管理設定ガイド

最終更新：2026 年 7 月 7 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

目 次

第 1 章 組み込みパケット キャプチャ 1

組み込みパケットキャプチャの概要 1

組み込みパケットキャプチャ 2

組み込みパケットキャプチャ設定の前提条件 2

組み込みパケットキャプチャの設定の制約事項 3

Wireshark とは何ですか？ 4

Wireshark の仕組み 5

キャプチャされたパケットのストレージ 7

Wireshark の機能 9

Wireshark 設定のガイドライン 10

Wireshark 設定の前提条件 13

Wireshark 設定の制約事項 13

パケットキャプチャの設定方法 15

パケット データ キャプチャの管理 15

キャプチャされたデータのモニタリングとメンテナンス 16

Wireshark の設定方法 17

キャプチャ ポイントの定義 18

キャプチャ ポイント パラメータの追加またはモニタリング 22

キャプチャ ポイント パラメータの削除 25

キャプチャ ポイントの削除 26

キャプチャ ポイントをアクティブまたは非アクティブにする 28

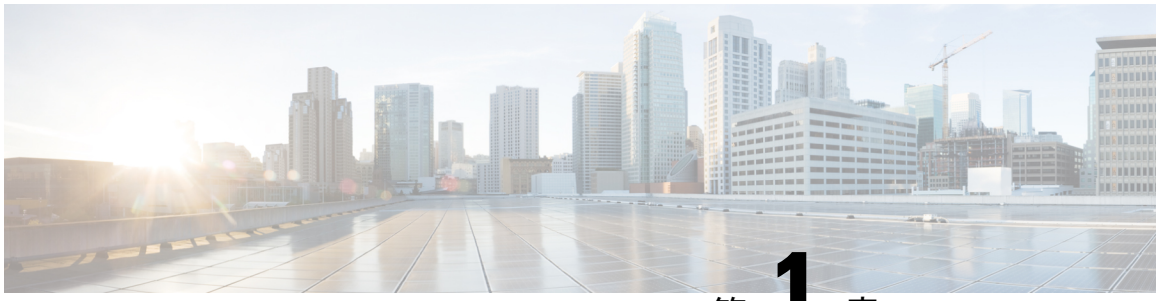
キャプチャ ポイント バッファのクリア 31

第 2 章 簡易ネットワーク管理プロトコル 33

SNMP の概要 33
SNMP とは何ですか? 33
SNMP バージョン 34
SNMP マネージャ機能 36
SNMP エージェント機能 37
SNMP MIB 変数アクセス 37
SNMP フラッシュ MIB 38
SNMP 通知 38
SNMP ifIndex MIB オブジェクト値 40
SNMP ENTITY-MIB 識別子 41
SNMP および Syslog Over IPv6 41
SNMP UDP ポート 41
SNMP のデフォルト設定 42
SNMP の制約事項 42
SNMP の設定方法 43
SNMP 設定時の注意事項 43
SNMP グループおよびユーザの設定 44
SNMP UDP ポートの開閉 49
エージェント コンタクトおよびロケーションの設定 50
SNMP を通して使用する TFTP サーバの制限 52
SNMP エージェントのディセーブル化 53
SNMP ステータスのモニタリング 54
SNMP の例 55

第 3 章 スイッチド ポート アナライザ 57
SPAN の概要 57
SPAN の仕組み 58
SPAN の概念および用語 59
SPAN と他の機能の相互作用 64
SPAN とデバイススタック 65
SPAN の制約事項 65

SPAN の設定方法 66
ローカル SPAN セッションの作成 66
SPAN のコンフィギュレーション例 69



第 1 章

組み込みパケット キャプチャ

- 組み込みパケットキャプチャの概要 （1 ページ）
- 組み込みパケットキャプチャ （2 ページ）
- Wireshark とは何ですか？ （4 ページ）
- パケットキャプチャの設定方法 （15 ページ）
- Wireshark の設定方法 （17 ページ）

組み込みパケットキャプチャの概要

パケットキャプチャは、ネットワークを通過するデータパケットを傍受して記録するために使用される、基本的なネットワーク診断技術です。これらのパケットには、プロトコルヘッダーやペイロードデータなど、デバイス間で交換される未加工の情報が含まれています。ネットワークエンジニアは、パケットをキャプチャして分析することにより、ネットワークの動作を可視化し、接続の問題、パフォーマンスのボトルネック、セキュリティインシデントのトラブルシュートを可能にします。

パケット データ キャプチャ

これは、分析のためにバッファに保存されるデータパケットをキャプチャするプロセスです。一意の名前を割り当て、特定のパラメータを定義することによって、パケットキャプチャを作成できます。

次の操作を実行できます。

- インターフェイスでのキャプチャのアクティブ化。
- キャプチャポイントへのアクセス コントロール リスト（ACL）やクラスマップの適用。
- 不要になった場合のキャプチャの破棄。
- サイズやタイプなどのバッファ ストレージ パラメータの指定。バッファサイズは 1MB ～ 100MB の範囲です。デフォルトのバッファタイプは線形であり、代替として循環バッファリングが利用可能です。
- キャプチャされたトラフィックをフィルタ処理するために、プロトコル、IP アドレス、またはポート番号に基づいて一致基準を定義します。

パケットキャプチャは、複雑なネットワークのやり取りを理解し、プロトコル運用を検証するのに必須です。従来、パケットキャプチャは、ネットワークタップやスパン/ミラーポートなどの外部デバイスを使用して実行され、分析ツールへのトラフィックを複製していました。

パケットキャプチャは、ローカルで使用することも、組み込みパケットキャプチャ（EPC）やWiresharkなどのツールを使用してオフライン分析用にエクスポートすることもできます。EPCとは、外部ハードウェアを必要とせずにデバイスでパケットを直接キャプチャおよびフィルタリングできるシスコのオンデバイスパケットキャプチャ機能のことを指します。一方、Wiresharkは、ネットワークトラフィックの詳細な検査と障害対応に広範に使用されている強力なオープンソース パケット アナライザです。

組み込みパケットキャプチャ

組み込みパケットキャプチャ（EPC）は、外部ハードウェアまたはポートミラーリングを必要とせずに、ネットワークデバイス上で直接パケットキャプチャを有効にするシスコの機能です。EPCは、デバイスの内部リソースを利用して、指定されたインターフェイスでトラフィックをキャプチャし、オンボードバッファまたはファイルに一時的にデータを保存します。その後、キャプチャしたパケットを分析用にエクスポートします。

EPC には以下の利点があります。

オンデバイスキャプチャ：追加のハードウェアは必要ありません。

詳細なフィルタリング：特定のトラフィックタイプまたはフローをキャプチャします。

低影響：デバイスのパフォーマンス低下を最小限に抑えるための効率的なリソースを使用します。

柔軟性：物理インターフェイスまたは他の論理インターフェイスでキャプチャします。

組み込みパケット キャプチャの利点

- このデバイスは、MACフィルタを使用して、または任意のMACアドレスに一致させて、IPv4 および IPv6 パケットだけでなく、非 IP パケットもキャプチャできます。
- パケット キャプチャ ポイントを有効にする拡張可能なインフラストラクチャキャプチャポイントは、パケットをキャプチャしてバッファに関連付けるために使用されるトラフィック中継の場所です。
- パケットキャプチャは、パケットキャプチャファイル（PCAP）形式でエクスポートできます。この形式は、外部ツールを使用した分析に適しています。
- さまざまな詳細レベルでキャプチャされたデータ パケットをデコードする方法。

組み込みパケットキャプチャ設定の前提条件

組み込みパケット キャプチャ（EPC）のソフトウェア サブシステムは、その動作で CPU とメモリ リソースを消費します。さまざまなタイプの操作を行うために十分なシステム リソース

を準備する必要があります。次の表は、システムリソースを使用するためのガイドラインを示しています。

表 *1 : EPC* サブシステムのシステム要件

システム リソース	要件
ハードウェア	CPU 利用率の要件は、プラットフォームによって異なります。
メモリ	DRAM はパケットバッファを保存します。パケットバッファのサイズは、ユーザーが指定します。
ディスク容量	パケットは外部のデバイスにエクスポートできます。フラッシュディスクでの中間保管は必要ありません。

組み込みパケットキャプチャの設定の制約事項

次の制約事項が組み込みパケットキャプチャ（EPC）に適用されます。

- VRF、管理ポート、またはプライベート VLAN を接続ポイントとして使用することはできません。
- シャットダウン状態の VLAN インターフェイスは EPC をサポートしていません。
- ユーザーがスイッチポートからルーテッドポート（レイヤ 2 からレイヤ 3）へ、またはその逆へインターフェイスを変更した場合、インターフェイスが再び起動したときに、そのキャプチャポイントを削除し、新しいファイルを作成する必要があります。キャプチャポイントの停止/開始が機能しません。
- インターフェイスの出力方向にキャプチャされたパケットは、デバイスの書き換えによって行われた変更（TTL、VLAN タグ CoS、チェックサム、および MAC アドレス、DSCP、プレシデント、UP など）が反映されないこともあります。
- パケットキャプチャの最小設定可能期間は 1 秒ですが、パケットキャプチャは少なくとも 2 秒間機能します。
- キャプチャがすでにアクティブである、または開始されている場合、キャプチャポイントパラメータを変更することはできません。
- EPC は、入力のマルチキャストパケットのみをキャプチャし、出力の複製パケットはキャプチャしません。
- 入力および出力の両方のパケットの書き換え情報はキャプチャされません。
- CPU が挿入されたパケットはコントロール プレーン パケットと見なされ、これらのタイプのパケットはインターフェイス出力キャプチャでキャプチャされません。
- コントロールプレーンパケットは、レート制限とパフォーマンスへの影響はありません。コントロール プレーン パケット キャプチャを制限するには、フィルタを使用してください。

- DNA Advantage は、ワイヤレスアクセスポイントの制御とプロビジョニング（CAPWAP）などのプロトコルの複合化をサポートしています。
- 最大 8 つのキャプチャポイントを定義できますが、一度にアクティブにできるのは 1 つだけです。もう一方を開始する前に、一方を停止してください。
- MAC フィルタは、MAC アドレスに一致しても IP パケットをキャプチャしません。これは、すべてのインターフェイス（レイヤ 2 スイッチポート、レイヤ 3 ルーテッドポート）に適用されます。
- MAC ACL は、ARP などの非 IP パケットだけに使用されます。レイヤ 3 ポートまたは SVI ではサポートされません。
- MAC フィルタは、レイヤ 3 インターフェイスとレイヤ 2 パケット（ARP）をキャプチャすることはできません。
- VACL は IPv6 ベースの ACL をサポートしません。
- EPC は、MPLS パケットの基礎となるルーティングプロトコルに基づいてキャプチャすることはできません。
- EPC は、Locator/ID Separation Protocol（LISP）インターフェイスおよびトンネルインターフェイスではサポートされていません。
- EPC は、Ethernet-over-MPLS（EoMPLS）ではサポートされていません。
- Network Based Application Recognition（NBAR）と MAC スタイルのクラスマップは、サポートされていません。

Wireshark とは何ですか？

Wireshark は、ネットワークキャプチャの詳細な調査に使用されるオープンソースのパケット分析ツールです。EPC はスイッチ自体でトラフィックをキャプチャしますが、Wireshark は PC またはワークステーションでそのデータを分析する環境を提供します。

一般的なワークフローは次のとおりです。

1. **キャプチャ**：EPC は、設定されたフィルタとインターフェイスに従ってデバイスでパケットを収集します。
2. **エクスポート**：キャプチャされたパケットは、プロトコルを使用して `.pcap` ファイルとしてスイッチからエクスポートされます。
3. **分析**：これらのキャプチャファイルは Wireshark で開き、プロトコルの復号、フィルタ、および可視化を適用して、ネットワークの問題を特定したり、動作を検証したりできます。

この統合プロセスにより、シスコの組み込みキャプチャ機能と Wireshark の分析ツールが統合され、専用のキャプチャアプライアンスを必要とせずに包括的なネットワークの障害対応が可能になります。

Wireshark の仕組み

Wireshark は、.pcap と呼ばれる既知の形式を使用してファイルへパケットをダンプし、個々のインターフェイスに対して適用されイネーブルになります。EXEC モードでインターフェイスを指定し、フィルタおよび他のパラメータも指定します。Wireshark アプリケーションは、**start** コマンドを入力した場合にだけ適用され、Wireshark が自動または手動でキャプチャを停止した場合にだけ削除されます。

次のセクションでは、Wireshark に関連するコンポーネントについて説明します。

キャプチャ ポイント

キャプチャ ポイントとは、Wireshark 機能の一元的なポリシー定義です。特定の Wireshark インスタンスの特性、例えばキャプチャするパケット、その送信元、キャプチャされたパケットに対して実行するアクション、停止条件などを概説します。キャプチャポイントは作成後に変更可能ですが、**start** コマンドを使用して明示的にアクティブ化しない限り、アクティブにはなりません。このプロセスは、キャプチャポイントのアクティブ化または開始と呼ばれます。キャプチャポイントは名前で識別され、手動または自動で非アクティブ化または停止することができます。

複数のキャプチャ ポイントを定義できますが、一度にアクティブにできるのは1つだけです。1 つ開始するには 1 つ停止する必要があります。

スタック構成のシステムの場合、キャプチャ ポイントはアクティブなメンバーによりアクティブ化されます。スイッチオーバーにより、アクティブなパケット キャプチャ セッションが終了するため、セッションを再開する必要があります。

接続ポイント

接続ポイントは、キャプチャ ポイントに関連付けられた論理パケットのプロセス パスのポイントです。接続ポイントはキャプチャポイントの属性であり、キャプチャポイントフィルタに対してテストされます。

フィルタに一致するパケットがコピーされ、関連する Wireshark インスタンスに送信されます。特定のキャプチャポイントは複数の接続ポイントに関連付けることができますが、異なるタイプの接続ポイントの混在に制限があります。一部の制限は、異なるタイプの添付ポイントを指定すると適用されます。接続ポイントは、常に双方向であるレイヤ 2 VLAN の接続ポイントを除き、方向性あり（入力/出力/両方）です。

スタック型システムの場合では、すべてのスタック メンバの接続ポイントに有効です。EPC は定義されたすべての接続ポイントからパケットをキャプチャします。ただし、これらのパケットはアクティブ メンバーでのみに処理されます。

フィルタ

フィルタは、キャプチャポイントの接続ポイントを通過するトラフィックのサブセットを識別して制限するキャプチャポイントの属性です。これらはコピーされ、Wireshark にパスされます。Wireshark では、パケットが接続ポイントを通過する場合、パケットと、キャプチャポイントに関連付けられているすべてのフィルタが表示されます。

キャプチャ ポイントには以下のタイプのフィルタがあります。

- コア システム フィルタ：コア システム フィルタはハードウェアによって適用され、一致基準はハードウェアによって制限されます。このフィルタは、ハードウェア転送トラフィックを Wireshark の目的でソフトウェアにコピーするかどうかを決定します。
- キャプチャフィルタ：Wireshark ではキャプチャフィルタが適用されます。一致基準は、コア システム フィルタによってサポートされるものよりも詳細に表示されます。コアフィルタを通過したものの、キャプチャフィルタを通過しなかったパケットはコピーされます。それらは CPU/ソフトウェアに送信されますが、Wireshark プロセスによって破棄されます。キャプチャ フィルタの構文は、表示フィルタの構文と同じです。
- 表示フィルタ：Wireshark では表示フィルタが適用されます。その一致基準は、キャプチャフィルタの基準に似ています。表示フィルタに失敗したパケットは表示されません。

（注） Wireshark はキャプチャフィルタの構文を使用しません。

コア システム フィルタ

クラス マップまたは ACL を使用して、または CLI を使用して明示的にコア システム フィルタの一致基準を指定できます。

（注） CAPWAP を接続ポイントとして指定すると、コア システム フィルタは使用されません。

一部のインストールでは、デバイス設定を変更する権限を取得する際、承認プロセスが長くなることで大幅な遅延が発生する可能性があります。これにより、ネットワーク管理者の機能がトラフィックの監視および分析に制限される場合があります。この状況に対処するため、Wireshark は、EXEC モード CLI から、コア システム フィルタ一致基準の明示的な仕様をサポートします。この対処方法の欠点は、指定できる一致基準が、クラスマップがサポートする対象の限定的なサブセットである（MAC、IP送信元アドレスおよび宛先アドレス、イーサネット タイプ、IP プロトコル、および TCP/UDP の発信元および宛先ポートなど）ことです。

コンフィギュレーション モードを使用する場合は ACL を定義するか、クラス マップでそこへキャプチャ ポイントを参照させることができます。明示的かつ ACL ベースの一致基準がクラス マップとポリシー マップの作成に内部的に使用されます。

ACL およびクラスマップの設定はシステムの一部であり、Wireshark 機能の一部ではありません。

表示フィルタ

表示フィルタを使用すると、.pcap ファイルからデコードして表示するときに表示するパケットの集合をさらに絞り込むように Wireshark に指示できます。

アクション

ライブトラフィックまたは既存の .pcap ファイルで Wireshark を呼び出すことができます。ライブトラフィックに対して起動されたとき、その表示フィルタを通過するパケットに対して次の 4 種類の処理を実行できます。

- デコード、分析、保存のためにメモリ内バッファへキャプチャ
- .pcap ファイルへ保存
- デコードおよび表示
- 保存および表示

デコードおよび表示アクションは、.pcap ファイルで呼び出された場合にのみ適用されます。

デフォルトの Wireshark の設定

次の表は、デフォルトの Wireshark の設定を示しています。

機能	デフォルト設定
持続時間	制限なし
パケット	制限なし
パケット長	制限なし（フル パケット）
ファイル サイズ	制限なし
リング ファイル ストレージ	なし
バッファのストレージ モード	直線

キャプチャされたパケットのストレージ

キャプチャ パケットのメモリ内のバッファへのストレージ

パケットをメモリのキャプチャバッファに保存できます。パケットは、後続の複合化、分析、または .pcap ファイルへの保存に使用できます。

キャプチャ バッファは線形モードまたは循環モードを選択できます。線形モードでは、バッファが上限に達すると、新しいパケットが廃棄されます。循環モードでは、バッファが上限に達すると、新しいパケットを格納するためにより古いほうのパケットが廃棄されます。必要に応じてバッファをクリアすることもできますが、このモードは、ネットワークトラフィックのデバッグに使用されます。ただし、これを削除せずに、バッファのコンテンツをクリアすることはできません。これを有効にするためには、現行のキャプチャを停止し、キャプチャをもう一度再起動します。

（注） パケットをバッファ内に保存する複数のキャプチャがある場合、メモリ ロスを避けるため、新しいキャプチャを開始する前にバッファをクリアしてください。

.pcap ファイルにキャプチャされたパケットのストレージ

Wireshark がスタック内のスイッチで使用される場合は、パケットキャプチャをアクティブスイッチに接続されたフラッシュまたは USB フラッシュデバイスにのみ保存できます。

たとえば、flash1 がアクティブなスイッチに接続されており、flash2 がセカンダリ スイッチに接続されている場合、flash1 にのみパケットキャプチャを保存できます。

アクティブ スイッチに接続されたフラッシュまたは USB フラッシュ デバイス以外のデバイスにパケット キャプチャを保存しようとすると、エラーが発生する場合があります。

Wireshark は .pcap ファイルにキャプチャされたパケットを保存できます。キャプチャ ファイルは次のストレージ デバイスに配置可能です。

- デバイス オンボード フラッシュ ストレージ（flash:）
- USB ドライブ（usbflash0:）

（注） サポートされていないデバイスまたはアクティブなスイッチに接続されていないデバイスにパケット キャプチャを保存しようとするとエラーが発生する可能性があります。

パケットのデコードおよび表示

Wireshark はコンソールにパケットをデコードして表示できます。この機能は、ライブ トラフィックに適用されるキャプチャ ポイントと前の既存 .pcap ファイルに適用されるキャプチャ ポイントで使用可能です。

（注） パケットをデコードして表示すると、CPU への負荷が高くなる場合があります。

Wireshark は、幅広い種類のパケット形式に対してパケット詳細をデコードおよび表示できます。詳細は、**monitor capture name start** コマンドを以下のキーワードオプション付きで入力することにより表示されます。これにより、表示およびデコードモードが開始します。

- 要約：パケット（デフォルト）ごとに 1 行を表示します。
- 詳細：プロトコルがサポートされているすべてのパケットのすべてのフィールドをデコードして表示します。詳細モードでは、他の 2 種類のモードよりも多くの CPU が必要です。
- (hexadecimal) dump：パケット データの 16 進ダンプおよび各パケットの印刷可能文字としてパケットごとに 1 行表示します。

capture コマンドをデコードおよび表示オプション付きで入力すると、Wireshark 出力が Cisco IOS に返され、変更なしでコンソールに表示されます。

ライブ トラフィックの表示

Wireshark はコア システムからパケットのコピーを受信します。Wireshark は、表示フィルタを適用して、不要なパケットを破棄し、残りのパケットをデコードおよび表示します。

.pcap ファイルの表示

Wireshark は、以前に保存された .pcap ファイルからのパケットをデコードして表示し、選択的にパケットを表示するように表示フィルタに指示できます。

パケットのストレージおよび表示

機能的には、このモードは以前の 2 種類のモードの組み合わせです。Wireshark は指定された .pcap ファイルにパケットを保存し、これらをコンソールにデコードおよび表示します。ここではコアフィルタだけが該当します。

Wireshark の機能

- ポートセキュリティと Wireshark を入力キャプチャに適用した場合、ポートセキュリティによってドロップされたパケットは Wireshark によって引き続きキャプチャされます。入力キャプチャにポートセキュリティを適用し、出力キャプチャに Wireshark を適用した場合、ポートセキュリティによってドロップされたパケットは Wireshark によってキャプチャされません。
- Wireshark は、Dynamic ARP Inspection（DAI）によってドロップされたパケットをキャプチャしません。
- STP ブロックステートにあるポートを接続ポイントとして使用し、コアフィルタが一致する場合、パケットがスイッチによってドロップされても、Wireshark はポートに着信するパケットをキャプチャします。
- 分類ベースのセキュリティ機能：入力分類ベースのセキュリティ機能によってドロップされたパケット（ACL および IPSG など）は同じ層の接続ポイントに接続する Wireshark キャプチャ ポイントでは検出されません。一方、出力分類ベースのセキュリティ機能によってドロップされたパケットは、同じ層の接続ポイントに接続されている Wireshark のキャプチャ ポイントでキャッチされます。論理モデルは、Wireshark の接続ポイントが、入力側のセキュリティ機能のルックアップ後、および出力側のセキュリティ機能のルックアップ前に発生することです。

  入力では、パケットはレイヤ 2 ポート、VLAN、およびレイヤ 3 ポート/SVI を介して送信されます。出力では、パケットはレイヤ 3 ポート/SVI、VLAN、およびレイヤ 2 ポートを介して送信されます。接続ポイントがパケットがドロップされるポイントの前にある場合、Wireshark はパケットをキャプチャします。これ以外の場合は、Wireshark はパケットをキャプチャしません。たとえば、入力方向のレイヤ2接続ポイントに接続される Wireshark のキャプチャ ポリシーはレイヤ3分類ベースのセキュリティ機能によってドロップされたパケットをキャプチャします。対照的に、出力方向のレイヤ 3 接続ポイントに接続する

Wiresharkのキャプチャ ポリシーは、レイヤ2分類ベースのセキュリティ機能によりドロップされたパケットをキャプチャします。

- ルーテッドポートおよびスイッチ仮想インターフェイス（SVIs）：SVIの出力から送信されるパケットはCPUで生成されるため、WiresharkはSVIの出力をキャプチャできません。これらのパケットをキャプチャするには、コントロールプレーンを接続ポイントとして含めます。
- VLAN：Cisco IOSリリース16.1以降、VLANがWiresharkの接続ポイントとして使用されている場合、パケットキャプチャは、入力方向と出力方向の両方のL2とL3でサポートされます。
- リダイレクション機能：入力方向では、レイヤ3（PBRおよびWCCPなど）でリダイレクトされる機能トラフィックは、レイヤ3のWiresharkの接続ポイントよりも論理的に後です。Wiresharkは、後で別のレイヤ3インターフェイスにリダイレクトされる可能性がある場合でも、これらのパケットをキャプチャします。対照的に、レイヤ3によってリダイレクトされる出力機能（出力WCCPなど）は論理的にレイヤ3接続ポイントの前にあり、Wiresharkではキャプチャされません。
- SPAN：Wiresharkは、SPAN宛先として設定されたインターフェイスでパケットをキャプチャできません。
- SPAN：Wiresharkは、入力方向のSPAN送信元として設定されたインターフェイスでパケットをキャプチャできます。出力方向でも使用できる可能性があります。
- ACLが適用されていない場合、最大1000のVLANからパケットを一度にキャプチャできます。ACLが適用されている場合、Wiresharkの使用できるハードウェア領域はより少なくなります。結果として、パケット キャプチャに一度に使用できるVLANの最大数は低くなります。1000以上のVLANトンネルを一度に使用したり、ACLを多数使用すると予測されない結果が生じる可能性があります。たとえば、モビリティがダウンする可能性があります。

（注） 過剰なCPU使用につながり、予測されないハードウェア動作の原因となる可能性があるため、過剰な数の接続ポイントを一度にキャプチャしないことを強くお勧めします。

Wireshark 設定のガイドライン

- Wireshark でのパケット キャプチャ中に、ハードウェア転送が同時に発生します。
- パケット転送はハードウェアで通常実行されるため、パケットは、ソフトウェア処理のためにCPUにコピーされません。Wiresharkのパケットキャプチャの場合、パケットがコピーされてCPUに送信されるため、CPU使用率が高くなります。.
- 次の場合に高いCPU（またはメモリ）使用率になる可能性があります。

- キャプチャセッションをイネーブルにし長期間不在のままにして、予期しないトラフィックのバーストが起きた場合。
- リング ファイルまたはキャプチャ バッファを使用してキャプチャ セッションを起動して、長期間不在のままにすると、パフォーマンスまたはシステム ヘルスの問題が引き起こされます。

- 高い CPU 使用率を最小限に抑えるには、次の手順を実行します。
  - 関連ポートだけに接続します。
  - 一致条件を表すにはクラス マップを使用し、二次的にアクセス リストを使用してください。いずれも実行可能でない場合は、明示的な、インライン フィルタを使用します。
  - フィルタルールに厳密に従ってください。不要なトラフィックを呼び込む可能性がある緩やかな ACL ではなく、厳密な ACL を使用してトラフィックタイプ（IPv4のみなど）を制限します。
  - ライブトラフィックのキャプチャに Wireshark を使用している場合、QoS ポリシーを一時的に適用して、キャプチャプロセスが終了するまで実際のトラフィックを制限することを考慮してください。

- パケット キャプチャを短い期間または小さなパケット番号に常に制限します。capture コマンドのパラメータにより、次を指定することができます。
  - キャプチャ期間
  - キャプチャされたパケットの数
  - ファイル サイズ
  - パケットのセグメント サイズ

- キャプチャセッション中に、デバイスのパフォーマンスやヘルスに影響する可能性のある Wireshark による高い CPU 使用率およびメモリ消費がないか監視します。こうした状況が発生した場合、Wireshark セッションをすぐに停止します。
- コアフィルタと一致するトラフィックが非常に少ないことが判明している場合は、制限なしでキャプチャセッションを実行します。
- Wireshark インスタンスは最大 8 個まで定義できます。.pcap ファイルまたはキャプチャ バッファからパケットをデコードして表示するアクティブな **show** コマンドは、1 個のインスタンスとしてカウントされます。ただし、アクティブにできるインスタンスは1つだけです。
- 実行中のキャプチャに関連付けられているアクセス制御リスト（ACL）を変更する場合は、キャプチャを再起動して変更を適用します。キャプチャを再起動しない場合、変更されていないかのように元の ACL が引き続き使用されます。

- フラッシュディスクへの書き込みは、CPUを集中的に使用する操作です。キャプチャレートが不十分な場合は、バッファキャプチャを使用することをお勧めします。
- 大きなファイルの .pcap ファイルからのパケットをデコードして表示することは避けてください。代わりに、PC に .pcap ファイルを転送し PC 上で Wireshark を実行します。
- ストレージ ファイルにパケットを保存する予定の場合、Wireshark キャプチャ プロセスを開始する前に十分なスペースが利用可能であることを確認してください。
- パケット損失を防ぐには、次の点を考慮します。
  - ライブパケットのキャプチャ中には、ストアのみ（表示オプションを指定しない場合）を使用します。CPUに負荷がかかる操作（特に詳細モード）である複合化と表示には使用しないでください。
  - パケットをバッファ内に保存する複数のキャプチャがある場合、メモリ ロスを避けるため、新しいキャプチャを開始する前にバッファをクリアしてください。
  - デフォルトバッファサイズを使用し、パケットが失われている場合、バッファサイズを増加してパケットの喪失を防ぐことができます。
- コンソールウィンドウのライブパケットを複合化して表示する場合は、短いキャプチャ期間で Wireshark セッションをバインドしていることを確認してください。
- コア フィルタは明示的なフィルタ、アクセス リスト、またはクラス マップにできます。これらのタイプの新しいフィルタを指定すると、既存のものを置き換えます。

（注） コア フィルタは、CAPWAP トンネル インターフェイスをキャプチャ ポイントの接続ポイントとして使用している場合を除き、必須です。

- キャプチャポイントを定義する場合、特定の順序は適用されません。CLI で許可されている場合は、キャプチャ ポイント パラメータを任意の順序で定義できます。Wireshark CLI では、単一行のパラメータ数に制限はありません。これはキャプチャ ポイントを定義するために必要なコマンドの数を制限します。
- 接続ポイントを除くすべてのパラメータは、単一の値を取ります。通常、コマンドを再入力することにより、値を新しいものに置き換えることができます。ユーザーの確認後にシステムが新しい値を受け入れ、古い値を上書きします。コマンドの **no** 形式は、新しい値の入力には必要はありませんが、パラメータの削除には必要です。
- Wireshark では 1 つ以上の接続ポイントを指定することができます。複数の接続ポイントを追加するには、新しい接続ポイントでコマンドを再入力します。接続ポイントを削除するには、コマンドの **no** 形式を使用します。接続ポイントとしてインターフェイス範囲を指定できます。

  たとえば、**monitor capture mycap interface GigabitEthernet1/0/1 in** と入力します。ここで、GigabitEthernet1/0/1 は接続ポイントです。インターフェイス GigabitEthernet1/0/2 も接続す

る必要がある場合は、次のように入力します **monitor capture mycap interface GigabitEthernet1/0/2 in**

- 実行する処理は、いずれのパラメータが必須であるかを決定します。Wireshark CLI では、**start** コマンドを入力する前に任意のパラメータを指定または変更することができます。**start** コマンドを入力すると、すべての必須パラメータが入力されたと判断した後にのみ Wireshark が開始します。
- キャプチャポイントの作成時にファイルが存在する場合、Wireshark はファイルを上書きできるかどうかを確認します。キャプチャポイントのアクティブ化時にファイルが存在する場合、Wireshark は既存のファイルを上書きします。
- 明示的な **stop** コマンドを使用するか、automore モードに **q** を入力して、Wireshark のセッションを終了します。セッションは、期間やパケットキャプチャの制限などの停止の条件が満たされたときに、自身を自動的に終了することができます。内部エラーが発生した場合、またはリソースがフルになった場合（特に、ディスクがファイルモードでフルの場合）に終了することがあります。
- ドロップされたパケットはキャプチャの最後に表示されません。ただし、ドロップされたサイズ超過のパケット数のみが表示されます。

Wireshark 設定の前提条件

- Wireshark は、DNA Advantage を実行しているスイッチのみでサポートされています。
- Wireshark のキャプチャ プロセスを開始する前に、CPU 使用率が妥当であり、十分なメモリ（少なくとも 200 MB）が使用可能であることを確認します。Wireshark キャプチャ中の CPU使用率は、指定された条件に一致するパケットの数によって異なります。また、一致したパケットの意図されたアクション（保存、複合化、表示、またはその両方）によっても異なります。

Wireshark 設定の制約事項

- Wireshark は、グローバル パケット キャプチャをサポートしていません。
- Wireshark は、ファイルサイズによる循環ファイルストレージの制限をサポートしていません。
- アクティブなキャプチャセッションで使用されているファイルを削除すると、キャプチャセッションで新しいファイルを作成できなくなります。キャプチャされた以降のパケットはすべて失われます。キャプチャポイントを再起動する必要があります。
- ファイル制限は、 のフラッシュのサイズに限定されます。
- Wireshark は宛先 SPAN ポートでパケットをキャプチャできません。
- Wireshark は、キャプチャ ポイントにアタッチされる接続ポイント（インターフェイス）のいずれかが動作を停止するとキャプチャを停止します。たとえば、接続ポイントに関連

付けられているデバイスがデバイスから切断された場合です。キャプチャを再開するには、手動で再起動します。

- ストリーミング キャプチャ モードは約 1000 pps をサポートし、ロックステップ モードは約 2 Mbps（256 バイト パケットで測定）をサポートします。一致するトラフィック レートがこの値を超えると、パケット損失が発生する可能性があります。
- キャプチャがアクティブな場合、キャプチャポイントを変更することはできません。
- Wireshark は floodblock によってドロップされるパケットをキャプチャしません。
- Wireshark クラスマップでは、1 つの ACL（IPv4、IPv6、または MAC）のみが許可されます。
- ACL ロギングおよび Wireshark には互換性がありません。Wireshark を有効にすると、それが優先されます。ポート上の ACL ロギングによってキャプチャされたトラフィックを含むすべてのトラフィックは、Wireshark にリダイレクトされます。Wireshark を開始する前に、ACL ロギングを非アクティブにすることをお勧めします。これを実行しないと、Wireshark のトラフィックは ACL ロギング トラフィックに汚染されます。
- 同じポートの PACL および RACL の両方をキャプチャすると、1 つのコピーだけが CPU に送信されます。DTLS 暗号化 CAPWAP インターフェイスをキャプチャすると、暗号化されたものと復号されたものの 2 つのコピーが Wireshark に送信されます。DTLS 暗号化 CAPWAP トラフィックを運ぶレイヤ 2 インターフェイスをキャプチャすると同じ動作が発生します。コア フィルタは外部 CAPWAP ヘッダーに基づいています。
- Wireshark を設定するための CLI では、機能を EXEC モードからのみ実行する必要があります。通常は設定サブモードで発生するアクション（キャプチャ ポイントの定義など）は、代わりに EXEC モードから処理されます。すべての主要コマンドは NVGEN の対象ではなく、NSF と SSO のシナリオではスタンバイスーパーバイザに同期されません。

組み込み型の Wireshark はサポートされていますが、次の制限があります。

- キャプチャフィルタと表示フィルタはサポートされません。
- アクティブなキャプチャの復号化は使用できません。
- 出力形式は、以前のリリースとは異なります。
- 期間制限がより長いまたはキャプチャ期間がない（**term len 0** コマンドを使用して auto-more サポートのない端末を使用した）Wireshark セッションでは、コンソールまたは端末が使用できなくなる場合があります。
- パケットキャプチャのフィルタとしてのパケット長の範囲は、非 IPv4/IPv6 パケットおよびフラグメント化されたパケットではサポートされません。
- フィルタとしてのパケット長の範囲は、他のフィルタとともに使用できません。

パケットキャプチャの設定方法

ここでは、パケットキャプチャの設定について説明します。

パケット データ キャプチャの管理

（注） アクティブなキャプチャを停止した後にのみ、アクティブなキャプチャポイントをエクスポートできます。

バッファ モードでパケット データ キャプチャを管理するには、次の手順を実行します。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **monitor capture** *capture-name* **access-list** *access-list-name*

例：

```
Device# monitor capture mycap access-list v4acl
```

アクセス リストをパケット キャプチャのコア フィルタとして指定し、モニター キャプチャを設定します。

ステップ 3 **monitor capture** *capture-name* **limit duration** *seconds*

例：

```
Device# monitor capture mycap limit duration 1000
```

モニター キャプチャの制限を設定します。

ステップ 4 **monitor capture** *capture-name* **interface** *interface-name* **both**

例：

```
Device# monitor capture mycap interface GigabitEthernet 0/0/1 both
```

接続ポイントおよびパケット フロー方向を指定して、モニター キャプチャを設定します。

ステップ 5 **monitor capture** *capture-name* **buffer circular size** *bytes*

例：

```
Device# monitor capture mycap buffer circular size 10
```

パケット データをキャプチャするようにバッファを設定します。

ステップ 6 **monitor capture** *capture-name* **start**

例：

```
Device# monitor capture mycap start
```

トラフィック トレース ポイントでパケット データのバッファへのキャプチャを開始します。

ステップ 7 **monitor capture** *capture-name* **stop**

例：

```
Device# monitor capture mycap stop
```

トラフィック トレース ポイントでパケット データのキャプチャを停止します。

ステップ 8 **monitor capture** *capture-name* **export** *file-location/file-name*

例：

```
Device# monitor capture mycap export tftp://10.1.88.9/mycap.pcap
```

分析のためにキャプチャされたデータをエクスポートします。

ステップ 9 **end**

例：

```
Device# end
```

特権 EXEC モードに戻ります。

キャプチャされたデータのモニタリングとメンテナンス

キャプチャされたパケットデータのモニタリングとメンテナンスを行うには、次の作業を実行します。キャプチャバッファとキャプチャポイントの詳細が、手順中に表示されます。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **show monitor capture** *capture-buffer-name* **buffer dump**

例：

```
Device# show monitor capture mycap buffer dump
```

（任意）キャプチャ パケットの 16 進数ダンプおよびそのメタデータを表示します。

ステップ 3 **show monitor capture** *capture-buffer-name* **parameter**

例：

```
Device# show monitor capture mycap parameter
```

（任意）キャプチャを指定するために使用されたコマンドのリストを表示します。

ステップ 4 **debug epc capture-point**

例：

```
Device# debug epc capture-point
```

（任意）パケット キャプチャ ポイントのデバッグを有効にします。

ステップ 5 **debug epc provision**

例：

```
Device# debug epc provision
```

（任意）パケット キャプチャ プロビジョニングのデバッグを有効にします。

ステップ 6 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

Wireshark の設定方法

Wireshark を設定するには、次の基本的な手順を実行します。

1. キャプチャ ポイントを定義します。
2. キャプチャポイントのパラメータを追加または変更します。

3. キャプチャ ポイントをアクティブ化または非アクティブ化します。
4. 不要になったらキャプチャポイントを削除します。

キャプチャ ポイントの定義

この手順の例では、非常にシンプルなキャプチャポイントを定義します。必要に応じて、キャプチャポイントのすべてのパラメータを 1 つの **monitor capture** コマンドで定義できます。

（注） 機能するキャプチャポイントを設定するには、接続ポイントを定義し、キャプチャの方向を指定して、コアフィルタを設定します。

CAPWAP トンネリング インターフェイスを使用してワイヤレス キャプチャ ポイントを作成する場合、コアフィルタを定義する必要はありません。

キャプチャポイントを定義するには、次の手順を実行します。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **monitor capture** {*capture-name*}{**interface** *interface-type interface-id* | **control-plane**}{**in** | **out** | **both**}

例：

```
Device# monitor capture mycap interface GigabitEthernet1/0/1 in
```

キャプチャ ポイントを定義し、キャプチャ ポイントが関連付けられている接続ポイントを指定し、キャプチャの方向を指定します。

キーワードの意味は次のとおりです。

- *capture-name*：定義するキャプチャ ポイントの名前を指定します（例では mycap が使用されています）。キャプチャ名の長さは 8 文字以下にしてください。英数字、アンダースコア（_）のみが許可されます
- （任意）**interface***interface-type interface-id*：キャプチャポイントが関連付けられる接続ポイントを指定します（例では GigabitEthernet1/0/1 が使用されています）。

（注）

オプションで、このコマンドインスタンス 1 つでこのキャプチャポイントの複数の接続ポイントおよびパラメータすべてを定義できます。これらのパラメータについては、キャプチャ ポイント パラメータの変更に関する手順で説明されています。範囲のサポートは、接続ポイントを追加および削除するためにも使用できます。

interface-type には次のいずれかのオプションを使用します。

- **AppGigabitEthernet**：接続ポイントを AppGigabitEthernet として指定します。
- **GigabitEthernet**：接続ポイントを GigabitEthernet として指定します。
- **vlan**：接続ポイントを VLAN として指定します。

  （注）
  このインターフェイスを接続ポイントとして使用する場合は、入力キャプチャのみが可能です。

- **capwap**：接続ポイントを CAPWAP トンネルとして指定します。

  （注）
  このインターフェイスを接続ポイントとして使用する場合、コアフィルタを使用することはできません。

- （任意）**control-plane**：接続ポイントとしてコントロールプレーンを指定します。
- **in** | **out** | **both**：キャプチャの方向を指定します。

ステップ 3 **monitor capture** {*capture-name*}[**match** {**any** | **ipv4 any any** | **ipv6**} **any any**}]

例：

```
Device# monitor capture mycap interface GigabitEthernet1/0/1 in match any
```

コア システムのフィルタを定義します。

（注）
コアフィルタが使用できなくなるため、CAPWAP のトンネリング インターフェイスを接続ポイントとして使用する場合はこの手順を実行しないでください。

キーワードの意味は次のとおりです。

- *capture-name*：定義するキャプチャ ポイントの名前を指定します（例では mycap が使用されています）。
- **match**：フィルタを指定します。定義されている最初のフィルタはコア フィルタです。

  （注）
  キャプチャポイントにコアシステムフィルタまたは接続ポイントが定義されていない場合、それをアクティブにすることはできません。キャプチャポイントがすべての要件を満たしており、アクティブ化前にエラーを回避していることを確認します。

- **ipv4**：IP バージョン 4 のフィルタを指定します。

- **ipv6**：IP バージョン 6 のフィルタを指定します。

ステップ 4 **show monitor capture** {*capture-name*}[**parameter**]

例：

```
Device# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/1 in
   monitor capture mycap match any
```

ステップ 2 で定義したキャプチャ ポイント パラメータを表示し、キャプチャポイントを定義したことを確認します。

ステップ 5 **show capwap summary**

例：

```
Device# show capwap summary
```

ワイヤレス キャプチャの接続ポイントとして使用できる CAPWAP トンネルを表示します。

（注）
このコマンドは、ワイヤレス キャプチャを実行するために CAPWAP トンネルを接続ポイントとして使用している場合にのみ使用します。例の項の CAPWAP の例を参照してください。

ステップ 6 **show running-config**

例：

```
Device# show running-config
```

入力を確認します。

ステップ 7 **copy running-config startup-config**

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

例

CAPWAP 接続ポイントでキャプチャ ポイントを定義するには次を実行します。

```
Device# show capwap summary

CAPWAP Tunnels General Statistics:
  Number of Capwap Data Tunnels       = 1
  Number of Capwap Mobility Tunnels   = 0
  Number of Capwap Multicast Tunnels  = 0


Name   APName                           Type PhyPortIf Mode      McastIf
------ -------------------------------- ---- --------- --------- -------
```

```
Ca0    AP442b.03a9.6715                 data Gi3/0/6   unicast   -


Name   SrcIP           SrcPort DestIP          DstPort DtlsEn MTU   Xact
------ --------------- ------- --------------- ------- ------ ----- ----
Ca0    10.10.14.32     5247    10.10.14.2      38514   No     1449  0

Device# monitor capture mycap interface capwap 0 both
Device# monitor capture mycap file location flash:mycap.pcap
Device# monitor capture mycap file buffer-size 1
Device# monitor capture mycap start

*Aug 20 11:02:21.983: %BUFCAP-6-ENABLE: Capture Point mycap enabled.on

Device# show monitor capture mycap parameter
   monitor capture mycap interface capwap  0 in
   monitor capture mycap interface capwap  0 out
   monitor capture mycap file location flash:mycap.pcap buffer-size 1
Device#
Device# show monitor capture mycap

Status Information for Capture mycap
  Target Type:
  Interface: CAPWAP,
    Ingress:
 0
    Egress:
 0
   Status : Active
  Filter Details:
    Capture all packets
  Buffer Details:
   Buffer Type: LINEAR (default)
  File Details:
   Associated file name: flash:mycap.pcap
   Size of buffer(in MB): 1
  Limit Details:
   Number of Packets to capture: 0 (no limit)
   Packet Capture duration: 0 (no limit)
   Packet Size to capture: 0 (no limit)
   Packets per second: 0 (no limit)
   Packet sampling rate: 0 (no sampling)
Device#
Device# show monitor capture file flash:mycap.pcap
  1   0.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  2   0.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  3   2.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  4   2.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  5   3.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  6   4.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  7   4.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  8   5.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
  9   5.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 10   6.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
```

```
 11   8.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 12   9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 13   9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 14   9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 15   9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 16   9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 17   9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 18   9.236987   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 19  10.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 20  10.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 21  12.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 22  12.239993   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 23  12.244997   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 24  12.244997   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 25  12.250994   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 26  12.256990   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 27  12.262987   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 28  12.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
 29  12.802012   10.10.14.3 -> 10.10.14.255 NBNS Name query NB WPAD.<00>
 30  13.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0,
FN=0, Flags=........
```

次のタスク

接続ポイントをさらに追加し、キャプチャポイントのパラメータを変更してから、アクティブ化できます。キャプチャポイントをそのまま使用する場合は、有効化できます。

（注） このトピックで説明されている方法を使用してキャプチャポイントのパラメータを変更することはできません。

ユーザーが間違ったキャプチャ名を入力した場合、または無効または存在しない接続ポイントを入力した場合、スイッチはエラーを表示します。たとえば、「キャプチャ名は*8*文字以下である必要があります。*Only alphanumeric characters and underscore (_) is permitted*」および「*% Invalid input detected at '^' marker*」のようなエラーを表示します。

キャプチャ ポイント パラメータの追加またはモニタリング

パラメータは順番にリストされていますが、任意の順番で手順を実行して値を指定できます。1 行、2 行、または複数行で指定できます。複数指定が可能な接続ポイントを除き、同じオプションを再定義することで、任意の値をより最近の値に置き換えることができます。すでに指定されている特定のパラメータを変更する場合は、インタラクティブに確認する必要があります。

キャプチャポイントのパラメータを変更するには、次の手順を使用します。

始める前に

これらの手順を使用する前に、キャプチャポイントを定義する必要があります。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **monitor capture** {*capture-name*} **match** {**any** | **mac** *mac-match-string* | **ipv4** {**any** | **host** | **protocol**}{**any** | **host**} | **ipv6** {**any** | **host** | **protocol**}{**any** | **host**}}

例：

```
Device# monitor capture mycap match ipv4 any any
```

ACL またはクラスマップで明示的に定義されたコアシステムフィルタ（**ipv4 any any**）を定義します。

ACL を使用してコアシステムフィルタを定義できます。ACL でプロトコルの Ethertype を設定できます。Wireshark で同じ ACL を設定して、特定の Ethertype を持つパケットのキャプチャを有効にすることができます。

ステップ 3 **monitor capture** {*capture-name*} **limit** {[**duration** *seconds*][**packet-length** *size*][**packets** *num*]}

例：

```
Device# monitor capture mycap limit duration 60 packet-len 400
```

秒単位のセッション制限（60）、キャプチャされたパケット、または Wireshark によって保持されるパケットセグメント長（400）を指定します。

ステップ 4 **monitor capture** {*capture-name*} **file** {**location** *filename*}

例：

```
Device# monitor capture mycap file location flash:mycap.pcap
```

キャプチャ ポイントがパケットを表示するだけでなくキャプチャできるようにする場合は、ファイルのアソシエーションを指定します。

（注）
ファイルが存在する場合、上書き可能か確認してください。

ステップ 5 **monitor capture** {*capture-name*} **file** {**buffer-size** *size*}

例：

```
Device# monitor capture mycap file buffer-size 100
```

トラフィック バーストの処理に Wireshark で使用されるメモリ バッファのサイズを指定します。

ステップ 6 **show monitor capture** {*capture-name*}[**parameter**]

例：

```
Device# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/1 in
   monitor capture mycap match ipv4  any any
   monitor capture mycap limit duration 60 packet-len 400
   monitor capture point mycap file location bootdisk:mycap.pcap
   monitor capture mycap file buffer-size 100
```

すでに定義されているキャプチャ ポイント パラメータを表示します。

ステップ 7 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

パラメータの変更

キャプチャ ファイルの関連付けまたは関連付け解除

```
Device# monitor capture point mycap file location flash:mycap.pcap
Device# no monitor capture mycap file
```

パケット バーストの処理にメモリ バッファ サイズを指定する

```
Device# monitor capture mycap buffer size 100
```

IPv4 と IPv6 の両方に一致するように、明示的なコア システム フィルタを定義する

```
Device# monitor capture mycap match any
```

パケットのイーサタイプの指定

```
MAC ACL:
Device(config)#mac access-list extended mac1
Device(config-ext-macl)#permit any any 0x806 0x0
Device(config-ext-macl)exit
Device(config)#monitor capture mycap access-list mac1

IP ACL:
Device#ip access-list extended ip1
Device(config-ext-nacl)#permit 1 any any icmp-message-type
Device(config-ext-nacl)# exit
Device#monitor capture mycap access-list ip1
```

次のタスク

キャプチャポイントに必要なパラメータがすべて含まれている場合はアクティブ化します。

キャプチャ ポイント パラメータの削除

順番に表示されていますが、パラメータを削除する手順は任意の順序で実行できます。1 行、2 行、または複数行で削除できます。複数のエントリを含む可能性のある接続ポイントを除くすべてのパラメータを削除できます。

キャプチャポイントパラメータを削除するには、次の手順を実行します。

始める前に

これらの手順を使用してキャプチャポイントを削除する前に、キャプチャポイントのパラメータを定義してください。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **no monitor capture** {*capture-name*} **match**

例：

```
Device# no monitor capture mycap match
```

キャプチャ ポイント（mycap）で定義されているすべてのフィルタを削除します。

ステップ 3 **no monitor capture** {*capture-name*} **limit** [**duration**] [**packet-length**] [**packets**]

例：

```
Device# no monitor capture mycap limit duration packet-len
Device# no monitor capture mycap limit
```

Wireshark が保持しているセッション時間制限とパケットセグメント長を削除します。その他の指定された制限はそのままになります。

Wireshark のすべての制限をクリアします。

ステップ 4 **no monitor capture** {*capture-name*} **file** [**location**] [**buffer-size**]

例：

```
Device# no monitor capture mycap file
Device# no monitor capture mycap file location
```

ファイルの関連付けを削除します。キャプチャ ポイントはパケットをキャプチャしなくなります。それらは表示されるのみです。

ファイル位置の関連付けを削除します。ファイルの場所は、キャプチャポイントに関連付けられなくなりました。ただし、他の定義されたファイルの関連付けは、このアクションの影響を受けません。

ステップ 5 **show monitor capture** {*capture-name*}[**parameter**]

例：

```
Device# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/1 in
```

パラメータの削除操作後にまだ定義されているキャプチャポイントパラメータを表示します。手順のどの段階でもこのコマンドを実行して、キャプチャポイントに関連付けられているパラメータを確認できます。

ステップ 6 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

次のタスク

キャプチャポイントに必要なパラメータがすべて含まれている場合はアクティブ化します。

（注） キャプチャポイントがアクティブなときにパラメータを削除すると、スイッチに「*Capture is active*」というエラーが表示されます。

キャプチャ ポイントの削除

キャプチャ ポイントを削除するには、次の手順を実行します。

始める前に

これらの手順を使用して削除する前に、キャプチャポイントを定義します。キャプチャポイントを削除する前に停止してください。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **no monitor capture** {*capture-name*}

例：

```
Device# no monitor capture mycap
```

指定されたキャプチャ ポイント（mycap）を削除します。

ステップ 3 **show monitor capture** {*capture-name*}[**parameter**]

例：

```
Device# show monitor capture mycap parameter
   Capture mycap does not exist
```

指定されたキャプチャポイントが削除されたため、存在しないことを示すメッセージを表示します。

ステップ 4 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 5 **show running-config**

例：

```
Device# show running-config
```

入力を確認します。

ステップ 6 **copy running-config startup-config**

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

次のタスク

削除したものと同じ名前の新規キャプチャポイントを定義します。キャプチャポイントの定義を最初からやり直す場合は、これらの手順を実行できます。

キャプチャ ポイントをアクティブまたは非アクティブにする

キャプチャ ポイントをアクティブまたは非アクティブにするには、次の手順を実行します。

始める前に

接続ポイント、コアシステムフィルタ、および関連付けられたファイル名が存在する場合でも、キャプチャポイントをアクティブ化できます。. このようなケースでは、既存のファイルは上書きされます。

関連するファイル名のないキャプチャポイントは、表示するためだけにアクティブにできます。ファイル名が指定されていないと、パケットはバッファに保管されます。ライブ表示（キャプチャ時の表示）は、ファイルおよびバッファモードの両方で使用できます。

表示フィルタが指定されていない場合、パケットはライブで表示されません。ただし、コアシステムフィルタでキャプチャされたパケットは表示され、デフォルトの表示モードは短時間です。

（注） CAPWAP トンネリングインターフェイスが接続ポイントとして使用される場合、コアフィルタは使用されません。したがって、それらを定義する必要はありません。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **monitor capture** {*capture-name*} **start** [**display** [**display-filter** *filter-string*]] [**brief** | **detailed** | **dump**]

例：

```
Device# monitor capture mycap start display display-filter "stp"
```

キャプチャポイントをアクティブ化し、「stp」を含むパケットだけが表示されるように表示をフィルタ処理します。

ステップ 3 **monitor capture** {*capture-name*} **stop**

例：

```
Device# monitor capture name stop
```

キャプチャ ポイントを非アクティブにします。

ステップ 4 end

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 5 show running-config

例：

```
Device# show running-config
```

入力を確認します。

ステップ 6 copy running-config startup-config

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

次のタスク

キャプチャ ポイントをアクティブおよび非アクティブにする際に、いくつかのエラーが発生する可能性があります。次に、発生する可能性のあるエラーのいくつかの例を示します。

接続ポイントが定義されていない場合に、「アクティベーション時に接続ポイントがありません」というエラーが発生します。

```
Device# monitor capture mycap match any
Device# monitor capture mycap start
No Target is attached to capture failed to disable provision featurefailed to remove
policyfailed to disable provision featurefailed to remove policyfailed to disable provision
 featurefailed to remove policy
Capture statistics collected at software (Buffer):
 Capture duration - 0 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0

Unable to activate Capture.
Device# unable to get action unable to get action unable to get action
Device# monitor capture mycap interface g1/0/1 both
Device#monitor capture mycap start
Device#
*Nov 5 12:33:43.906: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

アクティブ化する際にフィルタが不明

```
Device# monitor capture mycap int g1/0/1 both
Device# monitor capture mycap start
Filter not attached to capture
```

```
Capture statistics collected at software (Buffer):
 Capture duration - 0 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0

Unable to activate Capture.
Device# monitor capture mycap match any
Device# monitor capture mycap start
Device#
*Nov 5 12:35:37.200: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

キャプチャ ポイントがすでにアクティブ化されているのに、別のキャプチャ ポイントをアクティブ化しようとする

```
Device# monitor capture mycap start
PD start invoked while previous run is active Failed to start capture : Wireshark operation
 failure
Unable to activate Capture.
Device# show monitor capture

Status Information for Capture test
 Target Type:
 Interface: GigabitEthernet1/0/13, Direction: both
 Interface: GigabitEthernet1/0/14, Direction: both
 Status : Active
 Filter Details:
 Capture all packets
 Buffer Details:
 Buffer Type: LINEAR (default)
 Buffer Size (in MB): 10
 File Details:
 Associated file name: flash:cchh.pcap
 Limit Details:
 Number of Packets to capture: 0 (no limit)
 Packet Capture duration: 0 (no limit)
 Packet Size to capture: 0 (no limit)
 Maximum number of packets to capture per second: 1000
 Packet sampling rate: 0 (no sampling)

Status Information for Capture mycap
 Target Type:
 Interface: GigabitEthernet1/0/1, Direction: both
 Status : Inactive
 Filter Details:
 Capture all packets
 Buffer Details:
 Buffer Type: LINEAR (default)
 Buffer Size (in MB): 10
 File Details:
 File not associated
 Limit Details:
 Number of Packets to capture: 0 (no limit)
 Packet Capture duration: 0 (no limit)
 Packet Size to capture: 0 (no limit)
 Maximum number of packets to capture per second: 1000
 Packet sampling rate: 0 (no sampling)
Device# monitor capture test stop
Capture statistics collected at software (Buffer & Wireshark):
 Capture duration - 157 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0
```

```
Device#
*Nov 5 13:18:17.406: %BUFCAP-6-DISABLE: Capture Point test disabled.
Device# monitor capture mycap start
Device#
*Nov 5 13:18:22.664: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
Device#
```

キャプチャ ポイント バッファのクリア

次の手順に従ってバッファコンテンツをクリアするか、外部ファイルにストレージとして保存します。

（注） パケットをバッファ内に保存する複数のキャプチャがある場合、メモリ ロスを避けるため、新しいキャプチャを開始する前にバッファをクリアしてください。アクティブなキャプチャ ポイントのバッファをクリアしないでください。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **monitor capture** {*capture-name*} [**clear** | **export** *filename*]

例：

```
Device# monitor capture mycap clear
```

clear：完全にバッファを削除します。

（注）
clear コマンドを実行すると、

- DNA Advantage ライセンスでは、このコマンドはバッファを削除せずにバッファの内容をクリアします
- 他のすべてのライセンスでは、このコマンドはバッファ自体を削除します。

Export：バッファでキャプチャされたパケットを保存し、バッファを削除します。

ステップ 3 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 4　show running-config

例：

```
Device# show running-config
```

入力を確認します。

ステップ 5　copy running-config startup-config

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

例：キャプチャ ポイント バッファの処理

キャプチャのファイルへのエクスポート

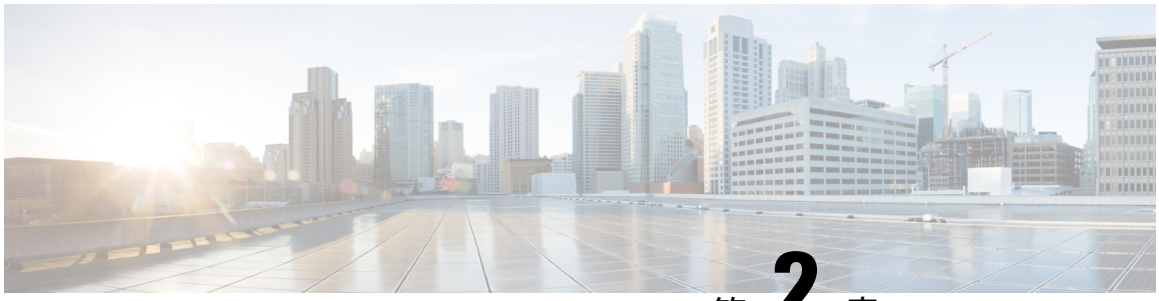
```
Device# monitor capture mycap export flash:mycap.pcap
Storage configured as File for this capture
```

キャプチャ ポイント バッファのクリア

```
Device# monitor capture mycap clear
 Capture configured with file options
```

次のタスク

（注）　DNA Advantage 以外のライセンスでキャプチャポイントのバッファをクリアしようとすると、スイッチは「*Failed to clear capture buffer : Capture Buffer BUSY*」エラーを表示します。



第 2 章

簡易ネットワーク管理プロトコル

- SNMP の概要 （33 ページ）
- SNMP マネージャ機能 （36 ページ）
- SNMP エージェント機能 （37 ページ）
- SNMP MIB 変数アクセス （37 ページ）
- SNMP フラッシュ MIB （38 ページ）
- SNMP 通知, on page 38
- SNMP ifIndex MIB オブジェクト値 （40 ページ）
- SNMP ENTITY-MIB 識別子 （41 ページ）
- SNMP および Syslog Over IPv6 （41 ページ）
- SNMP UDP ポート （41 ページ）
- SNMP のデフォルト設定, on page 42
- SNMP の制約事項, on page 42
- SNMP の設定方法, on page 43
- SNMP の例 （55 ページ）

SNMP の概要

Simple Network Management Protocol（SNMP）は、「IP ネットワーク上のデバイスを管理するためのインターネット標準プロトコル」です。

SNMP とは何ですか?

簡易ネットワーク管理プロトコル（SNMP）は、アプリケーション層プロトコルであり、マネージャとエージェントとの通信に使用されます。SNMP システムは、SNMP マネージャ、SNMP エージェント、および管理情報ベース（MIB）で構成されます。

SNMP マネージャは、Cisco Prime Infrastructure などのネットワーク管理システム（NMS）に統合できます。エージェントと MIB はネットワークデバイス上に存在します。デバイスに SNMP を設定するには、マネージャとエージェントの間の関係を定義します。

SNMP エージェントは MIB 変数を格納し、SNMP マネージャはこの変数の値を要求または変更できます。マネージャはエージェントから値を取得したり、エージェントに値を格納したりできます。エージェントは、デバイスパラメータやネットワーク データの保存場所である MIB から値を収集します。また、エージェントはマネージャのデータ取得またはデータ設定の要求に応答できます。

エージェントは非送信請求トラップをマネージャに送信できます。トラップは、ネットワーク上のある状態を SNMP マネージャに通知するメッセージです。トラップは不正なユーザー認証、再起動、リンク ステータス（アップまたはダウン）、MAC アドレス追跡、TCP 接続の終了、ネイバーとの接続の切断などの重要なイベントの発生を意味する場合があります。

SNMP バージョン

SNMP バージョンには、ネットワークデバイスを管理するためのさまざまな機能があります。このソフトウェアリリースは、SNMPv1、SNMPv2C、および SNMPv3 をサポートします。

- SNMPv1：RFC 1157 に規定された完全インターネット標準の簡易ネットワーク管理プロトコルです。
- SNMPv2C：SNMPv2Classic のパーティベースの管理およびセキュリティフレームワークを、コミュニティストリングベースのフレームワークに置き換えます。SNMPv2Classic の一括取得機能は保持され、エラー処理が改善されます。

  SNMPv2C には次が含まれます。

  - SNMPv2：RFC 1902 ～ 1907 に規定された SNMP バージョン 2（ドラフト版インターネット標準）
  - SNMPv2C：RFC 1901 に規定された SNMPv2 のコミュニティ ストリング ベースの管理フレームワーク（試験版インターネットプロトコル）

  SNMPv1 と SNMPv2C は、ともにコミュニティベースのセキュリティモデルを使用します。Management Information Base（MIB）にアクセスできるマネージャのコミュニティは、IP アドレスアクセスコントロールリストとパスワードによって定義されます。

  SNMPv2C には、テーブルや大量の情報を取得し、必要な往復回数を削減する一括取得機能が含まれています。また、SNMPv1 では単一のエラーコードで報告されるさまざまなエラー状態を区別する拡張エラーコードを提供し、エラー処理が改善されています。

- SNMPv3：SNMPv3（SNMP のバージョン 3）は、RFC 2273 ～ 2275 に規定されている相互運用可能な標準ベースのプロトコルです。SNMPv3 は、ネットワーク経由のパケットの認証と暗号化によって、デバイスへのセキュア アクセスを実現します。これには、次のセキュリティ機能が含まれています。

  - メッセージの完全性：パケットが伝送中に改ざんされないようにします。
  - 認証：メッセージが有効な送信元からのものであることを確認します。
  - 暗号化：パッケージの内容をミキシングし、許可されていない送信元に内容が読まれることを防止します。

（注） 暗号化を選択するには、priv キーワードを入力します。

SNMPv3 では、セキュリティ モデルとセキュリティ レベルの両方が提供されています。セキュリティモデルは、ユーザとユーザが属しているグループ用に設定された認証方式です。セキュリティ レベルとは、セキュリティ モデル内で許可されるセキュリティのレベルです。セキュリティレベルとセキュリティモデルの組み合わせにより、SNMP パケットを扱うときに使用するセキュリティ方式が決まります。使用可能なセキュリティ モデルは、SNMPv1、SNMPv2C、および SNMPv3 です。

次の表では、この特性を識別し、セキュリティ モデルとセキュリティ レベルの異なる組み合わせを比較します。

表 *2*：表 *1. SNMP* セキュリティ モデルおよびセキュリティ レベル

モデル	Level	認証	暗号化	結果
SNMPv1	noAuthNoPriv	コミュニティ ストリング	未対応	コミュニティ ストリングの照合を使用して認証します。
SNMPv2C	noAuthNoPriv	コミュニティ ストリング	未対応	コミュニティ ストリングの照合を使用して認証します。
SNMPv3	noAuthNoPriv	ユーザー名	未対応	ユーザ名の照合を使用して認証します。
SNMPv3	authNoPriv	Message Digest 5（MD5）または Secure Hash Algorithm（SHA）	未対応	HMAC-MD5 アルゴリズムまたは HMAC-SHA アルゴリズムに基づいて認証します。

モデル	Level	認証	暗号化	結果
SNMPv3	authPriv	MD5 または SHA	データ暗号規格（DES）または Advanced Encryption Standard（AES）	HMAC-MD5 アルゴリズムまたは HMAC-SHA アルゴリズムに基づいて認証します。 次の暗号化アルゴリズムで、User-based Security Model（USM）を指定できます。 • CBC-DES（DES-56）規格に基づく認証に加えた DES 56 ビット暗号化 • 3DES 168 ビット暗号化 • AES 128 ビット暗号化、192 ビット暗号化、または 256 ビット暗号化

SNMP マネージャ機能

SNMP マネージャは、MIB 情報を使用して、この表に示すようにさまざまな動作を実行します。

表 *3 : SNMP* の動作

動作	説明
get-request	特定の変数から値を取得します。
get-next-request	テーブル内の変数から値を取得します。この操作では、SNMP マネージャは正確な変数名を把握する必要はありません。テーブル内から必要な変数を見つけるために、シーケンシャル検索が実行されます。
get-bulk-request	テーブルの複数の行など、通常はサイズの小さい多数のデータ ブロックに分割して送信する必要がある巨大なデータ ブロックを取得します。 （注） このコマンドを使用できるのは、SNMPv2 以上に限られます。

動作	説明
get-response	NMS から送信される get-request、get-next-request、および set-request に対して応答します。
set-request	特定の変数に値を格納します。
trap	SNMP エージェントから SNMP マネージャに送られる、イベントの発生を伝える非送信請求メッセージです。

（注） パフォーマンスに関連する問題を回避するために、SNMP マネージャで ciscoFlashFileDate MIB オブジェクトをクエリから除外することを推奨します。これは、ciscoFlashFileDate オブジェクトが MIB で公開されていても、製品ではサポートされていないためです。

SNMP エージェント機能

SNMP エージェントは、1 つ以上の SNMP マネージャから要求を受信できます。すべての要求に、NMS の IP アドレス、NMS がエージェントをポーリングした回数、およびポーリングのタイムスタンプが含まれます。この情報は、IPv4 サーバーと IPv6 サーバーの両方で追跡できます。

SNMP エージェントは、次のようにして SNMP マネージャ要求に応答します。

- MIB 変数を取得する：SNMP エージェントは、NMS の要求に応じて要求された MIB 変数の値を取得し、その値で NMS に応答します。
- MIB 変数を設定する：SNMP エージェントは、NMS からのメッセージに応じて MIB 変数の値を要求された値に変更します。

show snmp stats hosts コマンドを使用して、キュー内の SNMP マネージャ要求のリストを表示します。**clear snmp stats hosts** コマンドを使用してキューをクリアします。

エージェントで重要なイベントが発生したことを NMS に通知するために、SNMP エージェントは非送信請求トラップ メッセージも送信します。トラップ条件の例には、ポートまたはモジュールがアップまたはダウン状態になった場合、スパニングツリートポロジが変更された場合、認証に失敗した場合などがあります。

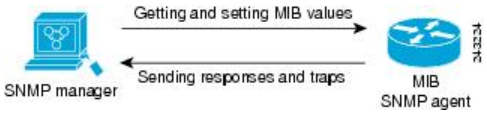
SNMP MIB 変数アクセス

Cisco Prime Infrastructure 3.1 ソフトウェアは NMS の例です。ソフトウェアは、デバイス MIB 変数を使用してデバイス変数を設定し、ネットワーク上のデバイスをポーリングして特定の情報を取得します。ポーリング結果は、グラフ形式で表示されます。この結果を解析して、イン

ターネットワーキング関連の問題のトラブルシューティング、ネットワーク パフォーマンスの改善、デバイス設定の確認、トラフィック負荷のモニターを行うことができます。

SNMP エージェントは MIB からデータを収集します。エージェントは SNMP マネージャに対し、トラップ（特定イベントの通知）を送信でき、SNMPマネージャはトラップを受信して処理します。トラップは、ネットワーク上で発生した不正なユーザー認証、再起動、リンクステータス（アップまたはダウン）、MACアドレストラッキングなどの状況をSNMPマネージャに通知します。SNMP エージェントはさらに、SNMP マネージャから、**get-request**、**get-next-request**、**set-request** 形式で送信される MIB 関連のクエリに応答します。

図 *1: SNMP* ネットワーク



SNMP フラッシュ MIB

Cisco フラッシュ MIB を使用すると、シスコ製のデバイスからフラッシュファイルデータを照会できます。フラッシュ MIB では、フラッシュファイルシステムからすべてのファイルを取得するようになりました。

フラッシュ MIB ウォークを実行するには、**snmp mib flash cache** コマンドを使用する必要があります。このコマンドは、すべてのファイルをローカルフラッシュ MIB キャッシュにプリフェッチします。

SNMP 通知

SNMPを使用すると、特定のイベントが発生した場合に、デバイスからSNMPマネージャに通知を送信できます。SNMP通知は、トラップまたは情報要求として送信できます。コマンド構文では、トラップまたは情報を選択するオプションがコマンドにない限り、キーワード traps はトラップ、情報、またはその両方を表します。**snmp-server host** コマンドを使用して、トラップまたは情報として SNMP 通知を送信するかどうかを指定します。

Note SNMPv1 は informs をサポートしていません。

トラップとインフォーム

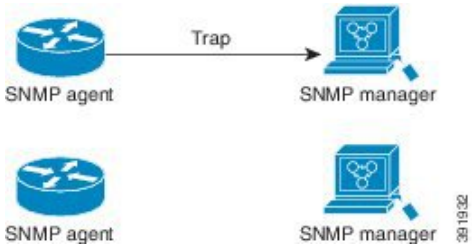
トラップは信頼性に欠けます。受信側はトラップを受信しても確認応答を送信しないので、トラップが受信されたかどうかが送信側にわからないからです。情報要求の場合、受信したSNMPマネージャは SNMP 応答プロトコル データ ユニット（PDU）でメッセージを確認します。送信側が応答を受信しなかった場合は、再び情報要求を送信できます。再送信できるので、情報の方がトラップより意図した宛先に届く可能性が高くなります。

情報の方がトラップより信頼性が高いのは、デバイスおよびネットワークのリソースを多く消費するという特性にも理由があります。送信と同時に廃棄されるトラップと異なり、情報要求は応答を受信するまで、または要求がタイムアウトになるまで、メモリ内に保持されます。トラップの送信は1回限りですが、情報は数回にわたって再送信つまり再試行が可能です。再送信の回数が増えるとトラフィックが増加し、ネットワークのオーバーヘッドが高くなる原因にもなります。したがって、トラップにするか情報にするかは、信頼性を取るかリソースを取るかという選択になります。SNMPマネージャですべての通知を受信することが重要な場合は、情報要求を使用してください。ネットワークまたはデバイスのメモリ上のトラフィックが問題になる場合で、なおかつ通知が不要な場合は、トラップを使用してください。

下図に、トラップとインフォームの違いを示します。

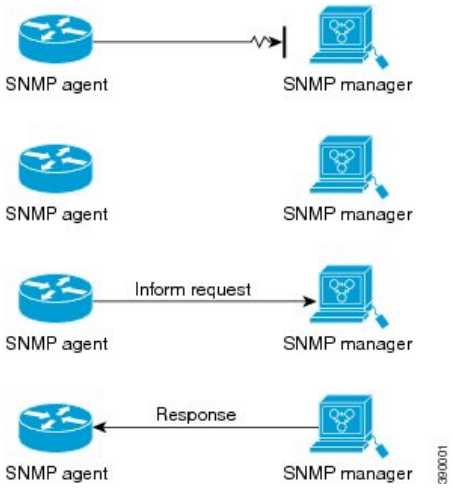
下図は、エージェントが SNMP マネージャへ正常にトラップを送信した場合を示します。マネージャはトラップを受信しても、確認応答を送信しません。エージェントには、トラップが宛先に到達したことを知る方法がありません。

Figure 2: SNMP マネージャに正常に送信されたトラップ



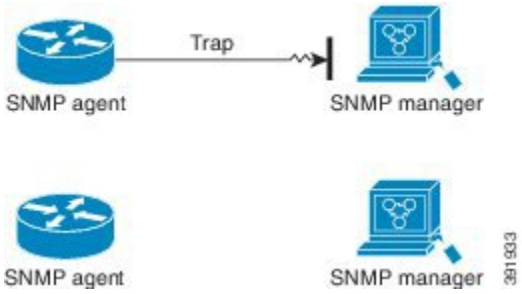
下図では、エージェントはマネージャへ正常にインフォームを送信しています。マネージャがインフォームを受信すると、応答がエージェントに送信されます。これにより、エージェントはインフォームが宛先に到達したことがわかります。この例では、上図に示すインタラクション内で 2 倍のトラフィックが生成されていることに注意してください。

Figure 3: SNMP マネージャに正常に送信されたインフォーム要求



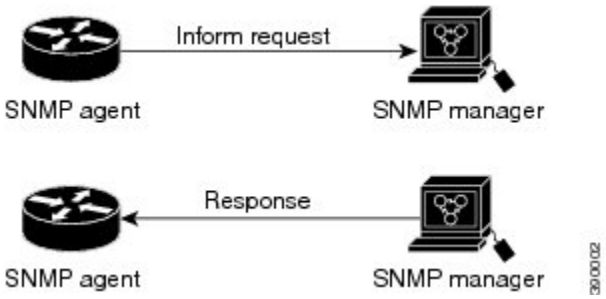
下図は、マネージャが受信しないトラップをマネージャに送信するエージェントを示します。エージェントには、トラップが宛先に到達しなかったことを知る方法がありません。トラップが再送信されないため、マネージャはトラップを受信しません。

Figure 4: SNMP マネージャに正常に送信されなかったトラップ



下図は、マネージャに到達しないインフォームをマネージャに送信するエージェントを示します。マネージャはインフォームを受信しなかったため、応答を送信しません。一定時間が経過すると、エージェントがインフォームを再送信します。マネージャは、2 番目の送信からインフォームを受信して応答します。この例では、上図に示すシナリオよりも多くのトラフィックが生成されますが、通知は SNMP マネージャに到達します。

Figure 5: SNMP マネージャに正常に送信されなかったインフォーム



Note SNMP プロセスが起動するたびに、予約ポート 161 および 162 が使用されます。これら 2 つの予約ポートに加えて、SNMP プロキシ フォワーダ アプリケーションを実行するためにダイナミックポートも開かれます。

SNMP ifIndex MIB オブジェクト値

SNMP エージェントの IF-MIB モジュールがリブート後すぐに起動されます。さまざまな物理インターフェイス ドライバが IF-MIB モジュールに登録を初期化し、ifIndex 番号を要求します。IF-MIB モジュールが先着順で使用可能な次の ifIndex 番号を割り当てます。1 つのリブートから他のリブートへのドライバの初期化順序のマイナーな違いが、同じ物理インターフェイスにリブートを行う以前のものとは別のインデックス番号を取得する可能性があるということです（インデックス持続が有効化されていない限り）。

SNMP ENTITY-MIB 識別子

ENTITY-MIBには、現場交換可能ユニット（FRU）、ファン、デバイスの電源装置などの物理エンティティを管理するための情報が含まれています。

各エンティティは、現在の MIB や他の MIB 内のエンティティに関する情報にアクセスする一意のインデックス番号（entPhysicalIndex）によって識別されます。エンティティの活性挿抜（OIR）により、新しいエンティティが挿入されたか、既存のエンティティが再挿入されたかに関係なく、エンティティには次に使用可能な entPhysicalIndex 番号が割り当てられます。

SNMP および Syslog Over IPv6

IPv4 と IPv6 の両方をサポートするには、IPv6 のネットワーク管理で IPv4 および IPv6 のトランスポートが必要になります。Syslog over IPv6 は、このトランスポートのアドレス データ タイプをサポートします。

Simple Network Management Protocol (SNMP) と syslog over IPv6 は、次の機能を提供します。

- IPv4 と IPv6 両方のサポート。
- SNMP に対する IPv6 トランスポート、および SNMP 変更による IPv6 ホストのトラップのサポート。
- IPv6 アドレス指定をサポートするための SNMP および syslog に関連する MIB。
- IPv6 ホストをトラップの受信者として設定。

Over IPv6 をサポートするため、SNMP は既存の IP トランスポート マッピングを変更して、IPv4 と IPv6 を同時にサポートします。次の SNMP 動作は、IPv6 トランスポート管理をサポートします。

- デフォルト設定のユーザー データグラム プロトコル（UDP）SNMP ソケットを開く。
- SR_IPV6_TRANSPORT と呼ばれる新しいトランスポートメカニズムを提供。
- IPv6 トランスポートによる SNMP 通知の送信。
- IPv6 トランスポートの SNMP 名のアクセスリストのサポート。
- IPv6 トランスポートを使用した SNMP プロキシ転送のサポート。
- SNMP マネージャ機能と IPv6 トランスポートの連動確認。

SNMP UDP ポート

SNMP プロセスはユーザーデータグラムプロトコル（UDP）ポート 161 および 162 を使用します。ポート 161 はデバイスのポーリングするために使用され、ポート 162 はエージェントから

サーバーに通知を送信するために使用されます。これらのポートは、いずれかの必須コマンドを設定しない限り、閉じたままになります。この設計により、必要な場合にのみポートが開くため、セキュリティが向上し、デバイスは不必要にポートをリッスンしなくなります。

SNMP のデフォルト設定

***Table 4:* 表 *3. SNMP* のデフォルト設定**

機能	デフォルト設定
SNMP エージェント	ディセーブル[1]
SNMP トラップ レシーバ	未設定
SNMP トラップ	TCP 接続のトラップ（tty）以外は、イネーブルではありません。
SNMP バージョン	バージョン キーワードがない場合、デフォルトはバージョン 1 になります。
SNMPv3 認証	キーワードを入力しなかった場合、セキュリティ レベルはデフォルトで **noauth**（noAuthNoPriv）になります。
SNMP 通知タイプ	タイプが指定されていない場合、すべての通知が送信されます。

[1] これは、デバイスが起動し、スタートアップ コンフィギュレーションに **snmp-server** グローバル コンフィギュレーション コマンドが設定されていない場合のデフォルトです。

SNMP の制約事項

- SNMPv1 は informs をサポートしていません。
- SNMPv3 認証は、次のシナリオではサポートされません。
  - スイッチ優先順位の変更後にスタックリロードが発生した場合。
  - 低い MAC アドレスを持つデバイスがスタックに追加された場合、スタック内のすべてのスイッチの優先順位が同じであれば、そのデバイスがアクティブスイッチとして選択されます。
- SNMPv3 認証の失敗を回避するには、SNMPv3 ユーザーを設定する前に、デバイスで SNMP engineID を手動で設定します。この設定により、ユーザーは engineID に関連付けられているためデバイスを管理できます。
- SNMP ENTITY-MIB は、イーサネット管理ポートではサポートされていません。

SNMP の設定方法

ここでは、SNMP の設定方法について説明します。

SNMP 設定時の注意事項

デバイスでは、SNMP User Datagram Protocol（UDP）ポート 161 および 162 を開き、SNMP エージェントを有効にするために、次のいずれかのグローバル コンフィギュレーション コマンドを設定する必要があります。

snmp-server host、**snmp-server user**、**snmp-server community**、または **snmp-server manager**

SNMP グループを設定するときには、次の注意事項に従ってください。

- SNMP グループを設定するときは、通知ビューを指定しません。snmp-server host グローバル コンフィギュレーション コマンドがユーザーの通知ビューを自動生成し、そのユーザーを対応するグループに追加します。グループの通知ビューを変更すると、そのグループに対応付けられたすべてのユーザが影響を受けます。
- リモートユーザを設定する場合は、ユーザが存在するデバイスのリモート SNMP エージェントに対応する IP アドレスまたはポート番号を指定します。
- 特定のエージェントのリモートユーザーを設定する前に、**snmp-serverengineID** グローバル コンフィギュレーション コマンドをオプションとともに使用して、SNMP エンジン ID を設定してください。リモートエージェントの SNMP エンジン ID およびユーザーパスワードを使用して認証およびプライバシーダイジェストが算出されます。先にリモートエンジン ID を設定しておかないと、コンフィギュレーション コマンドがエラーになります。
- SNMP 情報を設定するときには、プロキシ要求または情報の送信先となるリモートエージェントの SNMP エンジン ID を SNMP データベースに設定しておきます。
- ローカルユーザがリモートホストと関連付けられていない場合、デバイスは **auth**（authNoPriv）および **priv**（authPriv）の認証レベルの情報を送信しません。
- SNMP エンジン ID を変更する場合は注意が必要です。（コマンドラインで入力された）ユーザーのパスワードは、パスワードおよびローカルエンジン ID に基づいて、MD5 または SHA セキュリティダイジェストに変換されます。コマンドラインのパスワードは、RFC 2274 の規定に従って廃棄されます。エンジン ID の値を変更すると、SNMPv3 ユーザーのセキュリティダイジェストが無効になります。その後、snmp-server user username グローバル コンフィギュレーション コマンドを使用して SNMP ユーザーを再設定する必要があります。エンジン ID を変更した場合は、同様の制限によってコミュニティ ストリングも再設定する必要があります。
- snmp-server host コマンドをデフォルトの UDP ポート（162）で設定すると、show running-config コマンドの出力に UDP ポート値が表示されません。snmp-server host {host-addr} community-string udp-port value コマンドを使用してデフォルト以外の UDP ポート値を指定すると、UDP ポート番号がコマンド出力に表示されます。デフォルトの UDP ポート 162

を使用しても使用しなくても snmp-server host コマンドを設定できます。ただし、両方を同時に設定することはできません。

正しい例を次に示します。

```
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community udp-port 162
```

次の例は正しくありません。

```
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community
Device(config)# snmp-server host 10.10.10.10 community udp-port 162
Device(config)# snmp-server host 10.10.10.10 community udp-port 163
Device(config)# snmp-server host 10.10.10.10 community udp-port 162
Device(config)# snmp-server host 10.10.10.10 community
```

SNMP グループおよびユーザの設定

はじめる前に

デバイスのローカルまたはリモート SNMP サーバエンジンを表す識別名（エンジン ID）を指定できます。SNMP ユーザーを SNMP ビューにマッピングする、SNMP サーバーグループを設定し、新規ユーザーを SNMP グループに追加します。

このセクションでは、デバイスで SNMP グループとユーザーを設定する方法について説明します。

手順

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： Device> enable	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： Device# **configure terminal**	グローバル コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
ステップ 3：	**snmp-server engineID** {**local** engineid-string | **remote** ip-address [**udp-port** port-number] engineid-string} 例： Device(config)# **snmp-server engineID local 1234**	SNMP のローカル コピーまたはリモート コピーに名前を設定します。 • *engineid-string* は、SNMP のコピー名を指定する 24 文字の ID ストリングです。後続ゼロが含まれる場合は、24 文字のエンジン ID すべてを指定する必要はありません。指定するのは、エンジン ID のうちゼロのみが続く箇所を除いた部分だけです。手順例では、123400000000000000000000 のエンジン ID を設定します。 • **remote** を指定した場合、SNMP のリモートコピーが置かれているデバイスの ip-address を指定し、任意でリモートデバイスのユーザー データグラム プロトコル（UDP）ポートを指定します。デフォルトは 162 です。

	コマンドまたはアクション	目的
ステップ 4	**snmp-server group** *group-name* { **v1** | **v2c** | **v3** { **auth** | **noauth** | **priv**} } [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] 例： Device(config)# **snmp-server group public v2c access lmnop**	リモート デバイス上で新しい SNMP グループを設定します。 *group-name* には、グループの名前を指定します。 次のいずれかのセキュリティ モデルを指定します。 • **v1** は、最も安全性の低いセキュリティ モデルです。 • **v2c** は、2 番目に安全性の低いセキュリティ モデルです。標準の 2 倍の幅で情報および整数を伝送できます。 • **v3**、最も安全な場合には、次の認証レベルの 1 つを選択する必要があります。 **auth**：MD5 および SHA によるパケット認証が可能です。 **noauth**：noAuthNoPriv というセキュリティ レベルをイネーブルにします。キーワードを指定しなかった場合、これがデフォルトです。 **priv**：データ暗号規格（DES）によるパケット暗号化をイネーブルにします（privacy とも呼ばれます）。 （任意）**read** *readview* とともに、エージェントの内容を表示できるビュー名を表す文字列（64 文字以内）を入力します。 （任意）**write** *writeview* とともに、データを入力し、エージェントの内容を表示できるビュー名を表す文字列（64 文字以内）を入力します。 （任意）**notify** *notifyview* とともに、通知、情報、またはトラップを指定するビュー名を表す文字列（64 文字以内）を入力します。 （任意）**access** *access-list* とともに、アクセスリスト名を表す文字列（64 文字以内）を入力します。

	コマンドまたはアクション	目的
ステップ 5	**snmp-server user** username group-name {**remote** host [**udp-port** port]} { **v1** [**access** access-list] | **v2c** [**access** access-list] | **v3** [**encrypted**] [**access** access-list] [**auth** {**md5** | **sha**} auth-password]} [priv {**des** | **3des** | **aes** {**128** | **192** | **256**}} priv-password] 例： Device(config)# **snmp-server user Pat public v2c**	

	コマンドまたはアクション	目的
		SNMP グループに対して新規ユーザを追加します。 *username* は、エージェントに接続するホスト上のユーザ名です。 *group-name* は、ユーザが関連付けられているグループの名前です。 **remote** を入力して、ユーザーが所属するリモート SNMP エンティティおよびそのエンティティのホスト名または IP アドレスを指定し、任意で UDP ポート番号を指定します。デフォルトは 162 です。 SNMP バージョン番号（**v1**、**v2c**、または **v3**）を入力します。**v3** を入力する場合は、次のオプションを追加します。 • **encrypted** は、パスワードを暗号化形式で表示することを指定します。このキーワードは、**v3** キーワードが指定されている場合のみ使用可能です。 • **auth** は認証レベル設定セッションです。HMAC-MD5-96（**md5**）または HMAC-SHA-96（**sha**）のどちらかを指定でき、パスワードストリング *auth-password*（64 文字以下）が必要です。 **v3** を入力すると、次のキーワードを使用して（64 文字以内）、プライベート（**priv**）暗号化アルゴリズムおよびパスワード文字列 *priv-password* を設定することもできます。 • **priv**：User-based Security Model（USM）を指定します。 • **des**：56 ビット DES アルゴリズムの使用を指定します。 • **3des**：168 ビット DES アルゴリズムの使用を指定します。 • **aes**：DES アルゴリズムの使用を指定します。128 ビット暗号化、192 ビット暗号化、または 256 ビット暗号化のいずれかを選択する必要があります。

	コマンドまたはアクション	目的
		（任意）**access** *access-list* とともに、アクセスリスト名を表す文字列（64 文字以内）を入力します。 （注） コンプライアンスシールドが無効になっている場合、**md5**、**des**、**3des** のアルゴリズムは SNMPv3 グループでサポートされません。**crypto engine compliance shield enable** コマンドを使用してコンプライアンスシールドを有効にし、デバイスを再起動して、**md5**、**des**、および **3des** のアルゴリズムを設定する必要があります。
ステップ 6	**end** 例： Device(config)# **end**	特権 EXEC モードに戻ります。
ステップ 7	**show running-config** 例： Device# **show running-config**	入力を確認します。
ステップ 8	**copy running-config startup-config** 例： Device# **copy running-config startup-config**	（任意）コンフィギュレーション ファイルに設定を保存します。

SNMP UDP ポートの開閉

SNMP UDP ポートを開くには、ユーザー EXEC モードで次の手順を実行します。

Procedure

ステップ 1 **enable**

Example:

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **configure terminal**

Example:

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **snmp-server** {**host** | **user** | **community** | **manager**}

Example:

```
Device(config)# snmp-server host
```

SNMP UDP ポート 161 および 162 を開きます。

オプション（**host**、**user**、**community**、**manager**）のいずれかを設定すると、両方のポートが開きます。

ポートを閉じるには、設定したすべてのオプションの **no** 形式を入力します。キーワードが 1 つでも設定されていると、ポートは開いたままになります。

キーワードを指定せずに **no snmp-server** コマンドを入力すると、SNMP UDP ポートだけでなく、SNMP プロセスもシャットダウンされます。

ステップ 4 **end**

Example:

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 5 **show udp**

Example:

```
Device# show udp
```

SNMP UDP ポートを表示します。

必要なコマンドのいずれかが設定されている場合、ポート 161 および 162 は、リモートフィールドの下に値 listen を表示します。

ステップ 6 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

エージェント コンタクトおよびロケーションの設定

SNMP エージェントのシステム接点およびロケーションを設定して、コンフィギュレーション ファイルからこれらの記述にアクセスできるようにするには、次の手順を実行します。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **configure terminal**

例：

configure terminal

例：

```
Device# configure terminal
```

グローバル設定モードを開始します。

ステップ 3 **snmp-server contact** *text*

例：

```
Device(config)# snmp-server contact Dial System Operator at beeper 21555
```

システムの連絡先文字列を設定します。

ステップ 4 **snmp-server location** *text*

例：

```
Device(config)# snmp-server location Building 3/Room 222
```

システムの場所を表す文字列を設定します。

ステップ 5 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 6 **show running-config**

例：

```
Device# show running-config
```

入力を確認します。

ステップ 7 **copy running-config startup-config**

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

SNMP を通して使用する TFTP サーバの制限

SNMP を介したコンフィギュレーション ファイルの保存とロードに使用する TFTP サーバを、アクセス リストで指定されたサーバに限定するには、次の手順を実行します。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2 **configure terminal**

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **snmp-server tftp-server-list** *access-list-number*

例：

```
Device(config)# snmp-server tftp-server-list 44
```

SNMP を介したコンフィギュレーション ファイルのコピーに使用する TFTP サーバを、アクセス リストのサーバに限定します。

access-list-number には、1 ～ 99 および 1300 ～ 1999 の標準 IP アクセス リスト番号を入力します。

ステップ 4 **access-list** *access-list-number* { deny | permit} *source* [*source-wildcard*]

例：

```
Device(config)# access-list 44 permit 10.1.1.2
```

標準アクセス リストを作成し、コマンドを必要な回数だけ実行します。

- *access-list-number* には、ステップ 3 で指定したアクセス リスト番号を入力します。
- **deny** キーワードは、条件が一致した場合にアクセスを拒否します。**permit** キーワードは、条件が一致した場合にアクセスを許可します。
- *source* には、デバイスにアクセスできる TFTP サーバの IP アドレスを入力します。

- （任意）*source-wildcard* には、source に適用されるワイルドカード ビットをドット付き 10 進表記で入力します。無視するビット位置には 1 を設定します。

  アクセス リストの末尾には、すべてに対する暗黙の拒否ステートメントが常に存在します。

ステップ 5 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 6 **show running-config**

例：

```
Device# show running-config
```

入力を確認します。

ステップ 7 **copy running-config startup-config**

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

SNMP エージェントのディセーブル化

SNMP エージェントをディセーブルにするには、次の手順を実行します。

Before you begin

SNMP エージェントをディセーブルにする前にイネーブルにする必要があります。デバイス上で入力した最初の **first snmp-server** グローバル コンフィギュレーション コマンドによって SNMP エージェントがイネーブルになります。

no snmp-server グローバル コンフィギュレーション コマンドは、デバイス上で実行している SNMP エージェントのすべてのバージョン（バージョン 1、バージョン 2C、バージョン 3）をディセーブルにして、SNMP プロセスをシャットダウンします。グローバル コンフィギュレーション モードで、**snmp-server host**、**snmp-server user**、**snmp-server community**、**nmp-server manager** のいずれかのコマンドを入力して、SNMP エージェントのすべてのバージョンを再度イネーブルにできます。特に SNMP をイネーブルにするために指定された Cisco IOS コマンドはありません。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

パスワードを入力します（要求された場合）。

ステップ 2　configure terminal

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3　no snmp-server

例：

```
Device(config)# no snmp-server
```

SNMP エージェント動作をディセーブルにします。

ステップ 4　end

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 5　show running-config

例：

```
Device# show running-config
```

入力を確認します。

ステップ 6　copy running-config startup-config

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

SNMP ステータスのモニタリング

不正なコミュニティ ストリング エントリ、エラー、要求変数の数など、SNMP の入出力統計情報を表示するには、**show snmp** 特権 EXEC コマンドを使用します。また、次の表にリストされたその他の特権 EXEC コマンドを使用して、SNMP 情報を表示することもできます。

Table 5: SNMP 情報を表示するためのコマンド

コマンド	目的
show snmp	SNMP 統計情報を表示します。
show snmp engineID	デバイスに設定されているローカルSNMPエンジンおよびすべてのリモート エンジンに関する情報を表示します。
show snmp group	ネットワーク上の各 SNMP グループに関する情報を表示します。
show snmp pending	保留中の SNMP 要求の情報を表示します。
show snmp sessions	現在の SNMP セッションの情報を表示します。
show snmp user	SNMP ユーザ テーブルの各 SNMP ユーザ名に関する情報を表示します。 **Note** このコマンドは、**auth** | **noauth** | **priv** モードの SNMPv3 設定情報を表示する際に使用する必要があります。この情報は、**show running-config** の出力には表示されません。

SNMP の例

次に、SNMP の全バージョンをイネーブルにする例を示します。この設定では、任意の SNMP マネージャがコミュニティ ストリング *public* を使用して、読み取り専用権限ですべてのオブジェクトにアクセスできます。この設定では、デバイスはトラップを送信しません。

```
Device(config)# snmp-server community public
```

次に、任意の SNMP マネージャがコミュニティストリング *public* を使用して、読み取り専用権限ですべてのオブジェクトにアクセスする例を示します。デバイスはさらに、SNMPv1 を使用してホスト 192.180.1.111 および 192.180.1.33 に、SNMPv2C を使用してホスト 192.180.1.27 に VTP トラップを送信します。コミュニティ ストリング *public* は、トラップとともに送信されます。

```
Device(config)# snmp-server community public
Device(config)# snmp-server enable traps vtp
Device(config)# snmp-server host 192.180.1.27 version 2c public
Device(config)# snmp-server host 192.180.1.111 version 1 public
Device(config)# snmp-server host 192.180.1.33 public
```

次に、*comaccess* コミュニティ ストリングを使用するアクセス リスト 4 のメンバに、すべてのオブジェクトへの読み取り専用アクセスを許可する例を示します。その他の SNMP マネージャは、どのオブジェクトにもアクセスできません。SNMP 認証障害トラップは、SNMPv2C がコミュニティ ストリング *public* を使用してホスト *cisco.com* に送信します。

```
Device(config)# snmp-server community comaccess ro 4
Device(config)# snmp-server enable traps snmp authentication
Device(config)# snmp-server host cisco.com version 2c public
```

次に、エンティティ MIB トラップをホスト *cisco.com* に送信する例を示します。コミュニティストリングは制限されます。1 行目で、デバイスはすでにイネーブルになっているトラップ以外に、エンティティ MIB トラップを送信できるようになります。2行目はこれらのトラップの宛先を指定し、ホスト *cisco.com* に対する以前の **snmp-server** ホストコマンドを無効にします。

```
Device(config)# snmp-server enable traps entity
Device(config)# snmp-server host cisco.com restricted entity
```

次に、コミュニティストリング public を使用して、すべてのトラップをホスト myhost.cisco.com に送信するようにデバイスをイネーブルにする例を示します。

```
Device(config)# snmp-server enable traps
Device(config)# snmp-server host myhost.cisco.com public
```

次に、ユーザとリモート ホストを関連付けて、ユーザがグローバル コンフィギュレーション モードの際に auth（authNoPriv）認証レベルで情報を送信する例を示します。

```
Device(config)# snmp-server engineID remote 192.180.1.27 00000063000100a1c0b4011b
Device(config)# snmp-server group authgroup v3 auth
Device(config)# snmp-server user authuser authgroup remote 192.180.1.27 v3 auth md5
mypassword
Device(config)# snmp-server user authuser authgroup v3 auth md5 mypassword
Device(config)# snmp-server host 192.180.1.27 informs version 3 auth authuser config
Device(config)# snmp-server enable traps
Device(config)# snmp-server inform retries 0
```

次に、SNMPエージェントにポーリングされたSNMPマネージャのエントリを表示する例を示します。

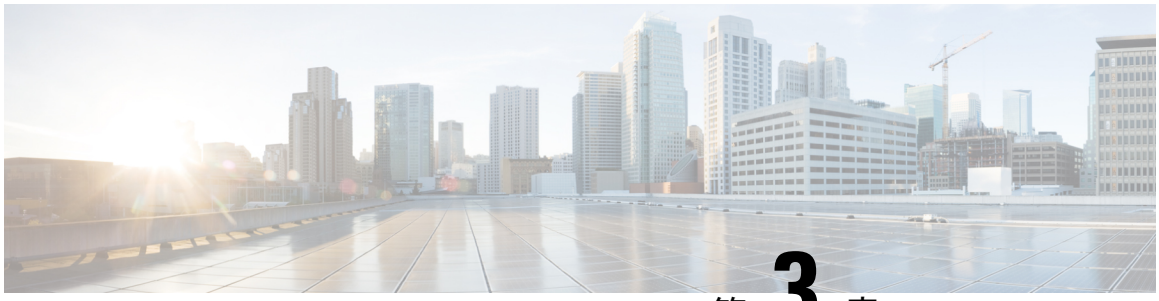
```
Device# show snmp stats host
Request Count Last Timestamp Address
2 00:00:01 ago 3.3.3.3
1 1w2d ago 2.2.2.2
```

次の例は、コンプライアンスシールドが無効になっている場合にSNMPv3 グループで3つのアルゴリズム（**md5**、**des**、**3des**）のいずれかを設定したときにデバイスに表示されるメッセージを示しています。

```
Device(config)# snmp-server user md5user grp v3 auth md5 cisco1234 priv des
Sep 1 00:14:51.582 IST: %SNMP-6-AUTHPROTOCOLMD5: Authentication protocol md5 support
will be deprecated in future
Sep 1 00:14:51.582 IST: %SNMP-6-PRIVPROTOCOLDES: Privacy protocol des support will be
deprecated in future
Sep 1 00:14:51.645 IST: %SNMP-5-WARMSTART: SNMP agent on host Switch is undergoing a
warm start
```

次の例は、コンプライアンスシールドが有効になっている場合にSNMPv3 グループで3つのアルゴリズム（**md5**、**des**、**3des**）のいずれかを設定したときにデバイスに表示されるメッセージを示しています。以下に示すように、暗号化アルゴリズムは警告メッセージとともにサポートされています。

```
Device(config)# snmp-server user md5user grp v3 auth md5 cisco1234
weaker algorithm MD5, DES and 3DES is not allowed for snmp user
```



第 3 章

スイッチドポートアナライザ

スイッチドポートアナライザ（SPAN）は、ネットワークトラフィックをモニタリング、分析、および障害対応するための強力なツールをネットワーク管理者に提供するシスコスイッチの機能です。この機能は、ライブ業務を中断することなく、データフローの優れた可視性を提供することで、ネットワークの正常性を維持し、セキュリティを確保し、パフォーマンスを最適化するのに不可欠です。

- SPAN の概要 （57 ページ）
- SPAN の仕組み （58 ページ）
- SPAN の概念および用語 （59 ページ）
- SPAN と他の機能の相互作用 （64 ページ）
- SPAN とデバイススタック （65 ページ）
- SPAN の制約事項 （65 ページ）
- SPAN の設定方法 （66 ページ）
- SPAN のコンフィギュレーション例 （69 ページ）

SPAN の概要

スイッチドポートアナライザ（SPAN）を使用すると、ネットワーク管理者は、トラフィックのコピーをデバイス上の別のポートに送信することにより、ポートまたは VLAN を通過するネットワークトラフィックを分析できます。この宛て先ポートは、通常、ネットワークアナライザ、その他のモニタリングまたはセキュリティデバイスに接続されています。SPAN は送信元ポート上または送信元 VLAN 上で受信、送信、または送受信されたトラフィックを指定された宛て先ポートにコピー（ミラーリング）して、解析します。SPAN の主な利点は、送信元ポートまたはVLAN上のネットワークトラフィックの通常のスイッチングには影響しません。

SPAN の仕組み

SPAN は、指定されたネットワークの場所から専用のモニタリングポートにトラフィックをミラーリングすることで動作します。次の段階では、SPAN の動作を説明します。

1. 送信元の特定：ネットワーク管理者は、トラフィックをミラーリングするための 1 つ以上の送信元ポートまたはVLANを設定します。これらの送信元には、スイッチに着信するトラフィック（入力）、スイッチから発信されるトラフィック（出力）、またはその両方を含めることができます。

（注） 送信元ポートに出入りするトラフィックや、送信元 VLAN に出入りするトラフィックが監視されます。送信元 VLAN にルーティングされるトラフィックはモニターできません。たとえば、着信トラフィックをモニターしている場合、別の VLAN から送信元 VLAN にルーティングされているトラフィックはモニターできません。ただし、送信元 VLAN で受信し、別の VLAN にルーティングされるトラフィックは、モニターできます。

2. 接続先を指定：管理者は、スイッチで単一の宛て先ポートを指定します。ネットワークアナライザや侵入検知システムなどのモニタリングデバイスがこの宛て先ポートに接続します。

（注） 宛先ポートは SPAN 専用にする必要があります。SPAN セッションに必要なトラフィック以外、宛て先ポートが他のネットワークトラフィックを受信したり転送したりすることはありません。

3. トラフィックミラーリング：スイッチは、設定済みの送信元ポートまたはVLANを通過するすべてのトラフィックを複製します。その後、これらの重複したパケットを指定された宛て先ポートに送信します。元のトラフィックは中断なしで意図したパスを継続します。
4. 分析とアクティブな使用：宛て先ポートに接続されているモニタリングデバイスは、ミラー済みトラフィックをキャプチャして分析します。これにより、ネットワークの動作、アプリケーションのパフォーマンス、および潜在的なセキュリティ脅威に関するインサイトが得られます。
5. トラフィック注入：ネットワーク セキュリティ デバイスからトラフィックを注入する場合、SPAN 宛て先ポートを使用できます。たとえば、Cisco 侵入検知システム（IDS）センサー装置を宛先ポートに接続した場合、IDS デバイスは TCP リセット パケットを送信して、疑わしい攻撃者の TCP セッションを停止させることができます。

SPAN の概念および用語

ローカル SPAN

ローカル SPAN は 1 つのデバイス内の SPAN セッション全体をサポートします。すべての送信元ポートまたは送信元 VLAN、および宛先ポートは、同じデバイスまたはデバイススタック内にあります。ローカル SPAN は、任意の VLAN 上の 1 つまたは複数の送信元ポートからのトラフィック、あるいは1つまたは複数の VLAN からのトラフィックを解析するために宛先ポートへコピーします。

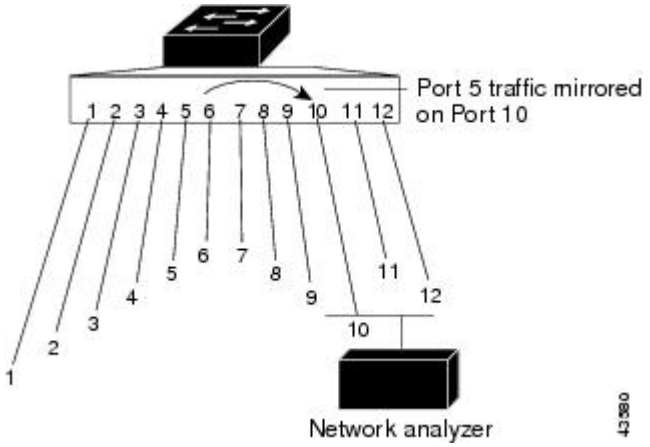
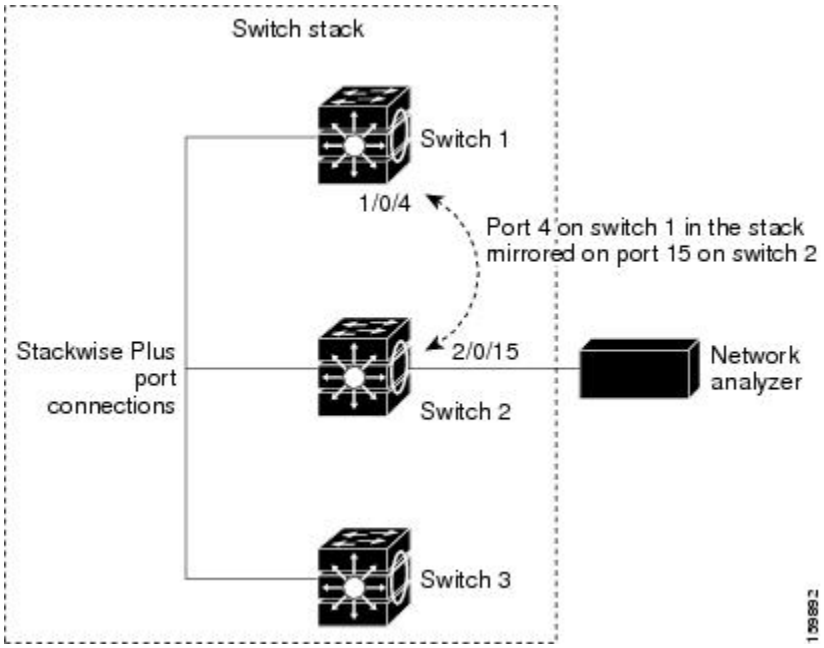


図 *6 :* 単一デバイスでのローカル *SPAN* の設定例

ポート 5（送信元ポート）上のすべてのトラフィックがポート 10（宛先ポート）にミラーリングされます。ポート 10 のネットワーク アナライザは、ポート 5 に物理的には接続されていませんが、ポート 5 からのすべてのネットワーク トラフィックを受信します。

図 *7:* デバイス スタックでのローカル ***SPAN*** の設定例



SPAN セッション

SPAN セッションを使用すると、1 つまたは複数のポート上または VLAN 上でトラフィックをモニターし、そのモニターしたトラフィックを 1 つまたは複数の宛て先ポートに送信できます。ローカル SPAN セッションは、宛て先ポートと送信元ポートまたは送信元 VLAN（すべて単一のネットワークデバイス上に設定されている）を結び付けたものです。これらのセッションでは、指定された入力および出力パケットが収集され、宛て先ポートに誘導される SPAN データのストリームに形成されます。SPAN セッションの主な特性は次のとおりです。

- スイッチド ポートおよびルーテッド ポートはいずれも SPAN 送信元および宛先として設定できます。
- SPAN セッションがデバイスの通常の動作を妨げることはありません。ただし、10 Mbps のポートで 100 Mbps のポートをトラフィック監視するなど、オーバーサブスクライブの SPAN 宛先は、パケットのドロップまたは消失を招くことがあります。
- SPAN が有効な場合、監視中の各パケットは 2 回送信されます（1 回は標準トラフィックとして、もう 1 回は監視されたパケットとして）。多数のポートまたは VLAN を監視すると、大量のネットワークトラフィックが生成されることがあります。
- 無効にされたポートで SPAN セッションを設定できます。ただし、SPAN セッションは、そのセッションの宛て先ポートと、少なくとも1つの送信元ポートまたは VLAN が有効になっている場合にのみアクティブになります。

モニター対象トラフィック

SPAN セッションは、次のトラフィック タイプを監視できます。

- 受信（Rx）SPAN：受信（または入力）SPANは、デバイスが変更または処理を行う前に、送信元インターフェイスまたは VLAN が受信したすべてのパケットをできるだけ多くモニタリングします。送信元が受信した各パケットのコピーがその SPAN セッションに対応する宛先ポートに送られます。
  - Diffserv コード ポイント（DSCP）の変更など、ルーティングや Quality of Service（QoS）が原因で変更されたパケットは、変更される前にコピーされます。
  - 受信処理中にパケットをドロップする可能性のある機能は、入力 SPAN には影響を与えません。宛先ポートは、実際の着信パケットがドロップされた場合でも、パケットのコピーを受信します。パケットをドロップする可能性のある機能は、標準および拡張 IP 入力アクセス コントロール リスト（ACL）、入力 QoS ポリシング、VLAN ACL、および出力 QoS ポリシングです。
- 送信（Tx）SPAN：送信（または出力）SPAN は、デバイスによる変更または処理がすべて実行されたあとに、送信元インターフェイスから送信されたすべてのパケットをできる限り多くモニタリングします。送信元が送信した各パケットのコピーがその SPAN セッションに対応する宛先ポートに送られます。コピーはパケットの変更後に用意されます。
  - ルーティングが原因で変更されたパケット（存続可能時間（TTL）、MAC アドレス、QoS 値の変更など）は、宛先ポートで（変更されて）コピーされます。
  - 送信処理中にパケットをドロップする可能性のある機能は、SPAN 用の複製コピーにも影響します。これらの機能には、標準および拡張 IP 出力 ACL、出力 QoS ポリシングがあります。
- 両方：SPAN セッションで、受信パケットと送信パケットの両方について、ポートまたは VLAN をモニタすることもできます。これはデフォルトです。

デフォルトでは、ローカル SPAN セッションはカプセル化とともに送信元パケットを複製します。

- 送信元ポートと同じカプセル化設定（タグなし、または IEEE 802.1Q）を使用して、パケットが宛先ポートに送信されます。
- BPDU やレイヤ 2 プロトコル パケットを含むすべてのタイプのパケットがモニタされます。

したがって、ローカル SPAN セッションでは、タグなし、および IEEE 802.1Q タグ付きパケットが宛て先ポートに混在することがあります。

デバイスの輻輳により、入力送信元ポート、出力送信元ポート、または SPAN 宛先ポートでパケットがドロップされることがあります。一般に、これらの特性は互いに無関係です。次に例を示します。

- パケットは通常どおり転送されますが、SPAN 宛先ポートのオーバーサブスクライブが原因でモニタされないことがあります。
- 入力パケットが標準転送されないにもかかわらず、SPAN 宛先ポートに着信することがあります。

- デバイスの輻輳が原因でドロップされた出力パケットは、出力 SPAN からもドロップされます。

SPAN の設定によっては、同一送信元のパケットのコピーが複数、SPAN 宛先ポートに送信されます。たとえば、ポート A での RX モニター用とポート B での TX モニター用に双方向（RX と TX）SPAN セッションが設定されているとします。パケットがポート A からデバイスに入ってポート B にスイッチされると、着信パケットも発信パケットも宛先ポートに送信されます。このため、両方のパケットは同じものになります。レイヤ 3 書き換えが行われた場合には、パケット変更のため異なるパケットになります。

送信元ポート

送信元ポート（別名モニター側ポート）は、ネットワーク トラフィック分析のために監視するスイッチド ポートまたはルーテッド ポートです。

1 つのローカル SPAN セッションでは、送信元ポートまたは VLAN のトラフィックを単一方向または双方向でモニターできます。

デバイスは、任意の数の送信元ポート（デバイスで使用可能なポートの最大数まで）および最大 1500 の送信元 VLAN をサポートしています。

送信元ポートの特性は、次のとおりです。

- モニターする方向（入力、出力、または両方）を指定して、各送信元ポートを設定できます。
- すべてのポート タイプ（EtherChannel、ギガビット イーサネットなど）が可能です。
- EtherChannel 送信元の場合は、EtherChannel 全体で、または物理ポートがポート チャネルに含まれている場合は物理ポート上で個別に、トラフィックをモニターできます。
- アクセス ポート、トランク ポート、ルーテッド ポート、または音声 VLAN ポートに指定できます。
- 宛先ポートにすることはできません。
- 送信元ポートは同じ VLAN にあっても異なる VLAN にあってもかまいません。
- 単一セッション内で複数の送信元ポートをモニターすることが可能です。

送信元 VLAN

VLAN ベースの SPAN（VSPAN）では、1 つまたは複数の VLAN のネットワーク トラフィックをモニターできます。VSPAN 内の SPAN 送信元インターフェイスが VLAN ID となり、トラフィックはその VLAN のすべてのポートでモニターされます。

VSPAN には次の特性があります。

- 送信元 VLAN 内のすべてのアクティブ ポートは送信元ポートとして含まれ、単一方向または双方向でモニターできます。

- 指定されたポートでは、モニター対象の VLAN 上のトラフィックのみが宛先ポートに送信されます。
- 宛先ポートが送信元 VLAN に所属する場合は、送信元リストから除外され、モニターされません。
- ポートが送信元 VLAN に追加または削除されると、これらのポートで受信された送信元 VLAN のトラフィックは、モニター中の送信元に追加または削除されます。
- モニターできるのは、イーサネット VLAN だけです。
- セッションあたりの送信元 VLAN の数は 1,500 以下である必要があります。この制限は、受信（RX）および送信（TX）方向の合計です。

宛先ポート

各ローカル SPAN セッションには、送信元ポートおよび VLAN からのトラフィックのコピーを受信し、SPAN パケットをユーザー（通常はネットワーク アナライザ）に送信する宛て先ポート（別名モニター側ポート）が必要です。

宛先ポートの特性は、次のとおりです。

- SPAN セッションには、セッションごとに 1 つの宛先ポートを設定できます。一度に 1 つの SPAN セッションにしか参加できません（ある SPAN セッションの宛先ポートは、別の SPAN セッションの宛先ポートになることはできません）。
- ローカル SPAN セッションの場合、宛先ポートは送信元ポートと同じデバイスまたはデバイススタックに存在している必要があります。
- ポートを SPAN 宛先ポートとして設定すると、元のポート設定が上書きされます。SPAN 宛先設定を削除すると、ポートは以前の設定に戻ります。ポートが SPAN 宛先ポートとして機能している間にポートの設定が変更されると、SPAN 宛先設定が削除されるまで、変更は有効になりません。
- ポートが EtherChannel グループに含まれていた場合、そのポートが宛先ポートとして設定されている間、グループから削除されます。削除されたポートがルーテッドポートであった場合、このポートはルーテッド ポートでなくなります。
- 任意のイーサネット物理ポートにできます。セキュアポートまたは送信元ポートにすることはできません。
- アクティブな場合、着信トラフィックはディセーブルになります。ポートは SPAN セッションに必要なトラフィック以外は送信しません。宛先ポートでは着信トラフィックを学習したり、転送したりしません。
- レイヤ 2 プロトコル（STP、VTP、CDP、DTP、PAgP）のいずれにも参加しません。
- 任意の SPAN セッションの送信元 VLAN に所属する宛先ポートは、送信元リストから除外され、モニターされません。
- デバイスまたはデバイススタックの宛先ポートの最大数は 64 です。

ローカル SPAN の場合、宛て先ポートでの送信元パケットはデフォルトで元のカプセル化（タグなし、ISL、または IEEE802.1Q）で表示されます。したがって、ローカル SPAN セッションの出力に、タグなし、ISL、または IEEE802.1Q タグ付きパケットが混在することがあります。

SPAN と他の機能の相互作用

SPAN は次の機能と相互に作用します。

- ルーティング：SPAN はルーテッド トラフィックを監視しません。VSPAN が監視するのはデバイスに出入りするトラフィックに限られ、VLAN間でルーティングされるトラフィックは監視しません。たとえば、VLAN が受信モニターされ、デバイスが別の VLAN から監視対象 VLAN にトラフィックをルーティングする場合、そのトラフィックは監視されず、SPAN 宛先ポートで受信されません。
- STP：SPAN セッションがアクティブな間、宛て先ポートは STP に参加しません。SPAN セッションが無効になると、宛て先ポートは STP に参加できます。送信元ポートでは、SPAN は STP ステータスに影響を与えません。
- CDP：SPAN セッションがアクティブな間、SPAN 宛先ポートは CDP に参加しません。SPAN セッションがディセーブルになると、ポートは再び CDP に参加します。
- VLAN およびトランキング：送信元ポート、または宛先ポートの VLAN メンバーシップまたはトランクの設定値を、いつでも変更できます。ただし、宛先ポートの VLAN メンバーシップまたはトランクの設定値に対する変更が有効になるのは、SPAN 宛先設定を削除してからです。送信元ポートの VLAN メンバーシップまたはトランクの設定値に対する変更は、ただちに有効になり、対応する SPAN セッションが変更に応じて自動的に調整されます。
- EtherChannel：EtherChannel グループを送信元ポートとして設定できます。グループが SPAN 送信元として設定されている場合、グループ全体が監視されます。
  - 監視対象の EtherChannel グループに物理ポートを追加すると、SPAN 送信元ポート リストに新しいポートが追加されます。監視対象の EtherChannel グループからポートを削除すると、送信元ポート リストからそのポートが自動的に削除されます。
  - EtherChannel グループに所属する物理ポートを SPAN 送信元ポートとして設定することはできません。ただし、EtherChannel グループに含まれる物理ポートを SPAN 宛先として設定した場合、その物理ポートはグループから削除されます。SPAN セッションからそのポートが削除されると、EtherChannel グループに再加入します。EtherChannel グループから削除されたポートは、グループ メンバのままですが、inactive または suspended ステートになります。
  - EtherChannel グループに含まれる物理ポートが宛先ポートであり、その EtherChannel グループが送信元の場合、ポートは EtherChannel グループおよび監視対象ポート リストから削除されます。

- マルチキャストトラフィックを監視できます。出力ポートおよび入力ポートの監視では、未編集のパケットが 1 つだけ SPAN 宛先ポートに送信されます。マルチキャスト パケットの送信回数は反映されません。
- プライベート VLAN ポートは、SPAN 宛先ポートには設定できません。
- セキュア ポートを SPAN 宛先ポートにすることはできません。
- IEEE 802.1x ポートは SPAN 送信元ポートにできます。SPAN 宛先ポート上で IEEE 802.1x を有効にできますが、SPAN宛先としてこのポートを削除するまで、IEEE 802.1x は無効に設定されます。

SPAN とデバイススタック

スイッチのスタックは 1 つの論理スイッチを表すため、ローカル SPAN の送信元ポートおよび宛先ポートは、スタック内の異なるスイッチである場合があります。したがって、スタック内でのスイッチの追加または削除は、ローカル SPAN セッションに影響を及ぼします。スイッチがスタックから削除されると、アクティブ セッションが非アクティブになります。また、スイッチがスタックに追加されると、非アクティブ セッションがアクティブになります。

SPAN の制約事項

SPAN セッションを設定する場合は、次の制限事項に従ってください。

- デバイスは、指定された送信元ポートまたは VLAN からのトラフィックをモニターおよびキャプチャするための送信元セッションとして指定された最大 8 セッションを含む、最大 66 モニタリングセッションをサポートします。
- 同じ SPAN セッション内に送信元ポートと送信元 VLAN を混在させないでください。
- 宛て先ポートを SPAN セッション内の送信元ポートにすることはできません。
- SPAN セッションには複数の宛て先ポートを設定できますが、デバイススタックは最大 64 個の宛て先ポートをサポートします。
- 10 Mbps のポートで 100 Mbps のポートをトラフィック監視するなど、オーバーサブスクライブの SPAN 宛先は、パケットのドロップまたは消失を招くことがあります。
- SPAN 送信元ポートまたは送信元 VLAN を複数の SPAN セッションの一部にすることはできません。
- SPAN セッションには 1 つの宛て先ポートしか設定できず、一意の宛て先ポートを使用する必要があります。
- EtherChannel グループを SPAN 宛て先ポートにすることはできません。
- EtherChannel メンバーを SPAN 送信元ポートにすることはできません。

SPAN の設定方法

SPAN は、指定されたネットワークの場所から専用のモニタリングポートにトラフィックをミラーリングすることで動作します。次の段階では、SPAN の動作を説明します。

1. 送信元の特定：ネットワーク管理者は、トラフィックをミラーリングするための 1 つ以上の送信元ポートまたはVLANを設定します。これらの送信元には、スイッチに着信するトラフィック（入力）、スイッチから発信されるトラフィック（出力）、またはその両方を含めることができます。

   送信元ポートに出入りするトラフィックや、送信元VLANに出入りするトラフィックが監視されます。送信元VLANにルーティングされるトラフィックはモニターできません。たとえば、着信トラフィックをモニターしている場合、別のVLANから送信元VLANにルーティングされているトラフィックはモニターできません。ただし、送信元 VLAN で受信し、別の VLAN にルーティングされるトラフィックは、モニターできます。

2. 接続先を指定：管理者は、スイッチで単一の宛て先ポートを指定します。ネットワークアナライザや侵入検知システムなどのモニタリングデバイスがこの宛て先ポートに接続します。

   宛先ポートは SPAN 専用にする必要があります。SPAN セッションに必要なトラフィック以外、宛て先ポートが他のネットワークトラフィックを受信したり転送したりすることはありません。

3. トラフィックミラーリング：スイッチは、設定済みの送信元ポートまたはVLANを通過するすべてのトラフィックを複製します。その後、これらの重複したパケットを指定された宛て先ポートに送信します。元のトラフィックは中断なしで意図したパスを継続します。

4. 分析とアクティブな使用：宛て先ポートに接続されているモニタリングデバイスは、ミラー済みトラフィックをキャプチャして分析します。これにより、ネットワークの動作、アプリケーションのパフォーマンス、および潜在的なセキュリティ脅威に関するインサイトが得られます。

   トラフィック注入：ネットワーク セキュリティ デバイスからトラフィックを注入する場合、SPAN 宛て先ポートを使用できます。たとえば、Cisco 侵入検知システム（IDS）センサー装置を宛先ポートに接続した場合、IDS デバイスは TCP リセット パケットを送信して、疑わしい攻撃者の TCP セッションを停止させることができます。

ローカル SPAN セッションの作成

SPAN セッションを作成し、送信元（監視対象）ポートまたは VLAN、および宛先（監視側）ポートを指定するには、次の手順を実行します。

手順

ステップ 1 有効

例：

```
Device# configure terminal
```

特権 EXEC モードを有効にします。

ステップ 2 configure terminal

例：

```
Device# configure terminal
```

グローバル設定モードを開始します。

ステップ 3 **no monitor session** {*session_number* | **all** | **local** | **remote**}

例：

```
Device(config)# no monitor session all
```

セッションに対する既存の SPAN 設定を削除します。

- **session_number** の範囲は、1 ～ 66 です。
- **all**：すべての SPAN セッションを削除します。
- **local**：すべてのローカルセッションを削除します。
- **remote**：すべてのリモート SPAN セッションを削除します。

ステップ 4 **monitor session** session_number **source** {**interface** interface-id | **vlan** vlan-id} [**,** | **-**] [**both** | **rx** | **tx**]

例：

```
Device(config)# monitor session 1 source interface gigabitethernet1/0/1
```

SPAN セッションおよび送信元ポート（モニター対象ポート）を指定します。

- **session_number** の範囲は、1 ～ 66 です。
- interface-id には、モニタリングする送信元ポートを指定します。有効なインターフェイスには、物理インターフェイスおよびポートチャネル論理インターフェイス（**port-channel** *port-channel-number*）があります。有効なポートチャネル番号は 1 ～ 48 です。
- **vlan-id** には、監視する送信元 VLAN を指定します。指定できる範囲は 1 ～ 4094 です。

（注）
1 つのセッションに、一連のコマンドで定義された複数の送信元（ポートまたは VLAN）を含めることができます。ただし、1 つのセッション内では送信元ポートと送信元 VLAN を併用できません。

- （任意）**[,|-]**：一連のインターフェイスまたはインターフェイスの範囲を指定します。カンマの前後およびハイフンの前後にスペースを 1 つずつ入力します。
- （任意）**both|rx|tx**：監視するトラフィックの方向を指定します。トラフィックの方向を指定しなかった場合、送信元インターフェイスは送信トラフィックと受信トラフィックの両方を送信します。
  - both：受信トラフィックと送信トラフィックの両方をモニターします。
  - rx：受信トラフィックをモニターします。
  - tx：送信トラフィックをモニターします。

（注）
monitor session *session_number***source** コマンドを複数回使用すると、複数の送信元ポートを設定できます。

ステップ 5 **monitor session** session_number **destination** {**interface** interface-id **[,** | **-]**

例：

```
Device(config)# monitor session 1 destination interface gigabitethernet1/0/2
```

SPAN セッションおよび宛先ポート（モニター側ポート）を指定します。設定変更が有効になると、ポートの LED がオレンジ色に変わります。LED は SPAN 宛先の設定を削除した後にのみ、元の状態（緑色）に戻ります。

- ローカル SPAN の場合は、送信元および宛先インターフェイスに同じセッション番号を使用する必要があります。
- **session_number** には、ステップ 4 で入力したセッション番号を指定します。
- **interface-id** には、宛て先ポートを指定します。宛先インターフェイスには物理ポートを指定する必要があります。EtherChannel や VLAN は指定できません。

  （任意）**[,|-]**：一連のインターフェイスまたはインターフェイスの範囲を指定します。カンマの前後およびハイフンの前後にスペースを 1 つずつ入力します。

ステップ 6 end

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 7 show running-config

例：

```
Device# show running-config
```

入力を確認します。

ステップ 8 copy running-config startup-config

例：

```
Device# copy running-config startup-config
```

（任意）コンフィギュレーション ファイルに設定を保存します。

SPAN のコンフィギュレーション例

次のセクションに SPAN の設定例を示します。

次に、SPAN セッション 1 を設定し、宛先ポートへ向けた送信元ポートのトラフィックをモニタする例を示します。最初に、セッション 1 の既存の SPAN 設定を削除し、カプセル化方式を維持しながら、双方向トラフィックを送信元ポート GigabitEthernet 1 から宛先ポート GigabitEthernet 2 にミラーリングします。

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1
Device(config)# monitor session 1 source interface gigabitethernet1/0/1
Device(config)# monitor session 1 destination interface gigabitethernet1/0/2
Device(config)# end
```

次に、SPAN セッション 1 の SPAN 送信元としてのポート 1 を削除する例を示します。

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1 source interface gigabitethernet1/0/1
Device(config)# end
```

次に、双方向モニタが設定されていたポート 1 で、受信トラフィックのモニタをディセーブルにする例を示します。

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 1 source interface gigabitethernet1/0/1 rx
```

ポート 1 で受信するトラフィックのモニタはディセーブルになりますが、このポートから送信されるトラフィックは引き続きモニタされます。

次に、SPAN セッション 2 内の既存の設定を削除し、VLAN 1 ～ 3 に属するすべてのポートで受信トラフィックをモニタするように SPAN セッション 2 を設定し、モニタされたトラフィックを宛先ポート GigabitEthernet 2 に送信する例を示します。さらに、この設定は VLAN 10 に属するすべてのポートですべてのトラフィックをモニタするよう変更されます。

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source vlan 1 - 3 rx

Device(config)# monitor session 2 destination interface gigabitethernet1/0/2
Device(config)# monitor session 2 source vlan 10
Device(config)# end
```

次に、SPAN セッション 2 内の既存の設定を削除し、Gigabit Ethernet トランクポート 2 で受信トラフィックをモニターするように SPAN セッション 2 を設定する例を示します。

```
Device> enable
Device# configure terminal
Device(config)# no monitor session 2
Device(config)# monitor session 2 source interface gigabitethernet1/0/2 rx
Device(config)# monitor session 2 destination interface gigabitethernet1/0/1
Device(config)# end
```

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