



オブジェクト トラッキングの設定

ここでは、Cisco IOS XR ネットワークでのオブジェクト トラッキングの設定について説明します。このモジュール内に記載されているコマンドの詳細については、「[その他の関連資料](#)」の項を参照してください。設定タスクを実行する手順の中で出現する可能性のあるその他のコマンドについて記載されたマニュアルを検索するには、トピック「[その他の関連資料](#)」の「[テクニカル ドキュメント](#)」の項を参照してください。

- [オブジェクト トラッキングの設定 \(1 ページ\)](#)
- [オブジェクト トラッキングの実装の前提条件 \(1 ページ\)](#)
- [オブジェクト トラッキングについて \(2 ページ\)](#)
- [オブジェクト トラッキングの実装方法 \(2 ページ\)](#)
- [オブジェクト トラッキングの設定例 \(13 ページ\)](#)

オブジェクト トラッキングの設定

ここでは、Cisco IOS XR ネットワークでのオブジェクト トラッキングの設定について説明します。このモジュール内に記載されているコマンドの詳細については、「[その他の関連資料](#)」の項を参照してください。設定タスクを実行する手順の中で出現する可能性のあるその他のコマンドについて記載されたマニュアルを検索するには、トピック「[その他の関連資料](#)」の「[テクニカル ドキュメント](#)」の項を参照してください。

オブジェクト トラッキングの実装の前提条件

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。

オブジェクトトラッキングについて

オブジェクトトラッキングとは、オブジェクトを追跡して、そのプロパティの変化に基づいて、トラッキング対象オブジェクトとは関係のない別のオブジェクトに対してアクションを実行する仕組みです。

各トラッキング対象オブジェクトは、トラッキングコマンドラインインターフェイス (CLI) で指定された一意の名前で識別されます。Cisco IOS XR が処理し、この名前を使用して特定のオブジェクトを追跡します。

トラッキングプロセスでは、定期的にトラッキング対象オブジェクトをポーリングして、ステータスのアップ、ダウンなどの変化をユーザの指定により即時または時間をおいてレポートします。

リストを使った方法で複数のオブジェクトを追跡することもできます。リストはオブジェクトの組み合わせにブール論理式を使った柔軟なメソッドです。リストでは次の演算を使用します。

- **ブールAND関数**：トラッキング対象リストにブールAND関数を指定した場合、サブセット内に定義された各オブジェクトはアップステートでなければならないため、トラッキング対象オブジェクトもアップステートになります。
- **ブールOR関数**：トラッキング対象リストにブールOR関数を指定した場合、サブセット内に定義されたオブジェクトのうち少なくとも1つがアップステートでなければならないため、トラッキング対象オブジェクトもアップステートであることを意味します。

オブジェクトトラッキングの実装方法

ここでは、さまざまなオブジェクトトラッキングの手順を説明します。

インターフェイスのラインプロトコルステートのトラッキング

インターフェイスのラインプロトコルステートをトラッキングするには、グローバルコンフィギュレーションモードで次の作業を実行します。

インターフェイスのラインプロトコルがアップしている場合は、トラッキング対象オブジェクトはアップ状態と見なされます。

トラッキング対象オブジェクトの設定後、そのステータスがトラッキング対象になっているインターフェイスを関連付けたり、トラッキングオブジェクトがインターフェイスをポーリングしてステータスを取得するまで待機する秒数を指定したりすることができます。

手順

	コマンドまたはアクション	目的
ステップ 1	<code>configure</code>	

	コマンドまたはアクション	目的
ステップ 2	track track-name 例 : RP/0/RP0/cpu 0: router(config)# track track1	トラック コンフィギュレーション モードを開始します。 • <i>track-name</i> : トラッキングの対象となるオブジェクト名を指定します。
ステップ 3	type line-protocol state 例 : RP/0/RP0/cpu 0: router(config-track)# type line-protocol state	インターフェイスのラインプロトコルに基づいてトラッキングを作成します。
ステップ 4	interface type interface-path-id 例 : RP/0/RP0/cpu 0: router(config-track-line-prot)# interface atm 0/2/0/0.1	プロトコルステートをトラッキングするインターフェイスを指定します。 • <i>type</i> : インターフェイスタイプを指定します。詳細については、疑問符 (?) オンラインヘルプ機能を使用します。 • <i>interface-path-id</i> : 物理インターフェイスまたは仮想インターフェイスを識別します。 (注) ルータに現在設定されている可能性があるすべてのインターフェイスのリストを表示するには、 show interfaces コマンドを使用します。 (注) ループバック インターフェイスおよびヌル インターフェイスは、常にアップステートであり、そのためトラッキングできません。
ステップ 5	exit 例 : RP/0/RP0/cpu 0: router(config-track-line-prot)# exit	トラック ラインプロトコル 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	(任意) delay {up seconds down seconds} 例 : RP/0/RP0/cpu 0: router(config-track)# delay up 10	オブジェクトがアップかダウンかのトラッキング間に発生可能な遅延をスケジューリングします。

	コマンドまたはアクション	目的
ステップ7	<p>次のいずれかのコマンドを使用します。</p> <ul style="list-style-type: none"> • end • commit <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track)# end</pre> <p>または</p> <pre>RP/0/RP0/cpu 0: router(config-track)# commit</pre>	<p>設定変更を保存します。</p> <ul style="list-style-type: none"> • end コマンドを実行すると、次に示す変更のコミットを求めるプロンプトが表示されます。 <pre>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> • yes と入力すると、実行コンフィギュレーションファイルに変更が保存され、コンフィギュレーションセッションが終了して、ルータがEXECモードに戻ります。 • no と入力すると、コンフィギュレーションセッションが終了して、ルータがEXECモードに戻ります。変更はコミットされません。 • cancel と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーションセッションは終了せず、設定変更もコミットされません。 • 実行コンフィギュレーションファイルに設定変更を保存し、コンフィギュレーションセッションを継続するには、commit コマンドを使用します。

IP ルートの到達可能性のトラッキング

ホストまたはネットワークがリモートサイトでダウン状態になった場合、ルーティングプロトコルはルータに通知し、ルーティングテーブルはそれに応じて更新されます。ルーティングプロセスは、ルーティングアップデートによってルートの状態が変わった場合にトラッキングプロセスに通知するように設定されます。

ルーティングテーブルエントリがルートに存在し、そのルートがアクセス可能であると、トラッキング対象オブジェクトはアップ状態にあると見なされます。

手順

	コマンドまたはアクション	目的
ステップ 1	configure	
ステップ 2	track track-name 例： RP/0/RP0/cpu 0: router(config)# track track1	トラック コンフィギュレーション モードを開始します。 • <i>track-name</i> : トラッキングの対象となるオブジェクト名を指定します。
ステップ 3	type route reachability 例： RP/0/RP0/cpu 0: router(config-track)# type route reachability vrf internet	ルーティングアップデートによってルートの状態が変わった場合にトラッキングプロセスに通知するようにルーティングプロセスを設定します。
ステップ 4	次のいずれかのコマンドを使用します。 • vrf vrf-table-name • route ipv4 IP-prefix/mask 例： RP/0/RP0/cpu 0: router(config-track-route)# vrf vrf-table-4 または RP/0/RP0/cpu 0: router(config-track-route)# route ipv4 10.56.8.10/16	トラッキングする IP ルートのタイプを設定します。これは、ルータのタイプによって次のいずれかで構成可能です。 • <i>vrf-table-name</i> : VRF テーブル名。 • <i>IP-prefix/mask</i> : ネットワークとサブネットマスクからなる IP プレフィックス (例: 10.56.8.10/16)。
ステップ 5	exit 例： RP/0/RP0/cpu 0: router(config-track-line-prot)# exit	トラック ラインプロトコル 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	(任意) delay {up seconds down seconds} 例： RP/0/RP0/cpu 0: router(config-track)# delay up 10	オブジェクトがアップかダウンかのトラッキング間に発生可能な遅延をスケジューリングします。
ステップ 7	commit	

オブジェクトリストに基づくトラッキングの設定

グローバル コンフィギュレーション モードでこのタスクを実行し、ブール式を使用してリストの状態を判断して、トラッキング対象オブジェクトリスト（ここではインターフェイスまたはプレフィックスのリスト）を作成します。

トラッキング対象リストには1つまたは複数のオブジェクトが含まれます。ブール式では、AND または OR 演算子を使用して2種類の演算を実行できます。たとえば、AND 演算子を使用して2つのインターフェイスをトラッキングする場合、アップは両方のインターフェイスがアップ状態であることを意味し、ダウンはいずれか一方のインターフェイスがダウン状態であることを意味します。



(注) トラッキング対象リストにオブジェクトを追加するには、そのオブジェクトが存在している必要があります。

NOT 演算子は、1つまたは複数のオブジェクトに指定し、そのオブジェクトの状態を否定します。

トラッキング対象オブジェクトを設定したら、状態をトラッキングするインターフェイスを関連付ける必要があります。オプションとして、トラッキングオブジェクトがインターフェイスをポーリングしてその状態を取得するまでの待機時間を秒数で指定できます。

手順

	コマンドまたはアクション	目的
ステップ 1	configure	
ステップ 2	track track-name 例： RP/0/RP0/cpu 0: router(config)# track track1	トラック コンフィギュレーション モードを開始します。 • track-name : トラッキングの対象となるオブジェクト名を指定します。
ステップ 3	type list boolean { and or } 例： RP/0/RP0/cpu 0: router(config-track-list)# type list boolean and	ブール リスト オブジェクトを設定し、トラッキング リスト コンフィギュレーション モードを開始します。 • boolean : トラッキング対象リストのステータスがブール式に基づいて決まることを指定します。 • and : リストについて、すべてのオブジェクトがアップの場合はアップ、ダウンのオブジェクトが1つ以上ある場合はダウンになるように指定します。たとえば2つのインターフェイスをトラッキングする場合、

	コマンドまたはアクション	目的
		<p>アップは両方のインターフェイスがアップ状態であることを意味し、ダウンはいずれか一方のインターフェイスがダウン状態であることを意味します。</p> <ul style="list-style-type: none"> • or : 少なくとも1つのオブジェクトがアップであればリストがアップになるように指定します。たとえば2つのインターフェイスをトラッキングする場合、アップはいずれか一方のインターフェイスがアップ状態であることを意味し、ダウンは両方のインターフェイスがダウン状態であることを意味します。
ステップ4	<p>object object-name [not]</p> <p>例 :</p> <pre>RP/0/RP0/cpu 0: router(config-track-list)# object 3 not</pre>	<p>リストによるトラッキングの対象となるオブジェクトを指定します。</p> <ul style="list-style-type: none"> • <i>object-name</i> : トラッキングするオブジェクトの名前。 • not : オブジェクトの状態を否定します。
ステップ5	<p>exit</p> <p>例 :</p> <pre>RP/0/RP0/cpu 0: router(config-track-line-prot)# exit</pre>	<p>トラック ラインプロトコル コンフィギュレーション モードを終了します。</p>
ステップ6	<p>(任意) delay {up seconds down seconds}</p> <p>例 :</p> <pre>RP/0/RP0/cpu 0: router(config-track)# delay up 10</pre>	<p>オブジェクトがアップかダウンかのトラッキング間に発生可能な遅延をスケジューリングします。</p>
ステップ7	<p>次のいずれかのコマンドを使用します。</p> <ul style="list-style-type: none"> • end • commit <p>例 :</p> <pre>RP/0/RP0/cpu 0: router(config-track)# end</pre> <p>または</p>	<p>設定変更を保存します。</p> <ul style="list-style-type: none"> • end コマンドを実行すると、次に示す変更のコミットを求めるプロンプトが表示されます。 <pre>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</pre>

	コマンドまたはアクション	目的
	RP/0/RP0/cpu 0: router(config-track)# commit	<ul style="list-style-type: none"> • yes と入力すると、実行コンフィギュレーションファイルに変更が保存され、コンフィギュレーションセッションが終了して、ルータがEXECモードに戻ります。 • no と入力すると、コンフィギュレーションセッションが終了して、ルータがEXECモードに戻ります。変更はコミットされません。 • cancel と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーションセッションは終了せず、設定変更もコミットされません。 • 実行コンフィギュレーションファイルに設定変更を保存し、コンフィギュレーションセッションを継続するには、commit コマンドを使用します。

オブジェクトリストに基づくトラッキングの設定：しきい値の割合

グローバル コンフィギュレーション モードでこのタスクを実行し、しきい値の割合を使用してリストの状態を判断して、トラッキング対象オブジェクトリスト（ここではインターフェイスまたはプレフィックスのリスト）を作成します。

手順

	コマンドまたはアクション	目的
ステップ 1	configure	
ステップ 2	track track-name 例： RP/0/RP0/cpu 0: router(config)# track track1	トラック コンフィギュレーション モードを開始します。 <ul style="list-style-type: none"> • track-name：トラッキングの対象となるオブジェクト名を指定します。

	コマンドまたはアクション	目的
ステップ 3	<p>type list threshold percentage</p> <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track-list)# type list threshold percentage</pre>	<p>トラッキングのタイプにしきい値の割合リストを設定します。</p>
ステップ 4	<p>object object-name</p> <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 1 RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 2 RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 3 RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 4</pre>	<p>トラック タイプ track1 のメンバーに object 1、object 2、object 3 および object 4 を設定します。</p>
ステップ 5	<p>threshold percentage up percentage down percentage</p> <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track-list-threshold)# threshold percentage up 50 down 33</pre>	<p>リストがそれぞれアップ状態またはダウン状態であると見なされるために、アップ状態またはダウン状態である必要があるオブジェクトの割合を設定します。</p> <p>たとえば、object 1、object 2、および object 3 がアップ状態にあり、object 4 がダウン状態にある場合、リストはアップ状態にあると見なされます。</p>
ステップ 6	<p>次のいずれかのコマンドを使用します。</p> <ul style="list-style-type: none"> • end • commit <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track)# end</pre> <p>または</p> <pre>RP/0/RP0/cpu 0: router(config-track)# commit</pre>	<p>設定変更を保存します。</p> <ul style="list-style-type: none"> • end コマンドを実行すると、次に示す変更のコミットを求めるプロンプトが表示されます。 <pre>Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:</pre> <ul style="list-style-type: none"> • yes と入力すると、実行コンフィギュレーションファイルに変更が保存され、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。

	コマンドまたはアクション	目的
		<ul style="list-style-type: none"> • no と入力すると、コンフィギュレーションセッションが終了して、ルータがEXECモードに戻ります。変更はコミットされません。 • cancel と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュレーションセッションは終了せず、設定変更もコミットされません。 • 実行コンフィギュレーションファイルに設定変更を保存し、コンフィギュレーションセッションを継続するには、commit コマンドを使用します。

オブジェクトリストに基づくトラッキングの設定：しきい値の重み

グローバル コンフィギュレーション モードでこのタスクを実行し、しきい値の重みを使用してリストの状態を判断して、トラッキング対象オブジェクトリスト（ここではインターフェイスまたはプレフィックスのリスト）を作成します。

手順

	コマンドまたはアクション	目的
ステップ 1	configure	
ステップ 2	track track-name 例： <pre>RP/0/RP0/cpu 0: router(config)# track track1</pre>	トラック コンフィギュレーション モードを開始します。 <ul style="list-style-type: none"> • track-name：トラッキングの対象となるオブジェクト名を指定します。
ステップ 3	type list threshold weight 例： <pre>RP/0/RP0/cpu 0: router(config-track-list)# type list threshold weight</pre>	トラッキングのタイプにしきい値の重みリストを設定します。

	コマンドまたはアクション	目的
ステップ 4	<p>object <i>object-name</i> weight <i>weight</i></p> <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 1 weight 10 RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 2 weight 5 RP/0/RP0/cpu 0: router(config-track-list-threshold)# object 3 weight 3</pre>	<p>track t1 のメンバーに object 1、object 2 および object 3 を設定し、それぞれに重み 10、5 および 3 を設定します。</p>
ステップ 5	<p>threshold weight up <i>weight</i> down <i>weight</i></p> <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track-list-threshold)# threshold weight up 10 down 5</pre>	<p>リストがそれぞれアップ状態またはダウン状態であると見なされるために、アップ状態またはダウン状態である必要があるオブジェクトの重みの範囲を設定します。この例では、object 1 および 2 がアップ状態にあり、累積の重みは 15 である（10～5 の範囲内ではない）ため、リストはダウン状態と見なされます。</p>
ステップ 6	<p>次のいずれかのコマンドを使用します。</p> <ul style="list-style-type: none"> • end • commit <p>例：</p> <pre>RP/0/RP0/cpu 0: router(config-track)# end</pre> <p>または</p> <pre>RP/0/RP0/cpu 0: router(config-track)# commit</pre>	<p>設定変更を保存します。</p> <ul style="list-style-type: none"> • end コマンドを実行すると、次に示す変更のコミットを求めるプロンプトが表示されます。 Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: • yes と入力すると、実行コンフィギュレーションファイルに変更が保存され、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。 • no と入力すると、コンフィギュレーションセッションが終了して、ルータが EXEC モードに戻ります。変更はコミットされません。 • cancel と入力すると、現在のコンフィギュレーションセッションが継続します。コンフィギュ

	コマンドまたはアクション	目的
		<p>レーションセッションは終了せず、設定変更もコミットされません。</p> <ul style="list-style-type: none"> • 実行コンフィギュレーションファイルに設定変更を保存し、コンフィギュレーションセッションを継続するには、commit コマンドを使用します。

IPSLA の到達可能性のトラッキング

IP サービス レベル契約 (SLA) 動作の戻りコードのトラッキングをイネーブルにするには、このタスクを使用します。

手順

	コマンドまたはアクション	目的
ステップ 1	configure 例： RP/0/RP0/cpu 0: router# configure	グローバル コンフィギュレーションモードを開始します。
ステップ 2	track track-name 例： RP/0/RP0/cpu 0: router(config)# track t1	トラック コンフィギュレーションモードを開始します。
ステップ 3	type rtr ipsla-no reachability 例： RP/0/RP0/cpu 0: router(config-track)# type rtr 100 reachability	到達可能性をトラッキングする IP SLA 動作 ID を指定します。 <i>ipsla-no</i> の有効値は、1 ~ 2048 の範囲です。
ステップ 4	commit	

IPSLA トラッキングの設定：例

次に、IPSLA のトラッキング設定の例を示します。

```
RP/0/RP0/cpu 0: router(config)# track track1
RP/0/RP0/cpu 0: router(config-track)# type rtr 1 reachability
RP/0/RP0/cpu 0: router(config-track)# delay up 5
RP/0/RP0/cpu 0: router(config-track)# delay down 10
```

オブジェクトトラッキングの設定例

インターフェイスがアップ状態かダウン状態かのトラッキング：実行コンフィギュレーションの例

```
track connection100
  type list boolean and
  object object3 not
  delay up 10
  !
interface service-ipsec 23
  line-protocol track connection100
  !
```

インターフェイスのラインプロトコルステートのトラッキング：実行コンフィギュレーションの例

この例では、トラフィックはインターフェイス `service-ipsec1` から到着し、インターフェイス `TenGigE0/11/0/3` を経由して終了します。

```
track IPsec1
  type line-protocol state
  interface TenGigE0/11/0/3
  !
interface service-ipsec 1
  ipv4 address 70.0.0.1 255.255.255.0
  profile vrfl_profile_ipsec
  line-protocol track IPsec1
  tunnel source 80.0.0.1
  tunnel destination 80.0.0.2
  service-location preferred-active 0/0/1
  !
```

次に、前述の例を実行した後の `show track` コマンドの出力例を示します。

```
RP/0/RP0/cpu 0: router# show run track

Track IPsec1
Interface GigabitEthernet0_0_0_3 line-protocol
!
  Line protocol is UP
  1 change, last change 10:37:32 UTC Thu Sep 20 2007
  Tracked by:
  service-ipsec1
  !
```

IP ルートの到達可能性のトラッキング：実行コンフィギュレーションの例

この例では、インターフェイス `service-ipsec1` から到着したトラフィックの宛先がネットワーク `7.0.0.0/24` にあります。このトラッキング手順は、ルーティングプロトコルプレフィックスの状態に従い、ルーティングテーブルに変更があったときに信号を送ります。

```
track PREFIX1
  type route reachability
    route ipv4 7.0.0.0/24
    !
  interface service-ipsec 1
  vrf 1
  ipv4 address 70.0.0.2 255.255.255.0
  profile vrf_1_ipsec
  line-protocol track PREFIX1
  tunnel source 80.0.0.2
  tunnel destination 80.0.0.1
  service-location preferred-active 0/2/0
```

オブジェクトのリストに基づいたトラックの構築：実行コンフィギュレーションの例

この例では、インターフェイス `service-ipsec1` から到着するトラフィックが、インターフェイス `TenGigE0/11/0/3` およびインターフェイス `ATM0/2/0/0.1` を介して終了します。トラフィックの宛先はネットワーク `7.0.0.0/24` です。

いずれかのインターフェイスまたはリモートネットワークがダウンした場合は、トラフィックフローが停止される必要があります。これを行うには、ブール AND 式を使用します。

```
track C1
  type route reachability
    route ipv4 3.3.3.3/32
    !
  !
track C2
  type route reachability
    route ipv4 1.2.3.4/32
    !
  !
track C3
  type route reachability
    route ipv4 10.0.20.2/32
    !
  !
track C4
  type route reachability
    route ipv4 10.0.20.0/24
    !
  !
track OBJ
  type list boolean and
    object C1
    object C2
  !
```

```
!  
track OBJ2  
  type list boolean or  
  object C1  
  object C2  
!
```

IPSLA ベースのオブジェクトトラッキングの設定：コンフィギュレーションの例

次に、ACL と IPSLA 設定を含む IPSLA ベースのオブジェクトトラッキングの設定例を示します。

ACL の設定：

```
RP/0/RP0/cpu 0: router(config)# ipv4 access-list abf-track  
RP/0/RP0/cpu 0: router(config-ipv4-acl)# 10 permit any nexthop track track1 1.2.3.4
```

オブジェクトトラッキングの設定：

```
RP/0/RP0/cpu 0: router(config)# track track1  
RP/0/RP0/cpu 0: router(config-track)# type rtr 1 reachability  
RP/0/RP0/cpu 0: router(config-track)# delay up 5  
RP/0/RP0/cpu 0: router(config-track)# delay down 10
```

IPSLA の設定：

```
RP/0/RP0/cpu 0: router(config)# ipsla  
RP/0/RP0/cpu 0: router(config-ipsla)# operation 1  
RP/0/RP0/cpu 0: router(config-ipsla-op)# type icmp echo  
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# source address 2.3.4.5  
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# destination address 1.2.3.4  
RP/0/RP0/cpu 0: router(config-ipsla-icmp-echo)# frequency 60  
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)# schedule operation 1  
RP/0/RP0/cpu 0: router(config-ipsla-sched)# start-time now  
RP/0/RP0/cpu 0: router(config-ipsla-sched)# life forever
```

