IP サービス レベル契約の実装

IP サービス レベル契約（IP SLA）は、Cisco IOS XR ソフトウェアを実行するほとんどのデバイスに組み込まれているテクノロジーのポートフォリオであり、ユーザはネットワーク アセスメントの実行、Quality of Service（QoS）の検証、新しいサービスの容易な展開、ネットワークのトラブルシューティングに関する管理者の支援を行うことができます。

この章では、この機能の詳細と、この機能の設定に必要なさまざま手順について説明します。

この章では、次のトピックについて取り上げます。

IP サービス レベル契約テクノロジーの概要

IP SLA は、連続的で、信頼でき、予測可能な方法でトラフィックを生成する、アクティブトラフィック モニタリングを使用してネットワークのパフォーマンスを測定します。IP SLA はネットワークにデータを送信し、複数のネットワーク間あるいは複数のネットワーク パス内のパフォーマンスを測定します。ネットワーク データおよびIPサービスをシミュレーションし、ネットワーク パフォーマンス情報をリアル タイムで収集します。次の情報が収集されます。

- 応答時間

- 単方向遅延、ジッター（パケット間の遅延のばらつき）
- パケット損失
- ネットワーク リソースのアベイラビリティ

IP SLA は、ルータ間またはルータとネットワーク アプリケーション サーバなどのリモート IP デバイスの間で、トラフィックを生成および分析してパフォーマンスを測定することにより、アクティブ モニタリングを行います。さまざまな IP SLA 動作によって得られる測定統計情報は、トラブルシューティング、問題分析、ネットワーク トポロジの設計に使用します。

この項では、次のトピックについて取り上げます。

サービス レベル契約

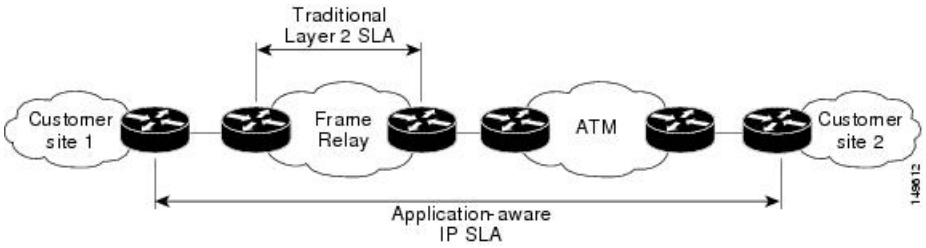
インターネット ショッピングはこの数年で急激に成長し、テクノロジーの進化により高速で信頼性の高いインターネット アクセスが提供されるようになりました。多くの機能がオンライン アクセスを必要とし、その業務のほとんどをオンラインで行っており、サービスが失われると企業の収益に影響を与えることがあります。インターネット サービス プロバイダー（ISP）だけでなく、社内の IT 部門も、定義されたサービス レベル（サービス レベル契約）を提供し、顧客にある程度の予測性を提供するようになっています。

ネットワーク管理者は、アプリケーション ソリューションをサポートするサービス レベル契約をサポートする必要があります。「図 1 : 従来のサービス レベル契約と IP SLA の範囲 （2 ページ）」に、アプリケーションのサポートも含め、エンドツーエンドのパフォーマンス測定をサポートするために、IP SLA がどのように従来のレイヤ 2 サービス レベル契約の概念を取り込み、より広い範囲に適用しているかを示します。

（注）

- アプリケーションおよび IP SLA の処理速度が対応している場合は、IP SLA フローエントリのフローレートを最大で 1500 に指定できます。
- IP SLA の高パフォーマンスの動作を実現するには、同じデバイスにおける複数の IP SLA 動作に同じ送信元ポートと宛先ポートを再利用しないようにします（特に大規模な場合）。

図 *1 :* 従来のサービス レベル契約と *IP SLA* の範囲



次の表に、IP SLA の従来のサービス レベル契約に対する改良点の一覧を示します。

表 *1*：従来のサービス レベル契約に対する *IP SLA* の改良点

改良の種類	説明
エンドツーエンド測定	ネットワークの端からもう一方の端までパフォーマンスを測定できることにより、エンドユーザによるネットワーク利用状況をより広い到達範囲でより正確に表現できます。
詳細化	遅延、ジッター、パケット シーケンス、レイヤ 3 接続、パスとダウンロード時間などの双方向のラウンドトリップの数値に詳細化される統計情報により、レイヤ2リンクの帯域幅だけよりも詳細なデータが得られます。
精度	ネットワーク パフォーマンスのわずかな変化にも影響を受けやすいアプリケーションは、ミリ秒以下の単位での IP SLA の測定精度を必要とします。
展開の簡易化	IP SLA は、大きいネットワーク内で既存のシスコ デバイスを活用することにより、従来のサービス レベル契約で必要になることが多い物理的な動作よりも、簡単に実装されます。
アプリケーション認識型の監視	IP SLA は、レイヤ 3 からレイヤ 7 の上で動作するアプリケーションによって生成されたパフォーマンス統計情報をシミュレートおよび測定できます。従来のサービス レベル契約では、レイヤ 2 パフォーマンスしか測定できません。
普及	IP SLA は、ローエンドからハイエンドまでのルータとスイッチに及ぶ、シスコ ネットワーキング デバイスでサポートされています。この幅広い展開により、IP SLA は、従来のサービス レベル契約よりも高い柔軟性を備えています。

IP サービス レベル契約の利点

次の表に、IP SLA を実装することの利点の一覧を示します。

表 *2*：*IP SLA* の利点の一覧

利点	説明
IP SLA のモニタリング	サービス レベル契約モニタリング、評価、および検証の提供
ネットワーク パフォーマンス モニタリング。	ネットワーク内のジッター、遅延、パケット損失が測定できる。また、IP SLA は、予防的な通知に加えて、連続的で、信頼性が高く、予測可能な測定を提供します。

利点	説明
IP サービス ネットワーク稼働状態評価	既存の QoS が新しい IP サービスに対して十分であることの検証
ネットワーク動作のトラブルシューティング	問題をただちに特定し、トラブルシューティング時間を節約する、一貫し、信頼性が高い測定を提供します。

IP サービス レベル契約を実装する前提条件

一般的なネットワーキング プロトコルおよび特定のネットワーク設計の知識が必要です。ネットワーク管理アプリケーションについての知識が役立ちます。すべての動作を同時にスケジューリングすると、パフォーマンスに悪影響を及ぼすおそれがあるため、お勧めしません。

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

IP サービス レベル契約の実装の制限

- Cisco IOS XR ソフトウェアでサポートされている IP SLA 動作の最大数は 2048 です。
- Cisco IOS XR ソフトウェアでサポートされている IP SLA 設定可能操作の最大数は 2000 です。
- 同じ開始時刻にすべての操作をスケジューリングすることは、パフォーマンスに影響する可能性があるため、推奨しません。開始時間が同じ場合は、1秒あたりの操作数を10以下にスケジューリングする必要があります。`start after` コンフィギュレーションを使用することをお勧めします。

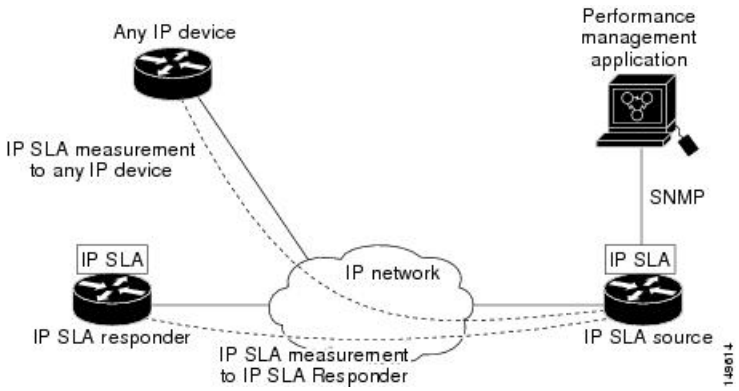
（注） 頻度を 60 秒未満に設定すると、送信されるパケット数が増加します。しかし、スケジュールされた動作の開始時刻が同じ場合、IP SLA 動作のパフォーマンスに悪影響を与える可能性があります。

- IP SLA は HA に対応していません。
- frequency、timeout、および threshold コマンドを設定する前に、次のガイドラインを検討してください。
- 制御無効モードでは、制御有効モードと比較すると、IP SLA の拡張性が向上します。

IPサービス レベル契約によるネットワーク パフォーマンスの測定

IP SLA は、ルータなどの 2 台のネットワーキング デバイス間のネットワーク パフォーマンスを測定するために、生成されたトラフィックを使用します。「図 2 : IP SLA の動作 （5 ページ）」に、IP SLA デバイスが宛先デバイスに生成パケットを送信するときに IP SLA が開始される手順を示します。宛先デバイスがパケットを受信した後、動作が IP SLA コンポーネントを受信側（たとえば IP SLA レスポンダ）で使用している場合、応答パケットにはターゲット デバイスでの遅延に関する情報が含まれています。送信元デバイスはこの情報を使用して測定の精度を向上させます。IP SLA 動作は、動作にユーザ データグラム プロトコル（UDP）などの特定のプロトコルを使用した、送信元デバイスからネットワークの宛先へのネットワーク測定です。

図 *2 : IP SLA* の動作



IP SLA ネットワーク パフォーマンス測定を実現するためには、次のタスクを実行します。

1. 必要に応じて IP SLA レスポンダをイネーブルにします。
2. 必要な IP SLA 動作タイプを設定します。
3. 指定された IP SLA 動作タイプのオプションを設定します。
4. 必要であれば、応答条件を設定します。
5. 動作の実行をスケジュールします。その後、一定の時間動作を実行して統計情報を収集します。
6. Cisco IOS-XR ソフトウェアの CLI または XML を使用して、または NMS システムで SNMP を使用して、動作の結果を表示および解釈します。

ここでは、次の内容について説明します。

IP SLA レスポンダおよび IP SLA 制御プロトコル

IP SLA レスポンダは宛先シスコ ルーティング デバイスに組み込まれたコンポーネントで、システムが IP SLA 要求パケットを予想して応答します。IP SLA レスポンダは、高い測定精度を提供します。特許取得済みの IP SLA 制御プロトコルは、IP SLA レスポンダによって使用され、応答側がどのポートで待ち受けと応答を行うか応答側に通知するメカニズムを提供します。Cisco IOS XR ソフトウェア デバイスまたはその他のシスコ プラットフォームのみが宛先 IP SLA レスポンダの送信元になることができます。

図 2 : IP SLA の動作 （5 ページ）に、IP SLA レスポンダが IP ネットワークのどこに適しているかを示します。IP SLA レスポンダは、IP SLA 動作から送信されたコントロール プロトコル メッセージを指定されたポートで受信します。レスポンダは、制御メッセージを受信すると、制御メッセージで指定された UDP ポートを指定された期間イネーブルにします。この間に、レスポンダは要求を受け付け、応答します。応答側は、IP SLA パケットに応答した後、あるいは指定された期間が経過すると、ポートをディセーブルにします。セキュリティを強化するために、コントロール メッセージの MD5 認証も使用できます。

（注） IP SLA レスポンダは、ソケットを開いてローカル パケット転送サービス（LPTS）をプログラムするために少なくとも 1 秒必要です。したがって、IP SLA タイムアウトを少なくとも 2000 ミリ秒に設定します。

UDP ジッター操作では IP SLA レスポンダを使用する必要があります。ターゲット ルータですでに提供されているサービスが選択された場合、IP SLA レスポンダをイネーブルにする必要はありません。シスコ以外のデバイスの場合は、IP SLA レスポンダを設定できず、IP SLA は、それらのデバイスにネイティブなサービスのみに動作パケットを送信できます。

IP SLA の応答時間の計算

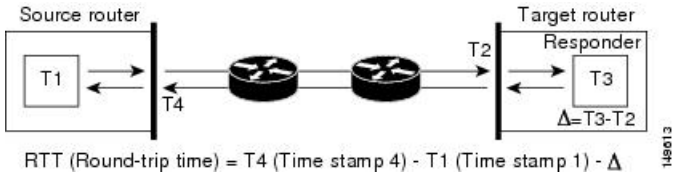
他の優先順位が高いプロセスにより、ルータは着信パケットを処理するために数十ミリ秒を要します。テスト パケットへの応答が処理を待つ間キューに格納される可能性があるため、この遅延は応答時間に影響を与えます。この場合、応答時間は正しいネットワーク遅延を反映しません。IP SLA は、送信元ルータとターゲット ルータ上でこれらの処理遅延を最小化し、真のラウンドトリップ遅延を判定します（IP SLA レスポンダが使用されている場合）。一部の IP SLA プローブ パケットには、測定をより正確にするために、最終的な計算で使用される遅延情報が含まれています。

IP SLA レスポンダをイネーブルにすると、ターゲット デバイスは、パケットがインターフェイスに到着した時点と出て行く時点の両方でタイムスタンプを取得し、統計情報を計算するときにそれを考慮します。このタイムスタンプ処理は、ミリ秒以下の単位で行われます。

図 3 : IP SLA レスポンダのタイムスタンプ処理 （7 ページ） に、レスポンダの動作を示します。T3 は、応答パケットが IP SLA レスポンダ ノードで送信された時刻であり、T1 は送信元ノードで要求が送信された時刻です。RTT を算出するためのタイム スタンプが 4 つ付けられます。ターゲット ルータでレスポンダ機能がイネーブルの場合、タイム スタンプ 3（TS3）からタイム スタンプ 2（TS2）を引いてテスト パケットの処理にかかった時間を求め、デルタ

（Δ）で表します。次に全体のRTTからこのデルタの値を引きます。精度を高めるため、優先度が高いパスで着信タイムスタンプ 4（TS4）が取得される送信元ルータ上で、同じ原理が IP SLA によって適用されることに注意してください。

図 *3 : IP SLA* レスポンダのタイムスタンプ処理



IP SLA 動作のスケジューリング

IP SLA動作の設定が完了したら、その動作をスケジューリングして、統計情報の取得とエラー情報の収集を開始する必要があります。動作をスケジューリングすると、動作はただちに開始されるか、特定の月の特定の日に開始されます。また、動作を保留状態にすることができます。これは、その動作が、トリガーされるのを待つ反応（しきい値）動作である場合に使用されます。IP SLA 動作の通常のスケジューリングでは、一度に 1 つの動作をスケジューリングできます。

IP サービス レベル契約の動作タイプ

IP SLA は、さまざまなタイプの動作を設定して、応答時間、ジッター、スループット、およびパケット損失を測定します。また、各動作は複数のアプリケーションにマッピングされます。

次の表に、さまざまな動作タイプの一覧を示します。

表 *3 : IP SLA* のための動作タイプ

動作	説明
UDP エコー	ラウンドトリップ遅延を測定し、UDP トラフィックの応答時間を正確に測定するために役立ちます。
UDP ジッター	ラウンドトリップ遅延、単方向遅延、単方向ジッター、双方向ジッター、単方向パケット損失を測定します。
ICMP エコー	パス全体のラウンドトリップ遅延を測定します。
ICMP パスエコー	ルータとネットワーク上の任意のIPデバイスの間のホップバイホップの応答時間を計算します。パスはtracerouteアルゴリズムを使用して検出され、送信元ルータとパス内の各中間ホップの間の応答時間が測定されます。送信元デバイスと宛先デバイスの間に複数の等コスト ルートがある場合、ICMP パスエコー動作は、設定可能なルーズ ソース ルーティング（LSR）オプションを使用していずれかのパスを選択します。

動作	説明
ICMP パスジッター	IP ネットワーク内のホップバイホップ ジッター、パケット損失、および遅延測定統計情報を測定します。
MPLS LSP ping	ラベル スイッチド パス（LSP）の接続をテストし、MPLS ネットワーク内の LSP のラウンドトリップ遅延を測定します。次の Forwarding Equivalence Class（FEC）がサポートされています。 • IPv4 ラベル配布プロトコル（LDP） • トラフィック エンジニアリング（TE）トンネル • 疑似回線 エコー要求は、FEC に属する他のパケットと同じデータ パスに沿って送信されます。エコー要求パケットがパスの終端に達すると、出力のラベル スイッチング ルータ（LSR）のコントロール プレーンに送信されます。LSR は、それが本当に FEC の出力であることを確認し、MPLS パスを確認する FEC に関する情報が格納されているエコー応答パケットを送信します。デフォルトの VRF テーブルのみがサポートされています。
MPLS LSP トレース	LSP パスのホップバイホップのルートをトレースし、MPLS ネットワーク内の IPv4 LDP プレフィックスと TE トンネル FEC のホップバイホップのラウンドトリップ遅延を測定します。 エコー要求パケットは、各中継 LSR のコントロール プレーンにデータを送信し、そこでこのパスの中継 LSR であるかどうかが確認されます。また、各中継 LSR は、テスト対象の LSP にバインドされているラベルに関する情報を返します。デフォルトの VRF テーブルのみがサポートされています。

IP SLA の VRF サポート

サービス プロバイダーは、コア ネットワークとカスタマー ネットワークの両方の観点からネットワークのパフォーマンスを監視および測定する必要があります。そのためには、デフォルトの VPN ルーティングおよび転送（VRF）テーブルに加えて、IP SLA 動作でデフォルト以外の VRF も使用する必要があります。「表 3 : IP SLA のための動作タイプ （7 ページ）」では、動作がデフォルト以外の VRF テーブルの使用をサポートしているかどうかなど、さまざまな IP SLA 動作について説明しています。

IP SLA：予防的しきい値モニタリング

ここでは、しきい値と反応トリガーを使用した IP SLA のプロアクティブな監視機能について説明します。IP SLA では、IP アプリケーションの監視、分析、IP サービス レベルの確認を

行って、生産性の向上、運用コストの削減、ネットワークの輻輳や停止の発生の低減を行うことができます。IP SLA は、アクティブ トラフィック監視を使用してネットワークのパフォーマンスを測定します。

IP SLA を使用したプロアクティブなしきい値監視を設定する必要があるタスクを実行するには、次の概念について理解する必要があります。

IP SLA 反応コンフィギュレーション

IP SLA は、特定の測定されたネットワーク条件に反応するように設定できます。たとえば、IP SLA が接続上で測定したジッターが大きすぎる場合、IP SLA はネットワーク管理アプリケーションに通知を生成したり、より多くのデータを収集するために別の IP SLA 動作をトリガーしたりできます。

IP SLA 反応を設定するには、**ipsla reaction operation** コマンドを使用します。

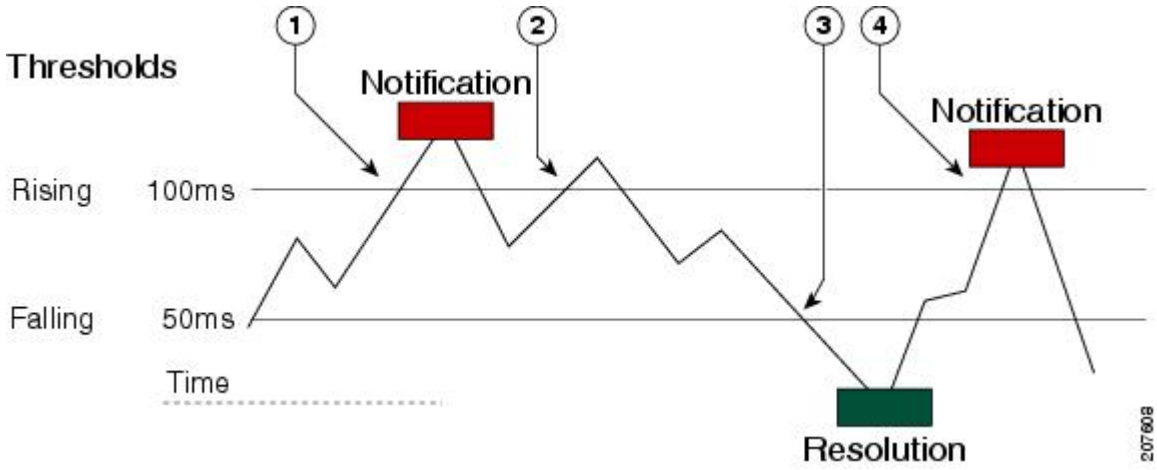
IP SLA しきい値モニタリングおよび通知

IP SLA では、ジッター平均、双方向ラウンドトリップ時間、接続性などのパフォーマンス パラメータのしきい値モニタリングがサポートされています。パケット損失とジッターでは、いずれかの方向（たとえば、送信元から宛先と、宛先から送信元）での違反またはラウンドトリップ値について、通知を生成できます。

通知は、しきい値違反が発生するたびに発行されるわけではありません。最初に上昇しきい値を超えたときに、イベントが送信され、通知が発行されます。後続のしきい値超過通知は、モニタリング対象の値が上昇しきい値を再び超える前に下限しきい値を下回った場合に限り発行されます。

次の図に、モニタリング対象要素が上限しきい値を超えたときに発生するトリガー反応の流れを示します。

図 *4 : IP SLA* のトリガーされた反応条件およびしきい値超過通知



1	最初に上昇しきい値を超えたときに、イベントが送信され、しきい値超過通知が発行されます。
2	上昇しきい値の超過違反が連続して発生しても、追加の通知は発行されません。
3	モニタリング対象の値が下限しきい値を下回っています。
4	上昇しきい値を超えたときに別のしきい値超過通知が発行されているのは、モニタリング対象の値が最初に下限しきい値を下回った後だけです。

同様に、モニタリング対象の要素が下限しきい値を最初に下回った時点で、下限しきい値超過通知が発行されます。下限しきい値超過違反に対する後続の通知が発行されるのは、上昇しきい値を超えた後で、モニタリング対象の値が下限しきい値を再び下回った場合に限られます。

Two-Way Active Measurement Protocol（TWAMP）

Two-Way Active Measurement Protocol（TWAMP）は、任意の 2 つのデバイス間のラウンドトリップ IP パフォーマンスを測定し、それにより IP SLA コンプライアンスをチェックする柔軟な方法を定義します。

TWAMP の利点

- TWAMP によって、完全な IP パフォーマンス測定が可能になります。
- TWAMP はネットワークに展開されたすべてのデバイスをサポートしているため、ソリューションを柔軟に選択できます。

（注） TWAMP v4 および v6 がサポートされています。

ここでは、次の内容について説明します。

TWAMP エンティティ

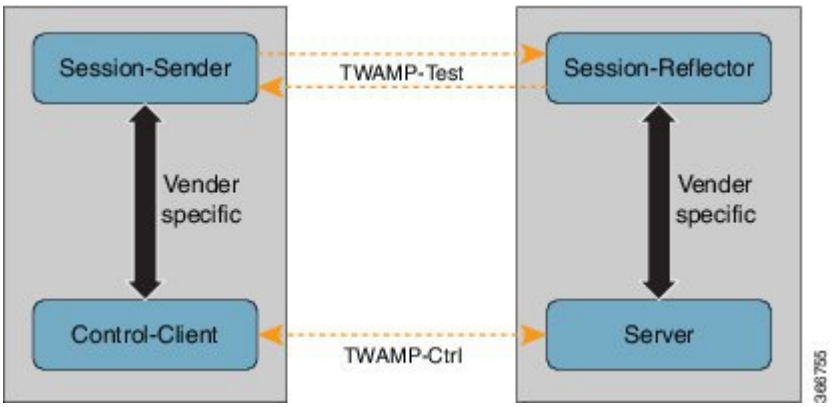
TWAMP システムは、4 つの論理エンティティで構成されています。

- サーバ：1 つ以上の TWAMP セッションを管理し、エンドポイント内のセッションごとのポートも構成します。

- セッション リフレクタ：TWAMP テスト パケットを受信するとすぐに測定パケットを反映します。
- 制御クライアント：TWAMP テスト セッションの開始と停止を開始します。
- セッション送信元：セッション リフレクタに送信された TWAMP テスト パケットをインスタンス化します。

次の図は、TWAMP が 2 つの個別のホスト上で動作する TWAMP の実装を示しています。一方は制御クライアントとセッション送信元の役割を果たし、もう一方はサーバとセッション リフレクタの役割を果たします。ルータでは、セッション サーバおよびセッション リフレクタの機能のみサポートされています。TWAMP を使用すると、基礎となるトランスポートの IP パフォーマンスを、TWAMP サポートを含むネットワーク要素間の協力によって測定できます。

図 *5 : TWAMP* エンティティ



TWAMP プロトコル

TWAMP プロトコルには、次の 3 つの異なるメッセージ交換カテゴリが含まれています。

- **接続セットアップ交換**：メッセージは、制御クライアントとサーバ間にセッション接続を確立します。最初に、通信ピアの ID が、チャレンジ応答メカニズムを介して確立されます。サーバがランダムに生成されたチャレンジを送信し、次に制御クライアントが共有秘密から派生したキーを使用してチャレンジを暗号化して応答を送信します。ID が確立された後、次のステップでは、後続の TWAMP 制御コマンドおよび TWAMP テスト ストリーム パケットをバインドするセキュリティ モードがネゴシエートされます。

  （注） サーバは、複数の制御クライアントからの接続要求を受け入れることができます。

- **TWAMP 制御交換**：TWAMP 制御プロトコルは TCP を介して実行され、測定セッションのインスタンス化および制御に使用されます。接続セットアップ交換とは異なり、TWAMP 制御コマンドは複数回送信できます。ただし、session-start コマンドの前に複数の

request-session コマンドを送信することはできますが、メッセージの順序が乱れることはありません。コマンドのシーケンスは、次のとおりです。

- request-session
- start-session
- stop-session

- **TWAMP テスト ストリーム交換**：TWAMP テストは UDP を介して実行され、セッション送信者とセッション リフレクタの間で TWAMP テスト パケットを交換します。これらのパケットには、パケットの出力と入力の瞬間を含むタイムスタンプ フィールドが含まれています。さらに、各パケットには、送信者（セッション送信元またはセッション リフレクタ）と外部の時刻源（GPS や NTP など）との同期の誤差を示すエラー推定値が含まれます。パケットにはシーケンス番号も含まれます。

TWAMP 制御および TWAMP テスト ストリームには、未認証、認証済み、および暗号化済みの 3 つのセキュリティ モードがあります。

ルータでの TWAMP の制限事項

- このルータでは、セッション サーバおよびセッション リフレクタの機能のみサポートされています。
- より精度の高いハードウェア タイムスタンプ機能はサポートされていません。
- ポート番号が 57343 より大きい値の場合、Twamp サーバ制御セッションはルータ上で作成されません。

ルータでの TWAMP の設定

セッション サーバの設定

```
Router# configure
Router(config)# ipsla server twamp
Router(config-ipsla-server-twamp)# port 862
Router(config-ipsla-server-twamp)# commit
```

セッション リフレクタの設定

```
Router# configure
Router(config)# ipsla responder twamp
Router(config-twamp-ref)# commit
```

実行コンフィギュレーション

```
ipsla
 responder
  twamp
  !
 !
```

```
 server twamp
  port 862
 !
!
```

TWAMP の確認

TWAMP 機能のステータスは、show ipsla twamp status コマンドを使用して確認できます。

```
Router# show ipsla twamp status
Thu Aug 17 12:42:38.923 IST
TWAMP Server is enabled
TWAMP Server port : 862
TWAMP Reflector is enabled
```

TWAMP セッションは、show ipsla twamp session コマンドを使用して検証できます。

```
Router# show ipsla twamp session
IP SLAs Responder TWAMP is: Enabled
Recvr Addr: 10.5.139.11
Recvr Port: 7222
Sender Addr: 172.27.111.233
Sender Port: 33243
Session Id: 10.5.139.11:70929508:88F7A620
Connection Id: 0
```

送信元 IP アドレスに基づく TWAMP テスト セッションは、**show ipsla twamp session source-ip <source ip-address> source-port <source port-number>** コマンドを使用して検証できます。

```
RP/0/0/CPU0:ios# show ipsla twamp session source-ip 172.27.111.233 source-port 33286
IP SLAs Responder TWAMP is: Enabled
Recvr Addr: 10.5.139.11
Recvr Port: 6198
Sender Addr: 172.27.111.233
Sender Port: 33286
Session Id: 10.5.139.11:71804476:F2721505
Connection Id: D
Mode: UnAuthorized
DSCP: 0
Pad Length: 0
Number of Packets Received: 8867
```

TWAMP Light

TWAMP Light は、制御セッションが不要な TWAMP の軽量モデルです。TWAMP 機能とは異なり、両方のエンドデバイスで TWAMP Light テストセッションのパラメータを設定する必要があります。これにより、制御セッションの確立と終了のオーバーヘッドが解消されます。さらに、リフレクタデバイスでサーバエンティティが不要であるため、サーバのメンテナンスのオーバーヘッドが軽減されます。

（注） TWAMP Light v4 および v6 がサポートされています。

Cisco NCS 540 シリーズ ルータ上の TWAMP Light の制限事項

- TWAMP Light テストセッションが Virtual Routing and Forwarding（VRF）インスタンスで実行されている場合、そのセッションは、インターフェイスで同じ VRF が設定されている場合にのみ機能します。
- デバイスで TWAMP Light テストセッションを設定すると、永続的なポートが開きます。このポートは、TWAMP Light の設定を削除するまで開いたままになります。この動作を希望しない場合は、タイムアウト時間後にセッションが非アクティブになるようにTWAMP Light テストセッションのタイムアウトを設定する必要があります。
- 同じローカル IP アドレスとローカルポートを持つ 2 つの異なるテストセッションを持つ 2 つのクライアントが同じ VRF に存在する場合、レスポンダには基盤となるソケットが 1 つだけ存在します。このようなシナリオでは、UDP の制限により、これら 2 つのクライアントの最大パケット数をサポートすることはできません。これにより、パフォーマンスが影響を受けます。したがって、同じ VRF で 2 つのテストセッションが同じローカル IP アドレスとローカルポートを持つことはできません。

NCS 540 シリーズ ルータでの TWAMP Light の設定

```
Router# configure
Router(config)# ipsla
Router(config-ipsla)# responder
Router(config-ipsla-resp)# twamp-light test-session 1
Router(config-ipsla-resp)# local-ip 192.0.2.10 local-port 13001 remote-ip 192.0.2.186
remote-port 13002 vrf default
Router(config-ipsla-resp)# timeout 60
Router(config-ipsla-resp)# commit
```

実行コンフィギュレーション

```
ipsla
 responder
  twamp-light test-session 1
  local-ip 192.0.2.10 local-port 13001 remote-ip 192.0.2.186 remote-port 13002 vrf default

  timeout 60
  !
 !
!
```

NCS 540 シリーズ ルータでの TWAMP Light の検証

TWAMP Light セッションは、**show ipsla twamp session** コマンドを使用して検証できます。コマンドの出力では、次に示すように、[Session status] フィールドを使用してセッションの状態が表示されます。

```
Router# show ipsla twamp session
***** TWAMP Sessions *****
No records matching query found
***** TWAMP-LIGHT Sessions *****
Session status: Active
```

```
Recvr Addr: 192.0.2.10
Recvr Port: 13001
Sender Addr: 192.0.2.186
Sender Port: 13002
Sender VRF Name: default
Session ID: 1
Mode: Unauthenticated
Number of Packets Received: 0
Session timeout: 60
```

MPLS LSP モニタリング

IP サービス レベル契約（SLA）ラベル スイッチド パス（LSP）のモニタ機能は、レイヤ 3 マルチプロトコル ラベル スイッチング（MPLS）バーチャル プライベート ネットワーク（VPN）を予防的に監視するための機能を備えています。この機能は、ネットワークのアベイラビリティを確認したり、MPLS VPN 内のプロバイダー エッジ（PE）ルータ間のネットワーク接続をテストするために便利です。MPLS LSP モニタを設定すると、ネットワーク トポロジに基づいて、自動的に IP SLA LSP ping または LSP traceroute 処理を生成または削除できます。

MPLS SLA モニタ機能では、IP SLA 動作の複数動作スケジューリングを実行することも可能であり、SNMP トラップ通知と Syslog メッセージを使用した予防的しきい値違反モニタリングもサポートされています。

MPLS LSP モニタ機能を使用するには、次の概念を理解しておく必要があります。

MPLS LSP モニタリングのしくみ

MPLS LSP モニタ機能では、レイヤ 3 MPLS VPN を予防的にモニタできます。MPLS LSP モニタの動作方法の一般的なプロセスは次のとおりです。

1. ユーザは、MPLS LSP モニタ インスタンスを設定します。

   MPLS LSP モニタ インスタンスを設定することは、標準的な IP SLA 動作の設定に似ています。たとえば、MPLS LSP モニタ インスタンスのすべての動作パラメータは、動作の ID 番号を指定した後で設定します。ただし、標準的な IP SLA 動作と異なり、これらの設定されたパラメータは、MPLS LSP モニタ インスタンスによって作成される個々の IP SLA LSP ping および LSP traceroute 動作の基本設定として使用されます。

   最初の MPLS LSP モニタ インスタンスが設定され、開始がスケジュールされると、BGP ネクストホップ ネイバー探索がイネーブルになります。「BGP ネクストホップ ネイバー探索 （16 ページ）」を参照してください。

2. ユーザが MPLS LSP モニタ インスタンスについて予防的しきい値違反モニタリングを設定します。

3. ユーザが、MPLS LSP モニタ インスタンスの複数動作スケジューリング パラメータを設定します。

4. 選択した設定オプションに応じて、MPLS LSP モニタ インスタンスは、該当する各 BGP ネクストホップ ネイバーに対する、個々の IP SLA LSP ping または LSP traceroute 動作を自動的に作成します。

   すべての MPLS LSP モニタ動作について、BGP ネクストホップ ネイバーあたり 1 つの IP SLA LSP ping 動作または LSP traceroute 動作のみが設定されます。しかし、特定の PE ルータで、同時に複数の MPLS LSP モニタ インスタンスを実行できます。（詳細については、この項の最後にある注記を参照してください）。

5. 各 IP SLA LSP ping または LSP traceroute 動作は、送信元 PE ルータと検出された宛先 PE ルータの間のネットワーク接続を測定します。

（注） 複数の MPLS LSP モニタ インスタンスを特定の PE ルータで同時に実行できます。たとえば、ある MPLS LSP モニタ インスタンスを、VPN1 という VRF に属する BGP ネクストホップ ネイバーを探索するように設定できます。同じ PE ルータで、別の MPLS LSP モニタ インスタンスを、VPN2 という VRF に属するネイバーを探索するように設定できます。この場合、BGP ネクストホップ ネイバーが VPN1 と VPN2 の両方に属していた場合、PE ルータはこのネイバーに対して 2 つの IP SLA 動作（1 つは VPN1 用、もう 1 つは VPN2 用）を作成します。

IP SLA 動作の MPLS LSP モニタ データベースへの追加と削除

MPLS LSP モニタ インスタンスは、特定の VPN に対して追加または削除された BGP ネクストホップ ネイバーについて定期的な通知を受けます。この情報は、MPLS LSP モニタが保持するキューに格納されます。キュー内の情報とユーザ指定の期間に基づき、新たに検出された PE ルータに対して新しい IP SLA 動作が自動的に作成され、有効でなくなった PE ルータに対する既存の IP SLA 動作は自動的に削除されます。

BGP ネクストホップ ネイバー探索

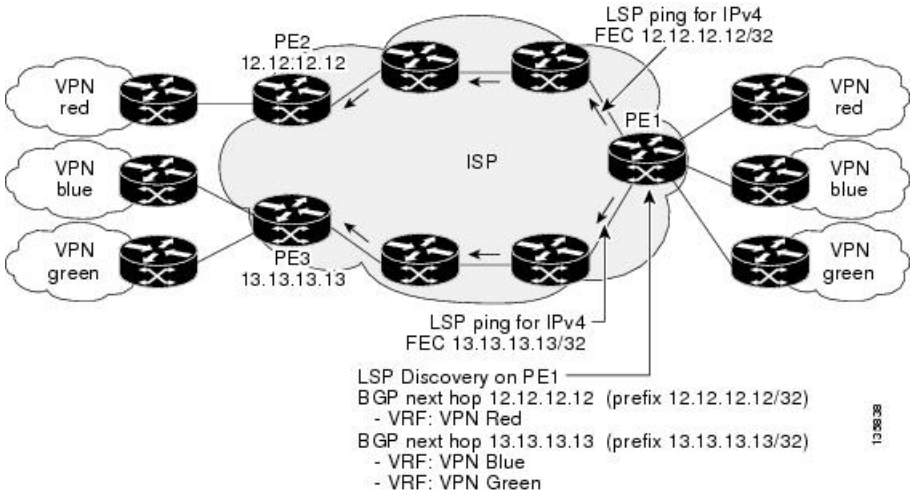
BGP ネクストホップ ネイバー探索は、送信元プロバイダー エッジ（PE）ルータに関連付けられているすべての VRF によって使用中の BGP ネクストホップ ネイバーを見つけるために使用されます。ほとんどの場合、これらのネイバーは PE ルータです。

BGP ネクストホップ ネイバー探索がイネーブルな場合、ローカル VRF とグローバル ルーティング テーブルの情報に基づいて、送信元 PE に関連付けられているすべての VRF によって使用中の BGP ネクストホップ ネイバーのデータベースが生成されます。ルーティング アップデートが受信されると、新しい BGP ネクスト ホップ ネイバーがただちにデータベースに追加されます。ただし、有効でなくなった BGP ネクストホップ ネイバーは、ユーザ定義に従って、定期的にデータベースから削除されます。

図 6 : 単純な VPN の BGP ネクストホップ ネイバー探索 （17 ページ） に、インターネット サービス プロバイダー（ISP）の単純な VPN シナリオでの BGP ネクストホップ ネイバー探索の動作を示します。この例で、ルータ PE1 に関連付けられた 3 つの VPN があります（赤、青、緑）。ルータ PE1 から見ると、これらの VPN には、BGP ネクスト ホップ ネイバー PE2（ルータ ID：12.12.12.12）および PE3（ルータ ID：13.13.13.13）を経由してリモートで到達可能で

す。BGP ネクストホップ ネイバー探索プロセスがルータ PE1 でイネーブルになっている場合、ローカル VRF とグローバル ルーティング テーブルに基づいてデータベースが生成されます。この例のデータベースには、2 つの BGP ネクスト ホップ ルータ エントリとして PE2 12.12.12.12 および PE3 13.13.13.13 が格納されます。ルーティング エントリは、どのネクスト ホップ ルータがどの特定の VRF 内に属しているか区別するために、ネクスト ホップ ルータ単位で維持されます。各ネクストホップ ルータ エントリに対し、グローバル ルーティング テーブル中の BGP ネクストホップ ルータの IPv4 Forward Equivalence Class（FEC）が、MPLS LSP ping 動作で使用するために提供されます。

図 *6 :* 単純な *VPN* の *BGP* ネクストホップ ネイバー探索



IP SLA LSP ping 動作と LSP traceroute 動作

この機能により、IP SLA LSP ping 動作と IP SLA LSP traceroute 動作に対するサポートが追加されます。これらの動作は、ネットワークの接続性の問題をトラブルシューティングし、MPLS VPN のネットワークのアベイラビリティを判定するために役立ちます。MPLS LSP モニタリングを使用する場合、送信元 PE ルータと検出された宛先 PE ルータの間のネットワーク接続を測定するために、IP SLA LSP ping 動作と LSP traceroute 動作が自動的に作成されます。個々の IP SLA LSP ping 動作と LSP traceroute 動作を手動で設定することもできます。これらの動作の手動の設定は、接続性の問題をトラブルシューティングするために役立ちます。

MPLS LSP モニタリングを使用して IP SLA LSP ping または LSP traceroute 動作を設定する方法の詳細については、「MPLS LSP モニタリング ping インスタンスの設定 （72 ページ）」および「MPLS LSP モニタリング トレース インスタンスの設定 （76 ページ）」を参照してください。

IP SLA LSP ping 動作と IP SLA LSP traceroute 動作は、それぞれ MPLS LSP ping 機能と MPLS LSP traceroute 機能で使用されるのと同じインフラストラクチャに基づいて、LSP をテストするためのエコー応答パケットとエコー要求パケットを送受信します。

MPLS LSP モニタリングの予防的しきい値モニタリング

MPLS LSP モニタの予防的しきい値モニタリング サポート機能では、ユーザ定義の応答条件（接続損失やタイムアウトなど）が満たされたときに、SNMP トラップ通知と Syslog メッセージをトリガーできます。MPLS LSP モニタ インスタンスのしきい値モニタリング動作の設定方法は、標準的な IP SLA 動作の設定方法と同様です。

LSP ヘルス モニタの複数動作スケジューリング

MPLS LSP モニタの複数動作スケジューリング サポート機能では、（各 MPLS LSP モニタ インスタンスに対して）自動的に作成された IP SLA 動作を、指定された期間（スケジュール期間）にわたって均等に分散される間隔で開始し、指定された頻度で再開するように簡単にスケジューリングできます。複数動作スケジューリングは、多数の PE ネイバーが存在し、その結果として多数の IP SLA 動作が同時に稼働している送信元 PE ルータ上で MPLS LSP モニタリングがイネーブルにされる場合に特に有用です。

（注） （新たに検出された BGP ネクスト ホップ ネイバーに対して）新たに作成された IP SLA 動作は、現在稼働している動作と同じスケジュール期間に追加されます。同時に開始する動作が多くなりすぎないように、複数動作スケジューリング機能は、それらの動作を、スケジュール期間にわたって均一に分散されるランダムな間隔で開始するようにスケジューリングします。

LSP パス ディスカバリ

LSP パス ディスカバリ（LPD）は、MPLS LSP モニタ（MPLSLM）の拡張で、MPLSLM インスタンスの一部である操作を許可し、パス ディスカバリ プロセスを開始してその結果が処理されます。この機能には、MPLS OAM インフラストラクチャにより LSPV サーバを通じて提供されるツリー トレース機能が必要です。

PEルータ間にコストが等しい複数のパスが存在する場合（これを等コストマルチパス（ECMP）と呼びます）、これら PE ルータの間にあるルータは、転送するトラフィックの特性（たとえばパケット中の宛先アドレス）に基づいて、ロードバランシングを行います。このようなネットワーク トポロジでは、PE ルータ間の使用可能なパスのうち 1 つ（またはいくつか）をモニタするだけでは、トラフィックが正しく転送される保証が得られません。

LPD は、**path discover** コマンドを使用して設定します。

（注） LPD機能を使用した場合、LSPV サーバが一度に多数のパス ディスカバリ要求を受信すると、CPU の負荷が高まる可能性があります。

IP サービスレベル契約の実装方法

UDP ジッター動作を使用した IP サービス レベルの設定

IP SLA UDP ジッター モニタリング動作は、VoIP、Video over IP、リアルタイム会議などのリアルタイム トラフィックに対するネットワークの適切さを診断するように設計されています。

ジッターはパケット間の遅延がばらつくことを指します。複数のパケットが送信元から宛先に連続的に送信された（たとえば 10 ms 間隔で送信された場合）、ネットワークが理想的に振る舞えば、宛先でも 10 ms 間隔でパケットを受信します。しかし、ネットワーク内に遅延（キューイング、代替ルートを介した受信など）が存在する場合、パケット間の到着遅延は、10 ms より大きい場合も、10 ms より小さい場合もあります。この例を使用すると、正のジッター値は、パケットが 10 ms を超える間隔で到着することを示します。パケットの到着が 12 ミリ秒の場合のジッター値は +2 ミリ秒（正の値）です。8 ミリ秒で到着する場合は、2 ミリ秒（負の値）です。Voice over IP（VoIP）など遅延に影響されやすいネットワークでは、正のジッター値は望ましくありません。0 のジッター値が理想的です。

しかし、IP SLA UDP ジッター動作の機能は、ジッターのモニタリングだけではありません。IP SLA が生成するパケットは、パケットの送信シーケンスと受信シーケンス情報を伝送し、送信元ターゲットと動作ターゲットとの間でタイム スタンプの送受信を行います。UDP ジッター動作は、次の機能を測定できます。

- 方向別ジッター（送信元から宛先へ、宛先から送信元へ）
- 方向別パケット損失
- 方向別遅延（一方向遅延）
- ラウンドトリップ遅延（平均 RTT）

データの送信と受信でパスが異なることがあるので（非対称）、方向別データを使用してネットワークの輻輳などの問題が発生している場所を簡単に特定できます。

UDP ジッター動作は、合成（シミュレーション）UDP トラフィックを生成して機能します。デフォルトでは、ペイロード サイズが 32 バイト（S）のパケット フレーム 10 個（N）を 20 ミリ秒（T）ごとに生成し、60 秒（F）ごとに動作を繰り返します。に示すように、これらのパラメータは、提供している IP サービスまたはこれから提供する IP サービスの最適なシミュレーションを行うようにそれぞれユーザ設定可能です。

この項では、次の手順について説明します。

宛先デバイスでの IP SLA レスポンダのイネーブル化

IP SLA レスポンダは、動作のターゲットであるターゲット デバイスでイネーブルにする必要があります。

ipsla responder コマンドを設定することにより、IP SLA レスポンダは、UDP ポート 1967 をオープンし、（プローブではなく）制御パケットを待ちます。ポートは、UDP ポート 1967 を

通じ、IP SLA 制御パケットを使用して動的にオープンまたはクローズできます。また、永続的なポートも設定できます。

永続的なポートは、設定が削除されるまでオープンされます。ポートは設定によってオープンされるため、エージェントは、制御要求パケットを使用せずに、IP SLA プローブ パケットを永続的なポートに直接送信できます。

永続的なポートを使用しない場合、**ipsla responder** コマンドのみを使用して設定する必要があります。

動的なポートを使用するには、次の例に示すように **ipsla responder** コマンドを使用します。

```
configure
ipsla responder
```

動的なポートは、エージェント側で動作を開始したときに、IP SLA 制御プロトコルを通じて、レスポンダ側でオープンされます。

例は、レスポンダ側の永続的なポートとして設定されています。UDP エコーと UDP ジッターは、動的なポートまたは永続的なポートを使用できます。UDP ジッターで永続的なポートを使用する場合、重複したパケットまたはアウトオブシーケンスパケットに対してチェックは実行されません。これは、プローブシーケンスの開始または終了を示す制御パケットがないためです。したがって、永続的なポートを使用する場合、シーケンス番号の検証はスキップされます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla responder**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla responder
RP/0/RP0/cpu 0: router(config-ipsla-resp)#
```

UDP エコーまたはジッター動作に対する IP SLA レスポンダをイネーブルにします。

ステップ 3 **type udp ipv4 address** *ip-address* **port** *port*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-resp)# type udp ipv4 address 12.25.26.10 port 10001
```

IP SLA レスポンダ上で永続的アドレスとポートをイネーブルにします。

ステップ 4 **commit**

次のタスク

IP SLA レスポンダをイネーブルにした後、「送信元デバイスでの UDP ジッター動作の設定およびスケジューリング （21 ページ）」を参照してください。

送信元デバイスでの UDP ジッター動作の設定およびスケジューリング

IP SLA 動作は、合成（シミュレーション）ネットワーク トラフィックを生成して機能します。1 つの IP SLA 動作（たとえば IP SLA 動作 10）は、動作の存続期間の間、指定された頻度で繰り返されます。

1 つの UDP ジッター動作は、指定された頻度 F で、送信元ルータからターゲットルータに、サイズ S の N 個の UDP パケットを T ミリ秒間隔で送信します。デフォルトでは、ペイロード サイズが 32 バイト（S）のパケット 10 個（N）を 20 ミリ秒（T）ごとに生成し、60 秒（F）ごとに動作を繰り返します。これらの各パラメータは、「表 4 : UDP ジッター動作パラメータ （21 ページ）」に示すように、ユーザが設定可能です。

表 *4 : UDP* ジッター動作パラメータ

UDP ジッター動作パラメータ	デフォルト	設定方法
パケット数（n）	10 パケット	• *operation-number* 引数を指定した **ipsla operation** コマンド • **type udp jitter**command • *count* 引数を指定した **packet count** コマンド
パケットあたりのペイロード サイズ（S）	32 バイト	• *operation-number* 引数を指定した **ipsla operation** コマンド • **type udp jitter**command • *size* 引数を指定した **datasize request** コマンド
パケット間隔（ミリ秒単位）（T）	20 ミリ秒	• *operation-number* 引数を指定した **ipsla operation** コマンド • **type udp jitter**command • *interval* 引数を指定した **packet interval** コマンド
動作を繰り返すまでの経過時間（秒単位）（F）	60 秒	• *operation-number* 引数を指定した **ipsla operation** コマンド • **type udp jitter**command • *seconds* 引数を指定した **frequency** コマンド

（注） IP SLA の設定中に **control disable** コマンドを使用して制御パケットを無効にした場合、送信元から送信されたパケットにはシーケンス番号がありません。ジッターを計算するには、シーケンス番号とタイムスタンプの値が必要です。したがって、**control disable** コマンドを使用する場合、ジッターは計算されません。

送信元デバイスで UDP ジッター動作を設定する前提条件

UDP ジッター動作の使用には、IP SLA レスポンダをターゲットのシスコ デバイスでイネーブルにする必要があります。IP SLA レスポンダをイネーブルにするには、「宛先デバイスでの IP SLA レスポンダのイネーブル化 （19 ページ）」のタスクを実行します。

送信元デバイスでの基本 UDP ジッター動作の設定およびスケジューリング

UDP ジッター動作を設定およびスケジューリングできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type udp jitter**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type udp jitter
```

動作を UDP ジッター動作として設定し、動作の特性を設定します。

ステップ 4 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# destination address 12.25.26.10
```

UDP ジッター動作の宛先の IP アドレスを指定します。

ステップ 5 **destination port** *port*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# destination port 11111
```

宛先ポート番号を指定します。範囲は 1 ～ 65535 です。

ステップ 6 **packet count** *count*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# packet count 30
```

（任意）プローブ中に送信されるパケット数を指定します。UDP ジッター動作の場合、範囲は 1 ～ 60000 です。ICMP パスジッター動作の場合、範囲は 1 ～ 100 です。

送信されるデフォルトのパケット数は 10 です。

ステップ 7 **packet interval** *interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# packet interval 30
```

（任意）パケット間隔を指定します。パケット間のデフォルト間隔は 20 ミリ秒です。

ステップ 8 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 9 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA コンフィギュレーション モードおよび動作モードを終了し、CLI をグローバル コンフィギュレーション モードに戻します。

ステップ 10 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 11 **life** { **forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 12 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 13 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 14 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルトは inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 15 **commit**

追加特性を指定した UDP ジッター動作の設定およびスケジューリング

UDP ジッター動作を設定およびスケジューリングできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type udp jitter**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type udp jitter
```

動作を UDP ジッター動作として設定し、動作の特性を設定します。

ステップ 4 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# vrf VPN-A
```

（任意）UDP ジッター動作の中で、デフォルト以外のルーティング テーブルを使用して VPN のモニタリングをイネーブルにします。最大 32 文字の英数字です。

ステップ 5 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 6 **destination port** *port*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# destination port 11111
```

宛先ポート番号を指定します。範囲は 1 ～ 65535 です。

ステップ 7 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 8 **statistics** [**hourly** | **interval** *seconds*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# statistics hourly
RP/0/RP0/cpu 0: router(config-ipsla-op-stats)#
```

（任意）UDP ジッター動作に対して統計情報収集パラメータを指定します。

ステップ 9 **buckets** *hours*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op-stats)# buckets 10
```

（任意）IP SLA 動作用に統計情報を保持する時間を設定します。このコマンドは、必ず **hourly** キーワードを指定した**statistics** コマンドとともに使用する必要があります。範囲は 0 ～ 25 時間です。デフォルト値は 2 時間です。

ステップ 10 **distribution count** *slot*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op-stats)# distribution count 15
```

（任意）IP SLA 動作のライフタイム中にホップごとに保持される統計情報の配布数を設定します。範囲は 1 ～ 20 です。デフォルト値は 1 配布です。

ステップ 11 **distribution interval** *interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op-stats)# distribution interval 20
```

（任意）統計情報の配布ごとのインターバルを設定します。指定できる範囲は 1 ～ 100 ms です。デフォルトの値は 20 ms です。

ステップ 12 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op-stats)# exit
```

IP SLA 統計情報コンフィギュレーション モードを終了します。

ステップ 13 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# datasize request 512
```

（任意）動作の要求パケットのペイロードのデータ サイズを設定します。UDP ジッターの場合は、範囲は 16 ～ 1500 バイトです。

ステップ 14 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# timeout 10000
```

指定された IP SLA 動作がその要求パケットからの応答を待機する時間を設定します。

- （任意）*milliseconds* 引数を使用して、動作が応答を待機するミリ秒数を指定します。

ステップ 15 **tos** *number*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# tos 255
```

タイプ オブ サービス番号を指定します。

ステップ 16 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-jitter)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA コンフィギュレーション モードおよび動作モードを終了し、CLI をグローバル コンフィギュレーション モードに戻します。

ステップ 17 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 18 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 19 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 20 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 21 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

（任意）動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルトは inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 22 **commit**

ステップ 23 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router # show ipsla statistics 432
```

現在の統計情報を表示します。

ステップ 24 **show ipsla statistics aggregated** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router # show ipsla statistics aggregated 432
```

ネットワークのパフォーマンスに関する 1 時間ごとの統計情報（集計データ）を返します。

UDP ジッター動作は、次の 1 時間ごとの統計情報を提供します。

- ジッター統計情報：テレフォニーおよびマルチメディア会議要件を解釈します。
- パケット損失およびパケット シーケンシング統計情報：テレフォニー、マルチメディア会議、ストリーミング メディア、およびその他の低遅延データ要件を解釈します。
- 単方向遅延統計情報：テレフォニー、マルチメディア会議、およびストリーミング メディア要件を解釈します。

UDP エコー動作のための IP SLA の設定

ネットワーク上の UDP パフォーマンスを測定するには、IP SLA UDP エコー動作を使用します。UDP エコー動作は、ラウンドトリップ遅延時間を測定し、シスコ デバイスとシスコ以外のデバイスの間の接続をテストします。UDP エコー動作の結果は、ビジネスクリティカル アプリケーションでの問題をトラブルシューティングするために役立ちます。

（注） UDP エコー動作では、IP SLA レスポンダが動作するシスコ デバイスか、UDP エコー サービスが動作する非シスコ デバイスが必要です。

基本的な UDP エコー動作を設定するのか、オプションのパラメータを使用した UDP エコー動作を設定するのかに応じて、次のいずれかのタスクを実行します。

送信元デバイスでの UDP エコー動作の設定のための前提条件

IP SLA Responder を使用する場合は、「宛先デバイスでの IP SLA レスポンダのイネーブル化（19 ページ）」のセクションを完了しておきます。

送信元デバイスでの UDP エコー動作の設定およびスケジューリング

オプション パラメータを指定せずに UDP エコー動作をイネーブルにできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type udp echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type udp echo
```

動作を UDP エコー動作として設定し、動作の特性を設定します。

ステップ 4 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。IP SLA レスポンダ側の永続的ポートを設定するか、UDP エコー サーバを使用できます。

ステップ 5 **destination port** *port*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# destination port 11111
```

宛先ポート番号を指定します。範囲は 1 ～ 65535 です。

ステップ 6 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 7 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 8 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 9 **life** [**forever** | *seconds*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 1
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 10 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 11 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 12 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

（任意）動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。これはデフォルト値です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 13 **commit**

ステップ 14 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

ステップ 15 **show ipsla statistics aggregated** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics aggregated 1
```

1 時間ごとの統計エラーと、すべての IP SLA 動作または指定した動作の 1 時間ごとの統計情報を表示します。

任意のパラメータを指定した、送信元デバイスでの UDP エコー動作の設定およびスケジューリング

送信元デバイスで UDP エコー動作をイネーブルにして、省略可能な IP SLA パラメータを設定できます。送信元デバイスは、測定統計情報が保存される場所です。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type udp echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type udp echo
```

動作を UDP エコー動作として設定し、動作の特性を設定します。

ステップ 4 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# vrf VPN-A
```

（任意）UDP エコー動作の中で、デフォルト以外のルーティング テーブルを使用して VPN のモニタリングをイネーブルにします。最大 32 文字の英数字です。

ステップ 5 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 6 **destination port** *port*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# destination port 11111
```

宛先ポート番号を指定します。範囲は 1 ～ 65535 です。

ステップ 7 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 8 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# datasize request 512
```

（任意）IP SLA 動作の要求パケットのペイロードにおけるプロトコル データ サイズを設定します。

- プロトコルデータサイズ（バイト単位）を指定するには、*size* 引数を使用します。範囲は 0 ～プロトコルの最大サイズです。デフォルト値は 1 バイトです。

ステップ 9 **tos** *number*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# tos 255
```

IP SLA 動作の IP ヘッダーに、タイプ オブ サービス（ToS）バイトを定義します。

（注） ToS バイトは DiffServ コード ポイント（DSCP）値に変換されますが、DSCP 値を直接入力することはできません。DSCP 値を使用するには、それに 4 を掛けて、結果を *number* 引数の値として入力します。

ステップ 10 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# timeout 10000
```

指定された IP SLA 動作がその要求パケットからの応答を待機する時間を設定します。

- *milliseconds* 引数を使用して、動作が応答を待機するミリ秒数を指定します。

ステップ 11 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# type udp echo tag ipsla
```

（任意）IP SLA 動作のユーザ指定 ID を作成します。

ステップ 12 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-udp-echo)# exit
```

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよびIPSLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 13 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本的なスケジュールを設定するか、グループ スケジューリングを使用して複数の動作をスケジューリングできます。

ステップ 14 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 15 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 16 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 17 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開

始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。

- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 18 **commit**

ステップ 19 **show ipsla statistics enhanced aggregated** [*operation-number*] **interval** *seconds*

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics enhanced aggregated 432
```

拡張された履歴統計情報を表示します。サンプル出力を表示するには、拡張された履歴統計情報を設定する必要があります。

ステップ 20 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

ICMP エコー動作の設定

デバイス上の IP 接続をモニタするには、IP SLA ICMP エコー動作を使用します。ICMP エコー動作は、シスコ ルータと IP を使用するデバイスの間のエンドツーエンド応答時間を測定します。ICMP エコーは、ネットワークの接続上の問題をトラブルシューティングするために使用します。

（注） ICMP エコー動作では、IP SLA レスポンダをイネーブルにする必要はありません。

基本的なICMPエコー動作を設定およびスケジューリングするのか、省略可能なパラメータを使用したICMPエコー動作を設定およびスケジューリングするのかに応じて、次のいずれかの手順を実行します。

送信元デバイスでの基本の ICMP エコー動作の設定およびスケジューリング

オプションパラメータを指定せずにICMPエコー動作をイネーブルにしてスケジューリングできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp echo
```

ICMP エコー動作タイプを定義します。

ステップ 4 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 5 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo) frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 6 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 7 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 8 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 9 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 10 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 11 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 12 **commit**

ステップ 13 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router # show ipsla statistics 432
```

現在の統計情報を表示します。

送信元デバイスでの省略可能なパラメータを使用した ICMP エコー動作の設定およびスケジューリング

送信元デバイスで ICMP エコー動作をイネーブルにして、省略可能な IP SLA パラメータを設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp echo
```

ICMP エコー動作タイプを定義します。

ステップ 4 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# vrf VPN-A
```

（任意）ICMP エコー動作の中で、デフォルト以外のルーティング テーブルを使用して VPN のモニタリングをイネーブルにします。最大 32 文字の英数字です。

ステップ 5 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 6 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 7 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# datasize request 512
```

（任意）指定した IP SLA 動作の要求パケットのペイロードにおけるプロトコル データ サイズを設定します。

- プロトコルデータサイズ（バイト単位）を指定するには、*bytes* 引数を使用します。範囲は 0 ～ 16384 です。ICMP エコー動作のデフォルトは 36 バイトです。

ステップ 8 **tos** *number*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# tos 1
```

IP SLA 動作の IP ヘッダーに、タイプ オブ サービス（ToS）バイトを定義します。

（注） ToS バイトは DiffServ コード ポイント（DSCP）値に変換できますが、DSCP 値を直接入力することはできません。DSCP 値を使用するには、それに 4 を掛けて、結果を *number* 引数の値として入力します。

ステップ 9 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# timeout 10000
```

IP SLA 動作がその要求パケットからの応答を待機する時間を設定します。

- *milliseconds* 引数を使用して、動作が応答を待機するミリ秒数を指定します。

ステップ 10 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# tag ipsla
```

（任意）IP SLA 動作のユーザ指定 ID を作成します。

ステップ 11 exit

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# exit
```

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 12 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 13 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 14 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 15 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 16 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。

- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 17 **commit**

ステップ 18 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router # show ipsla statistics 432
```

現在の統計情報を表示します。

ICMP パスエコー動作の設定

IP SLA ICMP パスエコー動作は、IP SLA 動作が宛先に到達するためにたどるパスに沿った各ホップの統計情報を記録します。ICMP パスエコー動作では、traceroute 機能を使用してパスを検出することにより、Cisco ルータとネットワーク上の IP デバイスの間のホップバイホップ応答時間が判断されます。

送信元 IP SLA デバイスは、traceroute を使用して宛先 IP デバイスへのパスを検出します。その後、ping を使用して、送信元 IP SLA デバイスと、宛先 IP デバイスへのパス中の以降の各ホップの間の応答時間が測定されます。

（注） ICMP パスエコー動作では、IP SLA レスポンダをイネーブルにする必要はありません。

基本的な ICMP パスエコー動作を設定およびスケジューリングするのか、省略可能なパラメータを使用した ICMP パスエコー動作を設定およびスケジューリングするのかに応じて、次のいずれかの手順を実行します。

送信元デバイスでの基本の ICMP パスエコー動作の設定およびスケジューリング

オプションパラメータを指定せずに ICMP パスエコー動作をイネーブルにしてスケジューリングできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp path-echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp path-echo
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)#
```

ICMP パスエコー動作タイプを定義します。

ステップ 4 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 5 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 6 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 7 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 8 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 9 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 10 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 11 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 12 **commit**

ステップ 13 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

送信元デバイスでの省略可能なパラメータを使用した ICMP パスエコー動作の設定およびスケジューリング

送信元デバイスで ICMP パスエコー動作をイネーブルにして、省略可能な IP SLA パラメータを設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp path-echo**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp path-echo
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)#
```

ICMP パスエコー動作タイプを定義します。

ステップ 4 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-imcp-path-echo)# vrf VPN-A
```

（任意）ICMP パスエコー動作の中で、デフォルト以外のルーティング テーブルを使用して VPN のモニタリングをイネーブルにします。最大 32 文字の英数字です。

ステップ 5 **lsr-path** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-imcp-path-echo)# lsr-path 20.25.22.1
```

ルーズ ソース ルーティング パスを使用することを指定します。

ステップ 6 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 7 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 8 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# datasize request 512
```

（任意）指定した IP SLA 動作の要求パケットのペイロードにおけるプロトコル データ サイズを設定します。

- プロトコルデータサイズ（バイト単位）を指定するには、*bytes* 引数を使用します。範囲は 0 ～ 16384 です。デフォルト値は 36 バイトです。

ステップ 9 **tos** *number*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# tos 5
```

IP SLA 動作の IP ヘッダーに、タイプ オブ サービス（ToS）バイトを定義します。

（注） ToS バイトは DiffServ コード ポイント（DSCP）値に変換できますが、DSCP 値を直接入力することはできません。DSCP 値を使用するには、それに 4 を掛けて、結果を *number* 引数として入力します。

ステップ 10 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# timeout 10000
```

IP SLA 動作がその要求パケットからの応答を待機する時間を設定します。

- *milliseconds* 引数を使用して、動作が応答を待機するミリ秒数を指定します。

ステップ 11 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# tag ipsla
```

（任意）IP SLA 動作のユーザ指定 ID を作成します。

ステップ 12 **lsr-path** *ipaddress1* {*ipaddress2* {*...* {*ipaddress8*}}}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# lsr-path 20.25.22.1
```

ICMP エコー応答時間を測定するパスを指定します。

- （任意）宛先へのパス中の中間ノードの *ip address* 引数を使用します。

ステップ 13 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-echo)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 14 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 15 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 1
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 16 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 17 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 18 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値はinactiveです。**start-time**コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 19 **commit**

ステップ 20 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

ICMP パスジッター動作の設定

IP SLA ICMP パスジッター動作は、IP ネットワーク内のホップバイホップ ジッター、パケット損失、および遅延測定統計情報を提供します。パスジッター動作は、一方向データの総計と往復データの総計を提供する標準的な UDP ジッター動作とは異なる機能を果たします。

ICMPパスジッター動作は、標準的なUDPジッター動作を補完するものとして使用できます。たとえば、UDPジッター動作から得られた結果が予期しない遅延や高いジッター値を示すことがあります。この場合にICMPパスジッター動作を使用すると、ネットワークパスのトラブルシューティングを行い、伝送パス沿いの特定のセグメントでトラフィックが渋滞していないかどうかを確認できます。

ICMP パス ジッター動作は、まず traceroute ユーティリティを使用して送信元から宛先までのホップバイホップ IP ルートを検出し、次に ICMP エコーを使用して、パス沿いの各ホップの応答時間、パケット損失、およびジッターの概算値を測定します。ICMP パスジッター動作を使用して得られたジッター値は、ターゲットノードでの遅延が考慮されていないため、近似値です。

ICMPパスジッター動作は、送信元デバイスから指定した宛先デバイスまでのIPパスをトレースし、次にそのトレース パス沿いの各ホップに N 個のエコー プローブを T ミリ秒間隔で送信します。動作全体は、F 秒ごとに 1 回の頻度で繰り返されます。次の表に示すように、属性はユーザ設定可能です。

表 *5 : ICMP* パスジッター動作のパラメータ

ICMP パスジッター動作のパラメータ	デフォルト	設定方法
エコー プローブの数（N）	10 個のエコー	• *operation-number* 引数を指定した **ipsla operation** コマンド • *count* 引数を指定した **packet count** コマンド
エコー プローブ間隔（ミリ秒単位）（T）	20 ミリ秒	• *operation-number* 引数を指定した **ipsla operation** コマンド • *interval* 引数を指定した **packet interval** コマンド
動作の繰り返し頻度（F）	60 秒に 1 回	• *operation-number* 引数を指定した **ipsla operation** コマンド • *seconds* 引数を指定した **frequency** コマンド

基本的な ICMP パスジッター動作を設定およびスケジューリングするのか、追加のパラメータを使用した ICMP ジッター動作を設定およびスケジューリングするのかに応じて、次のいずれかの手順を実行します。

基本的な ICMP パスジッター動作の設定およびスケジューリング

動作の一般的なデフォルト特性を使用して ICMP パスジッター動作を設定およびスケジューリングできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp path-jitter**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp path-jitter
```

ICMP パスジッター動作タイプを定義します。

ステップ 4 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 5 **packet count** *count*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# packet count 30
```

（任意）プローブ中に送信されるパケット数を指定します。UDP ジッター動作の場合、範囲は 1 ～ 60000 です。ICMP パスジッター動作の場合、範囲は 1 ～ 100 です。

送信されるデフォルトのパケット数は 10 です。

ステップ 6 **packet interval** *interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# packet interval 30
```

（任意）パケット間隔を指定します。パケット間のデフォルト間隔は 20 ミリ秒です。

ステップ 7 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 8 exit

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 9 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 10 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 11 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 12 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 13 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

（任意）動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 14 **commit**

ステップ 15 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

追加パラメータを指定した ICMP パスジッター動作の設定およびスケジューリング

送信元デバイスで ICMP パスエコー動作をイネーブルにして、省略可能な IP SLA パラメータを設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type icmp path-jitter**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp path-jitter
```

ICMP パスジッター動作タイプを定義します。

ステップ 4 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-imcp-path-jitter)# vrf VPN-A
```

（任意）ICMP パスジッター動作の中で、デフォルト以外のルーティング テーブルを使用して VPN のモニタリングをイネーブルにします。最大 32 文字の英数字です。

ステップ 5 **lsr-path** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-imcp-path-jitter)# lsr-path 20.25.22.1
```

ルーズ ソース ルーティング パスを使用することを指定します。

ステップ 6 **destination address** *ipv4address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# destination address 12.25.26.10
```

正しい動作タイプの宛先の IP アドレスを指定します。

ステップ 7 **packet count** *count*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# packet count 30
```

（任意）プローブ中に送信されるパケット数を指定します。UDP ジッター動作の場合、範囲は 1 ～ 60000 です。ICMP パスジッター動作の場合、範囲は 1 ～ 100 です。

送信されるデフォルトのパケット数は 10 です。

ステップ 8 **packet interval** *interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# packet interval 30
```

（任意）パケット間隔を指定します。パケット間のデフォルト間隔は 20 ミリ秒です。

ステップ 9 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# frequency 300
```

（任意）指定した IP SLA 動作がネットワークに送信されるレートを設定します。

- （任意）*seconds* 引数を使用して、IP SLA 動作間隔の秒数を指定します。有効な値の範囲は 1 ～ 12604800 秒です。デフォルトは 60 秒です。

ステップ 10 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# datasize request 512
```

（任意）指定した IP SLA 動作の要求パケットのペイロードにおけるプロトコル データ サイズを設定します。

- プロトコルデータサイズ（バイト単位）を指定するには、*size* 引数を使用します。ジッターのデフォルトは 36 バイトです。有効な範囲は 0 ～ 16384 バイトです。

ステップ 11 **tos** *number*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# tos 1
```

IP SLA 動作の IP ヘッダーに、タイプ オブ サービス（ToS）バイトを定義します。

（注） ToS バイトは DiffServ コード ポイント（DSCP）値に変換できますが、DSCP 値を直接入力することはできません。DSCP 値を使用するには、それに 4 を掛けて、結果を *number* 引数として入力します。

ステップ 12 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# timeout 10000
```

IP SLA 動作がその要求パケットからの応答を待機する時間を設定します。

- *milliseconds* 引数を使用して、動作が応答を待機するミリ秒数を指定します。

ステップ 13 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# tag ipsla
```

（任意）IP SLA 動作のユーザ指定 ID を作成します。

ステップ 14 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-icmp-path-jitter)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA 動作コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 15 **ipsla schedule operation** *op-num*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 16 **life** {**forever** | *seconds*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life 30
```

forever キーワードを指定すると、動作を無期限で実行するようにスケジューリングされます。*seconds* 引数を指定すると、動作のライフタイムが秒単位でスケジューリングされます。デフォルトの動作のライフタイムは 3600 秒（1 時間）です。

ステップ 17 **ageout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# ageout 3600
```

（任意）情報をアクティブに収集していない場合、動作をメモリに常駐させておく時間を秒数で指定します。デフォルト値の 0 秒は、動作がタイムアウトしないことを意味します。

ステップ 18 **recurring**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# recurring
```

（任意）動作が毎日指定された時刻に自動的に開始され、指定された期間実行されるように指定します。

ステップ 19 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 20 **commit**

ステップ 21 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の統計情報を表示します。

IP SLA MPLS LSP ping 動作およびトレース動作の設定

MPLS LSP ping 動作とトレース動作を使用すると、サービス プロバイダーは、ラベル スイッチド パス（LSP）をモニタし、MPLS フォワーディングの問題をすばやく切り分けることができます。送信元ルータとターゲット ルータの間のネットワーク接続の問題をトラブルシューティングするには、これらの IP SLA 動作を使用します。LSP をテストするため、MPLS LSP ping動作とトレース動作は、エコー要求パケットを送信しエコー応答パケットを受信します。

MPLS LSP ping 動作またはトレース動作を設定およびスケジューリングするには、次のいずれかのタスクを実行します。

MPLS LSP ping 動作の設定およびスケジューリング

MPLS LSP ping 動作は、LSP の終端にエコー要求（ユーザ データグラム プロトコル（UDP）パケット）を送信し、診断データが格納されたエコー応答を受信することで、MPLS ネットワーク内の LSP パスに沿ったルータの接続性をテストします。

MPLS エコー要求パケットは、検証対象の LSP に関連付けられた適切なラベル スタックを使用してターゲット ルータに送信されます。ラベル スタックを使用すると、パケットは LSP 自体を介して転送されます。

MPLS エコー要求パケットの宛先 IP アドレスは、ラベル スタックの選択に使用されるアドレスとは異なります。宛先 IP アドレスは、127.x.y.z/8 アドレスとして定義されます。127.x.y.z/8 アドレスを使用すると、LSP が切断された場合に IP パケットが宛先に IP スイッチングされるのを防ぐことができます。

MPLS エコー応答は、MPLS エコー要求に応じて送信されます。応答は IP パケットとして送信され、IP、MPLS、または両方のスイッチング タイプの組み合わせを使用して転送されます。MPLS エコー応答パケットの送信元アドレスは、エコー応答を生成するルータから取得されたアドレスです。宛先アドレスは、MPLS エコー要求パケットを送信したルータの送信元アドレスです。MPLS エコー応答の宛先ポートは、エコー要求の送信元ポートに設定されます。

MPLS LSP ping 動作では、サポートされているいずれかの Forwarding Equivalence Class（FEC; 転送等価クラス）エンティティを使用して、ping 送信元と各 FEC の出力ノード間の LSP の接続性が検証されます。MPLS LSP ping 動作では、次の FEC タイプがサポートされています。

- LDP IPv4 プレフィックス（**target ipv4** コマンドで設定）
- MPLS TE トンネル（**target traffic-eng tunnel** コマンドで設定）
- 疑似回線（**target pseudowire** コマンドで設定）

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

IP SLA 動作を設定し、動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type mpls lsp ping**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type mpls lsp ping
```

MPLS LSP ping 動作を設定し、IP SLA MPLS LSP ping 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 **output interface** *type interface-path-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# output interface pos 0/1/0/0
```

（任意）LSP ping 動作で使用されるエコー要求出力インターフェイスを設定します。

（注） MPLS LSP ping 動作で使用されるターゲットとして疑似回線が指定されている場合は、**output interface** コマンドを使用できません。

ステップ 5 **target** {**ipv4** *destination-address destination-mask* | **traffic-eng tunnel** *tunnel-interface* | **pseudowire** *destination-address circuit-id*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# target ipv4 10.25.26.10 255.255.255.255
```

または

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# target ipv4 10.25.26.10/32
```

または

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# target traffic-eng tunnel 12
```

または

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# target pseudowire 192.168.1.4 4211
```

MPLS LSP ping 動作のターゲット宛先を、LDP IPv4 アドレス、MPLS トラフィック エンジニアリング トンネル、または疑似回線として指定します。

ステップ 6 **lsp selector ipv4** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# lsp selector ipv4 127.0.0.2
```

（任意）MPLS LSP ping 動作の LSP を選択するために使用されるローカル ホスト IPv4 アドレスを指定します。

ステップ 7 **force explicit-null**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# force explicit-null
```

（任意）エコー要求を送信するときに、LSP のラベル スタックに明示的な null ラベルを追加します。

ステップ 8 **reply dscp** *dscp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# reply dscp 2
```

（任意）エコー応答パケットで使用する DiffServ コード ポイント（DSCP）値を指定します。有効な値は 0 ～ 63 です。

数値の代わりに、EF（緊急転送）やAF11（保証転送クラスAF11）などの予約されたキーワードを指定できます。

ステップ 9 **reply mode** {**control-channel** | **router-alert**}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# reply mode router-alert
```

または

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# reply mode control-channel
```

（任意）MPLS LSP ping 動作の制御チャネルを経由してエコー応答パケットを送信するか、IP ルータ アラートを含む IPv4 UDP パケットとして応答するように、エコー要求を設定します。ルータ アラート応答モードでは、エコー応答パケットが宛先に戻る場合に、中間ホップごとに中継 LSR ルータによって特別な処理が実行されるように強制されます。

（注） **control-channel** キーワードは、ターゲットが疑似回線に設定されている場合のみ使用できます。

ステップ 10 **exp** *exp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# exp 5
```

（任意）エコー応答パケットのヘッダーで使用するMPLS試験フィールド（EXP）値を指定します。有効な値の範囲は 0 ～ 7 です。

ステップ 11 **ttl** *time-to-live*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# ttl 200
```

（任意）エコー要求パケットの MPLS ラベルで使用する存続可能時間（TTL）値を指定します。有効な値は、1 ～ 255 です。

ステップ 12 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
```

```
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA MPLS LSP Ping コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 13 **ipsla schedule operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla schedule operation 432
RP/0/RP0/cpu 0: router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 14 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 15 **commit**

ステップ 16 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router# show ipsla statistics 432
```

現在の MPLS LSP ping 動作の IP SLA 統計情報を表示します。

MPLS LSP トレース動作の設定およびスケジューリング

MPLS LSP トレース動作は、エコー要求（UDP パケット）を各中継ラベル スイッチング ルータ（LSP）のコントロール プレーンに送信することにより、MPLS ネットワーク内のターゲット ルータへの LSP パスのホップバイホップ ルートをトレースします。中継 LSR では、さまざまなチェックが実行され、LSP パスの中継 LSR であることが特定されます。トレース動作では、ネットワーク接続のトラブルシューティングと、障害があるホップバイホップのローカライズを実行できます。

エコー要求パケットとエコー応答パケットが LSP を検証します。MPLS LSP トレース動作の成功は、ラベル付きパケットを受信したときに MPLS エコー要求を処理する中継ルータに依存します。

中継ルータは、存続可能時間（TTL）が期限切れになった MPLS パケットまたは LSP の切断に対応して、中継ホップに関する情報を含む MPLS エコー応答を返します。MPLS エコー応答の宛先ポートは、エコー要求の送信元ポートに設定されます。

MPLS LSP トレース動作では、各中継 LSR が、トレースされている Forwarding Equivalence Class（FEC;転送等価クラス）エンティティのタイプに関連する情報を返します。この情報により、トレース動作では、ローカルフォワーディングの情報がルーティングプロトコルによって LSP パスとして特定された情報と一致するかどうかをチェックできます。

MPLS ラベルは、LSP で使用されている FEC のタイプに従って、パケットにバインドされます。MPLS LSP トレース動作では、次の FEC タイプがサポートされています。

- LDP IPv4 プレフィックス（**target ipv4** コマンドで設定）
- MPLS TE トンネル（**target traffic-eng tunnel** コマンドで設定）

手順

ステップ 1 **configure**

ステップ 2 **ipsla operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla operation 432
```

IP SLA 動作を設定し、動作番号を指定します。範囲は 1 ～ 2048 です。

ステップ 3 **type mpls lsp trace**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-op)# type mpls lsp trace
```

MPLS LSP トレース動作を設定し、IP SLA MPLS LSP トレース 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 **output interface** *type interface-path-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-ping)# output interface pos 0/1/0/0
```

（任意）LSP トレース動作で使用されるエコー要求出力インターフェイスを設定します。

ステップ 5 次のいずれかを実行します。

- **target ipv4** *destination-address destination-mask*
- **target traffic-eng tunnel** *tunnel-interface*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# target ipv4 10.25.26.10
255.255.255.255

RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# target ipv4 10.25.26.10/32
```

または

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# target traffic-eng tunnel 12
```

MPLS LSP トレース動作のターゲット宛先を、LDP IPv4 アドレスまたは MPLS トラフィック エンジニアリング トンネルとして指定します。

ステップ 6 **lsp selector ipv4** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# lsp selector ipv4 127.0.0.2
```

（任意）IPv4 LSP ping 動作の LSP を選択するために使用されるローカル ホスト MPLS アドレスを指定します。

ステップ 7 **force explicit-null**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# force explicit-null
```

（任意）エコー要求を送信するときに、LSP のラベル スタックに明示的な null ラベルを追加します。

ステップ 8 **reply dscp** *dscp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# reply dscp 2
```

（任意）エコー応答パケットで使用する DiffServ コード ポイント（DSCP）値を指定します。有効な値は 0 ～ 63 です。

数値の代わりに、EF（緊急転送）や AF11（保証転送クラス AF11）などの予約されたキーワードを指定できます。

ステップ 9 **reply mode router-alert**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# reply mode router-alert
```

（任意）IP ルータ アラートを使用した IPv4 UDP パケットとして応答するようにエコー要求を設定します。ルータ アラート応答モードでは、エコー応答パケットが宛先に戻る場合に、中間ホップごとに中継 LSR ルータによって特別な処理が実行されるように強制されます。

ステップ 10 **exp** *exp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# exp 5
```

（任意）エコー応答パケットのヘッダーで使用する MPLS 試験フィールド（EXP）値を指定します。有効な値の範囲は 0 ～ 7 です。

ステップ 11 **ttl** *time-to-live*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# ttl 20
```

（任意）エコー要求パケットの MPLS ラベルで使用する存続可能時間（TTL）値を指定します。有効な値は、1 ～ 255 です。

ステップ 12 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-lsp-trace)# exit
RP/0/RP0/cpu 0: router(config-ipsla-op)# exit
RP/0/RP0/cpu 0: router(config-ipsla)# exit
RP/0/RP0/cpu 0: router(config)#
```

IP SLA MPLS LSP トレース コンフィギュレーション モードおよび IP SLA コンフィギュレーション モードを終了します。グローバル コンフィギュレーション モードに戻ります。

ステップ 13 **ipsla schedule operation** *operation-number*

例：

```
RP/0//CPU0:router(config)# ipsla schedule operation 432
RP/0//CPU0:router(config-ipsla-sched)#
```

動作の開始時間をスケジューリングします。基本スケジュールを設定できます。

ステップ 14 **start-time** [*hh:mm:ss* {*day* | *month day*} | **now** | **pending** | **after** *hh:mm:ss*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time 01:00:00
```

動作が開始される時刻を指定します。ここでは、次のキーワードについて説明します。

- （任意）**pending** キーワードを使用して、動作を保留（未開始）状態にしておくように設定します。デフォルト値は inactive です。**start-time** コマンドが指定されていない場合、開始時刻が設定されるか、即時スタートを実行するトリガーが発生するまで、情報は収集されません。
- （任意）**now** キーワードを使用して、動作を即時スタートする必要があることを示します。
- （任意）**after** キーワードおよび関連する引数を使用して時刻を指定します。ここで指定した時刻以降に、動作が情報の収集を開始します。

ステップ 15 **commit**

ステップ 16 **show ipsla statistics** [*operation-number*]

例：

```
RP/0/RP0/cpu 0: router # show ipsla statistics 432
```

トレース動作の現在の IP SLA 統計情報を表示します。

IP SLA 反応としきい値のモニタリングの設定

IP SLA でしきい値を設定して、しきい値違反を通知するようにするには、**ipsla reaction operation** コマンドと **ipsla reaction trigger** コマンドが必要です。次の手順を実行して、IP SLA 反応としきい値のモニタリングを設定します。

IP SLA 反応のモニタ対象の要素の設定

IP SLA 反応は、モニタ対象の値が指定レベルを上回ったり下回ったりした場合や、モニタ対象のイベント（タイムアウトやに接続の切断など）が発生した場合にトリガーされるように設定されます。これらのモニタ対象の値およびイベントは、モニタ対象の要素と呼ばれます。特定の動作で反応が発生するように、反応の条件を設定できます。

利用できるモニタ対象の要素のタイプは、次の項に示されています。

接続の切断違反のトリガーの設定

モニタ対象の動作に接続の切断がある場合の反応を設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react connection-loss
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

connection-loss キーワードを使用して、モニタ対象の動作で接続の切断がある場合に反応が発生するように指定します。

ステップ 4 **commit**

ジッター違反のトリガーの設定

ジッター値は送信元から宛先の値および宛先から送信元の値として計算されます。各方向または両方向のジッター値が指定しきい値を上回るか下回る場合に、トラップなどのイベントをトリガーできます。jitter-average をモニタ対象の要素として設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**jitter-average** {**dest-to-source** | **source-to-dest**}]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react jitter-average
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

反応は、平均ラウンドトリップジッター値が上限または下限のしきい値に違反している場合に発生します。**jitter-average** キーワードには、次のオプションが用意されています。

- **dest-to-source**：宛先から送信元（DS）のジッター平均を指定します。
- **source-to-dest**：送信元から宛先（SD）のジッター平均を指定します。

ステップ 4 **commit**

パケット損失違反のトリガーの設定

パケット損失値は送信元から宛先の値および宛先から送信元の値として計算されます。各方向のパケット損失値が指定しきい値を上回るか下回る場合に、トラップなどのイベントをトリ

ガーできます。パケット損失をモニタ対象の要素として設定するには、このタスクを実行します。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react**[**packet-loss** [**dest-to-source** | **source-to-dest**]]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react packet-loss dest-to-source
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

パケット損失値違反の反応が指定されます。**packet-loss** キーワードには、次のオプションが用意されています。

- **dest-to-source**：宛先から送信元（DS）のパケット損失違反を指定します。
- **source-to-dest**：送信元から宛先（SD）のパケット損失違反を指定します。

ステップ 4 **commit**

ラウンドトリップ違反のトリガーの設定

ラウンドトリップ時間（RTT）は、すべての IP SLA 動作のモニタ対象値です。rtt 値が指定しきい値を上回るか下回る場合に、トラップなどのイベントをトリガーできます。rtt をモニタ対象の要素として設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**rtt**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react rtt
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

rtt キーワードを使用して、ラウンドトリップ値が上限または下限のしきい値に違反する場合に発生する反応を指定します。

ステップ 4 **commit**

タイムアウト違反のトリガーの設定

タイムアウト違反のトリガーを設定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**timeout**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react timeout
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

timeout キーワードを使用して、モニタ対象の動作にタイムアウトがある場合に発生する反応を指定します。

ステップ 4 **commit**

エラー検証違反のトリガーの設定

エラー検証違反がある場合の反応を指定できます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react verify-error
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

verify-error キーワードを使用して、エラー検証違反があるときに発生する反応を指定します。

ステップ 4 **commit**

IP SLA 反応のしきい値違反タイプの設定

各モニタリング対象の要素では、次の項目を指定できます。

- しきい値をチェックするための条件
- 反応を発生させることができる条件の発生パターン（しきい値タイプなど）

たとえば、**threshold type immediate** コマンドを使用すると、対象の条件が確認されるとすぐに特定の要素で反応が発生するように指定できます。または、**threshold type consecutive** コマンドを使用すると、3 回連続して条件が確認されると反応が発生するように指定できます。

しきい値のタイプでは、イベントをトリガーするしきい値違反（またはしきい値違反の組み合わせ）のタイプを定義します。

この表では、しきい値違反タイプを一覧で示します。

表 *6 : IP SLA* 反応のしきい値違反タイプ

しきい値違反のタイプ	説明
連続	違反が何回か連続して発生した後にのみイベントをトリガーします。たとえば、連続した違反タイプを使用すると、タイムアウトが 5 回連続して発生した後や、ラウンドトリップ時間が上限のしきい値を 5 回連続して上回った後にアクションが実行されるように設定できます。詳細については、「連続した違反のイベントの生成 （68 ページ）」を参照してください。
即時	反応タイプ（応答時間など）の値が上限しきい値を上回るか、下限しきい値を下回る場合や、タイムアウト、接続の切断、verify-error イベントが発生した場合にイベントを即座にトリガーします。詳細については、「各違反のイベントの生成 （67 ページ）」を参照してください。
X / Y	y 回のプローブ動作以内に x 回の違反が発生すると（x 回/y 回）、イベントをトリガーします。詳細については、「X / Y 違反のイベントの生成 （69 ページ）」を参照してください。
平均	プローブ動作の X 回の値の平均合計が、指定された上限しきい値を上回るか、下限しきい値を下回るときにイベントをトリガーします。詳細については、「平均違反のイベントの生成 （70 ページ）」を参照してください。

各違反のイベントの生成

指定された条件が満たされるたびに、トラップ生成したり、別の動作をトリガーしたりできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react timeout
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

モニタ対象の動作にタイムアウトがあると、反応が指定されます。

ステップ 4 **threshold type immediate**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)# threshold type immediate
```

しきい値違反に対してただちにアクションを実行します。

ステップ 5 **commit**

連続した違反のイベントの生成

連続した回数の違反が発生した後に、トラップ生成したり、別の動作をトリガーしたりできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react connection-loss
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

モニタ対象の動作に接続の切断があると、反応が指定されます。

ステップ 4 **threshold type consecutive** *occurrences*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)# threshold type consecutive 8
```

連続した回数の違反が発生した後にアクションを実行します。反応条件が連続した発生回数に対して設定されている場合、デフォルト値はありません。発生回数は、しきい値タイプの指定時に設定されます。連続した違反回数は 1 ～ 16 です。

ステップ 5 **commit**

X/Y 違反のイベントの生成

y 回のプローブ動作以内に x 回の違反が発生した後に（x 回/y 回）、トラップ生成したり、別の動作をトリガーしたりできます。例として、**react** コマンドに **rtt** キーワードを指定して使用します。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react rtt
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

ラウンドトリップ値が上限しきい値または下限しきい値に違反している場合に反応が発生するように指定します。

ステップ 4 **threshold type xofy** *X value Y value*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)# threshold type xofy 7 7
```

モニタ対象の要素でしきい値違反などの反応条件が発生した場合、*y* 回のプローブ動作以内に *x* 回の違反が発生すると（*x* 回/*y* 回）、**action** コマンドでの定義に従って、アクションが実行されます。デフォルトは、*x-value* および *y-value* の両方とも 5 です（**xofy** 5 5）。各値の有効範囲は 1 ～ 16 です。

ステップ 5 **commit**

平均違反のイベントの生成

プローブ動作のXの平均合計数が下限しきい値または上限しきい値に違反する場合、トラップ生成したり、別の動作をトリガーしたりできます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss** | **jitter-average** { **dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react packet-loss dest-to-source
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

反応をモニタする要素を指定します。

パケット損失値違反の反応が指定されます。**packet-loss** キーワードには、次のオプションが用意されています。

- **dest-to-source**：宛先から送信元（DS）のパケット損失違反を指定します。
- **source-to-dest**：送信元から宛先（SD）のパケット損失違反を指定します。

ステップ 4 **threshold type average** *number-of-probes*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)# threshold type average 8
```

平均値がしきい値に違反した場合にアクションを実行します。

ステップ 5 **commit**

反応イベントの指定

反応条件が検出されるとき、**action** コマンドを使用して、発生するアクションのタイプを設定できます。次のアクション タイプが設定されます。

- **logging**：**logging** キーワードが設定されると、反応が発生したことを示すメッセージがコンソールに生成されます。
- **trigger**：**trigger** キーワードが設定されると、1 回以上の他の動作を開始できます。その結果、**ipsla reaction trigger** *op1 op2* コマンドで開始できる動作を制御できます。このコマンドは、*op1* がアクションタイプのトリガーを生成すると、動作 *op2* を開始できることを示します。

反応イベントを指定できます。例として、**react** コマンドに **connection-loss** キーワードを指定して使用します。

手順

ステップ 1 **configure**

ステップ 2 **ipsla reaction operation** *operation-number*

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla reaction operation 432
```

IP SLA エージェントが制御するイベントに基づいた特定のアクションを設定します。*operation-number* 引数は、設定されている反応に対する IP SLA 動作の数です。範囲は 1 ～ 2048 です。

ステップ 3 **react** [**connection-loss** | **jitter-average** {**dest-to-source** | **source-to-dest**} | **packet-loss** [**dest-to-source** | **source-to-dest**] | **rtt** | **timeout** | **verify-error**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react)# react connection-loss
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)#
```

モニタ対象の動作で接続の切断がある場合の反応を指定します。

ステップ 4 **action** [**logging** | **trigger**]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-react-cond)# action logging
```

react コマンドを設定した場合、またはしきい値イベントが発生した場合に実行されるアクションまたはアクションの組み合わせを指定します。次のアクション タイプが記述されます。

- **logging**：モニタ対象の要素で指定された違反タイプが発生した場合に、ロギングメッセージを送信します。IP SLA エージェントは syslog を生成し、SNMP に通知します。トラップを生成するかどうかは、SNMP エージェントによって決定されます。
- **trigger**：違反条件に一致した場合に保留からアクティブへの移行が発生する 1 つまたは複数の動作の動作ステートを決定します。トリガーされるターゲット動作は、**ipsla reaction trigger** コマンドを使用して指定されます。ターゲット動作は、そのターゲット動作の

lifetime 値で指定された存続期間が経過するまで継続します。トリガーされたターゲット動作は、存続期間が終了するまで、再度トリガーされることはありません。

ステップ 5 **commit**

送信元 PE ルータでの MPLS LSP モニタリング インスタンスの設定

このタスクを実行して、MPLS LSP モニタ（MPLSLM）インスタンスの動作パラメータを設定します。IP SLA 測定統計情報は送信元 PE ルータに保存されます。

MPLS LSP モニタ ping またはトレース インスタンスを設定するには、次のタスクのいずれかを実行します。

MPLS LSP モニタリング ping インスタンスの設定

始める前に

（注） MPLS LSP モニタリングは PE ルータで設定されます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla
```

IP SLA コンフィギュレーション モードを開始し、IP サービス レベル契約を設定します。

ステップ 3 **mpls discovery vpn**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls discovery vpn
```

（任意）MPLS VPN BGP ネクスト ホップ ネイバー探索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 **interval** *minutes*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-discovery-vpn)# interval 120
```

（任意）有効ではなくなったルーティング エントリが MPLS VPN の BGP ネクスト ホップ ネイバー探索データベースから削除される間隔を指定します。デフォルトの間隔は 60 分です。

ステップ 5 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-discovery-vpn)# exit
```

MPLS ディスカバリ VPN コンフィギュレーション モードを終了します。

ステップ 6 **mpls lsp-monitor**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls lsp-monitor
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)#
```

MPLS LSP モニタ モードを開始します。このモードから、LSP モニタ インスタンスの設定、LSP モニタ インスタンスの反応の設定、または LSP モニタ インスタンスのスケジューリングを実行できます。

ステップ 7 **monitor** *monitor-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)# monitor 1
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-def)#
```

MPLS LSP モニタ インスタンスを設定し、IP SLA MPLS LSP モニタ コンフィギュレーション モードを開始します。

ステップ 8 **type mpls lsp ping**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-def)# type mpls lsp ping
```

検出されたそれぞれの BGP ネクスト ホップ アドレスに対して、自動的に MPLS LSP ping 動作を作成し、対応するコンフィギュレーション モードを開始して、パラメータを設定します。

ステップ 9 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# vrf SANJOSE
```

（任意）ping 動作で特定のバーチャル プライベート ネットワーク（VPN）ルーティングおよび転送（VRF）インスタンスのモニタリングをイネーブルにします。VRF を指定しない場合、MPLS LSP モニタリング インスタンスはすべての VRF をモニタします。

ステップ 10 **scan interval** *scan-interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# scan interval 300
```

（任意）MPLS LSP モニタ インスタンスが BGP ネクスト ホップ ネイバーの更新のためにスキャン キューをチェックする間隔（分単位）を指定します。デフォルトの間隔は240分です。

各間隔では、MPLS LSP モニタ インスタンス スキャン キューにリストされている新しく検出された BGP ネクスト ホップ ネイバーごとに、新しい IP SLA 動作が自動的に作成されます。

ステップ 11 **scan delete-factor** *factor-value*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# scan delete-factor 2
```

（任意）有効ではなくなった BGP ネクスト ホップ ネイバーに対する IP SLA 動作を自動的に削除するまでに、MPLS LSP モニタ インスタンスがスキャン キューをチェックする回数を指定します。

デフォルトのスキャン ファクタは 1 です。つまり、MPLS LSP モニタ インスタンスがスキャン キューで更新をチェックするたびに、有効ではなくなった BGP ネクスト ホップ ネイバーの IP SLA 動作が削除されます。

スキャン ファクタが 0 に設定されると、MPLS LSP モニタ インスタンスによって IP SLA 動作は削除されません。この設定は推奨しません。

ステップ 12 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# timeout 50000
```

（任意）各 MPLS LSP 動作が LSP 検証（LSPV）サーバからの応答を待機する時間の長さを指定します。デフォルト値は 5000 ミリ秒です。

ステップ 13 **datasize request** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# datasize request 512
```

（任意）MPLS LSP エコー要求パケットのペイロード サイズを指定します。デフォルト値は 100 バイトです。

（注） このコマンドは、MPLS LSP ping モードだけで利用できます。

ステップ 14 **lsp selector ipv4** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# lsp selector ipv4 127.10.10.1
```

（任意）複数の LSP からラベルスイッチドパス（LSP）を選択するために使用するローカルホスト IP アドレス（127.*x.x.x*）を指定します。デフォルト値は 127.0.0.1 です。

ステップ 15 **force explicit-null**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# force explicit-null
```

（任意）MPLS LSP エコー要求パケットのラベル スタックに、明示的な Null ラベルが追加されるかどうかを指定します。デフォルトでは、無効になっています。

ステップ 16 **reply dscp** *dscp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# reply dscp 5
```

（任意）MPLS LSP エコー応答パケットの IP ヘッダーで使用される DiffServ サービス コード ポイント（DSCP）値を指定します。

ステップ 17 **reply mode router-alert**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# reply mode router-alert
```

（任意）MPLS LSP エコー応答パケットでルータ アラート オプションの使用をイネーブルにします。デフォルトでは、無効になっています。

ステップ 18 **ttl** *time-to-live*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# ttl 200
```

（任意）MPLS LSP 動作に使用されるエコー要求パケットの最大ホップ カウントを指定します。デフォルト値は 255 です。

ステップ 19 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# tag mplslm-tag
```

（任意）MPLS LSP 動作のユーザ指定 ID を作成します。

ステップ 20 **exp** *exp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# exp 7
```

（任意）エコー要求パケットの MPLS ヘッダーで使用される試験フィールド値を指定します。デフォルト値は 0 です。

ステップ 21 **statistics hourly** [**buckets** *hours*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# statistics hourly buckets 2
```

（任意）MPLS LSP モニタリング インスタンスでの動作の統計情報収集パラメータを指定します。時間のデフォルト値は 2 です。

ステップ 22 **commit**

次のタスク

- 反応条件を設定します。
- MPLS LSP モニタリング インスタンス動作のスケジュールを設定します。

MPLS LSP モニタリング トレース インスタンスの設定

始める前に

（注） MPLS LSP モニタリングは PE ルータで設定されます。

手順

ステップ 1 **configure**

ステップ 2 **ipsla**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla
```

IP SLA コンフィギュレーション モードを開始し、IP サービス レベル契約を設定します。

ステップ 3 **mpls discovery vpn**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls discovery vpn
```

（任意）MPLS VPN BGP ネクスト ホップ ネイバー探索をイネーブルにします。

ステップ 4 **interval** *minutes*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-discovery-vpn)# interval 120
```

（任意）有効ではなくなったルーティング エントリが MPLS VPN の BGP ネクスト ホップ ネイバー探索データベースから削除される間隔を指定します。デフォルトの間隔は 60 分です。

ステップ 5 **exit**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mpls-discovery-vpn)# exit
```

MPLS ディスカバリ VPN コンフィギュレーション モードを終了します。

ステップ 6 **mpls lsp-monitor**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls lsp-monitor
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)#
```

MPLS LSP モニタ モードを開始します。このモードから、LSP モニタ インスタンスの設定、LSP モニタ インスタンスの反応の設定、または LSP モニタ インスタンスのスケジューリングを実行できます。

ステップ 7 **monitor** *monitor-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)# monitor 1
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-def)#
```

MPLS LSP モニタ インスタンスを設定し、IP SLA MPLS LSP モニタ コンフィギュレーション モードを開始します。

ステップ 8 **type mpls lsp trace**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplsm-def)# type mpls lsp trace
```

検出されたそれぞれの BGP ネクスト ホップ アドレスに対して自動的に MPLS LSP トレース動作を作成し、対応するコンフィギュレーション モードを開始して、パラメータを設定します。

ステップ 9 **vrf** *vrf-name*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# vrf SANJOSE
```

（任意）traceroute 動作で特定のバーチャル プライベート ネットワーク（VPN）ルーティングおよび転送（VRF）インスタンスのモニタリングをイネーブルにします。VRF を指定しない場合、MPLS LSP モニタリング インスタンスはすべての VRF をモニタします。

ステップ 10 **scan interval** *scan-interval*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# scan interval 300
```

（任意）MPLS LSP モニタ インスタンスが BGP ネクスト ホップ ネイバーの更新のためにスキャン キューをチェックする間隔（分単位）を指定します。デフォルトの間隔は 240 分です。

各間隔では、MPLS LSP モニタ インスタンス スキャン キューにリストされている新しく検出された BGP ネクスト ホップ ネイバーごとに、新しい IP SLA 動作が自動的に作成されます。

ステップ 11 **scan delete-factor** *factor-value*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# scan delete-factor 2
```

（任意）有効ではなくなった BGP ネクスト ホップ ネイバーに対する IP SLA 動作を自動的に削除するまでに、MPLS LSP モニタ インスタンスがスキャン キューをチェックする回数を指定します。

デフォルトのスキャン ファクタは 1 です。つまり、MPLS LSP モニタ インスタンスがスキャン キューで更新をチェックするたびに、有効ではなくなった BGP ネクスト ホップ ネイバーの IP SLA 動作が削除されます。

スキャン ファクタが 0 に設定されると、MPLS LSP モニタ インスタンスによって IP SLA 動作は削除されません。この設定は推奨しません。

ステップ 12 **timeout** *milliseconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# timeout 50000
```

（任意）各 MPLS LSP 動作が LSP 検証（LSPV）サーバからの応答を待機する時間の長さを指定します。デフォルト値は 5000 ミリ秒です。

ステップ 13 **lsp selector ipv4** *ip-address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# lsp selector ipv4 127.10.10.1
```

（任意）複数の LSP からラベルスイッチドパス（LSP）を選択するために使用するローカルホスト IP アドレス（127.*x.x.x*）を指定します。デフォルト値は 127.0.0.1 です。

ステップ 14 **force explicit-null**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# force explicit-null
```

（任意）MPLS LSP エコー要求パケットのラベル スタックに、明示的な Null ラベルが追加されるかどうかを指定します。デフォルトでは、無効になっています。

ステップ 15 **reply dscp** *dscp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# reply dscp 5
```

（任意）MPLS LSP エコー応答パケットの IP ヘッダーで使用される DiffServ サービス コードポイント（DSCP）値を指定します。

ステップ 16 **reply mode router-alert**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# reply mode router-alert
```

（任意）MPLS LSP エコー応答パケットでルータ アラート オプションの使用をイネーブルにします。デフォルトでは、無効になっています。

ステップ 17 **ttl** *time-to-live*

例：

```
RP/0//CPU0:router(config-ipsla-mplslm-lsp-trace)# ttl 40
```

（任意）MPLS LSP 動作に使用されるエコー要求パケットの最大ホップ カウントを指定します。デフォルト値は 30 です。

ステップ 18 **tag** *text*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# tag mplslm-tag
```

（任意）MPLS LSP 動作のユーザ指定 ID を作成します。

ステップ 19 **exp** *exp-bits*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# exp 7
```

（任意）エコー要求パケットの MPLS ヘッダーで使用される試験フィールド値を指定します。デフォルト値は 0 です。

ステップ 20 **statistics hourly** [**buckets** *hours*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-trace)# statistics hourly buckets 2
```

（任意）MPLS LSP モニタリング インスタンスでの動作の統計情報収集パラメータを指定します。時間のデフォルト値は 2 です。

ステップ 21 **commit**

次のタスク

- 反応条件を設定します。
- MPLS LSP モニタリング インスタンス動作のスケジュールを設定します。

送信元 PE ルータでの MPLS LSP モニタリング インスタンスの反応条件の設定

このタスクを実行して、MPLS LSP モニタリング インスタンスの反応条件を設定します。

始める前に

MPLS LSP モニタリング インスタンスは、反応条件を設定する前に定義してください。

手順

ステップ 1 **configure**

ステップ 2 **ipsla**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla
```

IP SLA コンフィギュレーション モードを開始し、IP サービス レベル契約を設定します。

ステップ 3 **mpls lsp-monitor**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls lsp-monitor
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)#
```

MPLS LSP モニタ モードを開始します。このモードから、LSP モニタ インスタンスの設定、LSP モニタ インスタンスの反応の設定、または LSP モニタ インスタンスのスケジューリングを実行できます。

ステップ 4 **reaction monitor** *monitor-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)# reaction monitor 2
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-react)#
```

MPLS LSP モニタ インスタンスの反応を設定し、IP SLA MPLS LSP モニタ反応のコンフィギュレーション モードを開始します。

ステップ 5 **react** {**connection-loss** | **timeout**}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-react)# react connection-loss
```

一方向の接続が切断されたり、モニタ対象の動作にタイムアウトが発生したりすると、反応が発生するように指定します。自動的に作成された動作に条件が当てはまると、反応は適用されます。

ステップ 6 **action logging**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-react-cond)# action logging
```

反応条件およびしきい値の結果として、イベントがログに記録されるように指定します。

ステップ 7 **threshold type** {**consecutive** *occurrences* | **immediate**}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-react-cond)# threshold type consecutive
```

指定した回数の違反が連続して発生した場合や、違反が発生すると即座に指定されたアクションが実行されるように指定します。*occurrences* の有効な値の範囲は 1 ～ 16 です。

ステップ 8 **commit**

次のタスク

- MPLS LSP モニタリング インスタンス動作のスケジュールを設定します。

送信元PEルータでのMPLSLSPモニタリングインスタンスのスケジュール設定

このタスクを実行して、MPLS LSP モニタリング インスタンスでの動作のスケジュールを設定します。

手順

ステップ 1 **configure**

ステップ 2 **ipsla**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla
```

IP SLA コンフィギュレーション モードを開始し、IP サービス レベル契約を設定します。

ステップ 3 **mpls lsp-monitor**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls lsp-monitor
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)#
```

MPLS LSP モニタ モードを開始します。このモードから、LSP モニタ インスタンスの設定、LSP モニタ インスタンスの反応の設定、または LSP モニタ インスタンスのスケジューリングを実行できます。

ステップ 4 **schedule monitor** *monitor-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)# schedule monitor 2
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-sched)#
```

IP SLA MPLS LSP モニタ スケジュール コンフィギュレーション モードを開始して、MPLS LSP モニタ インスタンスのスケジュールを設定します。

ステップ 5 **frequency** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-sched)# frequency 600
```

（任意）スケジュール期間が実行される頻度を指定します。デフォルト値はスケジュール期間と同じです。スケジュール期間は **schedule period** コマンドを使用して指定されます。MPLS LSP MPLS LSP モニタ インスタンスの開始時刻のスケジュールを設定する前に、この値を指定する必要があります。

ステップ 6 **schedule period** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-sched)# schedule period 300
```

その期間ですべての動作が実行されるようにスケジュールを設定する時間を秒単位で指定します。すべての動作のスケジュールは、スケジュール期間中を通して均等の間隔で設定されます。

動作のセット全体が実行される頻度を指定するには、**frequency** コマンドを指定します。頻度の値は、スケジュール期間以上である必要があります。

MPLS LSP MPLS LSP モニタ インスタンスの開始時刻のスケジュールを設定する前に、この値を指定する必要があります。

ステップ 7 **start-time** *hh*:*mm*:*ss* [*day* | *month day*]

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-sched)# start-time 11:45:00 July 4
```

MPLS LSP モニタ インスタンスが情報の収集を開始するときを指定します。スケジュールを設定した時刻を指定する必要があります。指定しない場合、情報が収集されません。

ステップ 8 **commit**

LSP パス ディスカバリ

このタスクを実行して、LSP パス ディスカバリ（LPD）およびエコー間隔、パス、スキャンなどの必要なパラメータを設定します。

手順

ステップ 1 **configure**

ステップ 2 **ipsla**

例：

```
RP/0/RP0/cpu 0: router(config)# ipsla
```

IP SLA コンフィギュレーション モードを開始し、IP サービス レベル契約を設定します。

ステップ 3 **mpls lsp-monitor**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla)# mpls lsp-monitor
```

MPLS LSP モニタ モードを開始します。このモードから、LSP モニタ インスタンスの設定、LSP モニタ インスタンスの反応の設定、または LSP モニタ インスタンスのスケジューリングを実行できます。

ステップ 4 **monitor** *monitor-id*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm)# monitor 2
```

MPLS LSP モニタ インスタンスを設定します。

ステップ 5 **type mpls lsp ping**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-def)# type mpls lsp ping
```

ラベル スイッチド パス（LSP）のエンドツーエンドの接続と MPLS ネットワークの整合性を検証します。

ステップ 6 **path discover**

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-ping)# path discover
```

LSP パス ディスカバリをイネーブルにします。

ステップ 7 **echo interval** *time*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# echo interval 777
```

パス ディスカバリ中に送信される MPLS LSP エコー要求のインターバル（ミリ秒単位）を設定します。範囲は 0 ～ 3600000 です。デフォルトは 0 です。

ステップ 8 **echo maximum lsp selector ipv4** *host address*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# echo maximum lsp selector ipv4
host_one 127.100.100.100
```

パスディスカバリ中に使用される最大セレクタ値であるローカルホストIPアドレス（127.x.x.x）を設定します。デフォルトは 127.255.255.255 です。

ステップ 9 **echo multipath bitmap-size** *size*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# echo multipath bitmap-size 50
```

パス ディスカバリ中に MPLS LSP エコー要求のダウンストリーム マッピングで送信されるセレクタの最大数を設定します。範囲は 1 ～ 256 です。デフォルトは 32 です。

ステップ 10 **echo retry** *count*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# echo retry 3
```

パス ディスカバリ中に送信される MPLS LSP エコー要求のタイムアウト リトライ回数を設定します。範囲は 0 ～ 10 です。デフォルトは 3 です。

ステップ 11 **echo timeout** *value*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# echo timeout 300
```

パス ディスカバリ中のエコー要求のタイムアウト値を設定します。範囲は 0 ～ 3600（ミリ秒単位）です。デフォルトは 5 です。

ステップ 12 **path retry** *range*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# path retry 12
```

MPLS LSP パスの再試行範囲を設定します。範囲は 1 ～ 16 です。デフォルトは 1 です。

ステップ 13 **path secondary frequency** {**both** | **connection-loss** | **timeout**} *value*}

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# path secondary frequency both 600
```

次の secondary frequency をイネーブルにします。

- タイムアウトおよび接続の切断の両方
- 接続の切断のみ
- タイムアウトのみ

（注） デフォルト値はありません。

ステップ 14 **scan period** *value*

例：

```
RP/0/RP0/cpu 0: router(config-ipsla-mplslm-lsp-lpd)# scan period 60
```

MPLS LSP スキャン期間の値を設定します。範囲は 0 ～ 7200 分です。デフォルトは 5 です。

ステップ 15 **commit**

IP サービス レベル契約を実装するための設定例

ここでは、次の設定例を示します。

IP サービス レベル契約の設定：例

次の例では、UDP ジッター動作を設定およびスケジューリングする方法を示します。

```
configure
ipsla
 operation 101
  type udp jitter
   destination address 12.2.0.2
   statistics hourly
    buckets 5
    distribution count 5
    distribution interval 1
   !
   destination port 400
   statistics interval 120
    buckets 5
   !
  !
 !
 schedule operation 101
  start-time now
  life forever
 !
!


show ipsla statistics
```

```
Fri Nov 28 16:48:48.286 GMT
Entry number: 101
    Modification time: 16:39:36.608 GMT Fri Nov 28 2014
    Start time       : 16:39:36.633 GMT Fri Nov 28 2014
    Number of operations attempted: 10
    Number of operations skipped  : 0
    Current seconds left in Life  : Forever
    Operational state of entry    : Active
    Operational frequency(seconds): 60
    Connection loss occurred      : FALSE
    Timeout occurred              : FALSE
    Latest RTT (milliseconds)     : 3
    Latest operation start time   : 16:48:37.653 GMT Fri Nov 28 2014
    Next operation start time     : 16:49:37.653 GMT Fri Nov 28 2014
    Latest operation return code  : OK
    RTT Values:
      RTTAvg  : 3          RTTMin: 3          RTTMax : 4
      NumOfRTT: 10         RTTSum: 33         RTTSum2: 111
    Packet Loss Values:
      PacketLossSD       : 0          PacketLossDS : 0
      PacketOutOfSequence: 0          PacketMIA    : 0
      PacketLateArrival  : 0          PacketSkipped: 0
      Errors             : 0          Busies       : 0
      InvalidTimestamp   : 0
    Jitter Values :
      MinOfPositivesSD: 1          MaxOfPositivesSD: 1
      NumOfPositivesSD: 2          SumOfPositivesSD: 2
      Sum2PositivesSD : 2
      MinOfNegativesSD: 1          MaxOfNegativesSD: 1
      NumOfNegativesSD: 1          SumOfNegativesSD: 1
      Sum2NegativesSD : 1
      MinOfPositivesDS: 1          MaxOfPositivesDS: 1
      NumOfPositivesDS: 1          SumOfPositivesDS: 1
      Sum2PositivesDS : 1
      MinOfNegativesDS: 1          MaxOfNegativesDS: 1
      NumOfNegativesDS: 1          SumOfNegativesDS: 1
      Sum2NegativesDS : 1
      JitterAve: 1         JitterSDAve: 1      JitterDSAve: 1
      Interarrival jitterout: 0             Interarrival jitterin: 0
    One Way Values :
      NumOfOW: 0
      OWMinSD : 0          OWMaxSD: 0          OWSumSD: 0
      OWSum2SD: 0          OWAveSD: 0
      OWMinDS : 0          OWMaxDS: 0          OWSumDS: 0
      OWSum2DS: 0          OWAveDS: 0
```

IP SLA 反応としきい値のモニタリングの設定：例

次の例では、IP SLA 反応およびしきい値モニタリングを設定する方法を示します。次の操作を実行できます。

- true または false の条件をアクティブ化する属性の反応を設定します。たとえば、1、5、6です。
- しきい値を受け入れる属性の反応を設定します。
- 追加の threshold type オプションを設定します。
- アクション タイプのロギングまたはトリガーを設定します。

```
configure
ipsla operation 1
  type icmp echo
    timeout 5000
    destination address 223.255.254.254
    frequency 10
    statistics interval 30
    buckets 3
end

configure
ipsla operation 2
  type icmp path-echo
    destination address 223.255.254.254
    frequency 5
end

configure
ipsla reaction operation 1
  react timeout
   action trigger
   threshold type immediate
 exit
exit
  react rtt
   action logging
   threshold lower-limit 4 upper-limit 5
end
```

動作1はタイムアウトの発生をチェックします。適用される場合、動作1はトリガーイベントを生成します。**rtt** キーワードが 5 を超えると、エラーがログに記録されます。

動作1によってトリガーイベントが生成されると、動作2が開始されます。次の例では、**ipsla reaction trigger** コマンドを使用して、反応トリガー動作を設定する方法を示します。

```
configure
ipsla reaction trigger 1 2
end
```

IP SLA MPLS LSP モニタリングの設定：例

次の例では、IP SLA MPLS LSP モニタリングの設定方法を説明します。

```
ipsla
 mpls lsp-monitor
  monitor 1
   type mpls lsp ping
    vrf SANJOSE
    scan interval 300
    scan delete-factor 2
    timeout 10000
    datasize request 256
    lsp selector ipv4 127.0.0.10
    force explicit-null
    reply dscp af
    reply mode router-alert
    ttl 30
    exp 1
    statistics hourly
```

```
    buckets 1
   !
  !
 !
 reaction monitor 1
  react timeout
   action logging
   threshold type immediate
  !
  react connection-loss
   action logging
   threshold type immediate
  !
 !
 schedule monitor 1
  frequency 300
  schedule period 120
  start-time 11:45:00 July 4
 !
!
mpls discovery vpn
 interval 600
!
!
```

LSP パス ディスカバリの設定：例

次の例では、LSP パス ディスカバリの設定方法を説明します。

```
configure
ipsla
 mpls lsp-monitor
  monitor 1
   type mpls lsp ping
    path discover
     path retry 12
     path secondary frequency both 12
```