクリプション ID で応答します。これは動的サブスクリプションの応答 RPC メッセージのサンプルです。

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<subscription-result xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
xmlns:notif-bis="urn:ietf:params:xml:ns:yang:ietf-event-notifications">notif-bis:
ok</subscription-result>
<subscription-id xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications">2147484201</subscription-id>
</rpc-reply>
```

## NETCONF ダイアラインのサブスクリプションプッシュ更新の受信

デバイスからプッシュされるサブスクリプション更新は XML RPC 形式であり、それらが作成された同じ NETCONF セッションにより送信されます。サブスクライブ対象情報の要素またはツリーは **datastore-contents-xml** タグ内で返されます。次に、サブスクライブ対象情報を提供する RPC メッセージの例を示します。

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2017-05-09T21:34:51.74Z</eventTime>
  <push-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <subscription-id>2147483650</subscription-id>
    <datastore-contents-xml>
      <cpu-usage xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-process-cpu-oper"><cpu-utilization>
        <five-minutes>5</five-minutes></cpu-utilization></cpu-usage>
    </datastore-contents-xml>
  </push-update>
```

```
</notification>
```

サブスクリプションが行われる情報要素が空である場合、またはそれが動的（名前付きアクセスリストなど）であり存在しない場合、定期更新は空になり、自己終結 *datastore-contents-xml* タグを持つことになります。次に、定期更新が空である RPC メッセージの例を示します。

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2017-05-09T21:34:09.74Z</eventTime>
  <push-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <subscription-id>2147483649</subscription-id>
    <datastore-contents-xml />
  </push-update>
</notification>
```

## 動的サブスクリプションの削除

動的サブスクリプションを変更することはできませんが、いつでも終了できます。セッションが終了すると、動的サブスクリプションは自動的に終了します。動的サブスクリプションでは、サブスクライバとレシーバーは常に同じエンティティになります。

動的サブスクリプションを削除するには、インバンドの delete-subscription RPC、**clear telemetry subscription dynamic** コマンド、kill-subscription RPC を使用し、トランスポートセッションを切断します。

gNMI では、SubscribeRequest.subscribe.subscription の各サブスクリプションが個別の動的サブスクリプション ID として生成されます。kill-subscription RPC または **clear telemetry subscription dynamic** コマンドを使用してこれらのサブスクリプション ID のいずれかを強制終了すると、サブスクライブ要求で指定されたすべてのサブスクリプションが強制終了されます。

delete-subscription RPC は、サブスクライバのみが発行でき、そのサブスクライバが所有するサブスクリプションのみを削除します。**clear telemetry subscription dynamic** コマンドを使用して、動的サブスクリプションを削除することもできます。kill-subscription RPC は、**clear telemetry subscription dynamic** コマンドと同じ方法で動的サブスクリプションを削除します。

NETCONF では、親の NETCONF セッションが切断されると、サブスクリプションも削除されます。ネットワーク接続が中断された場合は、SSH または NETCONF セッションがタイムアウトしてその後にサブスクリプションが削除されるまで、多少の時間がかかることがあります。

kill-subscription RPC は delete-subscription RPC と類似しています。ただし、kill-subscription RPC は、delete-subscription RPC で使用される *subscription-id* 要素の代わりに、削除するサブスクリプションの ID を含む *identifier* 要素を使用します。ターゲット サブスクリプションで使用されるトランスポートセッションも、delete-subscription RPC で使用されているものと異なります。

## CLI を使用したサブスクリプションの削除

**show telemetry ietf subscription all** コマンドの出力には、使用可能なすべてのサブスクリプションが表示されます。

```
Device# show telemetry ietf subscription all
```

```
Telemetry subscription brief
```

| ID | Type | State | Filter type |
|----|------|-------|-------------|
|----|------|-------|-------------|

```
-----
2147483648      Dynamic      Valid      xpath
2147483649      Dynamic      Valid      xpath
```

次に、動的サブスクリプションを削除する例を示します。

```
Device# clear telemetry subscription dynamic 2147483648
```

## NETCONF delete-subscription RPC を使用したサブスクリプションの削除

この例では、delete-subscription RPC を使用してサブスクリプションを削除する方法を示します。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <delete-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
    xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
    <subscription-id>2147483650</subscription-id>
  </delete-subscription>
</rpc>
```

## NETCONF kill-subscription RPC を使用したサブスクリプションの削除

次に、kill-subscription RPC を使用してサブスクリプションを削除する例を示します。

```
<get>
<filter>
<mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
<mdt-subscriptions/>
</mdt-oper-data>
</filter>
</get>
```

\* Enter a NETCONF operation, end with an empty line

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
  <data>
    <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
      <mdt-subscriptions>
        <subscription-id>2147483652</subscription-id>
        <base>
...
          </base>
          <type>sub-type-dynamic</type>
          <state>sub-state-valid</state>
          <comments/>
          <mdt-receivers>
...
        </mdt-receivers>
        <last-state-change-time>2018-12-13T21:16:48.848241+00:00</last-state-change-time>
      </mdt-subscriptions>
      <mdt-subscriptions>
        <subscription-id>2147483653</subscription-id>
        <base>
...
          </base>
          <type>sub-type-dynamic</type>
          <state>sub-state-valid</state>
          <comments/>
```

```

        <mdt-receivers>
...
    </mdt-receivers>
    <last-state-change-time>2018-12-13T21:16:51.319279+00:00</last-state-change-time>
</mdt-subscriptions>
<mdt-subscriptions>
    <subscription-id>2147483654</subscription-id>
    <base>
...
    </base>
    <type>sub-type-dynamic</type>
    <state>sub-state-valid</state>
    <comments/>
    <mdt-receivers>
...
    </mdt-receivers>
    <last-state-change-time>2018-12-13T21:16:55.302809+00:00</last-state-change-time>
</mdt-subscriptions>
<mdt-subscriptions>
    <subscription-id>2147483655</subscription-id>
    <base>
...
    </base>
    <type>sub-type-dynamic</type>
    <state>sub-state-valid</state>
    <comments/>
    <mdt-receivers>
...
    </mdt-receivers>
    <last-state-change-time>2018-12-13T21:16:57.440936+00:00</last-state-change-time>
</mdt-subscriptions>
</mdt-oper-data>
</data>
</rpc-reply>
<kill-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
    xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <identifier>2147483653</identifier>
</kill-subscription>

* Enter a NETCONF operation, end with an empty line

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
    <subscription-result xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications"
        xmlns:notif-bis="urn:ietf:params:xml:ns:yang:ietf-event-notifications">notif-bis:ok</subscription-result>
</rpc-reply>
<get>
<filter>
<mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
<mdt-subscriptions/>
</mdt-oper-data>
</filter>
</get>

* Enter a NETCONF operation, end with an empty line

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
    <data>
        <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
            <mdt-subscriptions>
                <subscription-id>2147483652</subscription-id>
                <base>
...

```

```

        </base>
        <type>sub-type-dynamic</type>
        <state>sub-state-valid</state>
        <comments/>
        <mdt-receivers>
...
        </mdt-receivers>
        <last-state-change-time>2018-12-13T21:16:48.848241+00:00</last-state-change-time>
    </mdt-subscriptions>
    <mdt-subscriptions>
        <subscription-id>2147483654</subscription-id>
        <base>
...
        </base>
        <type>sub-type-dynamic</type>
        <state>sub-state-valid</state>
        <comments/>
        <mdt-receivers>
...
        </mdt-receivers>
        <last-state-change-time>2018-12-13T21:16:55.302809+00:00</last-state-change-time>
    </mdt-subscriptions>
    <mdt-subscriptions>
        <subscription-id>2147483655</subscription-id>
        <base>
...
        </base>
        <type>sub-type-dynamic</type>
        <state>sub-state-valid</state>
        <comments/>
        <mdt-receivers>
...
        </mdt-receivers>
        <last-state-change-time>2018-12-13T21:16:57.440936+00:00</last-state-change-time>
    </mdt-subscriptions>
    </mdt-oper-data>
    </data>
</rpc-reply>

```

## サービス gNMI

サービス gRPC ネットワーク管理インターフェイス (gNMI) は、gRPC (リモート プロシージャ コール フレームワーク) を使用してネットワークデバイスを管理する、Googleによって開発されたネットワーク管理プロトコルです。ネットワークデバイスの設定、動作状態データの取得、およびテレメトリデータのストリーミングのための統合サービスを提供します。

gNMI仕様は、ハイレベルRPCを含むgNMIという名前の単一のトップレベルサービスを識別します。これは、サブスクリプションサービス RPC を含むサービス定義です。

```

service gNMI {
    .
    .
    .
    rpc Subscribe(stream SubscribeRequest)
        returns (stream SubscribeResponse);
}

```

## SubscribeRequest メッセージ

このメッセージは、指定されたパスのセットに対するターゲットからの更新を要求するためにクライアントによって送信されます。これは、SubscribeRequest メッセージ定義のサンプルです。

```
message SubscribeRequest {
  oneof request {
    SubscriptionList subscribe = 1;
    PollRequest poll = 3;
    AliasList aliases = 4;
  }
  Repeated gNMI_ext.Extensions = 5;
}
```



(注)

---

request.subscribe のみがサポートされます。

---

## SubscribeResponse メッセージ

このメッセージは、確立された subscribe RPC を介してターゲットからクライアントに送信されます。これは、SubscribeResponse メッセージ定義のサンプルです。

```
message SubscribeResponse {
  oneof response {
    Notification update = 1;
    Bool sync_response = 3;
    Error error = 4 [deprecated=true];
  }
}
```



(注)

---

通知の更新のみがサポートされます。

---

## gNMI sync\_response メッセージ

SubscribeResponse メッセージは、確立された gNMI subscribe RPC を介して、ターゲット（スイッチまたはルータ）から gNMI クライアントまたはコレクタに送られます。sync\_response は、SubscribeResponse メッセージの一部であるブールフィールドで、SubscriptionList 内のパスに対応するすべてのデータ値が少なくとも 1 回送信されたことを示します。sync\_response メッセージは最初の更新メッセージ後に送信され、このフィールドは gNMI の変更時通知と定期的な通知の両方で有効になっています。

次のサンプル SubscribeResponse メッセージには、sync\_response フィールドが表示されています。

```
message SubscribeResponse {
  oneof response {
    Notification update = 1; // Changed or sampled value for a path.
    // Indicate target has sent all values associated with the subscription
    // at least once.
  }
}
```

```

    bool sync_response = 3;
    // Deprecated in favour of google.golang.org/genproto/googleapis/rpc/status
    Error error = 4 [deprecated = true];
}
// Extension messages associated with the SubscribeResponse. See the
// gNMI extension specification for further definition.
repeated gnmi_ext.Extension extension = 5;
}

```

## SubscriptionList メッセージ

このメッセージは、共通のサブスクリプション動作が必要なパスのセットを示すために使用されます。SubscriptionList メッセージの仕様内で、クライアントはモデル内の特定のプレフィックスに対する1つ以上のサブスクリプションを識別できます。これは、SubscriptionList メッセージの定義のサンプルです。

```

message SubscriptionList {
    Path prefix = 1;
    repeated Subscription subscription = 2;
    bool use_aliases = 3;
    QOSMarking qos = 4;
    enum Mode {
        STREAM = 0;
        ONCE = 1;
        POLL = 2;
    }
    Mode mode = 5;
    bool allow_aggregation = 6;
    repeated ModelData use_models = 7;
    Encoding encoding = 8;
    Bool updates_only = 9;
}

```



(注)

---

Path prefix (明示的な要素名のみ)、Subscription subscription、Mode mode STREAM、および Encoding encoding IETF\_JSON がサポートされています。

---

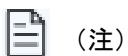
## プレフィックスメッセージ

有効なサブスクリプションリストには、XPath の (要求されたすべてのサブスクリプション間で) 共有部分で構成された入力済みのプレフィックスが含まれている場合と、含まれていない場合があります。

```

message Path {
    repeated string element = 1; [ deprecated ]
    string origin = 2;
    repeated PathElem elem = 3;
    optional string target = 4;
}

```



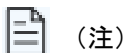
origin (サポートされる値は「openconfig」、「legacy」、および「rfc7951」)、elem (サポートされる要素名はプレフィックスなし)、および target がサポートされます。

「legacy」は、YANG モジュールプレフィックスをパスで使用する必要があることを示し、origin 「rfc7951」は、モジュール名プレフィックスがパスで使われる必要があることを示します。

## サブスクリプションメッセージ

このメッセージは、クライアントによってサブスクライブされるデータのセットを一般的に説明しています。通知動作を制御するために使用されるパスと属性が含まれます。これはサンプルのサブスクリプションメッセージ定義です。

```
message Subscription {  
    Path path = 1;  
    SubscriptionMode mode = 2;  
    uint64 sample_interval = 3;  
    bool suppress_redundant = 4;  
    uint64 heartbeat_interval = 5;  
}
```

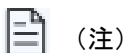


Path path、SubscriptionMode mode、Uint64 sample\_interval がサポートされます。

## パスメッセージ

有効なサブスクリプションには、パスが入力されています。これは、サブスクリプションリストに関連付けられたプレフィックスに追加されると、完全修飾パスを構成します。これは、サンプルのパスメッセージ定義です。

```
message Path {  
    repeated string element = 1; [ deprecated ]  
    string origin = 2;  
    repeated PathElem elem = 3;  
    optional string target = 4;  
}
```



origin (サポートされる値は「」と「openconfig」)、elem (サポートされる要素名はプレフィックスなし)、および target がサポートされます。

## SubscriptionMode メッセージ

このメッセージは、通知の更新をトリガーする方法をターゲットに通知します。これはサンプルの SubscriptionMode メッセージ定義です。

```
enum SubscriptionMode {  
    TARGET_DEFINED = 0;  
    ON_CHANGE      = 1;  
    SAMPLE        = 2;  
}
```



(注)

---

ON\_CHANGE と SAMPLE のみがサポートされています。

ON\_CHANGE のサポートは、特定のモデルパスに限定されています。パスが ON\_CHANGE をサポートしているかどうかを確認するには、Cisco-IOS-XE-MDT-capabilities-oper モデルでパスをクエリします。このモデルの詳細については、「変更時サブスクリプション機能の表示」セクションを参照してください。

---

## 通知メッセージ

このメッセージは、サブスクリプションターゲットからコレクタにテレメトリデータを配信します。これは通知メッセージの定義です。

```
message Notification {  
    int64 timestamp = 1;  
    Path prefix    = 2;  
    string alias    = 3;  
    repeated Update update = 4;  
    repeated Path delete = 5;  
    bool atomic      = 6;  
}
```



(注)

---

タイムスタンプ、プレフィックス、更新、および削除（主に変更時サブスクリプションに使用）がサポートされています。

---

## 設定済みサブスクリプション

設定済みサブスクリプションは、コントローラによるパブリッシャでの管理操作によって作成されたテレメトリまたはイベントのサブスクリプションで、サブスクリプションによって定義されたテレメトリデータのレシーバーの指定が明示的に含まれています。これらのサブスクリプションは、デバイスが再起動したりトランスポートセッションが中断されたりしても存続します。

設定済みサブスクリプションは複数の受信者を使用して設定できますが、最初の有効な受信者のみが使用されます。受信者がすでに接続されている場合、または接続中の場合は、他の受信者への接続は試行されません。その受信者が削除されると、別の受信者が接続されます。

設定されたサブスクリプションはダイヤルアウトサブスクリプションであり、これらのサブスクリプションは次の操作によってデバイスで設定されます。

- 設定 CLI を使用し、コンソール/VTY を使用してデバイス設定に変更を加えます。

- NETCONF または RESTCONF を使用し、目的のサブスクリプションを設定します。

Cisco-IOS-XE-mdt-cfg.yang モデルまたは ietf-event-notifications.yang モデルを使用して、サブスクリプションを設定します。

## 設定済みサブスクリプションの作成

このセクションでは、設定済みサブスクリプションを作成するための RPC の例を示します。

### 定期サブスクリプション

次の例は、CLI を使用して、設定済みサブスクリプションのトランスポートプロトコルとして gRPC を設定する方法を示しています。

```
telemetry ietf subscription 101
  encoding encode-kvgpb
  filter xpath /memory-ios-xe-oper:memory-statistics/memory-statistic
  stream yang-push
  update-policy periodic 6000
  source-vrf Mgmt-intf
  receiver ip address 10.28.35.45 57555 protocol grpc-tcp
```

次の RPC の例は、NETCONF を使用して定期的なサブスクリプションを作成し、60 秒ごとにテレメトリの更新を受信者に送信する方法を示します。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><edit-config>
  <target>
    <running/>
  </target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
      <mdt-subscription>
        <subscription-id>200</subscription-id>
        <base>
          <stream>yang-push</stream>
          <encoding>encode-kvgpb</encoding>
          <period>6000</period>
          <xpath>/memory-ios-xe-oper:memory-statistics/memory-statistic</xpath>
        </base>
        <mdt-receivers>
          <address>10.22.23.48</address>
          <port>57555</port>
          <protocol>grpc-tcp</protocol>
        </mdt-receivers>
      </mdt-subscription>
    </mdt-config-data>
  </config>
</edit-config>
</rpc>
```

次に、RESTCONF を使用して定期的なサブスクリプションを作成する RPC の例を示します。

```
URI:https://10.85.116.28:443/restconf/data/Cisco-IOS-XE-mdt-cfg:mdt-config-data
Headers:
application/yang-data.collection+json, application/yang-data+json, application/yang-data.errors+json
Content-Type:
```

```

application/yang-data+json
BODY:
{
  "mdt-config-data": {
    "mdt-subscription": [
      {
        "subscription-id": "102",
        "base": {
          "stream": "yang-push",
          "encoding": "encode-kvgpb",
          "period": "6000",
          "xpath": "/memory-ios-xe-oper:memory-statistics/memory-statistic"
        }
        "mdt-receivers": {
          "address": "10.22.23.48"
          "port": "57555"
        }
      }
    ]
  }
}

```

## 変更時サブスクリプション

次の RPC の例は、NETCONF を使用して変更時サブスクリプションを作成し、ターゲットデータベースに変更が生じた場合にのみ更新を送信する方法を示します。

```

<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><edit-config>
  <target>
    <running/>
  </target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
      <mdt-subscription>
        <subscription-id>200</subscription-id>
        <base>
          <stream>yang-push</stream>
          <encoding>encode-kvgpb</encoding>
          <no-synch-on-start-v2>false</no-synch-on-start-v2>
          <xpath>/cdp-ios-xe-oper:cdp-neighbor-details/cdp-neighbor-detail</xpath>
        </base>
        <mdt-receivers>
          <address>10.22.23.48</address>
          <port>57555</port>
          <protocol>grpc-tcp</protocol>
        </mdt-receivers>
      </mdt-subscription>
    </mdt-config-data>
  </config>
</edit-config>
</rpc>

```

次の RPC の例は、RESTCONF を使用して変更時サブスクリプションを作成する方法を示します。

```

URI:
https://10.85.116.28:443/restconf/data/Cisco-IOS-XE-mdt-cfg:mdt-config-data
Headers:
application/yang-data.collection+json, application/yang-data+json, application/yang-data.errors+json
Content-Type:
application/yang-data+json
BODY:
{

```

```

"mdt-config-data": {
  "mdt-subscription": [
    {
      "subscription-id": "102",
      "base": {
        "stream": "yang-push",
        "encoding": "encode-kvgpb",
        "dampening period": "0",
        "xpath": "/cdp-ios-xe-oper:cdp-neighbor-details/cdp
                  -neighbor-detail "
      }
    }
  ],
  "mdt-receivers": {
    "address": "10.22.23.48"
    "port": "57555"
  }
}
]
}
}

```

## gRPC の変更時サブスクリプションの設定

gRPC の変更時サブスクリプションを設定するには、このタスクを実行します。

### ステップ 1 enable

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

### ステップ 2 configure terminal

例：

```
Device# configure terminal
```

グローバル 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 telemetry ietf subscription *id*

例：

```
Device(config)# telemetry ietf subscription 8
```

テレメトリのサブスクリプションを作成し、テレメトリサブスクリプション モードを開始します。

### ステップ 4 stream yang-push

例：

```
Device(config-mdt-subs)# stream yang-push
```

サブスクリプションのストリームを設定します。

### ステップ 5 filter xpath *path*

例：

```
Device(config-mdt-subs)# filter xpath  
/iosxe-oper:ios-oper-db/hwidb-table
```

サブスクリプションの XPath フィルタを指定します。

#### ステップ 6 **update-policy {on-change | periodic period}**

例：

```
Device(config-mdt-subs)# update-policy on-change
```

サブスクリプションの変更時更新ポリシーを設定します。

#### ステップ 7 **encoding encode-kvgpb**

例：

```
Device(config-mdt-subs)# encoding encode-kvgpb
```

kvGPB エンコードを指定します。

#### ステップ 8 **receiver ip address ip-address receiver-port protocol protocol profile name**

例：

```
Device(config-mdt-subs)# receiver ip address 10.22.22.45 45000 protocol  
grpc_tls profile secure_profile
```

通知の受信者の IP アドレス、プロトコル、およびプロファイルを設定します。

#### ステップ 9 **end**

例：

```
Device(config-mdt-subs)# end
```

テレメトリサブスクリプションのコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

## 設定済みサブスクリプションの変更

サブスクリプションは、以下を使用して変更できます。

- NETCONF edit-config RPC などの管理プロトコル設定操作
- CLI（サブスクリプションの作成と同じ手順）

名前付きレシーバーを使用してサブスクリプションを変更します。詳細については、[設定済みサブスクリプションの名前付きレシーバー（32 ページ）](#) のセクションを参照してください。

有効なサブスクリプションの有効な受信者設定が切断状態にあり、管理側で受信者への接続のセットアップ時に新しい試行を強制する場合は、同一の特性を持つ受信者を書き換える必要があります。

## 設定済みサブスクリプションの削除

CLI または管理操作を使用して、設定済みサブスクリプションを削除できます。**no telemetry ietf subscription** コマンドは、設定済みサブスクリプションを削除します。設定されたサブスクリプションは、**kill-subscription** または **delete-subscription** RPC を使用して削除することはできません。設定インターフェイスからのみ削除できます。

## NETCONF を使用したサブスクリプションの削除

次の RPC の例は、設定済みサブスクリプションを削除する方法を示しています。

```
<edit-config>
  <target>
    <running/>
  </target>
  <config>
    <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
      <mdt-subscription operation="delete">
        <subscription-id>102</subscription-id>
      </mdt-subscription>
    </mdt-config-data>
  </config>
</edit-config>
```

## CLI を使用したサブスクリプションの削除

次に、サブスクリプションを削除する例を示します。

```
Device# configure terminal
Device(config)# no telemetry ietf subscription 101
Device(config)# end
```

## 設定済みサブスクリプションの管理

設定済みサブスクリプションを管理するには、このタスクを実行します。



(注)

gRPC は、サブスクリプションのトランスポートとして使用できますが、サブスクリプションの管理には使用できません。

### ステップ 1 enable

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

### ステップ 2 configure terminal

例：

```
Device# configure terminal
```

グローバル 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 telemetry ietf subscription id

例：

```
Device(config)# telemetry ietf subscription 101
```

テレメトリのサブスクリプションを作成し、テレメトリサブスクリプション モードを開始します。

#### ステップ 4 **stream yang-push**

例 :

```
Device(config-mdt-subs)# stream yang-push
```

サブスクリプションのストリームを設定します。

#### ステップ 5 **filter xpath path**

例 :

```
Device(config-mdt-subs)# filter xpath  
/memory-ios-xe-oper:memory-statistics/memory-statistic
```

サブスクリプションの XPath フィルタを指定します。

#### ステップ 6 **update-policy {on-change | periodic} period**

例 :

```
Device(config-mdt-subs)# update-policy periodic 6000
```

サブスクリプションの定期的な更新ポリシーを設定します。

#### ステップ 7 **encoding encode-kvgpb**

例 :

```
Device(config-mdt-subs)# encoding encode-kvgpb
```

key-value Google Protocol Buffers (kvGPB) エンコーディングを指定します。

- kvGPB は、GPB テクノロジーを使用してデータをキーと値のペアとして表すデータエンコーディング形式です。このエンコーディングでは、データがキーと値のペアとして編成され、各キーは対応する値にマッピングされます。

#### ステップ 8 **source-vrf vrf-id**

例 :

```
Device(config-mdt-subs)# source-address Mgmt-intf
```

ソースの VRF インスタンスを設定します。

#### ステップ 9 **source-address source-address**

例 :

```
Device(config-mdt-subs)# source-vrf 192.0.2.1
```

送信元アドレスを設定します。

#### ステップ 10 **receiver ip address ip-address receiver-port protocol protocol profile name**

例 :

```
Device(config-mdt-subs)# receiver ip address 10.28.35.45  
57555 protocol grpc-tcp
```

通知の受信者の IP アドレス、プロトコル、およびプロファイルを設定します。

## ステップ 11 end

例：

```
Device(config-mdt-subs)# end
```

テレメトリサブスクリプションのコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

# サブスクリプションの詳細の取得

## get RPC

このセクションでは、現在のサブスクリプションのリストを取得する方法と、サブスクリプションをモニターする方法について説明します。

現在のサブスクリプションの一覧を取得するには、get RPC を Cisco-IOS-XE-mdt-oper-v2 に送信します。これはサンプルの get RPC メッセージです。

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <filter>
      <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
        <mdt-subscriptions/>
      </mdt-oper-data>
    </filter>
  </get>
</rpc>
```

これはサンプルの RPC 応答です。

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <data>
    <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
      <mdt-subscriptions>
        <subscription-id>2147485164</subscription-id>
        <base>
          <stream>yang-push</stream>
          <encoding>encode-xml</encoding>
          <period>100</period>
          <xpath>/ios:native/router/ios-rip:rip/ios-rip:version</xpath>
        </base>
        <type>sub-type-dynamic</type>
        <state>sub-state-valid</state>
        <comments/>
        <updates-in>0</updates-in>
        <updates-dampened>0</updates-dampened>
        <updates-dropped>0</updates-dropped>
      </mdt-subscriptions>
    </mdt-oper-data>
  </data>
</rpc-reply>
```

サブスクリプションの詳細は、Cisco-IOS-XE-mdt-oper-v2 データベースへの RESTCONF GET 要求によっても取得できます。このサンプル RPC は、RESTCONF を使用してサブスクリプションの詳細を取得する方法を示しています。

```
URI:
https://10.85.116.28:443/restconf/data/Cisco-IOS-XE-mdt-oper-v2:mdt-oper-data/mdt-subscriptions
Headers:
application/yang-data.collection+json, application/yang-data+json, application/yang-data.errors+json
Content-Type:
application/yang-data+json
Returned output:
{
  "Cisco-IOS-XE-mdt-oper-v2:mdt-subscriptions": [
    {
      "subscription-id": 101,
      "base": {
        "stream": "yang-push",
        "encoding": "encode-kvgpb",
        "source-vrf": "",
        "no-synch-on-start-v2": false,
        "xpath": "/iosxe-oper:ios-oper-db/hwldb-table"
      },
      "type": "sub-type-static",
      "state": "sub-state-valid",
      "comments": "",
      "updates-in": "0",
      "updates-dampened": "0",
      "updates-dropped": "0",
      "mdt-receivers": [
        {
          "address": "10.28.35.35",
          "port": 57555,
          "protocol": "grpc-tcp",
          "state": "rcvr-state-connecting",
          "comments": "Connection retries in progress",
          "profile": ""
        }
      ]
    }
  ]
}
```

### show telemetry ietf subscription コマンド

現在のサブスクリプションの一覧を表示するには、**show telemetry ietf subscription** コマンドも使用できます。

これは **show telemetry ietf subscription dynamic brief** コマンドのサンプル出力です。

```
Device# show telemetry ietf subscription dynamic brief
```

```
Telemetry subscription brief
```

| ID         | Type    | State | Filter type |
|------------|---------|-------|-------------|
| 2147483667 | Dynamic | Valid | xpath       |
| 2147483668 | Dynamic | Valid | xpath       |
| 2147483669 | Dynamic | Valid | xpath       |

これは **show telemetry ietf subscription subscription-IDdetail** コマンドのサンプル出力です。

```
Device# show telemetry ietf subscription 2147483667 detail
```

Telemetry subscription detail:

```
Subscription ID: 2147483667
State: Valid
Stream: yang-push
Encoding: encode-xml
Filter:
  Filter type: xpath
  XPath: /mdt-oper:mdt-oper-data/mdt-subscriptions
Update policy:
  Update Trigger: periodic
  Period: 1000
Notes:
```

これは **show telemetry ietf subscription all detail** コマンドのサンプル出力です。

```
Device# show telemetry ietf subscription all detail
```

Telemetry subscription detail:

```
Subscription ID: 101
Type: Configured
State: Valid
Stream: yang-push
Encoding: encode-kvgpb
Filter:
  Filter type: xpath
  XPath: /iosxe-oper:ios-oper-db/hwldb-table
Update policy:
  Update Trigger: on-change
  Synch on start: Yes
  Dampening period: 0
Notes:
```

## サブスクリプションのモニタリング

パブリッシャーとサブスクライバー間のテレメトリサブスクリプションを管理するプロセスは、サブスクリプションモニタリングと呼ばれます。これには、テレメトリデータストリームのステータス、正常性、およびパフォーマンスの追跡が含まれます。CLIおよび管理プロトコル操作を使用して、すべてのタイプのサブスクリプションを監視できます。

## モデルを使用したモニタリング

次に、テレメトリのサブスクリプションに関する情報を表示する NETCONF メッセージの例を示します。

```
<get>
<filter>
<mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
<mdt-subscriptions/>
</mdt-oper-data>
</filter>
</get>
```

```

* Enter a NETCONF operation, end with an empty line
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
  <data>
    <mdt-oper-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
      <mdt-subscriptions>
        <subscription-id>101</subscription-id>
        <base>
          <stream>yang-push</stream>
          <encoding>encode-kvgpb</encoding>
          <source-vrf>RED</source-vrf>
          <period>10000</period>
          <xpath>/ios:native/interface/Loopback[name="1"]</xpath>
        </base>
        <type>sub-type-static</type>
        <state>sub-state-valid</state>
        <comments/>
        <mdt-receivers>
          <address>5.22.22.45</address>
          <port>57500</port>
          <protocol>grpc-tcp</protocol>
          <state>rcvr-state-connecting</state>
          <comments/>
          <profile/>
          <last-state-change-time>1970-01-01T00:00:00+00:00</last-state-change-time>
        </mdt-receivers>
        <last-state-change-time>1970-01-01T00:00:00+00:00</last-state-change-time>
      </mdt-subscriptions>
      <mdt-subscriptions>
        <subscription-id>2147483648</subscription-id>
        <base>
          <stream>yang-push</stream>
          <encoding>encode-xml</encoding>
          <source-vrf/>
          <period>1000</period>
          <xpath>/if:interfaces-state/interface[name="GigabitEthernet0/0"]/oper-status</xpath>
        </base>
        <type>sub-type-dynamic</type>
        <state>sub-state-valid</state>
        <comments/>
        <mdt-receivers>
          <address>5.22.22.45</address>
          <port>51259</port>
          <protocol>netconf</protocol>
          <state>rcvr-state-connected</state>
          <comments/>
          <profile/>
          <last-state-change-time>1970-01-01T00:00:00+00:00</last-state-change-time>
        </mdt-receivers>
        <last-state-change-time>1970-01-01T00:00:00+00:00</last-state-change-time>
      </mdt-subscriptions>
    </mdt-oper-data>
  </data>
</rpc-reply>

```

## CLI を使用したモニタリング

テレメトリのサブスクリプションに関する情報を表示するには、**show telemetry ietf subscription** コマンドを使用します。これはコマンドのサンプル出力です。

```
Device# show telemetry ietf subscription 2147483667 detail
```

```
Telemetry subscription detail:
```

```
Subscription ID: 2147483667
State: Valid
Stream: yang-push
Encoding: encode-xml
Filter:
  Filter type: xpath
  XPath: /mdt-oper:mdt-oper-data/mdt-subscriptions
Update policy:
  Update Trigger: periodic
  Period: 1000
Notes:
```

## 設定済みサブスクリプションの名前付きレシーバー

完全修飾ドメイン名 (FQDN) のサポートにより、名前付きレシーバー設定と呼ばれる新しいレシーバー設定方法が導入されました。名前付きレシーバーは、サブスクリプションとは無関係に存在できるトップレベルの設定エンティティです。名前付きレシーバーは名前で識別されます。この名前は任意の文字列で、システムで名前付きレシーバレコードのインデックスまたはキーとなります。名前付きレシーバー設定には、レシーバーに関連付けられているすべての設定が含まれます。これらの設定はサブスクリプションに依存しません。

名前付きレシーバーは、以下を行うことができます。

- さまざまなタイプのレシーバーのサポート
- より正確な状態および診断情報の取得
- IP アドレスの代わりにホスト名を使用した、プロトコルレシーバーに対するホストの指定
- 複数のサブスクリプションで使用されるレシーバーのパラメータに対する、1 つの場所での変更

次のプロトコルタイプ名前付きレシーバーだけがサポートされます。

- grpc-tcp : gRPC TCP プロトコル
- grpc-tls : gRPC TLS プロトコル

## YANG モデルを使用した名前付きレシーバーの状態の表示

名前付きレシーバーの状態は、Cisco-IOS-XE-mdt-oper-v2 YANG モデルを使用して取得できます。mdt-oper-v2-data コンテナには、すべての名前付きレシーバーの動作状態を含む mdt-named-receivers リストが含まれています。

次に、名前付きレシーバーの状態を取得する NETCONF 応答の例を示します。

```
<get>
<filter>
  <mdt-oper-v2-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
    <mdt-named-receivers/>
  </mdt-oper-v2-data>
</filter>
</get>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
  <data>
    <mdt-oper-v2-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-oper-v2">
      <mdt-named-receivers>
        <name>my-receiver</name>
        <profile>tls-profile</profile>
        <params>
          <protocol>grpc-tls</protocol>
        </params>
      </mdt-named-receivers>
    </mdt-oper-v2-data>
  </data>
</rpc-reply>
```

## CLI を使用した名前付きレシーバーの状態の表示

すべてのタイプの名前付きレシーバーの状態を表示するには、**show telemetry receiver** コマンドを使用します。**all** キーワードは、すべての名前付きレシーバーに関する情報を簡単な形式で表示し、**name** キーワードは、指定された名前付きレシーバーに関する詳細情報を表示します。

次に、**show telemetry receiver all** コマンドの出力例を示します。

```
Device# show telemetry receiver all
```

```
Telemetry receivers
```

| Name      | <...> | Type     | Profile        | State | Explanation |
|-----------|-------|----------|----------------|-------|-------------|
| receiver1 | <...> | protocol | tls-trustpoint | Valid |             |

次に、**show telemetry receiver name** コマンドの出力例を示します。

```
Device# show telemetry receiver name my-receiver
```

```
Name: my-receiver
Profile: tls-profile
State: Valid
State Description:
Last State Change: 09/18/24 15:50:54
Type: protocol
Protocol: grpc-tls
Host: collector.cisco.com
```

## 名前付きプロトコルレシーバー

名前付きプロトコルレシーバーは、プロトコルを使用するテレメトリの転送を指定します。名前付きプロトコルレシーバーは、プッシュ通知またはテレメトリデータの接続先を表す明示的に定義された再利用可能なレシーバーエンティティ

ティを指します。名前付きプロトコルレシーバは、レシーバを識別する名前に加えて、ホスト指定も使用します。ホスト指定には、ホスト名またはIPアドレス、および宛先ポート番号を指定します。各サブスクリプション内でレシーバー接続の詳細を直接指定する代わりに、名前付きプロトコルレシーバーでは、これらの詳細を一度定義し、複数のサブスクリプション内で名前によって参照できます。セキュアプロトコル転送もプロファイル文字列を使用します。



有効な名前付きプロトコルレシーバが作成されると、レシーバに自動的に接続されません。レシーバへの接続を作成するには、指定されたプロトコルレシーバが、少なくとも1つのサブスクリプションによって要求される必要があります。

## YANG モデルを使用した名前付きプロトコルレシーバーの設定

CLIまたはYANGモデルを使用して、名前付きプロトコルレシーバーを設定できます。YANGモデルCisco-IOS-XE-mdt-cfgには、名前付きプロトコルレシーバが含まれています。最上位の `mdt-config-data` コンテナ内のコンテナ `mdt-named-protocol-rcvrs` には、`mdt-named-protocol-rcvr` 構造体のリストがあります。

このグループには、次の5つのメンバーがあります。

- `hostname`
- リストキーである名前
- ポート番号
- プロトコル
- プロファイル

次に、NETCONF RPC の例を示します。名前付きプロトコルレシーバーの作成方法を示しています。

```
<edit-config>
<target>
  <running/>
</target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
    <mdt-named-protocol-rcvrs>
      <mdt-named-protocol-rcvr>
        <name>my-receiver</name>
        <protocol>grpc-tls</protocol>
        <profile>tls-profile</profile>
        <host>
          <hostname>collector.cisco.com</hostname>
        </host>
        <port>57500</port>
      </mdt-named-protocol-rcvr>
    </mdt-named-protocol-rcvrs>
  </mdt-config-data>
</config>
</edit-config>
```

## 名前付きプロトコルレシーバーの設定

名前付きプロトコルレシーバーを設定するには、このタスクを実行します。

### ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

### ステップ 2 **configure terminal**

例：

```
Device# configure terminal
```

グローバル 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 **telemetry receiver protocol receiver-name**

例：

```
Device(config)# telemetry receiver protocol receiver1
```

名前付きプロトコルレシーバを設定し、テレメトリ プロトコルレシーバ 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 **protocol {cloud-native | cntp-tcp | cntp-tls profile profile-name | grpc-tcp | grpc-tls profile profile-name | native | tls-native profile profile-name}**

例：

```
Device(config-mdt-protocol-receiver)# protocol grpc-tcp
```

名前付きプロトコルレシーバ接続のプロトコルを設定します。

### ステップ 5 **host {ip ip-address | name hostname} receiver-port**

例：

```
Device(config-mdt-protocol-receiver)# host name rcvr.test.com 45000
```

名前付きプロトコルレシーバのホスト名を設定します。

### ステップ 6 **end**

例：

```
Device(config-mdt-protocol-receiver)# end
```

テレメトリ プロトコルレシーバ コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

## 指定レシーバーサブスクリプション

サブスクリプションで名前付きレシーバを使用するには、レシーバタイプとレシーバ名の両方を指定する必要があります。すべてのレシーバー固有の設定は名前付きレシーバー設定の一部であるため、追加のレシーバー設定は必要ありません。

せん。ただし、名前付きプロトコルレシーバーは、接続解決プロセスの一部として、サブスクリプションの送信元 VRF および送信元アドレスを引き続き使用します。

サポートされる名前レシーバータイプは *protocol* のみです。

サブスクリプションは名前付きレシーバーまたはレガシーレシーバーのいずれかを使用できますが、両方を使用することはできません。レガシーレシーバーが設定されている場合、サブスクリプション レシーバー タイプと名前付きレシーバー名の設定はブロックされます。同様に、サブスクリプション レシーバー タイプまたは名前付きレシーバーを指定した場合は、レガシーレシーバーを設定できません。



(注)

複数のレシーバーが設定されている場合でも、サブスクリプションは 1 つのレシーバーのみを使用することに注意してください。

レガシーレシーバーを使用するサブスクリプションと名前付きレシーバーを使用するサブスクリプションは、同じ接続を使用できます。ただし、これは推奨されません。

## 名前付きレシーバーの動作と動作状態

名前付きレシーバオブジェクトとサブスクリプション レシーバオブジェクト（名前付きレシーバーを参照する）には、2つの異なる動作状態があります。具体的には、有効または無効の動作状態をとる可能性があります。名前付きレシーバーが無効になる最も一般的な理由は、設定が不完全であることですが、他の理由も考えられます。名前付きレシーバーの動作状態ビューには、レシーバーが無効である理由をテキストで説明するフィールドがあります。レシーバーの状態が有効な場合、このフィールドは空です。

## YANG モデルを使用した名前付きレシーバーサブスクリプションの設定

名前付きレシーバーが使用される場合、*rcvr-type* でサポートされる唯一の値は *rcvr-type-protocol* で、レガシーレシーバーが使用される場合、値はデフォルトの *rcvr-type-unspecified* です。

次に、NETCONF RPC の例を示します。名前付きプロトコルレシーバーを使用してサブスクリプションを作成する方法を示しています。

```
<edit-config>
  <target>
    <running/>
  </target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <mdt-config-data xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-mdt-cfg">
      <mdt-subscription>
        <subscription-id>1</subscription-id>
        <base>
          <rcvr-type>rcvr-type-protocol</rcvr-type>
        </base>
        <mdt-receiver-names>
          <mdt-receiver-name>
            <name>receiver1</name>
          </mdt-receiver-name>
        </mdt-receiver-names>
      </mdt-subscription>
    </mdt-config-data>
  </config>
```

</edit-config>

## CLI を使用した名前付きレシーバサブスクリプションの設定

指定レシーバサブスクリプションを設定するには、次のタスクを実行します。

### ステップ 1 enable

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

### ステップ 2 configure terminal

例：

```
Device# configure terminal
```

グローバル 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 telemetry ietf subscription id

例：

```
Device(config)# telemetry ietf subscription 101
```

テレメトリのサブスクリプションを作成し、テレメトリサブスクリプション モードを開始します。

### ステップ 4 receiver-type protocol

例：

```
Device(config-mdt-subs)# receiver-type protocol
```

プロトコルタイプレシーバを設定します。

### ステップ 5 receiver name name

例：

```
Device(config-mdt-subs)# receiver name receiver1
```

通知ためのレシーバの名前を設定します。

### ステップ 6 end

例：

```
Device(config-mdt-subs)# end
```

テレメトリのテレメトリサブスクリプション モードを終了し、特権 EXEC モードに戻ります。

## 名前付きレシーバー接続のトラブルシューティング

サブスクリプションが設定されている場合、一般的な問題の1つは、テレメトリ更新メッセージが受信されないことです。原因として、送信するイベントがないか、サブスクリプションが無効であることが考えられます。ここでは、名前付きレシーバ接続で発生する一般的な問題のトラブルシューティング方法について説明します。

テレメトリプロセスからのログ、および一部の **show** コマンドの出力には、名前付きレシーバー設定のトラブルシューティングに使用できる情報が含まれています。

表 4: 名前付きレシーバー接続のトラブルシューティング

| 問題                            | チェック方法/症状                                                                                        | 対処方法                                                                           |
|-------------------------------|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| サブスクリプション構成が無効。               | <b>show telemetry ietf subscription ID details</b>                                               | サブスクリプションの設定を修正します。                                                            |
| サブスクリプションレシーバが無効。             | <b>show telemetry ietf subscription ID receiver</b>                                              | 名前付きレシーバの設定を修正します。                                                             |
| サブスクリプションレシーバの接続パラメータを解決できない。 | <b>show telemetry ietf subscription ID receiver</b><br><br>サブスクリプションレシーバの状態が、解決中の状態から変わらない。      | レシーバ、ネットワーク設定、またはインターフェイスの状態を確認します。                                            |
| サブスクリプションレシーバ接続がアップ状態にならない。   | <b>show telemetry ietf subscription ID receiver</b><br><br>サブスクリプションレシーバの状態が、解決中から接続中に何度も変化する。   | 解決された接続が有効であり、レシーバまたはコレクタが到達可能で、指定されたトランスポートを使用してインバウンド接続を受け入れることができることを確認します。 |
| サブスクリプションレシーバの接続が拒否される。       | <b>show telemetry ietf subscription ID receiver</b><br><br>サブスクリプションレシーバの状態が、切断を除くすべての状態で変化し続ける。 | コレクタのタイプが正しいこと、および設定されている認証と許可が有効であることを確認します。                                  |

| 問題                                | チェック方法/症状                                                                                          | 対処方法                                                                                                          |
|-----------------------------------|----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| サブスクリプションレシーバが接続されているが、更新が受信されない。 | <b>show telemetry internal subscription ID stats</b><br><br>メッセージドロップカウントが増加しているが、送信されたレコードが増加しない。 | コレクタが更新通知のフローに追従できていることを確認します。                                                                                |
|                                   | <b>show telemetry internal subscription</b><br><br>カウントが変化しない。                                     | 変更時のサブスクリプションの場合は、実際にイベントが発生していないことを確認します。<br><br>定期的なサブスクリプションの場合は、更新期間が短く、時間は 100 分の 1 秒単位で指定されていることを確認します。 |

このセクションでは、問題のトラブルシューティングを行うための内部情報を提供するコマンドの詳細について説明します。

**show telemetry connection** : このコマンドは、オプションの接続インデックス値を取ります。インデックスが指定されていない場合は、使用されているすべての接続の基本的な接続パラメータ情報が表示されます。コマンドで接続インデックスを指定すると、接続に関する詳細が表示されます。コマンド出力はトランスポート固有であり、すべてのトランスポートで使用できるとは限りません。このコマンドの出力は変更される場合があります。

**show telemetry internal diagnostics** : このコマンドは、すべてのテレメトリログと動作状態のダンプを試みます。問題を報告するときは、可能な限り問題の発生時のすぐ後にこのコマンドを使用し、**show running-config | section telemetry** コマンドの出力も提供すると、解決に役立ちます。

## サブスクリプションレシーバー

サブスクリプションレシーバーは、実際のサブスクリプションレシーバーまたはコレクタに接続するサブスクリプション関連のオブジェクトです。コレクタに到達するために必要なメカニズムはレシーバタイプに固有ですが、接続は、サブスクリプションがレシーバまたはコレクタに到達できるようにするために使用されるエンティティです。

サブスクリプションレシーバの状態は、レシーバへの接続を要求および使用する機能に基づいており、複数の状態があります。それぞれの状態は、サブスクリプションがレシーバまたはコレクタに更新を送信できるようにするために必要な他のリソースの制御に関連付けられています。

サブスクリプションレシーバーは、テレメトリまたはテレメトリプロトコルを使用して接続を使用します

### サブスクリプションレシーバーの状態

サブスクリプションレシーバの動作状態は、設定された名前（接続のインデックス）、レシーバの状態、状態に関する説明またはメモ、および最後の状態変更の時刻で構成されます。説明文字列は常に使用されるわけではありません。

サブスクリプションレシーバーの考えられる状態を次の表に示します。

表 5: サブスクリプションレシーバーの状態

| サブスクリプションレシーバーの状態   |                                | 説明                                                             |
|---------------------|--------------------------------|----------------------------------------------------------------|
| CLI 値               | YANG 値                         |                                                                |
| Disconnected        | rcvr-state-disconnected        | レシーバは切断され、再接続は試行されません。                                         |
| Resolving           | rcvr-state-resolving           | レシーバに到達するために必要な接続パラメータを解決しています。                                |
| Transport requested | rcvr-state-transport-requested | レシーバに到達するための接続要求が、 <b>Resolving</b> 状態から決定された接続パラメータを使用していました。 |
| Connecting          | rcvr-state-connecting          | サブスクリプションをレシーバに接続するために必要なリソースが割り当てられています。                      |
| Connected           | rcvr-state-connecting          | サブスクリプションがレシーバに接続され、更新をレシーバに送信できるようになりました。                     |
| Disconnecting       | rcvr-state-disconnected        | 接続で使用されているリソースが再割り当てされています。                                    |

YANG 値 `rcvr-state-invalid` は、レガシーレシーバでのみ使用されます。無効なサブスクリプションレシーバは接続できないため、サブスクリプションレシーバの状態が無効な場合は `disconnected` に設定されます。説明文字列で、`invalid` のサブスクリプションレシーバと `disconnected` のサブスクリプションレシーバを区別できます。

サブスクリプションレシーバーは、次の理由で切断されることがあります。

- サブスクリプションの別のレシーバが切断されていない。
- 接続のセットアップが永続的に失敗した。
- 名前付きレシーバが存在しない。
- 名前付きレシーバがサブスクリプションで指定されたタイプではない。
- 名前付きレシーバが無効である。
- サブスクリプションが無効である。
- 要求された接続が別のレシーバによって使用されている。

## テレメトリ接続

テレメトリ接続は、サブスクリプションがレシーバに到達するために使用するトランスポートインスタンスを表し、単に動作状態を表します。テレメトリ接続は、整数のインデックス値で識別されます。接続に関するその他の情報は、サブスクリプションが使用するよう設定されているレシーバのタイプに基づく接続のタイプに固有です。

セキュアなダイヤルアウト トランスポートでは、設定された名前付きレシーバのホスト部分は、接続のセットアップ時にレシーバが提供する証明書の識別名（DN）と一致する必要があります。このため、同じ接続を使用する複数のレシーバを持つことはできません。

この項で説明するすべての状態は、すべてのタイプの接続で該当しますが、すべてが使用されるわけではありません。テレメトリ接続で可能な状態を次の表に示します。

### テレメトリの接続状態

表 6: 接続状態と説明

| 接続状態          |                          | 説明                                  |
|---------------|--------------------------|-------------------------------------|
| CLI 値         | YANG 値                   |                                     |
| Pending       | con-state-pending        | 接続が作成されましたが、まだ開始されていません。            |
| Connecting    | con-state-connecting     | 接続をセットアップする要求が進行中です。                |
| Active        | con-state-active         | 接続が確立されており、サブスクリプションレシーバで使用できます。    |
| Disconnecting | con-state -disconnecting | 接続が切断され、サブスクリプションレシーバによる解放を待機しています。 |

接続に関連付けられているその他の動作状態には、リモートレシーバ（使用可能な場合はピア）の ID、および最後の状態変更の時刻などがあります。

## テレメトリプロトコル接続

テレメトリプロトコル接続は、テレメトリデータを転送するために、テレメトリデータソースとテレメトリコレクタまたは管理システム間で確立される通信リンクおよびメカニズムです。これらの接続では、特定のプロトコルを使用して、モニタリングおよび分析用のネットワークパフォーマンス、状態、およびイベントデータを提供します。

ここでは、プロトコルタイプ接続と、これらが名前付きプロトコルレシーバに割り当てられたサブスクリプションレシーバによりどのように使用されるかについて説明します。

次の表に、プロトコル接続パラメータを示します。

表 7: プロトコルタイプ接続のパラメータ

| パラメータ                   | 発信元                  | 注                                                              |
|-------------------------|----------------------|----------------------------------------------------------------|
| Destination IP address  | 名前付きレシーバホスト          | ホストはドメイン名を使用するため、ドメイン名の解決が必要になる場合があります。                        |
| Destination port number | 名前付きレシーバポート          | 明示的に設定する必要があります。                                               |
| Source VRF              | サブスクリプション（指定されている場合） | VRF が指定されていない場合、デフォルトの VRF が使用されます。それ以外の場合、VRF 名は内部識別子に解決されます。 |
| Source IP address       | サブスクリプション（指定されている場合） | 指定しない場合、送信元 IP アドレスは VRF および宛先 IP アドレスに基づいて決定されます。             |

これらのパラメータの一部は、サブスクリプションレシーバの親サブスクリプションの設定に基づいています。

設定から接続パラメータを解決するときに、順序が指定されていない場合は、最初に VRF が決定され、次に宛先 IP アドレス、最後に送信元 IP アドレスが決定されます。解決の特定のステップが非永続的に失敗する場合、5 秒間隔で無限に再試行されます。

接続は、要求されるとすぐにインスタンス化されます。つまり、最初のサブスクリプションレシーバーが **resolving** 状態から **transport requested** 状態に移行するとすぐに、サブスクリプションレシーバーによって以前に解決されたパラメータを持つ接続インスタンスが作成されます。

要求された接続が正常に設定され、テレメトリで使用されると、接続状態は **connected** に変わります。これは、Cisco IOS XE デバイスとレシーバデバイスの間に接続が存在することを意味します。レシーバーが使用するリソースを再割り当てするために、リソースを使用するサブスクリプションレシーバーに、接続が設定されたことが通知されます。これらのサブスクリプションレシーバの状態は、**connecting** に移行して、サブスクリプションをレシーバに接続するために必要なリソースを設定します。これらのリソースが確保されると、サブスクリプションレシーバの状態が **connected** に変更され、レシーバが更新通知を受信します。

テレメトリ接続がアクティブになることができない理由の一部を次に示します。

- 認証が失敗した
- 接続先に到達できない
- リモートホストポートのリスナーのタイプが正しくない
- リモートホストポートにリスナーが存在しない



(注)

接続セットアップが進行中のときは、接続セットアップを開始するために必要なパラメータが正常に解決されているため、この接続を使用するサブスクリプションレシーバーはすべて **connecting** 状態になります。

接続セットアップが失敗したときに実行されるアクションは、プロトコルによって異なります。次の表に、単一のセットアップ要求内の接続の再試行動作と、接続セットアップ要求が失敗した場合の再解決要求の再試行動作を示します。この動作は、レガシーレシーバによって要求された接続でも同じです。

## プロトコルの再試行間隔

表 8: プロトコル固有の再試行間隔

| プロトコル                                                                                                                                            | 接続の再試行回数                   | 再解決要求                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|----------------------------------------------|
| <ul style="list-style-type: none"> <li>• grpc-tcp</li> <li>• grpc-tls</li> </ul>                                                                 | 各試行間を 1、3、4、7 秒の間隔で 5 回再試行 | 制限なし。接続の再試行が失敗すると、継続的に再解決を要求します試行ごとに 14 秒です。 |
| <ul style="list-style-type: none"> <li>• cloud-native</li> <li>• cntp-tcp</li> <li>• cntp-tls</li> <li>• native</li> <li>• tls-native</li> </ul> |                            | 5、10、15、20、25、および 30 秒。                      |

## テレメトリにおけるハイアベイラビリティ

テレメトリの動的な接続は、アクティブなスイッチかスイッチスタック内のメンバーへの SSH、またはハイアベイラビリティ対応デバイスでのアクティブなルートプロセッサへの SSH を介して NETCONF セッションで確立されます。また、スイッチオーバー後にすべてのダイナミック サブスクリプションを再作成する必要があります。gNMI ダイアログイン サブスクリプションも、SSH を介した NETCONF セッションと同様に機能します。

gRPC ダイアログアウト サブスクリプションは、アクティブなスイッチまたはスタック メンバの実行コンフィギュレーションの一部としてデバイスに設定されます。スイッチオーバーが発生すると、テレメトリ受信者への既存の接続が切断され、再接続されます（受信者へのルートが残っている限り）。サブスクリプションを再設定する必要はありません。

デバイスのリロード時には、サブスクリプションの設定をデバイスのスタートアップ コンフィギュレーションに同期させる必要があります。これにより、デバイスの再起動後もサブスクリプション設定がデバイス上にそのまま残ります。必要なプロセスが起動して実行されると、デバイスはテレメトリ受信者への接続を試行し、通常の動作を再開します。

## gRPC サブスクリプションの FQDN サポート

gRPC テレメトリ サブスクリプションは設定ベースです。つまり、ユーザーはデバイス設定の一部として受信ホストとその他のサブスクリプション パラメータを指定する必要があります。このレシーバ設定は、テレメトリの更新を送信するための接続の詳細を決定するために使用されます。FQDN サポートにより、IP アドレスに加え、完全修飾ドメイン名（FQDN）も gRPC サブスクリプションに使用できます。FQDN サポートはデフォルトで有効になっています。

前述したように、FQDNを使用するにはサブスクリプションで名前付きレシーバを使用する必要があります。この例は、gRPC サブスクリプションに使用される FQDN を示しています。

```
telemetry receiver protocol my-receiver
  host name receiver.cisco.com 12345
  protocol grpc-tcp

telemetry ietf subscription 101

  encoding encode-kvgpb
  filter xpath /mdt-oper-v2-data/mdt-subscriptions

  stream yang-push
  update-policy periodic 1000

  receiver name my-receiver

telemetry ietf subscription 102

  encoding encode-kvgpb
  filter xpath /mdt-oper-v2-data/mdt-connections

  stream yang-push
  update-policy periodic 1000

  receiver name my-receiver
```

## 翻訳について

このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。