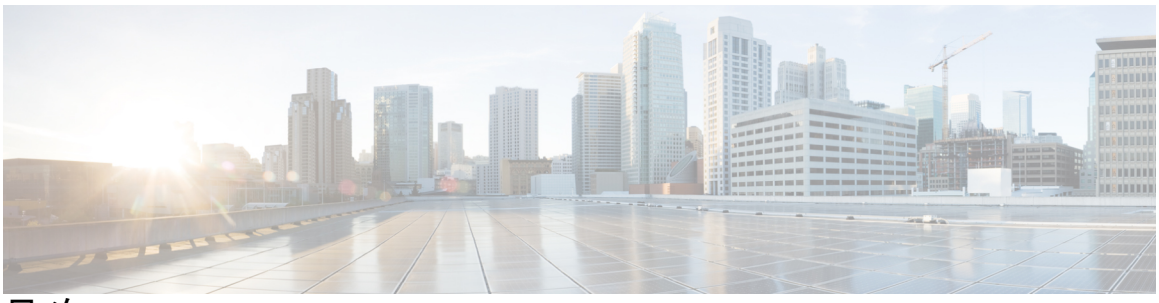
Cisco IOS XE 17.x システム管理コンフィグレーション ガイド

初版：2023 年 8 月 21 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/



目 次

はじめに

ここでは、このマニュアルの対象読者、構成、および表記法について説明します。また、他のマニュアルの入手方法についても説明します。

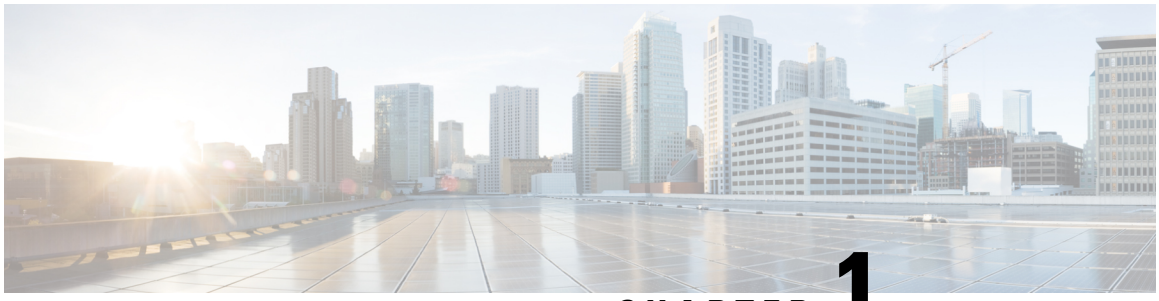
この前書きは、次の項で構成されています。

ここに参照前文マップ

はじめに

第 部

基本的なシステム管理



CHAPTER 1

基本的なシステム管理の実行

このモジュールでは、Cisco IOS ソフトウェアの一般的なシステム機能（つまり、通常は特定のプロトコルに固有でない機能）を管理するために実行できる基本的な管理作業について説明します。

基本的なシステム管理の実行について

システム名

システム名（ホスト名とも呼ばれます）は、ネットワーク内のシステムを一意に識別するために使用します。システム名はCLIプロンプトに表示されます。名前を設定していない場合は、システムのデフォルト名である Router になります。

コマンド エイリアス

コマンドエイリアスを使用して、コマンドの代替構文を設定できます。よく使用するコマンドや複雑なコマンドのエイリアスを作成することもできます。たとえば、エイリアス **save config** を **copy running-config startup-config** コマンドに割り当てると、タイプ量を減らすことができます。また、ユーザーにとって **save config** コマンドの方が覚えやすいはずです。自分またはユーザコミュニティのためにコマンド構文を調整する場合は、単語の置換または省略形を使用します。

設定するすべてのエイリアスはシステム上だけでイネーブルになること、および元のコマンド構文は設定ファイル内に表示されることに注意してください。

マイナー サービス

マイナー サービスは、ルーティング デバイス上で稼働する小規模なサービスであり、基本的なシステムテストおよび基本的なネットワーク機能の提供において役立ちます。マイナーサービスは、ネットワーク上の別のホストから接続テストを行う場合に便利です。

シスコのスモール サーバは、概念的にはデーモンと同じです。

Cisco IOS ソフトウェアベースのデバイスによって提供されるスモール サーバには、TCP、UDP、HTTP、ブートストラップ プロトコル（BOOTP）、Finger などがあります。HTTP サーバについては、『Cisco IOS Configuration Fundamentals Configuration Guide』の「Using the Cisco Web Browser User Interface」の章を参照してください。

TCP スモール サーバは、次のマイナー サービスを提供します。

- chargen：ASCII データのストリームを生成します。このサービスをテストするには、リモートホストから **telnet** *a.b.c.d* **chargen** コマンドを発行します。
- daytime：ネットワーク タイム プロトコル（NTP）が設定されている場合、または日付と時刻が手動で設定されている場合に、システムの日付と時刻を返します。このサービスをテストするには、リモートホストから **telnet** *a.b.c.d* **daytime** コマンドを発行します。
- discard：入力内容をすべて破棄します。このサービスをテストするには、リモートホストから **telnet** *a.b.c.d* **discard** コマンドを発行します。
- echo：すべての入力内容をエコーバックします。このサービスをテストするには、リモートホストから **telnet** *a.b.c.d* **echo** コマンドを発行します。

UDP スモール サーバは、次のマイナー サービスを提供します。

- chargen：送信されたデータグラムを廃棄し、CR+LF（復帰と改行）で終端された 72 文字の ASCII 文字列で応答します。
- discard：送信されたデータグラムを破棄します。
- echo：送信されたデータグラムのペイロードをエコーします。

マイナー サービスはデフォルトで無効になっています。

Caution マイナー サービスをイネーブルにすると、特定のタイプのサービス拒絶（DoS）攻撃（UDP 診断ポート攻撃など）が発生する可能性が生まれます。したがって、UDP、TCP、BOOTP または Finger サービスを提供するすべてのネットワーク デバイスをファイアウォールで保護するか、これらのマイナー サービスをディセーブルにしておく必要があります。UDP 診断ポート攻撃の防止については、Cisco.com で入手できる「Defining Strategies to Protect Against UDP Diagnostic Port Denial of Service Attacks」というタイトルのホワイトペーパーを参照してください。

BOOTP サーバ

ルーティング デバイスの非同期回線ブートストラップ プロトコル（BOOTP）サービスをイネーブルまたはディセーブルにできます。このスモール サーバは、デフォルトでイネーブルになっています。セキュリティ上の考慮事項により、このサービスを使用しない場合はディセーブルにしておく必要があります。

DHCP は BOOTP に基づいているため、これらのサービスは、（インターネット標準および RFC に準拠して）ウェルノウン UDP サーバ ポート 67 を共有します。Cisco IOS ソフトウェアにおける DHCP コンフィギュレーションの詳細については、『Cisco IOS IP Addressing Configuration Guide』を参照してください。BOOTP の詳細については、RFC 951 を参照してください。BOOTP と DHCP の相互運用性は、RFC 1534 で規定されています。DHCP は、RFC 2131 で規定されています。

Finger プロトコル

Fingerプロトコルを使用すると、ネットワーク全体のユーザは、現在特定のルーティング デバイスを使用しているユーザのリストを取得できます。表示される情報には、システムで稼働しているプロセス、回線番号、接続名、アイドル時間、終端位置などがあります。この情報は、Cisco IOS ソフトウェアの **show users** EXEC コマンドを通じて提供されます。

Telnet アドレスの非表示化

Telnet セッションの確立を試行する間、アドレスを非表示にできます。非表示機能によって、アドレスの表示が抑制され、接続の試行時に通常表示されるその他のすべてのメッセージ（接続に失敗した場合の詳細なエラー メッセージなど）は引き続き表示されます。

EXEC 起動遅延

ノイズの多い回線での EXEC プロセスの起動を、回線がアイドルになるまで 3 秒間遅らせるには、グローバル コンフィギュレーション モードで **service exec-wait** コマンドを使用します。

このコマンドはモデム回線のノイズが多い場合や、回線に接続されたモデムで Microcom Networking Protocol（MNP）または V.42 ネゴシエーションを無視するように設定されている場合や、MNP または V.42 モデムがダイヤル イン方式で使用されている場合に便利です。これらのケースでは、ノイズや MNP/V.42 パケットが誤ってユーザ名やパスワードとして認識され、ユーザがユーザ名とパスワードを入力する前に認証に失敗する可能性があります。このコマンドは、非モデム回線や、ログインが設定されていない回線には役立ちません。

アイドル Telnet 接続

通常、現在は使用中でない Telnet 接続に送信されたデータは、受け入れ後に廃棄されます。**service telnet-zero-idle** コマンドがイネーブルになっていてセッションが中断状態になると（つまり、他の接続がアクティブになると）、TCP ウィンドウが 0 に設定されます。この処理により、リモート ホストは、接続が再開されるまでデータを送信できません。このコマンドは、ホストから送信されたすべてのメッセージをユーザに表示する必要があり、ユーザが複数のセッ

ションを使用する可能性がある場合に使用します。ホストが最終的にタイムアウトし、ウィンドウの値が0のTCPユーザがログアウトする場合は、このコマンドを使用しないでください。

負荷データの間隔

一連のデータを負荷統計情報の計算に使用する期間を変更できます。ダイヤルバックアップなどの決定は、この統計情報に基づいて行われます。負荷間隔の値を減らすと、平均統計情報の計算期間が短くなり、トラフィックのバーストに対する応答性が高まります。

TCP トランザクションの数

標準のTCP実装を使用してマシン間のキーストロークを送信する場合、TCPは入力されたキーストロークごとに1パケットを送信する傾向があります。これにより、帯域幅を使い果たし、大規模ネットワークにおける輻輳の一因となる可能性があります。

John Nagle のアルゴリズム（RFC 896）は、TCP の小さなパケットの問題を軽減するうえで役立ちます。接続の確立後に入力された最初の文字は単一のパケットで送信されますが、TCP では、受信者が直前のパケットを確認するまで、その後入力されたすべての文字が保持されます。次に、より大きな2番めのパケットが送信され、確認応答が返信されるまで、その後入力されたすべての文字が保存されます。その効果は、文字をより大きなチャンクに蓄積し、任意の接続のラウンドトリップ時間と一致する速度にネットワークへの伝送速度を調整することです。この方法は、通常すべての TCP ベース トラフィックに推奨されます。

デフォルトでは、Nagle アルゴリズムはイネーブルになっていません。

スイッチングおよびスケジューリング プラオリティ

ネットワーク サーバの正常な動作では、スイッチング動作に必要なだけ中央処理装置を使用することが許容されます。プロセッサがルーティング プロトコルを処理する時間を許容しない、異常に重い負荷がネットワーク上で実行されている場合には、状況に応じて、システム プロセス スケジューラにプライオリティを与える必要があります。

システム バッファ サイズ

バッファ プールの初期設定および一時バッファを作成および破棄する際の制限値を調整できます。

通常のシステム運用においては、パブリック バッファ プールとインターフェイス バッファ プールの 2 つがあります。これらは、次のように動作します。

- パブリック プール内のバッファは、要求に基づいて拡大および収縮します。一部のパブリック プールは一時的なものであり、必要に応じて作成および破棄されます。その他のパブリック プールは永続的に割り当てられているため、破棄できません。パブリック バッファ プールには、small、middle、big、very big、large、および huge のラベルが付けられています。

- インターフェイス プールは静的です。つまり、すべて永続的です。インターフェイスごとに 1 つのインターフェイス プールが存在します。たとえば、Cisco 4000 1E 4T 構成には、1 つのイーサネット バッファ プールと 4 つのシリアル バッファ プールがあります。

サーバには、キューイング エレメントのプールが 1 つと、異なるサイズのパケット バッファのパブリック プールが 6 つあります。サーバは、プールごとに、未処理のバッファの数、フリー リスト上のバッファの数、およびフリー リストに許容される最大バッファ数を記録しています。

基本的なシステム管理の実行方法

基本的なシステム パラメータの設定

基本的なシステム パラメータを設定するには、次の手順を実行します。次の手順は、システムのカスタマイズ要件に応じて実行できます。

SUMMARY STEPS

1. **hostname** *name*
2. **prompt** *string*
3. **alias** *mode alias-name alias-command-line*
4. **service tcp-small-servers**
5. **service udp-small-servers**
6. **no ip bootp server**
7. **ip finger**
8. **ip finger rfc-compliant**
9. **service hide-telnet-address**
10. **line** *line-number*
11. exit
12. **exit**
13. **busy-message** *hostname message*
14. **service exec-wait**
15. **service telnet-zero-idle**
16. **load-interval** *seconds*
17. **service nagle**
18. **scheduler interval** *milliseconds*
19. **scheduler allocate** [*network-microseconds process-microseconds*]
20. **scheduler process-watchdog** {**hang** | **normal** | **reload** | **terminate**}
21. **buffers** {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number*
22. exit
23. **show aliases** [*mode*]
24. **show buffers**

DETAILED STEPS

ステップ 1 **hostname** *name*

デバイスに名前を割り当てる基本的なシステム管理作業を実行するには、**hostname** *name* コマンドを使用します。

Example:

```
Router(config)# hostname host1
```

ステップ 2 **prompt** *string*

または

no service prompt config

デフォルトでは、CLIプロンプトは、システム名とそれに続く山カッコ（>）（ユーザEXECモードの場合）またはポンド記号（#）（特権EXECモードの場合）で構成されます。システムのCLIプロンプトをカスタマイズするには、**prompt** *string* または **no service prompt config** コマンドを使用します。

Example:

```
Router(config)# prompt Router123
```

または

Example:

```
Router(config)# no service prompt config
```

ステップ 3 **alias** *mode alias-name alias-command-line*

コマンドエイリアスを作成するには、**alias** *mode alias-name alias-command-line* コマンドを使用します。

Example:

```
Router(config)# alias exec save config copy running-config startup-config
```

ステップ 4 **service tcp-small-servers**

Chargen、Daytime、Discard、Echo などのマイナー TCP サービスをイネーブルにするには、**service tcp-small-servers** コマンドを使用します。

Note これらの基本的サービスがディセーブルになっている場合はコンフィギュレーション ファイルに **service tcp-small-servers** コマンドの **no** 形式が表示されます。

Example:

```
Router(config)# service tcp-small-servers
```

ステップ 5 **service udp-small-servers**

Chargen、Daytime、Discard、Echo などのマイナー UDP サービスをイネーブルにするには、**service udp-small-servers** コマンドを使用します。

Note これらの基本的サービスがディセーブルになっている場合はコンフィギュレーション ファイルに **service udp-small-servers** コマンドの **no** 形式が表示されます。

Example:

```
Router(config)# service udp-small-servers
```

ステップ 6 **no ip bootp server**

プラットフォーム上の BOOTP サーバーをディセーブルにするには、**no ip bootp server** コマンドを使用します。08-12-2016 01:25

Example:

```
Router(config)# no ip bootp server
```

ステップ 7 **ip finger**

シスコデバイスが Finger（ポート 79）要求に応答できるようにするには、**ip finger** コマンドを使用します。**ip finger** コマンドが設定されている場合、ルータはリモートホストからの **telnet** *a.b.c.d* **finger** コマンドに応答し、**show users** コマンドの出力をただちに表示して接続を閉じます。

Example:

```
Router(config)# ip finger
```

ステップ 8 **ip finger rfc-compliant**

RFC 1288 に準拠するように Finger プロトコルを設定するには、**ip finger rfc-compliant** コマンドを使用します。20 人以上のユーザーが同時にログインするデバイスでは **ip finger rfc-compliant** コマンドを設定しないようにしてください。**ip finger rfc-compliant** コマンドを設定すると、ルータは、情報を表示する前に、入力を待ちます。その後、リモートユーザーは Return キーを押して **show users** コマンドの出力を表示したり、**/W** を入力して **show users wide** コマンドの出力を表示したりできます。この情報が表示されたら、接続が閉じます。

Example:

```
Router(config)# ip finger rfc-compliant
```

ステップ 9 **service hide-telnet-address**

Telnet アドレスを表示しないようにルータを設定するには、**service hide-telnet-address** コマンドを使用します。

Example:

```
Router(config)# service hide-telnet-address
```

ステップ 10 **line** *line-number*

ライン コンフィギュレーション モードを開始するには、line コマンドを使用します。

Example:

```
Router(config)# line 1
```

ステップ 11 exit

ライン コンフィギュレーション モードを終了してグローバル コンフィギュレーション モードに戻るには、**exit** コマンドを使用します。

Example:

```
Router(config-line)# exit
```

ステップ 12 **exit**

ライン コンフィギュレーション モードを終了してグローバル コンフィギュレーション モードに戻るには、**exit** コマンドを使用します。

Example:

```
Router(config-line)# exit
```

ステップ 13 **busy-message** *hostname message*

Telnet 接続の試行中に表示される情報をカスタマイズするには、**busy-message** コマンドを **service hide-telnet-address** コマンドとともに使用します。接続試行が失敗した場合、ルータではアドレスが抑制され、**busy-message** コマンドで指定されたメッセージが表示されます。

Example:

```
Router(config)# busy-message host1 message1
```

ステップ 14 **service exec-wait**

ノイズの多い回線でのEXECプロセスの起動を、回線がアイドルになってから3秒後まで遅らせるには、**service exec-wait** コマンドを使用します。

Example:

```
Router(config)# service exec-wait
```

ステップ 15 **service telnet-zero-idle**

Telnet 接続がアイドル状態のときに TCP ウィンドウをゼロに設定するように Cisco IOS ソフトウェアを設定するには、**service telnet-zero-idle** コマンドを使用します。

Example:

```
Router(config)# service telnet-zero-idle
```

ステップ 16 **load-interval** *seconds*

負荷統計情報の計算にデータセットを使用する時間の長さを変更するには、**load-interval** *seconds* コマンドを使用します。

Example:

```
Router(config)# load-interval 100
```

ステップ 17 **service nagle**

Nagleアルゴリズムをイネーブルにして TCP トランザクション数を減らすには、**service nagle** コマンドを使用します。

Example:

```
Router(config)# load-interval 100
```

ステップ 18 **scheduler interval** *milliseconds*

優先度が最も低いシステムプロセスの実行を停止できる最大時間を定義するには、**scheduler interval** *milliseconds* コマンドを使用します。

Example:

```
Router(config)# scheduler interval 100
```

ステップ 19 **scheduler allocate** [*network-microseconds process-microseconds*]

Cisco 7200 シリーズおよび Cisco 7500 シリーズ ルータで CPU が高速スイッチングとプロセスレベルの処理に費やす時間を変更するには、**scheduler allocate** コマンドを使用します。

Caution **scheduler allocate** コマンドのデフォルト値は変更しないようにすることを推奨します。

Example:

```
Router(config)# scheduler allocate 5000 200
```

ステップ 20 **scheduler process-watchdog** {**hang** | **normal** | **reload** | **terminate**}

scheduler process-watchdog {**hang** | **normal** | **reload** | **terminate**} コマンドを使用して、ループプロセスの特性を設定します。

Example:

```
Router(config)# scheduler process-watchdog hang
```

ステップ 21 **buffers** {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number*

buffers {**small** | **middle** | **big** | **verybig** | **large** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number* コマンドを使用して、システムバッファサイズを調整します。

Example:

```
Router(config)# buffers small permanent 10
```

Caution ただし、これらのパラメータを調整することは推奨されません。不適切な設定は、システムのパフォーマンスに悪影響を及ぼします。

ステップ 22 exit

グローバル コンフィギュレーション モードを終了して特権 EXEC モードに戻るには、**exit** コマンドを使用します。

Example:

```
Router(config)# exit
```

ステップ 23 **show aliases** [*mode*]

システムで現在設定されているコマンドエイリアスのリストと、それらのエイリアスの元のコマンド構文を表示するには、**show aliases** [*mode*] コマンドを使用します。

Example:

```
Router# show aliases exec
```

ステップ 24 **show buffers**

バッファ情報を表示するには、**show buffers** コマンドを使用します。このコマンドの詳細については、『Cisco IOS Configuration Fundamentals Command Reference』を参照してください。

Example:

```
Router# show buffers
Buffer elements:
     1119 in free list (1119 max allowed)
     641606 hits, 0 misses, 619 created
Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
     48 in free list (20 min, 150 max allowed)
     2976557 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Middle buffers, 600 bytes (total 25, permanent 25, peak 37 @ 2w0d):
     25 in free list (10 min, 150 max allowed)
     445110 hits, 4 misses, 12 trims, 12 created
     0 failures (0 no memory)
Big buffers, 1536 bytes (total 50, permanent 50):
     50 in free list (5 min, 150 max allowed)
     58004 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
VeryBig buffers, 4520 bytes (total 10, permanent 10):
     10 in free list (0 min, 100 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Large buffers, 5024 bytes (total 0, permanent 0):
     0 in free list (0 min, 10 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Huge buffers, 18024 bytes (total 0, permanent 0):
     0 in free list (0 min, 4 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Interface buffer pools:
Syslog ED Pool buffers, 600 bytes (total 282, permanent 282):
     257 in free list (282 min, 282 max allowed)
     32 hits, 0 misses
IPC buffers, 4096 bytes (total 2, permanent 2):
     1 in free list (1 min, 8 max allowed)
     1 hits, 0 fallbacks, 0 trims, 0 created
     0 failures (0 no memory)
Header pools:
Header buffers, 0 bytes (total 511, permanent 256, peak 511 @ 2w0d):
     255 in free list (256 min, 1024 max allowed)
     171 hits, 85 misses, 0 trims, 255 created
     0 failures (0 no memory)
     256 max cache size, 256 in cache
```

```
     0 hits in cache, 0 misses in cache
Particle Clones:
     1024 clones, 0 hits, 0 misses
Public particle pools:
F/S buffers, 128 bytes (total 512, permanent 512):
     0 in free list (0 min, 512 max allowed)
     512 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
     512 max cache size, 512 in cache
     0 hits in cache, 0 misses in cache
Normal buffers, 512 bytes (total 2048, permanent 2048):
     2048 in free list (1024 min, 4096 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Private particle pools:
HQF buffers, 0 bytes (total 2000, permanent 2000):
     2000 in free list (500 min, 2000 max allowed)
     0 hits, 0 misses, 0 trims, 0 created
     0 failures (0 no memory)
Serial2/0 buffers, 512 bytes (total 256, permanent 256):
     0 in free list (0 min, 256 max allowed)
     256 hits, 0 fallbacks
     256 max cache size, 132 in cache
     124 hits in cache, 0 misses in cache
     10 buffer threshold, 0 threshold transitions
Serial2/1 buffers, 512 bytes (total 256, permanent 256):
     0 in free list (0 min, 256 max allowed)
     256 hits, 0 fallbacks
     256 max cache size, 132 in cache
     124 hits in cache, 0 misses in cache
     10 buffer threshold, 0 threshold transitions
```

基本的なシステム管理の実行の設定例

基本的なシステム管理の実行に関連する設定例はありません。

その他の参考資料

関連資料

関連項目	マニュアル タイトル
ネットワーク管理コマンド	『*Cisco IOS Network Management Command Reference*』
Cisco IOS の基本設定コマンド	*Cisco IOS Configuration Fundamentals Command Reference*
Cisco IOS の基本設定	『*Cisco IOS Configuration Fundamentals Configuration Guide*』
UDP 診断ポートへの攻撃の防止	Defining Strategies to Protect Against UDP Diagnostic Port Denial of Service Attacks

関連項目	マニュアル タイトル
DHCP の設定	『*Cisco IOS IP Addressing Configuration Guide*』

標準

標準	タイトル
なし	--

MIB

MIB	MIB のリンク
なし	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャセットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
RFC 896	『*Congestion Control in IP/TCP Internetworks*』
RFC 951	『*Algorithms for Synchronizing Network Clocks*』
RFC 1288	『*The Finger User Information Protocol*』
RFC 1534	『*Interoperation Between DHCP and BOOTP*』
RFC 2131	『*Dynamic Host Configuration Protocol*』

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

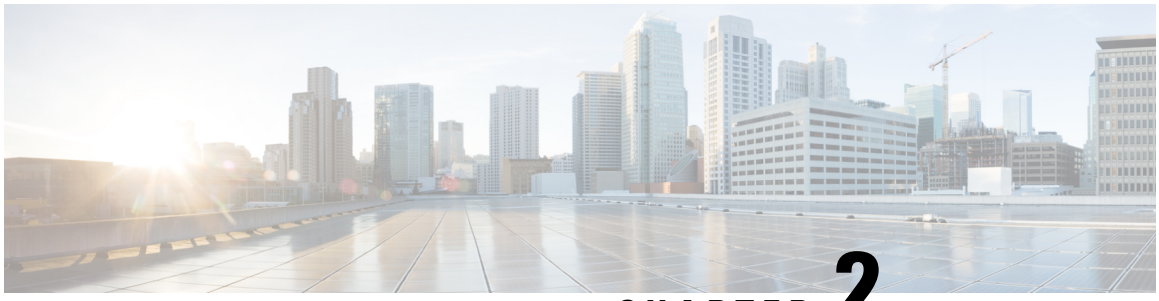
基本的なシステム管理の実行の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 1: 基本的なシステム管理の実行の機能情報

機能名	リリース	機能情報
基本的なシステム管理の実行		このモジュールでは、Cisco IOS ソフトウェアの一般的なシステム機能を管理するための基本的作業について説明します。



CHAPTER 2

メモリしきい値通知

メモリしきい値通知機能を使用すると、重要な通知のためにメモリを予約し、使用可能なメモリが指定したしきい値を下回ると通知を行うようルータを設定できます。

メモリしきい値通知について

メモリしきい値通知機能は、空きメモリが設定されたしきい値を下回っていることを示す通知を送信する方法と、重要な通知を行うために十分なメモリが使用できるようにメモリを予約する方法の2つの方法でルータ上のメモリ不足状態を軽減します。メモリしきい値通知機能を実装するには、次の概念を理解しておく必要があります。

メモリしきい値通知

メモリしきい値通知機能を使用すると、重要な通知のためにメモリを予約し、使用可能なメモリが指定したしきい値を下回ると通知を行うようルータを設定できます。

メモリ予約

重要な動作のメモリ予約によって、イベント ロギングなどの管理プロセスがルータ メモリが少なくなっても機能を続行できるようにします。

メモリしきい値通知の定義方法

空きメモリ不足しきい値の設定

空きメモリ不足しきい値を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **memory free low-watermark** [**processor** *threshold*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**memory free low-watermark** [**processor** *threshold* **Example:** `Router(config)# memory free low-watermark processor 20000`	空きプロセッサ メモリのしきい値を KB 単位で指定します。メモリしきい値に使用できる値を表示するには、次のコマンドを入力します。 • **memory free low-watermark processor ?**

重要な通知のためのメモリの予約

ルータがプロセスによって過負荷になると、使用可能なメモリの量が重要な通知を行うのに十分でないレベルまで落ち込む場合があります。ルータが重要な通知を行う際に使用するメモリ領域を予約するには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **memory reserve critical** *kilobytes*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**memory reserve critical** *kilobytes* **Example:** `Router(config)# memory reserve critical 1000`	ルータが重要な通知を行えるよう、キロバイトで指定したメモリの量を予約します。 • 重要な通知のために予約できるメモリの量は、使用可能なメモリ合計の25%以上にはできません。

メモリしきい値通知の設定例

空きメモリ不足しきい値の設定：例

空きプロセッサメモリのしきい値

次の例では、空きプロセッサ メモリ 20000 KB を、ルータが通知を行うしきい値に指定します。

```
Router(config)# memory free low-watermark processor 20000
```

使用可能な空きメモリが指定したしきい値を下回ると、ルータが次のような通知メッセージを送信します。

```
000029: *Aug 12 22:31:19.559: %SYS-4-FREEMEMLOW: Free Memory has dropped below 20000k
Pool: Processor  Free: 66814056  freemem_lwm: 204800000
```

使用可能な空きメモリがしきい値を5%上回ると、ルータが次のような通知メッセージを送信します。

```
000032: *Aug 12 22:33:29.411: %SYS-5-FREEMEMRECOVER: Free Memory has recovered 20000k
Pool: Processor  Free: 66813960  freemem_lwm: 0
```

重要な通知のためのメモリの予約：例

次の例では、重要な通知用にメモリを 1000 KB 予約します。

```
Router# memory reserved critical 1000
```

Note 重要な通知のために予約できるメモリの量は、使用可能なメモリ合計の 25% 以上にはできません。

その他の参考資料

CPU しきい値処理通知機能の詳細情報については、次の関連資料を参照してください。

関連資料

関連項目	マニュアル タイトル
SNMP トラップ	『*Configuration Fundamentals Command Reference*』

標準

標準	タイトル
この機能では、新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	--

MIB

MIB	MIB のリンク
CISCO-PROCESS-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャセットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
この機能では、新しい RFC または変更された RFC はサポートされていません。また、既存の RFC に対するサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

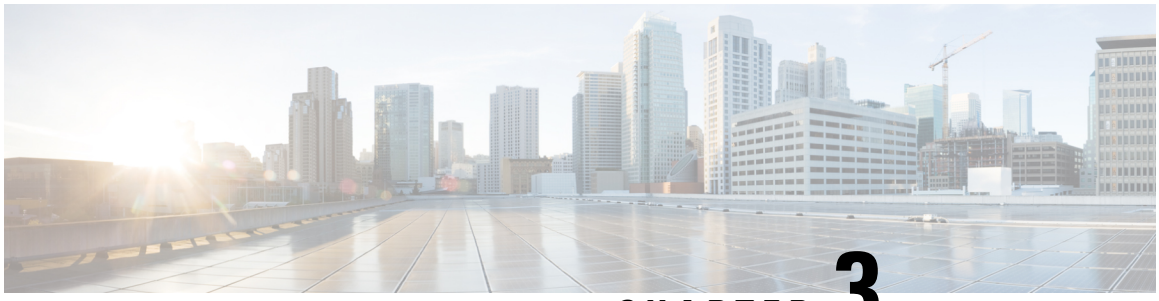
メモリしきい値通知の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェア イメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 2: メモリしきい値通知の機能情報

機能名	リリース	機能情報
メモリしきい値通知	Cisco IOS XE リリース 2.1	この機能は、Cisco ASR 1000 シリーズのアグリゲーション サービス ルータで導入されました。



CHAPTER 3

NTPv4 MIB

NTPv4 MIB 機能はシスコ ソフトウェアに Network Time Protocol バージョン 4（NTPv4）MIB を導入します。この機能では、NTP エンティティの現在のステータスを表すデータ オブジェクトが定義されています。これらのデータ オブジェクトへのアクセスには Simple Network Management Protocol（SNMP）が使用され、これらのデータ オブジェクトはローカル NTP エンティティの監視と管理に使用されます。

このモジュールでは、NTPv4 MIB について説明します。

NTPv4 MIB について

NTPv4 MIB

Network Time Protocol バージョン 4（NTPv4）MIB 機能では、RFC 5907 に基づき、NTP エンティティの現在のステータスを表すデータ オブジェクトが定義されています。これらのデータ オブジェクトへのアクセスには Simple Network Management Protocol（SNMP）が使用され、これらのデータ オブジェクトはローカル NTP エンティティの監視と管理に使用されます。

これらのデータ オブジェクトには NTP エンティティに関する次の情報が含まれます。

- アップストリーム NTP サーバおよびハードウェア基準クロックへの接続状態
- 製品
- ベンダー
- Version

これらのデータ オブジェクトに含まれる情報を使用すると、ネットワーク全体の時刻同期に影響が出る前に障害を検出することができます。

RFC 5907 で扱われている次のオブジェクト グループは NTPv4 MIB でサポートされています。

- ntpAssociation
- ntpEntInfo
- ntpEntStatus

RFC 5907 で扱われている次のオブジェクト グループは NTPv4 MIB でサポートされていません。

- ntpEntControl
- ntpEntNotifObjects

NTPv4 MIB の確認方法

この機能を使用するに当たって、特別な設定は必要ありません。この機能は、デフォルトでイネーブルにされています。

NTPv4 MIB の確認

NTPv4 MIB に関する情報を確認するには、次に示すコマンドのいずれか、またはすべてを任意の順序で実行します。

SUMMARY STEPS

1. **show ntp associations** [**detail**]
2. **show ntp status**
3. **show ntp info**
4. **show ntp packets**

DETAILED STEPS

ステップ 1 **show ntp associations** [**detail**]

Example:

```
Device> show ntp associations detail
```

（任意）NTP アソシエーションの詳細なステータスを表示します。

ステップ 2 **show ntp status**

Example:

```
Device> show ntp status
```

（任意）NTP のステータスを表示します。

ステップ 3 **show ntp info**

Example:

```
Device> show ntp info
```

（任意）NTP エンティティに関する情報を表示します。

ステップ 4 **show ntp packets**

Example:

```
Device> show ntp packets
```

（任意）NTP パケットに関する情報を表示します。

NTPv4 MIB の設定例

例：NTP4 MIB の確認

show ntp associations コマンドの出力例

```
Device> show ntp associations detail

172.31.32.2 configured, ipv4, our_master, sane, valid, stratum 1
ref ID .LOCL., time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
our mode active, peer mode passive, our poll intvl 16, peer poll intvl 16
root delay 0.00 msec, root disp 0.00, reach 377, sync dist 16.05
delay 0.00 msec, offset 0.0000 msec, dispersion 8.01, jitter 0.5 msec
precision 2**7, version 4
assoc ID 1, assoc name 192.0.2.1,
assoc in packets 60, assoc out packets 60, assoc error packets 0
org time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
rec time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
xmt time D2352248.2337CCB8 (06:12:24.137 IST Tue Oct 4 2011)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =     7.81    8.05    8.29    8.53    8.77    9.01    9.25    9.49
minpoll = 4, maxpoll = 4

192.168.13.33 configured, ipv6, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
our mode client, peer mode unspec, our poll intvl 1024, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 15951.96
delay 0.00 msec, offset 0.0000 msec, dispersion 15937.50,  jitter 1000.45 msec
precision 2**7, version 4
assoc ID 2, assoc name myserver
assoc in packets 0, assoc out packets 0, assoc error packets 0
org time D2351E93.2235F124 (05:56:35.133 IST Tue Oct 4 2011)
rec time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
xmt time 00000000.00000000 (00:00:00.000 IST Mon Jan 1 1900)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
```

show ntp status コマンドの出力例

```
Device> show ntp status

Clock is synchronized, stratum 2, reference assoc id 1, reference is 192.0.2.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**7
reference time is D2352258.243DDF14 (06:12:40.141 IST Tue Oct 4 2011)
clock offset is 0.0000 msec, root delay is 0.00 msec, time resolution 1000 (1 msec),
root dispersion is 15.91 msec, peer dispersion is 8.01 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 16, last update was 6 sec ago.
system uptime (00:00:00.000) UTC,
system time is D2352258.243DDF14 (06:12:40.141 IST Tue Oct 4 2011)
leap time is D2352258.243DDF14 (24:00:00.000 IST Tue Dec 31 2011)
leap direction is 1
```

show ntp info コマンドの出力例

```
Device> show ntp info

Ntp Software Name: Example
NtP Software Version: ntp-1.1
Ntp Software Vendor: Example
Ntp System Type: Example_System
```

show ntp packets コマンドの出力例

```
Device> show ntp packets

Ntp In packets: 100
Ntp Out packets: 110
Ntp bad version packets: 4
Ntp protocol error packets: 0
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
基本的なシステム管理コマンド	Basic System Management Command Reference
基本的なシステム管理の設定作業	『*Basic System Management Configuration Guide*』の「Setting Time and Calendar Services」モジュール

標準および RFC

標準/RFC	タイトル
RFC 5907	*Definitions of Managed Objects for Network Time Protocol Version 4 (NTPv4)*

MIB

MIB	MIB のリンク
NTPv4-MIB	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャ セットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

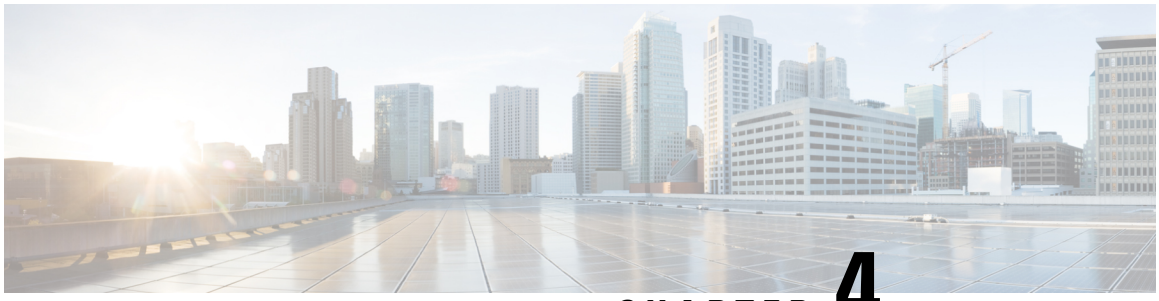
NTPv4 MIB の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 3: NTPv4 MIB の機能情報

機能名	リリース	機能情報
NTPv4 MIB		NTPv4 MIB 機能はシスコ ソフトウェアに Network Time Protocol バージョン 4（NTPv4）MIB を導入します。この機能では、NTP エンティティの現在のステータスを表すデータ オブジェクトが定義されています。これらのデータ オブジェクトへのアクセスには Simple Network Management Protocol（SNMP）が使用され、これらのデータ オブジェクトはローカル NTP エンティティの監視と管理に使用されます。



CHAPTER 4

ネットワーク タイム プロトコル

ネットワーク タイム プロトコル（NTP）は、マシンのネットワークの時刻同期を行うように設計されたプロトコルです。NTP は UDP で稼働し、UDP は IP 上で稼働します。NTP バージョン 3 は、RFC 1305 に記載されています。

このモジュールでは、シスコ デバイスで Network Time Protocol を設定する方法について説明します。

ネットワー タイム プロトコルについて

時刻サービスとカレンダーサービス

システム上の時刻データのプライマリ ソースは、ソフトウェア クロックです。このクロックはシステムが起動した瞬間から稼働して、現在の日付と時刻を追跡します。ソフトウェア クロックは多数のソースから設定でき、さまざまなメカニズムを介して他のシステムに現在の時刻を配信するために使用できます。ハードウェアクロックが内蔵されたデバイスを初期化または再起動すると、ハードウェアクロックの時刻に基づいてソフトウェアクロックが初期設定されます。その後、ソフトウェア クロックは次のソースによって更新できます。

- 手動設定（ハードウェア クロックを使用）
- ネットワーク タイム プロトコル（NTP）
- 簡易ネットワーク管理プロトコル（SNMP）
- Virtual Integrated Network Service（VINES）タイムサービス

ソフトウェアクロックは動的に更新できるため、ハードウェアクロックよりも正確である可能性があります。

ソフトウェア クロックは次のサービスに時刻を提供できます。

- アクセス リスト
- ログおよびデバッグ メッセージ
- NTP
- ハードウェア クロック
- user **show** コマンド
- VINES 時刻サービス

Note SNTP を使用してクロックを設定した場合、ソフトウェアクロックは NTP または VINES 時刻サービスに時刻を提供できません。

ソフトウェアクロックは、グリニッジ標準時（GMT）とも呼ばれる協定世界時（UTC）に基づいて内部的に時刻を追跡します。ローカル時間帯に対して時刻が正しく表示されるように、地域の時間帯とサマータイムに関する情報を設定できます。

ソフトウェアクロックは、時刻が「正規」であるかどうか（つまり、信頼できると見なされる時刻源によって設定されたかどうか）を追跡します。正規でない場合、時刻は表示のためだけに使用でき、再配信されません。

ネットワーク タイム プロトコル

ネットワーク タイム プロトコル（NTP）は、マシンのネットワークの時刻同期を行うように設計されたプロトコルです。NTP は UDP で稼働し、UDP は IP 上で稼働します。NTP バージョン 3（NTPv3）は、RFC 1305 に記載されています。

NTP ネットワークは通常、タイム サーバに接続されたラジオ クロックやアトミック クロックなど、正規の時刻源から時刻を取得します。NTP は、ネットワークにこの時刻を分配します。NTP はきわめて効率的です。毎分 1 パケットだけで、2 台のマシンが相互に 1 ミリ秒以内の精度で同期します。

NTP では、信頼できるタイム ソースから各マシンが何 NTP ホップ隔たっているかを表すために、ストラタムという概念が使用されます。Stratum 1 タイムサーバーには通常、正規の時刻源（電波時計、原子時計、Global Positioning System（GPS）時刻源など）が直接接続されています。Stratum 2 タイムサーバーは、Stratum 1 タイムサーバーから NTP を介して時刻を受信し、それ以降のサーバーも続きます。

NTP は、次の 2 つの方法により、時刻が正確でない可能性があるマシンへの同期を回避します。NTP は、NTP と同期していないマシンとは同期しません。複数のマシンから報告された時刻を比較し、他のマシンと時刻が大きく異なるマシンとは、そのストラタムがより低くても同期しません。このようにして、NTP サーバのツリーは効率よく自律的に編成されています。

シスコの NTP 実装では、Stratum 1 サービスをサポートしていないため、電波時計や原子時計に接続することはできません（ただし、いくつかの特定のプラットフォームでは、GPS 時刻源

デバイスに接続できます）。ネットワークのタイムサービスは、IPインターネットで利用できるパブリック NTP サーバーから取得することをお勧めします。

ネットワークがインターネットから分離されている場合、NTPの実装により、実際にはネットワークが他の手段を使用して時刻を決定している場合でも、あたかも NTP 経由で同期しているかのように動作するようにマシンを構成できます。これにより、他のマシンが NTP を介してそのマシンと同期できるようになります。

多くの製造業者のホストシステムで、NTP ソフトウェアが導入されています。また、UNIX システム向けに公開されているバージョンもあります。また、このソフトウェアにより UNIX 派生サーバーは原子時計から時刻を直接取得することができ、シスコデバイスに時刻情報を伝えるようにすることもできます。

NTPを実行しているマシン間の通信（アソシエーション）は通常、静的に設定されており、各マシンには、アソシエーションを形成する必要があるすべてのマシンの IP アドレスが通知されます。アソシエーションが設定されたマシンの各ペアの間で NTP メッセージを交換することにより、正確な時刻管理が可能になります。

ただし、LAN 環境では、代わりに IP ブロードキャスト メッセージを使用するように NTP を設定できます。この代替手段では、ブロードキャストメッセージを送受信するように各マシンを設定できるので、設定の複雑さが緩和されます。ただし、情報の流れが一方向のみであるため、計時精度はわずかに低下します。

マシン上の時刻は重要な情報であるため、NTPのセキュリティ機能を使用して、不正な時刻を誤って（または悪意を持って）設定できないように保護することを強く推奨します。アクセスリストベースの制約方式と、暗号化認証メカニズムの2つのセキュリティメカニズムが使用できます。

複数の時刻源（VINES、ハードウェア クロック、手動による設定）がある場合、NTP は常により信頼できる時刻源とされます。NTP の時刻は、他の方法による時刻に優先します。

NTP サービスは、デフォルトではすべてのインターフェイスで無効になっています。

NTP の詳細については、次の項を参照してください。

ポーリング ベースの NTP アソシエーション

NTP を実行しているネットワーキング デバイスは、時刻を基準時刻源と同期する際にさまざまなアソシエーションモードで動作するように設定できます。ネットワーキングデバイスは、2つの方法でネットワーク上の時刻情報を取得できます。それらは、ホスト サービスのポーリングと NTP ブロードキャストのリスニングです。ここでは、ポーリングベースのアソシエーション モードを中心に説明します。ブロードキャスト ベースの NTP アソシエーションの詳細については、「ブロードキャスト ベースの NTP アソシエーション」を参照してください。

最も一般的に使用される 2 つのポーリングベースのアソシエーション モードは次のとおりです。

- クライアント モード
- 対称アクティブ モード

クライアント モードと対称アクティブ モードは、高レベルの時刻の精度と信頼性を提供するために NTP が必要になる場合に使用します。

クライアント モードで動作しているネットワーキング デバイスは、自身に割り当てられている時刻提供ホストをポーリングして現在の時刻を取得します。次に、ネットワーキングデバイスは、ポーリングされたすべてのタイム サーバーから、同期に使用するホストを選択します。この場合は、確立された関係がクライアントホスト関係なので、ホストがローカル クライアントデバイスから送信された時刻情報をキャプチャしたり使用したりすることはありません。このモードが最も適しているのは、他のローカル クライアントにどのような形式の時刻同期も提供する必要のない、ファイルサーバーおよびワークステーションのクライアントです。ネットワーキングデバイスを同期させるタイムサーバーを個別に指定し、クライアントモードで動作するようにネットワーキングデバイスを設定するには、**ntp server** コマンドを使用します。

対称アクティブ モードで動作しているネットワーキング デバイスは、自身に割り当てられている時刻提供ホストをポーリングして現在の時刻を取得し、そのホストによるポーリングに応答します。これはピアツーピアの関係なので、ホストは、通信相手のローカル ネットワーキング デバイスの時刻関連情報も保持します。このモードは、さまざまなネットワーク パスを経由で多数の冗長サーバーが相互接続されている場合に使用します。インターネット上のほとんどの Stratum 1 および Stratum 2 サーバーは、この形式のネットワーク設定を採用しています。ネットワーキングデバイスを同期させる時刻提供ホストを個別に指定し、対称アクティブモードで動作するようにネットワーキングデバイスを設定するには、**ntp peer** コマンドを使用します。

各ネットワーキング デバイスの設定モードを決定する際には、タイムキーピング デバイスとしてのそのデバイスの役割（サーバーかクライアントか）と、そのデバイスが Stratum 1 タイムキーピング サーバーにどれだけ近いかを主に考慮してください。

ネットワーキング デバイスは、クライアント モードでクライアントまたはホストとして動作する場合、または対称アクティブ モードでピアとして動作する場合にポーリングに関与します。通常、ポーリングによってメモリおよび CPU リソース（帯域幅など）に負荷が生じることはありませんが、システム上で進行または同時実行しているポーリングの数がきわめて多い場合には、システムの性能に深刻な影響があったり、特定のネットワークの性能が低下したりする可能性があります。過剰な数のポーリングがネットワーク上で進行することを防止するには、直接的なピアツーピア アソシエーションまたはクライアントからサーバーへのアソシエーションを制限する必要があります。代わりに、局所的なネットワーク内に NTP ブロードキャストを使用して時刻情報を伝播することを検討します。

ブロードキャスト ベースの NTP アソシエーション

ブロードキャストベースの NTP アソシエーションは、時刻の精度および信頼性要件が適度であり、ネットワークが局所的であり、クライアント数が 20 を超える場合に使用します。また、帯域幅、システム メモリ、または CPU リソースが制限されているネットワークにおいても、ブロードキャストベースの NTP アソシエーションの使用をお勧めします。

ブロードキャスト クライアント モードで動作しているネットワーキング デバイスはポーリングに関与しません。代わりに、ブロードキャスト タイム サーバーによって転送される NTP ブロードキャスト パケットを待ち受けます。その結果、時刻情報の流れが一方向に限られるため、時刻の精度がわずかに低下する可能性があります。

ネットワークを通じて伝播される NTP ブロードキャスト パケットをリッスンするようにネットワーキング デバイスを設定するには、**ntp broadcast client** コマンドを使用します。ブロードキャスト クライアント モードが動作するためには、ブロードキャスト サーバーとそのクライアントが同じサブネット上に存在する必要があります。**ntp broadcast** コマンドを使用して、特定のデバイスのインターフェイスで NTP ブロードキャストケットを送信するタイムサーバーを有効にする必要があります。

NTP アクセス グループ

アクセスリストベースの制限スキームを使用すると、ネットワーク全体、ネットワーク内のサブネット、またはサブネット内のホストに対し、特定のアクセス権限を許可または拒否できます。NTP アクセスグループを設定するには、グローバル コンフィギュレーション モードで **ntp access-group** コマンドを使用します。

アクセス グループのオプションは、次の順序で制限の緩いものから厳しいものへとスキャンされます。

1. **ipv4**：IPv4 アクセスリストを設定します。
2. **ipv6**：IPv6 アクセスリストを設定します。
3. **peer**：時刻要求と NTP 制御クエリを許可し、システムがアクセスリストの基準を満たすアドレスを持つ別のシステムに同期することを許可します。
4. **serve**：時刻要求と NTP 制御クエリーを許可しますが、システムがアクセスリストの基準を満たすアドレスを持つ別のシステムに同期することは許可しません。
5. **serve-only**：アクセスリストの条件を満たすアドレスを持つシステムからの時刻要求のみを許可します。
6. **query-only**：アクセスリストの基準を満たすアドレスを持つ別のシステムからの NTP 制御クエリーのみを許可します。

送信元 IP アドレスが複数のアクセス タイプのアクセス リストに一致する場合は、最初のアクセス タイプのアクセスが認可されます。アクセス グループが指定されていない場合は、すべてのシステムへのアクセスがすべてのアクセス タイプに対して認可されます。アクセス グループが指定されている場合は、指定されたアクセス タイプに対してのみアクセスが認可されます。

NTP 制御クエリーの詳細については、RFC 1305（NTP バージョン 3）を参照してください。

信頼できる形式のアクセス コントロールが必要な場合は、暗号化された NTP 認証方式を使用する必要があります。IP アドレスに基づくアクセス リスト ベースの制約方式とは異なり、暗号化認証方式では、認証キーと認証プロセスを使用して、ローカル ネットワーク上の指定されたピアまたはサーバーによって送信された NTP 同期パケットが信頼できると見なされるかどうかを、一緒に伝送された時刻情報を受け入れる前に判断します。

認証プロセスは、NTP パケットが作成されるとすぐに開始されます。暗号チェックサム キーは、Message-Digest Algorithm 5（MD5）を使用して生成され、受信側クライアントに送信される NTP 同期パケットに埋め込まれます。パケットがクライアントによって受信されると、暗号チェックサム キーが復号され、信頼できるキーのリストに対してチェックされます。一致す

る認証キーがパケットに含まれる場合、受信側クライアントは、パケットに含まれるタイムスタンプ情報を受け入れます。一致するオーセンティケータ キーが含まれていない NTP 同期パケットは無視されます。

Note 信頼できるキーを多数設定する必要がある大規模なネットワークでは、信頼できるキーの範囲設定機能を使用して複数のキーを同時に有効にすることができます。

NTP 認証で使用される暗号化および復号化プロセスでは、CPU に非常に大きな負荷がかかる場合があり、ネットワーク内で伝播される時刻の精度が大きく低下する可能性があることに注意してください。より包括的なアクセス コントロール モデルを使用できるネットワーク構成の場合は、アクセス リスト ベースのコントロール方式を使用することを検討してください。

NTP認証が適切に設定されると、ネットワーキングデバイスは、信頼できる時刻源と同期し、信頼できる時刻源だけに同期を提供します。

特定のインターフェイス上の NTP サービス

Network Time Protocol（NTP）サービスは、デフォルトではすべてのインターフェイスで無効になっています。なんらかの NTP コマンドを入力すると、NTP がグローバルに有効になります。特定のインターフェイスを通じて特定のNTPパケットを受信しないように設定するには、インターフェイス コンフィギュレーション モードで **ntp disable** コマンドを使用します。

NTP パケットの送信元 IP アドレス

システムがNTPパケットを送信すると、通常、送信元IPアドレスは、そのNTPパケットの送信元であるインターフェイスのアドレスに設定されます。IP 送信元アドレスの取得元のインターフェイスを設定するには、グローバル コンフィギュレーション モードで **ntp source** *interface* コマンドを使用します。

このインターフェイスは、すべての宛先に送信されるすべてのパケットの送信元アドレスに使用されます。特定のアソシエーションに送信元アドレスを使用する場合は、**ntp peer** コマンドまたは **ntp server** コマンドで **source** キーワードを使用します。

正規の NTP サーバとしてのシステム

システムを正規の NTP サーバーにする場合は、グローバル コンフィギュレーション モードで **ntp** コマンドを使用します。これは、システムが外部の時刻源と同期されていない場合でも同じです。

Note **ntp primary** コマンドの使用には注意が必要です。このコマンドを使用すると、有効な時刻源が容易に上書きされてしまいます。低いストラタム番号を設定する際には、特に注意が必要です。**ntp primary** コマンドを使用して同じネットワーク内の複数のマシンを設定した場合は、それらのマシンの時刻が一致していないと、時刻管理が不安定になることがあります。

孤立モード

NTP サブネットは、ローカル基準クロックまたはインターネット クロック サーバーから分離されることがあります。この分離期間中、サブネットサーバーとクライアントは共通のタイムスケールに同期されます。ローカルクロックドライバは、UTC ソースをシミュレートして、共通のタイムスケールを提供します。ドライバに直接または間接的に接続されたサーバーは、サブネット内の他のホストを同期します。

ローカルクロックドライバを使用すると、サブネットの回復不能な障害が発生する可能性があり、複数のサーバーを使用して冗長性を維持することは現実的ではありません。このような欠点のない孤立モード機能により、ローカルクロックドライバが不要になります。孤立モード機能は、複数のサーバーを備えた単一のシミュレートされた UTC ソースと、サーバーが障害から回復する際のシームレスな切り替えメカニズムを提供します。

プライベート ネットワークでは、通常、最下位のストラタムで動作する 1 つまたは複数のコアサーバーが含まれます。これらの各サーバーは、対称モードまたはブロードキャストモードを使用する他のサーバーのバックアップとして設定する必要があります。1 つのコアサーバーが UTC ソースに到達した場合でも、サブネット全体がシミュレートしているサーバーに同期されます。どのサーバーも UTC ソースに到達しない場合、いずれかのサーバー（孤立した親と呼ばれる）が UTC ソースをシミュレートし、サブネット内の他のすべてのホスト（孤立した子と呼ばれる）のシミュレートされた UTC ソースとして機能できます。

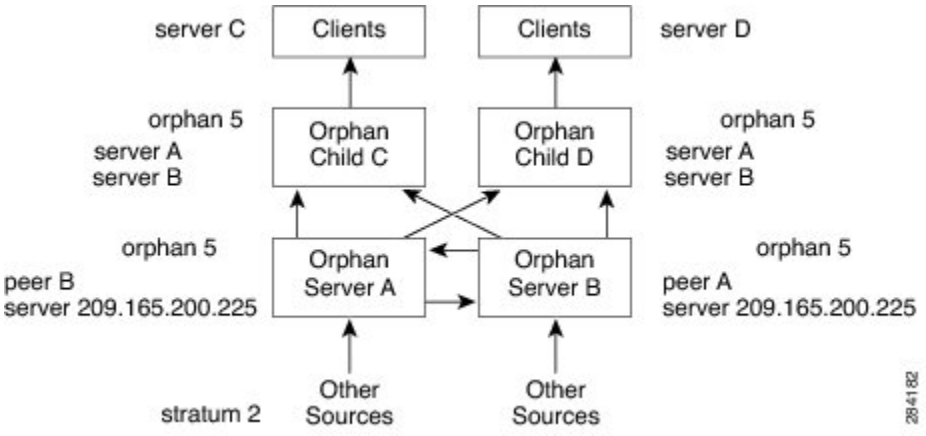
ntp orphan *stratum* コマンドを使用して、孤立モードのホストを有効にします。ここで、*stratum* は、16 未満で、設定されたインターネット タイム サーバーに出現するどのストラタム値よりも大きいストラタム値です。ただし、孤立した子に依存するすべてのサブネットホストのストラタム値が 16 未満になるように、十分なストラタムを指定する必要があります。他のサーバーまたは基準クロックのアソシエーションが設定されていない場合は、孤立ストラタム値を 1 に設定する必要があります。

ソースのないストラタム 1 で動作している孤立した親には、参照 ID LOOP が表示されます。ストラタム 1 で動作していない孤立した親は、UNIX ループバックアドレス 127.0.0.1 を表示します。通常の NTP クライアントは遅延と分散に基づく選択メトリックを使用しますが、孤立した子はサブネット内の各コアサーバーの IP アドレスから計算されたメトリックを使用します。各孤立した子は、最小のメトリックを持つ孤立した親をルートサーバーとして選択します。

すべてのソースを失ったサーバーは、ローカルクロックドライバを他のサーバーと継続的に同期させ、サーバーをバックアップします。コアサーバーと孤立した子でのみ孤立モードを有効にします。

次の図に、孤立モードのセットアップ方法とピアネットワーク設定を示します。この場合、2 台のプライマリまたはセカンダリ（ストラタム 2）サーバーが基準クロックまたはパブリック インターネット プライマリ サーバーで設定され、それぞれが対称モードを使用します。

Figure 1: 孤立モードの設定



孤立モードの前提条件

孤立モードをスムーズに機能させるには、同じストラタムで動作するように、使用可能なソースを使用して各コアサーバーを設定する必要があります。すべてのコアサーバーと孤立した子で **ntp orphan** コマンドを設定します。すべてのルートサーバーで孤立した子を設定します。

Simple Network Time Protocol

簡易ネットワーク タイム プロトコル（SNTP）とは、クライアント専用バージョンの簡易版 NTP です。SNTP は、NTP サーバから時刻を受信できるだけで、時刻サービスを他のシステムに提供できません。

通常、SNTP は 100 ミリ秒以内の精度で時刻を提供しますが、NTP のような複雑なフィルタリングや統計メカニズムは提供しません。また、拡張アクセスリストを設定することによってある程度の保護を提供できますが、トラフィックを認証できません。SNTP クライアントは、NTP クライアントよりも予期しない動作をするサーバーに対して脆弱であるため、強力な認証が必要ない状況でのみ使用する必要があります。

SNTP は、設定済みのサーバーからパケットを要求して受け入れるように設定するか、任意の送信元から NTP ブロードキャスト パケットを受け入れるように設定できます。複数の送信元が NTP パケットを送信している場合、最適な層にあるサーバーが選択されます（階層の説明については、3 ページの「*Network Time Protocol*」セクションを参照してください）。複数のサーバーのストラタムが同じだった場合は、ブロードキャスト サーバーよりも設定済みサーバーが優先されます。これらの両方を満たすサーバーが複数ある場合は、時刻パケットを最初に送信したサーバーが選択されます。SNTP が新しいサーバを選択するのは、現在選択しているサーバからのパケットの受信を停止している場合、または（上記の基準に従って）より適切なサーバが検出された場合だけです。

VINES 時刻サービス

Banyan VINES を設定すると、時刻サービスを使用できます。このプロトコルは、VINES の標準部分です。シスコの実装では、2つの方法でVINES時刻サービスを使用できます。最初の方

法では、他の時刻源から時刻を認識すると、システムは VINES タイム サーバとして動作し、VINES を実行している他のマシンに時刻を提供できます。2 番目の方法では、他の形式の時刻サービスを使用できない場合に、システムは VINES 時刻サービスを使用してソフトウェア クロックを設定できます。

Note すべてのリリースで、Banyan VINE および Xerox Network Systems（XNS）のサポートが利用できるわけではありません。

ハードウェア クロック

一部のデバイスは、システムの再起動から電源停止に至る日付および時刻を追跡するバッテリ駆動式のハードウェアクロックを内蔵しています。システムの再起動時には、ハードウェア クロックを常に使用してソフトウェア クロックが初期化されます。

Note CLI コマンド構文においては、ハードウェアクロックは「システムカレンダー」と呼ばれます。

他の時刻源を使用できない場合、ハードウェアクロックは正規の時刻源と見なされ、NTP を通じて再配信されます。NTP が実行されている場合、ハードウェアクロックは NTP から定期的に更新され、ハードウェアクロックが実行されたままになっている場合に一定のレートで一貫した時間の増加または損失である固有のドリフトを補正できます。

任意のデバイスのハードウェアクロック（システムカレンダー）がソフトウェアクロックから定期的に更新されるように設定できます。この設定は、NTP を使用するすべてのデバイスに推奨される方法です。それは、ハードウェアクロックの時刻設定は時間とともにわずかにドリフトする可能性があり、（NTP を使用して設定する）ソフトウェアクロックの時刻と日付の方がハードウェアクロックよりも正確であるためです。

ルーティングデバイスが NTP 経由で外部の時刻源と同期されている場合に、ハードウェアクロックを NTP 時刻に同期させるときは、グローバル コンフィギュレーション モードで次の **ntp update-calendar** コマンドを使用します。

時間範囲

シスコソフトウェアでは、時刻に基づいて機能を実装できます。**time-range** 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、特定の日/曜日の時間を定義します。この時間を関数から参照することにより、関数そのものに時間的制約を設定することができます。

リリースによっては、時間範囲を使用できる機能は、IP および Internetwork Packet Exchange（IPX）拡張アクセスリストだけです。時間範囲を使用すると、ネットワーク管理者はアクセス リストで permit 文または deny 文がいつ有効になるかを定義できます。この機能が導入されるまで、アクセスリストの文は、いったん適用すると常に有効になったままでした。時間範囲は、名前付きアクセス リストと番号付きアクセス リストの両方から参照できます。

Note 時間帯はシステムのソフトウェアクロックに基づきます。時間範囲機能が意図したとおりに機能するためには、信頼できるクロック ソースが必要になります。NTP を使用してシステムのソフトウェアクロックを同期させることを推奨します。

時間範囲の利点は次のとおりです。

- ネットワーク管理者は、リソースへのユーザーアクセスの許可または拒否の制御をより強化できます。これらのリソースとして、アプリケーション（IPアドレス/マスクペアとポート番号によって特定されます）、ポリシールーティング、またはオンデマンドリンク（ダイヤラへの関連トラフィックとして認識されます）があります。
- ネットワーク管理者は、次の内容を含む時間ベースのセキュリティポリシーを設定できます。
  - Cisco Firewall フィーチャセットまたはアクセスリストを使用する境界セキュリティなどがあります。
  - シスコ暗号化テクノロジーまたは IP セキュリティによるデータの機密性。
- ポリシーベース ルーティングおよび キューイング機能が拡張されています。
- プロバイダーのアクセスレートが時間帯によって異なる場合、トラフィックを自動的かつコスト効率よく再ルーティングできます。
- サービスプロバイダーは、特定の時間にネゴシエートされる Quality of Service（QoS）サービスレベル契約（SLA）をサポートするために専用アクセスレート（CAR）を動的に変更できます。

ネットワーク管理者は、ロギング メッセージを制御できます。アクセス リスト エントリは、一日の特定の時間帯にトラフィックをロギングすることはできますが、常にロギングすることはできません。したがって、管理者は、ピーク時に生成される多数のログを分析することなく、アクセスを拒否できます。

ネットワーク タイム プロトコルの設定方法

NTP の設定

ネットワーク タイム プロトコルに関する制約事項

Network Time Protocol（NTP）パッケージには、認証されていないリモート攻撃者がサービス妨害（DoS）状態を発生させる可能性がある脆弱性が含まれています。NTP バージョン 4.2.4p7 以前は脆弱です。

この脆弱性は、特定の不正メッセージの処理におけるエラーによるものです。認証されていないリモート攻撃者は、スプーフィングされた送信元 IP アドレスを使用して、悪意ある NTP パ

ケットを脆弱なホストに送信する可能性があります。このパケットを処理するホストは、送信者に応答パケットを返信します。この処理により、2 つのホスト間でメッセージのループが開始される可能性があります。その結果、両方のホストは、過剰な CPU リソースを消費し、ログファイルへのメッセージの書き込みにディスク スペースを使い切り、ネットワーク帯域幅を消費します。これにより、影響を受けたホスト上で DoS 状態が発生する可能性があります。

詳細については、Web ページ「Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability」を参照してください。

NTPv4 をサポートしている Cisco ソフトウェア リリースは影響を受けません。この問題は、その他すべての Cisco ソフトウェア バージョンに影響を及ぼします。

デバイスが NTP を使用するように設定されているかどうかを表示するには、**show running-config | include ntp** コマンドを使用します。出力に次のいずれかのコマンドが返された場合、そのデバイスは DoS 攻撃に対して脆弱です。

- **ntp broadcast client**
- **ntp primary**
- **ntp multicast client**
- **ntp peer**
- **ntp server**

Cisco ソフトウェア リリースの詳細については、『White Paper: Cisco IOS and NX-OS Software Reference Guide』を参照してください。

デバイスで NTP を無効にする以外にこの脆弱性に対する回避策はありません。この脆弱性を悪用できるのは、デバイス上の設定済み IP アドレスに宛てられたパケットだけです。中継トラフィックは、この脆弱性を悪用しません。

リリースによっては NTP モード 7 パケットが処理され、NTP のデバッグが有効になっている場合は「NTP: Receive: dropping message: Received NTP private mode 7 packet」というメッセージが表示されることがあります。**ntp allow mode private** コマンドを設定し、NTP モード 7 パケットを処理します。このコマンドは、デフォルトで無効になっています。

Note NTP ピア認証は回避策ではなく、脆弱な設定です。

NTP サービスは、デフォルトではすべてのインターフェイスで無効になっています。

NTP を実行しているネットワーキング デバイスは、時刻を基準時刻源と同期する際にさまざまなアソシエーションモードで動作するように設定できます。ネットワーキングデバイスは、2 つの方法でネットワーク上の時刻情報を取得できます。それらは、ホスト サービスのポーリングと NTP ブロードキャストのリスニングです。

Line Aux 0 オプションはデフォルトで無効になっています。

Cisco IOS XE で同じ NTP サーバーの IP アドレスと FQDN の両方を設定すると、FQDN が同じ IP アドレスに解決された後、FQDN 設定のみが **show running-config** コマンド出力に表示されます。

ポーリング ベースの NTP アソシエーションの設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp peer** *ip-address* [**normal-sync**] [**version** *number*] [**key** *key-id*] [**prefer**]
4. **ntp server** *ip-address* [**version** *number*] [**key** *key-id*] [**prefer**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**ntp peer** *ip-address* [**normal-sync**] [**version** *number*] [**key** *key-id*] [**prefer**] **Example:** `Device(config)# ntp peer 192.168.10.1 normal-sync version 2 prefer`	他のシステムとのピア アソシエーションを形成します。
ステップ 4	**ntp server** *ip-address* [**version** *number*] [**key** *key-id*] [**prefer**] **Example:** `Device(config)# ntp server 192.168.10.1 version 2 prefer`	他のシステムとのサーバー アソシエーションを形成します。
ステップ 5	**end** **Example:** `Device(config)# end`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

ブロードキャスト ベースの NTP アソシエーションの設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ntp broadcast version** *number*
5. **ntp broadcast client**
6. **ntp broadcastdelay** *microseconds*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0`	インターフェイスを設定し、インターフェイス コンフィギュレーション モードを開始します。
ステップ 4	**ntp broadcast version** *number* **Example:** `Device(config-if)# ntp broadcast version 2`	指定されたインターフェイスがNTPブロードキャスト パケットを送信するように設定します。
ステップ 5	**ntp broadcast client** **Example:** `Device(config-if)# ntp broadcast client`	指定されたインターフェイスがNTPブロードキャスト パケットを受信するように設定します。
ステップ 6	**ntp broadcastdelay** *microseconds* **Example:** `Device(config-if)# ntp broadcastdelay 100`	NTPブロードキャストの推定ラウンドトリップ遅延を調整します。
ステップ 7	**end** **Example:**	インターフェイス コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

	Command or Action	Purpose
	`Device(config-if)# end`	

NTP 認証の設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp authenticate**
4. **ntp authentication-key** *number* **md5** *key*
5. **ntp authentication-key** *number* **md5** *key*
6. **ntp authentication-key** *number* **md5** *key*
7. **ntp trusted-key** *key-number* [**-** *end-key*]
8. **ntp server** *ip-address* **key** *key-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**ntp authenticate** **Example:** `Device(config)# ntp authenticate`	NTP 認証機能を有効にします。
ステップ 4	**ntp authentication-key** *number* **md5** *key* **Example:**	認証キーを定義します。 • キーごとに、キー番号、タイプ、および値を 1 つずつ指定します。
ステップ 5	**ntp authentication-key** *number* **md5** *key* **Example:**	認証キーを定義します。 • キーごとに、キー番号、タイプ、および値を 1 つずつ指定します。

	Command or Action	Purpose
ステップ 6	**ntp authentication-key** *number* **md5** *key* **Example:**	認証キーを定義します。 • キーごとに、キー番号、タイプ、および値を 1 つずつ指定します。
ステップ 7	**ntp trusted-key** *key-number* [- *end-key*] **Example:** `Device(config)# ntp trusted-key 1 - 3`	信頼できる認証キーを定義します。 • キーを信頼できる場合、このデバイスは、このキーをNTPパケット内で使用する別のシステムに同期できます。
ステップ 8	**ntp server** *ip-address* **key** *key-id* **Example:** `Device(config)# ntp server 172.16.22.44 key 2`	NTP タイム サーバーによってソフトウェア クロックが同期されるように設定します。 **Note** 複数の NTP サーバーが設定され、ロギングが有効になっている場合、クロック同期損失メッセージがデバイスでランダムに表示されます。この問題を解決するには、**peer** キーワードを使用して NTP サーバーを設定します。 `Device(config)# ntp server ip-address [version number] [key key-id] [prefer]`
ステップ 9	**end** **Example:** `Device(config)# end`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

外部基準クロックの設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line aux** *line-number*
4. **end**
5. **show ntp associations**
6. **show ntp status**
7. **debug ntp refclock**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:**	特権 EXEC モードを有効にします。

	Command or Action	Purpose
	`Device> enable`	• 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	**line aux** *line-number* **Example:** `Device(config)# line aux 0`	補助ポート 0 のライン コンフィギュレーション モードを開始します。
ステップ 4	**end** **Example:** `Device(config-line)# end`	回線コンフィギュレーション モードを終了します。続いて、特権 EXEC モードに戻ります。
ステップ 5	**show ntp associations** **Example:** `Device# show ntp associations`	NTP アソシエーションのステータスを表示します（GPS 基準クロックのステータスを含みます）。
ステップ 6	**show ntp status** **Example:** `Device# show ntp status`	NTP のステータスを表示します。
ステップ 7	**debug ntp refclock** **Example:** `Device# debug ntp refclock`	デバッグを目的とした基準クロック動作の拡張モニタリングを許可します。

孤立モードの設定

孤立モードを設定するには、少なくとも 2 つのクライアントが必要です。次のタスクは、1 つのクライアントで孤立モードを設定する方法を示しています。他のクライアントで手順を繰り返します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ntp server** *ip-address*
4. **ntp peer** *ip-address*
5. **ntp orphan** *stratum*
6. 他のクライアントでも手順 1 ～ 5 を繰り返します。

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**ntp server** *ip-address* **Example:** `Router(config)# ntp server 10.1.1.1`	他のシステムとのサーバーアソシエーションを形成します。
ステップ 4	**ntp peer** *ip-address* **Example:** `Router(config)# ntp peer 172.16.0.1`	他のシステムとのピアアソシエーションを形成します。 **Note** 他のクライアントでピアを設定するときに、設定したばかりの IP アドレスとは異なる IP アドレス（172.16.0.2 など）を使用します。
ステップ 5	**ntp orphan** *stratum* **Example:** `Router(config)# ntp orphan 4`	ホストで孤立モードを有効にします。
ステップ 6	他のクライアントでも手順 1 ～ 5 を繰り返します。	

SNTP の設定

SNTP は通常、NTP をサポートしていないプラットフォームでサポートされます。SNTP は、デフォルトでディセーブルになっています。SNTP を構成するには、次のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sntp server** {*address* | *hostname*} [**version** *number*]
4. **sntp broadcast client**
5. **exit**
6. **show sntp**

DETAILED STEPS

	Command or Action	Purpose
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	**sntp server** {*address* | *hostname*} [**version** *number*] **Example:** `Device(config)# sntp server 192.168.2.1 version 2`	NTP サーバーからの NTP パケットを要求するように SNTP を設定します。 • 各 NTP サーバーについて、**sntp server** コマンドを 1 回入力します。NTP サーバーは、デバイスからの SNTP メッセージに応答するように設定する必要があります。
ステップ 4	**sntp broadcast client** **Example:** `Device(config)# sntp broadcast client`	任意の NTP ブロードキャストからの NTP パケットを受け入れるように SNTP を設定します。 **Note** **sntp server** コマンドと **sntp broadcast client** コマンドの両方を入力した場合、層が同じであるとすると、デバイスはブロードキャストサーバーからの時刻を受け入れますが、設定されたサーバーからの時刻を優先します。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 6	**show sntp** **Example:** `Device# show sntp`	SNTP に関する情報を表示します。

VINES 時刻サービスの設定

Banyan VINES を設定すると、時刻サービスを使用できます。このプロトコルは、VINES の標準部分です。VINE タイムサービスを設定するには、次のタスクを実行します。

Note リリースに応じて、Banyan VINE および XNS は Cisco ソフトウェアで使用できます。 **vines time set-system** および **vines time use-system** コマンドは、一部のリリースでは使用できません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vines time use-system**
4. **vines time set-system**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**vines time use-system** **Example:** `Device(config)# vines time use-system`	システムのソフトウェア クロック時刻を他の VINES システムに配信します。
ステップ 4	**vines time set-system** **Example:** `Device(config)# vines time set-system`	VINES タイムサービスから導出されたソフトウェア クロック システムの時刻と日付を設定します。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

日付と時刻の設定

他の時刻源を使用できない場合は、システムの再起動後に現在の時刻と日付を手動で設定できます。設定した時刻は、次回システムを再起動するまで正確に維持されます。手動設定は最後の手段としてのみ使用することを推奨します。

デバイスが同期できる外部の時刻源がある場合は、ソフトウェアクロックを手動で設定できないことがあります。時刻と日付を手動で設定するには、次のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clock timezone** *zone hours-offset* [*minutes-offset*]
4. **clock summer-time** *zone* **recurring** [*week day month hh*:*mm week day month hh*:*mm* [*offset*]]
5. **clock summer-time** *zone* **date** *date month year hh*:*mm date month year hh*:*mm* [*offset*]
6. **exit**
7. **clock set** *hh*:*mm*:*ss date month year*

DETAILED STEPS

	Command or Action	Purpose
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	**clock timezone** *zone hours-offset* [*minutes-offset*] **Example:** `Device(config)# clock timezone PST 2 30`	シスコソフトウェアで使用されるタイムゾーンを設定します。 **Note** **clock timezone** コマンドの *minutes-offset* 引数は、ローカル時間帯がUTC/GMTと1時間の何%異なるかによって表される場合に使用できます。たとえば、アトランティックカナダの一部の地域の時間帯（大西洋標準時（AST））はUTC-3.5です。この場合に必要なコマンドは、**clock timezone AST –3 30** です。
ステップ 4	**clock summer-time** *zone* **recurring** [*week day month hh*:*mm week day month hh*:*mm* [*offset*]] **Example:**	サマータイム（夏時間）を毎年特定の曜日に開始および終了する地域で設定します。

	Command or Action	Purpose
	`Device(config)# clock summer-time PST recurring 1 monday january 12:12 4 Tuesday december 12:12 120`	
ステップ 5	**clock summer-time** *zone* **date** *date month year hh:mm date month year hh:mm* [*offset*] **Example:** `Device(config)# clock summer-time PST date 1 january 1999 12:12 4 december 2001 12:12 120`	特定のサマー タイムの開始日と終了日を設定します。 • *offset* 引数は、UTC との時間帯の時差（時間数）です。
ステップ 6	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 7	**clock set** *hh:mm:ss date month year* **Example:** `Device# clock set 12:12:12 1 january 2011`	ソフトウェア クロックを設定します。 • 他の時刻源を使用できない場合は、次のコマンドを使用してください。このコマンドで指定する時刻は、設定されている時間帯に対応します。 **Note** 一般に、NTP や VINES クロック ソースなどの有効な外部の時刻メカニズムによってシステムが同期されている場合や、ハードウェアクロックを内蔵したデバイスを使用する場合には、ソフトウェアクロックを設定する必要があります。

ハードウェア クロックの設定

ほとんどのシスコデバイスは、ソフトウェアベースのクロックに加えて、別個のハードウェアベースのクロックを内蔵しています。ハードウェア クロックは、デバイスの各再起動間で時刻および日付情報を維持できる充電式バックアップ バッテリを備えたチップです。

ネットワーク上の正規の時刻源からの最も正確な時刻のアップデートを維持するため、ソフトウェア クロックは、ネットワーク上の正規の時刻源から時刻のアップデートを受信する必要があります。ハードウェア クロックは、システムが稼働している間、ソフトウェア クロック から定期的に更新される必要があります。

ハードウェア クロック（システム カレンダー）は、ソフトウェア クロックとは別に時刻を維持しています。システムを再起動した場合や、電源を遮断した場合でも、ハードウェア クロックは動作し続けます。通常、ハードウェアクロックは、システムのインストール時に1回だけ手動で設定する必要があります。

信頼できる外部時刻ソースにアクセスできる場合は、ハードウェアクロックを設定しないでください。代わりに、NTP を使用して時刻同期を確立する必要があります。

ハードウェアクロックを設定するには、次のタスクを実行します。

Before you begin

Note リリースに応じて、NTP は Linux カーネルの時刻を更新する IOS デーモン（IOSd）内で実行されます。Linux カーネルは 11 分ごとにハードウェアクロックを更新するため、NTP はハードウェアクロックと直接対話しません。したがって、カレンダー関連のコマンドは必要ありません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clock calendar-valid**
4. **exit**
5. **clock read-calendar**
6. **clock update-calendar**
7. **show calendar**
8. **show clock** [**detail**]
9. **show ntp associations** [**detail**]
10. **show ntp status**
11. **show sntp**

DETAILED STEPS

	Command or Action	Purpose
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	**clock calendar-valid** **Example:** `Device(config)# clock calendar-valid`	ネットワークピアを同期できる有効な時刻源としてデバイスが動作できるようにします。 • デフォルトでは、ソフトウェアクロックで維持される時刻は信頼できるものとみなされず、NTP または VINES タイムサービスと同期されません。ハードウェアクロックを有効な時刻源

	Command or Action	Purpose
		として設定するには、このコマンドを使用します。
ステップ 4	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 5	**clock read-calendar** **Example:** `Device# clock read-calendar`	ソフトウェアクロックを新しいハードウェアクロック設定に設定します。
ステップ 6	**clock update-calendar** **Example:** `Device# clock update-calendar`	新しいソフトウェアクロック設定でハードウェアクロックを更新します。
ステップ 7	**show calendar** **Example:** `Device# show calendar`	ハードウェア クロックの現在の時刻を表示します。
ステップ 8	**show clock** [**detail**] **Example:** `Device# show clock detail`	ソフトウェア クロックの現在の時刻を表示します。
ステップ 9	**show ntp associations** [**detail**] **Example:** `Device# show ntp associations detail`	NTP アソシエーションのステータスを表示します。
ステップ 10	**show ntp status** **Example:** `Device# show ntp status`	NTP のステータスを表示します。
ステップ 11	**show sntp** **Example:** `Device# show sntp`	SNTP に関する情報を表示します。

時間範囲の設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **time-range** *time-range-name*
4. 次のいずれか 1 つを入力します。
   - **absolute** [**start** *hh*:*mm date month year*] [**end** *hh*:*mm date month year*]
   - **periodic** *day-of-the-week hh*:*mm* **to** [*day-of-the-week*] *hh*:*mm*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**time-range** *time-range-name* **Example:** `Device(config)# time-range range1`	設定する時間範囲に名前を割り当て、時間範囲コンフィギュレーション モードを開始します。

ステップ 4	次のいずれか 1 つを入力します。
	• **absolute** [**start** *hh*:*mm date month year*] [**end** *hh*:*mm date month year*]
	• **periodic** *day-of-the-week hh*:*mm* **to** [*day-of-the-week*] *hh*:*mm*
	Example:
	`Device(config-time-range)# absolute start 12:12 30 January 1999 end 12:12 30 December 2000`
	`Device(config-time-range)# periodic monday 12:12 to friday 12:12`
		時間範囲が有効になる時期を指定します。
		• これらのコマンドをいくつか組み合わせて使用します。 **periodic** コマンドは複数指定できます。**absolute** コマンドは 1 つしか指定できません。

ステップ 5	**end** **Example:**	時間範囲コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

	Command or Action	Purpose
	`Device(config-time-range)# end`	

ネットワーク タイム プロトコルの確認

SUMMARY STEPS

1. **show clock** [**detail**]
2. **show ntp associations detail**
3. **show ntp status**

DETAILED STEPS

ステップ 1 show clock [detail]

このコマンドを使用すると、ソフトウェア クロックの現在の時刻が表示されます。次に、このコマンドの出力例を示します。

Example:

```
Device# show clock detail

*18:38:21.655 UTC Tue Jan 4 2011
Time source is hardware calendar
```

ステップ 2 show ntp associations detail

このコマンドを使用すると、NTP アソシエーションのステータスが表示されます。次に、このコマンドの出力例を示します。

Example:

```
Device# show ntp associations detail

192.168.10.1 configured, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
our mode active, peer mode unspec, our poll intvl 64, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 15940.56
delay 0.00 msec, offset 0.0000 msec, dispersion 15937.50
precision 2**24, version 4
org time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
rec time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
xmt time D0CDE881.9A6A9005 (18:42:09.603 UTC Tue Jan 4 2011)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
192.168.45.1 configured, insane, invalid, unsynced, stratum 16
ref ID .INIT., time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
our mode client, peer mode unspec, our poll intvl 64, peer poll intvl 1024
root delay 0.00 msec, root disp 0.00, reach 0, sync dist 16003.08
delay 0.00 msec, offset 0.0000 msec, dispersion 16000.00
precision 2**24, version 4
```

```
org time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
rec time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
xmt time 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
filtdelay =     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filtoffset =    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
filterror =  16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0 16000.0
minpoll = 6, maxpoll = 10
```

ステップ 3 show ntp status

このコマンドを使用すると、NTPのステータスが表示されます。次に、このコマンドの出力例を示します。

Example:

```
Device# show ntp status

Clock is synchronized, stratum 8, reference is 127.127.1.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
reference time is D25AF07C.4B439650 (15:26:04.294 PDT Tue Oct 21 2011)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 2.31 msec, peer dispersion is 1.20 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 16, last update was 10 sec ago.
```

ネットワーク タイム プロトコルの設定例

例：ネットワーク タイム プロトコルの設定

次の例では、ハードウェア クロックを内蔵したデバイスが、他の 2 つのシステムとのサーバアソシエーションを確立し、ブロードキャスト NTP パケットを送信し、ハードウェア クロックを定期的に更新し、時刻を VINES に再配信します。

```
clock timezone PST -8
clock summer-time PDT recurring

ntp server 192.168.13.57
ntp server 192.168.11.58
interface GigabitEthernet 0/0
 ntp broadcast
vines time use-system
```

次の例では、ハードウェア クロックを内蔵したデバイスは外部の時刻源を持たないため、ハードウェア クロックを正規の時刻源として使用し、NTP ブロードキャスト パケットを介して時刻を配信します。

```
clock timezone MET 2
clock calendar-valid
ntp master
interface vlan 3
 ntp broadcast
```

次の例は、Line Aux 0 オプションがデフォルトで無効になっていることを示しています。

```
config-register 0x0
reload
rommon 1 > set
rommon 2 >  AUX_PORT=1
rommon 3 > SYNC
rommon 4 > reset
rommon 1 > set
rommon 2 > confreg 0x2102
rommon 3 > reset
```

ネットワーク タイム プロトコルの関連資料

関連資料

関連項目	マニュアル タイトル
基本的なシステム管理コマンド	『Basic System Management Command Reference』
IPv6 の NTP4	『*Cisco IOS Basic System Management Guide*』
IP 拡張アクセス リスト	『*Cisco IOS IP Addressing Configuration Guide*』
IPX 拡張アクセス リスト	『*Novell IPX Configuration Guide*』
NTP パッケージの脆弱性	『*Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability*』
Cisco IOS および NX-OS ソフトウェア リリース	『*White Paper: Cisco IOS and NX-OS Software Reference Guide*』

標準および RFC

標準および RFC	タイトル
RFC 1305	『*Network Time Protocol (Version 3) Specification, Implementation and Analysis*』

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

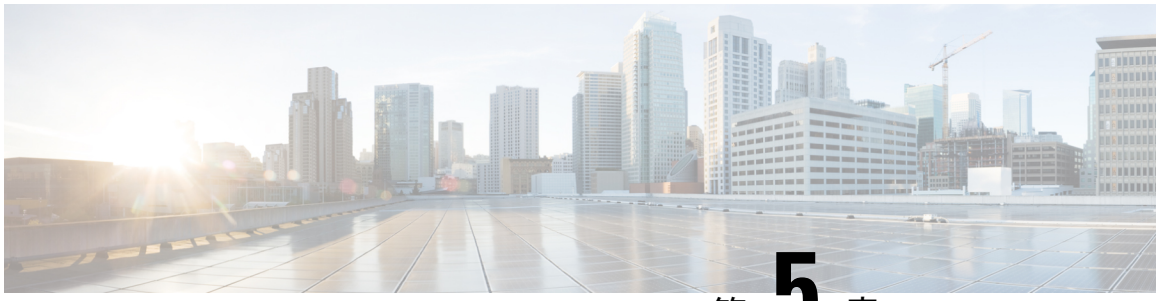
ネットワーク タイム プロトコルの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 4: ネットワーク タイム プロトコルの機能情報

機能名	リリース	機能情報
ネットワーク タイム プロトコル	11.2(1) 12.2(28)SB 12.2(33)SRA 12.2(33)SXI 12.2(33)SXJ 12.2(50)SY 12.2(58)SE 15.0(1)M 15.1(2)S 15.1(2)SG Cisco IOS XE リリース 3E	NTP は、ネットワーク接続されたマシンの時刻を同期させる目的で設計されたプロトコルです。NTP は UDP で稼働し、UDP は IP 上で稼働します。NTP は RFC 1305 で規定されています。 次のコマンドが導入または変更されました。**ntp access-group**、**ntp allow mode passive**、**ntp authenticate**、**ntp authentication-key**、**ntp broadcast, ntp broadcast client**、**ntp broadcastdelay**、**ntp clear drift**、**ntp clock-period**、**ntp disable**、**ntp logging**、**ntp primary**、**ntp max-associations**、**ntp multicast**、**ntp multicast client**、**ntp server**、**ntp source**、**ntp trusted-key** および **ntp update-calendar**。



第 5 章

Simple Network Time Protocol

Simple Network Time Protocol（SNTP）は、Network Time Protocol（NTP）の簡易バージョンです。このモジュールでは、シスコデバイスで Simple Network Time Protocol を設定する方法について説明します。

Simple Network Time Protocol に関する制約事項

- Simple Network Time Protocol（SNTP）と Network Time Protocol（NTP）は、同じポートを使用するため、同じマシン上で共存できません。つまり、これら2つのサービスをシステムで同時に設定することはできません。
- IPv6 アドレスのサポートは、イメージが IPv6 アドレッシングをサポートしている場合にのみ使用できます。

Simple Network Time Protocol について

Simple Network Time Protocol

簡易ネットワーク タイム プロトコル（SNTP）とは、クライアント専用バージョンの簡易版 NTP です。SNTP は、NTP サーバから時刻を受信できるだけで、時刻サービスを他のシステムに提供できません。

通常、SNTP は 100 ミリ秒以内の精度で時刻を提供しますが、NTP のような複雑なフィルタリングや統計メカニズムは提供しません。また、拡張アクセス リストを設定することによってある程度の保護を提供できますが、トラフィックを認証できません。SNTP クライアントは、NTP

クライアントよりも予期しない動作をするサーバーに対して脆弱であるため、強力な認証が必要ない状況でのみ使用する必要があります。

SNTP は、設定済みのサーバーからパケットを要求して受け入れるように設定するか、任意の送信元から NTP ブロードキャスト パケットを受け入れるように設定できます。複数の送信元が NTP パケットを送信している場合、最適な層にあるサーバーが選択されます（階層の説明については、3 ページの「*Network Time Protocol*」セクションを参照してください）。複数のサーバーのストラタムが同じだった場合は、ブロードキャスト サーバーよりも設定済みサーバーが優先されます。これらの両方を満たすサーバーが複数ある場合は、時刻パケットを最初に送信したサーバーが選択されます。SNTP が新しいサーバを選択するのは、現在選択しているサーバからのパケットの受信を停止している場合、または（上記の基準に従って）より適切なサーバが検出された場合だけです。

Simple Network Time Protocol の設定方法

Simple Network Time Protocol（SNTP）認証の設定

Simple Network Time Protocol（SNTP）は、Network Time Protocol（SNTP）の簡易バージョンです。このモジュールでは、シスコデバイスで SNTP を設定する方法について説明します。

手順の概要

1. **enable**
2. **configure terminal**
3. **sntp authenticate**
4. **sntp authentication-key** *number* **md5** *key*
5. **sntp trusted-key** *key-number* [- *end-key*]
6. **sntp server** *ip-address* **key** *key-id*
7. **end**

手順の詳細

	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。
ステップ 3	**sntp authenticate** 例： `Device(config)# sntp authenticate`	SNTP 認証機能をイネーブルにします。

	コマンドまたはアクション	目的
ステップ 4	**sntp authentication-key** *number* **md5** *key* 例： `Device(config)# sntp authentication-key 1 md5 key1`	認証キーを定義します。 • キーごとに、キー番号、タイプ、および値を 1 つずつ指定します。 • 追加の認証キーを定義するには、この手順を繰り返します。
ステップ 5	**sntp trusted-key** *key-number* [**-** *end-key*] 例： `Device(config)# sntp trusted-key 1 - 3`	信頼できる認証キーを定義します。 • キーを信頼できる場合、このデバイスは、このキーを SNTP パケット内で使用する別のシステムに同期できます。
ステップ 6	**sntp server** *ip-address* **key** *key-id* 例： `Device(config)# sntp server 172.16.22.44 key 2`	SNTP タイムサーバーによってソフトウェアクロックが同期されるように設定します。
ステップ 7	**end** 例： `Device(config)# end`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Simple Network Time Protocol の確認とトラブルシューティング

Simple Network Time Protocol の設定を確認してトラブルシューティングするには、次のコマンドを使用します。

•

手順の概要

1. **enable**
2. **debug sntp packets** [**detail**]
3. **debug sntp select**
4. **show sntp**

手順の詳細

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 **debug sntp packets** [**detail**]

例：

```
Device> debug sntp packets
```

送受信された NTP パケットを SNTP パケットフィールドとともに表示します。

ステップ 3 **debug sntp select**

例：

```
Device> debug sntp select
```

IPv4 および IPv6 サーバーの SNTP サーバーの選択を表示します。

ステップ 4 **show sntp**

例：

```
Device# show sntp

SNTP server     Stratum   Version    Last Receive
172.168.10.1      16        1        never
Broadcast client mode is enabled.
Multicast client 224.0.1.1 is enabled.
```

Cisco デバイスで使用可能な SNTP に関する情報を表示します。

Simple Network Time Protocol の設定例

例：Simple Network Time Protocol の設定

```
clock timezone PST -8
clock summer-time PDT recurring
sntp update-calendar
sntp server 192.168.13.57
sntp server 192.168.11.58
interface Ethernet 0/0
 sntp broadcast
```

Simple Network Time Protocol の追加資料

関連資料

関連項目	マニュアル タイトル
基本的なシステム管理コマンド	『Basic System Management Command Reference』
IPv6 の NTP4	『*Cisco IOS Basic System Management Guide*』

関連項目	マニュアル タイトル
IP 拡張アクセス リスト	『*Cisco IOS IP Addressing Configuration Guide*』
IPX 拡張アクセス リスト	『*Novell IPX Configuration Guide*』
NTP パッケージの脆弱性	『*Network Time Protocol Package Remote Message Loop Denial of Service Vulnerability*』
Cisco IOS および NX-OS ソフトウェア リリース	*White Paper: Cisco IOS and NX-OS Software Reference Guide*

標準および RFC

標準および RFC	タイトル
RFC 1305	『*Network Time Protocol (Version 3) Specification, Implementation and Analysis*』

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

SNTP の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

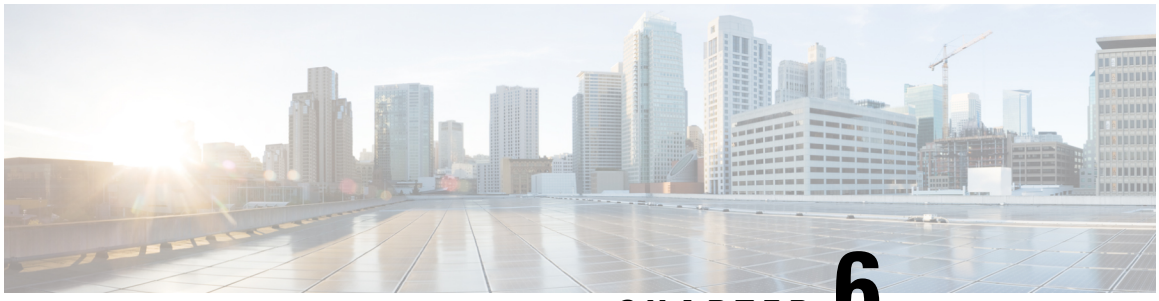
プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 5: SNTPv4 の機能情報

機能名	リリース	機能情報
Simple Network Time Protocol		Simple Network Time Protocol（SNTP）は、Network Time Protocol（NTP）の簡易バージョンです。このモジュールでは、シスコデバイスで Simple Network Time Protocol を設定する方法について説明します。 次のコマンドが導入または変更されました。**sntp server**、**sntp authenticate**、**sntp authentication-key**、**sntp multicast**、**sntp trusted-key**。

第 II 部

設定の基礎



CHAPTER 6

Cisco IOS コマンドライン インターフェイスの使用

Cisco IOS コマンドライン インターフェイス（CLI）は、シスコ デバイスの設定、監視、およびメンテナンスに使用される主要なユーザー インターフェイスです。このユーザー インターフェイスは、ルータ コンソールや端末、またはリモート アクセス方式を使用して、Cisco IOS コマンドを直接シンプルに実行することを可能にします。

この章では、Cisco IOS CLI の基本的な機能とその使用方法について説明します。この章で扱うトピックは、Cisco IOS コマンド モードの概要、ナビゲーションおよび編集機能、ヘルプ機能、コマンド履歴機能です。

追加ユーザー インターフェイスには、セットアップ モード（初回の起動に使用）、Cisco Web ブラウザ、およびシステム管理者が設定したユーザー メニューが含まれます。セットアップ モードの詳細については、「セットアップ モードを使用したシスコ ネットワーキング デバイスの設定」および「自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定」を参照してください。シスコ Web ブラウザを使用したコマンドの実行については、「Cisco Web ブラウザ ユーザー インターフェイスの使用」を参照してください。ユーザー メニューの詳細については、「接続、メニュー、およびシステム バナーの管理」を参照してください。

この章のユーザー インターフェイス コマンドの完全な説明については、『*Cisco IOS Configuration Fundamentals Command Reference*』を参照してください。この章で説明される他のコマンドの資料を検索するには、『Cisco IOS Master Command List, All Releases』を使用します。

Cisco IOS XE CLI コマンド モードの概要

シスコ デバイスの設定を支援するために、Cisco IOS XE コマンドライン インターフェイスは、さまざまなコマンド モードに分かれています。各コマンド モードには、ルータとネットワークの動作を設定、メンテナンス、モニタリングするための独自のコマンドセットがあります。常に使用可能なコマンドは、モードによって異なります。システム プロンプト（ルータ プロ

ンプト）で疑問符（**?**）を入力すると、各コマンド モードで使用できるコマンドのリストを取得できます。

特定のコマンドを使用すると、コマンドモードを変更できます。ユーザーがモードにアクセスする標準の順序は、ユーザーEXECモード、特権EXECモード、グローバル コンフィギュレーション モード、特定のコンフィギュレーション モード、コンフィギュレーション サブモード、およびコンフィギュレーション サブモードです。

ルータでセッションを開始するときは、通常、EXEC モードの 2 つあるアクセス レベルの 1 つであるユーザー *EXEC* モードから始めます。セキュリティのために、ユーザー EXEC モードで使用できる EXEC コマンドは制限されています。このアクセス レベルは、ルータのステータスを確認するなど、ルータの設定を変更しない作業のために予約されています。

すべてのコマンドにアクセスするには、EXEC モードの第 2 レベルである特権 *EXEC* モードを開始する必要があります。特権EXECモードを開始するには、通常、パスワードが必要です。特権 EXEC モードでは、任意の EXEC コマンドを入力できます。これは、特権 EXEC モードが、ユーザー EXEC モード コマンドのスーパーセットであるためです。

ほとんどの EXEC モード コマンドは、現在の設定ステータスを表示する **show** コマンドまたは **more** コマンドや、カウンタやインターフェイスをクリアする **clear** コマンドのように、1 回限りのコマンドです。EXEC モードのコマンドは、ルータをリブートすると保持されません。

特権EXECモードから、グローバル コンフィギュレーション モードを開始できます。このモードでは、一般的なシステム特性を設定するためのコマンドを実行できます。また、グローバル コンフィギュレーション モードを使用して特定のコンフィギュレーション モードを開始することもできます。グローバル コンフィギュレーション モードを含むコンフィギュレーション モードでは、実行コンフィギュレーションを変更できます。後で設定を保存すると、ルータをリブートしてもこれらのコマンドが保持されます。

グローバル コンフィギュレーション モードから、さまざまなプロトコル固有または機能固有のコンフィギュレーション モードを開始できます。CLI 階層では、グローバル コンフィギュレーション モードのみからこれらのコンフィギュレーション モードを開始できます。例として、この章では一般的に使用されるインターフェイス コンフィギュレーション モードについて説明します。

コンフィギュレーション モードから、コンフィギュレーション サブモードを開始できます。コンフィギュレーション サブモードは、特定のコンフィギュレーション モードの範囲内で特定の機能を設定するために使用します。たとえば、この章では、インターフェイス コンフィギュレーション モードのサブモードであるサブインターフェイス コンフィギュレーション モードについて説明します。

ROM モニター モードは、ルータが適切にブートできない場合に使用される、独立したモードです。システム（ルータ、スイッチ、またはアクセス サーバー）のブート時に適切なシステム イメージが見つからない場合、システムはROMモニターモードを開始します。ROMモニター（ROMMON）モードには、起動時にブート シーケンスに割り込むことでもアクセスできます。

Cisco IOS XE CLI の作業リスト

Cisco IOS XE CLI の機能に慣れるために、以降のセクションで説明する作業のいずれかを実行してください。

状況依存ヘルプの参照

システムプロンプトで疑問符（**?**）を入力すると、各コマンドモードで使用できるコマンドの一覧が表示されます。また、状況依存ヘルプ機能を使用して、任意のコマンドで使用できる引数とキーワードの一覧を参照できます。

コマンド モード、コマンド名、キーワード、または引数についてのヘルプ情報を表示するには、次のいずれかのコマンドを使用します。

コマンド	目的
(*prompt*)# **help**	ヘルプ システムの簡単な説明が表示されます。
(*prompt*)# *abbreviated-command-entry***?**	現在のモードの、特定の文字ストリングで始まるコマンドの一覧を表示します。
(*prompt*)# *abbreviated-command-entry*<**Tab**>	特定のコマンド名を補完します。
(*prompt*)# **?**	そのコマンド モードで使用できるすべてのコマンドの一覧を表示します。
(*prompt*)# *command***?**	コマンドに使用できる構文オプション（引数およびキーワード）の一覧を表示します。
(*prompt*)# *command keyword* **?**	コマンドに次に使用できる構文オプションの一覧を表示します。

システム プロンプトは、現在のコンフィギュレーション モードによって変わることに注意してください。

状況依存ヘルプが使用される場合は、疑問符（**?**）の前のスペースが重要です。特定の文字シーケンスで始まるコマンドのリストを表示するには、それらの文字を入力し、その直後に疑問符（**?**）を入力します。スペースは含めません。この形式のヘルプは、ユーザーに代わって 1 つの単語を完成させるため、ワードヘルプと呼びます。詳細については、この章の「部分的なコマンド名の補完」セクションを参照してください。

キーワードまたは引数のリストを表示するには、キーワードまたは引数の代わりに疑問符（**?**）を入力します。**?** の前にはスペースを挿入します。この形式のヘルプは、コマンド構文ヘルプと呼びます。これは、すでに入力したコマンド、キーワード、および引数に基づいて、使用できるキーワードや引数が表示されるためです。

コマンドおよびキーワードは、一意の省略形として認識可能な文字数まで省略できます。たとえば、**configureterminal** コマンドは **configt** に省略できます。コマンドの省略形が一意であるため、ルータによって省略形が受け付けられ、コマンドが実行されます。

help コマンド（どのコマンド モードでも使用できます）を実行すると、次のようにヘルプ システムの説明が表示されます。

```
Router#
 help
Help may be requested at any point in a command by entering
a question mark '?'. If nothing matches, the help list will
be empty and you must back up until entering a '?' shows the
available options.
Two styles of help are provided:
1. Full help is available when you are ready to enter a
   command argument (e.g. 'show ?') and describes each possible
   argument.
2. Partial help is provided when an abbreviated argument is entered
   and you want to know what arguments match the input
   (e.g. 'show pr?'.)
```

help コマンドの出力が示すように、疑問符（**?**）を使用して部分的なコマンド名を補完したり（部分ヘルプ）、現在のコマンドを補完する引数またはキーワードの一覧を表示したりできます。

次に、状況依存ヘルプ機能を使用して、コンフィギュレーション モードでアクセス リストを作成する例を示します。

システム プロンプトで、**co** に続けて疑問符（**?**）を入力します。最後の文字と疑問符との間にはスペースを入れません。システムには **co** で始まるコマンドが表示されます。

```
Router# co?
configure  connect  copy
```

configure コマンドの後にスペースと疑問符を入力すると、そのコマンドのキーワードと簡単な説明の一覧が表示されます。

```
Router# configure ?
  memory    Configure from NV memory
  network   Configure from a TFTP network host
  overwrite-network  Overwrite NV memory from TFTP network host
  terminal  Configure from the terminal
  <cr>
```

一覧内の <cr> 記号（「cr」は復帰を表します）は、Return キーまたは Enter キーを押して、キーワードを追加せずにコマンドを実行することが 1 つの選択肢であることを示します。この例の出力に、configure コマンドのオプションが **configurememory**（NVRAM から設定）、**configurenetwork**（ネットワーク上のファイルから設定）、**configureoverwrite-network**（ネットワーク上のファイルから設定し、NVRAM のファイルを置き換える）、または **configureterminal**（端末接続から手動で設定）であることが示されます。ほとんどのコマンド

で、<cr>記号は、入力済みの構文でコマンドを実行できることを示すために使用されます。ただし、configure コマンドは特殊であり、CLI によって不足している構文の入力を求められます。

```
Router# configure
Configuring from terminal, memory, or network [terminal]? terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

? プロンプトに対するデフォルトの応答は、CLI 出力中の行末にある角カッコで囲まれたオプションによって示されます。前の例では、Enter（またはReturn）キーを押すことは、「terminal」の単語を入力することと同じです。

configureterminal コマンドを入力して、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。

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

CLI では、エラー インジケータであるキャレット記号（^）を使用してエラーの位置が示されます。^ 記号は、コマンド構文中の、ユーザーが正しくないか認識されないコマンド構文を入力した場所に表示されます。たとえば、次の出力のキャレット記号は、コマンド中の入力ミスした文字を示しています。

```
Router# configure terminal
                   ^
% Invalid input detected at '^' marker.
Router#
```

エラー マーカーを警告するため、画面上にエラー メッセージ（% 記号によって示されます）が表示されることに注意してください。

access-list コマンドの後にスペースと疑問符を入力すると、コマンドで使用できるオプションの一覧が表示されます。

```
Router(config)# access-list ?
 <1-99>             IP standard access list
 <100-199>          IP extended access list
 <1100-1199>        Extended 48-bit MAC address access list
 <1300-1999>        IP standard access list (expanded range)
 <200-299>          Protocol type-code access list
 <2000-2699>        IP extended access list (expanded range)
 <700-799>          48-bit MAC address access list
 dynamic-extended   Extend the dynamic ACL absolute timer
 rate-limit         Simple rate-limit specific access list
```

山カッコ内の 2 つの数は包含範囲を表します。アクセス リスト番号 **99** を入力し、再度疑問符を入力すると、キーワードに該当する引数と簡単な説明が表示されます。

```
Router(config)# access-list 99 ?
  deny    Specify packets to reject
  permit  Specify packets to forward
```

deny 引数の後に疑問符（**?**）を入力すると、追加のオプションの一覧が表示されます。

```
Router(config)# access-list 99 deny ?
  A.B.C.D  Address to match
```

一般に大文字は変数（引数）を表します。IP アドレスに続けて疑問符（**?**）を入力すると、追加のオプション一覧が表示されます。

```
Router(config)# access-list 99 deny 172.31.134.0 ?
  A.B.C.D  Mask of bits to ignore
  <cr>
```

この出力では、A.B.C.Dは、ワイルドカードマスクの使用が可能であることを示します。ワイルドカード マスクは、IP アドレスまたは IP アドレスの範囲を照合するための方法の 1 つです。たとえば、0.0.0.255 のワイルドカード マスクは、IP アドレスの 4 番目のオクテットに表示される、0 ～ 255 の範囲の番号に一致します。

ワイルドカード マスクに続けて疑問符（**?**）を入力すると、その他のオプションの一覧が表示されます。

```
Router(config)# access-list 99 deny 172.31.134.0 0.0.0.255 ?
<cr>
```

<cr> 記号は、それ以上キーワードや引数がないことを示します。Enter（または Return）キーを押してコマンドを実行します。

```
Router(config)# access-list 99 deny 172.31.134.0 0.0.0.255
```

システムではエントリがアクセス リスト 99 に追加され、サブネット 172.31.134.0 上のすべてのホストへのアクセスが拒否され、0 ～ 255 の範囲で終わる IP アドレスに対するビットが無視されます。

コマンドの no 形式および default 形式の使用

ほぼすべてのコンフィギュレーション コマンドに **no** 形式があります。一般に、**no** 形式を使用すると、機能が無効になります。**no** キーワードなしでコマンドを使用すると、無効にされた機能を再度有効にしたり、デフォルトで無効になっている機能を有効にすることができます。たとえば、IP ルーティングはデフォルトで有効に設定されています。IP ルーティングを無効にするには、**iprouting** コマンドの **noiprouting** 形式を使用します。これを再度有効にするには、**iprouting** のプレーンな形式を使用します。Cisco IOS ソフトウェアのコマンド リファレンスの資料では、コマンドの **no** 形式が使用できる場合は常に **no** 形式の機能について説明しています。

多くの CLI コマンドには **default** 形式もあります。**default***command-name* コマンドを実行することで、コマンドをデフォルトの設定にすることができます。Cisco IOS ソフトウェアのコマンド リファレンス マニュアルでは、**default** 形式が、コマンドのプレーン形式か **no** 形式と異なる機能を実行する場合、一般にコマンドの **default** 形式の機能を説明しています。システムで使用できるデフォルト コマンドを表示するには、コマンドライン インターフェイスの該当するコマンド モードで **default?** と入力します。

コマンド履歴の使用

Cisco IOS CLI では、入力したコマンドの履歴（記録）が提供されます。この機能は、アクセスリストなど、長いまたは複雑なコマンドやエントリを呼び出す場合、特に便利です。コマンド履歴機能を使用するには、以降の項で説明するいずれかの作業を実行します。

CLI 編集機能とショートカットの使用

Cisco IOS CLI では、さまざまなショートカットと編集機能が使用できます。以降のサブセクションで次の機能について説明します。

コマンド ラインでのカーソルの移動

次の表に、修正または変更を加える際、コマンドラインでカーソルを移動するために使用できるキーの組み合わせまたはキー シーケンスを示します。Ctrl は Control キーを示し、対応する文字キーと同時に押す必要があります。Esc は Escape キーを示し、最初に押してから対応する文字キーを押します。キーの大文字と小文字は区別されません。CLI のナビゲーションと編集で使用される文字の多くは、その機能を簡単に覚えておけるように選択されています。次の表では、使用される文字と機能の関係を示すために「機能の概要」の列の文字が太字で示されています。

Table 6: カーソルを移動するために使用するキーの組み合わせ

キーストローク	機能の要約	機能の詳細
Left Arrow または **Ctrl+B**	**B** 1 文字戻る	カーソルを 1 文字左に移動します。複数行にわたってコマンドを入力するときは、左矢印キーまたは Ctrl+B キーを繰り返し押してシステム プロンプトまでスクロール バックして、コマンド エントリの先頭まで移動できます。あるいは Ctrl+A キーを押してコマンド エントリの先頭に移動します。
Right Arrow または **Ctrl+F**	**F** 1 文字進む	カーソルを 1 文字右に移動します。
Esc , **B**	**B** 1 単語戻る	カーソルを 1 単語後退させます。
Esc , **F**	**F** 1 単語進む	カーソルを 1 単語前進させます。
Ctrl -A	行の先頭	カーソルを行の先頭に移動します。
Ctrl -E	**E** 行末	カーソルをコマンド ラインの末尾に移動します。

部分的なコマンド名の補完

完全なコマンド名を思い出せない場合や、入力の作業量を減らす場合は、コマンドの先頭の数文字を入力して、Tab キーを押します。コマンドライン パーサーは、入力されたストリングが

コマンド モードで一意である場合に、コマンドを補完します。キーボード上に Tab キーがない場合は、代わりに **Ctrl** キーを押した状態で **I** キーを押します。

コマンドは、コマンドが一意になるのに十分な文字が入力されていれば認識されます。たとえば、特権 EXEC モードで **conf** と入力すると、CLI はエントリを **configure** コマンドと関連付けることができます。これは、**conf** で始まるコマンドが **configure** コマンドのみであるためです。

次の例で、Tab キーを押すと、特権 EXEC モードの **conf** に対する一意のストリングが認識されます。

```
Router# conf
<Tab
>
Router# configure
```

コマンド補完機能を使用すると、CLI により完全なコマンド名が表示されます。Return キーか Enter キーを押すまでコマンドは実行されません。これにより、完全なコマンドが省略形によって意図したものでない場合に、コマンドを修正できます。複数のコマンドに該当する文字列を入力した場合、テキスト ストリングが一意でないことを示すためにブザー音が鳴ります。

コマンドが補完できない場合は、疑問符（**?**）を入力して、その文字で始まるコマンドの一覧を表示します。入力した最後の文字と疑問符（**?**）の間にはスペースを入れません。

たとえば、**co?** を入力すると、現在のコマンド モードで使用可能なすべてのコマンドの一覧が表示されます。

```
Router# co?
configure connect copy
Router# co
```

疑問符の前に入力した文字は、コマンドを完全に入力できるように画面に表示されます。

削除したエントリの呼び出し

CLI では、削除したコマンドまたはキーワードが履歴バッファに格納されます。スペースで始まるかスペースで終わるストリングだけがバッファに格納され、削除した個別の文字（Backspace または Ctrl+D を使用）は格納されません。バッファには、Ctrl+K、Ctrl+U、または Ctrl+X で削除された最後の 10 個の項目が格納されます。これらの項目を呼び出してコマンドラインに貼り付けるには、次のキーの組み合わせを使用します。

キーストローク	目的
Ctrl -**Y**	バッファ内の最新のエントリを呼び出します（キーを同時に押します）。
Esc , **Y**	履歴バッファ内の前のエントリを呼び出します（キーは順番に押します）。

Esc、Y キー シーケンスは、最初に Ctrl+Y キーの組み合わせを押さない限り機能しません。Esc、Y を 11 回以上押すと、バッファ内の最新のエントリに戻ります。

画面幅よりも長いコマンドラインの編集

CLI には、画面上の 1 行を超えるコマンドに対する折り返し機能が備わっています。カーソルが右マージンに達すると、そのコマンドラインは 10 文字分だけ左へシフトされます。コマンドラインの先頭から 10 文字までは見えなくなりますが、左へスクロールして、コマンドの先頭部分の構文をチェックできます。スクロールで戻るには、Ctrl+B キーまたは←キーを繰り返し押してコマンド エントリの先頭に戻るか、Ctrl+A キーを押して直接行の先頭に戻ります。

次の例では、**access-list** コマンド エントリが 1 行を超えています。最初にカーソルが行末に達すると、その行は10 文字分だけ左へシフトされ、再表示されます。ドル記号（$）は、行が左にスクロールされたことを示しています。カーソルが行末に達するたびに、その行は再び 10 文字分だけ左へシフトされます。

```
Router(config)# access-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.1
Router(config)# $ 101 permit tcp 172.31.134.5 255.255.255.0 172.31.135.0 255.25
Router(config)# $t tcp 172.31.134.5 255.255.255.0 172.31.135.0 255.255.255.0 eq
Router(config)#
$31.134.5 255.255.255.0 172.31.135.0 255.255.255.0 eq 45
```

入力を完了したら、Return キーを押してコマンドを実行する前に、**Ctrl-A** キーを押して、完全な構文を確認します。行が右にスクロールしていることを示すため、ドル記号（$）が行末に表示されます。

```
Router(config)# access-list 101 permit tcp 172.31.134.5 255.255.255.0 172.31.1$
```

Cisco IOS XE ソフトウェアでは、幅が 80 カラムの端末画面を使用していると仮定しています。画面の幅が異なる場合は、**terminal width** ユーザー EXEC コマンドを使用して端末の幅を設定します。

ライン ラップとコマンド履歴機能を組み合わせることで、以前の複雑なコマンド エントリを呼び出したり修正したりできます。以前のコマンドエントリを呼び出す方法については、この章のコマンドのリコールに関するセクションを参照してください。

エントリの削除

入力を間違えた場合や考え直した場合に、コマンドエントリを削除するには、次のキーまたはキーの組み合わせを使用します。

キーストローク	目的
Delete または **Backspace**	カーソルの左にある文字を削除します。
Ctrl -D	カーソル位置にある文字を削除します。
Ctrl -K	カーソル位置からコマンド ラインの末尾までのすべての文字を削除します。
Ctrl +U または **Ctrl+X**	カーソル位置からコマンド ラインの先頭までのすべての文字を削除します。
Ctrl -W	カーソルの左にある単語を削除します。

キーストローク	目的
Esc , **D**	カーソルの位置から単語の末尾までを削除します。

--More-- プロンプトでの出力の続行

Cisco IOS XE CLI を使用する場合、出力が画面に表示可能な長さを超えることがあります。多くの **?** や **show** または **more** コマンドの出力などで画面の下端を超えて出力が続く場合は、出力が中断し、画面の最後の行に --More-- プロンプトが表示されます。出力を再開するには、Return キーを押して下に 1 行スクロールするか、スペースキーを押して出力の次の 1 画面分を表示します。

Tip 出力が画面上で一時停止していて、--More-- プロンプトが表示されない場合は、**length** ライン コンフィギュレーション コマンドまたは **terminal length** 特権 EXEC モード コマンドを使用して、画面の長さに入力する値を小さくします。**length** の値をゼロにすると、コマンド出力は一時停止しなくなります。

--More-- プロンプトからの出力のフィルタリングに関する情報については、この章の CLI 出力の検索とフィルタリングに関するモジュールを参照してください。

現在のコマンドラインの再表示

コマンドを入力していて、突然システムから画面にメッセージが表示された場合、現在のコマンド ライン エントリを簡単に呼び出すことができます。現在のコマンド ラインを再表示（画面を更新）するには、次のキーの組み合わせのうちいずれかを使用します。

キーストローク	目的
Ctrl +**L** または **Ctrl**+**R**	現在のコマンドラインを再表示します。

誤って入力した文字の置き換え

コマンド入力をミスした場合、入力ミスした文字を入れ替えることができます。文字を入れ替えるには、次のキーの組み合わせを使用します。

キーストローク	目的
Ctrl -**T**	カーソルの左にある文字を、カーソルの右にある文字と置き換えます。

大文字と小文字の制御

単純なキー シーケンスで単語を大文字または小文字にしたり、文字セットを大文字にすることができます。ただし、Cisco IOS XE コマンドでは、一般に大文字と小文字が区別されず、通常

はすべて小文字で入力します。コマンドの大文字と小文字を変更するには、次のキー シーケンスを使用します。

キーストローク	目的
Esc , **C**	カーソルの場所にある文字を大文字にします。
Esc , **L**	カーソルの場所にある単語を小文字にします。
Esc , **U**	カーソルの位置から単語の末尾までを大文字にします。

キーストロークをコマンド エントリとして指定

特定のキーストローク（キーの組み合わせまたはシーケンス）をコマンドエイリアスとして認識するようにシステムを設定できます。つまり、ストロークを、コマンドを実行するためのショートカットとして設定できます。システムにキーストロークをコマンドとして解釈させるには、コマンドシーケンスを入力する前に、次のいずれかのキーの組み合わせを使用します。

キーストローク	目的
Ctrl +**V** または **Esc**、**Q**	システムが次のキーストロークをユーザー コンフィギュレーション コマンド エントリとして受け付けるように設定します（編集コマンドとしてではありません）。

編集機能の無効化と再有効化

前のセクションで説明した編集機能はシステムで自動的に有効になります。しかし、これらの編集機能を無効にすることが望ましい状況がいくつかあります。たとえば、編集機能と競合するスクリプトがある場合です。編集機能をグローバルに無効にするには、ライン コンフィギュレーション モードで次のコマンドを使用します。

コマンド	目的
`Router(config-line)# no editing`	特定の回線に対して CLI 編集機能を無効にします。

現在の端末セッションに対して編集機能を無効にするには、ユーザーEXECモードで次のコマンドを使用します。

コマンド	目的
`Router# no terminal editing`	ローカル ラインに対して CLI 編集機能を無効にします。

現在の端末セッションに対して編集機能を再度有効にするには、ユーザーEXECモードで次のコマンドを使用します。

コマンド	目的
`Router# terminal editing`	現在の端末セッションに対してCLI編集機能を有効にします。

特定の回線に対して編集機能を再度有効にするには、ライン コンフィギュレーション モードで次のコマンドを使用します。

コマンド	目的
`Router(config-line)# editing`	CLI編集機能を有効にします。

CLI 出力の検索とフィルタリング

Cisco IOS CLI には、大量のコマンド出力を検索したり、出力をフィルタリングして不要な情報を除外するための手段が提供されています。これらの機能は、一般に大量のデータが表示される、**show** コマンドと **more** コマンドで使用できます。

Note **Show** コマンドと **more** コマンドは、常にユーザー EXEC モードまたは特権 EXEC モードで実行します。

画面に表示される内容を超えて出力が続く場合、Cisco IOS CLI では --More-- プロンプトが表示されます。Return キーを押すことで次の行が表示され、スペースキーを押すことで次の画面が表示されます。CLI ストリング検索機能を使用すると、--More-- プロンプトからの出力を検索またはフィルタリングできます。

Cisco IOS XE CLI の使用の例

コマンド構文の確認とコマンド履歴の使用の例

CLI では、エラー インジケータであるキャレット記号（^）を使用してエラーの位置が示されます。^記号は、コマンドストリング内の誤ったコマンド、キーワード、または引数が入力された位置に表示されます。

次の例では、クロックを設定するものとします。状況依存ヘルプを使用して、クロックを設定するための正しいコマンド構文を確認します。

```
Router# clock ?
  set  Set the time and date
Router# clock
```

ヘルプ出力により、**set** キーワードが必要であることが示されます。時刻を入力するための構文を確認します。

```
Router# clock set ?
hh:mm:ss   Current time
Router# clock set
```

現在の時刻を入力します。

```
Router# clock set 13:32:00
% Incomplete command.
```

コマンドを完了するために追加の引数を指定する必要があることがシステムによって示されます。Ctrl+P キーまたは↑キーを押して、以前のコマンド入力を自動的に繰り返します。次にスペースと疑問符（**?**）を追加し、他の引数を確認します。

```
Router# clock set 13:32:00 ?
  <1-31>     Day of the month
  MONTH Month of the year
```

これでコマンド入力を完了できます。

```
Router# clock set 13:32:00 February 01
                                      ^
% Invalid input detected at '^' marker.
```

キャレット記号（^）とヘルプ応答により、01 に誤りがあることが示されます。正しい構文の一覧を表示するために、エラーが発生した場所までコマンドを入力し、疑問符（**?**）を入力します。

```
Router# clock set 13:32:00 February ?
<1-31>     Day of the month
Router# clock set 13:32:00 February 23 ?
<1993-2035> Year
```

正しい構文を使用して年を入力し、Enter または Return を押してコマンドを実行します。

Router# clock set 13:32:00 February 23 2001

CLI 出力の検索とフィルタリングの例

次に、**morenvram:startup-config|begin** 特権 EXEC モード コマンドの部分的な出力例を示します。これは、正規表現を含む最初の行で、フィルタリングされていない出力が開始されています。--More-- プロンプトで、正規表現 ip を含む出力行を除外するためのフィルタを指定します。

```
Router# more nvram:startup-config | begin ip
address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
!
security passwords min-length 1
!
no aaa new-model
```

```
ip subnet-zero
no ip domain lookup
ip host sjc-tftp02 171.69.17.17
ip host sjc-tftp01 171.69.17.19
ip host dirt 171.69.1.129
!
!
multilink bundle-name authenticated
!
!
redundancy
 mode sso
!
!
bba-group pppoe global
!
!
interface GigabitEthernet0/0/0
 ip address 10.4.9.158 255.255.255.0
 media-type rj45
 speed 1000
 duplex full
 negotiation auto
 no cdp enable
!
interface GigabitEthernet0/0/1
 no ip address
 media-type rj45
 speed 1000
 duplex full
 negotiation auto
 no cdp enable
!
interface POS0/1/0
 no ip address
 shutdown
 no cdp enable
!
interface POS0/1/1
 no ip address
 shutdown
 no cdp enable
!
interface GigabitEthernet0
 vrf forwarding Mgmt-intf
 no ip address
 speed 1000
 duplex full
 negotiation auto
!
ip default-gateway 10.4.9.1
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 GigabitEthernet0/0/0
ip route 171.69.0.0 255.255.0.0 10.4.9.1
!
no ip http server
no ip http secure-server
!
!
snmp mib bulkstat schema E0
snmp mib bulkstat schema IFMIB
snmp mib bulkstat transfer 23
snmp mib bulkstat transfer bulkstat1
```

```
!
!
control-plane
!
!
line con 0
 exec-timeout 30 0
 logging synchronous
 stopbits 1
line aux 0
 stopbits 1
line vty 0 4
 privilege level 15
 password lab
 login
!
end
```

次に、**morenvram:startup-config|include** 特権 EXEC コマンドの部分的な出力例を示します。正規表現 ip を含む行だけが表示されています。

```
Router# more nvram:startup-config | include ip
ip subnet-zero
ip domain-name cisco.com
ip name-server 1192.168.48.48
ip name-server 172.16.2.132
```

次に、**morenvram:startup-config|exclude** 特権 EXEC コマンドの部分的な出力例を示します。正規表現 service を含む行が除外されています。--More-- プロンプトで、正規表現 Dialer1 をフィルタとして指定します。このフィルタを指定することにより、Dialer1 を含む最初の行で出力が再開されます。

```
Router# more nvram:startup-config | exclude service
!
version 12.2
!
hostname router
!
boot system flash
no logging buffered
!
ip subnet-zero
ip domain-name cisco.com
.
.
.
--More--
/Dialer1
filtering...
interface Dialer1
 no ip address
 no ip directed-broadcast
 dialer in-band
 no cdp enable
```

次に、出力の検索が指定された、**showinterface** ユーザー EXEC または特権 EXEC コマンドモードの出力例の一部を示します。パイプの後でキーワード **beginFastEthernet** を使用することで、正規表現 Fast Ethernet を含む最初の行でフィルタリングされていない出力が開始されます。--More-- プロンプトで、正規表現 **Serial** を含む行だけを表示するフィルタを指定します。

```
Router# show interface | begin FastEthernet
FastEthernet0/0 is up, line protocol is up
Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Description: ip address is 172.1.2.14 255.255.255.0
  Internet address is 172.1.2.14/24
.
.
.
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
--More--
+Serial
filtering...
Serial1 is up, line protocol is up
Serial2 is up, line protocol is up
Serial3 is up, line protocol is down
Serial4 is down, line protocol is down
Serial5 is up, line protocol is up
Serial6 is up, line protocol is up
Serial7 is up, line protocol is up
```

次に、**showbuffers|exclude** コマンドの出力例の一部を示します。正規表現 **0 misses** を含む行が除外されています。--More-- プロンプトで、フィルタされていない出力を、Serial0 を含む最初の行から続行するための検索を指定します。

```
Router# show buffers | exclude 0 misses
Buffer elements:
     398 in free list (500 max allowed)
Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
     50 in free list (20 min, 150 max allowed)
     551 hits, 3 misses, 0 trims, 0 created
Big buffers, 1524 bytes (total 50, permanent 50):
     49 in free list (5 min, 150 max allowed)
Very Big buffers, 4520 bytes (total 10, permanent 10):
.
.
.
Huge buffers, 18024 bytes (total 0 permanent 0):
     0 in free list (0 min, 4 max allowed)
--More--
/Serial0
filtering...
Serial0 buffers, 1543 bytes (total 64, permanent 64):
     16 in free list (0 min, 64 max allowed)
     48 hits, 0 fallbacks
```

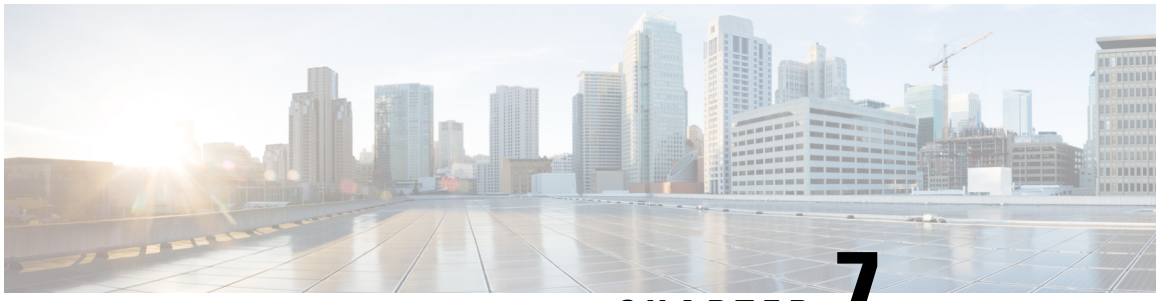
次に、**showinterface|include** ユーザー EXEC または特権 EXEC コマンド モードの部分的な出力例を示します。パイプ（|）の後で **include(is)** キーワードを使用することにより、正規表現 (is) が含まれる行だけが表示されます。カッコにより、is の前後にスペースが含まれることが指定されます。カッコを使用することで、isの前後にスペースを含む行だけが出力に含まれます（「disconnect」などの文字は検索から除外されます）。**) .**

```
router# show interface | include ( is )
ATM0 is administratively down, line protocol is down
  Hardware is ATMizer BX-50
Dialer0/1 is up (spoofing), line protocol is up (spoofing)
  Hardware is Unknown
  DTR is pulsed for 1 seconds on reset
```

```
FastEthernet0/0 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Internet address is 172.21.53.199/24
FastEthernet0/1 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.639c (bia 0060.837c.639c)
  Internet address is 10.5.5.99/24
Serial0:0 is down, line protocol is down
  Hardware is DSX1
.
.
.
 --More--
```

--More-- プロンプトで、Serial0:13 を含む最初の行でフィルタリングされた出力を続行する検索を指定します。

```
/Serial0:13
filtering...
Serial0:13 is down, line protocol is down
  Hardware is DSX1
  Internet address is 10.0.0.2/8
     0 output errors, 0 collisions, 2 interface resets
  Timeslot(s) Used:14, Transmitter delay is 0 flag
```



CHAPTER 7

show コマンド出力リダイレクション

show コマンド出力リダイレクション機能は、Cisco IOS コマンドライン インターフェイス（CLI）の **show** コマンドおよび **more** コマンドの出力をファイルにリダイレクトする機能を提供します。

show コマンド出力リダイレクションについて

この機能では Cisco IOS CLI の **show** コマンドを強化し、後から参照するために大量のデータ出力をファイルに直接書き込むことができます。このファイルはフラッシュ、SAN ディスク、あるいは外部メモリ デバイスなどのローカルまたはリモート ストレージ デバイスに保存できます。

発行される各 **show** コマンドにつき、新しいファイルを作成したり、出力を既存のファイルに追加したりできます。オプションで、**tee** キーワードを使用して、ファイルにリダイレクトしながらコマンド出力を画面表示できます。リダイレクトは、次のキーワードと組み合わせて、任意の **show** コマンドに続けてパイプ（|）文字を使用すると実行できます。

出力リダイレクション キーワード：

キーワード	使用法
append	URL（アペンド動作をサポートしている URL のみ）にリダイレクト出力をアペンドします
begin	一致する行から開始します
count	regexp に一致する行数をカウント
exclude	一致する行を除外

キーワード	使用法
format	指定されたスペック ファイルを使用して出力をフォーマットします
include	一致する行を含める
redirect	URL に出力をリダイレクトします
tee	URL に出力をコピーします

これらの拡張は **more** コマンドにも追加できます。

show コマンド拡張機能の使用方法

この機能拡張に関連付けられているコンフィギュレーション作業はありません。使用上のガイドラインについては、「関連資料」セクションに記載されているコマンドリファレンスを参照してください。

その他の参考資料

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
Cisco IOS コンフィギュレーション コマンド	*Cisco IOS Configuration Fundamentals Command Reference*

標準

標準	タイトル
この機能がサポートする新しい規格または変更された規格はありません。また、この機能による既存規格のサポートに変更はありません。	--

MIB

MIB	MIB のリンク
• 新しい MIB または変更された MIB はサポートされていません。また、既存の MIB に対するサポートに変更はありません。	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャ セットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
新しい RFC または変更された RFC はサポートされていません。また、既存の RFC に対するサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

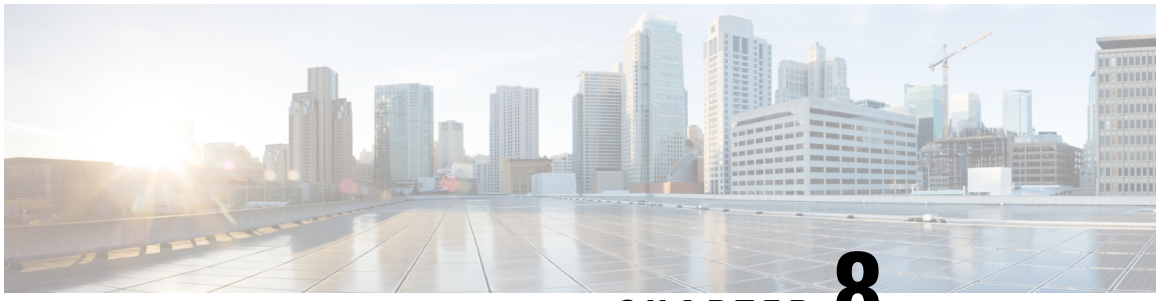
show コマンド出力リダイレクションの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェア イメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 7: show コマンド出力リダイレクション機能の機能情報

機能名	リリース	機能情報
show コマンド出力リダイレクション	12.0(21)S 12.2(13)T	• show コマンド出力リダイレクション機能は、Cisco IOS コマンドライン インターフェイス（CLI）の **show** コマンドおよび **more** コマンドの出力をファイルにリダイレクトする機能を提供します。 次のコマンドが導入または変更されました。**show**、**more**



CHAPTER 8

シスコ ネットワーキング デバイスの基本設定の概要

Cisco IOS ソフトウェアでは、Cisco IOS ベースのネットワーキング デバイスの設定を単純化するために、自動インストールとセットアップ モードの 2 つの機能が提供されています。自動インストールを使用すると、デバイス コンフィギュレーション ファイルを離れた場所から自動的にロードし、それを使用して複数のデバイスを同時に設定できます。セットアップは、システムの基本（スタートアップとも呼びます）設定をガイドする対話型の Cisco IOS ソフトウェア コマンドライン インターフェイス（CLI）モードですが、一度に設定できるのは 1 台のデバイスに制限されます。自動インストールは、設定するデバイスに対する自動的なプロセスですが、セットアップは設定するデバイスに対する手動のプロセスです。

このモジュールは各機能について紹介し、機能を詳細に説明するモジュールを示し、その使用方法について説明します。

初期設定という用語とスタートアップ コンフィギュレーションという用語は、同じ意味で使用されます。

シスコ ネットワーキング デバイスの基本設定における前提条件

Cisco IOS 自動インストールの前提条件

- 「自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定」モジュールは、Cisco IOS Release 12.4(1) 以降が動作するネットワーキング デバイス向けに

書かれています。しかし、このマニュアルのほとんどの情報は、自動インストールをサポートしている、Cisco IOS release 12.4(1) 以降が動作していないネットワーキング デバイスに対して使用できます。念頭に置くべき主な違いは次の 2 つです。

- 一部のシスコ ネットワーキング デバイスは、DHCP の代わりに BOOTP を使用して、LAN インターフェイス上で IP アドレスを要求します。DHCP サーバーで BOOTP のサポートを有効にすることで、この問題が解決されます。
- 一部のシスコ ネットワーキング デバイスでは、DHCP クライアント ID の形式が、Cisco IOS release 12.4(1) 以降が動作するネットワーキング デバイスのものと異なります。このマニュアルでは、Cisco IOS release 12.4(1) 以降が動作するネットワーキング デバイスで使用されている DHCP クライアント ID 形式についてだけ説明します。現在のシスコ ネットワーキング デバイスが使用している DHCP クライアント ID の形式を特定するには、「自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定」モジュールの「自動的な DHCP クライアント ID の特定」のセクションを参照してください

- 自動インストールを使用して設定するネットワーキング デバイス上の NVRAM にコンフィギュレーション ファイルが存在しないこと。
- 自動インストールを使用してネットワーキング デバイス上にロードするコンフィギュレーション ファイルが、ネットワークに接続されている TFTP サーバー上にあること。ほとんどの場合、ファイルは複数あります。たとえば、IP からホスト名へのマッピングが格納されたネットワーク ファイルと、デバイス固有のコンフィギュレーション ファイルです。
- 自動インストールを使用して設定するネットワーキング デバイスをネットワークに接続して電源を投入するために、リモート サイトに誰かがいること。
- 自動インストール プロセス中にネットワーキング デバイスが TFTP サーバーからコンフィギュレーション ファイルをロードできるように、ネットワークで IP 接続が可能であること。
- LAN 接続経由で自動インストールを使用してネットワーキング デバイスに IP アドレスを付与するため、ネットワーク上で DHCP サーバーが利用できること。

Cisco IOS セットアップ モードの前提条件

- 設定するデバイスのコンソール ポートに端末が接続されていること。
- 設定するインターフェイスがわかっていること。
- 有効にするルーティング プロトコルがわかっていること。

ルーティング プロトコルの詳細については、『*Cisco IOS IP Routing Protocols Configuration Guide*』を参照してください。

- 設定するデバイスがブリッジングを実行するかどうかがわかっていること。
- 設定するデバイスにプロトコル変換がインストールされているかどうかがわかっていること。
- 設定するプロトコルのネットワーク アドレスがわかっていること。

ネットワーク アドレスについては、『*Cisco IOS IP Addressing Services Configuration Guide*』を参照してください。

- ネットワーク環境のパスワード方針が決まっていること。

パスワードとデバイス セキュリティの詳細については、『*Cisco IOS Security Configuration Guide*』の「Configuring Security with Passwords, Privilege Levels, and Login User names for CLI Sessions on Networking Devices」を参照してください。

- 設定する製品のマニュアルが手元にあるか、アクセスできること。

シスコ ネットワーキング デバイスの基本設定における制約事項

Cisco IOS 自動インストールの制約事項

- （シリアル インターフェイスだけ）HDLC またはフレーム リレーを使用したシリアル インターフェイスでは、新しいデバイスの最初のシリアル ポート（シリアル インターフェイス 0 またはシリアル インターフェイス x/0）上だけで自動インストールを実行できます。
- （LAN インターフェイスだけ）物理的なジャンパを使用してリング速度を設定した LAN トークン リング インターフェイスだけで自動インストールがサポートされます。

Cisco IOS セットアップ モードの制約事項

- セットアップモードはハードウェア依存です。設定する製品のマニュアルに記載されている手順に従う必要があります。
- 一部のコンフィギュレーション パラメータは、ネットワーキング デバイスにプロトコル変換オプションがインストールされている場合にだけ適用されます。デバイスにプロトコル変換オプションがインストールされていない場合、これらのパラメータに対するプロンプトは表示されません。

シスコ ネットワーキング デバイスの基本設定に関する情報

基本設定を使用してネットワーキング デバイスを設定する前に、次の概念について理解し、要件に基づいて、自動インストールとセットアップ モードのどちらが最適な方法なのかを判断する必要があります。

Cisco IOS 自動インストールと Cisco IOS セットアップ モードの比較

Cisco IOS 自動インストールを使用すると、デバイス コンフィギュレーション ファイルを離れた場所から自動的にロードし、それを使用して複数のデバイスを同時に設定できます。セットアップは、システムの基本（スタートアップとも呼びます）設定をガイドする対話型の Cisco IOS ソフトウェア CLI モードですが、一度に設定できるのは 1 台のデバイスに制限されます。自動インストールは自動プロセスで、セットアップは手動プロセスです。

Cisco IOS 自動インストール

自動インストールは、中央のロケーションからリモート ネットワーキング デバイスの設定を可能にする Cisco IOS ソフトウェア機能です。コンフィギュレーション ファイルは、セットアップのために自動インストールを使用しているデバイスからアクセスできる TFTP サーバーに保存する必要があります。

自動インストールは、LAN、ハイレベル データリンク コントロール（HDLC）カプセル化を使用したシリアル インターフェイス、WAN 用のフレーム リレー カプセル化を使用したシリアル インターフェイス、および WIC-1-DSU-T1v2 カード（他の T1E1 カードでは自動インストールはサポートされていません）に対し、イーサネット、トークンリング、FDDI インターフェイス上でサポートされています。

自動インストールは、リモート サイトでの設置の中央での管理を容易にするように設計されています。自動インストール プロセスは、Cisco IOS ソフトウェアベースのデバイスの電源をオンにし、NVRAM に有効なコンフィギュレーション ファイルがない場合に開始されます。ネットワーキング デバイスに Cisco ルータと Security Device Manager（SDM）または Cisco Network Assistant がすでにインストールされている場合には、自動インストールは開始されません。この場合、自動インストールを有効にするには、SDM を無効にする必要があります。

『Using AutoInstall to Remotely Configure Cisco Networking Devices』モジュールでは、AutoInstall の動作、SDM を無効にする方法、AutoInstall を使用するようにデバイスを設定する方法が説明されています。

Cisco IOS セットアップ モード

Cisco IOS セットアップ モードを使用すると、Cisco IOS CLI またはシステム設定ダイアログを使用して初期設定ファイルを作成できます。初期設定手順がダイアログに表示されるため、シスコの製品や CLI に慣れておらず、CLI によって提供される詳細なレベルでの設定変更が不要な場合に便利です。

セットアップは、デバイスの NVRAM にコンフィギュレーション ファイルがなく、Cisco SDM を使用するように工場で事前設定されていない場合に開始されます。セットアップが完了すると、システム設定ダイアログが表示されます。ダイアログに従ってデバイスとネットワークに関する基本的な情報を入力することで初期設定が行われ、初期設定ファイルが作成されます。ファイルが作成された後、CLI を使用して追加の設定を行うことができます。

『Using Setup Mode to Configure a Cisco Networking Device』では、セットアップを使用して基本設定を作成する方法と、設定を変更する方法について説明しています。

次の作業

「自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定」モジュールまたは「セットアップ モードを使用したシスコ ネットワーキング デバイスの設定」モジュールに進んでください。

その他の参考資料

このセクションでは、シスコ ネットワーキング デバイスの基本設定に関する参考資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
設定の基本的なコマンド	『*Cisco IOS Configuration Fundamentals Command Reference*』
Cisco IOS ソフトウェアの自動インストール機能を使用した初めてのネットワーキング デバイスの設定	『*Cisco IOS Configuration Fundamentals Configuration Guide*』の「Using AutoInstall to Remotely Configure Cisco Networking Devices」モジュール
Cisco IOS セットアップ モードを使用したネットワーキング デバイスの設定	『*Cisco IOS Configuration Fundamentals Configuration Guide*』の「Using Setup Mode to Configure a Cisco Networking Device」モジュール

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

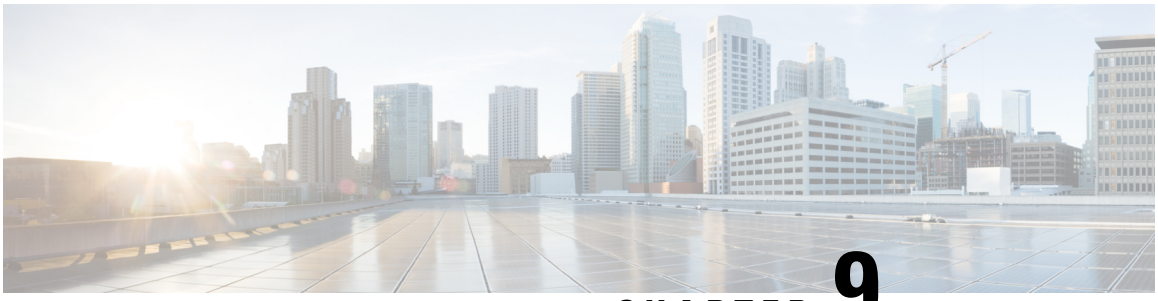
シスコ ネットワーキング デバイスの基本設定概要の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 8: 概要：シスコ ネットワーキング デバイスの基本設定の機能情報

機能名	リリース	機能情報
概要：シスコ ネットワーキング デバイスの基本設定	12.4(3)	Cisco IOS ソフトウェアでは、Cisco IOS ベースのネットワーキング デバイスの設定を単純化するために、自動インストールとセットアップ モードの 2 つの機能が提供されています。自動インストールを使用すると、デバイス コンフィギュレーション ファイルを離れた場所から自動的にロードし、それを使用して複数のデバイスを同時に設定できます。セットアップは、システムの基本（スタートアップとも呼びます）設定をガイドする対話型の Cisco IOS ソフトウェア コマンドライン インターフェイス（CLI）モードですが、一度に設定できるのは1台のデバイスに制限されます。自動インストールは、設定するデバイスに対する自動的なプロセスですが、セットアップは設定するデバイスに対する手動のプロセスです。



CHAPTER 9

自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定

自動インストールを使用すると、ネットワーキングデバイスをリモートから自動的に設定できます。一般に、自動インストールは、新しいネットワーキングデバイスをリモートからセットアップするために使用します。ただし、既存のネットワーキングデバイスについても、NVRAMからコンフィギュレーションファイルを削除した後で、自動インストールを使用して設定できます。自動インストール プロセスは、TFTP サーバーにあらかじめ格納されているコンフィギュレーション ファイルを使用します。

このモジュールでは、ネットワーキング デバイスという用語は、Cisco IOS ソフトウェアが動作するルータを指します。また、次の用語は同じ意味で使用されます。

- 初期設定およびスタートアップ コンフィギュレーション
- セットアップおよび設定

機能制限

- DHCP サーバーは、管理インターフェイス（ギガビット イーサネット 0）を介して到達可能である必要があります。
- 管理インターフェイス ギガビット イーサネット 0 だけがサポートされています。

  この機能を Cisco ASR 1000シリーズアグリゲーション サービス ルータで使用する場合は、ドキュメントのイーサネット インターフェイスをギガビット イーサネット インターフェイスと読み替えてください。

自動インストールを使用したシスコのネットワーキング デバイスのリモートでの設定に関する情報

自動インストールのIPアドレスのダイナミックな割り当てで使用するサービスとサーバー

ネットワークは、自動インストールを使用して設定するネットワーキング デバイスに対する IP アドレスのダイナミックな割り当てが可能であることが必要です。使用する IP アドレス割り当てサーバーの種類は、自動インストールを使用して設定するネットワーキング デバイスのネットワークに対する接続の種類によって変わります。

自動インストールは次の種類の IP アドレス サーバーを使用します。

DHCP Servers

LAN 接続上で自動インストールを使用するネットワーキング デバイスには、ダイナミックに IP アドレスを提供するために DHCP サーバーが必要です。この要件は、ファスト イーサネット、トークン リング、および FDDI のインターフェイスに適用されます。DHCP サーバーと、LAN 接続上で自動インストールを使用するすべてのデバイスとの間で、IP 接続が可能なようにネットワークが設定されている必要があります。

DHCP（RFC 2131 で規定）は、ブートストラップ プロトコル（RFC 951 で規定）により提供される機能を拡張したものです。DHCP は、設定情報を TCP/IP ネットワーク上のホストに渡すためのフレームワークを提供します。DHCP では、再利用可能なネットワーク アドレスと、ルータ（ゲートウェイ）の IP アドレス、TFTP サーバーの IP アドレス、ロードするブート ファイルの名前、使用するドメイン名など、追加の設定オプションを自動的に割り当てる機能が追加されています。DHCP サーバーは、ルータ、UNIX サーバー、Microsoft Windows ベースのサーバー、その他のプラットフォーム上で設定できます。

一般に DHCP サーバーは、IP アドレスのプールからランダムに IP アドレスを割り当てます。DHCP を使用するデバイスは、ネットワークに接続するたびに異なる IP アドレスを取得することがあります。これは、自動インストール プロセスの間、特定のデバイスに特定のホスト名

を割り当てる必要がある場合に問題になります。たとえば、リモートサイトの異なる階にルータを設置し、各ルータに、**ChicagoHQ-1st** や **ChicagoHQ-2nd** といった、その場所を示す名前を割り当てる場合、各デバイスの IP アドレスが、その正しいホスト名にマッピングされるようにする必要があります。

デバイスに特定の IP アドレスが割り当てられるようにするためのプロセスは、予約の作成と呼びます。予約とは、IP アドレスと、デバイス上の LAN インターフェイスの物理層アドレスの間の関係を、手動で設定することです。多くの Cisco IOS XE ベースのデバイスは、DHCP を通じて IP アドレスを要求する際に、その MAC アドレスを使用しません。代わりに、より長いクライアント ID を使用します。予約を事前に設定するためには、クライアント ID を特定しなくてはならず、新しいデバイスがその MAC アドレスとクライアント ID のどちらを使用するのかを知らなくてはなりません。デバイスが MAC アドレスとクライアント ID のどちらを使用しているかを特定するために、新しいデバイスが最初に DHCP 予約を使用せずに IP アドレスを取得できるようにすることを推奨します。新しいデバイスが DHCP サーバーに対して自身を識別する方法がわかったら、その形式をメモして、そのデバイス用の予約を作成します。次回デバイスがリブートした際に、予約した IP アドレスが取得され、新しいデバイスに正しいホスト名が割り当てられます。DHCP の予約の作成について、使用している DHCP サーバーソフトウェアに付属している情報を参照してください。Cisco IOS XE ベースの DHCP サーバーを使用して予約を作成する手順については、「自動インストールを使用した LAN に接続されているデバイス設定の例」のモジュールで説明しています。この項には、DHCP 予約を事前に設定できるように、デバイスがネットワークに接続される前にクライアント ID を特定するための手順が含まれています。

Note このマニュアルでは、自動インストールを使用して LAN に接続されているネットワーキングデバイスを設定するために、シスコのルータを DHCP サーバーとして使用します。別のデバイスを DHCP サーバーとして使用する場合は、設定時に参照できるように、そのユーザーマニュアルを手元に置いてください。

Note コンフィギュレーション パラメータには、TFTP サーバー アドレス、DNS サーバー アドレス、ドメイン名など、さまざまなものがあります。これらのパラメータは、DHCP サーバーにより、IP アドレスをクライアントに割り当てるプロセスの中で、LAN に接続されたクライアントに渡すことができます。これらのパラメータは自動インストールでは必要ないため、このマニュアルには記載されていません。これらのパラメータの使用方法を把握している場合は、ネットワーキング デバイスをセットアップするために自動インストールを使用しているときに、DHCP サーバーの設定に組み込むことができます。

DHCP サービスの詳細については、IETF RFC のサイト（http://www.ietf.org/rfc.html）で DHCP に関する RFC を参照してください。ほとんどのサーバー オペレーティング システムが DHCP サーバーをサポートしています。詳細については、使用しているオペレーティング システムに付属しているマニュアルを参照してください。

SLARP サーバー

HDLC カプセル化を使用してシリアル インターフェイス上で自動インストールを使用して設定するルータは、ステージング ルータに接続されているシリアル インターフェイス上の IP アドレスに対するシリアル ライン ARP（SLARP）要求を送信します。

ステージング ルータのシリアル インターフェイスには、192.168.10.1 や 192.168.10.2 など、ホスト ポートが 1 または 2 の IP アドレスが設定されている必要があります。ステージング ルータは、自動インストールで設定するルータに、ステージング ルータが使用していない値が格納された SLARP 応答を送信します。たとえば、自動インストールで設定するルータに接続されているステージング ルータ上のインターフェイスが、IP アドレスとして 192.168.10.1 を使用している場合、ステージング ルータは、自動インストールで設定するルータに対し、値が 192.168.10.2 の SLARP 応答を送信します。

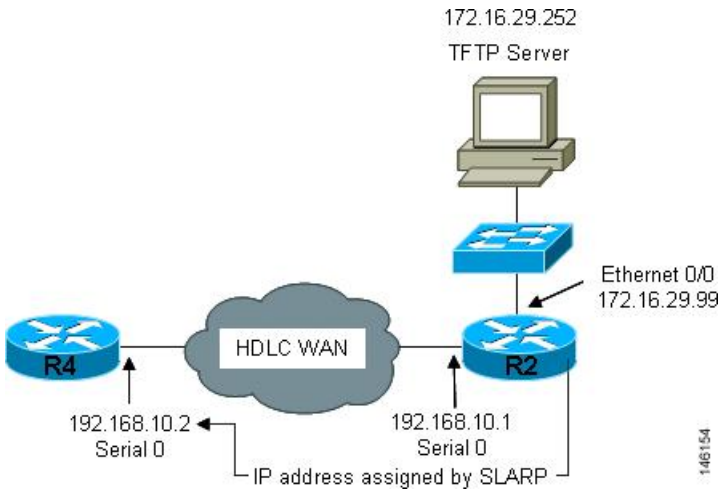
Tip ステージング ルータのシリアル インターフェイス上でマスク 255.255.255.252 を使用している場合、SLARP は使用可能な IP ホスト アドレスを新しいデバイスに割り当てます。たとえば、IP アドレス 198.162.10.5 255.255.255.252 をステージング ルータの serial 0 に割り当てる場合、SLARP は 198.162.10.6 を新しいデバイスに割り当てます。IP アドレス 198.162.10.6 255.255.255.252 をステージング ルータの serial 0 に割り当てる場合、SLARP は 198.162.10.5 を新しいデバイスに割り当てます。

次の図に、SLARP の例を示します。

次の図で、ステージング ルータ（R2）のシリアル インターフェイス 0 の IP アドレスは 192.168.10.1 です。そのため、SLARP は IP アドレス 192.168.10.2 を新しいルータのシリアル インターフェイス 0 に割り当てます。

Note このトポロジを Cisco ASR 1000 シリーズ アグリゲーション サービス ルータで使用することを計画している場合は、この図で使用されているイーサネット インターフェイスをギガビット イーサネット インターフェイスに置き換えます。

Figure 2: SLARP を使用した新しいデバイスへの *IP* アドレスの割り当て



Note HDLC を使用したシリアル インターフェイス上の自動インストールは、新しいデバイスの最初のシリアル ポート（シリアル インターフェイス 0 またはシリアル インターフェイス x/0）上だけで実行できます。ステージング ルータと新しいデバイスは、serial 0/0 や serial 2/0（シリアル ポートがデバイスの第 2 スロットにある場合）など、新しいデバイス上の最初のシリアル インターフェイス ポートを使用して直接接続されている必要があります。

Tip ステージング ルータから SLARP により自動インストールを使用して設定するルータに割り当てられる IP アドレスは、自動インストールの network-confg ファイルまたは cisconet.cfg ファイルの **ip host** *hostname ip-address* コマンドで使用する必要があります。これは、自動インストールを使用して設定するルータに正しいホスト名が割り当てられ、ホスト固有のコンフィギュレーション ファイルを要求できるようにするためです。

BOOTP サーバー

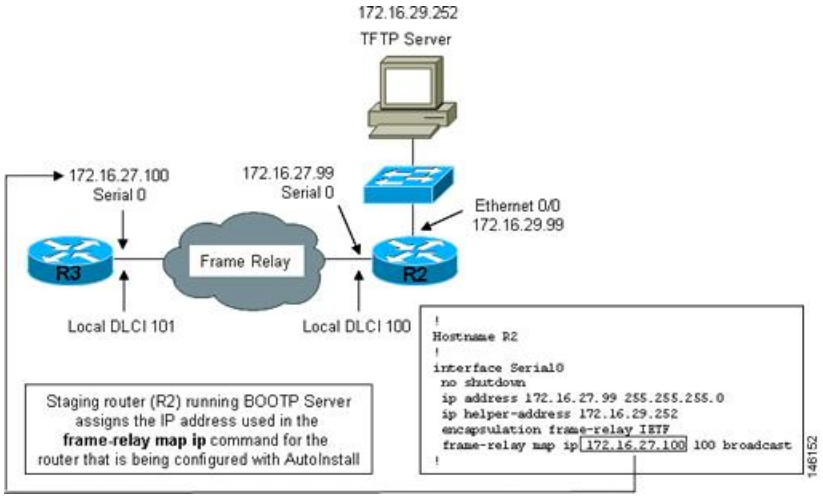
シリアル インターフェイス経由でフレーム リレー カプセル化を使用して自動インストールで設定するルータは、ステージング ルータに接続されているシリアル インターフェイス上で IP アドレスの BOOTP 要求を送信します。

ステージング ルータは、自動インストールで設定するルータに対する BOOTP 応答で提供する正しい IP アドレスを、自動インストールで設定するルータに接続するために使用しているインターフェイス上で設定されている **frame-relay map ip** *ip-address dlci* コマンドを調べることで取得します。

下の図で、R2 はステージング ルータです。R2 では、インターフェイス serial 0 上で **frame-relay map ip 172.16.27.100 100** ブロードキャスト コマンドが設定されています。R2 が自動インストール プロセス中に R3 から IP アドレスの BOOTP 要求を受信すると、R3 は 172.16.27.100 で応答します。

Note このトポロジを Cisco ASR 1000 シリーズ アグリゲーション サービス ルータで使用することを計画している場合は、この図で使用されているイーサネット インターフェイスをギガビット イーサネット インターフェイスに置き換えます。

Figure 3: フレーム リレー ネットワークを介した自動インストールで ***BOOTP*** を使用する例



Tip 新しいデバイスとステージング ルータの IP アドレスが .1 または .2 で終わっていなければならないという SLARP での制限は、BOOTP には適用されません。フレーム リレー上の自動インストールのための BOOTP は、自動インストールで設定するルータとステージング ルータの間のフレーム リレー回線に割り当てられた、IP アドレス サブネットに対するすべてのホスト アドレスをサポートします。

Tip ステージング ルータから BOOTP により自動インストールを使用して設定するルータに割り当てられる IP アドレスは、自動インストールの network-confg ファイルまたは cisconet.cfg ファイルの **ip host** *hostname ip-address* コマンドで使用する必要があります。これは、自動インストールを使用して設定するルータに正しいホスト名が割り当てられ、ホスト固有のコンフィギュレーション ファイルを要求できるようにするためです。

Note フレーム リレー カプセル化を使用したシリアル インターフェイス上の自動インストールは、新しいデバイスの最初のシリアル ポート（シリアル インターフェイス 0 またはシリアル インターフェイス x/0）上だけで実行できます。ステージング ルータと新しいデバイスは、serial 0/0 や serial 2/0（シリアル ポートがデバイスの第 2 スロットにある場合）など、新しいデバイス上の最初のシリアル インターフェイス ポートを使用して直接接続されている必要があります。

自動インストールのIPとホスト名のマッピングで使用されるサービスとサーバー

自動インストール プロセス中にネットワーキング デバイスに完全なコンフィギュレーション ファイルをロードするには、そのネットワーキング デバイス用に作成したコンフィギュレーション ファイルを要求できるように、ネットワーキング デバイスがそのホスト名を決定できる必要があります。

自動インストール用に IP アドレスからホスト名へのマッピングをプロビジョニングするためには、次の点に注意してください。

- 自動インストールで設定するネットワーキング デバイスは、そのいずれかの自動インストール ネットワーク コンフィギュレーション ファイル（network-confg または cisconet.cfg）を TFTP サーバーからロードすることで、そのホスト名を決定できます。このファイルには、**iphost***hostnameip-address* コマンドが含まれています。たとえば、ホスト R3 を IP アドレス 198.162.100.3 にマッピングするには、network-confg ファイルまたは cisconet.cfg ファイルに **iphostr3198.162.100.3** コマンドが含まれている必要があります。
- LAN インターフェイス上で自動インストールを使用して設定するネットワーキング デバイスは、DNS サーバーに問い合わせることでもそのホスト名を決定できます。DNS サーバーが同じ LAN に接続されていない場合、デバイスは、DHCP サーバーからダイナミックに割り当てられた IP アドレスを取得するプロセスの中で、DNS サーバーの IP アドレスを DHCP サーバーから取得する必要があります。

DNS サーバー

DNS サーバーは、ホスト名を IP アドレスに、IP アドレスをホスト名に（逆 DNS ルックアップ）マッピングするネットワーク サービスを提供するために使用します。PC がホスト名を使用してホストへの IP 接続を開始するときには、必ず接続先のホスト名に割り当てられている IP アドレスを特定する必要があります。たとえば、シスコの Web サイト（http://www.cisco.com/）を参照すると、PC は DNS サーバーに DNS クエリーを送信して、シスコの Web サイトに接続するために使用可能な現在の IP アドレスを知ります。

DNS サービスの詳細については、IETF RFC のサイト（http://www.ietf.org/rfc.html）で DNS に関する RFC を参照してください。ネーム サーバー ルックアップ ツール（nslookup）は、DNS の詳細を知るのに非常に便利です。検索すると、nslookup に関する優れた Web サイトがいくつも見つかります。

自動インストールのコンフィギュレーションファイルの格納と転送で使用されるサービスとサーバー

TFTP は、ネットワーク上のデバイス間でファイルを転送するために使用するプロトコルです。TFTP サーバは、TFTP を使用してデバイスにファイルを転送するデバイスです。TFTP サーバーは、UNIX サーバー、Microsoft Windows ベースの PC およびサーバー、その他のプラットフォーム上で設定できます。

Tip 使用可能な TFTP サーバーがない場合は、**tftp-serverfile-system**:*filename* コマンドを使用して、Cisco IOS ベースのルータを TFTP サーバーとして設定します。ルータを TFTP サーバーとして設定する方法の詳細については、『Configuring Basic File Transfer Services』を参照してください。

シスコのルータは、TFTP を使用して、自動インストールで必要なコンフィギュレーション ファイルをロードします。ファイルの格納と、自動インストールを使用するデバイスへのファイル転送のために、ネットワークに TFTP サーバーを配置する必要があります。

TFTP サービスの詳細については、IETF RFC のサイト（http://www.ietf.org/rfc.html）で TFTP に関する RFC を参照してください。検索すると、TFTP に関する優れた Web サイトがいくつも見つかります。インターネットでは、さまざまなオペレーティング システムおよびハードウェア プラットフォーム向けのフリーウェアとシェアウェア版の TFTP サーバーがいくつも利用できます。

自動インストール向けに TFTP サーバーをプロビジョニングする際には、次の点に注意してください。

- LAN 経由で自動インストールを使用するデバイス：TFTP サーバーと自動インストールを使用するデバイスが別々の LAN セグメント上にある場合、自動インストールを使用するデバイスからの TFTP セッション初期化要求を受信するすべてのインターフェイス上で、**iphelper-address** *address* コマンドを設定する必要があります。
- WAN 経由で自動インストールを使用するデバイス：自動インストールを使用するデバイスが WAN に接続されている場合、自動インストールを使用するデバイスからの TFTP セッション初期化要求を受信するすべてのインターフェイス上で、**iphelper-address** *address* コマンドを設定する必要があります。

ip helper-address

新しいデバイスが、TFTP サーバーの IP アドレスを、DHCP オプション 150 経由で取得しない場合、TFTP セッション初期化要求を、IP 宛先ブロードキャスト アドレス 255.255.255.255 を使用したネットワーク層ブロードキャストとして送信します。ルータはネットワーク層ブロードキャスト データグラムをブロックするため、TFTP セッション開始要求が TFTP サーバーに到達せず、自動インストールは失敗します。この問題を解決するには、**ip helper-address** *address* コマンドを使用します。**ip helper-address** *address* コマンドは、TFTP セッション開始要求のブロードキャスト アドレスを、255.255.255.255 から、*address* 引数で設定されるアドレスに変更します。たとえば、**ip helper-address 172.16.29.252** コマンドは、IP 宛先ブロードキャスト アドレス 255.255.255.255 を 172.16.29.252 に変更します。

自動インストールで使用されるネットワーキング デバイス

自動インストールで設定するデバイス

自動インストールで設定するデバイスは、自動インストールをサポートし、NVRAM にコンフィギュレーション ファイルがない、任意の Cisco IOS XE ベースのルータです。

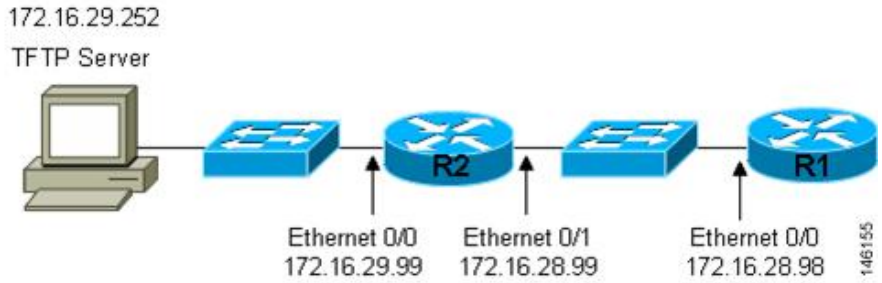
ステージング ルータ

ステージング ルータは、新しいデバイスと TFTP サーバーが異なるネットワークに接続されている場合に、TFTP サーバー（IP 接続可能であることが必要です）と、自動インストールで設定されるデバイスの間の仲介役として振る舞います。次の図で、R1 にはステージング ルータが必要です。これは、R1 が TFTP サーバーと異なる LAN セグメントに接続されているためです。

ステージング ルータは、次の状況で必要です。

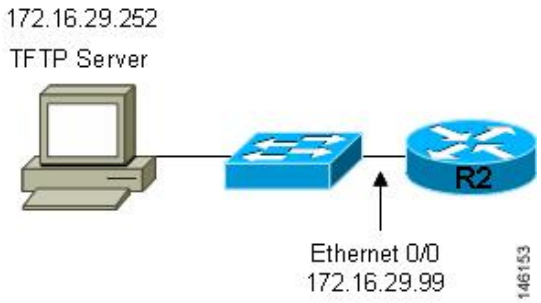
- LAN 経由で自動インストールを使用するデバイス：TFTP サーバーと DHCP サーバーのいずれかまたは両方と、自動インストールを使用するデバイスが異なる LAN セグメントにある場合は、ステージング ルータを使用する必要があります。
- WAN 経由で自動インストールを使用するデバイス：自動インストールを使用するデバイスが WAN に接続されている場合、自動インストールを使用するデバイスからの TFTP セッション初期化要求を受信するすべての直接接続インターフェイス上で、**ip helper-address** *address* コマンドを設定する必要があります。

Figure 4: **ステージング ルータが必要な自動インストールの例**



自動インストールで設定する新しいデバイスが、TFTP サーバーおよび DHCP サーバーと同じ LAN セグメントに接続されている場合には、ステージング ルータは不要です。次の図で、R2 は、TFTP サーバーと同じ LAN セグメント上にあるため、自動インストールを使用するためにステージング サーバーは必要ありません。

Figure 5: ステージング ルータが不要な自動インストールの例



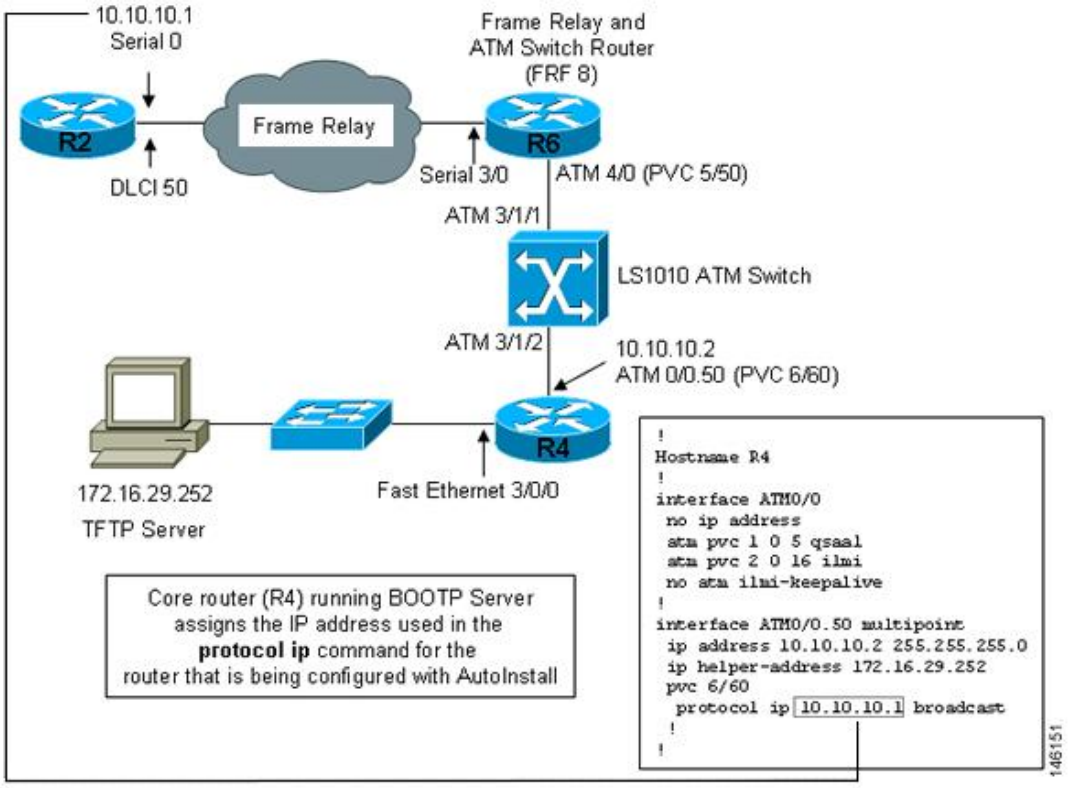
フレーム リレー/ATM 間スイッチング デバイス

フレームリレー/ATM間スイッチングデバイスは、ルーティングとスイッチング動作の両方を実行できるデバイスです。フレームリレー/ATM間スイッチングデバイスは、フレームリレーネットワークとATMネットワークを接続するために使用します。

フレーム リレー/ATM 間インターワーキング接続上の自動インストール機能は、自動インストールプロセスを、シスコが定義したフレーム リレー カプセル化ではなく、IETF標準で定義されたフレーム リレー カプセル化を使用するように、自動インストール プロセスを変えたものです。

次の図は、フレーム リレー/ATM 間インターワーキング接続上の自動インストール機能を使用するトポロジ例を示します。ルータ R6 は、フレーム リレー DLCI 50 から ATM VPI/VCI 5/50 への、フレームリレー/ATM間サービスインターワーキング（FRF8）変換を行います。LS1010 スイッチは、R6（5/50）が使用する VPI と VCI の組み合わせを、R4（6/60）が使用する VPI と VCI の組み合わせにルーティングします。

Figure 6: フレーム リレー/*ATM* 間インターワーキング接続上の自動インストールのトポロジ例



自動インストールの設定オプション

デバイスとサービスのいくつかの異なる組み合わせを使用して、自動インストールをサポートするようにネットワークをプロビジョニングできます。次に例を示します。

- 自動インストールで必要なすべてのサービス（シスコのルータで実行する必要がある、SLARP または BOOTP を使用したダイナミックな IP アドレスの割り当てを除く）を、1 台のネットワーク サーバー上にプロビジョニングすることも、各サービスを異なるネットワーク サーバーにプロビジョニングすることもできます。
- DHCP サービスは、シスコのルータ上にプロビジョニングできます。
- 自動インストールを使用するデバイスの IP アドレスを DNS サーバーから特定するか、**ip host** *hostname ip-address* コマンドを含むいずれかの自動インストール ネットワーク コンフィギュレーション ファイル（network-confg または cisconet.cfg）を使用できます。
- 自動インストールを使用するデバイスに、完全なコンフィギュレーションをロードするか部分的なコンフィギュレーションをロードするように自動インストールをプロビジョニングできます。

このモジュールでは、主に自動インストールをプロビジョニングするための最も一般的な方法のいくつかを扱います。自動インストールをプロビジョニングする最も一般的な方法について

は、「自動インストールを使用してシスコ ネットワーキング デバイスをリモートで設定する方法」のモジュールを参照してください。

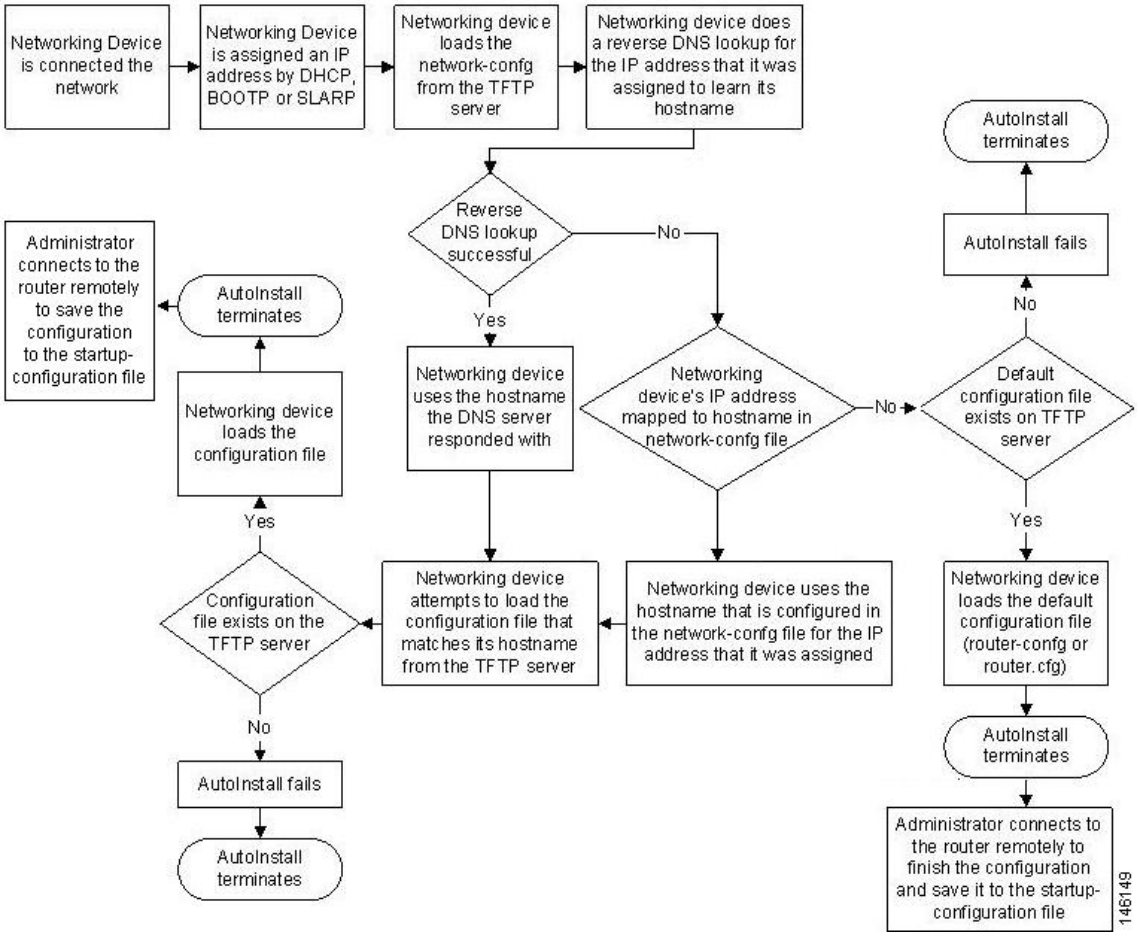
自動インストール プロセス

自動インストール プロセスは、NVRAM にファイルが何もないネットワーキング デバイスをネットワークに接続したときに開始されます。

Timesaver 自動インストールプロセスが終了するまでは、ネットワーキングデバイス上の自動インストールで使用するインターフェイスだけを接続することで、自動インストールが完了するまでに要する時間を短縮できます。たとえば、WAN インターフェイス経由でネットワーキング デバイスに対する自動インストールを実行する場合、その LAN インターフェイスと WAN インターフェイスを接続すると、ネットワーキング デバイスは、WAN インターフェイスの使用を試みる前に、LAN インターフェイス上で自動インストールの実行を試みます。自動インストール プロセスが完了するまで LAN インターフェイスを接続しないでおくことで、ネットワーキング デバイスはすぐに WAN インターフェイス上で自動インストール プロセスを開始します。

次の図は、コンフィギュレーション ファイルを使用する自動インストール プロセスの基本フローを示します。

Figure 7: 自動インストール プロセスのフローチャート（コンフィギュレーション ファイル使用）



自動インストールを使用してシスコ ネットワーキング デバイスをリモートで設定する方法

ここでは、自動インストールのためにルータを準備する方法について説明します。LAN、HDLC WAN、およびフレーム リレーのネットワークに接続された新しいルータのために自動インストールを使用する追加の例は、「自動インストールを使用してシスコのネットワーキング デバイスをリモートで設定する例」のモジュールを参照してください。

ほとんどの場合、自動インストールを実行する新規デバイスが TFTP、BOOTP、および DNS 要求を送信するときに経由するステージング ルータを設定する必要があります。

Tip いずれの場合にも、自動インストール プロセスが完了した後、ネットワーキング デバイス上でコンフィギュレーションを確認し保存する必要があります。コンフィギュレーションを保存しない場合、プロセス全体を繰り返す必要があります。

SDM デフォルト コンフィギュレーション ファイルの無効化

使用しているデバイスに SDM がプレインストールされているときに、セットアップを使用して、初期設定ファイルを作成する場合は、次の作業を実行します。SDM はデバイスに残ります。

使用しているデバイスにSDMがプレインストールされているときに、代わりに自動インストールを使用して、デバイスを設定する場合は、次の作業を実行します。SDM はデバイスに残ります。

SUMMARY STEPS

1. デバイスに付属しているコンソール ケーブルを、デバイスのコンソール ポートから PC のシリアル ケーブルに接続します。手順については、使用しているデバイスのハードウェア インストール ガイドを参照してください。
2. 電源モジュールをデバイスに接続し、この電源モジュールをコンセントに差し込んで、デバイスの電源をオンにします。手順については、使用しているデバイスのクイック スタート ガイドを参照してください。
3. 使用している PC の Hyperterminal またはこれに準じた端末エミュレーション プログラムで、次のように端末エミュレーション設定を行い、デバイスに接続します。
4. **enable**
5. **erase startup-config**
6. **reload**

DETAILED STEPS

ステップ 1 デバイスに付属しているコンソール ケーブルを、デバイスのコンソール ポートから PC のシリアル ケーブルに接続します。手順については、使用しているデバイスのハードウェア インストール ガイドを参照してください。

ステップ 2 電源モジュールをデバイスに接続し、この電源モジュールをコンセントに差し込んで、デバイスの電源をオンにします。手順については、使用しているデバイスのクイック スタート ガイドを参照してください。

ステップ 3 使用している PC の Hyperterminal またはこれに準じた端末エミュレーション プログラムで、次のように端末エミュレーション設定を行い、デバイスに接続します。

- 9600 ボー
- 8 データ ビット、パリティなし、1 ストップ ビット
- フロー制御なし

ステップ 4 **enable**

特権 EXEC モードを開始します。

enable

Example:

```
Router> enable
Router#
```

ステップ 5 **erase startup-config**

NVRAM から既存のコンフィギュレーションを消去します。

Example:

```
Router# erase startup-config
```

ステップ 6 **reload**

リロード プロセスを開始します。ルータはリロード プロセスの終了後、自動インストール プロセスを開始します。

Example:

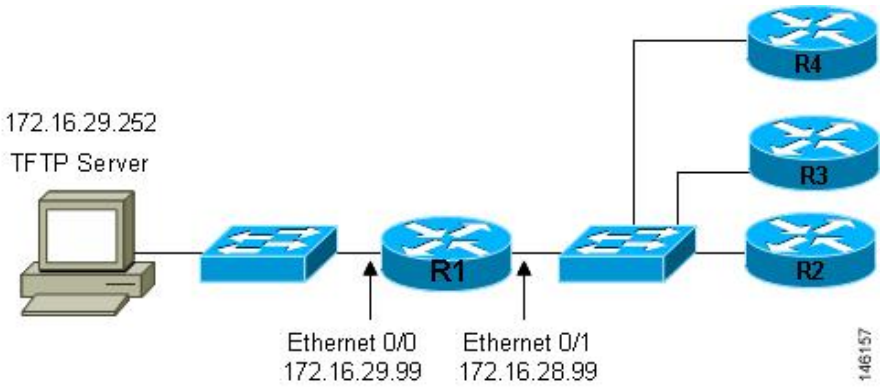
```
Router# reload
```

自動インストールを使用してシスコのネットワーキング デバイスをリモートで設定する例

自動インストールを使用したLANに接続されているデバイス設定の例

このタスクでは、次に示す図のネットワークを使用します。このタスクでは、自動インストールを使用してルータ R2、R3、および R4 を設定する方法を示します。ルータ R1 は、自動インストール プロセス中に新しいルータのファスト イーサネット 0/0 に IP アドレスを割り当てるために使用される DHCP サーバーです。

Figure 8: 特定のデバイスに対する自動インストール コンフィギュレーション ファイルを割り当てるためのネットワーク トポロジ



すべての DHCP クライアントには、固有の DHCP クライアント ID があります。DHCP クライアント ID は、DHCP サーバーによって、IP アドレスのリースを追跡し、IP アドレスの予約を設定するために使用されます。DHCP IP アドレス予約を設定するためには、自動インストールを使用して設定する各ネットワーキング デバイスの DHCP クライアント ID を知る必要があります。これにより、各デバイスに正しい IP アドレスが提供され、その後固有のコンフィギュレーション ファイルが提供されます。DHCP クライアント ID は手動または自動で特定できます。

自動インストールを使用してルータ R2、R3、および R4 を設定するには、次の作業を実行します。

手動での DHCP クライアント ID の値の特定

クライアント ID の値を自動的に特定する場合は、この作業を実行する必要はありません。「自動的な DHCP クライアント ID の特定」のモジュールに進みます。

クライアント ID を手動で特定するためには、自動インストール プロセス中にルータを LAN に接続するために使用されるファストイーサネットインターフェイスの MAC アドレスを知っておく必要があります。これには、**show interface** *interface-type interface-number* コマンドを入力できるように、端末をルータに接続し、電源をオンにする必要があります。

クライアント ID は次のように表示されます。

0063.6973.636f.2d30.3030.362e.3533.6237.2e38.6537.312d.4661.332f.30

形式は *nullcisco-0006.53b7.8e71-fa3/0* です。*0006.53b7.8e71* は MAC アドレスであり、*fa3/0* は IP アドレスを要求するインターフェイスの短いインターフェイス名です。

short-if-name フィールドの値は、Cisco MIB がインストールされた SNMP ワークステーションから取得できます。次に、ifIndex を Cisco IOS 上のインターフェイスにマッピングする例を示します。

```
snmpwalk -c public ponch ifName
IF-MIB::ifName.1 = STRING: AT2/0
IF-MIB::ifName.2 = STRING: Et0/0
IF-MIB::ifName.3 = STRING: Se0/0
IF-MIB::ifName.4 = STRING: BR0/0
```

show interface *interface-type interface-number* コマンドを使用して、ファスト イーサネット インターフェイスの情報と統計情報を表示します。

```
R6> show interface fastethernet 3/0
FastEthernet3/0 is up, line protocol is up
  Hardware is AmdFE, address is 0006.53b7.8e71 (bia 0006.53b7.8e71)
.
.
.
R6>
```

R6 上のファスト イーサネット 3/0 の MAC アドレスは 0006.53b7.8e71 です。このインターフェイスのクライアント ID の形式は nullcisco-0006.53b7.8e71-fa3/0 です。

Note ファスト イーサネット インターフェイスの短いインターフェイス名は fa です。

次の表に、文字を 16 進数の文字に変換するための値を示します。2 つ目の表の最後の行は、R6 上のファスト イーサネット 3/0 のクライアント ID（nullcisco-0006.53b7.8e71-fa3/0）を示します。

Table 9: 16 進数から文字への変換表

16 進数	文字	16 進数	文字	16 進数	文字	16 進数	文字	16 進数	文字
00	NUL	1a	SUB	34	4	4e	N	68	h
01	SOH	1b	ESC	35	5	4f	O	69	I
02	STX	1c	FS	36	6	50	P	6a	j
03	ETX	1d	GS	37	7	51	Q	6b	k
04	EOT	1e	RS	38	8	52	R	6c	l
05	ENQ	1f	US	39	9	53	S	6d	m
06	ACK	20		3a	:	54	T	6e	n
07	BEL	21	!	3b	;	55	U	6f	o
08	BS	22	"	3c	<	56	V	70	p
09	TAB	23	#	3d	=	57	W	71	q
0A	LF	24	$	3e	>	58	X	72	r
0B	VT	25	%	3f	?	59	Y	73	s
0C	FF	26	&	40	@	5a	Z	74	t
0D	CR	27	‘	41	A	5b	[	75	u
0E	SO	28	(	42	B	5c	\	76	v
0F	SI	29	)	43	C	5d	]	77	w
10	DLE	2a	*	44	D	5e	^	78	x
11	DC1	2b	+	45	E	5f	_	79	y
12	DC2	2c	,	46	F	60	`	7a	z
13	DC3	2d	-	47	G	61	a	7b	{
14	DC4	2e	.	48	H	62	b	7c	|

16 進数	文字	16 進数	文字	16 進数	文字	16 進数	文字	16 進数	文字
15	NAK	2f	/	49	I	63	c	7D	}
16	SYN	30	0	4a	J	64	d	7e	~
17	ETB	31	1	4b	K	65	e	7f	D
18	CAN	32	2	4c	L	66	f		
19	EM	33	3	4d	M	67	g		

***Table 10: nullcisco-0006.53b7.8e71-fa3/0* からクライアント *ID* への変換**

00	c	i	s	c	o	-	0	0	0	6	.	5	3	b	7	.	8	e	7	1	-	f	a	3	/	0
00	63	69	73	63	6f	2d	30	30	30	36	2e	35	33	62	37	2e	38	65	37	31	2d	46	61	33	2f	30

R4

show interface *interface-type interface-number* コマンドを使用して、R4 上のファストイーサネット 0/0 の情報と統計情報を表示します。

```
R4> show interface FastEthernet 0/0
FastEthernet0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb0e (bia 00e0.1eb8.eb0e)
```

R4 のファスト イーサネット 0/0 の MAC アドレスは 00e0.1eb8.eb0e です。このインターフェイスのクライアント ID の形式は nullcisco-00e0.1eb8.eb0e-et0 です。

Note ファスト イーサネット インターフェイスの短いインターフェイス名は et です。

上記の 1 つ目の表の 16 進数の文字に変換するための値を使用して、R4 上のファスト イーサネット 0/0 のクライアント ID を次の表の最後の行に示します。

***Table 11: null.cisco-00e0.1eb8.eb0e-et0* から *R4* のクライアント *ID* への変換**

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	0	e	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	30	65	2d	45	74	30

R3

show interface *interface-type interface-number* コマンドを使用して、R3 上のファストイーサネット 0/0 の情報と統計情報を表示します。

```
R3> show interface FastEthernet 0/0
```

```
FastEthernet0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb73 (bia 00e0.1eb8.eb73)
```

R3 のファスト イーサネット 0/0 の MAC アドレスは 00e0.1eb8.eb73 です。このインターフェイスのクライアント ID の形式は nullcisco-00e0.1eb8.eb73-et0 です。

上記の 1 つ目の表の 16 進数の文字に変換するための値を使用して、R3 上のファスト イーサネット 0/0 のクライアント ID を次の表の最後の行に示します。

***Table 12: null.cisco-00e0.1eb8.eb73-et0* から *R3* のクライアント *ID* への変換**

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	7	3	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	37	33	2d	45	74	30

R2

show interface *interface-type interface-number* コマンドを使用して、R2 上のファスト イーサネット 0/0 の情報と統計情報を表示します。

```
R2> show interface Fast Ethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is Lance, address is 00e0.1eb8.eb09 (bia 00e0.1eb8.eb09)
```

R2 のファスト イーサネット 0/0 の MAC アドレスは 00e0.1eb8.eb09 です。このインターフェイスのクライアント ID の形式は nullcisco-00e0.1eb8.eb09-et0 です。

上記の 1 つ目の表の 16 進数の文字に変換するための値を使用して、R2 上のファスト イーサネット 0/0 のクライアント ID を次の表の最後の行に示します。

***Table 13: null.cisco-00e0.1eb8.eb09-et0* から *R2* のクライアント *ID* への変換**

00	c	i	s	c	o	-	0	0	e	0	.	1	e	b	8	.	e	b	0	9	-	e	t	0
00	63	69	73	63	6f	2d	30	30	65	30	2e	31	65	62	38	2e	65	62	30	39	2d	45	74	30

これで各ルータのクライアント ID の値が特定できました。最後の手順は、次に示すように、左から右に 4 文字ずつのグループにし、その後にピリオドを追加することです。

- R4-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30
- R3-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
- R2-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30

DHCP クライアント ID の値の自動特定

クライアント ID の値を手動で特定する場合は、この作業を実行する必要はありません。「各ルータ用のプライベート DHCP プールの作成」のモジュールに進みます。

この作業では、R1 上に、1 つの IP アドレスだけを提供する DHCP サーバーを構築します。この IP アドレスは、ルータのクライアント ID の値を特定する間、新しい各ルータによって順番に使用されます。IP アドレスの範囲を単一の IP アドレスに制限することで、どのルータを操

作しているかに関する混乱を避けることができます。誰かが別のルータの電源をオンにし、自動インストール プロセスが開始されると、そのルータは IP アドレスを取得できません。

Tip network-confg またはルータ コンフィギュレーション ファイル（r4-confg、r3-confg、または r2-confg）は、まだ TFTP サーバーのルート ディレクトリに格納しないでください。ルータが正しいコンフィギュレーション ファイルをロードするように、各ルータが DHCP サーバーから正しい IP アドレスを取得することを確認するまでは、これらのファイルをルータがロードしないようにします。

このタスクは、分かりやすくするためにサブタスクに分かれています（すべてのサブタスクが必要）。

R1 上のインターフェイスの IP の設定

ファストイーサネット インターフェイスで IP アドレスを設定します。ファストイーサネット 0/1 上で **ip helper-address** *ip-address* コマンドを設定します。

```
!
interface FastEthernet0/0
 ip address 172.16.29.99 255.255.255.0
!
interface FastEthernet0/1
 ip address 172.16.28.99 255.255.255.0
 ip helper-address 172.16.29.252
!
```

R1 上の DHCP プールの設定

R1 上で一時的な DHCP サーバーをセットアップするには、次のコマンドを設定します。

Note これは、R1 で稼働する唯一の DHCP サーバーである必要があります。これは、自動インストールを使用して設定するルータがアクセスできる唯一の DHCP サーバーであることが必要です。

```
ip dhcp excluded-address vrf Mgmt-intf 172.16.28.1 172.16.28.10
ip dhcp pool DHCP_Pool
vrf Mgmt-intf
network 172.16.28.0 255.255.255.0
bootfile ASR-Bootup.cfg
option 150 ip 1.1.1.1
default-router 172.16.28.1
```

R1 上の DHCP プールからの 1 つを除くすべての IP アドレスの除外

DHCP サーバーからは常に 1 つの IP アドレスだけが利用できるようにする必要があります。DHCP プールから、172.16.28.1 以外のすべての IP アドレスを除外するには、次のコマンドを設定します。

```
!
```

```
ip dhcp excluded-address 172.16.28.2 172.16.28.255
!
```

R1 の設定の確認

R1 用のコンフィギュレーション ファイルに、1 つの IP アドレス（172.16.28.1）を DHCP クライアントに提供する、DHCP サーバー プールが設定されていることを確認します。

コンフィギュレーション ファイルに、ファスト イーサネット インターフェイスの IP アドレスと **ip helper-address** *ip-address* コマンドが含まれていることを確認します。

```
!
ip dhcp excluded-address 172.16.28.2 172.16.28.255
!
ip dhcp pool get-client-id
   network 172.16.28.0 255.255.255.0
!
interface FastEthernet0/0
 ip address 172.16.29.99 255.255.255.0
!
interface FastEthernet0/1
 ip address 172.16.28.99 255.255.255.0
 ip helper-address 172.16.29.252
!
```

R1 上での debug ip dhcp server events の有効化

R1 に接続された端末上で **debug ip dhcp server events** コマンドからの出力を使用し、各ルータのクライアント ID を特定します。

R1 上で **debug ip dhcp server events** コマンドを有効にします。

```
R1# debug ip dhcp server events
```

各ルータでのクライアント ID の値の特定

この手順は、各ルータで繰り返します。一度に 1 台のルータの電源だけをオンにする必要があります。ルータのクライアント ID フィールドの値を特定したら、そのルータの電源をオフにし、次のルータに進みます。

R4

R4 をファスト イーサネット ネットワークに接続し、電源をオンにします。R4 に IP アドレス 172.16.28.1 が割り当てられると、R1 に接続された端末に次のメッセージが表示されます。

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30.
```

クライアント ID 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30 をテキスト ファイルにコピーして保存します。テキスト ファイルは、次の 2 台のルータ用に開いたままにします。

R4 の電源をオフにします。

R1 上で **clear ip dhcp binding *** コマンドを使用し、R1 上の DHCP プールから R4 の IP アドレス バインディングを解放します。

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

R3

R3 をファスト イーサネット ネットワークに接続し、電源をオンにします。R3 に IP アドレス 172.16.28.1 が割り当てられると、R1 に接続された端末に次のメッセージが表示されます。

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30.
```

クライアント ID 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30 をテキスト ファイルにコピーして保存します。テキスト ファイルは、最後のルータ用に開いたままにします。

R3 の電源をオフにします。

R1 上で **clear ip dhcp binding *** コマンドを使用し、R1 上の DHCP プールから R3 の IP アドレス バインディングを解放します。

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

R2

R2 をファスト イーサネット ネットワークに接続し、電源をオンにします。R2 に IP アドレス 172.16.28.1 が割り当てられると、R1 に接続された端末に次のメッセージが表示されます。

```
DHCPD: assigned IP address 172.16.28.1 to client
0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30.
```

クライアント ID 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30 をテキスト ファイルにコピーして保存します。

R2 の電源をオフにします。

R1 上で **clear ip dhcp binding *** コマンドを使用し、R1 上の DHCP プールから R2 の IP アドレス バインディングを解放します。

```
R1# clear ip dhcp binding *
R1#
01:16:11: DHCPD: returned 172.16.28.1 to address pool get-client-id.
```

R4、R3、および R2 のクライアント ID

これで各ルータのクライアント ID の値が特定できました。

- R4-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30

- R3-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
- R2-0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30

ネットワーク 172.16.28.0 24 用の R1 上の DHCP プールの削除

ルータの一時的な DHCP プールは必要なくなり、削除する必要があります。

```
R1(config)# no ip dhcp pool get-client-id
```

R1 からの除外されたアドレス範囲の削除

172.16.28.1 以外のすべての IP アドレスをルータ上の DHCP プールから除外するコマンドは必要なくなり、削除する必要があります。

R1(config)# **no ip dhcp excluded-address 172.16.28.2 172.16.28.255**

各ルータ用のプライベート DHCP プールの作成

すべてのルータにネットワーク コンフィギュレーション ファイルでホスト名にマッピングされた IP アドレスが割り当てられるようにするために、各ルータ用のプライベート DHCP アドレス プールを作成する必要があります。

```
!
ip dhcp pool r4
   host 172.16.28.100 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.652d.4574.30
!
ip dhcp pool r3
   host 172.16.28.101 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6237.332d.4574.30
!
ip dhcp pool r2
   host 172.16.28.102 255.255.255.0
   client-identifier 0063.6973.636f.2d30.3065.302e.3165.6238.2e65.6230.392d.4574.30
```

各ルータ用のコンフィギュレーション ファイルの作成

各ルータ用のコンフィギュレーション ファイルを作成し、TFTP サーバーのルート ディレクトリに置きます。

Tip ルータにリモートからアクセスしてそのコンフィギュレーション ファイルを NVRAM に保存する場合は、リモート Telnet アクセスと特権 EXEC モードへのアクセス用のパスワードを設定するためのコマンドを含める必要があります。

r2-confg

```
!
hostname R2
!
enable secret 7gD2A0
```

```
!
interface FastEthernet0/0
 ip address 172.16.28.102 255.255.255.0
!
interface Serial0/0
 ip address 192.168.100.1 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.5 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0/0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

r3-confg

```
!
hostname R3
!
enable secret 7gD2A0
!
interface FastEthernet0/0
 ip address 172.16.28.101 255.255.255.0
!
interface Serial0/0
 ip address 192.168.100.9 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.13 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

r4-confg

```
!
hostname R3
!
enable secret 7gD2A0
!
interface FastEthernet0/0
 ip address 172.16.28.101 255.255.255.0
```

```
!
interface Serial0/0
 ip address 192.168.100.9 255.255.255.252
 no shutdown
!
interface Serial0/1
 ip address 192.168.100.13 255.255.255.252
 no shutdown
!
no ip http server
ip classless
ip default-network 0.0.0.0
ip route 0.0.0.0 0.0.0.0 FastEthernet0/0
!
line vty 0 4
 password 5Rf1k9
 login
!
end
```

ネットワーク コンフィギュレーション ファイルの作成

DHCP サーバーに割り当てる IP アドレスをホスト名にマップする **ip host** *hostname ip-address* コマンドでネットワーク コンフィギュレーション ファイルを作成します。

```
ip host r4 172.16.28.100
ip host r3 172.16.28.101
ip host r2 172.16.28.102
```

自動インストールによるルータのセットアップ

自動インストールを使用して、3 台のルータ（R4、R3、および R2）をセットアップする準備ができました。

自動インストールの進行状況を監視するには、ルータに端末を接続します。使用している PC の Hyperterminal またはこれに準じた端末エミュレーション プログラムで、次のように端末エミュレーション設定を行い、デバイスに接続します。

- 9600 ボー
- 8 データ ビット、パリティなし、1 ストップ ビット
- フロー制御なし

TFTP サーバーのルート ディレクトリに次のファイルを格納しておきます。

- network-confg
- r4-confg
- r3-confg
- r2-confg

TFTP サーバーが動作している必要があります。

各ルータの電源をオンにします。

Timesaver 3 台のルータを同時に設定できます。

R4

次に示すのは、自動インストール プロセス中に R4 のコンソール端末に表示されるメッセージの一部です。

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.100 to r4
Loading r4-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

R3

次に示すのは、自動インストール プロセス中に R3 のコンソール端末に表示されるメッセージの一部です。

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.101 to r3
Loading r3-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

R2

次に示すのは、自動インストール プロセス中に R2 のコンソール端末に表示されるメッセージの一部です。

```
Loading network-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 76 bytes]
Configuration mapped ip address 172.16.28.102 to r2
Loading r2-confg from 172.16.29.252 (via FastEthernet0/0): !
[OK - 687 bytes]
```

TFTP サーバー ログ

TFTP サーバー ログには、次のようなメッセージが出力されます。

```
Sent network-confg to (172.16.28.100), 76 bytes
Sent r4-confg to (172.16.28.100),687 bytes
Sent network-confg to (172.16.28.101), 76 bytes
Sent r3-confg to (172.16.28.101),687 bytes
Sent network-confg to (172.16.28.102), 76 bytes
Sent r2-confg to (172.16.28.102),687 bytes
```

ルータ上でのコンフィギュレーション ファイルの保存

各ルータに電源が再投入された場合にもそれぞれの設定を保持できるようにするために、各ルータで実行中の設定を保存してから設定を開始する必要があります。

R4

```
R1# telnet 172.16.28.100
Trying 172.16.28.100 ... Open
User Access Verification
Password:
R4> enable
Password:
R4# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R4# exit
[Connection to 172.16.28.100 closed by foreign host]
R1#
```

R3

```
R1# telnet 172.16.28.101
Trying 172.16.28.101 ... Open
User Access Verification
Password:
R3> enable
Password:
R3# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R3# exit
[Connection to 172.16.28.101 closed by foreign host]
R1#
```

R2

```
R1# telnet 172.16.28.102
Trying 172.16.28.102 ... Open
User Access Verification
Password:
R2> enable
Password:
R2# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
R2# exit
[Connection to 172.16.28.102 closed by foreign host]
R1#
```

R1 からのプライベート DHCP アドレス プールの削除

自動インストール プロセスの最後のステップは、R1 からプライベート DHCP アドレス プールを削除することです。

```
R1(config)# no ip dhcp pool r4
R1(config)# no ip dhcp pool r3
R1(config)# no ip dhcp pool r2
```

この作業は、自動インストールを使用して LAN に接続されたデバイスを設定するための最後の手順です。

その他の参考資料

このセクションでは、シスコ ネットワーキング デバイスの基本設定に関する参考資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS XE ソフトウェアの自動インストール機能を使用した初めてのネットワーキング デバイスの設定	『Using AutoInstall to Remotely Configure Cisco Networking Devices』
Cisco IOS XE セットアップ モードを使用したネットワーキング デバイスの設定	『Using Setup Mode to Configure a Cisco Networking Device』
設定の基本的なコマンドと関連コマンド	目的のリリースの『*Cisco IOS XE Configuration Fundamentals Configuration Guide*』と、リリースに依存しない『Cisco IOS Configuration Fundamentals Command Reference』

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

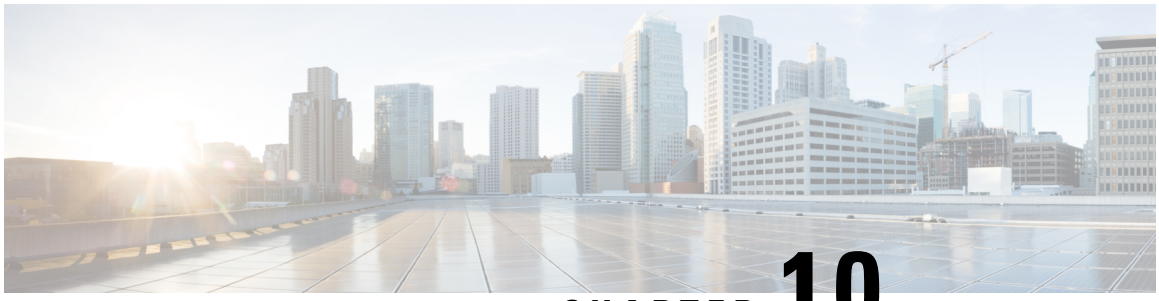
自動インストールを使用したシスコのネットワーキング デバイスの設定に関する機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 14: 自動インストールを使用したシスコ ネットワーキング デバイスのリモート設定の機能情報

機能名	リリース	機能の設定情報
LANインターフェイスにDHCPを使用した自動インストール	Cisco IOS XE Release 2.1	LAN インターフェイスに DHCP を使用した自動インストール機能では、LANインターフェイス（特にファストイーサネット、トークンリング、FDDI のインターフェイス）上での Cisco IOS 自動インストール用に、ブートストラッププロトコル（BOOTP）の使用を Dynamic Host Configuration Protocol（DHCP）の使用で置き換えることで、自動インストールの利点が強化されます。 この機能は、Cisco IOS XE Release 2.1 で、Cisco ASR 1000 シリーズ ルータに導入されました。
TCLスクリプトの自動インストール サポート	Cisco IOS XE Release 3.3SE	TCL スクリプトを使用する自動インストール機能では、インストールプロセスに柔軟性を持たせることで、自動インストール機能が強化されます。この機能を使用すると、ユーザーはダウンロードする対象に関する情報の取得、ファイル サーバーのタイプの選択、必須ファイル転送プロトコルの選択を行うようデバイスをプログラムすることができます。



CHAPTER 10

Unique Device Identifier の取得

Unique Device Identifier の取得機能は、この ID 情報を保存したシスコ製品から Unique Device Identifier（UDI）情報を取得および表示するための機能を提供します。

Unique Device Identifier の取得の前提条件

UDI 取得を使用するには、使用中のシスコ製品が UDI 対応である必要があります。UDI 対応のシスコ製品では、5 つの必須エンティティ MIB オブジェクトがサポートされます。5 つのエンティティ MIB v2（RFC-2737）オブジェクトは次のとおりです。

- entPhysicalName
- entPhysicalDescr
- entPhysicalModelName
- entPhysicalHardwareRev
- entPhysicalSerialNum

show inventory コマンドが使用可能な場合がありますが、UDI 対応ではないデバイスでそのコマンドを使用しても出力が生成されない可能性があります。

Unique Device Identifier の取得に関する情報

Unique Device Identifier の概要

識別可能な各製品は、エンティティ MIB（RFC-2737）およびそのサポート ドキュメントで定義されたエンティティです。シャーシなどの一部のエンティティには、スロットのようなサブエンティティがあります。ファスト イーサネット スイッチは、スタックなどのスーパーエンティティのメンバーである可能性があります。注文可能なシスコ製品のエンティティは、そのほとんどがUDIを割り当てられて出荷されます。UDI情報は、ラベルに印字され、ハードウェアデバイスに物理的に貼付されます。また、簡単にリモート検索できるよう、デバイス内に電子的に保存されます。

UDI は、次の要素で構成されています。

- 製品 ID（PID）
- バージョン ID（VID）
- シリアル番号（SN）

PIDは製品を発注するための名前で、従来は「製品名」または「部品番号」と呼ばれていました。これは、正しい交換部品を発注するために使用される ID です。

VIDは製品のバージョンです。製品が改訂されるたびに、VIDは増加します。VIDは、製品変更の通知を管理する業界のガイドラインである、Telcordia GR-209-CORE から取得された厳格なプロセスに従って増加されます。

SN はベンダー固有の製品の通し番号です。それぞれの製造済み製品には、現場では変更できない固有のシリアル番号が工場で割り当てられます。この番号は、製品の特定のインスタンスを個々に識別するための手段です。

Unique Device Identifier の取得機能の利点

- ネットワーク内の個別のシスコ製品を識別します。
- シスコ製品をシンプルに、クロスプラットフォームで、一貫して識別することで、資産管理の運用経費が削減されます。
- 交換可能な製品の PID を識別します。
- リコールまたはリビジョン対象の製品を容易に特定できます。
- シスコ製品のインベントリを自動化します（設備および資産管理）。
- 修理や交換サービスのためにシスコ製品のエンタイトルメント レベルを決定するためのメカニズムを提供します。

Unique Device Identifier の取得方法

Unique Device Identifier の取得

シスコ製品の ID 情報を取得および表示するには、このタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **show inventory** [**raw**] [*entity*]

DETAILED STEPS

ステップ 1 **enable**

特権 EXEC モードを開始します。パスワードを入力します（要求された場合）。

Example:

```
Router> enable
```

ステップ 2 **show inventory** [**raw**] [*entity*]

PID、VID、および SN が割り当てられているネットワーキング デバイスに取り付けられているすべてのシスコ製品についての情報を取得および表示するには、 **show inventory** コマンドを入力します。シスコ エンティティに PID が割り当てられていない場合、そのエンティティは取得または表示されません。

Example:

```
Router# show inventory
NAME: "Chassis", DESCR: "12008/GRP chassis"
PID: GSR8/40           ,  VID: V01,  SN: 63915640
NAME: "slot 0", DESCR: "GRP"
PID: GRP-B             ,  VID: V01,  SN: CAB021300R5
NAME: "slot 1", DESCR: "4 port ATM OC3 multimode"
PID: 4OC3/ATM-MM-SC    ,  VID: V01,  SN: CAB04036GT1
NAME: "slot 3", DESCR: "4 port 0C3 POS multimode"
PID: LC-4OC3/POS-MM    ,  VID: V01,  SN: CAB014900GU
NAME: "slot 5", DESCR: "1 port Gigabit Ethernet"
PID: GE-GBIC-SC-B      ,  VID: V01,  SN: CAB034251NX
NAME: "slot 7", DESCR: "GRP"
PID: GRP-B             ,  VID: V01,  SN: CAB0428AN4O
NAME: "slot 16", DESCR: "GSR 12008 Clock Scheduler Card"
PID: GSR8-CSC/ALRM     ,  VID: V01,  SN: CAB0429AUYH
NAME: "sfslot 1", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0428ALOS
NAME: "sfslot 2", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0429AU0M
NAME: "sfslot 3", DESCR: "GSR 12008 Switch Fabric Card"
PID: GSR8-SFC          ,  VID: V01,  SN: CAB0429ARD7
NAME: "PSslot 1", DESCR: "GSR 12008 AC Power Supply"
PID: FWR-GSR8-AC-B     ,  VID: V01,  SN: CAB041999CW
```

ネットワーキング デバイスに取り付けられている特定のタイプのシスコ エンティティの UDI 情報を表示するには、*entity* の引数値で **show inventory** コマンドを入力します。この例では、モジュールの RO 引数文字列に一致するシスコ エンティティのリストが表示されます。

Example:

```
Router# show inventory “module RO”
NAME: ''module R0'', DESCR: ''Cisco ASR1000 Route Processor 2''
PID: ASR1000-RP2 ,  VID: V01,  SN: JAE13041JEX
```

Note **raw** キーワード オプションの主な目的は、**show inventory** コマンド自体の問題をトラブルシューティングすることです。

Example:

```
Router# show inventory raw
NAME: “Chassis”, DESCR: “12008/GRP chassis”
PID:                    ,  VID: V01,  SN: 63915640
NAME: “slot 0”, DESCR: “GRP”
PID:                    ,  VID: V01,  SN: CAB021300R5
NAME: “slot 1”, DESCR: “4 port ATM OC3 multimode”
PID: 4OC3/ATM-MM-SC     ,  VID: V01,  SN: CAB04036GT1
NAME: “slot 3”, DESCR: “4 port 0C3 POS multimode”
PID: LC-4OC3/POS-MM     ,  VID: V01,  SN: CAB014900GU
```

トラブルシューティングのヒント

この章全体では、区切り文字（*d* 引数）の必要なコマンドが共通して使用されます。区切り文字にはどのような文字でも使用できますが、引用符（"）の使用を推奨します。これは、メッセージ自体の中でこの文字を使用することが通常はないためです。その他の一般に使用される区切り文字には、パーセント記号（%）またはスラッシュ（/）などがありますが、これらの文字は特定の Cisco IOS コマンド内で意味を持つため、推奨されません。たとえば、空きメッセージを「This terminal is idle」に設定するには、コマンド **vacant-message"Thisterminalisidle"** を入力します。

Unique Device Identifier の取得の設定例

UDI 取得機能の設定例はありません。**show inventory** コマンドの出力の表示例については、「Unique Device Identifier の取得」の項を参照してください。

その他の参考資料

このセクションでは、シスコ ネットワーキング デバイスの基本設定に関する参考資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
設定の基本的なコマンド	『*Cisco IOS Configuration Fundamentals Command Reference*』
Cisco IOS ソフトウェアの自動インストール機能を使用した初めてのネットワーキング デバイスの設定	『*Cisco IOS Configuration Fundamentals Configuration Guide*』の「Using AutoInstall to Remotely Configure Cisco Networking Devices」モジュール
Cisco IOS セットアップ モードを使用したネットワーキング デバイスの設定	『*Cisco IOS Configuration Fundamentals Configuration Guide*』の「Using Setup Mode to Configure a Cisco Networking Device」モジュール

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

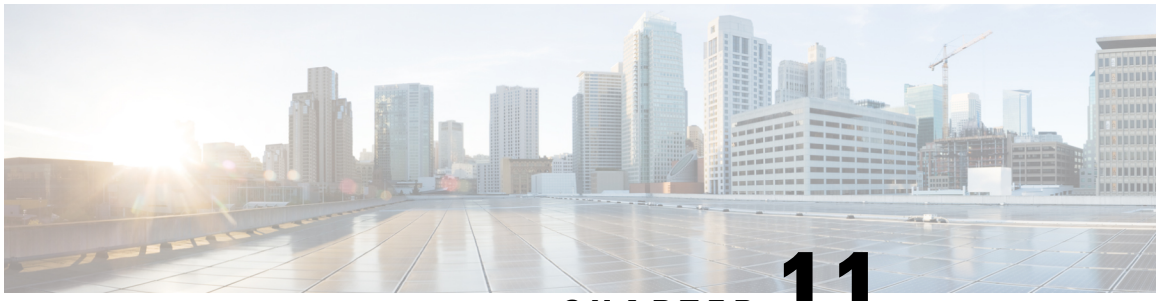
Unique Device Identifier の取得に関する機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

***Table 15: Unique Device Identifier* の取得に関する機能情報**

機能名	リリース	機能情報
Unique Device Identifier の取得	Cisco IOS XE Release 2.1	この機能が導入されました。



CHAPTER 11

CLI 出力の検索とフィルタリング

Cisco IOS CLI には、大量のコマンド出力を検索したり、出力をフィルタリングして不要な情報を除外するための手段が提供されています。これらの機能は、一般に大量のデータが表示される、**show** コマンドと **more** コマンドで使用できます。

Note **Show** コマンドと **more** コマンドは、常にユーザー EXEC モードまたは特権 EXEC モードで実行します。

画面に表示される内容を超えて出力が続く場合、Cisco IOS CLI では --More-- プロンプトが表示されます。Return キーを押すことで次の行が表示され、スペースキーを押すことで次の画面が表示されます。CLI ストリング検索機能を使用すると、--More-- プロンプトからの出力を検索またはフィルタリングできます。

機能情報の確認

Cisco Feature Navigator を使用すると、プラットフォーム、Cisco IOS ソフトウェア イメージ、および Catalyst OS ソフトウェア イメージの各サポート情報を検索できます。Cisco Feature Navigator には、http://www.cisco.com/go/cfn からアクセスします。Cisco.com のアカウントは必要ありません。

正規表現について

正規表現は、CLI ストリング検索機能によって、**show** コマンドまたは **more** コマンドの出力と照合されるパターン（句、数値、またはより複雑なパターン）です。正規表現では、大文字と小文字が区別され、複雑な一致要件を指定することが可能です。単純な正規表現には、Serial、

misses、138 などのエントリが含まれます。複雑な正規表現としては、00210... 、(is)、[Oo]utput などがあります。

正規表現は、単一文字パターンか複数文字パターンです。つまり、正規表現は、コマンド出力中の同じ 1 文字に一致する 1 つの文字か、コマンド出力中の同じ複数の文字に一致する複数の文字です。コマンド出力中のパターンをストリングと呼びます。この項では、単一文字パターンと複数文字パターンの作成について説明します。また、量指定子、選択、位置指定、カッコを使用したより複雑な正規表現についても説明します。

単一文字パターン

最も単純な正規表現は、コマンド出力内の同じ1つの文字と一致する単一文字です。任意の文字（A ～ Z、a ～ z）または数字（0 ～ 9）を 1 文字のパターンとして使用できます。また、その他のキーボード文字（「!」や「~」など）も 1 文字のパターンとして使用できますが、一部のキーボード文字は正規表現では特別な意味を持ちます。次の表に、特別な意味を持つキーボード文字の一覧を示します。

Table 16: 特別な意味を持つ文字

文字	特別な意味
.	スペースを含む任意の単一文字と一致します。
*	0 個以上のパターンのシーケンスと一致します。
+	1 個以上のパターンのシーケンスと一致します。
?	0 または 1 回のパターンと一致します。
^	ストリングの先頭と一致します。
$	ストリングの末尾と一致します。
_（アンダースコア）	カンマ（,）、左波カッコ（{）、右波カッコ（}）、左カッコ（(）、右カッコ（)）、ストリングの先頭、ストリングの末尾、またはスペースと一致します。

これらの特殊文字を単一文字パターンとして使用するときは、各文字の前にバックスラッシュ（\）を置いて特別な意味を除外してください。次の例は、それぞれドル記号、アンダースコア、プラス記号に一致する単一文字パターン マッチングの例です。

\$ _ \+

単一文字パターンを範囲指定して、コマンド出力とのマッチングを行うことができます。たとえば、文字 a、e、i、o、u のいずれかを含むストリングに一致する正規表現を作成できます。パターンマッチングが成功するためには、これらの文字のいずれかだけがストリング中に存在する必要があります。1 文字のパターンの範囲を指定するには、1 文字のパターンを角カッコ（**[]**）で囲みます。たとえば、**[aeiou]** は小文字アルファベットの 5 つの母音のうちの任意の 1

文字と一致しますが、**[abcdABCD]** は小文字または大文字アルファベットの最初の 4 つの文字のうちの任意の 1 文字と一致します。

ダッシュ（-）で区切って範囲の終点だけを入力することにより範囲を簡略化することができます。上の範囲は次のように単純化されます。

[a-dA-D]

ダッシュを範囲内の単一文字パターンとして追加するには、ダッシュをもう 1 つ追加し、その前にバックスラッシュを入力します。

[a-dA-D\-]

次に示すように、右角カッコ（]）を、範囲内の単一文字パターンとして追加することもできます。

[a-dA-D\-\]]

上の例は、大文字または小文字のアルファベットの最初の 4 文字、ダッシュ、右角カッコのいずれかに一致します。

範囲の先頭にキャレット（^）を追加することで、範囲の一致を反転させることができます。次の例は、その中の文字以外の文字に一致します。

[^a-dqsv]

次の例は、右角カッコ（]）または文字 d 以外のすべてと一致します。

[^\]d]

複数文字のパターン

正規表現を作成するとき、複数の文字を含むパターンを指定することもできます。複数文字正規表現は、文字、数字、特別な意味のないキーボード文字を組み合わせて作成します。たとえば、a4% は複数文字の正規表現です。文字をそのとおりに解釈することを指示するには、特別な意味のあるキーボード文字の前にバックスラッシュを挿入します。

複数文字パターンでは、順序が大切です。**a4%** という正規表現は、a という文字のあとに 4 が続き、そのあとに % 記号が続く文字と一致します。ストリングの中に a4% という文字がその順序で含まれていないと、パターン マッチングは失敗します。複数文字の正規表現 **a.** では、ピリオド文字の特別な意味を使用し、文字 a の後に任意の 1 文字が続く文字列と一致します。この例では、ab、a!、または a2 というストリングはすべてこの正規表現と一致します。

ピリオド文字の特別な意味を無効にするには、その前にバックスラッシュを挿入します。たとえば、表現 **a\.** がコマンド構文で使用されている場合、ストリング a. だけが一致します。

すべての文字、すべての数字、すべてのキーボード文字、文字と数字とその他のキーボード文字の組み合わせを含む複数文字正規表現を作成できます。たとえば、**telebit3107v32bis** は有効な正規表現です。

量指定子

Cisco IOS ソフトウェアに対して、指定した正規表現の複数の出現に一致させることを指示するため、より複雑な正規表現を作成できます。そのためには、単一文字パターンおよび複数文

字パターンとともに、いくつかの特殊文字を使用します。次の表は、「複数」の正規表現を示す特殊文字の一覧を示します。

Table 17: 量指定子として使用される特殊文字

文字	説明
*	0 以上の単一文字パターンまたは複数文字パターンと一致します。
+	1 以上の単一文字パターンまたは複数文字パターンと一致します。
?	1 以上の単一文字パターンまたは複数文字パターンの 0 回または 1 回の出現と一致します。

次の例は、空文字を含む文字 a の任意の回数の出現と一致します。

a*

次のパターンでは、ストリングが一致するためには、文字 a が少なくとも 1 文字含まれていることが必要です。

a+

次のパターンは、ストリング bb または bab と一致します。

ba?b

次のストリングは、任意の数のアスタリスク（*）と一致します。

******

複数文字パターンとともに量指定子を使用するには、パターンをカッコで囲みます。次の例で、パターンは複数文字ストリング ab の任意の回数の出現と一致します。

(ab)*

より複雑な例として、次のパターンは、英数字のペアの 1 つ以上のインスタンスに一致しますが、空文字には一致しません（つまり、空のストリングは一致しません）。

([A-Za-z][0-9])+

量指定子（*、+、または?）を使用した一致の順序は、最長構造優先です。ネストした構造は、外側から内側に一致します。連結された構造は、構造の左側から一致します。そのため、この正規表現は A9b3 に一致しますが、9Ab3 には一致しません。これは、英字が数字の前に指定されているためです。

代替

選択を使用すると、ストリングに対して一致する代替パターンを指定できます。選択肢は垂直線（|）で区切ります。代替パターンのうちの 1 つがストリングに一致します。たとえば、正規表現 **codex|telebit** は、ストリング codex またはストリング telebit に一致しますが、codex と telebit の両方には一致しません。

位置指定

Cisco IOS ソフトウェアに対し、ストリングの先頭または末尾に対して正規表現パターンを一致させることを指示できます。つまり、ストリングの先頭または末尾に特定のパターンが含まれていることを指定できます。ストリングの一部に対してこれらの正規表現を「位置指定」するには、次の表に示す特殊文字を使用します。

Table 18: 位置指定に用いられる特殊文字

文字	説明
^	ストリングの先頭と一致します。
$	ストリングの末尾と一致します。

たとえば、正規表現 **^con** は con で始まる任意のストリングと一致し、**$sole** は sole で終わる任意のストリングと一致します。

^ 記号は、ストリングの先頭を示すのに加えて、角カッコの中で使用された場合に論理的な「not」を示すものとして使用できます。たとえば、正規表現 **[^abcd]** は、a、b、c、または d 以外の任意の単一文字に一致する範囲を示します。

これらの位置指定文字は、特殊文字アンダースコア（_）とともに使用します。アンダースコアは、ストリングの先頭（^）、ストリングの末尾（$）、カッコ（( )）、スペース（ ）、波カッコ（{}）、カンマ（,）、アンダースコア（_）に一致します。アンダースコア文字を使用すると、パターンがストリング中のいずれかの場所に存在することを指定できます。たとえば、**_1300_** は、ストリング中のいずれかの場所に 1300 がある任意のストリングに一致します。ストリング 1300 の前後にスペース、波カッコ、カンマ、アンダースコアのいずれかがあってもかまいません。そのため、{1300_ は正規表現 **_1300_** に一致しますが、21300 や 13000 は一致しません。

アンダースコア文字を使用することで、長い正規表現リストを置き換えることができます。たとえば、**^1300()()1300${1300,,1300,{1300},1300,(1300** と指定する代わりに、**_1300_** と指定できます。

後方参照のためのカッコ

「繰り返し指定」のセクションに示したように、複数文字正規表現をカッコで囲み、パターンの出現を繰り返すことができます。また、単一文字パターンまたは複数文字パターンをカッコで囲み、Cisco IOS ソフトウェアに対して、正規表現の別の場所で使用するためにパターンを覚えておくことを指示できます。

前のパターンを後方参照する正規表現を作成するには、カッコを使用して特定のパターンの記憶を指示し、バックスラッシュ（\）の後に数字を使用して記憶したパターンを再利用します。数字は、正規表現パターン内のカッコの出現を指定します。正規表現内に複数のパターンがある場合、\1 は最初に記憶したパターンを示し、\2 は 2 番目に記憶したパターンとなり、以下同様となります。

次の正規表現では、後方参照のためにカッコを使用しています。

a(.)bc(.)\1\2

この正規表現は、後に任意の文字（文字番号 1 とする）が続き、その後に bc が続き、その後に任意の文字（文字番号 2 とする）が続き、そのまた後に文字番号 1 が再び続き、最後に文字番号が続く文字 a と一致します。2 が続くストリングに一致します。そのため、この正規表現は aZbcTZT に一致します。ソフトウェアは、文字番号 1 が Z であり、文字番号 2 が T であることを記憶し、正規表現の後半で Z と T を再度使用します。

show コマンドの検索とフィルタリング

show コマンドの出力を検索するには、特権 EXEC モードで次のコマンドを使用します。

コマンド	目的
`Router# show any-command | begin regular-expression`	**show** コマンドのフィルタリングされていない出力を、正規表現を含む最初の行で開始します。

Note Cisco IOS のマニュアルでは、縦線を、一般に構文の選択肢を示すために使用します。しかし、**show** コマンドと **more** コマンドの出力を検索するには、パイプ文字（縦線）を入力する必要があります。このセクションでは、パイプを入力する必要があることを示すために、太字（|）で表します。

show コマンドの出力をフィルタリングするには、特権 EXEC モードで次のコマンドのいずれかを使用します。

コマンド	目的
`Router# show any-command | exclude regular-expression`	正規表現を含まない出力行を表示します。
`Router# show any-command | include regular-expression`	正規表現を含む出力行を表示します。

ほとんどのシステムで、Ctrl+Z キーの組み合わせを使用して、いつでも出力を中断し特権 EXEC モードに戻ることができます。たとえば、**showrunning-config|beginhostname** コマンドを使用して、実行コンフィギュレーションファイルの、ホスト名の設定を含む行から表示を開始できます。次に、関心のある情報の最後まで確認し終えたら、Ctrl+Z を使用します。

Note 感嘆符（!）またはセミコロン（;）が続く文字は、コメントとして扱われ、コマンドでは無視されます。

more コマンドの検索とフィルタリング

more コマンドは、**show** コマンドと同様に検索できます（**more** コマンドは、**show** コマンドと同じ機能を実行します）。**more** コマンドの出力を検索するには、ユーザー EXEC モードで次のコマンドを使用します。

コマンド	目的
`Router# more` *any-command* `| begin` *regular-expression*	**more** コマンドのフィルタリングされていない出力を、正規表現を含む最初の行で開始します。

more コマンドは、**show** コマンドと同様にフィルタリングできます。**more** コマンドの出力をフィルタリングするには、ユーザー EXEC モードで次のコマンドのいずれかを使用します。

コマンド	目的
`Router# more` *any-command* `| exclude` *regular-expression*	正規表現を含まない出力行を表示します。
`Router# more` *any-command* `| include` *regular-expression*	正規表現を含む出力行を表示します。

--More-- プロンプトからの検索およびフィルタリング

--More-- プロンプトから出力を検索できます。**show** コマンドまたは **more** コマンドの出力を --More-- プロンプトから検索するには、ユーザー EXEC モードで次のコマンドを使用します。

コマンド	目的
`-More-` `/` *regular-expression*	フィルタリングされていない出力を、正規表現を含む最初の行で開始します。

--More-- プロンプトから出力をフィルタリングできます。ただし、各コマンドに対して 1 つのフィルタだけを指定できます。フィルタは、**show** コマンドまたは **more** コマンドの出力が終了するか、出力を中断（Ctrl+Z または Ctrl+6 を使用します）するまで継続されます。そのため、元のコマンドか前の --More-- プロンプトですでにフィルタを指定してある場合、--More-- プロンプトで別のフィルタを追加できません。

Note 検索とフィルタリングは異なる機能です。**begin** キーワードを使用してコマンド出力を検索し、同時に --More-- プロンプトでフィルタを指定することはできます。

--More-- プロンプトで **show** コマンドまたは **more** コマンドの出力をフィルタリングするには、ユーザー EXEC モードで次のコマンドのいずれかを使用します。

コマンド	目的
-More- **-** *regular-expression*	正規表現を含まない出力行を表示します。
-More- **+** *regular-expression*	正規表現を含む出力行を表示します。

CLI 出力の検索とフィルタリングの例

次に、**morenvram:startup-config|begin** 特権 EXEC モード コマンドの部分的な出力例を示します。これは、正規表現を含む最初の行で、フィルタリングされていない出力が開始されています。--More-- プロンプトで、正規表現 ip を含む出力行を除外するためのフィルタを指定します。

```
Router# more nvram:startup-config | begin ip
ip subnet-zero
ip domain-name cisco.com
ip name-server 192.168.48.48
ip name-server 172.16.2.132
!
isdn switch-type primary-5ess
.
.
.
interface Ethernet1
 ip address 10.5.5.99 10.255.255.0
 --More--
-ip
filtering...
 media-type 10BaseT
!
interface Serial0:23
 encapsulation frame-relay
 no keepalive
 dialer string 4001
 dialer-group 1
 isdn switch-type primary-5ess
 no fair-queue
```

次に、**morenvram:startup-config|include** コマンドの出力例の一部を示します。正規表現 ip を含む行だけが表示されています。

```
Router# more nvram:startup-config | include ip
ip subnet-zero
ip domain-name cisco.com
ip name-server 1192.168.48.48
ip name-server 172.16.2.132
```

次に、**morenvram:startup-config|exclude** コマンドの出力例の一部を示します。正規表現 service を含む行が除外されています。--More-- プロンプトで、正規表現 Dialer1 をフィルタとして指定します。このフィルタを指定することにより、Dialer1 を含む最初の行で出力が再開されます。

```
Router# more nvram:startup-config | exclude service
!
version 12.2
!
hostname router
!
boot system flash
no logging buffered
!
ip subnet-zero
ip domain-name cisco.com
.
.
.
--More--
/Dialer1
filtering...
interface Dialer1
 no ip address
 no ip directed-broadcast
 dialer in-band
 no cdp enable
```

次に、出力の検索が指定された、**showinterface** コマンドの部分的な出力例を示します。パイプの後でキーワード **beginEthernet** を使用することで、正規表現 Ethernet を含む最初の行でフィルタリングされていない出力が開始されます。--More-- プロンプトで、正規表現 Serial を含む行だけを表示するフィルタを指定します。

```
Router# show interface | begin Ethernet
Ethernet0 is up, line protocol is up
Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Description: ip address is 172.1.2.14 255.255.255.0
  Internet address is 172.1.2.14/24
.
.
.
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
--More--
+Serial
filtering...
Serial1 is up, line protocol is up
Serial2 is up, line protocol is up
Serial3 is up, line protocol is down
Serial4 is down, line protocol is down
Serial5 is up, line protocol is up
```

```
Serial6 is up, line protocol is up
Serial7 is up, line protocol is up
```

次に、**showbuffers|exclude** コマンドの出力例の一部を示します。正規表現 ip を含む行が除外されています。--More-- プロンプトで、フィルタされていない出力を、Serial0 を含む最初の行から続行するための検索を指定します。

```
Router# show buffers | exclude 0 misses
Buffer elements:
     398 in free list (500 max allowed)
Public buffer pools:
Small buffers, 104 bytes (total 50, permanent 50):
     50 in free list (20 min, 150 max allowed)
     551 hits, 3 misses, 0 trims, 0 created
Big buffers, 1524 bytes (total 50, permanent 50):
     49 in free list (5 min, 150 max allowed)
Very Big buffers, 4520 bytes (total 10, permanent 10):
.
.
.
Huge buffers, 18024 bytes (total 0 permanent 0):
     0 in free list (0 min, 4 max allowed)
--More--
/Serial0
filtering...
Serial0 buffers, 1543 bytes (total 64, permanent 64):
     16 in free list (0 min, 64 max allowed)
     48 hits, 0 fallbacks
```

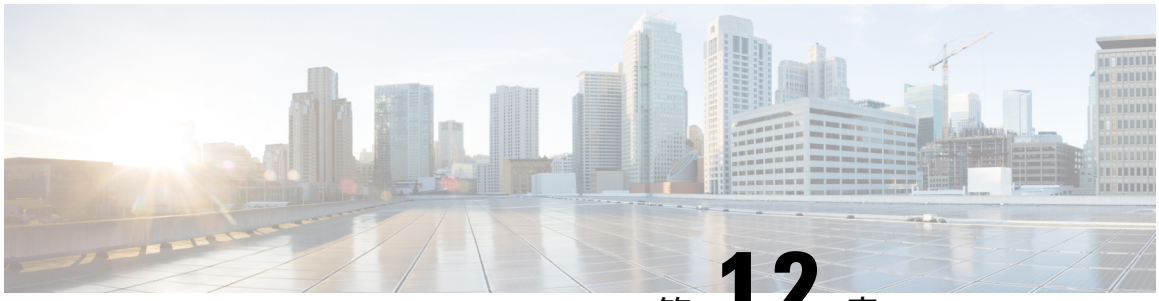
次に、**showinterface|include** コマンドの出力例の一部を示します。パイプ（|）の後で **include(is)** キーワードを使用することにより、正規表現 (is) が含まれる行だけが表示されます。カッコにより、is の前後にスペースが含まれることが指定されます。カッコを使用することで、is の前後にスペースを含む行だけが出力に含まれます（「disconnect」などの文字は検索から除外されます）。

```
router# show interface | include ( is )
ATM0 is administratively down, line protocol is down
  Hardware is ATMizer BX-50
Dialer1 is up (spoofing), line protocol is up (spoofing)
  Hardware is Unknown
  DTR is pulsed for 1 seconds on reset
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.6399 (bia 0060.837c.6399)
  Internet address is 172.21.53.199/24
Ethernet1 is up, line protocol is up
  Hardware is Lance, address is 0060.837c.639c (bia 0060.837c.639c)
  Internet address is 10.5.5.99/24
Serial0:0 is down, line protocol is down
  Hardware is DSX1
.
.
.
 --More--
```

--More-- プロンプトで、Serial0:13 を含む最初の行でフィルタリングされた出力を続行する検索を指定します。

```
/Serial0:13
filtering...
```

```
Serial0:13 is down, line protocol is down
  Hardware is DSX1
  Internet address is 10.0.0.2/8
     0 output errors, 0 collisions, 2 interface resets
  Timeslot(s) Used:14, Transmitter delay is 0 flag
```



第 12 章

同意トークン

同意トークンは、ネットワーク管理者と Cisco Technical Assistance Centre（Cisco TAC）の相互の同意により、システムシェルにアクセスする組織のネットワーク管理者を認証するために使用されるセキュリティ機能です。

同意トークンの制約事項

- 同意トークンはデフォルトで有効であり、無効にすることはできません。
- デバイスからチャレンジが送信された後、30 分以内に応答を入力する必要があります。入力しないとチャレンジが期限切れになり、新しいチャレンジの要求が必要になります。
- 単一の応答は、対応するチャレンジに対して 1 回だけ有効です。
- ルートシェルアクセスの最大承認タイムアウトは 7 日間です。
- スイッチオーバーイベント後、既存の同意トークンベースの承認はすべて期限切れとして処理されます。その後、サービスアクセスの新しい認証シーケンスを再起動する必要があります。
- シスコのチャレンジ署名サーバー上の同意トークン応答生成にアクセスできるのは、シスコ認定担当者のみです。
- システムシェルアクセスのシナリオでは、承認タイムアウトが発生するか、または同意トークン終了承認コマンドによってシェル承認が明示的に終了されるまで、シェルを終了しても承認は終了しません。

システムシェルアクセスの目的を達成したら、同意トークン終了コマンドを明示的に発行することによって、システムシェルの承認を強制終了することを推奨します。

同意トークンに関する情報

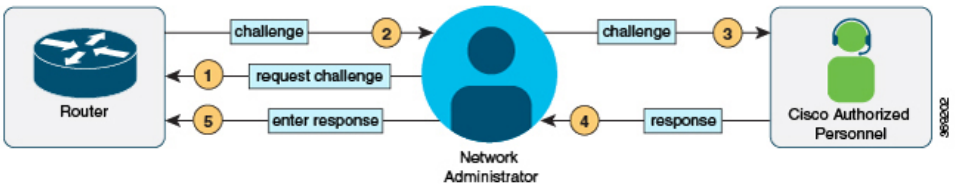
一部のデバッグシナリオでは、Cisco TAC エンジニアが特定のデバッグ情報を収集したり、実稼働システムでライブデバッグを実行する必要がある場合があります。このような場合、Cisco TACエンジニアは、デバイスのシステムシェルにアクセスするようユーザー（ネットワーク管理者）に依頼します。同意トークンは、システムシェルへの特権アクセス、制限アクセス、およびセキュアアクセスを提供する、ロック、ロック解除、および再ロックのメカニズムです。

システムシェルへのアクセスを要求する場合は、認証を受ける必要があります。最初にコマンドを実行し、デバイスの同意トークン機能を使用してチャレンジを生成する必要があります。デバイスは、固有のチャレンジを出力として生成します。このチャレンジ文字列をコピーし、電子メールまたはインスタントメッセージでシスコ認定担当者に送信する必要があります。

シスコ認定担当者は、一意のチャレンジ文字列を処理し、一意のレスポンスを生成します。シスコ認定担当者はこのレスポンス文字列をコピーし、電子メールまたはインスタントメッセージで送信します。

次に、このレスポンス文字列をデバイスに入力する必要があります。チャレンジ/レスポンスペアが一致すると、システムシェルへのアクセスが許可されます。一致しない場合は、エラーが表示され、認証プロセスを繰り返す必要があります。

システムシェルにアクセスしたら、Cisco TAC エンジニアが必要とするデバッグ情報を収集します。システムシェルへのアクセスが完了したら、セッションを終了し、デバッグプロセスを続行します。



システムシェルアクセスの同意トークン承認プロセス

ここでは、システムシェルにアクセスするための同意トークン承認のプロセスについて説明します。

手順の概要

1. 指定された期間、システムシェルへのアクセスを要求するチャレンジを生成します。
2. シスコ認定担当者にチャレンジ文字列を送信します。
3. デバイスにレスポンス文字列を入力します。

4. セッションを終了します。

手順の詳細

ステップ 1 指定された期間、システムシェルへのアクセスを要求するチャレンジを生成します。

例：

```
Device# request consent-token generate-challenge shell-access auth-timeout 900
zSSbIAAAQEBAQAAAPgAPAAAAAMCH86csUmTDlOBAQOFvd7CxqRUsD7B4AwWQABAAAC8GAAhDvEFBEANIwcAGNUQV9ERUlPXNUQV9TSUdOSUSkMOtnWQgCOM5CDAwILNMLb5CQAIOVpQIEAESESKRd=
Device#
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation attempt:
Shell access 0).
```

request consent-token generate-challenge shell-access *time-validity-slot* コマンドを使用して、チャレンジの要求を送信します。システムシェルへのアクセスを要求する期間（分単位）は、time-slot-period です。

この例の期間は、セッションの期限切れ後 900 分です。

デバイスは、固有のチャレンジを出力として生成します。このチャレンジは、base-64 形式の文字列です。

ステップ 2 シスコ認定担当者にチャレンジ文字列を送信します。

デバイスによって生成されたチャレンジ文字列を、電子メールまたはインスタントメッセージでシスコ認定担当者に送信します。

シスコ認定担当者は固有のチャレンジ文字列を処理し、レスポンスを生成します。レスポンスもまた、固有の base-64 文字列です。シスコ認定担当者はこのレスポンス文字列をコピーし、電子メールまたはインスタントメッセージで送信します。

ステップ 3 デバイスにレスポンス文字列を入力します。

例：

```
Device# request consent-token accept-response shell-access
[redacted]
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success: Shell
access 0).

Device# request platform software system shell
Activity within this shell can jeopardize the functioning of the system.
Are you sure you want to continue? [y/n] y
Device#
*Jan 18 02:56:59.714: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authorization for Shell access
0 will expire in 10 min).
```

request consent-token accept-response shell-access *response-string* コマンドを使用して、シスコ認定担当者から送信されたレスポンス文字列を入力します。

チャレンジ/レスポンスペアが一致すると、システムシェルへのアクセスが許可されます。チャレンジ/レスポンスペアが一致しない場合は、エラーが表示され、手順 1 ー 3 を繰り返す必要があります。

承認されると、要求されたタイムスロットのシステムシェルにアクセスできます。

承認セッションの残り時間が 10 分になると、デバイスはメッセージを送信します。

ステップ 4 セッションを終了します。

例：

```
Device# request consent-token terminate-auth
% Consent token authorization termination success

Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate authentication: Shell
 access 0).
Device#
```

システムシェルへのアクセスが終了したら、**request consent-token terminate-auth** コマンドを使用してセッションを終了できます。このコマンドを使用して、承認タイムアウトの前にセッションを強制終了することもできます。要求したタイムスロットが期限切れになると、セッションも自動的に終了します。

開発キーとリリースキー

Cisco IOS XE セキュアブート機能により、シスコの署名付きソフトウェアのみが Cisco IOS XE プラットフォームにロードされます。開発キーインストール機能を導入する前に、Cisco IOS XE プラットフォームには開発公開キーとリリース公開キーが付属しています。これらのキーは、対応する秘密キーによって署名されたイメージを検証するために使用されます。開発キーのインストール機能をサポートするCisco IOS XEプラットフォームのサブセットは、開発公開キーのないリリース公開キーのみで出荷されます。この機能の変更により、イメージ検証用の開発公開キーがないため、開発秘密キーで署名されたイメージは起動しません。ただし、何らかの理由で、Cisco IOS XEデバイスがシスコに返送された場合、製品の返品および交換（RMA）担当者は、開発秘密キーで署名されたイメージをロードする必要があります。これには、RMAスペシャリストがデバイスに開発公開キーをインストールして、開発秘密キーで署名されたイメージの検証に合格することを確認する必要があります。Dev公開キーをインストールするには、次のセクションで説明するコマンドを使用します。

開発キーアクセスのための同意トークン認証プロセス

ここでは、開発キーにアクセスするための同意トークン承認のプロセスについて説明します。

手順の概要

1. 指定された期間の開発キーへのアクセスを要求するチャレンジを生成します。
2. シスコ認定担当者にチャレンジ文字列を送信します。
3. デバイスにレスポンス文字列を入力します。
4. セッションを終了します。

手順の詳細

ステップ 1 指定された期間の開発キーへのアクセスを要求するチャレンジを生成します。

例：

```
Device# request consent-token generate-challenge dev-key auth-timeout 900
zSSbAAAQEBAQAAAEgAEAAAAAMCH8csUmDlOAAQFvd7CxqRUcD7B4AwWQAEAAAG8GAAhDAEFtEAUIwAGENQV9ERUlPXNUQ9TSUdSU5HX0tENQgAC0M5CDAwLUNMLUs5CQALOVpQLEAESESKkI=
Device#
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation attempt:
Dev key install).
```

request consent-token generate-challenge Dev-key *time-validity-slot* コマンドを使用して、チャレンジの要求を送信します。開発キーへのアクセスを要求する期間（分単位）は、time-slot-period です。

この例の期間は、セッションの期限切れ後 900 分です。

デバイスは、固有のチャレンジを出力として生成します。このチャレンジは、base-64 形式の文字列です。

（注） ゼロの認証タイムアウトは、永続的な開発公開キーのインストールを意味します。このような永続的なインストールは、セキュリティ上の理由から、シスコの内部デバイスにのみ許可されています。

ステップ 2 シスコ認定担当者にチャレンジ文字列を送信します。

デバイスによって生成されたチャレンジ文字列を、電子メールまたはインスタントメッセージでシスコ認定担当者に送信します。

シスコ認定担当者は固有のチャレンジ文字列を処理し、レスポンスを生成します。レスポンスもまた、固有の base-64 文字列です。シスコ認定担当者はこのレスポンス文字列をコピーし、電子メールまたはインスタントメッセージで送信します。

ステップ 3 デバイスにレスポンス文字列を入力します。

例：

```
Device# request consent-token accept-response dev-key
[redacted]
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success: dev key
 access 0).
```

request consent-token accept-response dev-key *response-string* コマンドを使用して、シスコ認定担当者から送信されたレスポンス文字列を入力します。

チャレンジ/レスポンスペアが一致すると、開発公開キーがインストールされます。チャレンジ/レスポンスペアが一致しない場合は、エラーが表示され、手順 1 ー 3 を繰り返す必要があります。

承認セッションの残り時間が 10 分になると、デバイスはメッセージを送信します。

ステップ 4 セッションを終了します。

例：

```
Device# request consent-token terminate-auth
% Consent token authorization termination success
```

```
Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate authentication: Dev
key install).
Device#
```

次に、システムが承認セッションの終了に失敗した場合の出力例を示します。

```
Router#request consent-token terminate-auth dev-key
% No in progress authorization, please generate challenge
Router#
```

開発キーへのアクセスが終了したら、**request consent-token terminate-auth** コマンドを使用してセッションを終了できます。このコマンドを使用して、承認タイムアウトの前にセッションを強制終了することもできます。要求したタイムスロットが期限切れになると、セッションも自動的に終了します。

インストール承認の検証

キーのインストールの承認を検証するには、**show platform software threat-token dev-key** コマンドを使用します。

```
Router#show platform software consent-token dev-key
Consent token statistics : dev-key
   Instance Id                      : 0
   Authorization remaining (minutes) : Permanent
   Challenge generation requests     : 1
   Challenge response timeouts       : 0
   Authentication success            : 1
   Authentication failure            : 0
   Authentication expiry             : 0
   Terminate authentication requests : 0
   Challenge generation errors       : 0
```

同意トークンの有効化または無効化

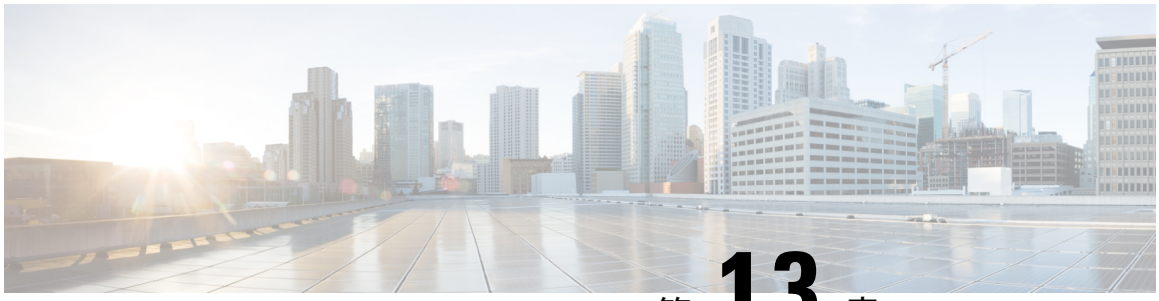
同意トークンをオンまたはオフにするには、次のデバッグコマンドを使用します。

- **debug platform software consent-token all**
- debug platform software consent-token errors

同意トークンの機能履歴と情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

リリース	機能情報
Cisco IOS XE Gibraltar 16.11.1	この機能が導入されました。
Cisco IOS XE Bengaluru 17.4.1	[開発キー（Dev Key）] および [リリースキー（Release Key）] オプションが導入されました。



第 13 章

ブート整合性の可視性

ブート整合性の可視性によって、シスコのプラットフォーム ID とソフトウェアの整合性情報が可視化され、実用可能になります。

ブート整合性の可視性について

プラットフォーム ID は、プラットフォームの製造元でインストールされた ID を示しています。ソフトウェアの整合性ではブート整合性の測定値が明らかになり、それを使用してプラットフォームが信頼できるコードを起動しているかどうかを評価できます。

ブート プロセス中に、ソフトウェアはブート ローダ アクティビティの各ステージのチェックサム レコードを作成します。

このレコードを取得して、シスコ認定レコードと比較し、ソフトウェアイメージが正規かどうかを確認できます。チェックサム値が一致していない場合は、シスコによって認定されていない、または未承認パーティによって改ざんされているソフトウェアイメージを実行している可能性があります。

ソフトウェア イメージとハードウェアの確認

このタスクでは、スイッチの起動時に作成されたチェックサム レコードを取得する方法について説明します。特権 EXEC モードで次のコマンドを入力します。

（注） 次のコマンドを実行した後で、メッセージ **% Please Try After Few Seconds** が CLI に表示されることがあります。これはCLIの障害を示すものではありませんが、必要な出力を取得するために必要な基盤となるインフラストラクチャの設定を示します。数分間待機して、コマンドを再度試すことをお勧めします。

メッセージ **% Error retrieving SUDI certificate** および **% Error retrieving integrity data** は、実際の CLI 障害を示します。

手順の概要

1. **show platform sudi certificate** ［**sign** ［**nonce** *nonce*］］
2. **show platform integrity** ［**sign** ［**nonce** *nonce*］］

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**show platform sudi certificate** ［**sign** ［**nonce** *nonce*］］ 例： `# show platform sudi certificate sign nonce 123`	特定の SUDI のチェックサム レコードを表示します。 • （オプション）**sign**：署名を示します • （オプション）**nonce**：ナンス値を入力します
ステップ 2	**show platform integrity** ［**sign** ［**nonce** *nonce*］］ 例： `# show platform integrity sign nonce 123`	ブート段階のチェックサム レコードを表示します。 • （オプション）**sign**：署名を示します • （オプション）**nonce**：ナンス値を入力します

プラットフォーム ID とソフトウェア整合性の確認

プラットフォーム ID の確認

次に、PEM形式でセキュアな固有デバイス識別子（SUDI）チェーンを表示する例を示します。最初の証明書はCisco Root CA 2048 で、2 番目はシスコの下位 CA（ACT2 SUDI CA）です。どちらの証明書も、https://www.cisco.com/security/pki/ で公開されているものと一致しているかを確認できます。3 番目は SUDI 証明書です。

```
Device#show platform sudi certificate sign nonce 123

-----BEGIN CERTIFICATE-----
MIIDQzCCAiugAwIBAgIQX/h7KCtU3I1CoxW1aMmt/zANBgkqhkiG9w0BAQUFADA1
MRYwFAYDVQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENB
IDIwNDgwHhcNMDQwNTE0MjAxNzEyWhcNMjkwNTE0MjAyNTQyWjA1MRYwFAYDVQQK
Ew1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgwggEg
```

```
MA0GCSqGSIb3DQEBAQUAA4IBDQAwggEIAoIBAQCwmrmrp68Kd6ficba0ZmKUeIhH
xmJVhEAyv8CrLqUccda8bnuoqrpu0hWISEWdovyD0My5jOAmaHBKeN8hF570YQXJ
FcjPFtolYYmUQ6iEqDGYeJu5Tm8sUxJszR2tKyS7McQr/4NEb7Y9JHcJ6r8qqB9q
VvYgDxFUl4F1pyXOWWqCZe+36ufijXWLbvLdT6ZeYpzPEApk0E5tzivMW/VgpSdH
jWn0f84bcN5wGyDWbs2mAag8EtKpP6BrXruOIIt6keO1aO6g58QBdKhTCytKmg9l
Eg6CTY5j/e/rmxrbU6YTYK/CfdfHbBcl1HP7R2RQgYCUTOG/rksc35LtLgXfAgED
o1EwTzALBgNVHQ8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUJ/PI
FR5umgIJFq0roIlgX9p7L6owEAYJKwYBBAGCNxUBBAMCAQAwDQYJKoZIhvcNAQEF
BQADggEBAJ2dhISjQal8dwy3U8pORFBi71R803UXHOjgxkhLtv5MOhmBVrBW7hmW
Yqpao2TB9k5UM8Z3/sUcuuVdJcr18JOagxEu5sv4dEX+5wW4q+ffy0vhN4TauYuX
cB7w4ovXsNgOnbFp1iqRe6lJT37mjpXYgyc81WhJDtSd9i7rp77rMKSsH0T8lasz
Bvt9YAretIpjsJyp8qS5UwGH0GikJ3+r/+n6yUA4iGe0OcaEb1fJU9u6ju7AQ7L4
CYNu/2bPPu8Xs1gYJQk0XuPL1hS27PKSb3TkL4Eq1ZKR4OCXPDJoBYVL0fdX4lId
kxpUnwVwwEpxYB5DC2Ae/qPOgRnhCzU=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEPDCCAySgAwIBAgIKYQlufQAAAAADDANBgkqhkiG9w0BAQUFADA1MRYwFAYD
VQQKEw1DaXNjbyBTeXN0ZW1zMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwNDgw
HhcNMTEwNjMwMTc1NjU3WhcNMjkwNTE0MjAyNTQyWjAnMQ4wDAYDVQQKEwVDaXNj
bzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEA0m5l3THIxA9tN/hS5qR/6UZRpdd+9aE2JbFkNjht6gfHKd477AkS
5XAtUs5oxDYVt/zEbslZq3+LR6qrqKKQVu6JYvH05UYLBqCj38s76NLk53905Wzp
9pRcmRCPuX+a6tHF/qRuOiJ44mdeDYZo3qPCpxzprWJDPclM4iYKHumMQMqmgmg+
xghHIooWS80BOcdiynEbeP5rZ7qRuewKMpl1TiI3WdBNjZjnpfjg66F+P4SaDkGb
BXdGj13oVeF+EyFWLrFjj97fL2+8oauV43Qrvnf3d/GfqXj7ew+z/sXlXtEOjSXJ
URsyMEj53Rdd9tJwHky8neapszS+r+kdVQIDAQABo4IBWjCCAVYwCwYDVR0PBAQD
AgHGMB0GA1UdDgQWBBRI2PHxwnDVW7t8cwmTr7i4MAP4fzAfBgNVHSMEGDAWgBQn
88gVHm6aAgkWrSugiWBf2nsvqjBDBgNVHR8EPDA6MDigNqA0hjJodHRwOi8vd3d3
LmNpc2NvLmNvbS9zZWN1cml0eS9wa2kvY3JsL2NyY2EyMDQ4LmNybDBQBggrBgEF
BQcBAQREMEIwQAYIKwYBBQUHMAKGNGh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3Vy
aXR5L3BraS9jZXJ0cy9jcmNhMjA0OC5jZXIwXAYDVR0gBFUwUzBRBgorBgEEAQkV
AQwAMEMwQQYIKwYBBQUHAgEWNWh0dHA6Ly93d3cuY2lzY28uY29tL3NlY3VyaXR5
L3BraS9wb2xpY2llcy9pbmRleC5odG1sMBIGA1UdEwEB/wQIMAYBAf8CAQAwDQYJ
KoZIhvcNAQEFBQADggEBAGh1qclr9tx4hzWgDERm371yeuEmqcIfi9b9+GbMSJbi
ZHc/CcCl0lJu0a9zTXA9w47H9/t6leduGxb4WeLxcwCiUgvFtCa51Iklt8nNbcKY
/4dw1ex+7amATUQO4QggIE67wVIPu6bgAE3Ja/nRS3xKYSnj8H5TehimBSv6TECi
i5jUhOWryAK4dVo8hCjkjEkzu3ufBTJapnv89g9OE+H3VKM4L+/KdkUO+52djFKn
hyl47d7cZR4DY4LIuFM2P1As8YyjzoNpK/urSRI14WdIlplR1nH7KND15618yfVP
0IFJZBGrooCRBjOSwFv8cpWCbmWdPaCQT2nwIjTfY8c=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIDhzCCAm+gAwIBAgIEAJT3DDANBgkqhkiG9w0BAQsFADAnMQ4wDAYDVQQKEwVD
aXNjbzEVMBMGA1UEAxMMQUNUMiBTVURJIENBMB4XDTE1MTExNDA5MzMzN1oXDTI1
MTExNDA5MzMzN1owczEsMCoGA1UEBRMjUElEOldTLUMzNjUwLTEyWDQ4VVEgU046
RkRPMTk0NkJHMDUxDjAMBgNVBAoTBUNpc2NvMRgwFgYDVQQLEw9BQ1QtMiBMaXRl
IFNVREkxGTAXBgNVBAMTEFdTLUMzNjUwLTEyWDQ4VVEwggEiMA0GCSqGSIb3DQEB
AQUAA4IBDwAwggEKAoIBAQC6SARWyImWrRV/x7XQogAE+02WmzKki+4arMVBvl9o
GgvJfkoJDdaHOROSUkEE3qXtd8N3lfKy3TZ+jtHD85m2aGz6+IRx/e/lLsQzi6dl
WIB+N94pgecFBONPR9wJriox1IGD3B43b0hMLkmro4R5Zrs8XFkDo9k1tBU7F2O7
GEzb/WkO5NLexznef2Niglx9fCDL0HC27BbsR5+03p8jhG0+mvrp8M9du1HKiGin
ZIV4XgTMp1/k/TVaIepEGZuWM3hxdUZjkNGG1c1m+oB8vLX3UlSL76sDBBoiaprD
rjXBgBIozyFW8tTjh50jMDG84hKD5s31ifOe4KpqEcnVAgMBAAGjbzBtMA4GA1Ud
DwEB/wQEAwIF4DAMBgNVHRMBAf8EAjAAME0GA1UdEQRGMESgQgYJKwYBBAEJFQID
oDUTM0NoaXBJRD1VWUpOTlZJMENBUkhVM1Z1SUVSbFl5QXlPQ0F4TXpvek5Ub3lN
U0EwS0NnPTANBgkqhkiG9w0BAQsFAAOCAQEADjtM8vdlf+p1WKSKX1C1qQ4aEnD5
p8T5e4iTer7Y1fbCrHIEEm3mnip+568j299z0H8V7PDp1ljuLHyMFTC+945F9RfA
eAuVWVb5A9dnGL8MssBJe2lVSnZwrWkT1EIdxLYrTiPAQHtll6CN77S4u/f71oYE
tzPE5AGfyGw7ro1MEPVGffaQmYUDAwKFNBH1uI7c2S1qlwk4WWZ6xxci+lhaQnIG
pWzapaiAYL1XrcBz4KwFc1ZZpQT6hHw24jzYaYimvCo+/kSKuA9xNdtSu18ycox0
zKnXQ17s6aChMMt7Y8Nh4iz9BDejoOF6/b3sM0wRi+2/4j+6/GhcMRs0Og==
-----END CERTIFICATE-----

Signature version: 1
Signature:
```

```
405C770D802B73947EDBF8DD0D2C8180F10D4B3EF9699444514219C579D2ED52F7D5
83E0F4408133FC4E9F549B2EB1C21725F7CB1C79F98271E47E780E703E674723880F
B52D4963E1D1FB9787B38E28B8E696570A180B7A2F1311B1F174EAA79F55DB4765DF
67386126D899E07EDF6C26E0A81272EAA114437DD03F26992937082756AE1F1BFAFB
BFACD6BE9CF9C84C961FACE9FA0FEE64D85AE4FA0086969D0702C536ABDB8FBFDC47
C14C17D02FEBF4F7F5BB24D2932FA876F56B4C07816270AA0B4195C53D975C85AEAE
3A74F2DBF293F52423ECB7B8539667080A9C57DA3E4B08B2B2CA623B2CBAF7080A0A
EB09B222E5B756970A3AA27E0F1D17C8A243
```

オプションの RSA 2048 署名は、3 つの証明書、署名のバージョンおよびユーザーにより提供されるナンスに対するものです

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
 2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

シスコの管理ソリューションには、上記の出力を解釈する機能が装備されています。しかし、OpenSSL コマンドを使用した簡単なスクリプトを使用してプラットフォームの ID を表示して署名を確認することもでき、それによってシスコの一意のデバイス ID を確保できます。

```
[linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:WS-C3650-12X48UQ SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite
SUDI/CN=WS-C3650-12X48UQ
```

ソフトウェア整合性の確認

次に、ブート段階のチェックサム レコードを表示する例を示します。ハッシュ測定値は、連続してブートされたソフトウェアの3つの段階それぞれについて表示されます。これらのハッシュをシスコが提供する基準値と比較できます。出力に署名するオプションを使用すると、出力は正規であり改ざんされていないことを保証する機能が検証ツールに付与されます。リプレイ攻撃から保護するために、ナンスを提供できます。

```
Device #show platform integrity sign nonce 456

Platform: WS-C3650-12X48UQ
Boot Loader Version: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.16, engineering
 software (D)
Boot Loader Hash: DB5A686E9F4CE358481DE3AF8B9C762F0A604E3B4764DF2A351F176E3D7
D3C60EB85C02906BD8CF28228C0DFC2AA8960CAFE6675D696E4ABA0CD687C0609E7E2
Boot 0 Version: F01062R15.0508d68fa2015-09-15
Boot 0 Hash: 6EF15CD54D3C66A8B644194A67B7ED57044C8C2E0EECB69736A7FFEC1F6D0EAD
OS Version: 2016-10-18_10.57_mundru
OS Hash: 4C85AECC88DAA49D940BBF65B1F17269F55C8D98DEFB4140F981923AA961140293E1
3B3E6E68CE3F8ED7F596CD858ACDD4BEF6538F59C1E243C351353026E6CD
PCR0: 90214167AAF35C06B2AC97292596E5669EAB72578FCDAD0B91746683BAA7B2B0
PCR8: FC2CE1BAC397F97008936DF372A2218BB16A798222B8FF55A7B6AEDA8018EDF5
Signature version: 1
Signature:
632A724F1AB6ADE134F6B0E8724D2052B3157F45B47E547763EE224A848E807CD737600587FF68
2526A8FE354A116CC9EDEBD9C659B9927336542EE4295084368327D01BD22AB4849BB3C007B6EB
B67708685FD6BC85DD045431E19A389FEB358894D4FBCF7C0FC960AC9133B61099DFD507F316C1
BF82F7F98687C7E7E8F99355DC1A95BD511B0B8DCB0CA909828F9EFBDF18847930392A8E3D072D
F3D90536880BAE9B7D7CF0E301D3F5AF16E7517FC2700E2F75911B836D6559A18E15B4CF452555
91656DF22DFF73392F777AEB796BCF9AC046C581ADEF19CA48A98F620BB58A79B32DA8B3BFB1CF
8399468A096E2F0C54B8B3ECD15EE3FE2C5ABDB5A029
```

オプションの RSA 2048 署名は SUDI 秘密キーで生成され、SUDI 証明書に含まれている SUDI 公開キーで確認できます。PCR 値全体の署名、署名のバージョンおよびユーザーにより提供されるナンスが表示されます。

```
RSA PKCS# 1 v1.5 Sign { <Nonce (UINT64)> || <Signature Version (UINT32)> || <PCR0 (32
bytes)> || <PCR8 (32 bytes)> }
```

シスコの管理ソリューションには、上記の出力を解釈する機能が装備されており、結果を公開されているシスコの値と比較し、署名を確認します。

ブート整合性の可視性の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

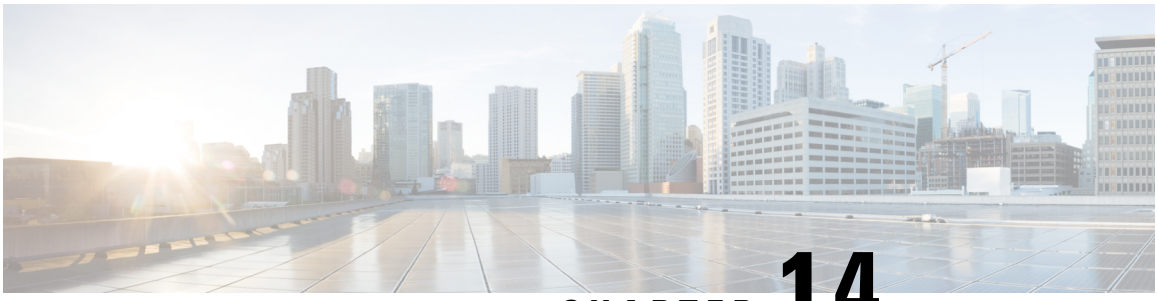
プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *19 : Open Plug-n-Play* エージェントの機能情報

機能名	リリース	機能情報
管理と制御：ブート整合性の可視性	Cisco IOS XE Everest 16.5.1	ブート整合性の可視性機能によって、シスコのプラットフォーム ID とソフトウェアの整合性情報が可視化され、実用可能になります。プラットフォーム ID は、プラットフォームの製造元でインストールされた ID を示しています。ソフトウェアの整合性ではブート整合性の測定値が明らかになり、それを使用してプラットフォームが信頼できるコードを起動しているかどうかを評価できます。 Cisco IOS XE Everest 16.5.1 では、Cisco ASR 1000 シリーズアグリゲーション ルータのサポートが追加されました。 このリリースで導入または変更されたコマンドはありません。

第 III 部

コンフィギュレーション ファイルの管理



CHAPTER 14

コンフィギュレーション ファイルの管理

コンフィギュレーションファイルを作成、ロード、維持することで、ユーザー設定のコマンドセットを生成し、現在のシスコ製ルーティングデバイスの機能性をカスタマイズできます。コンフィギュレーション ファイル管理コマンドの詳細については、『*Cisco IOS Configuration Fundamentals Command Reference*』を参照してください。

コンフィギュレーション ファイルの管理の前提条件

- ユーザーには、少なくとも Cisco IOS 環境とコマンドライン インターフェイスに関する基本的な知識が必要です。
- システムでは、少なくとも最小限の設定が実行されていることが必要です。**setup** コマンドを使用して基本的なコンフィギュレーション ファイルを作成できます（詳細については、「セットアップ モードを使用したシスコ ネットワーキング デバイスの設定」を参照してください）。

コンフィギュレーション ファイルの管理の制約事項

- このドキュメントで説明されている Cisco IOS コマンドの多くは、ルータの特定のコンフィギュレーション モードでのみ使用可能であり機能します。

コンフィギュレーション ファイルの管理について

コンフィギュレーション ファイルのタイプ

コンフィギュレーション ファイルには、現在のシスコ製ルーティング デバイス（ルータ、アクセス サーバー、スイッチなど）の機能をカスタマイズするために使用される、Cisco IOS ソフトウェア コマンドが含まれています。コマンドは、システムを起動したとき（startup-config ファイルから）、またはコンフィギュレーションモードでCLIにコマンドを入力したときに、Cisco IOS ソフトウェアによって解析（変換および実行）されます。

スタートアップ コンフィギュレーション ファイル（startup-config）は、ソフトウェアを設定するためにシステムの起動時に使用されます。実行コンフィギュレーション ファイル（running-config）には、ソフトウェアの現在の設定が含まれています。2つのコンフィギュレーション ファイルは別々の設定にできます。たとえば、コンフィギュレーションを永続的ではなく短期間だけ変更する場合があります。その場合は、**configure terminal** EXEC コマンドを使用して実行コンフィギュレーションを変更しますが、そのコンフィギュレーションは **copy running-config startup-config** EXEC コマンドを使用して保存しません。

実行コンフィギュレーションを変更するには、「CLI でのコンフィギュレーション ファイルの変更」の説明に従って、**configure terminal** コマンドを使用します。Cisco IOS コンフィギュレーション モードの使用時には、通常コマンドはすぐに実行され、入力直後またはコンフィギュレーション モードを終了した時点で実行コンフィギュレーション ファイルに保存されます。

スタートアップ コンフィギュレーション ファイルを変更するには、**copy running-config startup-config** EXEC コマンドを使用してスタートアップ コンフィギュレーションに実行コンフィギュレーション ファイルを保存するか、ファイル サーバーからスタートアップ コンフィギュレーションにコンフィギュレーション ファイルをコピーします（詳細については、「TFTP サーバーからルータへのコンフィギュレーション ファイルのコピー」を参照してください）。

コンフィギュレーション モードおよびコンフィギュレーション ソースの選択

ルータ上でコンフィギュレーション モードを開始するには、特権 EXEC プロンプトで **configure** コマンドを入力します。Cisco IOS ソフトウェアは次のプロンプトで応答し、端末、メモリ、またはネットワーク サーバー（ネットワーク）上に格納されたファイルのいずれかを、コンフィギュレーション コマンドのソースとして指定するように要求されます。

```
Configuring from terminal, memory, or network [terminal]?
```

端末からの設定では、コマンドラインにコンフィギュレーション コマンドを入力できます（次の項を参照してください）。メモリからの設定では、スタートアップ コンフィギュレーション ファイルがロードされます。詳細については、「スタートアップ コンフィギュレーション ファイルでのコンフィギュレーション コマンドの再実行」の項を参照してください。ネットワークからの設定では、ネットワーク経由でコンフィギュレーション コマンドをロードして実行でき

ます。詳細については、「TFTP サーバーからルータへのコンフィギュレーション ファイルのコピー」の項を参照してください。

CLI を使用したコンフィギュレーション ファイルの変更

Cisco IOS ソフトウェアは、1 行につき 1 つのコンフィギュレーション コマンドを受け入れます。コンフィギュレーションコマンドは、必要なだけ入力できます。コンフィギュレーションファイルには、入力したコマンドを説明するコメントを追加できます。コメントの先頭には、感嘆符（!）を付けます。コメントは NVRAM にもコンフィギュレーション ファイルのアクティブ コピーにも格納されないため、**show running-config or more system:running-config** EXEC コマンドでアクティブな設定のリストを表示しても、コメントは表示されません。**show startup-config** または **more nvram:startup-config** EXEC モード コマンドでスタートアップ コンフィギュレーションのリストを表示しても、コメントは表示されません。コメントは、コンフィギュレーション ファイルがルータにロードされたときにコンフィギュレーション ファイルから削除されます。ただし、ファイル転送プロトコル（FTP）、リモートコピープロトコル（RCP）、または Trivial File Transfer Protocol（TFTP）サーバ上に格納されているコンフィギュレーションファイルのコメントのリストは表示できます。CLI を使用してソフトウェアは設定するときは、ユーザの入力に従ってソフトウェアによりコマンドが実行されます。

コンフィギュレーション ファイルの場所

コンフィギュレーション ファイルは、次の場所に格納されます。

- 実行コンフィギュレーションは RAM に格納されます。
- クラス A フラッシュ ファイル システム プラットフォーム以外のすべてのプラットフォーム上では、スタートアップ コンフィギュレーションは不揮発性 RAM（NVRAM）に格納されます。
- クラス A フラッシュ ファイル システム プラットフォーム上では、スタートアップ コンフィギュレーションは CONFIG_FILE 環境変数で指定された場所に格納されます（詳細については、の項を参照してください）。CONFIG_FILE 変数は、デフォルトでは NVRAM になりますが、次のファイル システムのファイルも指定できます。
  - **nvram:**（NVRAM）

ネットワーク サーバからルータへのコンフィギュレーション ファイルのコピー

TFTP、rcp、または FTP サーバからルータの実行コンフィギュレーションまたはスタートアップ コンフィギュレーションへコンフィギュレーション ファイルをコピーできます。この機能は、次のいずれかの理由により実行する場合があります。

- バックアップ コンフィギュレーション ファイルを復元するため。

- 別のルータにコンフィギュレーションファイルを使用するため。たとえば、別のルータをネットワークに追加して、そのルータのコンフィギュレーションを元のルータと同様にする場合です。新しいルータにファイルをコピーすることにより、ファイル全体を再作成するのではなく、該当部分を変更できます。
- 同一のコンフィギュレーション コマンドをネットワーク内のすべてのルータにロードして、すべてのルータのコンフィギュレーションを同様にするため。

コマンドラインにコマンドを入力した場合と同様に、**copy**{**ftp:** | **rcp:** | **tftp:system:running-config**EXEC コマンドはルータにコンフィギュレーション ファイルをロードします。コマンドを追加する前に、ルータにより既存の実行コンフィギュレーションが消去されることはありません。コピーされたコンフィギュレーションファイル内のコマンドによって既存のコンフィギュレーションファイル内のコマンドが置き換えられた場合、既存のコマンドは消去されます。たとえば、コピーされたコンフィギュレーションファイルに含まれている特定のコマンドの IP アドレスが、既存のコンフィギュレーションと異なる場合は、コピーされたコンフィギュレーション内の IP アドレスが使用されます。ただし、既存のコンフィギュレーション内の一部のコマンドには、置き換えられたり無効になったりしないものもあります。このような場合、既存のコンフィギュレーション ファイルとコピーされたコンフィギュレーション ファイルで混成されたコンフィギュレーション ファイルが作成され、コピーされたコンフィギュレーション ファイルが優先されます。

コンフィギュレーションファイルをサーバ上に格納されているファイルの正確なコピーとして復元するには、そのコンフィギュレーションファイルをスタートアップ コンフィギュレーションに直接コピーし（**copyftp:**| **rcp:**| **tftp:**} **nvram:startup-config** コマンドを使用）、ルータをリロードする必要があります。

サーバからルータへコンフィギュレーションファイルをコピーするには、次の項で説明する作業を実行します。

使用するプロトコルは、使用中のサーバのタイプに応じて異なります。FTP および rcp のトランスポート メカニズムは、TFTP よりも高速でデータ配信の信頼性も優れています。これらの改善は、FTP および rcp のトランスポート メカニズムがコネクション型の TCP/IP スタック上に構築されており、これを使用しているために可能になりました。

ルータから TFTP サーバへのコンフィギュレーション ファイルのコピー

一部の TFTP 実装では、TFTP サーバー上にダミー ファイルを作成し、読み取り、書き込み、および実行を許可してから、ダミーファイルを上書きする形でファイルをコピーする必要があります。詳細については、ご使用の TFTP のマニュアルを参照してください。

ルータから FTP サーバーへのコンフィギュレーション ファイルのコピー

ルータから FTP サーバへコンフィギュレーション ファイルをコピーできます。

FTP ユーザ名およびパスワードの概要

FTPプロトコルでは、FTP要求ごとにリモートユーザー名およびパスワードを、クライアントがサーバーに送信する必要があります。FTPを使用してルータからサーバへコンフィギュレー

ション ファイルをコピーする場合、Cisco IOS ソフトウェアは次の順番で最初に発見した有効なユーザ名を送信します。

1. **copy** EXEC コマンドで指定されたユーザ名（ユーザ名が指定されている場合）。
2. **ip ftp username** グローバル コンフィギュレーション コマンドで設定されたユーザ名（コマンドが設定されている場合）。
3. Anonymous

ルータは次の順番で最初に発見した有効なパスワードを送信します。

1. **copy** コマンドで指定されたパスワード（パスワードが指定されている場合）。
2. **ip ftp password** コマンドで設定されたパスワード（コマンドが設定されている場合）。
3. ルータは、*username* *@routername* *.domain* というパスワードを生成します。変数 *username* は現在のセッションに関連付けられたユーザ名、*routername* は設定済みのホスト名、*domain* はルータのドメインです。

ユーザー名およびパスワードは、FTP サーバーのアカウントに関連付けられている必要があります。サーバに書き込む場合、ルータ上のユーザからの FTP 書き込み要求を受け入れるように、FTP サーバを適切に設定する必要があります。

このサーバーがディレクトリ構造をとっている場合、コンフィギュレーションファイルまたはイメージは、サーバー上のユーザー名と関連付けられたディレクトリに書き込まれるか、そのディレクトリからコピーされます。たとえば、システムイメージがサーバ上のユーザのホームディレクトリにある場合は、そのユーザの名前をリモート ユーザ名として指定します。

詳細については、ご使用の FTP サーバのマニュアルを参照してください。

すべてのコピー操作に使用するユーザ名およびパスワードを指定するには、**ip ftp username** および **ip ftp password** グローバル コンフィギュレーション コマンドを使用します。特定のコピー操作にのみ使用するユーザ名を指定する場合は、**copy** EXEC コマンド内でユーザ名を指定します。

Note パスワードには特殊文字「@」、「:」、および「/」を含めることはできません。これらの特殊文字が使用されている場合、コピーはサーバーの IP アドレスを解析できません。

VRF によるファイルのコピー

copy コマンドで指定した VRF インターフェイス経由でファイルをコピーできます。**copy** コマンドで VRF を指定するほうが、変更リクエストを使用して設定を変更しなくても送信元インターフェイスを直接変更でき、簡単で効率的です。

次の例に、**copy** コマンドを使用して VRF 経由でファイルをコピーする方法を示します。

```
Device# copy scp: slot0: vrf test-vrf
Device# copy scp: slot0: vrf test-vrf
Address or name of remote host [10.1.2.3]?
Source username [ScpUser]?
```

```
Source filename [/auto/tftp-server/ScpUser/vrf_test.txt]?
Destination filename [vrf_test.txt]?
Getting the vrf name as test-vrf
Password:
Sending file modes: C0644 10 vrf_test.txt
!
223 bytes copied in 22.740 secs (10 bytes/sec)
```

NVRAM より大きいコンフィギュレーション ファイル

NVRAM より大きいコンフィギュレーション ファイルを維持管理するには、以降の項の情報を知っておく必要があります。

コンフィギュレーション ファイルの圧縮

servicecompress-config グローバル コンフィギュレーション コマンドは、コンフィギュレーション ファイルを圧縮して NVRAM に格納することを指定します。コンフィギュレーション ファイルが圧縮されると、ルータは正常に機能します。システムの起動時に、システムはコンフィギュレーション ファイルが圧縮されていることを認識し、圧縮されたコンフィギュレーション ファイルを展開して、正常に処理を進めます。**morenvram:startup-config** EXEC コマンドにより、コンフィギュレーションが展開されてから表示されます。

コンフィギュレーション ファイルを圧縮する前に、適切なハードウェアのインストレーションおよびメンテナンス マニュアルを参照してください。ご利用のシステムの ROM がファイル圧縮をサポートしていることを確認します。サポートしていない場合、ファイル圧縮をサポートしている新しい ROM をインストールできます。

コンフィギュレーションのサイズは、NVRAM のサイズの 3 倍を超えてはいけません。NVRAM のサイズが 128 KB の場合、展開できる最大のコンフィギュレーション ファイルのサイズは 384 KB です。

servicecompress-config グローバル コンフィギュレーション コマンドは、Cisco IOS ソフトウェア Release 10 以降のブート ROM を使用している場合に限り実行できます。新しい ROM のインストールは 1 回限りの操作で、ROM に Cisco IOS Release 10 がない場合にのみ必要です。ブート ROM が圧縮コンフィギュレーションを認識しない場合は、次のメッセージが表示されます。

```
Boot ROMs do not support NVRAM compression Config NOT written to NVRAM
```

ネットワークからのコンフィギュレーション コマンドのロード

コンフィギュレーションが大きい場合は、FTP、TFTP のいずれかのサーバに格納しておき、システムの起動時にダウンロードすることもできます。ネットワーク サーバーを大きいコンフィギュレーションの保存に使用するためのコマンドの詳細については、「ルータから TFTP サーバーへのコンフィギュレーション ファイルのコピー」および「コンフィギュレーション ファイルをダウンロードするためのルータの設定」の各項を参照してください。

パーサー キャッシュの制御

Cisco IOS ソフトウェアの Cisco IOS コマンドライン パーサーは、コマンドラインを変換および実行（解析）します。パーサー キャッシュ機能は、大きいコンフィギュレーション ファイルを迅速に処理するために開発されました。これにより、ロード時間が大幅に改善されます。

パーサーキャッシュ機能では、簡略化された解析グラフをダイナミックに作成、キャッシュ、および再使用することにより、コンフィギュレーションファイル内の、前回使用された設定行と微妙に異なる設定行（たとえば pvc 0/100、pvc 0/101 など）が、迅速に認識および変換できるようになります。この改善は、主に同じようなコマンドを何百回、何千回と繰り返すコンフィギュレーション ファイルに役立ちます。このようなコンフィギュレーション ファイルには、サブインターフェイス用に何千もの仮想回線を設定する必要のある場合や、何百ものアクセスリストを設定する必要のある場合などがあります。数値の引数だけが異なる同一のコマンドが繰り返し使用されているファイルのほとんどで、性能が向上します。

パーサー キャッシュは、Cisco IOS Release 12.1(5)T 以降のリリースを使用するすべてのプラットフォームで、デフォルトでイネーブルにされています。ただし、大きいコンフィギュレーション ファイルを必要としないシスコ デバイスを使用しているユーザーの場合は、パーサーキャッシュをディセーブルにし、この機能で使用されるリソースを解放できます（この機能により使用されるメモリは、解析されるコンフィギュレーションファイルのサイズに依存しますが、通常は 512 KB 未満です）。

パーサーキャッシュを制御するには、いくつかの方法があります（これらはすべて任意です）。

- パーサー キャッシュのクリア：リソースを解放するか、またはパーサー キャッシュのメモリをリセットするために、パーサーキャッシュ機能に格納されている解析エントリおよびヒット数とミス数の統計情報をクリアすることもできます。
- パーサーキャッシュのディセーブル化：パーサーキャッシュ機能は、デフォルトでイネーブルされています。パーサー キャッシュ機能をディセーブルにするには、グローバル コンフィギュレーション モードで no parser cache コマンドを使用します。パーサー キャッシュがディセーブルになると、**noparsercache** コマンドラインが実行コンフィギュレーション ファイルに書き込まれます。システム リソースを解放するためにパーサー キャッシュをディセーブルにする場合は、**noparsercache** コマンドを発行する前にパーサー キャッシュをクリアする必要があります。パーサーキャッシュをディセーブルにした後は、パーサーキャッシュをクリアできません。
- パーサー キャッシュの再イネーブル化：パーサー キャッシュ機能をディセーブルにした後、再度イネーブルにするには、グローバル コンフィギュレーションモードで**parsercache** コマンドを使用します。
- パーサーのモニタリング：最後に解析されたコンフィギュレーションファイルに関する統計情報は、パーサーキャッシュ機能により解析されたコマンドのヒット数とミス数の統計情報とともにシステム メモリに格納されます。「hits（ヒット数）」および「misses（ミス数）」は、前回使用された類似するコマンドに対し、コンフィギュレーションセッション中にパーサーキャッシュが検出した一致数を示しています。一致したコマンド（「hits」）は、より効率的に解析されます。一致しなかったコマンド（「misses」）の解析時間は、パーサー キャッシュにより改善されることはありません。

コンフィギュレーション ファイルをダウンロードするルータの設定

システムの起動時に 1 つまたは 2 つのコンフィギュレーション ファイルをロードするようにルータを設定できます。コンフィギュレーションファイルは、コマンドラインにコマンドを入力した場合と同様に、メモリにロードされ読み込まれます。したがって、ルータのコンフィギュレーションは、元のスタートアップ コンフィギュレーションと 1 つまたは 2 つのダウンロードされたコンフィギュレーション ファイルで混成されたものになります。

ネットワークとホストのコンフィギュレーション ファイル

歴史的な理由から、ルータが最初にダウンロードするファイルは、ネットワーク コンフィギュレーション ファイルと呼ばれます。ルータが 2 番目にダウンロードするファイルは、ホスト コンフィギュレーションファイルと呼ばれます。2つのコンフィギュレーションファイルは、ネットワーク上のすべてのルータが、同一コマンドの多くを使用する場合に使用できます。ネットワーク コンフィギュレーション ファイルには、すべてのルータを設定するために使用される標準コマンドが含まれます。ホスト コンフィギュレーション ファイルには、特定の 1 つのホストに固有のコマンドが含まれます。2つのコンフィギュレーションファイルをロードする場合、ホスト コンフィギュレーション ファイルを、もう 1 つのファイルより優先させる必要があります。ネットワーク コンフィギュレーション ファイルおよびホスト コンフィギュレーションファイルは、両方とも TFTP、rcp、FTP のいずれかを介して到達可能なネットワーク サーバ上にあり、読み取り可能である必要があります。

コンフィギュレーション ファイル情報の管理方法

コンフィギュレーション ファイル情報の表示

コンフィギュレーションファイルに関する情報を表示するには、このセクションの手順を実行します。

SUMMARY STEPS

1. **enable**
2. **show boot**
3. **more** *file-url*
4. **show running-config**
5. **show startup-config**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
ステップ 2	**show boot** **Example:** `Device# show boot`	BOOT 環境変数の内容、CONFIG_FILE 環境変数によって指定されているコンフィギュレーション ファイルの名前、およびBOOTLDR環境変数の内容を示します。
ステップ 3	**more** *file-url* **Example:** `Device# more 10.1.1.1`	指定されたファイルの内容を表示します。
ステップ 4	**show running-config** **Example:** `Device# show running-config`	実行コンフィギュレーション ファイルの内容を表示します（**more system:running-config** コマンドのコマンド エイリアスです）。
ステップ 5	**show startup-config** **Example:** `Device# show startup-config`	スタートアップ コンフィギュレーション ファイルの内容を表示します。（**more nvram:startup-config** コマンドのコマンド エイリアスです）。 クラス A フラッシュ ファイル システム プラットフォーム以外のすべてのプラットフォーム上では、通常、デフォルトの startup-config ファイルはNVRAMに格納されます。クラス A フラッシュ ファイル システム プラットフォーム上では、CONFIG_FILE 環境変数はデフォルトの startup-config ファイルを指定します。CONFIG_FILE変数のデフォルトはNVRAMになります。

CLI でのコンフィギュレーション ファイルの変更

Cisco IOS ソフトウェアは、1 行につき 1 つのコンフィギュレーション コマンドを受け入れます。コンフィギュレーションコマンドは、必要なだけ入力できます。コンフィギュレーション ファイルには、入力したコマンドを説明するコメントを追加できます。コメントの先頭には、感嘆符（!）を付けます。コメントは NVRAM にもコンフィギュレーション ファイルのアクティブ コピーにも格納されないため、**show running-config or more system:running-config** EXEC コマンドでアクティブな設定のリストを表示しても、コメントは表示されません。**showstartup-config** または **more nvram:startup-config** EXEC モード コマンドでスタートアップ コンフィギュレーションのリストを表示しても、コメントは表示されません。コメントは、コンフィギュレーション ファイルがルータにロードされたときにコンフィギュレーション ファイルから削除されます。ただし、ファイル転送プロトコル（FTP）、リモート コピー プロトコル（RCP）、または Trivial File Transfer Protocol（TFTP）サーバ上に格納されているコンフィギュレーション ファイルのコメントのリストは表示できます。CLI を使用してソフトウェアは設定するときは、ユーザの入力に従ってソフトウェアによりコマンドが実行されます。CLI を

使用してソフトウェアを設定するには、特権EXECモードを開始して次のコマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. 次のいずれかを実行します。
   - **end**
   - **^Z**
4. **copy system:running-config nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
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。必要なコンフィギュレーション コマンドを入力します。Cisco IOS マニュアル セットに、テクノロジー別に編成されたコンフィギュレーション コマンドが説明されています。
ステップ 3	次のいずれかを実行します。 • **end** • **^Z** **Example:** `Device(config)# end`	コンフィギュレーションセッションを終了し、EXEC モードに戻ります。 **Note** Ctrl キーと Z キーを同時に押すと、画面に ^Z と表示されます。
ステップ 4	**copy system:running-config nvram:startup-config** **Example:** `Device# copy system:running-config nvram:startup-config`	実行コンフィギュレーション ファイルをスタートアップ コンフィギュレーション ファイルとして保存します。 **copy running-config startup-config** コマンドエイリアスも使用できますが、このコマンドは精度が高くないため、注意する必要があります。ほとんどのプラットフォーム上では、このコマンドによりコンフィギュレーションはNVRAMに保存されます。クラス A フラッシュ ファイル システムのプラットフォーム上では、この手順によりコンフィギュレーションは CONFIG_FILE 環境変数によって指定された場所に保存されます（デフォルトのCONFIG_FILE

	Command or Action	Purpose
		変数では、ファイルの保存先はNVRAMに指定されています）。

例

次の例では、デバイスのデバイスプロンプト名を設定しています。感嘆符（!）で示されたコメント行では、いずれのコマンドも実行されません。**hostname** コマンドを使用して、デバイス名を Device から new_name に変更しています。Ctrl-Z（^Z）キーを押すか、**end** コマンドを入力すると、コンフィギュレーションモードが終了します。**copy system:running-config nvram:startup-config** コマンドにより、現在のコンフィギュレーションがスタートアップ コンフィギュレーションに保存されます。

```
Device# configure terminal
Device(config)# !The following command provides the Device host name.
Device(config)# hostname new_name
new_name(config)# end
new_name# copy system:running-config nvram:startup-config
```

スタートアップ コンフィギュレーションが NVRAM にある場合は、スタートアップ コンフィギュレーションには現在の設定情報がコンフィギュレーション コマンドとしてテキスト形式で格納され、デフォルト以外の設定だけが記録されます。破損データから保護するために、メモリはチェックサム算出されます。

Note 一部の特定のコマンドは、NVRAM に保存されない場合があります。これらのコマンドは、マシンをリブートしたときに再入力する必要があります。これらのコマンドは、マニュアルに記載されています。リブート後にすばやくデバイスを再設定できるように、これらの設定のリストを保管しておくことを推奨します。

ルータからTFTPサーバへのコンフィギュレーションファイルのコピー

TFTP ネットワーク サーバー上の設定をコピーするには、以下の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*]
3. **copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*]

DETAILED STEPS

	Command or Action	Purpose
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	**copy system:running-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy system:running-config tftp: //server1/topdir/file10`	TFTP サーバーへ実行コンフィギュレーション ファイルをコピーします。
ステップ 3	**copy nvram:startup-config tftp:** [[[*//location*]*/directory*]*/filename*] **Example:** `Device# copy nvram:startup-config tftp: //server1/1stdir/file10`	TFTP サーバーへスタートアップ コンフィギュレーション ファイルをコピーします。

例

次に、デバイスからから TFTP サーバーへコンフィギュレーション ファイルをコピーする例を示します。

```
Tokyo# copy system:running-config tftp://172.16.2.155/tokyo-confg
Write file tokyo-confg on host 172.16.2.155? [confirm] y
Writing tokyo-confg!!! [OK]
```

次の作業

copy コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および**fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

ルータから FTP サーバへのコンフィギュレーション ファイルのコピー

ルータから FTP サーバーへスタートアップ コンフィギュレーション ファイルまたは実行コンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*

4. **ip ftp password** *password*
5. **end**
6. 次のいずれかを実行します。
   - **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]*/filename*]
   -
   -
   - **copy nvram:sta rtup-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]*/filename*]

DETAILED STEPS

	Command or Action	Purpose
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	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username user1`	（任意）デフォルトのリモート ユーザー名を指定します。
ステップ 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp username guessme`	（任意）デフォルトのパスワードを指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	（任意）グローバル コンフィギュレーション モードを終了します。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 2 および 3 を参照）。
ステップ 6	次のいずれかを実行します。 • **copy system:running-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]*/filename*] • • • **copy nvram:sta rtup-config ftp:** [[[//[*username* [**:***password*]**@**]*location/directory*]*/filename*] **Example:**	FTP サーバへ実行コンフィギュレーションまたはスタートアップ コンフィギュレーション ファイルをコピーします。

	Command or Action	Purpose
	`Device# copy system:running-config ftp: //user1:guessme@company.com /dir10/file1`	

例

FTP サーバーへの実行コンフィギュレーション ファイルの格納

次に、rtr2-confg という名前の実行コンフィギュレーション ファイルを、IP アドレスが 172.16.101.101 のリモート ホスト上の netadmin1 ディレクトリにコピーする例を示します。

```
Device# copy system:running-config ftp://netadmin1:mypass@172.16.101.101/Rtr2-confg
Write file rtr2-confg on host 172.16.101.101?[confirm]
Building configuration...[OK]
Connected to 172.16.101.101
Device#
```

FTP サーバーへのスタートアップ コンフィギュレーション ファイルの格納

次に、FTP を使用してファイルをコピーすることによって、サーバー上にスタートアップ コンフィギュレーション ファイルを格納する例を示します。

```
Rtr2# configure terminal

Rtr2(config)# ip ftp username netadmin2

Rtr2(config)# ip ftp password mypass

Rtr2(config)# end

Rtr2# copy nvram:startup-config ftp:

Remote host[]? 172.16.101.101

Name of configuration file to write [rtr2-confg]?
Write file rtr2-confg on host 172.16.101.101?[confirm]
![OK]
```

次の作業

copy EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

TFTPサーバからルータへのコンフィギュレーションファイルのコピー

TFTP サーバーからデバイスへコンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **copy tftp:** [[[*//location*]*/directory*]*/filename*] **system:running-config**
3. **copy tftp:** [[[*//location*]*/directory*]*/filename*] **nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
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	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **system:running-config** **Example:** `Device# copy tftp://server1/dir10/datasource system:running-config`	TFTP サーバーから実行コンフィギュレーションへコンフィギュレーション ファイルをコピーします。
ステップ 3	**copy tftp:** [[[*//location*]*/directory*]*/filename*] **nvram:startup-config** **Example:** `Device# copy tftp://server1/dir10/datasource nvram:startup-config`	TFTP サーバーからスタートアップ コンフィギュレーションへコンフィギュレーション ファイルをコピーします。

例

次に、IP アドレス 172.16.2.155 にある、tokyo-config という名前のファイルからソフトウェアを設定する例を示します。

```
Device1# copy tftp://172.16.2.155/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
```

次の作業

copy EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

FTP サーバーからルータへのコンフィギュレーション ファイルのコピー

FTP サーバーから実行コンフィギュレーションまたはスタートアップ コンフィギュレーションへコンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*
4. **ip ftp password** *password*
5. **end**
6. **copy ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*]**system:running-config**

DETAILED STEPS

	Command or Action	Purpose
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`	（任意）グローバル コンフィギュレーション モードを開始できます。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 2 および 3 を参照）。
ステップ 3	**ip ftp username** *username* **Example:** `Device(config)# ip ftp username user1`	（任意）デフォルトのリモート ユーザー名を指定します。
ステップ 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password guessme`	（任意）デフォルトのパスワードを指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	（任意）グローバル コンフィギュレーション モードを終了します。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 2 および 3 を参照）。

	Command or Action	Purpose
ステップ 6	**copy ftp:** [[[*//*[*username* [**:***password*]**@**]*location*]*/directory*]*/filename*]**system:running-config** **Example:** **Example:** or **Example:** **copy ftp:**[[[**//**[*username* [**:***password*]**@**]*location***/***directory*]**/***filename*] **nvram:startup-config** **Example:** Device# copy ftp://user1:guessme@company.com /dir10/datasource nvram:startup-config	FTP を使用して、ネットワーク サーバから実行メモリまたはスタートアップ コンフィギュレーションへコンフィギュレーション ファイルをコピーします。

例

FTP の Running-Config のコピー

次に、host1-confg という名前のホスト コンフィギュレーション ファイルを、IP アドレスが 172.16.101.101 のリモート サーバー上の netadmin1 ディレクトリからコピーし、デバイスでコマンドをロードして実行する例を示します。

```
Device# copy rcp://netadmin1:mypass@172.16.101.101/host1-confg system:running-config

Configure using host1-confg from 172.16.101.101? [confirm]
Connected to 172.16.101.101
Loading 1112 byte file host1-confg:![OK]
Device#
%SYS-5-CONFIG: Configured from host1-config by ftp from 172.16.101.101
```

FTP の Startup-Config のコピー

次に、リモート ユーザー名 netadmin1 を指定する例を示します。次に host2-confg という名前のコンフィギュレーション ファイルを、IP アドレスが 172.16.101.101 のリモート サーバー上の netadmin1 ディレクトリからスタートアップ コンフィギュレーションへコピーします。

```
Rtr2# configure terminal
Rtr2(config)# ip ftp username
 netadmin1
Rtr2(config)# ip ftp password
 mypass
Rtr2(config)# end
Rtr2# copy ftp: nvram:startup-config
Address of remote host [255.255.255.255]? 172.16.101.101
Name of configuration file[rtr2-confg]? host2-confg
Configure using host2-confg from 172.16.101.101?[confirm]
```

```
Connected to 172.16.101.101
Loading 1112 byte file host2-confg:![OK]
[OK]
Rtr2#
%SYS-5-CONFIG_NV:Non-volatile store configured from host2-config by ftp from 172.16.101.101
```

次の作業

copy EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

NVRAM より大きいコンフィギュレーション ファイルの保守

NVRAM のサイズを超えるコンフィギュレーション ファイルを保守するには、以降の項で説明する作業を実行します。

コンフィギュレーション ファイルの圧縮

コンフィギュレーション ファイルを圧縮するには、このセクションの手順を実行してください。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service compress-config**
4. **end**
5. 次のいずれかを実行します。
   - 新しいコンフィギュレーションをコピーするには、FTP、rcp、または TFTP を使用します。
   - **configure terminal**
6. **copy system:running-config nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
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。

	Command or Action	Purpose
ステップ 3	**service compress-config** **Example:** `Device(config)# service compress-config`	コンフィギュレーション ファイルを圧縮することを指定します。
ステップ 4	**end** **Example:** `Device(config)# end`	グローバル コンフィギュレーション モードを終了します。
ステップ 5	次のいずれかを実行します。 • 新しいコンフィギュレーションをコピーするには、FTP、rcp、または TFTP を使用します。 • **configure terminal** **Example:** `Device# configure terminal`	新しいコンフィギュレーションを入力します。 • NVRAM のサイズの 3 倍以上のコンフィギュレーションをロードしようとすると、次のエラー メッセージが表示されます。 「[buffer overflow - *file-size* /*buffer-size* bytes]。」
ステップ 6	**copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy system:running-config nvram:startup-config`	実行コンフィギュレーションの変更が終わったら、新しいコンフィギュレーションを保存します。

例

次に、129 KB のコンフィギュレーション ファイルを 11 KB に圧縮する例を示します。

```
Device# configure terminal

Device(config)# service compress-config

Device(config)# end

Device# copy tftp://172.16.2.15/tokyo-confg system:running-config

Configure using tokyo-confg from 172.16.2.155? [confirm] y

Booting tokyo-confg from 172.16.2.155:!!! [OK - 874/16000 bytes]
Device# copy system:running-config nvram:startup-config

Building configuration...
Compressing configuration from 129648 bytes to 11077 bytes
[OK]
```

パーサー キャッシュの管理

パーサーキャッシュ機能を制御するには、次の項で説明する作業を実行します。これらの作業はすべて任意です。

パーサー キャッシュのクリア

パーサーキャッシュ機能によって格納された情報をクリアするには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **clear parser cache**

DETAILED STEPS

	Command or Action	Purpose
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	**clear parser cache** **Example:** `Device# clear parser cache`	パーサー キャッシュ機能に格納されている解析キャッシュ エントリおよびヒット数とミス数の統計情報をクリアします。

パーサー キャッシュのディセーブル化

パーサー キャッシュ機能は、デフォルトでイネーブルにされています。パーサー キャッシュ機能を無効にするには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no parser cache**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
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	**no parser cache** **Example:** `Device(config)# no parser cache`	パーサー キャッシュ機能をディセーブルにします。 • パーサー キャッシュがディセーブルになると、**noparsercache** コマンド ラインが実行コンフィギュレーション ファイルに書き込まれます。 • システム リソースを解放するためにパーサー キャッシュをディセーブルにする場合は、**noparsercache** コマンドを発行する前にパーサー キャッシュをクリアする必要があります。パーサー キャッシュをディセーブルにした後は、パーサー キャッシュをクリアできません。

パーサー キャッシュの再イネーブル化

パーサー キャッシュ機能を無効にした後に再度有効にするには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parser cache**

DETAILED STEPS

	Command or Action	Purpose
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	**parser cache** **Example:** `Device(config)# parser cache`	パーサー キャッシュ機能をイネーブルにします。

次の作業

showparserstatistics コマンドにより、次の 2 セットのデータが表示されます。

- コンフィギュレーション ファイル内のコマンドのうち、最後に実行コンフィギュレーションにコピーされたコマンドの数、およびシステムがこれらのコマンドを解析するために要した時間（コンフィギュレーション ファイルはシステムの起動時または **copy***source***running-config** EXEC コマンドなどのコマンドを発行することによって実行コンフィギュレーションにロードされます）。
- パーサーキャッシュのステータス（イネーブルまたはディセーブル）、およびシステムの起動以降またはパーサーキャッシュのクリア以降に一致したコマンドの数（ヒット数またはミス数）。

フラッシュ メモリからスタートアップまたは実行コンフィギュレーションへのコンフィギュレーション ファイルのコピー

フラッシュ メモリから現在の NVRAM にあるスタートアップ コンフィギュレーションまたは実行コンフィギュレーションへコンフィギュレーション ファイルを直接コピーするには、ステップ 2 のいずれかのコマンドを入力します。

SUMMARY STEPS

1. **enable**
2. 次のいずれかを実行します。
   - **copy** *filesystem* **:** [*partition-number***:**][*filename*] **nvram:startup-config**
   - **copy** *filesystem* **:** [*partition-number***:**][*filename*] **system:running-config**

DETAILED STEPS

	Command or Action	Purpose
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	次のいずれかを実行します。 • **copy** *filesystem* **:** [*partition-number***:**][*filename*] **nvram:startup-config** • **copy** *filesystem* **:** [*partition-number***:**][*filename*] **system:running-config** **Example:** `Device# copy slot0:4:ios-upgrade-1 nvram:startup-config`	NVRAM にコンフィギュレーション ファイルを直接ロードします。 または 現在の実行コンフィギュレーションにコンフィギュレーション ファイルをコピーします。

例

次に、スロット 0 にあるフラッシュ メモリ PC カードのパーティション 4 からルータのスタートアップ コンフィギュレーションへ ios-upgrade-1 という名前のファイルをコピーする例を示します。

```
Device# copy slot0:4:ios-upgrade-1 nvram:startup-config

Copy '
ios-upgrade-1
' from flash device
  as 'startup-config' ? [yes/no] yes

[OK]
```

FTP サーバーからフラッシュ メモリ デバイスへのコンフィギュレーション ファイルのコピー

FTP サーバーからフラッシュ メモリ デバイスへコンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *username*
4. **ip ftp password** *password*
5. **end**
6. **copy ftp:** [[[//[*username*:*password*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*]

DETAILED STEPS

	Command or Action	Purpose
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`	（任意）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。この手順は、デフォルトのリモートユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 3 および 4 を参照）。
ステップ 3	**ip ftp username** *username* **Example:**	（任意）リモート ユーザー名を指定します。

	Command or Action	Purpose
	`Device(config)# ip ftp username user1`	
ステップ 4	**ip ftp password** *password* **Example:** `Device(config)# ip ftp password guessme`	（任意）リモート パスワードを指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	（任意）コンフィギュレーション モードを終了します。このステップが必要になるのは、デフォルトのリモートユーザー名を上書きする場合のみです（ステップ 3 および 4 を参照）。
ステップ 6	**copy ftp:** [[[//[*username*:*password*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*] **Example:** `Device> copy ftp:router-config slot0:new-config`	FTP を使用してネットワーク サーバからフラッシュ メモリ デバイスへコンフィギュレーション ファイルをコピーします。

次の作業

copy EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

rcp サーバからフラッシュ メモリ デバイスへのコンフィギュレーション ファイルのコピー

rcp サーバーからフラッシュ メモリ デバイスへコンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-username** *username*
4. **end**
5. **copy rcp:** [[[//[*username*@]*location*]/*directory*]/*filename*]
   flash-filesystem:[*partition-number*:][*filename*]

DETAILED STEPS

	Command or Action	Purpose
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`	（任意）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。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 3 および 4 を参照）。
ステップ 3	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username user1`	（任意）リモート ユーザー名を指定します。
ステップ 4	**end** **Example:** `Device(config)# end`	（任意）コンフィギュレーション モードを終了します。この手順は、デフォルトのリモート ユーザ名またはパスワードを上書きする場合にだけ必要です（ステップ 3 および 4 を参照）。
ステップ 5	**copy rcp:** [[[//[*username*@]*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*] **Example:** `Device# copy rcp:router-config slot0:new-config`	rcp を使用してネットワーク サーバからフラッシュ メモリ デバイスへコンフィギュレーション ファイルをコピーします。追加情報または確認を要求するルータからのプロンプトに対し応答します。このプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** コマンドの現在の設定によって異なります。

TFTP サーバーからフラッシュ メモリ デバイスへのコンフィギュレーション ファイルのコピー

TFTP サーバからフラッシュ メモリ デバイスへコンフィギュレーション ファイルをコピーするには、以下のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **copy tftp:** [[[//*location*]/*directory*]/*filename*] *flash-filesystem*:[*partition-number*:][*filename*]

DETAILED STEPS

	Command or Action	Purpose
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	**copy tftp:** [[[*//location*]*/directory*]*/filename*] *flash-filesystem***:**[*partition-number***:**][*filename*] **Example:** `Device# copy tftp:router-config slot0:new-config`	TFTP サーバからフラッシュ メモリ デバイスへファイルをコピーします。追加情報または確認を要求するデバイスからのプロンプトに対し応答します。このプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** コマンドの現在の設定によって異なります。

例

次の例は、TFTP サーバから Cisco 7500 シリーズ デバイスのネットワーク処理エンジン（NPE）またはルート スイッチ プロセッサ（RSP）カードのスロット 0 に挿入されたフラッシュ メモリ カードへDevice-config という名前のコンフィギュレーション ファイルをコピーする例を示します。コピーされたファイルの名前は new-config に変更されます。

```
Device# copy tftp:router-config slot0:new-config
```

スタートアップ コンフィギュレーション ファイルでのコンフィギュレーション コマンドの再実行

スタートアップ コンフィギュレーション ファイルのコマンドを再実行するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure memory**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
ステップ 2	**configure memory** **Example:** `Device# configure memory`	スタートアップ コンフィギュレーション ファイルにあるコンフィギュレーション コマンドを再実行します。

スタートアップ コンフィギュレーションのクリア

スタートアップ コンフィギュレーションから設定情報を消去できます。スタートアップ コンフィギュレーションなしでルータをリブートした場合は、ルータを最初から設定できるように、ルータは Setup コマンド ファシリティに移行します。スタートアップ コンフィギュレーションの内容を消去するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **erase nvram**

DETAILED STEPS

	Command or Action	Purpose
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	**erase nvram** **Example:**	スタートアップ コンフィギュレーションの内容をクリアします。

	Command or Action	Purpose
	`Device# erase nvram`	**Note** クラス A フラッシュ ファイル システムのプラットフォーム以外のすべてのプラットフォームでは、このコマンドにより NVRAM が消去されます。スタートアップ コンフィギュレーション ファイルは、いったん削除すると復元できません。クラス A フラッシュ ファイル システムのプラットフォーム上では、**erasestartup-config**EXEC コマンドを使用すると、CONFIG_FILE 環境変数により指定されたコンフィギュレーションが、デバイスにより削除されます。この変数が NVRAM を指定している場合は、デバイスにより NVRAM が消去されます。CONFIG_FILE 環境変数がフラッシュ メモリ デバイスとコンフィギュレーション ファイル名を指定している場合は、デバイスによりコンフィギュレーション ファイルが削除されます。つまり、そのコンフィギュレーション ファイルはデバイスにより消去されるのではなく、「削除済み」としてマークされます。この機能では、削除されたファイルを回復できます。

指定されたコンフィギュレーション ファイルの削除

特定のフラッシュ デバイスの指定された設定を削除するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **delete** *flash-filesystem* **:** *filename*

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
ステップ 2	**delete** *flash-filesystem* **:** *filename* **Example:** `Device# delete slot0:myconfig`	指定されたフラッシュ デバイス上の指定されたコンフィギュレーション ファイルを削除します。 **Note** クラス A および B フラッシュ ファイル システムでは、フラッシュ メモリ内の特定のファイルを削除すると、そのファイルは削除済みとしてシステムによりマークされます。これにより、**undelete** EXEC コマンドを使用して、削除したファイルを後で回復できるようになります。消去されたファイルは回復できません。コンフィギュレーション ファイルを完全に消去するには、**squeeze** EXEC コマンドを使用します。クラス C フラッシュ ファイル システムでは、削除されたファイルは回復できません。 CONFIG_FILE 環境変数で指定されたコンフィギュレーション ファイルを消去または削除しようとした場合、システムにより削除の確認を求めるプロンプトが表示されます。

CHAPTER 15

コンフィギュレーション生成のパフォーマンス拡張

コンフィギュレーション生成のパフォーマンス拡張機能は、実行中のコンフィギュレーションファイル情報の収集を高速化することでコンフィギュレーション管理を支援します。この機能は、多数のインターフェイスが構成された大規模なネットワークを管理する場合に特に便利です。

コンフィギュレーション生成のパフォーマンス拡張に関する制限事項

コンフィギュレーション生成のパフォーマンス拡張機能を使用するデバイスには、大規模インターフェイス コンフィギュレーション ファイルを保存（キャッシュ保存）するための十分なメモリが必要です。たとえば、インターフェイスのコンフィギュレーションが15 KBのメモリを占有する場合、この機能を使用するには、さらに15 KBのメモリ領域を使用できる必要があります。

コンフィギュレーション生成のパフォーマンス拡張について

Cisco IOS XE ソフトウェアのコンフィギュレーション ストレージ

Cisco IOS XE のソフトウェア コンフィギュレーション モデルでは、コンフィギュレーション状態は分散して維持され、各コンポーネントは独自のコンフィギュレーション状態を保持します。設定情報を取得するには、ソフトウェアは各コンポーネントをポーリングして、分散された情報を収集する必要があります。このコンフィギュレーション状態の取得操作は、不揮発生成（NVGEN）と呼ばれるプロセスによって実行され、実行中のシステム構成を表示またはコピーするためにコマンドライン インターフェイス（CLI）コマンド（**show running-configuration**、**write memory**、**copy system:running-configuration** など）で使用されます。NVGEN は、呼び出されると、各システムコンポーネントと、インターフェイスまたはその他の構成オブジェクトの各インスタンスを照会します。NVGEN がこれらのクエリーを実行しているシステムを通過するときに、実行コンフィギュレーション ファイルが作成されます。

コンフィギュレーション生成のパフォーマンス強化の利点

コンフィギュレーション生成のパフォーマンス強化機能が導入される前は、NVGEN は常にシステム全体を照会する必要があり、全体のコンフィギュレーションしか生成できませんでした。NVGEN 操作の完了には数分かかることがあるため、実行コンフィギュレーションの処理に必要な時間が原因となり、コンフィギュレーション管理上のパフォーマンスの問題が生じます。

コンフィギュレーション生成のパフォーマンス拡張機能はNVGEN処理の実行時間を短縮し、特に多数のインターフェイス コンフィギュレーションを含む大規模なコンフィギュレーション ファイルの管理で有用です。この機能はシステム メモリのインターフェイス コンフィギュレーション情報をキャッシュに保存し、変更された設定情報だけを取得することで、実行中のシステム構成を処理するコマンドの実行を高速化します。

コンフィギュレーション生成のパフォーマンス強化の設定方法

コンフィギュレーション生成のパフォーマンス強化の設定

コンフィギュレーション生成のパフォーマンス拡張をイネーブルにする作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parser config cache interface**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**parser config cache interface** **Example:** `Device(config)# parser config cache interface`	特に大規模コンフィギュレーション ファイルの場合に、実行中のシステム構成を管理するコマンドを CLI で実行するのに要する時間を短縮します。
ステップ 4	**end** **Example:** `Device(config)# end`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

コンフィギュレーション生成のパフォーマンス強化の設定例

コンフィギュレーション生成のパフォーマンス強化の設定例

次の例は、コンフィギュレーション生成のパフォーマンス強化機能を有効にする方法を示しています。

```
Device(config)# parser config cache interface
```

コンフィギュレーション生成のパフォーマンス強化の確認例

システム コンフィギュレーション ファイルのコマンドをチェックして、**parserconfigcacheinterface** コマンドがイネーブルになっていることを確認できます。これは **showrunning-configuration** EXEC コマンドを入力すると表示されます。

Note 初めてコンフィギュレーション ファイルを表示する場合は、インターフェイス キャッシュが少ないため、それほどパフォーマンスの改善は見られません。ただし、**showrunning-config** EXEC コマンドなどの後続の NVGEN タイプのコマンドを入力すると、パフォーマンスが向上することがわかります。インターフェイスの構成が変更されるたびに、指定したインターフェイスのキャッシュがフラッシュされます。その他のインターフェイスデータはそのままキャッシュに残ります。インターフェイス コンフィギュレーションの修正後にNVGENタイプのコマンドを入力すると、次回のNVGENタイプのコマンドが入力されるまで改善はほとんど見られません。

```
Device# show running-config
!
!
parser config cache interface
!
!
```

その他の参考資料

次の項に、コンフィギュレーション パーティショニング機能に関する参考資料を示します。

関連資料

関連項目	マニュアル タイトル
実行コンフィギュレーションのパフォーマンス強化：インターフェイスの **parserconfigcache**	コンフィギュレーション生成のパフォーマンス拡張
カスタマー サービスのプロビジョニング、コンフィギュレーション ロールバック、コンフィギュレーション ロッキング、およびコンフィギュレーション アクセス コントロール	コンフィギュレーションのコンテキスト差分ユーティリティ
コンフィギュレーション管理：コンフィギュレーション変更およびロギング	コンフィギュレーション変更通知およびロギング
コンフィギュレーション管理：コンフィギュレーション変更およびロギングのクイック保存：[1].	コンフィギュレーション ロガー永続性

関連項目	マニュアル タイトル
Cisco IOS ソフトウェア コンフィギュレーション アクセス制御およびコンフィギュレーション セッション ロック（「Config ロック」）。	排他的設定変更アクセスとアクセス セッション ロック

[1] 「コンフィギュレーション ロガー永続性」機能により、スタートアップ コンフィギュレーション全体を保存するのではなく、最後の startup-config ファイルが生成された時点から入力されたコマンドだけを保存できます。

標準

標準	タイトル
この機能に関連付けられている規格はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	--

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/techsupport

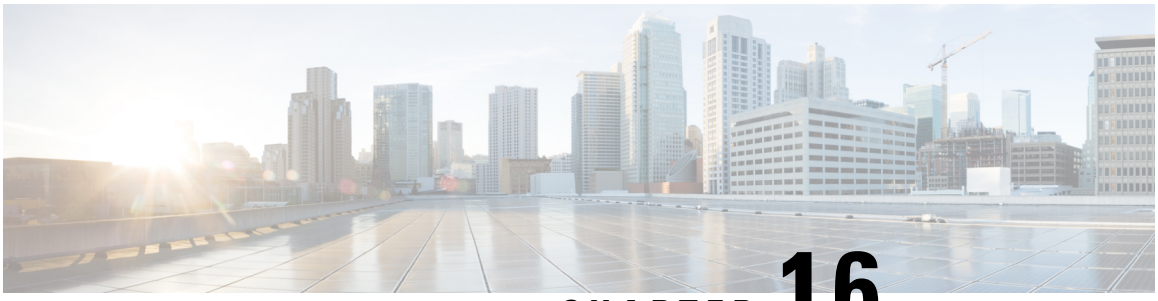
コンフィギュレーション生成のパフォーマンス強化の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 20: コンフィギュレーション生成のパフォーマンス強化機能の機能情報

機能名	リリース	機能情報
コンフィギュレーション生成のパフォーマンス拡張		コンフィギュレーション生成のパフォーマンス拡張機能は、実行中のコンフィギュレーション ファイル情報の収集を高速化することでコンフィギュレーション管理を支援します。この機能は、多数のインターフェイスが構成された大規模なネットワークを管理する場合に特に便利です。 この機能に関連付けられたコマンド： • **parser config cache interface** • **parser config partition** • **parser cache**



CHAPTER 16

排他的設定変更アクセスとアクセス セッション ロック

排他的設定変更アクセス機能（「コンフィギュレーション ロック」機能とも呼びます）を使用すると、Cisco IOS XE の実行コンフィギュレーションに排他的に変更アクセスし、複数のユーザが同時に設定を変更するのを防ぐことができます。

この機能に対してアクセス セッション ロッキングを追加することで、排他的設定変更アクセス機能が拡張され、設定ロックを保持しているユーザが実行する **show** コマンドと **debug** コマンドの実行が常に優先されるようになります。他のユーザによって入力される **show** コマンドと **debug** コマンドは、設定ロックの所有者が開始したプロセスが終了した後でしか実行を許可されません。

排他的設定変更アクセス機能（「コンフィギュレーション ロック」）は、コンフィギュレーションの置換とロールバック機能（「ロールバック ロック」）を補完するロック機構です。

設定のロックについて

排他的設定変更アクセスとアクセス セッション ロック

Cisco IOS ソフトウェアが動作するデバイスは、デバイスのコンフィギュレーション状態を決定する実行コンフィギュレーションを保持しています。実行コンフィギュレーションを変更すると、デバイスの動作が変わります。Cisco IOS ソフトウェアでは、複数のユーザがデバイス CLI（デバイス コンソール、telnet セキュア シェル（SSH）など）を介して実行コンフィギュレーションを変更することが可能です。そのため運用環境によっては、複数のユーザが同時に Cisco IOS の実行コンフィギュレーションに変更を加えるのを防ぐと役立ちます。Cisco IOS の実行コンフィギュレーションへのアクセスを一時的に制限することにより、不注意による競合

や、2 人のユーザが実行コンフィギュレーションの同じ部分を設定しようとするのを防ぐことができます。

排他的設定変更アクセス機能（「コンフィギュレーション ロック」機能とも呼びます）を使用すると、Cisco IOS の実行コンフィギュレーションに排他的に変更アクセスし、複数のユーザが同時に設定を変更するのを防ぐことができます。

この機能により、**configure terminal** コマンドを使用してグローバル コンフィギュレーション モードを開始した時点から、Cisco IOS の実行コンフィギュレーションへの排他的な変更アクセスが提供されます。これにより、「コンフィギュレーション ロック」の効果が得られ、他のユーザによる Cisco IOS の実行コンフィギュレーションの変更を防止できます。コンフィギュレーション ロックは、Cisco IOS コンフィギュレーション モードを終了すると自動的に解除されます。

排他的設定変更アクセス機能をイネーブルにするには、グローバル コンフィギュレーション モードで **configuration mode exclusive** コマンドを使用します。排他的設定変更アクセスは **auto** に設定できます。その場合は、他のユーザが **configure terminal** コマンドを使用するたびに Cisco IOS のコンフィギュレーション モードがロックされます。または **manual** に設定すると、**configure terminal lock** コマンドが発行されたときのみ Cisco IOS のコンフィギュレーション モードがロックされます。

排他的設定変更アクセス機能は、Cisco IOS リリース 12.2(25)S および 12.3(7)T で導入されたコンフィギュレーションの置換とロールバック機能を補完するロック機構です。

アクセス セッション ロック

アクセス セッション ロッキング機能は、設定のロックを保持しているユーザが入力した **show** コマンドと **debug** コマンドの実行が常に優先されるように、排他的設定変更アクセス機能を拡張します。この機能は、同時設定アクセスを防ぐとともに、別のユーザが入力した **show** コマンドのように、他のコンフィギュレーション コマンドの実行中に同時に処理が実行されるのを防ぐためのオプションも提供します。この機能をイネーブルにすると、設定ロックを保持しているユーザが入力したコマンド（コンフィギュレーション コマンドなど）が、他のユーザが入力したコマンドよりも常に優先されます。

排他的設定変更アクセスとアクセスセッションロックの設定方法

排他的設定変更アクセスとアクセス セッション ロックの有効化

Note Cisco IOS リリース 12.2(33)SRE から、排他的設定変更アクセスおよびアクセス セッション ロッキング機能は、Cisco IOS ソフトウェアで使用できなくなりました。この機能の代わりに、パーサーの並行処理およびロッキングの改善機能を使用してください。詳細については、「パーサーの同時実行とロックの改善の有効化」を参照してください。

排他的設定変更アクセスとアクセス セッション ロック機能を有効にするには、次のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **configuration mode exclusive**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**configuration mode exclusive** **Example:** `Router(config)# configuration mode exclusive`	排他的設定変更アクセス（設定ロック機能）をイネーブルにします。 • コマンドがイネーブルになると、コンフィギュレーション セッションがシングルユーザ（排他）モードで実行されます。

	Command or Action	Purpose
ステップ 4	**end** **Example:** `Router(config)# end`	コンフィギュレーション セッションを終了して、CLI を特権 EXEC モードに戻します。

排他的設定変更アクセスの取得

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **configure terminal lock**
4. 変更を実行コンフィギュレーションに入力してシステムを設定します。
5. 次のいずれかを実行します。
   - **end**
   - または
   - **exit**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**configure terminal lock** **Example:** `Router(config)# configure terminal lock`	（任意）Cisco IOS ソフトウェアを排他（シングルユーザ）モードでロックします。 • このコマンドは、**configuration mode exclusive** コマンドを使用して設定ロックをイネーブルにしてある場合にだけ使用できます。 • このコマンドは、Cisco IOS リリース 12.3(14)T 以降のリリースで使用できます。
ステップ 4	変更を実行コンフィギュレーションに入力してシステムを設定します。	--

	Command or Action	Purpose

ステップ 5	次のいずれかを実行します。
	• **end**
	• または
	• **exit**
	Example:
	`Router(config)# end`
	Example:
	Example:
	`Router(config)# exit`
		コンフィギュレーション セッションを終了し、ステップ 1 で取得したセッション ロックを解放し、特権 EXEC モードに戻ります。
		Note **end** コマンド、**exit** コマンド、Ctrl+Z のキーの組み合わせのいずれかで設定ロックを解放します。**end** コマンドの使用をお勧めします。

設定ロックのモニタリングとトラブルシューティング

排他的設定変更アクセスおよびアクセス セッション ロッキング機能をモニタリングまたはトラブルシューティングするには、この作業のいずれかの手順または両方の手順を実行します。

SUMMARY STEPS

1. **show configuration lock**
2. **debug configuration lock**

DETAILED STEPS

ステップ 1 **show configuration lock**

現在の設定ロックのステータスと詳細（所有者、ユーザー、端末、ロック状態、ロック クラスなど）を表示するには、次のコマンドを使用します。

グローバル コンフィギュレーション モードを開始できない場合は、このコマンドを使用して、コンフィギュレーション セッションが別のユーザによってロックされているかどうか、およびそのユーザーが誰なのかを調べることができます。

Example:

ステップ 2 **debug configuration lock**

Cisco IOS 設定ロックのデバッグをイネーブルにするには、このコマンドを使用します（公開クラス ロックまたはロールバック クラス ロック）。

Example:

```
Router# debug configuration lock

Session1 from console
```

```
==========================
Router# configure terminal lock
Configuration mode locked exclusively. The lock will be cleared once you exit out of configuration
 mode using end/exit
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Parser : LOCK REQUEST in EXCLUSIVE mode
Parser: <configure terminal lock> - Config. Lock requested by process <3> client <PARSER Client>
Parser: <configure terminal lock> - Config. Lock acquired successfully !
Router(config)#
```

コンフィギュレーションのロックの設定例

自動モードでの排他的ロックの設定例

次に、**configurationmodeexclusive** コマンドを使用し、シングルユーザ自動コンフィギュレーション モードに対して、自動モードで排他的ロックをイネーブルにする例を示します。Cisco IOS コンフィギュレーション ファイルが排他的にロックされたら、**showconfigurationlock** コマンドを使用してこのコンフィギュレーションを確認できます。

```
Router# configure terminal
Router(config)#
Router(config)# exit
Router# configure terminal
! Locks configuration mode exclusively.
Router# show configuration lock
Parser Configure Lock
Owner PID        :  10
User             :  User1
TTY              :  3
Type             :  EXCLUSIVE
State            :  LOCKED
Class            :  Exposed
Count            :  0
Pending Requests :  0
User debug info  :  0
```

手動モードでの排他的ロックの設定例

その他の参考資料

ここでは、設定のロックに関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
コンフィギュレーション ファイルを管理するためのコマンド	『*Cisco IOS Configuration Management Command Reference*』
コンフィギュレーション ファイルの管理についての情報	コンフィギュレーション ファイルの管理

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存MIBのサポートに変更はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

排他的設定変更アクセスとアクセスセッションロックの機能情報

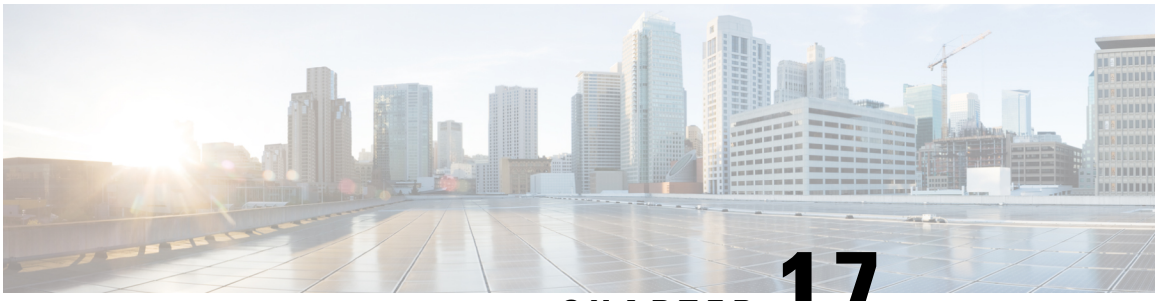
次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 21: 排他的設定変更アクセスとアクセス セッション ロックの機能情報

機能名	リリース	機能情報
排他的設定変更アクセスとアクセスセッションロック	12.3(14)T 12.0(31)S 12.2(33)SRA 12.4(11)T 12.2(33)SXH 12.2(33)SB	排他的設定変更アクセス機能（「コンフィギュレーション ロック」機能とも呼びます）を使用すると、Cisco IOS の実行コンフィギュレーションに排他的に変更アクセスし、複数のユーザーが同時に設定を変更するのを防ぐことができます。 この機能に対してアクセス セッション ロッキングを追加することで、排他的設定変更アクセス機能が拡張され、設定ロックを保持しているユーザが実行する **show** コマンドと **debug** コマンドの実行が常に優先されるようになります。他のユーザによって入力される **show** コマンドと **debug** コマンドは、設定ロックの所有者が開始したプロセスが終了した後でしか実行を許可されません。 排他的設定変更アクセス機能は、コンフィギュレーションの置換とロールバック機能（「ロールバック ロック」）を補完するロック機構です。 設定ロック機能はリリース 12.0S に統合され、アクセス セッション ロック機能の拡張が実装されました。**configuration mode exclusive** コマンドが、キーワード オプション **config_wait**、**expire**、**interleave**、**lock-show**、**retry_wait**、および **terminate** を含むように拡張されました。**show configuration lock** コマンドの出力が改良されました。 拡張機能は、リリース 12.2(33)SRA、12.4(11)T、12.2(33)SXH、および 12.2(33)SB に統合されました。 この機能に関する詳細については、次の各項を参照してください。 • 設定のロックについて • 設定ロックの設定方法 次のコマンドが導入または変更されました：**clear configuration lock**、**configuration mode exclusive**、**configure terminal lock**。

機能名	リリース	機能情報
パーサーの並行処理およびロッキングの改善	12.2(33)SRE 15.1(1)T	パーサーの並行処理およびロッキングの改善機能は、要求したプロセスに対して排他的なアクセスを許可し、他のプロセスがCisco IOS の設定に同時にアクセスできないようにするための、共通のインターフェイスを提供します。この機能により、ロックを保持しているユーザーにのみアクセスが許可され、他のクライアントがコンフィギュレーションにアクセスできなくなります。 この機能に関する詳細については、次の各項を参照してください。 • パーサーの並行処理およびロッキングの改善 • パーサーの並行処理およびロッキングの改善のイネーブル化 次のコマンドが導入または変更されました：**parser command serializer**、**test parser session-lock**。



CHAPTER 17

コンフィギュレーションの置換とロールバック

コンフィギュレーションの置換とロールバック機能により、現在の実行コンフィギュレーションを、保存しておいた Cisco IOS コンフィギュレーション ファイルで置換することができます。この機能は、コンフィギュレーションを保存しておいた状態へ戻すために使用でき、そのコンフィギュレーションファイルが保存された後にどのような変更が加えられても、効果的にロールバックさせることができます。

コンフィギュレーションの置換とロールバックの前提条件

コンフィギュレーションの置換とロールバックの機能に対する入力となるコンフィギュレーション ファイルの形式は、標準の Cisco ソフトウェア コンフィギュレーション ファイルの、次に示すインデント規則に準拠している必要があります。

- 新しい行のすべてのコマンドは、コマンドがコンフィギュレーションサブモードにない限り、インデントなしで開始します。
- レベル 1 コンフィギュレーション サブモード内のコマンドは、スペース 1 個分インデントします。
- レベル 2 コンフィギュレーション サブモード内のコマンドは、スペース 2 個分インデントします。

- 以下、続くサブモード内のコマンドは、同じようにインデントします。

これらのインデント規則には、ソフトウェアが **show running-config** や **copy running-config** *destination-url* などのコマンドのコンフィギュレーション ファイルを作成する方法が記述されています。シスコ デバイスで生成されるコンフィギュレーション ファイルは、いずれもこうした規則に従います。

2つのコンフィギュレーションファイル（現在の実行コンフィギュレーションと、保存された置換用コンフィギュレーション）を合わせたサイズより大きな空きメモリが必要です。

コンフィギュレーションの置換とロールバックの制約事項

デバイスに、2 つのコンフィギュレーション ファイル（現在の実行コンフィギュレーションと、保存された置換用コンフィギュレーション）を合わせたサイズより大きな空きメモリがない場合、コンフィギュレーション置換操作は実行されません。

ネットワーク デバイスの物理コンポーネント（物理インターフェイスなど）に関連する特定の Cisco コンフィギュレーション コマンドは、実行コンフィギュレーションについて追加または削除することはできません。たとえば、コンフィギュレーション置換操作を行っても、そのインターフェイスがデバイス上に物理的に存在する場合、現在の実行コンフィギュレーションから **interface ethernet 0** コマンド行を削除することはできません。同様に、**interface ethernet 1** コマンド行は、そのようなインターフェイスがデバイス上に物理的に存在しない場合、実行コンフィギュレーションに追加することはできません。コンフィギュレーション置換操作でこのタイプの変更を試行すると、その特定のコマンド行が失敗したことを示すエラー メッセージが表示されます。

非常にまれなケースですが、ルータをリロードしないと特定の Cisco コンフィギュレーション コマンドを実行コンフィギュレーションから削除できないことがあります。コンフィギュレーション置換操作でこのタイプのコマンドの削除を試行すると、その特定のコマンド行が失敗したことを示すエラー メッセージが表示されます。

署名証明書の検証に公開キーインフラストラクチャ（PKI）を使用する場合、**copy startup-config running-config** および **configure replace** コマンドはサポートされません。別のファイルから設定手順を置換またはロードする場合は、デバイスのリロードが必要です。

このタスクを実行するには、次の手順を実行します。

- **ステップ 1**：実行コンフィギュレーション ファイルのバックアップファイルを作成します。実行コンフィギュレーションファイルをスタートアップコンフィギュレーションファイルにコピーします。

```
Router#copy startup-config running-config
```

- **ステップ 2**：バックアップファイルから設定を復元します。スタートアップ コンフィギュレーション ファイルを実行コンフィギュレーション ファイルにコピーします。

```
Router#copy running-config startup-config
```

- **ステップ 3**：PKI 証明書を削除します。

```
Router#no crypto pki trustpoint trustpoint-name

% □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
 received from the related Certificate Authority.

Are you sure you want to do this? [yes/no]: yes
% Be sure to ask the CA administrator to revoke your certificates.
```

- **ステップ 4**：証明書を再度インポートします。

Note **configure replace** コマンドを発行して、現在の実行コンフィギュレーションを保存されている Cisco IOS コンフィギュレーション ファイルに置き換えると、CLI は、コンフィギュレーションを保持するためにコマンドを発行した後に、デバイスをリロードするように求めます。

copy startup-config running-config コマンドを使用して実行コンフィギュレーションをスタートアップ コンフィギュレーションにコピーする場合、CLIはコンフィギュレーションの変更を有効にするためにデバイスをリロードするように求めます。

コンフィギュレーションの置換とロールバックについて

コンフィギュレーション アーカイブ

Cisco IOS コンフィギュレーション アーカイブは、**configurereplace** コマンドにより提供されるコンフィギュレーションのロールバック機能を強化するために、Cisco IOS コンフィギュレーションファイルのアーカイブの保存、整理、管理を行うことを目的としたメカニズムです。この機能の導入前にも、実行コンフィギュレーションのコピーを **copyrunning-config***destination-url* コマンドを使用して保存し、ローカルやリモートに置換ファイルを保管できました。ただし、この方法ではファイルの自動管理を行うことはできませんでした。一方、コンフィギュレーションの置換とロールバック機能では、実行コンフィギュレーションファイルを自動的にCisco IOS コンフィギュレーション アーカイブに保存できます。アーカイブされたファイルはコンフィギュレーションのチェックポイントとして参照することができ、**configurereplace** コマンドを使用して以前のコンフィギュレーション状態に戻すために利用できます。

archiveconfig コマンドを使用すると、Cisco IOS コンフィギュレーションをコンフィギュレーションアーカイブに保存できます。その場合、標準のディレクトリとファイル名のプレフィクスが使用され、バージョン番号（およびオプションでタイムスタンプ）が自動的に付加されます。バージョン番号は連続したファイルを保存するごとに、1 つずつ大きくなります。この機能により、保存した Cisco IOS コンフィギュレーション ファイルを一貫して識別できます。アーカイブに保存する実行コンフィギュレーションの数は指定することができます。アーカイ

ブ内のファイル数が上限値に達すると、次に最新のファイルが保存されるときに、最も古いファイルが自動的に消去されます。**showarchive** コマンドを使用すると、Cisco IOS コンフィギュレーション アーカイブに保存されているすべてのコンフィギュレーション ファイルに関する情報が表示されます。

コンフィギュレーション ファイルを保存する Cisco IOS コンフィギュレーション アーカイブは、**configurereplace** コマンドで使用することによって、次のファイル システムに配置できます。

- お使いのプラットフォームが disk0--disk 0: disk1: ftp: pram: rcp: slavedisk0: slavedisk1: または tftp:
- disk0 がないプラットフォーム：ftp:、http:、pram:、rcp:、tftp:。

コンフィギュレーションの置換

configurereplace コマンドにより、現在の実行コンフィギュレーションを、保存しておいた Cisco IOS コンフィギュレーション ファイルで置換することができます。この機能は、コンフィギュレーションを保存しておいた状態へ戻すために使用することができ、そのコンフィギュレーション状態が保存された後にどのような変更が加えられても、効果的にロールバックさせることができます。

configurereplace コマンドを使用するときは、現在の実行コンフィギュレーションと置換するための、保存された Cisco IOS コンフィギュレーション ファイルを指定する必要があります。置換ファイルは、Cisco IOS デバイスによって作成された完全なコンフィギュレーション（**copyrunning-config***destination-url* コマンドによって作成されたものなど）であることが必要です。あるいは、置換ファイルを外部的に作成する場合は Cisco IOS デバイスが作成するファイル形式に完全に準拠していなければなりません。**configurereplace** コマンドを入力すると、現在の実行コンフィギュレーションが指定された置換コンフィギュレーションと比較され、一連の diff が生成されます。2 つのファイルの比較に使用されるアルゴリズムは、**showarchiveconfigdifferences** コマンドで使用されるものと同じです。置換コンフィギュレーションの状態になるよう、diff の結果が Cisco IOS パーサーによって適用されます。diff のみが適用されるため、現在の実行コンフィギュレーション上にすでに存在していた設定コマンドを再適用することにより生じる、潜在的なサービスの中断を避けられます。このアルゴリズムでは、順序に依存するコマンド（アクセス リストなど）へのコンフィギュレーション変更を、複数のパス プロセスを通して効果的に実行します。通常の環境では、コンフィギュレーション置換操作の完了に必要なパスは 3 つまでであり、ループ動作を防ぐためのパスは最大 5 つまでに制限されます。

Cisco IOS **copy***source-url***running-config** コマンドは、保存された Cisco IOS コンフィギュレーション ファイルを実行コンフィギュレーションへコピーするためによく使用されます。**copy***source-url***running-config** コマンドを **configurereplace***target-url* コマンドの代わりに使用する場合、主な相違点として次の点に注意が必要です。

- **copy***source-url***running-config** コマンドはマージ動作であり、ソース ファイルと現在の実行コンフィギュレーションの両方のコマンドをすべて保持します。このコマンドでは、現在の実行コンフィギュレーションにのみ含まれ、ソース ファイルには存在しないコマンドが削除されることはありません。これに対して、**configurereplace***target-url* コマンドでは、

置換ファイルに存在しないコマンドが現在の実行コンフィギュレーションから削除され、追加する必要のあるコマンドが現在の実行コンフィギュレーションに追加されます。

- **copy***source-url***running-config** コマンドでは、現在の実行コンフィギュレーションにすでに存在しているかどうかにかかわらず、ソースファイル中のすべてのコマンドが適用されます。このアルゴリズムは効率的でない上、場合によってはサービスの停止が発生します。これに対して、**configurereplace***target-url* コマンドでは適用が必要なコマンドのみを適用し、現在の実行コンフィギュレーションに存在しているコマンドは再適用されません。
- **copy***source-url***running-config** コマンドでは部分的なコンフィギュレーション ファイルもコピー元として使用できますが、**configurereplace***target-url* コマンドの置換ファイルとして使用できるのは、完全な Cisco IOS コンフィギュレーション ファイルのみです。

Cisco IOS リリース 12.2(25)S および 12.3(14)T では、コンフィギュレーション置換操作にロック機能が導入されました。**configurereplace** コマンドが使用されると、コンフィギュレーション置換の動作中、デフォルトで実行コンフィギュレーション ファイルがロックされます。このロック メカニズムによって、置換動作の実行中に他のユーザーが実行コンフィギュレーションを変更しようとしたために、置換動作の不正終了が発生することを防止できます。**nolock** キーワードを **configurereplace** コマンドの実行時に使用すると、実行コンフィギュレーションのロックをディセーブルにできます。

実行コンフィギュレーションのロックは、コンフィギュレーションの置換動作終了時に自動的にクリアされます。**showconfigurationlock** コマンドを使用すると、現在実行コンフィギュレーションに適用されているロックをすべて表示できます。

Note IOS から供給されていないコンフィギュレーション（カスタムに記述したコンフィギュレーションなど）を使用してコンフィギュレーションを置換するシナリオでは、ログインバナーに EXT 文字（ASCII コード 003）ではない区切り記号がある場合、バナーのコンフィギュレーションは拒否され、置換後のコンフィギュレーションには含まれません。正常に動作しない区切り文字には、^C、%、#、CC などがあります。

コンフィギュレーション ロールバック

ロールバックの概念は、データベースの操作ではトランザクション プロセス モデルに由来します。データベース トランザクションでは、あるデータベースのテーブルに一連の変更を加えることがあります。その後、変更を実行する（変更を恒久的に適用する）か、変更をロールバックする（変更を破棄してテーブルを以前の状態に戻す）かを選択することになります。ここでロールバックが意味するのは、変更のログを含んだジャーナル ファイルが破棄され、何の変更も加えられないということです。ロールバック操作の結果として、加えた変更が適用される前の状態に戻ります。

configurereplace コマンドを使用することで、以前のコンフィギュレーション状態へ戻ることが可能になり、コンフィギュレーション状態の保存後に加えた変更を効率的にロールバックさせることができます。Cisco IOS コンフィギュレーション ロールバックは、適用された一連の変更をもとにロールバック動作を行うのではなく、保存された Cisco コンフィギュレーション

ファイルに基づいた特定のコンフィギュレーション状態へ戻るというコンセプトを採用しています。このコンセプトは、チェックポイント（データベースの保存されたバージョン）に特定の状態を保存しておくという、データベースの考え方に類似しています。

コンフィギュレーションのロールバック機能が必要な場合、コンフィギュレーションの変更に先立って Cisco IOS 実行コンフィギュレーションを保存する必要があります。次に、コンフィギュレーションを変更した後に（**configurereplace** *target-url* コマンドを使用し）保存したコンフィギュレーション ファイルを使って変更をロール バックします。さらに、保存された Cisco IOS コンフィギュレーション ファイルならどれでも置換コンフィギュレーションとして指定できるため、ジャーナル ファイルによるロールバック モデルの一部のように、ロールバックの数が制限されることもありません。

コンフィギュレーション ロールバック変更確認の操作

コンフィギュレーション ロールバック変更確認機能は、コンフィギュレーションの変更に確認条件を追加できる機能です。この機能により、要求された変更の確認が設定済みの時間枠以内に受信されない場合にロールバックを行うことができます。コマンドの失敗を、コンフィギュレーションのロールバックをトリガーするように設定することもできます。

次に、このプロセスを実施するための手順の概要を示します。

1. 新しいオプションを使用すると、コンフィギュレーションの変更の確認を要求できます（確認の時間制限を指定する必要があります）。
2. 確認コマンドを入力する必要があります。要求された制限時間内に確認を入力しないと、コンフィギュレーションは以前の状態に戻ります。

コンフィギュレーションの置換とロールバックの利点

- コンフィギュレーションの変更を効率的にロールバックさせて、以前のコンフィギュレーション状態へ戻ることが可能。
- 現在の実行コンフィギュレーション ファイルをスタートアップ コンフィギュレーション ファイルに置き換えることができます。ファイルを置き換えた後、設定の変更を有効にするためにデバイスをリロードする必要があります。
- 保存しておいたどの Cisco IOS コンフィギュレーション状態に戻すことも可能。
- 追加や削除が必要なコマンドだけが影響される場合、ルータに完全なコンフィギュレーション ファイルを適用することができるため、コンフィギュレーションの変更がシンプルに。
- **configure replace** コマンドを **copy** *source-url* **running-config** コマンドの代用として使用すると、現在の実行コンフィギュレーションにある既存のコマンドが再度適用されないため、効率が向上し、サービス停止のリスクが回避されます。ファイルを置き換えた後、設定の変更を有効にするためにデバイスをリロードする必要があります。

コンフィギュレーションの置換とロールバックの使用方法

コンフィギュレーション アーカイブの作成

configurerepl ace コマンドを使用するうえで前提条件となる設定はありません。**configurerepl ace** コマンドと、Cisco IOS コンフィギュレーション アーカイブおよび **archiveconfig** コマンドとの併用は任意ですが、コンフィギュレーション ロールバックのシナリオでは大きな利点があります。**archiveconfig** コマンドを使用する前に、コンフィギュレーション アーカイブを設定しておく必要があります。コンフィギュレーションアーカイブの特性を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **path** *url*
5. **maximum** *number*
6. **time-period** *minutes*
7. **end**
8. **archive config**

DETAILED STEPS

	Command or Action	Purpose
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	**archive** **Example:** `Device(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**path** *url* **Example:**	Cisco IOS コンフィギュレーション アーカイブの場所と、ファイル名のプレフィックスを指定します。

	Command or Action	Purpose
	`Device(config-archive)# path flash:myconfig`	**Note** パスのところでファイルの代わりにディレクトリを指定する場合、ディレクトリ名は path flash:/directory/ のように後ろにスラッシュを付ける必要があります。このスラッシュはファイル名の後ろでは必要ありません。ディレクトリを指定する場合にだけ使います。
ステップ 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	（任意）Cisco IOS コンフィギュレーション アーカイブに保存される実行コンフィギュレーションのアーカイブ ファイル数の上限値を設定します。 • *number* 引数は、Cisco IOS コンフィギュレーション アーカイブに保存される実行コンフィギュレーションのアーカイブ ファイル数の上限値を示します。有効な値は 1 ～ 14 で、デフォルトは 10 です。 **Note** このコマンドを使用する前に、**path** コマンドを設定して Cisco IOS コンフィギュレーションアーカイブの位置とファイル名プレフィックスを指定しておく必要があります。
ステップ 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 10`	（任意）Cisco IOS コンフィギュレーション アーカイブに実行コンフィギュレーションのアーカイブ ファイルを自動保存する間隔を設定します。 • Cisco IOS コンフィギュレーション アーカイブに現在の実行コンフィギュレーションのアーカイブ ファイルをどれほどの頻度で自動保存するかを、minutes 引数により分単位で指定します。 **Note** このコマンドを使用する前に、**path** コマンドを設定して Cisco IOS コンフィギュレーションアーカイブの位置とファイル名プレフィックスを指定しておく必要があります。
ステップ 7	**end** **Example:** `Device(config-archive)# end`	特権 EXEC モードに戻ります。

	Command or Action	Purpose
ステップ 8	**archive config** **Example:** `Device# archive config`	現在の実行設定ファイルを設定アーカイブに保存します。 **Note** このコマンドを使用する前に、**path** コマンドを設定する必要があります。

コンフィギュレーションの置換やロールバック操作の実行

保存された Cisco IOS コンフィギュレーション ファイルで現在の実行コンフィギュレーション ファイルを置換するには、次の作業を実行します。

Note この手順の前に、コンフィギュレーションアーカイブを作成しておく必要があります。詳細については、コンフィギュレーションアーカイブの作成を参照してください。次に、現在の実行コンフィギュレーションで問題が生じた場合に、アーカイブしておいたコンフィギュレーションに戻す手順の詳細を示します。

SUMMARY STEPS

1. **enable**
2. **configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**reverttrigger[error]***[***timer***minutes***]**|**time***minutes*]
3. **configure revert {now** |**timer{***minutes*|**idle***minutes***}}**
4. **configure confirm**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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 replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**reverttrigger[error]***[***timer***minutes***]**|**time***minutes*] **Example:** `Device# configure replace flash:myconfig-1 list time 30`	保存しておいた Cisco IOS コンフィギュレーション ファイルで現在の実行コンフィギュレーション ファイルを置換します。ファイルを置き換えた後、設定の変更を有効にするためにデバイスをリロードする必要があります。 • *target -url* 引数は、**archiveconfig** コマンドで作成されたコンフィギュレーションファイルなど、現在の実行コンフィギュレーションと置換す

	Command or Action	Purpose

		る、保存されたCisco IOSコンフィギュレーション ファイルのURLです（Cisco IOSファイル システムでアクセス可能なもの）。
		• **list** キーワードは、コンフィギュレーション置換動作のパスごとに、Cisco IOS ソフトウェア パーサーによって適用されるコマンドラインのリストを表示します。実行されたパスの総数も表示されます。
		• **force** キーワードは、現在の実行コンフィギュレーションから指定したCisco IOSコンフィギュレーション ファイルへの置換を、確認プロンプトを出さずに実行します。
		• **time***minutes* キーワードおよび引数は、現在の実行コンフィギュレーション ファイルの置換確認のために **configureconfirm** コマンドを入力しなければならない制限時間（分単位）を指定します。**configureconfirm** コマンドが指定の制限時間内に入力されない場合、コンフィギュレーション置換操作は自動的に戻されます（つまり、現在の実行コンフィギュレーション ファイルが **configurereplace** コマンド入力以前のコンフィギュレーション状態へと回復されます）。
		• **nolock** キーワードは、コンフィギュレーション置換操作中に他のユーザーが実行コンフィギュレーションを変更しないように実行コンフィギュレーション ファイルをロックする機能をオフにします。
		• **reverttrigger** キーワードは、元のコンフィギュレーションへ戻すトリガーを次の内容から設定します。
		  • **error**：エラー時に元のコンフィギュレーションに戻します。
		  • **timer***minutes*：指定した時間が過ぎると元のコンフィギュレーションに戻します。
		• **ignorecase** キーワードで、コンフィギュレーションに確認コマンドの大文字と小文字の区別を無視させることができます。

	Command or Action	Purpose
ステップ 3	**configure revert {now |timer{***minutes*|**idle***minutes*}}** **Example:** `Device# configure revert now` **Example:**	（任意）時間指定ロールバックをキャンセルしてロールバックを即時トリガーする、または時間指定ロールバックのパラメータをリセットするには、特権 EXEC モードで **configurerevert** コマンドを使用します。 • **now** ：ロールバックをただちにトリガーします。 • **timer** ：コンフィギュレーションを元に戻すタイマーをリセットします。 • 元に戻す時間を分単位で新たに指定するには、*minutes* 引数を **timer** キーワードとともに使用します。 • 保存されたコンフィギュレーションに戻すまでに、操作が行われないアイドル時間を最大どれほど長く許容できるかを設定するには、分単位の時間とともに **idle** キーワードを使用します。
ステップ 4	**configure confirm** **Example:** `Device# configure confirm`	（任意）保存しておいた Cisco IOS コンフィギュレーション ファイルの現在の実行コンフィギュレーション ファイルへの置換を確認します。ファイルを置き換えた後、設定の変更を有効にするためにデバイスをリロードする必要があります。 **Note** このコマンドは、**configurerеplace** コマンドの **time***seconds* キーワードおよび引数が指定されている場合にのみ使用します。
ステップ 5	**exit** **Example:** `Device# exit`	ユーザー EXEC モードに戻ります。

機能のモニターリングおよびトラブルシューティング

コンフィギュレーションの置換とロールバック機能をモニターおよびトラブルシューティングするには、この手順を実行します。

SUMMARY STEPS

1. **enable**
2. **show archive**
3. **debug archive versioning**
4. **debug archive config timestamp**
5. **exit**

DETAILED STEPS

ステップ 1 enable

このコマンドを使用して、特権 EXEC モードをイネーブルにします。パスワードを入力します（要求された場合）。次に例を示します。

Example:

```
Device> enable
Device#
```

ステップ 2 show archive

Cisco IOS コンフィギュレーション アーカイブに保存されているファイルに関する情報を表示するには、次のコマンドを使用します。次に例を示します。

Example:

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfig-2
 Archive #  Name
   0
   1       flash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

次に、実行コンフィギュレーションのアーカイブ ファイルをいくつか保存した状態で **showarchive** コマンドを使用した場合の出力例を示します。この例では、保存されるアーカイブ ファイルの最大数が 3 に設定されています。

Example:

```
Device# show archive
There are currently 3 archive configurations saved.
The next archive file will be named flash:myconfig-8
 Archive #  Name
```

```
  0
  1       :Deleted
  2       :Deleted
  3       :Deleted
  4       :Deleted
  5       flash:myconfig-5
  6       flash:myconfig-6
  7       flash:myconfig-7 <- Most Recent
  8
  9
  10
  11
  12
  13
  14
```

ステップ 3 debug archive versioning

このコマンドを使用して、Cisco IOS コンフィギュレーション アーカイブのアクティビティのデバッグを有効にして、コンフィギュレーションの置換とロールバックをモニターおよびトラブルシューティングします。次に例を示します。

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file flash:myconfig-7
Jan  9 06:46:29.547: backup worked
```

ステップ 4 debug archive config timestamp

このコマンドを使用して、コンフィギュレーション置換操作の各必須段階の処理時間、および操作中のコンフィギュレーション ファイルのサイズのデバッグをイネーブルにします。次に例を示します。

Example:

```
Device# debug archive config timestamp
Device# configure replace flash:myconfig force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file slot0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

ステップ 5 exit

このコマンドを使用して、ユーザ EXEC モードに戻ります。次に例を示します。

Example:

```
Device# exit
Device>
```

コンフィギュレーションの置換とロールバックの設定例

コンフィギュレーション アーカイブの作成例

次の例は、Cisco IOS コンフィギュレーション アーカイブの初期設定を実行する方法を示しています。この例では、flash:myconfig がコンフィギュレーション アーカイブの保存位置およびファイル名のプレフィックスとして設定され、保存するアーカイブ ファイルが最大 10 個に設定されます。

```
configure terminal
!
archive
 path flash:myconfig
 maximum 10
 end
```

保存された Cisco IOS コンフィギュレーション ファイルでの現在の実行コンフィギュレーションの置換：例

次の例では、flash:myconfig という名前で保存された Cisco IOS コンフィギュレーション ファイルで現在の実行コンフィギュレーションを置換する方法を示します。**configure replace** コマンドでは、確認プロンプトでインタラクティブに操作を進めます。

```
Device# configure replace flash:myconfig
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
```

次の例では、コンフィギュレーション置換操作中に適用されるコマンドラインを表示するために、**list** キーワードを指定しています。

```
Device# configure replace flash:myconfig list
This will apply all necessary additions and deletions
to replace the current running configuration with the
```

```
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
!Pass 1
!List of Commands:
no snmp-server community public ro
snmp-server community mystring ro

end
Total number of passes: 1
Rollback Done
```

スタートアップ コンフィギュレーション ファイルの復元：例

次の例に、**configurereplace** コマンドを使用して Cisco IOS スタートアップ コンフィギュレーション ファイルへ復元する方法を示します。この例は、オプションの **force** キーワードを使用して、インタラクティブ ユーザー プロンプトをオーバーライドする方法を示しています。

```
Device# configure replace nvram:startup-config force
Total number of passes: 1
Rollback Done
```

例：**configure confirm** コマンドを使用したコンフィギュレーション置換操作の実行

次に、**configure replace** コマンドを **time** *minutes* キーワードおよび引数とともに使用する例を示します。現在の実行コンフィギュレーション ファイルの置換を実行するには、指定の制限時間内に **configure confirm** コマンドを入力する必要があります。**configure confirm** コマンドが指定の制限時間内に入力されない場合、コンフィギュレーション置換操作は自動的に戻されます（つまり、現在の実行コンフィギュレーション ファイルが **configure replace** コマンド入力以前のコンフィギュレーション状態へと回復されます）。

```
Device# configure replace nvram:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
Device# configure confirm
```

次に、**configure revert** コマンドを **timer** キーワードとともに使用する例を示します。時間指定ロールバックをキャンセルしてロールバックを即時トリガーする、または時間指定ロールバックのパラメータをリセットするには、**configure revert** コマンドを入力する必要があります。

```
Device# configure revert timer 100
```

コンフィギュレーション ロールバック操作の実行：例

次の例は、現在実行中のコンフィギュレーションへの変更を行い、その変更をロールバックする方法を示しています。コンフィギュレーション ロールバック操作の一部として、ファイルに変更を加える前に現在の実行コンフィギュレーションを保存する必要があります。この例では、現在の実行コンフィギュレーションの保存に**archiveconfig**コマンドが使用されています。**configurereplace** コマンドで生成された出力は、ロールバック操作を完了するために 1 つのパスのみが実行されたことを示します。

Note **archiveconfig** コマンドを使用する前に、**path** コマンドを設定して Cisco IOS コンフィギュレーション アーカイブの位置とファイル名プレフィックスを指定しておく必要があります。

次のように、設定アーカイブの現在実行中のコンフィギュレーションを保存します。

```
archive config
```

それから、次の例に示すようにコンフィギュレーションの変更を入力します。

```
configure terminal
!
user netops2 password rain
user netops3 password snow
exit
```

実行コンフィギュレーション ファイルに変更を加えた後、それらの変更をロールバックさせて、変更前のコンフィギュレーションに戻したくなったとします。**showarchive** コマンドは、交換ファイルとして使用される設定のバージョンを確認するために使用されます。次の例に示すように、**configurereplace** コマンドは交換コンフィギュレーション ファイルへ戻すために使用されます。

```
Device# show archive
There are currently 1 archive configurations saved.
The next archive file will be named flash:myconfig-2
 Archive #  Name
   0
   1       flash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
Device# configure replace flash:myconfig-1
Total number of passes: 1
Rollback Done
```

その他の参考資料

次の項に、コンフィギュレーションの置換とロールバック機能に関する参考資料を示します。

関連資料

関連項目	マニュアル タイトル
設定ロック	『*Exclusive Configuration Change Access and Access Session Locking*』
コンフィギュレーション ファイルを管理するためのコマンド	*Cisco IOS Configuration Fundamentals Command Reference*
コンフィギュレーション ファイルの管理についての情報	コンフィギュレーション ファイルの管理
コンフィギュレーションのコンテキスト差分ユーティリティ機能の使用	『*Contextual Configuration Diff Utility*』

標準

標準	タイトル
この機能でサポートされる新規の標準または変更された標準はありません。また、既存の標準のサポートは変更されていません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存MIBのサポートに変更はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

コンフィギュレーションの置換とロールバックの機能情報

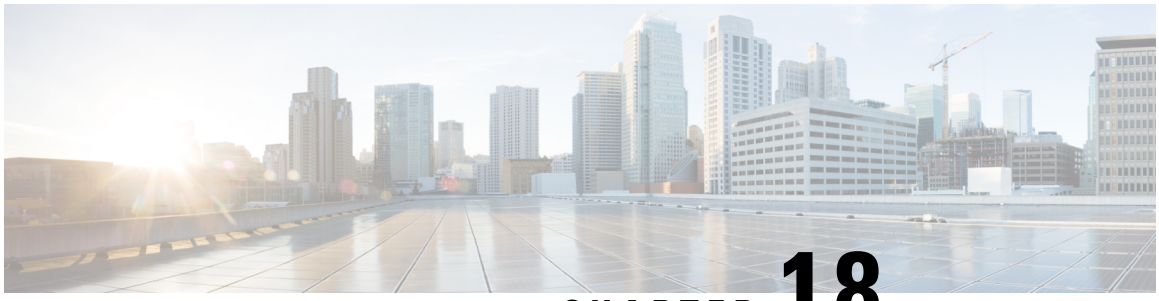
次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 22: コンフィギュレーションの置換とロールバックの機能情報

機能名	リリース	機能情報
コンフィギュレーションの置換とロールバック		コンフィギュレーションの置換とロールバック機能により、現在の実行コンフィギュレーションを、保存しておいたCisco IOS コンフィギュレーションファイルで置換することができます。この機能は、コンフィギュレーションを保存しておいた状態へ戻すために使用でき、そのコンフィギュレーションファイルが保存された後にどのような変更が加えられても、ロールバックさせることができます。 機能情報について、次の項で説明します。 この機能により、次のコマンドが変更されました。**archive config**、**configure confirm**、**configure replace**、**debug archive config timestamp**、**debug archive versioning**、**maximum**、**path (archive configuration)**、**show archive**、**show configuration lock**、**time-period**
コンフィギュレーションのバージョン管理		コンフィギュレーションのバージョン管理機能により、Cisco IOS 実行コンフィギュレーションのコピーをデバイス上やデバイス外で維持および管理することができます。コンフィギュレーション置換機能では、実行コンフィギュレーションの保存されたコピーへのロールバックを行うためにコンフィギュレーションバージョン管理機能を使用します。
排他的設定変更アクセス		排他的設定変更アクセス機能（「コンフィギュレーションロック」機能とも呼びます）を使用すると、Cisco IOS の実行コンフィギュレーションに排他的に変更アクセスし、複数のユーザーが同時に設定を変更するのを防ぐことができます。 この機能により、**show configuration lock** コマンドが変更され、コンフィギュレーションの置換とロールバック機能に適用されます。 詳しくは、別のモジュール『Exclusive Configuration Change Access and Access Session Locking』を参照してください。

機能名	リリース	機能情報
コンフィギュレーション ロールバック変更確認		コンフィギュレーション ロールバック変更確認機能により、コンフィギュレーション変更の実行に際して確認を要求するようオプションで設定できます。 この確認が受信できない場合、コンフィギュレーションは変更が適用される前の状態に戻されます。 このメカニズムは、ネットワーク デバイスとユーザまたは管理アプリケーションとの接続に、誤ったコンフィギュレーション変更に起因する切断を防止するものです。 この機能に関する詳細については、次の各項を参照してください。 この機能により、次のコマンドが変更されました。**configure confirm**、**configure replace**、**configure revert**、**configure terminal**



CHAPTER 18

コンフィギュレーションのコンテキスト差分ユーティリティ

コンフィギュレーションのコンテキスト差分ユーティリティ機能は、2 つのコンフィギュレーション ファイル（Cisco IOS XE Integrated File System（IFS）を通じてアクセス可能）を行ごとに比較し、その間の違いの一覧を生成する機能を提供します。生成される出力には、追加、変更、または削除されたコンフィギュレーション行に関する情報と、変更されたコンフィギュレーション行が存在するコンフィギュレーション モードが含まれます。

コンフィギュレーションのコンテキスト差分ユーティリティの前提条件

コンフィギュレーションのコンテキスト差分ユーティリティ機能で使用されるコンフィギュレーション ファイルの形式は、次に示す標準的な Cisco IOS XE コンフィギュレーション ファイルのインデント ルールに準拠している必要があります。

- 新しい行のすべてのコマンドは、コマンドがコンフィギュレーション サブモードにない限り、インデントなしで開始します。
- レベル 1 コンフィギュレーション サブモード内のコマンドは、スペース 1 個分インデントします。
- レベル 2 コンフィギュレーション サブモード内のコマンドは、スペース 2 個分インデントします。

- 以下、続くサブモード内のコマンドは、同じようにインデントします。

ルータには、比較対象の2つのコンフィギュレーションファイルの合計サイズよりも大きい連続したメモリ ブロックが必要です。

コンフィギュレーションのコンテキスト差分ユーティリティの制限事項

比較対象の2つのコンフィギュレーションファイルの合計サイズよりも大きい連続したメモリブロックがデバイスにない場合、diff 操作は失敗します。

コンフィギュレーションのコンテキスト差分ユーティリティについて

コンフィギュレーションのコンテキスト差分ユーティリティの利点

コンフィギュレーションのコンテキスト差分ユーティリティ機能は、2 つのコンフィギュレーション ファイル（Cisco IOS XE File System（IFS）を通じてアクセス可能）を行ごとに比較し、その間の違いの一覧を生成する機能を提供します。生成される出力には、次の項目に関する情報が含まれます。

- 追加、変更、削除された設定行。
- 変更された設定行が存在するコンフィギュレーション モード。
- 順序に依存する設定行の場所の変更。たとえば、**ip access-list** コマンドと **community-lists** コマンドは、コンフィギュレーション ファイル内での、同じ種類の他の Cisco IOS コマンドとの相対的な順序による影響を受けます。

コンフィギュレーションのコンテキスト差分ユーティリティの出力形式

diff 操作

コンフィギュレーションのコンテキスト差分ユーティリティ機能は、2 つのコンフィギュレーション ファイルのファイル名を入力として使用します。**show archive config differences** コマンドの使用により、指定されたファイルに対して diff 操作を実行し、2 つのファイル間の差分のリストを出力として生成します。出力の解釈は、コマンドで指定される2つのファイルの順序に依存します。ここでは、最初に入力されたファイルのファイル名を file1、2 番目に入力されたファイルのファイル名を file2 と仮定します。生成される出力リストの各エントリの前には、

見つかった差分の種類を示す固有のテキスト記号が付与されます。テキスト記号とその意味は次のとおりです。

- マイナス記号（–）は、設定行が file1 に存在するが file2 には存在しないことを示します。
- プラス記号（+）は、設定行が file2 に存在するが file1 には存在しないことを示します。
- 感嘆符（!）と説明用のコメントは、順序による影響を受ける設定行の場所が、file1 と file2 で異なることを示します。

差分比較操作

アプリケーションによっては、diff 操作で生成される出力に、変更されていない（つまり、マイナス記号やプラス記号のない）コンフィギュレーション行を含める必要があります。これらのアプリケーションでは、指定されたコンフィギュレーション ファイルを実行コンフィギュレーションファイルと比較する **show archive config incremental-diffs** コマンドを使用して、増分の diff 操作を実行できます。

増分diff操作が実行されると、実行コンフィギュレーションファイルに出現しない設定行（つまり、実行コンフィギュレーションと比較して指定されたファイルにのみ出現する設定行）のリストが出力として生成されます。感嘆符（!）と説明用のコメントは、順序による影響を受ける設定行の場所が、指定されたコンフィギュレーション ファイルと実行コンフィギュレーション ファイルで異なることを示します。

コンフィギュレーションのコンテキスト差分ユーティリティの使い方

コンフィギュレーションのコンテキスト差分ユーティリティを使用した行ごとのファイル比較の実行

SUMMARY STEPS

1. **enable**
2. 次のいずれか 1 つを入力します。
   - **show archive config differences** [*file1* [*file2*]]
   - **show archive config incremental-diffs** *file*
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	次のいずれか 1 つを入力します。
	• **show archive config differences** [*file1* [*file2*]]
	• **show archive config incremental-diffs** *file*
	Example:
	`Device# show archive config differences running-config startup-config`
	Example:
	`Device# show archive config incremental-diffs nvram:startup-config`
		2 つのコンフィギュレーション ファイル（Cisco IOS File System を通じてアクセス可能）を行ごとに比較し、その間の差分の一覧を生成します。
		または
		実行コンフィギュレーションファイルに対して、指定されたコンフィギュレーションファイルの行ごとの比較を実行し、実行コンフィギュレーションファイルに出現しないコンフィギュレーション行のリストを生成します。

ステップ 3	**exit** **Example:** `Device# exit`	ユーザー EXEC モードに戻ります。

コンフィギュレーションのコンテキスト差分ユーティリティの設定例

差分操作の例

この例では、実行コンフィギュレーション ファイルとスタートアップ コンフィギュレーションに対して比較操作を行います。次の表は、この例で使用しているコンフィギュレーション ファイルを示しています。

Table 23: diff 操作で使用するコンフィギュレーション ファイルの例

実行コンフィギュレーションファイル	スタートアップ コンフィギュレーション ファイル
no ip subnet-zero ip cef interface FastEthernet1/0 ip address 10.7.7.7 255.0.0.0 no ip route-cache no ip mroute-cache duplex half no ip classless snmp-server community public RO	ip subnet-zero ip cef ip name-server 10.4.4.4 voice dnis-map 1 dnis 111 interface FastEthernet1/0 no ip address no ip route-cache no ip mroute-cache shutdown duplex half ip default-gateway 10.5.5.5 ip classless access-list 110 deny ip any host 10.1.1.1 access-list 110 deny ip any host 10.1.1.2 access-list 110 deny ip any host 10.1.1.3 snmp-server community private RW

次に、**show archive config differences** コマンドの出力例を示します。この出力例は、以下の表のコンフィギュレーション ファイルに対して実行された diff 操作の結果を示しています。

```
Device# show archive config differences system:running-config nvram:startup-config

+ip subnet-zero

+ip name-server 10.4.4.4

+voice dnis-map 1
```

```
 +dnis 111

interface FastEthernet1/0

 +no ip address

 +shutdown

+ip default-gateway 10.5.5.5

+ip classless

+access-list 110 deny ip any host 10.1.1.1

+access-list 110 deny ip any host 10.1.1.2

+access-list 110 deny ip any host 10.1.1.3

+snmp-server community private RW

-no ip subnet-zero

interface FastEthernet1/0

 -ip address 10.7.7.7 255.0.0.0

-no ip classless

-snmp-server community public RO
```

差分比較操作の例

この例では、スタートアップ コンフィギュレーション ファイルと実行コンフィギュレーションファイルに対して増分diff操作を行います。次の表は、この例で使用しているコンフィギュレーション ファイルを示しています。

Table 24: 差分比較操作の例で使用するコンフィギュレーション ファイル

スタートアップコンフィギュレーション ファイル	実行コンフィギュレーションファイル
ip subnet-zero ip cef ip name-server 10.4.4.4 voice dnis-map 1 dnis 111 interface FastEthernet1/0 no ip address no ip route-cache no ip mroute-cache shutdown duplex half ip default-gateway 10.5.5.5 ip classless access-list 110 deny ip any host 10.1.1.1 access-list 110 deny ip any host 10.1.1.2 access-list 110 deny ip any host 10.1.1.3 snmp-server community private RW	no ip subnet-zero ip cef interface FastEthernet1/0 ip address 10.7.7.7 255.0.0.0 no ip route-cache no ip mroute-cache duplex half no ip classless snmp-server community public RO

次に、**show archive config incremental-diffs** コマンドの出力例を示します。この出力例は、以下の表のコンフィギュレーションファイルに対して実行された増分diff操作の結果を示しています。

```
Device# show archive config incremental-diffs startup-config

ip subnet-zero

ip name-server 10.4.4.4
```

```
voice dnis-map 1
 dnis 111
interface FastEthernet1/0
 no ip address
 shutdown
ip default-gateway 10.5.5.5
ip classless
 access-list 110 deny   ip any host 10.1.1.1
 access-list 110 deny   ip any host 10.1.1.2
 access-list 110 deny   ip any host 10.1.1.3
snmp-server community private RW
```

その他の参考資料

次の項に、コンフィギュレーション パーティショニング機能に関する参考資料を示します。

関連資料

関連項目	マニュアル タイトル
実行コンフィギュレーションのパフォーマンス強化：インターフェイスの **parserconfigcache**	コンフィギュレーション生成のパフォーマンス拡張
カスタマー サービスのプロビジョニング、コンフィギュレーション ロールバック、コンフィギュレーション ロッキング、およびコンフィギュレーション アクセス コントロール	コンフィギュレーションのコンテキスト差分ユーティリティ
コンフィギュレーション管理：コンフィギュレーション変更およびロギング	コンフィギュレーション変更通知およびロギング
コンフィギュレーション管理：コンフィギュレーション変更およびロギングのクイック保存：[2].	コンフィギュレーション ロガー永続性
Cisco IOS ソフトウェア コンフィギュレーション アクセス制御およびコンフィギュレーション セッション ロック（「Config ロック」）。	排他的設定変更アクセスとアクセス セッション ロック

2 「コンフィギュレーション ロガー永続性」機能により、スタートアップ コンフィギュレーション全体を保存するのではなく、最後の startup-config ファイルが生成された時点から入力されたコマンドだけを保存できます。

標準

標準	タイトル
この機能に関連付けられている規格はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	--

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/techsupport

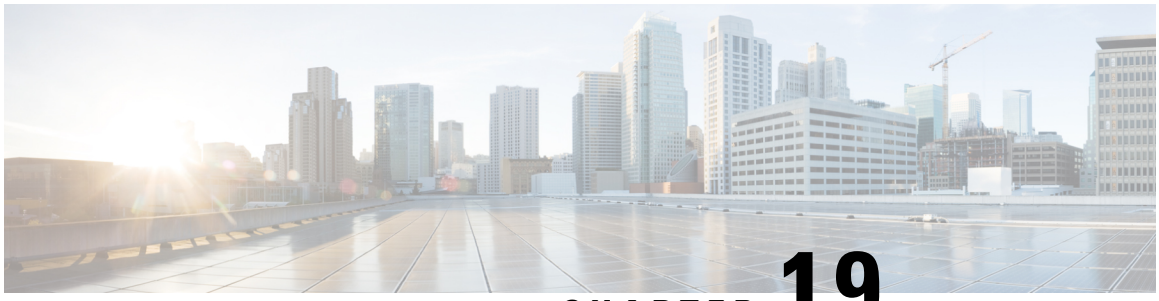
コンフィギュレーションのコンテキスト差分ユーティリティの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 25: コンフィギュレーションのコンテキスト差分ユーティリティの機能情報

機能名	リリース	機能情報
コンフィギュレーションのコンテキスト差分ユーティリティ	Cisco IOS XE リリース 2.1	コンフィギュレーションのコンテキスト差分ユーティリティ機能は、2つのコンフィギュレーションファイルを行ごとに比較し、その間の違いの一覧を生成する機能を提供します。生成される出力には、追加、変更、または削除されたコンフィギュレーション行に関する情報と、変更されたコンフィギュレーション行が存在するコンフィギュレーション モードが含まれます。 この機能は、Cisco IOS XE Release 2.1 で、Cisco ASR 1000 シリーズ ルータに導入されました。 この機能により、次のコマンドが変更されました。**show archive config differences**、**show archive config incremental-diffs**



CHAPTER 19

コンフィギュレーション変更通知およびロギング

コンフィギュレーション変更通知およびロギング（コンフィギュレーションログアーカイブ）機能を使用すると、アーカイブ機能を実装することにより、設定変更をセッションごとおよびユーザごとに追跡できます。このアーカイブでは、適用された各コンフィギュレーションコマンド、コマンドを適用した人、コマンドの Parser Return Code（PRC）、コマンドを適用した時刻を追跡する「設定ログ」が保存されます。また、この機能により、設定ログが変化したときに非同期通知を登録されたアプリケーションに送信する、通知メカニズムも追加されます。

コンフィギュレーション変更通知およびロギング機能が導入されるまでは、シスコ ソフトウェアの設定が変更されたかどうかを判断するための唯一の方法は、実行コンフィギュレーションとスタートアップ コンフィギュレーションのコピーをローカル コンピュータに保存し、行単位で比較することでした。この比較方法では、変更を特定できますが、変更が行われた順序や、変更に責任を持つ人は特定できません。

コンフィギュレーション変更通知およびロギングの制約事項

- コンフィギュレーション モードでの完全なコマンド入力のみがログに記録されます。
- **copy** コマンドを使用して適用されたコンフィギュレーション ファイルの一部であるコマンドは、ログに記録されません。

コンフィギュレーション変更通知およびロギングについて

設定ログ

コンフィギュレーション変更通知およびロギング機能は、設定ログを保持することで、シスコソフトウェアの実行コンフィギュレーションに加えられた変更を追跡します。この設定ログは、CLIまたはHTTPのみを介して開始される変更を追跡します。アクション ルーチンの呼び出しが発生する完全なコマンドが記録されます。次の種類の入力はログに記録されません。

- 結果的に構文エラー メッセージが表示されるコマンド
- デバイス ヘルプ システムを呼び出す一部のコマンド

実行される各設定コマンドでは次の情報が記録されます。

- 実行されたコマンド
- コマンドが実行されたコンフィギュレーション モード
- コマンドを実行したユーザーの名前
- コマンドが実行された時間
- 設定変更のシーケンス番号
- コマンドへのパーサー返還コード

設定ログの情報を表示するには、**show archive log config** コマンドを使用します。ただし、Parser Return Code は、シスコ アプリケーションの内部だけで使用されるため、除外されます。

コンフィギュレーション変更通知およびコンフィギュレーション変更ロギング

設定変更の通知をソフトウェア システム ロギング（syslog）プロセスに送信するように、コンフィギュレーション変更通知およびロギング機能を設定できます。syslog 通知機能を使用すると、ポーリングや情報収集作業を実行しなくても、設定ログ情報をモニタリングできます。

コンフィギュレーション変更通知およびロギング機能では、セッションごとまたはユーザごとにユーザが入力した設定変更を追跡できます。管理者はこのツールを使用して、ソフトウェアの実行コンフィギュレーションに加えられた設定変更をすべて追跡し、その変更を実行したユーザーを特定できます。

EAL4 + 認証用のコンフィギュレーション ロガーの機能強化

Evaluation Assurance Level 4+（EAL4+）認定のためのコンフィギュレーション ロガー機能拡張により、ロギング プロセスが Conformance to Common Criteria, EAL4+ Firewall Protection Profiles で規定されている要件を満たすことが保証されます。これらの機能拡張には、次の要件を満たすための変更が含まれています。

- ロギング パラメータを変更すると、それらの変更がログに記録されます。これは、実行コンフィギュレーションに対する各変更に対し、コピー操作（**copy** *source* **running-config** など）から syslog メッセージを送信することで実現されます。
- 管理ユーザ グループに対する変更がログに記録されます。たとえば、特権 EXEC モード（「イネーブル」モード）へのアクセスの失敗が記録されます。

Note EAL の認定はシスコが要求するものではありません。これらの機能拡張は、将来の認定に備えた土台となるものです。

前述のロギング アクションは、デフォルトでは無効になっています。これらのロギング特性を有効にするには、「コンフィギュレーション変更通知およびロギング」機能モジュールの「コンフィギュレーション変更通知およびロギング機能の設定」セクションに記載されているタスクを実行します。

コンフィギュレーション変更通知およびロギングの設定方法

コンフィギュレーション変更通知およびロギングの設定

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging enable**
6. **logging size** *entries*
7. **hidekeys**
8. **notify syslog**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**archive** **Example:** `Device(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**log config** **Example:** `Device(config-archive)# log config`	設定変更ロガー コンフィギュレーション モードを開始します。
ステップ 5	**logging enable** **Example:** `Device(config-archive-log-config)# logging enable`	設定変更のロギングをイネーブルにします。 • コンフィギュレーション変更のロギングは、デフォルトでは無効になっています。
ステップ 6	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 200`	（任意）設定ログに保持する最大エントリ数を指定します。 • *entries* 引数の有効な値の範囲は、1 ～ 1000 です。デフォルト値は 100 エントリです。 • 設定ログがいっぱいになると、新しいエントリが追加されるたびに最も古いエントリが削除されます。 **Note** 現在のログ サイズよりも小さいログ サイズが新たに指定された場合、ログ エントリの経過時間にかかわらず、新しいログ サイズになるまで最も古いログ エントリがすぐに削除されます。
ステップ 7	**hidekeys** **Example:**	（任意）パスワード情報が設定ログ ファイルに表示されないようにします。

	Command or Action	Purpose
	`Device(config-archive-log-config)# hidekeys`	**Note** **hidekeys** コマンドを有効にすると、設定ログ ファイルにパスワード情報が表示されなくなり、セキュリティが向上します。
ステップ 8	**notify syslog** **Example:** `Device(config-archive-log-config)# notify syslog`	（任意）設定変更の通知をリモート syslog に送信できるようにします。
ステップ 9	**end** **Example:** `Device(config-archive-log-config)# end`	特権 EXEC モードに戻ります。

設定ログ エントリおよび統計の表示

設定ログのエントリまたは設定ログのメモリ使用量に関する統計情報を表示するには、ここに示す作業を実行します。コマンドは任意の順序で入力できます。

設定ログ エントリを表示し、設定ログのメモリ使用量を監視するために、コンフィギュレーション変更通知およびロギング機能に **show archive log config** コマンドが用意されています。

SUMMARY STEPS

1. **enable**
2. **show archive log config** *number* [*end-number*]
3. **show archive log config all provisioning**
4. **show archive log config statistics**
5. **exit**

DETAILED STEPS

ステップ 1 enable

このコマンドを使用して、特権 EXEC モードをイネーブルにします。パスワードを入力します（要求された場合）。次に例を示します。

Example:

```
Device> enable
```

ステップ 2 show archive log config *number* [*end-number*]

このコマンドを使用して、設定ログ エントリをレコード番号ごとに表示します。オプションの *end-number* を指定すると、*number* 引数で入力した値から *end-number* 引数で入力した値までの範囲のレコード番号を持つすべてのログ エントリが表示されます。次に例を示します。

```
Device# show archive log config 1 2

idx   sess   user@line        Logged command
 1     1     user1@console    logging enable
 2     1     user1@console    logging size 200
```

Example:

この例では、設定ログ エントリ番号 1 と 2 が表示されています。*number* 引数と *end-number* 引数の範囲は 1 ～ 2147483647 です。

ステップ 3 show archive log config all provisioning

すべての設定ログ ファイルを、表形式ではなくコンフィギュレーション ファイルでの表示形式で表示するには、このコマンドを使用します。次に例を示します。

Example:

```
Device# show archive log config all provisioning

archive
 log config
  logging enable
  logging size 200
```

この表示では、ログに記録されたコマンドを正しく適用するために必要な、コンフィギュレーション モードを変更するために使用したコマンドも表示されています。

ステップ 4 show archive log config statistics

コンフィギュレーションのメモリ使用量の情報を表示するには、このコマンドを使用します。次に例を示します。

Example:

```
Device# show archive log config statistics

Config Log Session Info:
   Number of sessions being tracked: 1
   Memory being held: 3910 bytes
   Total memory allocated for session tracking: 3910 bytes
   Total memory freed from session tracking: 0 bytes
Config Log log-queue Info:
   Number of entries in the log-queue: 3
   Memory being held in the log-queue: 671 bytes
   Total memory allocated for log entries: 671 bytes
   Total memory freed from log entries:: 0 bytes
```

ステップ 5 exit

このコマンドを使用して、ユーザ EXEC モードに戻ります。次に例を示します。

Example:

```
Device# exit
Device>
```

設定ログ エントリのクリア

設定ログのエントリは、2 つのうちいずれかの方法でクリアできます。**logging size** コマンドを使用して設定ログのサイズを縮小するか、または **logging enable** コマンドを使用して設定ログを無効にしてから再び有効にすることができます。

ログサイズのリセットによる設定ログの消去

このタスクでは、 **logging size** コマンドを 2 回入力して、ログ サイズを 1 に減らしてから、ログ サイズを目的の値にリセットする方法を示します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging size** *entries*
6. **logging size** *entries*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**archive** **Example:** `Device(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**log config** **Example:**	設定変更ロガー コンフィギュレーション モードを開始します。

	Command or Action	Purpose
	`Device(config-archive)# log config`	
ステップ 5	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 1`	設定ログに保持する最大エントリ数を指定します。 **Note** 設定ログのサイズを1に設定すると、最新のエントリ以外はすべて消去されます。
ステップ 6	**logging size** *entries* **Example:** `Device(config-archive-log-config)# logging size 200`	設定ログに保持する最大エントリ数を指定します。 **Note** 設定ログを消去した後、設定ログのサイズを目的の値にリセットする必要があります。
ステップ 7	**end** **Example:** `Device(config-archive-log-config)# end`	特権 EXEC モードに戻ります。

設定ログをディセーブルすることによる設定ログのクリア

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **no logging enable**
6. **logging enable**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
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。

	Command or Action	Purpose
ステップ 3	**archive** **Example:** `Device(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**log config** **Example:** `Device(config-archive)# log config`	設定変更ロガー コンフィギュレーション モードを開始します。
ステップ 5	**no logging enable** **Example:** `Device(config-archive-log-config)# no logging enable`	コンフィギュレーション変更のロギングを無効にします。 **Note** 設定ログを無効にすると、すべてのレコードが消去されます。
ステップ 6	**logging enable** **Example:** `Device(config-archive-log-config)# logging enable`	設定変更のロギングをイネーブルにします。
ステップ 7	**end** **Example:** `Device(config-archive-log-config)# end`	特権 EXEC モードに戻ります。

自動ログ削除

この機能を使用すると、設定可能な時間が経過すると、ロギングバッファからエントリを自動的に削除できます。エントリがデバイスから消去されるまでのローカルsyslog保持期間を設定する必要があります。特定の時間が経過した後にロギングデータを自動的に消去するには、**logging purge-log buffer days x time <x:y>** コマンドを使用します。ログエントリの最大保持期間は、1～120日の範囲で日単位で設定できます。この機能では、1日に1回のバッファクリーンアップも許可されます。これにより、24時間ごとに設定された期間に基づいてバッファログがクリーンアップされます。

（注） コマンドで保存期間を日単位でのみ指定した場合、ログの削除は翌日のコマンドの設定と同時に行われます。

自動ログ削除を設定するには、次の手順を実行します。

手順の概要

1. **enable**

2. **configure terminal**
3. **logging purge-log buffer days entries**
4. **logging purge-log buffer days x time <x:y>**
5. **end**

手順の詳細

	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。プロンプトが表示されたらパスワードを入力します。
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	**logging purge-log buffer days entries** 例： `Device(config)#logging purge-log buffer days 90`	ログエントリの最大保持時間を指定します。 （注） 有効な値の範囲は 1 ～ 120 です。
ステップ 4	**logging purge-log buffer days x time <x:y>** 例： `Device(config)#logging purge-log buffer days 90 time 15:45`	（任意）ログを自動削除する特定の時間を指定します。 （注） • ログは削除されます。 • 時刻が現在のシステム時刻よりも小さい場合、削除は翌日の指定された時刻に行われます。
ステップ 5	**end** 例： `Device(config)# end`	特権 EXEC モードに戻ります。

ログ自動削除の設定例

次に、自動ログ削除を有効にして、90 日前のデータのみを保持する例を示します。ログの削除は、指定された時刻（15:45）に行われます。

```
Router (config)# logging purge-log buffer days 90 time 15:45
*May 18 20:20:20 UTC: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration
change requiring running configuration sync detected - ' logging purgelog
buffer days 90 time 15:45
'. The running configuration will be sy
nchronized to the NETCONF running data store.
◦ May 18 20:20:21 UTC: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization
of the running configuration to the NETCONF running data store has
started.
```

```
May 18 20:20:26 UTC: %DMI-5-SYNC_COMPLETE: R0/0: dmiauthd: The running
configuration has been synchronized to the NETCONF running data store.
```

次に、自動ログ削除を有効にして、10 日前のデータのみを保持し、残りのログをバッファから削除する例を示します。

```
Router(config)# logging purge-log buffer days 10
Jul  5 19:48:16.974: %PARSER-5-CFGLOG_LOGGEDCMD: User:test  logged command:logging
purge-log buffer days 10
*Jul  5 19:48:17.330: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration change requiring
 running configuration sync detected - '  logging purge-log buffer days 10'.
The running configuration will be synchronized  to the NETCONF running data store.
*Jul  5 19:48:17.451: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization of the running
 configuration to the NETCONF running data store has started.
```

no logging purge-log buffer コマンドの出力例。

```
Router(config)# no logging purge-log buffer
Jul  5 19:49:29.601: %PARSER-5-CFGLOG_LOGGEDCMD: User:test  logged command:no logging
purge-log buffer
*Jul  5 19:49:29.980: %DMI-5-SYNC_NEEDED: R0/0: dmiauthd: Configuration change requiring
 running configuration sync detected - '  no logging purge-log buffer '.
The running configuration will be synchronized  to the NETCONF running data store.
*Jul  5 19:49:30.110: %DMI-5-SYNC_START: R0/0: dmiauthd: Synchronization of the running
 configuration to the NETCONF running data store has started.
```

コンフィギュレーション変更通知およびロギングの設定例

例：コンフィギュレーション変更通知およびロギングの設定

次に、設定ログの最大エントリ数を 200 にして設定ロギングをイネーブルにする例を示します。この例では、**hidekeys** コマンドを使用して設定ログ レコード内のパスワード情報の表示を抑止することでセキュリティを向上させ、**notify syslog** コマンドで syslog 通知を有効にしています。

```
configure terminal
archive
 log config
 logging enable
 logging size 200
 hidekeys
 notify syslog
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
コンフィギュレーションファイルの管理についての情報	『コンフィギュレーションファイルの管理コンフィギュレーションガイド』の「コンフィギュレーション ファイルの管理」モジュール
コンフィギュレーションファイルを管理するためのコマンド	Cisco IOS Configuration Fundamentals Command Reference

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

コンフィギュレーション変更通知およびロギングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 26: コンフィギュレーション変更通知およびロギングの機能情報

機能名	リリース	機能情報
コンフィギュレーション変更通知およびロギング		コンフィギュレーション変更通知およびロギング（コンフィギュレーションロギング）機能を使用すると、設定ログを実装することで、セッションごとまたはユーザごとに設定変更を追跡できます。設定ログには、適用された各コンフィギュレーション コマンド、コマンドを適用した人、コマンドのParser Return Code（PRC）、および、コマンドを適用した時刻が記録されます。また、この機能により、設定ログが変化したときに非同期通知を登録されたアプリケーションに送信する、通知メカニズムも追加されます。 次のコマンドが導入または変更されました。**archive**、**hidekeys**、**log config**、**logging enable**、**logging size**、**notify syslog**、**show archive log config**
自動ログ削除のサポート	Cisco IOS XE Dublin 17.12.1a	この機能を使用すると、ロギングバッファからエントリを削除できます。エントリがデバイスから自動的に消去されるまでのローカル syslog 保持期間を設定できます。この機能を有効にするには、**logging purge-log buffer days** コマンドを使用します。

CHAPTER 20

コンフィギュレーション パーティショニング

コンフィギュレーションパーティショニング機能によって実行コンフィギュレーション状態をモジュール化（「パーティショニング」）して、Cisco IOS ソフトウェアで実行コンフィギュレーションに柔軟にアクセスできるようにします。

この機能が搭載された Cisco IOS ソフトウェア イメージではデフォルトでオンになっています。

デバイスのコンフィギュレーション状態は、**showrunning-config** コマンドがユーザによって実行されると動的に取得されます。コンフィギュレーションパーティショニング機能がイネーブルの場合、システムによってデバイスのコンフィギュレーション状態が分割され、グループ化されます（「パーティション」と呼ばれます）。これにより、実行コンフィギュレーションで表示されるコマンドリストの生成時にユーザが確認したいコンフィギュレーション状態のみを取得できます。この機能により、システムのコンフィギュレーション状態全体が処理される従来の処理方法とは異なり、実行コンフィギュレーション コマンドのリストの生成時に実行コンフィギュレーション状態の一部のみが処理されるため、コンフィギュレーションが複雑なハイエンド システムのパフォーマンスを向上できます。

デフォルトのコンフィギュレーションパーティションはこの機能を導入することで提供されます。将来のリリースでは、他の Cisco IOS ソフトウェア機能によって独自のコマンド パーティションが提供される可能性があります。

コンフィギュレーション パーティショニングについて

システム実行コンフィギュレーション

Cisco IOS ソフトウェアベース デバイスのコンフィギュレーション管理には、不揮発性メモリに格納されたスタートアップ コンフィギュレーション（startup-config）およびシステムに適用されているすべてのコンフィギュレーション オプションである実行コンフィギュレーション（running-config）を管理する必要があります。通常、スタートアップ コンフィギュレーション ファイルはシステム起動時にロードされ、コマンドライン インターフェイス（CLI）を使用して適用されたシステムに対する実行コンフィギュレーションの変更は、実行コンフィギュレーションをコンフィギュレーション ファイルにコピーすることで保存されます（ローカルまたはネットワーク上）。ファイルは、起動時にデバイスをコンフィギュレーションする場合、または他のデバイスをコンフィギュレーションする場合に使用されます。

実行コンフィギュレーションを取得して表示またはコピーする

Cisco IOS のソフトウェア コンフィギュレーション モデルでは、コンフィギュレーション状態は分散して維持され、各コンポーネントは独自のコンフィギュレーション状態を保持します。グローバル コンフィギュレーション情報を取得するには、ソフトウェアは各コンポーネントをポーリングして、分散された情報を収集する必要があります。このコンフィギュレーション状態の取得処理は不揮発性生成（NVGEN）として知られる処理によって実行され、現在のコンフィギュレーション状態を表示する **showrunning-config** などのコマンドや、実行コンフィギュレーションをファイルにコピーして保存する **copysystem:running-configuration** コマンドによって呼び出されます。取得処理が呼び出されると、NVGEN 処理によって各システム コンポーネント、各インターフェイス インスタンス、およびその他すべてのコンフィギュレーションされたコンポーネント オブジェクトが標準の順序でクエリーされます。NVGEN がこれらのクエリーを実行しているシステムを通過するときに、実行コンフィギュレーション ファイルが作成されます。表示およびコピーには作成された「仮想ファイル」が使用されます。

実行コンフィギュレーションをパーティショニングする利点

コンフィギュレーション パーティショニング機能は、Cisco IOS ソフトウェアに追加された一連のコンフィギュレーション生成のパフォーマンス拡張機能の最新機能です（関連する機能については、「関連ドキュメント」セクションを参照してください）。この機能によって、**showrunning-config** コマンドの実行時に表示したいシステム コンポーネントのみがクエリーされるため、システム応答時間が短縮されます。

コンフィギュレーション パーティショニング機能がイネーブルの場合、システムによってデバイスのコンフィギュレーション状態が分割され、グループ化されます（「パーティション」と呼ばれます）。これにより、仮想実行コンフィギュレーション ファイル（コンフィギュレーション コマンドのリスト）が生成されます。新しいコマンド **showrunning-configpartition** を使用すると、一度に実行コンフィギュレーションをすべて表示したり、特定のストリングに一致

する行のみを表示するのではなく、検証したい実行コンフィギュレーションの部分のみを表示することができます。

この機能は、ユーザが表示したいシステム コンポーネントのグループ（特定のインターフェイスなど）のみのNVGEN処理をシステムで実行してシステムのパフォーマンスを向上できることが主な利点であると言えます。この特徴は、システム コンポーネントをすべて処理した後に生成されたリストをフィルタ処理するだけの**showrunning-config**コマンドのその他の拡張とは対照的です。

実行コンフィギュレーションを部分的に生成するため、システムのコンフィギュレーション状態を選択的に処理することを「コンフィギュレーション パーティショニング」と呼びます。

コンフィギュレーション情報に柔軟にアクセスできることで、サイズの大きいコンフィギュレーション ファイルがあるハイエンドなルーティング プラットフォームにパフォーマンスの重大な利点をもたらし、同時に詳細なコンフィギュレーション機能を細かに実装することでコンフィギュレーション管理を強化します。詳細なコンフィギュレーション オプションには、Cisco IOS ソフトウェアのカスタマー サービスのプロビジョニング、コンフィギュレーション ロールバック、コンフィギュレーション ロッキング、およびコンフィギュレーション アクセス コントロールのサポートが含まれます。

コンフィギュレーションパーティショニング機能を使用するには

コンフィギュレーション パーティションの表示

この機能を活用するには、主に特権 EXEC モードで **showrunning-configpartition***part* コマンドを使用します。このコマンドは、**showrunning-config** コマンド専用の拡張です。

Note **partition***part* コマンドの拡張は、**more:systemrunning-config** コマンドでは利用できません。

この機能は既存のコマンドのパフォーマンスを向上するので、この機能が搭載されたCisco IOSソフトウェア イメージではデフォルトでオンになっています。お使いのシステムでサポートおよび実行されているかどうかを簡単に判断するには、特権 EXEC モードで **showrunning-configpartition?** コマンドを実行します。

SUMMARY STEPS

1. **show running-config partition ?**
2. **show running-config partition** *part*

DETAILED STEPS

ステップ 1 **show running-config partition ?**

このコマンドを実行すると、システムに表示できる実行コンフィギュレーションの部分が表示されます。

コンフィギュレーション パーティショニング機能がシステムでサポートされており、イネーブルの場合は、ヘルプ出力の 1 行目に「config partition is TRUE」というストリングが表示されます。

ここに示すコマンド構文を入力するとエラー メッセージが表示される場合は、この機能はシステムでサポートされていません。実行コンフィギュレーションの部分のみを表示できる他のリリースで利用可能な **showrunning-config** コマンドの既存の拡張については、コマンドのマニュアルを参照してください。

Note 利用できるコンフィギュレーションの部分は、ソフトウェア イメージによって異なり、コンフィギュレーションされている機能に依存します。

Example:

```
Router# show running-config partition ?
 config partition is TRUE
  access-list       All access-list configurations
  boot              All boot configurations
  class-map         All class-map configurations
  common            All remaining unregistered configurations
  global-cdp        All global cdp configurations
  interface         All Interface specific Configurations
  ip-as-path        All IP as-path configurations
  ip-community      All IP community list configurations
  ip-domain-list    All ip domain list configurations
  ip-prefix-list    All ip prefix-list configurations
  ip-static-routes  All IP static configurations
  line              All line mode configurations
  policy-map        All policy-map configurations
  route-map         All route-map configurations
  router            All routing configurations
  snmp              All SNMP configurations
  tacacs            All TACACS configurations
```

表示する実行コンフィギュレーションの部分を選択して、ステップ 2 で関連キーワードを part 引数として使用します。

ステップ 2 **show running-config partition** *part*

たとえば、システムで NVGEN 処理を実行コンフィギュレーション状態の access-list 部分に関連するコンポーネントのみで実行して、access-list に関連するコンポーネントのみを表示する場合は、**showrunning-configpartitionaccess-list** コマンドを入力します。

Example:

```
Router# show running-config partition access-list
        Building configuration...
Current configuration : 127 bytes
!
Configuration of Partition access-list
!
!
!
```

```
access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
!
end
```

Note このコマンドを使用すると、NVGEN 処理を実行して、特定のインターフェイスに関する結果出力を表示します。複数のインターフェイスがアクティブなシステムで使用できる設計のこの動作がコンフィギュレーション パーティショニング機能の主な役割です。

次の例では、メインのコンフィギュレーション パーティションはインターフェイス コンフィギュレーションです。生成される特定のコンフィギュレーション部分は、ファスト イーサネット インターフェイス 0/0 のコンフィギュレーションです。

Example:

```
Router# show running-config partition interface fastethernet0/0
Building configuration...
Current configuration : 213 bytes
!
Configuration of Partition interface FastEthernet0/0
!
!
interface FastEthernet0/0
 ip address 10.4.2.39 255.255.255.0
 no ip route-cache cef
 no ip route-cache
 duplex half
 ipv6 enable
 no cdp enable
!
!
end
```

コンフィギュレーションパーティショニング機能をディセーブルにする

この機能は既存のコマンドのパフォーマンスを向上させるので、この機能が搭載された Cisco IOS ソフトウェア イメージではデフォルトでオンになっています。しかし、この機能は少量のシステム リソース（メモリおよび CPU）を消費するため不要な場合、ディセーブルにしたい場合があります。コンフィギュレーション パーティショニングをディセーブルにするには、次の手順を実行してください。手順はユーザ EXEC モードで起動されていることを前提としています。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no parser config partition**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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 parser config partition**
	Example:
	`Router(config)# no parser config partition`
	Example:
	`Disabling config partitioning`
	Example:
	`Router(config)#`
		コンフィギュレーションパーティショニング機能をディセーブルにします。

What to do next

次の作業

機能をディセーブルにした後、イネーブルにするには、グローバル コンフィギュレーション モードで **parserconfigpartition** コマンドを使用します。

Note この機能はデフォルトでイネーブルになっているので、実行コンフィギュレーションファイルには、**no** 形式のみが表示されます。または、**copyrunning-configstartup-config** コマンドを実行するとスタートアップ コンフィギュレーション ファイルに書き込まれます。

コンフィギュレーションパーティショニングするためのコンフィギュレーション例

ここでは、**show running-config partition** コマンドを使用してコンフィギュレーション パーティションを表示する例を示します。

コンフィギュレーション パーティションの表示例

この例では、管理者が特定のインターフェイスの状態、およびシステムの他のコンポーネントの一部のコンフィギュレーションを確認するために実行する一連の手順で **showrunning-configpartition** と関連コマンドを一緒に使用しています。標準の **showrunning-config** コマンド（例：**showrunning-config|includeaccess-list**）による、同等のフィルタされた出力もデモとして含まれます。

Note *part* 引数には **showrunning-configpartroutereigrp1** のように複数のパーティション名キーワードを含めることができます。

```
gt3-7200-3# show running-config partition ?
  access-list       All access-list configurations
  boot              All boot configurations
  class-map         All class-map configurations
  global-cdp        All global cdp configurations
  interface         All Interface specific Configurations
  ip-as-path        All IP as-path configurations
  ip-community      All IP community list configurations
  ip-domain-list    All ip domain list configurations
  ip-static-routes  All IP static configurations
  line              All line mode configurations
  policy-map        All policy-map configurations
  route-map         All route-map configurations
  router            All routing configurations
  service           All service configurations
  snmp              All SNMP configurations
gt3-7200-3# show running-config partition access-list

Building configuration...
Current configuration : 87 bytes
!
!
!
!
access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
!
end
gt3-7200-3# show running-config | include access-list

access-list 90 permit 0.0.0.0 1.2.3.5
access-list 100 permit 10 any any
gt3-7200-3#
gt3-7200-3# show running-config partition boot

Building configuration...
Current configuration : 51 bytes
!
boot network tftp:/service_config.txt
!
!
!
end
gt3-7200-3# show running-config partition class-map
```

```
Building configuration...
Current configuration : 78 bytes
!
!
!
class-map match-all abc
  match any
class-map match-all xyz
!
!
!
end
gt3-7200-3# show running-config | begin class-map

class-map match-all abc
  match any
class-map match-all xyz
!
!
gt3-7200-3# show running-config partition global-cdp

Building configuration...
Current configuration : 43 bytes
!
!
!
cdp timer 20
cdp holdtime 100
!
end
gt3-7200-3# show running-config | include

global-cdp

cdp timer 20
cdp holdtime 100
gt3-7200-3#
gt3-7200-3# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES NVRAM  administratively down down
Ethernet2/0                10.4.2.32       YES NVRAM  up                    up
Ethernet2/1                unassigned      YES NVRAM  administratively down down
Ethernet2/2                unassigned      YES NVRAM  administratively down down
Ethernet2/3                unassigned      YES NVRAM  administratively down down
Serial3/0                  unassigned      YES NVRAM  administratively down down
Serial3/1                  unassigned      YES NVRAM  administratively down down
Serial3/2                  unassigned      YES NVRAM  administratively down down
Serial3/3                  unassigned      YES NVRAM  administratively down down
Loopback0                  unassigned      YES NVRAM  administratively down down
Loopback234                unassigned      YES NVRAM  administratively down down
gt3-7200-3# show running-config partition interface fastethernet0/0
Building configuration...
Current configuration : 98 bytes
!
!
!
interface FastEthernet0/0
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
```

```
gt3-7200-3# show running-config partition interface ethernet2/0

Building configuration...
Current configuration : 122 bytes
!
!
!
interface Ethernet2/0
 ip address 10.4.2.32 255.255.255.0
 no ip proxy-arp
 no ip route-cache
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/1
Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/1
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/2

Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/2
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface ethernet2/3
Building configuration...
Current configuration : 94 bytes
!
!
!
interface Ethernet2/3
 no ip address
 no ip route-cache
 shutdown
 duplex half
!
!
end
gt3-7200-3# show running-config partition interface serial3/0
Building configuration...
Current configuration : 103 bytes
!
!
!
```

```
interface Serial3/0
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface serial3/1
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/1
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface serial3/2
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/2
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface serial3/3
Building configuration...
Current configuration : 103 bytes
!
!
!
interface Serial3/3
 no ip address
 no ip route-cache
 shutdown
 serial restart-delay 0
!
!
end
gt3-7200-3# show running-config partition interface loopback0
Building configuration...
Current configuration : 79 bytes
!
!
!
interface Loopback0
 no ip address
 no ip route-cache
 shutdown
!
!
end
gt3-7200-3# show running-config partition interface loopback1
```

```
                                                                ^
% Invalid input detected at '^' marker.
gt3-7200-3# show running-config partition interface loopback234
Building configuration...
Current configuration : 81 bytes
!
!
!
interface Loopback234
 no ip address
 no ip route-cache
 shutdown
!
!
end
gt3-7200-3# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
gt3-7200-3(config)# interface ethernet 2/0.1
gt3-7200-3(config-subif)# exit
gt3-7200-3(config)# exit
gt3-7200-3#
00:13:05: %SYS-5-CONFIG_I: Configured from console by console
gt3-7200-3# show running-config partition interface ethernet2/0.1
Building configuration...
Current configuration : 58 bytes
!
!
!
interface Ethernet2/0.1
 no ip route-cache
!
!
end
gt3-7200-3# show run partition ip?
ip-as-path  ip-community  ip-domain-list  ip-static-routes
gt3-7200-3#sh run part ip-as
gt3-7200-3#sh run part ip-as-path

Building configuration...
Current configuration : 125 bytes
!
!
!
ip as-path access-list 2 permit $ABC
ip as-path access-list 2 permit $xyz*
ip as-path access-list 2 permit qwe*
!
end
gt3-7200-3# show running-config partition ip-community

Building configuration...
Current configuration : 92 bytes
!
!
!
ip community-list standard asd permit
ip community-list expanded qwe deny uio*
!
end
gt3-7200-3# show running-config | include ip community
ip community-list standard asd permit
ip community-list expanded qwe deny uio*
gt3-7200-3#
```

```
gt3-7200-3# show running-config partition ip-domain-list

Building configuration...
Current configuration : 70 bytes
!
ip domain-list iop
ip domain-list tyu
ip domain-list jkl
!
!
!
end
gt3-7200-3# show running-config partition
 ip-static-routes

Building configuration...
Current configuration : 98 bytes
!
!
!
ip route 0.0.0.0 0.0.0.0 Ethernet2/0
ip route 171.69.1.129 255.255.255.255 10.4.29.1
!
end
gt3-7200-3# show running-config partition line
Building configuration...
Current configuration : 489 bytes
!
!
!
!
line con 0
 exec-timeout 0 0
 transport output lat pad v120 mop telnet rlogin udptn nasi
 stopbits 1
line aux 0
 transport output lat pad v120 mop telnet rlogin udptn nasi
 stopbits 1
line vty 0
 password lab
 login
 transport input lat pad v120 mop telnet rlogin udptn nasi
 transport output lat pad v120 mop telnet rlogin udptn nasi
line vty 1 4
 login
 transport input lat pad v120 mop telnet rlogin udptn nasi
 transport output lat pad v120 mop telnet rlogin udptn nasi
!
end
gt3-7200-3# show running-config partition policy-map
Building configuration...
Current configuration : 162 bytes
!
!
!
policy-map qwer
 description policy-map qwer.
  class xyz
    shape peak 8000 32 32
policy-map p1
policy-map sdf
  class abc
   set precedence 4
!
```

```
!
!
end
gt3-7200-3# show running-config partition route-map
Building configuration...
Current configuration : 65 bytes
!
!
!
route-map iop permit 10
!
route-map rty permit 10
!
!
end
gt3-7200-3#sh run part router bgp 1
Building configuration...
Current configuration : 111 bytes
!
!
!
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 distance bgp 2 2 2
 no auto-summary
!
!
end
gt3-7200-3#sh run part router egp ?
<0-65535>  Remote autonomous system number
gt3-7200-3#sh run part router egp 1
Building configuration...
Current configuration : 46 bytes
!
!
!
router egp 1
 timers egp 20 20
!
!
end
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router eigrp ?
<1-65535>  Autonomous system number
gt3-7200-3# show running-config partition router eigrp 1
Building configuration...
Current configuration : 13 bytes
!
!
!
!
end
gt3-7200-3#
gt3-7200-3# sh run part router eigrp 2
```

```
Building configuration...
Current configuration : 57 bytes
!
!
!
router eigrp 2
 variance 10
 auto-summary
!
!
end
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router isis ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router isis qwe
Building configuration...
Current configuration : 86 bytes
!
!
!
router isis qwe
 set-attached-bit route-map qwer
 use external-metrics
!
!
end
gt3-7200-3# show running-config partition router isis ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router iso
gt3-7200-3# show running-config partition router iso-igrp ?
  WORD  ISO routing area tag
  |     Output modifiers
  <cr>
gt3-7200-3# show running-config partition router iso-igrp

Building configuration...
Current configuration : 31 bytes
!
!
!
router iso-igrp
!
!
end
gt3-7200-3# show running-config | begin iso
router iso-igrp
!
router isis qwe
 set-attached-bit route-map qwer
 use external-metrics
```

```
!
router egp 1
 timers egp 20 20
!
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 distance bgp 2 2 2
 no auto-summary
!

gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router mobile ?
  |  Output modifiers
  <cr>
gt3-7200-3# show running-config partition router mobile

Building configuration...
Current configuration : 42 bytes
!
!
!
router mobile
 distance 20
!
!
end
gt3-7200-3# sh run | include router

router mobile
router odr
router eigrp 2
router ospf 4
router iso-igrp
router isis qwe
router egp 1
router bgp 1
gt3-7200-3# show running-config partition router ?
  bgp       Border Gateway Protocol (BGP)
  egp       Exterior Gateway Protocol (EGP)
  eigrp     Enhanced Interior Gateway Routing Protocol (EIGRP)
  isis      ISO IS-IS
  iso-igrp  IGRP for OSI networks
  mobile    Mobile routes
  odr       On Demand stub Routes
  ospf      Open Shortest Path First (OSPF)
  rip       Routing Information Protocol (RIP)
gt3-7200-3# show running-config partition router ospf ?
  <1-65535>  Process ID
gt3-7200-3# show running-config partition router ospf 4
Building configuration...
Current configuration : 64 bytes
!
!
!
```

```
router ospf 4
 log-adjacency-changes
 distance 4
!
!
end
gt3-7200-3# sh run part service

Building configuration...
Current configuration : 190 bytes
!
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service internal
service udp-small-servers
service tcp-small-servers
!
!
!
end
gt3-7200-3# sh run part snmp

Building configuration...
Current configuration : 84 bytes
!
!
!
snmp-server community user101 RW
snmp mib target list qwe host 0.0.0.0
!
end
```

その他の参考資料

次の項に、コンフィギュレーション パーティショニング機能に関する参考資料を示します。

関連資料

関連項目	マニュアル タイトル
実行コンフィギュレーションのパフォーマンス強化：インターフェイスの **parserconfigcache**	コンフィギュレーション生成のパフォーマンス拡張
カスタマー サービスのプロビジョニング、コンフィギュレーション ロールバック、コンフィギュレーション ロッキング、およびコンフィギュレーション アクセス コントロール	コンフィギュレーションのコンテキスト差分ユーティリティ
コンフィギュレーション管理：コンフィギュレーション変更およびロギング	コンフィギュレーション変更通知およびロギング
コンフィギュレーション管理：コンフィギュレーション変更およびロギングのクイック保存：[3].	コンフィギュレーション ロガー永続性

関連項目	マニュアル タイトル
Cisco IOS ソフトウェア コンフィギュレーション アクセス制御およびコンフィギュレーション セッション ロック（「Config ロック」）。	排他的設定変更アクセスとアクセス セッション ロック

[3] 「コンフィギュレーション ロガー永続性」機能により、スタートアップ コンフィギュレーション全体を保存するのではなく、最後の startup-config ファイルが生成された時点から入力されたコマンドだけを保存できます。

標準

標準	タイトル
この機能に関連付けられている規格はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	--

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/techsupport

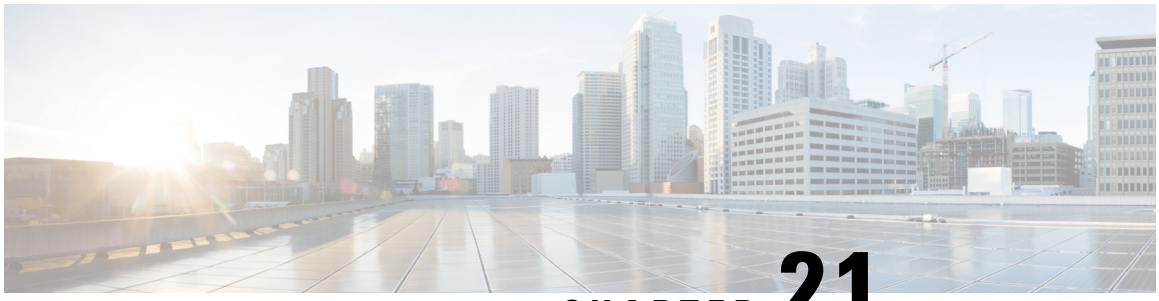
コンフィギュレーションパーティショニングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 27: コンフィギュレーション パーティショニングの機能情報

機能名	リリース	機能情報
コンフィギュレーション パーティショニング	12.2(33)SRB 12.2(33)SB 12.2(33)SXI	コンフィギュレーション パーティショニング機能によって実行コンフィギュレーション状態をモジュール化（「パーティショニング」）して、Cisco IOS ソフトウェアで実行コンフィギュレーションに柔軟にアクセスできるようにします。この機能が搭載された Cisco IOS ソフトウェア イメージではデフォルトでオンになっています。 12.2(33)SB では、この機能が Cisco 10000 シリーズに実装されました。 この機能に関する詳細については、次の各項を参照してください。 • コンフィギュレーション パーティショニングについて • コンフィギュレーション パーティショニング機能を使用するには



CHAPTER 21

コンフィギュレーションのバージョン管理

コンフィギュレーションのバージョン管理機能により、シスコの実行コンフィギュレーションのコピーをデバイス上やデバイス外で維持および管理することができます。コンフィギュレーション置換機能では、実行コンフィギュレーションの保存されたコピーへのロールバックを行うためにコンフィギュレーション バージョン管理機能を使用します。

コンフィギュレーションのバージョン管理について

コンフィギュレーション アーカイブ

シスコのコンフィギュレーション アーカイブは、**configure replace** コマンドにより提供されるコンフィギュレーションのロールバック機能を強化するために、シスコのコンフィギュレーション ファイルのアーカイブを保存、整理、管理するメカニズムです。この機能の導入前にも、実行コンフィギュレーションのコピーを **copy running-config** *destination-url* コマンドを使用して保存し、ローカルやリモートに置換ファイルを保管できました。ただし、この方法ではファイルの自動管理を行うことはできませんでした。コンフィギュレーションの置換とロールバック機能により、実行コンフィギュレーションのコピーを自動的にコンフィギュレーション アーカイブに保存できます。アーカイブされたファイルはコンフィギュレーションのチェックポイントとして参照することができ、**configure replace** コマンドを使用してコンフィギュレーションを以前の状態に戻せます。

archive config コマンドを使用すると、シスコのコンフィギュレーションをコンフィギュレーション アーカイブに保存できます。その場合、標準のディレクトリとファイル名のプレフィクスが使用され、バージョン番号（およびオプションでタイムスタンプ）が自動的に付加されます。バージョン番号は連続したファイルを保存するごとに、1 つずつ大きくなります。この機能により、保存した コンフィギュレーション ファイルを一貫して識別できます。アーカイブ

に保存する実行コンフィギュレーションの数は指定することができます。アーカイブ内のファイル数が上限値に達すると、次に最新のファイルが保存されるときに、最も古いファイルが自動的に消去されます。**show archive** コマンドを使用すると、コンフィギュレーション アーカイブに保存されているすべてのコンフィギュレーション ファイルに関する情報が表示されます。

コンフィギュレーション ファイルを保存するコンフィギュレーション アーカイブは、**configure replace** コマンドで使用することによって、お使いのプラットフォームに応じて次のファイル システムに配置できます。

- disk0 があるプラットフォーム：disk0:、disk1:、ftp:、pram:、rcp:、slavedisk0:、slavedisk1:、または tftp:
- disk0 がないプラットフォーム：bootflash:、ftp:、harddisk:、http:、pram:、rcp:、tftp:、usb0:、または usb1:

コンフィギュレーションのバージョン管理の設定方法

設定アーカイブの特性の設定

archive config コマンドを使用する前に、コンフィギュレーション アーカイブを設定しておく必要があります。コンフィギュレーション アーカイブの特性を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **path** *url*
5. **maximum** *number*
6. **time-period** *minutes*
7. **end**
8. **archive config**

DETAILED STEPS

	Command or Action	Purpose
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:**	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。

	Command or Action	Purpose
	`Device# configure terminal`	
ステップ 3	**archive** **Example:** `Device(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**path** *url* **Example:** `Device(config-archive)# path bootflash:myconfig`	コンフィギュレーションアーカイブに、ファイルのディレクトリとファイル名プレフィックスを指定します。 • ハードウェアプラットフォームによって、ファイルシステムの名前は、例に示しているものとは異なる可能性があります。 **Note** パスの部分でファイルの代わりにディレクトリを指定する場合、ディレクトリ名は path flash:/*directory*/ のように後ろにスラッシュを付ける必要があります。このスラッシュはファイル名の後ろでは必要ありません。ディレクトリを指定する場合にだけ使います。
ステップ 5	**maximum** *number* **Example:** `Device(config-archive)# maximum 14`	（任意）設定アーカイブに保存する実行設定のアーカイブ ファイルの最大数を指定します。 • *number* 引数は、コンフィギュレーション アーカイブに保存される実行コンフィギュレーションのアーカイブ ファイル数の上限値を示します。指定できる範囲は 1 ～ 14 です。デフォルトは 10 です。 **Note** このコマンドを使用する前に、**path** コマンドを設定して、コンフィギュレーション アーカイブの位置とファイル名プレフィックスを指定しておく必要があります。
ステップ 6	**time-period** *minutes* **Example:** `Device(config-archive)# time-period 10`	（任意）コンフィギュレーションアーカイブに現在実行中のコンフィギュレーションのアーカイブ ファイルを自動保存する間隔を設定します。 • 設定アーカイブに現在の実行設定のアーカイブファイルをどれほどの頻度で自動保存するかを、*minutes* 引数により分単位で指定します。

	Command or Action	Purpose
		Note このコマンドを使用する前に、**path** コマンドを設定して、コンフィギュレーション アーカイブの位置とファイル名プレフィックスを指定しておく必要があります。
ステップ 7	**end** **Example:** `Device(config-archive)# end`	特権 EXEC モードに戻ります。
ステップ 8	**archive config** **Example:** `Device# archive config`	現在の実行設定ファイルを設定アーカイブに保存します。 **Note** **archive config** コマンドを使用する前に、**path** コマンドを設定する必要があります。

コンフィギュレーションのモニタリングとトラブルシューティング

SUMMARY STEPS

1. **enable**
2. **show archive**
3. **debug archive versioning**
4. **debug archive config timestamp**
5. **exit**

DETAILED STEPS

ステップ 1 **enable**

このコマンドを使用して、特権 EXEC モードをイネーブルにします。パスワードを入力します（要求された場合）。次に例を示します。

Example:

```
Device> enable
Device#
```

ステップ 2 **show archive**

コンフィギュレーション アーカイブに保存されているファイルに関する情報を表示するには、次のコマンドを使用します。次に例を示します。

Example:

```
Device# show archive

There are currently 1 archive configurations saved.
The next archive file will be named bootflash:myconfig-2
 Archive #  Name
   0
   1       bootflash:myconfig-1 <- Most Recent
   2
   3
   4
   5
   6
   7
   8
   9
   10
   11
   12
   13
   14
```

次に、実行コンフィギュレーションのアーカイブ ファイルをいくつか保存した状態で **show archive** コマンドを使用した場合の出力例を示します。この例では、保存されるアーカイブ ファイルの最大数が 3 に設定されています。

Example:

```
Device# show archive

There are currently 3 archive configurations saved.
The next archive file will be named bootflash:myconfig-8
 Archive #  Name
   0
   1       :Deleted
   2       :Deleted
   3       :Deleted
   4       :Deleted
   5       bootflash:myconfig-5
   6       bootflash:myconfig-6
   7       bootflash:myconfig-7 <- Most Recent
   8
   9
   10
   11
   12
   13
   14
```

ステップ 3 debug archive versioning

このコマンドを使用して、コンフィギュレーション アーカイブのアクティビティのデバッグを有効にして、コンフィギュレーションの置換とロールバックをモニタおよびトラブルシューティングします。次に例を示します。

Example:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file bootflash:myconfig-7
Jan  9 06:46:29.547: backup worked
```

ステップ 4 debug archive config timestamp

このコマンドを使用して、コンフィギュレーション置換操作の各必須段階の処理時間、および操作中のコンフィギュレーション ファイルのサイズのデバッグをイネーブルにします。次に例を示します。

Example:

```
Device# debug archive config timestamp
Device# configure replace bootflash:myconfig force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file slot0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

ステップ 5 exit

このコマンドを使用して、ユーザ EXEC モードに戻ります。次に例を示します。

Example:

```
Device# exit
Device>
```

コンフィギュレーションのバージョン管理の設定例

例：コンフィギュレーション アーカイブの作成

次の例は、コンフィギュレーションアーカイブの初期設定を実行する方法を示しています。この例では、bootflash:myconfig がコンフィギュレーション アーカイブの保存位置およびファイル名のプレフィックスとして設定され、保存するアーカイブ ファイルが最大 10 個に設定されます。ハードウェア プラットフォームによって、ファイル システムの名前は、例に示しているものとは異なる可能性があります。

```
configure terminal
```

```
!
archive
 path bootflash:myconfig
 maximum 10
 end
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
コンフィギュレーションファイルの管理についての情報	『コンフィギュレーションファイルの管理コンフィギュレーションガイド』の「コンフィギュレーション ファイルの管理」モジュール
コンフィギュレーションファイルを管理するためのコマンド	Cisco IOS Configuration Fundamentals Command Reference

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

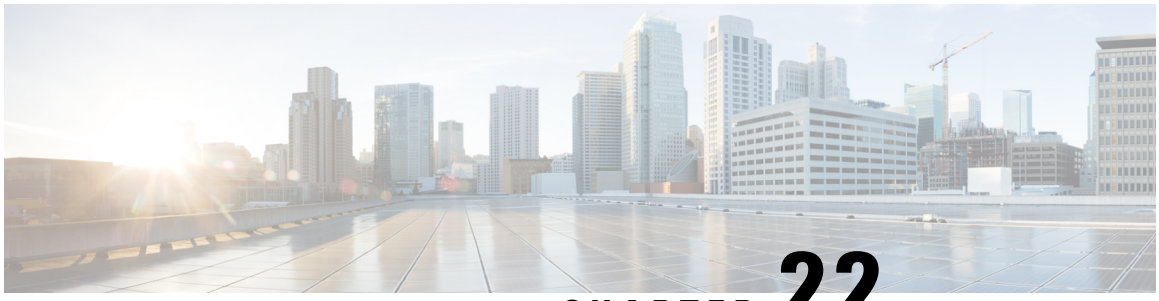
コンフィギュレーションのバージョン管理の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 28: コンフィギュレーションのバージョン管理の機能情報

機能名	リリース	機能情報
コンフィギュレーションのバージョン管理	12.2(25)S 12.2(33)SRA 12.3(7)T Cisco IOS XE Release 2.1	コンフィギュレーションのバージョン管理機能により、シスコの実行コンフィギュレーションのコピーをデバイス上やデバイス外で維持および管理することができます。コンフィギュレーション置換機能では、実行コンフィギュレーションの保存されたコピーへのロールバックを行うためにコンフィギュレーション バージョン管理機能を使用します。 次のコマンドが導入または変更されました。**archive config**、**debug archive versioning**、**log config**、**maximum**、**path**（アーカイブの設定）、**show archive**、**time-period**、**write-memory**



CHAPTER 22

コンフィギュレーション ロールバック変更確認

コンフィギュレーションロールバック変更確認機能により、コンフィギュレーション変更の実行に際して確認を要求するようオプションで設定できます。この確認が受信できない場合、コンフィギュレーションは変更が適用される前の状態に戻されます。このメカニズムは、ネットワークデバイスとユーザーまたは管理アプリケーションとの接続において、コンフィギュレーション変更に起因する切断を防止するものです。

コンフィギュレーションロールバック変更確認について

コンフィギュレーション ロールバック変更確認の操作

コンフィギュレーションロールバック変更確認機能は、コンフィギュレーションの変更に確認条件を追加できる機能です。この機能により、要求された変更の確認が設定済みの時間枠以内に受信されない場合にロールバックを行うことができます。コマンドの失敗を、コンフィギュレーションのロールバックをトリガーするように設定することもできます。

次に、このプロセスを実施するための手順の概要を示します。

1. 新しいオプションを使用すると、コンフィギュレーションの変更の確認を要求できます（確認の時間制限を指定する必要があります）。
2. 確認コマンドを入力する必要があります。要求された制限時間内に確認を入力しないと、コンフィギュレーションは以前の状態に戻ります。

コンフィギュレーションロールバック変更確認の設定方法

コンフィギュレーションの置換またはコンフィギュレーションのロールバック操作の確認を伴う実行

現在の実行コンフィギュレーション ファイルを保存済みのシスコのコンフィギュレーション ファイルに置換するには、次のタスクを実行します。

Note この手順の前に、コンフィギュレーションアーカイブを設定しておく必要があります。手順の詳細については、『コンフィギュレーションファイルの管理コンフィギュレーションガイド』の「コンフィギュレーション アーカイブの特性の設定」モジュールを参照してください。次に、現在の実行コンフィギュレーションで問題が生じた場合に、アーカイブしておいたコンフィギュレーションに戻す手順の詳細を示します。

SUMMARY STEPS

1. **enable**
2. **configure replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**revert trigger** [**error**] [**timer** *minutes*] | **time** *minutes*]
3. **configure revert** {**now** | **timer** {*minutes* | **idle** *minutes*}}
4. **configure confirm**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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 replace** *target-url* [**nolock**] [**list**] [**force**] [**ignorecase**] [**revert trigger** [**error**] [**timer** *minutes*]	**time** *minutes*] **Example:** `Device# configure replace bootflash:myconfig-1 list time 30`	現在の実行コンフィギュレーションファイルを保存済みのコンフィギュレーションファイルに置換します。 • *target-url*：**archive config** コマンドで作成されたコンフィギュレーションファイルなど、現在の実行コンフィギュレーションを置き換える、保存済みのコンフィギュレーション ファイルの URL を指定します（シスコのファイル システ

	Command or Action	Purpose

		ムでアクセス可能なもの）。ハードウェア プラットフォームによって、ファイル システムの名前は、例に示しているものとは異なる可能性があります。
		• **nolock**：コンフィギュレーション置換操作中に他のユーザが実行コンフィギュレーションを変更しないように実行コンフィギュレーション ファイルをロックする機能をオフにします。
		• **list**：コンフィギュレーション置換動作のパスごとに、シスコのソフトウェア パーサーによって適用されるコマンドラインのリストを表示します。実行されたパスの総数も表示されます。
		• **force**：現在の実行コンフィギュレーション ファイルと指定した保存済みコンフィギュレーション ファイルの交換を確認なしで実行します。
		• **ignorecase**：コンフィギュレーションに確認コマンドの大文字と小文字の区別を無視させることができます。
		• **time** *minutes*：現在の実行コンフィギュレーション ファイルの置換確認のために **configure confirm** コマンドを入力しなければならない制限時間（分単位）を指定します。**configure confirm** コマンドが指定の制限時間内に入力されない場合、コンフィギュレーション置換操作は自動的に戻されます（つまり、現在の実行コンフィギュレーション ファイルが **configure replace** コマンド入力以前のコンフィギュレーション状態へと回復されます）。
		• **revert trigger**：元のコンフィギュレーションへ戻すトリガーを次の内容から設定します。
		  • **error**：エラー時に元のコンフィギュレーションに戻します。
		  • **timer** *minutes*：指定した時間が過ぎると元のコンフィギュレーションに戻します。

ステップ 3	**configure revert** {**now** | **timer** {*minutes* | **idle** *minutes*}} **Example:** `Device# configure revert now`	（任意）時間指定ロールバックをキャンセルしてロールバックを即時トリガーするか、または時間指定ロールバックのパラメータをリセットします。 • **now**：ロールバックをただちにトリガーします。

	Command or Action	Purpose
		• **timer**：コンフィギュレーションを元に戻すタイマーをリセットします。 • 元に戻す時間を分単位で新たに指定するには、*minutes* 引数を **timer** キーワードとともに使用します。 • 保存されたコンフィギュレーションに戻すまでに、操作が行われないアイドル時間を最大どれほど長く許容できるかを設定するには、分単位の時間とともに **idle** キーワードを使用します。
ステップ 4	**configure confirm** **Example:** `Device# configure confirm`	（任意）現在の実行コンフィギュレーションファイルが保存済みのコンフィギュレーションファイルに置換されることを確認します。 **Note** このコマンドは、**configure replace** コマンドの **time** *minutes* キーワードおよび引数が指定されている場合にのみ使用してください。
ステップ 5	**exit** **Example:** `Device# exit`	ユーザー EXEC モードに戻ります。

コンフィギュレーション ロールバック変更確認の設定例

例：**configure confirm** コマンドを使用したコンフィギュレーション置換操作の実行

次に、**configure replace** コマンドを **time** *minutes* キーワードおよび引数とともに使用する例を示します。現在の実行コンフィギュレーションファイルの置換を実行するには、指定の制限時間内に **configure confirm** コマンドを入力する必要があります。**configure confirm** コマンドが指定の制限時間内に入力されない場合、コンフィギュレーション置換操作は自動的に戻されます（つまり、現在の実行コンフィギュレーションファイルが **configure replace** コマンド入力以前のコンフィギュレーション状態へと回復されます）。

```
Device# configure replace nvram:startup-config time 120
This will apply all necessary additions and deletions
to replace the current running configuration with the
contents of the specified configuration file, which is
assumed to be a complete configuration, not a partial
```

```
configuration. Enter Y if you are sure you want to proceed. ? [no]: Y
Total number of passes: 1
Rollback Done
Device# configure confirm
```

次に、**configure revert** コマンドを **timer** キーワードとともに使用する例を示します。時間指定ロールバックをキャンセルしてロールバックを即時トリガーする、または時間指定ロールバックのパラメータをリセットするには、**configure revert** コマンドを入力する必要があります。

```
Device# configure revert timer 100
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
コンフィギュレーションファイルの管理についての情報	『コンフィギュレーションファイルの管理コンフィギュレーションガイド』の「コンフィギュレーション ファイルの管理」モジュール
コンフィギュレーションファイルを管理するためのコマンド	Cisco IOS Configuration Fundamentals Command Reference

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

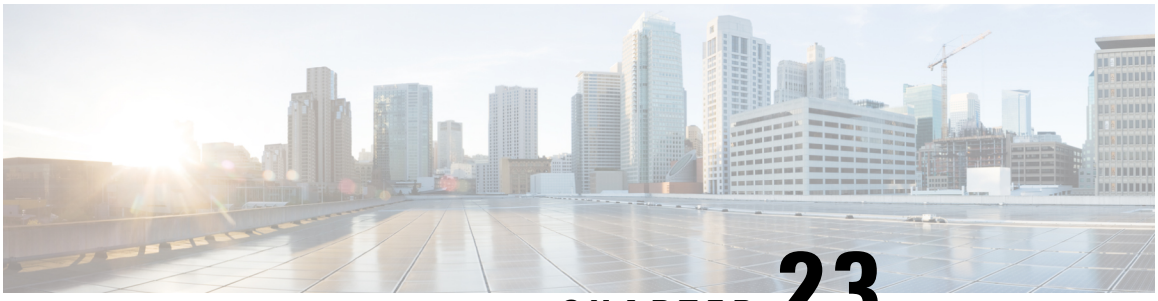
コンフィギュレーションロールバック変更確認の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 29: コンフィギュレーション ロールバック変更確認の機能情報

機能名	リリース	機能情報
コンフィギュレーション ロールバック変更確認	12.2(33)SB 12.2(33)SRC 12.2(33)SXI 12.4(20)T Cisco IOS XE Release 2.1	コンフィギュレーション ロールバック変更確認機能により、コンフィギュレーション変更の実行に際して確認を要求するようオプションで設定できます。 このメカニズムは、ネットワーク デバイスとユーザまたは管理アプリケーションとの接続に、誤ったコンフィギュレーション変更に起因する切断を防止するものです。 次のコマンドが導入または変更されました。**configure confirm**、**configure replace**、**configure revert**、**configure terminal**



CHAPTER 23

コンフィギュレーション ロガー永続性

コンフィギュレーション ロガー永続性機能は「クイック保存」機能を実装することで、Cisco IOS コンフィギュレーションとプロビジョニング アクションの運用上の堅牢性を高めます。コンフィギュレーション ロガー永続性機能を設定すると、Cisco IOS ソフトウェアはスタートアップ コンフィギュレーション全体を保存するのではなく、最後の startup-config ファイルが生成された時点から入力されたコマンドだけを保存します。

コンフィギュレーション ロガー永続性の前提条件

コンフィギュレーション ロガー永続性機能をイネーブルにするには、disk0: を構成し、ルータ上に外部フラッシュ カードを挿入する必要があります。

コンフィギュレーション ロガー永続性機能の最適な結果を実現するためには、Cisco IOS Release 12.2(33)SRA、Release 12.4(11)T、Release 12.2(33)SXH、または Release 12.2(33)SB をシステムにインストールする必要があります。

コンフィギュレーション ロガー永続性について

コンフィギュレーション ロガー永続性を使用したコンフィギュレーション ファイルの保存

Cisco IOS ソフトウェアは startup-config コンフィギュレーション ファイルを使用して、リロード全体でルータ コンフィギュレーション コマンドを保存します。この単一のファイルには、ルータのリブート時に適用する必要があるすべてのコマンドが含まれています。スタートアップ コンフィギュレーション ファイルは、**writememory** コマンドまたは **copy***url***startup-config** コマンドを入力するたびに更新されます。running-config ファイルのサイズが大きくなると、startup-config ファイルを NVRAM ファイル システムに保存する時間が長くなります。スタートアップ コンフィギュレーション ファイルは、1 MB 以上にすることができます。このサイズのファイルの場合、startup-config ファイルの 1 行を変更すると、ほとんどのコンフィギュレーションが変更されていない場合でも、全体の startup-config ファイルを再度保存する必要があります。

コンフィギュレーション ロガー永続性機能は「クイック保存」機能を実装しています。この目的は、startup-config ファイルの変更を保存する時間が保存する（startup-config ファイルと相対した）差分変更のサイズに比例する「コンフィギュレーション保存」のメカニズムを提供することです。

Cisco IOS コンフィギュレーション ロガーは、コマンドライン プロンプトで手動で入力されたすべての変更をログに記録します。この機能では、ログの変更が発生したときに登録済みのクライアントに通知します。設定ログの内容はランタイム メモリに保存されます。ログの内容は再起動後は保持されません。

コンフィギュレーション ロガー永続性機能は、リロード全体でユーザが入力したコンフィギュレーション コマンドを保持するメカニズムです。リロード後も保持されるのは、コマンドライン インターフェイス（CLI）で入力したコマンド（コンフィギュレーション モードで入力したコマンド）のみです。この機能は Cisco IOS のセキュア ファイル システムを使用して、生成されるコンフィギュレーション コマンドを保持します。

Note Cisco IOS コンフィギュレーション ロガーはシステム メッセージ ロギング（syslog）機能とは別のものです。Syslog はシステム メッセージを追跡するための一般的なログ ファシリティです。コンフィギュレーション ロガーは、CLI で入力された設定コマンドに関する情報を記録します。

保持されたコマンド

Cisco IOS コンフィギュレーション ロガーで保持されたコマンドはスタートアップ コンフィギュレーションの拡張として使用されます。これらの保存されたコマンドでは、クイック保存

機能が提供されます。startup-config ファイル全体を保存するのではなく、Cisco IOS ソフトウェアは最後の startup-config ファイル生成以降入力されたコマンドだけを保存します。

ログ出力されたコマンドだけが保持されます。コンフィギュレーション ロガーの次の追加データは保持されません。

- コマンドを出力したユーザ
- ユーザがログインした IP アドレス
- ログに記録されたコマンドのセッションおよびログ インデックス
- コマンドが入力された時刻
- 入力されたコマンドに関連付けられている前後の NVGEN 出力
- 入力されたコマンドに対するパーサーからの戻りコード

コマンドを保持する主な目的は、startup-config ファイルのクイック保存の拡張として使用することです。コンフィギュレーション コマンドに関連付けられている追加情報はクイック保存目的では有用ではありません。（監査の目的で）再起動後に追加情報を保持する必要がある場合は、次の手順を実行します。

1. Syslog へのコンフィギュレーション ロガー通知をイネーブルにします。
2. Syslog 保持機能のイネーブル化

代わりに、Cisco Networking Services、CiscoView、または、Cisco IOS デバイスを管理して標準外のストレージ ソリューションのコンフィギュレーション変更を追跡するその他のネットワーク管理システムを使用できます。

デフォルトでは、リロード時に、保持されたコマンドが startup-config ファイルの末尾に追加されます。CLI コンフィギュレーション コマンドを使用して明示的にこの動作を設定した場合にだけこれらのコマンドが適用されます。

コンフィギュレーション ロガー永続性機能を設定する方法

コンフィギュレーション ロガー永続性機能のイネーブル化

コンフィギュレーション ロガー永続性機能はクイック保存メカニズムを実装するため、スタートアップ コンフィギュレーションの変更を保存するためにかかる時間が、保存する必要がある（スタートアップ コンフィギュレーションと相対した）差分変更のサイズに比例します。Cisco IOS コンフィギュレーション ロガーで保持されたコマンドはスタートアップ コンフィギュレーションの拡張として使用されます。保存されたコマンドは、スタートアップ コンフィギュレーションの拡張として使用され、クイック保存の機能を提供します。startup-config ファイル全体

を保存するのではなく、Cisco IOS ソフトウェアは最後の startup-config ファイル生成以降入力されたコマンドだけを保存します。

コンフィギュレーション ロガー永続性機能をイネーブルにするには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **archive**
4. **log config**
5. **logging persistent auto manual**
6. **logging persistent reload**
7. **logging persistent** *size threshold*
8. **logging size** *entries*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**archive** **Example:** `Router(config)# archive`	アーカイブ コンフィギュレーション モードを開始します。
ステップ 4	**log config** **Example:** `Router(config-archive)# log config`	アーカイブ configuration-log コンフィギュレーション モードをイネーブルにします。
ステップ 5	**logging persistent auto manual** **Example:** `Router(config-archive-log-cfg)# logging persistent auto`	コンフィギュレーション ロガー永続性機能をイネーブルにします。 • **auto** キーワードは、各コンフィギュレーション コマンドが自動的に Cisco IOS セキュア ファイル システムに保存されることを指定します。 • **manual** キーワードは、コンフィギュレーション コマンドを Cisco IOS セキュア ファイル システムにオンデマンドで保存できることを指定し

	Command or Action	Purpose
		ます。これを行うには、**archivelogconfigpersistentsave** コマンドを使用する必要があります。 **Note** **loggingpersistentauto** コマンドをイネーブルにするには、disk0: を構成し、ルータに外部フラッシュ カードを挿入する必要があります。
ステップ 6	**logging persistent reload** **Example:** `Router(config-archive-log-cfg)# logging persistent reload`	続いて、リロード後に、コンフィギュレーション ロガー データベースに保存された（最後の **writememory** コマンド以降の）コンフィギュレーション コマンドを実行コンフィギュレーション ファイルに適用します。
ステップ 7	**logging persistent** *size threshold* **Example:** `Router(config-archive-log-cfg)# logging persistent size threshold`	コンフィギュレーション ロガー データベースにログ メッセージを書き込むためのディスク領域のサイズを指定します。ログ サイズがしきい値（パーセントで指定）を超えると、コンソールまたは syslog サーバーでアラートがトリガーされます。
ステップ 8	**logging size** *entries* **Example:** `Router(config-archive-log-cfg)# logging size 10`	設定ログに保持する最大エントリ数を指定します。 • 有効な値の範囲は、1 ～ 1000 です。 • デフォルト値は 100 エントリです。

コンフィギュレーション ロガー永続性機能の検証とトラブルシューティング

3 つのコマンドを使用して、設定ログの内容を検証、アーカイブ、クリアできます。トラブルシューティングでは、ステップ 4 のコマンドでデバッグをオンにします。

SUMMARY STEPS

1. **show archive log config persistent**
2. **clear archive log config persistent**
3. **archive log config persistent save**
4. **debug archive log config persistent**

DETAILED STEPS

ステップ 1 **show archive log config persistent**

このコマンドは設定ログに保持されたコマンドを表示します。このコマンドは configlet 形式で表示されます。次に、このコマンドの出力例を示します。

Example:

```
Router# show archive log config persistent
!Configuration logger persistentarchive
 log config
 logging persistent auto
 logging persistent reload
archive
 log config
logging size 10
logging console
interface loop 101
 ip address 10.1.1.1 255.255.255.0
 ip address 10.2.2.2 255.255.255.0
 no shutdown
```

ステップ 2 clear archive log config persistent

このコマンドはコンフィギュレーション ロギング永続データベース エントリをクリアします。コンフィギュレーション ロギング データベース ファイルのエントリだけが削除されます。ファイル自体は、新しいエントリを記録するために使用されるため、削除されません。このコマンドを入力すると、アーカイブログがクリアされたことを示すメッセージが表示されます。

Example:

```
Router# clear archive log config persistent
Purged the config log persist database entries successfully
Router#
```

ステップ 3 archive log config persistent save

このコマンドは Cisco IOS セキュア ファイル システムに設定ログを保存します。このコマンドを有効にするには、**archivelogconfigpersistentsave** コマンドを設定する必要があります。

ステップ 4 debug archive log config persistent

このコマンドはデバッグ機能をオンにします。デバッグがオンになっていることを示すメッセージが返されます。

Example:

```
Router# debug archive log config persistent
debug archive log config persistent debugging is on
```

コンフィギュレーション ロガー永続性機能の設定例

Cisco 7200 シリーズ ルータでのコンフィギュレーション ロガー永続性機能の設定例

この例では、各コンフィギュレーション コマンドが自動的に Cisco IOS セキュア ファイル システムに保存され、（最後の **writememory** コマンドの実行以降）コンフィギュレーション ロガーデータベースに保存されたコンフィギュレーション コマンドが実行コンフィギュレーション ファイルに適用され、設定ログに保持される最大エントリ数が 10 に設定されます。

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# archive
Router(config-archive)# log config
Router(config-archive-log-config)# logging persistent auto
configuration log persistency feature enabled. Building configuration... [OK]
Router(config-archive-log-config)# logging persistent reload
Router(config-archive-log-config)# logging persistent size 16384 threshold 10
Router(config-archive-log-config)# logging size 10
Router(config-archive-log-config)# archive log config persistent save
Router(config-archive-log-config)# end
Router#
```

その他の参考資料

次の項に、コンフィギュレーション ロガー永続性機能に関する参考資料を示します。

関連資料

関連項目	マニュアル タイトル
包括的なコマンドリファレンス情報	『*Cisco IOS Configuration Fundamentals Command Reference*』

標準

標準	タイトル
この機能がサポートする新しい規格または変更された規格はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
この機能でサポートが追加または変更された RFC はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

コンフィギュレーション ロガー永続性の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 30: コンフィギュレーション ロガー永続性の機能情報

機能名	リリース	機能情報
コンフィギュレーション ロガー永続性	12.2(33)SRA 12.4(11)T 12.2(33)SXH 12.2(33)SB Cisco IOS XE Release 3.9S	コンフィギュレーション ロガー永続性機能は「クイック保存」機能を実装することで、シスコのコンフィギュレーションとプロビジョニング アクションの運用上の堅牢性を高めます。 Cisco IOS Release 12.2(33)SRA、Release 12.4(11)T、Release 12.2(33)SXH、Release 12.2(33)SB で有効なシスコのソフトウェアは、スタートアップ コンフィギュレーション全体を保存するのではなく、最後の startup-config ファイルが生成された時点から入力されたコマンドだけを保存します。 この機能は、Cisco IOS XE Release 3.9S に統合されました。

用語集

API ：アプリケーション プログラミング インターフェイス。

CAF ：コマンド アクション機能。

CDP ：Cisco Discovery Protocol。

CSB ：コマンド ステータス ブロック。

HA ：高可用性アーキテクチャ。

MIB ：管理情報ベース。

NAF ：NVGEN アクション機能。

NVGEN ：不揮発生成。

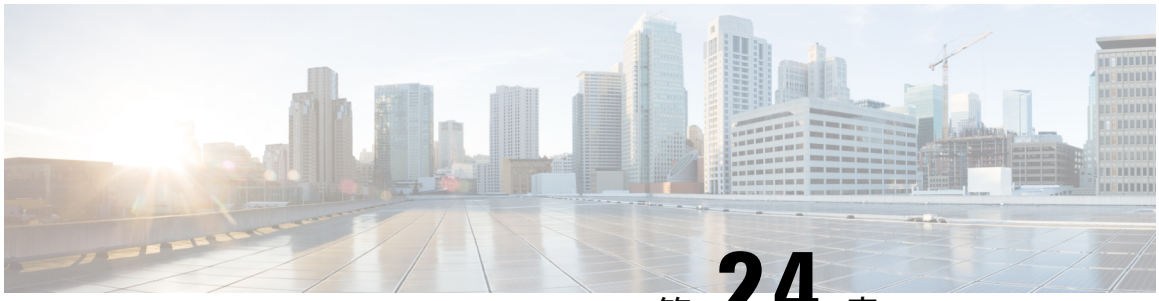
NVRAM ：不揮発性ランダム アクセス メモリ

parse chain ：Cisco IOS コマンドの構文を定義する一連の C 言語マクロ。

RP ：ルート プロセッサ。

SNMP ：簡易ネットワーク管理プロトコル。

XML ：拡張マークアップ言語。



第 24 章

ソフトウェア メンテナンス アップグレード

ソフトウェア メンテナンス アップグレード（SMU）は、システムにインストールしてパッチ修正やセキュリティ解決をリリースされたイメージに提供できるパッケージです。

ソフトウェア メンテナンス アップグレードについて

ソフトウェア メンテナンス アップグレード

ソフトウェア メンテナンス アップグレード（SMU）は、システムにインストールしてパッチ修正やセキュリティ解決をリリースされたイメージに提供できるパッケージです。

SMU パッケージはリリースごとおよびコンポーネントごとに提供され、プラットフォームに固有です。

SMU はネットワークの問題に迅速に対応できるようにするとともに、必要なテストの時間と範囲を削減するため、従来の IOS ソフトウェアには多大なメリットがあります。Cisco IOS XE プラットフォームでは SMU の互換性を内部的に検証し、互換性のない SMU はインストールできません。

すべて SMU が後続の Cisco IOS XE ソフトウェア メンテナンス リリースに統合されています。SMU は独立した自己完結型パッケージであり、前提条件や依存関係はありません。SMU はどのような順序でもインストールまたはアンインストールできます。

Cisco IOS XE Everest 16.6.1 以降、SMU は拡張メンテナンス リリースでのみ、基盤となるソフトウェア リリースのライフサイクルにわたってサポートされます。

次に、SMU をインストールする 3 つの基本ステップを示します。

- ファイルシステムへの SMU の追加
- システムでの SMU のアクティブ化
- リロード後も保持するための SMU の変更のコミット

サポートされるプラットフォーム

Cisco IOS XE Everest 16.6.1 以降、ソフトウェア メンテナンス アップグレードで次のプラットフォームがサポートされています。

- Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ（ASR1001-X、ASR1002-X、ASR1001-HX、ASR1002-HX、ASR1000-RP2、ASR1000-RP3）
- Cisco ISR 4000 シリーズ サービス統合型ルータ（ISR4351、ISR4331、ISR4431、ISR4321、ISR4451）
- Cisco CSR 1000v シリーズ クラウド サービス ルータ
- Cisco Catalyst 9300 シリーズ スイッチ
- Cisco Catalyst 9400 シリーズ スイッチ

Cisco IOS XE Dublin 17.10.1a 以降、ソフトウェア メンテナンス アップグレードで次のプラットフォームがサポートされています。

- Cisco Catalyst 8200 シリーズ エッジ プラットフォーム
- Cisco Catalyst 8300 シリーズ エッジ プラットフォーム

ソフトウェア メンテナンス アップ グレード パッケージ

SMU パッケージには、SMU が要求されている報告済みの問題のメタデータと修正が含まれています。

ソフトウェア メンテナンス アップグレードのワークフロー

SMU プロセスは、SMU Committee への要求によって開始されます。カスタマー サポートに連絡し、SMU 要求を行います。

リリース時に、SMU パッケージは次の情報とともにシスコのソフトウェア ダウンロード ページに公開されます。

- 対処済みの不具合
- 不具合の種類 - PSIRT など

SMU のリロード

SMUのタイプは、SMUのインストール後のシステムへの影響を説明します。SMUはトラフィックに影響を与えない場合もありますが、デバイスのリロードやスイッチオーバーを引き起こす可能性もあります。

システムのコールドリロードでは、オペレーティングシステムの完全なリロードが必要です。このアクションは、リロードの間（現在は最大5分間）、トラフィックフローに影響を与えます。リロードにより、SMU の一部としてインストールされている正しいライブラリとファイルですべてのプロセスが起動します。

ソフトウェア メンテナンス アップグレードの設定方法

SMU の追加、アクティブ化、コミット

SMUパッケージをインストールするには、ダウンロードしたSMUパッケージをデバイス上の該当するディレクトリにコピーします。次のコマンドを使用して、SMU を追加、アクティブ化、およびコミットします。

- **install add**：ファイルで基本的な互換性チェックを実行し、SMU パッケージがプラットフォームでサポートされていることを確認します。また、パッケージ/SMU の .sta ファイル内にエントリを追加することで、それ以降ステータスを監視または維持できるようにします。install add コマンドは、パッケージ ファイルの場所とダウンロード方法（tftp、ftp など）を入力として受け取ります。
- **install active**：互換性チェックを実行し、パッケージをインストールして、パッケージのステータスの詳細を更新します。再起動可能なパッケージの場合は、適切なポストインストールスクリプトをトリガーして必要なプロセスを再起動します。また、再起動できないパッケージの場合は、リロードをトリガーします。
- **install commit**リロードが繰り返されても保持されるようにアクティブ化の変更をコミットします。アクティブ化の後で、システムがアップしている間、または最初のリロード後にコミットできます。パッケージがアクティブになっていてもコミットされなかった場合は、最初のリロード後はアクティブの状態を保ちますが、2 回目のリロード後はアクティブ状態を保ちません。

次の設定を実行して、SMU を追加、アクティブ化、およびコミットします。

```
enable
install add file bootflash:isr4300-universalk9.BLD_
SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

show install summary  // Shows the installed SMU package as inactive package in the
command output

install activate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

show version // Shows the image version tagged with the  "SMU Patched" phrase
```

```
show install summary // Shows the installed SMU package as an active package in the
command output

install commit

show install summary // Shows the installed SMU package as a committed package in the
command output.
```

SMU のロールバック、非アクティブ化、または削除

次のコマンドを使用して、SMU をロールバック、非アクティブ化、および削除します。

- **install rollback**：デバイスを以前のインストール状態に戻します。このロールバックにはリロードが必要です。
- **install deactivate**：アクティブなパッケージを非アクティブ化し、パッケージ ステータスを更新し、再起動またはリロードするプロセスをトリガーします。
- **install remove**：すべての、または指定した非アクティブな SMU パッケージをファイル システムから削除します。

次のタスクを実行して、SMU をロールバック、非アクティブ化、および削除します。

```
enable
install rollback to committed

install deactivate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin


install remove file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin
```

ソフトウェア メンテナンス アップグレードの設定例

例：SMU の追加、アクティブ化、コミット

SMU の追加、アクティブ化、コミット

次の例は、SMU の追加、アクティブ化、コミットのワークフローを示しています。

```
Device# install add file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin
install_add: START Tue Aug  1 04:22:48 UTC 2017
install_add: Adding SMU


*Aug  1 04:22:54.492: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:22:54 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install add
bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin--- Starting SMU Add operation
 ---
Performing SMU_ADD on Active/Standby
```

```
  [R0] SMU_ADD package(s) on R0
  [R0] Finished SMU_ADD on R0
Checking status of SMU_ADD on [R0]
SMU_ADD: Passed on [R0]
Finished SMU Add operation

SUCCESS: install_add  Tue Aug  1 04:23:10 UTC 2017


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install activate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxx.SSA.smu.bin

install_activate: START Tue Aug  1 04:24:42 UTC 2017
install_activate: Activating SMU

*Aug  1 04:24:48.682: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:24:48 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install activate
bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Activate operation ---
Performing SMU_ACTIVATE on Active/Standby
  [R0] SMU_ACTIVATE package(s) on R0
    DMP package.
  [R0] Finished SMU_ACTIVATE on R0
Checking status of SMU_ACTIVATE on [R0]
SMU_ACTIVATE: Passed on [R0]
Finished SMU Activate operation

install_activate: Reloading the box to complete activation of the SMU...
install_activate will reload the system now!
Aug  1 04:25:36
Aug  1 04:25:45.742 RP0/0: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install activate
 SMU bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin

<after reload>

Device# show version
Cisco IOS XE Software, Version 16.06.01 - SMU-PATCHED
Cisco IOS Software [Everest], ISR Software (X86_64_LINUX_IOSD-UNIVERSALK9-M), Version
16.6.1, RELEASE SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2017 by Cisco Systems, Inc.
Compiled Sat 22-Jul-17 05:55 by mcpre

...

Active SMU Information:
  State (St):  C - Committed, U - Uncommitted
--------------------------------------------------------------------------------
Type  Defect_ID    Version   St Filename
--------------------------------------------------------------------------------
SMU   CSCxxXXXXX     16.6.1.0. U  isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu
--------------------------------------------------------------------------------
```

```
cisco ISR4351/K9 (2RU) processor with 7941107K/6147K bytes of memory.
Processor board ID FLM2007W0MJ
3 Gigabit Ethernet interfaces
32768K bytes of non-volatile configuration memory.
16777216K bytes of physical memory.
14659583K bytes of flash memory at bootflash:.
0K bytes of WebUI ODM Files at webui:.

Configuration register is 0x0

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# show install active
 R0 ] Active Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   U    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install commit
install_commit: START Tue Aug  1 04:48:03 UTC 2017
install_commit: Committing SMU


*Aug  1 04:48:10.042: %IOSXE-5-PLATFORM:  SIP2: Aug  1 04:48:10 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install commit--- Starting SMU Commit operation
---
Performing SMU_COMMIT on Active/Standby
  [R0] SMU_COMMIT package(s) on R0
  [R0] Finished SMU_COMMIT on R0
Checking status of SMU_COMMIT on [R0]
SMU_COMMIT: Passed on [R0]
Finished SMU Commit operation
SUCCESS: install_commit  Tue Aug  1 04:48:33 UTC 2017

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0
```

例：SMU のロールバック、非アクティブ化、または削除

例：SMU のロールバック、非アクティブ化、または削除

```
Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#show install rollback
ID      Label          Description
-----------------------------------------------
4       No Label        No Description

Device# install rollback to committed

install_rollback: START Tue Aug  1 05:00:37 UTC 2017

*Aug  1 05:00:44.038: %IOSXE-5-PLATFORM:  SIP2: Aug  1 05:00:44 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install rollbackinstall_rollback: Rolling back
SMU

This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Rollback operation ---
Performing SMU_ROLLBACK on Active/Standby
  [R0] SMU_ROLLBACK package(s) on R0
  [R0] Finished SMU_ROLLBACK on R0
Checking status of SMU_ROLLBACK on [R0]
SMU_ROLLBACK: Passed on [R0]
Finished SMU Rollback operation

install_rollback will reload the system now!
Aug  1 05:01:40.43
Aug  1 05:01:53.558 RP0/0: %INSTALL-5-INSTALL_COMPLETED_INFO: Completed install rollback
 SMU

<after reload>


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
IMG   C    16.6.1.0


//install deactivate: Deactivates an active package and triggers a process restart or a
 reload.

Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
```

```
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   C    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device# install deactivate file
bootflash:isr4300-universalk9.BLD_SMU_LATEST_20170128_040557.1.CSCxxXXXX.SSA.smu.bin
install_deactivate: START Tue Aug  1 05:28:47 UTC 2017
install_deactivate: Deactivating SMU

This operation requires a reload of the system. Do you want to proceed? [y/n]y

--- Starting SMU Deactivate operation ---
Performing SMU_DEACTIVATE on Active/Standby
  [R0] SMU_DEACTIVATE package(s) on R0
    DMP package.
  [R0] Finished SMU_DEACTIVATE on R0
Checking status of SMU_DEACTIVATE on [R0]
SMU_DEACTIVATE: Passed on [R0]
Finished SMU Deactivate operation

install_deactivate: Reloading the box to complete activation of the SMU...
install_deactivate will reload the system now!

<after reload>


Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   D    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#install commit
install_commit: START Tue Aug  1 05:39:29 UTC 2017
install_commit: Committing SMU


*Aug  1 05:39:35.222: %IOSXE-5-PLATFORM:  SIP2: Aug  1 05:39:35 install_engine.sh:
%INSTALL-5-INSTALL_START_INFO: Started install commit--- Starting SMU Commit operation
---
Performing SMU_COMMIT on Active/Standby
  [R0] SMU_COMMIT package(s) on R0
  [R0] Finished SMU_COMMIT on R0
Checking status of SMU_COMMIT on [R0]
SMU_COMMIT: Passed on [R0]
Finished SMU Commit operation

SUCCESS: install_commit  Tue Aug  1 05:39:58 UTC 2017
Completed install commit SMU

Device#show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
```

```
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0
```

//install remove: Deletes the inactive SMU file from the file system.

```
Device# show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0

Device#install remove file bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
install_remove: START Tue Aug  1 05:43:22 UTC 2017
install_remove: Removing SMU

--- Starting SMU Remove operation ---
Performing SMU_REMOVE on Active/Standby
  [R0] SMU_REMOVE package(s) on R0
  [R0] Finished SMU_REMOVE on R0
Checking status of SMU_REMOVE on [R0]
SMU_REMOVE: Passed on [R0]
Finished SMU Remove operation

SUCCESS: install_remove  Tue Aug  1 05:43:43 UTC 2017
```

//Remove inactive: Deletes all inactive packages from the file system

```
Device#show install summary
[ R0 ] Installed Package(s) Information:
State (St): I - Inactive, U - Activated & Uncommitted,
            C - Activated & Committed, D - Deactivated & Uncommitted
--------------------------------------------------------------------------------
Type  St   Filename/Version
--------------------------------------------------------------------------------
SMU   I    bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
IMG   C    16.6.1.0


Device#install remove inactive
install_remove: START Tue Aug  1 05:52:31 UTC 2017
Cleaning up unnecessary package files
  Scanning boot directory for packages ... done.
  Preparing packages list to delete ...
    isr4300-universalk9.16.06.01.SPA.bin
      File is in use, will not delete.
    packages.conf
      File is in use, will not delete.
  done.

The following files will be deleted:
[R0]:
/bootflash/isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin

Do you want to remove the above files? [y/n]y
[R0]:
Deleting file bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin ... done.
SUCCESS: Files deleted.
--- Starting Post_Remove_Cleanup ---
```

```
Performing Post_Remove_Cleanup on Active/Standby
  [R0] Post_Remove_Cleanup package(s) on R0
  [R0] Finished Post_Remove_Cleanup on R0
Checking status of Post_Remove_Cleanup on [R0]
Post_Remove_Cleanup: Passed on [R0]
Finished Post_Remove_Cleanup

SUCCESS: install_remove  Tue Aug  1 05:53:19 UTC 2017


///Show install package

Device#show install package bootflash:isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
Name: isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin
Version: 16.6.1.0.202.1500742946..Everest
Platform: ISR4300
Package Type: SMU
Defect ID: CSCxxXXXXX
Package State: Not Installed
Supersedes List: {}
SMU ID: 0
SMU Type: reload
SMU Compatible with Version: 16.6.1.0.202


//Show install log
Device#show install log
[0|install_op_boot]: START Tue Aug  1 05:34:59 Universal 2017
[0|install_op_boot(INFO, )]: SMU
/bootflash/isr4300-universalk9.16.06.01.CSCxxXXXXX.SPA.smu.bin will be activated upon
reload.
[0|install_op_boot]: END SUCCESS  Tue Aug  1 05:35:06 Universal 2017
```

ソフトウェア メンテナンス アップグレードの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

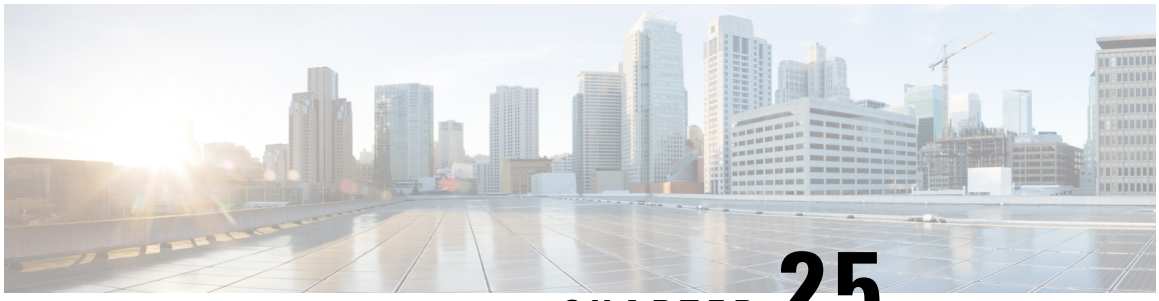
プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *31* : ソフトウェア メンテナンス アップグレードの機能情報

機能名	リリース	機能情報
ソフトウェア メンテナンス アップグレード	Cisco IOS XE Everest 16.6.1	ソフトウェア メンテナンス アップグレード（SMU）は、システムにインストールしてパッチ修正やセキュリティ解決をリリースされたイメージに提供できるパッケージです。 次のコマンドが導入または変更されました。 **install**、**show install**

第 IV 部

統合ファイルシステム Cisco IOS



CHAPTER 25

基本ファイル転送サービスの設定

基本ファイル転送サービスを使用すると、ルータを簡易ファイル転送プロトコル（TFTP）または逆アドレス解決プロトコル（RARP）サーバーとして設定、そのルータが拡張 BOOTP 要求を非同期インターフェイス経由で転送するよう設定、および rcp、rsh、FTP を設定することが可能です。

基本ファイル転送サービスの前提条件

- ユーザーには、少なくとも Cisco IOS 環境とコマンドライン インターフェイスに関する基本的な知識が必要です。
- システムでは、少なくとも最小限の設定が実行されていることが必要です。

基本ファイル転送サービスに関する制約事項

- ネットワークが稼働していて、Cisco IOS リリース 12.2 以降のリリースがすでにインストールされている必要があります。
- Cisco IOS コンフィギュレーション コマンドのいくつかは、特定のルータ プラットフォームでのみ使用可能であり、コマンド構文はプラットフォームによって異なる可能性があります。

基本ファイル転送サービスに関する情報

TFTP または RARP サーバーとしてのルータの使用

サーバーとしてだけ機能するマシンをネットワークの各セグメントに配置するのは、コストがかかり、非効率的です。しかし、すべてのセグメントにサーバーがあるのではない場合、ネットワークセグメントを超えたネットワークの操作によって相当の遅延が引き起こされることがあります。ルータを RARP または TFTP サーバーとして機能するよう設定することで、ルータの通常の機能を使用しながらコストと遅延時間を削減できます。

多くの場合、TFTP または RARP サーバーとして設定されたルータは、フラッシュ メモリから他のルータにシステムイメージまたはルータ コンフィギュレーション ファイルを提供します。リクエストのような他のタイプのサービス要求に応答するよう、ルータを設定することもできます。

TFTP サーバーとしてのルータの使用

TFTP サーバー ホストとして、ルータは TFTP 読み取り要求メッセージに応答し、ROM に含まれるシステム イメージのコピー、またはフラッシュ メモリに含まれるシステム イメージの 1 つを、要求したホストに送出します。TFTP 読み取り要求メッセージは、コンフィギュレーションで指定されたファイル名のいずれかを使用する必要があります。

Note Cisco 7000 ファミリでは、使用されるファイル名はフラッシュ メモリ内に存在するソフトウェア イメージを表している必要があります。フラッシュ メモリ内にイメージが存在しない場合、クライアント ルータはデフォルトとしてサーバーの ROM イメージをブートします。

フラッシュ メモリは、ネットワーク内の他のネットワークの TFTP ファイル サーバーとして使用できます。この機能により、リモートのルータをフラッシュ サーバー メモリ内に存在するイメージを使用してブートすることが可能になります。

シスコ デバイスの中には、TFTP サーバーとして、さまざまなフラッシュ メモリ位置（**bootflash:**、**slot0:**、**slot1:**、**slavebootflash:**、**slaveslot0:**、または **slaveslot1:**）から 1 つを選択できるものもあります。

RARP サーバーとしてのルータの使用

逆アドレス解決プロトコル（RARP）は、MAC（物理）アドレスをもとに IP アドレスを検索する方法をそなえた、TCP/IP スタックのプロトコルです。ブロードキャスト Address Resolution Protocol（ARP）の逆であるこの機能により、ネットワーク層の特定の IP アドレスに対応する MAC レイヤ アドレスをホストが動的に検出できます。RARP はさまざまなシステムをディスクなしで起動させることを可能にします（たとえば、クライアントとサーバーが別のサブネットにあるネットワークの Sun ワークステーションや PC のように、起動時点では IP アドレスが

わからないディスクレス ワークステーション）。RARP は、MAC レイヤから IP アドレスへのマッピングのキャッシュされたエントリの表を持つRARP サーバーの存在に依存しています。

Cisco ルータは RARP サーバーとして設定できます。この機能で、Cisco IOS ソフトウェアは RARP 要求に応答することができます。

Rsh および rcp 用ルータの使用

リモート シェル（rsh）により、コマンドをリモートで実行できるようになります。リモート コピー（RCP）を使用すると、ユーザーはネットワーク上のリモート ホストやサーバーに存在するファイル システムへのファイル コピーや、ファイル システムからのコピーが行えます。シスコの rsh および rcp の実装は、業界標準の実装と相互運用できます。シスコでは、rsh と rcpの両方を示すために、省略形 RCMD（Remote Command、リモート コマンド）を使用します。

RCMD 送信の発信元インターフェイス

RCMD（rsh と rcp）通信の発信元インターフェイスを指定できます。たとえば、RCMD接続でループバック インターフェイスをルータから送信されるすべてのパケットの送信元アドレスとして使用するように、ルータを設定できます。source-interface を指定するのは、ループバック インターフェイスの指定に最も一般的に使用される方法です。これにより、RCMD通信にパーマネント IP アドレスを関連付けることができます。パーマネント IP アドレスを持つことは、セッションの識別に役立ちます（リモートデバイスがセッションの間パケットの送信元を一貫して識別できます）。「既知の」IPアドレスも、アドレスを含めてリモートデバイスにアクセス リストを作成できるよう、セキュリティの目的で使用できます。

RCMD の DNS 逆引き参照について

基本的なセキュリティ チェックとして、Cisco IOS ソフトウェアでは、リモート コマンド（RCMD）アプリケーション（rsh および rcp）の DNS を使用してクライアント IP アドレスの逆引き参照を実行します。このチェックは、ホスト認証プロセスを使用して実行されます。

イネーブルにされている場合、システムは要求元のクライアントのアドレスを記録します。アドレスは、DNS を使用してホスト名にマッピングされます。次に、そのホスト名の IP アドレスに対する DNS リクエストが行われます。受け取った IP アドレスが、元の要求元アドレスと照合されます。そのアドレスが、DNS から受信したアドレスのいずれにも一致しない場合、RCMD 要求は処理されません。

この逆引き参照は、「スプーフィング」に対する保護を促進するためのものです。ただし、このプロセスでは当該 IP アドレスが有効かつルーティング可能なアドレスであることを確認するのみであり、ハッカーは引き続き既知のホストの有効な IP アドレスをスプーフィングできるということに注意してください。

rsh の導入

rsh（リモート シェル）を使用すると、アクセス可能なリモート システム上でコマンドを実行できます。**rsh** コマンドを発行すると、リモート システム上でシェルが起動します。シェルに

より、ターゲット ホストにログインすることなくリモート システム上でコマンドを実行できます。

そのシステムへの接続、ルータ、アクセス サーバー、さらにコマンド実行後の切断も、rsh を使えば必要ありません。たとえば、rsh を使用すれば、ターゲット デバイスへの接続やコマンドの実行、切断といった手順なしに、リモートで他のデバイスのステータスを見ることができます。この機能は、多数の異なるルータの統計情報を見る場合に役立ちます。rsh を有効化するコンフィギュレーション コマンドは、「remote command（リモート コマンド）」の略語である「rcmd」を使用します。

rsh セキュリティの維持

rshが動作しているリモートシステム（UNIXホストなど）にアクセスするためには、そのユーザーがリモートからそのシステムでコマンドを実行する権限を与えられていることを示すエントリが、システムの *.rhosts* ファイルまたはそれに相当するものに存在している必要があります。UNIX システムでは、*.rhosts* ファイルはシステムのコマンドをリモートで実行できるユーザーを特定します。

ルータ上の rsh サポートを有効化すると、リモート システム上のユーザーがコマンドを実行できるようになります。しかし、シスコの rsh の実装は、*.rhosts* ファイルをサポートしていません。その代わりに、rshを使用してリモートでコマンドを実行しようとするユーザーによるルータへのアクセスを制御するため、ローカルの認証データベースを設定する必要があります。ローカルの認証データベースは、UNIX *.rhosts* ファイルに似ています。認証データベースで設定する各エントリでは、ローカル ユーザー、リモート ホスト、およびリモート ユーザーを特定します。

rcp の導入

リモート コピー（rcp）コマンドは、リモート システムの rsh サーバー（またはデーモン）に依存します。RCP を使用してファイルをコピーする場合、TFTP と異なり、ファイル配布用のサーバーを作成する必要はありません。必要なのは、リモート シェル（rsh）をサポートするサーバーへのアクセスだけです（ほとんどの UNIX システムが rsh をサポートしています）。ある場所から別の場所にファイルをコピーするため、コピー元のファイルに対する読み取り権限とコピー先のディレクトリに対する書き込み権限が必要です。コピー先ファイルが存在しない場合は、rcp により作成されます。

シスコの rcp 実装は UNIX の rcp 実装（ネットワーク上のシステム間でファイルをコピー）の関数をエミュレートしたものですが、シスコのコマンド構文は UNIX の rcp コマンド構文とは異なります。Cisco IOS ソフトウェアには、rcpをトランスポートメカニズムとして使用する一群のコピー コマンドがあります。これらの rcp コピー コマンドは Cisco IOS TFTP コピー コマンドと類似していますが、より高速なパフォーマンスと信頼性の高いデータ配信を可能にする代替案になっています。このような改善が可能なのは、rcp トランスポート メカニズムが組み込まれており、Transmission Control Protocol/Internet Protocol（TCP/IP）スタックを使用しているためです。rcp コマンドを使用して、ルータからネットワークサーバー（またはその逆）へシステム イメージおよびコンフィギュレーション ファイルをコピーできます。

また、rcp サポートをイネーブルにすることで、リモート システムのユーザーによるルータへの、またはルータからのファイル コピーを許可できます。

/user キーワードおよび引数を指定しない場合、Cisco IOS ソフトウェアはデフォルトのリモートユーザー名を送信します。リモートユーザー名のデフォルト値として、現在の TTY プロセスと関連付けられたリモートユーザー名が有効である場合、ソフトウェアはそのユーザー名を送信します。TTY リモート ユーザー名が無効な場合、ソフトウェアはリモートとローカルのユーザー名の両方にルータのホスト名を使用します。

rcp 要求の送信側リモート クライアントの設定

rcp プロトコルでは、クライアントは rcp 要求ごとにリモート ユーザー名をサーバーに送信する必要があります。rcp を使用してコンフィギュレーション ファイルをサーバーからルータへコピーする場合、Cisco IOS ソフトウェアは次のリストから、最初の有効なユーザー名を送信します。

1. **iprcmdremote-username** コマンドで設定されたユーザー名（このコマンドが設定されている場合）。
2. 現在の TTY（端末）プロセスに関連付けられているリモートユーザー名。たとえば、ユーザーが Telnet を介してルータに接続されており、**username** コマンドを介して認証された場合は、リモート ユーザー名として Telnet ユーザー名がルータ ソフトウェアによって送信されます。

Note シスコ製品では、TTY がサーバーへのアクセスに広く使用されています。TTY の概念は、UNIX に由来します。UNIX システムでは、各物理デバイスがファイル システムで表現されます。端末は *tty* デバイスと呼ばれます（tty は、UNIX 端末の *teletype* が元になった省略形です）。

1. ルータのホスト名。

rcp を使用した **boot** コマンドで、ソフトウェアはルータ ホスト名を送信します。リモートユーザー名の明示的な設定はできません。

rcp コピー要求が正常に実行されるためには、ネットワーク サーバー上でリモート ユーザー名のアカウントが定義されている必要があります。

サーバーに書き込む場合、ルータ上のユーザーからの rcp 書き込み要求を受け入れるように、rcp サーバーを適切に設定する必要があります。UNIX システムの場合は、rcp サーバー上のリモートユーザーの *.rhosts* ファイルに対しエントリを追加する必要があります。たとえば、ルータに次の設定行が含まれているとします。

```
hostname Rtr1
ip rcmd remote-username User0
```

そのルータの IP アドレスを Router1.company.com と変換するとすれば、rcp サーバーの User0 の *.rhosts* ファイルは、次の行を含んでいる必要があります。

```
Router1.company.com Rtr1
```

詳細については、ご使用の RCP サーバーのマニュアルを参照してください。

このサーバーがディレクトリ構造をとっている場合、コンフィギュレーションファイルまたはイメージは、サーバー上のリモートユーザー名と関連付けられたディレクトリに関連して書き込まれるか、そのディレクトリからコピーされます。サーバー上で使用するディレクトリを指定するには、**ip rcmd remote-username** コマンドを使用します。たとえば、システムイメージがサーバー上のあるユーザーのホーム ディレクトリに存在する場合、そのユーザーの名前をリモート ユーザー名として指定します。

ファイル サーバーとして使用されているパーソナル コンピュータにコンフィギュレーションファイルをコピーする場合、このコンピュータでは rsh がサポートされている必要があります。

FTP 接続用ルータの使用

ネットワーク上のシステム間で File Transfer Protocol（FTP）を使用してファイルを転送するよう、ルータを設定できます。Cisco IOS に実装された FTP により、次の FTP 特性を設定できます。

- パッシブ モード FTP
- ユーザー名
- パスワード
- IP アドレス

基本ファイル転送サービスの設定方法

TFTP サーバーとしてのルータの使用の設定

ルータが TFTP サーバーとして使用されるよう設定するには、このセクションのタスクを実行します。

Before you begin

TFTP 機能の実装前に、サーバーとクライアント ルータは互いに到達可能である必要があります。**ping***a.b.c.d* コマンドを使用して（*a.b.c.d* はクライアント デバイスのアドレス）サーバーとクライアント ルータとの接続をテストし（いずれかの方向で）、この接続を確認します。**ping** コマンドが発行されると、接続されたことが、一連の感嘆符（!）によって表示されます。接続に失敗した場合は、一連のピリオド（.）に加えて [timed out] または [failed] が表示されます。接続に失敗し、インターフェイスを再設定する場合は、フラッシュ サーバーとクライアント ルータとの間の物理的な接続をチェックし、ping を再実行します。

接続をチェックした後、TFTP ブート可能イメージがサーバー上に存在することを確認します。これは、クライアント ルータがブートするシステム ソフトウェア イメージです。最初のクライアント ブートの後で確認できるように、そのソフトウェア イメージの名前を記録しておきます。

Caution すべての機能を使用するために、クライアントに送信されるソフトウェアイメージは、クライアントルータにインストールされたROMソフトウェアと同一のタイプのものである必要があります。たとえば、サーバーにはX.25ソフトウェアがあり、クライアントのROMにはX.25ソフトウェアがない場合、フラッシュメモリ内にあるサーバーのイメージからブートしてからも、クライアントにはX.25の機能がありません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. 次のいずれかを実行します。
   - **tftp-server flash** [*partition-number*:]*filename1* [**alias***filename2*] [*access-list-number*]
   - **tftp-server flash** *device* **:** *filename*（Cisco 7000 ファミリのみ）
   - **tftp-server flash** [*device*:][*partition-number*:]*filename* （Cisco 1600 シリーズと Cisco 3600 シリーズのみ）
   - **tftp-server rom alias** *filename1* [*access-list-number*]
4. **end**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
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	次のいずれかを実行します。 • **tftp-server flash** [*partition-number*:]*filename1* [**alias***filename2*] [*access-list-number*] • **tftp-server flash** *device* **:** *filename*（Cisco 7000 ファミリのみ） • **tftp-server flash** [*device*:][*partition-number*:]*filename* （Cisco 1600 シリーズと Cisco 3600 シリーズのみ） • **tftp-server rom alias** *filename1* [*access-list-number*] **Example:**	読み取り要求の応答として送信されるシステム イメージを指定します。複数行を入力して複数のイメージを指定することができます。

	Command or Action	Purpose
	`Device(config)# tftp-server flash version-10.3 22`	
ステップ 4	**end** **Example:** `Device(config)# end`	コンフィギュレーションセッションを終了して、特権 EXEC モードに戻ります。
ステップ 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	実行コンフィギュレーションをスタートアップ コンフィギュレーション ファイルに保存します。

例

次の例では、フラッシュ メモリ ファイル *version-10.3* の TFTP 読み取りリクエストへの応答として、システムは TFTP を使用してこのファイルのコピーを送信できます。要求送出ホストはアクセス リスト 22 でチェックされます。

```
tftp-server flash version-10.3 22
```

次の例では、ROM イメージ *gs3-k.101* ファイルについての TFTP 読み取り要求への応答として、システムは TFTP を使用して *gs3-k.101* ファイルのコピーを送信できます。

```
tftp-server rom alias gs3-k.101
```

次の例では、TFTP 読み取り要求への応答として、ルータがフラッシュ メモリ内のファイル *gs7-k.9.17* のコピーを送信します。クライアント ルータはアクセス リスト 1 で指定されたネットワーク内に存在している必要があります。したがって、この例では、ネットワーク 172.16.101.0 にあるすべてのクライアントがファイルへのアクセスを許可されます。

```
Server# configure terminal

Enter configuration commands, one per line. End with CTRL/Z
Server(config)# tftp-server flash gs7-k.9.17 1

Server(config)# access-list 1 permit 172.16.101.0 0.0.0.255

Server(config)# end

Server# copy running-config startup-config

[ok]
Server#
```

トラブルシューティング

TFTP セッションには障害が発生することがあります。TFTP は TFTP セッション障害の原因判別のために、次の特別な文字を生成します。

- 文字「E」は、TFTP サーバーがエラーを含むパケットを受信したことを示します。
- 文字「O」は、TFTP サーバーがシーケンスに合わないパケットを受信したことを示します。
- ピリオド（.）はタイムアウトを示します。

転送中の不適当な遅延を診断するために、この出力が役立ちます。トラブルシューティングの手順については、マニュアル『*Internetwork Troubleshooting Guide*』を参照してください。

クライアント ルータの設定

最初にサーバーからシステムイメージをロードし、次にバックアップとして、サーバーからのロードに失敗した場合に自身の ROM イメージをロードするようクライアント ルータを設定するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no boot system**
4. **boot system** [**tftp**] *filename* [*ip-address*]
5. **boot system rom**
6. **config-register** *value*
7. **end**
8. **copy running-config startup-config**
9. **reload**

DETAILED STEPS

	Command or Action	Purpose
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	**no boot system** **Example:**	（任意）これまでの **bootsystem** 文をすべてコンフィギュレーション ファイルから削除します。

	Command or Action	Purpose
	`Device(config)# no boot system`	
ステップ 4	**boot system** [**tftp**] *filename* [*ip-address*] **Example:** `Device(config)# boot system c5300-js-mz.121-5.T.bin 172.16.1.1`	クライアントルータがサーバーからシステムイメージをロードするよう指定します。
ステップ 5	**boot system rom** **Example:** `Device(config)# boot system rom`	クライアントルータがサーバーからのロードに失敗した場合に、自身の ROM イメージをロードするように指定します。
ステップ 6	**config-register** *value* **Example:** `Device(config)# config-register 0x010F`	クライアント ルータがネットワーク サーバーからシステム イメージをロードできるよう、コンフィギュレーション レジスタを設定します。
ステップ 7	**end** **Example:** `Device(config)# end`	グローバル コンフィギュレーション モードを終了します。
ステップ 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	コンフィギュレーションファイルをスタートアップコンフィギュレーションに保存します。
ステップ 9	**reload** **Example:** `Device# reload`	（任意）変更を有効にするため、ルータをリロードします。

例

次の例では、ルータは指定の TFTP サーバーからブートするよう設定されます。

```
Client# configure terminal

Enter configuration commands, one per line. End with CTRL/Z
Client(config)# no boot system

Client(config)# boot system c5300-js-mz.121-5.T.bin 172.16.1.1

Client(config)# boot system rom

Client(config)# config-register 0x010F
```

```
Client(config)# end

Client# copy running-config startup-config

[ok]
Client# reload
```

この例では、**nobootsystem** コマンドによって、現在コンフィギュレーション メモリ内にある他の **bootsystem** コマンドがすべて無効化され、このコマンドの後に入力される **bootsystem** コマンドが先に実行されるようになります。2 番目のコマンドである **bootsystem***filename address* は、クライアント ルータに対し、IP アドレスが 172.16.111.111 の TFTP サーバーにあるファイル c5300-js-mz.121-5.T.bin を探すよう指示しています。これが失敗した場合、クライアント ルータは、ネットワーク障害が生じた場合のバックアップとして含まれている **bootsystemrom** コマンドに応答して、自身のシステム ROM からブートします。**copyrunning-configstartup-config** コマンドは、コンフィギュレーションをスタートアップ コンフィギュレーションへコピーし、**reload** コマンドがシステムをブートします。

Note サーバーからブートするためのシステム ソフトウェアは、サーバーのフラッシュ メモリ内に存在している必要があります。フラッシュ メモリにない場合、クライアント ルータはサーバーのシステム ROM からブートします。

次の例に、ルータの再起動後に **showversion** コマンドを実行した場合の出力例を示します。

```
Device> show version
Cisco Internetwork Operating System Software
Cisco IOS (tm) 5300 Software (C5300-JS-M), Version 12.1(5)T,  RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2000 by Cisco Systems, Inc.
Compiled Sat 11-Nov-00 03:03 by joe
Image text-base: 0x60008958, data-base: 0x611C6000
ROM: System Bootstrap, Version 11.2(9)XA, RELEASE SOFTWARE (fc2)
BOOTFLASH: 5300 Software (C5300-BOOT-M), Version 12.0(7)T,  RELEASE SOFTWARE (f)
Router uptime is 8 weeks, 4 days, 22 hours, 36 minutes
System returned to ROM by power-on
System restarted at 00:37:38 UTC Thu Feb 22 2001
System image file is "flash:c5300-js-mz.121-5.T.bin"
.
.
.
Configuration register is 0x010F
```

この例の重要情報は、最初の行の「Cisco IOS (tm)..」と「System image file....」で始まる行とに含まれています。「Cisco IOS (tm)...」という行では、NVRAM のオペレーティングシステムのバージョンが表示されています。「System image file....」という行は、TFTP サーバからロードされたシステム イメージのファイル名を表示しています。

次の作業

システムをリロードしたら、**showversion** EXEC モード コマンドを使用して、目的とするイメージでシステムがブートしたことを確認する必要があります。

Caution 次の例にあるとおり、**nobootsystem** コマンドを使用すると、現在クライアント ルータのシステム コンフィギュレーションにある他のブート システム コマンドがすべて無効化されます。次に進む前に、バックアップ コピーの目的でクライアント ルータに格納されたシステム コンフィギュレーションを先に TFTP ファイル サーバーに保存するか（アップロードするか）を決定します。

RARP サーバーとしてのルータの設定

ルータを RARP サーバーに設定するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type* [*slot*/]*port*
4. **ip rarp-server** *ip-address*

DETAILED STEPS

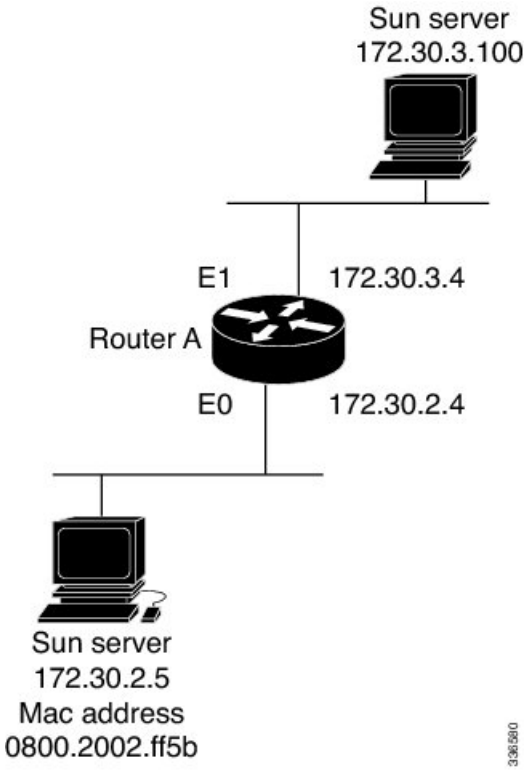
	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Device> enable`	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** **Example:** `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**interface** *type* [*slot*/]*port* **Example:** `Device(config)# interface Gigabitethernet 0/0`	RARP サービスを設定するインターフェイスを指定し、指定したインターフェイスのインターフェイス コンフィギュレーション モードを開始します。
ステップ 4	**ip rarp-server** *ip-address* **Example:** `Device(config-if)# ip rarp-server 172.30.3.100`	ルータの RARP サービスを有効化します。

例

以下の図は、ルータがディスクレス ワークステーションの RARP サーバーとして機能するネットワークの設定を示しています。この例では、Sun ワークステーションは自

身の MAC（ハードウェア）アドレスを IP アドレスに解決するために SLARP 要求を送信し、要求はルータによって Sun サーバーへ転送されます。

***Figure 9: RARP* サーバーとしてのルータの設定**



ルータ A は次のように設定されています。

```
! Allow the router to forward broadcast portmapper requests
ip forward-protocol udp 111
! Provide the router with the IP address of the diskless sun
arp 172.30.2.5 0800.2002.ff5b arpa
interface Gigabitethernet 0/0
! Configure the router to act as a RARP server, using the Sun Server's IP
! address in the RARP response packet.
ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
ip helper-address 172.30.3.100
```

Sun のクライアントとサーバーの IP アドレスには、現在の SunOS デーモン *rpc.bootparamd* での制限により、同じメジャー ネットワーク番号を使用する必要があります。

次の例では、アクセス サーバーが RARP サーバーとして機能するよう設定されています。

```
! Allow the access server to forward broadcast portmapper requests
 ip forward-protocol udp 111
! Provide the access server with the IP address of the diskless sun
```

```
 arp 172.30.2.5 0800.2002.ff5b arpa
 interface Gigabitethernet 0/0
! Configure the access server to act as a RARP server, using the Sun Server's
! IP address in the RARP response packet.
 ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
 ip helper-address 172.30.3.100
```

rsh および rcp を使用するためのルータの設定

RCMD 送信での送信元インターフェイスの指定

RCMD 接続でルータから送信されるすべてのパケットの送信元アドレスとしてループバック インターフェイスを使用するようにルータを設定するには、このセクションのタスクを実行することにより、RCMD 通信に関連付けられているインターフェイスを指定します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd source-interface** *interface-id*

DETAILED STEPS

	Command or Action	Purpose
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	**ip rcmd source-interface** *interface-id* **Example:** `Device(config)# ip rcmd source-interface`	rsh と rcp のすべての送信トラフィックにラベル付けするために使用するインターフェイス アドレスを指定します。

RCMD の DNS 逆引き参照の無効化

rcmd の DNS 逆引き参照はデフォルトで有効化されています。このセクションのタスクを実行することにより、RCMD（rsh および rcp）アクセスの DNS チェックを無効化できます。

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **no ip rcmd domain-lookup**

DETAILED STEPS

	Command or Action	Purpose
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	**no ip rcmd domain-lookup** **Example:** `Device(config)# no ip rcmd domain-lookup`	リモート コマンド（RCMP）アプリケーション（rsh および rcp）の Domain Name Service（DNS）逆ルックアップ機能をディセーブルにします。

リモート ユーザーが rsh を使用してコマンドを実行できるようにするためのルータの設定

リモート ユーザーが rsh を使用してコマンドを実行できるようにルータを設定するには、このセクションのタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]]
4. **ip rcmd rsh-enable**

DETAILED STEPS

	Command or Action	Purpose
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。

	Command or Action	Purpose
ステップ 3	**ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]] **Example:** `Device(config)# ip rcmd remote-host Router1 172.16.101.101 netadmin4 enable`	ローカル認証データベースで、rsh コマンド実行を許可するリモートユーザーそれぞれにエントリを作成します。
ステップ 4	**ip rcmd rsh-enable** **Example:** `Device(config)# ip rcmd rsh-enable`	ソフトウェアの受信rshコマンドのサポートをイネーブルにします。 **Note** ソフトウェアの受信 rsh コマンドのサポートを無効化するには、**noiprcmdrsh-enable** コマンドを使用します。 **Note** 受信 rsh コマンドのサポートがディセーブルにされた場合でも、リモート シェル プロトコルをサポートする他のルータおよびネットワーク上のUNIXホストで実行される rsh コマンドを発行することができます。

例

次に、リモート ユーザーのために 2 つのエントリを認証データベースに追加し、リモート ユーザーからの rsh コマンドをサポートするようルータをイネーブルにする例を示します。

```
ip rcmd remote-host Router1 172.16.101.101 rmtnetad1
ip rcmd remote-host Router1 172.16.101.101 netadmin4 enable
ip rcmd rsh-enable
```

名前が *rmtnetad1* というユーザーと *netadmin4* というユーザーはいずれも、リモート ホストの IP アドレス 172.16.101.101 に存在します。ユーザーはいずれも同じリモート ホスト上にいますが、各ユーザーに対して一意のエントリを含める必要があります。ルータを rsh に対して有効化すると、いずれのユーザーも、そのルータに接続してリモートで rsh コマンドを実行できるようになります。*netadmin4* という名前のユーザーは、ルータ上での特権 EXEC モード コマンドの実行を許可されます。認証データベース上のいずれのエントリも、ローカルのユーザー名として、ルータのホスト名 *Router1* を使用します。最後のコマンドで、リモート ユーザーが発行した rsh コマンドのルータでのサポートを有効化します。

rsh を使用したリモートでのコマンド実行

rshを使用してリモートからネットワーク サーバーでコマンドを実行するには、ユーザーEXECモードで次のコマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **rsh** {*ip-address* | *host* } [/**user***username*] *remote-command*

DETAILED STEPS

	Command or Action	Purpose
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	**rsh** {*ip-address* | *host* } [/**user***username*] *remote-command* **Example:** `Device# rsh mysys.cisco.com /user sharon ls -a`	rshを使用してリモートからコマンドを実行します。

例

次の例では、mysys.cisco.com 上で、ユーザー sharon のホーム ディレクトリから rsh を使用して「ls -a」コマンドを実行します。

```
Device# enable
Device# rsh mysys.cisco.com /user sharon ls -a
.
..
.alias
.cshrc
.emacs
.exrc
.history
.login
.mailrc
.newsrc
.oldnewsrc
.rhosts
.twmrc
.xsession
jazz
Device#
```

リモート ユーザーからの rcp 要求受け入れのためのルータ設定

Cisco IOS ソフトウェアが受信 rcp 要求をサポートするよう設定するには、グローバル コンフィギュレーション モードで次のコマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]]
4. **ip rcmd rcp-enable**

DETAILED STEPS

	Command or Action	Purpose
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	**ip rcmd remote-host** *local-username* {*ip-address* | *host* } *remote-username* [**enable**[*level*]] **Example:** `Device(config)# ip rcmd remote-host Router1 172.16.101.101 netadmin3`	ローカルの認証データベースで、rcp コマンドの実行を許可されているリモートユーザーそれぞれにエントリを作成します。 **Note** ソフトウェアの受信 rcp 要求のサポートを無効化するには、**noiprcmdrcp-enable** コマンドを使用します。 **Note** 受信 rcp 要求のサポートをディセーブルにした場合でも、rcp コマンドを使用してリモートサーバーへイメージをコピーできます。受信 rcp 要求のサポートは、発信 rcp 要求を扱う際の機能とは異なっています。
ステップ 4	**ip rcmd rcp-enable** **Example:** `Device(config)# ip rcmd rcp-enable`	ソフトウェアの受信 rcp 要求のサポートをイネーブルにします。

例

次の例に、認証データベースにリモート ユーザー用の 2 つのエントリを追加してから、ソフトウェアでリモートユーザーからのリモートコピー要求のサポートを有効化する方法を示します。IP アドレス 172.16.15.55 のリモート ホストの *netadmin1* というユーザーと、IP アドレス 172.16.101.101 のリモートホストの *netadmin3* というユーザーは両方とも、ルータへの接続、およびルータが rcp サポートをイネーブル化した後にリモートから rcp コマンドを実行することを許可されます。認証データベース上のいずれのエントリも、ローカルのユーザー名として、ホスト名 *Router1* を使用します。最後のコマンドで、リモート ユーザーからの rcp 要求のルータでのサポートをイネーブルにします。

```
ip rcmd remote-host Router1 172.16.15.55 netadmin1
ip rcmd remote-host Router1 172.16.101.101 netadmin3
ip rcmd rcp-enable
```

rcp 要求の送信側リモートの設定

rcp 要求で送信されるデフォルトのリモート ユーザー名を上書きするには、グローバル コンフィギュレーション モードで次のコマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rcmd remote-username** *username*

DETAILED STEPS

	Command or Action	Purpose
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	**ip rcmd remote-username** *username* **Example:** `Device(config)# ip rcmd remote-username sharon`	リモート ユーザー名を指定します。 **Note** リモート ユーザー名を削除してデフォルト値に戻すには、**noiprcmdremote-username** コマンドを使用します。

FTP 接続使用時のルータ設定

ネットワークのシステム間で File Transfer Protocol（FTP）を使用してファイルを転送するようルータを設定して、このセクションのタスクである FTP 特性の設定を完了するには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ftp username** *string*
4. **ip ftp password** [*type*] *password*
5. 次のいずれかを実行します。
   - **ip ftp passive**
   -
   -
   - **no ip ftp passive**
6. **ip ftp source-interface** *interface*

DETAILED STEPS

	Command or Action	Purpose
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	**ip ftp username** *string* **Example:** `Device(config)# ip ftp username zorro`	FTP 接続で使用されるユーザー名を指定します。
ステップ 4	**ip ftp password** [*type*] *password* **Example:** `Device(config)# ip ftp password sword`	FTP 接続で使用されるパスワードを指定します。
ステップ 5	次のいずれかを実行します。 • **ip ftp passive** • •	パッシブ モード FTP 接続のみを使用するようルータを設定します。 または

	Command or Action	Purpose
	• **no ip ftp passive** **Example:** `Device(config)# ip ftp passive`	すべてのタイプのFTP 接続（デフォルト）を許可します。
ステップ 6	**ip ftp source-interface** *interface* **Example:** `Device(config)# ip ftp source-interface to1`	FTP 接続の発信元 IP アドレスを指定します。

例

次の例に、Cisco IOS の FTP 機能を使用してコア ダンプを取り込む方法を示します。ルータはログイン名 zorro とパスワード sword により IP アドレス 192.168.10.3 でサーバーにアクセスします。デフォルトのパッシブ モード FTP が使用され、コア ダンプが発生するルータ上のトークン リング インターフェイス to1 を使用してサーバーへのアクセスが行われます。

```
ip ftp username zorro
ip ftp password sword
ip ftp passive
ip ftp source-interface to1
! The following command allows the core-dump code to use FTP rather than TFTP or RCP
exception protocol ftp
! The following command identifies the FTP server
! 192.168.10.3 crashes
exception dump 192.168.10.3
```

CHAPTER 26

HTTP または HTTPS を使用したファイルの転送

Cisco IOS Release 12.4 には、Cisco IOS ソフトウェアベースのデバイスとリモート HTTP サーバーとの間で HTTP/HTTP セキュア（HTTPS）プロトコルを使用してファイル転送を行う機能があります。ファイルシステムプレフィックスを使用する Cisco IOS コマンドライン インターフェイス（CLI）コマンド（**copy** コマンドなど）で、送信元や宛先に HTTP や HTTPS を指定できるようになりました。

HTTP または HTTPS を使用したファイル転送の前提条件

リモート HTTP サーバーへ、またはサーバーからファイルをコピーするためには、使用するシステムが HTTP クライアント機能をサポートしている必要があります。この機能はほとんどの Cisco IOS ソフトウェア イメージに統合されています。HTTP クライアントはデフォルトでイネーブルになっています。現在のシステムが HTTP クライアントをサポートしているかどうかを判断するには、**show ip http client all** コマンドを発行します。このコマンドを実行できれば、HTTP クライアントがサポートされています。

埋め込み HTTP クライアントのオプション設定と HTTPS クライアントのためのコマンドも存在しますが、HTTP または HTTPS を使用したファイル転送機能を使用する場合は、デフォルトの設定で十分です。HTTP または HTTP クライアントのオプション特性の設定については、「関連資料」セクションを参照してください。

HTTP または HTTPS を使用したファイル転送に関する制約事項

- **copy** コマンドの既存の制限（ネットワーク間のコピーができないなど）は、HTTP または HTTPS を使用したファイル転送機能でも有効です。

Note Cisco IOS リリース 12.4T の **copy** コマンドは、古いバージョンの Apache サーバー ソフトウェアと組み合わせて動作させることができません。copy コマンドを使用するには、Apache サーバー ソフトウェアをバージョン 2.0.49 以降にアップグレードする必要があります。

- Cisco リリース 17.3.1 以降、TLS 接続は、ホスト名が証明書のサブジェクト代替名（SAN）または共通名（CN）と一致する場合にのみ確立されます。サーバーがこれらの期待を満たさず、無効な属性を送信した場合、TLS 接続が確立されないため、SSL ハンドシェイクは拒否されます。したがって、HTTPS コピーは成功しません。

HTTP または HTTPS を使用したファイル転送に関する情報

HTTP または HTTPS を使用してファイルを転送するには、次の概念について理解しておく必要があります。

HTTP または HTTPS を使用したファイル転送機能は、Cisco IOS の **copy** コマンドおよびコマンドライン インターフェイスを使用して、リモート サーバーからローカル ルーティング デバイスへ、またはその逆の方向に、Cisco IOS イメージ ファイル、コア ファイル、コンフィギュレーション ファイル、ログ ファイル、スクリプトなどのファイルをコピーする機能を提供します。HTTP コピー操作は、FTP や TFTP など、他のリモート ファイル システムからのコピーと同じように動作します。

HTTP コピー操作では、HTTP セキュア転送に組み込み HTTPS クライアントを使用できるので、Public Key Infrastructure（PKI）のコンテキスト内で安全かつ認証されたファイル転送が実現されます。

HTTP または HTTPS を使用したファイル転送方法

ここでは、次の手順について説明します。

Note 接続にユーザー名とパスワードを要求するサーバーとのHTTP接続では、HTTPを使用したファイル転送機能を使用するために、ユーザー名とパスワードの指定が必要な場合があります。デフォルト設定を使用できますが、カスタム接続特性を指定するコマンドも使用できます。接続とファイルの監視とメンテナンスのためのコマンドも準備されています。

ファイル転送の HTTP 接続特性の設定

HTTP ファイル転送用に、デフォルト値が設定されています。次の作業では、接続特性を使用中のネットワーク用にカスタマイズし、使用するユーザー名とパスワード、接続プライオリティ、リモート プロキシ サーバー、発信元インターフェイスを指定します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http client connection** {**forceclose** | **idletimeout***seconds* | **timeout***seconds*}
4. **ip http client username** *username*
5. **ip http client password** *password*
6. **ip http client proxy-server** {*proxy-name* | *ip-address*} [**proxy-port***port-number*]
7. **ip http client source-interface** *interface-id*
8. **do copy running-config startup-config**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**ip http client connection** {**forceclose** | **idletimeout***seconds* | **timeout***seconds*} **Example:** `Router(config)# ip http client connection timeout 15`	すべてのファイル転送について、リモート HTTP サーバーへの HTTP クライアント接続の特性を設定します。 • **forceclose** ：デフォルトの永続的接続を無効化します。

	Command or Action	Purpose
		• **idle timeout** *seconds*：アイドル接続が許容される時間を 1 秒から 60 秒の範囲で設定します。デフォルト タイムアウトは 30 秒です。 • **timeout** *seconds*：HTTP クライアントの接続待ち時間の上限を 1 秒から 60 秒の範囲で設定します。デフォルトは 10 秒です。
ステップ 4	**ip http client username** *username* **Example:** `Router(config)# ip http client username user1`	ユーザー認証を要求する HTTP クライアント接続で使用するユーザー名を指定します。 **Note** CLI で **copy** コマンドを発行する際、ユーザー名を指定することもできます。その場合、そこで入力されるユーザー名がこのコマンドの設定を上書きします。例については、「HTTP またはHTTPS を使用したリモートサーバーからのファイルのダウンロードの例」セクションを参照してください。
ステップ 5	**ip http client password** *password* **Example:** `Router(config)# ip http client password letmein`	ユーザー認証を要求する HTTP クライアント接続で使用するパスワードを指定します。 **Note** CLI で **copy** コマンドを発行する際、パスワードを指定することもできます。その場合、そこで入力されるパスワードがこのコマンドの設定を上書きします。例については、「HTTP またはHTTPS を使用したリモートサーバーからのファイルのダウ ンロードの例」セクションを参照してください。
ステップ 6	**ip http client proxy-server** {*proxy-name* | *ip-address*} [**proxy-port***port-number*] **Example:** `Router(config)# ip http client proxy-server edge2 proxy-port 29`	HTTP ファイル システム クライアント接続のために HTTP クライアントをリモート プロキシ サーバーに接続するよう設定します。 • オプションの **proxy-port***port-number* キーワードおよび引数で、リモート プロキシ サーバーのプロキシ ポート番号を指定します。
ステップ 7	**ip http client source-interface** *interface-id* **Example:** `Router(config)# ip http client source-interface Ethernet 0/1`	すべての HTTP クライアント コネクションの送信元アドレスにインターフェイスを指定します。

	Command or Action	Purpose
ステップ 8	**do copy running-config startup-config** **Example:** `Router(config)# do copy running-config startup-config`	（任意）実行コンフィギュレーションをスタートアップ コンフィギュレーション ファイルとして保存します。 • **do** コマンドを使用すると、グローバル コンフィギュレーション モードで特権 EXEC モード コマンドを実行できます。

ステップ 9	**end**
	Example:
	`Router(config)# end`
	Example:
	`Router#`
		コンフィギュレーション セッションを終了し、CLI をユーザー EXEC モードに戻します。

HTTP または HTTPS を使用したリモート サーバーからのファイルのダウンロード

HTTP または HTTPS を使用してリモート サーバーからファイルをダウンロードするには、次の作業を実行します。**copy** コマンドで、どのようなファイルでもコピー元からコピー先へコピーすることができます。

SUMMARY STEPS

1. **enable**
2. 次のいずれかを実行します。
   - **copy** [**/erase**] [**/noverify**] **http://***remote-source-urllocal-destination-url*
   - **copy https://** *remote-source-url local-destination-url*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> enable` **Example:**	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	次のいずれかを実行します。 • **copy** [**/erase**] [**/noverify**] **http://***remote-source-urllocal-destination-url*	HTTP または HTTPS を使用して、リモート Web サーバーからローカル ファイル システムへファイルをコピーします。

	Command or Action	Purpose

	• **copy https://** *remote-source-url local-destination-url*
	Example:
	`Router# copy http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx flash:c7200-i-mx`
	Example:
	`Router# copy`
	Example:
	`copy https://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx flash:c7200-i-mx`
		• **/erase** ：コピー前にローカルのコピー先ファイルシステムを消去します。このオプションは、限られたメモリ容量のクラスBファイルシステム プラットフォーム用に準備されたもので、ローカルのフラッシュ メモリ スペースを簡単にクリアできます。
		• **/noverify** ：コピーするファイルがイメージファイルの場合、このキーワードを使用すると、イメージがコピーされた後に発生するイメージの自動確認が無効化されます。
		• *remote-source-url* 引数は、コピーするファイルのコピー元の位置を示すURL（またはエイリアス）であり、標準の Cisco IOS ファイル システムの HTTP 構文では次のようになります。
		http:// [[*username*:*password*]**@**] {*hostname* | *host-ip*}[*/filepath*]*/filename*
		Note オプションの *username* 引数および *password* 引数は、ユーザー認証が必要な HTTP サーバーにログインするときに使用され、グローバル コンフィギュレーション コマンド **iphttpclientusername** および **iphttpclientpassword** の設定により当該の認証文字列を指定する代わりになります。
		• *local-destination-url* は、コピーするファイルを置く位置のURL（またはエイリアス）であり、標準の Cisco IOS ファイル システムの HTTP 構文では次のようになります。
		filesystem **:** [*/filepath*][*/filename*]
		Note **copy** コマンド使用時の URL 構文についての詳細は、「その他の参考資料」セクションを参照してください。

トラブルシューティングのヒント

リモート Web サーバーからのファイル転送に失敗した場合、次の点を確認します。

- ルータとインターネットとの接続はアクティブか。
- 正しいパスとファイル名が指定されているか。

- リモート サーバーがユーザー名とパスワードを要求しているか。
- リモートサーバーに非標準のコミュニケーション ポートが設定されていないか（HTTP のデフォルト ポートは 80、HTTPS のデフォルト ポートは 443）。

失敗したコピー要求の原因を判別できるよう、CLI はエラー メッセージを返します。コピー プロセスについての追加情報は、**debugiphttpclientall** コマンドで表示できます。

HTTP または HTTPS を使用したリモート サーバーへのファイルのアップロード

HTTP または HTTPS を使用してリモート サーバーへファイルをアップロードするには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. 次のいずれかを実行します。
   - **copy** [**/erase**] [**/noverify**] *local-source-url***http://***remote-destination-url*
   - **copy** *local-source-url* **https://** *remote-destination-url*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	次のいずれかを実行します。
	• **copy** [**/erase**] [**/noverify**] *local-source-url***http://***remote-destination-url*
	• **copy** *local-source-url* **https://** *remote-destination-url*
	Example:
	`Router# http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx_backup`
	Example:
	`Router# copy flash:c7200-i-mx http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx_backup`
	Example:
		HTTP または HTTPS を使用して、ローカル ファイル システムからリモート Web サーバーへファイルをコピーします。
		• **/erase** ：コピー前にローカルのコピー先ファイル システムを消去します。このオプションは、限られたメモリ容量のクラス B ファイル システム プラットフォーム用に準備されたもので、ローカルのフラッシュ メモリ スペースを簡単にクリアできます。
		• **/noverify** ：コピーするファイルがイメージ ファイルの場合、このキーワードを使用すると、イメージがコピーされた後に発生するイメージの自動確認が無効化されます。

	Command or Action	Purpose
		• *local-source-url* 引数は、コピーするファイルのコピー元の位置を示す URL（またはエイリアス）であり、標準の Cisco IOS ファイル システムの構文では次のようになります。 **http://** [[*username*:*password*]**@**] {*hostname* | *host-ip*}[*/filepath*]*/filename* **Note** オプションの *username* 引数および *password* 引数は、ユーザー認証が必要な HTTP サーバーにログインするときに使用され、グローバル コンフィギュレーション コマンド **iphttpclientusername** および **iphttpclientpassword** の設定により当該の認証文字列を指定する代わりになります。 • *remote-destination-url* は、コピーするファイルを置く URL（またはエイリアス）であり、標準の Cisco IOS ファイル システムの HTTP 構文では次のようになります。 *filesystem* **:** [*/filepath*][*/filename*] **Note** **copy** コマンド使用時の URL 構文についての詳細は、「その他の参考資料」セクションを参照してください。

トラブルシューティングのヒント

リモート Web サーバーからのファイル転送に失敗した場合、次の点を確認します。

- ルータとインターネットとの接続はアクティブか。
- 正しいパスとファイル名が指定されているか。
- リモート サーバーがユーザー名とパスワードを要求しているか。
- リモート サーバーに非標準のコミュニケーション ポートが設定されていないか（HTTP のデフォルト ポートは 80、HTTPS のデフォルト ポートは 443）。

失敗したコピー要求の原因を判別できるよう、CLI はエラー メッセージを返します。コピー プロセスについての追加情報は、**debugiphttpclientall** コマンドで表示できます。

HTTP を使用したファイル転送の維持とモニタリング

HTTP 接続の維持と監視を行うには、次の作業を実行します。ステップ 2 から 4 は任意の順序で実行できます。

SUMMARY STEPS

1. **enable**
2. **show ip http client connection**
3. **show ip http client history**
4. **show ip http client session-module**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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 ip http client connection** **Example:** `Router# show ip http client connection`	アクティブな HTTP クライアント接続の詳細を表示します。
ステップ 3	**show ip http client history** **Example:** `Router# show ip http client history`	HTTP クライアントがアクセスした URL のうち最新の 20 を表示します。
ステップ 4	**show ip http client session-module** **Example:** `Router# show ip http client session-module`	HTTP クライアントで登録されたセッション（アプリケーション）の詳細を表示します。

HTTP または HTTPS を使用したファイル転送の設定例

ファイル転送の HTTP 接続特性の設定：例

次の例に、全ユーザーの認証を行うリモート サーバーへの接続のために HTTP パスワードとユーザー名を設定する方法を示します。この例はまた、接続のアイドル時間制限を 20 秒に設定する方法も示しています。HTTP クライアントの接続待ち時間の上限は、デフォルトの 10 秒のままです。

```
Router(config)# ip http client connection idle timeout 20
Router(config)# ip http client password Secret
Router(config)# ip http client username User1
Router(config)# do show running-config | include ip http client
```

HTTP または HTTPS を使用したリモート サーバーからのファイルのダウンロードの例

次の例に、ファイル c7200-i-mx をリモート サーバーから HTTP を使用してフラッシュ メモリへコピーする設定方法を示します。この例はまた、ユーザー認証を行う HTTP サーバー用にコマンドラインからユーザー名とパスワードを入力する方法も示しています。

```
Router# copy http://user1:mypassword@209.165.202.129:8080/image_files/c7200-i-mx
flash:c7200-i-mx
```

フラッシュからリモート HTTP サーバーへのファイル アップロードの例

次の例は、フラッシュ メモリからリモート HTTP サーバーにファイルをコピーする方法を示しています。この例では、**copy** 特権 EXEC コマンドを使用したファイル転送で予期されるプロンプトと表示について示しています。

```
Router# copy flash:c7200-js-mz.ELL2 http://172.19.209.190/user1/c7200-js-mz.ELL2
Address or name of remote host [172.19.209.190]?
Destination filename [user1/c7200-js-mz.ELL2]?
Storing http://172.19.209.190/user1/c7200-js-mz.ELL2 !!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
17571956 bytes copied in 57.144 secs (307503 bytes/sec)
```

リモート HTTP サーバーからフラッシュ メモリへのファイルのダウンロードの例

次の例は、リモート HTTP サーバーからフラッシュ メモリへファイルをコピーする方法を示しています。この例では、**copy** 特権 EXEC コマンドを使用したファイル転送で予期されるプロンプトと表示について示しています。

```
Router# copy http://172.19.209.190/user1/c7200-i-mz.test  flash:c7200-i-mz.test
Destination filename [c7200-i-mz.test]?
Loading http://172.19.209.190/user1/c7200-i-mz.test
.
.
.
```

```
11272788 bytes copied in 527.104 secs (21386 bytes/sec)
```

HTTP または HTTPS を使用したリモート サーバーへのファイルのアップロード

次の例は、HTTP または HTTPS を使用してファイルをリモート サーバーにコピーする方法を示しています。

```
router#copy flash
: http:
Source filename []? running-config
Address or name of remote host []? 10.1.102.1 Destination filename [pilot-confg]?file1
...
```

その他の参考資料

ここでは、HTTP または HTTPS を使用したファイル転送に関する情報について説明します。

関連資料

関連項目	マニュアル タイトル
セキュア HTTP 通信	『*HTTPS —HTTP Server and Client with SSL 3.0*』
Cisco IOS 埋め込み Web サーバー	『*HTTP 1.1 Web Server and Client*』
Cisco IOS 組み込み Web クライアント	『*HTTP 1.1 Client*』
ネットワーク管理コマンド：コマンド構文の詳細、コマンドモード、コマンド履歴、デフォルト設定、使用上の注意事項、および例	『*Cisco IOS Network Management Command Reference*』
コンフィギュレーション基礎コマンド：完全なコマンド構文、コマンドモード、コマンド履歴、デフォルト、使用上の注意、例	『*Cisco IOS Configuration Fundamentals Command Reference*』

標準

標準	タイトル
この機能でサポートされる新規または改訂された標準規格はありません。また、この機能による既存の標準規格サポートの変更はありません。	--

MIB

MIB	MIB のリンク
なし	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、URL http://www.cisco.com/go/mibs にある Cisco MIB Locator を使用します。

RFC

RFC	タイトル
RFC 2616	『*Hypertext Transfer Protocol -- HTTP/1.1*』R. Fielding 他
RFC 2617	『*HTTP Authentication: Basic and Digest Access Authentication*』J. Franks 他

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

HTTP または HTTPS を使用したファイル転送の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

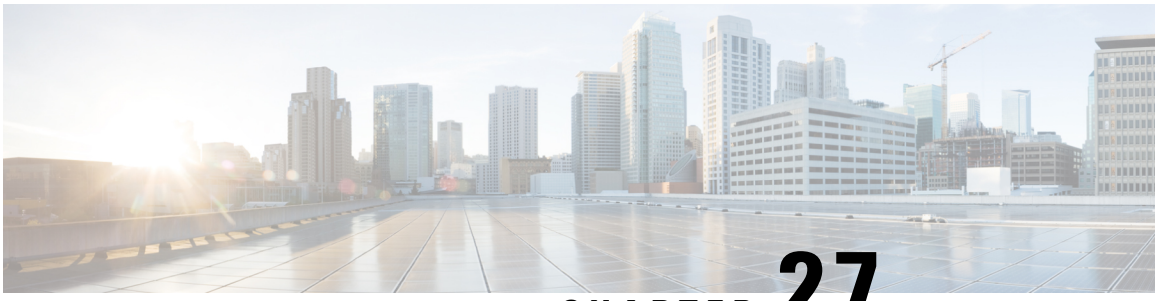
プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 32: HTTP または *HTTPS* を使用したファイル転送の機能情報

機能名	リリース	機能情報
HTTP を使用したファイルのダウンロード	12.3(2)T	HTTPを使用したファイルのダウンロード機能により、HTTPサーバーから Cisco IOS ソフトウェアベースのプラットフォームにファイルをコピーすることができます。
HTTP を使用したファイルのアップロード	12.3(7)T	
HTTP を使用したファイル転送	12.3(7)T	HTTP を使用したファイル転送機能は、Cisco IOS **copy** コマンドおよびコマンドライン インターフェイスを使用して、リモート サーバーから使用するローカル ルーティング デバイスへ、またはその逆の方向に、Cisco IOS イメージファイル、コア ファイル、コンフィギュレーション ファイル、ログ ファイル、スクリプトなどのファイルをコピーする機能を提供します。HTTP コピー操作は、FTP や TFTP など、他のリモート ファイル システムからのコピーと同じように動作します。 これにより、HTTP または HTTPS を使用して Cisco IOS ソフトウェアベースのプラットフォームから HTTP サーバーへファイルをコピーする機能がサポートされます。

第 V 部

システム イメージのロードと管理



CHAPTER 27

デジタル署名付き Cisco ソフトウェア

デジタル署名付き Cisco ソフトウェア機能では、デジタル署名付き Cisco ソフトウェアの識別、デジタル署名付きイメージに関するソフトウェア認証情報の収集、およびキー失効の実行について説明します。デジタル署名付き Cisco ソフトウェアは、セキュアな非対称（公開キー）暗号化を使用してデジタル署名されたソフトウェアです。

デジタル署名付き Cisco ソフトウェアの目的は、自分のシステム内で動作しているソフトウェアが改ざんされていないセキュアなもので、信頼できる送信元のものであることを、お客様に確信していただくことです。

デジタル署名付き Cisco ソフトウェアに関するソフトウェア アップデートについてお客様が不安を抱えているかもしれませんが、向上した保護機能を有効にするのに特別な作業は必要ありません。システム操作の大部分は、現行方針に対する透明性が概ね確保されています。デジタル署名付き Cisco ソフトウェアの使用を反映して、システム表示に小さな変更が加えられています。

デジタル署名付き Cisco ソフトウェアに関する制限事項

Cisco IOS XE ソフトウェアを実行する Cisco Catalyst 4500 E+Series スイッチには、このドキュメントで説明する機能（デジタル署名付きソフトウェアのキーの失効と置換を除く）が含まれています。

デジタル署名付き Cisco ソフトウェアに関する情報

デジタル署名付き Cisco ソフトウェアの機能と利点

3 つの主な要因によって、デジタル署名付き Cisco ソフトウェアとソフトウェア整合性検証が推進されています。

- 米国政府は、連邦情報処理標準（FIPS）140 の改訂版を公表しています。FIPS-140-3 は最新の草稿であり、2010 年に批准し、2011 年に発効するようにスケジュールされています。この標準では、ソフトウェアをロードおよび実行する前に、そのソフトウェアで信頼性と整合性を証明し、デジタル署名することが求められています。
- 製品のセキュリティに焦点を合わせることにより、シスコ製品への攻撃や脅威からの保護を強化しています。デジタル署名付き Cisco ソフトウェアは、破損している、または変更されているソフトウェアのインストールおよびロードを防止する保護機能の強化を提供します。
- デジタル署名付き Cisco ソフトウェアは、お客様の購入した機器が主張どおりのものであることを保証する、偽造防止機能です。

デジタル署名付き Cisco ソフトウェアの識別

デジタル署名付き Cisco IOS ソフトウェアは、イメージ名に含まれる 3 文字の拡張子によって識別されます。Cisco IOS イメージ ファイルは、Cisco ソフトウェア ビルド プロセスによって作成されます。このファイルに含まれるファイル拡張子は、イメージを署名するために使用された署名キーに基づいています。これらのファイル拡張子は次のようになります。

- .SPA
- .SSA

ファイル拡張子の各文字の意味を以下の表に示します。

Table 33: デジタル署名付き *Cisco* ソフトウェア イメージのファイル拡張子における文字の意味

ファイル拡張子の文字	文字の意味
S（最初の文字）	デジタル署名付きソフトウェアであることを表します。
P または S（2 番目の文字）	P または S はそれぞれ、製品および特別（開発）イメージであることを表します。製品イメージは、一般リリースが承認された Cisco ソフトウェアを指します。特別イメージは、特別な条件下で限定的に使用される開発用ソフトウェアを指します。

ファイル拡張子の文字	文字の意味
A（3 番めの文字）	イメージのデジタル署名に使用されているキー バージョンを示します。キー バージョンは A、B、C のようなアルファベット文字で識別されます。

デジタル署名付き Cisco ソフトウェアのキー タイプとバージョン

デジタル署名付き Cisco ソフトウェアのキーは、キーのタイプとバージョンによって識別されます。キーのタイプには、特別キー、製品キー、ロールオーバー キーがあります。特別キーと製品キーは、失効させることができます。ロールオーバー キーは、特別キーまたは製品キーを失効させるために使用します。ファイル拡張子の 2 番目の文字は、キー タイプ（特別キーまたは製品キー）を示します。キー タイプが製品キーの場合は「P」となり、特別キーの場合は「S」となります。

製品キーおよび特別キーの各タイプには、それぞれキー バージョンが関連付けられています。ファイル拡張子の 3 番目の文字（A、B、C のようなアルファベット文字）によって、キー バージョンが定義されます。キーを置換すると、キー バージョンのアルファベットが 1 つ進みます。たとえば、キー バージョンが「A」で、キー タイプが「P」（製品キー）のキーが失効すると、新しいイメージはキー バージョン「B」で署名されます。キー タイプとキー バージョンは、デバイスのキー ストレージにキー レコードの一部として保存されます。

デジタル署名付き Cisco ソフトウェアのキーの失効と置換

Note キーの失効と置換は、IOS XE ソフトウェアを実行している Catalyst 4500 E+Series スイッチではサポートされていません。

キー失効

キーの失効は、デジタル署名付き Cisco ソフトウェア内で動作中のキーを削除するプロセスです。

キーが侵害された場合、または使用されなくなった場合に、キー失効が発生します。キーの失効と置換は、特定の脆弱性またはシスコのセキュア キー インフラストラクチャに深刻な損失が発生した場合にのみ必要となります。そのような状況を修復する操作手順は、シスコによって通知され、指示された場合にのみ必要になります。通知と指示は、www.cisco.com での勧告の掲載またはフィールド通知によって行われます。

失効されるキーのタイプによって異なる 2 つのキー失効プロセスが存在します。

- 無効化イメージと製品イメージを使用する製品キーの置換
- 製品イメージを使用する特別キーの置換

キーの置換

キーの置換は、侵害されたキーと置き換えるための新しいキーを作成するプロセスです。侵害されたキーを失効させる前に、新しいキーが追加されます。キーの置換は2段階のプロセスです。

1. 新しいキーがキー ストレージに追加され、失効したキーを置き換えます。
2. イメージが新しいキーで正しく動作することが確認されると、侵害されたキーはキー ストレージから失効されます。

キー失効イメージ

失効イメージは、新しい製品キーをキー ストレージ領域に追加する機能を持つ、通常イメージの基本バージョンとなります。失効イメージに他の機能はありません。キーを失効させ、置換する場合に、キーごとに 1 つの失効イメージが作成されます。

失効イメージには、その中でバンドルされている新しい製品キーが含まれます。

プラットフォームに保存されたロールオーバー キーは、失効イメージの署名を検証するために使用されます。有効な失効イメージは同じロールオーバー キーを使用して署名されます。

Note 失効イメージが使用できるのは、製品キーの失効だけです。

失効イメージに関する重要なタスク

失効イメージに関して、2 つの重要な作業があります。

- 新しい製品キーのキー ストレージ領域への追加。
- 製品キーのアップグレードチェックの実行。詳細については、「製品キーの失効」の手順 2 を参照してください。

新しい製品キーのキー ストレージ領域への追加：

失効イメージは、バンドルされた製品キーをキー ストレージに追加します。追加されるキーはキー ストレージ内の既存のキー セットの一部ではないことが失効イメージによって確認された後、キーはプライマリおよびバックアップのキー ストレージ領域に書き込まれます。

キーのアップグレード チェックの実行：

新しいキーが追加され、お客様がソフトウェア（Cisco IOS および ROMmon）をアップグレードした後、show software authenticity upgrade-status コマンドを実行する必要があります。ユーザーは、製品キーが正常にアップグレードされ、次回のブート時に選択できるようになっているか確認するため、コマンド出力を確認できます。

製品キーの失効

侵害された製品キーを使用して署名されたイメージは信頼できないため、ロールオーバー キーによって署名された失効イメージを使用して、製品キー（リリース キーとも呼ばれます）は失効および置換されます。ROMmon はロールオーバー キーを使用して署名されたイメージを起動することができます。製品キーの失効と置換のプロセスに、4 つの手順が関係しています。

1. 新しい製品キーをキー ストレージに追加する。新しい製品キーは、失効イメージ内でバンドルされます。
2. show software authenticity upgrade-status コマンドを使用してソフトウェア アップグレード チェックを実行し、以下を確認します。
   - 新しい製品キー バージョンがインストールされたこと。
   - 新しい製品キーがプライマリ キー ストレージに追加されたこと（されていない場合、既存の失効イメージで software authenticity key add production コマンドを再発行する）。
   - 新しい製品キーがバックアップ キー ストレージに追加されたこと（されていない場合、既存の失効イメージで software authenticity key add production コマンドを再発行する）。
   - イメージが新しい製品キーで署名され、オートブートするように（boot system コマンドを使用）設定されたこと（されていない場合、新しい製品イメージをボックスにコピーし、新しいイメージをポイントするように boot system コマンドが変更されていることを確認する）。
   - アップグレード可能な ROMmon が新しい製品キーによって署名されていること（されていない場合、新しい製品キーによって署名された ROMMON にアップグレードする）。
3. すべてを確認したら、reload コマンドを使用して、新しい製品キーで署名された製品イメージをロードします。
4. 新しい製品イメージをロードしたら、software authenticity key revoke production コマンドを使用して侵害されたキーを失効させることができます。

手順 1 と 2 は、特別失効イメージを使用して実行します。いずれかのソフトウェアが古いキーを使用している場合、リブートしても（手順 3）、古いキーは失効されないため、手順 2 でこれらを確認することは重要です。この作業によって、新しいキーのインストールが完了し、次のリブート（手順 3）では新しいリリースのソフトウェアと新しい ROMmon が使用されることを確認できます。古い製品キーの失効（手順 4）は、新しいキーと新しいソフトウェアがシステムにインストールされてからでなければ、実行できません。

特別キーの失効

特別キーの失効には製品キーで署名された製品イメージが使用されます。特別キーの失効に使用される各製品イメージには、バンドルされた特別キー（製品イメージの作成時の最新）があります。特別キーの失効と置換のプロセスには、3 つの手順が含まれます。

1. バンドルされた新しい特別キーのキー ストレージ領域への追加。

2. 侵害された特別キーを使用して署名された ROMmon の、新しい特別キーを使用して署名された新しい ROMmon へのアップグレード。
3. キー ストレージからの侵害されたキーの失効。

手順3ではリブートする必要はありません。製品イメージ自体を使用して実行されることに注意してください。これは、お客様がすでに製品イメージを実行していて、無効化自体が稼働中の製品イメージから発生することによります。どのようなキーについても、特別イメージに追加や無効化の機能はありません。

デジタル署名付き Cisco ソフトウェア イメージの作業方法

デジタル署名付き Cisco ソフトウェアの識別

以下のタスクを実行して、デジタル署名付き Cisco ソフトウェアを識別します。このタスクでは、show version コマンドのコマンド出力でイメージ ファイル名を調べ、「デジタル署名付き Cisco ソフトウェアの識別」セクションで説明されている条件に基づいて判断します。

Note イメージファイルの名前がユーザーによって変更された場合、デジタル署名されたイメージであることを示す条件をユーザーが上書きしたために、イメージを識別できない可能性があります。

SUMMARY STEPS

1. **enable**
2. **show version**

DETAILED STEPS

	Command or Action	Purpose
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	**show version** **Example:** `Device# show version`	ルーティング デバイスで実行している Cisco IOS ソフトウェアのバージョン、ROM モニタとブートフラッシュ ソフトウェアのバージョン、およびシステム メモリの量を含むハードウェア構成についての情報が表示されます。

デジタル署名付き Cisco ソフトウェア署名情報の表示

以下のタスクを実行して、起動に使用する現在の ROMmon および Cisco IOS イメージ ファイルのソフトウェア認証に関する情報を表示します。この表示には、イメージのクレデンシャル情報、確認に使用されるキータイプ、署名情報、署名エンベロープのその他の属性が含まれます。

SUMMARY STEPS

1. **enable**
2. **show software authenticity running**

DETAILED STEPS

	Command or Action	Purpose
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	**show software authenticity running** **Example:** `Device# show software authenticity running`	起動に使用する現在の ROMmon および Cisco IOS イメージファイルのソフトウェア認証に関する情報を表示します。

特定のイメージ ファイルのデジタル署名情報の表示

以下のタスクを実行して、特定のイメージファイルのソフトウェア認証に関連したデジタル署名情報を表示します。

SUMMARY STEPS

1. **enable**
2. **show software authenticity file {flash0:filename | flash1:filename | flash:filename | nvram:filename | flash0:filename | flash1:filename}**

DETAILED STEPS

	Command or Action	Purpose
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	**show software authenticity file {flash0:filename | flash1:filename | flash:filename | nvram:filename | flash0:filename | flash1:filename}**	特定のイメージ ファイルのデジタル署名とソフトウェア認証に関連した情報を表示します。

	Command or Action	Purpose
	Example: `Device# show software authenticity file flash0:c3900-universalk9-mz.SPA`	

デジタル署名付き Cisco ソフトウェア キー情報の表示

以下のタスクを実行して、デジタル署名付き Cisco ソフトウェア キー情報を表示します。キータイプとともにストレージ内にあるソフトウェア公開キーの詳細情報を表示します。

SUMMARY STEPS

1. **enable**
2. **show software authenticity keys**

DETAILED STEPS

	Command or Action	Purpose
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	**show software authenticity keys** **Example:** `Device# show software authenticity keys`	デジタル署名付き Cisco ソフトウェアのキー タイプとともにストレージ内にあるソフトウェア公開キーを表示します。

デジタル署名付き Cisco ソフトウェア イメージのトラブルシューティング

以下のタスクを実行して、デジタル署名付き Cisco ソフトウェア イメージをトラブルシューティングします。

SUMMARY STEPS

1. **enable**
2. **debug software- authenticity errors {envelope | errors | key | revocation | show | verbose}**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:**	特権 EXEC モードを有効にします。

	Command or Action	Purpose
	`Device> enable`	• パスワードを入力します（要求された場合）。
ステップ 2	**debug software- authenticity errors {envelope | errors | key | revocation | show | verbose}** **Example:** `Device# debug software-authenticity errors`	デジタル署名付き Cisco ソフトウェアでデバッグ メッセージの表示をイネーブルにします。

デジタル署名付き Cisco ソフトウェアの設定例

デジタル署名付き Cisco ソフトウェアの識別例

次に、デジタル署名付き Cisco ソフトウェアのイメージ ファイル名を表示する例を示します。この方法によって、デジタル署名付き Cisco ソフトウェアの識別条件に基づいて識別することができます。

```
Device# show version
Cisco IOS Software, C3900 Software (C3900-UNIVERSALK9-M),
12.4(20090904:044027) [i12 577]
Copyright (c) 1986-2009 by Cisco Systems, Inc.
Compiled Fri 04-Sep-09 09:22 by xxx
ROM: System Bootstrap, Version 12.4(20090303:092436)
C3900-2 uptime is 8 hours, 41 minutes
System returned to ROM by reload at 08:40:40 UTC Tue May 21 1901!
System image file is "xxx.SPA"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Cisco xxx (revision 1.0) with CISCxxx with 987136K/61440K bytes of memory.
Processor board ID xxx
3 Gigabit Ethernet interfaces
1 terminal line
1 Virtual Private Network (VPN) Module
1 cisco Integrated Service Engine(s)
DRAM configuration is 72 bits wide with parity enabled.
255K bytes of non-volatile configuration memory.
1020584K bytes of USB Flash usbflash0 (Read/Write)
1020584K bytes of USB Flash usbflash1 (Read/Write)
500472K bytes of ATA System CompactFlash 0 (Read/Write)
License Info:
License UDI:
```

```
-----------------------------------------------
Device#   PID                   SN
-----------------------------------------------
xx        xxx                   xxxx
Technology Package License Information for Module:'xxx'
-----------------------------------------------------------------
Technology    Technology-package          Technology-package
              Current       Type          Next reboot
-----------------------------------------------------------------
ipbase        ipbasek9      Permanent     ipbasek9
security      securityk9    Evaluation    securityk9
uc            None          None          None
data          None          None          None
Configuration register is 0x2102
```

デジタル署名付きイメージ ファイルは、以下の行で識別されます。

```
System image file is "xxx.SPA"
```

イメージの特性として、ファイル名にデジタル署名付き Cisco ソフトウェアの 3 文字の拡張子（.SPA）が付きます。「デジタル署名付き Cisco ソフトウェアの識別」セクションのガイドラインに基づいて、ファイル拡張子の先頭の文字「S」はイメージがデジタル署名付きソフトウェア イメージであること、2 番目の文字「P」はイメージが製品キーを使用してデジタル署名されたこと、3 番目の文字「A」はキー バージョンがバージョン A であることが示されています。

デジタル署名付き Cisco ソフトウェア署名情報の表示例

次に、起動に使用する現在の ROMmon および Cisco IOS イメージ ファイルのソフトウェア認証に関する情報を表示する例を示します。

```
Device# show software authenticity running
SYSTEM IMAGE
--------------------
Image type                    : Development
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : xxx
    Signature Algorithm       : 2048-bit RSA
    Key Version               : xxx

    Verifier Information
        Verifier Name         : ROMMON 2
        Verifier Version      :  System Bootstrap, Version 12.4(20090409:084310)
ROMMON 2
---------------
Image type                    : xxx
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : xxx
    Signature Algorithm       : 2048-bit RSA
    Key Version               : xx
```

```
    Verifier Information
        Verifier Name         : ROMMON 2
        Verifier Version      :  System Bootstrap, Version 12.4(20090409:084310) [
```

次の表で、この出力に表示される重要なフィールドを説明します。

***Table 34: show software authenticity running* フィールドの説明**

フィールド	説明
SYSTEM IMAGE	システム イメージ情報を表示する出力のセクション。
Image type	イメージのタイプを表示する。
Common Name	ソフトウェア製造業者の名前を表示する。
Organization Unit	ソフトウェア イメージが導入されているハードウェアを表示する。
Organization Name	ソフトウェア イメージの所有者を表示する。
Certificate Serial Number	デジタル署名の証明書シリアル番号を表示する。
Hash Algorithm	デジタル署名の確認に使用されるハッシュ アルゴリズムの種類を表示する。
Signature Algorithm	デジタル署名の確認に使用される署名アルゴリズムの種類を表示する。
Key Version	確認に使用されるキー バージョンを表示する。
Verifier Name	デジタル署名の確認を受け持つプログラムの名前を表示する。
Verifier Version	デジタル署名の確認を受け持つプログラムのバージョンを表示する。
ROMMON 2	現在の ROMmon 情報を表示する出力のセクション。

特定のイメージ ファイルのデジタル署名情報の表示例

次に、特定のイメージファイルのソフトウェア認証に関連したデジタル署名情報を表示する例を示します。

Device# **show software authenticity file flash0:**c3900-universalk9-mz.SSA

```
File Name                     : flash0:c3900-universalk9-mz.SSA
Image type                    : Development
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : SHA512
    Signature Algorithm       : 2048-bit RSA
    Key Version               : A
```

The table below describes the significant fields shown in the display.

***Table 35: show software authenticity file* フィールドの説明**

フィールド	説明
File Name	メモリのファイル名。たとえば、flash0:c3900-universalk9-mz.SSA は、フラッシュ メモリ（flash0:）内のファイル名 c3900-universalk9-mz.SSA を指します。
Image type	イメージのタイプを表示する。
Signer Information	署名情報。
Common Name	ソフトウェア製造業者の名前を表示する。
Organization Unit	ソフトウェア イメージが導入されているハードウェアを表示する。
Organization Name	ソフトウェア イメージの所有者を表示する。
Certificate Serial Number	デジタル署名の証明書シリアル番号を表示する。
Hash Algorithm	デジタル署名の確認に使用されるハッシュ アルゴリズムの種類を表示する。
Signature Algorithm	デジタル署名の確認に使用される署名アルゴリズムの種類を表示する。
Key Version	確認に使用されるキー バージョンを表示する。

デジタル署名付き Cisco ソフトウェア キー情報の表示例

次の例では、デジタル署名付き Cisco ソフトウェア キー情報を表示します。キー タイプを含むストレージ内にあるソフトウェア公開キーの詳細情報を表示します。

```
Device# show software authenticity keys
Public Key #1 Information
-------------------------
Key Type              : Release  (Primary)
Public Key Algorithm : RSA
Modulus :
        CC:CA:40:55:8C:71:E2:4A:3A:B6:9D:5C:94:1D:02:BA:
       ...
        26:04:6B:33:EB:70:2B:18:24:C7:D9:31:3E:77:24:85
Exponent : xxx
Key Version           : A
Public Key #2 Information
-------------------------
Key Type              : Development  (Primary)
Public Key Algorithm : RSA
Modulus :
        CC:CA:40:55:8C:71:E2:4A:3A:B6:9D:5C:94:1D:02:BA:
       ....
```

```
        26:04:6B:33:EB:70:2B:18:24:C7:D9:31:3E:77:24:85
Exponent : xxx
Key Version          : A
```

The table below describes the significant fields shown in the display.

***Table 36: show software authenticity keys* フィールドの説明**

フィールド	説明
Public Key #	公開キー番号。
Key Type	イメージの確認に使用されるキー タイプを表示する。
Public Key Algorithm	公開キーの暗号化に使用されるアルゴリズム名を表示します。
Modulus	公開キー アルゴリズムの係数。
Exponent	公開キー アルゴリズムの指数。
Key Version	確認に使用されるキー バージョンを表示する。

デジタル署名付き Cisco ソフトウェア イメージ キー情報のデバッグの有効化：例

次に、デジタル署名付き Cisco ソフトウェアのキー情報に関連するソフトウェア認証イベントのデバッグを有効にする例を示します。

```
Device# debug software authenticity key
```

その他の参考資料

ここでは、デジタル署名付き Cisco ソフトウェアの機能の関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Command List, All Releases』
『System Management Command Reference』	http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst3850/software/release/3se/system_manag

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

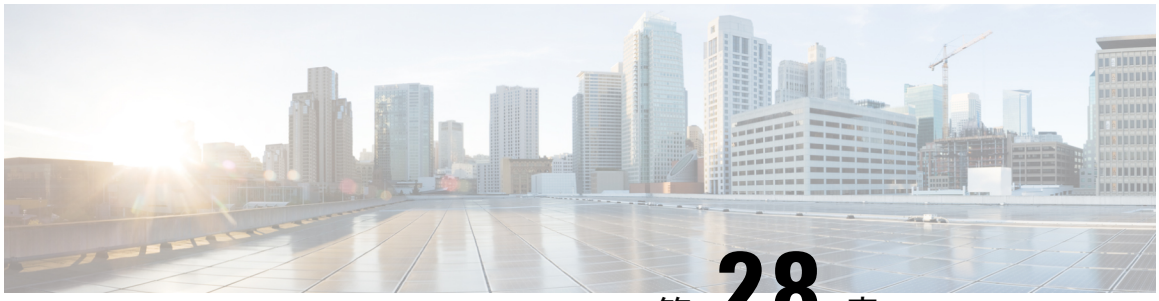
デジタル署名付き Cisco ソフトウェアの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 37: デジタル署名付き *Cisco* ソフトウェアの機能情報

機能名	リリース	機能情報
デジタル署名付き Cisco ソフトウェア		デジタル署名付き Cisco ソフトウェア機能では、デジタル署名付き Cisco ソフトウェアの識別、デジタル署名付きイメージに関するソフトウェア認証情報の収集、およびキー失効の実行について説明します。デジタル署名付き Cisco ソフトウェアは、セキュアな非対称（公開キー）暗号化を使用してデジタル署名されたソフトウェアです。 次のコマンドが導入または変更されました。 **debug software authenticity, show software authenticity file, show software authenticity keys, show software authenticity running.**
キー失効機能のサポート		キー失効機能のサポートが追加されました。キー失効では、プラットフォームのキーストレージからキーを削除します。プラットフォームは製品イメージまたは特別イメージをホストでき、製品キー（製品イメージから）または特別キー（特別イメージから）はキー失効の過程で失効させられます。 この機能に関する詳細については、次の項を参照してください。 • デジタル署名付き Cisco ソフトウェアのキーの失効と置換 次のコマンドが導入または変更されました。 **debug software authenticity, show software authenticity upgrade-status, software authenticity key add, software authenticity key revoke, upgrade rom-monitor file.**



第 28 章

FTP を使用したシステム イメージの管理

このモジュールには、FTP を使用したシステム イメージの管理に関する情報が含まれています。

フラッシュ メモリから FTP サーバーへのイメージのコピー

FTP プロトコルでは、FTP 要求ごとにリモート ユーザー名およびパスワードを、クライアントがサーバーに送信する必要があります。FTP を使用して、ルータからサーバーにコンフィギュレーション ファイルをコピーする場合、Cisco IOS ソフトウェアは、次のうち、最初に発見した有効なユーザー名を送信します。

1. **copy** 特権 EXEC コマンドで指定されたユーザー名（ユーザー名が指定されている場合）。
2. **ipftpusername** グローバル コンフィギュレーション コマンドで設定されたユーザ名（コマンドが設定されている場合）。
3. Anonymous

ルータは次のうち、最初に発見した有効なパスワードを送信します。

1. **copy** 特権 EXEC コマンドで指定されたパスワード（パスワードが指定されている場合）
2. **ipftppassword** グローバル コンフィギュレーション コマンドで設定されたパスワード（コマンドが設定されている場合）。

ルータは、*username* *@routername* *.domain* というパスワードを生成します。変数 *username* は現在のセッションに関連付けられたユーザー名、*routername* は設定済みのホスト名、*domain* はルータのドメインです。

ユーザー名およびパスワードは、FTP サーバーのアカウントに関連付けられている必要があります。サーバに書き込む場合、ルータ上のユーザからの FTP 書き込み要求を受け入れるように、FTP サーバを適切に設定する必要があります。

このサーバーがディレクトリ構造をとっている場合、コンフィギュレーション ファイルまたはイメージは、サーバー上のユーザー名と関連付けられたディレクトリに書き込まれるか、そのディレクトリからコピーされます。たとえば、システム イメージがサーバー上のユーザーのホーム ディレクトリにある場合は、そのユーザーの名前をリモート ユーザー名として指定します。

詳細については、ご使用の FTP サーバーのマニュアルを参照してください。

すべてのコピー操作に使用するユーザー名およびパスワードを指定するには、**ipftpusername** および **ipftppassword** コマンドを使用します。特定のコピー操作にのみ使用するユーザー名を指定する場合は、**copy** コマンド内でユーザー名を指定します。

FTP サーバーからフラッシュ メモリ ファイル システムへのイメージのコピー

FTP サーバーからフラッシュ メモリ ファイル システムへシステム イメージをコピーできます。

FTP ユーザー名とパスワード

FTP プロトコルでは、FTP 要求ごとにリモートユーザー名およびパスワードを、クライアントがサーバーに送信する必要があります。FTP を使用して、ルータからサーバーにコンフィギュレーション ファイルをコピーする場合、Cisco IOS ソフトウェアは、次のうち、最初に発見した有効なユーザー名を送信します。

1. **copy** 特権 EXEC コマンドで指定されたユーザー名（ユーザー名が指定されている場合）。
2. **ipftpusername** グローバル コンフィギュレーション コマンドで設定されたユーザ名（コマンドが設定されている場合）。
3. Anonymous

ルータは次のうち、最初に発見した有効なパスワードを送信します。

1. **copy** 特権 EXEC コマンドで指定されたパスワード（パスワードが指定されている場合）
2. **ip ftp password** コマンドで設定されたパスワード（コマンドが設定されている場合）。

ルータは、*username* *@routername* *.domain* というパスワードを生成します。変数 *username* は現在のセッションに関連付けられたユーザ名、*routername* は設定済みのホスト名、*domain* はルータのドメインです。

ユーザー名およびパスワードは、FTP サーバーのアカウントに関連付けられている必要があります。サーバに書き込む場合、ルータ上のユーザからの FTP 書き込み要求を受け入れるように、FTP サーバを適切に設定する必要があります。

このサーバーがディレクトリ構造をとっている場合、コンフィギュレーションファイルまたはイメージは、サーバー上のユーザー名と関連付けられたディレクトリに書き込まれるか、そのディレクトリからコピーされます。たとえば、システム イメージがサーバー上のユーザーのホーム ディレクトリにある場合は、そのユーザーの名前をリモート ユーザー名として指定します。

詳細については、ご使用の FTP サーバーのマニュアルを参照してください。

すべてのコピー操作に使用するユーザー名およびパスワードを指定するには、**ip ftp username** および **ip ftp password** コマンドを使用します。特定のコピー操作にのみ使用するユーザー名を指定する場合は、**copy** コマンド内でユーザー名を指定します。

フラッシュ メモリから FTP サーバーにイメージをコピー

FTP ネットワーク サーバー上のシステム イメージをコピーするには、以下の手順を実行します。

ステップ 1 **enable**

Example:

```
Router> 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:

```
Router# configure terminal
```

（任意）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。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 2 および 3 を参照）。

ステップ 3 **ip ftp username** *username*

Example:

```
Router(config)# ip ftp username user1
```

（任意）デフォルトのリモート ユーザー名を変更します。

ステップ 4 **ip ftp password** *password*

Example:

```
Router(config)# ip ftp password guessme
```

（任意）デフォルトのパスワードを変更します。

ステップ 5 **end**

Example:

```
Router(config)# end
```

（任意）グローバル コンフィギュレーション モードを終了します。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 2 および 3 を参照）。

ステップ 6 **show** *flash-filesystem* **:**

Example:

```
Router# show flash:
```

（任意）指定されたフラッシュ ディレクトリのシステム イメージ ファイルを表示します。フラッシュ メモリ内のシステムイメージファイル名を知らない場合は、このファイル名の正確なスペルをメモしておきます。

ステップ 7 **copy** *flash-filesystem* **:** *filename* **ftp:** [[[//[*username* [**:***password*]**@**]*location*]/*directory*]/*filename*]

Example:

```
Router# copy slot0:1:your-ios ftp://myuser:mypass@172.23.1.129/dirt/sysadmin/your-ios
```

このイメージを FTP サーバーにコピーします。

Note **copy** 特権 EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

例

この例では、**showslot1:privileged**EXEC コマンドを使用して 2 番目の PCMCIA スロットにあるシステム イメージ ファイルの名前を表示し、ファイル（test）を FTP サーバーにコピーします。

```
Router# show slot1:
-#- ED --type-- --crc--- -seek-- nlen -length- -----date/time------ name
1   .. 1        46A11866 2036C   4    746      May 16 1995 16:24:37 test
Router# copy slot1:test ftp://thisuser:thatpass@172.16.13.110/test
writing test!!!!...
successful ftp write.
```

この例では、your-ios という名前のファイルを、スロット 0 にあるフラッシュ メモリ PC カードのパーティション 1 から、172.23.1.129 にある TFTP サーバーにコピーします。このファイ

ルは、リモート ユーザー名を持つディレクトリに対する dirt/sysadmin ディレクトリに your-ios という名前で保存されます。

```
Router# show slot0: partition 1
PCMCIA Slot0 flash directory, partition 1:
File  Length   Name/status
  1   1711088 your-ios
[1711152 bytes used, 2483152 available, 4194304 total]
Router# copy slot0:1:your-ios ftp://myuser:mypass@172.23.1.129/dirt/sysadmin/your-ios
Verifying checksum for 'your-ios' (file # 1)...  OK
Copy 'your-ios' from Flash to server
  as 'dirt/sysadmin/ios-2'? [yes/no] yes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Upload to server done
Flash device copy took 00:00:23 [hh:mm:ss]
```

FTP サーバーからフラッシュ メモリへのコピー

FTP サーバーからフラッシュ メモリ ファイル システムへシステム イメージをコピーするには、以下の手順を実行します。

ステップ 1 **enable**

Example:

```
Router> 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** *flash-filesystem* **:**

Example:

```
Router# show flash:
```

（任意）フラッシュ メモリ内のシステム イメージ ファイル名を表示します。このコマンドを使用して、この次のコマンドで使用するために、ファイルの URL パスとシステム イメージ ファイル名の正確なスペルを確認します。

ステップ 3 **copy** *flash-url* **tftp** :[[[*//location*]*/directory*]*/filename*]

Example:

```
Router# copy slot0:1:your-ios tftp://172.23.1.129/dirt/sysadmin/your-ios
```

フラッシュ メモリから TFTP サーバーにシステム イメージをコピーします。ファイルの場所とファイル名を *flash-url* 引数として指定します。

Note **copy** 特権 EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

ステップ 4 **configure terminal**

Example:

```
Router# configure terminal
```

（任意）端末からグローバル コンフィギュレーション モードを開始します。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 3 および 4 を参照）。

ステップ 5 **ip ftp username** *username*

Example:

```
Router(config)# ip ftp username netuser1
```

（任意）デフォルトのリモート ユーザー名を変更します。

ステップ 6 **ip ftp password** *password*

Example:

```
Router(config)# ip ftp password guessme
```

（任意）デフォルトのパスワードを変更します。

ステップ 7 **end**

Example:

```
Router(config)# end
```

（任意）グローバル コンフィギュレーション モードを終了します。この手順は、デフォルトのリモート ユーザー名またはパスワードを上書きする場合にだけ必要です（ステップ 3 および 4 を参照）。

ステップ 8 **copy ftp:** [[[//[*username* [**:***password*]**@**]*location*] */directory*]*/filename*]*flash-filesystem***:**[*filename*]

Example:

```
Router# copy ftp://myuser:mypass@theserver/tftpboot/sub3/c7200-js-mz slot1:c7200-js-mz
```

コンフィギュレーション ファイルをネットワーク サーバーから稼働中のメモリ、または rcp を使用してスタートアップ コンフィギュレーションにコピーします。

Note **copy** 特権 EXEC コマンドを発行した後、追加情報またはアクションの確認を求めるプロンプトが表示される場合があります。表示されるプロンプトは、**copy** コマンドで入力した情報量および **fileprompt** グローバル コンフィギュレーション コマンドの現在の設定によって異なります。

例

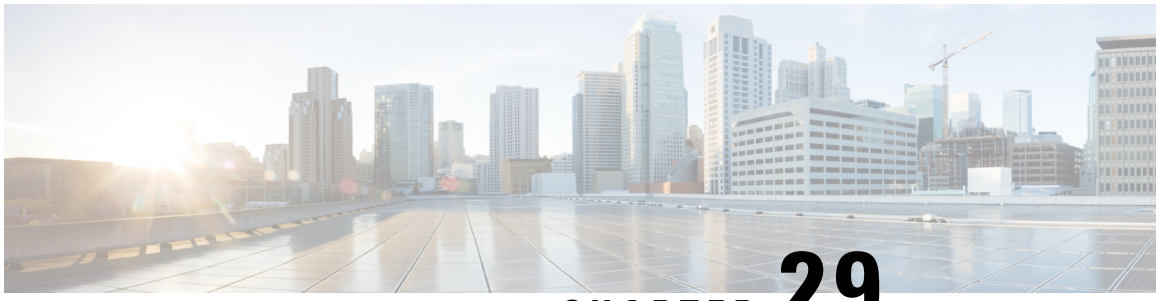
次に、**reload** コマンドを使用して、ルータでソフトウェアを現在の日の午後 7 時 30 分にリロードする例を示します。

```
Router# reload at 19:30
Reload scheduled for 19:30:00 UTC Wed Jun 5 1996 (in 2 hours and 25 minutes)
Proceed with reload? [confirm]
```

次に、**reload** コマンドを使用して、ルータでソフトウェアを将来リロードする例を示します。

```
Router# reload at 02:00 jun 20
Reload scheduled for 02:00:00 UTC Thu Jun 20 1996 (in 344 hours and 53 minutes)
Proceed with reload? [confirm]
```

例



CHAPTER 29

Cisco IOS Auto-Upgrade Manager の設定

Cisco IOS Auto-Upgrade Manager（AUM）機能を使用すると、新しい Cisco IOS イメージを指定、ダウンロード、アップグレードするための単純なインターフェイスが利用できるようになり、ソフトウェア イメージのアップグレード プロセスが単純化されます。

Auto-Upgrade Manager の指示に従ってプロセスを進めることにより、対話モードで新しい Cisco IOS イメージにアップグレードできます。また、単一の Cisco IOS コマンドまたは一連のコマンドを実行してアップグレードを行うこともできます。3 つの方法すべてで、ウォーム アップグレード機能を使用してアップグレードが行われ、ダウンタイムが最小化されます。

Cisco IOS Auto-Upgrade Manager のための前提条件

- シスコからダウンロードするために、ルータ上で DNS サーバーの IP アドレスを設定する必要があります。詳細については、「DNS サーバーの IP アドレスの設定：例」セクションおよび「関連資料」セクションを参照してください。
- シスコからダウンロードするために、ルータ上でシスコの Web サイト（www.cisco.com）から取得した Secure Socket Layer（SSL）証明書を設定する必要があります。この設定は、シスコ以外のサーバーからダウンロードする場合は不要です。詳細については、「シスコダウンロードの SSL 証明書の設定」セクションおよび「関連資料」セクションを参照してください。
- 暗号化 Cisco IOS ソフトウェア イメージをダウンロードする場合は、暗号化ソフトウェアのダウンロードのために、シスコに登録する必要があります。

Cisco IOS Auto-Upgrade Manager の制約事項

要求された Cisco IOS ソフトウェア イメージをロードおよび格納するための十分なメモリ リソースがルータにない場合、Cisco IOS Auto-Upgrade Manager は最後まで完了しません。Cisco IOS ソフトウェア イメージは、ルータで現在動作している Cisco IOS ソフトウェア イメージが暗号化イメージの場合にだけ www.cisco.com からダウンロードできます。

Cisco IOS Auto-Upgrade Manager について

Cisco IOS Auto-Upgrade Manager の概要

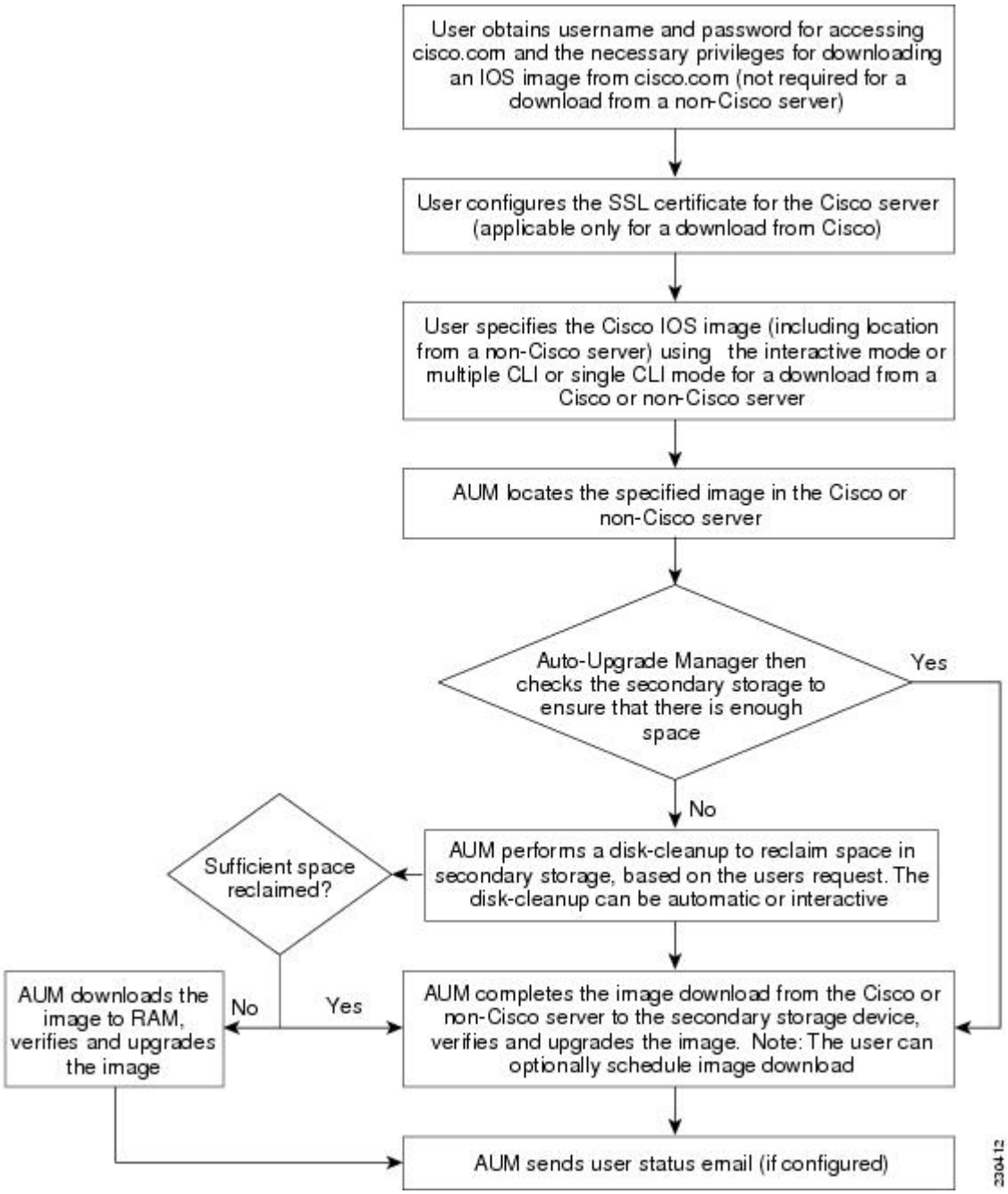
Cisco IOS Auto-Upgrade Manager は、新しい Cisco IOS ソフトウェア イメージのアップグレード プロセスを効率化します。Cisco IOS Auto-Upgrade Manager は、コマンドライン インターフェイス（CLI）を通じて実行できます。AUM では、ルータをシスコの Web サイト（www.cisco.com）に接続し、cisco.com のユーザー名とパスワードを認証のために送信できます。認証後、ルータは、ユーザーが指定した Cisco IOS ソフトウェア イメージの名前をシスコのサーバーに渡します。シスコのサーバーは、Cisco IOS ソフトウェア イメージの完全な URL をルータに返します。

ルータで設定された Cisco IOS Auto-Upgrade Manager は、Cisco IOS ソフトウェア イメージへのアップグレード プロセス全体を管理します。AUM は、次の作業を実行することにより、ユーザーによって指定された時刻に、ソフトウェアイメージを使用してルータをアップグレードします。

- Cisco IOS ソフトウェア イメージの検索とダウンロード
- すべての要件の確認
- 第 2 記憶域の管理
- Cisco IOS ソフトウェア イメージの検証
- ウォームアップグレードのスケジューリング

下の図に、Cisco IOS Auto-Upgrade Manager のワークフローを示します。

Figure 10: Cisco IOS Auto-Upgrade Manager のワークフロー



Note ルータが、ユーザーが指定した Cisco IOS ソフトウェア イメージのロードに失敗すると、コンソール ウィンドウと syslog バッファに、エラーの理由を示すエラー メッセージが表示されます。ユーザーが暗号化ソフトウェアをダウンロードする許可を持っていない場合、このサービスに登録するようユーザーに求めるエラー メッセージが生成されます。同様に、いずれかの CLI 設定文がブート時にパーサーに理解されない場合、エラー メッセージが生成され、無効な設定行のログが nvram:invalid-config ファイルに格納されます。このエラー メッセージは、ユーザーが指定した Cisco IOS ソフトウェア イメージが、以前の Cisco IOS ソフトウェア イメージと同じフィーチャ セットをサポートしていないことを示します。ルータに、両方のイメージをサポートするために十分な第 2 記憶域がなく、新しいイメージのアップグレードに成功した場合、再度シスコのサーバーに接続して、第 2 記憶域に Cisco IOS ソフトウェア イメージをダウンロードします。このプロセスにより既存のイメージが消去されます。

シスコの Web サイトからの特定の Cisco IOS ソフトウェア イメージのダウンロード

www.cisco.com から特定の Cisco IOS ソフトウェア イメージをダウンロードできます。AUM は、セキュアな接続のために Secure Socket Layer（SSL）を使用するため、ユーザー側で証明書を設定する必要があります。ルータは、Cisco IOS ソフトウェア イメージの名前を、www.cisco.com サーバーにログインするためのユーザー名およびパスワードとともに渡します。シスコのサーバーは、特定の Cisco IOS ソフトウェア イメージの完全な URL をルータに返します。

Cisco IOS Auto-Upgrade Manager は、ユーザーが指定した Cisco IOS ソフトウェア イメージを自動的に www.cisco.com からダウンロードして確認し、ダウンロードしたイメージでルータをアップグレードします。

Note Intelligent Download Application（IDA）は、AUM に対するシスコのインターフェイスであり、AUM に関してはシスコのサーバーと同じ意味で使用されます。

また、Cisco IOS Auto-Upgrade Manager では、次のオプション サービスが提供されます。

- ディスク クリーンアップ ユーティリティ
- アップグレードのスケジューリング

これらのサービスは、シスコのサーバーとシスコ以外のサーバーからのダウンロードに対して、対話モードとコマンド ラインモードの両方で使用できます。

シスコ以外のサーバーからの特定の Cisco IOS ソフトウェア イメージのダウンロード

ローカルまたはシスコ以外の TFTP サーバーまたは FTP サーバーに存在する Cisco IOS ソフトウェア イメージをダウンロードできます。FTP ダウンロードのための FTP ユーザー名とパスワードは、**ipftpusername** および **ipftppassword** 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指定します。Cisco IOS Auto-Upgrade Manager では、特定の Cisco IOS ソフトウェア イメージのシスコ以外のサーバーからのダウンロードとウォーム アップグレード サービスのプロセスが自動化されます。また、新しい Cisco IOS ソフトウェア イメージをダウンロードするために必要な領域が十分でない場合に使用する、ファイルを削除するためのディスク クリーンアップ ユーティリティも提供されています。

対話型およびシングル コマンド ライン モード

CLI を使用するか、次のユーザー インターフェイスを通じて、特定の Cisco IOS ソフトウェア イメージを www.cisco.com からダウンロードできます。

対話モード

Auto-Upgrade Manager に従って、対話モードで新しい Cisco IOS イメージにアップグレードできます。自動アップグレードを選択すると、対話モードでいくつかの問題に答えるだけでデバイスのアップグレードが完了します。対話モードを開始するには、オプションなしで **upgradeautomatic** コマンドを実行します。詳細については、『*Cisco IOS Configuration Fundamentals Command Reference*』を参照してください。

シングル コマンド ライン モード

対話型でないシングル ライン CLI は、上級ユーザー向けです。**upgradeautomaticgetversion** コマンドを使用し、必要なすべての引数を指定することで、シスコのサーバーまたはシスコ以外のサーバーから新しい Cisco IOS ソフトウェア イメージをダウンロードし、アップグレードできます。詳細については、『*Cisco IOS Configuration Fundamentals Command Reference*』を参照してください。

対話モードとシングル ライン CLI モードは、シスコのサーバーとシスコ以外のサーバーからのダウンロードに適用されます。

Cisco IOS Auto-Upgrade Manager を使用した Cisco IOS ソフトウェア イメージのアップグレード方法

シスコからのダウンロードのための SSL 証明書の設定

この作業では、シスコからダウンロードするための SSL 証明書を設定します。

Before you begin

SSL 証明書を、cisco.com からダウンロードするように設定しておく必要があります。証明書は、セキュアな HTTP 通信のために必要です。SSL 証明書は、シスコの Web サイト（www.cisco.com）からダウンロードしてルータ上で設定します。

シスコの Web サイトから SSL 証明書を取得するには、次の作業を実行します。

1. Internet Explorer（IE）の [Tools] メニューから [Internet Options] を選択します。
2. [Advanced] タブで [Warn if changing between secure and not secure mode] を選択します。
3. IE に URL として https://www.cisco.com/ と入力します。セキュリティ警告のポップアップボックスが表示され、「You are about to leave a secure Internet connection. Do you want to continue?」というメッセージが表示されたら、[No] をクリックします。
4. IE のステータスバーにある鍵のアイコンをダブルクリックします。これにより、証明書の詳細を示すダイアログボックスが表示されます。
5. [Certification Path] タブをクリックします。タブには証明書チェーンが表示されます。
6. CA 証明書をそれぞれ選択して [View Certificate] をクリックします。これにより、証明書の詳細を示すウィンドウが表示されます。
7. 表示された証明書ウィンドウの [Details] タブを選択して、[Copy to File] をクリックします。これにより、証明書のエクスポート ウィザードが開きます。
8. 証明書を Base-64 符号化形式でファイル（cisco.cert など）に保存します。
9. cisco.cert ファイルをメモ帳で開き、ルータを設定するために必要な証明書データを取得します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal**
5. **revocation-check none**
6. **exit**
7. **crypto ca authenticate** *name*

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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）。

	Command or Action	Purpose
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	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint cisco_ssl_cert`	認証局（CA）を宣言し、CA トラストポイント コンフィギュレーション モードを開始します。
ステップ 4	**enrollment terminal** **Example:** `Device(ca-trustpoint)# enrollment terminal`	コンソール端末上に証明書要求を表示し、発行された証明書データを端末上に入力できるようにします。
ステップ 5	**revocation-check none** **Example:** `Device(ca-trustpoint)# revocation-check none`	証明書の確認が必要ないことを指定します。
ステップ 6	**exit** **Example:** `Device(ca-trustpoint)# exit`	CA トラストポイント コンフィギュレーション モードを終了し、グローバル コンフィギュレーション モードに戻ります。
ステップ 7	**crypto ca authenticate** *name* **Example:** `Device(config)# crypto ca authenticate cisco_ssl_cert`	CA の自己署名証明書を取得することで、CA がルータに対して認証されます。

Cisco IOS Auto-Upgrade Manager の設定

Cisco IOS Auto-Upgrade Manager を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **autoupgrade disk-cleanup** {**crashinfo** | **core** | **image** | **irrecoverable**}
4. **autoupgrade ida url** *url*
5. **autoupgrade status email** {**recipient***email-address* | **smtp-server***name-address*}

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**autoupgrade disk-cleanup** {**crashinfo** | **core** | **image** | **irrecoverable**} **Example:** `Device(config)# autoupgrade disk-cleanup crashinfo`	Cisco IOS Auto-Upgrade Manager のディスク クリーンアップ ユーティリティを設定します。
ステップ 4	**autoupgrade ida url** *url* **Example:** `Device(config)# autoupgrade ida url https://www.cisco.com/cgi-bin/new-ida/locator/locator.pl`	Cisco IOS Auto-Upgrade Manager によってイメージ ダウンロード要求が送信される、www.cisco.com 上で動作しているシスコのサーバーの URL を設定します。 **Note** この手順は、デフォルトの URL が変更された場合にだけ必要です。
ステップ 5	**autoupgrade status email** {**recipient***email-address* | **smtp-server***name-address*} **Example:** `Device(config)# autoupgrade status email smtp-server smtpserver.abc.com`	ルータからのステータス電子メールの宛先となる電子メール アドレスと電子送信サーバーを設定します。

Cisco IOS ソフトウェア イメージのダウンロード

Cisco IOS ソフトウェア イメージをシスコの Web サイト（www.cisco.com）またはシスコ以外のサーバーからダウンロードするには、この作業を実行します。

SUMMARY STEPS

1. **enable**
2. **upgrade automatic getversion** {**ciscousername***username***password***password***image***image* | *url*} [**at***hh***:***mm* | **now** | **in***hh***:***mm*] [**disk-management**{**auto** | **confirm** | **no**}]

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**upgrade automatic getversion** {**ciscousername***username***password***password***image***image* | *url*} [**at***hh***:***mm* | **now** | **in***hh***:***mm*] [**disk-management**{**auto** | **confirm** | **no**}] **Example:** `Device# upgrade automatic getversion tftp://abc/tom/c3825-adventerprisek9-mz.124-2.XA.bin at now disk-management auto`	www.cisco.com またはシスコ以外のサーバーから、直接イメージをダウンロードします。

新しい Cisco IOS ソフトウェア イメージを使用したルータのリロード

新しい Cisco IOS ソフトウェア イメージを使用してルータをリロードするには、ここで説明する作業を実行します。

SUMMARY STEPS

1. **enable**
2. **upgrade automatic runversion** [**at***hh***:***mm* | **now** | **in***hh***:***mm*]

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**upgrade automatic runversion** [**at***hh***:***mm* | **now** | **in***hh***:***mm*]	新しいイメージでルータをリロードします。

	Command or Action	Purpose
	Example: `Device# upgrade automatic runversion at 7:30`	**Note** また、**upgradeautomaticgetversion** コマンドを使用して、新しい Cisco IOS ソフトウェア イメージでルータをリロードすることもできます。ただし、**upgradeautomaticgetversion** コマンドを使用してすでに Cisco IOS ソフトウェア イメージをダウンロードしてある場合は、**upgradeautomaticrunversion** コマンドを使用してルータをリロードする必要があります。

Cisco IOS ソフトウェア イメージのリロードの取り消し

特定の Cisco IOS ソフトウェア イメージのスケジューリングされたリロードを取り消すには、この作業を実行します。

次の状況でイメージのリロードを取り消すことができます。

- ルータをリロードするようスケジューリングされた時刻が十分でない場合。
- ルータを新しいイメージにアップグレードしない場合。

SUMMARY STEPS

1. **enable**
2. **upgrade automatic abortversion**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**upgrade automatic abortversion** **Example:** `Device# upgrade automatic abortversion`	Cisco IOS ソフトウェア イメージのアップグレードを取り消します。

Cisco IOS Auto-Upgrade Manager の設定例

DNS サーバーの IP アドレスの設定：例

Cisco IOS Auto-Upgrade Manager を設定する前に、ルータ上で DNS サーバーの IP アドレスを設定する必要があります。これらの一連のイベントでは、ルータで **ping** コマンドを実行するときに、IP アドレスの代わりにホスト名を使用できます。ルータ上で DNS サーバーの IP アドレスを設定した後、シスコの Web サイト（www.cisco.com）に正常に ping できるようになります。このアクションにより、ルータがインターネットに接続されていることも確認できます。

次に、ルータ上で DNS サーバーの IP アドレスを設定する例を示します。DNS サーバーの IP アドレスを設定した後、www.cisco.com に正常に ping できるようになります。

```
configure terminal
 ip domain name mycompany.com
 ip name-server 10.2.203.1
 end
  ping www.cisco.com
```

シスコからのダウンロードのための SSL 証明書の設定：例

Cisco IOS Auto-Upgrade Manager を使用してシスコの Web サイトからイメージをダウンロードする前に、ルータ上でシスコのサーバーの SSL 証明書を設定する必要があります。

次に、SSL 証明書を設定する例を示します。

```
configure terminal
 crypto pki trustpoint cisco_ssl_cert
  enrollment terminal
  revocation-check none
  exit
 crypto ca authenticate cisco_ssl_cert
!Enter the base 64 encoded CA certificate and end this with a blank line or the word
quit
. !The console waits for the user input. Paste the SSL certificate text and press Return.


-----BEGIN CERTIFICATE-----


<The content of the certificate>

-----END CERTIFICATE-----


!Trustpoint 'cisco_ssl_cert' is a subordinate CA and holds a non self signed cert
!Trustpoint 'cisco_ssl_cert' is a subordinate CA.
!but certificate is not a CA certificate.
!Manual verification required
!Certificate has the following attributes:
      ! Fingerprint MD5: 49CE9018 C0CC41BA 1D2FBEA7 AD3011EF
      ! Fingerprint SHA1: A88EAA5D 73D63CB7 BF25197B 9C35ED97 023BB57B

% Do you accept this certificate? [yes/no]: yes
```

```
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Cisco IOS Auto-Upgrade Manager の設定：例

次に、ルータ上で Cisco IOS Auto-Upgrade Manager を設定する例を示します。

```
configure terminal
autoupgrade disk-cleanup crashinfo
autoupgrade ida url https://www.cisco.com/cgi-bin/new-ida/locator/locator.pl
autoupgrade status status email smtp-server
```

その他の参考資料

次の項では、Cisco IOS Auto-Upgrade Manager の関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS Auto-Upgrade Manager コマンド：完全なコマンド構文、コマンドモード、コマンド履歴、デフォルト、使用ガイドライン、および例	『Cisco IOS Configuration Fundamentals Command Reference』
Cisco ルータでの DNS の設定	『Configuring DNS on Cisco Routers』テクニカル ノート
ウォーム アップグレード	機能モジュールのウォーム アップグレード

標準

標準	タイトル
なし	--

MIB

MIB	MIB のリンク
なし	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
なし	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Cisco IOS Auto-Upgrade Manager の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 38: Cisco IOS Auto-Upgrade Manager の機能情報

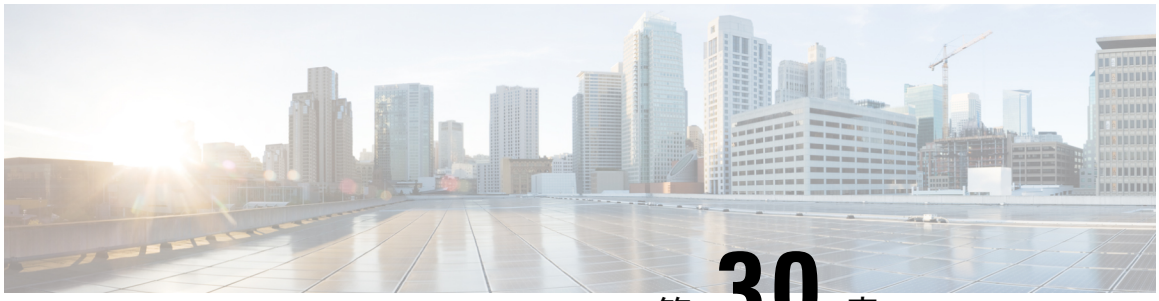
機能名	リリース	機能情報
Cisco IOS Auto-Upgrade Manager	12.4(15)T Cisco IOS XE Release 3.9S	Cisco IOS Auto-Upgrade Manager を使用すると、新しい Cisco IOSイメージを指定し、ダウンロードして、アップグレードするための単純なインターフェイスが利用できるようになり、ソフトウェア イメージのアップグレード プロセスが単純化されます。 12.4(15)T で、この機能が Cisco 1800、Cisco 2800、および Cisco 3800 シリーズ ルータに追加されました。 この機能は、Cisco IOS XE Release 3.9S に統合されました。 この機能により、次のコマンドが導入または変更されました。**autoupgrade disk-cleanup**、**autoupgrade ida url**、**autoupgrade status email**、**debug autoupgrade**、**show autoupgrade configuration unknown**、**upgrade automatic abortversion**、**upgrade automatic getversion**、**upgrade automatic runversion**

用語集

CLI -- コマンドライン インターフェイス

IDA or Cisco server -- Intelligent Download Application

Cisco IOS -- Cisco Internetworking Operating System



第 30 章

ブート整合性の可視性について

ブート整合性の可視性によって、シスコのプラットフォーム ID とソフトウェアの整合性情報が可視化され、実用可能になります。プラットフォーム ID は、プラットフォームの製造元でインストールされた ID を提供します。ソフトウェアの整合性ではブート整合性の測定値が明らかになり、それを使用してプラットフォームが信頼できるコードを起動しているかどうかを評価できます。

ブートプロセス中に、ソフトウェアはブートローダーアクティビティの各ステージのチェックサムレコードを作成します。

このレコードを取得して、シスコ認定レコードと比較し、ソフトウェアイメージが正規かどうかを確認できます。チェックサム値が一致していない場合は、シスコによって認定されていない、または未承認パーティによって改ざんされているソフトウェアイメージを実行している可能性があります。

（注） ブート整合性の可視性は、アクティブなスーパーバイザでのみサポートされます。高可用性シナリオはサポートしていません。

- ソフトウェアイメージとハードウェアの確認 （379 ページ）
- プラットフォーム ID とソフトウェア整合性の確認 （380 ページ）

ソフトウェアイメージとハードウェアの確認

このタスクでは、ルータの起動時に作成されたチェックサムレコードを取得する方法について説明します。特権 EXEC モードで次のコマンドを入力します。

（注） 次のコマンドを実行した後で、メッセージ % Please Try After Few Seconds が CLI に表示されることがあります。これはCLIの障害を示すものではありませんが、必要な出力を取得するために必要な基盤となるインフラストラクチャの設定を示します。数分間待機して、コマンドを再度試すことをお勧めします。

メッセージ % Error retrieving SUDI certificate および % Error retrieving integrity data は、実際の CLI 障害を示します。

1. show platform sudi certificate [sign [nonce nonce]]
2. show platform integrity [sign [nonce nonce]]

プラットフォーム ID とソフトウェア整合性の確認

プラットフォーム ID の確認

```
Store-4451# show platform integrity
Platform: ISR4451-X/K9
Boot 0 Version: F01001R06.03c1d3d202013-01-18
Boot 0 Hash: 82597CE130610B8016A6A0FF2851919279857C86966540170E1132C6872A6274
Boot Loader Version: 16.7(4r)
Boot Loader Hash:
5F44054A51B69312283CE03255929D38D938351FDBE7F26A45DCEF6CB7F39C3078C65CB966D71DCF984865D30880AB8D65DD70DB31910B94B0AE290E8DA675E3
OS Version: BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127
OS Hashes:
isr4400-universalk9.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.bin:
8448067652482B991F562E7CB99FC1B1C1437BA7FC968A22C717AD1B5D36D1FE1331B6CCF5C5427FF9D88847D3B849DF482D92D0F631D00BD9A853C065DABEA1
isr4400-firmware_sm_dsp_sp2700.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

A667AFCD2B9819CE88725B90399131BDA06A0B9BFC0DC4835F02E6EFC23347C717DDB6A4659A8C33692344191931D32407EEAA1604F0C152222DE243D5E21D29
isr4400-firmware_nim_shdsl.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

A6E5D11706801FEF7B87B67B71A591176B05955CBA031EAABA23CC41AC715970819F06BD9A85AF945A338E99400211A5061D919C85FA3EC428457F0E498C06C0
isr4400-firmware_nim_ssd.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

F6F6418037171A6C941830BE8481A768C7CFB205F6A807B0196A54B8A2A607C78E6CA26F34BFEAB0C04D0CCA05A1AA5E8AECB6BC8CF7659E826A2F2DC39888DE
isr4400-firmware_nim_ge.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

97752B79EB8AE4925B74A94603CE5FEE5BF899945310 74C55935BF1C79065C474D21F3CF35A9F755110A6875ED425C0A14CA3400D3FB76C47CBEFA1B2A7E3216
isr4400-firmware_sm_async.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

0044338EE6A3E8A8AE61DAA599EB9A2A1B1BD78FBCE2880459FCCF9E750FD5852396777 55CC6ACE4BFDA1CED40A0F63D8A0DF5BAB4DF34DE11D4A42D8FCCFBCC
isr4400-firmware_sm_1t3e3.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

54DA6469D00FF20596FDAD7A2687ED6424180E73DA95A87848CE61143BAB51011866759B7CD21F4C77BFCB2219ECE6918A5F60F245E68BA2E22DFB3831CB1B2B
isr4400-firmware_dsp_analogbri.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

9B7B92DBF5B9E2574FC3668AGF2E4F1A0C20D4C895EF99016F51055E56D6195BA41DE31596E8F2D31B5C4B409207F3C04104304E9AEBC4461606B3614CD57F8C
isr4400-firmware_nim_xdsl.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

A2957CD30054993166 38B0AE943F77B02882F1B490899EB43E0D052ED57E299AD3FD82D58589AEC97275DB9AB6D12382C99DF41FD3722D40E01AB7E0201B739D
isr4400-firmware_dsp_sp2700.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

4A6422975EDDBD6367F40A0FB6C20888414BCD2D9C78A615F8C853584CE360079533B63B2AE9D10C1C4BCBE2F46F409525927A416E7275A34B2D513635486F54
isr4400-firmware_ngwic_t1e1.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

001B48A89716B10B6B506AAF562495DFD67E8DBF5BF4385A870E8A8B08BAB7A4F7D67230084A344AA9E40B037974E425A8CE289CDB47D06FDF759F56B5D30DB6
isr4400-firmware_sm_10g.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

1B9D88DC2E708015D65A913B42CEF7D42981D2E09EF9B9CEBDC94714F23C6D19D66B9CD5C72F51434A719BDDC640D9F88972BE9CAC742C894A69EBE55694FF67
isr4400-firmware_prince.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

E82220CCB45DD66C2A7A99DFA10758FE5AB8C217624BA623A83D1ADAF87FD08E4FC533C028D8C86B093184479BB064E36DB6255AA15A91381AE287070C1226E4
```

```
isr4400-firmware_dreamliner.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

50AE70E6C115E5339A1299E4ED8C123DE8EBC04CA9A45CA11B716C3013FFDCACD73D53FF043D6EEA36655A56F687247AF2D57176BE2142E0ACC506E64BD2A7DD
isr4400-firmware_nim_bri_st_fw.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

338EDCB41132394919D045E6B957D485F3ACBD160C7561EEF0A8155036DC695F6300291E1444E240975D9D02B45F4DFD36F36C5973D4DD9091DF6F71D9BB4157
isr4400-firmware_nim_cwan.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

AF6FD9A79D92382994FFA292E3129C47024E907E1AC05E13BA44F519D1B95863E7BC2E0BF9A2DD82D153A0D0159131CE034253ADC8ECA8E4662787834BC8DA5E
isr4400-firmware_nim_async.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

DE69E388865BC5B0144FBC96996F35143CA1E3920D84EDA1D97A08281289575B1FA0664CC7B81FC834B4FFA8C91DCF177CD5CA8323ED078B85374374F63DFD16
isr4400-mono-universalk9.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

6E9CCDCA9AD205B2713C0097A0B90B95B61FF267E3BC231916E8E1DE1650131F8168188E7F1CEE4F17A412B83C73D890A9EDC0409AB6EBB6F5AA687E043FE154
isr4400-firmware_dsp_tilegx.BLD_V1612_THROTTLE_LATEST_20190517_003908_V16_12_0_127.SSA.pkg:

BA452BCB66B279A97519397D6B90C8CF9C4CDF3BF74F41900EEDF0000D711EF03CE62C3B9878C314B5A339C16E0C963FD41C4DE86C3A36BDBD2481C49467B485
PCR0: C0F992411527603FE21E89331F95A1B9427B396C3210CFE47CD75B144A8A950E
PCR8: D767C72CEC698669B4A909423C56CA5527CF232217CF23B503B60D5C89275B20
```

ソフトウェア整合性の確認

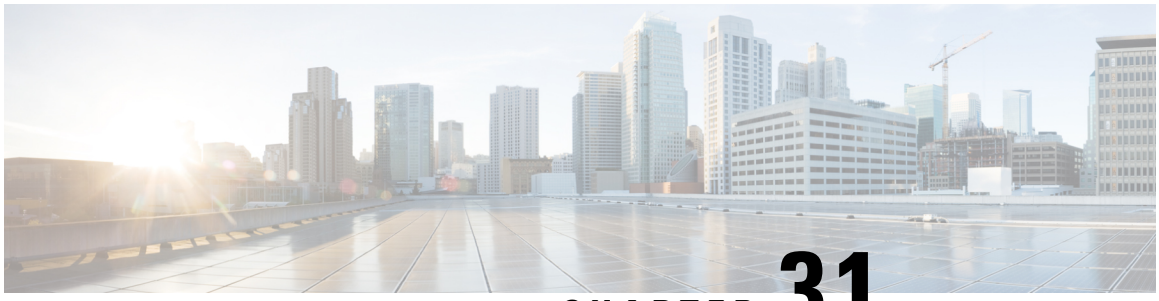
次に、ブート段階のチェックサム レコードを表示する例を示します。ハッシュ測定値は、連続してブートされたソフトウェアの 3 つの段階それぞれについて表示されます。これらのハッシュをシスコが提供する基準値と比較できます。出力に署名するオプションを使用すると、出力は正規であり改ざんされていないことを保証する機能が検証ツールに付与されます。リプレイ攻撃から保護するために、ナンスを提供できます。

```
show platform integrity sign nonce 123
Platform: C9300-24U
Boot 0 Version: F01144R16.216e68ad62019-02-13
Boot 0 Hash: 523DD459C650AF0F5AB5396060605E412C1BE99AF51F4FA88AD26049612921FF
Boot Loader Version: System Bootstrap, Version 17.1.1r, RELEASE SOFTWARE (P)
Boot Loader Hash:
34A2070D9EAE97E4FC4315A9BAF0E31FFD285E09F0B7F621955607A0FBC1D134ACC0068D8918F15B01975187458F6A46DF0F3DF9BA1593A3CD7BB4DF12487473
OS Version: BLD_POLARIS_DEV_LATEST_20191023_070152
OS Hashes:
cat9k_iosxe.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.bin:
8656F31DE26886F555B93258ADA7F354E083F1AFD22E676D3D83E956F6AA3307F9553E0D94FF752BD6E08DED5DAE067528CE44B16F3DD30A9FB4793E38BAE952
cat9k-wlc.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
33DDC53F932C9EC4CED2B402DA600511D2E2C5F4EF8037CE5D7D8E70B7050936D060467E7533FC7064073F6B3D9ED5AE53F756DD3493A38D564E96E7A49E25E5
cat9k-guestshell.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
4F2057EC660DCDE8FAE08CBE932E035338C7DE0A482B12CB443B506FA2298DE3B8FA1F805A28C0BBBFCDA089AE280E6953870161DD5E7F0C16C66A75FEB48546
cat9k-webui.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
45F3315C88E57A45F21A508C3771FADF0C8DB952F8848CA1C81F5588FFE466B9AF96295A8247DEFC47CD26A39D1802F0507109897297A4B5A86EFCADB3CFC261
cat9k-cc_srdriver.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EE6B1B1920145F5C978374ECB8374917E4E2825B059B7C95D409312C2C19271317AB349F775D4E1860DD0E22E2F68A961566A00466259D93323972F98E8B17E9
cat9k-srdriver.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EAF591B3945F14596A8C8AE8022722B6FC2073DFCEC4D24FE2518CAD7338F73A26F4AD29D00602A56E0B8EF6FAA4463239094EA8446D7B074AAF00930253C281
cat9k-sipbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
27155ECC5007A7A457C3E3263257613231 7EBF905972454C0305932B9A97591D37AFFC7AB40EC19E7B82DE042B31078309C38F4B81AA756F8D4180662D10F051
cat9k-sipspa.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
EDC255EC04D267055BE433D60F8CB4CCC426773C12442A291B15838E0D742F99CD45FD01B7E03AC139FDCA3143D83630052B45CCCBC834A84778CFEFF938CBC9
cat9k-espbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
65B0C8305E572247AAFE188A8C0B5081697CDD60BD8501FC2C88A8101862A63FED8B4AEF276D008F03F28978175FC3C4BF0B8FB3C238CDB619952F46CCF19CF1
cat9k-rpbase.BLD_POLARIS_DEV_LATEST_20191023_070152.SSA.pkg:
A297AA546323F63751F1CDA42558975D549E83A8D928A6CAFBD5A77AE19C6645620488E5A40E99FD8BE0F9726B12FF9591D3107825B885C9F7C7244FB31491F9
PCR0: 32E782AF9D75D12AC55BA5F67E9E8F375589CAF9C3558BC90E0EB969A84CDE95
PCR8: F0637823517D08D145F3E4DF207673D194FCB437E8B07170887E7AE279F88178
```

```
Signature version: 1
Signature:
```

第 VI 部

Cisco Discovery Protocol



CHAPTER 31

Cisco Discovery Protocol バージョン 2

Cisco Discovery Protocol（旧称 CDP）は、シスコ デバイス上で動作する、メディア独立型かつネットワーク独立型のレイヤ 2 プロトコルです。このプロトコルにより、ネットワーキング アプリケーションは直接接続された付近のデバイスに関して学習することができます。このプロトコルによってシスコ デバイスが検出されてその設定状態が特定され、異なるネットワーク層プロトコルを使用するシステムが相互に学習できるようになることで、デバイスの管理が容易になります。

ここでは、Cisco Discovery Protocol バージョン 2、およびその簡易ネットワーク管理プロトコル（SNMP）での動作について説明します。

Cisco Discovery Protocol の使用に関する前提条件

- インターフェイスがサブネットワーク アクセス プロトコル（SNAP）ヘッダーをサポートしている必要があります。

Cisco Discovery Protocol の使用に関する制約事項

- Cisco Discovery Protocol は、シスコ デバイス上でのみ動作します。
- Cisco Discovery Protocol は、フレームリレー マルチポイント サブインターフェイス上ではサポートされません。
- Cisco Discovery Protocol が有効になっているインターフェイス上にネイバーの IP アドレスがない場合、別のインターフェイスの IP アドレスが、非 IP アドレスインターフェイスの IP アドレスとして更新されます。

- Cisco Discovery Protocol は、カプセル化のデフォルトインターフェイスではサポートされていません。

Cisco Discovery Protocol の使用について

VLAN Trunking Protocol; VLANトランキング プロトコル

VLAN トランキング プロトコル（VTP）は、スイッチによって使用される検出技術です。スイッチは自身の管理ドメイン、コンフィギュレーション リビジョン番号、VLAN、および独自のパラメータをトランク ポートでアドバタイズします。VTP ドメインは、同じ VTP ドメイン名を共有する単一のデバイスまたは相互接続された複数のデバイスで構成されます。1 つのスイッチは 1 つの VTP ドメインにのみ属することができます。

Type-Length-Value フィールド

Type-Length-Value（TLV）フィールドは、Cisco Discovery Protocol アドバタイズメントに埋め込まれた情報ブロックです。アドバタイズメント内の情報はさまざまであり、必要に応じて、TLV フレーム フォーマットを使用してアドバタイズメントを拡張できます。次の表で TLV の定義を要約します。

***Table 39: Cisco Discovery Protocol* バージョン *2* の *Type-Length-Value* の定義**

TLV	定義
アドレス TLV	受信デバイスと送信デバイスの両方のネットワーク アドレスが含まれます。
アプリケーション TLV	Cisco Discovery Protocol を介してアプリケーション固有の TLV を送信するメカニズムを提供します。
機能 TLV	デバイスの機能を示すデバイス タイプを識別します（スイッチなど）。
デバイス ID TLV	文字列形式のデバイス名を識別します。
全二重/半二重 TLV	Cisco Discovery Protocol ブロードキャスト インターフェイスのデュプレックス設定を示します。この情報は、ネットワーク オペレータが隣接するネットワーク デバイス間の接続の問題を診断する際に使用します。
IP ネットワーク プレフィックス TLV	送信デバイスが IP パケットを転送できるネットワーク プレフィックスのリストが含まれます。プレフィクスには、インターフェイス プロトコルとポート番号が含まれます（Ethernet 1/0 など）。

TLV	定義
ロケーション TLV	Cisco Discovery Protocol を使用し、アクセス デバイス（スイッチまたはルータ）を通じてエンドポイントデバイスにロケーションベースの情報を提供します。ロケーション TLV では次の種類の情報を送信できます。 • 都市ロケーション情報：住所情報および郵便情報を提供します。たとえば、地名、番地、郵便番号などがあります。 • ELIN ロケーション情報：発信者のロケーション情報を提供します。ロケーションは、緊急ロケーション識別番号（ELIN）によって特定されます。ELIN は、緊急通報を現地の公安応答局（PSAP）にルーティングする電話番号で、PSAP はこれを使用して緊急通報者にコールバックすることができます。 Cisco Discovery Protocol によってエンドポイント デバイスにロケーションベース情報を提供するには、デバイスでロケーション TLV を設定しておく必要があります。ロケーション TLV の設定の詳細については、『*Using Link Layer Discovery Protocol in Multivendor Networks*』を参照してください。
ロケーションサーバ TLV	ロケーション サーバがネイバー デバイスに必要な情報を転送するためのメカニズムを提供します。
ネイティブ VLAN TLV	インターフェイス上の非タグ付きパケットに対して想定される VLAN をインターフェイス単位で示します。Cisco Discovery Protocol は、インターフェイスのネイティブ VLAN を認識します。 このフィールドは、IEEE 802.1Q プロトコルをサポートするインターフェイスに対してのみ実装されます。
プラットフォーム TLV	デバイスのハードウェア プラットフォームを識別します。たとえば Cisco 4500 などです。
ポート ID TLV	Cisco Discovery Protocol パケットが送信されるポートを識別します。
バージョン TLV	デバイスのソフトウェア リリース情報が含まれます。
VTP 管理ドメイン TLV	システムに設定された VLAN トランキング プロトコル（VTP）管理ドメイン名をアドバタイズします。この名前は、ネットワーク オペレータが隣接するネットワーク ノードの VTP ドメイン構成を確認する際に使用します。

Cisco Discovery Protocol

Cisco Discovery Protocol は、メディア独立型かつネットワーク独立型のレイヤ 2 プロトコルであり、ネットワーキング アプリケーションで、直接接続された付近のデバイスに関して学習するために使用されます。Cisco Discovery Protocol はデフォルトでイネーブルになっています。Cisco Discovery Protocol 用に設定された各デバイスは、メッセージを受信できるアドレスを 1 つ以上アドバタイズし、定期的なアドバタイズメント（メッセージ）を既知のマルチキャスト アドレス 01:00:0C:CC:CC:CC に送信します。デバイスは、このアドレスをリッスンすることによって相互に検出します。また、メッセージをリッスンすることにより、他のデバイス上のインターフェイスがアップまたはダウン状態になった時期を認識します。

アドバタイズメントには、存続可能時間情報が含まれます。この情報は、受信デバイスが Cisco Discovery Protocol 情報を廃棄するまでの保持時間の長さを示します。デフォルトで、シスコ ソフトウェアでサポートされている設定済みアドバタイズメントは、サブネットワーク アクセス プロトコル（SNAP）ヘッダーをサポートするインターフェイス上で 60 秒ごとに送信されます。シスコ デバイスは、Cisco Discovery Protocol パケットを転送しません。Cisco Discovery Protocol をサポートしているシスコ デバイスは、受信した情報をテーブルに保存します。このテーブル内の情報はアドバタイズメントを受信するたびに更新されます。また、アドバタイズメントの送信に 3 回失敗したデバイスに関する情報は廃棄されます。

Cisco Discovery Protocol アドバタイズメントに含まれる情報は、デバイス タイプおよびインストールされているオペレーティング システムのバージョンによって異なります。Cisco Discovery Protocol で学習できる情報には次のようなものがあります。

- シスコ デバイスで実行されている Cisco IOS バージョン
- デバイスのハードウェア プラットフォーム
- デバイス上のインターフェイスの IP アドレス
- Cisco Discovery Protocol をアドバタイズする、ローカル接続されているデバイス
- シスコ デバイス上のアクティブなインターフェイス（カプセル化タイプを含む）
- ホスト名
- デュプレックス設定
- VLAN トランキング プロトコル（VTP）ドメイン
- ネイティブ VLAN

Cisco Discovery Protocol バージョン 2 は、バージョン 1 よりさらにインテリジェントなデバイス トラッキング機能を備えています。使用できる機能の 1 つに、より迅速なエラー追跡を可能にする拡張レポート メカニズムがあります。これはネットワーク ダウンタイムの削減に役立ちます。レポートされるエラーには、接続ポートのネイティブ VLAN ID（IEEE 802.1Q）の不一致や、接続デバイス間のポートデュプレックス状態の不一致が含まれます。レポートされるエラーに関するメッセージが、コンソールまたはロギング サーバに送信される可能性があります。

show コマンドを使用して、ネイバーデバイスの VTP 管理ドメインとデュプレックスモード、Cisco Discovery Protocol に関連するカウンタ、接続ポートの VLAN ID に関する詳細な出力を取得できます。

Cisco Discovery Protocol と SNMP との併用

Cisco Discovery Protocol と簡易ネットワーク管理プロトコル（SNMP）を併用すると、ネットワーク管理アプリケーションはネイバー デバイスのデバイス タイプおよび SNMP エージェントアドレスを学習できます。アプリケーションはこれらのネイバーデバイスに SNMP クエリーを送信することもできます。

SNMP 管理アプリケーションは、これらのデバイスの SNMP エージェントから Cisco Discovery Protocol テーブルを取得することにより、プロトコルアドレスとネイバーデバイスのタイプを学習します。ネットワーク管理モジュール（NMM）の SNMP エージェントをイネーブルにすると、ネイバー デバイスが検出され、それらのデバイスに関する情報を含むローカル キャッシュが構築されます。管理ワークステーションは、SNMP 要求を送信して CISCO-CDP-MIB にアクセスすることにより、このキャッシュを取得できます。

ATM PVC の Cisco Discovery Protocol およびオンデマンド ルーティング サポート

Cisco Discovery Protocol およびオンデマンド ルーティング（ODR）は、ATM ポイントツーポイント相手先固定接続（PVC）でサポートされます。ODR では、Cisco Discovery Protocol を使用して、ハブアンドスポーク トポロジ内で IP アドレス情報が伝播されます。ODR がイネーブルになっていると、スポーク ルータは Cisco Discovery Protocol を使って自身のサブネットを自動的にアドバタイズします。

Cisco Discovery Protocol は、ATM PVC インターフェイス上ではデフォルトでディセーブルになっています。Cisco Discovery Protocol をイネーブルにするには、グローバル コンフィギュレーション モードで **cdp run** コマンド、インターフェイス コンフィギュレーション モードで **cdp enable** コマンドを使用します。これは、PVC の両端で行います。ODR をイネーブルにするには、ハブルータでグローバル コンフィギュレーション モードで **router odr** コマンドを使用し、スポークルータではすべてのダイナミックルーティング プロトコルをオフにします。ODR の設定の詳細については、『*IP Routing: ODR Configuration Guide*』の「Configuring On-Demand Routing」の項を参照してください。

IPv6 での Cisco Discovery Protocol のサポート

IPv6 でも、Cisco Discovery Protocol は IPv4 の場合と同様に機能し、同じ利点が提供されます。IPv6 の機能拡張により、Cisco Discovery Protocol は IPv6 情報およびネイバー アドレッシング情報を交換できます。この拡張機能は、ネットワーク管理製品およびトラブルシューティング ツールにも IPv6 情報を提供します。

Cisco Discovery Protocol の利点

Cisco Discovery Protocol には次の利点があります。

- 異なるネットワーク層プロトコルを使用するシステムが相互に学習できるようになります。
- シスコデバイスとその設定状況が検出されることで、デバイスの管理が容易になります。
- Type-Length-Value（TLV）フィールドのトラブルシューティングに役立ちます。
- SNMP エージェント アドレスを学習して SNMP クエリーを送信することにより、SNMP と連携します。

Cisco Discovery Protocol バージョン 2 の使用方法

シスコデバイスでのCiscoDiscoveryProtocolのディセーブル化とイネーブル化

Note システムの再起動時に、CDP はデフォルトで無効になります。CDP が有効になっている場合、CDP TLV アプリケーションもデフォルトで有効になります。CDP が無効な場合、CDP アプリケーション TLV も無効になります。ただし、CDP を再度有効にすると「no CDP app TLV」と表示され、これはデフォルトの動作とは異なります。

サポートされているデバイス上での Cisco Discovery Protocol のディセーブル化

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no cdp run**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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。

	Command or Action	Purpose
ステップ 3	**no cdp run** **Example:** `Device(config)# no cdp run`	サポートされているデバイス上で Cisco Discovery Protocol をディセーブルにします。
ステップ 4	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

サポートされているデバイス上での Cisco Discovery Protocol のイネーブル化

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp run**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**cdp run** **Example:** `Device(config)# cdp run`	サポートされているデバイス上で Cisco Discovery Protocol をイネーブルにします。
ステップ 4	**end** **Example:** `Device(config)# end`	コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

サポートされているインターフェイスでの Cisco Discovery Protocol のディセーブル化とイネーブル化

サポートされているインターフェイス上での Cisco Discovery Protocol のディセーブル化

インターフェイスのカプセル化を変更すると、Cisco Discovery Protocol を事前にディセーブル化していても、そのインターフェイスで Cisco Discovery Protocol が再度イネーブルになります。たとえばインターフェイスのカプセル化をPPPからハイレベルデータリンク制御（HDLC）に変更すると、そのインターフェイスで **no cdp run** コマンドによって Cisco Discovery Protocol を明示的にディセーブル化していても、再度イネーブルになります。この動作は設計によるものです。カプセル化により、そのインターフェイスに設定されているレイヤ2プロトコルが変更されて、インターフェイス コンフィギュレーションが Cisco Discovery Protocol のデフォルト状態であるイネーブルにリセットされます。このとき、Cisco Discovery Protocol はデバイス上でグローバルにイネーブルであると見なされます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **no cdp enable**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**interface** *type number* [*name-tag*] **Example:**	指定したインターフェイスを設定し、インターフェイス コンフィギュレーション モードを開始します。
ステップ 4	**no cdp enable** **Example:**	インターフェイス上で Cisco Discovery Protocol をディセーブルにします。

	Command or Action	Purpose
	`Device(config-if)# no cdp enable`	**Note** インターフェイスのカプセル化を変更すると、Cisco Discovery Protocol を事前にディセーブル化していても、そのインターフェイスで Cisco Discovery Protocol が再度イネーブルになります。
ステップ 5	**end** **Example:** `Device(config-if)# end`	特権 EXEC モードに戻ります。

例

次の例では、最初に Cisco Discovery Protocol をディセーブルにします。

```
Device(config)#
Device(config-if)# no ip address

Device(config-if)# shutdown
Device(config-if)# no cdp enable
! Cisco Discovery Protocol is disabled.
Device(config-if)# end
```

サポートされているインターフェイス上での Cisco Discovery Protocol のイネーブル化

Note インターフェイスのカプセル化を変更すると、Cisco Discovery Protocol を事前にディセーブル化していても、そのインターフェイスで Cisco Discovery Protocol が再度イネーブルになります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **cdp enable**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:**	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）。

	Command or Action	Purpose
	`Device> enable`	
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	**interface** *type number* [*name-tag*] **Example:** `Device(config)# interface Gigabitethernet 1/0/1`	指定されたインターフェイスを設定し、CLI をインターフェイス コンフィギュレーション モードにします。 **Note** インターフェイスのカプセル化を変更すると、Cisco Discovery Protocol を事前にディセーブル化していても、そのインターフェイスで Cisco Discovery Protocol が再度イネーブルになります。
ステップ 4	**cdp enable** **Example:** `Device(config-if)# cdp enable`	インターフェイスで Cisco Discovery Protocol をイネーブルにします。
ステップ 5	**end** **Example:** `Device(config-if)# end`	CLI を特権 EXEC モードに戻します。

送信タイマーと保持時間の設定

Cisco Discovery Protocol の送信周波数と Cisco Discovery Protocol パケットの保持時間を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp timer** *seconds*
4. **cdp holdtime** *seconds*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**cdp timer** *seconds* **Example:** `Device(config)# cdp timer 30`	Cisco Discovery Protocol パケットの送信頻度を指定します。
ステップ 4	**cdp holdtime** *seconds* **Example:** `Device(config)# cdp holdtime 90`	受信デバイスが情報を廃棄するまでの保持時間を指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	特権 EXEC モードを開始します。

Cisco Discovery Protocol バージョン 2 アドバタイズメントのディセーブル化と再イネーブル化

シスコ デバイス上では、Cisco Discovery Protocol バージョン 2 アドバタイズメントのブロードキャストはデフォルトでイネーブルになっています。このブロードキャストをディセーブルまたは再度イネーブルにするには、次の作業を実行します。

Cisco Discovery Protocol バージョン 2 アドバタイズメントのディセーブル化

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no cdp advertise-v2**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**no cdp advertise-v2** **Example:** `Device(config)# no cdp advertise-v2`	Cisco Discovery Protocol バージョン 2 アドバタイズメントのブロードキャストをディセーブルにします。
ステップ 4	**end** **Example:** `Device(config)# end`	特権 EXEC モードに戻ります。

Cisco Discovery Protocol バージョン 2 アドバタイズメントのイネーブル化

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **cdp advertise-v2**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**cdp advertise-v2** **Example:**	Cisco Discovery Protocol バージョン 2 アドバタイズメントのブロードキャストをイネーブルにします。

	Command or Action	Purpose
	`Device(config)# cdp advertise-v2`	
ステップ 4	**end** **Example:** `Device(config)# end`	特権 EXEC モードに戻ります。

Cisco Discovery Protocol のモニタリングとメンテナンス

デバイスで Cisco Discovery Protocol 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。この作業およびすべての手順は省略可能です。また、手順は任意の順序で実行できます。

SUMMARY STEPS

1. **enable**
2. **clear cdp counters**
3. **clear cdp table**
4. **show cdp**
5. **show cdp entry** *device-name* [**protocol** | **version**]
6. **show cdp interface** [*type number*]
7. **show cdp neighbors** [*type number*] [**detail**]
8. **show cdp traffic**
9. **show debugging**

DETAILED STEPS

	Command or Action	Purpose
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	**clear cdp counters** **Example:** `Device# clear cdp counters`	Cisco Discovery Protocol のトラフィック カウンタを 0 にリセットします。
ステップ 3	**clear cdp table** **Example:** `Device# clear cdp table`	ネイバーに関する Cisco Discovery Protocol 情報を含むテーブルをクリアします。

	Command or Action	Purpose
ステップ 4	**show cdp** **Example:** `Device# show cdp`	アドバタイズメントの間隔、指定されたポートに対するアドバタイズメントの有効期間（秒単位）、アドバタイズメントのバージョンを表示します。
ステップ 5	**show cdp entry** *device-name* [**protocol** | **version**] **Example:** `Device# show cdp entry test-device protocol`	特定のネイバーに関する情報を表示します。
ステップ 6	**show cdp interface** [*type number*] **Example:** `Device# show cdp interface`	Cisco Discovery Protocol がイネーブルになっているインターフェイスに関する情報を表示します。
ステップ 7	**show cdp neighbors** [*type number*] [**detail**] **Example:** `Device# show cdp neighbors`	検出されたデバイスのタイプ、デバイスの名前、ローカルインターフェイス（ポート）の番号とタイプ、インターフェイスに対する Cisco Discovery Protocol アドバタイズメントの有効期間（秒数）、デバイスのタイプ、デバイスの製品番号、およびポート ID を表示します。 • **detail** キーワードを使用すると、ネイティブ VLAN ID、デュプレックスモード、およびネイバーデバイスに関連付けられた VTP ドメイン名に関する情報が表示されます。
ステップ 8	**show cdp traffic** **Example:** `Device# show cdp traffic`	Cisco Discovery Protocol トラフィックに関する情報（送受信されたパケット数とチェックサムエラー数を含む）を表示します。
ステップ 9	**show debugging** **Example:** `Device# show debugging`	デバイスに対してイネーブルになっているデバッグのタイプに関する情報を表示します。

Cisco Discovery Protocol バージョン 2 の設定例

例：送信タイマーと保持時間の設定

次の例では、30秒ごとにアップデートを送信するようにタイマーが設定され、アップデートが有効であることを確認するために **show cdp interface** コマンドが使用されます。

```
Device(config)# cdp timer 30
Device(config)# end
Device# show cdp interface

Serial0 is up, line protocol is up
Encapsulation is HDLC
Sending CDP packets every 30 seconds
Holdtime is 180 seconds
```

I次の例では、保持時間が 90 秒に設定され、アップデートが有効であることを確認するために **show cdp interface** コマンドが使用されます。

```
Device(config)# cdp holdtime 90
Device(config)# end
Device# show cdp interface

Serial0 is up, line protocol is up
Encapsulation is HDLC
Sending CDP packets every 30 seconds
Holdtime is 90 seconds
```

例：Cisco Discovery Protocol のモニタリングとメンテナンス

次に、Cisco Discovery Protocol 情報を表示するために使用できる一連のコマンドの例を示します。

Cisco Discovery Protocol バージョン 2 に関する追加情報

関連資料

関連項目	マニュアル タイトル
Cisco Discovery Protocol のコマンド	Cisco IOS Cisco Discovery Protocol Command Reference
SNMP サポートの設定タスク	「Configuring SNMP Support」モジュール
オンデマンドルーティングの設定タスク	「Configuring On-Demand Routing」の章
debug コマンド	Cisco IOS Debug Command Reference

標準

標準	タイトル
IEEE 802.1Q	仮想 *LAN*

MIB

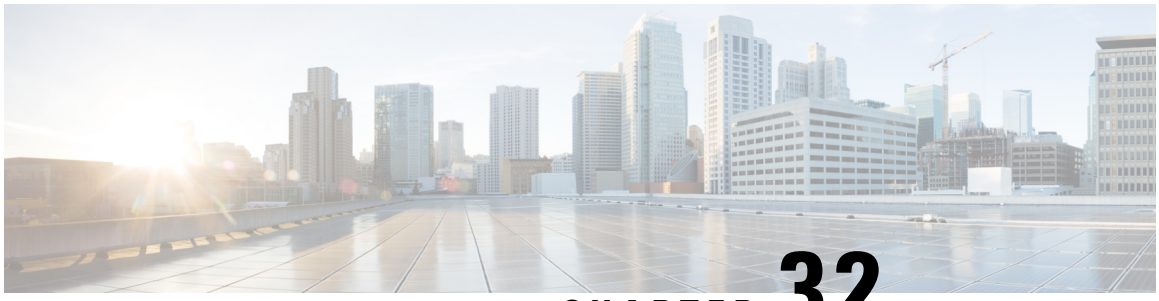
MIB	MIB のリンク
CISCO-CDP-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

第 VII 部

メディア モニタリング



CHAPTER 32

Cisco Mediatrace の設定

この章には、Cisco Mediatrace の設定に関する情報と説明が記載されています。

Cisco Mediatrace を使用すると、データ ストリームに関するネットワーク パフォーマンス低下の問題の切り分けを行ってトラブルシューティングできます。任意のタイプのフローをモニタするために使用できますが、主にビデオ フローで使用されます。また、メディア フロー パスに沿った、フロー以外に関連する監視にも使用できます。

Cisco Mediatrace の設定に関する情報

Cisco Mediatrace の概要

Note Mediatrace は M&T トレインではサポートされなくなりました。パフォーマンスモニタリングについては、「Cisco Performance Monitor の設定, on page 439」の章を参照してください。

Cisco Mediatrace は、IP フローのパスをネットワーク管理者が検出できるようにしてパフォーマンスの低下の問題を切り分けてトラブルシューティングを行う場合や、パス上のノードでモニタリング機能を動的に有効にする際や、ネットワークホップバイホップベースで情報を収集するのに役立ちます。この情報には、特に、フロー統計情報の他、着信および発信インターフェイス、CPU、ならびにメモリの使用率情報、さらに IP ルートまたは Cisco Mediatrace のモニタリング状態の変更が含まれます。

この情報は、次の 2 つのうちのいずれかの方法で取得できます。

- exec コマンドを発行して、メディア フロー上のホップからの統計情報のオンデマンド収集を実行します。この単発の操作では、メディア フロー上のホップが検出され、指定された他の一連の情報と共に表示されます。
- 特定の日の特定の時刻に定期モニタリング セッションが開始されるように Cisco Mediatrace を設定します。セッションを設定して、収集対象のメトリックおよびデータの収集頻度を指定することができます。パス上のホップの検出は操作の一部として自動的に実行されます。

指定したメトリックが収集されたら、それらのメトリックに関するレポートを表示できます。

Cisco Mediatrace は、Cisco Medianet 製品ファミリの一部です。Medianet を他のシスコ製品と併せて使用する場合の設計、設定、およびトラブルシューティングの詳細については、クイック スタート ガイドや導入ガイドも含めて、Cisco Medianet ナレッジ ベース ポータル（http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html）を参照してください。

Cisco Mediatrace を使用して収集できるメトリック

Mediatrace を使用して、次のカテゴリのメトリックを収集できます。

- 各レスポンダの共通メトリック
- システム メトリック：TCP プロファイル（System Metrics: TCP Profile）
- システム メトリック：RTP プロファイル（System Metrics: RTP Profile）
- システム メトリック：INTF プロファイル（System Metrics: INTF Profile）
- システム メトリック：CPU プロファイル（System Metrics: CPU Profile）
- システム メトリック：メモリ プロファイル（System Metrics: MEMORY Profile）
- アプリケーション ヘルス メトリック：Mediatrace ヘルス プロファイル（App-Health Metrics: MEDIATRACE-HEALTH Profile）
- イニシエータからの Mediatrace 要求サマリに関するメトリック（Metrics for Mediatrace Request Summary from Initiator）

これらのカテゴリそれぞれに含まれる個々のメトリックを、以下の該当セクションに示します。

イニシエータからの Mediatrace 要求サマリに関するメトリック（Metrics for Mediatrace Request Summary from Initiator）

- 要求タイムスタンプ（Request Timestamp）
- リクエストのステータス（Request Status）
- 応答ホップの数（Number of Hops Responded）
- 有効データを含むホップの数（Number of Hops with Valid Data）
- エラーを含むホップの数（Number of Hops with Error）

- データ レコードがないホップの数（Number of hops with no data record）
- 前回のルート変更のタイムスタンプ（Last Route Change Timestamp）
- ルート インデックス（Route Index）

各レスポンダの共通メトリック

- メトリック収集ステータス（Metrics Collection Status）
- 到達可能性アドレス（Reachability address）
- 入力インターフェイス（Ingress Interface）
- 出力インターフェイス（Egress Interface）
- Mediatrace IP TTL
- ホスト名（Hostname）
- Mediatrace ホップ数（Mediatrace Hop Count）

パフォーマンス モニタ メトリック：TCP プロファイル（Perf-Monitor Metrics: TCP Profile）

- フロー サンプリング開始タイムスタンプ（Flow Sampling Start Timestamp）
- 測定の信頼性の喪失（Loss of measurement confidence）
- 発生したメディア停止イベント（Media Stop Event Occurred）
- IP パケット ドロップ数（IP Packet Drop Count）
- IP バイト数（IP Byte Count）
- IP パケット数（IP Packet Count）
- IP バイト レート（IP Byte Rate）
- IP DSCP
- IP TTL
- IP プロトコル（IP Protocol）
- メディア バイト数（Media Byte Count）
- TCP 接続ラウンド トリップ遅延（TCP Connect Round Trip Delay）
- TCP 損失イベント数（TCP Lost Event Count）

パフォーマンス モニタ メトリック：RTP プロファイル（Perf-Monitor Metrics: RTP Profile）

- フロー サンプリング開始タイムスタンプ（Flow Sampling Start Timestamp）
- 測定の信頼性の喪失（Loss of measurement confidence）

- 発生したメディア停止イベント（Media Stop Event Occurred）
- IP パケット ドロップ数（IP Packet Drop Count）
- IP バイト数（IP Byte Count）
- IP パケット数（IP Packet Count）
- IP バイト レート（IP Byte Rate）
- パケット ドロップの理由（Packet Drop Reason）
- IP DSCP
- IP TTL
- IP プロトコル（IP Protocol）
- メディア バイト レート（平均）（Media Byte Rate Average）
- メディア バイト数（Media Byte Count）
- メディア パケット数（Media Packet Count）
- RTP 到着間ジッター（平均）（RTP Interarrival Jitter Average）
- RTP パケット損失（RTP Packets Lost）
- 予想 RTP パケット（pkts）（RTP Packets Expected (pkts)）：
- RTP パケット損失イベント数（RTP Packet Lost Event Count）：
- RTP 損失率（RTP Loss Percent）

システム メトリック：INTF プロファイル（System Metrics: INTF Profile）

- 収集タイムスタンプ（Collection timestamp）
- オクテット入力（入力）（Octet input at Ingress）
- オクテット出力（出力）（Octet output at Egress）
- エラーを含む受信パケット（入力）（Packets received with errors at Ingress）
- エラーを含むパケット（出力）（Packets with errors at Egress）
- 廃棄されたパケット（入力）（Packets discarded at Ingress）
- 廃棄されたパケット（出力）（Packets discarded at Egress）
- 入力インターフェイス速度（Ingress interface speed）
- 出力インターフェイス速度（Egress interface speed）

システム メトリック：CPU プロファイル（System Metrics: CPU Profile）

- CPU 使用率（1 分間）（CPU Utilization (1min)）
- CPU 使用率（5 分間）（CPU Utilization (5min)）
- 収集タイムスタンプ（Collection timestamp）

システム メトリック：メモリ プロファイル（System Metrics: MEMORY Profile）

- プロセッサ メモリ使用率（%）（Processor memory utilization %）
- 収集タイムスタンプ（Collection timestamp）

アプリケーション ヘルス メトリック：Mediatrace ヘルス プロファイル（App-Health Metrics: MEDIATRACE-HEALTH Profile）

- 受信された要求（Requests Received）
- 前回の要求受信時刻（Time Last Request Received）
- 前回の要求のイニシエータ（Initiator of Last Request）
- ドロップされた要求（Requests Dropped）
- サポートされている最大同時セッション数（Max Concurrent Sessions supported）
- 現在アクティブなセッション（Sessions currently active）
- 切断されたセッション（Sessions Teared down）
- タイムアウトが発生したセッション（Sessions Timed out）
- 受信されたホップ情報要求（Hop Info Requests Received）
- 受信された Performance Monitor 要求（Performance Monitor Requests Received）
- 失敗した Performance Monitor 要求（Performance Monitor Requests failed）
- 受信された静的ポリシー要求（Static Policy Requests Received）
- 失敗した静的ポリシー要求（Static Policy Requests Failed）
- 受信されたシステム データ要求（System Data Requests Received）
- 失敗したシステム データ要求（System Data Requests Failed）
- 受信されたアプリケーション ヘルス要求（Application Health Requests Received）
- ローカル ルート変更イベント（Local route change events）
- 前回のルート変更イベントの時刻（Time of last route change event）
- 受信された不明な要求の数（Number of unknown requests received）

Cisco Mediatrace の設定の概要

Cisco Mediatrace では、次のいずれかの方法で情報を取得できます。

- 事前にスケジュールされた定期モニタリング セッション。
- Mediatrace ポーリングとして知られる、オンデマンドでの単発の統計情報の収集。

Mediatrace セッションまたはポーリングを実装する前に、フロー情報の収集先の各ネットワーク ノードで Mediatrace を有効にしておく必要があります。Mediatrace セッションまたはポーリングを設定、開始、および制御するために使用するネットワーク ノードで Mediatrace Initiator をイネーブルにする必要があります。情報の収集先の各ネットワーク ノードで、Mediatrace Responder を有効にする必要があります。

Cisco Mediatrace セッションを設定するには、事前にパッケージ化されている次の 2 つのタイプのプロファイルのいずれかをセッションと関連付けることにより、セッションパラメータを設定できます。

- ビデオ モニタリング プロファイル
- システム データ プロファイル

また、次のタイプのプロファイルを設定してセッションと関連付けることにより、独自のパラメータを Cisco Mediatrace セッションに設定できます。

- パス指定子プロファイル
- フロー指定子プロファイル
- セッション パラメータ プロファイル

したがって、次のセクションでは、Cisco Mediatrace セッションを設定するために以下の作業を実行する方法について説明します。

1. Mediatrace の有効化
2. ビデオ モニタリング プロファイルのセットアップ
3. システム データ プロファイルのセットアップ
4. パス指定子プロファイルのセットアップ
5. フロー指定子プロファイルのセットアップ
6. セッション パラメータ プロファイルのセットアップ
7. プロファイルと Mediatrace セッションとの関連付ける
8. Mediatrace セッションのスケジュール設定

また、次のセクションでは、特定のパスのホップからデータをオンデマンドで取得する Mediatrace ポーリングを実行する方法についても説明します。

さらに、次のセクションでは、以下の作業を実行してMediatraceセッションを管理する方法について説明します。

- 未完了の Cisco Mediatrace セッションのクリア
- Cisco Mediatrace セッションのトラブルシューティング

制限事項

- Mediatrace は IPv6 をサポートしていません。
- Resource Reservation Protocol（RSVP）は、同じインターフェイス上では着信 Path メッセージを転送しません（つまり、Pathメッセージの送信元のインターフェイス経由では転送されないということです）。その場合、「出力インターフェイスが入力インターフェイスと同じです（ingress interface = egress interface）」というエラー メッセージが表示されます。ただし、Performance Routing（PfR）境界ルータの場合は、着信インターフェイスで Path メッセージが送信されます。

Cisco Mediatrace の設定方法

Cisco Mediatrace の有効化

Cisco Mediatrace を使用してモニタするノードごとに、少なくとも 1 つの Cisco Mediatrace レスポンダを有効にする必要があります。また、Mediatrace セッションまたはポーリングを開始するすべてのノードについて、Cisco Mediatrace イニシエータを有効にする必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace initiator** {**source-ip** ip-address | **source-interface** *interface-name*} [**force**] **[max-sessions** *number* **]**
4. **mediatrace responder [max-sessions** *number* **]**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:**	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。

	Command or Action	Purpose
	`Router# configure terminal`	
ステップ 3	**mediatrace initiator** {**source-ip** ip-address | **source-interface** *interface-name*} [**force**] **[max-sessions** *number* **]** **Example:** `Router(config)# mediatrace initiator source-ip 10.10.1.1 max-sessions 4`	Cisco Mediatrace またはイニシエータを有効にします。次のキーワードを使用することもできます。 • *ip-address*：常に到達可能な IP アドレス。 • *interface-name*：イニシエータに接続する任意のローカルインターフェイス。 • **max-sessions** ：Cisco Mediatrace セッションの数を設定します。
ステップ 4	**mediatrace responder [max-sessions** *number* **]** **Example:** `Router(config)# mediatrace responder max-sessions 4`	Cisco Mediatrace レスポンダを有効にします。次のキーワードを使用することもできます。 • **max-sessions** ：Cisco Mediatrace セッションの数を設定します。
ステップ 5	**end** **Example:** `Router(config)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace responder app-health コマンドを使用して、レスポンダがイベント、要求、および Cisco Mediatrace に関連するその他の統計情報を正しく収集しているかどうかを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Mediatrace イニシエータでの Cisco Mediatrace ビデオ プロファイルの設定

Cisco Mediatrace には、事前にパッケージ化されたビデオ モニタリング プロファイルが用意されています。このパッケージには、ビデオ メディア モニタリング セッションを開始するために必要なすべてのパラメータ設定が含まれています。また、Mediatrace イニシエータに独自のビデオ モニタリング プロファイルを設定することもできます。

新しいビデオ メディア モニタリング セッションを開始するため、それらのプロファイルの 1 つを関連付けて Cisco Mediatrace セッションを設定することができます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace profile perf-monitor** *name*
4. admin-params
5. **sampling-interval** *seconds*
6. **exit**
7. **metric-list** {**tcp | rtp**}
8. **clock-rate** {*type-number* | *type-name*} *rate*
9. **max-dropout** *number*
10. **max-reorder** *number*
11. **min-sequential** *number*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace profile perf-monitor** *name* **Example:** `Router(config)# mediatrace profile perf-monitor vprofile-2`	パフォーマンス プロファイル コンフィギュレーション モードを開始して、事前にパッケージ化されている Cisco Mediatrace ビデオ モニタリング プロファイルのパラメータを設定できるようにします。
ステップ 4	admin-params **Example:** `Router(config-mt-prof-perf)# admin-params`	管理者パラメータ コンフィギュレーション モードを開始して、ビデオ モニタリング管理者パラメータを設定できるようにします。
ステップ 5	**sampling-interval** *seconds* **Example:** `Router(config-mt-prof-perf-params)# sampling-interval 40`	ビデオ モニタリング メトリックのサンプリング間隔（秒）を指定します。

	Command or Action	Purpose
ステップ 6	**exit** **Example:** `Router(config-mt-prof-perf-params)# exit`	現在のコンフィギュレーション モードを終了し、パフォーマンスプロファイル コンフィギュレーション モードに戻ります。
ステップ 7	**metric-list** {**tcp** | **rtp**} **Example:** `Router(config-mt-prof-perf)# metric-list rtp`	モニタ対象のメトリックが TCP と RTP のどちらに関するものであるかを指定します。
ステップ 8	**clock-rate** {*type-number* | *type-name*} *rate* **Example:** `Router(config-mt-prof-perf-rtp-params)# clock-rate 64`	（任意）RTP ビデオ モニタリング メトリックのサンプリングに使用するクロック レートを指定します。各ペイロード タイプには、関連付けられている特定のクロック レートがあります。それらは、タイプ番号とタイプ名のいずれかを使用して指定できます。ペイロード タイプ名で使用できる値の詳細については、『Cisco Media Monitoring Command Reference』を参照してください。
ステップ 9	**max-dropout** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# max-dropout 2`	（任意）RTP ビデオ モニタリング メトリックのサンプリング時に許可されるドロップアウトの最大数を指定します。ドロップアウトは、シーケンス番号が現在のパケットよりも古いものとして無視されるパケットの数を指定します。
ステップ 10	**max-reorder** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# max-reorder 4`	（任意）RTP ビデオ モニタリング メトリックのサンプリング時に許可される順序変更の最大数を指定します。順序変更は、シーケンス番号が現在のパケットよりも新しいものとして無視されるパケットの数を指定します。
ステップ 11	**min-sequential** *number* **Example:** `Router(config-mt-prof-perf-rtp-params)# min-sequential 2`	（任意）RTP フローの分類に使用される連続パケットの最小数を指定します。
ステップ 12	**end** **Example:** `Router(config-mt-prof-perf-rtp-params)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace profile perf-monitor を使用して、事前にパッケージ化されているビデオモニタリング プロファイルのパラメータ値が正しく設定されていることを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace のシステム プロファイルの設定

Cisco Mediatrace には、事前にパッケージ化されたシステム データ モニタリング プロファイルが用意されています。このパッケージには、システム データ モニタリング セッションを開始するために必要なすべてのパラメータ設定が含まれています。また、独自のシステム データ モニタリング プロファイルを設定することもできます。新しいシステム データ モニタリング セッションを開始するため、それらのプロファイルの 1 つを関連付けて Cisco Mediatrace セッションを設定することができます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace profile system** *name*
4. **metric-list** {**intf** | **cpu** | **memory**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace profile system** *name* **Example:** `Router(config)# mediatrace profile system system-2`	システムプロファイルコンフィギュレーションモードを開始して、Cisco Mediatrace のシステム プロファイルのパラメータを設定できるようにします。
ステップ 4	**metric-list** {**intf**	**cpu**	**memory**} **Example:** `Router(config-sys)# metric-list memory`	モニタ対象のメトリックが、インターフェイス、CPU、メモリのいずれに関するものであるかを指定します。
ステップ 5	**end** **Example:** `Router(config-sys)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace profile system コマンドを使用して、事前にパッケージ化されているシステム データプロファイルのパラメータ値が正しく設定されていることを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace のパス指定子プロファイルの設定

Cisco Mediatraceセッションの設定では、パス指定子プロファイルを指定する必要があります。このプロファイルは、トラブルシューティングのためにモニタされるネットワーク ホップの検出に使用されるパラメータを定義します。オプションの**disc-proto** キーワードで指定するRSVP トランスポートプロトコルは、このホップ検出を実行するために使用されます。フロー指定子のパラメータ値は、トレース対象のメディア フローの値と一致している必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace path-specifier** *name* [**disc-proto rsvp**] {**gsid** *gsid* | **destination ip** *ip-address* **port** *nnnn* }
4. **source ip** *ip-address* **port** *nnnn*
5. **l2-params gateway** *ip-address* **vlan** *vlan-id*
6. **gsid** *gsid*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace path-specifier** *name* [**disc-proto rsvp**] {**gsid** *gsid* | **destination ip** *ip-address* **port** *nnnn* } **Example:** `Router(config)# mediatrace path-specifier path-4 disc-proto rsvp destination ip 10.1.1.1 port 400`	パス指定子コンフィギュレーション モードを開始して、Cisco Mediatrace のパス指定子プロファイルのパラメータを設定できるようにします。このコマンドでは、パスの名前、宛先アドレス、およびポートを指定する必要があります。

	Command or Action	Purpose
ステップ 4	**source ip** *ip-address* **port** *nnnn* **Example:** `Router(config-mt-path)# source ip 10.1.1.2 port 600`	モニタ対象のメトリックの送信元IPアドレスを指定します。
ステップ 5	**l2-params gateway** *ip-address* **vlan** *vlan-id* **Example:** `Router(config-mt-path)# l2-params gateway 10.10.10.4 vlan 22`	レベル 2 ゲートウェイの仮想 LAN の IP アドレスと ID を指定します。 **Note** このコマンドは、Catalystプラットフォームのみで使用できます。
ステップ 6	**gsid** *gsid* **Example:** `Router(config-mt-path)# gsid 60606060`	モニタ対象のフローのメタデータ グローバル セッション ID を指定します。
ステップ 7	**end** **Example:** `Router(config-mt-path)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace path-specifier コマンドを使用して、パス指定子プロファイルのパラメータ値が正しく設定されていることを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace のフロー指定子プロファイルの設定

Cisco Mediatrace セッションの設定では、フロー指定子プロファイルを指定する必要があります。このプロファイルは、フローを識別するための送信元 IP アドレス、宛先 IP アドレス、送信元ポート、宛先ポート、およびプロトコルを定義します。プロファイルは、後で実際のCisco Mediatrace セッションを設定するときに関連付けることができます。

RTP メディア フローについては、UDP をプロトコルとして選択します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace flow-specifier** *name*
4. **source-ip** *ip-address* [**source-port** *port*]
5. **dest-ip** *ip-address* [**dest-port** *port*]

6. **gsid** *gsid*
7. **ip-protocol** {**tcp** | **udp**}
8. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace flow-specifier** *name* **Example:** `Router(config)# mediatrace flow-specifier flow-6`	フロー指定子コンフィギュレーションモードを開始して、Cisco Mediatrace のフロー指定子プロファイルのパラメータを設定できるようにします。
ステップ 4	**source-ip** *ip-address* [**source-port** *port*] **Example:** `Router(config-mt-flowspec)# source-ip 10.1.1.2 source-port 600`	（任意）モニタ対象のメトリックの送信元 IP アドレスを指定します。
ステップ 5	**dest-ip** *ip-address* [**dest-port** *port*] **Example:** `Router(config-mt-flowspec)# dest-ip 10.1.1.2 dest-port 600`	モニタ対象のメトリックの宛先 IP アドレスを指定します。
ステップ 6	**gsid** *gsid* **Example:** `Router(config-mt-flowspec)# gsid 60606060`	モニタ対象のフローのメタデータ グローバル セッション ID を指定します。
ステップ 7	**ip-protocol** {**tcp**	**udp**} **Example:** `Router(config-mt-flowspec)# ip-protocol tcp`	モニタ対象のメトリックが TCP と UDP のどちらに関するものであるかを指定します。
ステップ 8	**end** **Example:** `Router(config-mt-flowspec)# end`	現在のコンフィギュレーションモードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace flow-specifier コマンドを使用して、パス指定子プロファイルのパラメータ値が正しく設定されていることを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace のセッション パラメータ プロファイルの設定

Cisco Mediatrace セッションの設定では、セッション パラメータ プロファイルを指定する必要があります。このプロファイルは、Cisco Mediatrace セッションの特性を定義し、Cisco Mediatrace セッションの動作を円滑化するのに役立ちます。プロファイルは、後で実際の Cisco Mediatrace セッションを設定するときに関連付けることができます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace session-params** *name*
4. **response-timeout** *seconds*
5. **frequency** {*frequency* | **on-demand**} **inactivity-timeout** *seconds*
6. **history** *buckets*
7. **route-change reaction-time** *seconds*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace session-params** *name* **Example:** `Router(config-mt-sesparam)# mediatrace session-params qos-2`	セッションパラメータ コンフィギュレーションモードを開始して、Cisco Mediatrace のセッション パラメータ プロファイルのパラメータを設定できるようにします。
ステップ 4	**response-timeout** *seconds* **Example:**	イニシエータがレスポンダからの応答を待機する時間（秒）を指定します。

	Command or Action	Purpose
	`Router(config-mt-sesparam)# response-timeout 8`	
ステップ 5	**frequency** {*frequency* | **on-demand**} **inactivity-timeout** *seconds* **Example:** `Router(config-mt-sesparam)# frequency 4 inactivity-timeout 2`	セッション パラメータ メトリックのサンプリング間隔（秒）、およびレスポンダからのアクティビティがない場合にイニシエータがアクティブ状態を維持する時間（秒）を指定します。
ステップ 6	**history** *buckets* **Example:** `Router(config-mt-sesparam)# history 2`	保持する履歴データ セットの数を指定します（最大値は 10）。
ステップ 7	**route-change reaction-time** *seconds* **Example:** `Router(config-mt-sesparam)# route-change reaction-time 8`	追加のルート変更に対するレスポンダの反応をイニシエータが待機する時間（秒）を指定します。範囲は秒単位です。
ステップ 8	**end** **Example:** `Router(config-mt-sesparam)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace session-param コマンドを使用して、セッション パラメータ プロファイルのパラメータ値が正しく設定されていることを確認します。

このコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace セッションの設定

Cisco Mediatrace セッションの設定は、さまざまなプロファイルをセッションに関連付けます。Cisco Mediatrace セッションに関連付けできるプロファイルは各タイプで 1 つだけです。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace** *session-number*
4. **trace-route**
5. **path-specifier** {[**forward**] *path-name* | **reverse** *path-name* }
6. **session-params** *name*

7. **profile system** *name*
8. **profile perf-monitor** *name* **flow-specifier** *flow-specifier-name*
9. **profile snmp** *name*
10. **profile custom** *name*
11. **last-node** { **auto** | **address** *address* }
12. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace** *session-number* **Example:** `Router(config)# mediatrace 157`	セッション コンフィギュレーション モードを開始します。
ステップ 4	**trace-route** **Example:** `Router(config-mt-session)# trace-route`	Cisco Mediatrace セッションのトレース ルートの実行を有効にします。デフォルトでは、トレースルートは有効になっています。トレースルートの実行を停止にするには、このコマンドの **no** 形式を使用します。
ステップ 5	**path-specifier** {[**forward**] *path-name* | **reverse** *path-name* } **Example:** `Router(config-mt-session)# path-specifier path-4`	パス指定子プロファイルを Cisco Mediatrace セッションに関連付けます。
ステップ 6	**session-params** *name* **Example:** `Router(config-mt-session)# session-params session-6`	セッション パラメータ プロファイルを Cisco Mediatrace セッションに関連付けます。
ステップ 7	**profile system** *name* **Example:** `Router(config-mt-session)# profile system sys-2`	システム プロファイルを Cisco Mediatrace セッションに関連付けます。

	Command or Action	Purpose
ステップ 8	**profile perf-monitor** *name* **flow-specifier** *flow-specifier-name* **Example:** `Router(config-mt-session)# profile perf-monitor monitor-6 flow-specifier flow-4`	パフォーマンス モニタ プロファイルおよびフロー指定子を Cisco Mediatrace セッションに関連付けます。
ステップ 9	**profile snmp** *name* **Example:** `Router(config-mt-session)# profile snmp snmp-2`	SNMP プロファイルを Cisco Mediatrace セッションに関連付けます。
ステップ 10	**profile custom** *name* **Example:** `Router(config-mt-session)# profile custom cp-2`	SNMP プロファイルを Cisco Mediatrace セッションに関連付けます。
ステップ 11	**last-node** { **auto** | **address** *address* } **Example:** `Router(config-mt-session)# last-node address 10.1.1.1`	Cisco Mediatrace セッションの最後のノードを設定します。
ステップ 12	**end** **Example:** `Router(config-mt-session)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace session コマンドを使用して、特定のセッションまたはすべてのセッションのパラメータ設定を表示します。

show mediatrace responder app-health コマンドと **show mediatrace responder sessions** コマンドを使用して、モニター対象のノードのステータスを確認します。

必要なすべてのデータが Cisco Mediatrace で収集されない場合は、**debug mediatrace** コマンドを使用します。

これらのコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace セッションのスケジュール設定

Cisco Mediatrace セッションを設定したら、必要なときにデータの収集が開始されるようにスケジュールを設定することができます。Cisco Mediatrace セッションがパフォーマンス モニタ

リング メトリックの収集を目的とするものである場合は、セッションの開始時に Performance Monitor の有効化が試行されます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mediatrace schedule** *session ID* [*life* {**forever** | *secs*}] [**start-time** {*hh*:*mm*:[:*ss*][*month day*| *day month*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *secs*] [**recurring**]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**mediatrace schedule** *session ID* [*life* {**forever** | *secs*}] [**start-time** {*hh*:*mm*:[:*ss*][*month day*| *day month*]	**pending**	**now**	**after** *hh*:*mm*:*ss*}] [**ageout** *secs*] [**recurring**] **Example:** `Router(config)# mediatrace schedule 22 life 40 start-time 10:00:00 AUG 20 recurring`	セッションの実行日時を指定します。次の設定を使用します。 • *sessionID*：実行するセッション。 • **life** ：セッションの継続時間（秒数または永久のいずれか)。 • **start-time** ：セッションの開始時刻（指定の日時、イベントの保留、即時、指定の日時の後のいずれか）。 • **ageout** ：イニシエータでタイムアウトが発生してセッション設定が削除されるまでの時間。 • **recurring** ：セッションが指定時刻に繰り返し実行されます。
ステップ 4	**end** **Example:** `Router(config-mt-sched)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

show mediatrace session コマンドを使用して、特定のセッションまたはすべてのセッションのパラメータに意図したとおりの値が設定されていることを確認します。

show mediatrace responder app-health コマンドと **show mediatrace responder sessions** コマンドを使用して、モニター対象のノードのステータスを確認します。

必要なすべてのデータが Cisco Mediatrace で収集されない場合は、**debug mediatrace** コマンドを使用します。

これらのコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace セッションのクリア

以下の説明に従って **clear mediatrace incomplete-sessions** コマンドを使用して、イニシエータで完了していない Mediatrace セッションをクリアすることができます。また、このコマンドは、Cisco Mediatrace で設定したすべての Performance Monitor の設定をクリアします。config コマンドで作成したセッションについては、**no mediatrace schedule** コマンドを使用します。クリーンアップを実行すると、「セッションが切断されました（session teardown）」というメッセージが RSVP に対して出力され、続けて、ローカルの Mediatrace セッション データベースのクリーンアップが実行されます。

SUMMARY STEPS

1. **enable**
2. **clear mediatrace incomplete-sessions**
3. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**clear mediatrace incomplete-sessions** **Example:** `Router# clear mediatrace incomplete-sessions`	未完了の Mediatrace セッションをクリアします。
ステップ 3	**end** **Example:** `Router# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

Cisco Mediatrace セッションのステータスをチェックするには、**show mediatrace responder sessions** コマンドを使用します。

これらのコマンドの詳細については、Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法, on page 426を参照してください。

Cisco Mediatrace ポーリングの実行

Cisco Mediatrace ポーリングは、特定のパスのホップからオンデマンドでデータを取得するために使用します。いくつかの使用例を以下に示します。

- 事前設定済みのセッションを使用してデータを取得する場合。この場合、他のパラメータをインラインで指定する必要はありません。事前設定済みのセッションでは、オンデマンドを頻度のタイプとして設定する必要があります。
- 特定のパスのホップからシステム データ、ホップ、またはビデオ モニタリング情報を取得する場合。設定モード特権がない場合、パスを事前設定済みパス指定子またはインライン パス仕様として指定することができます。ただし、デフォルトでは、Cisco Mediatrace は、パスのノードからパッシブ モニタリング メトリックが報告されるように設定しようとし、次に、設定可能な時間待機した後に再びデータを収集しようとします。
- Performance Monitor コマンドを使用して既に Performance Monitor ポリシーが設定されているメディアパスのノードからデータを取得するため、**configless** キーワードを使用することができます。この方法を使用してデータを取得する場合に留意すべきいくつかの重要な点として、次のようなものがあります。
  - デフォルトのパフォーマンス モニタリング プロファイルまたは関連付けられているパフォーマンス モニタリング プロファイルに設定されているサンプリング間隔が適用されます。静的ポリシーのサンプリング間隔が関連付けられているパフォーマンス モニタリング プロファイルのサンプリング間隔と一致しない場合、データは返されません。
  - レスポンダ ノードで Performance Monitor ポリシーが設定されていない場合、Cisco Mediatrace レスポンダは Performance Monitor を設定しようとせず、単にエラーをイニシエータに報告します。

SUMMARY STEPS

1. **enable**
2. **mediatrace poll** {**no-traceroute** | **session** *number* | [**timeout** *value*] **path-specifier** {**name** *path-name* | **gsid** *gsid* | {[**disc-proto rsvp**] **destination ip** *ip-address* [**port** *nnnnn*] | **source ip** *ip-address* [**port** *nnnnn*] **destination ip** *ip-address* [**port** *nnnn*] [**ip-protocol** {**tcp** | **udp**}]} {**app-health** | **hops** | **l2-params gateway** *ip-address* | **system** [**profile** *system-profile-name*] | [**configless**] **perf-monitor** [**profile** *profile-name*]} {**flow-specifier** *name* | **source-ip** *ipaddress* [**source-port** *nnnnn*] **dest-ip** *ipaddress* [**dest-port** *nnnnn*] **ip-protocol** {**tcp** | **udp**}}}}
3. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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	**mediatrace poll** {**no-traceroute** | **session** *number* | [**timeout** *value*] **path-specifier**{**name** *path-name* | **gsid** *gsid* | {[**disc-proto rsvp**] **destination ip** *ip-address* [**port** *nnnnn*] | **source ip** *ip-address* [**port** *nnnnn*] **destination ip** *ip-address* [**port** *nnnn*] [**ip-protocol**{**tcp** | **udp**}]} {**app-health** | **hops** | **l2-params gateway** *ip-address* | **system** [**profile** *system-profile-name*] | [**configless**] **perf-monitor** [**profile** *profile-name*]} {**flow-specifier** *name* | **source-ip** *ipaddress* [**source-port** *nnnnn*] **dest-ip** *ipaddress* [**dest-port** *nnnnn*] **ip-protocol** {**tcp** | **udp**}}}} **Example:** **Example:** `Router# mediatrace poll session 22`	特定のパスのホップからオンデマンドでデータを取得します。次のいずれかのタイプの情報を使用して、ホップを指定できます。 • セッション定義またはその構成パラメータ • システム定義プロファイルまたはその構成パラメータ • パス指定子プロファイル定義とパフォーマンスモニタリングプロファイル定義の組み合わせ、またはそれらの構成パラメータの組み合わせ **Note** **l2-params gateway** キーワードは、Catalyst プラットフォームのみで使用できます。
ステップ 3	**end** **Example:** `Router# end`	現在のコンフィギュレーションモードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

必要なすべてのデータが Cisco Mediatrace で収集されない場合は、次のようにします。

- **show mediatrace session** コマンドを使用して、特定のセッションまたはすべてのセッションのパラメータに意図したとおりの値が設定されていることを確認します。
- **show mediatrace responder app-health** コマンドと **show mediatrace responder sessions** コマンドを使用して、モニター対象のノードのステータスを確認します。
- **debug mediatrace** コマンドを使用して、エラーメッセージを表示します。

例

Tip ポーリングの出力例については、Cisco Mediatrace の設定例, on page 434を参照してください。

例えば、送信元 IP アドレス、送信元ポート、および宛先ポートが不明な場合にデフォルトのシステム メトリックを取得するには、次のようにします。Cisco Mediatrace は、最適なローカル IP アドレスを送信元 IP アドレスとして使用して、RSVP を使用しているホップを検出します。

mediatrace poll path dest *ip-address* **system**

例えば、送信元ポート番号と宛先ポートが不明な場合にデフォルトのシステム メトリックを取得するには、次のようにします。指定した送信元と宛先の間のホップが RSVP によって検出されます。

mediatrace poll path source *ip-address* **dest** *ip-address* **system**

例えば、送信元ポート番号と宛先ポートがわかっている場合にデフォルトのシステム メトリックを取得するには、次のようにします。この情報が RSVP で使用されてホップが検出されます。

mediatrace poll path source-ip *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp system**

例えば、送信元ポート番号と宛先ポートが不明な場合にデフォルトの RTP メトリック セットを取得するには、次のようにします。Cisco Mediatrace は、パスの送信元 IP アドレスと宛先 IP アドレスを使用して、Performance Monitor のデータをフィルタリングすると共にホップを検出します。

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor**

例えば、デフォルトの RTP メトリック セットを取得するには、次のようにします。Cisco Mediatrace は、パス パラメータを使用してホップを検出し、インライン フロー指定子プロファイルを Performance Monitor データのフィルタとして使用します。

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp**

例えば、デフォルトの TCP メトリック セットを取得するには、次のようにします。Cisco Mediatrace は、パス パラメータを使用してホップを検出し、インライン フロー指定子プロファイルを Performance Monitor データのフィルタとして使用します。

mediatrace poll path source *ip-address* **dest** *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol tcp**

例えば、デフォルトの RTP メトリック セットを取得するには、次のようにします。Cisco Mediatrace は、最適なローカル IP アドレスを送信元 IP アドレスとして使用してパス上のホップを検出し、インライン フロー指定子プロファイルを Performance Monitor データのフィルタとして使用します。

mediatrace poll path dest *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol udp**

例えば、デフォルトの TCP メトリック セットを取得するには、次のようにします。Cisco Mediatrace は、最適なローカル IP アドレスを送信元 IP アドレスとして使用してパス上のホップを検出し、インライン フロー指定子プロファイルを Performance Monitor データのフィルタとして使用します。

mediatrace poll path dest *ip-address* **perf-monitor source-ip** *ip-address* **source - port** *nnnn* **dest-ip** *ip-address* **dest - port** *nnnn* **ip-protocol tcp**

例えば、ホップで既に設定されている静的ポリシーからデフォルトの RTP メトリックセットを取得するには、次のようにします。このコマンドを使用しても Performance Monitor が設定されることはありません。Cisco Mediatrace は、パス パラメータを使用してホップを検出し、インライン フロー指定子プロファイルを Performance Monitor データのフィルタとして使用します。

mediatrace poll path source *ip-address* **dest** *ip-address* **configless perf-monitor flow-specifier source** *ip-address* **port** *nnnn* **dest** *ip-address* **port** *nnnn* **ip-protocol udp**

ポーリングの出力例

この例は、次のホップ ポーリング コマンドの出力を示しています。

```
mediatrace poll path-specifier source 10.10.130.2 destination 10.10.132.2 hops
Started the data fetch operation.
Waiting for data from hops.
This may take several seconds to complete...
Data received for hop 1
Data received for hop 2
Data fetch complete.
Results:
Data Collection Summary:
  Request Timestamp: 22:47:56.788 PST Fri Oct 29 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 2
  Number of hops with valid data report: 2
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
    Number of Mediatrace hops in the path: 2
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Reachability Address: 10.10.12.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Reachability Address: 10.10.34.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
```

Cisco Mediatrace セッションのトラブルシューティングとモニタリングの方法

ここで説明する **show** コマンドを使用して、Cisco Mediatrace セッションのトラブルシューティングとモニタリングを実行します。

Tip 出力例については、この章の「例」セクションを参照してください。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **show mediatrace profile perf-monitor** [*name*]
4. **show mediatrace profile system** [*name*]
5. **show mediatrace flow-specifier** [*name*]
6. **show mediatrace path-specifier** [*name*]
7. **show mediatrace initiator**
8. **show mediatrace session-params** [*name*]
9. **show mediatrace session** [**config**| **data**| **stats**| **hops**] [**brief**| *ID*]
10. **show mediatrace responder app-health**
11. **show mediatrace responder sessions [** *global-session-id* **| brief | details]**
12. **debug mediatrace** {**event** | **trace** | **error**} [**initiator** | **responder**| *session-id*]
13. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	**show mediatrace profile perf-monitor** [*name*] **Example:** `Router(config)# show mediatrace profile perf-monitor vprofile-4`	事前にパッケージ化されているすべてのビデオ モニタリングプロファイルまたは指定されたプロファイルに設定されているパラメータを表示します。
ステップ 4	**show mediatrace profile system** [*name*] **Example:** `Router(config)# show mediatrace profile system system-8`	事前にパッケージ化されているすべてのシステム データ プロファイルまたは指定されたプロファイルに設定されているパラメータを表示します。
ステップ 5	**show mediatrace flow-specifier** [*name*] **Example:** `Router(config)# show mediatrace flow-specifier flow-2`	すべてのフロー指定子プロファイルまたは指定されたフロー指定子プロファイルに設定されているパラメータを表示します。

	Command or Action	Purpose
ステップ 6	**show mediatrace path-specifier** [*name*] **Example:** `Router(config)# show mediatrace path-specifier path-6`	すべてのパス指定子プロファイルまたは指定されたパス指定子プロファイルに設定されているパラメータを表示します。
ステップ 7	**show mediatrace initiator** **Example:** `Router(config)# show mediatrace initiator`	イニシエータ プロファイルに設定されているパラメータを表示します。
ステップ 8	**show mediatrace session-params** [*name*] **Example:** `Router(config)# show mediatrace session-params sysparams-2`	頻度や応答タイムアウトなど、セッションのモニタリング パラメータを表示します。 事前にパッケージ化されているすべてのシステム データ プロファイルまたは指定されたプロファイルに設定されているパラメータ。
ステップ 9	**show mediatrace session** [**config**| **data**| **stats**| **hops**] [**brief**| *ID*] **Example:** `Router(config)# show mediatrace session data 1002`	すべてのセッション プロファイルまたは指定されたセッション プロファイルに設定されているパラメータを表示します。次のキーワードを使用して、該当の情報を表示します。 • **config** ：セッションのコンフィギュレーション。 • **data**：イニシエータで収集されたすべてのデータレコードと、まだキャッシュに残っているすべてのデータレコード。 • **stats** ：このサービスパスまたはセッションの統計情報。 • **hops** ：以前のサービスパス（可能な場合）および検出された現在のサービスパス。また、前回のルート変更の場所と日時も表示されます。 • **brief** ：ID、送信元および宛先のアドレスとポート、ならびにそれらに関連付けられている役割（イニシエータまたはレスポンダ）のみを含むセッションのリスト。 • *ID* ：セッション ID および何らかの状態情報。
ステップ 10	**show mediatrace responder app-health** **Example:**	レスポンダの現在のステータスを表示します。

	Command or Action	Purpose
	`Router(config)# show mediatrace responder app-health`	
ステップ 11	**show mediatrace responder sessions [** *global-session-id* **| brief | details]** **Example:** `Router(config)# show mediatrace responder sessions`	ローカル レスポンダのすべてのアクティブなセッションまたは特定のアクティブなセッションに関する情報を表示します。次のキーワードを使用して、対応する情報を表示します。 • *global-session-id* ：情報を表示するセッションの ID。 • **brief** ：パスの宛先および送信元のアドレスとポート、それらの役割（イニシエータまたはレスポンダ）、ならびに何らかの状態情報のみを表示します。 • **details** ：すべての情報を表示します。
ステップ 12	**debug mediatrace** {**event** | **trace** | **error**} [**initiator** | **responder**| *session-id*] **Example:** `Router(config)# debug mediatrace event 24`	特定のパス、特定のセッション、またはすべてのイニシエータ機能とレスポンダ機能について、デバッグを有効にします。次のオプションを使用できます。 • **event** ：イベント情報のみを表示します。 • **trace** ：トレース情報のみを表示します。 • **error** ：エラーのみを表示します。 • **initiator** ：イニシエータのみの情報を表示します。 • **responder** ：レスポンダのみの情報を表示します。 • *session-id* ：セッションのみの情報を表示します。
ステップ 13	**end** **Example:** `Router(config)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

例

Note 以下の show コマンドの完全な説明については、『*Cisco Media Monitoring Command Reference*』を参照してください。

例えば、ビデオ モニタリング プロファイルを表示するには、次のようにします。

```
Router# show mediatrace profile perf-monitor
Perf-monitor Profile: vprof-4
Metric List: rtp
RTP Admin Parameter:
  Max Dropout: 5
  Max Reorder: 5
  Min Sequential: 5
Admin Parameter:
  Sampling Interval (sec): 30
```

例えば、システム データ プロファイルを表示するには、次のようにします。

```
Router# show mediatrace profile
system

System Profile: sys-1
Metric List: intf
```

例えば、フロー指定子プロファイルを表示するには、次のようにします。

```
Router# show mediatrace
 flow-specifier flow-1
Flow Specifier: flow-1
    Source address/port:
    Destination address/port:
    Protocol: udp
```

例えば、パス指定子プロファイルを表示するには、次のようにします。

```
Router# show mediatrace
 path-specifier flow-1
Path Configuration: ps1
    Destination address/port: 10.10.10.1
    Source address/port: 10.10.10.4
    Gateway address/vlan:
    Discovery protocol: rsvp
```

例えば、イニシエータ プロファイルを表示するには、次のようにします。

```
Router# show mediatrace
 initiator
Version: Mediatrace 1.0
Mediatrace Initiator status: enabled
Source IP: 1.1.1.1
Number of Maximum Allowed Active Session: 127
Number of Configured Session: 1
Number of Active Session    : 0
```

```
Number of Pending Session   : 0
Number of Inactive Session  : 1
Note: the number of active session may be higher than max active session
      because the max active session count was changed recently.
```

例えば、セッション プロファイルを表示するには、次のようにします。

```
Router# show mediatrace session-params
Session Parameters: s-1
    Response timeout (sec): 60
    Frequency: On Demand
    Inactivity timeout (sec): 300
    History statistics:
       Number of history buckets kept: 3
    Route change:
       Reaction time (sec): 5
```

例えば、Mediatrace セッションの統計情報を表示するには、次のようにします。

```
Router# show mediatrace session stats 2
Session Index: 2
Global Session Id: 86197709
Session Operation State: Active
Operation time to live: Forever
Data Collection Summary:
  Request Timestamp: 23:55:04.228 PST Fri Oct 29 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 2
  Number of Non Mediatrace hops responded: 0
  Number of hops with valid data report: 2
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
  Route Index: 0
    Number of Mediatrace hops in the path: 2
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Metrics Collection Status: Success
      Reachability Address: 10.10.12.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
      Traceroute data:
        Address List: 1.2.2.3
        Round Trip Time List (msec): 12 msec
```

Note ホップ 1 の残りのデータは、次に示すホップ 2 のデータと同様です。

```
Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Metrics Collection Status: Success
      Reachability Address: 10.10.34.3
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
      Metrics Collected:
        Collection timestamp: 23:55:04.237 PST Fri Oct 29 2010
        Octet input at Ingress (KB): 929381.572
        Octet output at Egress (MB): 1541.008502
        Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
```

```
        Pkts discarded at Egress (pkts): 0
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
```

例えば、Mediatrace セッションの設定情報を表示するには、次のようにします。

```
Router# show mediatrace session config 2
Global Session Id: 93642270
---------------------------
Session Details:
    Path-Specifier: ps1
    Session Params: sp1
    Collectable Metrics Profile: intf1
    Flow Specifier:
Schedule:
   Operation frequency (seconds): 30  (not considered if randomly scheduled)
   Next Scheduled Start Time: Start Time already passed
   Group Scheduled : FALSE
   Randomly Scheduled : FALSE
   Life (seconds): Forever
   Entry Ageout (seconds): never
   Recurring (Starting Everyday): FALSE
   Status of entry (SNMP RowStatus): Active
History Statistics:
   Number of history Buckets kept: 10
```

例えば、Mediatrace セッションのホップを表示するには、次のようにします。

```
show mediatrace session hops 2
Session Index: 2
Global Session Id: 93642270
Session Operation State: Active
Data Collection Summary:
  Request Timestamp: 13:40:32.515 PST Fri Jun 18 2010
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 3
  Number of hops with valid data report: 3
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
  Route Index: 0
    Number of Mediatrace hops in the path: 3
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
    Mediatrace Hop Number: 3 (host=responder3, ttl=252)
      Ingress Interface: Gi0/1
      Egress Interface: Gi0/2
```

例えば、Mediatrace セッションのデータを表示するには、次のようにします。

```
Router# show mediatrace session data 2
Session Index: 2
Global Session Id: 35325453
Session Operation State: Active
Bucket index: 1
Data Collection Summary:
  Request Timestamp: 13:02:47.969 PST Fri Jun 18 2010
```

```
  Request Status: Completed
  Number of hops responded (includes success/error/no-record): 3
  Number of hops with valid data report: 3
  Number of hops with error report: 0
  Number of hops with no data record: 0
Detailed Report of collected data:
  Last Route Change Timestamp:
  Route Index: 0
    Number of Mediatrace hops in the path: 3
    Mediatrace Hop Number: 1 (host=responder1, ttl=254)
      Metrics Collection Status: Success
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
      Metrics Collected:
        Collection timestamp: 13:04:57.781 PST Fri Jun 18 2010
        Octet input at Ingress (KB): 10982.720
        Octet output at Egress (KB): 11189.176
    Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
        Pkts discarded at Egress (pkts): 0
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
    Mediatrace Hop Number: 2 (host=responder2, ttl=253)
      Metrics Collection Status: Success
      Ingress Interface: Gi0/1
      Egress Interface: Gi1/0
      Metrics Collected:
        Collection timestamp: 13:04:57.792 PST Fri Jun 18 2010
        Octet input at Ingress (MB): 1805.552836
        Octet output at Egress (MB): 1788.468650
        Pkts rcvd with err at Ingress (pkts): 0
        Pkts errored at Egress (pkts): 0
        Pkts discarded at Ingress (pkts): 0
        Pkts discarded at Egress (pkts): 0
        Ingress i/f speed (mbps): 1000.000000
        Egress i/f speed (mbps): 1000.000000
```

例えば、Mediatrace レスポンダのアプリケーション ヘルス情報を表示するには、次のようにします。

```
Router# show mediatrace responder app-health
Mediatrace App-Health Stats:
   Number of all requests received: 0
   Time of the last request received:
   Initiator ID of the last request received:  0
   Requests dropped due to queue full: 0
   Responder current max sessions: 45
   Responder current active sessions: 0
   Session down or tear down requests received: 0
   Session timed out and removed: 0
   HOPS requests received: 0
   VM dynamic polling requests received: 0
   VM dynamic polling failed: 0
   VM configless polling requests received: 0
   VM configless polling failed: 0
   SYSTEM data polling requests received: 0
   SYSTEM data polling requests failed: 0
   APP-HEALTH polling requests received: 0
   Route Change or Interface Change notices received: 0
   Last time Route Change or Interface Change:
   Unknown requests received: 0
```

例えば、Mediatrace レスポンダの簡潔なセッション情報を表示するには、次のようにします。

```
Router# show mediatrace responder sessions brief
Local Responder configured session list:
Current configured max sessions: 45
Current number of active sessions: 0
session-id initiator-name       src-ip          src-port   dst-ip          dst-port det-l

  2   host-18     10.10.10.2     200   10.10.10.8     200
```

Cisco Mediatrace の設定例

例：Mediatrace の基本設定

この例のトポロジには、次のものが含まれています。

- Mediatrace イニシエータ（10.10.12.2）1 つ
- 以下の間のレスポンダ 2 つ
  - メディア送信元（10.10.130.2）
  - 宛先（10.10.132.2）

この例では、送信元（アドレス 10.10.130.2、ポート 1000）から宛先（アドレス 10.10.132.2、ポート 2000）への RTP トラフィック ストリームがあります。

Mediatrace レスポンダの基本設定は、次のとおりです。

```
mediatrace responder
snmp-server community public RO
```

Mediatrace イニシエータの基本設定は、次のとおりです。

```
mediatrace initiator source-ip 10.10.12.2
mediatrace profile system intf1
mediatrace profile perf-monitor rtp1
mediatrace path-specifier path1 destination ip 10.10.132.2 port 2000
 source ip 10.10.130.2 port 1000
mediatrace flow-specifier flow1
 source-ip 10.10.130.2 source-port 1000
 dest-ip 10.10.132.2 dest-port 2000
mediatrace session-params sp1
 response-timeout 10
 frequency 60 inactivity-timeout 180
mediatrace 1
 path-specifier path1
 session-params sp1
 profile perf-monitor rtp1 flow-specifier flow1
mediatrace schedule 1 life forever start-time now
mediatrace 2
 path-specifier path1
 session-params sp1
```

```
 profile system intf1
mediatrace schedule 2 life forever start-time now
```

サンプルのリバース Mediatrace 設定は、次のとおりです。

```
Device# show mediatrace initiator
Mediatrace Initiator Software Version: 3.0
Mediatrace Protocol Version: 1
Mediatrace Initiator status: enabled

Source IP: 10.10.1.1
Source IPv6:

Number of Maximum Allowed Active Session: 8
Number of Configured Session: 3
Number of Active Session    : 2
Number of Pending Session   : 0
Number of Inactive Session  : 1
Number of Total Proxy Session     : 1
Number of Active Proxy Session    : 1
Number of Pending Proxy Session   : 0
Number of Inactive Proxy Session  : 0

Note: the number of active session may be higher than max active session
      because the max active session count was changed recently.

Device# show run
Device# show running-config | show mediatrace
mediatrace responder
mediatrace initiator source-ip 10.10.1.1
mediatrace profile perf-monitor MT_PERF_RTP
mediatrace path-specifier MT_PATH destination ip 10.11.1.10 port 21064
 source ip 10.10.1.11 port 28938
mediatrace path-specifier MT_PATH2 destination ip 10.10.10.10 port 16514
 source ip 10.10.1.10 port 16558
mediatrace flow-specifier MT_FLOW
 source-ip 10.10.1.11 source-port 28938
 dest-ip 10.10.1.50 dest-port 21064
mediatrace flow-specifier MT_FLOW2
 source-ip 10.1.1.50 source-port 21064
 dest-ip 10.1.1.11 dest-port 28938
mediatrace session-params MT_PARAMS
 response-timeout 50
 frequency 60 inactivity-timeout 180
 history data-sets-kept 10
mediatrace reverse 155
 path-specifier forward/reverse MT_PATH/MT_PATH2
 session-params MT_PARAMS
 profile perf-monitor MT_PERF_RTP flow-specifier MT_FLOW2
mediatrace schedule 155 life forever start-time now
mediatrace 157
 path-specifier MT_PATH
 session-params MT_PARAMS
 profile perf-monitor MT_PERF_RTP flow-specifier MT_FLOW
mediatrace schedule 157 life forever start-time now
```

次の作業

Medianet 製品ファミリの製品設定の詳細については、このガイドの他の章または『*Cisco Media Monitoring Configuration Guide*』を参照してください。

その他の参考資料

関連資料

関連項目	マニュアル タイトル
Cisco Mediatrace およびその他の Cisco Medianet 製品の設計、設定、ならびにトラブルシューティングに関する資料（クイック スタート ガイドや導入ガイドなど）。	Cisco Medianet ナレッジ ベース ポータル サイト（http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html）を参照してください。
IP アドレッシング コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト、使用上の注意事項、および例	『*Cisco Media Monitoring Command Reference*』

標準

標準	タイトル
この機能がサポートする新しい規格または変更された規格はありません。また、この機能による既存規格のサポートに変更はありません。	--

MIB

MIB	MIB のリンク
この機能がサポートする新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	--

RFC

RFC[4]	タイトル
RFC 2205	*Resource Reservation Protocol*（*RSVP*） http://www.ietf.org/rfc/rfc2205.txt

[4] これらの参考資料は、IP アドレッシングおよび IP ルーティングに関連する項目で使用できる多くの RFC の例です。RFC の完全なリストについては、IETF RFC のサイト（http://www.ietf.org/rfc.html）を参照してください。

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

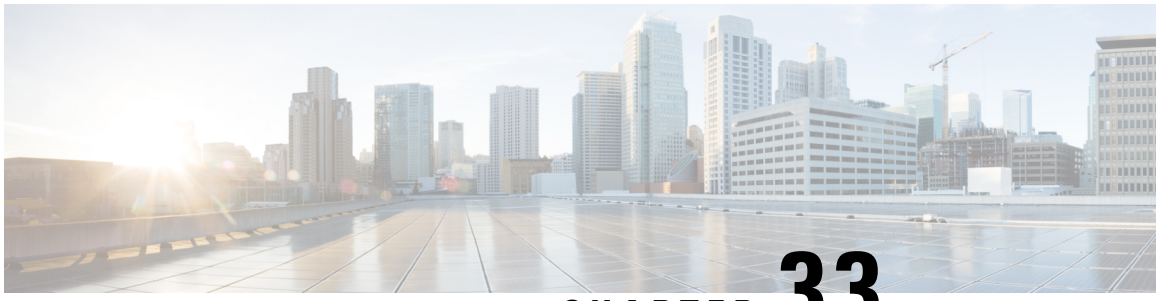
Cisco Mediatrace の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 40: Cisco Mediatrace の機能情報

機能名	リリース	機能情報
Cisco Mediatrace 1.0	15.1(3)T 12.2(58)SE 15.1(4)M1 15.0(1)SY 15.1(1)SY 15.1(1)SY1 15.2(1)S Cisco IOS XE Release 3.5S 15.1(2)SY	この機能を使用すると、データ ストリームに関するネットワーク パフォーマンス低下の問題の切り分けを行って、トラブルシューティングを実行できます。 この機能により、次のコマンドが導入または変更されました。**admin-params**、**clear mediatrace**、**incomplete-sessions**、**clock-rate** (RTP parameters)、**dest-ip** (flow)、**frequency** (session parameters)、**history** (session parameters)、**ip-protocol** (flow)、**max-dropout**、**max-reorder**、**mediatrace**、**mediatrace initiator**, **mediatrace responder**、**mediatrace path-specifier**、**mediatrace poll**、**mediatrace profile perf-monitor**、**mediatrace profile system**、**mediatrace schedule**、**mediatrace session-params**、**metric-list** (monitoring profile)、**metric-list** (system profile)、**min-sequential**、**path-specifier**, **profile perf-monitor**、**profile system**、**response-timeout** (session parameters)、**route-change reaction-time**、**sampling-interval**、**session-params**、**show mediatrace flow-specifier**、**show mediatrace initiator**、**show mediatrace path-specifier**、**show mediatrace profile system**、**show mediatrace profile perf-monitor**、**show mediatrace responder app-health**、**show mediatrace responder sessions**、**show mediatrace session**、**show mediatrace session-params**、**source-ip** (flow)、および **source ip** (path)。



CHAPTER 33

Cisco Performance Monitor の設定

このドキュメントには、Cisco Performance Monitor の設定に関する情報と説明が記載されています。

Cisco Performance Monitor に関する情報

Cisco Performance Monitor の概要

Cisco Performance Monitor では、ネットワーク内のパケット フローをモニタすることで、対象のアプリケーションのパフォーマンスに重大な影響が現れる前に、そのフローに影響をおよぼす可能性がある問題点を認識できます。高品質で対話型のビデオ トラフィックはネットワークの問題点の影響を非常に受けやすいため、ビデオ トラフィックに対しては特にパフォーマンス モニタリングの重要性は高くなります。他のアプリケーションに影響を与えることがほとんどない軽度の問題であっても、ビデオの品質には大きな影響をおよぼす可能性があります。

Cisco Performance Monitor は Cisco NetFlow や Cisco Flexible NetFlow と同様のソフトウェア コンポーネントとコマンドを使用するので、それらの製品について熟知していると、Cisco Performance Monitor の設定方法について理解するのに役立ちます。これらの製品は、ルータを通過するパケットの統計情報を提供し、IP ネットワークから IP 運用データを取得するための定番製品です。これらは、ネットワークとセキュリティのモニタリング、ネットワーク計画、トラフィック分析、および IP アカウンティングをサポートするためのデータを提供します。Cisco NetFlow および Cisco Flexible NetFlow の詳細については、「その他の参考資料」に記載されているドキュメントを参照してください。

Performance Monitor およびその他の Cisco Medianet 製品の設計、設定、ならびにトラブルシューティングの詳細については、クイック スタート ガイドや導入ガイドも含めて、Cisco Medianet ナレッジ ベース ポータル サイト（http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html）を参照してください。

Cisco Performance Monitor の設定の前提条件

Cisco Performance Monitor を設定する前に、次の前提条件を満たしておく必要があります。

IPv4 トラフィック

- ネットワーキング デバイスが IPv4 ルーティング用に設定されていること。
- 使用しているルータおよび Flexible NetFlow を有効にするすべてのインターフェイスで Cisco Express Forwarding または分散型 Cisco Express Forwarding が有効になっていること。

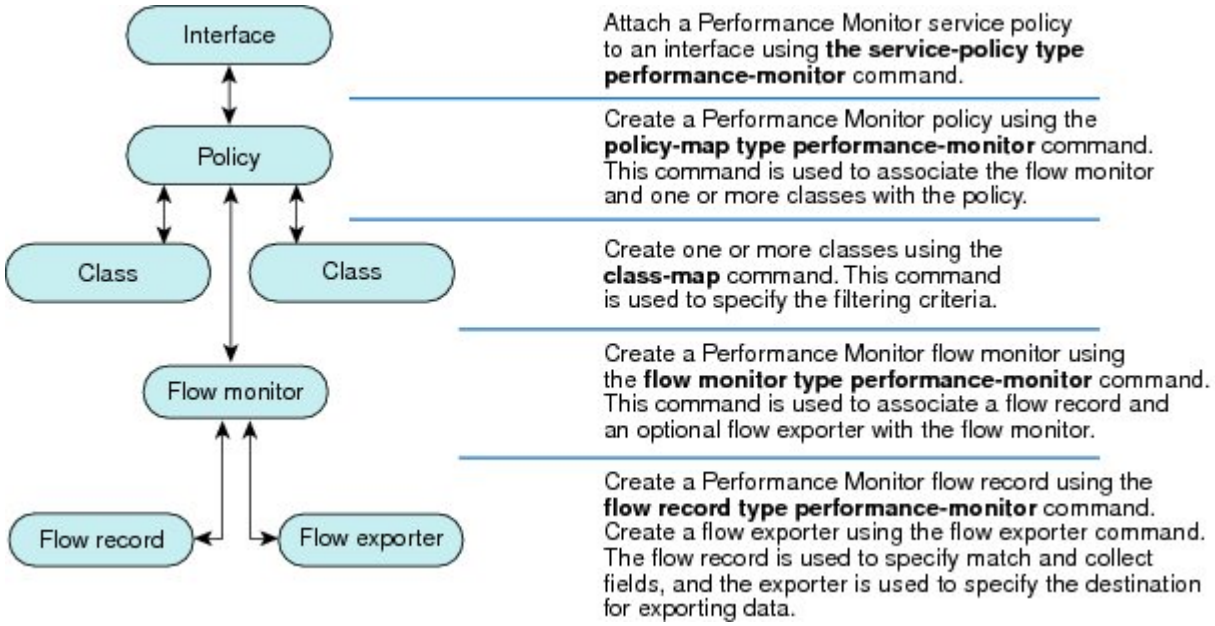
Cisco Performance Monitor の構成コンポーネント

Cisco Performance Monitor を設定するには、Flexible NetFlow で通常設定するのと同じ基本要素の多くを設定します。

- インターフェイス
- ポリシー
- クラス
- フロー モニタ
- フロー レコード
- フロー エクスポータ

次の図に、それらの要素がどのように相関しているかを示します。図の最下部にある要素を最初に設定します。

Figure 11: Cisco Performance Monitor のコンポーネント



上記のように、ポリシーには1つ以上のクラスが含まれます。各クラスにはそのクラスに関連付けられているフロー モニタがあり、各フロー モニタにはフロー レコードとフロー モニタに関連付けられているオプションのフローエクスポータがあります。これらの要素は、次の順序で設定します。

1. フロー レコードを設定して、モニタする非キー フィールドとキー フィールドを指定します。これは、**match** および **collect** コマンドを使用して設定します。また、オプションで、フローエクスポータを設定してエクスポート先を指定することもできます。Cisco Performance Monitor では、**performance-monitor** タイプのフローレコードを設定する必要があります。
2. フロー レコードおよびフロー エクスポータを含むフロー モニタを設定します。Cisco Performance Monitor では、**performance-monitor** タイプのフローモニターを設定する必要があります。
3. **class-map** コマンドを使用して、クラスを設定してフィルタリング基準を指定します。
4. **policy-map** コマンドを使用して、ポリシーを設定して 1 つ以上のクラスと 1 つ以上の **performance-monitor**タイプのフローモニターを含めます。Cisco Performance Monitor では、**performance-monitor** タイプのポリシーを設定する必要があります。
5. **service-policy type performance-monitor** コマンドを使用して、**performance-monitor** タイプポリシーを適切なインターフェイスに関連付けます。

Cisco Performance Monitor を使用してモニタできるデータ

collect コマンドまたは **match** コマンドを使用して、対応する非キーフィールドについてフローレコードを設定することにより、以下の情報をモニターできます。

Tip これらの統計の詳細については、『*Cisco Media Monitoring Command Reference*』の **show performance monitor status** コマンドを参照してください。

- IP パケット数（IP Packet Count）
- IP TTL
- IP TTL（最小）（IP TTL minimum）
- IP TTL（最大）（IP TTL maximum）
- インターフェイス マッピングへのフロー（Flow to Interface Mapping）
- IP フローの宛先アドレスとポート、送信元アドレスとポート、およびプロトコル
- RTP 同期ソース（SSRC）（RTP Synchronization Source (SSRC)）
- IP オクテット数（IP Octets Count）
- メディア ストリーム パケット数（Media Stream Packet Count）
- メディア ストリーム オクテット数（Media Stream Octect Count）
- メディア バイト レート（Media Byte Rate）
- メディア バイト数（Media Byte Count）
- メディア パケット レート（Media Packet Rate）
- メディア パケット損失数（Media Packet Loss Count）
- メディア パケット損失レート（Media Packet Loss Rate）
- 予想パケット数（Packets Expected Count）
- 測定レート（Measured Rate）
- メディア損失イベント数（Media Loss Event Count）
- ラウンドトリップ時間（RTT）
- 到着間ジッター（RFC3550）（最大）（Interarrival Jitter (RFC3550) max）
- 到着間ジッター（RFC3550）（最小 2）（Interarrival Jitter (RFC3550) min 2）
- 到着間ジッター（RFC3550）（平均）（Interarrival Jitter (RFC3550) mean）
- メディア レート変動（Media Rate Variation）
- モニタ イベント（Monitor Event）
- メディア エラー（Media Error）
- メディア停止（Media Stop）

- IP バイト数（IP Byte Count）
- IP バイト レート（IP Byte Rate）
- IP Source Mask
- IP Destination Mask
- モニタリング インターバルのエポック（Epoch of A Monitoring Interval）
- パケット転送ステータス（Packet Forwarding Status）
- Packet Drops
- DSCP および IPv6 トラフィック クラス（DSCP and IPv6 Traffic Class）
- TCP 最大セグメントサイズ（TCP Maximum Segment Size）
- TCP：最大ウィンドウサイズ
- TCP：最大ウィンドウサイズ
- TCP：平均ウィンドウサイズ
- 不正なバイト数
- 不正なパケット数

Cisco Performance Monitor の SNMP MIB サポート

Cisco Performance Monitor は、メディア ストリームをモニタするため、業界標準の Simple Network Management Protocol（SNMP）の使用をサポートします。このサポート機能は、次に示すシスコ独自の SNMP Management Information Base（MIB）モジュールの追加と共に実装されます。

- CISCO-FLOW-MONITOR-TC-MIB：以下の MIB モジュールに共通するテキスト規則を定義します。
- CISCO-FLOW-MONITOR-MIB：システムでサポートされているフローモニタを表すフレームワーク、システムで学習されたフロー、それらのフローに関して収集されるフローメトリックを定義します。
- CISCO-RTP-METRICS-MIB：RTCP Receiver Report パケット（RFC 3550）によって表されるメトリックと同様の、RTP ストリームに関して収集される品質メトリックを表すオブジェクトを定義します。
- CISCO-IP-CBR-METRICS-MIB：固定ビット レート（CBR）をもつ IP ストリームに関して収集される品質メトリックを表すオブジェクトを定義します。

これらの MIB の詳細について、また、特定のプラットフォーム、Cisco IOS リリース、およびフィーチャ セットの MIB を検索してダウンロードするには、Cisco MIB Locator（http://www.cisco.com/go/mibs）を使用してください。

また、この機能には、新しいコマンドライン インターフェイス（CLI）コマンド 2 つと、変更された CLI コマンド 1 つも含まれています。これらのコマンドは、次のとおりです。

- **snmp-server host**：受信者へのフローモニタリング SNMP 通知の配信を有効にします。
- **snmp-server enable traps flowmon**：フロー監視の SNMP 通知を有効にします。デフォルトでは、フロー モニタリング SNMP 通知は無効になっています。
- **snmp mib flowmon alarm history**：フローモニターアラーム履歴ログによって維持されるエントリの最大数を設定します。

Catalyst 6500 プラットフォームに関する制限事項

Cisco Performance Monitor には Catalyst 6000 プラットフォームに関する次の制限事項があります。

- モニタできるインターフェイスのタイプについて、いくつかの制限事項があります。以下の 2 つの表に、Catalyst 6500 プラットフォームにおける入力モニタリングと出力モニタリングでサポートされているインターフェイスのタイプを示します。

Table 41: 入力インターフェイスのサポート

インターフェイス タイプ	サポート
レイヤ 3 ルーテッド ポート	あり
レイヤ 3 サブインターフェイス (a)	なし
レイヤ 3 ポート チャネル	あり
レイヤ 3 ポートチャネル サブインターフェイス (a)	なし
レイヤ 3 SVI (b)	一部（以下の箇条書きの 3 番目の項目を参照）
L3 トンネル	なし
レイヤ 2 物理（スイッチド）ポート	あり
レイヤ 2 ポート チャネル	あり
レイヤ 2 VLAN	あり

Table 42: 出力インターフェイスのサポート

インターフェイス タイプ	サポート
レイヤ 3 ルーテッド ポート	あり
レイヤ 3 サブインターフェイス (a)	あり

インターフェイス タイプ	サポート
レイヤ 3 ポート チャネル	あり
レイヤ 3 ポートチャネル サブインターフェイス (a)	あり
レイヤ 3 SVI (b)	あり
L3 トンネル	なし
レイヤ 2 物理（スイッチド）ポート	なし
レイヤ 2 ポート チャネル	なし
レイヤ 2 VLAN	あり

- VRF でのパフォーマンス モニタリングはサポートされていません。
- マルチキャストフローのパフォーマンス監視はサポートされていません。
- VLANインターフェイスのトランク ポートからのルーテッド トラフィックは、トラフィックの送信元 VLAN インターフェイスを特定できないため、モニタできません。「Routed traffic from trunk ports will not be monitored by ingress policy on VLAN interface（トランク ポートからのルーテッド トラフィックは、VLAN インターフェイスの入力ポリシーにより、モニタされません）」という syslog メッセージが表示されます。

  回避策として、トランク インターフェイスでパフォーマンス モニタリング ポリシーを設定できます。このモニタリングを利用すると、CPU 使用率が増加することになります。
- match all タイプのクラスマップを使用することはできません。サポートされているのは、match any タイプの検索のみです。match all タイプのクラス マップを使用するようにパフォーマンス モニタリングを設定した場合、パケットの複製が CPU に送られます。その結果、match-all クラスが正常に適用されると、再びパケットが CPU で分類されて、必要に応じてドロップされます。そのため、CPU 使用率が予想よりも高くなります。
- VLAN インターフェイスの出力のパフォーマンス モニタリング ポリシーでは、VLAN 内でブリッジされるトラフィックはモニタされません。これは、ハードウェアの制限によるものです。回避策は、VLANインターフェイスの出力だけでなく入力でもポリシーを適用することです。VLANインターフェイスの入力のポリシーでは、ブリッジされたパケットがモニタされます。
- 出力ポリシーによって複製されるパケットについては、ソフトウェアによるレート制限のみが可能です。それらのパケットについてハードウェアベースの保護を使用することはできません。したがって、多くのフローをモニタする場合のシナリオでは、CPU割り込み使用率が高くなる可能性があります。
- 出力パフォーマンス モニタリングでは、Catalyst 6500 プラットフォームの再循環メカニズムを利用します。その結果、フレーム スイッチングの遅延が数マイクロ秒増加します。
- 高速（CEF）パスを使用してスイッチングされるパケットについては、パフォーマンス モニタリングはサポートされていません。

- 合法的傍受およびパフォーマンス モニタリングでは、パケットの複製と同じメカニズムを利用します。合法的傍受機能は、パフォーマンス モニタリングよりも優先されます。したがって、パフォーマンス モニタリングは、合法的傍受機能が有効になっている場合には機能しません。そのようなことが発生すると、syslog メッセージが作成されます。
- パフォーマンス モニタリングでは、最適化 ACL ロギング、VACL キャプチャ、IPv6 コピーなどの他の機能と同じメカニズムを利用します。最初に有効にされた機能が優先されます。その他の機能はブロックされて設定できなくなり、syslog メッセージが作成されます。

IPv6 サポートの制限事項

パフォーマンスモニターでの IPv6 のサポートには、次の制限があります。

- IPv6 でサポートされるトポロジは、非MPLS、DMVPN（ほとんどのプラットフォーム）、およびデュアルスタックです。
- 次のトポロジは、IPv6 ではサポートされていません。MPLS/VRF（6PE および 6VPE）、GETVPN、および IPV4 トンネルを介した IPV6。
- Mediatrace は IPv6 をサポートしていません。
- IPv6 アドレスへのデータのエクスポートは、ASR1K プラットフォームではサポートされていません。
- Flexible NetFlow は IPv6 マルチキャストをサポートしていません。
- DMVPN は、ASR1K プラットフォームの IPv6 ではサポートされていません。

Cisco Performance Monitor の設定、トラブルシューティング、およびメンテナンスの方法

Note これらの作業で使用する Flexible NetFlow のコマンド、キーワード、および引数の多くは、以前のリリースでも利用できます。これらの既存の Flexible NetFlow コマンド、キーワード、および引数の詳細については、『*Cisco IOS Flexible NetFlow Command Reference*』を参照してください。

Cisco Performance Monitor のフロー エクスポータの設定

フロー エクスポータは、Cisco Performance Monitor で収集されるデータを NetFlow Collection Engine などのリモート システムへ送信するために使用されます。エクスポータでは、転送プ

ロトコルとして User Datagram Protocol（UDP）が使用され、バージョン 9 エクスポート フォーマットが使用されます。

詳細な分析や保管を目的として、Cisco Performance Monitor によって収集されるデータをリモートシステムにエクスポートするためにフローモニタ用のフローエクスポータを設定するには、次のオプション作業を実行します。Cisco Performance Monitor では、フロー エクスポータは Cisco IOS Flexible NetFlow の場合と同様の方法で設定します。詳細については、『*Configuring Data Export for Cisco IOS Flexible NetFlow with Flow Exporters*』を参照してください。

Note IPv4 アドレスと IPv6 アドレスのいずれを使用しても宛先にエクスポートできます。

Note フローエクスポータごとに、1 つの宛先のみがサポートされます。複数の宛先にデータをエクスポートする場合は、複数のフロー エクスポータを設定してフロー モニターに割り当てる必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow exporter** *exporter-name*
4. **description** *description*
5. **destination** {*ip-address* | *hostname*} [**vrf** *vrf-name*]
6. **export-protocol** {**netflow-v5** | **netflow-v9** | **ipfix** }
7. **dscp** *dscp*
8. **source** *interface-type interface-number*
9. **option** {**application-attributes** | **application table** | **exporter-stats** | **interface-table** | **metadata-table** | **sampler-table** | **vrf-table**} [**timeout** *seconds*]
10. **output-features**
11. **template data timeout** *seconds*
12. **transport udp** *udp-port*
13. **ttl** *seconds*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
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	**flow exporter** *exporter-name* **Example:** `Device(config)# flow exporter EXPORTER-1`	フロー エクスポータを作成し、Flexible NetFlow フロー エクスポータ コンフィギュレーション モードを開始します。 • このコマンドでは、既存のフロー エクスポータを変更することもできます。
ステップ 4	**description** *description* **Example:** `Device(config-flow-exporter)# description Exports to the datacenter`	（任意）設定および **show flow exporter** コマンドの出力に表示されるエクスポータの説明を設定します。
ステップ 5	**destination** {*ip-address* | *hostname*} [**vrf** *vrf-name*] **Example:** `Device(config-flow-exporter)# destination 172.16.10.2`	エクスポータでデータを送信する宛先システムの IP アドレスまたはホスト名を指定します。 **Note** IPv4 アドレスと IPv6 アドレスのいずれを使用しても宛先にエクスポートできます。
ステップ 6	**export-protocol** {**netflow-v5** | **netflow-v9** | **ipfix** } **Example:** `Device(config-flow-exporter)# export-protocol netflow-v9`	エクスポータで使用されるプロトコルを指定します。 **Note** NBAR から抽出されたフィールドのエクスポートは、IPFIX経由でのみサポートされます。
ステップ 7	**dscp** *dscp* **Example:** `Device(config-flow-exporter)# dscp 63`	（任意）エクスポータによって送信されるデータグラムの Diffserv コード ポイント（DSCP）パラメータを設定します。 • *dscp* 引数の範囲は 0 ～ 63 です。デフォルト：0。
ステップ 8	**source** *interface-type interface-number* **Example:** `Device(config-flow-exporter)# source ethernet 0/0`	（任意）エクスポータで、エクスポートされたデータグラムの送信元 IP アドレスとして IP アドレスを使用するローカルインターフェイスを指定します。

	Command or Action	Purpose
ステップ 9	**option** {**application-attributes** | **application table** | **exporter-stats** | **interface-table** | **metadata-table** | **sampler-table** | **vrf-table**} [**timeout** *seconds*] **Example:** `Device(config-flow-exporter)# option exporter-stats timeout 120`	（任意）エクスポートされるデータの量を減らすためのオプションテーブルの使用を有効にします。これらのテーブルにより、エクスポータは、メタデータの完全な値を表し、オプションテーブルによって値にマッピングされる ID をエクスポートできます。たとえば、インターフェイステーブルはSNMP インデックスをインターフェイス名にマッピングし、VRF テーブルはVRF ID を名前にマッピングします。 • オプションテーブルの任意の組み合わせを同時に使用できるようにすることができます。 • *seconds* 引数の範囲は、1 ～ 86,400 です。デフォルト値：600。
ステップ 10	**output-features** **Example:** `Device(config-flow-exporter)# output-features`	（任意）Quality of Service（QoS）と暗号化を使用してエクスポート パケットを送信できるようにします。
ステップ 11	**template data timeout** *seconds* **Example:** `Device(config-flow-exporter)# template data timeout 120`	（任意）タイムアウトに基づくテンプレートの再送を設定します。 • *seconds* 引数の範囲は、1 ～ 86400 です（86400 秒 = 24 時間）。
ステップ 12	**transport udp** *udp-port* **Example:** `Device(config-flow-exporter)# transport udp 650`	UDP をトランスポート プロトコルとして設定し、エクスポートされるデータグラムを宛先システムがリスニングする UDP ポートを指定します。 • *udp-port* 引数の範囲は 1 ～ 65536 です。
ステップ 13	**ttl** *seconds* **Example:** `Device(config-flow-exporter)# ttl 15`	（任意）エクスポータによって送信されるデータグラムの存続可能時間（TTL）値を設定します。 • *seconds* 引数の範囲は、1 ～ 255 です。
ステップ 14	**end** **Example:** `Device(config-flow-exporter)# end`	フロー エクスポータ コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フローエクスポータの設定とステータスをチェックするには、**show flow exporter** コマンドを使用します。

Cisco Performance Monitor のフロー レコードの設定

Cisco Performance Monitor のフロー レコードの設定に関する基本概念と手法は、Flexible NetFlow のフロー レコードの場合と同じです。フロー レコードは、収集されたデータを集約して表示する方法を指定します。唯一の大きな違いは、Cisco Performance Monitor の場合、コマンドに **type performance-monitor** が含まれていることです。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow record type performance-monitor** *record-name*
4. **match application** {**name** [**account-on-resolution**] | **vendor** | **version**}
5. **match connection transaction-id**
6. **match flow** {**direction** | **sampler**}
7. **match interface** {**input** | **output**}
8. **match ipv4** {**destination**{**address** | **prefix** [**minimum-mask** *mask*]} | **protocol** | **source** {**address** | **prefix** [**minimum-mask** *mask*]}
9. **match ipv4 fragmentation** {**flags** |**offset**}
10. **match ipv4** {**section** {**header size** *header-size* | **payload size** *payload-size*}
11. **match ipv4 total-length**
12. **match ipv4 ttl**
13. **match ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**}
14. **match ipv6 destination** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]}
15. **match ipv6 extension map**
16. **match ipv6 fragmentation** {**flags** | **id** | **offset**}
17. **match ipv6 hop-limit**
18. **match ipv6 length** {**header** | **payload** | **total**}
19. **match ipv6** {**section** {**header size** *header-size* | **payload size** *payload-size*}
20. **match ipv6 source** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]}
21. **match metadata** {**global-session-id** | **multi-party-session-id**}
22. **match routing** {**destination** | **source**}
23. **match routing is-multicast**
24. **match routing multicast replication-factor**
25. **match transport** {**destination-port** | **igmp** | **rtp** [**ssrc**] | **source-port**}
26. **match transport icmp ipv4** {**code** | **type**}
27. **match transport icmp ipv6** {**code** | **type**}
28. **match transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** |

urgent-pointer | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**}

29. **match transport udp** {**destination-port** | **message-length** | **source-port**}
30. **collect application media** {**bytes**{**rate** | **counter**}| **packets** {**rate**|**counter**} | **events**}
31. **collect application** {**name** [**account-on-resolution**]| **description** | **http host** | **nntp group-name** | **pop3 server** | **rstp host-name** | **sip** {**destination** | **source**} | **smtp** {**sender** | **server**} | **vendor** | **version**}
32. **collect connection**
33. **collect counter** {**bytes** [**long** | **rate**] |**packets**[**dropped** [**long**] | **long**]}
34. **collect datalink mac source address** {**input** | **output**}
35. **collect flow direction**
36. **collect interface** {**input** | **output**}
37. **collect ipv4** {**destination mask** [**minimum-mask** *mask*]} | **dscp** | **source mask** [**minimum-mask** *mask*] | **ttl** [**minimum** | **maximum**]}
38. **collect ipv4 fragmentation** {**flags** | **offset**}
39. **collect ipv4** {**section** {**header size** *header-size* | **prefix**[**payload size** *payload-size*}
40. **collect ipv4 total-length** [**maximum** | **minimum**]
41. **collect ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**}
42. **collect ipv6 destination** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]}
43. **collect ipv6 extension-map**
44. **collect ipv6 fragmentation** {**flags** | **offset**}
45. **collect ipv6 hop-limit** [**maximum**] [**minimum**]
46. **collect ipv6 length**{**header** | **payload** | **total** [**maximum**] [**minimum**] }
47. **collect ipv6** {**section** {**header size** *header-size* | **prefix** [**payload size** *payload-size*}
48. **collect ipv6 source** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]}
49. **collect metadata** {**global-session-id** | **multi-party-session-id**}
50. **collect monitor event**
51. **collect routing forwarding-status** [**reason**]
52. **collect routing is-multicast**
53. **collect routing multicast replication-factor**
54. **collect timestamp internal**
55. **collect timestamp sys-uptime** {**first** | **last**}
56. **collect transport** {**destination-port** | **igmp type** | **source-port** | **event packet-loss counter** | **packets** {**expected counter** | **lost** {**counter** | **rate**} | **out-of-order**} | **round-trip-time** | **rtp jitter** {**minimum** | **mean** | **maximum**}}
57. **collect transport icmp ipv4**
58. **collect transport icmp ipv6**
59. **collect transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**}
60. **collect transport udp** {**destination-port** | **message-length** | **source-port**}
61. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**flow record type performance-monitor** *record-name* **Example:** `Device(config)# flow record type performance-monitor record-8`	フロー レコードを作成し、フロー レコード コンフィギュレーション モードを開始します。 • このコマンドでは、既存のフロー レコードを変更することもできます。
ステップ 4	**match application** {**name** [**account-on-resolution**] | **vendor** | **version**} **Example:** `Device(config-flow-record)# match application name`	アプリケーション名、ベンダー、またはバージョンをキーフィールドとして使用することを指定します。
ステップ 5	**match connection transaction-id** **Example:** `Device(config-flow-record)# match connection transaction-id`	アプリケーション名をキーフィールドとして使用することを指定します。
ステップ 6	**match flow** {**direction** | **sampler**} **Example:** `Device(config-flow-record)# match flow direction`	フロー方向フィールドをキーフィールドとして使用することを指定します。
ステップ 7	**match interface** {**input** | **output**} **Example:** `Device(config-flow-record)# match flow direction`	入力インターフェイスフィールドをキーフィールドとして使用することを指定します。
ステップ 8	**match ipv4** {**destination**{**address** | **prefix** [**minimum-mask** *mask*]} | **protocol** | **source** {**address** | **prefix** [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv4 destination address`	1 つ以上の IPv4 フィールドをキー フィールドとして使用することを指定します。

	Command or Action	Purpose
ステップ 9	**match ipv4 fragmentation** {**flags** |**offset**} **Example:** `Device(config-flow-record)# match ipv4 fragmentation flags`	1 つ以上の IPv4 フィールドをキー フィールドとして使用することを指定します。
ステップ 10	**match ipv4** {**section** {**header size** *header-size* | **payload size** *payload-size*} **Example:** `Device(config-flow-record)# match ipv4 section header size 8`	1 つ以上の IPv4 フィールドをキー フィールドとして使用することを指定します。
ステップ 11	**match ipv4 total-length** **Example:** `Device(config-flow-record)# match ipv4 total-length`	IPv4 の全長フィールドをキーフィールドとして使用することを指定します。
ステップ 12	**match ipv4 ttl** **Example:** `Device(config-flow-record)# match ipv4 ttl`	IPv4 ttl フィールドがキーフィールドとして使用されることを指定します。
ステップ 13	**match ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**} **Example:** `Device(config-flow-record)# match ipv6 dscp`	IPv6 DSCP フィールドがキーフィールドとして使用されることを指定します。
ステップ 14	**match ipv6 destination** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv4 destination address`	IPv6 宛先アドレスフィールドをキーフィールドとして使用することを指定します。
ステップ 15	**match ipv6 extension map** **Example:** `Device(config-flow-record)# match ipv6 extension map`	IPv6 拡張マップフィールドをキーフィールドとして使用することを指定します。
ステップ 16	**match ipv6 fragmentation** {**flags** | **id** | **offset**} **Example:** `Device(config-flow-record)# match ipv6 fragmentation flags`	IPv6 フラグメンテーション フラグ フィールドをキーフィールドとして使用することを指定します。

	Command or Action	Purpose
ステップ 17	**match ipv6 hop-limit** **Example:** `Device(config-flow-record)# match ipv6 hop-limit`	IPv6 ホップリミットフィールドをキーフィールドとして使用することを指定します。
ステップ 18	**match ipv6 length** {**header** | **payload** | **total**} **Example:** `Device(config-flow-record)# match ipv6 length total`	IPv6 の全長フィールドをキーフィールドとして使用することを指定します。
ステップ 19	**match ipv6** {**section** {**header size** *header-size* | **payload size** *payload-size*} **Example:** `Device(config-flow-record)# match ipv6 section header size 8`	IPv6セクションのヘッダーサイズフィールドをキーフィールドとして使用することを指定します。
ステップ 20	**match ipv6 source** {**address** | {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# match ipv6 source address`	IPv6 送信元アドレスフィールドをキーフィールドとして使用することを指定します。
ステップ 21	**match metadata** {**global-session-id** | **multi-party-session-id**} **Example:** `Device(config-flow-record)# match metadata global-session-id`	メタデータセッション ID フィールドをキーフィールドとして使用することを指定します。
ステップ 22	**match routing** {**destination** | **source**} **Example:** `Device(config-flow-record)# match routing source`	ルーティング送信元フラグフィールドをキーフィールドとして使用することを指定します。
ステップ 23	**match routing is-multicast** **Example:** `Device(config-flow-record)# match routing is-multicast`	ルーティング is-multicast フラグフィールドをキーフィールドとして使用することを指定します。
ステップ 24	**match routing multicast replication-factor** **Example:** `Device(config-flow-record)# match routing multicast replication-factor`	ルーティング マルチキャスト レプリケーション ファクタ フラグ フィールドをキーフィールドとして使用することを指定します。

	Command or Action	Purpose
ステップ 25	**match transport** {**destination-port** | **igmp** | **rtp** [**ssrc**] | **source-port**} **Example:** `Device(config-flow-record)# match transport destination-port`	Real-time Transport Protocol（RTP）パケット ヘッダーの Synchronization Source（SSRC）フィールドを含め、1つ以上のトランスポート層フィールドをキーフィールドとして使用することを指定します。
ステップ 26	**match transport icmp ipv4** {**code** | **type**} **Example:** `Device(config-flow-record)# match transport icmp ipv4 code`	IPv4 ICMP トランスポート コードフィールドがキーフィールドとして使用されることを指定します。
ステップ 27	**match transport icmp ipv6** {**code** | **type**} **Example:** `Device(config-flow-record)# match transport icmp ipv6 code`	IPv6 ICMP トランスポート コードフィールドがキーフィールドとして使用されることを指定します。
ステップ 28	**match transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**} **Example:** `Device(config-flow-record)# match transport tcp destination-port`	IPv6 TCP トランスポート宛先ポートフィールドがキーフィールドとして使用されることを指定します。
ステップ 29	**match transport udp** {**destination-port** | **message-length** | **source-port**} **Example:** `Device(config-flow-record)# match transport udp destination-port`	IPv6 UDP トランスポート宛先ポートフィールドがキーフィールドとして使用されることを指定します。
ステップ 30	**collect application media** {**bytes**{**rate** | **counter**}| **packets** {**rate**|**counter**} | **events**} **Example:** `Device(config-flow-record)# collect application media events`	アプリケーション メディアのバイト、パケット、またはイベントを非キー フィールドとして使用することを指定します。アプリケーション イベントは、フローの反応ステートメントで指定されているいずれかのしきい値をモニタリング インターバルで少なくとも 1 回超えることがあった場合や、メディア パケットが検出されなかった場合に発生します。

	Command or Action	Purpose
ステップ 31	**collect application** {**name** [**account-on-resolution**]| **description** | **http host** | **nntp group-name** | **pop3 server** | **rstp host-name** | **sip** {**destination** | **source**} | **smtp** {**sender** | **server**} | **vendor** | **version**} **Example:** `Device(config-flow-record)# collect application name`	アプリケーション名を非キーフィールドとして使用することを指定します。
ステップ 32	**collect connection** **Example:** `Device(config-flow-record)# collect connection initiator`	接続イニシエーターが非キーフィールドとして使用されることを指定します。
ステップ 33	**collect counter** {**bytes** [**long** | **rate**] |**packets**[**dropped** [**long**] | **long**]} **Example:** `Device(config-flow-record)# collect counter bytes long`	非キー フィールドとして使用するバイトまたはパケットの数を指定します。
ステップ 34	**collect datalink mac source address** {**input** | **output**} **Example:** `Device(config-flow-record)# collect flow direction`	フロー方向フィールドを非キー フィールドとして使用することを指定します。
ステップ 35	**collect flow direction** **Example:** `Device(config-flow-record)# collect flow direction`	フロー方向フィールドを非キー フィールドとして使用することを指定します。
ステップ 36	**collect interface** {**input** | **output**} **Example:** `Device(config-flow-record)# collect interface input`	入力インターフェイスまたは出力インターフェイスを非キー フィールドとして使用することを指定します。
ステップ 37	**collect ipv4** {**destination mask** [**minimum-mask** *mask*]} | **dscp** | **source mask** [**minimum-mask** *mask*] | **ttl** [**minimum** | **maximum**]} **Example:** `Device(config-flow-record)# collect ipv4 dscp`	IPv4 DSCP フィールドが非キーフィールドとして使用されることを指定します。

	Command or Action	Purpose
ステップ 38	**collect ipv4 fragmentation** {**flags** | **offset**} **Example:** `Device(config-flow-record)# collect ipv4 fragmentation flags`	IPv4 フラグメンテーション フラグ フィールドが非キーフィールドとして使用されることを指定します。
ステップ 39	**collect ipv4** {**section** {**header size** *header-size* | **prefix**[**payload size** *payload-size*} **Example:** `Device(config-flow-record)# collect ipv4 section header size 8`	IPv4 セクションのヘッダーサイズフィールドを非キーフィールドとして使用することを指定します。
ステップ 40	**collect ipv4 total-length** [**maximum** | **minimum**] **Example:** `Device(config-flow-record)# collect ipv4 total-length`	IPv4 全長フィールドが非キーフィールドとして使用されることを指定します。
ステップ 41	**collect ipv6** {**dscp** | **flow-label** | **next-header** | **payload-length** | **precedence** | **protocol** | **traffic-class** | **version**} **Example:** `Device(config-flow-record)# collect ipv6 dscp`	IPv6 DSCP フィールドが非キーフィールドとして使用されることを指定します。
ステップ 42	**collect ipv6 destination** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# collect ipv6 destination mask`	IPv6 宛先マスクフィールドが非キーフィールドとして使用されることを指定します。
ステップ 43	**collect ipv6 extension-map** **Example:** `Device(config-flow-record)# collect ipv6 extension-map`	IPv6 拡張マップフィールドが非キーフィールドとして使用されることを指定します。
ステップ 44	**collect ipv6 fragmentation** {**flags** | **offset**} **Example:** `Device(config-flow-record)# collect ipv6 fragmentation flags`	IPv6 フラグメンテーション フラグ フィールドが非キーフィールドとして使用されることを指定します。
ステップ 45	**collect ipv6 hop-limit** [**maximum**] [**minimum**] **Example:**	IPv6 ホップ制限フィールドが非キーフィールドとして使用されることを指定します。

	Command or Action	Purpose
	`Device(config-flow-record)# collect ipv6 hop-limit`	
ステップ 46	**collect ipv6 length** {**header** | **payload** | **total** [**maximum**] [**minimum**] } **Example:** `Device(config-flow-record)# collect ipv6 length total`	IPv6 全長フィールドが非キーフィールドとして使用されることを指定します。
ステップ 47	**collect ipv6** {**section** {**header size** *header-size* | **prefix** [**payload size** *payload-size*} **Example:** `Device(config-flow-record)# collect ipv6 section header size 8`	IPv6 セクションのヘッダーサイズフィールドを非キーフィールドとして使用することを指定します。
ステップ 48	**collect ipv6 source** {**address** {**mask** | **prefix**} [**minimum-mask** *mask*]} **Example:** `Device(config-flow-record)# collect ipv6 source mask`	IPv6 送信元マスクフィールドが非キーフィールドとして使用されることを指定します。
ステップ 49	**collect metadata** {**global-session-id** | **multi-party-session-id**} **Example:** `Device(config-flow-record)# collect meatdata global-session-id`	メタデータセッションIDフィールドを非キーフィールドと使用することを指定します。
ステップ 50	**collect monitor event** **Example:** `Device(config-flow-record)# collect monitor event`	モニタ イベント フィールドを非キー フィールドとして使用することを指定します。モニタ イベントは、メディア アプリケーション パケットが検出されない場合に発生します
ステップ 51	**collect routing forwarding-status** [**reason**] **Example:** `Device(config-flow-record)# collect routing forwarding-status`	1 つ以上のルーティング属性を非キー フィールドとして使用することを指定します。
ステップ 52	**collect routing is-multicast** **Example:** `Device(config-flow-record)# collect routing is-multicast`	ルーティング is-multicast フィールドが非キーフィールドとして使用されることを指定します。

	Command or Action	Purpose
ステップ 53	**collect routing multicast replication-factor** **Example:** `Device(config-flow-record)# collect routing multicast replication-factor`	ルーティング マルチキャスト レプリケーション ファクタ フィールドを非キーフィールドとして使用することを指定します。
ステップ 54	**collect timestamp internal** **Example:** `Device(config-flow-record)# collect timestamp internal`	フローで最初または最後に検出されたパケットのシステム タイムスタンプを非キー フィールドとして使用することを指定します。
ステップ 55	**collect timestamp sys-uptime** {**first** | **last**} **Example:** `Device(config-flow-record)# collect timestamp sys-uptime`	sys-uptime のシステムタイムスタンプが非キー フィールドとして使用されることを指定します。
ステップ 56	**collect transport** {**destination-port** | **igmp type** | **source-port** | **event packet-loss counter** | **packets** {**expected counter** | **lost** {**counter** | **rate**} | **out-of-order**} | **round-trip-time** | **rtp jitter** {**minimum** | **mean** | **maximum**}} **Example:** `Device(config-flow-record)# collect transport packets expected counter`	1 つ以上のトランスポート層フィールドを非キー フィールドとして使用することを指定します。これらのフィールドには、次のメトリックが含まれます。 • パケット損失カウンタ • 予想パケット カウンタ • ジッター
ステップ 57	**collect transport icmp ipv4** **Example:** `Device(config-flow-record)# collect transport icmp ipv4`	トランスポート ICMP IPv4 フィールドが非キー フィールドとして使用されることを指定します。
ステップ 58	**collect transport icmp ipv6** **Example:** `Device(config-flow-record)# collect transport icmp ipv6`	トランスポート ICMP IPv6 フィールドが非キー フィールドとして使用されることを指定します。
ステップ 59	**collect transport tcp** {**acknowledgement-number** | **destination-port** | **flags** {[**ack**] | [**cwr**] | [**ece**] | [**fin**] | [**psh**] | [**syn**] | [**urg**]} | **header-length** | **maximum-segment-size** | **sequence-number** | **urgent-pointer** | **window-size** | **window-size-maximum** | **window-size-minimum** | **window-size-average**} **Example:**	

	Command or Action	Purpose
	`Device(config-flow-record)# collect transport tcp destination-port`	
ステップ 60	**collect transport udp** {**destination-port** | **message-length** | **source-port**} **Example:** `Device(config-flow-record)# collect transport udp destination-port`	トランスポートUDP宛先ポートフィールドが非キーフィールドとして使用されることを指定します。
ステップ 61	**end** **Example:** `Device(config-flow-record)# end`	フロー レコード コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フローポリシーの設定とステータスを確認するには、**show flow record type performance-monitor** コマンドを使用します。

AVC フェーズ 2 の使用状況レコードの設定

入力使用状況レコードを設定するには、次の必須タスクを実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. flow record flow-record-name
4. match interface input
5. match flow direction
6. match connection client {ipv4 | ipv6} address
7. match connection client transport port
8. match connection server {ipv4 | ipv6} address
9. match connection server transport port
10. match ipv4 {initiator | responder} address
11. match ipv6 {initiator | responder} address
12. match transport {initiator | responder} port
13. match routing vrf {input | output}
14. match datalink {destination-vlan-id | source-vlan-id}
15. match datalink vlan {input | output}
16. match datalink mac {destination | source} address {input | output}
17. match flow {class | qos-class}
18. match policy performance-monitor classification hierarchy
19. match services waas segment

20. collect interface output
21. collect flow direction
22. collect timestamp sys-uptime first
23. collect timestamp sys-uptime last
24. collect counter bytes long
25. collect counter packets
26. collect connection client {ipv4 | ipv6} address
27. collect connection client counter {bytes long | packets long | packets retransmitted}
28. collect connection client transport port
29. collect connection new-connections
30. collect connection sum-duration
31. collect routing vrf {input | output}
32. collect connection delay application {sum | min | max}
33. collect connection delay network {client-to-server | to-server [histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7}] {sum | min | max}
34. collect connection delay response {client-to-server | to-client | to-server} {sum | min | max}
35. collect connection performance application-delay {sum | min | max}
36. collect connection performance initiator bytes long
37. collect connection performance initiator count re-transmitted-packets
38. collect connection performance initiator network-delay {sum | min | max}
39. collect connection performance initiator packets long
40. collect connection performance network-delay {sum | min | max}
41. collect connection performance new-transaction-time
42. collect connection performance total-transaction-time {sum | min | max}
43. collect connection performance total-transaction-time {sum | min | max}
44. collect connection performance responder bytes long
45. collect connection performance responder response-time {sum | min | max}
46. collect connection performance responder network-delay {sum | min | max}
47. collect connection performance responder count {histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7} | late-responses | responses}
48. collect connection performance responder packets long
49. collect connection performance total-delay {sum | min | max}
50. collect connection performance total-transaction-time {sum | min | max}
51. collect connection server {ipv4 | ipv6} address
52. collect connection server counter {bytes long | packets long | packets retransmitted}
53. collect connection server transport port
54. collect connection transaction {counter complete | duration {sum | min | max}}
55. collect datalink {destination-vlan-id | source-vlan-id}
56. collect datalink mac {destination | source} address {input | output}
57. collect datalink vlan {input | output}
58. collect policy performance-monitor classification hierarchy
59. collect services waas {passthrough-reason | segment}
60. collect timestamp absolute {first | last}
61. collect transport tcp {option map | window-size {sum | minimum | maximum} | maximum-segment-size}

62. end

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:** `Router> 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:** `Router# 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	flow record flow-record-name **Example:** `Router(config)# flow record my-input-usage-monitor`	フロー レコードを作成し、フロー レコード コンフィギュレーション モードを開始します。
ステップ 4	match interface input **Example:** `Router(config-flow-record)# match interface input`	パケットの入力インターフェイスをフロー レコードのキーフィールドとして設定します。 input：トラフィックはCiscoルータの入力インターフェイスに到着します。
ステップ 5	match flow direction **Example:** `Router(config-flow-record)# match flow direction`	フローレコードの方向をキーフィールドとして設定します。方向は input または output のいずれかです。
ステップ 6	match connection client {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# match connection client ipv6 address`	クライアントの IPv6 アドレスをフローレコードのキーフィールドとして設定します。
ステップ 7	match connection client transport port **Example:** `Router(config-flow-record)# match connection client transport port`	クライアントの接続ポートをフローレコードのキーフィールドとして設定します。
ステップ 8	match connection server {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# match connection server ipv6 address`	サーバーの IPv6 アドレスをフローレコードのキーフィールドとして設定します。

	Command or Action	Purpose
ステップ 9	match connection server transport port **Example:** `Router(config-flow-record)# match connection server transport port`	サーバーの接続ポートをフローレコードのキーフィールドとして設定します。
ステップ 10	match ipv4 {initiator | responder} address **Example:** `Router(config-flow-record)# match ipv4 initiator address`	（任意）IPv4 ネットワークの場合、イニシエータまたはレスポンダの IPv4 アドレスをキーフィールドとして設定します。方向は input または output のいずれかです。
ステップ 11	match ipv6 {initiator | responder} address **Example:** `Router(config-flow-record)# match ipv6 initiator address`	（任意）IPv6 ネットワークの場合、イニシエータまたはレスポンダの IPv6 アドレスをキーフィールドとして設定します。方向は input または output のいずれかです。
ステップ 12	match transport {initiator | responder} port **Example:** `Router(config-flow-record)# match transport initiator port`	（任意）イニシエータまたはレスポンダのトランスポートポートをキーフィールドとして設定します。
ステップ 13	match routing vrf {input | output} **Example:** `Router(config-flow-record)# match routing vrf input`	（任意）着信パケットまたは発信パケットの Virtual Routing and Forwarding（VRF）ID をキーフィールドとして設定します。
ステップ 14	match datalink {destination-vlan-id | source-vlan-id} **Example:** `Router(config-flow-record)# match datalink destination-vlan-id`	（任意）宛先 VLAN ID をキーフィールドとして設定します。
ステップ 15	match datalink vlan {input | output} **Example:** `Router(config-flow-record)# match datalink vlan input`	（任意）着信パケットまたは発信パケットの VLAN ID をキーフィールドとして設定します。
ステップ 16	match datalink mac {destination | source} address {input | output} **Example:** `Router(config-flow-record)# match datalink mac destination address output`	（任意）宛先 MAC アドレスをキーフィールドとして設定します。

	Command or Action	Purpose
ステップ 17	match flow {class | qos-class} **Example:** `Router(config-flow-record)# match flow class`	クラス ID をフローレコードのキーフィールドとして使用するように設定します。
ステップ 18	match policy performance-monitor classification hierarchy **Example:** `Router(config-flow-record)# match policy performance-monitor classification hierarchy`	フローレコードのキーフィールドとしてパフォーマンス モニター ポリシー分類階層の使用を設定します。
ステップ 19	match services waas segment **Example:** `Router(config-flow-record)# match services waas segment`	WAAS セグメントをフローレコードのキーフィールドとして使用するように設定します。
ステップ 20	collect interface output **Example:** `Router(config-flow-record)# collect interface output`	出力インターフェイスをフローレコードの非キーフィールドとして設定し、フローレコードのフローから出力インターフェイスフィールドを収集できるようにします。
ステップ 21	collect flow direction **Example:** `Router(config-flow-record)# collect flow direction`	フロー方向をフローレコードの非キーフィールドとして設定します。
ステップ 22	collect timestamp sys-uptime first **Example:** `Router(config-flow-record)# collect timestamp sys-uptime first`	フロー内で最初に検出されたパケットのシステム稼働時間を、フローレコードの非キーフィールドとして設定します。 • first：フローの最初のパケットが確認されたときのシステム稼働時間を非キーフィールドとして設定し、フローの最初のパケットが確認されたときのシステム稼動時間に基づいてタイムスタンプを収集します。
ステップ 23	collect timestamp sys-uptime last **Example:** `Router(config-flow-record)# collect timestamp sys-uptime last`	フロー内で最後に検出されたパケットのシステム稼働時間を、フローレコードの非キーフィールドとして設定します。 • last：フローの最後のパケットが確認されたときのシステム稼働時間を非キーフィールドとして設定し、フローの最後のパケットが確認されたときのシステム稼働時間に基づいてタイムスタンプを収集します。

	Command or Action	Purpose
ステップ 24	collect counter bytes long **Example:** `Router(config-flow-record)# collect counter bytes long`	フローレコードの非キーフィールドとしてフロー内のバイト数またはパケット数を設定します。 • bytes：フローの確認されたバイト数を非キーフィールドとして設定し、フローの合計バイト数を収集します。 • long：32 ビットカウンタではなく 64 ビットカウンタを使用して、フローからバイトまたはパケットの合計数を収集できるようにします。
ステップ 25	collect counter packets **Example:** `Router(config-flow-record)# collect counter packets`	フロー内のパケット数をフローレコードの非キーフィールドとして設定します。 • packets：フローで確認されるパケット数を非キーフィールドとして設定し、フローから合計パケット数を収集します。
ステップ 26	collect connection client {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# collect connection client ipv6 address`	クライアントの IPv6 アドレスをフローレコードの非キーフィールドとして設定します。
ステップ 27	collect connection client counter {bytes long | packets long | packets retransmitted} **Example:** `Router(config-flow-record)# collect connection client counter packets retransmitted`	フローレコードの非キーフィールドとして再送信されるクライアントパケットの数を設定します。
ステップ 28	collect connection client transport port **Example:** `Router(config-flow-record)# collect connection client transport port`	クライアント接続ポートをフローレコードの非キーフィールドとして設定します。
ステップ 29	collect connection new-connections **Example:** `Router(config-flow-record)# collect connection new-connections`	観測期間中に開かれた TCP または UDP 接続の数をカウントします。観測期間は、フローの開始タイムスタンプと終了タイムスタンプで指定できます。
ステップ 30	collect connection sum-duration **Example:** `Router(config-flow-record)# collect connection sum-duration`	観測期間中に使用されていたすべての TCP または UDP 接続の合計時間（秒単位）を集約します。たとえば、5 つの同時接続がそれぞれ 10 秒間ある場合、値は 50 秒になります。

	Command or Action	Purpose
ステップ 31	collect routing vrf {input | output} **Example:** `Router(config-flow-record)# collect routing vrf output`	フローレコードの非キーフィールドとして、着信または発信パケット出力の Virtual Routing and Forwarding（VRF）ID を設定します。
ステップ 32	collect connection delay application {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay application sum`	アプリケーション遅延の合計量をフローレコードの非キーフィールドとして設定します。
ステップ 33	collect connection delay network {client-to-server | to-server [histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7}] {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay network client-to-server sum`	クライアントとサーバー間のネットワーク遅延の合計量を、フローレコードの非キーフィールドとして設定します。
ステップ 34	collect connection delay response {client-to-server | to-client | to-server} {sum | min | max} **Example:** `Router(config-flow-record)# collect connection delay response client-to-server sum`	クライアントとサーバー間の応答遅延の合計をフローレコードの非キーフィールドとして設定します。
ステップ 35	collect connection performance application-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance application-delay sum`	フローレコードの非キーフィールドとして合計アプリケーション遅延を設定します。
ステップ 36	collect connection performance initiator bytes long **Example:** `Router(config-flow-record)# collect connection performance initiator bytes long`	Mediatraceイニシエータのロングバイト数をフローレコードの非キーフィールドとして設定します。
ステップ 37	collect connection performance initiator count re-transmitted-packets **Example:** `Router(config-flow-record)# collect connection performance initiator count re-transmitted-packets`	フローレコードの非キーフィールドとして Mediatraceイニシエータの再転送パケット数を設定します。

	Command or Action	Purpose
ステップ 38	collect connection performance initiator network-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance initiator network-delay sum`	Mediatraceイニシエータの合計ネットワーク遅延をフローレコードの非キーフィールドとして設定します。
ステップ 39	collect connection performance initiator packets long **Example:** `Router(config-flow-record)# collect connection performance initiator packets long`	Mediatrace イニシエータのロングパケット数をフローレコードの非キーフィールドとして設定します。
ステップ 40	collect connection performance network-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance network-delay sum`	フローレコードの非キーフィールドとして合計ネットワーク遅延を設定します。
ステップ 41	collect connection performance new-transaction-time **Example:** `Router(config-flow-record)# collect connection performance new-transaction`	新しいトランザクションフィールドをフローレコードの非キーフィールドとして設定します。
ステップ 42	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	合計トランザクション時間をフローレコードの非キーフィールドとして設定します。
ステップ 43	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	合計トランザクション時間をフローレコードの非キーフィールドとして設定します。
ステップ 44	collect connection performance responder bytes long **Example:** `Router(config-flow-record)# collect connection performance responder bytes long`	Mediatrace レスポンダのロングバイト数をフローレコードの非キーフィールドとして設定します。
ステップ 45	collect connection performance responder response-time {sum | min | max} **Example:**	Mediatrace レスポンダの合計応答時間をフローレコードの非キーフィールドとして設定します。

	Command or Action	Purpose
	`Router(config-flow-record)# collect connection performance responder response-time sum`	
ステップ 46	collect connection performance responder network-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance responder network-delay sum`	Mediatrace レスポンダの合計ネットワーク遅延をフローレコードの非キーフィールドとして設定します。
ステップ 47	collect connection performance responder count {histogram { bucket1 | bucket2 | bucket3 | bucket4 | bucket5 | bucket6 | bucket7} | late-responses | responses} **Example:** `Router(config-flow-record)# collect connection performance responder count late-responses`	Mediatrace レスポンダの遅延応答数をフローレコードの非キーフィールドとして設定します。
ステップ 48	collect connection performance responder packets long **Example:** `Router(config-flow-record)# collect connection performance responder packets long`	Mediatrace レスポンダのロングパケット数をフローレコードの非キーフィールドとして設定します。
ステップ 49	collect connection performance total-delay {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-delay sum`	フローレコードの非キーフィールドとして合計接続遅延を設定します。
ステップ 50	collect connection performance total-transaction-time {sum | min | max} **Example:** `Router(config-flow-record)# collect connection performance total-transaction-time sum`	合計トランザクション時間をフローレコードの非キーフィールドとして設定します。
ステップ 51	collect connection server {ipv4 | ipv6} address **Example:** `Router(config-flow-record)# collect connection server ipv6 address`	サーバーの IPv6 アドレスをフローレコードの非キーフィールドとして設定します。
ステップ 52	collect connection server counter {bytes long | packets long | packets retransmitted} **Example:**	フローレコードの非キーフィールドとして再送信されるサーバーパケットの数を設定します。

	Command or Action	Purpose
	`Router(config-flow-record)# collect connection server counter packets retransmitted`	
ステップ 53	collect connection server transport port **Example:** `Router(config-flow-record)# collect connection server transport port`	サーバー接続ポートをフローレコードの非キーフィールドとして設定します。
ステップ 54	collect connection transaction {counter complete | duration {sum | min | max}} **Example:** `Router(config-flow-record)# collect connection transaction duration sum`	トランザクションの合計期間をフローレコードの非キーフィールドとして設定します。
ステップ 55	collect datalink {destination-vlan-id | source-vlan-id} **Example:** `Router(config-flow-record)# collect datalink destination-vlan-id`	（任意）宛先 VLAN ID を非キーフィールドとして設定します。
ステップ 56	collect datalink mac {destination | source} address {input | output} **Example:** `Router(config-flow-record)# collect datalink mac destination address input`	（任意）宛先MACアドレスを非キーフィールドとして設定します。
ステップ 57	collect datalink vlan {input | output} **Example:** `Router(config-flow-record)# collect datalink vlan input`	（任意）着信パケットまたは発信パケットのVLAN ID を非キーフィールドとして設定します。
ステップ 58	collect policy performance-monitor classification hierarchy **Example:** `Router(config-flow-record)# collect policy performance-monitor classification hierarchy`	フローレコードの非キーフィールドとしてパフォーマンス モニター ポリシー分類階層の使用を設定します。
ステップ 59	collect services waas {passthrough-reason | segment} **Example:** `Router(config-flow-record)# collect services waas segment`	WAASセグメントをフローレコードの非キーフィールドとして使用するように設定します。

	Command or Action	Purpose
ステップ 60	collect timestamp absolute {first | last} **Example:** `Router(config-flow-record)# collect timestamp absolute first`	最初のタイムスタンプをフローレコードの非キーフィールドとして使用するように設定します。
ステップ 61	collect transport tcp {option map | window-size {sum | minimum | maximum} | maximum-segment-size} **Example:** `Router(config-flow-record)# collect connection performance initiator network-delay sum`	Mediatraceイニシエータの合計ネットワーク遅延をフローレコードの非キーフィールドとして設定します。
ステップ 62	end **Example:** `Router(config-flow-record)# end`	フロー レコード コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Cisco Performance Monitor のフロー モニタの設定

Cisco Performance Monitor のフロー モニタの設定に関する基本概念は、Flexible NetFlow のフロー モニタの場合と同じです。各フロー モニタには、別々のキャッシュが割り当てられ、キャッシュ エントリの内容とレイアウトを定義するレコードが必要です。

フロー モニタを設定する場合は、次のいずれかを使用する必要があります:

- 設定済みの既存のフロー レコード
- 次のいずれかのデフォルトの事前定義済みレコード
  - デフォルト RTP レコード（**default-rtp**）
  - デフォルト TCP レコード（**default-tcp**）
  - Flexible NetFlow の「NetFlow IPv4 original input」

Note フロー レコードを変更するには、関連付けられているすべてのフロー モニタから削除する必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow monitor type performance-monitor** *monitor-name*
4. **description** *description*
5. **cache** {**entries**| **timeout**| **type**}

6. **statistics** {**packet**}
7. **exporter** *exporter-name*
8. **record** {*record-name*| **default-rtp**| **default-tcp**|**netflow ipv4 original-input**}
9. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**flow monitor type performance-monitor** *monitor-name* **Example:** `Device(config)# flow monitor type performance-monitor FLOW-MONITOR-2`	フローモニターを作成し、フローモニターコンフィギュレーション モードを開始します。 • このコマンドでは、既存のフローモニターを変更することもできます。
ステップ 4	**description** *description* **Example:** `Device(config-flow-monitor)# description Used for monitoring IPv4 traffic`	（任意）フロー モニターの説明を作成します。
ステップ 5	**cache** {**entries**| **timeout**| **type**} **Example:** `Device(config-flow-monitor)# cache timeout 20`	（任意）フローモニタのキャッシュを作成します。
ステップ 6	**statistics** {**packet**} **Example:** `Device(config-flow-monitor)# statistics`	（任意）フローモニタの統計情報を収集するかどうかを指定します。
ステップ 7	**exporter** *exporter-name* **Example:** `Device(config-flow-monitor)# exporter export-4`	フローモニタのフローエクスポータを指定します。
ステップ 8	**record** {*record-name*| **default-rtp**| **default-tcp**|**netflow ipv4 original-input**} **Example:**	フロー モニタのフロー レコードを指定します。

	Command or Action	Purpose
	`Device(config-flow-monitor)# record default-rtp`	
ステップ 9	**end** **Example:** `Device(config-flow-monitor)# end`	フロー モニタ コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フローモニターの設定とステータスを確認するには、**show flow monitor type performance-monitor** コマンドと **show running-config flow monitor** コマンドを使用します。

Cisco Performance Monitor のフロー クラスの設定

Cisco Performance Monitor のクラスの設定に関する基本概念と手法は、他のタイプのクラスの場合と同じです。クラスは、モニタリング対象のフロー トラフィックを決定するフィルタを指定します。フィルタは、さまざまな match コマンドをクラス マップ モードで使用して設定します。

まだフロー モニタを設定していない場合は、次のいずれかを実行できます。

Note ネスト形式のクラス マップはサポートされていません。つまり、**class-map** コマンドはクラス マップ コンフィギュレーション モード（config-cmap）では使用できません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-name*
4. **description** *description*
5. **match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map** *class-map-name* | **cos** *cos-value* | **destination-address mac** *address* | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}}
6. **rename** *class-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**class-map** *class-name* **Example:** `Device(config)# class-map class-4`	ポリシーに含めるクラスを指定します。ポリシーに含める各クラスについて、このコマンドを繰り返し実行します。
ステップ 4	**description** *description* **Example:** `Device(config-cmap)# description match any packets`	（任意）フロー クラスの説明を作成します。
ステップ 5	**match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map** *class-map-name* | **cos** *cos-value* | **destination-address mac** *address* | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}} **Example:** `Device(config-cmap)# match any`	分類基準を指定します。 詳細および例については、『*Cisco Media Monitoring Command Reference*』を参照してください。
ステップ 6	**rename** *class-name* **Example:** `Device(config-cmap)# rename class-4`	フロー クラスの新しい名前を指定します。

	Command or Action	Purpose
ステップ 7	**end** **Example:** `Device(config-cmap)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フロークラスの設定とステータスを確認するには、**show policy-map type performance-monitor** または **show class-map** コマンドを使用します。

既存のフロー モニタを使用した Cisco Performance Monitor のフロー ポリシーの設定

Cisco Performance Monitor のクラスの設定に関する基本概念と手法は、他のタイプのクラスの場合と同じです。クラスは、どのフロー モニタを含めるかを指定します。唯一の大きな違いは、Cisco Performance Monitor の場合、**policy-map** コマンドに **type performance-monitor** が含まれていることです。

フロー モニタをまだ設定していない場合や、既存のフロー モニタを新しいクラスに使用しない場合は、flow monitor inline オプションを使用し、どのフロー レコードおよびフロー エクスポータを含めるかを指定して、フロー モニタを設定できます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type performance-monitor** *policy-name*
4. **parameter-map type performance-monitor system-default-aor**
5. **class** {*class-name* | **class-default**}
6. **flow monitor** *monitor-name*
7. **monitor metric ip-cbr**
8. **rate layer3** {byte-*rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
9. **exit**
10. **monitor metric rtp**
11. **clock-rate** {*type-number* | *type-name* | **default**} *rate*
12. **max-dropout** *number*
13. **max-reorder** *number*
14. **min-sequential** *number*
15. **ssrc maximum** *number*
16. exit
17. **monitor parameters**
18. **flows** *number*
19. **interval duration** *number*

20. **history** *number*
21. **timeout** *number*
22. **exit**
23. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
24. **action** {**snmp** | **syslog**}
25. **alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**}
26. **alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*}
27. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*}
28. **description** *description*
29. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**policy-map type performance-monitor** *policy-name* **Example:** `Device(config)# policy-map type performance-monitor FLOW-MONITOR-4`	ポリシーを作成し、ポリシーコンフィギュレーション モードを開始します。 • このコマンドでは、既存のポリシーを変更することもできます。
ステップ 4	**parameter-map type performance-monitor system-default-aor** **Example:** `Device(config-pmap)# parameter-map type performance-monitor system-default-aor`	Performance Monitor のパラメータ マップを作成します。使用可能な唯一のマップは system-default -aor マップです。
ステップ 5	**class** {*class-name*	**class-default**} **Example:** `Device(config-pmap)# class class-4`	ポリシーに含めるクラスを指定します。ポリシーに含める各クラスについて、このコマンドを繰り返し実行します。
ステップ 6	**flow monitor** *monitor-name* **Example:** `Device(config-pmap-c)# flow monitor FLOW-MONITOR-4`	フロー モニタ コンフィギュレーション モードを開始します。既存のフローモニターを使用しない場合は、既存のフロー ポリシーを使用せずに Cisco Performance Monitor ポリシーをインターフェイスに適用する方法, on page 486の説明に従って、**inline** オ

	Command or Action	Purpose
		プションを使用して新しいフローモニターを設定できます。
ステップ 7	**monitor metric ip-cbr** **Example:** `Device(config-pmap-c)# monitor metric ip-cbr`	（任意）IP-CBR モニタ メトリック コンフィギュレーション モードを開始します。
ステップ 8	**rate layer3** {byte-*rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-pmap-c-mipcbr)# rate layer3 248 mbps`	（任意）メトリックのモニタリングのレートを指定します。 • *byte-rate*：データレート（単位：Bps、kBps、mBps、または gBps）。指定できる範囲は 1 ～ 65535 です。 • **packet**：パケットレート（単位：pps）。
ステップ 9	**exit** **Example:** `Device(config-pmap-c-mipcbr)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 10	**monitor metric rtp** **Example:** `Device(config-pmap-c)# monitor metric rtp`	RTP モニタ メトリック コンフィギュレーション モードを開始します。
ステップ 11	**clock-rate** {*type-number* | *type-name* | **default**} *rate* **Example:** `Device(config-pmap-c-mrtp)# clock-rate 8 9600`	RTP ビデオ モニタリング メトリックのサンプリングに使用するクロック レートを指定します。 クロックタイプの番号と名前の詳細については、『*Cisco Media Monitoring Command Reference*』を参照してください。 *rate* の範囲は 1 ～ 192 kHz です。
ステップ 12	**max-dropout** *number* **Example:** `Device(config-pmap-c-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.
ステップ 13	**max-reorder** *number* **Example:** `Device(config-pmap-c-mrtp)# max-reorder 4`	RTP ビデオ モニタリング メトリックのサンプリング時に許可される順序変更の最大数を指定します。
ステップ 14	**min-sequential** *number* **Example:**	ストリームを RTP フローとして識別するために必要な連続パケットの最小数を指定します。

	Command or Action	Purpose
	`Device(config-pmap-c-mrtp)# min-sequential 2`	
ステップ 15	**ssrc maximum** *number* **Example:** `Device(config-pmap-c-mrtp)# ssrc maximum 20`	同じフロー内でモニタできるSSRCの最大数を指定します。フローは、プロトコル、送信元と宛先のアドレス、および送信元と宛先のポートによって定義されます。
ステップ 16	exit **Example:** `Device(config-pmap-c-mrtp)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 17	**monitor parameters** **Example:** `Device(config-pmap-c)# monitor parameters`	モニタ パラメータ コンフィギュレーション モードを開始します。
ステップ 18	**flows** *number* **Example:** `Device(config-pmap-c-mparam)# flows 40`	各モニタ キャッシュのフローの最大数を指定します。
ステップ 19	**interval duration** *number* **Example:** `Device(config-pmap-c-mparam)# interval duration 40`	ビデオ モニタリング メトリックのサンプリング間隔（秒）を指定します。
ステップ 20	**history** *number* **Example:** `Device(config-pmap-c-mparam)# history 4`	収集されるビデオ モニタリング メトリックの履歴バケットの数を指定します。
ステップ 21	**timeout** *number* **Example:** `Device(config-pmap-c-mparam)# timeout 20`	停止したフローがデータベースから削除されるまでのインターバルの数を指定します。
ステップ 22	**exit** **Example:** `Device(config-pmap-c-mparam)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 23	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:**	次のメトリックのしきい値を超えた場合の反応を指定できるモードを開始します。 • *ID*：反応設定のID。有効値の範囲は1～65535です。

	Command or Action	Purpose
	`Device(config-pmap-c)# react 41 rtp-jitter-average`	• **media-stop**：フローのトラフィックが検出されません。 • **mrv**：実際のレートと予想レートの差を予想レートで割ることによって算出されるレート。 • **rtp-jitter-average**：平均ジッター。 • **transport-packets-lost-rate** ：損失パケット数を予想パケット数で割ることによって算出されるレート。
ステップ 24	**action** {**snmp** | **syslog**} **Example:** `Device(config-pmap-c-react)# action syslog`	しきい値を超えた場合の報告方法を指定します。
ステップ 25	**alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**} **Example:** `Device(config-pmap-c-react)# alarm severity critical`	報告されるアラームのレベルを指定します。デフォルト設定は**info**です。
ステップ 26	**alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*} **Example:** `Device(config-pmap-c-react)# alarm type discrete`	報告が必要なアラームと見なされるレベルのタイプを指定します。デフォルト設定は**discrete**です。
ステップ 27	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*} **Example:** `Device(config-pmap-c-react)# threshold value ge 20`	報告が必要なアラームと見なされるしきい値のタイプを指定します。 値が指定されておらず、アプリケーション名がキーフィールドとして設定されている場合は、デフォルトのマップで検出されるしきい値が使用されます。値が指定されておらず、また、アプリケーション名がキー フィールドとして設定されていない場合、しきい値にはデフォルト値が使用されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されているが、しきい値が指定されているのは1つの反応設定のみである場合は、設定されている反応の値が優先され、残りのしきい値は無視されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されており、しきい値が1つも設定されてい

	Command or Action	Purpose
		ない場合は、最も小さい反応 ID が割り当てられている設定にデフォルトのしきい値が適用されます。
ステップ 28	**description** *description* **Example:** `Device(config-cmap-c-react)# description rtp-jitter-average above 40`	（任意）反応の説明を作成します。
ステップ 29	**end** **Example:** `Device(config-pmap-c-react)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フローポリシーの設定とステータスを確認するには、**show policy-map type performance-monitor** コマンドを使用します。

既存のフロー モニタを使用しない Cisco Performance Monitor のフローポリシーの設定

Cisco Performance Monitor のクラスの設定に関する基本概念と手法は、他のタイプのクラスの場合と同じです。クラスは、どのフロー モニタを含めるかを指定します。唯一の大きな違いは、Cisco Performance Monitor の場合、**policy-map** コマンドに **type performance-monitor** が含まれていることです。

フロー モニタをまだ設定していない場合や、既存のフロー モニタを新しいクラスに使用しない場合は、クラス コンフィギュレーション モードで、どのフロー レコードおよびフロー エクスポータを含めるかを指定して、フロー モニタを設定できます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type performance-monitor** *policy-name* **class** class-name
4. **parameter-map type performance-monitor system-default-aor**
5. **class** {*class-name* | **class-default**}
6. **flow monitor inline**
7. **record** {*record-name* | **default-rtp** | **default-tcp**}
8. **exporter** *exporter-name*
9. **exit**
10. monitor metric ip-cbr
11. **rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
12. **exit**

13. **monitor metric rtp**
14. **clock-rate** {*type-number*| *type-name*} *rate*
15. **max-dropout** *number*
16. **max-reorder** *number*
17. **min-sequential** *number*
18. **ssrc maximum** *number*
19. exit
20. **monitor parameters**
21. **flows** *number*
22. **interval duration** *number*
23. **history** *number*
24. **timeout** *number*
25. **exit**
26. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
27. **action** {**snmp** | **syslog**}
28. **alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**}
29. **alarm type** {**discrete** | **grouped {count** *number* | **percent** *number*}
30. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*
31. **description** *description*
32. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**policy-map type performance-monitor** *policy-name* **class** class-name **Example:** `Device(config)# policy-map type performance-monitor FLOW-MONITOR-4`	ポリシーを作成し、ポリシーコンフィギュレーション モードを開始します。 • このコマンドでは、既存のポリシーを変更することもできます。
ステップ 4	**parameter-map type performance-monitor system-default-aor** **Example:** `Device(config-pmap)# parameter-map type performance-monitor system-default-aor`	Performance Monitor のパラメータ マップを作成します。使用可能な唯一のマップは system-default -aor マップです。

	Command or Action	Purpose
ステップ 5	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class class-4`	ポリシーに含めるクラスを指定します。ポリシーに含める各クラスについて、このコマンドを繰り返し実行します。
ステップ 6	**flow monitor inline** **Example:** `Device(config-pmap-c)# flow monitor inline`	インライン モードを開始し、新しいフロー モニタを設定できるようにします。
ステップ 7	**record** {*record-name* | **default-rtp** | **default-tcp**} **Example:** `Device(config-pmap-c-flowmon)# record default-tcp`	フロー モニタに関連付けるフロー レコードを指定します。
ステップ 8	**exporter** *exporter-name* **Example:** `Device(config-pmap-c-flowmon)# exporter exporter-4`	フロー エクスポータに関連付けるフロー レコードを指定します。
ステップ 9	**exit** **Example:** `Device(config-pmap-c-flowmon)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 10	monitor metric ip-cbr **Example:** `Device(config-pmap-c)# monitor metric ip-cbr`	（任意）IP-CBR モニタ メトリック コンフィギュレーション モードを開始します。
ステップ 11	**rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-pmap-c-mipcbr)# rate layer3 248 mbps`	（任意）メトリックのモニタリングのレートを指定します。 • *byte-rate*：データレート（単位：Bps、kBps、mBps、または gBps）。指定できる範囲は 1 ～ 65535 です。 • **packet**：パケットレート（単位：pps）。
ステップ 12	**exit** **Example:** `Device(config-pmap-c-mipcbr)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 13	**monitor metric rtp** **Example:**	RTP モニタ メトリック コンフィギュレーション モードを開始します。

	Command or Action	Purpose
	`Device(config-pmap-c)# monitor metric rtp`	
ステップ 14	**clock-rate** {*type-number*| *type-name*} *rate* **Example:** `Device(config-pmap-c-mrtp)# clock-rate 8 9600`	RTP ビデオ モニタリング メトリックのサンプリングに使用するクロック レートを指定します。 クロックタイプの番号と名前の詳細については、『*Cisco Media Monitoring Command Reference*』を参照してください。 *rate* の範囲は 1 ～ 192 kHz です。
ステップ 15	**max-dropout** *number* **Example:** `Device(config-pmap-c-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.
ステップ 16	**max-reorder** *number* **Example:** `Device(config-pmap-c-mrtp)# max-reorder 4`	RTP ビデオ モニタリング メトリックのサンプリング時に許可される順序変更の最大数を指定します。
ステップ 17	**min-sequential** *number* **Example:** `Device(config-pmap-c-mrtp)# min-sequential 2`	ストリームを RTP フローとして識別するために必要な連続パケットの最小数を指定します。
ステップ 18	**ssrc maximum** *number* **Example:** `Device(config-pmap-c-mrtp)# ssrc maximum 20`	同じフロー内でモニタできる SSRC の最大数を指定します。フローは、プロトコル、送信元と宛先のアドレス、および送信元と宛先のポートによって定義されます。
ステップ 19	exit **Example:** `Device(config-pmap-c-mrtp)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 20	**monitor parameters** **Example:** `Device(config-pmap-c)# monitor parameters`	モニタ パラメータ コンフィギュレーション モードを開始します。
ステップ 21	**flows** *number* **Example:** `Device(config-pmap-c-mparam)# flows 40`	各モニタ キャッシュのフローの最大数を指定します。

	Command or Action	Purpose
ステップ 22	**interval duration** *number* **Example:** `Device(config-pmap-c-mparam)# interval duration 40`	モニタリング メトリックを収集するためのインターバルの長さ（秒）を指定します。
ステップ 23	**history** *number* **Example:** `Device(config-pmap-c-mparam)# history 4`	収集されたビデオ モニタリング メトリックについて表示する履歴インターバルの数を指定します。
ステップ 24	**timeout** *number* **Example:** `Device(config-pmap-c-mparam)# timeout 20`	停止したフローがデータベースから削除されるまでのインターバルの数を指定します。
ステップ 25	**exit** **Example:** `Device(config-pmap-c-mparam)# exit`	ポリシー クラス コンフィギュレーション モードに戻ります。
ステップ 26	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:** `Device(config-pmap-c)# react 41 rtp-jitter-average`	次のメトリックのしきい値を超えた場合の反応を指定できるモードを開始します。 • *ID*：反応設定の ID。有効値の範囲は 1 ～ 65535 です。 • **media-stop**：フローのトラフィックが検出されません。 • **mrv**：実際のレートと予想レートの差を予想レートで割ることによって算出されるレート。 • **rtp-jitter-average**：平均ジッター。 • **transport-packets-lost-rate** ：損失パケット数を予想パケット数で割ることによって算出されるレート。
ステップ 27	**action** {**snmp** | **syslog**} **Example:** `Device(config-pmap-c-react)# action syslog`	しきい値を超えた場合の報告方法を指定します。
ステップ 28	**alarm severity** {**alert** | **critical** | **emergency** | **error** | **info**} **Example:**	報告されるアラームのレベルを指定します。デフォルト設定は、**info** です。

	Command or Action	Purpose
	`Device(config-pmap-c-react)# alarm severity critical`	
ステップ 29	**alarm type** {**discrete** | **grouped** {**count** *number* | **percent** *number*} **Example:** `Device(config-pmap-c-react)# alarm severity critical`	報告が必要なアラームと見なされるレベルのタイプを指定します。デフォルト設定は**discrete**です。
ステップ 30	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end* **Example:** `Device(config-pmap-c-react)# threshold value ge 20`	報告が必要なアラームと見なされるしきい値のタイプを指定します。 値が指定されておらず、アプリケーション名がキーフィールドとして設定されている場合は、デフォルトのマップで検出されるしきい値が使用されます。値が指定されておらず、また、アプリケーション名がキー フィールドとして設定されていない場合、しきい値にはデフォルト値が使用されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されているが、しきい値が指定されているのは1つの反応設定のみである場合は、設定されている反応の値が優先され、残りのしきい値は無視されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されており、しきい値が1つも設定されていない場合は、最も小さい反応 ID が割り当てられている設定にデフォルトのしきい値が適用されます。
ステップ 31	**description** *description* **Example:** `Device(config-cmap-c-react)# description rtp-jitter-average above 40`	（任意）反応の説明を作成します。
ステップ 32	**end** **Example:** `Device(config-pmap-c-react)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

フローポリシーの設定とステータスを確認するには、**show policy-map type performance-monitor** コマンドを使用します。

既存のフロー ポリシーを使用して Cisco Performance Monitor ポリシーをインターフェイスに適用する方法

Cisco Performance Monitor ポリシーをアクティブにする前に、そのポリシーを 1 つ以上のインターフェイスに適用する必要があります。Cisco Performance Monitor をアクティブにするには、次の必須作業を実行します。

Note Cisco Performance Monitor ポリシーを IPv6 インターフェイスに適用できます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-policy type performance-monitor** {**input** | **output**} *policy-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	インターフェイスを指定し、インターフェイス コンフィギュレーション モードを開始します。 IPv6 インターフェイスを指定できます。
ステップ 4	**service-policy type performance-monitor** {**input** | **output**} *policy-name* **Example:** **Example:** `Device(config-if)# service-policy type performance-monitor input mypolicy-map-4` **Example:**	インターフェイスまたは仮想回線（VC）のサービス ポリシーとして使用されるポリシーマップを入力のインターフェイスまたは VC（あるいは出力のインターフェイスまたは VC）に付加します。 • **input**：指定されたポリシーマップを入力インターフェイスまたは入力 VC に対応付けます。 • **output**：指定されたポリシーマップを出力インターフェイスまたは出力 VC に対応付けます。

	Command or Action	Purpose
		• *policy-name*：関連付けるサービスポリシーマップ（**policy-map** コマンドで作成）の名前。名前には最大40文字までの英数字を指定できます。
ステップ5	**end** **Example:** `Device(config-if)# end`	現在のコンフィギュレーションモードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

サービス ポリシーの設定とステータスをチェックするには、次のコマンドを使用します。

- **show performance monitor history**
- **show performance monitor status**
- **show policy-map ypre performance-monitor interface**

既存のフローポリシーを使用せずにCiscoPerformanceMonitorポリシーをインターフェイスに適用する方法

Cisco Performance Monitor ポリシーをアクティブにする前に、そのポリシーを 1 つ以上のインターフェイスに適用する必要があります。Cisco Performance Monitor をアクティブにするには、次の必須作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-policy type performance-monitor inline** {**input** | **output**}
5. **match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map***class-map-name* | **cos** *cos-value* | **destination-address mac** address | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}}
6. **flow monitor** {*monitor-name*| **inline**}
7. {r ecord-name | | } **record default-rtp default-tcp**
8. **exporter** *exporter-name*
9. **exit**

10. **monitor metric ip-cbr**
11. **rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**}
12. **exit**
13. **monitor metric rtp**
14. **clock-rate** {*type-number*| *type-name*} *rate*
15. **max-dropout** *number*
16. **max-reorder** *number*
17. **min-sequential** *number*
18. **ssrc maximum** *number*
19. exit
20. **monitor parameters**
21. **flows** *number*
22. **interval duration** *number*
23. **history** *number*
24. **timeout** *number*
25. **exit**
26. **react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**}
27. **action** {**snmp** | **syslog**}
28. **alarm severity** {**alert** | **critical** | **emergency**| **error** | **info**}
29. **alarm type** {**discrete**| **grouped**{**count** *number* | **percent** *number*}}
30. **threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*}
31. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	インターフェイスを指定し、インターフェイス コンフィギュレーション モードを開始します。 IPv6 インターフェイスを指定できます。
ステップ 4	**service-policy type performance-monitor inline** {**input** | **output**} **Example:**	インターフェイスまたは仮想回線（VC）のサービス ポリシーとして使用されるポリシー マップを入力のインターフェイスまたはVC（あるいは出力のインターフェイスまたは VC）に付加します。

	Command or Action	Purpose
	Example: `Device(config-if)# service-policy type performance-monitor inline input`	• **input**：指定されたポリシーマップを入力インターフェイスまたは入力VCに対応付けます。 • **output**：指定されたポリシーマップを出力インターフェイスまたは出力VCに対応付けます。
ステップ 5	**match** {*access-group* {*access-group* | **name** *access-group-name*} | **any** | **class-map***class-map-name* | **cos** *cos-value* | **destination-address mac** address | **discard-class** *class-number* | **dscp** *dscp-value* | **flow** {**direction** | **sampler**} | **fr-de** | **fr-dlci** *dlci-number* | **input-interface** *interface-name* | **ip** {**rtp** *starting-port-number port-range* | **precedence** | **dscp**} | **mpls experimental topmost** *number* | **not** *match-criterion*| **packet length** {**max** *maximum-length-value* [**min** *minimum-length-value*] | **min** *minimum-length-value* [**max** *maximum-length-value*]} | **precedence** {*precedence-criteria1* | *precedence-criteria2* | *precedence-criteria3* | *precedence-criteria4*} | **protocol** *protocol-name* | **qos-group** *qos-group-value* | **source-address** *mac address-destination*| **vlan** {*vlan-id* | *vlan-range* | *vlan-combination*}} **Example:** `Device(config-if-spolicy-inline)# match any`	分類基準を指定します。 詳細および例については、『*Cisco Media Monitoring Command Reference*』を参照してください。
ステップ 6	**flow monitor** {*monitor-name*| **inline**} **Example:** `Device(config-if-spolicy-inline)# flow monitor inline`	フロー ポリシーと関連付ける既存のフロー モニタを指定します。既存のフローモニターを使用しない場合は、**inline** オプションを使用して新しいフローモニターを設定できます。 必要な場合は、**inline** オプションを使用してフローレコードおよびフローエクスポータを指定することもできます。
ステップ 7	{r ecord-name | | } **record default-rtp default-tcp** **Example:** `Device(config-spolicy-inline-flowmon)# record default-tcp`	（任意）既存のフローモニターを使用せず、代わりに **inline** オプションを使用する場合は、このコマンドを使用してフローレコードを設定します。
ステップ 8	**exporter** *exporter-name* **Example:** `Device(config-spolicy-inline-flowmon)# exporter exporter-4`	（任意）既存のフローモニターを使用せず、代わりに **inline** オプションを使用する場合は、このコマンドを使用してフローエクスポータを設定します。

	Command or Action	Purpose
ステップ 9	**exit** **Example:** `Device(config-spolicy-inline-flowmon)# exit`	サービス ポリシー インライン コンフィギュレーション モードに戻ります。
ステップ 10	**monitor metric ip-cbr** **Example:** `Device(config-if-spolicy-inline)# monitor metric ip-cbr`	IP-CBR モニタ メトリック コンフィギュレーション モードを開始します。
ステップ 11	**rate layer3** {*byte-rate* {**bps** | **kbps** | **mbps** | **gbps**} | **packet**} **Example:** `Device(config-spolicy-inline-mipcbr)# rate layer3 248 mbps`	メトリック モニタリング レートを指定します。 • *byte-rate*：データレート（単位：Bps、kBps、mBps、または gBps）。指定できる範囲は 1 ～ 65535 です。 • **packet**：パケットレート（単位：pps）。
ステップ 12	**exit** **Example:** `Device(config-spolicy-inline-mipcbr)# exit`	サービス ポリシー インライン コンフィギュレーション モードに戻ります。
ステップ 13	**monitor metric rtp** **Example:** `Device(config-if-spolicy-inline)# monitor metric rtp`	RTP モニタ メトリック コンフィギュレーション モードを開始します。
ステップ 14	**clock-rate** {*type-number*| *type-name*} *rate* **Example:** `Device(config-spolicy-inline-mrtp)# clock-rate 8 9600`	RTP ビデオ モニタリング メトリックのサンプリングに使用するクロック レートを指定します。 クロックタイプの番号と名前の詳細については、『*Cisco Media Monitoring Command Reference*』を参照してください。 *rate* の範囲は 1 ～ 192 kHz です。
ステップ 15	**max-dropout** *number* **Example:** `Device(config-spolicy-inline-mrtp)# max-dropout 2`	Specifies the maximum number of dropouts allowed when sampling RTP video-monitoring metrics.
ステップ 16	**max-reorder** *number* **Example:**	RTP ビデオ モニタリング メトリックのサンプリング時に許可される順序変更の最大数を指定します。

	Command or Action	Purpose
	`Device(config-spolicy-inline-mrtp)# max-reorder 4`	
ステップ 17	**min-sequential** *number* **Example:** `Device(config-spolicy-inline-mrtp)# min-sequential 2`	ストリームを RTP フローとして識別するために必要な連続パケットの最小数を指定します。
ステップ 18	**ssrc maximum** *number* **Example:** `Device(config-spolicy-inline-mrtp)# ssrc maximum 20`	同じフロー内でモニタできる SSRC の最大数を指定します。フローは、プロトコル、送信元と宛先のアドレス、および送信元と宛先のポートによって定義されます。
ステップ 19	exit **Example:** `Device(config-spolicy-inline-mrtp)# exit`	サービス ポリシー インライン コンフィギュレーション モードに戻ります。
ステップ 20	**monitor parameters** **Example:** `Device(config-if-spolicy-inline)# monitor parameters`	モニタ パラメータ コンフィギュレーション モードを開始します。
ステップ 21	**flows** *number* **Example:** `Device(config-spolicy-inline-mparam)# flows 40`	各モニタ キャッシュのフローの最大数を指定します。
ステップ 22	**interval duration** *number* **Example:** `Device(config-spolicy-inline-mparam)# interval duration 40`	モニタリング メトリックを収集するためのインターバルの長さ（秒）を指定します。
ステップ 23	**history** *number* **Example:** `Device(config-spolicy-inline-mparam)# history 4`	収集されたビデオ モニタリング メトリックについて表示する履歴インターバルの数を指定します。
ステップ 24	**timeout** *number* **Example:** `Device(config-spolicy-inline-mparam)# timeout 20`	停止したフローがデータベースから削除されるまでのインターバルの数を指定します。

	Command or Action	Purpose
ステップ 25	**exit** **Example:** `Device(config-spolicy-inline-mparam)# exit`	サービス ポリシー インライン コンフィギュレーション モードに戻ります。
ステップ 26	**react** *ID* {**media-stop** | **mrv** | **rtp-jitter-average** | **transport-packets-lost-rate**} **Example:** `Device(config-if-spolicy-inline)# react 6 rtp-jitter-average`	次のメトリックのしきい値を超えた場合の反応を指定できるモードを開始します。 • *ID* ：反応設定の ID。有効値の範囲は 1 ～ 65535 です。 • **media-stop** ：フローのトラフィックが検出されません。 • **mrv** ：実際のレートと予想レートの差を予想レートで割ることによって算出されるレート。 • **rtp-jitter-average**：平均ジッター。 • **transport-packets-lost-rate** ：損失パケット数を予想パケット数で割ることによって算出されるレート。
ステップ 27	**action** {**snmp** | **syslog**} **Example:** `Device(config-spolicy-inline-react)# action syslog`	しきい値を超えた場合の報告方法を指定します。
ステップ 28	**alarm severity** {**alert** | **critical** | **emergency**| **error** | **info**} **Example:** `Device(config-spolicy-inline-react)# alarm severity critical`	報告されるアラームのレベルを指定します。
ステップ 29	**alarm type** {**discrete**| **grouped**{**count** *number* | **percent** *number*}} **Example:** `Device(config-pspolicy-inline-react)# alarm severity critical`	報告が必要なアラームと見なされるレベルのタイプを指定します。
ステップ 30	**threshold value** {**ge** *number* | **gt** *number* | **le** *number* | **lt** *number* | **range** *rng-start rng-end*} **Example:** `Device(config-spolicy-inline-react)# threshold value ge 20`	報告が必要なアラームと見なされるしきい値のタイプを指定します。 値が指定されておらず、アプリケーション名がキーフィールドとして設定されている場合は、デフォルトのマップで検出されるしきい値が使用されます。

	Command or Action	Purpose
		値が指定されておらず、また、アプリケーション名がキー フィールドとして設定されていない場合、しきい値にはデフォルト値が使用されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されているが、しきい値が指定されているのは1つの反応設定のみである場合は、設定されている反応の値が優先され、残りのしきい値は無視されます。 同じポリシーとクラスに対して複数の反応コマンドが設定されており、しきい値が1つも設定されていない場合は、最も小さい反応 ID が割り当てられている設定にデフォルトのしきい値が適用されます。
ステップ 31	**end** **Example:** `Device(config-spolicy-inline-react)# end`	現在のコンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

What to do next

サービスポリシーの構成とステータスを確認するには、**show performance monitor status** コマンドおよび **show performance monitor history** コマンドを使用します。

Cisco Performance Monitor のデータ収集の確認

Cisco Performance Monitor がデータを収集していることを確認するには、次のオプション作業を実行します。

Note フローが相互に関連付けられるので、同じポリシーが同じ入力インターフェイスと出力インターフェイスに適用されている場合に **show** コマンドを実行すると、その入力インターフェイスと出力インターフェイスについて単一のフローが表示され、フローのインターフェイス名と方向は表示されません。

データが収集されていない場合は、このセクションの残りの作業を完了させます。

Before you begin

フロー モニタ キャッシュ内のフローを表示するには、オリジナルのフロー レコードで定義された基準に適合するトラフィックを受信するインターフェイスに、入力フローモニタを適用する必要があります。

この場合、*filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

SUMMARY STEPS

1. **enable**
2. **show policy-map type performance-monitor** [**interface** *interface-name*][**class** *class-name*][**input | output**]
3. **show performance monitor status** [**interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]
4. **show performance monitor history** [**interval**{**all**| *number*[**start** *number*]} | **interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show policy-map type performance-monitor [interface *interface-name*][class *class-name*][input | output]

このコマンドによって表示されるフィールドの説明については、『*Cisco Media Monitoring Command Reference*』を参照してください。

次に、あるフロー ポリシーの出力例を示します。

Example:

```
Policy Map type performance-monitor PM-POLICY-4
  Class PM-CLASS-4
    flow monitor PM-MONITOR-4
      record PM-RECORD-4
      exporter PM-EXPORTER-4
    monitor parameters
      interval duration 30
      timeout 10
      history 10
      flows 8000
    monitor metric rtp
      min-sequential 5
      max-dropout 5
      max-reorder 5
      clock-rate default 90000
      ssrc maximum 5
```

Table 43: policy-map type performance-monitor のフィールドの説明

フィールド	説明
Policy Map type performance-monitor	Cisco Performance Monitor のフロー ポリシーの名前。
flow monitor	Cisco Performance Monitor のフロー モニタの名前。
record	Cisco Performance Monitor のフロー レコードの名前。
exporter	Cisco Performance Monitor のフロー エクスポータの名前。
monitor parameter	フロー ポリシーのパラメータ。
interval duration	ポリシーで設定されている収集間隔の時間。
timeout	ポリシーで設定されているデータ収集時の応答待機時間。
history	ポリシーで設定されている収集履歴の保持数。
flows	ポリシーで設定されているフローの収集数。
monitor metric rtp	フロー ポリシーの RTP メトリック。
min-sequential	RTP フローの分類に使用される連続パケットについて設定されている最小数。
max-dropout	現在のパケットよりもシーケンス番号が小さいものとして無視されるパケットについて設定されている最大数。
max-reorder	現在のパケットよりもシーケンス番号が大きいものとして無視されるパケットについて設定されている最大数。
clock-rate default	パケット到着遅延の計算に使用される RTP パケット タイムスタンプ クロック用に設定されているクロック レート。
ssrc maximum	同じフロー内でモニタできる SSRC について設定されている最大数。フローは、プロトコル、送信元と宛先のアドレス、および送信元と宛先のポートによって定義されます。範囲は 1 ～ 50 です。

ステップ 3 **show performance monitor status** [**interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

この場合、*filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

このコマンドは、指定された数の最新のインターバルの累積統計情報を表示します。インターバルの数は、**history** コマンドを使用して設定します。このコマンドのデフォルト設定は、最新 10 の収集インターバルです。収集インターバルの長さは、 **interval duration** コマンドを使用して指定します。

他のインターバルの統計情報を表示するには、次のステップの説明に従って、**show performance monitor history** コマンドを使用します。これらのコマンドの詳細については、『*Cisco Media Monitoring Command Reference*』を参照してください。

ステップ 4 **show performance monitor history** [**interval**{**all**| *number*[**start** *number*]} | **interface** *interface name*[*filter*] | **policy** *policy-map-name* **class** *class-map-name*[*filter*]} | *filter*]

この場合、*filter* = {**ip** {*source-addr source-prefix* | **any**} {*dst-addr dst-prefix* | **any**} | {**tcp** | **udp**} {*source-addr source-prefix* | **any**} {[**eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**} | {{*dst-addr dst-prefix* | **any**} **eq**| **lt**| **gt** *number*| **range** *min max*| **ssrc** {*ssrc-number* | **any**}}

このコマンドは、最新のものを含めて、任意またはすべてのインターバルで Cisco Performance Monitor によって収集された統計情報を表示します。収集インターバルの長さは、 **interval duration** コマンドを使用して指定します。

このコマンドの詳細については、『*Cisco Media Monitoring Command Reference*』を参照してください。

次の例は、**show performance monitor history** コマンド**:**のサンプル出力を示しています。

Note 同じポリシーが同じ入力インターフェイスと出力インターフェイスに適用されている場合、その入力インターフェイスと出力インターフェイスについて単一のフローが表示され、フローのインターフェイス名と方向は表示されません。

Example:

```
Codes: *   - field is not configurable under flow record
       NA  - field is not applicable for configured parameters
Match: ipv4 source address = 21.21.21.1, ipv4 destination address = 1.1.1.1,
transport source-port = 10240, transport destination-port = 80, ip protocol = 6,
 Policy: RTP_POL, Class: RTP_CLASS

 start time                                    14:57:34
                                               ============
 *history bucket number                       : 1
 routing forwarding-status                    : Unknown
 transport packets expected counter           : NA
 transport packets lost counter               : NA
 transport round-trip-time           (msec)   : 4
 transport round-trip-time sum       (msec)   : 8
 transport round-trip-time samples            : 2
 transport event packet-loss counter          : 0
 interface input                              : Null
 interface output                             : Null
 counter bytes                                : 8490
 counter packets                              : 180
 counter bytes rate                           : 94
 counter client bytes                        : 80
 counter server bytes                        : 200
 counter client packets                      : 6
 counter server packets                      : 6
 transport tcp window-size minimum           : 1000
 transport tcp window-size maximum           : 2000
 transport tcp window-size average           : 1500
 transport tcp maximum-segment-size          : 0
 application media bytes counter              : 1270
 application media bytes rate                 : 14
 application media packets counter            : 180
 application media event                      : Stop
 monitor event                                : false
```

```
[data set,id=257] Global session ID|Multi-party session ID|
[data] 11                         |22
```

Table 44: show performance monitor status and show performance-monitor history のフィールドの説明

フィールド	説明
history bucket number	収集された履歴データのバケットの数。

フィールド	説明
routing forwarding-status reason	

フィールド	説明
	2 桁の最上位ビットがステータスを表し、残りの 6 ビットが理由コードを表す 8 ビットを使用して、転送状態が検出されます。 ステータスは、Unknown（00）、Forwarded（10）、Dropped（10）、Consumed（11）のいずれかです。 次に、各ステータス カテゴリの転送ステータス値を示します。 **Unknown** • 0 **Forwarded** • Unknown 64 • Forwarded Fragmented 65 • Forwarded not Fragmented 66 **Dropped** • Unknown 128 • Drop ACL Deny 129 • Drop ACL drop 130 • Drop Unroutable 131 • Drop Adjacency 132 • Drop Fragmentation & DF set 133 • Drop Bad header checksum 134 • Drop Bad total Length 135 • Drop Bad Header Length 136 • Drop bad TTL 137 • Drop Policer 138 • Drop WRED 139 • Drop RPF 140 • Drop For us 141 • Drop Bad output interface 142 • Drop Hardware 143 **Consumed** • Unknown 192 • Terminate Punt Adjacency 193

フィールド	説明
	• Terminate Incomplete Adjacency 194 • Terminate For us 195
transport packets expected counter	予想パケット数。
transport packets lost counter	パケット損失数。
transport round-trip-time (msec)	ラウンド トリップを完了させるためにかかった時間（ミリ秒）。
transport round-trip-time sum (msec)	すべてのサンプリングのラウンド トリップを完了させるためにかかった合計時間（ミリ秒）。
transport round-trip-time samples	ラウンド トリップ時間の計算に使用されたサンプルの合計数。
transport event packet-loss counter	損失イベントの数（損失パケットの連続セットの数）。
interface input	着信インターフェイス インデックス。
interface output	発信インターフェイス インデックス。
counter bytes	すべてのフローで収集されたバイトの合計数。
counter packets	すべてのフローで送信された IP パケットの合計数。
counter bytes rate	すべてのフローのモニタリング インターバルでモニタリング システムによって 1 分間に処理されたパケットまたはビット（設定によって異なる）の平均数。
counter client bytes	クライアントの送信バイト数。
counter server bytes	サーバの送信バイト数。
counter client packets	クライアントの送信パケット数。
counter servers packets	サーバの送信パケット数。
transport tcp window-size-maximum	TCP ウィンドウの最大サイズ。
transport tcp window-size-minimum	TCP ウィンドウの最小サイズ。
transport tcp window-size-average	TCP ウィンドウの平均サイズ。
transport tcp maximum-segment-size	最大 TCP セグメント サイズ。
application media bytes counter	特定のメディア ストリームでメディア アプリケーションから受信された IP バイトの数。
application media bytes rate	モニタリング インターバルにおけるすべてのフローの平均メディア ビット レート（bps）。

フィールド	説明
application media packets counter	特定のメディア ストリームでメディア アプリケーションから受信された IP パケットの数。
application media event	ビット 1 は使用されません。ビット 2 は、メディア アプリケーション パケットが検出されなかったこと、つまり、メディア停止イベントが発生したことを示します。
monitor event	ビット 1 は、フローの反応ステートメントで指定されているいずれかのしきい値をモニタリング インターバルで少なくとも 1 回超えることがあったこと示します。ビット 2 は、測定の信頼性の喪失が発生したことを示します。

オプションテーブルを表示します。

次の **show** コマンドを使用して、さまざまなオプションテーブルに含まれるマッピングを表示できます。

SUMMARY STEPS

1. **enable**
2. **show metadata** {**application attributes** | **application table** | **exporter stats** | **interface table** | **metadata version table** | **sampler table** | **vrf table**}

DETAILED STEPS

	Command or Action	Purpose
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	**show metadata** {**application attributes** | **application table** | **exporter stats** | **interface table** | **metadata version table** | **sampler table** | **vrf table**}
	Example:
		次の **show metadata application table** コマンドを使用して、アプリケーション ID とアプリケーション名のマッピングを表示する例を示します。
		`ID Name Vendor Version`
		`100673296 webex-audio - -`
		`100673297 webex-video - -`

Catalyst 6500 プラットフォームに固有の情報の表示

Feature Manager および Catalyst 6500 プラットフォームに固有のその他の機能の情報を表示またはクリアするには、次の任意のタスクを実行します。

SUMMARY STEPS

1. **enable**
2. **clear fm performance-monitor counters**
3. **debug fm performance-monitor** {**all** | **dynamic** | **event** | **unusual** | **verbose** | **vmr**}
4. **platform performance-monitor rate-limit pps** *number*
5. **show platform software feature-manager performance-monitor** {**all** | **counters** | **interface** *interface-type interface-number* | **rdt-indices** }
6. **show platform software feature-manager tcam dynamic performance-monitor** {**handle ip** *ip-address* | **interface** *interface-type interface-number* }
7. **show platform hardware acl entry interface** *interface-type interface-number* **security** {**in** | **out** } {**ip** | **ipv6** } [**detail**]
8. **show platform software ccm interface** *interface-type interface-number* **security** {**interface** *interface-type interface-number* | **class-group** *class-group-ID* }

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 clear fm performance-monitor counters

clearfm performance-monitor counters コマンドは、機能モニターのパフォーマンス モニター コンポーネントのカウンタをクリアします。

Example:

```
Device# clear fm performance-monitor counters
Device#
```

ステップ 3 debug fm performance-monitor {all | dynamic | event | unusual | verbose | vmr}

このコマンドは、Feature Manager のパフォーマンス モニター コンポーネントのすべてのレベルのデバッグを有効にします。

Example:

```
Device# debug fm performance-monitor all
Device#
```

ステップ 4 **platform performance-monitor rate-limit pps** *number*

このコマンドは、機能モニターのパフォーマンス モニター コンポーネントのレート制限を設定します。

Example:

```
Device# platform performance-monitor rate-limit pps 2000
Device#
```

ステップ 5 **show platform software feature-manager performance-monitor** {**all** | **counters** | **interface** *interface-type interface-number* | **rdt-indices** }

このコマンドは、Feature Manager のパフォーマンス モニター コンポーネントに関する情報を表示します。

Example:

```
Device# show platform software feature-manager performance-monitor all
Device#

Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
--------------------------------------------------------------------------------
Feature: VM Ingress L3
================================================================================
DPort  - Destination Port  SPort  - Source Port       Pro    - Protocol
RFTCM  - R-Recirc. Flag    MRLCS  - M-Multicast Flag  Res    - VMR Result
       - F-Fragment flag           - R-Reflexive flag  Prec   - Drop Precedence
       - T-Trailing Fragments      - L-Layer 3 only    GrpId  - Qos Group Id
       - C-From CPU                - C-Capture Flag    Adj.   - Adj. Index
       - M-L2 Lookup Miss          - S-RPF suppress    Pid    - NF Profile Index
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V       224.0.0.0         0.0.0.0                 0          0      0         -----
0       -----
       M       240.0.0.0         0.0.0.0                  0          0      0         00000    0
        0
      PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                 0          0      0         -----
  0      -----
       M          0.0.0.0         0.0.0.0                  0          0      0         00000
0        0
      L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
      ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
  0
      PERMIT_RESULT

     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
      ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
```

```
  0
      PERMIT_RESULT

     3 V          0.0.0.0           0.0.0.0                   0           0       0          -----
  0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000
0         0
L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V          0.0.0.0           0.0.0.0                   0           0       0          -----
  0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000
0         0
      PERMIT_RESULT



Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Egress L3
==============================================================================
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V          0.0.0.0           0.0.0.0                   0           0       0          -----
  0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000
0         0
PERMIT_RESULT

     2 V          0.0.0.0           0.0.0.0                   0           0       0          -----
  0       -----
       M           0.0.0.0          0.0.0.0                    0           0       0          00000
0         0
      L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V          0.0.0.0           10.10.10.0             0           0       17         -----         0
      -----
       M           0.0.0.0           255.255.255 0       0           0       255       00000      0
  0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     2 V          0.0.0.0           10.10.20.0             0           0       17         -----         0
      -----
       M           0.0.0.0           255.255.255 0       0           0       255       00000      0
  0
      PERMIT_RESULT Adjacency: 0x5512D8F4
```

```
     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000
0        0
       L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000
0        0
     PERMIT_RESULT Adjacency: 0x5512D8F4


Adjacency: 0x5512D8F4
        FeatureId: 0x84  AdjId: 0xFFFFFFFF Flags: RecirculationAdj|

        Cause: 0x0 Priority: 0xC Device#

Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Ingress L3
==============================================================================
DPort  - Destination Port  SPort  - Source Port       Pro    - Protocol
RFTCM  - R-Recirc. Flag    MRLCS  - M-Multicast Flag  Res    - VMR Result
       - F-Fragment flag           - R-Reflexive flag  Prec   - Drop Precedence
       - T-Trailing Fragments      - L-Layer 3 only    GrpId  - Qos Group Id
       - C-From CPU                - C-Capture Flag    Adj.   - Adj. Index
       - M-L2 Lookup Miss          - S-RPF suppress    Pid    - NF Profile Index
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V       224.0.0.0        0.0.0.0                 0          0      0         -----
0       -----
       M       240.0.0.0        0.0.0.0                   0          0      0         00000    0
        0
      PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0       -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000
0        0
      L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
      ---C-
       M          0.0.0.0          255.255.255 0      0          0      255      00000     0
  0
      PERMIT_RESULT
```

```
     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
      ---C-
       M          0.0.0.0          255.255.255.255 0      0          0      255      00000     0
  0
      PERMIT_RESULT

     3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0          0.0.0.0                   0          0      0         00000
0        0
L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0          0.0.0.0                   0          0      0         00000
0        0
      PERMIT_RESULT


Interface: FastEthernet2/3
Policy: video-flow-test          Group ID: A0000001
------------------------------------------------------------------------------
Feature: VM Egress L3
==============================================================================
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0          0.0.0.0                   0          0      0         00000
0        0
PERMIT_RESULT

     2 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0          0.0.0.0                   0          0      0         00000
0        0
      L3_DENY_RESULT

+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+-----------------+-------+-------+----+---------+------+---------+-----+--------+


     1 V         0.0.0.0          10.10.10.0            0          0      17        -----        0
      -----
       M          0.0.0.0          255.255.255.255 0      0          0      255      00000     0
  0
      PERMIT_RESULT Adjacency: 0x5512D8F4

     2 V         0.0.0.0          10.10.20.0            0          0      17        -----        0
      -----
```

```
      M          0.0.0.0          255.255.255 0      0          0      255      00000     0
  0
     PERMIT_RESULT Adjacency: 0x5512D8F4

    3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000
0        0
       L3_DENY_RESULT

+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+
| Indx | T |   Dest Ip Addr | Source Ip Addr | DPort | SPort | Pro | RFTCM | Prec | MRLCS | Pid |
Stats Id|
+-----+--+----------------+------------------+-------+-------+----+---------+------+---------+-----+--------+


    3 V         0.0.0.0          0.0.0.0                  0          0      0         -----
  0      -----
       M          0.0.0.0         0.0.0.0                   0          0      0         00000
0        0
     PERMIT_RESULT Adjacency: 0x5512D8F4


Adjacency: 0x5512D8F4
        FeatureId: 0x84  AdjId: 0xFFFFFFFF Flags: RecirculationAdj|

        Cause: 0x0 Priority: 0xC
```

ステップ 6 **show platform software feature-manager tcam dynamic performance-monitor** {**handle ip** *ip-address* | **interface** *interface-type interface-number* }

このコマンドは、特定のホストの動的ポリシーと静的ポリシーに関する情報を表示します。

Example:

```
Device# show platform software feature-manager tcam dynamic performance-monitor handle ip 10.1.1.0
-------------------------------------------------------------------------
HANDLE                      Feature ID   No of entries     MD5
-------------------------------------------------------------------------
10.1.1.0             VM Ingress L3                        2
```

ステップ 7 **show platform hardware acl entry interface** *interface-type interface-number* **security** {**in** | **out** } {**ip** | **ipv6** } [**detail**]

このコマンドは、インターフェイス上のIPの受信アクセスコントロールリスト（ACL）エントリを表示します。

Example:

```
Device# show platform hardware acl entry interface fastEthernet 1/1 security in ip detail

mls_if_index:2000400A dir:0 feature:0 proto:0


pass#0 features
UAPRSF: U-urg, A-ack, P-psh, R-rst, S-syn, F-fin
MLGFI: M-mpls_plus_ip_pkt, L-L4_hdr_vld, G-gpid_present,F-global_fmt_match, I-ife/ofe
's' means set; 'u' means unset; '-' means don't care
-------------------------------------------------------------------------------------------------
INDEX  LABEL FS ACOS    AS     IP_SA      SRC_PORT       IP_DA      DST_PORT        F FF
L4PROT
```

```
TCP-F:UAPRSF MLGFI OtherL4OPs                                    RSLT                CNT
-------------------------------------------------------------------------------------------

fno:0

tcam:B, bank:0, prot:0    Aces

I  V  16375   2049  0    0     0         0.0.0.0              -         0.0.0.0              - 0
  0      0      -       -----          -
0x0000000800000038            10331192<-
I  M  16375 0x1FFF  0 0x00 0x000         0.0.0.0              -         0.0.0.0              - 0
  0    0x0
```

ステップ 8 **show platform software ccm interface** *interface-type interface-number* **security** {**interface** *interface-type interface-number* | **class-group** *class-group-ID* }

このコマンドは、インターフェイス上の TCAM（Ternary Content Addressable Memory）Cisco CallManager（CCM）エントリに関する情報を表示します。

Example:

```
Device# show platform software ccm interface fastEthernet 2/3 in

 Target-Class : id 0xA0000000, dir CCM_INPUT, if_type 1, if_info 0x14823998

 Class-Group List: 0xA0000001
b1-cs217#

b1-cs217#sh platform software ccm interface fastEthernet 2/3 out

 Target-Class : id 0xA0000002, dir CCM_OUTPUT, if_type 1, if_info 0x14823998

 Class-Group List: 0xA0000001
```

このコマンドは、クラスグループの TCAM（Ternary Content Addressable Memory）Cisco CallManager（CCM）エントリに関する情報を表示します。

Example:

```
Device# show platform software ccm class-group A0000001
 Class-group    : video-flow-test, id 0xA0000001
 Target input   : 0xA0000000
 Target Output  : 0xA0000002
        Class   : video-flow, id 0xA98681, type 1
                Filter          : type MATCH_NUMBERED_ACCESS_GROUP, id 0xF0000002
                Filter params   : ACL Index: 101 Linktype: 7

                Feature         : PERFORMANCE_MONITOR
                Params          :
                  Feature Object : 0x54224218
                    Name           :
                    Meter context  : 0x54264440
                    Sibling        : 0x0
                    Dynamic        : FALSE
                  Feature Object : 0x54221170
                    Name           :
                    Meter context  : 0x54263858
                    Sibling        : 0x0
                    Dynamic        : FALSE
                Intf List       : 0xA0000000  0xA0000002
        Class   : class-default, id 0xADA3F1, type 39
                Filter          : type MATCH_ANY, id 0xF0000003
```

```
Filter params   : any

Feature         : FEATURE_EMPTY
Params          :
  Feature Object : 0x1741629C
    Name          :
    Meter context  : 0x0
    Sibling        : 0x0
    Dynamic        : FALSE
Intf List       : 0xA0000000  0xA0000002
```

Performance Monitor のキャッシュとクライアントの表示

Cisco Performance Monitor のキャッシュとクライアントを表示するには、次のオプション作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show performance monitor cache** [**policy** *policy-map-name* **class** *class-map-name*][**interface** *interface name*]
3. **show performance monitor clients detail all**

DETAILED STEPS

ステップ 1 **enable**

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 **show performance monitor cache** [**policy** *policy-map-name* **class** *class-map-name*][**interface** *interface name*]

Example:

```
MMON Metering Layer Stats:
  static pkt cnt: 3049
  static cce sb cnt: 57
  dynamic pkt cnt: 0
  Cache type:                              Permanent
  Cache size:                                   2000
  Current entries:                                 8
  High Watermark:                                  9
  Flows added:                                     9
  Updates sent            (  1800 secs)            0
IPV4 SRC ADDR    IPV4 DST ADDR    IP PROT  TRNS SRC PORT  TRNS DST PORT
ipv4 ttl ipv4 ttl min ipv4 ttl max  ipv4 dscp bytes long perm pktslong perm  user space vm
==============================================================================================
10.1.1.1         10.1.2.3              17           4000           1967
```

```
0             0             0  0x00                          80
1 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           6000           1967
0             0             0  0x00                          80
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           4000           2000
0             0             0  0x00                          44
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3               6           6000           3000
0             0             0  0x00                          84
2  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           1967           6001
0             0             0  0x00                          36
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           1967           4001
0             0             0  0x00                          36
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3               6           3001           6001
0             0             0  0x00                         124
3  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
10.1.1.1         10.1.2.3              17           2001           4001
0             0             0  0x00                          44
```

```
1  0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x00000000
```

ステップ 3 show performance monitor clients detail all

Example:

```
Client name for ID 1 : Mediatrace-131419052
   Type: Mediatrace
   Age: 443 seconds
   Monitor Object: _MMON_DYN_-class-map-69
        Flow spec: (dvmc-acl#47) 10.10.130.2 1000 10.10.132.2 2000 17
        monitor parameters
                interval duration 60
                timeout 2
                history 1
                flows 100
        monitor metric rtp
                min-sequential 10
                max-dropout 5
                max-reorder 5
                clock-rate 112 90000
                clock-rate default 90000
                ssrc maximum 20
        monitor metric ip-cbr
                rate layer3 packet 20
        Flow record: dvmc_fnf_fdef_47
                Key fields:
                        ipv4 source address
                        ipv4 destination address
                        transport source-port
                        transport destination-port
                        ip protocol
                Non-key fields:
                        monitor event
                        application media event
                        routing forwarding-status
                        ip dscp
                        ip ttl
                        counter bytes rate
                        application media bytes rate
                        transport rtp jitter mean
                        transport packets lost counter
                        transport packets expected counter
                        transport event packet-loss counter
                        transport packets lost rate
                        timestamp interval
                        counter packets dropped
                        counter bytes
                        counter packets
                        application media bytes counter
                        application media packets counter
        Monitor point: _MMON_DYN_-policy-map-70 GigabitEthernet0/3 output
        Classification Statistic:
                matched packet: 545790
                matched byte: 64403220
```

Cisco Performance Monitor クラスのクロック レートの表示

1 つ以上のクラスのクロック レートを表示するには、次のオプション作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show performance monitor clock rate** [**policy** *policy-map-name* **class** *class-map-name*]

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show performance monitor clock rate [policy *policy-map-name* class *class-map-name*]

クラス名を指定しない場合は、すべてのチャネルの情報が表示されます。

Example:

```
Device# show performance monitor clock rate policy all-apps class telepresence-CS4
Load for five secs: 6%/2%; one minute: 5%; five minutes: 5% Time source is NTP, 17:41:35.508 EST
Wed Feb 16 2011
RTP clock rate for Policy: all-apps, Class: telepresence-CS4
     Payload type     Clock rate(Hz)
     pcmu     (0  )     8000
     gsm      (3  )     8000
     g723     (4  )     8000
     dvi4     (5  )     8000
     dvi4-2   (6  )     16000
     lpc      (7  )     8000
     pcma     (8  )     8000
     g722     (9  )     8000
     l16-2    (10 )     44100
     l16      (11 )     44100
     qcelp    (12 )     8000
     cn       (13 )     8000
     mpa      (14 )     90000
     g728     (15 )     8000
     dvi4-3   (16 )     11025
     dvi4-4   (17 )     22050
     g729     (18 )     8000
     celb     (25 )     90000
     jpeg     (26 )     90000
     nv       (28 )     90000
     h261     (31 )     90000
     mpv      (32 )     90000
     mp2t     (33 )     90000
     h263     (34 )     90000
              (96 )     48000
```

```
          (112)      90000
default              90000
```

フロー モニタの現在のステータスの表示

フロー モニタの現在のステータスを表示するには、次のオプション作業を実行します。

Before you begin

フロー モニタ キャッシュ内のフローを表示するには、オリジナルのフロー レコードで定義された基準に適合するトラフィックを受信するインターフェイスに、入力フローモニタを適用する必要があります。

SUMMARY STEPS

1. **enable**
2. **show flow monitor type performance-monitor**

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show flow monitor type performance-monitor

show flow monitor type performance-monitor コマンドでは、指定するフローレコードの現在のステータスを表示します。

Example:

```
Device# show flow monitor type performance-monitor
Flow Monitor type performance-monitor monitor-4:
  Description:           User defined
  Flow Record:           record-4
  Flow Exporter:         exporter-4
  No. of Inactive Users: 0
  No. of Active Users:   0
```

フロー モニタの設定の確認

入力したコンフィギュレーション コマンドを確認するには、次のオプション作業を実行します。

Before you begin

フロー モニタ キャッシュ内のフローを表示するには、オリジナルのフロー レコードで定義された基準に適合するトラフィックを受信するインターフェイスに、入力フローモニタを適用する必要があります。

SUMMARY STEPS

1. **enable**
2. **show running-config flow monitor**

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show running-config flow monitor

show running-config flow monitor コマンドでは、指定するフロー レコードのコンフィギュレーション コマンドを表示します。

Example:

```
Device# show running-config flow monitor
Current configuration:
!
flow monitor FLOW-MONITOR-1
 description Used for basic IPv4 traffic analysis
 record netflow ipv4 original-input
!
!
flow monitor FLOW-MONITOR-2
 description Used for basic IPv6 traffic analysis
 record netflow ipv6 original-input
!
```

インターフェイスで Cisco IOS Flexible NetFlow および Cisco Performance Monitor が有効になっていることの確認

インターフェイスで Flexible NetFlow および Cisco Performance Monitor が有効になっていることを確認するには、次のオプション作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show flow interface** *type number*

DETAILED STEPS

ステップ 1 **enable**

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Router> enable
Router#
```

ステップ 2 **show flow interface** *type number*

show flow interface コマンドを使用して、インターフェイスで Flexible NetFlow および Cisco Performance Monitor が有効になっていることを確認します。

Example:

```
Router# show flow interface ethernet 0/0
Interface Ethernet0/0
  FNF:  monitor:          FLOW-MONITOR-1
        direction:        Input
        traffic(ip):      on
  FNF:  monitor:          FLOW-MONITOR-2
        direction:        Input
        traffic(ipv6):    on
```

フロー モニタ キャッシュの表示

フロー モニタ キャッシュのデータを表示するには、次のオプション作業を実行します。

Before you begin

フロー モニタ キャッシュ内のフロー データを表示するには、NetFlow original レコードで定義された基準に適合するトラフィックを受信するインターフェイスに、入力フローモニタを適用する必要があります。

SUMMARY STEPS

1. **enable**
2. **show flow monitor name** *monitor-name* **cache format record**

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show flow monitor name *monitor-name* cache format record

show flow monitor name *monitor-name* **cache format record** コマンド文字列では、フローモニターのステータス、統計情報、およびキャッシュ内のフローデータを表示します。

Example:

```
Device# show flow monitor name FLOW-MONITOR-1 cache format record
Cache type:                               Normal
  Cache size:                               4096
  Current entries:                             8
  High Watermark:                              8
  Flows added:                                24
  Flows aged:                                 16
    - Active timeout   (  1800 secs)           0
    - Inactive timeout (    15 secs)          16
    - Event aged                               0
    - Watermark aged                           0
    - Emergency aged                           0
IPV4 SOURCE ADDRESS:       10.251.10.1
IPV4 DESTINATION ADDRESS:  172.16.10.2
TRNS SOURCE PORT:          0
TRNS DESTINATION PORT:     2048
INTERFACE INPUT:           Et0/0
FLOW SAMPLER ID:           0
IP TOS:                    0x00
IP PROTOCOL:               1
ip source as:              0
ip destination as:         0
ipv4 next hop address:     172.16.7.2
ipv4 source mask:          /0
ipv4 destination mask:     /24
tcp flags:                 0x00
interface output:          Et1/0
counter bytes:             733500
counter packets:           489
timestamp first:           720892
timestamp last:            975032
.
.
.
IPV4 SOURCE ADDRESS:       172.16.6.1
```

```
IPV4 DESTINATION ADDRESS:  224.0.0.9
TRNS SOURCE PORT:          520
TRNS DESTINATION PORT:     520
INTERFACE INPUT:           Et0/0
FLOW SAMPLER ID:           0
IP TOS:                    0xC0
IP PROTOCOL:               17
ip source as:              0
ip destination as:         0
ipv4 next hop address:     0.0.0.0
ipv4 source mask:          /24
ipv4 destination mask:     /0
tcp flags:                 0x00
interface output:          Null
counter bytes:             52
counter packets:           1
timestamp first:           973804
timestamp last:            973804
Device# show flow monitor name FLOW-MONITOR-2 cache format record
Cache type:                               Normal
  Cache size:                               4096
  Current entries:                             6
  High Watermark:                              8
  Flows added:                              1048
  Flows aged:                               1042
    - Active timeout   (  1800 secs)          11
    - Inactive timeout (    15 secs)        1031
    - Event aged                                0
    - Watermark aged                            0
    - Emergency aged                            0
IPV6 FLOW LABEL:           0
IPV6 EXTENSION MAP:        0x00000040
IPV6 SOURCE ADDRESS:       2001:DB8:1:ABCD::1
IPV6 DESTINATION ADDRESS:  2001:DB8:4:ABCD::2
TRNS SOURCE PORT:          3000
TRNS DESTINATION PORT:     55
INTERFACE INPUT:           Et0/0
FLOW DIRECTION:            Input
FLOW SAMPLER ID:           0
IP PROTOCOL:               17
IP TOS:                    0x00
ip source as:              0
ip destination as:         0
ipv6 next hop address:     ::
ipv6 source mask:          /48
ipv6 destination mask:     /0
tcp flags:                 0x00
interface output:          Null
counter bytes:             521192
counter packets:           9307
timestamp first:           9899684
timestamp last:            11660744
.
.
.
IPV6 FLOW LABEL:           0
IPV6 EXTENSION MAP:        0x00000000
IPV6 SOURCE ADDRESS:       FE80::A8AA:BBFF:FEBB:CC03
IPV6 DESTINATION ADDRESS:  FF02::9
TRNS SOURCE PORT:          521
TRNS DESTINATION PORT:     521
INTERFACE INPUT:           Et0/0
FLOW DIRECTION:            Input
FLOW SAMPLER ID:           0
```

```
IP PROTOCOL:                17
IP TOS:                     0xE0
ip source as:               0
ip destination as:          0
ipv6 next hop address:      ::
ipv6 source mask:           /10
ipv6 destination mask:      /0
tcp flags:                  0x00
interface output:           Null
counter bytes:              92
counter packets:            1
timestamp first:            11653832
timestamp last:             11653832
```

フロー エクスポータの現在のステータスの表示

フローエクスポータの現在のステータスを表示するには、次のオプション作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show flow exporter** [*exporter-name*]

DETAILED STEPS

ステップ 1 enable

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 show flow exporter [*exporter-name*]

show flow exporter コマンドでは、指定するフローエクスポータの現在のステータスを表示します。

Example:

```
Device# show flow exporter EXPORTER-1
Flow Exporter EXPORTER-1:
  Description:              Exports to Chicago datacenter
  Transport Configuration:
    Destination IP address: 172.16.10.2
    Source IP address:      172.16.7.1
    Transport Protocol:     UDP
    Destination Port:       65
    Source Port:            56041
    DSCP:                   0x0
    TTL:                    255
```

フロー エクスポータの設定の確認

フロー エクスポータを設定するために入力したコンフィギュレーション コマンドを確認するには、次のオプション作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show running-config flow exporter** *exporter-name*

DETAILED STEPS

ステップ 1 **enable**

enable コマンドを使用して、特権 EXEC モードを開始します（プロンプトが表示されたらパスワードを入力します）。

Example:

```
Device> enable
Device#
```

ステップ 2 **show running-config flow exporter** *exporter-name*

show running-config flow exporter コマンドでは、指定するフローエクスポータのコンフィギュレーションコマンドを表示します。

Example:

```
Device# show running-config flow exporter EXPORTER-1
Building configuration...
!
flow exporter EXPORTER-1
 description Exports to datacenter
 destination 172.16.10.2
 transport udp 65
!
```

デバッグの有効化

Cisco Performance Monitor のデバッグを有効にするには、特権 EXEC モードで、次のオプション作業を実行します。

SUMMARY STEPS

1. **debug performance monitor** {**database** | **dynamic** | **event** | **export** | **flow-monitor** | **metering** | **provision** | **sibling** | **snmp** | **tca** | **timer**}

DETAILED STEPS

debug performance monitor {**database** | **dynamic** | **event** | **export** | **flow-monitor** | **metering** | **provision** | **sibling** | **snmp** | **tca** | **timer**}

debug performance monitor コマンドは、次のパフォーマンス モニター コンポーネントのデバッグを有効にします。

- フロー データベース
- ダイナミック モニタリング
- パフォーマンス イベント
- エクスポート
- フロー モニタ
- 測定層
- プロビジョニング
- 兄弟管理
- SNMP
- TCA
- タイマー

次に、ダイナミック モニタリングを有効にする方法の例を示します。

Example:

```
Device# debug performance monitor dynamic
```

Cisco Performance Monitor の設定例

例：損失 RTP パケットおよび RTP ジッターのモニタリング

この例では、**gig1** インターフェイスの損失 RTP パケットの数、RTP ジッターの量、およびその他の基本統計情報をモニタする設定を示します。また、この例では、次のいずれかのイベントがインターフェイスで発生した場合に syslog でエントリが作成されるように Cisco Performance Monitor が設定されています。

- 損失 RTP パケットの割合が 5～9% です。
- 損失 RTP パケットの割合が 10% を超えています。

- メディア停止イベントが発生しました。

```
! Set the filter spec for the flows to monitor.
access-list 101 ip permit host 10.10.2.20 any
! Use the flow record to define the flow keys and metric to collect.
flow record type performance-monitor video-monitor-record
 match ipv4 source
 match ipv4 destination
 match transport source-port
 match transport destination-port
 match rtp ssrc
 collect timestamp
 collect counter byte
 collect counter packet
 collect mse
 collect media-error
 collect counter rtp interval-jitter
 collect counter rtp packet lost
 collect counter rtp lost event
! Set the exporting server. The export message format is based on FNFv.9.
flow export video-nms-server
 export-protocol netflow-v9
 destination cisco-video-management
 transport udp 32001
! Set the flow filter in the class-map.
class-map match-all video-class
 access-group  ipv4 101
! Set the policy map with the type performance-monitor for video monitor.
policy-map type performance-monitor video-monitor
 ! Set the video monitor actions.
 class video-class
  ! Specify where the metric data is being exported to.
  export  flow video-nms-server
  flow monitor inline
   record video-monitor-record
! Set the monitoring modeling parameters.
monitor parameters
 ! Set the measurement timeout to 10 secs.
 interval duration 10
 ! Set the timeout to 10 minutes.
 timeout 10
 ! Specify that 30 flow intervals can be kept in performance database.
 history 30
 priority 7
 ! Set rtp flow verification criteria.
 monitor metric rtp
 ! Configure a RTP flow criteria: at least 10 packets in sequence.
 min-sequential   10
 ! Ignore packets that are more than 5 packet ahead in terms of seq  number.  max-dropout
  5
 ! Ignore packets that are more than 5 packets behind in terms of seq  number.
 max-reorder 5
 ! Set the clock rate frequency for rtp packet timestamp clock.
 clock-rate 89000
 ! Set the maximum number of ssrc allowed within this class.
 ssrc maximum  100
 ! Set TCA for alarm.
 react 100 transport-packets-lost-rate
  description critical  TCA
  ! Set the threshold to greater than 10%.
  threshold gt 10
  ! Set the threshold to the average number based on the last five intervals.
  threshold type average 5
```

```
 action  syslog
 alarm severity critical
react 110 transport-packets-lost-rate
 description medium TCA
 ! Set the threshold to between 5% and 9% of packet lost.
 threshold range gt 5 le 9
 threshold type average 10
 action  syslog
 alarm type grouped percent 30
react 3000 media-stop
 action syslog
 alarm severity critical
 alarm type grouped percent 30

interface gig1
 service-policy type performance-monitor video-mon in
```

次の作業

Medianet製品ファミリの製品設定の詳細については、このガイドの他の章または『*Cisco Media Monitoring Configuration Guide*』を参照してください。

その他の参考資料

関連資料

関連項目	マニュアル タイトル
Performance Monitor およびその他の Cisco Medianet 製品の設計、設定、ならびにトラブルシューティングに関する資料（クイック スタート ガイドや導入ガイドなど）。	Cisco Medianet ナレッジ ベース ポータル サイト（http://www.cisco.com/web/solutions/medianet/knowledgebase/index.html）を参照してください。
IP アドレッシング コマンド：コマンド構文の詳細、コマンド モード、コマンド履歴、デフォルト、使用上の注意事項、および例	『*Cisco Media Monitoring Command Reference*』
Flexible NetFlow のコンフィギュレーション コマンド	『*Cisco IOS Flexible NetFlow Command Reference*』

関連項目	マニュアル タイトル
Flexible NetFlow の概要	「Cisco IOS Flexible NetFlow Overview」
Flexible NetFlow の機能ロードマップ	「Cisco IOS Flexible NetFlow Features Roadmap」
Flexible NetFlow データをエクスポートするためのフロー エクスポータの設定	「Configuring Data Export for Cisco IOS Flexible NetFlow with Flow Exporters」
Flexible NetFlow のカスタマイズ	「Customizing Cisco IOS Flexible NetFlow Flow Records and Flow Monitors」
Flexible NetFlow のトラフィック監視によるオーバーヘッド軽減のためのフロー サンプリング設定	「Using Cisco IOS Flexible NetFlow Flow Sampling to Reduce the CPU Overhead of Analyzing Traffic」
事前定義済みレコードを使用した Flexible NetFlow の設定	「Configuring Cisco IOS Flexible NetFlow with Predefined Records」
Flexible NetFlow Top N Talkers を使用したネットワーク トラフィックの分析	「Using Cisco IOS Flexible NetFlow Top N Talkers to Analyze Network Traffic」
Flexible NetFlow 用の IPv4 マルチキャスト統計情報サポートの設定	「Configuring IPv4 Multicast Statistics Support for Cisco IOS Flexible NetFlow」

標準

標準	タイトル
なし	—

MIB

MIB	MIB のリンク
• CISCO-FLOW-MONITOR-TC-MIB • CISCO-FLOW-MONITOR-MIB • CISCO-RTP-METRICS-MIB • CISCO-IP-CBR-METRICS-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
RFC 3954	『*Cisco Systems NetFlow Services Export Version 9*』 http://www.ietf.org/rfc/rfc3954.txt
RFC 3550	『*RTP: A Transport Protocol for Real-Time Applications*』 http://www.ietf.org/rfc/rfc3550.txt

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Cisco Performance Monitor の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 45: Cisco Performance Monitor の機能情報

機能名	リリース	機能情報
Cisco Performance Monitor 1.0	15.1(3)T 12.2(58)SE 15.1(4)M1 15.0(1)SY Cisco IOS XE Release 3.5S 15.1(1)SG Cisco IOS XE Release 3.3 SG	

機能名	リリース	機能情報
		この機能を使用すると、ネットワーク内のパケットフローをモニタして、ご使用のアプリケーションのパフォーマンスに重大な影響が現れる前に、そのフローに影響を及ぼす可能性がある問題を認識できるようになります。 この機能のサポートは、Cisco IOS XE Release 3.5S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ用に追加されました。 Cisco IOS XE Release 3.3 SG および Cisco IOS release 15.1(1)SG の場合、特定のタイプのインターフェイスでは入力データまたは出力データのモニタリングに関する制限事項がいくつかあります。詳細については、「制限事項」を参照してください。 他のすべてのリリースでは、次のコマンドがこの機能によって導入または変更されました。**action**(policy react and policy inline react)、**alarm severity** (policy react and policy inline react)、 **alarm type**(policy react and policy inline react)、 **class-map**、 **clock-rate**(policy RTP)、 **collect application media**、**clear fm performance-monitor counters**、 **collect counter**、 **collect flow direction**、**collect interface**、 **collect ipv4**、 **collect ipv4 destination**、 **collect ipv4 source**、**collect ipv4 ttl**、 **collect monitor event**、 **collect routing**、 **collect timestamp interval**、 **collect transport event packet-loss counter**、 **collect transport packets**、 **collect transport rtp jitter**、**debug fm performance-monitor counters**、**debug performance-monitor counters**、 **description** (Performance Monitor)、**destination dscp** (Flexible NetFlow)、**export-protocol**、 **exporter**、**flow monitor type performance-monitor**、 **flow record type performance-monitor**、 **flows**、 **history** (monitor parameters)、**interval duration**、 **match access-group**、 **match any**、 **match class-map**、 **match cos**、 **match destination-address mac**、 **match discard-class**、 **match dscp**、**match flow**、 **match fr-de**、 **match fr-dlci**、 **match input-interface**、 **match ip dscp**、 **match ip precedence**、 **match ip rtp**、 **match ipv4**、 **match ipv4 destination**、 **match ipv4 source**、 **match mpls experimental topmost**、 **match not**、 **match packet length** (class-map)、**match precedence**、 **match protocol**、**match qos-group**、 **match source-address mac**、 **match transport destination-port**、 **match transport rtp ssrc**、 **match transport source-port**、**match vlan**、 **max-dropout** (policy RTP)、**max-reorder** (policy RTP)、**min-sequential** (policy RTP)、**monitor metric ip-cbr**、 **monitor metric rtp**、**monitor parameters**、 **option** (Flexible NetFlow)、**output-features**, **platform performance-monitor rate-limit**、 **policy-map type performance-monitor**、**rate layer3**、 **react** (policy)、 **record** (Performance Monitor)、**rename** (policy)、**service-policy type performance-monitor**、 **show performance monitor history**、 **show performance monitor status**、**show platform hardware acl entry interface**、**show platform software ccm**、**show platform software feature-manager performance-monitor**、**show platform software feature-manager tcam**、 **show policy-map type performance-monitor**, **snmp-server host**, **snmp-server enable traps flowmon**、**snmp mib flowmon**

機能名	リリース	機能情報
		alarm history 、 **source**(Flexible NetFlow)、**ssrc maximum**、 **template data timeout**、 **threshold value** (policy react and policy inline react)、**timeout** (monitor parameters)、**transport** (Flexible NetFlow)、および **ttl** (Flexible NetFlow)。
Performance Monitor（フェーズ 2）	15.2(2)T Cisco IOS XE Release 3.5S	この機能を使用すると、IPv6 フィールドをモニタできるようになります。また、以前のリリースではサポートされていない Flexible NetFlow の他のすべての collect コマンドと match コマンドを使用できます。 現在では、フローが相互に関連付けられるので、同じポリシーが同じ入力インターフェイスと出力インターフェイスに適用されている場合に show コマンドを実行すると、その入力インターフェイスと出力インターフェイスについて単一のフローが表示されます。 この機能のサポートは、Cisco IOS XE Release 3.5S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ用に追加されました。 この機能により、次のコマンドが導入または変更されました。 **collect datalink mac**、 **collect ipv4 fragmentation**、 **collect ipv4 section**、 **collect ipv4 total-length**、 **collect ipv6**、 **collect ipv6 destination**、 **collect ipv6 extensionmap**、 **collect ipv6 fragmentation**、 **collect ipv6 hop-count**, **collect ipv6 length**、 **collect ipv6 section**, **collect ipv6 source**、 **collect routing is-multicast**、 **collect routing multicast replication-factor**、 **collect timestamp sys-uptime**、 **collect transport**、 **collect transport icmp ipv4**、 **collect transport icmp ipv6**、 **collect transport tcp**、 **collect transport udp**、 **match application name**、 **match connection transaction-id**、 **match datalink dot1q vlan**、 **match datalink mac**、 **match datalink vlan**、 **match interface**、 **match ipv4 fragmentation**、 **match ipv4 section**、 **match ipv4 total-length**、 **match ipv4 ttl**、 **match ipv6**、 **match ipv6 destination**、 **match ipv6 extension map**、 **match ipv6 fragmentation**、 **match ipv6 hop-limit**、 **match ipv6 length**、 **match ipv6 section**、 **match ipv6 source**、 **match routing**、 **match routing is-multicast**、 **match routing multicast replication-factor**、 **match transport**、 **match transport icmp ipv4**、 **match transport icmp ipv6**、 **match transport tcp**、 **match transport udp**

機能名	リリース	機能情報
Performance Monitor（フェーズ 3）	15.2(3)T Cisco IOS XE Release 3.7S	この機能を使用すると、複数のエクスポータを設定し、メタデータ フィールドと新しい TCP メトリックをモニタできます。 この機能のサポートは、Cisco IOS XE リリース 3.7S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに追加されました。 この機能により、次のコマンドが導入または変更されました。**collect application**、**collect transport tcp bytes out-of-order**、**collect transport packets out-of-order**、**collect transport tcp maximum-segment-size**、**collect transport tcp window-size maximum**、**collect transport tcp window-size minimum**、**collect transport tcp window-size average**、**match application**、**match transport tcp bytes out-of-order**、**match transport packets out-of-order**、**match transport tcp maximum-segment-size**、**match transport tcp window-size maximum**、**match transport tcp window-size minimum**、**match transport tcp window-size average**
パフォーマンス モニタリング：IPv6 サポート	Cisco IOS XE Release 3.6S	この機能を使用すると、モニタを IPv6 インターフェイスに接続できます。 この機能のサポートは、Cisco IOS XE Release 3.6S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ用に追加されました。
パフォーマンス モニタリング：誤った順序でのパケットのトランスポート	Cisco IOS XE Release 3.6S	この機能を使用すると、誤った順序で送信された TCP パケットの合計数をモニタできます。 この機能のサポートは、Cisco IOS XE Release 3.6S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ用に追加されました。 この機能により、次のコマンドが導入または変更されました。**collect transport tcp bytes out-of-order** および **collect transport packets out-of-order**。
Flexible NetFlow：IPFIX エクスポート フォーマット	15.2(4)M Cisco IOS XE リリース 3.7S	IPFIX エクスポート プロトコルを使用したエクスポート パケットの送信を有効化します。NBAR から抽出されたフィールドのエクスポートは、IPFIX 経由でのみサポートされます。 この機能のサポートは、Cisco IOS XE リリース 3.7S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに追加されました。 次のコマンドが導入されました：**export-protocol**
IPv6 アドレスへの Flexible NetFlow エクスポート	Cisco IOS XE リリース 3.7S	この機能では、Flexible NetFlow で IPv6 アドレスを使用してデータを宛先にエクスポートできます。 この機能のサポートは、Cisco IOS XE リリース 3.7S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに追加されました。 次のコマンドが導入されました：**destination**

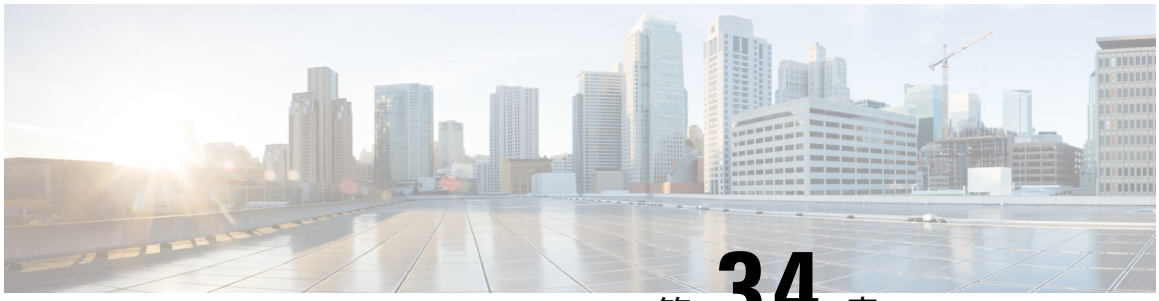
機能名	リリース	機能情報
Flexible NetFlow：抽出フィールドのサポート	Cisco IOS XE リリース 3.7S	NBAR を使用した抽出フィールドの収集を有効にします。抽出されたフィールドのエクスポートは、IPFIX 経由でのみサポートされます。 この機能のサポートは、Cisco IOS XE リリース 3.7S で Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに追加されました。 この機能により、次のコマンドが導入または変更されました。**collect http host**、**collect nntp group-name**、**collect pop3 server**、**collect rtsp host-name**、**collect sip destination**、**collect sip source**、**collect smtp server** および **collect smtp sender**。

機能名	リリース	機能情報
	Cisco IOS XE Release 3.8S	AVC 2.0 は、AVC とメディア モニタリング テクノロジーの統合など、広範な新機能を提供します。 このマニュアルでは、AVC 2.0 のフローレコードの設定方法についてのみ説明します。AVC 2.0 の詳細については、「*http://www.cisco.com/en/US/docs/ios-xml/ios/avc/configuration/xe-3s/avc-xe-3s-book.html*」を参照してください。

機能名	リリース	機能情報
Application Visibility and Control（AVC）2.0 には次の機能が含まれます。 • パフォーマンス モニタリング ポリシーでのアプリケーション使用状況の可視化を有効にする • アプリケーション使用のパフォーマンスを有効にす		

機能名	リリース	機能情報
る • Prime とルータパケットキャプチャの統合を有効にする • サービスパスの可視化を有効にする • FNF：WAAS セグメントの Account On Resolution (ACR)		

機能名	リリース	機能情報
• FNF：パフォーマンス モニタリング ポリシーマップ用の Account On Resolution (ACR)		



第 34 章

アシュアランス モニタリングのメトリック

アシュアランスモニタリングのメトリックとは、Cisco DNA Center によるアシュアランスモニタリングをサポートするために、特定のインターフェイスを介して転送されるフローについて、ネットワークアプリケーションごとに収集されるアシュアランス関連のメトリックを指します。FNF は、このデータを収集するためのレコードタイプのペア（IPv4 と IPv6 用）を提供します。Monitoring for Assurance は、FNF モニターの一般的なパフォーマンスよりも優れたパフォーマンスを提供するように最適化されています。

アシュアランス モニタリングのメトリックの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *46 :* アシュアランス モニタリングのメトリックの機能情報

機能名	リリース	機能情報
アシュアランス モニタリングのメトリック	Cisco IOS XE Gibraltar 16.10.1	FNF は、アシュアランスのデータを収集するためのレコードタイプのペアを提供し、FNF モニターの一般的なパフォーマンスよりも優れたパフォーマンスを提供するように最適化されています。

アシュアランス モニタリングのメトリックについて

概要

DNA Center アシュアランス

Cisco DNA Center Assurance は、ネットワークデータを収集して分析し、より優れた一貫性のあるネットワークパフォーマンスを提供します。Cisco DNA Center は Flexible NetFlow（FNF）を使用して Assurance の特定のネットワークメトリックを収集し、ネットワーク内のデバイスに関する定量的および定性的な情報を提供します。アシュアランス関連のメトリック用に設計された FNF レコードは、パフォーマンスを向上させるために特別に最適化されています。

FNF は、アシュアランスのデータを収集するためのレコードタイプのペア（IPv4 と IPv6 用）を提供します。これらの専用レコードタイプを使用したアシュアランスメトリックのモニタリングは、同じメトリックを収集するように設定された一般的な FNF モニターと比較して、パフォーマンスが向上するように最適化されています。（レコードを変更すると、アシュアランス専用のパフォーマンス拡張がキャンセルされ、モニターをインターフェイスに接続できなくなる場合があります）。

手動設定

通常の使用では、Cisco DNA Center は、ユーザー入力を必要とせずに Assurance のデータを収集するようにモニターを設定します。ただし、これらのレコードタイプを手動で使用することもできます。

アシュアランスのために収集されるメトリック

アシュアランス用に収集されるメトリックのほとんどは、FNF およびその他のモニタータイプを介して使用できるメトリックですが、アシュアランスレコード専用に収集される場合、一部のメトリックの動作が若干異なる場合があります。

表 *47 : Metrics*

Metric	情報
match ipv4/ipv6 version	IPv4/IPv6 ヘッダーからの IPv4/IPv6 バージョン。 [1]
match ipv4/ipv6 protocol	IPv4/IPv6 ヘッダーからのレイヤ 4 プロトコル。
match application name	Application ID
match connection client ipv4/ipv6 address	フィールド名：clientIPv4/IPv6Address IP パケットヘッダー内の IPv4/IPv6 クライアントアドレス。クライアントは、セッションの作成をトリガーしたデバイスであり、セッションの期間中も同じです。 [2]
match connection server ipv4/ipv6 address	フィールド名：serverIPv4/IPv6Address IP パッカーヘッダー内の IPv4/IPv6 サーバーアドレス。サーバーは、クライアントに応答するデバイスであり、セッションの期間中も同じです。 [2]
match connection server transport port	フィールド名：serverTransportPort サーバーの転送ポート ID。これは、送信元または宛先の転送ポートになります。サーバーは、クライアントに応答するデバイスであり、セッションの期間中も同じです。 [2]
match flow observation point	フィールド名：observationPointId 観測ドメインごとに一意の観測ポイントの識別子。 [2]
collect connection initiator	フィールド名：biflowDirection Biflow の送信元と宛先を割り当てるために使用される方向割り当て方式の説明。 [2]
collect flow direction	観測ポイントで観測されたフロー

Metric	情報
collect routing vrf input	フィールド名：ingressVRFID （ルータにのみ適用され、ワイヤレスコントローラには適用されません） ルータの着信パケットからの VRF ID。パケットが VRF に属していないインターフェイスに到着すると、VRF ID 0 が記録されます。
collect wireless client mac address	（ワイヤレスコントローラにのみ適用） フィールド名：staMacAddress ワイヤレスステーション（STA）の IEEE 802 MAC アドレス。
collect timestamp absolute first	フィールド名：flowStartMilliseconds フローの先頭パケットの絶対タイムスタンプ。
collect timestamp absolute last	フィールド名：flowEndMilliseconds フローの最終パケットの絶対タイムスタンプ。
collect connection new-connections	フィールド名：connectionCountNew この情報エレメントは、観測期間中に開かれた TCP または UDP 接続の数をカウントします。観測期間は、フローの開始タイムスタンプと終了タイムスタンプで指定できます。 [2]
collect connection server counter packets long	フィールド名：serverPackets サーバーからのフロー内のレイヤ 4 パケットの数。サーバーは、クライアントに応答するデバイスであり、セッションの期間中も同じです。 [2]
collect connection server counter bytes network long	フィールド名：serverOctets サーバーからのフローの IP パケット全体のバイト数。サーバーは、クライアントに応答するデバイスであり、セッションの期間中も同じです。 [2]
collect connection client counter packets long	フィールド名：clientPackets クライアントからのフロー内のレイヤ 4 パケットの数。クライアントは、セッションの作成をトリガーしたデバイスであり、セッションの期間中も同じです。 [2]

Metric	情報
collect connection client counter bytes network long	クライアントからサーバーへの IP パケット全体のバイト数。 [2]
collect connection delay network client-to-server sum	フィールド名：sumNwkTime ネットワーク遅延は、観測ポイントによって測定されるクライアントとサーバー間のラウンドトリップ時間であり、セッションごとに 1 回計算されます。この情報要素の値は、このフローのセッションで観測されたすべてのネットワーク遅延の合計です。 [2] [3]
collect connection delay network to-server sum	フィールド名：sumServerNwkTime サーバーネットワーク遅延は、観測ポイントとサーバー間のラウンドトリップ時間であり、セッションごとに 1 回計算されます。この情報要素の値は、このフローのセッションで観測されたすべてのサーバーネットワーク遅延の合計です。 [2] [3]
collect connection client counter packets retransmitted	フィールド名：retransClientPackets クライアントによって再送信されたパケットの数。 [2] [3]
collect connection server counter packets retransmitted	フィールド名：retransServerPackets サーバーによって再送信されたパケットの数。 [3]
collect connection delay application sum	フィールド名：sumServerRespTime フローのすべての応答で観測されたすべてのアプリケーション遅延の合計。 [2] [3]
collect connection server counter responses	フィールド名：numRespsCountDelta サーバーによって送信された応答の合計数。 [2] [3]

注記

[1]『Cisco IOS Flexible NetFlow Command Reference』を参照してください。

[2]『Cisco AVC Field Definition Guide』を参照してください。

[3] このメトリックは、Cisco Performance Monitor レコードタイプで使用できます。FNF では、特別に最適化されたアシュアランス関連レコードの一部としてのみ使用できます。別

のFNF レコードタイプでこのメトリックを使用しようとすると、インターフェイスにアタッチするときにレコードが拒否されます。

アシュアランス モニタリングのメトリックの設定方法

Cisco DNA Center の外部でのアシュアランスモニターの設定

通常の使用では、Cisco DNA Center は追加のユーザー入力を必要とせずにモニターを設定しますが、アシュアランス関連のメトリックのモニターを手動で設定することもできます。

Assurance 関連のメトリックを手動でモニタリングする方法：

Method	適用対象	参照セクション
ezPM プロファイル	ezPM をサポートするプラットフォーム 非ワイヤレスコントローラ	ezPM を使用したアシュアランスモニターの設定 （538 ページ）
Assurance 用の定義済み FNF レコード	ルータ ワイヤレスコントローラ	事前定義された FNF レコードを使用したアシュアランスモニターの設定 （539 ページ）

ezPM を使用したアシュアランスモニターの設定

ルータに適用され、ワイヤレスコントローラには適用されません

アプリケーションアシュアランス ezPM プロファイルは、アシュアランス関連のメトリック用に設計されたアプリケーション パフォーマンス監視（APM）FNF レコードを利用します。ezPM を使用して APM を設定すると、FNF レコードを直接操作する場合と比較して、設定が大幅に簡素化されます。

1. ezPM コンテキストを設定します。

   performance monitor context *context-name* **profile application-assurance**

   traffic-monitor assurance-monitor ipv4

   traffic-monitor assurance-monitor ipv6

2. コンテキストをインターフェイスに接続します。次に、パフォーマンスモニターをインターフェイスに接続し、入力と出力の両方をモニターします。

   interface *interface*

   performance monitor context *context-name*

結果

これにより、モニターがインターフェイスに接続され、アシュアランス関連のメトリックが収集されます。

例

次の例では、apm というモニターがギガビットイーサネット 1 インターフェイスに接続されています。

```
performance monitor context apm profile application-assurance
traffic-monitor assurance-monitor ipv4
traffic-monitor assurance-monitor ipv6

interface GigabitEthernet1
performance monitor context apm
```

事前定義された FNF レコードを使用したアシュアランスモニターの設定

ルータに適用され、ワイヤレスコントローラには適用されません

ezPM は、アシュアランス関連のメトリックのモニターを設定するための推奨される方法ですが、これらのメトリックに対して事前定義された FNF レコードを使用することもできます。ezPM をサポートしていないプラットフォームでは、この方法が推奨されます。

アシュアランス関連のメトリック用に設計された FNF レコードは、パフォーマンスを向上させるために特別に最適化されています。

ルーティング プラットフォームでの設定方法

（注） ワイヤレスプラットフォームには適用されません。

1. アシュアランス関連のメトリック用に 2 つのフローモニターを定義します。1 つは IPv4 用、もう 1 つは IPv6 用です。

   flow monitor *monitor-name-for-ipv4*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record netflow ipv4 assurance

   flow monitor *monitor-name-for-ipv6*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record netflow ipv6 assurance

2. コンテキストをインターフェイスに接続します。次に、パフォーマンスモニターをインターフェイスに接続し、入力と出力の両方をモニターします。

   interface *interface*

ipv4 flow monitor *monitor-name-for-ipv4* **input**

ipv4 flow monitor *monitor-name-for-ipv4* **output**

ipv6 flow monitor *monitor-name-for-ipv6* **input**

ipv6 flow monitor *monitor-name-for-ipv6* **output**

結果

これにより、保証に必要なメトリックを収集するために、2 つの IPv4 モニターと 2 つの IPv6 モニターがインターフェイスに接続されます。

例

この例では、assurance-ipv4 および Assurance-ipv6 というモニターを定義し、これらのモニターを GigabitEthernet1 インターフェイスに接続します。

```
flow monitor assurance-ipv4
cache entries 100000
record netflow ipv4 assurance

flow monitor assurance-ipv6
cache entries 100000
record netflow ipv6 assurance

interface GigabitEthernet1
ipv4 flow monitor assurance-ipv4 input
ipv4 flow monitor assurance-ipv4 output
ipv6 flow monitor assurance-ipv6 input
ipv6 flow monitor assurance-ipv6 output
```

ワイヤレスプラットフォームでの設定方法

（注） ルーティングプラットフォームには適用されません。

1. 関連するワイヤレスプロファイルの設定モードを開始します。

   interface *policy-name*

2. ワイヤレスコントローラ用に 2 つのモニターを定義します。1 つは IPv4 用、もう 1 つは IPv6 用です。

   flow monitor *monitor-name-wlc-for-ipv4*

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record wireless avc ipv4 assurance

   flow monitor monitor-name-wlc-ipv6

   cache entries 100000 {Optional. Recommended value depends on platform.}

   record wireless avc ipv6 assurance

3. 入力トラフィックと出力トラフィックを含む 2 つのフローモニターをワイヤレスプロファイルに接続します。

   wireless profile policy *policy-name*

   ipv4 flow monitor *monitor-name-for-wireless-ipv4* **input**

   ipv4 flow monitor *monitor-name-for-wireless-ipv4* **output**

   ipv6 flow monitor monitor-name-for-wireless-ipv6 **input**

   ipv6 flow monitor *monitor-name-for-wireless-ipv6* **output**

例

この例では、assurance-wlc-ipv4 および Assurance-wlc-ipv6 というモニターを定義し、モニターをワイヤレスプロファイルに接続します。

```
flow monitor assurance-wlc-ipv4
cache entries 100000
record wireless avc ipv4 assurance

flow monitor assurance-wlc-ipv6
cache entries 100000
record wireless avc ipv6 assurance

wireless profile policy AVC_POL
central association
central switching
ipv4 flow monitor assurance-wlc-ipv4 input
ipv4 flow monitor assurance-wlc-ipv4 output
ipv6 flow monitor assurance-wlc-ipv6 input
ipv6 flow monitor assurance-wlc-ipv6 output
no shutdown
```

インターフェイスへのアシュアランスモニターの接続について

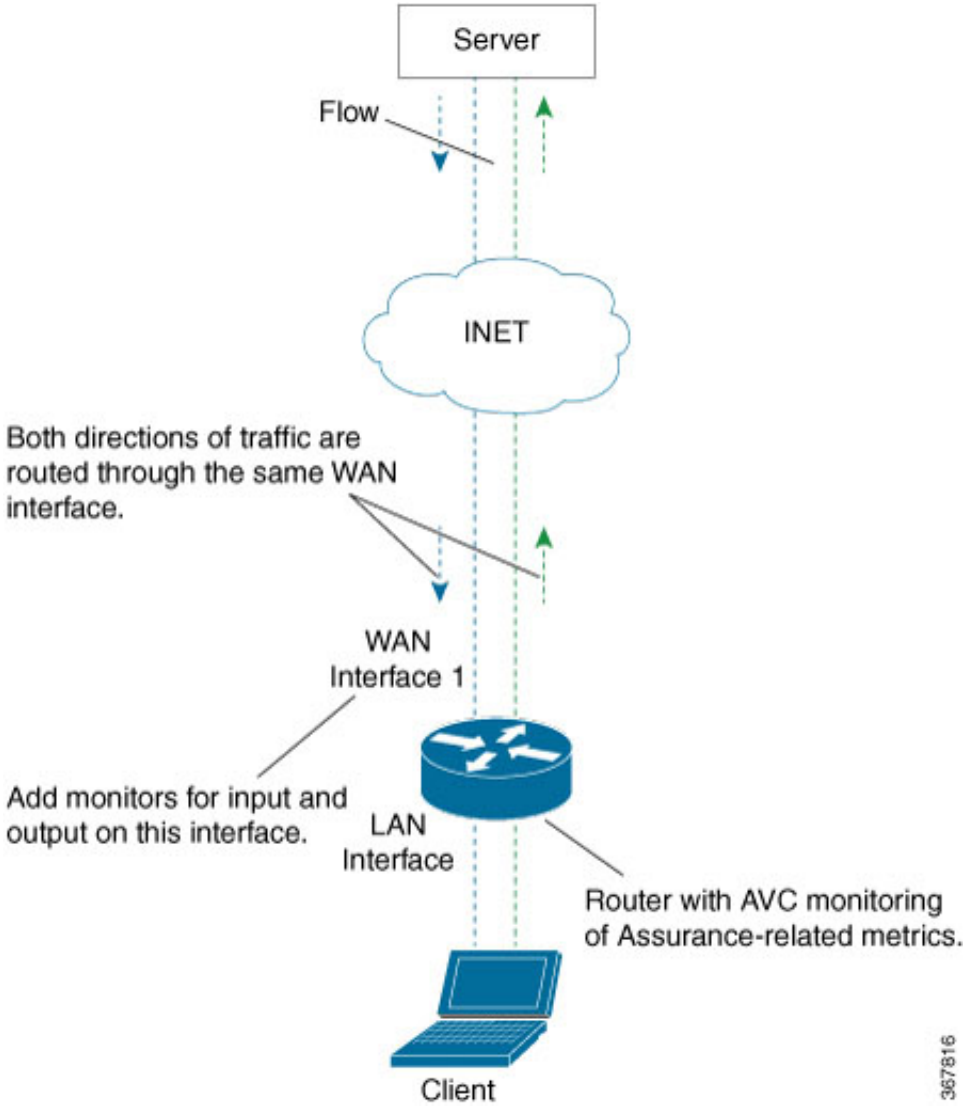
1 つのインターフェイスのみでのフローのモニタリング

アシュアランス関連のメトリックのモニターには、1 つのフローのみが 1 回表示されます。一般的な対称ルーティングのシナリオでは、1 つのインターフェイスでのみフローをモニターする必要があります。

同じフローの両方向を処理する 2 つの個別のインターフェイスに、アシュアランス関連のメトリックのモニターを接続しないでください。これを行うと、誤ったトラフィックメトリックが報告されます。たとえば、トラフィックがインターフェイス A のデバイスに入り、インターフェイス B から出る場合、インターフェイス A と B の両方にアシュアランス関連のメトリックのモニターを接続しないでください。

同じインターフェイスで入力と出力のモニターを使用する一般的な対称ルーティング：

図 *12* : 対称ルーティング

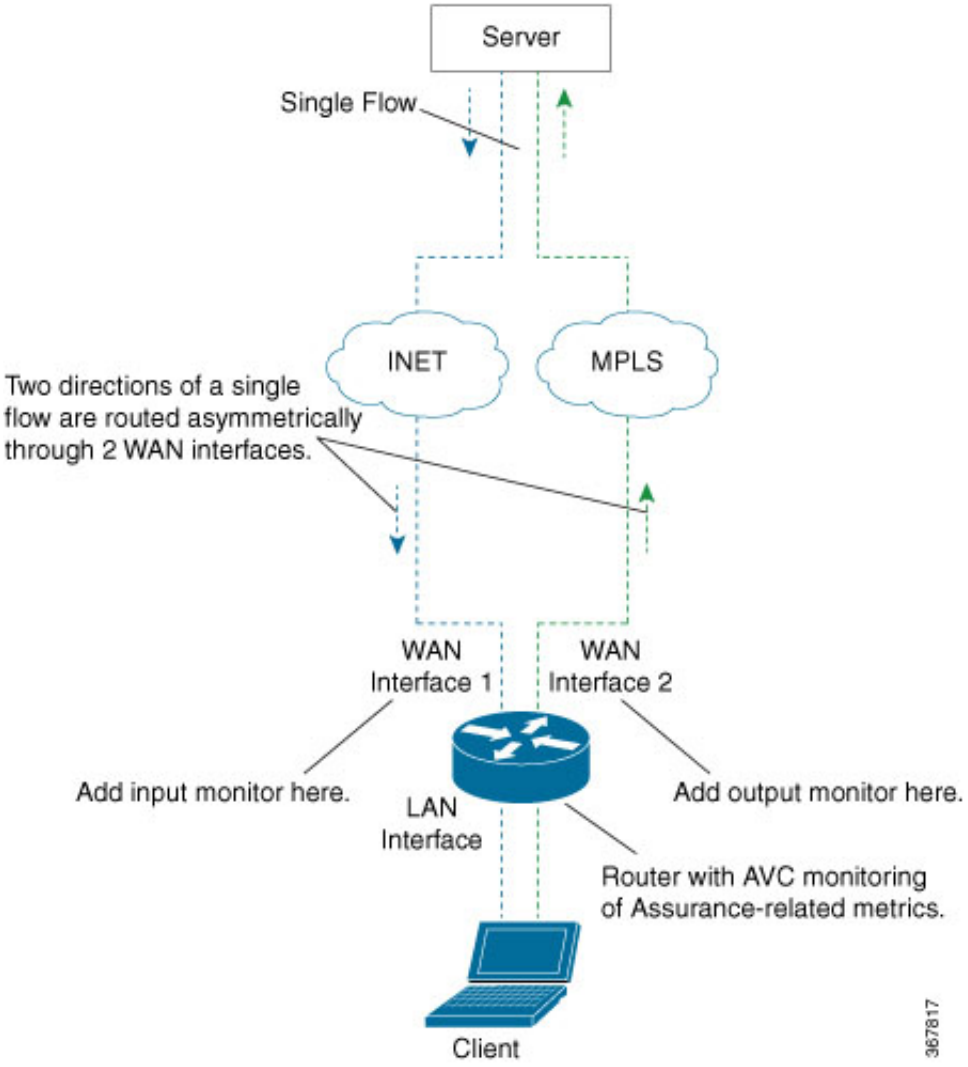


非対称ルーティング

非対称ルーティングなど、場合によっては、1 つのインターフェイスに入力用のモニターを接続し、別のインターフェイスに出力用のモニターを接続する必要があります。

一部のシナリオでは、単一のフローが非対称にルーティングされ、フローのアップストリーム トラフィックとダウンストリーム トラフィックが2つの異なるインターフェイスで発生する場合があります。この場合、入力と出力のモニターを 2 つの個別のインターフェイスに配置して、フロー全体をモニターします。

図 *13* : 非対称ルーティング



アシュアランスレコードとコンテキストの詳細の表示

概要

コンテキストをインターフェイスに付加した後、2 つの **show** コマンドを使用して、アシュアランスレコードまたはコンテキストに関する情報を表示できます。

アシュアランスレコードの構造の表示

次のコマンドは、事前定義されたアシュアランスレコード（IPv4 および IPv6）の構造を表示します。

show fnf record netflow {**ipv4** | **ipv6**} **assurance**

コンテキストの設定の表示

次のコマンドは、指定されたコンテキストの完全なコンフィギュレーションを表示します。

show performance monitor context *context-name* **configuration**

次の出力は、ルータインターフェイスに接続された ApmContext と呼ばれる ezPM コンテキストを介したアシュアランス関連のモニタリングを示しています。

```
Device#show performance monitor context ApmContext configuration
!================================================================================
!                Equivalent Configuration of Context ApmContext                !
!================================================================================
!Exporters
!==========
!
flow exporter ApmContext-1
description performance monitor context ApmContext exporter
destination 64.103.113.128 vrf FNF
source GigabitEthernet2/2/0
transport udp 2055
export-protocol ipfix
template data timeout 300
option interface-table timeout 300
option vrf-table timeout 300
option sampler-table timeout 300
option application-table timeout 300
option application-attributes timeout 300
!
!Access Lists
!=============
!Class-maps
!===========
!Samplers
!=========
!Records and Monitors
!====================
!
flow record ApmContext-app_assurance_ipv4
description ezPM record
match ipv4 version
match ipv4 protocol
match application name
match connection client ipv4 address
match connection server ipv4 address
match connection server transport port
match flow observation point
collect routing vrf input
collect flow direction
collect timestamp absolute first
collect timestamp absolute last
collect connection initiator
collect connection new-connections
collect connection server counter responses
collect connection delay network to-server sum
collect connection client counter packets retransmitted
collect connection delay network client-to-server sum
collect connection delay application sum
collect connection server counter packets long
collect connection client counter packets long
```

```
collect connection server counter packets retransmitted
collect connection server counter bytes network long
collect connection client counter bytes network long
!
!
flow monitor ApmContext-app_assurance_ipv4
description ezPM monitor
exporter ApmContext-1
cache timeout active 60
cache entries 100000
record ApmContext-app_assurance_ipv4
!
!
flow record ApmContext-app_assurance_ipv6
description ezPM record
match ipv6 version
match ipv6 protocol
match application name
match connection client ipv6 address
match connection server transport port
match connection server ipv6 address
match flow observation point
collect routing vrf input
collect flow direction
collect timestamp absolute first
collect timestamp absolute last
collect connection initiator
collect connection new-connections
collect connection server counter responses
collect connection delay network to-server sum
collect connection client counter packets retransmitted
collect connection delay network client-to-server sum
collect connection delay application sum
collect connection server counter packets long
collect connection client counter packets long
collect connection server counter packets retransmitted
collect connection server counter bytes network long
collect connection client counter bytes network long
!
!
flow monitor ApmContext-app_assurance_ipv6
description ezPM monitor
exporter ApmContext-1
cache timeout active 60
cache entries 100000
record ApmContext-app_assurance_ipv6
!
!Interface Attachments
!======================
interface TenGigabitEthernet2/0/0
ip flow monitor ApmContext-app_assurance_ipv4 input
ip flow monitor ApmContext-app_assurance_ipv4 output
ipv6 flow monitor ApmContext-app_assurance_ipv6 input
ipv6 flow monitor ApmContext-app_assurance_ipv6 output
```

注意事項と制限事項

アシュアランス関連のメトリックとエレファントフロー

ネットワーキングでは、特に長いフローは「エレファントフロー」と呼ばれ、ネットワーキングリソースに課題をもたらす可能性があります。

単一の高バーストフローが大量の QFP リソースを消費する場合、保証メトリックを収集しているモニターは、他のトラフィック用のリソースを確保するために、フローの定性メトリックの収集を停止する可能性があります。他のトラフィックは影響を受けません。

定量的メトリックは次のように完全に収集されます。

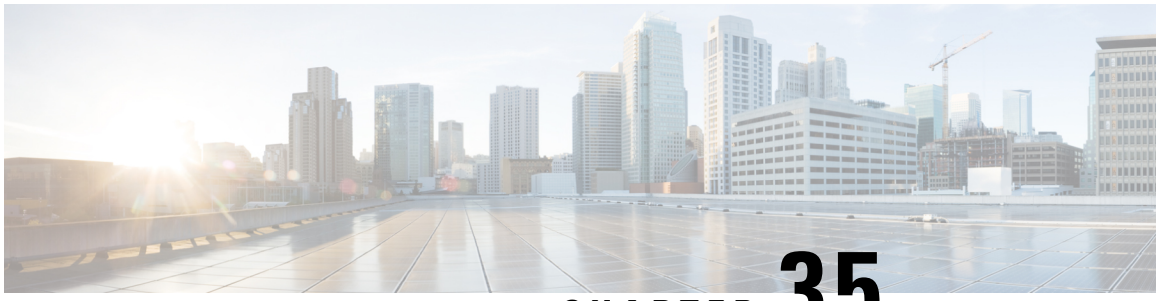
- フローパケットの開始時刻
- フローパケットの終了時刻
- パケット
- Bytes

定性的メトリックは完全には収集されません。

- 合計ネットワーク遅延合計 (TCP ハンドシェイク時)
- ネットワークからサーバーへの遅延合計（TCP ハンドシェイク時）
- 再送信されたクライアントパケット
- 再送信されたサーバーパケット
- アプリケーション遅延合計
- サーバーアプリケーションの応答数

第 VIII 部

組み込まれている Event Manager



CHAPTER 35

Embedded Event Manager Overview

Embedded Event Manager（EEM）は、イベント検出と回復を Cisco IOS 内部で直接行うための分散型でカスタマイズされた手法です。EEM はイベントをモニターし、モニター対象イベントが発生したり、しきい値に達したりすると、情報提供や訂正などの必要な EEM 処理を実行します。EEM ポリシーは、イベントおよびイベントが発生した場合に行う処理を定義するエンティティです。

この章では、EEM の技術的概要を説明します。EEM は単体でも使用できますが、ネットワークのモニターとメンテナンスのための他のネットワーク管理テクノロジーと合わせて使用することもできます。EEM の実装を開始する前に、このモジュールに示す情報を理解することが重要です。

Embedded Event Manager について

組み込まれている Event Manager

従来、イベント トラッキングおよびイベント管理はネットワーキング デバイスの外部のデバイスによって実施されてきました。Embedded Event Manager（EEM）は、イベント管理を Cisco IOS デバイス内で直接実施できるように設計されました。障害によってはデバイスと外部ネットワーク管理デバイスとの通信が損なわれることがあるため、デバイス外ですべてのイベント管理ができるわけではないことから、EEM のデバイス上での予防的なイベント管理機能は有用です。このような状況でデバイスの状態をキャプチャすることは、迅速な回復アクションの実行、および根本原因の分析実施のための情報収集に非常に役立ちます。ルーティング デバイスを完全にリブートすることなしに自動回復アクションが実施されれば、ネットワーク可用性も向上します。

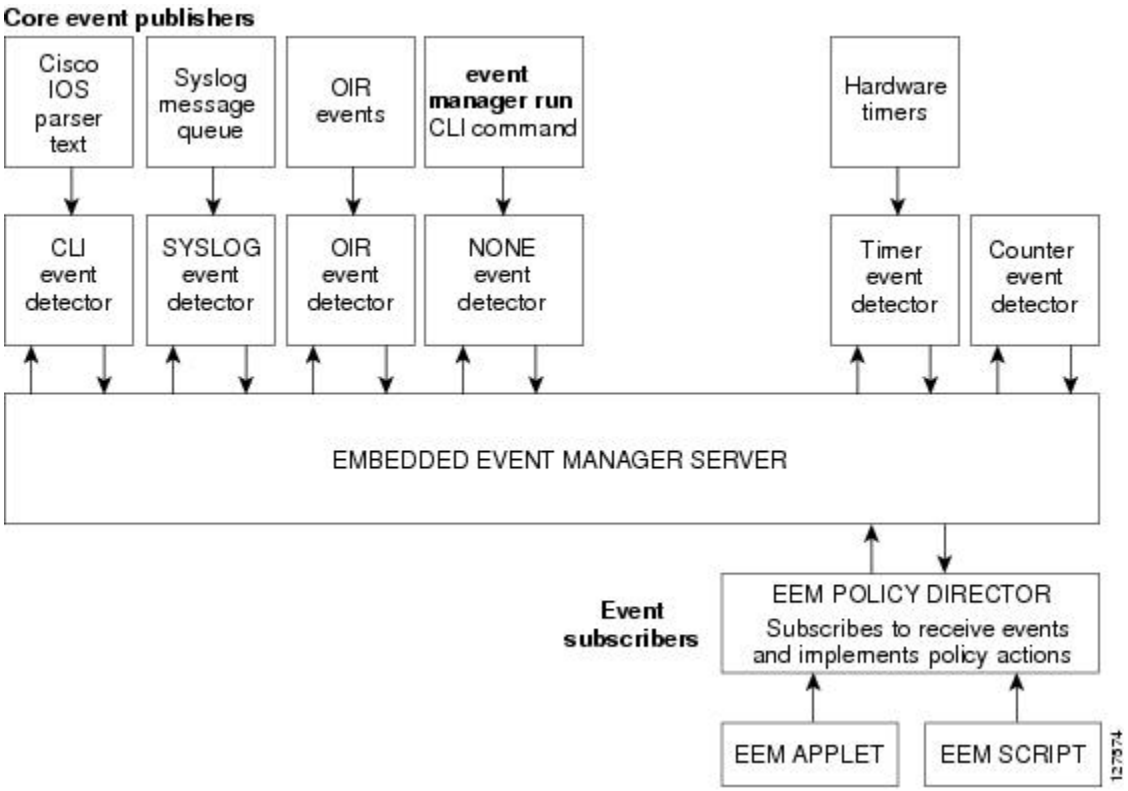
EEM は、イベント ディテクタと呼ばれるソフトウェア エージェントを使用してシステム内の異なるコンポーネントのモニターリングをサポートする、柔軟でポリシードリブンのフレーム

組み込まれている Event Manager

ワークです。次の図に、EEMサーバー、コアイベントパブリッシャ（イベントディテクタ）、およびイベントサブスクライバ（ポリシー）の関係を示します。基本的に、イベントパブリッシャはイベントをスクリーニングして、イベントサブスクライバから提供されたイベント仕様に一致したときにイベントをパブリッシュします。イベントディテクタは、注目するイベントが発生したときにEEMサーバーに通知します。Ciscoコマンドラインインターフェイス（CLI）を使用して設定されたEEMポリシーは、現在のシステムの状態と、該当するイベントのポリシーで指定されたアクションに基づいて回復を実施します。

EEMでは、イベントをモニターし、イベント発生が検出されたとき、およびしきい値を超えたときに情報通知や是正アクションを実施できます。EEMポリシーは、イベントおよびイベントが発生した場合に行う処理を定義するエンティティです。EEMポリシーにはアプレットとスクリプトの2つのタイプがあります。アプレットは、CLI設定に定義された、ポリシーの単純な形式です。スクリプトは、Tool Command Language（Tcl）で記述されたポリシーの形式です。

***Figure 14: Embedded Event Manager* コア イベント ディテクタ**



Note ネットワークにEEMの上位バージョンがある場合、そのバージョンは以前のリリースのEEMバージョンを含みます。

Embedded Event Manager 1.0

EEM 1.0 では、Embedded Event Manager が導入されました。EEM 1.0 は次のイベント ディテクタを追加しました。

- SNMP：簡易ネットワーク管理プロトコル（SNMP）イベント ディテクタによって、標準 SNMP MIB オブジェクトを監視し、オブジェクトが指定された値と一致するとき、または指定されたしきい値を超えたときにイベントを生成することができます。
- Syslog：syslog イベント ディテクタは、正規表現パターン マッチに対して syslog メッセージをスクリーニングできます。

EEM 1.0 は、次のアクションを追加しました。

- 優先化された syslog メッセージの生成。
- Cisco Networking Services（CNS）デバイスによるアップストリーム処理に対し CNS イベントの生成。
- シスコのソフトウェアをリロードします。
- 完全冗長ハードウェア構成におけるセカンダリ プロセッサへのスイッチング。

Embedded Event Manager 2.0

EEM 2.0 で、いくつかの新機能が導入されました。EEM 2.0 では次のイベント ディテクタが追加されました。

- Application-Specific：Application-Specific イベント ディテクタによって、Embedded Event Manager ポリシーはイベントをパブリッシュできます。
- Counter：Counter イベント ディテクタは、名前付きカウンタが指定されたしきい値を超えたときにイベントをパブリッシュします。
- Interface Counter：Interface Counter イベント ディテクタは、指定されたインターフェイスの汎用 Cisco IOS インターフェイス カウンタが定義されたしきい値を超えたときにイベントをパブリッシュします。
- Timer：Timer イベント ディテクタは、absolute-time-of-day、countdown、watchdog、および CRON の 4 種類のタイマーのイベントをパブリッシュします。
- Watchdog System Monitor（IOSWDSysMon）：Cisco IOS Watchdog System Monitor イベント ディテクタは、Cisco IOS プロセスの CPU またはメモリの使用率がしきい値を超えたときにイベントをパブリッシュします。

EEM 2.0 では次のアクションが追加されました。

- 名前付きカウンタの設定または変更。
- アプリケーション特有のイベントのパブリッシュ

- SNMP トラップの生成。

Tool Command Language（Tcl）を使用して記述された、Cisco 定義のサンプル ポリシー実行機能が追加されました。システム ポリシー ディレクトリに格納可能なサンプル ポリシーが提供されました。

Embedded Event Manager 2.1

EEM 2.1 では、いくつかの新機能が導入されました。EEM 2.1 は次の新しいイベント ディテクタを追加しました。

- CLI：CLI イベント ディテクタは、正規表現と一致するコマンドライン インターフェイス（CLI）コマンドをスクリーニングします。
- None：None イベントディテクタは、Cisco IOS **event manager run** コマンドが EEM ポリシーを実行したときに、イベントをパブリッシュします。
- OIR：Online Insertion and Removal（OIR）イベント ディテクタは、特定のハードウェアの挿入または削除のイベント発生時にイベントをパブリッシュします。

EEM 2.1 は次のアクションを追加しました。

- Cisco CLI コマンドの実行。
- イベント発生時のシステム情報要求。
- ショート メールの送信。
- 手動による EEM ポリシーの実行。

EEM 2.1 は、新しい **event manager scheduler script** コマンドを使用した、複数の同時実行ポリシーの実行も許可します。SNMP イベントディテクタ比率ベース イベントのサポートは、Tool Command Language（Tcl）を使用してポリシーを作成する機能として提供されます。

Embedded Event Manager 2.1（ソフトウェア モジュール方式）

EEM 2.1（ソフトウェア モジュール）は、Cisco ソフトウェア モジュラリティ イメージでサポートされます。EEM 2.1（ソフトウェア モジュール方式）は、次のイベント ディテクタを追加しました。

- GOLD：Generic Online Diagnostics（GOLD）イベント ディテクタは、GOLD 障害イベントが指定されたカードおよびサブカードで検出されたときにイベントをパブリッシュします。
- System Manager：System Manager イベント ディテクタは、Cisco IOS ソフトウェア モジュール方式プロセスの開始、通常停止、異常停止、および再起動のイベントに対してイベントを生成します。System Manager によって生成されたイベントによって、ポリシーはプロセス再起動のデフォルトの動作を変更できます。

- Watchdog System Monitor（WDSysMon）：Cisco Software Modularity Watchdog System Monitor イベント ディテクタは、Cisco IOS ソフトウェア モジュール方式プロセスにおける無限ループ、デッドロック、メモリリークを検出します。

EEM 2.1 ソフトウェア モジュール方式では、プロセスに対する EEM 信頼性メトリック データの表示機能が追加されました。

Note EEM 2.1 ソフトウェア モジュール方式イメージは、Resource イベント ディテクタおよび RF イベント ディテクタを EEM 2.2 に追加しましたが、EOT イベント ディテクタ、またはトラッキング対象オブジェクトの読み込みおよび設定のアクションをサポートしません。

Embedded Event Manager 2.2

EEM 2.2 で、いくつかの新機能が導入されました。EEM 2.2 では次のイベント ディテクタが追加されました。

- Enhanced Object Tracking：Enhanced Object Tracking イベント ディテクタは、トラッキング対象オブジェクトが変更されたときにイベントをパブリッシュします。拡張オブジェクト トラッキングは、トラッキング対象オブジェクトと、トラッキング対象オブジェクトが変更されたときにクライアントが実施するアクションとを全面的に分離します。
- Resource：Resource イベント ディテクタは、Embedded Resource Manager（ERM）が、指定されたポリシーのイベントをレポートしたときにイベントをパブリッシュします。
- RF：Redundancy Framework（RF）イベント ディテクタは、デュアル ルートプロセッサ（RP）システムにおける同期の間に、1 つ以上の RF イベントが発生したときにイベントをパブリッシュします。RF イベント ディテクタは、デュアル RP システムが一方の RP からもう一方の RP に継続的にスイッチしている（ピンポン状態と呼ばれる）ときもイベントを検出できます。

EEM 2.2 では次のアクションが追加されました。

- トラッキング対象オブジェクトの状態の読み取り。
- トラッキング対象オブジェクトの状態の設定。

Embedded Event Manager 2.3

EEM 2.3 は、Cisco Catalyst 6500 シリーズ スイッチでサポートされ、その製品での汎用オンライン診断（GOLD）イベント ディテクタのための拡張が追加されています。

- **event gold** コマンドは、GOLD テスト失敗および条件への対応を改善するための **action-notify**、**testing-type**、**test-name**、**test-id**、**consecutive-failure**、**platform-action**、および **maxrun** キーワードが追加され、拡張されました。

- 次のプラットフォーム全体の GOLD イベント ディテクタ情報には、新しい読み込み専用 EEM 組み込み環境変数を通じてアクセスできます。
  - 起動診断レベル
  - カード インデックス、名前、シリアル番号
  - ポート数
  - テスト数
- 次のテスト固有 GOLD イベント ディテクタ情報は、新しい読み込み専用 EEM 組み込み環境変数（EEM アプレットだけが利用可能）を通じてアクセスできます。
  - テスト名、属性、総実行回数
  - テストごと、ポートごと、またはデバイスごとのテスト結果
  - 合計障害カウント、最終障害時間
  - エラー コード
  - 連続的障害の発生

これらの拡張の結果、オートメーションと障害検出が改善され、平均修復時間（MTTR）が削減され、可用性が向上しました。

Embedded Event Manager 2.4

EEM 2.4 は次のイベント ディテクタを追加しました。

- SNMP Notification：SNMP 通知イベント ディテクタには、デバイスが受信した SNMP トラップおよび SNMP インフォーム メッセージを代行受信する機能があります。SNMP 通知イベントは、受信 SNMP トラップまたは SNMP インフォーム メッセージが指定された値に一致するか、指定されたしきい値を超えたときに生成されます。
- RPC：リモート プロシージャ コール（RPC）イベント ディテクタには、EEM ポリシーをセキュア シェル（SSH）を使用して暗号化された接続経由でデバイスの外から起動する機能があります。RPC イベント ディテクタは、XML ベースのメッセージ交換に Simple Object Access Protocol（SOAP）データ エンコーディングを使用します。このイベント ディテクタは、EEM ポリシーの実行および SOAP XML フォーマット化された応答内の出力の受信に使用できます。

EEM 2.4 は、次のイベント ディテクタに拡張を追加しました。

- インターフェイス カウンタ 比率ベース トリガー：この機能によって、インターフェイス イベントが期間中の変更の比率に基づいてトリガーされる機能が追加されました。entry 値または exit 値の両方に対して比率が指定できます。この機能は、現在、SNMP イベント ディテクタに存在する比率ベースの機能をコピーします。
- SNMP デルタ値：モニターリング期間の開始時のモニター対象オブジェクト識別子（OID）の値と、イベントがパブリッシュされた時点での実際の OID の差が、SNMP イベント ディテクタとインターフェイス カウンタ イベント ディテクタの両方の **event reqinfo** データで提供されます。

EEM 2.4 は次のアクションを追加しました。

- 複数イベントのサポート：複数のイベントを実行する機能が導入されました。さらに、**show event manager** コマンドは複数のイベントを表示するように拡張されました。
- パラメータのサポート：パラメータ引数が **event manager run** コマンドに追加されました。最大 15 個のパラメータを使用できます。
- ジョブ ID と完了ステータスの表示：**show event manager** のコマンドの一部が、ジョブ ID と完了ステータスを表示するように拡張されました。
- バイトコードのサポート：Tcl 8 は、特殊なバイトコード言語（BCL）を定義し、Tcl スクリプトを BCL に変換する Just-In-Time コンパイラを備えています。バイト シーケンスが「virtual machine」、Tcl_ExecuteByteCode()、または TEBC によって、必要に応じて短縮して実行されます。現在、EEM は、ユーザー ポリシーのファイル拡張子として、*.tcl を、また、システム ポリシーのファイル拡張子として *.tm を認めています。bytecode スクリプトの Tcl 標準拡張子は、*.tbc です。現在、EEM は *.tbc を有効な EEM ポリシーとして認めます。
- 登録置換拡張：単一の環境変数によるイベント登録文での複数のパラメータの置換をサポートします。
- Tcl パッケージのサポート

Embedded Event Manager 3.0

EEM 3.0 は次の新しいイベント ディテクタを追加しました。

- Custom CLI：Custom CLI イベント ディテクタは、既存の CLI コマンド構文を追加、拡張するためにイベントをパブリッシュします。
- Routing：Routing イベント ディテクタは、ルーティング情報ベース（RIB）のルート エントリが変化したときにイベントをパブリッシュします。
- NetFlow：NetFlow イベント ディテクタは、NetFlow イベントがトリガーされたときにイベントをパブリッシュします。
- IP SLA：IP SLA イベント ディテクタは、IP SLA 応答がトリガーされたときにイベントをパブリッシュします。

EEM 3.0 は、次の機能を追加しました。

- クラスベース スケジューリング：EEM ポリシーは、登録されるときに **class** キーワードを使用してクラスが割り当てられます。クラスなしで登録された EEM ポリシーは、デフォルト クラスに割り当てられます。
- 高パフォーマンス Tcl ポリシー：**event_completion**、**event_wait**、および **event_completion_with_wait** の 3 つの新しい Tcl コマンドが導入されました。
- インタラクティブ CLI サポート：同期アプレットが、ローカル コンソール（TTY）とのインタラクションをサポートするように拡張されました。2 つの新しい IOS コマンド、

action gets と **action puts,** が導入され、ユーザーがコンソールに直接入力し、それを表示できるようになりました。

- アプレット用の可変ロジック：EEM アプレット用の可変ロジック機能は、EEM アプレット内に条件付きロジックを適用する機能を追加します。条件付きロジックは、アプレット内のアクションのフローを条件式に従って変更する制御構造を追加します。
- デジタル署名サポート：新しい API は、Tcl スクリプトの実行の前に、スクリプトが Cisco によって署名されていることを確認するために、Tcl スクリプトのデジタル署名検証を実行します。
- 電子メールサーバーの認証のサポート：オプションのユーザー名とパスワードを含めるように、**action mail** コマンドが変更されました。
- SMTP IPv6 サポート：Tcl 電子メールテンプレートに、IPv6 または IPv4 アドレスのいずれかを指定するためのキーワード **sourceaddr** が追加されました。
- SNMP ライブラリの機能拡張：EEM アプレット **action info** と Tcl **sys_reqinfo_snmp** コマンドが拡張され、SNMP getid、inform、trap、および set-type 動作用の機能が組み込まれました。
- SNMP 通知 IPv6 サポート：送信元 IP アドレスと宛先 IP アドレスでの IPv6 アドレスがサポートされます。
- CLI Library XML-PI サポート：異なるシスコ製品間で矛盾のない方法で、IOS コマンドライン インターフェイス（CLI）show コマンドを XML 形式にカプセル化した、プログラム可能なインターフェイスを提供します。XML-PI を使用する場合は、既知のキーワードを使用して IOS show コマンドの出力を Tcl スクリプトから解析できます。正規表現サポートを使用する必要はありません。

Embedded Event Manager 3.1

EEM 3.1 は、新しいイベント ディテクタを追加しました。

- SNMP Object：簡易ネットワーク管理プロトコル（SNMP）Object Trap イベント ディテクタは、指定された SNMP オブジェクト ID（OID）を持つ SNMP トラップが特定のインターフェイスまたはアドレスで発生したときに、値を置き換えるように拡張されました。

EEM 3.1 は、次のイベント ディテクタに拡張を追加しました。

- SNMP Notification：SNMP 通知イベント ディテクタは、出力 SNMP トラップおよび SNMP インフォームを待ち、代行受信できるようになりました。

EEM 3.1 は、次のアクションに拡張機能を追加しました。

- ファシリティの指定：**action syslog** コマンドが拡張され、syslog ファシリティを指定できるようになりました。

EEM 3.1 は、次の機能を追加しました。

- 登録されたポリシーの簡単な説明を作成するための機能を提供：ポリシーを簡単な説明とともに Cisco IOS CLI と Tcl ポリシーに登録するための新しい **description** コマンドが導入されました。**show event manager policy available** コマンドと **show event manager policy registered** コマンドが拡張され、登録されたアプレットの説明を表示するための **description** キーワードが追加されました。
- EEM ポリシーでの AAA 認証のバイパスが可能：**event manager application** コマンドが拡張され、承認を提供し、AAA を無効にするキーワードをバイパスできるようになりました。
- CLI ライブラリ拡張機能の導入：CLI ライブラリに 2 つの新しいコマンドが提供されました：**cli_run** および **cli_run_interactive**。

Embedded Event Manager 3.2

EEM 3.2 は次の新しいイベント ディテクタを追加しました。

- ネイバー探索：ネイバー探索イベントディテクタによって、次の場合に自動ネイバー検出に応答するポリシーをパブリッシュできます。
  - Cisco Discovery Protocol（CDP）のキャッシュ エントリが追加、削除、または更新された場合。
  - リンク層検出プロトコル（LLDP）のキャッシュ エントリが追加、削除、または更新された場合。
  - インターフェイスのリンク ステータスが変更された場合。
  - インターフェイスのライン ステータスが変更された場合。
- ID：ID イベント ディテクタは、AAA の許可および認証が成功した場合、障害が発生した場合、またはポート上で通常のユーザー トラフィックの送信が許可された後にイベントを生成します。
- Mac-Address-Table：Mac-Address-Table イベント ディテクタは、MAC アドレスが MAC アドレス テーブルで学習された場合にイベントを生成します。

Note Mac-Address-Table イベント検出器は、スイッチ プラットフォームでだけサポートされており、MAC アドレスが学習されたレイヤ 2 インターフェイスだけで使用できます。レイヤ 3 インターフェイスはアドレスを学習しません。デバイスは通常、学習された MAC アドレスの EEM を通知する必要がある mac-address-table インフラストラクチャをサポートしません。

EEM 3.2 では、新しいイベント ディテクタで動作するアプレットをサポートするための新しい CLI コマンドも導入されています。

Embedded Event Manager 4.0

EEM 4.0 では、次の新機能が導入されます。

- EEM 電子メール アクションの機能
  - SMTP 電子メールアクションの TLS サポート：新しいオプションの **secure** キーワードが、**tls** および **none** キーワードとともに、**action mail** CLI に追加されました。対応する Tcl ポリシーには更新はありません。
  - SMTP 電子メールアクションのカスタムポート：新しいオプションの **port** キーワードが **action mail** CLI に追加されました。Tcl ポリシーでは、電子メール テンプレートに行を追加することでポート番号を指定できます。
- EEM セキュリティの機能拡張
  - チェックサムベースのスクリプトの整合性：デジタル署名がサポートされていない、または使用できない場合、ユーザーは引き続き Unix コマンド **openssl sha1** を使用して、TCL ポリシーにいくつかの基本的な整合性チェックを適用できます。新しいオプションの **checksum**、**md5**、および **sha-1** キーワードが **event manager policy** コマンドに追加されました。
  - サードパーティのデジタル署名のサポート：署名を確認するには、Tcl セキュア モードとトラストポイントを TCL スクリプトに関連付ける必要があります。
  - スクリプト所有者の識別：ポリシーが正常にデジタル署名に登録されている場合は、**show event manager policy registered** コマンドを使用して、show 出力で **Dsig** キーワードを確認することで、ポリシーの所有者（または署名者）を識別できます。
  - リモート Tcl ポリシーの登録：新しいオプションの **remote** キーワードが **event manager policy** コマンドに追加されました。
- EEM リソース管理
  - リソース消費量のスロットリング：新しいオプションの **resource-limit** キーワードが **event manager scheduler** コマンドに追加されました。
  - イベントごとにトリガーされるポリシーのレート制限：新しいオプションの **rate-limit** キーワードが **event syslog** コマンドに追加されました。
- EEM ユーザービリティの機能拡張
  - EEM アプレットアクションでのファイル操作：新しい CLI **action file** が追加され、ファイルを選択できるようになりました。
  - **show event manager statistics** EXEC コマンドを使用して、キューサイズ、ドロップされたイベント、および実行時間の統計情報を追跡するために、EEM に新しいフィールドが追加されました。イベントマネージャのキューカウンタをクリアするために、一連の新しい clear コマンド（**clear event manager detector counters** および **clear event manager server counters**）が導入されました。
- EEM イベント ディテクタの機能拡張
  - CLI イベント ディテクタの機能拡張：ユーザーがイベント CLI コマンドを入力したセッションを検出する機能を提供します。4 つの新しいキーワードと組み込み環境変数：**username**、**host**、**privilege**、および **tty** が **event cli** アプレットに、event_reqinfo アレイ名が **event_register_cli** イベントディテクタに追加されました。**show event manager detector** EXEC コマンドも、この機能強化を反映するように変更されました。

- Syslog イベント ディテクタの機能拡張：特定のログ メッセージ フィールドで文字列の照合を実行するオプションを提供します。4 つの新しいキーワード（**facility**、**mnemonic**、**sequence**、および **timestamp** キーワード）が、**action syslog** コマンド、**event syslog** コマンド、および **event_register_syslog** イベントディテクタに追加されました。**show event manager detector** EXEC コマンドも、この機能強化を反映するように変更されました。

Cisco IOS Release ごとの利用可能な EEM イベント ディテクタ

EEM は、イベントディテクタと呼ばれるソフトウェア プログラムを使用して、EEM イベントの発生したときを判断します。一部のイベント ディテクタは、すべての Cisco IOS Release で利用できますが、イベントディテクタの多くは、特定のリリースに導入されています。次の表を使用して、特定の Cisco IOS リリースで使用可能なイベント ディテクタを特定します。ブランク エントリ（--）は、そのイベント ディテクタが利用できないことを示します。「Yes」の文字はイベント ディテクタが利用できることを示します。この表に示されているイベント ディテクタは、同じ Cisco IOS リリース トレインの最新のリリースでサポートされています。各イベントディテクタの詳細については、「Embedded Event Manager の概要」の章のイベント ディテクタの概念を参照してください。

***Table 48: Cisco IOS Release* ごとのイベント ディテクタの可用性**

イベント ディテクタ	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS ソフトウェア モジュール方式	12.2(33)SXH	12.4(20)T 12.2(33)SXI	12.4(22)T 12.2(33)SRE	15.0(1)M 15.1(3)T	12.2(54)SY	15 E XE 3E
Application-Specific	対応	対応	対応	対応	対応	対応	対応	対応	対応	対応
CLI	--	対応	対応	対応	対応	対応	対応	対応	--	対応
Counter	対応	対応	対応	対応	対応	対応	対応	対応	対応	対応
Custom CLI	--	--	--	--	--	--	対応	対応	--	--
Enhanced Object Tracking	--	--	対応	--	対応	対応	対応	対応	--	--
Environmental	--	--	--	--	--	--	--	--	--	対応
GOLD	--	--	--	対応	対応	対応	対応	対応	--	対応
Identity	--	--	--	--	--	--	--	対応	対応	対応
Interface Counter	対応	対応	対応	対応	対応	対応	対応	対応	--	対応
IPSLA	--	--	--	--	--	--	対応	対応	--	対応

イベント ディテクタ	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS ソフトウェアモジュール方式	12.2(33)SXH	12.4(20)T 12.2(33)SXI	12.4(22)T 12.2(33)SRE	15.0(1)M 15.1(3)T	15.1(2)SY	15 E XE 3E
Mac-Address-Table	--	--	--	--	--	--	--	対応	対応	対応
Neighbor Discovery	--	--	--	--	--	--	--	対応	対応	対応
NF	--	--	--	--	--	--	対応	対応	--	--
なし	--	対応	対応	対応	対応	対応	対応	対応	対応	対応
OIR	--	対応	対応	対応	対応	対応	対応	対応	対応	対応
Resource	--	--	対応	対応	対応	対応	対応	対応	--	--
RF	--	--	対応	対応	対応	対応	対応	対応	--	対応
Routing	--	--	--	--	--	--	対応	対応	--	対応
RPC	--	--	--	--	--	対応	対応	対応	対応	--
SNMP	対応	対応	対応	対応	対応	対応	対応	対応	--	対応
SNMP Proxy	--	--	--	--	--	--	--	--	対応	--
SNMP Notification	--	--	--	--	--	対応	対応	対応	--	対応
SNMP Object	--	--	--	--	--	--	--	対応	--	対応
Syslog	対応	対応	対応	対応	対応	対応	対応	対応	対応	対応
System Manager	--	--	--	対応	対応	対応	対応	対応	対応	--
Timer	対応	対応	対応	対応	対応	対応	対応	対応	対応	対応
IOSWDSysMon（Cisco IOS Watchdog）	対応	対応	対応	対応	対応	対応	対応	対応	--	対応
WDSysMon（Cisco IOS Software Modularity Watchdog）	--	--	--	対応	--	--	--		--	--

イベント検出器

Embedded Event Manager（EEM）は、イベント ディテクタと呼ばれるソフトウェア プログラムを使用して、EEMイベントの発生したときを判断します。イベントディテクタは、モニターされるエージェント（たとえば、簡易ネットワーク管理プロトコル（SNMP））と、アクションが実施される EEM ポリシーの間のインターフェイスを提供する、独立したシステムです。一部のイベント ディテクタは、すべての Cisco IOS Release で利用できますが、イベント ディテクタの多くは、特定のリリースに導入されています。各 Cisco IOS Release でサポートされるイベント ディテクタの詳細については、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」または「Tcl を使用した Embedded Event Manager ポリシーの記述」の章の Cisco IOS Release ごとの利用可能な EEM イベント ディテクタについての記述を参照してください。EEM には次のイベント ディテクタがあります。

Application-Specific イベント ディテクタ

Application-Specific イベント ディテクタによって、任意の Embedded Event Manager ポリシーがイベントをパブリッシュできます。EEM ポリシーがイベントをパブリッシュするとき、任意のイベントタイプで、EEMサブシステム番号798を使用する必要があります。既存のポリシーがサブシステム798と指定されたイベントタイプに対して登録されている場合、同じイベントタイプの別のポリシーは、指定されたイベントがパブリッシュされたときに第1のポリシーをトリガーして実行します。

CLI イベント ディテクタ

CLIイベントディテクタは、コマンドライン インターフェイス（CLI）コマンドを正規表現に一致するかスクリーニングします。一致が見つかったとき、イベントがパブリッシュされます。コマンドが正常に解析されたあと、コマンドが実施される前に、完全に展開されたCLIコマンドで一致ロジックが実施されます。CLIイベントディテクタは次の3種類のパブリッシュモードをサポートします。

- CLIイベントの同期パブリッシング：CLIコマンドは、EEMポリシーが終了するまで実行されません。EEMポリシーは、コマンドが実行されるかどうかをコントロールできます。読み取り/書き込み変数 _exit_status では、ポリシー 終了時に同期イベントからトリガーされたポリシーの終了ステータスを設定できます。_exit_status が 0 の場合、コマンドはスキップされ、_exit_status が 1 の場合はコマンドが実行されます。
- CLI イベントの非同期パブリッシング：CLI イベントは、パブリッシュされ、続いて CLI コマンドが実行されます。
- CLI イベントの非同期パブリッシングかつコマンド スキップ：CLI イベントがパブリッシュされますが、CLI コマンドは実行されません。

Counter イベント ディテクタ

Counter イベント ディテクタは、名前付きカウンタが指定されたしきい値を超えたときにイベントをパブリッシュします。カウンタ処理に影響を与える関係タスクが 2 つ以上あります。Counterイベントディテクタは、カウンタを変更でき、1つ以上のサブスクライバは、イベント

をパブリッシュする条件を定義します。カウンタイベントがパブリッシュされた後、カウンタモニターリングロジックをリセットして、すぐにカウンタの監視を開始できます。また、別のしきい値（exit 値と呼ばれる）を超えたときにリセットすることもできます。

Custom CLI イベント ディテクタ

Custom CLI イベント ディテクタは、既存の CLI コマンド構文を追加、拡張するためにイベントをパブリッシュします。特別なパーサー キャラクタである Tab、?（疑問符）、および Enter が入力された場合、パーサーは処理のために入力を Custom CLI イベント ディテクタに送信します。（疑問符）、および Enter が入力された場合、パーサーは処理のために入力を Custom CLI イベント ディテクタに送信します。続いて Custom CLI イベント ディテクタは、この入力を登録された文字列と比較して、新しい、または拡張された CLI コマンドかどうかを判断します。一致すると、カスタム CLI イベント ディテクタが適切なアクションを実行します。たとえば、? が入力された場合はコマンドのヘルプを表示する、タブが入力された場合はコマンド全体を表示する、Enter が入力された場合はコマンドを実行するなどです。一致しなかった場合は、パーサーはコントロールを回復し、通常どおりに情報を処理します。

Enhanced Object Tracking イベント ディテクタ

Enhanced Object Tracking（EOT）イベント ディテクタは、トラッキング対象のオブジェクトのステータスが変更されたときイベントをパブリッシュします。オブジェクト トラッキングは、当初、ユーザーが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できるだけの単純なトラッキングメカニズムとして、ホットスタンバイ ルータ プロトコル（HSRP）に導入されました。インターフェイスのラインプロトコル ステートがダウンになった場合、デバイスの HSRP 優先度は削減され、より高い優先度のもう 1 つの HSRP デバイスがアクティブになることができます。

オブジェクト トラッキングはトラッキング対象オブジェクトと、トラッキング対象オブジェクトが変更されたときにクライアントが実施するアクションとを全面的に分離するように拡張されました。したがって、HSRP、仮想ルータ冗長プロトコル（VRRP）、または Gateway Load Balancing Protocol（GLBP）などの複数のクライアントが、トラッキング プロセスの対象を登録でき、同一オブジェクトをトラッキング可能であり、さらに、オブジェクト変更時に異なるアクションを実行できます。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番号で識別されます。クライアント プロセスは、この番号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、値の変更を確認します。トラッキング対象オブジェクトの変更は、すぐに、または指定された遅延後に、対象のクライアント プロセスに通知されます。オブジェクトの値は、アップまたはダウンとして報告されます。

拡張オブジェクト トラッキングが EEM と統合され、EEM は追跡対象オブジェクトのステータス変更を報告して、拡張オブジェクト トラッキングが EEM オブジェクトを追跡できるようになりました。新しいタイプのトラッキング オブジェクト、スタブ オブジェクトが作成されます。現在追跡対象オブジェクトを操作できるようにしている既存の CLI コマンドを使用して、スタブ オブジェクトを操作できます。

Generic Online Diagnostics（GOLD）イベント ディテクタ

GOLD イベント ディテクタは、GOLD 障害イベントが指定されたカードおよびサブカードで検出されたときにイベントをパブリッシュします。

Interface Counter イベント ディテクタ

Interface Counter イベント ディテクタは、指定されたインターフェイスの汎用 Cisco IOS インターフェイス カウンタが、定義されたしきい値を超えたときにイベントをパブリッシュします。しきい値は絶対値か増分値で指定できます。たとえば、増分値を 50 に設定した場合、インターフェイス カウンタが 50 増えると、イベントがパブリッシュされます。

インターフェイス カウンタ イベントがパブリッシュされた後、インターフェイス カウンタ モニターリング ロジックは 2 つの方法でリセットされます。インターフェイス カウンタは、別のしきい値（exit 値と呼ばれる）を超えたとき、または、期間の経過が発生したときにリセットされます。

IP SLA イベント ディテクタ

IP SLA イベント ディテクタは、IP SLA 応答がトリガーされたときにイベントをパブリッシュします。

NetFlow イベント ディテクタ

NetFlowイベントディテクタは、NetFlowイベントがトリガーされたときにイベントをパブリッシュします。

None イベント ディテクタ

None イベントディテクタは、Cisco IOS **event manager run** CLI コマンドが EEM ポリシーを実行すると、イベントをパブリッシュします。EEM は、ポリシーそのものに含まれるイベント仕様に基づいてポリシーをスケジューリングし、実行します。EEM ポリシーは識別される必要があり、手動での実行が許可されるように、**event manager run** コマンドが実行される前に登録される必要があります。

OIR イベント ディテクタ

Online Insertion and Removal（OIR）イベント ディテクタは、次のハードウェアの挿入または削除のいずれかのイベント発生時にイベントをパブリッシュします。

- カードが削除されました。
- カードが挿入されました。

ルートプロセッサ（RP）、ラインカード、またはフィーチャ カードは、OIR イベントでモニターできます。

Resource イベント ディテクタ

Resource イベント ディテクタは、Embedded Resource Manager（ERM）が指定されたポリシーのイベントをレポートしたときにイベントをパブリッシュします。ERMインフラストラクチャは、プロセス間およびシステム内のリソースの枯渇とリソースの依存関係を追跡し、さまざまなエラー状態を処理します。エラー状態は、さまざまなアプリケーション間でリソースを等分に共有することで処理されます。ERM フレームワークは、リソース エンティティに通信メカニズムを提供して、さまざまなロケーションからこれらのリソースエンティティ間での通知が行えるようにします。ERM フレームワークは、CPU およびメモリ関連の問題のデバッグにも役立ちます。ERM は、CPU、バッファ、およびメモリなどのリソースに対してユーザーがしきい値を設定できるようにすることで、スケーラビリティ ニーズを理解するためにシステム リソース使用率をモニターリングします。ERMイベントディテクタは、Cisco ソフトウェアのリソースを監視するためのより望ましい方法ですが、ERM イベント ディテクタはソフトウェア モジュラリティ イメージをサポートしません。ERM の詳細については、「Embedded Resource Manager」の章を参照してください。

RF イベント ディテクタ

Redundancy Framework（RF）イベント ディテクタは、デュアル ルート プロセッサ（RP）システムにおける同期の間に、1 つ以上の RF イベントが発生したときにイベントをパブリッシュします。RF イベント ディテクタは、デュアル RP システムが一方の RP からもう一方の RP に継続的にスイッチしている（ピンポン状態と呼ばれる）ときもイベントを検出できます。

Remote Procedure Call（RPC）イベント ディテクタ

リモート プロシージャ コール（RPC）イベント ディテクタには、EEM ポリシーをセキュア シェル（SSH）を使用して暗号化された接続経由でデバイスの外から起動する機能があります。RPC イベント ディテクタは、XML ベースのメッセージ交換に Simple Object Access Protocol（SOAP）データ エンコーディングを使用します。このイベント ディテクタは、EEM ポリシーの実行および SOAP XML フォーマット化された応答内の出力の受信に使用できます。

ルーティング イベント ディテクタ

ルーティング イベント ディテクタは、ルーティング情報ベース（RIB）のルート エントリが変化したときにイベントをパブリッシュします。

SNMP イベント ディテクタ

SNMP イベント ディテクタによって、標準 SNMP MIB オブジェクトを監視し、オブジェクトが指定された値と一致するとき、または指定されたしきい値を超えたときにイベントを生成することができます。

SNMP 通知イベント ディテクタ

SNMP 通知イベント ディテクタには、デバイスが受信した SNMP トラップおよび SNMP インフォーム メッセージを代行受信する機能があります。SNMP 通知イベントは、受信または送信 SNMP トラップまたは SNMP インフォーム メッセージが指定された値に一致するか、指定さ

れたしきい値を超えたときに生成されます。SNMPイベントディテクタは、送信SNMPトラップおよびSNMPインフォームを待ち、代行受信できます。

SNMP Object イベント ディテクタ

簡易ネットワーク管理プロトコル（SNMP）Object Trapイベント ディテクタは、指定されたSNMPオブジェクトID（OID）を持つSNMPトラップが特定のインターフェイスまたはアドレスで発生したときに、値を置き換えるように拡張されました。

syslog イベント ディテクタ

syslogイベントディテクタは、正規表現パターンマッチに対してsyslogメッセージをスクリーニングできます。選別されたメッセージをさらに限定し、指定された時間内に特定の回数の発生を記録するように要求できます。指定されたイベント基準での一致により、設定されたポリシー処理がトリガーされます。

System Manager イベント ディテクタ

System Managerイベント ディテクタは、Cisco IOS ソフトウェア モジュール方式プロセスの開始、通常停止、異常停止、および再起動のイベントに対してイベントを生成します。System Managerによって生成されたイベントによって、ポリシーはプロセス再起動のデフォルトの動作を変更できます。

Timer イベント ディテクタ

timerイベント ディテクタは、次の4種類のタイマーのイベントをパブリッシュします。

- absolute-time-of-day タイマーは、指定された絶対的な日時が発生したとき、イベントをパブリッシュします。
- countdown タイマーは、タイマーがカウント ダウンしてゼロ（0）になったときにイベントをパブリッシュします。
- watchdog タイマーは、タイマーがカウント ダウンしてゼロ（0）になったときにイベントをパブリッシュし、自動的にタイマーを初期値にリセットして、再びカウントダウンを開始します。
- CRON タイマーは、UNIX標準CRON仕様を使用してイベントをパブリッシュするときを指定して、イベントをパブリッシュします。CRON タイマーは、1分間にイベントを複数回パブリッシュすることはありません。

Cisco IOS の Watchdog System Monitor（IOSWDSysMon）イベント ディテクタ

Cisco IOS Watchdog System Monitorイベント ディテクタは、次のいずれかが発生したときにイベントをパブリッシュします。

- Cisco IOS タスクの CPU 使用率がしきい値を超えたとき。
- Cisco IOS タスクのメモリ使用率がしきい値を超えたとき。

Note Cisco IOS プロセスは、現在、Cisco IOS ソフトウェア モジュール方式プロセスから区別するために、タスクと呼ばれています。

同時に2つのイベントがモニターリングされることがあります。指定されたしきい値を超えるために1つのイベントを必要とするか、両方のイベントを必要とするかを、イベントパブリッシング基準で指定できます。

Cisco IOS Software Modularity の Watchdog System Monitor (WDSysMon) イベント ディテクタ

Cisco IOS Software Modularity Watchdog System Monitor イベント ディテクタは、Cisco IOS ソフトウェア モジュラリティ プロセスにおける無限ループ、デッドロック、メモリ リークを検出します。

各 Cisco IOS リリースで利用可能な EEM アクション

イベントディテクタがイベントを報告したときに実行される是正アクションはCLIベースで、強力なオンデバイスのイベント管理メカニズムを実現します。一部のアクションは、すべてのCisco IOS Release で利用できますが、アクションの多くは、特定のリリースに導入されています。次の表を使用して、特定の Cisco IOS リリースで使用可能なアクションを特定します。ブランク エントリ（--）は、そのアクションが使用できないことを示します。「Yes」のテキストはそのアクションが使用できることを示します。この表に示されているアクションは、同じ Cisco IOS リリース トレインの最新のリリースでサポートされています。各アクションの詳細については、「Embedded Event Manager Overview」の章の Embedded Event Manager アクションの概念を参照してください。

Table 49: 各 ***Cisco IOS*** リリースで利用可能なアクション

アクション	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS ソフトウェア モジュール方式	12.2(33)SXH	12.4(20)T	12.4(22)T	15.0(1)M	15E XE 3E
CLI コマンドの実行	--	対応	対応	対応	対応	対応	対応	対応	対応
CNS イベントの生成	対応	対応	対応	対応	対応	対応	対応	対応	対応
優先化された syslog メッセージの生成	対応	対応	対応	対応	対応	対応	対応	対応	対応
SNMP トラップの生成	対応	対応	対応	対応	対応	対応	対応	対応	対応
手動による EEM ポリシーの実行	--	対応	対応	対応	対応	対応	対応	対応	対応

アクション	12.2(25)S	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	12.4(2)T 12.2(31)SB3 12.2(33)SRB	12.2(18)SXF4 Cisco IOS ソフトウェア モジュール方式	12.2(33)SXH	12.4(20)T	12.4(22)T	15.0(1)M	15E XE 3E
アプリケーション固有のイベントのパブリッシュ	対応	対応	対応	対応	対応	対応	対応	対応	対応
トラッキング対象オブジェクトの状態の読み取り	--	--	対応	--		対応	対応	対応	対応
シスコのソフトウェアのリロード	対応	対応	対応	対応	対応	対応	対応	対応	対応
システム情報の要求	--	対応	対応	対応	対応	対応	対応	対応	対応
ショート メールの送信	--	対応	対応	対応	対応	対応	対応	対応	対応
名前付きカウンタの設定または変更	対応	対応	対応	対応	対応	対応	対応	対応	対応
トラッキング対象オブジェクトの状態の設定	--	--	対応	--		対応	対応	対応	対応
セカンダリ RP へのスイッチ	対応	対応	対応	対応	対応	対応	対応	対応	対応

Embedded Event Manager のアクション

イベントディテクタがイベントを報告したときに実行される是正アクションはCLIベースで、強力なオンデバイスのイベント管理メカニズムを実現します。一部の EEM アクションは、すべての Cisco IOS Release で利用できますが、EEM アクションの多くは、特定のリリースに導入されています。各 Cisco IOS Release でサポートされる EEM アクションの詳細については、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」または「Writing Embedded Event Manager Policies Using Tcl」の章の Cisco IOS Release ごとの利用可能な EEM アクションについての記述を参照してください。EEM がサポートするアクションは、次のとおりです。

- Cisco IOS コマンドライン インターフェイス（CLI）コマンドの実行。
- Cisco CNS デバイスによるアップストリーム処理に対し CNS イベントの生成。
- 名前付きカウンタの設定または変更。

- 完全冗長ハードウェア構成におけるセカンダリ プロセッサへのスイッチング。
- イベント発生時のシステム情報要求。
- ショート メールの送信。
- 手動による EEM ポリシーの実行。
- アプリケーション特有のイベントのパブリッシュ。
- シスコのソフトウェアをリロードします。
- SNMP トラップの生成。
- 優先化された syslog メッセージの生成。
- トラッキング対象オブジェクトの状態の読み取り。
- トラッキング対象オブジェクトの状態の設定。

EEM アクション CLI コマンドには、任意の文字列値が可能で一意の ID である EEM アクション ラベルが含まれます。アクションは、ラベルをソート キーとして使用して、英数字のキーの昇順（辞書順）にソートされ、実行されます。ラベルとして数字を使用している場合は、英数字ソートは、10.0は1.0よりも後ですが、2.0よりも前になることに注意してください。このような場合、01.0、02.0 のような数字を使用する、または頭文字の後に同様の数字を続けることを推奨します。

Embedded Event Manager の環境変数

EEM では、EEM ポリシーに環境変数を使用できます。Tool Command Language（Tcl）では、Tcl スクリプト内のすべてのプロシージャで既知のグローバル変数を定義できます。EEM では、CLI コマンドの **event manager environment** コマンドを使用して、EEM ポリシー内で使用するための環境変数を定義できます。EEM環境変数は、Tclスクリプトの実行前に、Tclグローバル変数に自動的に割り当てられます。Embedded Event Managerに関連する環境変数には次の3 種類があります。

- ユーザー定義：ユーザーが記述したポリシー内の環境変数を作成する場合にユーザーが定義できます。
- シスコ定義：特定のサンプル ポリシーのためにシスコが定義しました。
- シスコ組み込み（EEM アプレット内で利用可能）：シスコが定義し、読み取り専用、または読み取り/書き込み可能です。読み取り専用変数は、アプレットの実行開始前にシステムによって設定されます。単一の読み取りと書き込みの変数 _exit_status では、同期イベントからトリガーされたポリシーの終了ステータスを設定できます。

シスコ定義環境変数（次の表を参照）およびシスコ システム定義環境変数は、1つの特定イベント ディテクタまたはすべてのイベント ディテクタに適用できます。ユーザー定義の環境変数、またはサンプルポリシーでCiscoによって定義された環境変数は、**event manager environment** コマンドを使用して設定されます。EEM ポリシーで使用される変数は、ポリシーを登録する

前に定義する必要があります。Tcl ポリシーには、ポリシーの実行前に必要な環境変数がすべて定義されているかどうかを確認するために定義される「Environment Must Define」と呼ばれるセクションがあります。

シスコ組み込み環境変数は、シスコ定義の環境変数のサブセットです。組み込み変数は、EEM アプレットでだけ利用できます。組み込み変数は、読み込み専用であるか、または読み込みおよび書き込み用のいずれかです。これらの変数は、1 個の特定のイベント ディテクタまたはすべてのイベント ディテクタに適用されます。シスコ システム定義変数の詳細と、一覧表については、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章を参照してください。

Note シスコ定義環境変数は、アンダースコア（_）で始まります。付ける名前の競合を防止するため、ユーザー間での同じ命名規則の使用は避けることを強く推奨します。

次の表に、サンプル EEM ポリシーで使用されるシスコ定義変数の説明を示します。一部の環境変数は、対応サンプル ポリシーで実行のために指定される必要はありません。これらは任意として示されています。

Table 50: **シスコ定義環境変数と例**

環境変数	説明	例
_config_cmd1	実行される 1 番めのコンフィギュレーション コマンド。	**interface Ethernet1/0**
_config_cmd2	（任意）実行される 2 番めのコンフィギュレーション コマンド。	**no shutdown**
_crash_reporter_debug	（任意）tm_crash_reporter.tcl のデバッグ情報がイネーブルであるかどうかを決定する値。	1
_crash_reporter_url	クラッシュ レポートが送信される URL 位置。	http://www.yourdomain.com/fm/interface_tm.cgi

環境変数	説明	例
_cron_entry	ポリシーが実行されるときを決定する CRON 仕様。cron エントリを指定する方法の詳細については、「Tcl を使用した Embedded Event Manager ポリシーの記述」の章を参照してください。	0-59/1 0-23/1 * * 0-7
_email_server	E メール送信に使用されるシンプル メール転送プロトコル（SMTP）メール サーバー。	mailserver.yourdomain.com
_email_to	E メールの送信先アドレス。	engineer@yourdomain.com
_email_from	E メールの送信元アドレス。	devtest@yourdomain.com
_email_cc	E メールのコピーの送信先アドレス。	manager@yourdomain.com
_email_ipaddr	受信者の送信元 IP アドレス。	209.165.201.1 または（IPv6 アドレス）2001:0DB8::1
_info_snmp_oid	SNMP オブジェクト ID。	1.3.6.1.2.1.2 または iso.internet.mgmt.mib-2.interfaces
_info_snmp_value	割り当てられた SNMP データ エレメントの値文字列。	
_show_cmd	ポリシーの実行時に実行される CLI **show** コマンド。	**show version**
_syslog_pattern	ポリシー実行時を決定するために syslog メッセージを比較するために使用する正規表現パターン マッチ文字列。	.*UPDOWN.*FastEthernet 0/0.*

環境変数	説明	例
_tm_fsys_usage_cron	（オプション）**event_register** キーワード拡張機能で使用されるCRON仕様。指定されない場合、_tm_fsys_usage.tcl ポリシーが1分に1回、トリガーされます。	0-59/1 0-23/1 * * 0-7
_tm_fsys_usage_debug	（任意）この変数が値1に設定された場合、システムのすべてのエントリのディスク使用率情報が表示されます。	1
_tm_fsys_usage_freebytes	（任意）システムまたは特定のプレフィックスの空きバイト数しきい値。空きスペースが所定の値を下回ると、警告が表示されます。	disk2:98000000
_tm_fsys_usage_percent	（任意）システムまたは特定のプレフィックスのディスク使用割合しきい値。ディスク使用割合が所定の割合を超えると、警告が表示されます。指定されない場合、すべてのシステムのデフォルトのディスク使用割合は、80% です。	nvram:25 disk2:5

Embedded Event Manager ポリシーの作成

EEM は、Cisco ソフトウェア システムで障害またはその他のイベントが発生したときに EEM ポリシーエンジンが通知を受け取るポリシードリブンプロセスです。Embedded Event Manager ポリシーは、システムの現在の状態に基づいて回復を実行し、該当するイベントのポリシーに指定されたアクションを実行します。回復アクションはポリシーが実行されたときにトリガーされます。

いくつかの EEM CLI 設定と **show** コマンドはありますが、EEM はポリシーの作成を通じて実装されます。EEM ポリシーは、イベントおよびイベントが発生した場合に行う処理を定義するエンティティです。EEMポリシーにはアプレットとスクリプトの2つのタイプがあります。

アプレットは、CLI 設定に定義された、ポリシーの単純な形式です。スクリプトは、Tcl で記述された、ポリシーの形式です。

EEM ポリシーの作成には次の項目が含まれます。

- ポリシーが実行されるイベントの選択。
- イベントの記録およびイベントへの対応に関連付けられたイベント ディテクタ オプションの定義。
- 必要に応じて、環境変数の定義。
- イベント発生時に実行されるアクションの選択。

EEM ポリシーの作成には 2 つの方法があります。第 1 の方法は、CLI コマンドを使用してアプレットを記述する方法で、第 2 の方法は、Tcl スクリプトを記述する方法です。シスコは、Tcl に EEM ポリシー開発を促進する Tcl コマンド拡張機能を加えました。スクリプトは、ネットワーキング デバイスの外部で ASCII エディタを使用して定義します。続いてスクリプトはネットワーキング デバイスにコピーされ EEM に登録されます。Embedded Event Manager にポリシーが登録されると、ソフトウェアはポリシーを調べ、指定されたイベントの発生時に起動するために登録します。ポリシーは、未登録または中断にできます。両方のタイプのポリシーとも、ネットワークの EEM 実装に使用できます。

Cisco IOS CLI を使用して EEM ポリシーを記述する方法については、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章を参照してください。

Tcl を使用して EEM ポリシーを記述する方法の詳細については、「Writing Embedded Event Manager Policies Using Tcl」の章を参照してください。

次の作業

- Cisco IOS CLI を使用して EEM ポリシーを記述するには、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章を参照してください。
- Tcl を使用して EEM ポリシーを記述する方法については、「Writing Embedded Event Manager Policies Using Tcl」の章を参照してください。

Embedded Event Manager 4.0 の機能情報の概要

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 51: Embedded Event Manager 4.0 の機能情報の概要

機能名	リリース	機能情報
Embedded Event Manager 1.0		EEM 1.0 は、Embedded Event Manager アプレット作成を SNMP イベント ディテクタ Syslog イベント ディテクタとともに追加しました。EEM 1.0 は、次のアクションも追加しました。優先化された syslog メッセージの生成、Cisco CNS デバイスによるアップストリーム処理に対し CNS イベントの生成、Cisco IOS ソフトウェアのリロード、および完全冗長ハードウェア構成におけるセカンダリ プロセッサへのスイッチング。 次のコマンドがこの機能で導入されました。**action cns-event**、 **action force-switchover**、 **action reload**、**action syslog**、 **debug event manager**、 **event manager applet**、 **event snmp**、 **event syslog**、 **show event manager policy registered**。
Embedded Event Manager 2.0		EEM 2.0 は、Application-Specific イベント ディテクタ、Counter イベント ディテクタ、Interface Counter イベント ディテクタ、Timer イベント ディテクタ、および watchdog イベント ディテクタを追加しました。新しいアクションには、名前付きカウンタの変更、アプリケーション固有イベントのパブリッシュ、SNMP トラップの生成が含まれました。環境変数定義機能、および、Tcl を使用して記述されたサンプル EEM ポリシーの実行機能が追加され、2 個のサンプル ポリシーがソフトウェアに追加されました。 次のコマンドがこの機能で導入されました。**action counter**、 **action publish-event**、**action snmp-trap**、**event application**、**event counter**、**event interface**、**event ioswdsysmon**、**event manager environment**、**event manager history size**、**event manager policy**、**event manager scheduler suspend**、**event timer**, **show event manager environment**、**show event manager history events**、**show event manager history traps**、**show event manager policy available**、**show event manager policy pending**。

機能名	リリース	機能情報
Embedded Event Manager 2.1		EEM 2.1 は複数の新しいイベント ディテクタおよびアクション、EEM ポリシーを手動で起動する新しい機能と複数の共存ポリシーを起動する機能を追加しました。簡易ネットワーク管理プロトコル（SNMP）イベント ディテクタ比率ベース イベントのサポートが、Tool Command Language（Tcl）を使用してポリシーを作成する機能として導入されました。 次のコマンドがこの機能で導入されました。**action cli**、**action counter**、 **action info**、 **action mail**、**action policy**、**debug event manager**、 **event cli**、 **event manager directory user**, **event manager policy**、**event manager run**、**event manager scheduler script**、**event manager session cli username**、**event none**、**event oir**、**event snmp**, **event syslog**、**set**(EEM)、**show event manager directory user**、**show event manager policy registered**、**show event manager session cli username**。
Embedded Event Manager 2.1（ソフトウェア モジュール方式）	12.2(18)SXF4 Cisco IOS ソフトウェアモジュール方式のイメージ	EEM 2.1 ソフトウェア モジュール方式イメージは、GOLD、system manager、および WDSysMon（Cisco IOS Software Modularity watchdog）イベント ディテクタ、および Cisco IOS ソフトウェア モジュール方式プロセスとプロセス メトリックを表示する機能を導入しました。 次のコマンドがこの機能で導入されました。**event gold**、**event process**、**show event manager metric process**。 **Note** EEM 2.1 ソフトウェア モジュール方式イメージは、Resource イベント ディテクタおよび RF イベント ディテクタを EEM 2.2 に追加しましたが、EOT イベント ディテクタ、またはトラッキング対象オブジェクトの読み込みおよび設定のアクションをサポートしません。
Embedded Event Manager 2.2	12.2(31)SB3 12.2(33)SRB	EEM 2.2 は、Enhanced Object Tracking、Resource、および RF イベント ディテクタを追加しました。トラッキング対象オブジェクトの状態の読み取りおよび設定のアクションも追加されました。 この機能により、次のコマンドが導入または変更されました。**action track read**、**action track set**、**default-state**、**event resource**、**event rf**、**event track**、**show track**、**track stub-object**。

機能名	リリース	機能情報
SNMP イベント ディテクタ delta 環境変数		新しい SNMP イベント ディテクタ環境変数、_snmp_oid_delta_val が追加されました。 これはマイナーな拡張です。マイナーな拡張は、通常 Feature Navigator に記載されません。
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB	EEM 2.3 では、Cisco Catalyst 6500 シリーズ スイッチ上の Generic Online Diagnostics (GOLD) イベント ディテクタに関連する新しい機能が追加されました。 **event gold** コマンドは、GOLD テスト失敗および条件への対応を改善するための **action-notify**、**testing-type**、**test-name**、**test-id**、**consecutive-failure**、**platform-action**、および **maxrun** キーワードが追加され、拡張されました。 検出されたイベントのプラットフォーム全体、および、テスト特有の GOLD イベントディテクタ情報へのアクセスを実現するために、読み取り専用変数が **GOLD Event Detector** カテゴリに追加されました。
Embedded Event Manager 2.4	12.2(33)SXI 12.2(33)SRE	EEM 2.4 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが追加されました。 **attribute (EEM)** 、**correlate**、**event manager detector rpc**、**event manager directory user repository**、**event manager update user policy**、 **event manager scheduler clear**、**event manager update user policy**、**event owner**、**event rpc**、**event snmp-notification**、**show event manager detector**、**show event manager version**、**trigger (EEM)**。

機能名	リリース	機能情報
Embedded Event Manager 3.0	12.2(33)SRE	EEM 3.0 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが導入または変更されました。 **action add** 、**action append**、**action break**、**action comment**、**action context retrieve**、**action context save**、**action continue**、**action decrement**、**action divide**、**action else**、**action elseif**、**action end**、**action exit**、**action foreach**、**action gets**、**action if**, **action if goto**、**action increment**、**action info type interface-names**、**action info type snmp getid**、**action info type snmp inform**、**action info type snmp oid**、**action info type snmp trap**、**action info type snmp var**、**action multiply**、**action puts**、**action regexp**、**action set (EEM)**、**action string compare**、**action string equal**、**action string first**、**action string index**、**action string last**、**action string length**、**action string match**、**action string range**、**action string replace**、**action string tolower**、**action string toupper**、**action string trim**、**action string trimleft**、**action string trimright**、**action subtract**、**action while**、**event cli**、**event ipsla**、**event manager detector routing**、**event manager scheduler**、**event manager scheduler clear**、**event manager scheduler hold**、**event manager scheduler modify**、**event manager scheduler release**、**event nf**、**event routing**、**show event manager policy active**、**show event manager policy pending**、および **show event manager scheduler**。
Embedded Event Manager 3.1		EEM 3.1 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが導入または変更されました。**action syslog**、**description (EEM)**、**event manager applet**、**event manager policy**、**event snmp-notification**、**event snmp-object**、**show event manager policy registered** および **show event manager policy available**。
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG	EEM 3.2 で、いくつかの新機能が導入されました。 次のコマンドが導入または変更されました。**debug event manager**、**event identity**、**event mat**、**event neighbor-discovery**、**show event manager detector**。

機能名	リリース	機能情報
Embedded Event Manager 4.0		EEM 4.0 で、いくつかの新機能が導入されました。 次のコマンドが導入または変更されました。**action file**、**action mail**、**action syslog**、**clear event manager detector counters**、**clear event manager server counters**、**event cli**、**event manager policy**、**event manager scheduler**、**event syslog**、**show event manager detector**、**show event manager policy registered**、**show event manager statistics**。

その他の参考資料

EEM に関連する参考資料については、次の各項を参照してください。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
EEM コマンド：コマンド構文の詳細、デフォルト、コマンド モード、コマンド履歴、使用上の注意事項、および例	Cisco IOS Embedded Event Manager のコマンド リファレンス
CLI を使用して Embedded Event Manager ポリシーを記述する	「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章
Tcl を使用して Embedded Event Manager ポリシーを記述する	「Tcl を使用した Embedded Event Manager ポリシーの記述」の章
Embedded Resource Manager	「Embedded Resource Manager」の章

標準

標準	タイトル
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	--

MIB

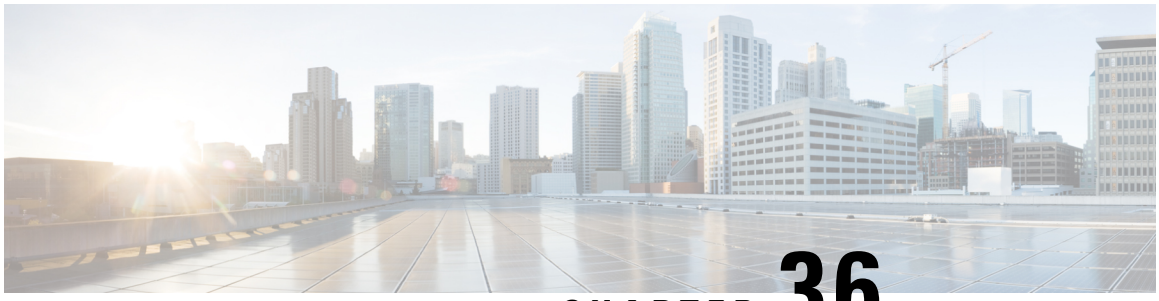
MIB	MIB のリンク
CISCO-EMBEDDED-EVENT-MGR-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
新しい RFC または変更された RFC はサポートされていません。また、既存の RFC に対するサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 36

Writing Embedded Event Manager Policies Using the Cisco IOS CLI

この章では、Cisco IOS コマンドライン インターフェイス（CLI）アプレットを使用して、Cisco IOS ソフトウェア障害およびイベントを処理する Embedded Event Manager（EEM）ポリシーを記述する方法について説明します。EEM は、イベント検出と回復を Cisco IOS 内部で直接行うための分散型でカスタマイズされた手法です。EEM では、イベントをモニタし、イベント発生が検出されたとき、およびしきい値を超えたときに情報通知や是正などの任意のアクションを実施できます。EEM ポリシー エンジンは、障害およびその他のイベントが発生したときに通知を受け取ります。EEM ポリシーは、システムの現在の状態に基づいて回復を実行し、該当するイベントのポリシーに指定されたアクションを実行します。回復アクションはポリシーが実行されたときにトリガーされます。

Cisco IOS CLI を使用した EEM ポリシーの記述に関する前提条件

- EEM ポリシーを記述する前に、「Embedded Event Manager の概要」の章で説明されている概念を十分に理解しておく必要があります。
- **action cns-event** コマンドを使用する場合は、Cisco Networking Services（CNS）イベント ゲートウェイへのアクセスを設定する必要があります。
- **action force-switchover** コマンドを使用する場合は、デバイスでセカンダリ プロセッサを設定する必要があります。

- **action snmp-trap** コマンドを使用した場合、**snmp-server enable traps event-manager** コマンドを有効にして、SNMP トラップが Cisco IOS デバイスから SNMP サーバーに送信されることを許可する必要があります。その他の関連する **snmp-server** コマンドを設定する必要もあります。詳細については、**action snmp-trap** コマンドのページを参照してください。

Cisco IOS CLI を使用した EEM ポリシーの記述について

Embedded Event Manager ポリシー

EEM では、イベントを監視し、監視対象のイベントが発生したときやしきい値を超えたときに情報通知や是正アクションを実施できます。EEM ポリシーは、イベントおよびイベントが発生した場合に行う処理を定義するエンティティです。EEM ポリシーにはアプレットとスクリプトの 2 つのタイプがあります。アプレットは、CLI 設定に定義された、ポリシーの単純な形式です。スクリプトは、Tool Command Language（Tcl）で記述されたポリシーの形式です。

EEM アプレット

EEM アプレットは、イベント スクリーニング基準とイベント発生時に実行するアクションを定義する簡潔な方法です。アプレット コンフィギュレーション モードでは、3 種類のコンフィギュレーション ステートメントがサポートされています。**event** コマンドを使用して実行するアプレットをトリガーするイベント基準を指定し、**action** コマンドを使用して、EEM アプレットがトリガーされるときに実行されるアクションを指定し、**set** コマンドを使用して EEM アプレット変数の値を設定します。現在、_exit_status 変数だけが、**set** コマンドでサポートされます。

アプレット コンフィギュレーション内では、**event** コンフィギュレーション コマンドを 1 つだけが使用できます。アプレット コンフィギュレーション モードが終了し、**event** コマンドが存在しない場合は、このアプレットにイベントが関連付けられていないことを示す警告が表示されます。イベントが指定されない場合、このアプレットは登録されたと見なされません。このアプレットにアクションが割り当てられない場合、イベントはトリガーされますが、アクションは実行されません。1 つのアプレット コンフィギュレーション内で複数の **action** コンフィギュレーション コマンドが使用できます。登録済みのアプレットを表示するには、**show event manager policy registered** コマンドを使用します。

EEM アプレットを修正する前に、アプレット コンフィギュレーション モードを終了するまで既存のアプレットを置き換えられないことに注意してください。アプレット コンフィギュレーション モードでアプレットを修正中であっても、既存のアプレットを実行できます。アプレットを登録解除することなく修正することが安全な方法です。アプレット コンフィギュレーション モードを終了すると、古いアプレットが登録解除され、新しいバージョンが登録されます。

action コンフィギュレーション コマンドは、*label* 引数を使用して一意に識別できます。この引数には任意の文字列値が使用できます。アクションは *label* 引数を使用してソート キーとして、英数字のキーの昇順に並べ替えられ、この順序で実行されます。

Embedded Event Manager は、ポリシーそのものに含まれるイベント仕様に基づいてポリシーをスケジューリングし、実行します。アプレット コンフィギュレーション モードが終了するとき、EEM は、入力された **event** コマンドと **action** コマンドを検査し、指定されたイベントの発生時に実行されるようにアプレットを登録します。

EEM スクリプト

スクリプトは、ネットワーキング デバイスの外部で ASCII エディタを使用して定義します。続いてスクリプトはネットワーキング デバイスにコピーされ EEM に登録されます。Tcl スクリプトは EEM でサポートされます。

EEM では、Tcl を使用して独自のポリシーを記述、実装できます。EEM ポリシーの記述には、次の作業が含まれます。

- ポリシーが実行されるイベントの選択。
- イベントの記録およびイベントへの対応に関連付けられたイベント ディテクタ オプションの定義。
- イベント発生後に実行されるアクションの選択。

シスコは、Tcl に EEM ポリシー開発を促進するキーワード拡張機能の形式を加えました。キーワードの主要なカテゴリでは、検出されたイベント、後続のアクション、ユーティリティ情報、カウンタの値、システム情報が特定されます。Tcl を使用して EEM ポリシーを記述する方法については、「Tcl を使用した Embedded Event Manager ポリシーの記述」の章を参照してください。

EEM アプレットに使用される Embedded Event Manager 組み込み環境変数

EEM 組み込み環境変数は、シスコ定義の環境変数のサブセットです。組み込み変数は、EEM アプレットでだけ利用できます。組み込み変数は、読み込み専用であるか、または読み込みおよび書き込み用のいずれかです。これらの変数は、1 個の特定のイベント ディテクタまたはすべてのイベント ディテクタに適用されます。次の表に、イベント ディテクタおよびサブイベントごとの読み込み専用のシスコ組み込み環境変数の一覧をアルファベット順に示します。

***Table 52: EEM* 組み込み環境変数（読み取り専用）**

環境変数	説明
すべてのイベント	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
_event_id	
_event_type	イベントのタイプ。

環境変数	説明
_event_type_string	イベントをトリガーしたイベントの種類を識別する ASCII 文字列。
_event_pub_sec _event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
_event_severity	イベントの重大度。
Application-Specific イベント ディテクタ	
_application_component_id	イベント アプリケーション コンポーネント ID。
_application_data1	イベントがパブリッシュされたときにアプリケーション固有のイベントに渡される、環境変数の値、文字テキスト、またはその両方の組み合わせ。
_application_data2	イベントがパブリッシュされたときにアプリケーション固有のイベントに渡される、環境変数の値、文字テキスト、またはその両方の組み合わせ。
_application_data3	イベントがパブリッシュされたときにアプリケーション固有のイベントに渡される、環境変数の値、文字テキスト、またはその両方の組み合わせ。
_application_data4	イベントがパブリッシュされたときにアプリケーション固有のイベントに渡される、環境変数の値、文字テキスト、またはその両方の組み合わせ。
_application_sub_system	イベント アプリケーション サブシステム番号。
_application_type	アプリケーションのタイプ。
CLI イベント ディテクタ	
_cli_msg	CLIイベントをトリガーした、完全に展開されたメッセージ。
_cli_msg_count	イベントがパブリッシュされる前にメッセージ一致が発生した回数。
Counter イベント ディテクタ	
_counter_name	カウンタの名前。
_counter_value	カウンタの値。

環境変数	説明
Enhanced Object Tracking イベント ディテクタ	
_track_number	トラッキング対象オブジェクトの数。
_track_state	トラッキング対象オブジェクトの状態（ダウン、またはアップ）。
Generic Online Diagnostics（GOLD）イベント ディテクタ	
_action_notify	GOLD イベント フラグのアクション通知情報（False または True）。
_event_severity	イベントの重大度（Normal、Minor、またはMajor）。
_gold_bl	起動診断レベル（次のいずれかの値）。 • 0：完全診断 • 1：最小診断 • 2：バイパス診断
_gold_card	GOLD 障害イベントが検出されたカード。
_gold_cf *testnum*	連続的な障害。*testnum* はテスト番号。たとえば、**_gold_cf3** は、テスト 3 の連続的な障害の EEM 組み込み環境変数です。
_gold_ci	カード インデックス。
_gold_cn	カードの名前。
_gold_ec *testnum*	テストエラー コード。*testnum* はテスト番号。たとえば、**_gold_ec3** は、テスト 3 のエラー コードの EEM 組み込み環境変数です。
_gold_lf *testnum*	最終障害時間。*testnum* はテスト番号。たとえば、**_gold_lf3** は、テスト 3 の最終障害時間の EEM 組み込み環境変数です。 タイムスタンプの形式は *mmm dd yyyy hh:mm:ss* です。例：Mar 11 2005 08:47:00。
_gold_new_failure	GOLDイベントフラグの新しいテスト障害情報（False または True）。

環境変数	説明
_gold_overall_result	総合診断結果、次のいずれかの値である。 • 0：OK • 3：マイナー エラー • 4：メジャー エラー • 14：結果不明
_gold_pc	ポート数。
_gold_rc *testnum*	テスト総実行回数。*testnum* はテスト番号。たとえば、**_gold_rc3** は、テスト 3 の総実行回数の EEM 組み込み変数です。
_gold_sn	カード シリアル番号。
_gold_sub_card	GOLD 障害イベントが検出されたサブカード。
_gold_ta *testnum*	テスト属性名。*testnum* はテスト番号。たとえば、**_gold_ta3** は、テスト 3 の属性の EEM 組み込み環境変数です。
_gold_tc	テスト数。
_gold_tf *testnum*	合計障害回数。*testnum* はテスト番号。たとえば、**_gold_tf3** は、テスト 3 の合計障害回数の EEM 組み込み変数です。
_gold_tn *testnum*	テストの名前。*testnum* はテスト番号。たとえば、**_gold_tn3** は、テスト 3 の名前の EEM 組み込み環境変数です。
_gold_tr *testnum*	テストの結果。*testnum* はテスト番号。たとえば、**_gold_tr6** はテスト 6 用の EEM 組み込み変数です。テスト 6 はポート単位のテストでも、デバイス単位のテストでもありません。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown

環境変数	説明
_gold_tr *testnum* **d** *devnum*	デバイスごとのテスト結果。*testnum* はテスト番号で、*devnum* はデバイス番号です。たとえば、**_gold_tr3d20** は、テスト 3、デバイス 20 のテスト結果の EEM 組み込み環境変数です。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown
_gold_tr *testnum* **p** *portnum*	ポートごとのテスト結果。*testnum* はテスト番号で、*portnum* はポート番号です。たとえば、**_gold_tr5p20** は、テスト 5、ポート 20 のテスト結果の EEM 組み込み環境変数です。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown
_gold_tt	テストのタイプ。次のうちのいずれかです。 • 1：起動診断 • 2：オンデマンド診断 • 3：スケジュール診断 • 4：モニターリング診断
Interface Counter イベント ディテクタ	
_interface_is_increment	現在のインターフェイスカウンタ値が、絶対値（0）か増分値（1）かを示す値。
_interface_name	モニターされるインターフェイスの名前。
_interface_parameter	モニターされるインターフェイス カウンタの名前。
_interface_value	現在のインターフェイスカウンタ値と比較される値。
None イベント ディテクタ	

環境変数	説明
_event_id	1 であれば挿入イベントを示し、2 であれば削除イベントを示す値。
_none_argc **_none_arg1** **_none_arg2** **_none_arg3** **_none_arg4** **_none_arg5** **_none_arg6** **_none_arg7** **_none_arg8** **_none_arg9** **_none_arg10** **_none_arg11** **_none_arg12** **_none_arg13** **_none_arg14** **_none_arg15**	Extensible Markup Language（XML）Simple Object Access Protocol（SOAP）コマンドからスクリプトに渡されるパラメータ。
OIR イベント ディテクタ	
_oir_event	1 であれば挿入イベントを示し、2 であれば削除イベントを示す値。
_oir_slot	OIR イベントのスロット番号。
Resource イベント ディテクタ	
_resource_configured_threshold	設定されている ERM しきい値。
_resource_current_value	ERM によって報告された、現在の値。
_resource_dampen_time	ERM 減衰時間、ナノ秒単位。
_resource_direction	ERM イベント方向。イベント方向は、アップ、ダウン、または、変更なしのうちのいずれかです。
_resource_level	ERM イベント レベル。イベント レベルは、Normal、Minor、Major、および Critical の 4 つです。
_resource_notify_data_flag	ERM 通知データ フラグ。

環境変数	説明
_resource_owner_id	ERM リソース オーナー ID。
_resource_policy_id	ERM ポリシー ID。
_resource_policy_violation_flag	ERM ポリシー違反フラグ（False または True）。
_resource_time_sent	ERM イベント時間、ナノ秒単位。
_resource_user_id	ERM リソース ユーザー ID。
RF イベント ディテクタ **_rf_event**	0 であれば RF イベントでないことを示し、1 であれば RF イベントであることを示す値。
Remote Procedure Call（RPC）イベント ディテクタ **_rpc_event**	値 0 はエラーがないことを示し、値 1 ～ 83 はエラーを示します。
_rpc_argc **_rpc_arg0** **_rpc_arg1** **_rpc_arg2** **_rpc_arg3** **_rpc_arg4** **_rpc_arg5** **_rpc_arg6** **_rpc_arg7** **_rpc_arg8** **_rpc_arg9** **_rpc_arg10** **_rpc_arg11** **_rpc_arg12** **_rpc_arg13** **_rpc_arg14**	XML SOAP コマンドからアプレットに渡されるパラメータ。
SNMP イベント ディテクタ **_snmp_exit_event**	0 であれば exit イベントでないことを示し、1 であれば exit イベントであることを示す値。

環境変数	説明
_snmp_oid	パブリッシュされるイベントの原因となった SNMP オブジェクト ID。
_snmp_oid_delta_val	現在の SNMP オブジェクト ID の値と、イベントが最後にトリガーされたときの実際の増分差異。
_snmp_oid_val	イベントがパブリッシュされたときの SNMP オブジェクト ID 値。
SNMP 通知イベント ディテクタ	
_snmp_notif_oid	ユーザー指定オブジェクト ID。
_snmp_notif_oid_val	ユーザー指定オブジェクト ID 値。
_snmp_notif_src_ip_addr	SNMP プロトコル データ ユニット（PDU）の発信元 IP アドレス。
_snmp_notif_dest_ip_addr	SNMP PDU の宛先の IP アドレス。
_x_x_x_x_x_x_x(varbinds)	SNMP PDU varbind 情報。
_snmp_notif_trunc_vb_buf	バッファの領域不足から varbind 情報が切り捨てられているかどうかを示します。
syslog イベント ディテクタ	
_syslog_msg	パブリッシュされるイベントの原因となる syslog メッセージ。
System Manager（Process）イベント ディテクタ	
_process_dump_count	Posix プロセスがダンプされた回数。
_process_exit_status	終了時の Posix プロセスの状態。
_process_fail_count	Posix プロセスが失敗した回数。
_process_instance	Posix プロセスのインスタンス数。
_process_last_respawn	最後に再生成された Posix プロセス。
_process_node_name	Posix プロセスのノード名。
_process_path	Posix プロセスのパス。
_process_process_name	Posix プロセスの名前。
_process_respawn_count	Posix プロセスが再生成された回数。

環境変数	説明
Timer イベント ディテクタ	
_timer_remain	タイマーの期限が切れるまでの使用可能時間。 **Note** この環境変数は、CRON タイマーには使用できません。
_timer_time	最後のイベントがトリガーされた時刻。
_timer_type	タイマーのタイプ。
Watchdog System Monitor（IOSWDSysMon）イベント ディテクタ	
_ioswd_node	ルート プロセッサ（RP）レポーティング ノードのスロット番号。
_ioswd_num_subs	存在するサブイベントの数。
全 Watchdog System Monitor（IOSWDSysMon）サブイベント	
_ioswd_sub1_present **_ioswd_sub2_present**	サブイベント 1 またはサブイベント 2 の存在を示す値。値 1 は、サブイベントが存在することを示し、値 0 はサブイベントが存在しないことを示します。
_ioswd_sub1_type _ioswd_sub2_type	イベントのタイプ（cpu_proc、または mem_proc）。
Watchdog System Monitor（IOSWDSysMon）cpu_proc サブイベント	
_ioswd_sub1_path _ioswd_sub2_path	サブイベントのプロセス名。
_ioswd_sub1_period **_ioswd_sub2_period**	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_ioswd_sub1_pid _ioswd_sub2_pid	サブイベントのプロセス ID。
_ioswd_sub1_taskname **_ioswd_sub2_taskname**	サブイベントのタスク名。
_ioswd_sub1_value _ioswd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
Watchdog System Monitor（IOSWDSysMon）mem_proc サブイベント	

環境変数	説明
_ioswd_sub1_diff _ioswd_sub2_diff	イベントをトリガーした差のパーセンテージの値。 **Note** この変数は、**_ioswd_sub1_is_percent** 変数または **_ioswd_sub2_is_percent** 変数が 1 である場合に限って設定されます。
_ioswd_sub1_is_percent **_ioswd_sub2_is_percent**	値がパーセンテージであるかどうかを識別する番号。0 であれば値がパーセンテージではないことを意味し、1 であれば値がパーセンテージであることを意味します。
_ioswd_sub1_path _ioswd_sub2_path	サブイベントのプロセス名。
_ioswd_sub1_pid _ioswd_sub2_pid	サブイベントのプロセス ID。
_ioswd_sub1_taskname **_ioswd_sub2_taskname**	サブイベントのタスク名。
_ioswd_sub1_value _ioswd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
Watchdog System Monitor（WDSysMon）イベント ディテクタ	
_wd_sub1_present _wd_sub2_present	サブイベント 1 またはサブイベント 2 の存在を示す値。値 1 は、サブイベントが存在することを示し、値 0 はサブイベントが存在しないことを示します。
_wd_num_subs	存在するサブイベントの数。
_wd_sub1_type _wd_sub2_type	イベントのタイプ（cpu_proc、cpu_tot、deadlock、dispatch_mgr、mem_proc、mem_tot_avail、または mem_tot_used）。
Watchdog System Monitor（WDSysMon）cpu_proc サブイベント	
_wd_sub1_node _wd_sub2_node	サブイベント RP レポーティング ノードのスロット番号。
_wd_sub1_period _wd_sub2_period	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_wd_sub1_procname **_wd_sub2_procname**	サブイベントのプロセス名。
_wd_sub1_value _wd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。

環境変数	説明
Watchdog System Monitor（WDSysMon）cpu_tot サブイベント	
_wd_sub1_node _wd_sub2_node	サブイベントRP レポーティング ノードのスロット番号。
_wd_sub1_period _wd_sub2_period	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_wd_sub1_value _wd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
Watchdog System Monitor（WDSysMon）deadlock サブイベント	
_wd_sub1_entry_ [1-N]**_b_node** **_wd_sub2_entry_**[1-N]**_b_node**	サブイベントRP レポーティング ノードのスロット番号。
_wd_sub1_entry_ [1-N]**_b_pid** **_wd_sub2_entry_**[1-N]**_b_pid**	サブイベントのプロセス ID。
_wd_sub1_entry_ [1-N]**_b_procname** **_wd_sub2_entry_**[1-N]**_b_procname**	サブイベントのプロセス名。
_wd_sub1_entry_ [1-N]**_b_tid** **_wd_sub2_entry_**[1-N]**_b_tid**	サブイベントの時間 ID。
_wd_sub1_entry_ [1-N]**_node** **_wd_sub2_entry_**[1-N]**_node**	サブイベントRP レポーティング ノードのスロット番号。
_wd_sub1_entry_ [1-N]**_pid** **_wd_sub2_entry_**[1-N]**_pid**	サブイベントのプロセス ID。
_wd_sub1_entry_ [1-N]**_procname** **_wd_sub2_entry_**[1-N]**_procname**	サブイベントのプロセス名。
_wd_sub1_entry_ [1-N]**_state** **_wd_sub2_entry_**[1-N]**_state**	サブイベントの時間 ID。
_wd_sub1_entry_ [1-N]**_tid** **_wd_sub2_entry_**[1-N]**_tid**	サブイベントの時間 ID。
_wd_sub1_num_entries **_wd_sub2_num_entries**	サブイベントの数。
Watchdog System Monitor（WDSysMon）dispatch manager サブイベント	
_wd_sub1_node _wd_sub2_node	サブイベントRP レポーティング ノードのスロット番号。

環境変数	説明
_wd_sub1_period _wd_sub2_period	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_wd_sub1_procname **_wd_sub2_procname**	サブイベントのプロセス名。
_wd_sub1_value _wd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
Watchdog System Monitor（WDSysMon）mem_proc サブイベント	
_wd_sub1_diff _wd_sub2_diff	イベントをトリガーした差のパーセンテージの値。 **Note** この変数は、**_wd_sub1_is_percent** 変数または **_wd_sub2_is_percent** 変数が 1 である場合に限って設定されます。
_wd_sub1_is_percent **_wd_sub2_is_percent**	値がパーセンテージであるかどうかを識別する番号。0 であれば値がパーセンテージではないことを意味し、1 であれば値がパーセンテージであることを意味します。
_wd_sub1_node _wd_sub2_node	サブイベント RP レポーティング ノードのスロット番号。
_wd_sub1_period _wd_sub2_period	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_wd_sub1_pid _wd_sub2_pid	サブイベントのプロセス ID。
_wd_sub1_procname **_wd_sub2_procname**	サブイベントのプロセス名。
_wd_sub1_value _wd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
Watchdog System Monitor（WDSysMon）mem_tot_avail and mem_tot_used サブイベント	
_wd_sub1_avail _wd_sub2_avail	サブイベントに使用可能なメモリ。
_wd_sub1_diff _wd_sub2_diff	イベントをトリガーした差のパーセンテージの値。 **Note** この変数は、**_wd_sub1_is_percent** 変数または **_wd_sub2_is_percent** 変数が 1 である場合に限って設定されます。

環境変数	説明
_wd_sub1_is_percent **_wd_sub2_is_percent**	値がパーセンテージであるかどうかを識別する番号。0 であれば値がパーセンテージではないことを意味し、1 であれば値がパーセンテージであることを意味します。
_wd_sub1_node _wd_sub2_node	サブイベント RP レポーティング ノードのスロット番号。
_wd_sub1_period _wd_sub2_period	サブイベントの測定に使用される時間間隔（秒単位、オプションでミリ秒単位）。
_wd_sub1_value _wd_sub2_value	パーセンテージで測定されたサブイベントの CPU 使用率。
_wd_sub1_used _wd_sub2_used	サブイベントが使用したメモリ。

Cisco IOS CLI を使用した EEM ポリシーの記述方法

Embedded Event Manager アプレットの登録と定義

アプレットを Embedded Event Manager に登録し、Cisco IOS CLI **event** コマンドと **action** コマンドを使用して定義するには、次の作業を実行します。EEM アプレットでは、**event** コマンドが 1 つだけ許可されます。**action** コマンドは複数許可されます。**event** コマンドと **action** コマンドが指定されていない場合、コンフィギュレーション モードの終了時にアプレットが削除されます。

この作業で使用する SNMP イベントディテクタと syslog **action** コマンドは、任意のイベントディテクタと **action** コマンドを表しています。他のイベントディテクタや **action** コマンドの使用例については、Embedded Event Manager アプレットの設定例, on page 640 を参照してください。

EEM 環境変数

EEM ポリシーの EEM 環境変数は、EEM **event manager environment** コンフィギュレーション コマンドを使用して定義されます。慣例として、すべてのシスコ EEM 環境変数は、「_」で始まります。将来的な競合を避けるため、「_」で始まる新しい変数を定義しないことを推奨します。

show event manager environment 特権 EXEC コマンドを使用して、システムの EEM 環境変数セットを表示できます。

たとえば、イベント発生時に E メールを送信する EEM ポリシーを作成できます。次の表に、EEM ポリシーで使用できる電子メール特有の環境変数の説明を示します。

Table 53: EEM 電子メール固有の環境変数

環境変数	説明
_email_server	E メール送信に使用されるシンプル メール転送プロトコル（SMTP）メール サーバー。
_email_to	E メールの送信先アドレス。
_email_from	E メールの送信元アドレス。
_email_cc	E メールのコピーの送信先アドレス。

EEM アクション ラベルのアルファベット順

EEM アクション ラベルは一意の ID で、任意の文字列値が可能です。アクションは、ラベルをソート キーとして使用して、英数字のキーの昇順（辞書順）にソートされ、実行されます。ラベルとして数字を使用している場合は、英数字ソートは、10.0 は 1.0 よりも後ですが、2.0 よりも前になることに注意してください。このような場合、01.0、02.0 のような数字を使用する、または頭文字の後に同様の数字を続けることを推奨します。

SUMMARY STEPS

1. **enable**
2. **show event manager environment** [**all**| *variable-name*]
3. **configure terminal**
4. **event manager environment** *variable-name string*
5. 手順 4 を、必要なすべての環境変数に繰り返します。
6. **event manager applet** *applet-name*
7. 次のいずれかを実行します。
   - **event snmp oid** *oid-value* **get-type** {**exact**| **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb**| **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
8. **action** *label* **cli command** *cli-string* [**pattern** *pattern-string*]
9. **action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string*
10. **action** *label* **mail server** *server-address* **to** *to-address* **from** *from-address* [**cc** *cc-address*] **subject** *subject* **body** *body-text*
11. 必要に応じて action コマンドを追加します。
12. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager environment** [**all**| *variable-name*] **Example:** `Device# show event manager environment all`	（任意）EEM 環境変数の名前と値を表示します。 • オプションの **all** キーワードは、すべての EEM 環境変数を表示します。 • オプションの *variable-name* 引数は、指定された環境変数に関する情報を表示します。
ステップ 3	**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 4	**event manager environment** *variable-name string* **Example:** `Device(config)# event manager environment _email_to engineering@example.com`	指定された EEM 環境変数の値を設定します。 • この例では、E メール送信先の E メール アドレスを保持する環境変数は、engineering@example.com に設定されます。
ステップ 5	手順 4 を、必要なすべての環境変数に繰り返します。	ステップ 4 を繰り返して、ステップ 6 で登録されるポリシーに必要なすべての環境変数を設定します。
ステップ 6	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet memory-fail`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 7	次のいずれかを実行します。 • **event snmp oid** *oid-value* **get-type** {**exact**| **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb**| **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value* **Example:** `Device(config-applet)# event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val 5120000 poll-interval 90`	EEM アプレットの実行の原因となる、イベント基準を指定します。 • この例では、空きメモリの値が 5120000 を下回ったときに EEM イベントがトリガーされます。 • 終了基準はオプションです。指定されない場合、イベントのモニターリングは、すぐに再び有効になります。

	Command or Action	Purpose

ステップ 8	**action** *label* **cli command** *cli-string* [**pattern** *pattern-string*]
	Example:
	`Device(config-applet)# action 1.0 cli command "enable"`
	Example:
	`Device(config-applet)# action 2.0 cli command "clear counters Ethernet0/1" pattern "confirm"`
	Example:
	`Device(config-applet)# action 3.0 cli command "y"`
		EEM アプレットがトリガーされたときに Cisco IOS CLI コマンドを実行するアクションを指定します。
		pattern キーワードはオプションで、コマンド文字列が入力を求める場合にだけ使用します。**action cli** コマンドは、オプションの **pattern** キーワードで指定されているとおりの応答プロンプトを受信した時点で終了します。次の応答プロンプトに一致する正規表現パターンを指定する必要があります。正しくないパターンを指定すると、**action cli** コマンドが、maxrun タイマー期限切れによるアプレット実行タイムアウトまで、待ち続けることになります。
		• 実行されるアクションは、**pattern** キーワードが **clear counters Ethernet0/1** コマンドの *confirm* 引数を指定するときに実行される EEM アプレットを指定するためのものです。この場合、コマンド文字列は「confirm」という入力を要求します。その入力は、「yes」または「no」で完了する必要があります。

ステップ 9	**action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string*
	Example:
	`Device(config-applet)# action 1.0 syslog priority critical msg "Memory exhausted; current available memory is $_snmp_oid_val bytes"`
	Example:
	`Device(config-applet)# action 1.0 syslog priority errors facility EEM-FAC message "TEST MSG"`
		EEM アプレットがトリガーされたときに実行されるアクションを指定します。
		この例では、実行されるアクションは syslog にメッセージを書き込むことです。
		• オプションの **priority** キーワードは syslog メッセージの優先度レベルを指定します。選択した場合は、*priority-level* 引数を定義する必要があります。
		• *msg-text* 引数は、文字テキスト、環境変数、またはその両方の組み合わせが可能です。
		• **facility** キーワードは生成したメッセージの場所を指定します。
		• *string* 引数は、キャラクタ テキスト、環境変数、またはその両方の組み合わせが可能です。

ステップ 10	**action** *label* **mail server** *server-address* **to** *to-address* **from** *from-address* [**cc** *cc-address*] **subject** *subject* **body** *body-text* **Example:** `Device(config-applet)# action 2.0 mail server`	EEM アプレットがトリガーされたときにショートメールを送信するアクションを指定します。 • *server-address* 引数は、電子メールの転送に使用する電子メール サーバーの完全修飾ドメイン名を指定します。

	Command or Action	Purpose
	`192.168.1.10 to engineering@example.com from devtest@example.com subject "Memory failure" body "Memory exhausted; current available memory is $_snmp_oid_val bytes"`	• *to-address* 引数は、電子メールの送信先の電子メール アドレスを指定します。 • *from-address* 引数は、電子メール送信元の電子メール アドレスを指定します。 • *subject* 引数は、英数字の文字列で、電子メールのサブジェクトラインの内容を指定します。 • *body-text* 引数は、英数字の文字列で、電子メールのテキストの内容を指定します。
ステップ 11	必要に応じて action コマンドを追加します。	--
ステップ 12	**end** **Example:** `Device(config-applet)# end`	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

トラブルシューティングのヒント

特権 EXEC モードで **debug event manager** コマンドを使用して、EEM コマンド操作のトラブルシューティングを行います。debugging コマンドは注意して使用してください。生成される出力量によってデバイスの動作が遅くなったり、停止したりすることがあります。シスコエンジニアの管理下に限ってこのコマンドを使用することを推奨します。

手動で実行する Embedded Event Manager ポリシーの登録と定義

EEM ポリシーを手動で実行するには 2 つの方法があります。EEM は、通常、ポリシーそのものに含まれるイベント仕様に基づいてポリシーをスケジューリングし、実行します。**event none** コマンドでは、EEM が手動でトリガー可能な EEM ポリシーを識別できます。ポリシーを実行するには、アプレット コンフィギュレーション モードで **action policy** コマンドを使用するか、または特権 EXEC モードで **event manager run** コマンドを実行します。

event manager run コマンドを使用して手動で実行される EEM ポリシーを登録するには、次の作業を実行します。**action policy** コマンドを使用して手動でポリシーを実行する方法については、「Embedded Event Manager の手動によるポリシー実行の例, on page 645」を参照してください。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string*

6. **end**
7. **event manager run** *applet-name*

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet manual-policy`	Embedded Event Manager にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**event none** **Example:** `Device(config-applet)# event none`	EEM に登録して手動で起動される EEM ポリシーを指定します。
ステップ 5	**action** *label* **syslog** [**priority** *priority-level*] **msg** *msg-text* **facility** *string* **Example:** `Device(config-applet)# action 1.0 syslog msg "Manual-policy triggered"`	EEM アプレットがトリガーされたときに実行されるアクションを指定します。 この例では、実行されるアクションは syslog にメッセージを書き込むことです。 • オプションの **priority** キーワードは syslog メッセージの優先度レベルを指定します。選択した場合は、*priority-level* 引数を定義する必要があります。 • *msg-text* 引数は、文字テキスト、環境変数、またはその両方の組み合わせが可能です。 • **facility** キーワードは生成したメッセージの場所を指定します。 • *string* 引数は、キャラクタテキスト、環境変数、またはその両方の組み合わせが可能です。
ステップ 6	**end** **Example:**	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

	Command or Action	Purpose
	`Device(config-applet)# end`	
ステップ 7	**event manager run** *applet-name* **Example:** `Device# event manager run manual-policy`	登録された EEM ポリシーを手動で実行します。

Embedded Event Manager ポリシーの登録解除

EEM ポリシーを実行コンフィギュレーション ファイルから削除するには、次の作業を実行します。ポリシーの実行はキャンセルされます。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**description** [*policy-name*] | **detailed** *policy-filename* [**system** | **user**] | [**event-type** *event-name*] [**system** | **user**] [**time-ordered** | **name-ordered**]]
3. **configure terminal**
4. **no event manager policy** *policy-filename*
5. **exit**
6. ステップ 2 を繰り返して、ポリシーが削除されたことを確認します。

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy registered** [**description** [*policy-name*]	**detailed** *policy-filename* [**system**	**user**]	[**event-type** *event-name*] [**system**	**user**] [**time-ordered**	**name-ordered**]] **Example:** `Device# show event manager policy registered`	（任意）現在登録されている EEM ポリシーを表示します。 • オプションの **system** キーワードおよび **user** キーワードは登録されているシステム ポリシーおよびユーザー ポリシーを表示します。 • キーワードが指定されない場合は、すべてのイベント タイプに対する登録された EEM ポリシーが時間順に表示されます。
ステップ 3	**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。

	Command or Action	Purpose
ステップ 4	**no event manager policy** *policy-filename* **Example:** `Device(config)# no event manager policy IPSLAping1`	ポリシーを登録解除するために EEM ポリシーを設定から削除します。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 6	ステップ 2 を繰り返して、ポリシーが削除されたことを確認します。 **Example:** `Device# show event manager policy registered`	--

例

次に、**show event manager policy registered** 特権 EXEC コマンドを使用して、現在登録されている 2 個の EEM アプレットを表示する例を示します。

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  snmp                Off   Fri Aug 12 17:42:52 2005  IPSLAping1
 oid {1.3.6.1.4.1.9.9.42.1.2.9.1.6.4} get-type exact entry-op eq entry-val {1}
 exit-op eq exit-val {2} poll-interval 90.000
 action 1.0 syslog priority critical msg "Server IPecho Failed: OID=$_snmp_oid_val"
 action 1.1 snmp-trap strdata "EEM detected server reachability failure to 10.1.88.9"
 action 1.2 publish-event sub-system 88000101 type 1 arg1 "10.1.88.9" arg2 "IPSLAEcho"
arg3 "fail"
 action 1.3 counter name _IPSLA1F op inc value 1
2    applet  system  snmp                Off   Thu Sep 15 05:57:16 2005  memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg Memory exhausted; current available memory is
$_snmp_oid_val bytes
 action 2.0 force-switchover
```

次の例では、**show event manager policy registered** 特権 EXEC コマンドを使用して、アプレット IPSLAping1 が **no event manager policy** コマンドの入力後に削除されていることを示します。

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  snmp                Off   Thu Sep 15 05:57:16 2005  memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg Memory exhausted; current available memory is
$_snmp_oid_val bytes
 action 2.0 force-switchover
```

すべての Embedded Event Manager ポリシーの実行の一時停止

すべての EEM ポリシーの実行をただちに一時停止するには、次の作業を実行します。一時的なパフォーマンスまたはセキュリティ面での理由から、ポリシーの登録解除ではなく一時停止が必要なことがあります。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**description** [*policy-name*] | **detailed** *policy-filename* [**system** | **user**] | [**event-type** *event-name*] [**system** | **user**] [**time-ordered** | **name-ordered**]]
3. **configure terminal**
4. **event manager scheduler suspend**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy registered** [**description** [*policy-name*]	**detailed** *policy-filename* [**system**	**user**]	[**event-type** *event-name*] [**system**	**user**] [**time-ordered**	**name-ordered**]] **Example:** `Device# show event manager policy registered`	（任意）現在登録されている EEM ポリシーを表示します。 • オプションの **system** キーワードおよび **user** キーワードは登録されているシステム ポリシーおよびユーザー ポリシーを表示します。 • キーワードが指定されない場合は、すべてのイベント タイプに対する登録された EEM ポリシーが時間順に表示されます。
ステップ 3	**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 4	**event manager scheduler suspend** **Example:** `Device(config)# event manager scheduler suspend`	すべての EEM ポリシーの実行がすぐに一時停止されます。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Embedded Event Manager 履歴データの表示

履歴テーブルのサイズを変更し、EEM 履歴データを表示するには、次の任意の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager history size** {**events** | **traps**} [*size*]
4. **exit**
5. **show event manager history events** [**detailed**] [**maximum** *number*]
6. **show event manager history traps** {**server** | **policy**}

DETAILED STEPS

ステップ 1 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）。

Example:

```
Device> enable
```

ステップ 2 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。

Example:

```
Device# configure terminal
```

ステップ 3 event manager history size {events | traps} [size]

このコマンドを使用して、EEM イベント履歴テーブルのサイズ、または、EEM SNMP トラップ履歴テーブルのサイズを変更します。次に、EEM イベント履歴テーブルのサイズを 30 エントリに変更する例を示します。

Example:

```
Device(config)# event manager history size events 30
```

ステップ 4 exit

グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Example:

```
Device(config)# exit
```

ステップ 5 show event manager history events [detailed] [maximum number]

このコマンドを使用して、各 EEM イベントの詳細情報を表示します。次に例を示します。

Example:

```
Device# show event manager history events
No.  Time of Event             Event Type          Name
1    Fri Aug13  21:42:57 2004  snmp                applet: SAAping1
2    Fri Aug13  22:20:29 2004  snmp                applet: SAAping1
3    Wed Aug18  21:54:48 2004  snmp                applet: SAAping1
4    Wed Aug18  22:06:38 2004  snmp                applet: SAAping1
5    Wed Aug18  22:30:58 2004  snmp                applet: SAAping1
6    Wed Aug18  22:34:58 2004  snmp                applet: SAAping1
7    Wed Aug18  22:51:18 2004  snmp                applet: SAAping1
8    Wed Aug18  22:51:18 2004  application         applet: CustApp1
```

ステップ 6 **show event manager history traps** {**server** | **policy**}

このコマンドを使用して、EEM サーバーまたは EEM ポリシーのいずれかから送信された EEM SNMP トラップを表示します。次に、EEM ポリシー内からトリガーされた EEM SNMP トラップが表示される例を示します。

Example:

```
Device# show event manager history traps policy
No.  Time                       Trap Type           Name
1    Wed Aug18  22:30:58 2004  policy              EEM Policy Director
2    Wed Aug18  22:34:58 2004  policy              EEM Policy Director
3    Wed Aug18  22:51:18 2004  policy              EEM Policy Director
```

Embedded Event Manager 登録済みポリシーの表示

登録済みの EEM ポリシーを表示するには、次の任意の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**]

DETAILED STEPS

ステップ 1 **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）。

Example:

```
Device> enable
```

ステップ 2 **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**]

このコマンドを **time-ordered** キーワードとともに使用して、現在登録されているポリシーの情報を時間でソートして表示します。次に例を示します。

Example:

```
Device# show event manager policy registered time-ordered
No.  Type    Event Type          Time                    Registered Name
1    applet  snmp                Thu May30 05:57:16 2004 memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val
{5120000} poll-interval 90
action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
2    applet  syslog              Wed Jul16 00:05:17 2004 intf-down
 pattern {.*UPDOWN.*Ethernet1/0.*}
 action 1.0 cns-event msg "Interface state change: $_syslog_msg"
```

このコマンドを**name-ordered**キーワードとともに使用して、現在登録されているポリシーの情報を名前でソートして表示します。次に例を示します。

Example:

```
Device# show event manager policy registered name-ordered
No.  Type    Event Type          Time Registered          Name
1    applet  syslog              Wed Jul16  00:05:17 2004 intf-down
 pattern {.*UPDOWN.*Ethernet1/0.*}
 action 1.0 cns-event msg "Interface state change: $_syslog_msg"
2    applet  snmp                Thu May30 05:57:16 2004   memory-fail
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val
{5120000} poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
```

このコマンドを **event-type** キーワードとともに使用して、*event-name* 引数で指定されたイベントタイプの現在登録されているポリシーに関する情報を表示します。次に例を示します。

Example:

```
Device# show event manager policy registered event-type syslog
No.  Type    Event Type          Time Registered           Name
1    applet  syslog              Wed Jul16  00:05:17 2004 intf-down
 pattern {.*UPDOWN.*Ethernet1/0.*}
 action 1.0 cns-event msg "Interface state change: $_syslog_msg"
```

イベント SNMP 通知の設定

SNMP 通知を設定するには、次の作業を実行します。

Before you begin

- SNMP イベントマネージャは、**snmp-server manager** コマンドを使用して設定する必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **event manager applet** *applet-name*
4. **event** [**tag** *event-tag*] **snmp-notification oid** *oid-string* **oid-val** *comparison-value* **op** *operator* [**maxrun** *maxruntime-number*] [**src-ip-address** *ip-address*] [**dest-ip-address** *ip-address*] [**default** *seconds*] [**direction** {**incoming** | **outgoing**}] [**msg-op** {**drop** | **send**}]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Event Manager にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**event** [**tag** *event-tag*] **snmp-notification oid** *oid-string* **oid-val** *comparison-value* **op** *operator* [**maxrun** *maxruntime-number*] [**src-ip-address** *ip-address*] [**dest-ip-address** *ip-address*] [**default** *seconds*] [**direction** {**incoming** | **outgoing**}] [**msg-op** {**drop** | **send**}] **Example:** `Device(config-applet)# event snmp-notification dest-ip-address 192.168.1.1 oid 1 op eq oid-val 10`	簡易ネットワーク管理プロトコル（SNMP）通知のサンプリングによって実行される Embedded Event Manager（EEM）アプレットのイベント基準を指定します。
ステップ 5	**end** **Example:** `Device(config-applet)# end`	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

複数イベント サポートの設定

複数イベントサポート機能は、EEMサーバーに複数のイベントを登録する機能を追加します。複数イベントサポートには、1個以上のイベントの発生、1個以上のトラッキング対象オブジェクトの状態、および、発生するイベントの時間間隔が含まれます。イベント パラメータは、CLIコマンドで指定されます。複数イベントを扱うためのデータ構造には、複数のイベントID

と相関関係ロジックが含まれます。このデータは、EEM サーバーに複数のイベントを登録するために使用されます。

イベント設定パラメータの設定

trigger コマンドは、トリガー アプレット コンフィギュレーション モードを開始し、EEM アプレットの複数イベント設定ステートメントを指定します。トリガーステートメントは、各イベント文に指定される *tag* 引数を使用して複数イベント ステートメントを関連付けます。イベントは指定されたパラメータに基づいて発生します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event** [**tag** *event-tag*] **cli pattern** *regular-expression* **sync** {**yes** | **no skip** {**yes** | **no**}} [**occurs** *num-occurrences*] [**period** *period-value*] [**maxrun** *maxruntime-number*]
5. **trigger** [**occurs** *occurs-value*] [**period** *period-value*] [**period-start** *period-start-value*] [**delay** *delay-value*]
6. **correlate** {**event** *event-tag* | **track** *object-number*} [*boolean-operator* **event** *event-tag*]
7. **attribute tag** *event-tag* [**occurs** *occurs-value*]
8. **action** *label* **cli command** *cli-string*

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet EventInterface`	EEM にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**event** [**tag** *event-tag*] **cli pattern** *regular-expression* **sync** {**yes**	**no skip** {**yes**	**no**}} [**occurs** *num-occurrences*] [**period** *period-value*] [**maxrun** *maxruntime-number*] **Example:** `Device(config-applet)# event tag 1.0 cli pattern`	Cisco IOS コマンドライン インターフェイス（CLI）コマンドの一致によって実行される EEM アプレットのイベント基準を指定します。

	Command or Action	Purpose
	"show bgp all" sync yes occurs 32 period 60 maxrun 60	
ステップ 5	**trigger** [**occurs** *occurs-value*] [**period** *period-value*] [**period-start** *period-start-value*] [**delay** *delay-value*] **Example:** `Device(config-applet)# trigger occurs 1 period-start "0 8 * * 1-5" period 60`	EEMアプレットの複雑なイベント設定パラメータを指定します。
ステップ 6	**correlate** {**event** *event-tag* | **track** *object-number*} [*boolean-operator* **event** *event-tag*] **Example:** `Device(config-applet)# correlate event 1.0 or event 2.0`	EEM アプレットのトリガー モードで複雑なイベント関連付けを指定します。 **Note** 「and」を使用して、トラップや syslog メッセージなどのイベントをグループ化した場合、デフォルトのトリガー発生時間枠は 3 分です。
ステップ 7	**attribute tag** *event-tag* [**occurs** *occurs-value*] **Example:** `Device(config-applet)# attribute tag 1.0 occurs 1`	EEMアプレットの複雑なイベントをビルドする最大 8 個の属性文を指定します。
ステップ 8	**action** *label* **cli command** *cli-string* **Example:** `Device(config-applet)# action 1.0 cli command "show pattern"`	EEM アプレットがトリガーされたときに CLI コマンドを実行するアクションを指定します。

例

次に、**show bgp all** CLI コマンドと「COUNT」文字列を含む syslog メッセージが 60 秒以内に発生した場合にアプレットが実行される例を示します。

```
event manager applet delay_50
 event tag 1.0 cli pattern "show bgp all" sync yes occurs 32 period 60 maxrun 60
 event tag 2.0 syslog pattern "COUNT"
 trigger occurs 1 delay 50
  correlate event 1.0 or event 2.0
  attribute tag 1.0 occurs 1
  attribute tag 2.0 occurs 1
 action 1.0 cli command "show pattern"
 action 2.0 cli command "enable"
 action 3.0 cli command "config terminal"
 action 4.0 cli command " ip route 192.0.2.0 255.255.255.224 192.0.2.12"
 action 91.0 cli command "exit"
 action 99.0 cli command "show ip route | incl 192.0.2.5"
```

EEM クラスベース スケジューリングの設定

Embedded Event Manager（EEM）ポリシーをスケジュールし、ポリシー スケジュール オプションを設定するには、次の作業を実行します。このタスクでは、2 個の EEM 実行スレッドが作成され、デフォルト クラスに割り当てられたアプレットが実行されます。

EEM ポリシーは、登録時に **class** キーワードを使用して、クラスに割り当てられます。クラスなしで登録された EEM ポリシーは、デフォルト クラスに割り当てられます。デフォルト クラスを保持するスレッドは、スレッドが作業に利用可能であるとき、デフォルト クラスをサービスします。特定のクラス文字に割り当てられたスレッドは、スレッドが作業に利用可能であるとき、クラス文字が一致する任意のポリシーをサービスします。

EEM 実行スレッドが、指定されたクラスのポリシー実行に利用可能でない場合で、クラスのスケジューラ ルールが設定されている場合は、ポリシーは該当クラスのスレッドが実行可能になるまで待ちます。同じ入力イベントからトリガーされた同期ポリシーは、同一の実行スレッドにスケジュールされなければなりません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. {| |} クラスオプションスレッド **event manager scheduler appletaxpcall-homethread class** *class-options* **number** 番号
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	{| |} クラスオプションスレッド **event manager scheduler appletaxpcall-homethread class** *class-options* **number** 番号 **Example:** `Device(config)# event manager scheduler applet thread class default number 2`	EEM ポリシーをスケジュールし、ポリシー スケジューリング オプションを設定します。 • この例では、2 個の EEM 実行スレッドが作成され、デフォルト クラスに割り当てられたアプレットが実行されます。
ステップ 4	**exit** **Example:**	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

	Command or Action	Purpose
	`Device(config)# exit`	

スケジュール済み EEM ポリシー イベントまたはイベント キューの保留

EEM スケジューラで、スケジュールされた EEM ポリシー イベントまたはイベント キューをホールドするには、次の作業を実行します。このタスクでは、すべての保留 EEM ポリシーが表示されます。ジョブ ID 2 を使用して特定されるポリシーは、EEM スケジューラでホールドされています。最初のステップは、ジョブ ID 2 のポリシーは、状態が Pending から Held に変更されていることを示しています。

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending** [**queue-type**{**applet** | **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]
3. **event manager scheduler hold** {**all**| **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary**| **rp_standby**}]
4. **show event manager policy pending** [**queue-type**{**applet** | **call-home** | **axp** | **script**} **class** *class-options* | **detailed**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy pending** [**queue-type**{**applet**	**call-home**	**axp**	**script**} **class** *class-options*	**detailed**] **Example:** `Device# show event manager policy pending`	保留 EEM ポリシーを表示します。
ステップ 3	**event manager scheduler hold** {**all**| **policy** *job-id*	**queue-type** {**applet**	**call-home**	**axp**	**script**} **class** *class-options*} [**processor** {**rp_primary**| **rp_standby**}] **Example:** `Device# event manager scheduler hold policy 2`	EEM スケジューラで、スケジュールされた EEM ポリシー イベントまたはイベント キューをホールドします。 • この例では、ジョブ ID 2 のポリシーがホールドされます。
ステップ 4	**show event manager policy pending** [**queue-type**{**applet**	**call-home**	**axp**	**script**} **class** *class-options*	**detailed** **Example:** `Device# show event manager policy pending`	他の保留ポリシーとともに、手順 3 でホールドされた EEM ポリシーのステータスが Held と表示されます。

例

次に、すべての保留 EEM ポリシーの表示方法とジョブ ID 2 の EEM ポリシーをホールドする例を示します。

```
Device# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
Device# event manager scheduler hold policy 2
Device# show event manager policy pending

no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2  2      held   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

EEM ポリシー イベントまたはイベント キューの実行の再開

EEM ポリシー イベントまたはイベント キューの実行を再開するには、次の作業を実行します。このタスクでは、スケジュール済み EEM ポリシー イベントまたはイベント キューの保留で保留状態となっていたポリシーは、実行を再開できるようになっています。

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler release** {**all** | **policy** *policy-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**}} **class** *class-options* [**processor** {*rp_primary* | *rp_standby*}]
4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	保留およびホールドされた EEM ポリシーを表示します。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。

	Command or Action	Purpose
ステップ 3	**event manager scheduler release** {**all** | **policy** *policy-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**}} **class** *class-options* [**processor** {*rp_primary* | *rp_standby*}] **Example:** `Device# event manager scheduler release policy 2`	指定された EEM ポリシーの実行を再開します。 • 例では、ジョブ ID 2 のポリシーの実行を再開する方法を示しています。
ステップ 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	他の保留ポリシーとともに、手順 3 で再開された EEM ポリシーの状態が pending と表示されます。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。

例

次に、すべての保留 EEM ポリシーの表示方法、および実行を再開するポリシーを指定する方法、ポリシーが保留状態に戻っていることを確認する例を示します。

```
Device# show event manager policy pending

no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2  2      held   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
Rotuer# event manager scheduler release policy 2
Rotuer# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

保留 EEM ポリシー イベントまたはイベント キューのクリア

実行中または実行を保留中の EEM ポリシー イベントをクリアするには、次の作業を実行します。このタスクでは、ジョブ ID 2 のポリシーが保留キューからクリアされます。ポリシーがクリアされる前後に保留中のポリシーを表示するには、**show event manager policy pending** コマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler clear** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary** | **rp_standby**}]

4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	保留 EEM ポリシーを表示します。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。
ステップ 3	**event manager scheduler clear** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} **class** *class-options*} [**processor** {**rp_primary** | **rp_standby**}] **Example:** `Device# event manager scheduler clear policy 2`	実行中または実行を保留中の EEM ポリシーをクリアします。 • この例では、ジョブ ID 2 のポリシーが保留キューからクリアされます。
ステップ 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	手順 3 でクリアされたポリシーを除く、保留中のすべての EEM ポリシーを表示します。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。

例

次に、実行を保留されたジョブ ID 2 のポリシーをクリアする例を示します。ポリシーがクリアされる前後に保留中のポリシーを表示するには、**show** コマンドを使用します。

```
Device# show event manager policy pending
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   2      pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three

Device# event manager scheduler clear policy 2
Device# show event manager policy pending
```

```
no. job id status time of event             event type     name
1   1      pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
3   3      pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

EEM ポリシー イベントまたはイベント キューのスケジューリング パラメータの変更

EEM ポリシー イベントのスケジューリング パラメータを変更するには、次の作業を実行します。**show event manager policy pending** コマンドは、B またはデフォルトクラスに割り当てられているポリシーを表示します。現在保留されているすべてのポリシーがクラスAに変更されます。設定変更後、**show event manager policy pending** コマンドはクラス A として割り当てられているすべてのポリシーを表示します。

SUMMARY STEPS

1. **enable**
2. **show event manager policy pending**
3. **event manager scheduler modify** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} | **class** *class-options*} [**queue-priority** {**high** | **last** | **low** | **normal**}][**processor** {**rp_primary** | **rp_standby**}]
4. **show event manager policy pending**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	保留 EEM ポリシーを表示します。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。
ステップ 3	**event manager scheduler modify** {**all** | **policy** *job-id* | **queue-type** {**applet** | **call-home** | **axp** | **script**} | **class** *class-options*} [**queue-priority** {**high** | **last** | **low** | **normal**}][**processor** {**rp_primary** | **rp_standby**}] **Example:** `Device# event manager scheduler modify all class A`	EEM ポリシーのスケジューリング パラメータを変更します。 • この例では、現時点での保留 EEM ポリシーはすべてクラス A に割り当てられています。

	Command or Action	Purpose
ステップ 4	**show event manager policy pending** **Example:** `Device# show event manager policy pending`	他の保留ポリシーとともに、手順 3 で変更された EEM ポリシーが表示されます。 **Note** この例では、このタスクに適用可能な構文だけが使用されています。詳細については、『Cisco IOS Network Management Command Reference』を参照してください。

例

次に、EEMポリシーのスケジューリングパラメータを変更する例を示します。この例では、**show event manager policy pending** コマンドは、B またはデフォルトクラスに割り当てられているポリシーを表示します。現在保留されているすべてのポリシーがクラス A に変更されます。設定変更後、**show event manager policy pending** コマンドはクラス A として現在割り当てられているすべてのポリシーを確認します。

```
Device# show event manager policy pending
no. class   status time of event             event type     name
1   default pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   default pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   B       pend   Thu Sep 7  02:54:04 2006  syslog         applet: three

Device# event manager scheduler modify all class A
Device# show event manager policy pending

no. class status time of event             event type     name
1   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: one
2   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: two
3   A     pend   Thu Sep 7  02:54:04 2006  syslog         applet: three
```

クラスベースでスケジュールされた EEM ポリシーのアクティビティの確認

EEM ポリシーのスケジュールされたアクティビティを確認するには、**show event manager scheduler** コマンドを使用します。

SUMMARY STEPS

1. **show event manager scheduler thread** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager scheduler thread [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

このコマンドは、スケジューラの視点からのすべてのEEM実行スレッドと実行中ポリシーの詳細を表示します。このコマンドには、オプションの **detailed** および **queue-type** キーワードが含まれています。次に、このコマンドの出力例を示します。

Example:

```
Device# show event manager scheduler thread
1 Script threads service class default
  total: 1 running: 1 idle: 0
2 Script threads service class range A-D
  total: 3 running: 0 idle: 3
3 Applet threads service class default
  total: 32 running: 0 idle: 32
4 Applet threads service class W X
  total: 5 running: 0 idle: 5
```

スケジューラスレッドを使用している実行中ポリシーの詳細を表示するには、**detailed** キーワードを使用します。次に、このキーワードの出力例を示します。

Example:

```
Device# show event manager scheduler thread detailed

1 Script threads service class default
total: 5 running: 5 idle: 0
1 job id: 12341, pid: 101, name: loop.tcl
2 job id: 12352, pid: 52, name: loop.tcl
3 job id: 12363, pid: 55, name: loop.tcl
4 job id: 12395, pid: 53, name: loop.tcl
5 job id: 12588, pid: 102, name: loop.tcl
2 Applet threads service class default
total: 32 running: 5 idle: 27
1 job id: 15585, pid: 104, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 job id: 15586, pid: 105, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 job id: 15587, pid: 106, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 job id: 15589, pid: 107, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 job id: 15590, pid: 80, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

キュータイプのスケジューラスレッドを表示するには、**queue-type** キーワードを使用します。次に、このキーワードの出力例を示します。

Example:

```
Device# show event manager sched thread queue-type applet
1 Applet threads service class default
total: 32 running: 7 idle: 25
Device# show event manager sched thread queue-type applet detailed
1 Applet threads service class default
total: 32 running: 5 idle: 27
1 job id: 15700, pid: 103, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 job id: 15701, pid: 104, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 job id: 15703, pid: 106, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 job id: 15704, pid: 107, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 job id: 15706, pid: 55, name: WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

クラスベースのアクティブ EEM ポリシーの確認

アクティブな EEM ポリシーか、または実行中の EEM ポリシーを確認するには、**show event manager policy active** コマンドを使用します。

SUMMARY STEPS

1. **show event manager policy active** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager policy active [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

このコマンドは、実行中の EEM ポリシーだけを表示します。このコマンドには、オプションの **class** キーワード、**detailed** キーワード、および **queue-type** キーワードが含まれています。次に、このコマンドの出力例を示します。

Example:

```
Device# show event manager policy active
no. job id p s status time of event event type name
1 12598 N A running Mon Oct29 20:49:37 2007 timer watchdog loop.tcl
2 12609 N A running Mon Oct29 20:49:42 2007 timer watchdog loop.tcl
3 12620 N A running Mon Oct29 20:49:46 2007 timer watchdog loop.tcl
4 12650 N A running Mon Oct29 20:49:59 2007 timer watchdog loop.tcl
5 12842 N A running Mon Oct29 20:51:13 2007 timer watchdog loop.tcl
default class - 6 applet events
no. job id p s status time of event event type name
1 15852 N A running Mon Oct29 21:11:09 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
2 15853 N A running Mon Oct29 21:11:09 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
3 15854 N A running Mon Oct29 21:11:10 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
4 15855 N A running Mon Oct29 21:11:10 2007 timer watchdog WDOG_SYSLG_CNTR_TRACK_INTF_APPL
5 15856 N A running Mon Oct29 21:11:11 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
6 15858 N A running Mon Oct29 21:11:11 2007 counter WDOG_SYSLG_CNTR_TRACK_INTF_APPL
```

保留 EEM ポリシーの確認

実行が保留中の EEM ポリシーを確認するには、**show event manager policy pending** コマンドを使用します。EEM クラス ベースのスケジュール オプションを指定するには、オプションのキーワードを使用します。

SUMMARY STEPS

1. **show event manager policy pending** [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

DETAILED STEPS

show event manager policy pending [**queue-type** {**applet**| **call-home** | **axp** | **script**} **class** *class-options* | **detailed**]

このコマンドは、保留中の EEM ポリシーのみを表示します。このコマンドには、オプションの **class** キーワード、**detailed** キーワード、および **queue-type** キーワードが含まれています。次に、このコマンドの出力例を示します。

Example:

```
Device# show event manager policy pending
no. job id p s status time of event event type name
1    12851   N A    pend    Mon Oct29   20:51:18 2007   timer watchdog   loop.tcl
2    12868   N A    pend    Mon Oct29   20:51:24 2007   timer watchdog   loop.tcl
3    12873   N A    pend    Mon Oct29   20:51:27 2007   timer watchdog   loop.tcl
4    12907   N A    pend    Mon Oct29   20:51:41 2007   timer watchdog   loop.tcl
5    13100   N A    pend    Mon Oct29   20:52:55 2007   timer watchdog   loop.tcl
```

EEM アプレット（インタラクティブ CLI）サポートの設定

同期アプレットは、2 つのコマンド、**action gets** および **action puts** を使用してローカルコンソール（tty）との連携をサポートするように拡張されました。これらのコマンドによってコンソールへの直接入力と表示が可能です。同期アプレットの出力は、System Logger をバイパスします。ローカル コンソールは、アプレットによって開かれ、対応する同期イベント ディテクタ pty によってサービスされます。同期出力は、開かれたコンソールに向けられます。

同期 EEM アプレットのアクティブ コンソールからの入力の読み取りと書き込み

次のタスクを使用して、EEM アプレットのインタラクティブ CLI サポートを実装します。

アクティブなコンソールからの入力の読み取り

同期ポリシーがトリガーされたとき、関連するコンソールがパブリッシュ情報仕様に格納されます。ポリシー ディテクタは、この情報を event_reqinfo コール内で問い合わせ、**action gets** コマンドで使用するために与えられたコンソール情報を格納します。

action gets コマンドは、アクティブコンソールからの入力の 1 行を読み、入力を変数に格納します。後続の改行文字は戻されません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **gets** *variable*
6. **action** *label* **syslog** [**priority** *priority-level* **msg** *msg-text*
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet action`	EEM にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**event none** **Example:** `Device(config-applet)# event none`	EEM に登録して手動で起動される EEM ポリシーを指定します。
ステップ 5	**action** *label* **gets** *variable* **Example:** `Device(config-applet)# action label2 gets input`	EEMアプレットがトリガーされたときに、同期アプレットのローカルコンソールから入力を取得し、与えられた変数に値を格納します。
ステップ 6	**action** *label* **syslog** [**priority** *priority-level* **msg** *msg-text* **Example:** `Device(config-applet)# action label3 syslog msg "Input entered was \"$input\""`	EEMアプレットがトリガーされたときに実行されるアクションを指定します。 • この例では、実行されるアクションは手順 5 で指定された変数の値を syslog に書き込むことです。
ステップ 7	**exit** **Example:** `Device(config-applet)# exit`	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

例

次に、同期アプレットのローカルttyから入力を取得して値を格納する例を示します。

```
Device(config)# event manager applet action
Device(config-applet)# event none
Device(config-applet)# action label2 gets input
```

```
Device(config-applet)# action label3 syslog msg “Input entered was \”$input\“”
```

アクティブなコンソールへの入力の書き込み

同期ポリシーがトリガーされたとき、関連するコンソールがパブリッシュ情報仕様に格納されます。ポリシーディテクタは、この情報を event_reqinfo コール内で問い合わせ、**action puts** コマンドで使用するために与えられたコンソール情報を格納します。

action puts コマンドは、アクティブコンソールに文字列を書き込みます。**nonewline** キーワードが指定されない限り、改行文字が表示されます。同期アプレットの **action puts** コマンドからの出力は、直接コンソールに表示され、System Logger をバイパスします。非同期アプレットの **action puts** コマンドの出力は、System Logger に向けられます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event none**
5. **action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]
6. **action** *label* **puts** [**nonewline**] *string*
7. **exit**
8. **event manager run** *applet-name*

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet action`	EEM にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**event none** **Example:** `Device(config-applet)# event none`	EEM に登録して手動で起動される EEM ポリシーを指定します。

	Command or Action	Purpose
ステップ 5	**action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]] **Example:** `Device(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1`	EEMアプレットがトリガーされたときに入力文字列の正規表現パターンと比較するアクションを指定します。
ステップ 6	**action** *label* **puts** [**nonewline**] *string* **Example:** `Device(config-applet)# action 2 puts "match is $_match"`	EEMアプレットがトリガーされたときにデータを直接ローカルコンソールに出力するアクションを指定します。 • **nonewline** キーワードはオプションであり、改行文字を表示しないために使用します。
ステップ 7	**exit** **Example:** `Device(config-applet)# exit`	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 8	**event manager run** *applet-name* **Example:** `Device# event manager run action`	登録された EEM ポリシーを手動で実行します。 • この例では、手順 3 で登録されたポリシーがトリガーされ、手順 5 および手順 6 で指定された、関連付けられたアクションが実行されます。

例

次に、**action puts** コマンドがデータを直接ローカルコンソールに出力する例を示します。

```
Device(config-applet)# event manager applet puts
Device(config-applet)# event none
Device(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
Device(config-applet)# action 2 puts "match is $_match"
Device(config-applet)# action 3 puts "submatch 1 is $_sub1"
Device# event manager run puts
match is one two three
submatch 1 is one
```

SNMP ライブラリ拡張の設定

リリースに応じて、SNMP ライブラリ拡張機能で次の設定を実行できます。

前提条件

この機能を使用するには、Cisco IOS Release12.4(22)T 以降のリリースを実行している必要があります

SNMP Get および Set オペレーション

SNMP ライブラリ拡張機能により、EEM アプレットの **action info** コマンドと Tcl の **sys_reqinfo_snmp** コマンドが拡張され、SNMP の get-one、get-next、getid および set-any オペレーションのための機能が追加されます。

SNMP Get オペレーション

SNMP イベント マネージャは SNMP get オペレーションを実行して、管理対象オブジェクトの 1 つ以上の変数を取得します。**action info type snmp oid get-type** コマンドと **action info type snmp getid** コマンドを使用すると、取得する変数とエージェントの IP アドレスを指定して SNMP get 要求を送信するように SNMP イベントマネージャを設定できます。

たとえば、OID の値が 1.3.6.1.2.1.1.1 である変数を取得する場合、変数値、1.3.6.1.2.1.1.1 を指定する必要があります。指定された値が一致しない場合、トラップが生成され、エラーメッセージが syslog 履歴に書き込まれます。

action info type snmp oid get-type コマンドは、実行する get オペレーションのタイプを指定します。正確な変数を取得するには、get オペレーションのタイプを **exact** に指定する必要があります。指定された OID 値の辞書順での後続値を取得するには、get オペレーションのタイプを **next** に設定する必要があります。

次の表に、SNMP get オペレーションから取得された値が保存される組み込み変数を示します。

***Table 54: action info type snmp oid* コマンドの組み込み変数**

組み込み変数	説明
_info_snmp_oid	SNMP オブジェクト ID。
_info_snmp_value	割り当てられた SNMP データ エレメントの値文字列。

GetID の動作

action info type snmp getid コマンドは SNMP エンティティから次の変数を取得します。

- sysDescr.0
- sysObjectID.0
- sysUpTime.0
- sysContact.0
- sysName.0
- sysLocation.0

次の表に、SNMP getID オペレーションから取得された値が保存される組み込み変数を示します。

Table 55: action info type snmp getid コマンドの組み込み変数

組み込み変数	説明
_info_snmp_syslocation_oid	sysLocation 変数の OID 値。
_info_snmp_syslocation_value	sysLocation 変数の値文字列。
_info_snmp_sysdescr_oid	sysDescr 変数の OID 値。
_info_snmp_sysdescr_value	sysDescr 変数の値文字列。
_info_snmp_sysobjectid_oid	sysObjectID 変数の OID 値。
_info_snmp_sysobjectid_value	sysObjectID 変数の値文字列。
_info_snmp_sysuptime_oid	sysUptime 変数の OID 値。
_info_snmp_sysuptime_value	sysUptime 変数の値文字列。
_info_snmp_syscontact_oid	sysContact 変数の OID 値。
_info_snmp_syscontact_value	sysContact 変数の値文字列。

get オペレーション要求は、ローカル ホストとリモート ホストの両方に送信できます。

SNMP Set オペレーション

MIB ビューでは、すべての SNMP 変数にデフォルト値が割り当てられています。SNMP イベント マネージャは、set オペレーションによってこれらの MIB 変数の値を変更できます。set オペレーションは、読み取りと書き込みアクセスが許可されたシステムでだけ実行できます。

set オペレーションを実行するには、変数のタイプと変数に割り当てられる値を指定する必要があります。

次の表に、有効な OID タイプと各 OID タイプの値を示します。

Table 56: set オペレーションの *OID* タイプおよび値

OID タイプ	説明
counter32	最小値が 0 の 32 ビットの数値。最大 4294967295 の範囲の整数値が有効で
gauge	最小値が 0 の 32 ビットの数値。たと ンターフェイスの速度を測定できます

OID タイプ	説明
integer	管理対象オブジェクトのコンテキ 数字が使用されます。たとえば、 はアップ、2 に設定した場合はダ
ipv4	IP バージョン 4 アドレス。ドット
octet string	物理アドレスを表すために使用さ
string	テキスト文字列を表すために使用 効です。
unsigned32	10 進の値を表すために使用される 効です。

set オペレーションは、ローカル ホストとリモート ホストの両方で実行できます。

SNMP トラップ要求および通知要求

トラップは、SNMP マネージャまたは NMS にネットワーク状態を警告する SNMP 通知です。

SNMP インフォーム要求は、SNMP マネージャにネットワーク状態を警告する SNMP 通知を参照し、SNMP マネージャからの受信の確認を要求します。

SNMP イベントは、SNMP MIB オブジェクト ID 値がサンプリングされたとき、または、SNMP カウンタが定義されたしきい値を超えたときに発生します。通知がイネーブルであり、該当するイベントが設定されている場合、SNMP トラップまたはインフォーム メッセージが生成されます。イベント マネージャ サーバーによって SNMP トラップまたはインフォーム メッセージが受信されたとき、SNMP 通知イベントがトリガーされます。

Embedded Event Manager（EEM）アプレットがトリガーされたときに SNMP トラップまたは通知メッセージを送信するには、 **action info type snmp trap** コマンドと **action info type snmp inform** コマンドを使用します。CISCO-EMBEDDED-EVENT-MGR-MIB.my を使用して、トラップおよびインフォーム メッセージが定義されます。

SNMP Get および Set オペレーションの EEM Applet 設定

ポリシーをイベント マネージャ サーバーに登録する一方で、SNMP イベントに関連付けられたアクションを設定できます。

SNMP set および get オペレーションの EEM アプレットを設定するには、次の作業を実行します。

Before you begin

- SNMP イベントマネージャは、**snmp-server manager** コマンドを使用して設定する必要があります。

- SNMP エンティティへのアクセスを有効にするためには、**snmp-server community** コマンドを使用して、SNMP コミュニティストリングを設定する必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. 次のいずれかを実行します。
   - **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
5. **action label info type snmp oid** *oid-value* **get-type** {**exact**| **next**} [**community** *community-string*] [**ipaddr** *ip-address*]
6. **action label info type snmp oid** *oid-value* **set-type** *oid-type* *oid-type-value* **community** *community-string* [**ipaddr** *ip-address*]
7. **action label info type snmp getid** *oid-value* [**community** *community-string*] [**ipaddr** *ip-address*]
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Event Manager にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	次のいずれかを実行します。 • **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value* **Example:** `Device(config-applet)# event snmp oid` **Example:**	EEM アプレットの実行の原因となる、イベント基準を指定します。 • この例では、空きメモリの値が 5120000 を下回ったときに EEM イベントがトリガーされます。 • 終了基準はオプションです。指定されない場合、イベントのモニターリングは、すぐに再び有効になります。

	Command or Action	Purpose

	`1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact`
	Example:
	`entry-op lt entry-val 5120000 poll-interval 90`	

ステップ 5	**action label info type snmp oid** *oid-value* **get-type** {**exact**| **next**} [**community** *community-string*] [**ipaddr** *ip-address*]
	Example:
	`Device(config-applet)# action 1.3 info type`
	Example:
	`snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type`
	Example:
	`exact community public ipaddr 172.17.16.69`
		実行する get オペレーションのタイプを指定します。
		• この例では、get オペレーションのタイプが exact と指定され、コミュニティ ストリングが public と指定されます。

ステップ 6	**action label info type snmp oid** *oid-value* **set-type** *oid-type oid-type-value* **community** *community-string* [**ipaddr** *ip-address*]
	Example:
	`Device(config-applet)# action 1.4 info type`
	Example:
	`snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 set-type`
	Example:
	`integer 42220 sysName.0 community rw ipaddr`
	Example:
	`172.17.16.69`
		（任意）設定される変数を指定します。
		• この例では、sysName.0 変数が set オペレーションに指定され、コミュニティ ストリングに rw が指定されます。
		Note set オペレーションの場合、SNMP コミュニティ ストリングを指定する必要があります。

ステップ 7	**action label info type snmp getid** *oid-value* [**community** *community-string*] [**ipaddr** *ip-address*]
	Example:
	`Device(config-applet)# action 1.3 info type`
	Example:
	`snmp getid community public ipaddr 172.17.16.69`
		（任意）個々の変数が getid オペレーションによって取得される必要があるかどうかを指定します。

ステップ 8	**exit** **Example:**	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

	Command or Action	Purpose
	`Device(config)# exit`	

SNMP OID 通知の EEM アプレットの設定

SNMP 通知を設定するには、次の作業を実行します。

Before you begin

- SNMP イベントマネージャを、**snmp-server manager** コマンドを使用して設定し、SNMP エージェントが EEM ポリシーのために生成された SNMP トラップを送受信するように設定する必要があります。
- SNMP トラップとインフォームを **snmp-server enable traps event-manager** および **snmp-server enable traps** コマンドを使用して有効にして、トラップ要求とインフォーム要求をデバイスからイベントマネージャサーバーに送信できるようにする必要があります。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. 次のいずれかを実行します。
   - **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
5. **action label info type snmp var** *variable-name* **oid** *oid-value* *oid-type oid-type-value*
6. **action label info type snmp trap enterprise-oid** *enterprise-oid-value* **generic-trapnum** *generic-trap-number* **specific-trapnum** *specific-trap-number* **trap-oid** *trap-oid-value* **trap-var** *trap-variable*
7. **action label info type snmp inform trap-oid** *trap-oid-value* **trap-var** *trap-variable* **community** *community-string* **ipaddr** *ip-address*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
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:**	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。

	Command or Action	Purpose
	`Device# configure terminal`	
ステップ 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet snmp`	Event Manager にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。

ステップ 4	次のいずれかを実行します。
	• **event snmp oid** *oid-value* **get-type** {**exact** | **next**} **entry-op** *operator* **entry-val** *entry-value*[**exit-comb** | **and**}] [**exit-op** *operator*] [**exit-val** *exit-value*] [**exit-time** *exit-time-value*] **poll-interval** *poll-int-value*
	Example:
	`Device(config-applet)# event snmp oid`
	Example:
	`1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact`
	Example:
	`entry-op lt entry-val 5120000 poll-interval 90`
		EEM アプレットの実行の原因となる、イベント基準を指定します。
		• この例では、空きメモリの値が 5120000 を下回ったときに EEM イベントがトリガーされます。
		• 終了基準はオプションです。指定されない場合、イベントのモニターリングは、すぐに再び有効になります。

ステップ 5	**action label info type snmp var** *variable-name* **oid** *oid-value* *oid-type oid-type-value*
	Example:
	`Device(config-applet)# action 1.3 info type`
	Example:
	`snmp var sysDescr.0 oid`
	Example:
	`1.3.6.1.4.1.9.9.48.1.1.1.6.1 integer 4220`
		管理対象オブジェクトのインスタンスとその値を指定します。
		• この例では、sysDescr.0 変数が使用されています。

ステップ 6	**action label info type snmp trap enterprise-oid** *enterprise-oid-value* **generic-trapnum** *generic-trap-number* **specific-trapnum** *specific-trap-number* **trap-oid** *trap-oid-value* **trap-var** *trap-variable* **Example:**	EEM アプレットがトリガーされたときに SNMP トラップを生成します。 • この例では、authenticationFailure トラップが生成されます。

	Command or Action	Purpose

	`Device(config-applet)# action 1.4 info type`
	Example:
	`snmp trap enterprise-oid 1.3.6.1.4.1.1`
	Example:
	`generic-trapnum 4 specific-trapnum 7 trap-oid`
	Example:
	`1.3.6.1.4.1.1.226.0.2.1 trap-var sysUpTime.0`
		Note 固有のトラップ番号は、enterprise イベントが発生したときに生成される enterprise-specific トラップを示します。標準トラップ番号が 6 に設定されていない場合、指定した固有のトラップ番号がトラップの生成に使用されます。

ステップ 7	**action label info type snmp inform trap-oid** *trap-oid-value* **trap-var** *trap-variable* **community** *community-string* **ipaddr** *ip-address*
	Example:
	`Device(config-applet)# action 1.4 info type`
	Example:
	`snmp inform trap-oid 1.3.6.1.4.1.1.226.0.2.1`
	Example:
	`trap-var sysUpTime.0 community public ipaddr`
	Example:
	`172.69.16.2`
		EEM アプレットがトリガーされたときに SNMP インフォーム要求を生成します。
		• この例では、sysUpTime.0 変数のインフォーム要求が生成されます。

ステップ 8	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権モードに戻ります。

EEM アプレットの可変ロジックの設定

EEM アプレットの可変ロジック機能は、EEM アプレット内に条件付きロジックを適用する機能を追加します。アプレットには、可変ロジックが導入される前は、イベントがトリガーされたときに各アクションが設定された順に実行されるリニア構造だけがありました。条件付きロジックは、アプレット内のアクションのフローを条件式に従って変更する制御構造を追加します。各制御構造には、ループ アクションや、構造を実行するかどうかを決定する if/else アクションを含むアプレット アクションのリストが含まれます。

アプレットコンフィギュレーションモードの情報は、action コマンドの内容を設定するための背景として示されます。

Tool Command Language（Tcl）とアプレット（CLI）ベースの EEM ポリシーの間で一貫したユーザー インターフェイスを実現するには、次の基準に従います。

- Tcl ベースの実装では、イベント仕様基準は TCL で記述されます。
- アプレット ベースの実装では、イベント仕様データは CLI アプレット サブモード コンフィギュレーション文を使用して記述されます。

アプレット コンフィギュレーション モードは、event manager applet コマンドを使用して開始します。アプレット コンフィギュレーション モードでは、config プロンプトが、(config-applet)# に変わります。アプレット コンフィギュレーション モードでは、2 種類のコンフィギュレーション文がサポートされます。

- event：アプレットが実行される原因となるイベント基準を指定するために使用します。
- action：実行する組み込みアクションを指定するために使用します。

1 つのアプレット コンフィギュレーション内で複数の **action** アプレット コンフィギュレーション コマンドを使用できます。**action** アプレット コンフィギュレーション コマンドが存在しない場合は、終了時に、このアプレットに文が割り当てられていないことを示す警告が表示されます。このアプレットに文が割り当てられない場合、イベントはトリガーされますが、アクションは実行されません。アプレット コンフィギュレーション モードでコマンドが指定されない場合は、終了時にアプレットが削除されます。exit applet config コマンドは、アプレット コンフィギュレーション モードを終了するために使用されます。

リリースに応じて、EEM アプレットの可変ロジック機能は次の設定を実行できます。

前提条件

この機能を使用するには、Cisco IOS Release12.4(22)T 以降のリリースを実行している必要があります

EEM アプレットの可変ロジックの設定

EEM 3.0 は、アプレット内で単純な可変ロジックを可能にするための新しいアプレット action コマンドを追加しました。

action コマンドを使用して可変ロジックを設定するには、次の作業を実行します。

条件付きブロックのループの指定

EEM アプレットがトリガーされたときに、条件付きブロックのループを指定するには、次の作業を実行します。次のタスクでは、変数の値が 10 よりも小さいかどうかを確認するために、条件付きループが設定されます。変数の値が 10 よりも小さい場合は、メッセージ「i is $_i」が syslog に書き込まれます。

Note リリースに応じて、**set**（EEM）コマンドは **action set** コマンドに置き換えられます。詳細については、 **action** *label* **set** コマンドを参照してください。特定のリリースで set（EEM）コマンドを入力した場合、IOS パーサーは **set** コマンドを **action** *label* **set** コマンドに変換します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set**
5. **action** *label* **while** *string_op1 operator string_op2*
6. 必要に応じてアクションを追加します。
7. **action** *label* **end**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet condition`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**action** *label* **set** **Example:** `Device(config-applet)#action 1.0 set i 2`	イベントに対するアクションを設定します。 • この例では、変数 i の値が 2 に設定されます。
ステップ 5	**action** *label* **while** *string_op1 operator string_op2* **Example:** `Device(config-applet)# action 2 while $i lt 10`	条件付きブロックのループを指定します。 • この例では、変数 i の値が 10 よりも小さいかどうかを確認するために、ループが設定されます。
ステップ 6	必要に応じてアクションを追加します。 **Example:**	action コマンドで指示されたアクションを実行します。

	Command or Action	Purpose
	`Device(config-applet)# action 3 syslog msg "i is $i"`	• この例では、メッセージ「i is $_i」が syslog に書き込まれます。
ステップ 7	**action** *label* **end** **Example:** `Device(config-applet)# action 3 end`	実行中のアクションを終了します。

if else 条件付きブロックの指定

if 条件付き文の開始とそれに続く else 条件付き文を指定するには、次の作業を実行します。if 条件付き文と else 条件付き文は、それぞれを結合して使用することも、別々に使用することもできます。このタスクでは、変数の値が 5 に設定されます。次に、変数の値が 10 よりも小さいかどうかを確認するために、if 条件付きブロックが指定されます。if 条件付きブロックが満たされる場合にメッセージ「x is less than 10」を出力する action コマンドが指定されます。

if 条件付きブロックに続いて、else 条件付き部位ロックが指定されます。if 条件付きブロックが満たされない場合にメッセージ「x is greater than 10」を出力する action コマンドが指定されます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set** *variable-name variable-value*
5. **action** *label* **if** [*stringop1*] {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} [*stringop2*]
6. 必要に応じてアクションを追加します。
7. **action** *label* **else**
8. 必要に応じてアクションを追加します。
9. **end**

DETAILED STEPS

	Command or Action	Purpose
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。

	Command or Action	Purpose
ステップ 3	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet ifcondition`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**action** *label* **set** *variable-name variable-value* **Example:** `Device(config-applet)# action 1.0 set x 5`	イベントに対するアクションを設定します。 • この例では、変数 x の値が 5 に設定されます。
ステップ 5	**action** *label* **if** [*stringop1*] {**eq** | **gt** | **ge** | **lt** | **le** | **ne**} [*stringop2*] **Example:** `Device(config-applet)# action 2.0 if $x lt 10`	if 条件付き文を指定します。 • この例では、if条件付き文は変数の値が 10 よりも小さいかどうかを確認します。
ステップ 6	必要に応じてアクションを追加します。 **Example:** `Device(config-applet)# action 3.0 puts "$x is less than 10"`	action コマンドで指示されたアクションを実行します。 • この例では、メッセージ「5 is less than 10」が画面に表示されます。
ステップ 7	**action** *label* **else** **Example:** `Device(config-applet)# action 4.0 else`	else 条件付きステートメントを指定します。
ステップ 8	必要に応じてアクションを追加します。 **Example:** `Device(config-applet)# action 5.0`	action コマンドで指示されたアクションを実行します。 • この例では、メッセージ「5 is greater than 10」が画面に表示されます。
ステップ 9	**end** **Example:** `Device(config-applet)# end`	実行中のアクションを終了します。

foreach 反復文の指定

デリミタをトークン化パターンとして使用して入力文字列上で繰り返す条件付き文を指定するには、次の作業を実行します。foreach反復文は目的の情報を取得するためにコレクションを使用して繰り返すために使用されます。デリミタは、正規表現パターン文字列です。各反復で見つかったトークンは、与えられた iterator 変数に割り当てられます。すべての算術演算は、長整数としてオーバーフローのチェックなしで実行されます。この例では、変数 x の値が 5 に設

定されます。反復文は、入力文字列 red、blue、green、orange の間、実行するように設定されます。入力文字列の各エレメントに対して、対応するメッセージが画面に表示されます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **foreach** [*string-iterator*] [*string-input*] [*string-delimiter*]
5. 任意の action コマンドを指定します。
6. **action** *label* **end**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet iteration`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**action** *label* **foreach** [*string-iterator*] [*string-input*] [*string-delimiter*] **Example:** `Device(config-applet)# action 2.0 foreach iterator "red blue green orange"`	デリミタをトークン化パターンとして使用して、入力文字列上で繰り返します。 • この例では、入力のエレメント、red、blue、green、および、orange の間、反復が実行されます。
ステップ 5	任意の action コマンドを指定します。 **Example:** `Device(config-applet)# action 3.0 puts "Iterator is $iterator"`	action コマンドで指示されたアクションを実行します。 • この例では、次のメッセージが画面に表示されます。 Iterator is red Iterator is blue Iterator is green Iterator is orange

	Command or Action	Purpose
ステップ 6	**action** *label* **end** **Example:** `Device(config-applet)# action 4.0 end`	実行中のアクションを終了します。

正規表現の使用

正規表現パターンを入力文字列と比較するには、次の作業を実行します。正規表現を使用すると、比較される文字列として可能性のある文字列のセットを表す規則を指定できます。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet regexp`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**action** *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]] **Example:** `Device(config-applet)# action 2.0 regexp "(.*) (.*) (.*)" "red blue green" _match _sub1`	入力文字列と比較する表現パターンを指定します。 • この例では、「red blue green」の入力文字列が指定されます。表現パターンが入力文字列と一致すると、**red blue green** の結果全体が変数の **_match** に格納され、部分一致の **red** は変数の **_sub1** に格納されます。

変数の値の増加

変数の値を増加させるには、次の作業を実行します。このタスクでは変数の値が 20 に設定され、次に値が 12 だけ増加します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **action** *label* **set**
5. **action** *label* **increment** *variable-name long-integer*

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet increment`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**action** *label* **set** **Example:** `Device(config-applet)#action 1.0 set varname 20`	イベントに対するアクションを設定します。 • この例では、変数の値が 20 に設定されます。
ステップ 5	**action** *label* **increment** *variable-name long-integer* **Example:** `Device(config-applet)#action 2.0 increment varname 12`	変数の値が指定された長整数だけ増加します。 • この例では、変数の値が 12 だけ増加します。

イベント SNMP オブジェクトの設定

SNMP オブジェクトのサンプリングによって実行される Embedded Event Manager（EEM）アプレットの簡易ネットワーク管理プロトコル（SNMP）オブジェクトイベントを登録するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event snmp-object oid** *oid-value* **type** *value* **sync** {**yes** | **no**} **skip** {**yes** | **no**} **istable** {**yes** | **no**} [**default** *seconds*] [**maxrun** *maxruntime-number*]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet manual-policy`	Embedded Event Manager にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。

ステップ 4	**event snmp-object oid** *oid-value* **type** *value* **sync** {**yes** | **no**} **skip** {**yes** | **no**} **istable** {**yes** | **no**} [**default** *seconds*] [**maxrun** *maxruntime-number*]
	Example:
	`Device(config-applet)# event snmp-object oid 1.9.9.9.9 type gauge sync yes`
	Example:
	`action 1 syslog msg "oid = $_snmp_oid"`
	Example:
	`action 2 syslog msg "request = $_snmp_request"`
	Example:
		Embedded Event Manager（EEM）アプレット用の簡易ネットワーク管理プロトコル（SNMP）オブジェクトイベントを登録し、オブジェクトの SNMP GET および SET 要求を代行受信します。
		デフォルトでは、このコマンドは設定されていません。このコマンドが設定されると、デフォルトは構文オプションの説明と同一になります。
		• **oid** キーワードは、SNMP オブジェクト識別子（object ID）を指定します。
		• *oid-value* 引数は、SNMP ドット付き表記のデータ エレメントのオブジェクト ID 値です。OID は、関連する MIB

	Command or Action	Purpose
	`action 3 syslog msg "request_type = $_snmp_request_type"`	（CISCO-EMBEDDED-EVENT-MGR-MIB）内にタイプとして定義され、各タイプはオブジェクト値を保持します。 • **istable** キーワードは、OID が SNMP テーブルかどうかを指定します。 • sync キーワードは、アプレットを同期モードで実行するよう指定します。アプレットからの戻りコードは、SNMP 要求に応答するかどうかを示します。コード 0 は「要求に応答しない」、コード 1 は「要求に応答する」を意味します。アプレットからの戻りコードが要求に応答すると、**action snmp-object-value** コマンドを使用して、オブジェクトのアプレットで値が指定されます。 • **type** キーワードは、オブジェクトのタイプを指定します。 • *value* 引数はオブジェクトの値です。 • skip キーワードは、CLI コマンドの実行をスキップするかどうかを指定します • **default** キーワードは、アプレットが通常処理する SET 要求または GET 要求の時間を指定します。**default** キーワードが指定されない場合は、デフォルトの時間が 30 秒に設定されます。 • *milliseconds* 引数は、SNMP オブジェクト イベントディテクタがポリシーの終了を待つ時間です。 • **maxrun** キーワードは、アプレットの最大ランタイムを指定します。**maxrun** キーワードを指定した場合、*maxruntime-number* 値を指定する必要があります。**maxrun** キーワードが指定されていない場合、デフォルトのアプレットランタイムは 20 秒です。 • *milliseconds* 引数は、ミリ秒単位のアプレットの最大ランタイムです。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

	Command or Action	Purpose
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

AAA 認証の無効化

トリガーされたときに、EEM ポリシーが AAA 認証をバイパスするようにするには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name* [**authorization bypass**] [**class** *class-options*] [**trap**]
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* [**authorization bypass**] [**class** *class-options*] [**trap**] **Example:** `Device(config)# event manager applet one class A authorization bypass`	Embedded Event Manager（EEM）にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**exit** **Example:** `Device(config-aaplet)# exit`	デバイス コンフィギュレーション アプレット モードを終了し、特権 EXEC モードに戻ります。

Embedded Event Manager アプレットの説明の設定

EEM アプレットについて記述するには、次の作業を実行します。アプレットの説明は、他のアプレット設定の前でも後でも、任意の順序で追加できます。すでに説明があるアプレットに新しい説明を設定すると、現在の説明が上書きされます。アプレットの説明はオプションです。

アプレットに新しい説明を設定するには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **description** *line*
5. **event syslog pattern** *regular-expression*
6. **action** *label* **syslog msg** *msg-text*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet increment`	EEM にアプレットを登録し、アプレット コンフィギュレーション モードを開始します。
ステップ 4	**description** *line* **Example:** `Device(config-applet)# description "This applet looks for the word count in syslog messages"`	簡易ネットワーク管理プロトコル（SNMP）のサンプリングによって実行される EEM アプレットの説明を追加または変更します。
ステップ 5	**event syslog pattern** *regular-expression* **Example:** `Device(config-applet)# event syslog pattern "count"`	syslog メッセージの一致によって実行される Embedded Event Manager（EEM）アプレットのイベント基準を指定します。

	Command or Action	Purpose
ステップ 6	**action** *label* **syslog msg** *msg-text* **Example:** `Device(config-applet)# action 1 syslog msg hi`	EEM アプレットがトリガーされたときに実行されるアクションを指定します。 • この例では、実行されるアクションは syslog にメッセージを書き込むことです。 • *msg-text* 引数は、文字テキスト、環境変数、またはその両方の組み合わせが可能です。
ステップ 7	**end** **Example:** `Device(config-applet)# end`	アプレット コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Cisco IOS CLI を使用して EEM ポリシーを記述する設定例

Embedded Event Manager アプレットの設定例

次に、一部の EEM イベント ディテクタの EEM アプレット作成例を示します。次の例は、Embedded Event Manager アプレットの登録と定義, on page 593で説明した手順に従っています。

Application-Specific イベント ディテクタ

次に、EventPublish_A という名前のポリシーが、20 秒ごとに実行され、番号が 1 のイベントタイプを、番号 798 のサブシステムにパブリッシュする例を示します。サブシステムの値、798 は、イベントのパブリッシュが EEM ポリシーから発生することを指定します。EventPublish_B という名前の別のポリシーは、EEM イベント タイプ 1 が発生したときに実行されるように、subsystem 798 に登録されます。EventPublish_B ポリシーは実行されるときに、EventPublish_A から引数として渡されたデータを含むメッセージを syslog に送信します。

```
event manager applet EventPublish_A
 event timer watchdog time 20.0
 action 1.0 syslog msg "Applet EventPublish_A"
 action 2.0 publish-event sub-system 798 type 1 arg1 twenty
 exit
event manager applet EventPublish_B
 event application sub-system 798 type 1
 action 1.0 syslog msg "Applet EventPublish_B arg1 $_application_data1"
```

CLI イベント ディテクタ

次に、Cisco IOS **write memory** CLI コマンドが実行されたときに実行する EEM アプレットを指定する例を示します。アプレットは、このイベントが syslog メッセージによって生成した通知を提供します。この例では、**sync** キーワードが yes 引数とともに設定されています。これは、このポリシーの実行が完了したときに、イベントディテクタに通知されることを意味します。

ポリシーの終了状態が、CLI コマンドが実行されるかどうかを決定します。この例では、ポリシーの終了状態は 1 に設定され、CLI コマンドは実行されます。

```
event manager applet cli-match
 event cli pattern "write mem.*" sync yes
 action 1.0 syslog msg "$_cli_msg Command Executed"
 set 2.0 _exit_status 1
```

次に、**cli pattern** と test 引数を照合するアプレットの例を示します。**show access-list test** が入力されると、CLI イベントディテクタは、test 引数を照合し、アプレットがトリガーされます。**debug event manager detector cli** 出力が追加され、num_matches が 1 に設定されていることが示されます。

```
!
event manager applet EEM-PIPE-TEST
 event cli pattern "test" sync yes
 action 1.0 syslog msg "Pattern matched!"
!
*Aug 23 23:19:59.827: check_eem_cli_policy_handler: command_string=show access-lists
test
*Aug 23 23:19:59.827: check_eem_cli_policy_handler: num_matches = 1, response_code = 4
*Aug 23 23:19:59.843: %HA_EM-6-LOG: EEM-PIPE-TEST: Pattern matched!
```

Note CLI イベント ディテクタによる機能は、有効な IOS CLI コマンドでの正規表現パターン比較機能だけです。これには、リダイレクションが使用される場合のパイプ記号（|）以降のテキストは含まれません。

次に、**show version | include test** が入力された場合にアプレットがトリガーされなかった例を示します。CLI イベントディテクタでパイプ（|）文字の後ろに入力された文字との一致がなく、**debug event manager detector cli** 出力で num_matches がゼロと表示されいるためにトリガーされませんでした。

```
*Aug 23 23:20:16.827: check_eem_cli_policy_handler: command_string=show version
*Aug 23 23:20:16.827: check_eem_cli_policy_handler: num_matches = 0, response_code = 1
```

Counter イベント ディテクタおよび Timer イベント ディテクタ

次に、EventCounter_A ポリシーが 1 分に 1 回実行されるように設定され、既知のカウンタ critical_errors を増加させる例を示します。2 番目のポリシー、EventCounter_B は、critical_errors がという既知のカウンタがしきい値 3 を超えたときにトリガーされるように登録されます。EventCounter_B ポリシーが実行されたとき、カウンタは 0 にリセットされます。

```
event manager applet EventCounter_A
 event timer watchdog time 60.0
 action 1.0 syslog msg “EventCounter_A”
 action 2.0 counter name critical_errors op inc value 1
 exit
event manager applet EventCounter_B
 event counter name critical_errors entry-op gt entry-val 3 exit-op lt exit-val 3
 action 1.0 syslog msg “EventCounter_B”
 action 2.0 counter name critical_errors op set value 0
```

Interface Counter イベント ディテクタ

次に、EventInterface という名前のポリシーが、ファスト イーサネット インターフェイス 0/0 の receive_throttle カウンタが 5 ずつ増加するたびに、トリガーされる例を示します。カウンタをチェックするポーリング間隔は、90 秒ごとに 1 回実行するように指定されます。

```
event manager applet EventInterface
 event interface name FastEthernet0/0 parameter receive_throttle entry-op ge entry-val
5
 entry-val-is-increment true poll-interval 90
 action 1.0 syslog msg "Applet EventInterface"
```

RF イベント ディテクタ

RF イベント ディテクタは、デュアル ルート プロセッサ（RP）を備えたネットワーキング デバイスでだけ利用できます。次に、RF 状態変化通知に基づいてイベント基準を指定する例を示します。

```
event manager applet start-rf
 event rf event rf_prog_initialization
 action 1.0 syslog msg "rf state rf_prog_initialization reached"
```

Remote Procedure Call（RPC）イベント ディテクタ

RPC イベント ディテクタによって、外部エンティティがデバイスに対して Simple Object Access Protocol（SOAP）要求を作成でき、定義された EEM ポリシーまたはスクリプトを実行できます。次に、Event_RPC という名前の EEM アプレットが EEM スクリプトを実行するように登録されている例を示します。

```
event manager applet Event_RPC
 event rpc
 action print puts "hello there"
```

次に、SOAP 要求と返信メッセージの形式の例を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP:Envelope xmlns:SOAP="http://www.cisco.com/eem.xsd">
  <SOAP:Body>
    <run_eemscript>
      <script_name>Event_RPC</script_name>
    </run_eemscript>
  </SOAP:Body>
</SOAP:Envelope>
]]>]]>
<?xml version="1.0" encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.cisco.com/eem.xsd"><SOAP:Body>
<run_eemscript_response><return_code>0</return_code><output></output></run_eemscript_response></SOAP:Body></SOAP:Envelope>]]>]]>
```

SNMP イベント ディテクタ

次に、CPU 使用率が 75% を上回ったときに実行する EEM アプレットを指定する例を示します。EEM アプレットを実行すると、CLI コマンドの **enable** と **show cpu processes** が実行され、**show cpu processes** コマンドの結果が含まれている電子メールがエンジニアに送信されます。

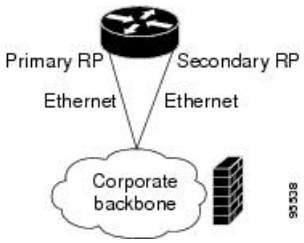
```
event manager applet snmpcpuge75
 event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3.1 get-type exact entry-op ge entry-val 75
poll-interval 10
 action 1.0 cli command "enable"
 action 2.0 cli command "show process cpu"
 action 3.0 mail server "192.168.1.146" to "engineer@cisco.com" from "devtest@cisco.com"
subject "B25 PBX Alert" body "$_cli_result"
```

次の例はより複雑で、プライマリ ルート プロセッサ（RP）がメモリ不足で実行されているときに、セカンダリ（冗長）RP に切り替えるように EEM アプレットを設定する例を示します。

次に、メモリ リークの原因となるソフトウェア障害に対する予防措置を実施する例を示します。ここで実行されるアクションは、メモリ リークの可能性が検出されたときに、冗長 RP へ切り替えることによってダウンタイムを削減することを意図しています。

次の図は、EEM イメージを実行しているデュアル RP デバイスを示しています。EEM アプレットは、**event manager applet** コマンドを使用して CLI によって登録されています。プライマリ RP の使用可能なメモリが、指定されたしきい値 5,120,000 バイトを下回ったときに、アプレットは実行されます。アプレットのアクションは、利用可能なメモリのバイト数を示すメッセージを syslog に書き込み、セカンダリ RP へスイッチします。

Figure 15: デュアル *RP* トポロジ



ポリシーの登録に使用されるコマンドは、次のとおりです。

```
event manager applet memory-demo
 event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val 5120000
poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory is
$_snmp_oid_val bytes"
 action 2.0 force-switchover
```

登録済みのアプレットは、**show event manager policy registered** コマンドを使用して表示できます。

```
Device# show event manager policy registered
No.  Type    Event Type          Time Registered           Name
1    applet  snmp                Thu Jan30  05:57:16 2003  memory-demo
 oid {1.3.6.1.4.1.9.9.48.1.1.1.6.1} get-type exact entry-op lt entry-val {5120000}
poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory is
$_snmp_oid_val bytes"
 action 2.0 force-switchover
```

この例を示すため、デバイスでメモリを強制的に枯渇させ、一連の **show memory** コマンドを実行させてメモリの枯渇を監視します。

```
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212348444   119523060    92825384    92825384    92365916
Fast        53565260      131080       70360       60720       60720       60668
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212364664   164509492    47855172    47855172    47169340
Fast        53565260      131080       70360       60720       60720       60668
Device# show memory
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   53585260   212369492   179488300    32881192    32881192    32127556
Fast        53565260      131080       70360       60720       60720       60668
```

しきい値に達したときに、EEM イベントがトリガーされます。memory-demo という名前のアプレットが実行され、これによって、syslog メッセージがコンソールに出力され、セカンダリ RP へのスイッチが発生します。次のメッセージが記録されます。

```
00:08:31: %HA_EM-2-LOG: memory-demo: Memory exhausted; current available memory is
4484196 bytes
00:08:31: %HA_EM-6-FMS_SWITCH_HARDWARE: fh_io_msg: Policy has requested a hardware
switchover
```

次に、プライマリ RP とセカンダリ（冗長）RP の両方での **show running-config** コマンドの出力の一部を示します。

```
redundancy
 mode sso
.
.
!
event manager applet memory-demo
 event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val
5120000 poll-interval 90
 action 1.0 syslog priority critical msg "Memory exhausted; current available memory
is $_snmp_oid_val bytes"
 action 2.0 force-switchover
```

SNMP 通知イベント ディテクタ

次に、**event snmp-notification** を設定する前に、**snmp-server community** パブリック RW コマンドと **snmp-server manager** コマンドを設定する例を示します。

```
snmp-server community public RW
 snmp-server manager
```

次に、値が 10 であるオブジェクト ID 1 の宛先 IP アドレス 192.168.1.1 でデバイスが SNMP 通知を受け取ったときに、EEM スクリプトを実行するように SNMP_Notification という名前の EEM アプレットを登録する例を示します。

```
event manager applet SNMP_Notification
 event snmp-notification dest_ip_address 192.168.1.1 oid 1 op eq oid-value 10
 action 1 policy eem_script
```

syslog イベント ディテクタ

次に、syslog がイーサネット インターフェイス 1/0 のダウンを認識したときに実行する EEM アプレットを指定する例を示します。アプレットはインターフェイスに関するメッセージを syslog に送信します。

```
event manager applet interface-down
 event syslog pattern "".*UPDOWN.*Ethernet1/0.*" occurs 4
 action 1.0 syslog msg "Ethernet interface 1/0 changed state 4 times"
```

Embedded Event Manager アプレットの設定例

ID イベント ディテクタの例

次に、「EventIdentity」というポリシーが、ファスト イーサネット インターフェイス 0 で認証が成功するたびにトリガーされる例を示します。

```
event manager applet EventIdentity
 event identity interface FastEthernet0 authc success
action 1.0 syslog msg "Applet EventIdentity"
```

MAT イベント ディテクタの例

次に、「EventMat」というポリシーが、mac-address-table で MAC アドレスが学習されるたびにトリガーされる例を示します。

```
event manager applet EventMat
 event mat interface FastEthernet0
action 1.0 syslog msg "Applet EventMat"
```

ネイバー検出イベント ディテクタの例

次に、「EventNeighbor」というポリシーが、Cisco Discovery Protocol（CDP）キャッシュ エントリが変化するときにトリガーされる例を示します。

```
event manager applet EventNeighbor
 event neighbor-discovery interface FastEthernet0 cdp all
action 1.0 syslog msg "Applet EventNeighbor"
```

Embedded Event Manager の手動によるポリシー実行の例

次に、手動で実行する EEM ポリシー（アプレットまたはスクリプト）の設定に None イベント ディテクタを使用する例を示します。

イベント マネージャ run コマンドの使用

次に、**event manager run** コマンドを使用して、手動でポリシーを実行する例を示します。ポリシーはアプレット コンフィギュレーション モードで **event none** コマンドを使用して登録さ

れてから、グローバル コンフィギュレーション モードで **event manager run** コマンドを使用して実行されます。

```
event manager applet manual-policy
 event none
 action 1.0 syslog msg "Manual-policy triggered"
 end
!
event manager run manual-policy
```

action policy コマンドの使用

次に、**action policy** コマンドを使用して、手動でポリシーを実行する例を示します。ポリシーはアプレット コンフィギュレーション モードで **event none** コマンドを使用して登録されてから、アプレット コンフィギュレーション モードで **action policy** コマンドを使用して実行されます。

```
event manager applet manual-policy
 event none
 action 1.0 syslog msg "Manual-policy triggered"
 exit
!
event manager applet manual-policy-two
 event none
 action 1.0 policy manual-policy
 end
!
event manager run manual-policy-two
```

Embedded Event Manager Watchdog System Monitor（Cisco IOS）イベント ディテクタの設定例

次に、Cisco IOS watchdog system monitor (IOSWDSysMon) イベント ディテクタの動作を具体的に表示する 3 個の EEM アプレットの設定例を示します。

Watchdog System Monitor サンプル 1 ポリシー

第 1 のポリシーは、IP Input という名前のプロセスの平均 CPU 使用率が 10 秒間 1% 以上になったときにアプレットをトリガーします。

```
event manager applet IOSWD_Sample1
 event ioswdsysmon sub1 cpu-proc taskname "IP Input" op ge val 1 period 10
 action 1.0 syslog msg "IOSWD_Sample1 Policy Triggered"
```

Watchdog System Monitor サンプル 2 ポリシー

第 2 のポリシーは、Net Input という名前のプロセスによる合計メモリ使用量が 100 kb を超えたときアプレットをトリガーします。

```
event manager applet IOSWD_Sample2
 event ioswdsysmon sub1 mem-proc taskname "Net Input" op gt val 100 is-percent false
 action 1.0 syslog msg "IOSWD_Sample2 Policy Triggered"
```

Watchdog System Monitor サンプル 3 ポリシー

第 3 のポリシーは、IP RIB Update という名前のプロセスによる合計メモリ使用量が、60 秒のサンプリング時間全体で、50% を超えて増加したときにアプレットをトリガーします。

```
event manager applet IOSWD_Sample3
 event ioswdsysmon sub1 mem-proc taskname "IP RIB Update" op gt val 50 is-percent true
period 60
 action 1.0 syslog msg "IOSWD_Sample3 Policy Triggered"
```

3 個のポリシーが設定され、複数のワークステーションからネットワーキング デバイスに対して繰り返し大量の ping が実行されます。そのためネットワーキング デバイスは一定の利用量を記録します。これにより、ポリシー 1 およびポリシー 2 がトリガーされ、コンソールに次のメッセージが表示されます。

```
00:42:23: %HA_EM-6-LOG: IOSWD_Sample1: IOSWD_Sample1 Policy Triggered
00:42:47: %HA_EM-6-LOG: IOSWD_Sample2: IOSWD_Sample2 Policy Triggered
```

登録したポリシーを表示するには、**show event manager policy registered** コマンドを使用します。

```
Device# show event manager policy registered
No.  Class   Type    Event Type          Trap  Time Registered           Name
1    applet  system  ioswdsysmon         Off   Fri Jul 23 02:27:28 2004  IOSWD_Sample1
 sub1 cpu_util {taskname {IP Input} op ge val 1 period 10.000 }
 action 1.0 syslog msg "IOSWD_Sample1 Policy Triggered"
2    applet  system  ioswdsysmon         Off   Fri Jul 23 02:23:52 2004  IOSWD_Sample2
 sub1 mem_used {taskname {Net Input} op gt val 100 is_percent FALSE}
 action 1.0 syslog msg "IOSWD_Sample2 Policy Triggered"
3    applet  system  ioswdsysmon         Off   Fri Jul 23 03:07:38 2004  IOSWD_Sample3
 sub1 mem_used {taskname {IP RIB Update} op gt val 50 is_percent TRUE period 60.000 }
 action 1.0 syslog msg "IOSWD_Sample3 Policy Triggered"
```

SNMP ライブラリ拡張の設定例

SNMP get オペレーションの例

次に、get 要求をローカル ホストに送信する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact
community
 public
Device(config-applet)# action 1.3 info type snmp oid
 1.3.6.1.2.1.1.4.0 get-type next community
public
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d03h:%HA_EM-6-LOG: lg: 1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgn: 1.3.6.1.2.1.1.5.0
```

次に、get 要求をリモート ホストに送信する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 get-type next community
 public ipaddr
 172.17.16.69
Device(config-applet)# action 1.3 info type snmp getid
 1.3.6.1.2.1.1.1.0 community
 public ipaddr
 172.17.16.69
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d03h:%HA_EM-6-LOG: lg: 1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgn: 1.3.6.1.2.1.1.5.0
```

SNMP GetID オペレーションの例

次に、getid 要求をローカル ホストに送信する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp getid
 community
 public
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_oid=1.3.6.1.2.1.1.5.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_value=jubjub.cisco.com
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_oid=1.3.6.1.2.1.1.6.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_value=
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysdescr_oid=1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_oid=1.3.6.1.2.1.1.2.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_value=products.222
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=1.3.6.1.2.1.1.3.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=10131676
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_oid=1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_value=YYY
```

次に、getid 要求をリモート ホストに送信する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
```

```
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp getid
 1.3.6.1.2.1.1.1.0 community
 public ipaddr
 172.17.16.69
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_oid=1.3.6.1.2.1.1.5.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysname_value=jubjub.cisco.com
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_oid=1.3.6.1.2.1.1.6.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syslocation_value=
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysdescr_oid=1.3.6.1.2.1.1.1.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_oid=1.3.6.1.2.1.1.2.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysobjectid_value=products.222
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=1.3.6.1.2.1.1.3.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_sysuptime_oid=10131676
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_oid=1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lgid: _info_snmp_syscontact_value=YYY
```

set オペレーションの例

次に、set オペレーションをローカル ホストで実行する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 set-type
 integer
 5 sysName.0 community
 public
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d04h:%HA_EM-6-LOG: lset: 1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lset: XXX
```

次に、set オペレーションをリモート ホストで実行する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
1.3.6.1.2.1.1.1.0 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp oid
1.3.6.1.2.1.1.4.0 set-type integer
 5 sysName.0 community
 public ipaddr
172.17.16.69
```

次のログ メッセージが SNMP イベント マネージャ ログに書き込まれます。

```
1d04h:%HA_EM-6-LOG: lset: 1.3.6.1.2.1.1.4.0
1d04h:%HA_EM-6-LOG: lset: XXX
```

SNMP 通知の生成の例

次に、sysUpTime.0 変数の SNMP トラップを設定する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op
lt entry-val
 5120000 poll-interval
90
Device(config-applet)# action 1.3 info type snmp var
 sysUpTime.0 oid
1.3.6.1.4.1.9.9.43.1.1.6.1.3.41 integer
 2
Device(config-applet)# action 1.4 info type snmp trap
 enterprise-oid
 ciscoSyslogMIB.2 generic-trapnum
6 specific-trapnum
1 trap-oid
 1.3.6.1.4.1.9.9.41.2.0.1 trap-var
sysUpTime.0
```

debug snmp packets コマンドがイネーブルにされている場合、次の出力が生成されます。

```
Device# debug snmp packets
1d04h: SNMP: Queuing packet to 172.69.16.2
1d04h: SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 172.19.rap 1
clogHistoryEntry.3 = 4
clogHistoryEntry.6 = 9999
1d04h: SNMP: Queuing packet to 172.19.208.130
1d04h: SNMP: V1 Trap, ent ciscoSyslogMIB.2, addr 172.19.rap 1
clogHistoryEntry.3 = 4
clogHistoryEntry.6 = 9999
1d04h: SNMP: Packet sent via UDP to 172.69.16.2
1d04h: SNMP: Packet sent via UDP to 172.69.16.2
infra-view10:
Packet Dump:
30 53 02 01 00 04 04 63 6f 6d 6d a4 48 06 09 2b
06 01 04 01 09 09 29 02 40 04 ac 13 d1 17 02 01
06 02 01 01 43 04 00 9b 82 5d 30 29 30 12 06 0d
2b 06 01 04 01 09 09 29 01 02 03 01 03 02 01 04
30 13 06 0d 2b 06 01 04 01 09 09 29 01 02 03 01
06 02 02 27 0f
Received SNMPv1 Trap:
Community: comm
Enterprise: ciscoSyslogMIBNotificationPrefix
Agent-addr: 172.19.209.23
Enterprise Specific trap.
Enterprise Specific trap: 1
Time Ticks: 10191453
clogHistSeverity = error(4)
clogHistTimestamp = 9999
```

次に、sysUpTime.0 変数の SNMP インフォーム要求を設定する例を示します。

```
Device(config)# event manager applet snmp
Device(config-applet)# event snmp oid
```

```
1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op
 lt entry-val
 5120000 poll-interval
 90
Device(config-applet)# action 1.3 info type snmp var
 sysUpTime.0 oid
1.3.6.1.4.1.9.9.43.1.1.6.1.3.41 integer
 2
Device(config-applet)# action 1.4 info type snmp inform
trap-oid
 1.3.6.1.4.1.9.9.43.2.0.1 trap-var
 sysUpTime.0 community
 public ipaddr
 172.19.209.24
```

debug snmp packets コマンドがイネーブルにされている場合、次の出力が生成されます。

```
Device# debug snmp packets
1d04h: SNMP: Inform request, reqid 24, errstat 0, erridx 0
sysUpTime.0 = 10244391
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.40 = 1
1d04h: SNMP: Packet sent via UDP to 172.19.209.24.162
1d04h: SNMP: Packet received via UDP from 172.19.209.24 on FastEthernet0/0
1d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
1d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
1d04h: SNMP: Inform request, reqid 25, errstat 0, erridx 0
sysUpTime.0 = 10244396
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.41 = 2
1d04h: SNMP: Packet sent via UDP to 172.19.209.24.162
1d04h: SNMP: Packet received via UDP from 172.19.209.24 on FastEthernet0/0
1d04h: SNMP: Response, reqid 25, errstat 0, erridx 0
1d04h: SNMP: Response, reqid 25, errstat 0, erridx 0
Device# debug snmp packets
5d04h: SNMP: Packet received via UDP from 172.19.209.23 on FastEthernet0/0
5d04h: SNMP: Inform request, reqid 24, errstat 0, erridx 0
sysUpTime.0 = 10244391
snmpTrapOID.0 = ciscoConfigManMIB.2.0.1
ccmHistoryEventEntry.3.40 = 1
5d04h: dest if_index = 1
5d04h: dest ip addr= 172.19.209.24
5d04h: SNMP: Response, reqid 24, errstat 0, erridx 0
5d04h: SNMP: Packet sent via UDP to 172.19.209.23.57748
5d04h: SNMP: Packet received via UDP from 172.19.209.23 on FastEthernet0/0
5d04h: SNMP: Inform request, reqid 25, errstat 0, erridx 0
```

EEM アプレットの可変ロジックの設定例

このセクションでは、一部の選択された action コマンドの例を示します。アプレット内の可変ロジックをサポートするすべての action コマンドについては、次の表を参照してください。

この例では、条件付きのループである **while**、**if** および **foreach** を使用してデータを出力します。**action divide**、**action increment** および **action puts** のようなその他のアクションコマンドは、条件が満たされている場合に実行されるアクションを定義するために使用します。

```
event manager applet printdata
event none
action 100 set colors "red green blue"
```

```
action 101 set shapes "square triangle rectange"
action 102 set i "1"
action 103 while $i lt 6
action 104  divide $i 2
action 105  if $_remainder eq 1
action 106   foreach _iterator "$colors"
action 107    puts nonewline "$_iterator "
action 108   end
action 109   puts ""
action 110  else
action 111   foreach _iterator "$shapes"
action 112    puts nonewline "$_iterator "
action 113   end
action 114   puts ""
action 115  end
action 116  increment i
action 117 end
```

イベント マネージャ アプレット ex が実行されると、次の出力が得られます。

```
event manager run printdata
red green blue
square triangle rectange
red green blue
square triangle rectange
red green blue
```

次の例では、2 個の環境変数、poll_interface と max_rx_rate が、それぞれ、F0/0 と 3 に設定されます。30 秒ごとに、インターフェイスで rx 比率の調査が行われます。rx 比率がしきい値を上回った場合は、syslog メッセージが表示されます。

このアプレットは、インターフェイスの調査に foreach 条件付き文を使用します。また、RXPS に属する値を EEM 環境変数に設定された max_rx_rate と比較するために、if 条件付きブロックを使用します。

```
event manager environment poll_interfaces F0/0
event manager environment max_rx_rate 3
ev man app check_rx_rate
ev timer watchdog name rx_timer time 30
action 100 foreach int $poll_interfaces
action 101  cli command "en"
action 102  cli command "show int $int summ | beg ------"
action 103  foreach line $_cli_result "\n"
action 105  regexp ".*[0-9]+\s+[0-9]+\s+[0-9]+\s+[0-9]+\s+[0-9]+\s+([0-9]+)\s+.*" $line
 junk rxps
action 106   if $_regexp_result eq 1
action 107    if $rxps gt $max_rx_rate
action 108     syslog msg "Warning rx rate for $int is > than threshold. Current value
is $rxps
 (threshold is $max_rx_rate)"
action 109    end
action 110   end
action 111  end
action 112 end
```

syslog メッセージ例：

```
Oct 16 09:29:26.153: %HA_EM-6-LOG: c: Warning rx rate for F0/0 is > than threshold.
 Current value is 4 (threshold is 3)
The output of show int F0/0 summ is of the format:
```

```
#show int f0/0 summ

 *: interface is up
 IHQ: pkts in input hold queue     IQD: pkts dropped from input queue
 OHQ: pkts in output hold queue    OQD: pkts dropped from output queue
 RXBS: rx rate (bits/sec)          RXPS: rx rate (pkts/sec)
 TXBS: tx rate (bits/sec)          TXPS: tx rate (pkts/sec)
 TRTL: throttle count

  Interface               IHQ   IQD  OHQ   OQD  RXBS RXPS  TXBS TXPS TRTL
------------------------------------------------------------------------------
* FastEthernet0/0          0 87283    0     0     0    0     0    0    0
```

Note アプレット内の可変ロジックをサポートするその他の action コマンドを使用するには、次の表にあるコマンドを使用してください。

Table 57: 使用できる ***action*** コマンド

Action コマンド	目的
action add	EEM アプレットがトリガーされたときに
action append	EEM アプレットがトリガーされたときに、
action break	EEM アプレットがトリガーされたときに、
action comment	EEM アプレットがトリガーされたときに
action context retrieve	EEMアプレットがトリガーされたときに、します。
action context save	EEM アプレットがトリガーされたときに、
action continue	EEM アプレットがトリガーされたときに、
action decrement	EEM アプレットがトリガーされたときに、
action divide	EEM アプレットがトリガーされたときに、
action else	EEM アプレットがトリガーされたときに、ブロックの開始を指定します。
action elseif	EEM アプレットがトリガーされたときに、ブロックの開始を特定します。
action end	EEM アプレットがトリガーされたときに、クション ブロック終了の ID を指定します
action exit	EEM アプレットがトリガーされたときに、ことを指定します。

Action コマンド	目的
action foreach	EEMアプレットがトリガーされたときに、デ の反復を指定します。
action gets	EEM アプレットがトリガーされたときに、同 変数に値を格納します。
action if	EEM アプレットがトリガーされたときに、if
action if goto	EEM アプレットがトリガーされたときに、指 にジャンプすることを指示します。
action increment	EEM アプレットがトリガーされたときに、変
action info type interface-names	EEM アプレットがトリガーされたときに、イ
action info type snmp getid	SNMP get オペレーション中に簡易ネットワー ます。
action info type snmp inform	EEM アプレットがトリガーされたときに、SI
action info type snmp oid	EEM アプレットがトリガーされたときに、SI ション中に取得するオブジェクトを指定しま
action info type snmp trap	EEM アプレットがトリガーされたときに、SI
action info type snmp var	SNMP オブジェクト ID（OID）の変数、およ
action multiply	EEM アプレットがトリガーされたときに、変
action puts	EEMアプレットがトリガーされたときにデー
action regexp	EEM アプレットがトリガーされたときに入力 す。
action set（EEM）	EEM アプレットがトリガーされたときに変数
action string compare	EEM アプレットがトリガーされたときに 2 個
action string equal	EEM アプレットがトリガーされたときに 2 個 す。
action string first	EEMアプレットがトリガーされたときにstrin ンを指定します。
action string index	EEMアプレットがトリガーされたときに与え 定します。
action string last	EEM アプレットがトリガーされたときに strin ションを指定します。

Action コマンド	目的
action string length	EEM アプレットがトリガーされたときに
action string match	EEMアプレットがトリガーされたときに ンを指定します。
action string range	EEM アプレットがトリガーされたときに
action string replace	EEMアプレットがトリガーされたときに 格納するアクションを指定します。
action string tolower	EEMアプレットがトリガーされたときに ます。
action string toupper	EEMアプレットがトリガーされたときに ます。
action string trim	EEM アプレットがトリガーされたときに
action string trimleft	EEMアプレットがトリガーされたときに、 を指定します。
action string trimright	EEMアプレットがトリガーされたときに、 を指定します。
action subtract	EEM アプレットがトリガーされたときに、
action while	EEMアプレットがトリガーされたときに す。

イベント SNMP オブジェクトの設定例

次の例は、SET オペレーション、および、設定される値が $_snmp_value にありスクリプトで管理されることを示します。次の例は、oid とその値を、後で取得されるコンテキストとして格納します。

```
event manager applet snmp-object1
 description "APPLET SNMP-OBJ-1"
 event snmp-object oid 1.3.6.1.2.1.31.1.1.1.18 type string sync no skip no istable yes
default 0
 action 1 syslog msg "SNMP-OBJ1:TRIGGERED" facility "SNMP_OBJ"
 action 2 context save key myoid variable "_snmp_oid"
 action 3 context save key myvalue variable "_snmp_value"
```

EEM アプレットの説明の設定例

次に、簡易ネットワーク管理プロトコル（SNMP）のサンプリングによって実行される Embedded Event Manager（EEM）アプレットの説明を追加または変更する例を示します。

```
event manager applet test
 description "This applet looks for the word count in syslog messages"
 event syslog pattern "count"
 action 1 syslog msg hi
```

その他の参考資料

ここでは、Cisco IOS CLI を使用した EEM ポリシーの記述に関する関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
EEM コマンド：コマンド構文の詳細、デフォルト、コマンド モード、コマンド履歴、使用上の注意事項、および例	Cisco IOS Embedded Event Manager のコマンド リファレンス
Embedded Event Manager 概要	「Embedded Event Manager の概要」の章
Tcl を使用して Embedded Event Manager ポリシーを記述する	「Tcl を使用した Embedded Event Manager ポリシーの記述」の章
拡張オブジェクト トラッキングの設定	「Configuring Enhanced Object Tracking」の章

標準

標準	タイトル
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	--

MIB

MIB	MIB のリンク
CISCO-EMBEDDED-EVENT-MGR-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
新しい RFC または変更された RFC はサポートされていません。また、既存の RFC に対するサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Cisco IOS CLI を使用した EEM 4.0 ポリシーの記述の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 58: Cisco IOS CLI を使用した EEM 4.0 ポリシーの記述の機能情報

機能名	リリース	機能情報
Embedded Event Manager 1.0	12.0(26)S 12.3(4)T	EEM 1.0 は、Embedded Event Manager アプレット作成を SNMP イベント ディテクタ Syslog イベント ディテクタとともに追加しました。EEM 1.0 は、次のアクションも追加しました。優先化された syslog メッセージの生成、Cisco CNS デバイスによるアップストリーム処理に対し CNS イベントの生成、Cisco ソフトウェアのリロード、および完全冗長ハードウェア構成におけるセカンダリプロセッサへのスイッチング。 次のコマンドがこの機能で導入されました。**action cns-event**、**action force-switchover**、 **action reload**、 **action syslog**、 **debug event manager**、 **event manager applet**、 **event snmp**、 **event syslog**、 **show event manager policy registered**。
Embedded Event Manager 2.0	12.2(25)S	EEM 2.0 は、Application-Specific イベント ディテクタ、Counter イベント ディテクタ、Interface Counter イベント ディテクタ、Timer イベント ディテクタ、および watchdog イベント ディテクタを追加しました。新しいアクションには、名前付きカウンタの変更、アプリケーション固有イベントのパブリッシュ、SNMP トラップの生成が含まれました。環境変数定義機能、および、Tcl を使用して記述されたサンプル EEM ポリシーの実行機能が追加され、2 個のサンプル ポリシーがソフトウェアに追加されました。 次のコマンドがこの機能で導入されました。**action counter**、**action publish-event**、**action snmp-trap**、**event application**、**event counter**、**event interface**、**event ioswdsysmon**、**event manager environment**、**event manager history size**、**event manager policy**、**event manager scheduler suspend**、**event timer**, **show event manager environment**、**show event manager history events**、**show event manager history traps**、**show event manager policy available**、**show event manager policy pending**。

機能名	リリース	機能情報
Embedded Event Manager 2.1	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	EEM 2.1 は複数の新しいイベント ディテクタおよびアクション、EEM ポリシーを手動で起動する新しい機能と複数の共存ポリシーを起動する機能を追加しました。簡易ネットワーク管理プロトコル（SNMP）イベント ディテクタ比率ベース イベントのサポートが、Tool Command Language（Tcl）を使用してポリシーを作成する機能として導入されました。 次のコマンドがこの機能で導入されました。**action cli**、**action counter**、 **action info**、 **action mail**、**action policy**、**debug event manager**、 **event cli**、 **event manager directory user**, **event manager policy**、**event manager run**、**event manager scheduler script**、**event manager session cli username**、**event none**、**event oir**、**event snmp**, **event syslog**、**set**(EEM)、**show event manager directory user**、**show event manager policy registered**、**show event manager session cli username**。
Embedded Event Manager 2.1（ソフトウェア モジュール方式）	12.2(18)SXF4 Cisco IOS ソフトウェアモジュール方式のイメージ	EEM 2.1 ソフトウェア モジュール方式イメージは、GOLD、system manager、および WDSysMon（Cisco IOS Software Modularity watchdog）イベント ディテクタ、および Cisco IOS ソフトウェア モジュール方式プロセスとプロセス メトリックを表示する機能を導入しました。 次のコマンドがこの機能で導入されました。**event gold**、**event process**、**show event manager metric process**。 **Note** EEM 2.1 ソフトウェア モジュール方式イメージは、Resource イベント ディテクタおよび RF イベント ディテクタを EEM 2.2 に追加しましたが、EOT イベント ディテクタ、またはトラッキング対象オブジェクトの読み込みおよび設定のアクションをサポートしません。
Embedded Event Manager 2.2	12.4(2)T 12.2(31)SB3 12.2(33)SRB	EEM 2.2 は、Enhanced Object Tracking、Resource、および RF イベントディテクタを追加しました。トラッキング対象オブジェクトの状態の読み取りおよび設定のアクションも追加されました。 この機能により、次のコマンドが導入または変更されました。**action track read**、**action track set**、**default-state**、**event resource**、**event rf**、**event track**、**show track**、**track stub-object**。
SNMP イベント ディテクタ delta 環境変数	12.4(11)T	新しい SNMP イベント ディテクタ環境変数、_snmp_oid_delta_val が追加されました。 これはマイナーな拡張です。マイナーな拡張は、通常 Feature Navigator に記載されません。

機能名	リリース	機能情報
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB 15.1(2)SY	EEM 2.3 では、Cisco Catalyst 6500 シリーズ スイッチ上の Generic Online Diagnostics (GOLD) イベント ディテクタに関連する新しい機能が追加されました。 **event gold** コマンドは、GOLD テスト失敗および条件への対応を改善するための **action-notify**、**testing-type**、**test-name**、**test-id**、**consecutive-failure**、**platform-action**、および **maxrun** キーワードが追加され、拡張されました。 検出されたイベントのプラットフォーム全体、および、テスト特有の GOLD イベントディテクタ情報へのアクセスを実現するために、読み取り専用変数が **GOLD Event Detector** カテゴリに追加されました。
Embedded Event Manager 2.4	12.4(20)T 12.2(33)SXI 12.2(33)SRE 15.1(2)SY	EEM 2.4 は Cisco IOS Release 12.4(20)T 以降のリリースでサポートされ、複数の新しい機能が追加されました。 この機能により、次のコマンドが追加されました。 **attribute (EEM)** 、**correlate**、**event manager detector rpc**、**event manager directory user repository**、**event manager update user policy**、 **event manager scheduler clear**、**event manager update user policy**、**event owner**、**event rpc**、**event snmp-notification**、**show event manager detector**、**show event manager version**、**trigger (EEM)**。

機能名	リリース	機能情報
Embedded Event Manager 3.0	12.4(22)T 12.2(33)SRE 12.2(50)SY	EEM 3.0 は、Cisco IOS Release 12.4(22)T 以降のリリースでサポートされ、複数の新しい機能が追加されました。 この機能により、次のコマンドが導入または変更されました。 **action add** 、**action append**、**action break**、**action comment**、**action context retrieve**、**action context save**、**action continue**、**action decrement**、**action divide**、**action else**、**action elseif**、**action end**、**action exit**、**action foreach**、**action gets**、**action if**, **action if goto**、**action increment**、**action info type interface-names**、**action info type snmp getid**、**action info type snmp inform**、**action info type snmp oid**、**action info type snmp trap**、**action info type snmp var**、**action multiply**、**action puts**、**action regexp**、**action set (EEM)**、**action string compare**、**action string equal**、**action string first**、**action string index**、**action string last**、**action string length**、**action string match**、**action string range**、**action string replace**、**action string tolower**、**action string toupper**、**action string trim**、**action string trimleft**、**action string trimright**、**action subtract**、**action while**、**event cli**、**event ipsla**、**event manager detector routing**、**event manager scheduler**、**event manager scheduler clear**、**event manager scheduler hold**、**event manager scheduler modify**、**event manager scheduler release**、**event nf**、**event routing**、**show event manager policy active**、**show event manager policy pending**、および **show event manager scheduler**。
Embedded Event Manager 3.1	15.0(1)M 15.1(1)SY 15.1(2)SY	EEM 3.1 は、Cisco IOS Release 15.0(1)M 以降のリリースでサポートされ、複数の新しい機能が追加されました。 この機能により、次のコマンドが導入または変更されました。**action syslog**、**description (EEM)**、**event manager applet**、**event manager policy**、**event snmp-notification**、**event snmp-object**、**show event manager policy registered**、および **show event manager policy available** 。
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG 15.1(3)T 15.1(1)SY 15.1(2)SY	EEM は、イベント検出と回復を Cisco IOS 内部で直接行うための分散型でカスタマイズされた手法です。 この機能に関する詳細については、次の各項を参照してください。 次のコマンドが導入または変更されました。**debug event manager**、**event identity**、**event mat**、**event neighbor-discovery**、**show event manager detector**。

機能名	リリース	機能情報
Embedded Event Manager 4.0	15.2(2)T 15.1(1)SY 15.1(2)SY	EEM 4.0 は 15.2(2)T 以降のリリースでサポートされ、いくつかの新機能が導入されました。 次のコマンドが導入または変更されました。**action file**、**action mail**、**action syslog**、**clear event manager detector counters**、**clear event manager server counters**、**event cli**、**event manager policy**、**event manager scheduler**、**event syslog**、**show event manager detector**、**show event manager policy registered**、**show event manager statistics**。

CHAPTER 37

Writing Embedded Event Manager Policies Using Tcl

この章では、ソフトウェア開発者が Tool command language（Tcl）スクリプトを使用して Embedded Event Manager（EEM）ポリシーを記述およびカスタマイズし、Cisco ソフトウェアの障害とイベントを処理できるようにする方法について説明します。EEM は、定義済みの Application Programming Interface（API）を介してレポートされる Cisco ソフトウェア システムの障害による、ポリシー方式のプロセスです。EEM ポリシー エンジンは、障害およびその他のイベントが発生したときに通知を受け取ります。EEM ポリシーは、システムの現在の状態に基づいて回復を実行し、該当するイベントのポリシーに指定されたアクションを実行します。回復アクションはポリシーが実行されたときにトリガーされます。

Tcl を使用した Embedded Event Manager ポリシーの記述に関する前提条件

- EEM ポリシーを記述するには、その前に「Embedded Event Manager Overview」の章を理解しておく必要があります。
- コマンドライン インターフェイス（CLI）コマンドを使用して EEM ポリシーを記述するときは、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章をよく理解しておいてください。

Tcl を使用した Embedded Event Manager ポリシー記述について

EEM ポリシー

EEM では、イベントをモニターし、イベント発生が検出されたとき、およびしきい値を超えたときに、情報通知や是正などの任意のアクションを実施できます。EEM ポリシーは、イベントおよびイベントが発生した場合に行う処理を定義するエンティティです。EEM ポリシーにはアプレットとスクリプトの2つのタイプがあります。アプレットは、コマンドラインインターフェイス（CLI）設定に定義された、ポリシーの単純な形式です。スクリプトは、Tool Command Language（Tcl）で記述されたポリシーの形式です。

EEM アプレット

EEM アプレットは、イベント スクリーニング基準とイベント発生時に実行するアクションを定義する簡潔な方法です。EEM アプレット コンフィギュレーション モードでは、3 種類のコンフィギュレーション文がサポートされます。event コマンドを使用して実行するアプレットをトリガーするイベント基準を指定し、action コマンドを使用して、EEM アプレットがトリガーされるときに実行されるアクションを指定し、**set** コマンドを使用して EEM アプレット変数の値を設定します。現在、_exit_status 変数だけが、**set** コマンドでサポートされます。

アプレット コンフィギュレーションでは、event コンフィギュレーション コマンドを 1 つだけ使用できます。アプレット コンフィギュレーション サブモードが終了し、event コマンドが存在しない場合は、アプレットにイベントが割り当てられていないことを示す警告が表示されます。イベントが指定されない場合、アプレットは登録されたと見なされません。アプレットにアクションが割り当てられない場合、イベントはトリガーされますが、アクションは実行されません。1 つのアプレット コンフィギュレーション内で複数の action コンフィギュレーション コマンドが使用できます。登録済みのアプレットを表示するには、**show event manager policy registered** コマンドを使用します。

EEM アプレットを修正する前に、アプレット コンフィギュレーション モードを終了するまで既存のアプレットを置き換えられないことに注意してください。アプレットコンフィギュレーションモードでアプレットを修正中であっても、既存のアプレットを実行できます。変更は一時ファイルに書き込まれるため、登録を解除しないでアプレットを変更するのが安全です。アプレット コンフィギュレーション モードを終了すると、古いアプレットが登録解除され、新しいバージョンが登録されます。

アプレット内の action コンフィギュレーション コマンドは、*label* 引数を使用することで一意に識別できます。label 引数には任意の文字列値が使用できます。アクションは、*label* 引数をソートキーとして、アプレット内で英数字のキーの昇順に並べ替えられ、この順序で実行されます。同じ *label* 引数を異なるアプレットで使用できます。ラベルは 1 つのアプレット内でのみ一意にする必要があります。

Embedded Event Manager は、ポリシーそのものに含まれるイベント仕様に基づいてポリシーをスケジューリングし、実行します。アプレット コンフィギュレーション モードが終了するとき、EEM は、入力された event コマンドと action コマンドを検査し、指定されたイベントの発生時に実行されるようにアプレットを登録します。

Cisco IOS CLI を使用して EEM ポリシーを記述する方法については、「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章を参照してください。

EEM スクリプト

すべての Embedded Event Manager スクリプトは、Tcl で記述されます。Tcl は文字列ベースのコマンド言語で、実行時に解釈されます。Tclがサポートされるバージョンは、Tclバージョン 8.3.4 に、スクリプト サポートが追加されたものです。スクリプトは、ネットワーキング デバイスではなく、別のデバイスで、ASCIIエディタを使用して定義されます。続いてスクリプトはネットワーキング デバイスにコピーされ EEM に登録されます。Tcl スクリプトは EEM でサポートされます。強制適用される規則としての Embedded Event Manager ポリシーは、経過時間 20 秒未満で解釈および実行される必要がある、存続時間の短い実行時ルーチンです。20 秒よりも長い経過時間が必要な場合、event_register 文で maxrun パラメータを使用して、必要な値を指定する必要があります。

EEM ポリシーでは、すべての Tcl 言語機能が使用されます。ただし、シスコでは、EEM ポリシーの記述に活用できる Tcl コマンド拡張の形式で、Tcl 言語の機能を拡張しています。Tcl コマンド拡張のキーワードの主要なカテゴリでは、検出されたイベント、後続のアクション、ユーティリティ情報、カウンタの値、システム情報が特定されます。

EEM では、Tcl を使用して独自のポリシーを記述、実装できます。EEM スクリプトの記述には、次の作業が含まれます。

- ポリシーの実行時に決定に使用される基準を確立する、イベント Tcl コマンド拡張の選択。
- イベントの検出に関連付けられているイベント ディテクタ オプションの定義。
- 検出されたイベントのリカバリまたは検出されたイベントに対する応答を実装するアクションを選択すること。

EEM ポリシーの Tcl コマンド拡張のカテゴリ

EEM ポリシーの Tcl コマンド拡張には、さまざまなカテゴリがあります。

Note すべての EEM ポリシーで使用するこれらの各カテゴリで使用可能な Tcl コマンドは、この資料の以降の項で説明します。

Table 59: EEM ポリシーの *Tcl* コマンド拡張のカテゴリ

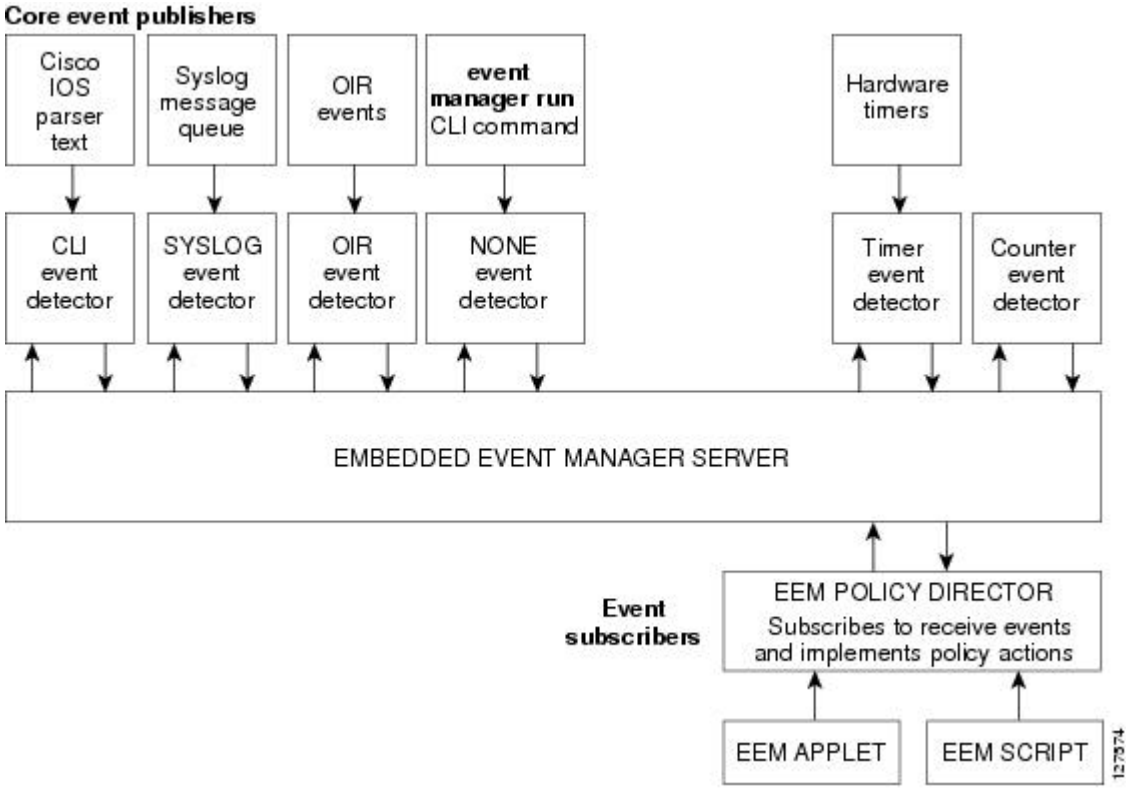
カテゴリ	定義
EEM イベントの Tcl コマンド拡張（イベント情報、イベント登録、イベント パブリッシュの 3 タイプ）	このカテゴリは、イベント固有のコマンドの **event_register_** *xxx* ファミリによって表されます。このカテゴリには、別のイベント情報 Tcl コマンド拡張の **event_reqinfo** もあります。これは、イベントについての情報を EEM に問い合わせるためにポリシーで使用されるコマンドです。アプリケーション固有のイベントをパブリッシュする、EEM イベントパブリッシュ Tcl コマンド拡張 **event_publish**> もあります。
EEM アクションの Tcl コマンド拡張	これらの Tcl コマンド拡張（たとえば、**action_syslog** など）は、イベントまたは障害への応答か、あるいは、イベントまたは障害からの回復のために、ポリシーによって使用されます。これらの拡張に加え、開発者は、Tcl 言語を使用して、必要なアクションを実装できます。
EEM ユーティリティの Tcl コマンド拡張	これらの Tcl コマンド拡張は、アプリケーション情報、カウンタ、またはタイマーの取得、保存、設定、または変更に使用されます。
EEM システム情報の Tcl コマンド拡張	このカテゴリは、システム固有の情報コマンドの **sys_reqinfo _** *xxx* ファミリによって表されます。これらのコマンドは、システム情報を収集する目的で、ポリシーによって使用されます。
EEM コンテキストの Tcl コマンド拡張	これらの Tcl コマンド拡張は、Tcl コンテキスト（可視変数およびその値）の保存および取得に使用されます。

EEM イベントの検出および回復の一般的なフロー

EEM は、イベント ディテクタと呼ばれるソフトウェア エージェントを使用してシステム内の異なるコンポーネントのモニターリングをサポートする、柔軟でポリシードリブンのフレームワークです。次の図に、EEM サーバー、コア イベント パブリッシャ（イベント ディテクタ）、およびイベント サブスクライバ（ポリシー）の関係を示します。基本的に、イベント パブリッシャはイベントをスクリーニングして、イベント サブスクライバから提供されたイベント仕様に一致したときにイベントをパブリッシュします。イベント ディテクタは、注目するイベントが発生したときに EEM サーバーに通知します。

イベントまたは障害が検出されると、Embedded Event Manager によって、たとえば次の図の OIR イベント パブリッシャなどのイベント パブリッシャから、検出された障害またはイベントの登録が発生しているかどうかが判断されます。EEM によって、イベント登録情報が、イベント データそのものと、照会されます。ポリシーによって、検出されたイベントが Tcl コマンド拡張 **event_register _** *xxx* で登録されます。イベント情報 Tcl コマンド拡張 **event_reqinfo** は、検出されたイベントに関する情報について Embedded Event Manager に問い合わせるために、ポリシーで使用されます。

Figure 16: Embedded Event Manager コア イベント ディテクタ



Safe-Tcl

Safe-Tcl は、安全モードで作成されたインタープリタで、非信頼 Tcl スクリプトを実行できる、安全メカニズムです。安全インタープリタには、一部のシステム リソースへのアクセスや、ホストおよび他のアプリケーションに害が及ぼされることを防ぐ、制限されたコマンドのセットがあります。たとえば、コマンドは、重要な Cisco IOS ファイル システム ディレクトリにはアクセスできません。

シスコ定義のスクリプトはフル Tcl モードで実行されますが、ユーザー定義のスクリプトは Safe-Tcl モードで実行されます。Safe-Tcl を使用すると、シスコでは、個々の Tcl コマンドのディセーブルまたはカスタマイズを行えます。Tcl コマンドの詳細については、http://www.tcl.tk/man/ を参照してください。

次のリストにある Tcl コマンドは、一部の例外によって制約されます。各コマンドまたはコマンド キーワードに対する制約事項は、次のとおりです。

- **cd** ：制約付きの Cisco ディレクトリ名の 1 つへのディレクトリ移動はできません。
- **-- encoding** コマンド **names** 、 **encoding** 、 **convertfrom**および **encoding** が許可されます **convertto** 。 **encoding** 引数のない **encoding system** コマンドは許可されています が、**?encoding?** キーワードを使用した **encoding system** コマンドは使用できません。
- **exec** ：使用できません。

- **fconfigure** ：使用できます。
- **file** ：以下は使用できます。
  - **file dirname**
  - **file exists**
  - **file extension**
  - **file isdirectory**
  - **file join**
  - **file pathtype**
  - **file rootname**
  - **file split**
  - **file stat**
  - **file tail**
- **file** ：以下は使用できません。
  - **file atime**
  - **file attributes**
  - **file channels**
  - **file copy**
  - **file delete**
  - **file executable**
  - **file isfile**
  - **file link**
  - **file lstat**
  - **file mkdir**
  - **file mtime**
  - **file nativename**
  - **file normalize**
  - **file owned**
  - **file readable**
  - **file readlink**
  - **file rename**
  - **file rootname**
  - **file separator**
  - **file size**
  - **file system**
  - **file type**
  - **file volumes**
  - **file writable**
- **glob** ：制約付きの Cisco ディレクトリの 1 つで検索する場合、**glob** コマンドは使用できません。これ以外の場合は使用できます。
- **load** ：ユーザー ポリシー ディレクトリまたはユーザー ライブラリ ディレクトリにあるファイルのみがロードできます。静的パッケージ（たとえば、C コードで構成されるライブラリ）は、**load** コマンドではロードできません。

- **open** ：**open** コマンドは、制約付きの Cisco ディレクトリの 1 つにあるファイルでは使用できません。
- **pwd** ：**pwd** コマンドは使用できません。
- **socket** ：**socket** コマンドは使用できます。
- **source** ：**source** コマンドは、ユーザーポリシーディレクトリまたはユーザー ライブラリディレクトリにあるファイルで使用できます。

EEM 2.4 のバイトコード サポート

EEM 2.4 で、標準バイトコード スクリプト拡張子 .tbc のファイルを受け付けることによって、Bytecode Language（BCL）サポートが導入されています。Tcl バージョン 8.3.4 では、BCL が定義され、Tcl スクリプトが BCL に変換されるコンパイラが含まれています。EEM 2.4 のユーザー ポリシーおよびシステム ポリシーで有効な EEM ポリシーのファイル拡張子は、.tcl（Tcl テキスト ファイル）と .tbc（Tcl バイトコード ファイル）です。

バイトコードの Tcl スクリプトを格納すると、ポリシーの実行速度が向上します。これは、コードが事前にコンパイルされ、ポリシーサイズが小さくなり、コードを隠蔽するためです。難読化はスクリプトの変更を若干難しくし、論理を隠して知的財産権を保護します。

サポート コードおよび信頼済みコードのリリースのために別のオプションを提供するため、バイトコードのサポートが追加されています。十分に理解しているソフトウェア、信頼できるソフトウェア、またはサポートされているソフトウェアのみをネットワーク デバイスで実行することを推奨します。IOS EEM サポートの Tcl バイトコードを生成するには、TclPro バージョン 1.4 または 1.5 を使用します。

Tcl スクリプトをバイトコードに変換するには、procomp、Free TclPro Compiler の一部、または Active State Tcl Development Kit を使用できます。Tcl スクリプトを procomp を使用してコンパイルする場合、コードはスクランブルされ、.tbc ファイルが生成されます。バイトコード ファイルはプラットフォームに依存せず、Windows、Linux、および UNIX などの、TclPro を使用できるすべてのオペレーティング システムで生成できます。Procomp は TclPro の一部であり、http://www.tcl.tk/software/tclpro で入手できます。

登録の置き換え

通常の Tcl の置き換えの他に、EEM 2.3 では、EEM イベント登録ステートメントの行内の個別のパラメータを環境変数に置き換えることができます。

EEM 2.4 では、イベント登録ステートメントの行にある複数パラメータを 1 つの環境変数で置き換える機能が導入されています。

Note 1 つめの環境変数のみで、複数パラメータの置き換えがサポートされます。個別のパラメータを指定することも引き続き可能です。それを行うには最初の変数の後に追加の環境変数を追加します。

置き換えを示すために、1 つの環境変数 $_eem_syslog_statement が次のとおりに設定されています。

```
::cisco::eem::event_register_syslog pattern COUNT
```

登録の置き換えを使用すると、$_eem_syslog_statement 環境変数が、次の EEM ユーザー ポリシーで使用されます。

```
$_eem_syslog_statement occurs $_eem_occurs_val
action_syslog "this is test 3"
```

環境変数は、それらを使用するポリシーを登録する前に定義しておく必要があります。$_eem_syslog_statement 環境変数を定義するには、次を実行します。

```
Device(config)# event manager environment eem_syslog_statement
::cisco::eem::event_register_syslog pattern COUNT
Device(config)# event manager environment eem_occurs_val 2
```

EEM 用のシスコ ファイル命名規則

すべての Embedded Event Manager ポリシー名、ポリシー サポート ファイル（たとえば、E メール テンプレート ファイル）、およびライブラリ ファイル名は、シスコのファイル命名規則に従う必要があります。このため、Embedded Event Manager ポリシー ファイル名は、次の仕様に従っています。

- オプションのプレフィックス Mandatory. がある場合、これは、システム ポリシーがまだ登録されていない場合に、自動的に登録される必要があるシステム ポリシーであることを示します。たとえば、Mandatory.sl_text.tcl などです。
- 指定された 1 つめのイベントの 2 文字の省略形が含まれるファイル名の本体部（下の表を参照）、下線部、および、ポリシーをさらに示す説明フィールド部。
- ファイル名拡張子部は .tcl と定義されます。

Embedded Event Manager の E メール テンプレート ファイルは、email_template のファイル名のプレフィックスと、後続の E メール テンプレートの使用状況を示す省略形で構成されます。

Embedded Event Manager ライブラリ ファイル名は、ライブラリの使用状況を示す説明フィールドを含むファイル名の本体部と、後続の _lib、および .tcl というファイル名拡張子で構成されます。

***Table 60: 2* 文字の省略形の指定**

ap	event_register_appl
cl	event_register_cli
ct	event_register_counter
go	event_register_gold
if	event_register_interface

io	event_register_ioswdsysmon
la	event_register_ipsla
nf	event_register_nf
no	event_register_none
oi	event_register_oir
pr	event_register_process
rf	event_register_rf
rs	event_register_resource
rt	event_register_routing
rp	event_register_rpc
sl	event_register_syslog
sn	event_register_snmp
st	event_register_snmp_notification
so	event_register_snmp_object
tm	event_register_timer
tr	event_register_track
ts	event_register_timer_subscriber
wd	event_register_wdsysmon

Tcl を使用した Embedded Event Manager ポリシーの記述方法

EEM Tcl スクリプトの登録と定義

環境変数を設定し、EEMポリシーを登録するには、この作業を実行します。EEMは、ポリシーそのものに含まれるイベント仕様に基づいてポリシーをスケジューリングし、実行します。EEM ポリシーが登録されると、ソフトウェアによって、ポリシーが調べられ、指定されたイベントの発生時に実行されるよう、登録されます。

Before you begin

Tcl スクリプト言語で記述されたポリシーが使用できる状態である必要があります。サンプルポリシーを示します。使用している Cisco IOS リリースのイメージで使用可能なポリシーについては、*EEM* サンプルポリシータスクを参照してください。これらのサンプルポリシーは、システム ポリシー ディレクトリに保存されています。

SUMMARY STEPS

1. **enable**
2. **show event manager environment** [**all**| *variable-name*]
3. **configure terminal**
4. **event manager environment** *variable-name string*
5. ステップ 4 を繰り返して、ステップ 6 で登録されるポリシーに必要なすべての環境変数を設定します。
6. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager environment** [**all**| *variable-name*] **Example:** `Device# show event manager environment all`	（任意）EEM 環境変数の名前と値を表示します。 • オプションの **all** キーワードは、すべての EEM 環境変数を表示します。 • オプションの *variable-name* 引数は、指定された環境変数に関する情報を表示します。
ステップ 3	**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 4	**event manager environment** *variable-name string* **Example:** `Device(config)# event manager environment _cron_entry 0-59/2 0-23/1 * * 0-6`	指定された EEM 環境変数の値を設定します。 • この例では、ソフトウェアによって、CRON タイマー環境変数が、毎日、毎時の 2 分目に設定されます。
ステップ 5	ステップ 4 を繰り返して、ステップ 6 で登録されるポリシーに必要なすべての環境変数を設定します。	--

	Command or Action	Purpose
ステップ 6	**event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**] **Example:** `Device(config)# event manager policy tm_cli_cmd.tcl type system`	ポリシー内で定義された指定イベントが発生した場合に、EEMポリシーを実行するよう、定義します。 • **system** キーワードを使用して、シスコ定義のシステムポリシーを登録します。 • **user** キーワードを使用して、ユーザー定義のシステムポリシーを登録します。 • **trap** キーワードを使用して、ポリシーがトリガーされた場合の SNMP トラップを生成します。 • この例では、tm_cli_cmd.tcl という名前の EEM サンプル ポリシーが、システム ポリシーとして定義されます。
ステップ 7	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

例

次に、**show event manager environment** 特権 EXEC コマンドを使用して、すべての EEM 環境変数の名前と値を表示する例を示します。

```
Device# show event manager environment all
No.  Name                           Value
1    _cron_entry                    0-59/2 0-23/1 * * 0-6
2    _show_cmd                      show ver
3    _syslog_pattern                .*UPDOWN.*Ethernet1/0.*
4    _config_cmd1                   interface Ethernet1/0
5    _config_cmd2                   no shut
```

登録済みの EEM ポリシーの表示

登録済みの EEM ポリシーを表示するには、次の任意の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]

DETAILED STEPS

ステップ 1 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）。

Example:

```
Device> enable
```

ステップ 2 show event manager policy registered [event-type *event-name*] [time-ordered| name-ordered] [detailed *policy-filename*]

このコマンドを **time-ordered** キーワードとともに使用して、現在登録されているポリシーの情報を時間でソートして表示します。次に例を示します。

Example:

```
Device# show event manager policy registered time-ordered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Wed May11  01:43:18 2005  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240
2    system  syslog              Off   Wed May11  01:43:28 2005  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90
3    system  proc abort          Off   Wed May11  01:43:38 2005  pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20
```

このコマンドを **name-ordered** キーワードとともに使用して、現在登録されているポリシーの情報を名前でソートして表示します。次に例を示します。

Example:

```
Device# show event manager policy registered name-ordered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  proc abort          Off   Wed May11  01:43:38 2005  pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20
2    system  syslog              Off   Wed May11  01:43:28 2005  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90
3    system  timer cron          Off   Wed May11  01:43:18 2005  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240
```

このコマンドを **event-type** キーワードとともに使用して、*event-name* 引数で指定されたイベントタイプの現在登録されているポリシーに関する情報を表示します。次に例を示します。

Example:

```
Device# show event manager policy registered event-type syslog
No.  Type    Event Type          Time Registered           Name
1    system  syslog              Wed May11  01:43:28 2005  sl_intf_down.tcl
```

```
occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
nice 0 priority normal maxrun 90
```

EEM ポリシーの登録解除

EEM ポリシーを実行コンフィギュレーション ファイルから削除するには、次の作業を実行します。ポリシーの実行はキャンセルされます。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]
3. **configure terminal**
4. **no event manager policy** *policy-filename*
5. **exit**
6. ステップ 2 を繰り返して、ポリシーが削除されたことを確認します。

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*] **Example:** `Device# show event manager policy registered`	（任意）現在登録されている EEM ポリシーを表示します。 • オプションの **system** キーワードまたは **user** キーワードによって、登録済みのシステム ポリシーまたはユーザー ポリシーが表示されます。 • キーワードが指定されない場合は、すべてのイベント タイプに対する登録された EEM ポリシーが時間順に表示されます。
ステップ 3	**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 4	**no event manager policy** *policy-filename* **Example:**	ポリシーを登録解除するために EEM ポリシーを設定から削除します。

	Command or Action	Purpose
	`Device(config)# no event manager policy pr_cdp_abort.tcl`	• この例では、コマンドの **no** 形式を使用して、指定されたポリシーの登録が解除します。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。
ステップ 6	ステップ 2 を繰り返して、ポリシーが削除されたことを確認します。 **Example:** `Device# show event manager policy registered`	--

例

次に、**show event manager policy registered** 特権 EXEC コマンドを使用して、現在登録されている 3 個の EEM ポリシーを表示する例を示します。

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Tue Oct11  01:43:18 2005 tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Tue Oct11  01:43:28 2005 sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
3    system  proc abort          Off   Tue Oct11  01:43:38 2005 pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20.000
```

現在のポリシーが表示されたら、**no** 形式の **event manager policy** コマンドを使用して pr_cdp_abort.tcl ポリシーの削除が決定されます。

```
Device# configure terminal
Device(config)# no event manager policy pr_cdp_abort.tcl
Device(config)# exit
```

show event manager policy registered 特権 EXEC コマンドを再度入力すると、現在登録されている EEM ポリシーが表示されます。ポリシー pr_cdp_abort.tcl は、登録されていません。

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Tue Oct11  01:45:17 2005 tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Tue Oct11  01:45:27 2005 sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
```

EEM ポリシー実行の一時停止

すべての EEM ポリシーの実行をただちに一時停止するには、次の作業を実行します。一時的なパフォーマンスまたはセキュリティ面での理由から、ポリシーの登録解除ではなく一時停止が必要なことがあります。

SUMMARY STEPS

1. **enable**
2. **show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*]
3. **configure terminal**
4. **event manager scheduler suspend**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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	**show event manager policy registered** [**event-type** *event-name*][**system**| **user**] [**time-ordered**| **name-ordered**] [**detailed** *policy-filename*] **Example:** `Device# show event manager policy registered`	（任意）現在登録されている EEM ポリシーを表示します。 • オプションの **system** キーワードまたは **user** キーワードによって、登録済みのシステム ポリシーまたはユーザー ポリシーが表示されます。 • キーワードが指定されない場合は、すべてのイベント タイプに対する登録された EEM ポリシーが時間順に表示されます。
ステップ 3	**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 4	**event manager scheduler suspend** **Example:** `Device(config)# event manager scheduler suspend`	すべての EEM ポリシーの実行がすぐに一時停止されます。
ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

例

次に、**show event manager policy registered** 特権 EXEC コマンドを使用して、EEM のすべての登録済みポリシーを表示する例を示します。

```
Device# show event manager policy registered
No.  Type    Event Type          Trap  Time Registered           Name
1    system  timer cron          Off   Sat Oct11  01:43:18 2003  tm_cli_cmd.tcl
 name {crontimer2} cron entry {0-59/1 0-23/1 * * 0-7}
 nice 0 priority normal maxrun 240.000
2    system  syslog              Off   Sat Oct11  01:43:28 2003  sl_intf_down.tcl
 occurs 1 pattern {.*UPDOWN.*Ethernet1/0.*}
 nice 0 priority normal maxrun 90.000
3    system  proc abort          Off   Sat Oct11  01:43:38 2003  pr_cdp_abort.tcl
 instance 1 path {cdp2.iosproc}
 nice 0 priority normal maxrun 20.000
```

すべての EEM ポリシーの実行をすぐに一時停止するには、**event manager scheduler suspend** コマンドを入力します。

```
Device# configure terminal
Device(config)# event manager scheduler suspend
*Nov  2 15:34:39.000: %HA_EM-6-FMS_POLICY_EXEC: fh_io_msg: Policy execution has been
suspended
```

EEM ポリシーの管理

ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定するには、この作業を実行します。

Note この作業は、Tcl スクリプトを使用して記述される EEM ポリシーのみに適用されます。

SUMMARY STEPS

1. **enable**
2. **show event manager directory user** [**library**| **policy**]
3. **configure terminal**
4. **event manager directory user** {**library** *path*| **policy** *path*}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
ステップ 2	**show event manager directory user** [**library**| **policy**] **Example:** `Device# show event manager directory user library`	（任意）EEM ユーザー ライブラリまたはポリシー ファイルの保存に使用するディレクトリを表示します。 • オプションの **library** キーワードによって、ユーザーライブラリファイルに使用されるディレクトリが表示されます。 • オプションの **policy** キーワードによって、ユーザー定義 EEM ポリシーに使用されるディレクトリが表示されます。
ステップ 3	**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 4	**event manager directory user** {**library** *path*| **policy** *path*}
	Example:
	`Device(config)# event manager directory user library disk0:/user_library`
	`Device(config)# event manager directory user library bootflash:/user_library`
		ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定します。
		• ユーザーディレクトリへの絶対パス名を指定するには、*path* 引数を指定します。

ステップ 5	**exit** **Example:** `Device(config)# exit`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

例

次に、**show event manager directory user** 特権 EXEC コマンドを使用して、EEM ユーザーライブラリファイルの保存に使用されるディレクトリがある場合に、そのディレクトリを表示する例を示します。

```
Device# show event manager directory user library
disk0:/user_library

Device# show event manager directory user library
bootflash:/user_library
```

履歴テーブル サイズの変更と EEM 履歴データの表示

履歴テーブルのサイズを変更し、EEM 履歴データを表示するには、次の任意の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager history size** {**events** | **traps**} [*size*]
4. **exit**
5. **show event manager history events** [**detailed**] [**maximum** *number*]
6. **show event manager history traps** [**server** | **policy]**

DETAILED STEPS

ステップ 1 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）。

Example:

```
Device> enable
```

ステップ 2 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。

Example:

```
Device# configure terminal
```

ステップ 3 event manager history size {events | traps} [size]

このコマンドを使用して、EEM イベント履歴テーブルのサイズ、または、EEM SNMP トラップ履歴テーブルのサイズを変更します。次に、EEM イベント履歴テーブルのサイズを 30 エントリに変更する例を示します。

Example:

```
Device(config)# event manager history size events 30
```

ステップ 4 exit

グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

Example:

```
Device(config)# exit
```

ステップ 5 show event manager history events [detailed] [maximum number]

このコマンドを使用して、トリガーされた各 EEM イベントについての情報を表示します。

Example:

```
Device# show event manager history events
No.  Time of Event             Event Type          Name
1    Fri Sep  9 13:48:40 2005  syslog              applet: one
2    Fri Sep  9 13:48:40 2005  syslog              applet: two
3    Fri Sep  9 13:48:40 2005  syslog              applet: three
4    Fri Sep  9 13:50:00 2005  timer cron          script: tm_cli_cmd.tcl
5    Fri Sep  9 13:51:00 2005  timer cron          script: tm_cli_cmd.tcl
```

ステップ 6 **show event manager history traps** [**server** | **policy]**

このコマンドを使用して、EEM サーバーまたは EEM ポリシーのいずれかから送信された EEM SNMP トラップを表示します。

Example:

```
Device# show event manager history traps
No.  Time                      Trap Type           Name
1    Fri Sep  9 13:48:40 2005  server              applet: four
2    Fri Sep  9 13:57:03 2005  policy              script: no_snmp_test.tcl
```

EEM を使用したソフトウェア モジュール方式プロセスの信頼性メトリック

Cisco IOS ソフトウェアモジュール方式プロセスの信頼性メトリックを表示するには、この任意の作業を実行します。この **show event manager metric processes** コマンド拡張は、ソフトウェアモジュール方式イメージでのみサポートされます。

SUMMARY STEPS

1. **enable**
2. **show event manager metric process** {**all**| *process-name*}

DETAILED STEPS

ステップ 1 **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）。

Example:

```
Device> enable
```

ステップ 2 **show event manager metric process** {**all**| *process-name*}

このコマンドを使用して、プロセスの信頼性メトリック データを表示します。システムでは、プロセスの開始時と終了時にレコードが保存され、このデータが、信頼性分析の基本データとして使用されます。この部分の例では、システムに挿入されているすべてのカード上でのプロセスのメトリック データを示す、最初と最後のエントリが表示されます。

Example:

```
Device# show event manager metric process all
====================================
process name: devc-pty, instance: 1
sub_system id: 0, version: 00.00.0000
--------------------------------
last event type: process start
recent start time: Fri Oct10  20:34:40 2005
recent normal end time: n/a
recent abnormal end time: n/a
number of times started: 1
number of times ended normally: 0
number of times ended abnormally: 0
most recent 10 process start times:
--------------------------
Fri Oct10  20:34:40 2005
--------------------------
most recent 10 process end times and types:
cumulative process available time: 6 hours 30 minutes 7 seconds 378 milliseconds
cumulative process unavailable time: 0 hours 0 minutes 0 seconds 0 milliseconds
process availability:  0.100000000
number of abnormal ends within the past 60 minutes (since reload): 0
number of abnormal ends within the past 24 hours (since reload): 0
number of abnormal ends within the past 30 days (since reload): 0
.
.
.
====================================
process name: cdp2.iosproc, instance: 1
sub_system id: 0, version: 00.00.0000
--------------------------------
last event type: process start
recent start time: Fri Oct10  20:35:02 2005
recent normal end time: n/a
recent abnormal end time: n/a
number of times started: 1
number of times ended normally: 0
number of times ended abnormally: 0
most recent 10 process start times:
--------------------------
Fri Oct10  20:35:02 2005
--------------------------
most recent 10 process end times and types:

cumulative process available time: 6 hours 29 minutes 45 seconds 506 milliseconds
cumulative process unavailable time: 0 hours 0 minutes 0 seconds 0 milliseconds
process availability:  0.100000000
number of abnormal ends within the past 60 minutes (since reload): 0
number of abnormal ends within the past 24 hours (since reload): 0
number of abnormal ends within the past 30 days (since reload): 0
```

トラブルシューティングのヒント

特権 EXEC モードで **debug event manager** コマンドを使用して、EEM コマンド操作のトラブルシューティングを行います。デバッグ コマンドは注意して使用してください。生成される出力量によってデバイスの動作が遅くなったり、停止したりすることがあります。シスコエンジニアの管理下に限ってこのコマンドを使用することを推奨します。

EEM サンプル ポリシーの変更

サンプル ポリシーの 1 つを変更するには、この作業を実行します。Cisco ソフトウェアには、Embedded Event Manager が含まれるイメージに、いくつかのサンプル ポリシーが含まれています。EEM ポリシーの開発者は、ポリシーが実行されるイベントと、イベントの記録および応答に関連付けられているオプションを、カスタマイズすることによって、これらのポリシーを変更できます。さらに、開発者は、ポリシーの実行時に実装されるアクションを選択できます。

EEM サンプル ポリシー

シスコには、次の表に示されているように、サンプル ポリシーのセットが含まれています。ユーザーは、サンプルポリシーをユーザーディレクトリにコピーし、ポリシーを変更するか、または、独自にポリシーを記述することができます。現時点でポリシー作成のためにシスコでサポートされているスクリプト言語は、Tcl だけです。Tcl ポリシーは、Emacs などのテキストエディタを使用して変更できます。ポリシーは、定義されている経過時間の秒数以内で実行する必要があり、時間変数はポリシー内で設定できます。現在のデフォルト値は 20 秒です。

次の表で、サンプル EEM ポリシーについて説明します。

***Table 61: EEM* サンプル ポリシーの説明**

ポリシーの名前	説明
pr_cdp_abort.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。このポリシーでは、cdp2.iosproc プロセスの終了イベントがモニターされます。SYSLOG にメッセージが記録され、終了の詳細が E メールで送信されます。
pr_crash_reporter.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。このポリシーでは、すべてのプロセスの終了イベントがモニターされます。イベントが発生すると、ポリシーによって、クラッシュダンプ ファイルを含むクラッシュ情報が、CGI スクリプトによってデータが処理される指定された URL に、送信されます。
pr_iprouting_abort.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。このポリシーでは、iprouting.iosproc プロセスの終了イベントがモニターされます。SYSLOG にメッセージが記録され、終了の詳細が E メールで送信されます。
sl_intf_down.tcl	このポリシーは、設定可能な Syslog メッセージが記録されるときに実行されます。設定可能な CLI コマンドが実行され、結果が E メールで送信されます。
tm_cli_cmd.tcl	このポリシーは、設定可能な CRON エントリを使用して実行されます。設定可能な CLI コマンドが実行され、結果が E メールで送信されます。

ポリシーの名前	説明
tm_crash_history.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。このポリシーは、毎日夜中に実行され、指定された E メール アドレスにプロセス クラッシュ履歴レポートが E メールで送信されます。
tm_crash_reporter.tcl	このポリシーは、登録後 5 秒間実行されます。ポリシーが設定に保存される場合、デバイスがリロードされるたびにも実行されます。ポリシーによって、リロード理由を示すプロンプトが表示されます。クラッシュが原因でリロードされる場合、ポリシーによって最新の crashinfo ファイルが検索され、この情報が指定された URL に送信されます。
tm_fsys_usage.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。このポリシーは、設定可能な CRON エントリを使用して実行され、ディスク領域の使用状況がモニターされます。ディスク領域の使用状況が、設定可能なしきい値を超えると、Syslog メッセージが表示されます。
wd_mem_reporter.tcl	Cisco ソフトウェア モジュラリティ イメージを使用して導入されました。使用可能なメモリ容量が、使用可能な初期システム メモリの 20% を下回った場合、このポリシーによって、システム メモリ低下の状態がレポートされます。Syslog メッセージが表示され、オプションで、E メールが送信されます。

SUMMARY STEPS

1. **enable**
2. **show event manager policy available detailed** *policy-filename*
3. 画面に表示されたサンプル ポリシーの内容を、テキスト エディタにカット アンド ペーストします。
4. ポリシーを編集し、新しいファイル名で保存します。
5. 新しいファイルを、デバイスのフラッシュ メモリにコピーして戻します。
6. **configure terminal**
7. **event manager directory user** {**library** *path*| **policy** *path*}
8. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

DETAILED STEPS

ステップ 1 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）。

Example:

```
Device> enable
```

ステップ 2 show event manager policy available detailed *policy-filename* detailed

ポリシーによって使用される環境変数と、ポリシーの実行方法の説明の詳細を含む、指定された実際のサンプル ポリシーを表示します。**show event manager policy available** および **show event manager policy registered** コマンドに対してキーワードが導入されました。お使いのリリースによっては、2 つの Tcl スクリプトのいずれかをこのドキュメントの設定例セクションからコピーしなければならない場合があります。次に、サンプル ポリシー tm_cli_cmd.tcl についての詳細が画面上に表示される例を示します。

Example:

```
Device# show event manager policy available detailed tm_cli_cmd.tcl
```

ステップ 3 画面に表示されたサンプル ポリシーの内容を、テキスト エディタにカット アンド ペーストします。

編集機能とコピー機能を使用して、デバイスから別のデバイス上のテキスト エディタに、内容を移動します。

ステップ 4 ポリシーを編集し、新しいファイル名で保存します。

テキスト エディタを使用して、ポリシーを Tcl スクリプトとして変更します。ファイルの命名規則については、EEM 用のシスコ ファイル命名規則, on page 670を参照してください。

ステップ 5 新しいファイルを、デバイスのフラッシュ メモリにコピーして戻します。

デバイスのフラッシュ ファイル システム（通常は disk0:）にファイルをコピーします。ファイルのコピーの詳細については、『*Configuration Fundamentals Configuration Guide*』の「Using the Cisco IOS File System」の章を参照してください。

デバイスのフラッシュ ファイル システム（通常は bootflash:）にファイルをコピーします。ファイルのコピーの詳細については、『*Configuration Fundamentals Configuration Guide*』の「Using the Cisco IOS File System」の章を参照してください。

ステップ 6 **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。

Example:

```
Device# configure terminal
```

ステップ 7 **event manager directory user** {**library** *path*| **policy** *path*}

ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定します。次に、disk0 の user_library ディレクトリが、ユーザー ライブラリ ファイルを保存するディレクトリとして指定されます。

ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定します。次に、bootflash の user_library ディレクトリが、ユーザー ライブラリ ファイルを保存するディレクトリとして指定されます。

Example:

```
Device(config)# event manager directory user library disk0:/user_library

Device(config)# event manager directory user library bootflash:/user_library
```

ステップ 8 **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

ポリシー内で定義された指定イベントが発生した場合に、EEM ポリシーを実行するよう、定義します。次に、test.tcl という名前の EEM ポリシーが、ユーザー定義ポリシーとして登録される例を示します。

Example:

```
Device(config)# event manager policy test.tcl type user
```

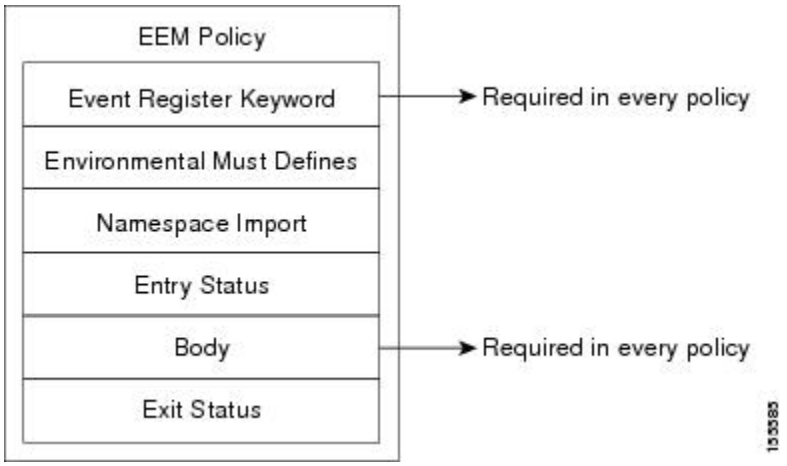
Tcl を使用した EEM ポリシーのプログラミング

Tcl コマンド拡張を使用してポリシーをプログラムするには、この作業を実行します。既存のポリシーをコピーし、変更することを推奨します。EEM Tcl ポリシーには、**event_register** Tcl コマンド拡張と本体の 2 つの必須部分が存在する必要があります。Tcl ポリシーの構造と要件, on page 686 の概念にある他のすべてのセクションは、オプションです。

Tcl ポリシーの構造と要件

すべての EEM ポリシーでは、次の図に示されているように、同じ構造が共有されます。EEM ポリシーには、**event_register** Tcl コマンド拡張と本体という 2 つの必須部分が存在します。ポリシーの残りの部分の、環境定義必須、名前空間のインポート、開始ステータス、および終了ステータスは、オプションです。

***Figure 17: Tcl* ポリシーの構造と要件**



各ポリシーの開始部分では、**event_register** Tcl コマンド拡張を使用して検出するイベントを記述し登録する必要があります。ポリシーのこの部分によって、ポリシーの実行がスケジュールされます。次に、**event_register_timer** Tcl コマンド拡張を登録する Tcl コードの例を示します。

```
::cisco::eem::event_register_timer cron name crontimer2 cron_entry $_cron_entry maxrun
240
```

環境定義必須セクションはオプションで、環境変数の定義が含まれます。次に、一部の環境変数をチェックし、定義する Tcl コードの例を示します。

```
# Check if all the env variables that we need exist.
# If any of them does not exist, print out an error msg and quit.
if {![info exists _email_server]} {
    set result \
      "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
      "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
      "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
```

名前空間のインポートセクションはオプションで、コードライブラリが定義されます。次に、名前空間インポート セクションを設定する Tcl コードの例を示します。

```
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
```

ポリシーの本体は必須の構造で、次のものを含める必要があります。

- 検出されたイベントに関する情報の EEM への問い合わせに使用される **event_reqinfo** イベント情報の Tcl コマンド拡張。
- EEM 特有のアクションの指定に使用される、**action_syslog** などのアクション Tcl コマンド拡張。
- 一般的なシステム情報の取得に使用される、**sys_reqinfo_routername** などのシステム情報の Tcl コマンド拡張。
- ポリシーからの、SMTP ライブラリ（電子メール通知を送信）または CLI ライブラリ（CLI コマンドを実行）の使用。
- 他のポリシーによって使用される Tcl 変数の保存に使用される **context_save** および **context_retrieve** の Tcl コマンド拡張。

次に、イベントを問い合わせ、本体セクションの一部としてメッセージを記録するコードの Tcl コードの例を示します。

```
# Query the event info and log a message.
array set arr_einfo [event_reqinfo]

if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)

# Log a message.
```

```
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]

action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

EEM 開始ステータス

EEM ポリシーの開始ステータスの部分は、前のポリシーが同じイベントに対して実行されたかどうかや、前のポリシーの終了ステータスを特定するために、使用されます。_entry_status 変数が定義されている場合、このイベントに対して前のポリシーがすでに実行されています。_entry_status 変数の値によって、前のポリシーの戻りコードが特定されます。

開始ステータス指定には、0（前のポリシーが正常終了した）、Not=0（前のポリシーに障害が発生した）、およびUndefined（実行された前のポリシーがない）の、3つの値のうちいずれか1つを使用できます。

EEM 終了ステータス

ポリシーでそのコードの実行を終了すると、終了値が設定されます。終了値は、Embedded Event Managerによって使用され、このイベントのデフォルトアクションがある場合に、それが適用されたかどうかが判断されます。ゼロの値は、デフォルトアクションが実行されていないことを意味します。ゼロではない値は、デフォルトアクションが実行されたことを意味します。終了ステータスは、同じイベントで実行される後続ポリシーに渡されます。

EEM ポリシーと Cisco エラー番号

一部の EEM Tcl コマンド拡張によって、Cisco エラー番号の Tcl グローバル変数の _cerrno が設定されます。_cerrno が設定されるたびに、他の 4 つの Tcl グローバル変数が _cerrno から分岐し、それとともに設定されます（_cerr_sub_num、_cerr_sub_err、_cerr_posix_err、および _cerr_str）。

たとえば、次の例の **action_syslog** コマンドでは、コマンド実行の副次的な影響としてこれらのグローバル変数が設定されます。

```
action_syslog priority warning msg “A sample message generated by action_syslog”
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

_cerrno：32 ビット エラー戻り値

コマンドによって設定された _cerrno は、次の形式の 32 ビットの整数を表す場合があります。

```
XYSSSSSSSSSSSSSEEEEEEEEPPPPPPPPP
```

たとえば、次のエラー戻り値は、EEM Tcl コマンド拡張から戻される場合があります。

```
862439AE
```

この数字は、次の 32 ビット値として解釈されます。

```
10000110001001000011100110101110
```

この 32 ビットの整数は、次の表に示されているように、5 つの変数に分けられます。

***Table 62: _cerrno*：*32* ビット エラー戻り値の変数**

変数	説明
XY	エラー クラス（エラーの重大度を示します）。この変数は、32 ビットのエラー戻り値の最初の 2 ビットに対応しています。前述のケースの 10 は、CERR_CLASS_WARNING を示します。 この変数固有の 4 つのエラー クラス エンコーディングについては、次の表を参照してください。
SSSSSSSSSSSSS	最新のエラーが生成されたサブシステム番号（13 ビット = 値 8192）。これは、32 ビット シーケンスの次の 13 ビットで、その整数値は $_cerr_sub_num に含まれています。
変数	説明
EEEEEEEE	サブシステム固有のエラー番号（8 ビット = 値 256）。このセグメントは、32 ビット シーケンスの次の 8 ビットで、このエラー番号に対応する文字列は、$_cerr_sub_err に含まれています。
PPPPPPPP	パススルー POSIX エラー コード（9 ビット = 値 512）。これは、32 ビット シーケンスの最後で、このエラー コードに対応する文字列は、$_cerr_posix_err に含まれています。

XY のエラー クラス エンコーディング

最初の変数 XY は、次の表に示されているように、エラー クラス エンコーディングを参照しています。

***Table 63:* エラー クラス エンコーディング**

00	CERR_CLASS_SUCCESS
01	CERR_CLASS_INFO
10	CERR_CLASS_WARNING
11	CERR_CLASS_FATAL

ゼロのエラー戻り値は、SUCCESS を示します。

SUMMARY STEPS

1. **enable**
2. **show event manager policy available detailed** *policy-filename*
3. 画面に表示されたサンプル ポリシーの内容を、テキスト エディタにカット アンド ペーストします。
4. 必要な **event_register** Tcl コマンド拡張を定義します。
5. 適切な名前空間を、::cisco 階層構造に追加します。
6. Must Define セクションをプログラムし、このポリシーで使用される各環境変数をチェックします。
7. スクリプトの本体をプログラムします。
8. 開始ステータスをチェックし、ポリシーがこのイベントに対して前に実行されたかどうかを判断します。
9. 終了ステータスをチェックし、デフォルト アクションが存在する場合に、このイベントのデフォルト アクションが適用されたかどうかを判断します。
10. Cisco エラー番号（_cerrno）の Tcl グローバル変数を設定します。
11. 新しいファイル名で Tcl スクリプトを保存し、Tcl スクリプトをデバイスにコピーします。
12. **configure terminal**
13. **event manager directory user** {**library** *path*| **policy** *path*}
14. **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]
15. ポリシーを実行し、ポリシーを観察します。
16. ポリシーが正しく実行されていない場合、デバッグのテクニックを使用します。

DETAILED STEPS

ステップ 1 **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）。

Example:

```
Device> enable
```

ステップ 2 **show event manager policy available detailed** *policy-filename*

ポリシーによって使用される環境変数と、ポリシーの実行方法の説明の詳細を含む、指定された実際のサンプル ポリシーを表示します。**detailed** キーワードが **show event manager policy available** コマンドと **show event manager policy registered** コマンドに導入されました。リリースに応じて、このドキュメントの設定例セクションから 2 つの Tcl スクリプトのいずれかをコピーする必要があります。次に、サンプル ポリシー tm_cli_cmd.tcl についての詳細が画面上に表示される例を示します。

Example:

```
Device# show event manager policy available detailed tm_cli_cmd.tcl
```

ステップ 3 画面に表示されたサンプル ポリシーの内容を、テキスト エディタにカット アンド ペーストします。

編集機能とコピー機能を使用して、デバイスから別のデバイス上のテキスト エディタに、内容を移動します。テキスト エディタを使用して、ポリシーを Tcl スクリプトとして編集します。

ステップ 4 必要な **event_register** Tcl コマンド拡張を定義します。

検出するイベントについて、適切な **event_register** Tcl コマンド拡張を次の表から選択し、ポリシーに追加します。

***Table 64: EEM* イベント登録の *Tcl* コマンド拡張**

イベント登録の Tcl コマンド拡張
event_register_appl
event_register_cli
event_register_counter
event_register_gold
event_register_interface
event_register_ioswdsysmon
event_register_ipsla
event_register_nf
event_register_none
event_register_oir
event_register_process
event_register_resource
event_register_rf
event_register_routing
event_register_rpc
event_register_snmp
event_register_snmp_notification
event_register_snmp_object
event_register_syslog
event_register_timer
event_register_timer_subscriber
event_register_track

イベント登録の Tcl コマンド拡張
event_register_wdsysmon

ステップ 5 適切な名前空間を、::cisco 階層構造に追加します。

ポリシーの開発者は、Cisco IOS EEM によって使用されるすべての拡張をグループ化するため、Tcl ポリシーで新しい名前空間 ::cisco を使用できます。::cisco 階層構造の下には、2 つの名前空間があります。次の表に、各名前空間の下に属する EEM Tcl コマンド拡張のカテゴリを示します。

***Table 65: Cisco IOS EEM* 名前空間グルーピング**

Namespace	Tcl コマンド拡張のカテゴリ
::cisco::eem	EEM イベント登録
	EEM イベント情報
	EEM イベント パブリッシュ
	EEM アクション
	EEM ユーティリティ
	EEM コンテキスト ライブラリ
	EEM システム情報
	CLI ライブラリ
::cisco::lib	SMTP ライブラリ

Note 前述のコマンドの使用時に、適切な名前空間をインポートするか、または、認定コマンド名を使用します。

ステップ 6 Must Define セクションをプログラムし、このポリシーで使用される各環境変数をチェックします。

この手順は任意です。Must Define は、ポリシーによって必要とされるすべての EEM 環境変数が、回復アクションの実行前に定義されているかどうかをテストする、ポリシーのセクションです。ポリシーによって EEM 環境変数が使用されない場合、Must Define セクションは不要です。EEM スクリプトの EEM 環境変数は、ポリシーの実行前にポリシーに対して外部定義された Tcl グローバル変数です。EEM 環境変数を定義するには、Embedded Event Manager コンフィギュレーション コマンド **event manager environment** CLI コマンドを使用します。規則として、すべてのシスコ EEM 環境変数の先頭は、「_」（アンダースコア）になっています。将来的な競合を避けるため、「_」で始まる新しい変数を定義しないことを推奨します。

Note **show event manager environment** 特権 EXEC コマンドを使用して、システムの Embedded Event Manager 環境変数セットを表示できます。

たとえば、サンプル ポリシーで定義されている Embedded Event Manager 環境変数には、E メール変数が含まれます。適切に動作させるためには、電子メールを送信するサンプル ポリシーに、次の表に示す変数が設定されている必要があります。

次の表で EEM サンプル ポリシーで使用される電子メール特有の環境変数について説明します。

Table 66: サンプル ポリシーで使用される電子メール特有の環境変数

環境変数	説明	例
_email_server	E メール送信に使用されるシンプル メール転送プロトコル（SMTP）メール サーバー。	E メール サーバー名は、次のテンプレート形式のいずれかで使用できます。 • username:password@host • username@host • ホスト
_email_to	E メールの送信先アドレス。	engineering@example.com
_email_from	E メールの送信元アドレス。	devtest@example.com
_email_cc	E メールのコピーの送信先アドレス。	manager@example.com

次に、E メール特有の環境変数のプログラムをチェックする Must Define セクションの例を示します。

Must Define の例

Example:

```
if {![info exists _email_server]} {
    set result \
        "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
        "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
        "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
}
if {![info exists _email_cc]} {
    set result \
        "Policy cannot be run: variable _email_cc has not been set"
    error $result $errorInfo
}
```

ステップ 7 スクリプトの本体をプログラムします。

スクリプトのこのセクションでは、次のいずれかを定義できます。

- 検出されたイベントに関する情報の EEM への問い合わせに使用される **event_reqinfo** イベント情報の Tcl コマンド拡張。
- EEM 特有のアクションの指定に使用される、**action_syslog** などのアクション Tcl コマンド拡張。
- 一般的なシステム情報の取得に使用される、**sys_reqinfo_routername** などのシステム情報の Tcl コマンド拡張。
- 他のポリシーによって使用される Tcl 変数の保存に使用される **context_save** および **context_retrieve** の Tcl コマンド拡張。
- ポリシーからの、SMTP ライブラリ（電子メール通知を送信）または CLI ライブラリ（CLI コマンドを実行）の使用。

ステップ 8 開始ステータスをチェックし、ポリシーがこのイベントに対して前に実行されたかどうかを判断します。

前のポリシーが正常終了した場合、現在のポリシーは、実行が必要な場合と、実行が不要な場合があります。開始ステータス指定には、0（前のポリシーが正常終了した）、Not=0（前のポリシーに障害が発生した）、および Undefined（実行された前のポリシーがない）の、3 つの値のうちいずれか 1 つを使用できます。

ステップ 9 終了ステータスをチェックし、デフォルトアクションが存在する場合に、このイベントのデフォルトアクションが適用されたかどうかを判断します。

ゼロの値は、デフォルト アクションが実行されていないことを意味します。ゼロではない値は、デフォルト アクションが実行されたことを意味します。終了ステータスは、同じイベントで実行される後続ポリシーに渡されます。

ステップ 10 Cisco エラー番号（_cerrno）の Tcl グローバル変数を設定します。

一部の EEM Tcl コマンド拡張によって、Cisco エラー番号の Tcl グローバル変数の _cerrno が設定されます。_cerrno が設定されるたびに、他の 4 つの Tcl グローバル変数が _cerrno から分岐し、それとともに設定されます（_cerr_sub_num、_cerr_sub_err、_cerr_posix_err、および _cerr_str）。

たとえば、次の例の **action_syslog** コマンドでは、コマンド実行の副次的な影響としてこれらのグローバル変数が設定されます。

Example:

```
action_syslog priority warning msg "A sample message generated by action_syslog
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

ステップ 11 新しいファイル名で Tcl スクリプトを保存し、Tcl スクリプトをデバイスにコピーします。

Embedded Event Manager ポリシー ファイル名は、次の仕様に従っています。

- オプションのプレフィックス Mandatory. がある場合、これは、システム ポリシーがまだ登録されていない場合に、自動的に登録される必要があるシステムポリシーであることを示します。たとえば、Mandatory.sl_text.tcl などです。

- 指定された 1 つめのイベントの 2 文字の省略形が含まれるファイル名の本体部、下線文字部、および、ポリシーをさらに示す説明フィールド部。
- ファイル名拡張子部は .tcl と定義されます。

詳細については、「*EEM* 用のシスコファイル命名規則」を参照してください。

デバイスのフラッシュ ファイル システム（通常は disk0:）にファイルをコピーします。ファイルのコピーの詳細については、『Cisco IOS Configuration Fundamentals Configuration Guide』の「Using the Cisco IOS File System」の章を参照してください。

デバイスのフラッシュ ファイル システム（通常は bootflash:）にファイルをコピーします。ファイルのコピーの詳細については、『Cisco IOS Configuration Fundamentals Configuration Guide』の「Using the Cisco IOS File System」の章を参照してください。

ステップ 12 **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。

Example:

```
Device# configure terminal
```

ステップ 13 **event manager directory user** {**library** *path*| **policy** *path*}

ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定します。次に、disk0 の user_library ディレクトリが、ユーザー ライブラリ ファイルを保存するディレクトリとして指定されます。

ユーザー ライブラリ ファイルまたはユーザー定義 EEM ポリシーの保存に使用するディレクトリを指定します。次に、bootflash の user_library ディレクトリが、ユーザー ライブラリ ファイルを保存するディレクトリとして指定されます。

Example:

```
Device(config)# event manager directory user library disk0:/user_library

Device(config)# event manager directory user library bootflash:/user_library
```

ステップ 14 **event manager policy** *policy-filename* [**type** {**system**| **user**}] [**trap**]

ポリシー内で定義された指定イベントが発生した場合に、EEM ポリシーを実行するよう、定義します。次に、cl_mytest.tcl という名前の EEM ポリシーが、ユーザー定義ポリシーとして登録される例を示します。

Example:

```
Device(config)# event manager policy cl_mytest.tcl type user
```

ステップ 15 ポリシーを実行し、ポリシーを観察します。

ポリシーの実行をテストするには、ポリシーが実行される原因となる条件を生成し、ポリシーが想定どおりに実行されていることを確認します。

ステップ 16 ポリシーが正しく実行されていない場合、デバッグのテクニックを使用します。

Cisco IOS **debug event manager** CLI コマンドをそのさまざまなキーワードとともに使用して、問題をデバッグします。Tcl 特有のキーワード使用の詳細については、*Troubleshooting Tips* セクションを参照してください。

トラブルシューティングのヒント

- Tcl 拡張コマンドの問題をデバッグするには、**debug event manager tcl commands** コマンドを使用します。イネーブルの場合、このコマンドによって、CLI のやり取りを処理する TTY セッションに渡され、TTY セッションから読み戻される、すべてのデータが表示されます。このデータを使用すると、ユーザーが CLI に渡すコマンドが有効になります。
- CLI ライブラリを使用すると、ユーザーは、CLI コマンドを実行し、Tcl のコマンドの出力を取得できます。**debug event manager tcl cli-library** CLI コマンドを使用して、CLI ライブラリの問題をデバッグします。
- SMTP ライブラリを使用すると、ユーザーは、SMTP E メール サーバーへ、E メール メッセージを送信できます。**debug event manager tcl smtp_library** CLI コマンドを使用して、SMTP ライブラリの問題をデバッグします。イネーブルの場合、このコマンドによって、SMTP ライブラリ ルーチンに渡され、SMTP ライブラリ ルーチンから読み戻される、すべてのデータが表示されます。このデータを使用すると、ユーザーが SMTP ライブラリに渡すコマンドが有効になります。
- Tcl は、コマンドを上書きできる融通性のある言語です。たとえば、**set** コマンドを変更し、スカラ変数が設定されたときにメッセージを表示する **set** コマンドのバージョンを作成します。ポリシーに **set** コマンドが入力されると、スカラ変数が設定されたときにはいつでもメッセージが表示され、スカラ変数をデバッグする方法が示されます。このデバッグテクニックの例を参照するには、Tcl set コマンド操作のトレースの例, on page 726を参照してください。

これらのデバッグ テクニックのいくつかの例を参照するには、Embedded Event Manager ポリシーのデバッグの例, on page 724を参照してください。

EEM ユーザー Tcl ライブラリ索引の作成

Tcl ファイルのライブラリに含まれているすべての手順のディレクトリが含まれている、索引ファイルを作成するには、この作業を実行します。この作業を行うと、EEM Tcl のライブラリ サポートをテストできます。この作業では、Tcl ライブラリ ファイルが含まれるライブラリ ディレクトリが作成され、ファイルがディレクトリにコピーされ、ライブラリ ファイルにあるすべての手順のディレクトリが含まれる索引 tclIndex が作成されます。索引が作成されない場合、Tcl 手順を参照する EEM ポリシーを実行するときに、Tcl 手順は見つかりません。

SUMMARY STEPS

1. ワークステーション（UNIX、Linux、PC、またはMac）で、ライブラリディレクトリを作成し、Tcl ライブラリ ファイルをディレクトリにコピーします。
2. **tclsh**
3. **auto_mkindex** *directory_name* ***.tcl**
4. ターゲットデバイス上のユーザーライブラリファイルの保存に使用されるディレクトリに Tcl ライブラリファイルと tclIndex ファイルをコピーします。
5. Tcl で記述されたユーザー定義EEMポリシーファイルを、ターゲットデバイス上でユーザー定義 EEM ポリシーの保存に使用されるディレクトリにコピーします。
6. **enable**
7. **configure terminal**
8. **event manager directory user library** *path*
9. **event manager directory user policy** *path*
10. **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap]**
11. **event manager run** *policy-name*

DETAILED STEPS

ステップ 1 ワークステーション（UNIX、Linux、PC、または Mac）で、ライブラリ ディレクトリを作成し、Tcl ライブラリ ファイルをディレクトリにコピーします。

次の例ファイルを使用すると、Tcl シェルが実行されているワークステーション上で、tclIndex を作成できます。

lib1.tcl

Example:

```
proc test1 {} {
    puts "In procedure test1"
}

proc test2 {} {
    puts "In procedure test2"
}
```

lib2.tcl

Example:

```
proc test3 {} {
    puts "In procedure test3"
}
```

ステップ 2 **tclsh**

このコマンドを使用して、Tcl シェルを開始します。

Example:

```
workstation% tclsh
```

ステップ 3 **auto_mkindex** *directory_name* ***.tcl**

auto_mkindex コマンドを使用して、tclIndex ファイルを作成します。すべての手順のディレクトリが含まれる tclIndex ファイルは、Tcl ライブラリ ファイルに含まれていました。どのディレクトリにも 1 つの tclIndex ファイルのみを存在させることができ、他の Tcl ファイルはグループ化しておくことが可能であるため、ディレクトリ内で auto_mkindex を実行することを推奨します。ディレクトリ内で auto_mkindex を実行すると、特定の tclIndex を使用してどの Tcl ソース ファイルを索引化できるかが判断されます。

Example:

```
workstation% auto_mkindex eem_library *.tcl
```

lib1.tcl ファイルと lib2.tcl ファイルがライブラリ ファイル ディレクトリにあり、**auto_mkindex** コマンドが実行されたときに、次の例に示す tclIndex が作成されます。

tclIndex

Example:

```
# Tcl autoload index file, version 2.0
# This file is generated by the "auto_mkindex" command
# and sourced to set up indexing information for one or
# more commands.  Typically each line is a command that
# sets an element in the auto_index array, where the
# element name is the name of a command and the value is
# a script that loads the command.

set auto_index(test1) [list source [file join $dir lib1.tcl]]
set auto_index(test2) [list source [file join $dir lib1.tcl]]
set auto_index(test3) [list source [file join $dir lib2.tcl]]
```

ステップ 4 ターゲットデバイス上のユーザーライブラリファイルの保存に使用されるディレクトリに Tcl ライブラリファイルと tclIndex ファイルをコピーします。

ステップ 5 Tcl で記述されたユーザー定義 EEM ポリシー ファイルを、ターゲット デバイス上でユーザー定義 EEM ポリシーの保存に使用されるディレクトリにコピーします。

ユーザー定義 EEM ポリシーを保存するディレクトリは、ステップ 4 で使用されるディレクトリと同じディレクトリを使用できます。次に、EEM でサポートされる Tcl ライブラリのテストに、ユーザー定義 EEM ポリシーを使用できる例を示します。

libtest.tcl

Example:

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

global auto_index auto_path

puts [array names auto_index]

if { [catch {test1} result]} {
    puts "calling test1 failed result = $result $auto_path"
}

if { [catch {test2} result]} {
```

```
    puts "calling test2 failed result = $result $auto_path"
}
if { [catch {test3} result]} {
    puts "calling test3 failed result = $result $auto_path"
}
```

ステップ 6 **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）。

Example:

```
Device> enable
```

ステップ 7 **configure terminal**

グローバル コンフィギュレーション モードをイネーブルにします。

Example:

```
Device# configure terminal
```

ステップ 8 **event manager directory user library** *path*

このコマンドを使用して、EEM ユーザー ライブラリ ディレクトリを指定します。これは、ファイルがコピーされたディレクトリです。

Example:

```
Device(config)# event manager directory user library disk2:/eem_library
```

ステップ 9 **event manager directory user policy** *path*

このコマンドを使用して、EEM ユーザー ポリシー ディレクトリを指定します。これは、ファイルがコピーされたディレクトリです。

Example:

```
Device(config)# event manager directory user policy disk2:/eem_policies
```

ステップ 10 **event manager policy** *policy-name* **[type** {**system** | **user**} **[trap]**

このコマンドを使用して、ユーザー定義 EEM ポリシーを登録します。この例では、libtest.tcl という名前のポリシーが登録されます。

Example:

```
Device(config)# event manager policy libtest.tcl
```

ステップ 11 **event manager run** *policy-name*

このコマンドを使用して、手作業で EEM ポリシーを実行します。この例では、libtest.tcl という名前のポリシーが実行され、EEM の Tcl サポートがテストされます。次に、EEM の Tcl サポートが正常終了した出力例を示します。

Example:

```
Device(config)# event manager run libtest.tcl
```

```
The following output is displayed:
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test1
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test2
01:24:37: %HA_EM-6-LOG: libtest.tcl: In procedure test3
```

EEM ユーザー Tcl パッケージ索引の作成

すべての Tcl パッケージのディレクトリと、Tcl パッケージ ファイルのライブラリに含まれるバージョン情報が含まれる、Tcl パッケージの索引ファイルを作成するには、この作業を実行します。使用しているリリースによっては、Tcl **package** キーワードを使用することで Tcl パッケージがサポートされます。

Tcl パッケージは、EEM システム ライブラリ ディレクトリまたは EEM ユーザー ライブラリ ディレクトリのいずれかにあります。**package require** Tcl コマンドが実行されると、ユーザー ライブラリ ディレクトリで、まず、pkgIndex.tcl ファイルが検索されます。pkgIndex.tcl ファイルがユーザー ディレクトリで見つからない場合、システム ライブラリ ディレクトリが検索されます。この作業では、**pkg_mkIndex** コマンドを使用して、適切なライブラリディレクトリに Tcl パッケージディレクトリ（pkgIndex.tcl ファイル）が作成され、バージョン情報とともに、ディレクトリ内にあるすべての Tcl パッケージについての情報が含められます。索引が作成されない場合、**package require** Tcl コマンドが含まれる、EEM ポリシーが実行されたときに、Tcl パッケージは見つかりません。

EEM で Tcl パッケージ サポートを使用すると、ユーザーは、Tcl の XML_RPC などのパッケージにアクセスできます。Tcl パッケージ インデックスが作成されるとき、Tcl スクリプトは、外部エンティティに対する XML-RPC 呼び出しを容易に行うことができます。

Note C プログラミング コードで実装されるパッケージは、EEM ではサポートされません。

SUMMARY STEPS

1. ワークステーション（UNIX、Linux、PC、またはMac）で、ライブラリ ディレクトリを作成し、Tcl パッケージ ファイルをディレクトリにコピーします。
2. **tclsh**
3. **pkg_mkindex** *directory_name* ***.tcl**
4. ターゲット デバイス上のユーザー ライブラリ ファイルの保存に使用されるディレクトリに Tcl ライブラリ ファイルと pkgIndex ファイルをコピーします。
5. Tcl で記述されたユーザー定義EEMポリシー ファイルを、ターゲットデバイス上でユーザー定義 EEM ポリシーの保存に使用されるディレクトリにコピーします。ディレクトリは、使用されるディレクトリと同じにすることができます。
6. **enable**
7. **configure terminal**
8. **event manager directory user library** *path*
9. **event manager directory user policy** *path*
10. **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap**]

11. **event manager run** *policy-name*

DETAILED STEPS

ステップ 1 ワークステーション（UNIX、Linux、PC、またはMac）で、ライブラリ ディレクトリを作成し、Tcl パッケージ ファイルをディレクトリにコピーします。

ステップ 2 **tclsh**

このコマンドを使用して、Tcl シェルを開始します。

Example:

```
workstation% tclsh
```

ステップ 3 **pkg_mkindex** *directory_name* ***.tcl**

pkg_mkindex コマンドを使用して、pkgIndex ファイルを作成します。すべてのパッケージのディレクトリが含まれる pkgIndex ファイルは、Tcl ライブラリ ファイルに含まれていました。どのディレクトリにも 1 つの pkgIndex ファイルのみを存在させることができ、他の Tcl ファイルはグループ化しておくことが可能であるため、ディレクトリ内で pkg_mkindex を実行することを推奨します。ディレクトリ内で pkg_mkindex を実行すると、特定の pkgIndex を使用してどの Tcl パッケージ ファイルを索引化できるかが判断されます。

Example:

```
workstation% pkg_mkindex eem_library *.tcl
```

次に、いくつかの Tcl パッケージがライブラリ ファイル ディレクトリにあり、**pkg_mkindex** コマンドが実行されたときに、pkgIndex が作成される例を示します。

pkgIndex

Example:

```
# Tcl package index file, version 1.1
# This file is generated by the "pkg_mkIndex" command
# and sourced either when an application starts up or
# by a "package unknown" script. It invokes the
# "package ifneeded" command to set up package-related
# information so that packages will be loaded automatically
# in response to "package require" commands. When this
# script is sourced, the variable $dir must contain the
# full path name of this file's directory.
package ifneeded xmlrpc 0.3 [list source [file join $dir xmlrpc.tcl]]
```

ステップ 4 ターゲット デバイス上のユーザー ライブラリ ファイルの保存に使用されるディレクトリに Tcl ライブラリ ファイルと pkgIndex ファイルをコピーします。

ステップ 5 Tcl で記述されたユーザー定義 EEM ポリシー ファイルを、ターゲット デバイス上でユーザー定義 EEM ポリシーの保存に使用されるディレクトリにコピーします。ディレクトリは、使用されるディレクトリと同じにすることができます。

ユーザー定義EEMポリシーを保存するディレクトリは、ディレクトリと同じディレクトリを使用できます。次に、EEM でサポートされる Tcl パッケージのテストに、ユーザー定義 EEM ポリシーを使用できる例を示します。

packagetest.tcl

Example:

```
::cisco::eem::event_register_none maxrun 1000000.000
#
# test if xmlrpc available
#
#
# Namespace imports
#
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
#
package require xmlrpc
puts "Did you get an error?"
```

ステップ 6 **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）。

Example:

```
Device> enable
```

ステップ 7 **configure terminal**

グローバル コンフィギュレーション モードをイネーブルにします。

Example:

```
Device# configure terminal
```

ステップ 8 **event manager directory user library** *path*

このコマンドを使用して、EEM ユーザー ライブラリ ディレクトリを指定します。これは、ファイルがコピーされたディレクトリです。

Example:

```
Device(config)# event manager directory user library disk2:/eem_library
```

ステップ 9 **event manager directory user policy** *path*

このコマンドを使用して、EEM ユーザー ポリシー ディレクトリを指定します。これは、ファイルがコピーされたディレクトリです。

Example:

```
Device(config)# event manager directory user policy disk2:/eem_policies
```

ステップ 10 **event manager policy** *policy-name* [**type** {**system** | **user**} [**trap**]

このコマンドを使用して、ユーザー定義 EEM ポリシーを登録します。この例では、packagetest.tcl という名前のポリシーが登録されます。

Example:

```
Device(config)# event manager policy packagetest.tcl
```

ステップ 11 **event manager run** *policy-name*

このコマンドを使用して、手作業で EEM ポリシーを実行します。この例では、packagetest.tcl という名前のポリシーが実行され、EEM の Tcl パッケージ サポートがテストされます。

Example:

```
Device(config)# event manager run packagetest.tcl
```

Tcl を使用した Embedded Event Manager（EEM）ポリシー記述の設定例

Tcl セッションへのユーザー名割り当ての例

次に、Tcl セッションに関連付けられるユーザー名を設定する例を示します。認証、認可、カウンティング（AAA）セキュリティを使用し、コマンドベースで認可を実装する場合、**event manager session cli username** コマンドを使用して、Tcl セッションに関連付けられるユーザー名を設定する必要があります。Tcl ポリシーによって CLI コマンドが実行されるときに、ユーザー名が使用されます。TACACS+ では、ポリシーを実行している Tcl セッションに関連付けられているユーザー名を使用して、各 CLI コマンドが確認されます。ポリシーを登録するには、デバイスが特権 EXEC モードである必要があるため、Tcl ポリシーからのコマンドは、通常、確認されません。この例では、ユーザー名は yourname で、これは、CLI コマンドセッションが EEM ポリシー内から開始されるたびに使用されるユーザー名です。

```
configure terminal
 event manager session cli username yourname
 end
```

EEM イベント ディテクタのデモの例

EEM サンプル ポリシーの説明

この設定例では、一部の EEM サンプル ポリシーについて説明します。

- ap_perf_test_base_cpu.tcl：EEM ポリシーの CPU パフォーマンスを測定するために実行されます。

- no_perf_test_init.tcl：EEM ポリシーの CPU パフォーマンスを測定するために実行されます。
- sl_intf_down.tcl：設定可能な syslog メッセージが記録されるときに実行されます。最大で 2 つまでの CLI コマンドを実行し、結果が E メールで送信されます。
- tm_cli_cmd.tcl：設定可能な CRON エントリを使用して実行されます。設定可能な CLI コマンドが実行され、結果が電子メールで送信されます。
- tm_crash_reporter.tcl：登録後の 5 秒間と、デバイスの起動後の 5 秒間に実行されます。トリガーされると、スクリプトによって、リロード原因の検索が試行されます。リロードの原因がクラッシュの場合、ポリシーによって、関連する crashinfo ファイルが検索され、環境変数 _crash_reporter_url でユーザーによって指定された URL へ、この情報が送信されます。
- tm_fsys_usage.tcl：このポリシーは、設定可能な CRON エントリを使用して実行され、ディスク領域の使用状況を監視します。ディスク領域の使用状況が、設定可能なしきい値を超えると、Syslog メッセージが表示されます。

サンプル ポリシーのイベント マネージャ環境変数

イベント マネージャ環境変数は、ポリシーの登録および実行の前に EEM ポリシーに対して外部定義された Tcl グローバル変数です。サンプルポリシーでは、電子メール環境変数のうち 3 つを設定する必要があります（電子メール変数のリストについては、上記のセクションを参照してください）。_email_cc のみがオプションです。他の必須および任意の変数設定については、次の表で説明します。

次の表に、ap_perf_test_base_cpu.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 67: ap_perf_test_base_cpu.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_perf_iterations	測定を反復する回数。	**100**
_perf_cmd1	測定テストの一部として実行される最初の非インタラクティブ CLI コマンド。この変数は任意で、指定する必要はありません。	**enable**
_perf_cmd2	測定テストの一部として実行される 2 番目の非インタラクティブ CLI コマンド。_perf_cmd2 を使用するには、_perf_cmd1 を定義する必要があります。この変数は任意で、指定する必要はありません。	**show version**
_perf_cmd3	測定テストの一部として実行される 3 番目の非インタラクティブ CLI コマンド。_perf_cmd3 を使用するには、_perf_cmd1 を定義する必要があります。この変数は任意で、指定する必要はありません。	**show interface counters protocol status**

次の表に、no_perf_test_init.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 68: no_perf_test_init.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_perf_iterations	測定を反復する回数。	**100**
_perf_cmd1	測定テストの一部として実行される最初の非インタラクティブ CLI コマンド。この変数は任意で、指定する必要はありません。	**enable**
_perf_cmd2	測定テストの一部として実行される 2 番目の非インタラクティブ CLI コマンド。_perf_cmd2 を使用するには、_perf_cmd1 を定義する必要があります。この変数は任意で、指定する必要はありません。	**show version**
_perf_cmd3	測定テストの一部として実行される 3 番目の非インタラクティブ CLI コマンド。_perf_cmd3 を使用するには、_perf_cmd1 を定義する必要があります。この変数は任意で、指定する必要はありません。	**show interface counters protocol status**

次の表に、sl_intf_down.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 69: sl_intf_down.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_config_cmd1	実行される 1 番めのコンフィギュレーション コマンド。	**interface Ethernet1/0**
_config_cmd2	実行される 2 番めのコンフィギュレーション コマンド。この変数は任意で、指定する必要はありません。	**no shutdown**
_syslog_pattern	ポリシー実行時を決定するために syslog メッセージを比較するために使用する正規表現パターン マッチ文字列。	.*UPDOWN.*FastEthernet0/0.*

次の表に、tm_cli_cmd.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 70: tm_cli_cmd.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_cron_entry	ポリシーが実行されるときを決定する CRON 仕様。	0-59/1 0-23/1 * * 0-7

環境変数	説明	例
_show_cmd	ポリシーの実行時に実行される CLI コマンド。	**show version**

次の表に、tm_crash_reporter.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 71: tm_crash_reporter.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_crash_reporter_debug	tm_crash_reporter.tcl のデバッグ情報がイネーブルであるかどうかを決定する値。この変数は任意で、指定する必要はありません。	1
_crash_reporter_url	クラッシュ レポートが送信される URL 位置。	http://www.example.com/fm/interface_tm.cgi

次の表に、tm_fsys_usage.tcl サンプル ポリシーの実行前に設定する必要がある EEM 環境変数を示します。

***Table 72: tm_fsys_usage.tcl* ポリシーで使用される環境変数**

環境変数	説明	例
_tm_fsys_usage_cron	**event_register** Tcl コマンド拡張で使用される CRON 仕様。指定されない場合、tm_fsys_usage.tcl ポリシーが 1 分に 1 回トリガーされます。この変数は任意で、指定する必要はありません。	0-59/1 0-23/1 * * 0-7
_tm_fsys_usage_debug	この変数が値 1 に設定された場合、システムのすべてのエントリのディスク使用率情報が表示されます。この変数は任意で、指定する必要はありません。	1
_tm_fsys_usage_freebytes	システムまたは特定のプレフィックスの空きバイト数しきい値。空きスペースが所定の値を下回ると、警告が表示されます。この変数は任意で、指定する必要はありません。	disk2:98000000
_tm_fsys_usage_percent	システムまたは特定のプレフィックスのディスク使用割合しきい値。ディスク使用割合が所定の割合を超えると、警告が表示されます。指定されない場合、すべてのシステムのデフォルトのディスク使用割合は、80% です。この変数は任意で、指定する必要はありません。	nvram:25 disk2:5

一部の EEM ポリシーの登録

ポリシーの登録後に EEM 環境変数が変更された場合、一部の EEM ポリシーは、登録を解除し、再登録する必要があります。ポリシーの開始時に表示される event_register_*xxx* ステートメントには、一部の EEM 環境変数が含まれ、このステートメントは、ポリシーが実行される条件の確立に使用されます。ポリシーの登録後に環境変数が変更された場合、条件は無効になります。いかなるエラーも回避するには、ポリシーの登録を解除し、再登録する必要があります。次の変数に影響が及ぼされます。

- _cron_entry in the tm_cli_cmd.tcl policy
- _syslog_pattern in the sl_intf_down.tcl policy

すべてのサンプル ポリシーの基本設定の詳細

Embedded Event Manager から電子メールを送信できるようにするには、**hostname** コマンドと **ip domain-name** コマンドを設定する必要があります。EEM 環境変数も設定する必要があります。Cisco IOS イメージのブート後、次の初期設定を使用し、ネットワークで適切な値を置き換えます。tm_fsys_usage サンプル ポリシーの環境変数（上の表を参照）はすべて任意で、ここではそのリストは示されていません。

```
hostname cpu
ip domain-name example.com
event manager environment _email_server ms.example.net
event manager environment _email_to username@example.net
event manager environment _email_from engineer@example.net
event manager environment _email_cc projectgroup@example.net
event manager environment _cron_entry 0-59/2 0-23/1 * * 0-7
event manager environment _show_cmd show event manager policy registered
event manager environment _syslog_pattern .*UPDOWN.*FastEthernet0/0
event manager environment _config_cmd1 interface Ethernet1/0
event manager environment _config_cmd2 no shutdown
event manager environment _crash_reporter_debug 1
event manager environment _crash_reporter_url
http://www.example.com/fm/interface_tm.cgi
end
```

サンプル ポリシーの使用

ここでは、次の設定シナリオを使用して、Tcl サンプル ポリシーを使用する方法について説明します。

Mandatory.go_*.tcl サンプル ポリシーの実行

GOLD EEM ポリシーの一部として実行される各テストに GOLD TCL スクリプトがあります。この TCL スクリプトをテスト用に変更したり、連続障害回数を指定することができ、また、デフォルトの是正アクションを変更することもできます。たとえば、他の CLI ベースのアクションをリセットするのではなく、ラインカードの電源を切ることができます。

登録済みのテストごとにデフォルトの TCL スクリプトを使用できます。このスクリプトはシステムに登録し、デフォルトのアクションと一致させることができます。これは、これらのスクリプトによってオーバーライドできます。

次の表は、GOLD が EEM にインストールした必須ポリシーのリストです。各ポリシーが、カードのリセットやポートの無効化といった何らかのアクションを実行します。

GOLD Tcl スクリプト	テスト
Mandatory.go_asicsync.tcl	TestAsicSync
Mandatory.go_bootup.tcl	すべての ブートアップ テストに共通。
Mandatory.go_fabric.tcl	TestFabricHealth
Mandatory.go_fabrich0.tcl	TestFabricCh0Health
Mandatory.go_fabrich1.tcl	TestFabricCh1Health
Mandatory.go_ipsec.tcl	TestIPSecEncrypDecrypPkt
Mandatory.go_mac.tcl	TestMacNotification
Mandatory.go_nondislp.tcl	TestNonDisruptiveLoopback
Mandatory.go_scratchreg.tcl	TestScratchRegister
Mandatory.go_sprping.tcl	TestSPRPInbandPing

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザー EXEC モードを開始し、デバイス プロンプトで **enable** コマンドを入力します。デバイスは特権 EXEC モードを開始します。このモードで **show event manager policy registered** コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する **show event manager policy available** コマンドです。**configure terminal** コマンドを入力してグローバル コンフィギュレーション モードが開始されたら、**event manager policy** コマンドを使用して mandatory.go_*.tcl ポリシーを EEM に登録できます。グローバル コンフィギュレーション モードを終了し、もう一度 **show event manager policy registered** コマンドを入力してポリシーが登録されていることを確認します。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy Mandatory.go_spuriousisr.tcl
 end
show event manager policy registered
show event manager environment
```

ap_perf_test_base_cpu.tcl および no_perf_test_init.tcl サンプル ポリシーの実行

これらのサンプル ポリシーは、EEM ポリシーの CPU パフォーマンスを測定します。これらのポリシーは、各 EEM ポリシーの標準実行時間の検出に役立ち、CLI ライブラリ コマンドを使用して EEM 環境変数の perf_cmd1（任意で _perf_cmd2 および _perf_cmd3）で指定されているコンフィギュレーション コマンドを実行します。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザー EXEC モードを開始し、デバイス プロンプトで **enable** コマンドを入力します。デバイスは特権 EXEC モー

ドを開始します。このモードで **show event manager policy registered** コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する **show event manager policy available** コマンドです。**configure terminal** コマンドを入力してグローバル コンフィギュレーション モードが開始された後に **service timestamps debug datetime msec** コマンドを入力すると、**event manager policy** コマンドを使用して EEM に ap_perf_test_base_cpu.tcl ポリシーと no_perf_test_init.tcl ポリシーを登録できます。グローバル コンフィギュレーション モードを終了し、もう一度 **show event manager policy registered** コマンドを入力してポリシーが登録されていることを確認します。

ポリシー ap_perf_test_base_cpu.tcl および no_perf_test_init.tcl はセットで実行されるので、一緒に登録する必要があります。no_perf_test_init.tcl ポリシーを実行し、テストを開始することができます。反復ごとに返ってくる syslog メッセージを使用して結果を分析します。反復の総回数は、変数 _perf_iterations で指定します。時間の差を測り、反復の総回数で除算して、各 EEM ポリシーの平均実行時間を計算します。

```
enable
show event manager policy registered
show event manager policy available
show event manager environment
configure terminal
 service timestamps debug datetime msec
 event manager environment _perf_iterations 100
 event manager policy ap_perf_test_base_cpu.tcl
 event manager policy no_perf_test_init.tcl
 end
show event manager policy registered
show event manager policy available
show event manager environment
event manager run no_perf_test_init.tcl
```

no_perf_test_init.tcl サンプル ポリシーの実行

このサンプル ポリシーでは、EEM ポリシーの CPI パフォーマンスを測定します。このポリシーは、各 EEM ポリシーの標準実行時間の検出に役立ち、CLI ライブラリ コマンドを使用して EEM 環境変数の perf_cmd1（任意で _perf_cmd2 および _perf_cmd3）で指定されているコンフィギュレーション コマンドを実行します。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザー EXEC モードを開始し、デバイス プロンプトで **enable** コマンドを入力します。デバイスは特権 EXEC モードを開始します。このモードで **show event manager policy registered** コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する **show event manager policy available** コマンドです。**configure terminal** コマンドを入力してグローバル コンフィギュレーション モードが開始されたら、**event manager policy** コマンドを使用して no_perf_test_init.tcl ポリシーを EEM に登録できます。グローバル コンフィギュレーション モードを終了し、もう一度 **show event manager policy registered** コマンドを入力してポリシーが登録されていることを確認します。

反復ごとに返ってくる syslog メッセージを使用して結果を分析します。反復の総回数は、変数 _perf_iterations で指定します。時間の差を測り、反復の総回数で除算して、各 EEM ポリシーの平均実行時間を計算します。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy no_perf_test_init.tcl
 end
show event manager policy registered
show event manager environment
```

sl_intf_down.tcl サンプル ポリシーの実行

このサンプル ポリシーでは、特定のパターンで Syslog メッセージが記録されるときに設定を変更する機能について説明します。ポリシーでは、イベントについての詳細情報が収集され、CLI ライブラリを使用して、EEM 環境変数 _config_cmd1 と、任意で _config_cmd2 で指定された、コンフィギュレーション コマンドが実行されます。CLI コマンドの結果とともに、電子メール メッセージが送信されます。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザー EXEC モードを開始し、デバイス プロンプトで **enable** コマンドを入力します。デバイスは特権 EXEC モードを開始します。このモードで **show event manager policy registered** コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する **show event manager policy available** コマンドです。**configure terminal** コマンドを入力してグローバル コンフィギュレーション モードが開始されたら、**event manager policy** コマンドを使用して sl_intf_down.tcl ポリシーを EEM に登録できます。グローバル コンフィギュレーション モードを終了し、もう一度 **show event manager policy registered** コマンドを入力してポリシーが登録されていることを確認します。

インターフェイスがダウンするときに、ポリシーが実行されます。**show event manager environment** コマンドを入力して現在の環境変数の値を表示します。_syslog_pattern EEM 環境変数で指定されたインターフェイスのケーブルを取り外します（またはシャットダウンを設定します）。インターフェイスがダウンし、インターフェイスがダウンしていることについての Syslog メッセージを記録する Syslog デーモンのプロンプトが表示されて、Syslog イベント ディテクタが呼び出されます。

Syslog イベント ディテクタによって、未解決のイベント仕様が見直され、インターフェイス ステータス変更に対する一致が検索されます。EEM サーバーに通知され、サーバーでは、このイベント sl_intf_down.tcl を処理するために登録されたポリシーが実行されます。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy sl_intf_down.tcl
 end
show event manager policy registered
show event manager environment
```

tm_cli_cmd.tcl サンプル ポリシーの実行

このサンプル ポリシーでは、定期的に CLI コマンドを実行し、結果を E メールで送信する機能について説明します。CRON 仕様「0-59/2 0-23/1 * * 0-7」を使用すると、このポリシーは、毎時 2 分目に実行されます。ポリシーでは、イベントについての詳細情報が収集され、CLI ラ

イブラリを使用して、EEM 環境変数 _show_cmd で指定された、コンフィギュレーション コマンドが実行されます。CLI コマンドの結果とともに、電子メール メッセージが送信されます。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザー EXEC モードを開始し、デバイス プロンプトで **enable** コマンドを入力します。デバイスは特権 EXEC モードを開始します。このモードで **show event manager policy registered** コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する **show event manager policy available** コマンドです。**configure terminal** コマンドを入力してグローバル コンフィギュレーション モードが開始されたら、**event manager policy** コマンドを使用して tm_cli_cmd.tcl ポリシーを EEM に登録できます。グローバル コンフィギュレーション モードを終了し、**show event manager policy registered** コマンドを入力してポリシーが登録されていることを確認します。

EEM 環境変数 _cron_entry に設定されている CRON 文字列に従って、タイマー イベント ディテクタによって、定期的にこのケースのイベントがトリガーされます。EEM サーバーに通知され、サーバーでは、このイベント tm_cli_cmd.tcl を処理するために登録されたポリシーが実行されます。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_cli_cmd.tcl
 end
show event manager policy registered
```

tm_crash_reporter.tcl サンプル ポリシーの実行

このサンプル ポリシーでは、ある URL へ HTTP 形式のクラッシュ レポートを送信する機能について説明します。ポリシー登録がスタートアップ コンフィギュレーション ファイルに保存されている場合、ポリシーは、ブートの 5 秒後にトリガーされます。トリガーされると、スクリプトによって、リロード原因の検索が試行されます。リロードの原因がクラッシュの場合、ポリシーによって、関連する crashinfo ファイルが検索され、環境変数 _crash_reporter_url でユーザーによって指定された URL へ、この情報が送信されます。CGI スクリプト interface_tm.cgi は、tm_crash_reporter.tcl ポリシーから URL を受け取るために作成され、ターゲット URL マシン上のローカル データベースにクラッシュ情報が保存されます。

Perl CGI スクリプト interface_tm.cgi が作成され、HTTP サーバーが含まれているマシン上で実行するために設計され、tm_crash_reporter.tcl ポリシーが実行されているデバイスからアクセスできます。interface_tm.cgi スクリプトによって、tm_crash_reporter.tcl から渡されたデータが解析され、テキスト ファイルの末尾にクラッシュ情報が追加され、これによって、システムのすべてのクラッシュの履歴が作成されます。さらに、各クラッシュの詳細情報は、ユーザーが指定したクラッシュ データベース ディレクトリの 3 つのファイルに保存されます。別の Perl CGI スクリプト crash_report_display.cgi は、interface_tm.cgi スクリプトによって作成されたデータベースに保存されている情報を表示するために作成されました。crash_report_display.cgi スクリプトは、interface_tm.cgi が含まれているマシンと同じマシンに置く必要があります。そのマシンでは、Internet Explorer または Netscape などのブラウザが実行されている必要があります。crash_report_display.cgi スクリプトが実行されると、読み取り可能な形式でクラッシュ情報が表示されます。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザーEXECモードを開始し、デバイスプロンプトで**enable**コマンドを入力します。デバイスは特権EXECモードを開始します。このモードで**show event manager policy registered**コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する**show event manager policy available**コマンドです。**configure terminal**コマンドを入力してグローバルコンフィギュレーションモードが開始されたら、**event manager policy**コマンドを使用してtm_crash_reporter.tclポリシーをEEMに登録できます。グローバルコンフィギュレーションモードを終了し、**show event manager policy registered**コマンドを入力してポリシーが登録されていることを確認します。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_crash_reporter.tcl
 end
show event manager policy registered
```

tm_fsys_usage.tcl サンプル ポリシーの実行

このサンプルポリシーでは、ディスク領域の使用状況を定期的にモニターし、値が設定可能なしきい値に近くなったときにSyslogを介してレポートする機能について説明します。

次に、このポリシーの使用方法を示すサンプル設定について説明します。ユーザーEXECモードを開始し、デバイスプロンプトで**enable**コマンドを入力します。デバイスは特権EXECモードを開始します。このモードで**show event manager policy registered**コマンドを入力すると、現在登録されているポリシーがないことを確認できます。次のコマンドはどのポリシーがインストールできるかを表示する**show event manager policy available**コマンドです。**configure terminal**コマンドを入力してグローバルコンフィギュレーションモードが開始されたら、**event manager policy**コマンドを使用してtm_fsys_usage.tclポリシーをEEMに登録できます。グローバルコンフィギュレーションモードを終了し、もう一度**show event manager policy registered**コマンドを入力してポリシーが登録されていることを確認します。tm_fsys_usage.tclポリシーで使用されるオプション環境変数のいずれかを設定した場合、**show event manager environment**コマンドによって、設定された変数が表示されます。

```
enable
show event manager policy registered
show event manager policy available
configure terminal
 event manager policy tm_fsys_usage.tcl
 end
show event manager policy registered
show event manager environment
```

Tcl のサンプル スクリプトを使用したポリシーのプログラミングの例

ここでは、EEMシステムポリシーとして含まれているいくつかのサンプルポリシーについて説明します。これらのポリシーの詳細については、EEM イベント ディテクタのデモの例, on page 703を参照してください。

Mandatory.go_ipsec.tcl サンプル ポリシー

次のサンプルポリシーは、TestIPSecEncrypDecrypPkt テスト用です。

```
::cisco::eem::event_register_gold card all testing_type monitoring test_name Tes
tIPSecEncrypDecrypPkt consecutive_failure 6 platform_action 0 queue_priority las
t
#
#  GOLD TestIPSecEncrypDecrypPkt Test TCL script
#
#  March 2005, Hai Qiu
#
#  Copyright (c) 2005-2007 by cisco Systems, Inc.
#  All rights reserved.
#
#
# Register for TestIPSecEncrypDecrypPkt test even
# the elements for register the event
# card [all | card #]
# sub_card [all | sub_card #]
# severity_major | severity_minor | severity_normal  default :  severity_normal
# new_failure [true | false]  default: dont_care
# testing_type [bootup | ondemand | schedule | monitoring]
# test_name [ test name ]
# test_id [ test # ]
# consecutive_failure [ consecutive_failure # ]
# platform_action [action_flag]
# action_flag [ 0 | 1 | 2 ]
# queue_priority [ normal | low | high | last]   default: normal
#
# Note:
#    1: "card" element is required. If other elements are not specified,
#       treat them as dont care, or default.
#
#    2: action_flag is platform specific. It is up to platform to
#       determine what action need to be taken based on the value
#       For Cat6k platform
#       action_flag 0 : TCL script take action to reset card
#       action_flag 1 : TCL script doesn't take action to reset card
#       action_flag 2 : TCL script takes action to reset card for bootup diag
#                       when there is major error
#       action_flag 3 : TCL script doesn't take action to reset card for
#                       bootup diag when there is major error
#
#    3: "queue_priority last" would guarantee this policy will be executed last
#       if there are other EEM events in queue with queue priority other
#       than "last"
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# 1. query the information of latest triggered eem event
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
puts "GOLD EEM TCL policy for TestIPSecEncrypDecrypPkt"
#set msg [format "array=%s", array names arr_einfo]
#puts "msg $msg"
#set msg $arr_einfo(msg)
set card $arr_einfo(card)
set sub_card $arr_einfo(sub_card)
#set overall_result $arr_einfo(overall_result)
```

```
#puts "GOLD event msg recieved: $card/$sub_card overall_result= $overall_result"
# 2. execute the user-defined config commands
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# Use "diagn action mod mod# test testname default" command
# for default platform action
if [catch {cli_exec $cli1(fd) "diagnostic action mod $card test TestIPSecEncrypD
ecrypPkt default"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}
```

ap_perf_test_base_cpu.tcl サンプル ポリシー

次のサンプル ポリシーは、EEM ポリシーの CPU パフォーマンスを測定します。

```
::cisco::eem::event_register_appl sub_system 798 type 9999
#----------------------------------
# EEM policy used for measuring the cpu performance of EEM policies.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005, 2006 by cisco Systems, Inc.
# All rights reserved.
#------------------
###
### Input arguments:
###
### arg1 $iter                   - current iteration count
###
### The following EEM environment variables are used:
###
### _perf_iterations (mandatory) - number of iterations over which we
###                                will run our measurement.
### Example:
### event manager environment _perf_iterations 100
###
### _perf_cmd1 (optional)        - optional non interactive cli command
###                                to be executed as part of the
###                                measurement test.
### Example:
### event manager environment _perf_cmd1 enable
###
### _perf_cmd2 (optional)        - optional non interactive cli command
###                to be executed as part of the
###                                measurement test.
###                                To use _perf_cmd2, _perf_cmd1 MUST
###                be defined.
### Example:
### event manager environment _perf_cmd2 show ver
###
### _perf_cmd3 (optional)        - optional non interactive cli command
###                                to be executed as part of the
```

```
###                                 measurement test.
###                                 To use _perf_cmd3, _perf_cmd1 MUST
###                                 be defined.
### Example:
### event manager environment _perf_cmd3 show int counters protocol status
###
### Description:
###   Iterate through _perf_iterations of this policy.
###   It is up to the user to calculate the average
###   execution time based on the system timestamps.
###   Optional commands _perf_cmd1,
###   _perf_cmd2 and _perf_cmd3 are executed if defined.
###
###   A value of 100 is a good starting point.
###
### Outputs:
###   Console output.
###
### Usage example:
###   >conf t
###   >service timestamps debug datetime msec
###   >event manager environment _perf_iterations 100
###   >event manager policy ap_perf_base_cpu.tcl
###   >event manager policy no_perf_test_init.tcl
###   >end
###   2d19h: %SYS-5-CONFIG_I: Configured from console by console
###   >event manager run no_perf_test_init.tcl
###
###   Oct 16 14:57:17.284: %SYS-5-CONFIG_I: Configured from console by console
###   >event manager run no_perf_test_init.tcl
###
###   Oct 16 19:32:02.772: %HA_EM-6-LOG:
###     eem_policy/no_perf_test_init.tcl: EEM performance test start
###   Oct 16 19:32:03.115: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 1
###   Oct 16 19:32:03.467: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 2
###   ...
###   Oct 16 19:32:36.936: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test iteration 100
###   Oct 16 19:32:36.936: %HA_EM-6-LOG:
###     eem_policy/ap_perf_test_base_cpu.tcl: EEM performance test end
###
###   The user must calculate execution time and average time of execution.
###   In this example, total time = 19:32:36.936 - 19:32:02.772 = 34.164
###   Average script execution time = 341.64 milliseconds
###
# check if all the env variables we need exist
# If any of them doesn't exist, print out an error msg and quit
if {![info exists _perf_iterations]} {
    set result \
        "Policy cannot be run: variable _perf_iterations has not been set"
    error $result $errorInfo
}
# ensure our target iteration count > 0
if {$_perf_iterations <= 0} {
    set result \
        "Policy cannot be run: variable _perf_iterations <= 0"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info
array set arr_einfo [event_reqinfo]
```

```
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
set iter $arr_einfo(data1)
set iter [expr $iter + 1]
# if _perf_cmd1 is defined
if {[info exists _perf_cmd1]} {
    # open the cli library
    if [catch {cli_open} result] {
        error $result $errorInfo
    } else {
        array set cli1 $result
    }
    # execute the comamnd defined in _perf_cmd1
    if [catch {cli_exec $cli1(fd) $_perf_cmd1} result] {
        error $result $errorInfo
    }
    # if _perf_cmd2 is defined
    if {[info exists _perf_cmd2]} {
        # execute the comamnd defined in _perf_cmd2
        if [catch {cli_exec $cli1(fd) $_perf_cmd2} result] {
            error $result $errorInfo
        } else {
            set cmd_output $result
        }
    }
    # if _perf_cmd3 is defined
    if {[info exists _perf_cmd3]} {
        # execute the comamnd defined in _perf_cmd3
        if [catch {cli_exec $cli1(fd) $_perf_cmd3} result] {
            error $result $errorInfo
        } else {
            set cmd_output $result
        }
    }
    # close the cli library
    if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
        error $result $errorInfo
    }
}

# log a message
set msg [format "EEM performance test iteration %s" $iter]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# use the context info from the previous run to determine when to end
if {$iter >= $_perf_iterations} {
    #log the final messages
    action_syslog priority info msg "EEM performance test end"
    if {$_cerrno != 0} {
        set result [format \
                "component=%s; subsys err=%s; posix err=%s;\n%s" \
                $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
        error $result
    }
    exit 0
}
# cause the next iteration to run
```

```
event_publish sub_system 798 type 9999 arg1 $iter
if {$_cerrno != 0} {
    set result [format \
            "component=%s; subsys err=%s; posix err=%s;\n%s" \
            $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

tm_cli_cmd.tcl サンプル ポリシー

次に、設定可能な CRON エントリが実行されるサンプル ポリシーについて説明します。ポリシーでは、設定可能な Cisco IOS CLI コマンドが実行され、結果が電子メールで送信されます。タイムスタンプとともに出力が末尾に追加される任意のログ ファイルを定義することができます。

```
::cisco::eem::event_register_timer cron name crontimer2 cron_entry $
_cron_entry maxrun 240
#------------------------------------------------------------------
# EEM policy that will periodically execute a cli command and email the
# results to a user.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#------------------------------------------------------------------
### The following EEM environment variables are used:
###
### _cron_entry (mandatory)            - A CRON specification that determines
###                                      when the policy will run. See the
###                                      IOS Embedded Event Manager
###                                      documentation for more information
###                                      on how to specify a cron entry.
### Example: _cron_entry                 0-59/1 0-23/1 * * 0-7
###
### _log_file (mandatory without _email_....)
###                                    - A filename to append the output to.
###                                      If this variable is defined, the
###                                      output is appended to the specified
###                                      file with a timestamp added.
### Example: _log_file                   disk0:/my_file.log
###
### _email_server (mandatory without _log_file)
###                                    - A Simple Mail Transfer Protocol (SMTP)
###                                      mail server used to send e-mail.
### Example: _email_server               mailserver.example.com
###
### _email_from (mandatory without _log_file)
###                                    - The address from which e-mail is sent.
### Example: _email_from                 devtest@example.com
###
### _email_to (mandatory without _log_file)
###                                    - The address to which e-mail is sent.
### Example: _email_to                   engineering@example.com
###
### _email_cc (optional)               - The address to which the e-mail must
###                                      be copied.
### Example: _email_cc                   manager@example.com
###
### _show_cmd (mandatory)              - The CLI command to be executed when
###                                      the policy is run.
```

```
### Example: _show_cmd                   show version
###
# check if all required environment variables exist
# If any required environment variable does not exist, print out an error msg and quit
if {![info exists _log_file]} {
    if {![info exists _email_server]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_server has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_from]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_from has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_to]} {
 set result \
  "Policy cannot be run: variable _log_file ore _email_to has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_cc]} {
 #_email_cc is an option, must set to empty string if not set.
 set _email_cc ""
    }
}
if {![info exists _show_cmd]} {
    set result \
        "Policy cannot be run: variable _show_cmd has not been set"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info and log a message
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)
# log a message
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
 $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 1. execute the command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# save exact execution time for command
set time_now [clock seconds]
# execute command
if [catch {cli_exec $cli1(fd) $_show_cmd} result] {
```

```
    error $result $errorInfo
} else {
    set cmd_output $result
    # format output: remove trailing router prompt
    regexp {\n*(.*\n)([^\n]*)$} $result dummy cmd_output
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}

# 2. log the success of the CLI command
set msg [format "Command \"%s\" executed successfully" $_show_cmd]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 3. if _log_file is defined, then attach it to the file
if {[info exists _log_file]} {
    # attach output to file
    if [catch {open $_log_file a+} result] {
        error $result
    }
    set fileD $result
    # save timestamp of command execution
    #      (Format = 00:53:44 PDT Mon May 02 2005)
    set time_now [clock format $time_now -format "%T %Z %a %b %d %Y"]
    puts $fileD "%%% Timestamp = $time_now"
    puts $fileD $cmd_output
    close $fileD
}
# 4. if _email_server is defined send the email out
if {[info exists _email_server]} {
    set routername [info hostname]
    if {[string match "" $routername]} {
 error "Host name is not configured"
    }
    if [catch {smtp_subst [file join $tcl_library email_template_cmd.tm]} \
     result] {
 error $result $errorInfo
    }
    if [catch {smtp_send_email $result} result] {
 error $result $errorInfo
    }
}


::cisco::eem::event_register_timer cron name crontimer2 cron_entry $
_cron_entry maxrun 240
#-------------------------------------------------------------------
# EEM policy that will periodically execute a cli command and email the
# results to a user.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#-------------------------------------------------------------------
### The following EEM environment variables are used:
###
### _cron_entry (mandatory)            - A CRON specification that determines
###                                      when the policy will run. See the
###                                      IOS Embedded Event Manager
###                                      documentation for more information
```

```
###                                     on how to specify a cron entry.
### Example: _cron_entry                0-59/1 0-23/1 * * 0-7
###
### _log_file (mandatory without _email_....)
###                                     - A filename to append the output to.
###                                       If this variable is defined, the
###                                       output is appended to the specified
###                                       file with a timestamp added.
### Example: _log_file                  bootflash:/my_file.log
###
### _email_server (mandatory without _log_file)
###                                     - A Simple Mail Transfer Protocol (SMTP)
###                                       mail server used to send e-mail.
### Example: _email_server              mailserver.example.com
###
### _email_from (mandatory without _log_file)
###                                     - The address from which e-mail is sent.
### Example: _email_from                devtest@example.com
###
### _email_to (mandatory without _log_file)
###                                     - The address to which e-mail is sent.
### Example: _email_to                  engineering@example.com
###
### _email_cc (optional)                - The address to which the e-mail must
###                                       be copied.
### Example: _email_cc                  manager@example.com
###
### _show_cmd (mandatory)               - The CLI command to be executed when
###                                       the policy is run.
### Example: _show_cmd                  show version
###
# check if all required environment variables exist
# If any required environment variable does not exist, print out an error msg and quit
if {![info exists _log_file]} {
    if {![info exists _email_server]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_server has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_from]} {
 set result \
  "Policy cannot be run: variable _log_file or _email_from has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_to]} {
 set result \
  "Policy cannot be run: variable _log_file ore _email_to has not been set"
 error $result $errorInfo
    }
    if {![info exists _email_cc]} {
 #_email_cc is an option, must set to empty string if not set.
 set _email_cc ""
    }
}
if {![info exists _show_cmd]} {
    set result \
        "Policy cannot be run: variable _show_cmd has not been set"
    error $result $errorInfo
}
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# query the event info and log a message
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
```

```
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
global timer_type timer_time_sec
set timer_type $arr_einfo(timer_type)
set timer_time_sec $arr_einfo(timer_time_sec)
# log a message
set msg [format "timer event: timer type %s, time expired %s" \
        $timer_type [clock format $timer_time_sec]]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
 $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 1. execute the command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
# save exact execution time for command
set time_now [clock seconds]
# execute command
if [catch {cli_exec $cli1(fd) $_show_cmd} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
    # format output: remove trailing router prompt
    regexp {\n*(.*\n)([^\n]*)$} $result dummy cmd_output
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}

# 2. log the success of the CLI command
set msg [format "Command \"%s\" executed successfully" $_show_cmd]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# 3. if _log_file is defined, then attach it to the file
if {[info exists _log_file]} {
    # attach output to file
    if [catch {open $_log_file a+} result] {
        error $result
    }
    set fileD $result
    # save timestamp of command execution
    #       (Format = 00:53:44 PDT Mon May 02 2005)
    set time_now [clock format $time_now -format "%T %Z %a %b %d %Y"]
    puts $fileD "%%% Timestamp = $time_now"
    puts $fileD $cmd_output
    close $fileD
}
# 4. if _email_server is defined send the email out
if {[info exists _email_server]} {
    set routername [info hostname]
```

```
    if {[string match "" $routername]} {
 error "Host name is not configured"
    }
    if [catch {smtp_subst [file join $tcl_library email_template_cmd.tm]} \
     result] {
 error $result $errorInfo
    }
    if [catch {smtp_send_email $result} result] {
 error $result $errorInfo
    }
}
```

sl_intf_down.tcl サンプル ポリシー

次に、設定可能な Syslog メッセージが記録されるときに実行されるサンプル ポリシーを示します。ポリシーでは、設定可能な CLI コマンドが実行され、結果が電子メールで送信されます。

```
::cisco::eem::event_register_syslog occurs 1 pattern $_syslog_pattern maxrun 90

#-------------------------------------------------------------------
# EEM policy to monitor for a specified syslog message.
# Designed to be used for syslog interface-down messages.
# When event is triggered, the given config commands will be run.
#
# July 2005, Cisco EEM team
#
# Copyright (c) 2005 by cisco Systems, Inc.
# All rights reserved.
#-------------------------------------------------------------------

### The following EEM environment variables are used:
###
### _syslog_pattern (mandatory)        - A regular expression pattern match string
###                                      that is used to compare syslog messages
###                                      to determine when policy runs
### Example: _syslog_pattern             .*UPDOWN.*FastEthernet0/0.*
###
### _email_server (mandatory)          - A Simple Mail Transfer Protocol (SMTP)
###                                      mail server used to send e-mail.
### Example: _email_server               mailserver.example.com
###
### _email_from (mandatory)            - The address from which e-mail is sent.
### Example: _email_from                 devtest@example.com
###
### _email_to (mandatory)              - The address to which e-mail is sent.
### Example: _email_to                   engineering@example.com
###
### _email_cc (optional)               - The address to which the e-mail must
###                                      be copied.
### Example: _email_cc                   manager@example.com
###
### _config_cmd1 (optional)            - The first configuration command that
###                                      is executed.
### Example: _config_cmd1                interface Ethernet1/0
###
### _config_cmd2 (optional)            - The second configuration command that
###                                      is executed.
### Example: _config_cmd2                no shutdown
###

# check if all the env variables we need exist
```

```
# If any of them doesn't exist, print out an error msg and quit
if {![info exists _email_server]} {
    set result \
        "Policy cannot be run: variable _email_server has not been set"
    error $result $errorInfo
}
if {![info exists _email_from]} {
    set result \
        "Policy cannot be run: variable _email_from has not been set"
    error $result $errorInfo
}
if {![info exists _email_to]} {
    set result \
        "Policy cannot be run: variable _email_to has not been set"
    error $result $errorInfo
}
if {![info exists _email_cc]} {
     #_email_cc is an option, must set to empty string if not set.
     set _email_cc ""
}

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

# 1. query the information of latest triggered eem event
array set arr_einfo [event_reqinfo]

if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

set msg $arr_einfo(msg)
set config_cmds ""

# 2. execute the user-defined config commands
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
if [catch {cli_exec $cli1(fd) "config t"} result] {
    error $result $errorInfo
}

if {[info exists _config_cmd1]} {
    if [catch {cli_exec $cli1(fd) $_config_cmd1} result] {
        error $result $errorInfo
    }
    append config_cmds $_config_cmd1
}

if {[info exists _config_cmd2]} {
    if [catch {cli_exec $cli1(fd) $_config_cmd2} result] {
        error $result $errorInfo
    }
    append config_cmds "\n"
    append config_cmds $_config_cmd2
}
```

```
if [catch {cli_exec $cli1(fd) "end"} result] {
    error $result $errorInfo
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}

after 60000
# 3. send the notification email
set routername [info hostname]
if {[string match "" $routername]} {
    error "Host name is not configured"
}

if [catch {smtp_subst [file join $tcl_library email_template_cfg.tm]} result] {
    error $result $errorInfo
}
if [catch {smtp_send_email $result} result] {
    error $result $errorInfo
}
```

次に、前述の EEM サンプル ポリシーで使用される電子メール テンプレート ファイルの使用例を示します。

```
email_template_cfg.tm
Mailservername: $_email_server
From: $_email_from
To: $_email_to
Cc: $_email_cc
Subject: From router $routername: Periodic $_show_cmd Output
$cmd_output
```

Embedded Event Manager ポリシーのデバッグの例

次に、CLI ライブラリおよび SMTP ライブラリのデバッグ例を示します。

CLI ライブラリのデバッグ

CLI ライブラリを使用すると、ユーザーは、CLI コマンドを実行し、Tcl のコマンドの出力を取得できます。Embedded Event Manager の **debug** コマンドは、このライブラリのユーザー向けに用意されています。CLI ライブラリのデバッグを有効にするコマンドは、**debug event manager tcl cli_library** です。イネーブルの場合、このコマンドによって、CLI のやり取りを処理する TTY セッションに渡され、TTY セッションから読み戻される、すべてのデータが表示されます。このデータを使用すると、ユーザーが CLI に渡すコマンドが有効になります。

デバッグ イベント マネージャ tcl cli_library コマンドの例

この例では、サンプル ポリシー sl_intf_down.tcl が使用されます。トリガーされると、sl_intf_down.tcl によって、CLI ライブラリを介して CLI にコンフィギュレーション コマンドが渡されます。次で渡されるコマンドは、**show event manager environment** です。このコマンドは、コンフィギュレーション モードでは有効ではありません。**debug** コマンドが有効ではない場合、出力は次のとおりです。

```
00:00:57:sl_intf_down.tcl[0]:config_cmds are show eve man env
00:00:57:%SYS-5-CONFIG_I:Configured from console by vty0
```

前述の出力で、ユーザーは、CLI でコマンドが正常終了したかどうかはわかりません。**debug event manager tcl cli_library** コマンドが有効である場合は、次が表示されます。

```
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : CTL : cli_open called.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson>
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson>enable
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson#configure terminal
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : Enter configuration commands, one
per line.  End with CNTL/Z.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson(config)#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson(config)#show event manager
environment
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT :                 ^
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : % Invalid input detected at '^'
marker.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson(config)#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson(config)#end
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : OUT : nelson#
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : CTL : cli_close called.
01:17:07: sl_intf_down.tcl[0]: DEBUG(cli_lib) : IN  : nelson#exit
01:17:07: sl_intf_down.tcl[0]: config_cmds are show event manager environment
01:17:07: %SYS-5-CONFIG_I: Configured from console by vty0
```

前述の出力には、**show event manager environment** コマンドがコンフィギュレーション モードでは無効であることが示されています。IN キーワードによって、CLI ライブラリを介して TTY へすべてのデータが渡されることが指定されます。OUT キーワードによって、CLI ライブラリを介して TTY からすべてのデータが読み戻されることが指定されます。CTL キーワードによって、CLI ライブラリで使用されるヘルパー機能が指定されます。これらのヘルパー機能は、CLI への接続の設定や、接続の削除に使用されます。

SMTP ライブラリのデバッグ

SMTP ライブラリを使用すると、ユーザーは、SMTP E メール サーバーへ、E メール メッセージを送信できます。Embedded Event Manager の **debug** コマンドは、このライブラリのユーザー向けに用意されています。SMTP ライブラリのデバッグを有効にするコマンドは、**debug event manager tcl smtp_library** です。イネーブルの場合、このコマンドによって、SMTP ライブラリ ルーチンに渡され、SMTP ライブラリ ルーチンから読み戻される、すべてのデータが表示されます。このデータを使用すると、ユーザーが SMTP ライブラリに渡すコマンドが有効になります。

デバッグ イベント マネージャ tcl smtp_library コマンドの例

この例では、サンプル ポリシー tm_cli_cmd.tcl が使用されます。トリガーされると、tm_cli_cmd.tcl は CLI ライブラリを介して **show event manager policy available system** コマンドを実行します。結果は、SMTP ライブラリを介してメールでユーザーに送信されます。出力を参考に、SMTP ライブラリを使用して、関連する問題をデバッグできます。

debug event manager tcl smtp_library コマンドが有効の場合は、コンソールに次が表示されます。

```
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 220 XXXX.example.com ESMTP
XXXX 1.1.0; Tue,
25 Jun 2002 14:20:39 -0700 (PDT)
```

```
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : HELO XXXX.example.com
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 XXXX.example.com Hello
XXXX.example.com [XXXX],
pleased to meet you
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : MAIL FROM:<XX@example.com>
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>...
Sender ok
00:39:46: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : RCPT TO:<XX@example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>...
Recipient ok
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : RCPT TO:<XX@example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 <XX@example.com>...
Recipient ok
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : DATA
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 354 Enter mail, end with "."
 on a line by itself
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Date: 25 Jun 2002 14:35:00
UTC
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Message-ID:
<20020625143500.2387058729877@XXXX.example.com>
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : From: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : To: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Cc: XX@example.com
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : Subject: From router nelson:

Periodic show eve man po ava system Output
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : No. Type    Time Created
          Name
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 1   system  Fri May3
20:42:34 2002  pr_cdp_abort.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 2   system  Fri May3
20:42:54 2002  pr_iprouting_abort.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 3   system  Wed Apr3
02:16:33 2002  sl_intf_down.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 4   system  Mon Jun24
23:34:16 2002  tm_cli_cmd.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : 5   system  Wed Mar27
05:53:15 2002  tm_crash_hist.tcl
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : nelson#
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write :
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : .
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 250 ADE90179 Message accepted
 for delivery
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_write : QUIT
00:39:47: tm_cli_cmd.tcl[0]: DEBUG(smtp_lib) : smtp_read  : 221 XXXX.example.com closing
 connection
```

Tcl set コマンド操作のトレースの例

Tcl は、融通性のある言語です。Tcl の融通性の 1 つは、コマンドを上書きできることです。この例では、Tcl **set** コマンドの名前が _set に変更されます。また、テキスト「setting」が含まれるメッセージを表示し、設定しているスカラ変数を末尾に追加する、新バージョンの **set** コマンドが作成されます。この例を使用すると、設定しているスカラ変数のすべてのインスタンスをトレースできます。

```
rename set _set
proc set {var args} {
   puts [list setting $var $args]
   uplevel _set $var $args
};
```

これがポリシーに置かれると、スカラ変数が設定されるたびに、たとえば次のようなメッセージが表示されます。

```
02:17:58: sl_intf_down.tcl[0]: setting test_var 1
```

RPC イベント ディテクタのの例

```
TCL script (rpccli.tcl):
::cisco::eem::event_register_rpc
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
proc run_cli { clist } {
    set rbuf ""
    if {[llength $clist] < 1} {
    return -code ok $rbuf
    }
    if {[catch {cli_open} result]} {
        return -code error $result
    } else {
    array set cliarr $result
    }
    if {[catch {cli_exec $cliarr(fd) "enable"} result]} {
        return -code error $result
    }
    if {[catch {cli_exec $cliarr(fd) "term length 0"} result]} {
        return -code error $result
    }
    foreach cmd $clist {
    if {[catch {cli_exec $cliarr(fd) $cmd} result]} {
            return -code error $result
    }
    append rbuf $result
    }
    if {[catch {cli_close $cliarr(fd) $cliarr(tty_id)} result]} {
        puts "WARNING: $result"
    }
    return -code ok $rbuf
}
proc run_cli_interactive { clist } {
    set rbuf ""
    if {[llength $clist] < 1} {
    return -code ok $rbuf
    }
    if {[catch {cli_open} result]} {
        return -code error $result
    } else {
    array set cliarr $result
    }
    if {[catch {cli_exec $cliarr(fd) "enable"} result]} {
        return -code error $result
    }
    if {[catch {cli_exec $cliarr(fd) "term length 0"} result]} {
        return -code error $result
    }
    foreach cmd $clist {
        array set sendexp $cmd
    if {[catch {cli_write $cliarr(fd) $sendexp(send)} result]} {
            return -code error $result
    }
    foreach response $sendexp(responses) {
```

```
        array set resp $response
        if {[catch {cli_read_pattern $cliarr(fd) $resp(expect)} result]} {
                return -code error $result
        }
        if {[catch {cli_write $cliarr(fd) $resp(reply)} result]} {
                return -code error $result
        }
    }
    if {[catch {cli_read $cliarr(fd)} result]} {
            return -code error $result
    }
    append rbuf $result
    }
    if {[catch {cli_close $cliarr(fd) $cliarr(tty_id)} result]} {
        puts "WARNING: $result"
    }
    return -code ok $rbuf
}
array set arr_einfo [event_reqinfo]
set args $arr_einfo(argc)
set cmds [list]
for { set i 0 } { $i < $args } { incr i } {
    set arg "arg${i}"
    # Split each argument on the '^' character.  The first element is
    # the command, and each subsequent element is a prompt followed by
    # a response to that prompt.
    set cmdlist [split $arr_einfo($arg) "^"]
    set cmdarr(send) [lindex $cmdlist 0]
    set cmdarr(responses) [list]
    if { [expr ([llength $cmdlist] - 1) % 2] != 0 } {
    return -code 88
    }
    set cmdarr(responses) [list]
    for { set j 1 } { $j < [llength $cmdlist] } { incr j 2 } {
    set resps(expect) [lindex $cmdlist $j]
    set resps(reply) [lindex $cmdlist [expr $j + 1]]
    lappend cmdarr(responses) [array get resps]
    }
    lappend cmds [array get cmdarr]
}
set rc [catch {run_cli_interactive $cmds} output]
if { $rc != 0 } {
    error $output $errorInfo
    return -code 88
}
puts $output
```

その他の参考資料

次の項では、Tcl を使用した Embedded Event Manager ポリシー記述についての関連資料を示します。

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』

関連項目	マニュアル タイトル
EEM コマンド：コマンド構文の詳細、デフォルト、コマンドモード、コマンド履歴、使用上の注意事項、および例	Cisco IOS Embedded Event Manager のコマンド リファレンス
Embedded Event Manager 概要	「Embedded Event Manager の概要」の章
CLI を使用して Embedded Event Manager ポリシーを記述する	「Writing Embedded Event Manager Policies Using the Cisco IOS CLI」の章
Embedded Resource Manager	「Embedded Resource Manager」の章

MIB

MIB	MIB のリンク
CISCO-EMBEDDED-EVENT-MGR-MIB	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
この機能によりサポートされた新規 RFC または改訂 RFC はありません。またこの機能による既存 RFC のサポートに変更はありません。	--

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Tcl を使用した Embedded Event Manager（EEM）4.0 ポリシー記述の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

***Table 73: Tcl** を使用した **Embedded Event Manager**（**EEM**）**4.0** ポリシー記述の機能情報*

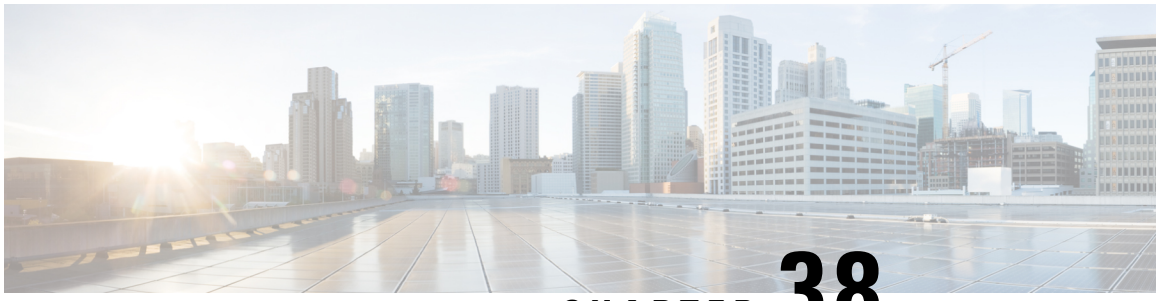
機能名	リリース	機能情報
Embedded Event Manager 1.0	12.0(26)S 12.3(4)T	EEM 1.0 は、Embedded Event Manager アプレット作成を SNMP イベント ディテクタ Syslog イベント ディテクタとともに追加しました。EEM 1.0 は、次のアクションも追加しました。優先化された syslog メッセージの生成、Cisco CNS デバイスによるアップストリーム処理に対し CNS イベントの生成、Cisco ソフトウェアのリロード、および完全冗長ハードウェア構成におけるセカンダリプロセッサへのスイッチング。 次のコマンドがこの機能で導入されました。**action cns-event**、**action force-switchover**、 **action reload**、 **action syslog**、 **debug event manager**、 **event manager applet**、 **event snmp**、 **event syslog**、 **show event manager policy registered**。

機能名	リリース	機能情報
Embedded Event Manager 2.0	12.2(25)S	EEM 2.0 は、Application-Specific イベント ディテクタ、Counter イベント ディテクタ、Interface Counter イベント ディテクタ、Timer イベント ディテクタ、および watchdog イベント ディテクタを追加しました。新しいアクションには、名前付きカウンタの変更、アプリケーション固有イベントのパブリッシュ、SNMP トラップの生成が含まれました。環境変数定義機能、および、Tcl を使用して記述されたサンプル EEM ポリシーの実行機能が追加され、2 個のサンプル ポリシーがソフトウェアに追加されました。 次のコマンドがこの機能で導入されました。**action counter**、**action publish-event**、**action snmp-trap**、**event application**、**event counter**、**event interface**、**event ioswdsysmon**、**event manager environment**、**event manager history size**、**event manager policy**、**event manager scheduler suspend**、**event timer**, **show event manager environment**、**show event manager history events**、**show event manager history traps**、**show event manager policy available**、**show event manager policy pending**。
Embedded Event Manager 2.1	12.3(14)T 12.2(18)SXF5 12.2(28)SB 12.2(33)SRA	EEM 2.1 は複数の新しいイベント ディテクタおよびアクション、EEM ポリシーを手動で起動する新しい機能と複数の共存ポリシーを起動する機能を追加しました。簡易ネットワーク管理プロトコル（SNMP）イベント ディテクタ比率ベース イベントのサポートが、Tool Command Language（Tcl）を使用してポリシーを作成する機能として導入されました。 次のコマンドがこの機能で導入されました。**action cli**、**action counter**、 **action info**、 **action mail**、**action policy**、**debug event manager**、 **event cli**、 **event manager directory user**, **event manager policy**、**event manager run**、**event manager scheduler script**、**event manager session cli username**、**event none**、**event oir**、**event snmp**, **event syslog**、**set**(EEM)、**show event manager directory user**、**show event manager policy registered**、**show event manager session cli username**。

機能名	リリース	機能情報
Embedded Event Manager 2.1（ソフトウェア モジュール方式）	12.2(18)SXF4 Cisco IOS ソフトウェアモジュール方式のイメージ	EEM 2.1 ソフトウェア モジュール方式イメージは、GOLD、system manager、および WDSysMon（Cisco IOS Software Modularity watchdog）イベント ディテクタ、および Cisco IOS ソフトウェア モジュール方式プロセスとプロセス メトリックを表示する機能を導入しました。 次のコマンドがこの機能で導入されました。**event gold**、**event process**、**show event manager metric process**。 **Note** EEM 2.1 ソフトウェア モジュール方式イメージは、Resource イベント ディテクタおよび RF イベント ディテクタを EEM 2.2 に追加しましたが、EOT イベント ディテクタ、またはトラッキング対象オブジェクトの読み込みおよび設定のアクションをサポートしません。
Embedded Event Manager 2.2	12.4(2)T 12.2(31)SB3 12.2(33)SRB	EEM 2.2 は、Enhanced Object Tracking、Resource、および RF イベント ディテクタを追加しました。トラッキング対象オブジェクトの状態の読み取りおよび設定のアクションも追加されました。 この機能により、次のコマンドが導入または変更されました。**action track read**、**action track set**、**default-state**、**event resource**、**event rf**、**event track**、**show track**、**track stub-object**。
SNMP イベント ディテクタ delta 環境変数	12.4(11)T	新しい SNMP イベント ディテクタ環境変数、_snmp_oid_delta_val が追加されました。 これはマイナーな拡張です。マイナーな拡張は、通常 Feature Navigator に記載されません。
Embedded Event Manager 2.3	12.2(33)SXH 12.2(33)SB 15.1(2)SY	EEM 2.3 では、Cisco Catalyst 6500 シリーズ スイッチ上の Generic Online Diagnostics (GOLD) イベント ディテクタに関連する新しい機能が追加されました。 **event gold** コマンドは、GOLD テスト失敗および条件への対応を改善するための **action-notify**、**testing-type**、**test-name**、**test-id**、**consecutive-failure**、**platform-action**、および **maxrun** キーワードが追加され、拡張されました。 検出されたイベントのプラットフォーム全体、および、テスト特有の GOLD イベントディテクタ情報へのアクセスを実現するために、読み取り専用変数が **GOLD Event Detector** カテゴリに追加されました。

機能名	リリース	機能情報
Embedded Event Manager 2.4	12.4(20)T 12.2(33)SXI 12.2(33)SRE 15.1(2)SY	EEM 2.4 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが追加されました。 **attribute (EEM)** 、**correlate**、**event manager detector rpc**、**event manager directory user repository**、**event manager update user policy**、 **event manager scheduler clear**、**event manager update user policy**、**event owner**、**event rpc**、**event snmp-notification**、**show event manager detector**、**show event manager version**、**trigger (EEM)**。
Embedded Event Manager 3.0	12.4(22)T 12.2(33)SRE 12.2(50)SY	EEM 3.0 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが導入または変更されました。 **action add** 、**action append**、**action break**、**action comment**、**action context retrieve**、**action context save**、**action continue**、**action decrement**、**action divide**、**action else**、**action elseif**、**action end**、**action exit**、**action foreach**、**action gets**、**action if**, **action if goto**、**action increment**、**action info type interface-names**、**action info type snmp getid**、**action info type snmp inform**、**action info type snmp oid**、**action info type snmp trap**、**action info type snmp var**、**action multiply**、**action puts**、**action regexp**、**action set (EEM)**、**action string compare**、**action string equal**、**action string first**、**action string index**、**action string last**、**action string length**、**action string match**、**action string range**、**action string replace**、**action string tolower**、**action string toupper**、**action string trim**、**action string trimleft**、**action string trimright**、**action subtract**、**action while**、**event cli**、**event ipsla**、**event manager detector routing**、**event manager scheduler**、**event manager scheduler clear**、**event manager scheduler hold**、**event manager scheduler modify**、**event manager scheduler release**、**event nf**、**event routing**、**show event manager policy active**、**show event manager policy pending**、および **show event manager scheduler**。
Embedded Event Manager 3.1	15.0(1)M 15.1(1)SY 15.1(2)SY	EEM 3.1 で、いくつかの新機能が導入されました。 この機能により、次のコマンドが導入または変更されました。**action syslog**、**description (EEM)**、**event manager applet**、**event manager policy**、**event snmp-notification**、**event snmp-object**、**show event manager policy registered**、および **show event manager policy available** 。

機能名	リリース	機能情報
Embedded Event Manager 3.2	12.2(52)SE 12.2(54)SG 15.1(3)T 15.1(1)SY 15.1(2)SY	EEMは、イベント検出と回復をCisco IOS内部で直接行うための分散型でカスタマイズされた手法です。 この機能に関する詳細については、次の各項を参照してください。 次のコマンドが導入または変更されました。**debug event manager**、**event identity**、**event mat**、**event neighbor-discovery**、**show event manager detector**。
Embedded Event Manager 4.0	15.2(2)T 15.1(1)SY 15.1(2)SY 12.2(2)E	EEM 4.0 で、いくつかの新機能が導入されました。 次のコマンドが導入または変更されました。**action file**、**action mail**、**action syslog**、**clear event manager detector counters**、**clear event manager server counters**、**event cli**、**event manager policy**、**event manager scheduler**、**event syslog**、**show event manager detector**、**show event manager policy registered**、**show event manager statistics**。



CHAPTER 38

署名済み Tcl スクリプト

署名付き TCL スクリプト機能を使用すると、デジタル署名を生成する証明書を作成し、そのデジタル署名を使用してツールコマンド言語（TCL）スクリプトに署名することが可能になります。この機能は、既存のスクリプトおよび証明書でも動作します。デジタル署名の認証が確認されてから、Tcl インタープリタへの信頼できるアクセスでスクリプトが実行されます。スクリプトにデジタル署名がない場合、そのスクリプトは信頼できないスクリプト用の限定モードで実行されるか、まったく実行されません。

署名済み Tcl スクリプトに関する前提条件

この機能が動作するには、Cisco Public Key Infrastructure（PKI）設定のトラストポイント コマンドを有効にする必要があります。

署名付き TCL スクリプトの制約事項

この機能が動作するには、次を実行している必要があります。

- Cisco IOS 暗号イメージ
- OpenSSL Version 0.9.7a 以降
- Expect

署名済み Tcl スクリプトについて

署名済み Tcl スクリプト機能は Tcl スクリプトにセキュリティを導入します。この機能を使用すると、デジタル署名を生成する証明書を作成し、そのデジタル署名を使用して Tcl スクリプトに署名することが可能になります。この証明書は、Tcl スクリプトを実行する前にそれらを検査します。スクリプトに Cisco 発行のデジタル証明書が含まれているかどうかを確認します。さらに、第三者がデジタル署名でスクリプトに署名することもできます。独自に社内で開発した TCL スクリプトに署名したい場合や、サード パーティ製が開発したスクリプトを使用したい場合もあります。スクリプトに正しいデジタル署名が含まれている場合は本物であると見なされ、Tcl インタープリタにフル アクセスで実行されます。スクリプトにデジタル署名がない場合、そのスクリプトはセーフ Tcl モードという限定されたモードで実行されるか、またはまったく実行されません。

署名付き Tcl スクリプトを作成し、使用するには、次の概念を理解する必要があります。

Cisco PKI

Cisco PKI を使用すると、IP セキュリティ（IPSec）、セキュア シェル（SSH）、セキュア ソケット レイヤ（SSL）などのセキュリティ プロトコルをサポートする証明書管理を実現できます。PKI は以下のエンティティで構成されています。

- セキュアなネットワークで通信する複数のピア
- 証明書を発行および維持する認証局（CA）を最低 1 つ
- デジタル証明書（証明書の有効期間、ピアの ID 情報、セキュアな通信に使用する暗号キー、CA 発行のシグニチャなどで構成）
- 登録要求を処理し CA の負荷を軽減する登録局（RA）（任意）
- 証明書失効リスト（CRL）を配信するメカニズム（Lightweight Directory Access Protocol（LDAP）、HTTP など）

PKI を使用すると、セキュアなデータ ネットワークで暗号化情報と ID 情報を配信、管理、失効するためのスケーラブルでセキュアなメカニズムを実現できます。セキュアな通信に関係するルーティング デバイスはすべて、あるプロセスを経て PKI に登録されます。そのプロセスでは、ルーティング デバイスが Rivest, Shamir, and Adelman（RSA）キー ペア（秘密キーが 1 つ、公開キーが 1 つ）を生成し、信頼されているルーティング デバイス（CA またはトラストポイントともいいます）でキーの ID を確認します。

各ルーティング デバイスが PKI に登録されると、PKI のすべてのピア（エンド ホストともいいます）は、CA が発行したデジタル証明書を付与されます。セキュアな通信セッションをネゴシエーションする必要があるときは、ピアはデジタル証明書を交換します。ピアは証明書内の情報を基に他のピアの ID を確認し、証明書内の公開キーを使って、暗号化されたセッションを確立します。

RSA キーペア

RSA キー ペアは、公開キーと秘密キーで構成されます。PKI を設定する場合、証明書登録要求に公開キーを含める必要があります。証明書が付与された後、ピアが公開キーを使用して、デバイスに送信されるデータを暗号化できるように、公開キーが証明書に組み込まれます。秘密キーはデバイスに保持され、ピアによって送信されたデータの復号化と、ピアとネゴシエーションするときの、トランザクションのデジタル署名に使用されます。

RSA キーペアには、キーのモジュラス値が含まれています。モジュラス値に応じて、RSA キーのサイズが決まります。モジュラス値が大きいほど、RSA キーの安全性が高まります。ただし、モジュラス値が大きくなると、キーの生成にかかる時間が長くなり、キーのサイズが大きくなると暗号化処理および復号化処理にかかる時間が長くなります。

証明書およびトラストポイント

認証局（CA。トラストポイントともいいます）は、証明書要求を管理し、参加ネットワークデバイスに証明書を発行します。証明書要求の管理や証明書発行などのサービスにより、参加デバイスを一元的に管理します。またこれらのサービスによって受信者は、明示的に信頼してアイデンティティを確認し、デジタル証明書を作成できます。PKI の動作を開始する前に、CA は独自の公開キー ペアを生成し、自己署名 CA 証明書を作成します。その後、CA は、証明書要求に署名し、PKI に対してピア登録を開始できます。

CA は、サードパーティの CA ベンダーが提供する CA を使用するか、内部の CA、つまり Cisco 証明書サーバーを使用します。

署名済み Tcl スクリプトの設定方法

キー ペアの生成

キーペアは、秘密キーと公開キーで構成されます。秘密キーは公開されず、作成者のみがアクセス可能にすることを意図しています。公開キーは秘密キーから生成され、公開されることを前提としています。

キーペアを生成するには、**openssl genrsa** コマンドを使用した後、**openssl rsa** コマンドを使用します。

SUMMARY STEPS

1. **openssl genrsa -out** *private-key-file bit-length*
2. **ls -l**
3. **openssl rsa -in** *private-key-file* **-pubout -out** *public-key-file*
4. **ls -l**

DETAILED STEPS

ステップ 1 **openssl genrsa -out** *private-key-file bit-length*

このコマンドは、*bit-length* ビット長の秘密キーを生成し、そのキーを *private-key-file* ファイルに書き込みます。

```
Host% openssl genrsa -out privkey.pem 2048
```

Example:

```
Generating RSA private key, 2048 bit long modulus
.........+++
.....................................................................+++
e is 65537 (0x10001)
```

ステップ 2 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 8
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
```

privkey.pem ファイルには、**openssl genrsa** コマンドを使用して生成した秘密キーが含まれています。

ステップ 3 **openssl rsa -in** *private-key-file* **-pubout -out** *public-key-file*

このコマンドは、*private-key-file* ファイル内の指定された秘密キーに基づいて公開キーを作成し、その公開キーを *public-key-file* ファイルに書き込みます。

Example:

```
Host% openssl rsa -in privkey.pem -pubout -out pubkey.pem

writing RSA key
```

ステップ 4 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 16
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

pubkey.pem ファイルには、**openssl rsa** コマンドを使用して生成された公開キーが含まれます。

証明書の生成

証明書を生成するには、次のタスクを実行します。X.509 証明書を生成するには、**openssl req** コマンドを使用します。

SUMMARY STEPS

1. **openssl req -new -x509 -key** *private-key-file* **-out** *certificate-file* **-days** *expiration-days*
2. **ls -l**

DETAILED STEPS

ステップ 1 **openssl req -new -x509 -key** *private-key-file* **-out** *certificate-file* **-days** *expiration-days*

このコマンドは、*private-key-file* ファイルに保存された秘密キーにフルアクセスできる X.509 証明書を作成し、*certificate-file* ファイルに証明書を保存します。証明書は *expiration-days* 日内に期限が切れるように設定されます。

コマンドを実行するには、プロンプトが表示された時点で次の識別名（DN）情報を入力します。

- 国名
- 州、行政区分（都道府県）名
- 組織名
- 組織部署名
- 共通名
- メール アドレス

各プロンプトの角括弧で囲まれたテキストは、Enter を押す前に値を入力しなかった場合に使用されるデフォルト値を示します。

次に、privkey.pem ファイル内の秘密キーに対するフルアクセスを持つ X.509 証明書を生成する方法の例を示します。証明書は cert.pem ファイルに書き込まれ、生成日の 1095 日後に期限切れになります。

Example:

```
Host% openssl req -new -x509 -key privkey.pem -out cert.pem -days 1095

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value, If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:US
```

```
State or Province Name (full name) [Berkshire]:California

Locality Name (eg, city) [Newbury]:San Jose

Organization Name (eg, company) [My Company Ltd]:Cisco Systems, Inc.

Organizational Unit Name (eg, section) []:DEPT_ACCT

Common Name (eg, your name or your server's hostname) []:Jane

Email Address []:janedoe@company.com
```

ステップ 2 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 24
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

cert.pem ファイルには、**openssl req** コマンドを使用して作成された X.509 証明書が含まれています。

Tcl スクリプトの署名

Tcl スクリプトに署名するには、次のタスクを実行します。TCL ファイル、および OpenSSL ドキュメントの出力に、pkcs7（PKCS#7）フォーマットで署名する必要があります。

Tcl ファイルに署名するには、**openssl smime** コマンドと **-sign** キーワードを使用します。

SUMMARY STEPS

1. **openssl smime -sign -in** *tcl-file* **-out** *signed-tcl-file* **-signer** *certificate-file* **-inkey** *private-key-file* **-outform DER -binary**
2. **ls -l**

DETAILED STEPS

ステップ 1 **openssl smime -sign -in** *tcl-file* **-out** *signed-tcl-file* **-signer** *certificate-file* **-inkey** *private-key-file* **-outform DER -binary**

このコマンドは、*certificate-file* に保存されている証明書と、*private-key-file* に保存されている秘密キーを使用して Tcl ファイル名 *tcl-file* に署名し、署名済みの Tcl ファイルを *signed-tcl-file* ファイルに DER PKCS#7 形式で書き込みます。

Example:

```
Host% openssl smime -sign -in hello -out hello.pk7 -signer cert.pem -inkey privkey.pem -outform DER
 -binary
```

ステップ 2 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

hello.pk7 ファイルには、hello という名前の未署名の TCL ファイルから **openssl smime** コマンドと cert.pem ファイル内の X.509 証明書を使用して作成された、署名済み Tcl ファイルが含まれています。

署名の確認

署名がデータと一致していることを確認するには、**openssl smime** コマンドと **-verify** キーワードを使用して次のタスクを実行します。Tcl の元の内容を入力ファイルに提供する必要があります。これは、ファイルに元の内容が含まれていないためです。

SUMMARY STEPS

1. **openssl smime -verify -in** *signed-tcl-file* **-CAfile** *certificate-file* **-inform DER -content** *tcl-file*
2. **ls -l**

DETAILED STEPS

ステップ 1 **openssl smime -verify -in** *signed-tcl-file* **-CAfile** *certificate-file* **-inform DER -content** *tcl-file*

このコマンドは、*certificate-file* 内の信頼認証局（CA）証明書を使用して DER PKCS#7 形式で *signed-tcl-file* に保存されている署名付き Tcl ファイルを確認した後、デタッチされた内容を *tcl-file* ファイルに書き込みます。

次に、入力ファイルの hello.pk7 を使用して署名を確認する例を示します。

Example:

```
Host% openssl smime -verify -in hello.pk7 -CAfile cert.pem -inform DER -content hello

puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
```

```
puts "tcl_interactive = $tcl_interactive"
Verification successful
```

Note SSL コマンドページでは、**-in** *filename* が暗号化または署名される入力メッセージか、復号または確認される MIME メッセージとして説明されています。詳細については、http://www.openssl.org/ を参照してください。

ステップ 2 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 13 10:18 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:17 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

hello ファイルには、**openssl smime** コマンドを **-verify** キーワードで実行して、署名付き Tcl ファイル hello.pk7 からデタッチされた内容が含まれています。確認に成功した場合、署名者の証明書が cert.pem ファイルの X.509 証明書に書き込まれます。

シグニチャの非バイナリデータへの変換

バイナリから非バイナリ データにシグニチャを変換するには、次のタスクを実行します。

SUMMARY STEPS

1. **xxd -ps** *signed-tcl-file* > *nonbinary-signature-file*
2. **#Cisco Tcl Signature V1.0** を最初の行に表示するスクリプトを作成し、コメント文字（#）を入力ファイルの各行の先頭に挿入し、入力ファイルの名前にテキスト文字列「_sig」を追加して形成された名前のファイルに各行を書き込みます。
3. 非バイナリ シグニチャ ファイルを含むファイルの名前（*nonbinary-signature-file*）を入力引数として指定して、スクリプトを実行します。
4. **ls -l**
5. **cat** *signed-tcl-file* *commented-nonbinary-signature-file* > *signed-tcl-script*
6. **cat** *signed-tcl-script*

DETAILED STEPS

ステップ 1 **xxd -ps** *signed-tcl-file* > *nonbinary-signature-file*

このコマンドは、*signed-tcl-file* のシグニチャをバイナリから非バイナリのデータに変換して *nonbinary-signature-file* ファイルに 16 進ダンプとして保存します。

Example:

```
Host% xxd -ps hello.pk7 > hello.hex
```

ステップ 2 **#Cisco Tcl Signature V1.0** を最初の行に表示するスクリプトを作成し、コメント文字（#）を入力ファイルの各行の先頭に挿入し、入力ファイルの名前にテキスト文字列「_sig」を追加して形成された名前のファイルに各行を書き込みます。

次に、**cat** コマンドを使用して、my_append という名前のスクリプトファイルの内容を表示する例を示します。

Example:

```
Host% cat my_append

#!/usr/bin/env expect
set my_first {#Cisco Tcl Signature V1.0}
set newline {}
set my_file [lindex $argv 0]
set my_new_file ${my_file}_sig
set my_new_handle [open $my_new_file w]
set my_handle [open $my_file r]
puts $my_new_handle $newline
puts $my_new_handle $my_first
foreach line [split [read $my_handle] "\n"]  {
   set new_line {#}
   append new_line $line
   puts $my_new_handle $new_line
}

close $my_new_handle
close $my_handle
```

ステップ 3 非バイナリ シグニチャ ファイルを含むファイルの名前（*nonbinary-signature-file*）を入力引数として指定して、スクリプトを実行します。

この例では、my_append スクリプトが、入力として指定された非バイナリ シグニチャ ファイル hello.hex を使用して実行されています。出力ファイルには、hello.hex_sig という名前が付けられます。

Example:

```
Host% my_append hello.hex
```

ステップ 4 **ls -l**

このコマンドは、現在のディレクトリ内の各ファイルに対する詳細な情報（権限、所有者、サイズ、最終変更日時を含む）を表示します。

Example:

```
Host% ls -l

total 80
-rw-r--r--   1 janedoe eng12       1659 Jun 13 10:18 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:17 hello
-rw-r--r--   1 janedoe eng12       3815 Jun 13 10:20 hello.hex
-rw-r--r--   1 janedoe eng12       3907 Jun 13 10:22 hello.hex_sig
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rwxr--r--   1 janedoe eng12        444 Jun 13 10:22 my_append
```

```
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

hello.hex ファイルには、署名済み Tcl ファイル hello.pk7 のバイナリ シグニチャから変換された非バイナリ データ（16 進数のダンプとして格納）が含まれています。my_append ファイルには、入力ファイルの各行の先頭にコメント文字を挿入するスクリプトが含まれています。この hello.hex_sig ファイルは、非バイナリ シグニチャ ファイルで my_append スクリプトを実行して作成されたファイルです。

ステップ 5 **cat** *signed-tcl-file commented-nonbinary-signature-file* > *signed-tcl-script*

このコマンドは非バイナリ シグニチャ ファイル（*commented-nonbinary-signature-file*）の内容を、DER PKCS#7 形式で保存された署名済みの Tcl ファイル（*signed-tcl-file* ファイル）に追加します。連結された出力が *signed-tcl-script* ファイルに書き込まれます。

Example:

```
Host% cat hello hello.hex_sig > hello.tcl
```

ステップ 6 **cat** *signed-tcl-script*

このコマンドは、署名済み Tcl ファイルと非バイナリ シグニチャ ファイルから分離された内容を連結した *signed-tcl-script* ファイルの内容を表示します。

Example:

```
Host% cat hello.tcl

puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
#Cisco Tcl Signature V1.0
#3082075006092a864886f70d010702a08207413082073d020101310b3009
#06052b0e03021a0500300b06092a864886f70d010701a08204a13082049d
#30820385a003020102020100300d06092a864886f70d0101040500308195
#310b300906035504061302555331133011060355040813 0a43616c69666f
#726e69613111300f060355040713 0853616e204a6f7365311c301a060355
#040a1313436973636f2053797374656d732c20496e632e310e300c060355
#040b13054e53535447310d300b060355040313044a6f686e3121301f0609
#2a864886f70d01090116126a6c6175746d616e40636973636f2e636f6d30
#1e170d3037303631323232303134335a170d3130303631313232303133
#5a308195310b300906035504061302555331133011060355040813 0a4361
#6c69666f726e69613111300f060355040713 0853616e204a6f7365311c30
#1a060355040a1313436973636f2053797374656d732c20496e632e310e30
#0c060355040b13054e53535447310d300b060355040313044a6f686e3121
#301f06092a864886f70d01090116126a6c6175746d616e40636973636f2e
#636f6d30820122300d06092a864886f70d0101010500 0382010f00308201
#0a0282010100a751eb5ec1f3009738c88a55987c07b759c36f3386342283
#67ea20a89d9483ae85e0c63eeded8ab3eb7a08006689f09136f172183665
#c971099ba54e77ab47706069bbefaaab8c50184396350e4cc870c4c3f477
#88c55c52e2cf411f05b59f0eaec0678ff5cc238fdce2263a9fc6b6c244b8
#ffaead865c19c3d3172674a13b24c8f2c01dd8b1bd491c13e84e29171b85
#f28155d81ac8c69bb25ca23c2921d85fbf745c106e7aff93c72316cbc654
#4a34ea88174a8ba7777fa60662974e1fbac85a0f0aeac925dba6e5e850b8
#7caffce2fe8bb04b61b62f532b5893c081522d538005df81670b931b0ad0
#e1e76ae648f598a9442d5d0976e67c8d55889299147d0203010001a381f5
#3081f2301d0603551d0e04160414bc34132be952ff8b9e1af3b93140a255
#e54a667c3081c20603551d230481ba3081b78014bc34132be952ff8b9e1a
#f3b93140a255e54a667ca1819ba48198308195310b300906035504061302
```

```
#5553311330110603550408130a43616c69666f726e69613111300f060355
#0407130853616e204a6f7365311c301a060355040a1313436973636f2053
#797374656d732c20496e632e310e300c060355040b13054e53535447310d
#300b060355040313044a6f686e3121301f06092a864886f70d0109011612
#6a6c6175746d616e40636973636f2e636f6d820100300c0603551d130405
#30030101ff300d06092a864886f70d010104050003820101000c83c1b074
#6720929c9514af6d5df96f0a95639f047c40a607c83d8362507c58fa7f84
#aa699ec5e5bef61b2308297a0662c653ff446acfbb6f5cb2dd162d939338
#a5e4d78a5c45021e5d4dbabb8784efbf50cab0f5125d164487b31f5cf933
#a9f68f82cd111cbab1739d7f372ec460a7946882874b0a0f22dd53acbd62
#a944a15e52e54a24341b3b8a820f23a5bc7ea7b2278bb56838b8a4051926
#af9c167274ff8449003a4e012bcf4f4b3e280f85209249a390d14df47435
#35efabce720ea3d56803a84a2163db4478ae19d7d987ef6971c8312e280a
#aac0217d4fe620c6582a48faa8ea5e3726a99012e1d55f8d61b066381f77
#4158d144a43fb536c77d6a3182027730820273020101308190b308195310b
#3009060355040613025553311330110603550408130a43616c69666f726e
#69613111300f060355040713085616e204a6f7365311c301a060355040a
#1313436973636f2053797374656d732c20496e632e310e300c060355040b
#13054e53535447310d300b060355040313044a6f686e3121301f06092a86
#4886f70d01090116126a6c6175746d616e40636973636f2e636f6d020100
#300906052b0e03021a0500a081b1301806092a864886f70d010903310b06
#092a864886f70d010701301c06092a864886f70d010905310f170d303730
#3631333137313634385a302306092a864886f70d01090431160414372cb3
#72dc607990577fd0426104a42ee4158d2b305206092a864886f70d01090f
#31453043300a06082a864886f70d0307300e06082a864886f70d03020202
#0080300d06082a864886f70d030202014030070605 2b0e030207300d0608
#2a864886f70d0302020128300d06092a864886f70d0101010500048201 00
#72db6898742f449b26d3ac18f43a1e7178834fb05ad13951bf042e127eea
#944b72b96f3b8ecf7eb52f3d0e383bf63651750223efe69eae04287c9dae
#b1f31209444108b31d34e46654c6c3cc10b5baba887825c224ec6f376d49
#00ff7ab2d9f88402dab9a2c2ab6aa3ecceeaf5a594bdc7d3a822c55e7daa
#aa0c2b067e06967f22a20e406fe21d9013ecc6bd9cd6d402c2749f8bea61
#9f8f87acfbc9e10d6ce91502e34629adca6ee855419afafe6a8233333e14
#ad4c107901d1f2bca4d7ffaadddbc54192a25da662f8b8509782c76977b8
#94879453fbb00486ccc55f88db50fcc149bae066916b350089cde51a6483
#2ec14019611720fc5bbe2400f24225fc
```

証明書を使用したデバイスの設定

証明書を使用してデバイスを設定するには、次のタスクを実行します。

Before you begin

すでに、Cisco IOS 暗号化イメージが用意されている必要があります。用意されていない場合は、証明書を設定できません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal**
5. **exit**
6. **crypto pki authenticate** *name*
7. プロンプトで、ベースが暗号化された CA 証明書を入力します。

8. **scripting tcl secure-mode**
9. **scripting tcl trustpoint name** *name*
10. **scripting tcl trustpoint untrusted** {**execute** | **safe-execute** | **terminate**}
11. **exit**
12. **tclsafe**

DETAILED STEPS

ステップ 1 **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）。

Example:

```
Device> enable
```

ステップ 2 **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。

Example:

```
Device# configure terminal
```

ステップ 3 **crypto pki trustpoint** *name*

デバイスが認証局（CA）*mytrust* を使用して、CA トラストポイント コンフィギュレーション モードを開始することを宣言します。

Example:

```
Device(config)# crypto pki trustpoint mytrust
```

ステップ 4 **enrollment terminal**

カットアンドペーストによる手動での証明書登録を指定します。このコマンドが有効になると、デバイスはコンソール端末に証明書要求を表示します。これにより、このターミナルに発行済みの証明証が入力できるようになります。

Example:

```
Device(ca-trustpoint)# enrollment terminal
```

ステップ 5 **exit**

CA トラストポイント コンフィギュレーション モードを終了し、グローバル コンフィギュレーション モードに戻ります。

Example:

```
Device(ca-trustpoint)# exit
```

ステップ 6 **crypto pki authenticate** *name*

CA 証明書を取得して、認証します。証明書フィンガープリントをチェックするよう求められた場合、証明書フィンガープリントをチェックします。

Note CA では、証明書が自己署名されるため、このコマンドを実行するときは、CA 管理者に問い合わせて、CA の公開キーを手動で認証する必要があります。

Example:

```
Device(config)# crypto pki authenticate mytrust
```

ステップ 7 プロンプトで、ベースが暗号化された CA 証明書を入力します。

Example:

```
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIEuDCCA6CgAwIBAgIBADANBgkqhkiG9w0BAQQFADCBnjELMAkGA1UEBhMCVVMx
EzARBgNVBAgTCkNhbGlmb3JuaWExETAPBgNVBAcTCFNhbiBKb3NlMRwwGgYDVQQK
ExNDaXNjbyBTeXN0ZW1zLCBJbmMuMQ4wDAYDVQQLEwVOU1NURzEWMBQGA1UEAxMN
Sm9obiBMYXV0bWFubjEhMB8GCSqGSIb3DQEJARYSamxhdXRtYW5AY2lzY28uY29t
MB4XDTA2MTExNzE3NTgwMVoXDTA5MTExNjE3NTgwMVowgZ4xCzAJBgNVBAYTAlVT
MRMwEQYDVQQIEwpDYWxpZm9ybmlhMREwDwYDVQQHEwhTYW4gSm9zZTEcMBoGA1UE
ChMTQ2lzY28gU3lzdGVtcywgSW5jLjEOMAwGA1UECxMFTlNTVEcxFjAUBgNVBAMT
DUpvaG4gTGF1dG1hbm4xITAfBgkqhkiG9w0BCQEWEmpsYXV0bWFuQGNpc2NvLmNv
bTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALxtqTMCirMb+CdyWLuH
oWAM8CEJDwQggL7MWBhoi3TSMd/ww2XBB9biBtdlH6jHsjCiOwAR5OorakwfPyf7
mvRJ2PqJALs+Vn93VBKIG6rZUl4+wdOx686BVddIZvEJQPbROiYTzfazWV70aLMV
bd7/B7vF1SG1YK9y1tX9p9nZyZ0x47OAXetwOaGinvlG7VNuTXaASBLUjCRZsIlz
SBrXXedBzZ6+BuoWm1FK45EYSlag5Rt9RGXXMBqzx91iyhrJ3zDDmkExa45yKJET
mAgDVMcpeteJtif47UDZJK30g4MbMyx/c8WGhmJ54qRL9BZEPmDxMQkNP1Ol8MAl
Q8sCAwEAAaOB/jCB+zAdBgNVHQ4EFgQU9/ToDvbMR3JfJ4xEa4X47oNFq5kwgcsG
A1UdIwSBwzCBwIAU9/ToDvbMR3JfJ4xEa4X47oNFq5mhgaSkgaEwgZ4xCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMREwDwYDVQQHEwhTYW4gSm9zZTEc
MBoGA1UEChMTQ2lzY28gU3lzdGVtcywgSW5jLjEOMAwGA1UECxMFTlNTVEcxFjAU
BgNVBAMTDUpvaG4gTGF1dG1hbm4xITAfBgkqhkiG9w0BCQEWEmpsYXV0bWFuQGNp
c2NvLmNvbYIBADAMBgNVHRMEBTADAQH/MA0GCSqGSIb3DQEBBAUAA4IBAQBtEs/4
MQeN9pT+XPCPg2ObQU8y2AadI+I34YK+fDHsFOh68hZhpszTN2VpNEvkFXpADhgr
7DkNGtwTCla481v70iNFViQVL+inNrZwWMxoTnUNCK7Hc5kHkXt6cj0mvsefVUzx
Xl70mauhESRVlmYWrJxSsrEILerZYsuv5HbFdand+/rErmP2HVyfdntLnKdSzmXJ
5lwE/Et2QtYNGor0OBlLesowfslR3LhHi4wn+5is7mALgNw/NuTiUr1zH18OeB4m
wcpBIJsLaJu6ZUJQl7IqdswSa3fHd5qq0/k8P9z0YAYrf3+MFQr4ibvsYvHlO087
o2Js1gW4qz34pqNh
Certificate has the following attributes:
       Fingerprint MD5: 1E327DBB 330936EB 2FB8EACB 4FD1133E
      Fingerprint SHA1: EE7FF9F4 05148842 B9D50FAC D76FDC9C E0703246
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

ステップ 8 **scripting tcl secure-mode**

インタラクティブ Tcl スクリプトのシグニチャ確認を有効にします。

```
Device(config)# scripting tcl secure-mode
```

ステップ 9 **scripting tcl trustpoint name** *name*

設定済みの既存のトラストポイント名と証明書を関連付け、Tcl スクリプトを確認します。

```
Device(config)# scripting tcl trustpoint name mytrust
```

ステップ 10 scripting tcl trustpoint untrusted {execute | safe-execute | terminate}

（任意）シグニチャ確認に失敗したか、信頼できないモードであるかにかかわらず、**execute**、**safe-execute** または **terminate** の 3 つのキーワードのいずれかを使用してインタラクティブ Tcl スクリプトを実行できます。

- **execute** ：シグニチャの確認に失敗しても、Tcl スクリプトを実行します。 **execute** キーワードを設定すると、シグニチャの確認は一切実行されません。

Note シグニチャの確認が実行されないため、通常、このキーワードの使用は推奨されません。

execut キーワードは、内部テスト用に提供されており、これにより柔軟性が向上します。たとえば、証明書の期限が切れていても、他の設定が有効であり、既存の設定で作業したい場合は、execute キーワードを使用して、期限の切れた証明書で対処することができます。

- **safe-execute** ：スクリプトをセーフ モードで実行できます。tclsafe コマンドを使用し、インタラクティブ Tcl シェル セーフ モードを開始すると、使用可能なセーフ モード Tcl コマンドを確認できます。この限定されたセーフ モードで何が使用できるかをより深く理解するには、tclsafe Exec コマンドを使用してオプションを確認します。
- **terminate** ：すべてのスクリプトの実行を停止し、デフォルトの動作に戻します。デフォルト ポリシーは終端します。最後のトラストポイント名が削除されると、信頼できないアクションも削除されます。信用できないアクションは、TCL 用に最低でも 1 つのトラストポイント名が設定されていない場合は開始されません。

次に、シグニチャ確認が失敗した場合に、**safe-execute** キーワードを使用して Tcl スクリプトをセーフモードで実行する例を示します。

```
Device(config)# scripting tcl trustpoint untrusted safe-execute
```

ステップ 11 exit

グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

```
Device(config)# exit
```

ステップ 12 tclsafe

（任意）インタラクティブ Tcl シェルの信頼できないセーフ モードを有効にします。これにより、Cisco コマンドライン インターフェイスから信頼できないセーフ モードで手動により Tcl コマンドを実行できるようになります。

```
Device# tclsafe
```

Example:

トラストポイントの確認

デバイス内に設定されているトラストポイントを表示するには、**show crypto pki trustpoints** コマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **show crypto pki trustpoints**

DETAILED STEPS

ステップ 1 **enable**

このコマンドでは、特権 EXEC モードをイネーブルにします。

Example:

```
Device> enable
```

ステップ 2 **show crypto pki trustpoints**

このコマンドは、デバイスに設定されているトラストポイントを表示します。

Example:

```
Device# show
 crypto pki trustpoints

Trustpoint mytrust:
    Subject Name:
    ea=janedoe@cisco.com
    cn=Jane
    ou=DEPT_ACCT
    o=Cisco
    l=San Jose
    st=California
    c=US
          Serial Number: 00
    Certificate configured.
```

署名済み Tcl スクリプトの確認

署名済み Tcl スクリプトが正しく実行していることを確認するには、**debug crypto pki transactions** コマンドと **tclsh** コマンドを使用します。

SUMMARY STEPS

1. **enable**
2. **debug crypto pki transactions**
3. **tclsh** *flash:signed-tcl-file*

DETAILED STEPS

ステップ 1 enable

このコマンドでは、特権 EXEC モードをイネーブルにします。

Example:

```
Device> enable
```

ステップ 2 debug crypto pki transactions

このコマンドは、CA とデバイス間のやり取りのトレース（メッセージ タイプ）のデバッグ メッセージを表示します。

Example:

```
Device# debug crypto pki transactions

Crypto PKI Trans debugging is on
```

ステップ 3 tclsh *flash:signed-tcl-file*

このコマンドは、Tcl シェルで Tcl スクリプトを実行します。

Note ファイルは、署名付きの Tcl ファイルである必要があります。

Example:

```
Device# tclsh flash:hello.tcl

hello
argc = 0
argv =
argv0 = flash:hello.tcl
tcl_interactive = 0
device#
*Apr 21 04:46:18.563: CRYPTO_PKI: locked trustpoint mytrust, refcount is 1
*Apr 21 04:46:18.563: The PKCS #7 message has 0 verified signers.
*Apr 21 04:46:18.563: CRYPTO_PKI: Success on PKCS7 verify!
*Apr 21 04:46:18.563: CRYPTO_PKI: unlocked trustpoint mytrust, refcount is 0
```

次の作業

- 暗号の概要については、『*Security Configuration Guide*』の「Part 5: Implementing and Managing a PKI」の項を参照してください。

署名済み Tcl スクリプトの設定例

キー ペアの生成の例

次に、キー ペア（秘密キーと公開キー）を生成する方法の例を示します。

秘密キーの生成：例

```
Host% openssl genrsa -out privkey.pem 2048
Generating RSA private key, 2048 bit long modulus
.........+++
.....................................................................................+++
e is 65537 (0x10001)
Host% ls -l
total 8
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
Host%
```

秘密キーからの公開キーの生成

```
Host% openssl rsa -in privkey.pem -pubout -out pubkey.pem
writing RSA key
Host% ls -l
total 16
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

証明書の生成の例

次に、証明書を生成する例を示します。

```
Host% openssl req -new -x509 -key privkey.pem -out cert.pem -days 1095
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value, If you enter '.', the field will be left
 blank.
-----
Country Name (2 letter code) [GB]:US
State or Province Name (full name) [Berkshire]:California
Locality Name (eg, city) [Newbury]:San Jose
Organization Name (eg, company) [My Company Ltd]:Cisco Systems, Inc.
Organizational Unit Name (eg, section) []:DEPT_ACCT
Common Name (eg, your name or your server's hostname) []:Jane
Email Address []:janedoe@company.com
Host% ls -l
total 24
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

Tcl スクリプトの署名の例

次に、Tcl スクリプトに署名する例を示します。

```
Host% openssl smime -sign -in hello -out hello.pk7 -signer cert.pem -inkey privkey.pem
-outform DER -binary
Host% ls -l
total 40
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
```

署名の確認の例

次に、署名を確認する例を示します。

```
Host% openssl smime -verify -in hello.pk7 -CAfile cert.pem -inform DER -content hello
puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
Verification successful
```

非バイナリデータを使用した署名の変換の例

次に、TCL シグニチャを非バイナリ データに変換する方法の例を示します。

```
#Cisco Tcl Signature V1.0
Then append the signature file to the end of the file.
Host% xxd -ps hello.pk7 > hello.hex
Host% cat my_append
#!/usr/bin/env expect
set my_first {#Cisco Tcl Signature V1.0}
set newline {}
set my_file [lindex $argv 0]
set my_new_file ${my_file}_sig
set my_new_handle [open $my_new_file w]
set my_handle [open $my_file r]

puts $my_new_handle $newline
puts $my_new_handle $my_first
foreach line [split [read $my_handle] "\n"]  {
   set new_line {#}
   append new_line $line
   puts $my_new_handle $new_line
}

close $my_new_handle
close $my_handle
Host% my_append hello.hex
Host% ls -l
total 80
-rw-r--r--   1 janedoe eng12       1659 Jun 12 15:01 cert.pem
-rw-r--r--   1 janedoe eng12        115 Jun 13 10:16 hello
```

```
-rw-r--r--   1 janedoe eng12       3815 Jun 13 10:20 hello.hex
-rw-r--r--   1 janedoe eng12       3907 Jun 13 10:22 hello.hex_sig
-rw-r--r--   1 janedoe eng12       1876 Jun 13 10:16 hello.pk7
-rwxr--r--   1 janedoe eng12        444 Jun 13 10:22 my_append
-rw-r--r--   1 janedoe eng12       1679 Jun 12 14:55 privkey.pem
-rw-r--r--   1 janedoe eng12        451 Jun 12 14:57 pubkey.pem
Host% cat hello hello.hex_sig > hello.tcl
Host% cat hello.tcl
puts hello
puts "argc = $argc"
puts "argv = $argv"
puts "argv0 = $argv0"
puts "tcl_interactive = $tcl_interactive"
#Cisco Tcl Signature V1.0
#3082075006092a864886f70d010702a08207413082073d020101310b3009
#06052b0e03021a0500300b06092a864886f70d010701a08204a13082049d
#30820385a003020102020100300d06092a864886f70d01010405003081 95
#310b300906035504061302555331133011060355040813 0a43616c69666f
#726e69613111300f060355040713 0853616e204a6f7365311c301a060355
#040a1313436973636f2053797374656d732c20496e632e310e300c060355
#040b13054e53535447310d300b060355040313044a6f686e3121301f0609
#2a864886f70d01090116126a6c6175746d616e40636973636f2e636f6d30
#1e170d3037303631323232303134335a170d3130303631313232303313433
#5a308195310b3009060355040613025553311330110603550408130a4361
#6c69666f726e69613111300f060355040713 0853616e204a6f7365311c30
#1a060355040a1313436973636f2053797374656d732c20496e632e310e30
#0c060355040b13054e53535447310d300b060355040313044a6f686e3121
#301f06092a864886f70d01090116126a6c6175746d616e40636973636f2e
#636f6d30820122300d06092a864886f70d01010105000382010f00308201
#0a0282010100a751eb5ec1f3009738c88a55987c07b759c36f3386342283
#67ea20a89d9483ae85e0c63eeded8ab3eb7a08006689f09136f172183665
#c971099ba54e77ab47706069bbefaaab8c50184396350e4cc870c4c3f477
#88c55c52e2cf411f05b59f0eaec0678ff5cc238fdce2263a9fc6b6c244b8
#ffaead865c19c3d3172674a13b24c8f2c01dd8b1bd491c13e84e29171b85
#f28155d81ac8c69bb25ca23c2921d85fbf745c106e7aff93c72316cbc654
#4a34ea88174a8ba7777fa60662974e1fbac85a0f0aeac925dba6e5e850b8
#7caffce2fe8bb04b61b62f532b5893c081522d538005df81670b931b0ad0
#e1e76ae648f598a9442d5d0976e67c8d55889299147d0203010001a381f5
#3081f2301d0603551d0e04160414bc34132be952ff8b9e1af3b93140a255
#e54a667c3081c20603551d230481ba3081b78014bc34132be952ff8b9e1a
#f3b93140a255e54a667ca1819ba48198308195310b300906035504061302
#5553113300110603550408130a43616c69666f726e69613111300f060355
#0407130853616e204a6f7365311c301a060355040a1313436973636f2053
#797374656d732c20496e632e310e300c060355040b13054e53535447310d
#300b060355040313044a6f686e3121301f06092a864886f70d0109011612
#6a6c6175746d616e40636973636f2e636f6d820100300c0603551d130405
#30030101ff300d06092a864886f70d01010405000382010100c83c1b074
#6720929c9514af6d5df96f0a95639f047c40a607c83d8362507c58fa7f84
#aa699ec5e5bef61b2308297a0662c653ff446acfbb6f5cb2dd162d939338
#a5e4d78a5c45021e5d4dbabb8784efbf50cab0f5125d164487b31f5cf933
#a9f68f82cd111cbab1739d7f372ec460a7946882874b0a0f22dd53acbd62
#a944a15e52e54a24341b3b8a820f23a5bc7ea7b2278bb56838b8a4051926
#af9c167274ff8449003a4e012bcf4f4b3e280f85209249a390d14df47435
#35efabce720ea3d56803a84a2163db4478ae19d7d987ef6971c8312e280a
#aac0217d4fe620c6582a48faa8ea5e3726a99012e1d55f8d61b066381f77
#4158d144a43fb536c77d6a3182027730820273020101 30819b308195310b
#300906035504061302555331133011060355040813 0a43616c69666f726e
#69613111300f060355040713 0853616e204a6f7365311c301a060355040a
#1313436973636f2053797374656d732c20496e632e310e300c060355040b
#13054e53535447310d300b060355040313044a6f686e3121301f06092a86
#4886f70d01090116126a6c6175746d616e40636973636f2e636f6d020100
#300906052b0e03021a0500a081b1301806092a864886f70d010903310b06
#092a864886f70d010701301c06092a864886f70d010905310f170d303730
#3631333137313634385a302306092a864886f70d01090431160414372cb3
```

```
#72dc607990577fd0426104a42ee4158d2b305206092a864886f70d01090f
#31453043300a06082a864886f70d0307300e06082a864886f70d03020202
#0080300d06082a864886f70d030202014030070605 2b0e030207300d0608
#2a864886f70d0302020128300d06092a864886f70d010101050004820100
#72db6898742f449b26d3ac18f43a1e7178834fb05ad13951bf042e127eea
#944b72b96f3b8ecf7eb52f3d0e383bf63651750223efe69eae04287c9dae
#b1f31209444108b31d34e46654c6c3cc10b5baba887825c224ec6f376d49
#00ff7ab2d9f88402dab9a2c2ab6aa3ecceeaf5a594bdc7d3a822c55e7daa
#aa0c2b067e06967f22a20e406fe21d9013ecc6bd9cd6d402c2749f8bea61
#9f8f87acfbc9e10d6ce91502e34629adca6ee855419afafe6a8233333e14
#ad4c107901d1f2bca4d7ffaadddbc54192a25da662f8b8509782c76977b8
#94879453fbb00486ccc55f88db50fcc149bae066916b350089cde51a6483
#2ec14019611720fc5bbe2400f24225fc
```

証明書を使用したデバイスの設定の例

次に、証明書でデバイスを設定する例を示します。

```
crypto pki trustpoint mytrust
 enrollment terminal
!
!
crypto pki authentication mytrust
crypto pki certificate chain mytrust
 certificate ca 00
  308204B8 308203A0 A0030201 02020100 300D0609 2A864886 F70D0101 04050030
  819E310B 30090603 55040613 02555331 13301106 03550408 130A4361 6C69666F
  726E6961 3111300F 06035504 07130853 616E204A 6F736531 1C301A06 0355040A
  13134369 73636F20 53797374 656D732C 20496E63 2E310E30 0C060355 040B1305
  4E535354 47311630 14060355 0403130D 4A6F686E 204C6175 746D616E 6E312130
  1F06092A 864886F7 0D010901 16126A6C 6175746D 616E4063 6973636F 2E636F6D
  301E170D 30363131 31373137 35383031 5A170D30 39313131 36313735 3830315A
  30819E31 0B300906 03550406 13025553 31133011 06035504 08130A43 616C6966
  6F726E69 61311130 0F060355 04071308 53616E20 4A6F7365 311C301A 06035504
  0A131343 6973636F 20537973 74656D73 2C20496E 632E310E 300C0603 55040B13
  054E5353 54473116 30140603 55040313 0D4A6F68 6E204C61 75746D61 6E6E3121
  301F0609 2A864886 F70D0109 0116126A 6C617574 6D616E40 63697363 6F2E636F
  6D308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201 0A028201
  0100BC6D A933028A B31BF827 7258BB87 A1600CF0 21090F04 2080BECC 5818688B
  74D231DF F0C365C1 07D6E206 D7651FA8 C7B230A2 3B0011E4 EA2B6A4C 1F3F27FB
  9AF449D8 FA8900BB 3E567F77 5412881B AAD9525E 3EC1D3B1 EBCE8155 D74866F1
  0940F6D1 3A2613CD F6B3595E F468B315 6DDEFF07 BBC5D521 B560AF72 D6D5FDA7
  D9D9C99D 31E3B380 5DEB7039 A1A29EF9 46ED536E 4D768048 12D48C24 59B08973
  481AD75D E741CD9E BE06EA16 9B514AE3 91184A56 A0E51B7D 4465D730 1AB3C7DD
  62CA1AC9 DF30C39A 41316B8E 72289113 98080354 C7297AD7 89B627F8 ED40D924
  ADF48383 1B332C7F 73C58686 6279E2A4 4BF41644 3E60F131 090D3F5D 25F0C025
  43CB0203 010001A3 81FE3081 FB301D06 03551D0E 04160414 F7F4E80E F6CC4772
  5F278C44 6B85F8EE 8345AB99 3081CB06 03551D23 0481C330 81C08014 F7F4E80E
  F6CC4772 5F278C44 6B85F8EE 8345AB99 A181A4A4 81A13081 9E310B30 09060355
  04061302 55533113 30110603 55040813 0A43616C 69666F72 6E696131 11300F06
  03550407 13085361 6E204A6F 7365311C 301A0603 55040A13 13436973 636F2053
  79737465 6D732C20 496E632E 310E300C 06035504 0B13054E 53535447 31163014
  06035504 03130D4A 6F686E20 4C617574 6D616E6E 3121301F 06092A86 4886F70D
  01090116 126A6C61 75746D61 6E406369 73636F2E 636F6D82 0100300C 0603551D
  13040530 030101FF 300D0609 2A864886 F70D0101 04050003 82010100 6D12CFF8
  31078DF6 94FE5CF0 8F83639B 414F32D8 069D23E2 37E182BE 7C31EC14 E87AF216
  61A6CCD3 37656934 4BE4157A 400E182B EC390D1A DC130A56 B8F35BFB D2234556
  24152FE8 A736B670 58CC684E 750D08AE C7739907 917B7A72 3D26BEC7 9F554CF1
  5E5EF499 ABA11124 55966616 AC9C52B2 B1082DEA D962CBAF E476C575 A9DDFBFA
  C4AE63F6 1D5C9F76 7B4B9CA7 52CE65C9 E65C04FC 4B7642D6 0D1A8AF4 38194B7A
  CA307EC9 51DCB847 8B8C27FB 98ACEE60 0B80DC3F 36E4E252 BD731F5F 0E781E26
```

```
  C1CA4120 9B0B689B BA654250 97B22A76 CC126B77 C7779AAA D3F93C3F DCF46006
  2B7F7F8C 150AF889 BBEC62F1 E53B4F3B A3626CD6 05B8AB3D F8A6A361
   quit
archive
 log config
scripting tcl trustpoint name mytrust
scripting tcl secure-mode
!
!
end
```

その他の参考資料

ここでは、署名付き TCL スクリプト機能の関連資料について説明します。

関連資料

関連項目	マニュアル タイトル
Cisco PKI の概要：PKI の理解と計画 PKI の実装と管理	*Security Configuration Guide, Release 12.4*
PKI コマンド：コマンド構文、コマンド モード、コマンド履歴、デフォルト、使用に関する注意事項、および例	*Cisco IOS Security Command Reference, Release 12.4*

標準

標準	タイトル
なし	--

MIB

MIB	MIB のリンク
なし	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャ セットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

RFC

RFC	タイトル
なし	--

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/en/US/support/index.html

署名済み Tcl スクリプトの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 74: 署名済み *Tcl* スクリプトの機能情報

機能名	リリース	機能情報
署名済みTclスクリプト		署名付き TCL スクリプト機能を使用すると、デジタル署名を生成する証明書を作成し、そのデジタル署名を使用してTclスクリプトに署名することが可能になります。 次のコマンドがこの機能で導入されました。**scripting tcl secure-mode**、 **scripting tcl trustpoint name**、 **scripting tcl trustpoint untrusted**、 および **tclsafe.**

用語集

CA：認証局。証明書要求の管理と、関係する IP セキュリティ ネットワーク デバイスへの証明書の発行を担当しているサービス。このサービスは、参加デバイスを一元的に管理します。またこれらのサービスによって受信者は、明示的に信頼してアイデンティティを確認し、デジタル証明書を作成できます。

証明書：ユーザー名またはデバイス名を公開キーにバインドする電子ドキュメント。証明書は、一般的にデジタル署名を確認するために使用されます。

CRL：証明書失効リスト。失効した証明書のリストが含まれる電子ドキュメントです。CRL は、証明書を発行したCAによって作成され、デジタル署名されます。CRLには、証明書の発行日と失効日が含まれています。現行の CRL が失効すると、新しい CRL が発行されます。

IPsec：IP セキュリティ。

ピア証明書：ピアが提示する証明書で、ピアの公開キーが含まれており、トラストポイント CA によって署名されています。

PKI：公開キー インフラストラクチャ。セキュアに設定された通信に使用されているネットワーク コンポーネントの暗号キーと ID 情報を管理するシステムです。

RA：登録局。CA のプロキシとして機能するサーバーで、CA がオフラインのときでも CA の機能を継続できます。RA は CA サーバー上に設定するのが通常ですが、別アプリケーションとして、稼働のための別デバイスを必要とする場合もあります。

RSA キー：公開キー暗号化システムで、Ron Rivest（ロナルド・リベスト）、Adi Shamir（アディ・シャミア）、Leonard Adleman（レオナルド・エーデルマン）の3人によって開発されました。デバイスの証明書を取得するには、RSAキーペア（公開キーと秘密キー）が必要です。

SHA1：Secure Hash Algorithm 1。

SSH：セキュア シェル。

SSL：Secure Socket Layer。

注意事項

本ソフトウェア ライセンスに関連する通知内容を以下に示します。

OpenSSL/Open SSL Project

本製品には、OpenSSL Toolkit（http://www.openssl.org/）で使用するために OpenSSL Project によって開発されたソフトウェアが含まれています。

本製品には、Eric Young 氏（eay@cryptsoft.com）によって作成された暗号化ソフトウェアが含まれています。

本製品には、Tim Hudson 氏（tjh@cryptsoft.com）によって作成されたソフトウェアが含まれています。

ライセンスの問題

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License:

Copyright © 1998-2007 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. 本ソフトウェアの機能または使用に言及するすべての広告資料には、以下の謝辞が表示される必要があります。「本製品には、OpenSSL Toolkit で使用するために OpenSSL Project によって開発されたソフトウェアが含まれています（http://www.openssl.org/）」。
4. The names “OpenSSL Toolkit” and “OpenSSL Project” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:

「本製品には、OpenSSL Toolkit（http://www.openssl.org/）で使用するために OpenSSL プロジェクトによって開発されたソフトウェアが含まれています」。

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT “AS IS”' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

本製品には、Eric Young 氏（eay@cryptsoft.com）によって作成された暗号化ソフトウェアが含まれています。本製品には、Tim Hudson 氏（tjh@cryptsoft.com）によって作成されたソフトウェアが含まれています。

Original SSLeay License:

Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young’s, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

“This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)”.

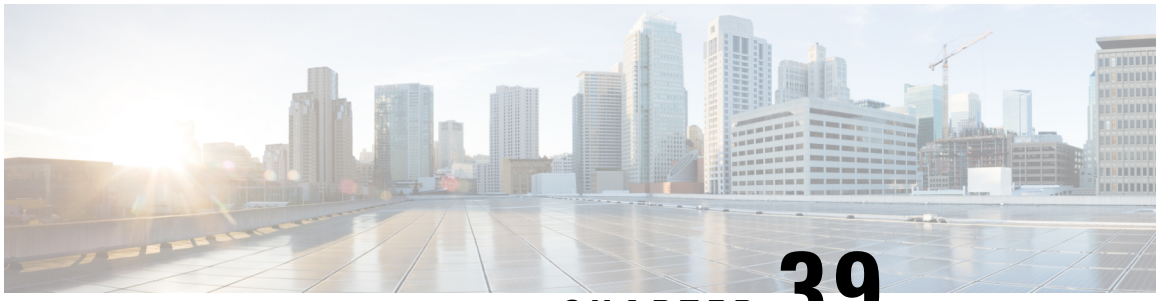
The word ‘cryptographic’ can be left out if the routines from the library being used are not cryptography-related.

1. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: “This product includes software written by Tim Hudson (tjh@cryptsoft.com)”.

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The license and distribution terms for any publicly available version or derivative of this code cannot be changed, i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License.]



CHAPTER 39

EEM アクションの Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

action_policy

Tcl スクリプトで、None イベント ディテクタで登録された Embedded Event Manager（EEM）ポリシーを実行できるようにします。EEM ポリシーを実行するアクションは、**event manager run** コマンドを使用して実行することもできます。

構文

```
action_policy ?
```

引数

?（文字列を表す）	（必須）実行がスケジュールされる EEM ポリシーの名前。ポリシーは、None イベント ディテクタを使用して事前に登録しておく必要があります。

なし

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err = 2) FH_ESYSERR (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID (unknown event ID)
```

このエラーは、ポリシーが登録されていないため、未知であることを意味します。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

action_process

ソフトウェア モジュール方式プロセスを起動、再起動、または停止します。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
action_process start|restart|kill [job_id ?]
[process_name ?] [instance ?]
```

引数

start	（必須）プロセスが起動されるよう指定します。
restart	（必須）プロセスが再起動されるよう指定します。
kill	（必須）プロセスが停止されるよう指定します。
job_id	（任意）システム マネージャによってプロセスに割り当てられるジョブ ID。この引数を指定する場合、1 ～ 4294967295 の範囲の整数である必要があります。
process_name	（任意）プロセス名。job_id を指定するか、または、process_name および instance を指定する必要があります。
instance	（任意）プロセス インスタンス ID。この引数を指定する場合、1 ～ 4294967295 の範囲の整数である必要があります。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

```
(_cerr_sub_num = 425, _cerr_sub_err = 1) SYSMGR_ERROR_INVALID_ARGS  (Invalid arguments
passed)
```

このエラーは、渡された引数が無効であったことを意味します。

```
(_cerr_sub_num = 425, _cerr_sub_err = 2) SYSMGR_ERROR_NO_MEMORY  (Could not allocate
required memory)
```

このエラーは、メモリの内部 SYSMGR 要求に障害が発生したことを意味します。

```
(_cerr_sub_num = 425, _cerr_sub_err = 5) SYSMGR_ERROR_NO_MATCH  (This process is not
known to sysmgr)
```

このエラーは、プロセス名が未知であったことを意味します。

```
(_cerr_sub_num = 425, _cerr_sub_err = 14) SYSMGR_ERROR_TOO_BIG  (outside the valid limit)
```

このエラーは、オブジェクト サイズがその最大値を超えたことを意味します。

```
(_cerr_sub_num = 425, _cerr_sub_err = 15) SYSMGR_ERROR_INVALID_OP  (Invalid operation
for this process)
```

このエラーは、その動作がプロセスに対して無効であったことを意味します。

action_program

Tcl スクリプトで、POSIX プロセス（プログラム）を実行できるようにします。任意で、該当する引数文字列、環境文字列、標準入力（stdin）パス名、標準出力（stdout）パス名、または標準エラー（stderr）パス名を使用します。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
action_program path ? [argv ?] [envp ?] [stdin ?] [stdout ?] [stderr ?]
```

引数

path	（必須）実行するプログラムのパス名。
argv	（任意）プログラムの引数文字列。
envp	（任意）プログラムの環境文字列。
stdin	（任意）stdin のパス名。
stdout	（任意）stdout のパス名。
stderr	（任意）stderr のパス名。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

```
(_cerr_sub_err = 34)    FH_EMAXLEN  (maximum length exceeded)
```

このエラーは、オブジェクト長またはオブジェクト数が、最大値を超えたことを意味します。

action_reload

デバイスがリロードされます。

構文

```
action_reload
```

引数

なし

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

action_script

Tcl スクリプトで、すべての Tcl スクリプトの実行をイネーブルまたはディセーブルにします（スクリプト スケジューラをイネーブルまたはディセーブルにします）。

構文

```
action_script [status enable|disable]
```

引数

status	（任意）スクリプト実行ステータスを示すフラグ。この引数がイネーブルに設定されている場合、スクリプト実行がイネーブルにされます。この引数がディセーブルに設定されている場合、スクリプト実行がディセーブルにされます。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

```
(_cerr_sub_err = 52)    FH_ECONFIG  (configuration error)
```

このエラーは、設定エラーが発生したことを意味します。

action_snmp_trap

Embedded Event Manager Notification MIB を使用して簡易ネットワーク管理プロトコル（SNMP）トラップを送信します。

構文

```
action_snmp_trap [intdata1 ?] [intdata2 ?] [strdata ?]
```

引数

intdata1	（任意）トラップで送信される任意の整数。
intdata2	（任意）トラップで送信される任意の整数。
strdata	（任意）トラップで送信される任意の文字列データ。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

action_snmp_object_value

SNMP get 要求で返される簡易ネットワーク管理プロトコル（SNMP）オブジェクト ID および値を設定します。

構文

```
action_snmp_object_value {int|uint|counter|gauge|ipv4|octet|counter64|string} ?
[next_oid ?]
```

引数

int	管理対象オブジェクトのコンテキスト内の番号が付けられたタイプを指定する場合は、32 ビットの数字が使用されます。
uint	10 進数の値を表す 32 ビット番号。
counter	最小値が 0 の 32 ビットの数値。
gauge	最小値が 0 の 32 ビットの数値。
ipv4	IP バージョン 4 アドレス。
octet	物理アドレスを表すために使用される、16 進表記のオクテット文字列。
counter 64	最小値が 0 の 64 ビットの数値。
string	テキスト文字列を表すために使用される、テキスト表記のオクテット文字列。
next_oid	テールブルにある次のオブジェクトの OID。テーブルの最後のオブジェクトの場合は NULL です。

結果文字列

なし

_cerrno を設定

対応

action_switch

完全冗長環境でセカンダリ プロセッサで処理するよう切り替えます。**action_switch** Tcl コマンド拡張を使用する前に、デバイスでバックアッププロセッサをインストールする必要があります。ハードウェアが完全冗長ではない場合、切り替えアクションは実行されません。

構文

```
action_switch
```

引数

なし

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err = 2) FH_ESYSERR (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 14)    FH_ENOSUCHACTION  (unknown action type)
```

このエラーは、要求されたアクション コマンドが未知であることを示します。

action_syslog

EEM スクリプトがトリガーされるときに、指定された機能を使用して定期的な Syslog メッセージを生成します。

構文

```
action_syslog [priority emerg|alert|crit|err|warning|notice|info|debug]
[msg ?] [facility ?]
```

引数

priority	（任意）action_syslog メッセージ ファシリティ レベル。この引数が指定されない場合、デフォルトのプライオリティは LOG_INFO です。
msg	（任意）記録されるメッセージ。
facility	（任意）Syslog の機能。

結果文字列

なし

_cerrno を設定

対応

action_track_read

Embedded Event Manager（EEM）スクリプトがトリガーされるときにトラックされるオブジェクトの状態を読み取ります。

構文

```
action_track_read ?
```

引数

?（番号を表す）	（必須）1 から 500 の範囲でトラックされるオブジェクト番号。

結果文字列

```
number {%u}
state {%s}
```

_cerrno を設定

対応

```
FH_ENOTRACK
```

このエラーは、トラックされるオブジェクト番号が見つからなかったことを意味します。

action_track_set

Embedded Event Manager（EEM）スクリプトがトリガーされるときにトラックされるオブジェクトの状態を設定します。

構文

```
action_track_set ? state up|down
```

引数

?（番号を表す）	（必須）1 から 500 の範囲でトラックされるオブジェクト番号。
state	（必須）トラックされるオブジェクトの状態が設定されるよう指定します。up と指定されている場合、トラックされているオブジェクトの状態はアップです。down と指定されている場合、トラックされているオブジェクトの状態はダウンです。

結果文字列

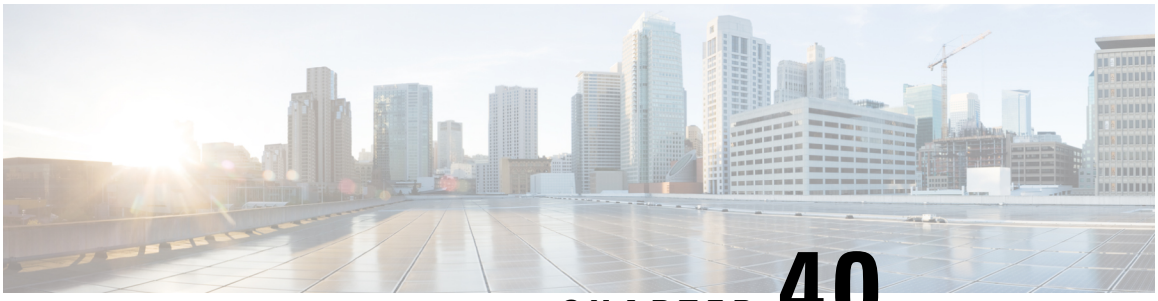
なし

_cerrno を設定

対応

```
FH_ENOTRACK
```

このエラーは、トラックされるオブジェクト番号が見つからなかったことを意味します。



CHAPTER 40

EEM CLI ライブラリのコマンド拡張

すべてのコマンドラインインターフェイス（CLI）ライブラリ コマンド拡張は、::cisco::eem 名前空間に属します。

このライブラリによって、ユーザーに対し、CLI コマンドを実行し、Tcl でコマンドの出力を取得する機能が用意されます。コマンドが exec によって実行され、コマンドの出力が読み戻されるようにするため、ユーザーは、このライブラリでコマンドを使用して、exec を生成し、それに対して仮想端末チャネルをオープンし、コマンドを記述してチャネルに対して実行できます。

CLI コマンドには、対話式コマンドと非対話式コマンドの、2 つのタイプがあります。

対話式コマンドでは、コマンドの入力後、デバイスによって異なるユーザー オプションが質問される「Q&A」フェーズがあり、ユーザーは、各質問に対する答えを入力する必要があります。すべての質問が適切に答えられた後、ユーザーのオプションに従って、完了するまでコマンドが実行されます。

非対話式コマンドでは、コマンドが一度入力されると、コマンドが完了まで実行されます。EEM スクリプトを使用してさまざまなタイプのコマンドを実行するには、異なる CLI ライブラリ コマンド シーケンスを使用する必要があります。詳細については、cli_write Tcl コマンドの「CLI ライブラリを使用した非対話式コマンドの実行」の項および「CLI ライブラリを使用した対話式コマンドの実行」の項を参照してください。

vty 行は、**line vty** CLI コンフィギュレーション コマンドを使用して設定された vty 行のプールから割り当てられます。EEM によって vty 行が使用されていない場合で、使用可能な vty 行がある場合、EEM では、vty 行が使用されます。EEM によって vty 行がすでに使用されている場合で、使用可能な 3 行以上の vty 行がある場合も、EEM では、vty 行が使用されます。3 行よりも少ない vty 行が使用可能な場合、残りの vty 行は Telnet で使用するために予約されているので、接続は失敗することに注意してください。

お使いのリリースで XML-PI がサポートされている場合があります。XML-PI サポート、新しい CLI ライブラリ コマンド拡張、および、XML-PI の実装方法の例については、「EEM CLI ライブラリ XML-PI サポート」を参照してください。

cli_close

exec プロセスをクローズし、コマンドライン インターフェイス（CLI）に接続された、vty および指定されたチャネル ハンドラをリリースします。

構文

```
cli_close fd tty_id
```

引数

fd	（必須）CLI チャネル ハンドラ。
tty_id	（必須）**cli_open** コマンド拡張から返された TTY ID。

結果文字列

なし

_cerrno を設定

チャネルをクローズできない。

cli_exec

指定されたチャネル ハンドラにコマンドを記述し、コマンドを実行します。次に、チャネルからコマンドの出力を読み取り、出力を返します。

構文

```
cli_exec fd cmd
```

引数

fd	（必須）コマンドライン インターフェイス（CLI）チャネル ハンドラ。
cmd	（必須）実行する CLI コマンド。

結果文字列

実行された CLI コマンドの出力。

_cerrno を設定

チャネルを読み取れない。

cli_get_ttyname

該当する TTY ID の実際と疑似の TTY の名前を返します。

構文

```
cli_get_ttyname tty_id
```

引数

tty_id	（必須）**cli_open** コマンド拡張から返された TTY ID。

結果文字列

```
pty %s tty %s
```

_cerrno を設定

なし

cli_open

vty を割り当て、EXEC コマンドライン インターフェイス（CLI）セッションを作成し、vty をチャネル ハンドラに接続します。チャネル ハンドラを含む配列を返します。

Note **cli_open** への各コールによって、Cisco IOS vty 回線を割り当てる Cisco IOS EXEC セッションが開始されます。vty は、**cli_close** ルーチンが呼び出されるまで、使用中のままです。vty 行は、**line vty** CLI コンフィギュレーション コマンドを使用して設定された vty 行のプールから割り当てられます。EEM によって vty 行が使用されていない場合で、使用可能な vty 行がある場合、EEM では、vty 行が使用されます。EEM によって vty 行がすでに使用されている場合で、使用可能な 3 行以上の vty 行がある場合も、EEM では、vty 行が使用されます。3 行よりも少ない vty 行が使用可能な場合、残りの vty 行は Telnet で使用するために予約されているので、接続は失敗することに注意してください。

構文

```
cli_open
```

引数

なし

結果文字列

```
"tty_id {%s} pty {%d} tty {%d} fd {%d}"
```

イベント タイプ	説明
tty_id	TTY ID。
pty	PTY デバイス名。
tty	TTY デバイス名。
fd	CLI チャネル ハンドラ。

_cerrno を設定

- EXEC の pty を取得できない。
- EXEC CLI セッションを作成できない。
- 最初のプロンプトを読み取れない。

cli_read

読み取られている内容でデバイス プロンプトのパターンが発生するまで、指定されたコマンドライン インターフェイス（CLI）のチャネル ハンドラからコマンド出力を読み取ります。一致するまで、読み取られたすべての内容を返します。

構文

```
cli_read fd
```

引数

fd	（必須）CLI チャネル ハンドラ。

結果文字列

読み取られたすべての内容。

_cerrno を設定

デバイス名を取得できない。

Note この Tcl コマンド拡張によって、デバイス プロンプトを待つ状態がブロックされ、読み取られた内容が表示されます。

cli_read_drain

指定されたコマンドライン インターフェイス（CLI）のチャネル ハンドラのコマンド出力を読み取り、排出します。読み取られたすべての内容を返します。

構文

```
cli_read_drain fd
```

引数

fd	（必須）CLI チャネル ハンドラ。

結果文字列

読み取られたすべての内容。

_cerrno を設定

なし

cli_read_line

指定されたコマンドライン インターフェイス（CLI）のチャネル ハンドラから、コマンド出力の 1 行を読み取ります。読み取られた回線を返します。

構文

```
cli_read_line fd
```

引数

fd	（必須）CLI チャネル ハンドラ。

結果文字列

読み取られた回線。

_cerrno を設定

なし

Note この Tcl コマンド拡張によって、行の末尾を待つ状態がブロックされ、読み取られた内容が表示されます。

cli_read_pattern

読み取られている内容でパターンが発生するまで、指定されたコマンドライン インターフェイス（CLI）のチャネル ハンドラからコマンド出力を読み取ります。一致するまで、読み取られたすべての内容を返します。

Note パターン マッチ ロジックで、Cisco IOS コマンドから配信されるコマンド出力データを探すことによって、照会が試行されます。照会は、出力バッファの最新の 256 文字で常に行われます。ただし、使用可能な文字がより少ない場合は、より少ない文字で照会が行われます。正常な一致に 256 よりも多い文字が必要な場合、パターン マッチは実行されません。

構文

```
cli_read_pattern fd ptn
```

引数

fd	（必須）CLI チャネル ハンドラ。
ptn	（必須）チャネルからコマンド出力を読み取るときに、パターンが照会されます。

結果文字列

読み取られたすべての内容。

_cerrno を設定

なし

Note この Tcl コマンド拡張によって、指定されたパターンを待つ状態がブロックされ、読み取られた内容が表示されます。

cli_run

clist にある回数を繰り返し、それぞれが、イネーブル モードで実行されるコマンドライン インターフェイス（CLI）であることを前提とします。正常に実行されると、実行されたすべてのコマンドの出力を返します。失敗すると、失敗からのエラーを返します。

構文

```
cli_run clist
```

引数

clist	（必須）実行されるコマンドのリスト。

結果文字列

出力されるすべてのコマンドの出力、またはエラー メッセージ。

_cerrno を設定

なし。

使用例

次に、**cli_run** コマンド拡張の使用例を示します。

```
set clist [list {sh run} {sh ver} {sh event man pol reg}]
cli_run { clist }
```

cli_run_interactive

3 つの項目がある clist のサブリストを提供します。正常に実行されると、実行されたすべてのコマンドの出力を返します。失敗すると、失敗からのエラーを返します。可能な場合には、配列も使用します。予測と応答を別々に保持することによって、より簡単に後で読み取ることができます。

構文

```
cli_run_interactive clist
```

引数

clist	（必須）3 つの項目のリスト：
	• command：実行するコマンド
	• expect：予想される応答プロンプトの正規表現パターンマッチ
	• responses：2 つの項目のアレイとして構成された応答プロンプトに対して可能性がある応答のリスト
	  • expect：可能性がある応答プロンプトの正規表現パターン マッチ
	  • reply：その予測されるプロンプトの応答

結果文字列

出力されるすべてのコマンドの出力、またはエラーメッセージ。各コマンドが実行されると、その出力が結果の変数に追加されます。入力リストが枯渇すると、CLI チャネルが閉じ、集約結果が返されます。

_cerrno を設定

なし。

使用例

次に、cli_run_ interactive コマンド拡張を使用してインターフェイス fa0/0 のカウンタをクリアする例を示します。

```
set cmdarr(command) "clear counters fa0/0"
set cmdarr(responses) [list]
set resps(expect) {[confirm]}
set resps(reply) "y"
lappend cmdarr(responses) [array get resps]
set rc [catch {cli_run_interactive [list [array get cmdarr]]} result]
```

発生する可能性があるエラーには、次のようなものがあります。

- exec の pty を取得できない。
- exec を生成できない。
- 最初のプロンプトを読み取れない。
- チャネルを読み取れない。
- チャネルをクローズできない。

cli_write

指定されたCLIチャネルハンドラに対して実行されるコマンドを書き込みます。CLIチャネルハンドラによって、コマンドが実行されます。

構文

```
cli_write fd cmd
```

引数

fd	（必須）CLI チャネル ハンドラ。
cmd	（必須）実行する CLI コマンド。

結果文字列

なし

_cerrno を設定

なし

使用例

たとえば、次のように、コンフィギュレーション CLI コマンドを使用して、イーサネット インターフェイス 1/0 をアップにします。

```
if [catch {cli_open} result] {
puts stderr $result
exit 1
} else {
array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "config t"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "interface Ethernet1/0"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "no shut"} result] {
puts stderr $result
exit 1
}
if [catch {cli_exec $cli1(fd) "end"} result] {
puts stderr $result
exit 1
```

```
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} } result] {
puts stderr $result
exit 1
```

CLI ライブラリを使用した非対話式コマンドの実行

非対話式コマンドを実行するには、**cli_exec** コマンド拡張を使用して、コマンドを発行し、次に、出力とデバイスプロンプトを待ちます。たとえば、コンフィギュレーション CLI コマンドを使用して、イーサネット インターフェイス 1/0 をアップにする例を示します。

```
if [catch {cli_open} result] {
error $result $errorInfo
} else {
set fd $result
}
if [catch {cli_exec $fd "en"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "config t"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "interface Ethernet1/0"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "no shut"} result] {
error $result $errorInfo
}
if [catch {cli_exec $fd "end"} result] {
error $result $errorInfo
}
if [catch {cli_close $fd} result] {
error $result $errorInfo
}
```

CLI ライブラリを使用した対話式コマンドの実行

対話式コマンドを実行するには、次の 3 つのフェーズが必要です。

- フェーズ 1：**cli_write** コマンド拡張を使用して、コマンドを発行します。
- フェーズ 2：Q&A フェーズ。**cli_read_pattern** コマンド拡張を使用して質問を読み取り（質問テキストの照合に指定される通常パターン）、**cli_write** コマンド拡張を使用して、代わりに回答を書き戻します。
- フェーズ 3：非対話式フェーズ。すべての質問が回答され、完了までコマンドが実行されます。**cli_read** コマンド拡張を使用して、コマンドの出力とデバイスプロンプトを待ちます。

たとえば、CLI コマンドを使用して、ブートフラッシュをまとめます。Tcl 変数 cmd_output に、このコマンドの出力を保存します。

```
if [catch {cli_open} result] {
error $result $errorInfo
} else {
array set cli1 $result
}
```

```
if [catch {cli_exec $cli1(fd) "en"} result] {
error $result $errorInfo
}

# Phase 1: issue the command
if [catch {cli_write $cli1(fd) "squeeze bootflash:"} result] {
error $result $errorInfo
}

# Phase 2: Q&A phase
# wait for prompted question:
# All deleted files will be removed. Continue? [confirm]
if [catch {cli_read_pattern $cli1(fd) "All deleted"} result] {
error $result $errorInfo
}
# write a newline character
if [catch {cli_write $cli1(fd) "\n"} result] {
error $result $errorInfo
}
# wait for prompted question:
# Squeeze operation may take a while. Continue? [confirm]
if [catch {cli_read_pattern $cli1(fd) "Squeeze operation"} result] {
error $result $errorInfo
}
# write a newline character
if [catch {cli_write $cli1(fd) "\n"} result] {
error $result $errorInfo
}

# Phase 3: noninteractive phase
# wait for command to complete and the router prompt
if [catch {cli_read $cli1(fd) } result] {
error $result $errorInfo
} else {
set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
error $result $errorInfo
}
```

次に、CLI **reload** コマンドを使用して、デバイスがリロードされる例を示します。EEM **action_reload** コマンドによって、より効率的な方法で同じ結果が達成されますが、この例は、対話式コマンド実行での CLI ライブラリでの柔軟性を示すために示します。

```
# 1. execute the reload command
if [catch {cli_open} result] {
    error $result $errorInfo
} else {
    array set cli1 $result
}
if [catch {cli_exec $cli1(fd) "en"} result] {
    error $result $errorInfo
}
if [catch {cli_write $cli1(fd) "reload"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_read_pattern $cli1(fd) ".*(System configuration has been modified. Save\\\?
 \\\[yes/no\\\]: )"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
```

```
}
if [catch {cli_write $cli1(fd) "no"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_read_pattern $cli1(fd) ".*(Proceed with reload\\\? \\\[confirm\\\])"}
result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_write $cli1(fd) "y"} result] {
    error $result $errorInfo
} else {
    set cmd_output $result
}
if [catch {cli_close $cli1(fd) $cli1(tty_id)} result] {
    error $result $errorInfo
}
```

CHAPTER 41

EEM CLI ライブラリ XML-PI サポート

XML プログラマチック インターフェイス（XML-PI）が Cisco IOS Release 12.4(22)T で導入されました。XML-PI は異なるシスコ製品間で矛盾のない方法で、IOS コマンドライン インターフェイス（CLI）show コマンドを XML 形式にカプセル化した、プログラム可能なインターフェイスを提供します。XML-PI を使用する場合は、既知のキーワードを使用して IOS show コマンドの出力を Tcl スクリプトから解析できます。「スクリーン スクレイピング」出力に対する正規表現サポートを使用する必要はありません。

XML-PI コマンド拡張を使用する利点は、CLI **show** コマンドを使用して生成される特定の出力情報の抽出を容易にすることです。ほとんどの show コマンドは出力内の多くのフィールドを返しますが、現在のところ、行の中央に表示される可能性がある特定の情報を抽出するには正規表現を使用する必要があります。XML-PI サポートは一連の Tcl ライブラリ関数を提供し、次の形式の IOS CLI 形式の拡張からの出力の解析を容易にします。

```
show
 <
show-command
> | format
 {
spec-file
}
```

ここで、spec-file は現在サポートされている各 **show** コマンドのすべての SPEC ファイルエントリ（SFE）を連結したものです。XML-PI プロジェクトの一環として、デフォルトの spec-file が IOS リリース 12.4(22)T イメージに組み込まれます。デフォルトの spec-file には、一連の少数のコマンドが組み込まれ、それらのコマンドの SFE には考えられるタグのサブセットが組み込まれます。format コマンドで spec-file が提供されない場合、デフォルトの spec-file が使用されます。

XML-PI に関するより全般的な詳細については、「XML-PI」の章を参照してください。

xml_pi_exec

fd 引数を使用してハンドラが指定され、spec_file 引数によって spec-file が指定されてコマンドが実行されるチャネルに対して、cmd 引数を使用して指定された XML-PI コマンドを書き込みます。コマンドの未加工XML出力データが、チャネルから読み取られ、XML出力が返されます。

構文

```
xml_pi_show fd cmd [spec_file]
```

引数

fd	（必須）cli_open から取得された CLI ライブラリ ファイル記述子。
cmd	（必須）IOS 表示コマンド。
spec_file	（任意）IOS CLI 表示コマンド spec_file。

結果文字列

IOS 表示コマンドの結果（XML フォーマット）。

_cerrno を設定

発生する可能性があるエラーは、次のとおりです。

1.チャンネル読み取りエラー

xml_pi_parse

この機能に xml_data として渡される XML 表示コマンドの未加工出力を処理し、xml_tags_list によって指定されたこれらのフィールドを取得します。次の処理が発生します。

ステップ 1：XML タグ リストは、Tcl リストとして有効にされます。低い順序の名前が、該当するコマンドであいまいな場合、XML タグは、低い順序の XML タグ名または完全修飾 XML タグ名として指定できます。

タグ例：<Interface> <ShowIpInterfaceBrief><IPInterfaces><entry><Interface>

ステップ 2：xml_data は、有効な XML として有効にされ、XML パース ツリーに解釈されます。

ステップ 3：XML パース ツリーをウォークし、各タグが、XML タグ リストのエントリと比較されます。照会が発生する場合、タグ名が、現在の Tcl スコープ内で定義されている Tcl 手順と一致しているかどうかが判断されます。一致する場合、Tcl 手順は、現在の結果で呼び出さ

れます。一致しない場合、タグ名と、タグ名に関連付けられているデータは、現在の結果の末尾に追加されます。

構文

```
xml_pi_parse fd xml_show_cmd_output xml_tags_list
```

引数

fd	（必須）cli_open から取得された CLI ライブラリ ファイル記述子。
xml_show_cmd_output	（必須）xml 形式での、xml_pi_show コマンド拡張の出力。
xml_tags_list	（必須）関連するタグのリスト。

結果文字列

XML タグ名によって索引化される Tcl 配列のデータ。

Note 現在の結果は、Tcl 手順の呼び出し後にリセットされます。

_cerrno を設定

発生する可能性があるエラーは、次のとおりです。

1. XML タグリストの分割中にエラーが発生する 2. XML タグリストが null に指定されている 3. XML タグツリーが 20 レベルを超えている 4. 呼び出された Tcl プロシージャがエラーを返した 5. メモリ割り当ての失敗 6. XML 解析エラー 7. XML ドメインの作成に失敗

xml_pi_read

読み取られている内容でルータプロンプトのパターンが発生するまで、ファイル記述子によって指定されているハンドラがある CLI チャネルから（指定された表示コマンドから）XML-PI コマンド出力を読み取ります。XML 形式で、一致するまで、読み取られたすべての内容を返します。

構文

```
xml_pi_read fd
```

引数

fd	（必須）cli_open から取得された CLI ライブラリ ファイル記述子。

結果文字列

XML 形式で読み取られるすべての内容。

_cerrno を設定

発生する可能性があるエラーは、次のとおりです。

1. ルータ名を取得できない 2. コマンドエラー

xml_pi_write

fd 引数を使用してハンドラが指定され、spec_file 引数によって仕様ファイルが指定されてコマンドが実行されるチャネルに対して、cmd 引数を使用して指定された XML-PI コマンドを書き込みます。

構文

```
xml_pi_write fd cmd spec_file
```

引数

fd	（必須）cli_open から取得された CLI ライブラリ ファイル記述子。
cmd	（必須）IOS 表示コマンド。
spec_file	（任意）IOS CLI 表示コマンド spec_file。

結果文字列

なし

_cerrno を設定

なし

XML-PI 機能のサンプル使用

次の EEM ポリシー（sample.tcl）で、新しい EEM XML-PI 機能の 5 つの異なる実装の 1 つの例を示します。odm spec-file（例 2）が、このポリシーに続きます。

```
::cisco::eem::event_register_none maxrun 60
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# open the cli_lib.tcl channel
if [catch {cli_open} result] {
error $result $errorInfo
} else {
array set cli1 $result
}
```

```
# enter "enable" privilege mode
if [catch {cli_exec $cli1(fd) "en"} result] {
error $result $errorInfo
}
# Example 1:
#
# Detect if XML-PI is present in this image
# Invoke xml_pi_exec with the default spec file for the "show inventory"
# command. After the command executes $result contains the raw XML data if
# the command is successful.
if [catch {xml_pi_exec $cli1(fd) "show inventory" ""} result] {
puts "Example 1: XML-PI support is not present in this image - exiting"
exit
} else {
puts "Example 1: XML-PI support is present in this image"
}
# Example 2:
#
# In the next example we demonstrate how to extract two data elements
# from the "show version" command using the specified XML-PI spec file.
# The raw output from this command is as follows:
#
# Device#show version | format disk2:speceemtest.odm
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowVersion>
# <Version>12.4(20071029:194217)</Version>
# <Compiled>Thu 08-Nov-07 11:28</Compiled>
# <ROM>System Bootstrap, Version 12.2(20030826:190624) [BLD-npeg1_rommon_r11 102],
DEVELOPMENT</ROM>
# <uptime>17 minutes</uptime>
# <processor>NPE-G1</processor>
# <bytesofmemory>983040K/65536K</bytesofmemory>
# <CPU>700MHz</CPU>
# <L2Cache>0.2</L2Cache>
# <GigabitEthernetinterfaces>3</GigabitEthernetinterfaces>
# <bytesofNVRAM>509K</bytesofNVRAM>
# <bytesofATAPCMCIAcard>125952K</bytesofATAPCMCIAcard>
# <Sectorsize>512 bytes</Sectorsize>
# <bytesofFlashinternalSIMM>16384K</bytesofFlashinternalSIMM>
# <Configurationregister>0x2100</Configurationregister>
# </ShowVersion>
#
# Invoke xml_pi_exec with the spec file "disk2:speceemtest.odm" for the
# "show version" command. After the command executes $result contains
# the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show version" "disk2:speceemtest.odm"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <processor> and <CPU> fields be returned.
array set xml_result [xml_pi_parse $cli1(fd) $result "<processor> <CPU>"]
puts "Example 2: Processor is $xml_result(<processor>) CPU is $xml_result(<CPU>)"
}
# Example 3:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show inventory" command using the default built-in
# XML-PI spec file. Sample raw output from this command is as follows:
#
# Device#show inventory | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowInventory>
# <SpecVersion>built-in</SpecVersion>
```

```
# <InventoryEntry>
# <ChassisName>"Chassis"</ChassisName>
# <Description>"Cisco 7206VXR, 6-slot chassis"</Description>
# <PID>CISCO7206VXR</PID>
# <VID>
# </VID>
# <SN>31413378 </SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>"NPE-G1 0"</ChassisName>
# <Description>"Cisco 7200 Series Network Processing Engine
NPE-G1"</Description>
# <PID>NPE-G1</PID>
# <VID>
# </VID>
# <SN>31493825 </SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>"disk2"</ChassisName>
# <Description>"128MB Compact Flash Disk for NPE-G1"</Description>
# <PID>MEM-NPE-G1-FLD128</PID>
# <VID>
# </VID>
# <SN>NAME: "module 1"</SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>"module 1"</ChassisName>
# <Description>"Dual Port FastEthernet (RJ45)"</Description>
# <PID>PA-2FE-TX</PID>
# <VID>
# </VID>
# <SN>JAE0827NGKX</SN>
# </InventoryEntry>
# <InventoryEntry>
# <ChassisName>"Power Supply 2"</ChassisName>
# <Description>"Cisco 7200 AC Power Supply"</Description>
# <PID>PWR-7200-AC</PID>
# <VID>
# </VID>
# </InventoryEntry>
# </ShowInventory>
#
# Define a procedure to be called every time the <InventoryEntry> tag
# is processed. Since this tag precedes each new output record, the data
# that is passed into this procedure contains the fields that have been
# requested via xml_pi_parse since the previous time this procedure was
# called.
proc <InventoryEntry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
set output [format "Example 3: Item %2d %-18s %s" \
$num $xml_result(<PID>) $xml_result(<Description>)]
puts $output
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show inventory" command. After the command executes $result contains
# the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show inventory"} result] {
```

```
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <PID> and <Description> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <InventoryEntry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<InventoryEntry> <PID> <Description>"]
# Display the data from the last record.
incr num
set output [format "Example 3: Item %2d %-18s %s" \
$num $xml_result(<PID>) $xml_result(<Description>)]
puts $output
}
# Example 4:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show ip interface brief" command using the default
# built-in XML-PI spec file. Sample raw output from this command is as
# follows:
#
# Device#show ip interface brief | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowIpInterfaceBrief>
# <SpecVersion>built-in</SpecVersion>
# <IPInterfaces>
# <entry>
# <Interface>GigabitEthernet0/1</Interface>
# <IP-Address>172.19.209.34</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>up</Status>
# <Protocol>up</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/2</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/3</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/0</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/1</Interface>
# <IP-Address>unassigned</IP-Address>
```

```
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# </IPInterfaces>
# </ShowIpInterfaceBrief>
#
# Define a procedure to be called every time the fully qualified name
# <ShowIpInterfaceBrief><IPInterfaces><entry> tag is processed. Since
# this tag precedes each new output record, the data that is passed into
# this procedure contains the fields that have been requested via
# xml_pi_parse since the previous time this procedure was called.
proc <ShowIpInterfaceBrief><IPInterfaces><entry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
set output [format "Example 4: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
} else {
puts "Example 4: Display All Interfaces"
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show ip interface brief" command. After the command executes $result
# contains the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show ip interface brief"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <Interface> and <Status> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <entry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<ShowIpInterfaceBrief><IPInterfaces><entry> <Interface> <Status>"]
# Display the data from the last record.
incr num
set output [format "Example 4: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
# Example 5:
#
# In the next example we demonstrate how to extract two data elements
# from the multi-record "show ip interface brief" command using the default
# built-in XML-PI spec file. Sample raw output from this command is as
# follows:
#
# Device#show ip interface brief | format
# <?xml version="1.0" encoding="UTF-8"?>
# <ShowIpInterfaceBrief>
# <SpecVersion>built-in</SpecVersion>
# <IPInterfaces>
# <entry>
# <Interface>GigabitEthernet0/1</Interface>
# <IP-Address>172.19.209.34</IP-Address>
```

```
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>up</Status>
# <Protocol>up</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/2</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>GigabitEthernet0/3</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/0</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# <entry>
# <Interface>FastEthernet1/1</Interface>
# <IP-Address>unassigned</IP-Address>
# <OK>YES</OK>
# <Method>NVRAM</Method>
# <Status>administratively down</Status>
# <Protocol>down</Protocol>
# </entry>
# </IPInterfaces>
# </ShowIpInterfaceBrief>
#
# Note: This example is the same as Example 4 with the exception that
# the new record procedure is called by the un-qualified tag name. The
# ability to specify the un-qualified tag names is simplier but only works
# if the un-qualified name is used once per Tcl program. In this example
# the unqualified new record tag name is "<entry>" which is a very
# common name in the Cisco spec file.
# Define a procedure to be called every time the <entry> tag
# is processed. Since this tag precedes each new output record, the data
# that is passed into this procedure contains the fields that have been
# requested via xml_pi_parse since the previous time this procedure was
# called.
proc <entry> {xml_line} {
global num
# The first time that this function is called there is no data and
# xml_line will be null.
if [string length $xml_line] {
array set xml_result $xml_line
incr num
if ([string equal $xml_result(<Status>) "up"]) {
set output [format "Example 5: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
} else {
```

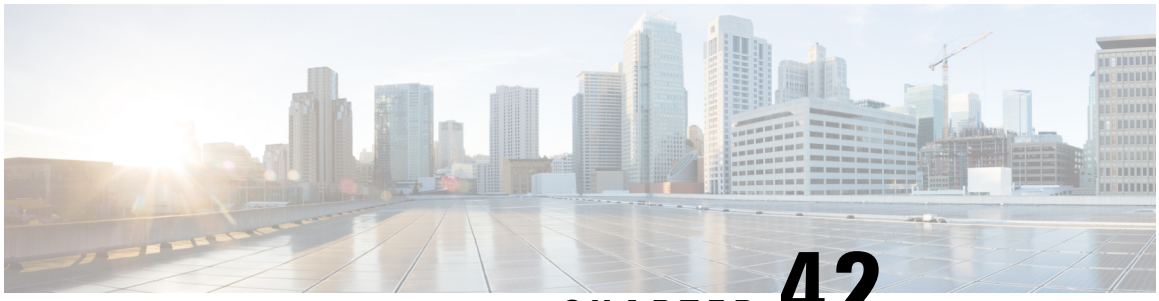
```
puts "Example 5: Display All Interfaces That Are Up"
}
}
set num 0
# Invoke xml_pi_exec with the default built-in spec file for the
# "show ip interface brief" command. After the command executes $result
# contains the raw XML data.
if [catch {xml_pi_exec $cli1(fd) "show ip interface brief"} result] {
error $result $errorInfo
} else {
# Pass the raw XML data to the xml_pi_parse routine to extract fields
# of interest:
# we ask that only the <Interface> and <Status> fields be returned.
# If an XML tag name is requested and a Tcl proc exists with that name,
# the Tcl proc will be called every time that tag is encountered in the
# output data. Specify the <entry> tag and define the proc
# before executing the xml_pi_parse statement.
array set xml_result [xml_pi_parse $cli1(fd) $result \
"<entry> <Interface> <Status>"]
# Display the data from the last record.
incr num
if ([string equal $xml_result(<Status>) "up"]) {
set output [format "Example 5: Interface %2d %-30s %s" \
$num $xml_result(<Interface>) $xml_result(<Status>)]
puts $output
}
}
```

XML-PI 仕様 eemtest.odm ODM ファイルの例

```
###
show version
<?xml version='1.0' encoding='utf-8'?>
<ODMSpec>
<Command>
<Name>show version</Name>
</Command>
<OS>ios</OS>
<DataModel>
<Container name="ShowVersion">
<Property name="Version" distance = "1.0" length = "1" type = "IpAddress"/>
<Property name="Technical Support" distance = "1.0" length = "1" type = "IpAddress"/>
<Property name="Compiled" distance = "1.0" length = "3" type = "String"/>
<Property name="ROM" distance = "1.0" length = "7" type = "IpAddress"/>
<Property name="uptime" distance = "2" length = "8" type = "String"/>
<Property name="image" distance = "4" length = "1" type = "IpAddress"/>
<Property name="processor" distance = "-1" length = "1" type = "String"/>
<Property name="bytes of memory" distance = "-1" length = "1" type = "Port"/>
<Property name="CPU" distance = "2" length = "1" end-delimiter = "," type = "String"/>
<Property name="L2 Cache" distance = "-2" length = "1" end-delimiter = "," type =
"String"/>
<Property name="Gigabit Ethernet interfaces" distance = "-1" length = "1" type =
"Integer"/>
<Property name="bytes of NVRAM" distance = "-1" length = "1" type = "String"/>
<Property name="bytes of ATA PCMCIA card" distance = "-1" length = "1" type = "String"/>
<Property name="Sector size" distance = "1.0" length = "2" end-delimiter = ")" type =
"String"/>
<Property name="bytes of Flash internal SIMM" distance = "-1" length = "1" type =
"String"/>
<Property name="Configuration register" distance = "2" length = "1" type = "String"/>
</Container>
</DataModel>
</ODMSpec>
```

sample.tcl の実行例

```
Device#config t
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)#event manager policy sample.tcl
Device(config)#end
Device#
Oct 10 20:21:26: %SYS-5-CONFIG_I: Configured from console by console
Device#event manager run sample.tcl
Example 1: XML-PI support is present in this image
Example 2: Processor is NPE-G1 CPU is 700MHz
Example 3: Item 1 CISCO7206VXR "Cisco 7206VXR, 6-slot chassis"
Example 3: Item 2 NPE-G1 "Cisco 7200 Series Network Processing Engine NPE-G1"
Example 3: Item 3 MEM-NPE-G1-FLD128 "128MB Compact Flash Disk for NPE-G1"
Example 3: Item 4 PA-2FE-TX "Dual Port FastEthernet (RJ45)"
Example 3: Item 5 PWR-7200-AC "Cisco 7200 AC Power Supply"
Example 4: Display All Interfaces
Example 4: Interface 1 GigabitEthernet0/1 up
Example 4: Interface 2 GigabitEthernet0/2 administratively down
Example 4: Interface 3 GigabitEthernet0/3 administratively down
Example 4: Interface 4 FastEthernet1/0 administratively down
Example 4: Interface 5 FastEthernet1/1 administratively down
Example 4: Interface 6 SSLVPN-VIF0 up
Example 5: Display All Interfaces That Are Up
Example 5: Interface 1 GigabitEthernet0/1 up
Example 5: Interface 6 SSLVPN-VIF0 up
```



CHAPTER 42

EEM コンテキスト ライブラリのコマンド拡張

すべての Tcl コンテキスト ライブラリ コマンド拡張は、::cisco::eem 名前空間に属します。

context_retrieve

該当するコンテキスト名、使用されている可能性があるスカラ変数名、配列型変数名、および配列の索引によって指定される Tcl 変数を取得します。取得される情報は、自動的に削除されます。

Note 保存される情報が一度取得されると、自動的に削除されます。その情報が別のポリシーで必要な場合、（**context_retrieve** コマンド拡張を使用して）それを取得するポリシーも、（**context_save** コマンド拡張を使用して）再度保存する必要があります。

構文

```
context_retrieve ctxt [var] [index_if_array]
```

引数

ctxt	（必須）コンテキスト名。
var	（任意）スカラ変数名または配列型変数名。この引数が指定されない場合、ヌル文字列を定義します。
index_if_array	（任意）配列の索引。

Note var 引数がスカラ変数の場合、index_if_array 引数は無視されます。

var が未指定の場合、コンテキストに保存されている変数テーブル全体を取得します。

var が指定され、index_if_array が指定されない場合、または、index_if_array が指定されるが var がスカラ変数の場合、var の値を取得します。

var が指定され、index_if_array が指定され、var が配列変数の場合、指定された配列エレメントの値を取得します。

結果文字列

保存が実行されたときの状態に、Tcl グローバル変数をリセットします。

_cerrno を設定

- appl_reqinfo エラーが原因で、_cerrno、_cerr_sub_num、_cerr_sub_err、_cerr_posix_err、_cerr_str を表示する文字列。
- 変数がコンテキストにない。

使用例

次に、**context_save** コマンド拡張機能および **context_retrieve** コマンド拡張機能を使用して、データを保存し、取得する例を示します。例は、保存と取得のペアで示されます。

例 1：保存

var が未指定か、またはパターンが指定される場合、複数の変数をコンテキストに保存します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvara 123
set testvarb 345
set testvarc 789
if {[catch {context_save TESTCTX "testvar*"} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 1：取得

var が未指定の場合、複数の変数をコンテキストから取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

if {[catch {foreach {var value} [context_retrieve TESTCTX] {set $var $value}} errmsg]}
```

```
{
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvara]} {
      action_syslog msg "testvara exists and is $testvara"
} else {
      action_syslog msg "testvara does not exist"
}
if {[info exists testvarb]} {
      action_syslog msg "testvarb exists and is $testvarb"
} else {
      action_syslog msg "testvarb does not exist"
}
if {[info exists testvarc]} {
      action_syslog msg "testvarc exists and is $testvarc"
} else {
      action_syslog msg "testvarc does not exist"
}
```

例 2：保存

var が指定される場合、var の値を保存します。

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvar 123
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 2：取得

var が指定され、index_if_array が指定されない場合、または、index_if_array が指定されるが var がスカラ変数の場合、var の値を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar does not exist"
}
```

例 3：保存

var が指定される場合、それが配列の場合でも、var の値を保存します。

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 3：取得

var が指定され、index_if_array が指定されず、var が配列変数の場合、配列全体を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {array set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is [array get testvar]"
} else {
      action_syslog msg "testvar does not exist"
}
```

例 4：保存

var が指定される場合、それが配列の場合でも、var の値を保存します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 4：取得

var が指定され、index_if_array が指定され、var が配列変数の場合、指定された配列エレメントの値を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar testvar1]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
```

```
} else {
      action_syslog msg "testvar doesn't exist"
}
```

context_save

現在およびグローバルな名前空間で、指定されたパターンが、識別情報として指定されたコンテキスト名と一致する、Tcl変数を保存します。このTclコマンド拡張を使用すると、ポリシー外の情報が保存されます。保存された情報は、**context_retrieve**コマンド拡張を使用して、異なるポリシーによって取得できます。

Note 保存される情報が一度取得されると、自動的に削除されます。その情報が別のポリシーで必要な場合、（**context_retrieve** コマンド拡張を使用して）それを取得するポリシーも、（**context_save** コマンド拡張を使用して）再度保存する必要があります。

構文

```
context_save ctxt [pattern]
```

引数

ctxt	（必須）コンテキスト名。
pattern	（任意）**string match** Tcl コマンドによって使用される、glob-style パターン。この引数が指定されない場合、パターンのデフォルトは、ワイルドカード * です。 glob パターンで使用されている、3 つの構成があります。 • * = すべての文字 • ? = 1 文字 • [abc] = 文字のセットの 1 つと照合

結果文字列

なし

_cerrno を設定

appl_setinfo エラーが原因で、_cerrno、_cerr_sub_num、_cerr_sub_err、_cerr_posix_err、_cerr_str を表示する文字列。

使用例

次に、**context_save** コマンド拡張機能および **context_retrieve** コマンド拡張機能を使用して、データを保存し、取得する例を示します。例は、保存と取得のペアで示されます。

例 1：保存

varが未指定か、またはパターンが指定される場合、複数の変数をコンテキストに保存します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set testvara 123
set testvarb 345
set testvarc 789
if {[catch {context_save TESTCTX "testvar*"} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 1：取得

var が未指定の場合、複数の変数をコンテキストから取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

if {[catch {foreach {var value} [context_retrieve TESTCTX] {set $var $value}} errmsg]}
{
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvara]} {
      action_syslog msg "testvara exists and is $testvara"
} else {
      action_syslog msg "testvara does not exist"
}
if {[info exists testvarb]} {
      action_syslog msg "testvarb exists and is $testvarb"
} else {
      action_syslog msg "testvarb does not exist"
}
if {[info exists testvarc]} {
      action_syslog msg "testvarc exists and is $testvarc"
} else {
      action_syslog msg "testvarc does not exist"
}
```

例 2：保存

var が指定される場合、var の値を保存します。

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
```

```
set testvar 123
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 2：取得

var が指定され、index_if_array が指定されない場合、または、index_if_array が指定されるが var がスカラ変数の場合、var の値を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar does not exist"
}
```

例 3：保存

var が指定される場合、それが配列の場合でも、var の値を保存します。

```
::cisco::eem::event_register_none

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 3：取得

var が指定され、index_if_array が指定されず、var が配列変数の場合、配列全体を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {array set testvar [context_retrieve TESTCTX testvar]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is [array get testvar]"
} else {
      action_syslog msg "testvar does not exist"
}
```

例 4：保存

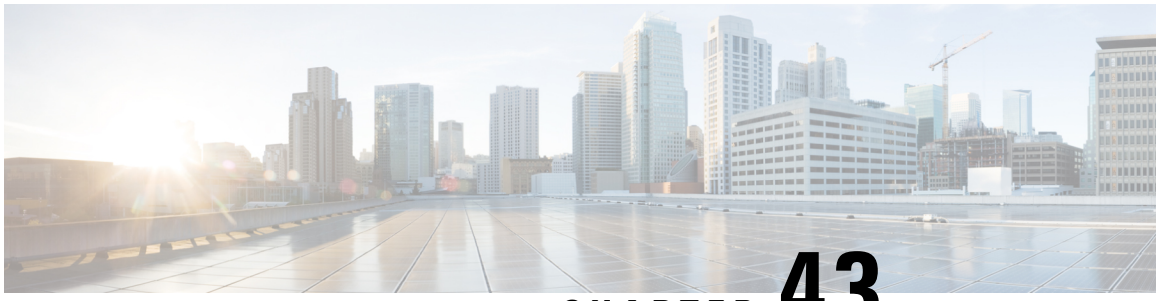
var が指定される場合、それが配列の場合でも、var の値を保存します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
array set testvar "testvar1 ok testvar2 not_ok"
if {[catch {context_save TESTCTX testvar} errmsg]} {
      action_syslog msg "context_save failed: $errmsg"
} else {
      action_syslog msg "context_save succeeded"
}
```

例 4：取得

var が指定され、index_if_array が指定され、var が配列変数の場合、指定された配列エレメントの値を取得します。

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
if {[catch {set testvar [context_retrieve TESTCTX testvar testvar1]} errmsg]} {
      action_syslog msg "context_retrieve failed: $errmsg"
} else {
      action_syslog msg "context_retrieve succeeded"
}
if {[info exists testvar]} {
      action_syslog msg "testvar exists and is $testvar"
} else {
      action_syslog msg "testvar doesn't exist"
}
```



CHAPTER 43

EEM イベント登録の Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

event_register_appl

アプリケーション イベントの登録を行います。この Tcl コマンド拡張は、**event_publish** Tcl コマンド拡張の別のポリシーの実行に続いて、アプリケーションイベントがトリガされたときにポリシーを実行するために使用します。**event_publish** コマンド拡張によって、アプリケーションイベントがパブリッシュされます。

アプリケーション イベントを登録するためには、サブシステムを指定する必要があります。Tcl ポリシーまたは内部 Embedded Event Manager（EEM）API のいずれかによって、アプリケーションイベントをパブリッシュできます。イベントがポリシーによってパブリッシュされている場合、ポリシーで予約される sub_system 引数は 798 です。

構文

```
event_register_appl [tag ?] sub_system ? type ? [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
sub_system	（必須）アプリケーション イベントをパブリッシュした EEM ポリシーに割り当てられる番号。他のすべての番号は Cisco での使用のために予約されているため、番号は 798 に設定されます。この引数が指定されない場合、すべてのコンポーネントが照会されます。

type	（必須）指定されたイベント内のイベント サブタイプ。sub_system 引数および type 引数によって、アプリケーション イベントが一意に識別されます。この引数が指定されない場合、すべてのタイプが照会されます。この引数を指定する場合、1 ～ 4294967295 の整数を選択する必要があります。 パブリッシュと登録が機能するためには、**event_publish** コマンド拡張と **event_register_appl** コマンド拡張の間でコンポーネントとタイプが一致する必要があります。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

複数の条件が存在する場合、すべての条件が満たされたときに、アプリケーションイベントが発生します。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"sub_system 0x%x type %u data1 {%s} data2 {%s} data3 {%s} data4 {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	イベントが Embedded Event Manager（EEM）にパブリッシュされたときの、秒単位およびミリ秒単位の時間。
sub_system	アプリケーション イベントをパブリッシュした EEM ポリシーに割り当てられる番号。他のすべての番号は Cisco での使用のために予約されているため、番号は 798 に設定されます。
type	指定されたコンポーネント内のイベント サブタイプ。
data1 data2 data3 data4	イベントがパブリッシュされるときに、アプリケーション固有のイベントに渡される、引数データ。データは、文字テキスト、環境変数、または、この 2 つの組み合わせです。

event_register_cli

CLI イベントの登録を行います。この Tcl コマンドを使用すると、拡張 CLI コマンドに対して実行されるパターン マッチに基づいて、特定パターンの CLI コマンドが入力されるときに、ポリシーが実行されます。

Note ユーザーは、**sh mem summary** などの省略形の CLI コマンドを入力できます。パーサーによってコマンドが **show memory summary** に拡張され、照会が実行されます。

Note CLI イベント ディテクタによる機能は、有効な IOS CLI コマンドでの正規表現パターン比較機能だけです。これには、リダイレクションが使用される場合のパイプ記号（|）以降のテキストは含まれません。

構文

```
event_register_cli [tag ?] sync yes|no skip yes|no
[occurs ?] [period ?] pattern ? [default ?] [enter] [questionmark] [tab] [mode]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
sync	（必須）「yes」は、ポリシー（イベント パブリッシュ）が、CLI コマンドと同期的に実行することを意味します。「no」は、イベント パブリッシュが CLI コマンドと非同期に実行されることを意味します。ポリシーの実行が完了すると、イベントディテクタによって通知されます。ポリシーの終了ステータスは、CLI コマンドを実行する必要があるかどうかを示します。終了ステータスがゼロの場合は、ポリシーが正常に実行されたことを意味し、CLI コマンドは実行されません。それ以外の場合は、CLI コマンドが実行されます。
skip	sync 引数が no の場合は必須で、sync 引数が yes の場合は不要ですskip 引数が yes の場合、CLI コマンドを実行する必要がないことを意味します。skip 引数が no の場合、CLI コマンドを実行する必要があることを意味します。 **Caution** skip 引数が yes の場合、パターン マッチがコンフィギュレーション コマンドに対して行われる場合、正規表現に一致する CLI コマンドは実行されないため、想定外の結果が生成される場合があります。
occurs	（任意）イベントが発生する前の発生回数。この引数が指定されない場合、イベントは 1 回目から発生します。この引数が指定される場合は、1 ～ 4294967295 の範囲の整数である必要があります。
period	（任意）イベントがパブリッシュされるようにするために、すべての CLI イベントが発生する必要がある（occurs 句を満たす必要がある）逆方向検索時間ウィンドウを指定します（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のイベントが使用されます。
pattern	（必須）CLI コマンド パターン マッチの実行に使用される正規表現を指定します。

デフォルト	（任意）CLI イベント ディテクタがポリシーの終了を待つ時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。ポリシーが終了する前にデフォルトの時間の期限が切れると、デフォルト アクションが実行されます。デフォルト アクションによって、コマンドが実行されます。この引数が指定されない場合、デフォルトの時間は 30 秒に設定されます。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
enter	（任意）ユーザーが Enter キーを押したときにイベント照会が実行されるよう、指定します。このパラメータが使用されると、照会前には入力文字列は拡張されません。
questionmark	（任意）ユーザーが ? キーを押したときにイベント照会が実行されるよう、指定します。このパラメータが使用されると、照会前には入力文字列は拡張されません。
タブ	（任意）ユーザーが Tab キーを押したときにイベント照会が実行されるよう、指定します。このパラメータが使用されると、照会前には入力文字列は拡張されません。
mode	（任意）パーサーが指定されたパーサー モードの場合のみ、イベントが生成されます。使用可能なモードのリストは、**show parser dump** CLI コマンドを使用して表示できます。オプション パラメータの enter、questionmark、または tab のいずれか 1 つが指定されている場合、mode パラメータが確認されます。

maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

複数の条件が存在する場合、すべての条件が一致したときに、CLI イベントが発生します。

結果文字列

なし

_cerrno を設定

なし

Note このポリシーは、CLI コマンドの実行前に実行されます。たとえば、**copy** コマンドが入力されると、policy_CLI が実行のために登録されるとします。**copy** コマンドが入力されると、CLI イベントディテクタがパターンの一致を検出し、このポリシーの実行がトリガーされます。ポリシーの実行が終了すると、CLI イベントディテクタは、「sync」、「skip」（ポリシーで設定）、および、必要に応じてポリシー実行の終了ステータスに従って、**copy** コマンドを実行する必要があるかどうかを判断します。

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u msg {%s} msg_count %d line %u key %u tty %u error_code %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
msg	CLI プロンプトで入力されるテキスト。

イベント タイプ	説明
msg_count	イベントがトリガーされる前にパターン マッチされた回数。
line	一致したキーが入力されたポイントまで、パーサーによって拡張できたテキスト。
key	Enter キー、疑問符、または Tab キー。
tty	ユーザーがコマンドを実行する行番号に対応します。
error_code	CLI のエラー コード。 0：パーサーからキーが入力されたポイントまで、エラーはありません。 1：キーが入力されたポイントまで、コマンドはあいまいです。 4：キーが入力されたポイントまで、未知のコマンドです。

event_register_counter

パブリッシャとサブスクライバの両方として、カウンタ イベントの登録を行います。この Tcl コマンド拡張を使用すると、しきい値に近くなった名前付きカウンタに基づいて、ポリシーが実行されます。サブスクライバとして、このイベントカウンタによって、登録に必要なカウンタの名前が指定され、別のポリシーまたは別のプロセスに依存して、カウンタが実際に操作されます。たとえば、policyB をカウンタポリシーとして動作させる一方、policyA（カウンタ ポリシーである必要はない）では、**register_counter**、**counter_modify**、または **unregister_counter** の各 Tcl コマンド拡張を使用して、policyB で定義されているカウンタを操作します。

構文

```
event_register_counter [tag ?] name ? entry_op gt|ge|eq|ne|lt|le entry_val ?
exit_op gt|ge|eq|ne|lt|le exit_val ? [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
name	（必須）カウンタの名前。
entry_op	（必須）現在のカウンタの値を開始値と比較するために使用される開始比較演算子。真の場合、イベントが発生し、終了基準を満たすまでイベントモニターリングがディセーブルにされます。

entry_val	（必須）カウンタ イベントを発生させる必要があるかどうかを判断するために、現在のカウンタの値を比較する必要がある値。
exit_op	（必須）現在のカウンタの値を終了値と比較するために使用される終了比較演算子。真の場合、このイベントのイベントモニターリングが再度イネーブルにされます。
exit_val	（必須）終了基準を満たすかどうかを判断するために、現在のカウンタの値を比較する必要がある値。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"name {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
name	カウンタ名。

event_register_gold

Generic Online Diagnostic（GOLD）障害イベントの登録を行います。この Tcl コマンド拡張を使用すると、指定されたカードおよびサブカードの Generic Online Diagnostic（GOLD）障害イベントに基づいて、ポリシーが実行されます。

構文

```
event_register_gold card all|card_number
[subcard all|subcard_number]
[new_failure TRUE|FALSE]
[severity_major TRUE]
[severity_minor TRUE]
[severity_normal TRUE]
[action_notify TRUE|FALSE]
[testing_type [bootup|ondemand|schedule|monitoring]]
[test_name [testname]]
[test_id [testnumber]]
[consecutive_failure consecutive_failure_number]
[platform_action [action_flag]]
[maxrun ?]
[queue_priority low|normal|high|last]
[nice 0|1]
```

引数

card	（必須）すべてのカードまたは 1 つのカードがモニターされるよう指定します。 • card all：すべてのカードを監視対象に指定します。これはデフォルトです。 • card-number：card-number の番号によって指定されたカードを監視対象に指定します。 **event_register_gold** Tcl コマンド拡張を完了させるには、この引数を指定する必要があります。
subcard	（任意）1 つまたは複数のサブカードがモニターされるよう指定します。 • subcard all：すべてのサブカードを監視対象に指定します。 • subcard-number：subcard-number の番号によって指定されたサブカードを監視対象に指定します。 この引数が指定されない場合、すべてのサブカードがデフォルトでモニターされます。
new_failure	（任意）GOLD からの新しいテスト障害情報に基づいて、イベント基準を指定します。 • new_failure TRUE：GOLD からの新しいテスト障害のイベント基準が真であると指定します。 • new_failure FALSE：GOLD からの新しいテスト障害のイベント基準が偽であると指定します。 この引数が指定されない場合、GOLD からの新しいテスト障害情報は、イベント基準で考慮されません。
severity_major	（任意）診断結果のイベント基準が、GOLD からの診断（メジャーエラー）と合致するよう指定します。
severity_minor	（任意）診断結果のイベント基準が、GOLD からの診断（マイナーエラー）と合致するよう指定します。
severity_normal	（任意）診断結果のイベント基準が、GOLD からの診断（通常）と合致するよう指定します。これはデフォルトです。

action_notify	（任意）GOLD からのアクション通知情報に基づいて、イベント基準を指定します。 • action_notify TRUE：GOLD からのアクション通知のイベント基準が真であると指定します。 • action_notify FALSE：GOLD からのアクション通知のイベント基準が偽であると指定します。 この引数が指定されない場合、GOLD からのアクション通知情報は、イベント基準で考慮されません。
testing_type	（任意）GOLD からの診断のテスト タイプに基づいて、イベント基準を指定します。 • testing_type bootup：システム ブート時に実行される診断テストを指定します。 • testing_type ondemand：カードがオンライン後に CLI から実行される診断テストを指定します。 • testing_type schedule：スケジュールされる診断テストを指定します。 • testing_type monitoring：システムの状態を監視するためにバックグラウンドで定期的に実行される診断テストを指定します。 この引数が指定されない場合、GOLD からのテスト タイプ情報は、イベント基準で考慮されず、ポリシーは、すべての診断テスト タイプに適用されます。
test_name	（任意）名前 test-name でのテストに基づいて、イベント基準を指定します。 • test_name test-name：名前 test-name でのテストに基づいて、イベント基準を指定します。 この引数が指定されない場合、GOLD からのテスト名情報は、イベント基準で考慮されません。

test_id	（任意）テスト ID に基づいて、イベント基準を指定します。 • test_id test-id：ID 番号 test-id のテストに基づいてイベント基準を指定します。test-id の最大値は 65535 です。 **Note** テスト ID は、異なるラインカード上での同じテストについて、異なる可能性があるため、通常は、代わりに test_name キーワードを使用する必要があります。テスト ID が指定され、指定されたテスト名と矛盾する場合、テスト名によって、テスト ID が上書きされます。 この引数が指定されない場合、GOLD からのテスト ID 情報は、イベント基準で考慮されません。
consecutive_failure	（任意）GOLD からの連続テスト障害情報に基づいて、イベント基準を指定します。 • consecutive_failure consecutive-failure-number：イベント障害が、consecutive-failure-number 連続テスト障害の発生に基づくよう、指定します。 この引数が指定されない場合、GOLD からの連続テスト障害情報は、イベント基準で考慮されません。
platform_action	（任意）すべてのイベント基準が一致した場合に、プラットフォームへのコールバックが必要かどうかを指定します。コールバックが必要な場合、プラットフォームでは、指定されたレジストリを介してコールバック機能を登録する必要があります。 • platform_action action-flag-number：プラットフォームへのコールバックが必要な場合に、特定の情報がプラットフォーム特有の action-flag-number の値によって指定されるよう、指定します。action-flag-number の最大値は 65535 です。 **Note** プラットフォームにより、フラグに基づいて行われる必要があるアクションが判断されます。 この引数が指定されない場合、コールバックはありません。
maxrun	（任意）スクリプトの最大実行時間を指定します。 • maxrun max-run-time-number：スクリプトの最大実行時間を、max-run-time-number 秒と指定します。max-run-time-number の最大値は 4294967295 秒です。 この引数が指定されない場合、デフォルトの実行時間は 20 秒です。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
nice	（任意）次のような、ポリシー実行時間のプライオリティ設定。 • nice 0：ポリシーがデフォルトの実行時間優先度レベルで実行されるよう指定します。 • nice 1：ポリシーがデフォルト優先度レベルよりも低い実行時間優先度で実行されるよう指定します。 この引数が指定されない場合、デフォルトの実行時間プライオリティが使用されます。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u card %u sub_card %u"
"event_severity {%s} event_pub_sec %u event_pub_msec %u overall_result %u"
"new_failure {%s} action_notify {%s} tt %u tc %u bl %u ci %u pc %u cn {%s}"
"sn {%s} tn# {%s} ta# %s ec# {%s} rc# %u lf# {%s} tf# %u cf# %u tr# {%s}"
"tr#p# {%s} tr#d# {%s}"
```

イベント タイプ	説明
action_notify	TRUE または FALSE の、GOLD イベントでのアクション通知情報。
bl	起動診断レベル、次のいずれかの値である。 • 0：完全診断 • 1：最小診断 • 2：バイパス診断
card	GOLD イベントのカード情報。
cf *testnum*	連続的な障害。*testnum* はテスト番号。たとえば、**cf3** は、テスト 3 の連続的な障害の EEM 組み込み環境変数です。
ci	カード インデックス。
cn	カードの名前。
ec *testnum*	テスト エラー コード。*testnum* はテスト番号。たとえば、**ec3** は、テスト 3 のエラー コードの EEM 組み込み環境変数です。
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_pub_msec **event_pub_sec**	イベントが EEM にパブリッシュされたときの、ミリ秒単位および秒単位の時間。
event_severity	GOLD イベントの重大度で、次のいずれかの値を指定できます。 • normal • minor • major
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
lf *testnum*	最終障害時間。*testnum* はテスト番号。たとえば、**lf3** は、テスト 3 の最終障害時間の EEM 組み込み環境変数です。 タイムスタンプの形式は *mmm dd yyyy hh*:*mm*:*ss* です。例：Mar 11 1960 08:47:00。
new_failure	GOLD イベント フラグの新しいテスト障害情報（False または True）。

イベント タイプ	説明
overall_result	総合診断結果、次のいずれかの値である。 • 0：OK • 3：マイナー エラー • 4：メジャー エラー • 14：結果不明
pc	ポート数。
rc *testnum*	テスト総実行回数。*testnum* はテスト番号。たとえば、**rc3** は、テスト 3 の総実行回数の EEM 組み込み変数です。
sn	カード シリアル番号。
sub_card	GOLD 障害イベントが検出されたサブカード。
ta *testnum*	テスト属性名。*testnum* はテスト番号。たとえば、**ta3** は、テスト 3 の属性の EEM 組み込み環境変数です。
tc	テスト数。
tf *testnum*	合計障害回数。*testnum* はテスト番号。たとえば、**tf3** は、テスト 3 の合計障害回数の EEM 組み込み変数です。
tn *testnum*	テストの名前。*testnum* はテスト番号。たとえば、**tn3** は、テスト 3 の名前の EEM 組み込み環境変数です。
tr *testnum*	テストの結果。*testnum* はテスト番号。たとえば、**tr6** はテスト 6 用の EEM 組み込み変数です。テスト 6 はポート単位のテストでも、デバイス単位のテストでもありません。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown

イベント タイプ	説明
tr *testnum* **d** *devnum*	デバイスごとのテスト結果。*testnum* はテスト番号で、*devnum* はデバイス番号です。たとえば、**tr3d20** は、テスト 3、デバイス 20 のテスト結果の EEM 組み込み環境変数です。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown
tr *testnum* **p** *portnum*	ポートごとのテスト結果。*testnum* はテスト番号で、*portnum* はデバイス番号です。たとえば、**tr5p20** は、テスト 5、ポート 20 のテスト結果の EEM 組み込み環境変数です。 テスト結果は、次の値のうちのいずれかです。 • P：診断結果 Pass • F：診断結果 Fail • U：診断結果 Unknown
tt	テストのタイプ。次のうちのいずれかです。 • 1：起動診断 • 2：オンデマンド診断 • 3：スケジュール診断 • 4：モニターリング診断

event_register_identity

ID イベントの登録を行います。この Tcl コマンド拡張を使用して、AAA 認証または許可が成功または失敗したときや、ポート上での通常のユーザー トラフィックのフローが許可された後にイベントを生成します。

構文

```
event_register_identity [tag ?] interface ?
[aaa-attribute ?]
[authc {all | fail | success}]
[authz {all | fail | success}]
[authz-complete]
[mac-address ?]
```

```
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
interface	インターフェイス名と照合する正規表現パターン。
aaa-attribute	（任意）特定の AAA 属性によってイベントをフィルタリングするために使用可能な正規表現。
authc	（任意）成功した認証、失敗した認証、または成功と失敗の両方の認証で、イベントをトリガーします。
authz	（任意）成功した許可、失敗した許可、または成功と失敗の両方の許可で、イベントをトリガーします。
authz-complete	（任意）インターフェイスに接続されたデバイスが完全に認証、許可され、通常のトラフィックがそのインターフェイスで流れ始めたときにイベントをトリガーします。
mac-address	（任意）リモートデバイスの MAC アドレスによってイベントをフィルタリングするために使用可能な正規表現パターン。
maxrun	（任意）スクリプトの最大実行時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 31536000 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

EEM_EVENT_IDENTITY の Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u identity_stage %u identity_status %u interface %u identity_mac %u
identity_<attribute> {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。

イベント タイプ	説明
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
identity_stage	authentication、authentication、または autherization-complete のステージのうちのいずれか。
identity_status	Success または fail_authc、fail_aaa_server、fail_no_response、fail_timeout、fail_authz のいずれかの障害タイプ。 autherization-complete は常に success になります。
interface	イベントのインターフェイス。
identity_mac	イベントのリモート デバイスの MAC アドレス。
identity_<attribute>	属性リストまたは値リスト内のその AAA 属性に対応する値に対する AAA 属性ごとの一連のダイナミック変数。

event_register_interface

インターフェイス カウンタ イベントの登録を行います。この Tcl コマンド拡張を使用すると、指定されたインターフェイス カウンタが指定されたしきい値を超えたときに、イベントが生成されます。

構文

```
event_register_interface [tag ?] name ?
parameter ? entry_op gt|ge|eq|ne|lt|le
entry_val ? entry_val_is_increment TRUE|FALSE
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le]
[exit_val ?] [exit_val_is_increment TRUE|FALSE]
[exit_type value|increment|rate]
[exit_time ?] [poll_interval ?]
[average_factor ?] [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

name	（必須）イーサネット 0/0 など、モニターされるインターフェイスの名前。省略形と空白は使用できません。
parameter	（必須）比較されるカウンタの名前は、次のとおりです。 • input_errors：ラント、ジャイアント、バッファなし、CRC、フレーム、オーバーラン、および無視されたカウントが含まれます。他の入力関連のエラーも、入力エラー カウントが大きくなる場合があります。一部のデータグラムには、複数のエラーがあります。したがって、この合計は、列挙型入力エラー カウントとのバランスが取れない場合があります。 • input_errors_crc：発信元 LAN ステーションまたは遠隔エンド デバイスによって生成される巡回冗長検査が、受信したデータから計算されるチェックサムに一致しません。 • input_errors_frame：受信した不正確なパケット数。CRC エラーが発生し、8 ビットの非整数の数です。 • input_errors_overrun：入力レートが、レシーバのデータ処理能力を超えたために、レシーバ ハードウェアによって、受信データをハードウェア バッファに渡せなかった回数。 • input_packets_dropped：入力キューがいっぱいのため、廃棄されたパケット数。 • interface_resets：インターフェイスが完全にリセットされた回数。 • output_buffer_failures：障害が発生したバッファ数およびスワップされたバッファ数。 • output_buffer_swappedout：DRAM にスワップされたパケット数。

parameter（続き）	• output_errors：調べられているインターフェイスからのデータグラムの最終的な送信が妨害されたすべてのエラーの合計。一部のデータグラムには、複数のエラーがある場合があり、また、他のデータグラムには、特に表形式のカテゴリに当てはまらないエラーがある場合があるため、これは、列挙型出力エラーの合計とのバランスが取れないことがあります。 • output_errors_underrun：トランスミッタが、デバイスが処理可能な速度よりも高速だった回数。 • output_packets_dropped：出力キューがいっぱいのため、廃棄されたパケット数。 • receive_broadcasts：インターフェイスによって受信されたブロードキャスト パケットまたはマルチキャスト パケットの数。 • receive_giants：メディアの最大パケット サイズを超過したために廃棄されたパケット数。 • receive_rate_bps：1 秒あたりのバイト単位でのインターフェイス受信レート。 • receive_rate_pps：1 秒あたりのパケット単位でのインターフェイス受信レート。 • receive_runts：メディアの最小パケット サイズよりも小さいために廃棄されたパケット数。 • receive_throttle：バッファまたはプロセッサが過負荷などの理由で、ポート上のレシーバが無効にされた回数。 • reliability：5 分間の幾何平均で計算される、255 の分数でのインターフェイスの信頼性（255/255 が 100% の信頼性）。 • rxload：255 の分数でのインターフェイスの受信レート（255/255 が 100%）。 • transmit_rate_bps：1 秒あたりのバイト単位でのインターフェイス送信レート。 • transmit_rate_pps：1 秒あたりのパケット単位でのインターフェイス送信レート。 • txload：255 の分数でのインターフェイスの送信レート（255/255 が 100%）。
entry_op	（必須）現在のインターフェイスの値を開始値と比較するために使用される比較演算子。真の場合、イベントが発生し、終了基準を満たすまでイベント モニターリングがディセーブルにされます。
entry_val	（必須）イベントがトリガーされる値。

entry_val_is_increment	（必須）TRUE の場合、entry_val フィールドは増分差異として処理され、現在のカウンタの値とイベントが最後に真であったとき（これが新しいイベントの場合は最初にポーリングされたサンプル）の値との差異と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。FALSE の場合、entry_val フィールドが現在のカウンタの値に対して比較されます。 **Note** このキーワードは廃止されました。これを指定した場合、その構文は同等な entry-type キーワード構文に変換されます。
entry-type	entry-val 引数によって指定されたオブジェクト ID に適用される操作のタイプを指定します。 値は、entry-val 引数の実際の値として定義されます。 増分では、entry-val フィールドは増分差異として使用され、entry-val は、現在のカウンタの値と、イベントが最後に真であったとき（これが新しいイベントの場合は最初にポーリングされたサンプル）の値との間の差と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。 レートは、ある期間の変更の平均レートとして定義されます。期間は、average-factor の値に、poll-interval の値を乗じたものです。ポーリング間隔ごとに、現在のサンプルと前のサンプルとの間の差が取得され、絶対値として記録されます。前の average-factor 値サンプルの平均は、変更のレートとして取得されます。
exit_comb	（任意）イベント トリガーの再準備に必要な終了条件テストの組み合わせを示すために使用されます。and 演算子が指定される場合、再準備のためには、終了値と終了時間テストの両方が真である必要があります。or 演算子が指定される場合、イベント モニターリングの再準備のためには、終了値または終了時間テストのいずれかが真である可能性があります。
exit_op	（任意）現在のインターフェイスの値を終了値と比較するために使用される比較演算子。真の場合、このイベントのイベント モニターリングが再度イネーブルにされます。
exit_val	（任意）イベントが再度監視されるように再準備される値。

exit_val_is_increment	（任意）TRUE の場合、exit_val フィールドは増分差異として処理され、現在のカウンタの値と、イベントが最後に真であったときの値との差異と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。FALSE の場合、exit_val フィールドが現在のカウンタの値に対して比較されます。 **Note** Cisco IOS Release 12.4(20)T では、このキーワードは廃止予定で、指定された場合、構文は同等の exit-type キーワード構文に変換されます。
exit-type	（任意）exit-val 引数によって指定されたオブジェクト ID に適用される操作のタイプを指定します。指定されない場合、値が仮定されます。 値は、exit-val 引数の実際の値として定義されます。 増分では、exit-val フィールドは増分差異として使用され、exit-val は、現在のカウンタの値と、イベントが最後に真であったとき（これが新しいイベントの場合は最初にポーリングされたサンプル）の値との間の差と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。 レートは、ある期間の変更の平均レートとして定義されます。期間は、average-factor の値に、poll-interval の値を乗じたものです。ポーリング間隔ごとに、現在のサンプルと前のサンプルとの間の差が取得され、絶対値として記録されます。前の average-factor 値サンプルの平均は、変更のレートとして取得されます。
exit_time	（任意）イベントが再度監視されるように再準備される時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。
poll_interval	（任意）サンプルが収集される頻度（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、60 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。ポーリング間隔の値には、1 秒よりも小さい値は使用できません。デフォルト値は 1 秒です。
average-factor	（任意）レート ベースの計算に使用される期間の計算に使用される 1 から 64 の範囲の数。average-factor の値は、poll-interval の値を乗じた値で、ミリ秒単位で導き出される期間です。最少平均係数値は 1 です。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} name {%s} parameter {%s} value %d"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	インターフェイスイベントの重大度で、次のいずれかの値を指定できます。 • normal • minor • major
name	インターフェイスの名前。
parameter	パラメータの名前。
value	指定された entry_val_is_increment によって、トリガーされた最後のイベントに対する増加または減少の差異、または、監視されているパラメータの絶対値。

event_register_ioswdsysmon

IOSWDSysMon イベントの登録を行います。この Tcl コマンド拡張を使用すると、Cisco IOS タスクが指定された CPU 使用率またはメモリしきい値を超えたときに、イベントが生成されます。Cisco IOS タスクは、ネイティブ Cisco IOS の Cisco IOS プロセスと呼ばれます。

構文

```
event_register_ioswdsysmon [tag ?] [timewin ?] [sub12op and|or] [sub1 ?] [sub2 ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

timewin	（任意）イベントが生成されるようにするために、すべてのサブイベントが発生する必要がある時間ウィンドウを定義します（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。
sub12_op	（任意）サブイベント 1 とサブイベント 2 とを比較する組み合わせ演算子。
sub1	（任意）サブイベント 1 の指定。
sub2	（任意）サブイベント 2 の指定。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

サブイベントの構文

```
cpu_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [period ?]
mem_proc path ? taskname ? op gt|ge|eq|ne|lt|le val ? [is_percent TRUE|FALSE] [period
?]
```

サブイベントの引数

cpu_proc	（必須）CPU 統計情報のサンプル収集の使用を指定します。
path	（必須）ソフトウェア モジュール方式イメージのみ。監視される Cisco IOS スケジューラが含まれる POSIX プロセスのパス名。たとえば、/sbin/cdp2.iosproc など。
taskname	（必須）監視される Cisco IOS タスクの名前。
op	（必須）収集される使用サンプルを指定値と比較するために使用される比較演算子。真の場合、このイベントが発生します。
val	（必須）比較される値。
period	（任意）収集されるサンプルの平均が計算される経過時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。
mem_proc	（必須）メモリ統計情報のサンプル収集の使用を指定します。
is_percent	（任意）指定値がパーセンテージかどうか。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
num_subs	サブイベントの番号。

サブイベント情報文字列は、次のような、CPU_UTIL サブイベント用です。

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u value %u sec %ld msec %ld}"
```

サブイベントタイプ	説明
type	サブイベントのタイプ。
procname	このサブイベントの POSIX プロセス名。
pid	このサブイベントの POSIX プロセス ID。
taskname	このサブイベントの Cisco IOS タスク名。
taskid	このサブイベントの Cisco IOS タスク ID。
value	測定された間隔での、実際の平均CPU使用率。
sec , **msec**	この測定間隔の経過時間。

サブイベント情報文字列は、次のような、MEM_UTIL サブイベント用です。

```
"{type %s procname {%s} pid %u taskname {%s} taskid %u is_percent %s value %u diff %d"
"sec %ld msec %ld}"
```

サブイベントタイプ	説明
type	サブイベントのタイプ。
procname	このサブイベントの POSIX プロセス名。
pid	このサブイベントの POSIX プロセス ID。
taskname	このサブイベントの Cisco IOS タスク名。
taskid	このサブイベントの Cisco IOS タスク ID。
is_percent	値がパーセント値かどうかによって、TRUE または FALSE。
value	この測定された間隔の KB 単位でのメモリ使用量の合計、または実際のメモリ使用率の平均。
diff	測定された間隔で最も古いサンプルと、最新のサンプルとの、パーセンテージでの違い。負の値は、減少を表します。
sec , **msec**	この測定間隔の経過時間。

event_register_ipsla

event ipsla コマンドによってトリガーされるイベントの登録を行います。この Tcl コマンド拡張を使用すると、IPSLA の応答がトリガーされるときに、イベントがパブリッシュされます。イベントの登録には、グループ ID または動作 ID が必要です。

構文

```
event_register_ipsla [tag ?] group_name ? operation_id ? [reaction_type ?]
[dest_ip_addr ?][queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
group_name	（必須）IP SLA グループ名を指定します。
operation_id	（必須）IP SLA 動作 ID を指定します。番号は 1 から 2147483647 の範囲の整数です。

reaction_type	（任意）指定した IP SLA 動作に対する応答を指定します。 IP SLA 反応のタイプ：次のキーワードのいずれかを指定できます：**connectionLoss**、 **icpif**、 **jitterAvg**、 **jitterDSAvg**、 **jitterSDAvg**、 **maxOfNegativeDS**、 **maxOfNegativeSD**、 **maxOfPositiveDS**、 **maxOfPositiveSD**、 **mos**、 **packetLateArrival**、 **packetLossDS**、 **packetLossSD**、 **packetMIA**、 **packetOutOfSequence**、 **rtt**、 **timeout** または **verifyError** を指定できます。 IP SLA の応答。次のキーワードの 1 つを指定できます。 • connectionLoss • icpif • jitterAvg • jitterDSAvg • jitterSDAvg • maxOfNegativeDS • maxOfNegativeSD • maxOfPositiveDS • maxOfPositiveSD • mos • packetLateArrival • packetLossDS • packetLossSD • packetMIA • packetOutOfSequence • rtt • timeout • verifyError
dest_ip_address	（任意）IP SLA イベントが監視される宛先ポートの宛先 IP アドレスを指定します。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大実行時間（SSSSSSSSSS[.MMM]形式で指定します。SSSSSSSSSS は、0 ～ 31536000 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

"event_ID %u event_type %u event_pub_sec %u event_pub_msec %u event_severity %u" "group_name %u operation_id %u condition %u reaction_type %u dest_ip_addr %u" "threshold_rising %u threshold_falling%u measured_threshold_value %u" "threshold_count1 %u threshold count2 %u"

Event Type	Description
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	フローの作成、アップデート、および削除を監視するイベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
group_name	IPSLA グループの名前。
operation_id	IPSLA 動作 ID。
condition	IPSLA の条件で、次の 1 つを使用できます。 • cleared • occurred
reaction_type	IPSLA 応答タイプ。
dest_ip_address	IPSLA 宛先 IP アドレス。
threshold rising	IPSLA で設定されている上昇しきい値。
threshold falling	IPSLA で設定されている下限しきい値。
measured_threshold_value	IPSLA 動作の測定されたしきい値。
threshold_count1	しきい値 type1 の引数に対応します。
threshold_count2	しきい値 type2 の引数に対応します。

event_register_mat

MATイベントの登録を行います。このTclコマンド拡張を使用して、mac-address-tableでMACアドレスが学習されたときにイベントを生成します。

構文

```
event_register_identity [tag ?] interface ?
```

```
[mac-address ?]
[type {add | delete}]
[hold-down ?]
[maxrun ?]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
interface	インターフェイス名と照合する正規表現パターン。
mac-address	インターフェイス パラメータを指定していない場合には必須です。リモート デバイスの MAC アドレスによってイベントをフィルタリングするために使用可能な正規表現パターン。
type	（任意）追加または削除の mac-address-table イベント タイプに基づいてフィルタリングします。指定しなかった場合、イベントをトリガーするかどうかの判断にそのイベント タイプが使用されません。
hold-down	（任意）mac-address-table イベントが着信した場合、ポリシーを処理する前にそのイベントを 1 ～ 4294967295 秒間待機させるようにホールドダウン タイマーを設定できます。このタイマーを設定しなかった場合は、ポリシーの処理は遅延しません。
maxrun	（任意）スクリプトの最大実行時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 31536000 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

結果文字列

なし

_cerrno を設定

なし

EEM_EVENT_MAT の Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u notification %u intf_name %u mac_address {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。

イベント タイプ	説明
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
notification	通知のタイプ：追加または削除。
intf_name	アドレス テーブル エントリのインターフェイス名。
mac_address	アドレス テーブルのエントリの MAC アドレス。

event_register_neighbor_discovery

ネイバー探索イベントの登録この Tcl コマンド拡張を使用して、Cisco Discovery Protocol（CDP）または Link Layer Discovery Protocol（LLDP）のキャッシュ エントリまたはインターフェイス リンク ステータスが変わった場合にイベントを生成します。

構文

```
event_register_neighbor_discovery [tag ?] interface ?
[cdp {add | update | delete | all}]
[lldp {add | update | delete | all}]
[link-event]
[line-event]
[queue_priority {normal | low | high | last}]
[maxrun ?] [nice {0 | 1}]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
interface	インターフェイス名と照合する正規表現パターン。

cdp	CDP のマッチング イベント発生時にイベントをトリガーします。次のオプションのいずれかを指定する必要があります。 • add：新しい CDP キャッシュ エントリが CDP テーブルに作成された場合にイベントをトリガーします。 • all：CDP キャッシュ エントリが CDP キャッシュ テーブルに追加された場合、または削除された場合、および CDP キャッシュ エントリをアップデートするためにリモート CDP デバイスがキープアライブを送信した場合にイベントをトリガーします。 • delete：CDP キャッシュ エントリが CDP テーブルから削除された場合だけイベントをトリガーします。 • update：CDP キャッシュ エントリが CDP テーブルに追加された場合、または CDP キャッシュ エントリをアップデートするためにリモート CDP デバイスが CDP キープアライブを送信した場合にイベントをトリガーします。
lldp	LLDP のマッチング イベント発生時にイベントをトリガーします。次のオプションのいずれかを指定する必要があります。 • add：新しい CDP キャッシュ エントリが CDP テーブルに作成された場合にイベントをトリガーします。 • all：CDP キャッシュ エントリが CDP キャッシュ テーブルに追加された場合、または削除された場合、および CDP キャッシュ エントリをアップデートするためにリモート CDP デバイスがキープアライブを送信した場合にイベントをトリガーします。 • delete：CDP キャッシュ エントリが CDP テーブルから削除された場合だけイベントをトリガーします。 • update：CDP キャッシュ エントリが CDP テーブルに追加された場合、または CDP キャッシュ エントリをアップデートするためにリモート CDP デバイスが CDP キープアライブを送信した場合にイベントをトリガーします。
line-event	インターフェイス回線プロトコルのステータスが変わった場合にイベントをトリガーします。
link-event	インターフェイス リンクのステータスが変わった場合にイベントをトリガーします。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大実行時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 31536000 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

EEM_EVENT_NEIGHBOR_DISCOVERY の Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u nd_notification {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec　event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
共通の Event_Reqinfo	
nd_notification	通知のタイプ：cdp-add、cdp-update、cdp-delete、lldp-add、lldp-update、lldp-delete、link、line。
nd_intf_linkstatus	現在のインターフェイス リンクのステータス。up または down。
nd_intf_linestatus	現在のインターフェイス回線のステータス。down、goingdown、init、testing、up、reset、admindown、deleted。
nd_local_intf_name	イベントのローカル インターフェイスの名前。
nd_short_local_intf_name	イベントのローカル インターフェイスの短い名前。
nd_port_id	CDP プロトコルまたは LLDP プロトコルのいずれかで識別されたポート ID。これは、リンクまたは回線プロトコルのイベントには設定されません。
CDP-specific Event_reqinfo	
nd_protocol	イベントをトリガーしたプロトコルを識別します。CDP の場合は常に cdp に設定されます。
nd_proto_notif	イベント、追加、更新、または削除をトリガーしたプロトコル イベントのタイプを特定します。
nd_proto_new_entry	1 に設定されている場合、キャッシュ エントリは新規であるため、イベントはトリガーされており、それ以外の場合は 0 に設定されます。
nd_cdp_entry_name	CDP テーブル内の CDP キャッシュ エントリの名前。

イベント タイプ	説明
nd_cdp_hold_time	CDP キャッシュ エントリが期限切れになり、CDP テーブルから削除されるまでの残り時間。この時間は、CDP ネイバーからの更新によって最大値にリセットされます。新しいエントリの場合は通常、0 に設定されます。
nd_cdp_mgmt_domain	CDP VTP 管理ドメイン。
nd_cdp_platform	リモート デバイスによって報告されるプラットフォームの名前。
nd_cdp_version	リモートデバイスで実行されているコードのバージョン。
nd_cdp_capabilities_string	文字列形式の CDP capabilities フィールドのコンテンツ：ルータ、トランスブリッジ、ソースルートブリッジ、スイッチ、ホスト、IGMP、リピータ、電話、リモートで管理されているデバイス、CVTA 電話ポート、2 ポート MAC リレー、または、カンマで区切ったこれらの組み合わせ。
nd_cdp_capabilities_bits	先頭に 0x が付加された 16 進数内の CDP 機能ビット。
nd_cdp_capabilities_bit_[0-31]	capabilities フィールドのそのビットが設定されている場合は YES に、設定されていない場合は NOT に設定される一連の値。
LLDP-specific Event_reqinfo	
nd_protocol	イベントをトリガーしたプロトコルを識別します。LLDP の場合は常に lldp に設定されます。
nd_proto_notif	イベント、追加、更新、または削除をトリガーしたプロトコル イベントのタイプを特定します。
nd_proto_new_entry	1 に設定されている場合、キャッシュ エントリは新規であるため、イベントはトリガーされており、それ以外の場合は 0 に設定されます。
nd_lldp_chassis_id	LLDP キャッシュ エントリからの chassis id フィールド。
nd_lldp_system_name	LLDP キャッシュ エントリからのシステム名。
nd_lldp_system_description	LLDP キャッシュ エントリからの system description フィールド。
nd_lldp_ttl	LLDP キャッシュ エントリからの LLDP time to live フィールド。
nd_lldp_port_description	LLDP キャッシュ エントリからの port description フィールド。

イベント タイプ	説明
nd_lldp_system_capabilities_string	LLDP キャッシュ エントリからの LLDP system capabilities フィールド。この文字列には、O、P、B、W、R、T、C、S、またはこれらの任意の組み合わせをカンマで区切って含めることができます。
nd_lldp_enabled_capabilities_string	LLDP キャッシュ エントリからの LLDP enabled system capabilities フィールド。この文字列には、O、P、B、W、R、T、C、S、またはこれらの任意の組み合わせをカンマで区切って含めることができます。
nd_lldp_system_capabilities_bits	LLDP キャッシュ エントリからの LLDP system capabilities bits フィールド。先頭に 0x が付加された 16 進数です。
nd_lldp_enabled_capabilities_bits	LLDP キャッシュ エントリからの LLDP enabled capabilities bits フィールド。先頭に 0x が付加された 16 進数です。
nd_lldp_capabilities_bits	LLDP キャッシュ エントリからの LLDP capabilities bits フィールド。先頭に 0x が付加された 16 進数です。
nd_lldp_capabilities_bit_[0-31]	capabilities フィールドのそのビットが設定されている場合は YES に、設定されていない場合は NOT に設定される一連の値。

event_register_nf

NetFlow イベントが **event nf** コマンドによってトリガーされるときのイベントの登録を行います。この Tcl コマンド拡張を使用すると、NetFlow の応答がトリガーされるときに、イベントがパブリッシュされます。

構文

```
event_register_nf [tag ?] monitor_name ? event_type create|update|delete
exit_event_type create|update|delete event1-event4 ? [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
monitor_name	（必須）NetFlow モニターの名前。
event_type	（必須）フローの作成、アップデート、および削除を監視するイベントのタイプ。

exit_event_type	（必須）監視のためにイベントが再準備されるイベントタイプ（create、delete、update）。
event1- event4	（必須）監視するイベントとその属性を指定します。有効な値は **event1**、**event2**、**event3**、および **event4** です。 サブイベント キーワードは、単独でも、一緒でも、それぞれの任意の組み合わせでも使用できますが、各キーワードは 1 回のみ使用できます。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice引数が1に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

サブイベントの構文

```
field ? rate_interval ? event1 only entry_value ? entry_op eq|ge|gt|le|lt|wc
[exit_value ?] [exit_op eq|ge|gt|le|lt|wc] [exit_rate_interval ? event1 only]
```

サブイベントの引数

field	（必須）監視されるキャッシュまたはフィールド属性を指定します。次の属性の 1 つを指定できます。 • **counter** {**bytes** | **packets**}：カウンタフィールドを指定します。 • **datalink** {**dot1q** | **mac**}：データリンク（レイヤ 2）フィールドを指定します。 • **flow** {**direction** | **sampler**}：フロー識別フィールドを指定します。 • **interface** {**input** | **output**}：インターフェイスフィールドを指定します。 • **ipv4** *field-type*： IPv4 フィールドを指定します。 • **ipv6** *field-type*： IPv6 フィールド • **routing** *routing-attrribute* **--** ：ルーティング属性を指定します。 • **timestamp sysuptime** {**first** | **last**}**--**：タイムスタンプ フィールドを指定します。 • **transport** *field-type*： トランスポート レイヤ フィールドを指定します。
rate_interval	（必須）レートの計算に使用されるレート間隔値を秒単位で指定します。このフィールドは、event1 でのみ有効です。

entry_value	（必須）フィールドまたはレートの値を指定します。
entry_op	（必須）フィールド演算子を指定します。 次の比較演算子の値が有効です。 • **eq** ：次の値と等しい • **ge** ：次の値以上 • **gt** ：次の値より大きい • **le** ：次の値以下 • **lt** ：次の値より小さい • **wc** ：ワイルドカード
exit_value	（任意）イベントが再度監視されるように再準備される値。
exit_op	（任意）現在のイベントフィールドまたはレートの値を終了値と比較するために使用される比較演算子。真の場合、このイベントのイベント監視が再度イネーブルにされます。 次の比較演算子の値が有効です。 • **eq** ：次の値と等しい • **ge** ：次の値以上 • **gt** ：次の値より大きい • **le** ：次の値以下 • **lt** ：次の値より小さい • **wc** ：ワイルドカード
exit_rate_interval	（任意）終了レート値の計算に使用される終了レート間隔値を秒単位で指定します。このフィールドは、event1 でのみ有効です。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_ID %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u monitor_name %u event1-event4_field %u event1-event4_value
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	フローの作成、アップデート、および削除を監視するイベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	NetFlow イベントの重大度。
montior_name	NetFlow モニターの名前。
event1-event4_field	監視するイベントとその属性を指定します。有効な値は **event1**、**event2**、**event3**、および **event4** です。
event1-event4_value	監視するイベント値とその属性を指定します。有効な値は **event1**、**event2**、**event3**、および **event4** です。

event_register_none

event manager run コマンドによってトリガーされるイベントの登録を行います。これらのイベントは、このイベントをスクリーニングする None イベント ディテクタによって処理されます。

構文

```
event_register_none [tag ?] [sync {yes|no}] [default ?] [queue_priority
low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
sync	（任意）このキーワードを完了するには、「yes」または「no」が必要です。 • yes キーワードが指定されている場合、ポリシーは、CLI コマンドと同期的に実行されます。 • no キーワードが指定されている場合、ポリシーは、CLI コマンドと非同期的に実行されます。

デフォルト	（任意）CLI イベント ディテクタがポリシーの終了を待つ時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。ポリシーが終了する前にデフォルトの時間の期限が切れると、デフォルト アクションが実行されます。デフォルト アクションによって、コマンドが実行されます。この引数が指定されない場合、デフォルトの時間は 30 秒に設定されます。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
event_severity %u arg %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
argc **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14** **arg15**	Extensible Markup Language（XML）Simple Object Access Protocol（SOAP）コマンドからスクリプトに渡されるパラメータ。

event_register_oir

活性挿抜（OIR）イベントの登録を行います。このTclコマンド拡張を使用すると、ハードウェア カードOIRイベントの発生時に発生するイベントに基づいて、ポリシーが実行されます。

これらのイベントは、このイベントをスクリーニングする OIR イベント ディテクタによって処理されます。

構文

```
event_register_oir [tag ?] [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"slot %u event %s"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一のイベント ID を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
slot	影響が及ぼされるカードのスロット番号。
event	OIR の削除イベントまたは OIR の挿入イベントを表す、removed または online の文字列を示します。

event_register_process

プロセス イベントの登録を行います。この Tcl コマンド拡張を使用すると、Cisco IOS ソフトウェア モジュール方式プロセスの開始時と停止時に発生するイベントに基づいて、ポリシーが実行されます。これらのイベントは、このイベントをスクリーニングする System Manager イベント ディテクタによって処理されます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
event_register_process [tag ?] abort|term|start|user_restart|user_shutdown
[sub_system ?] [version ?] [instance ?] [path ?] [node ?]
[queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

abort	（必須）プロセスの異常な終了。ゼロではない終了ステータスでの終了、カーネル生成信号の受信、またはユーザー要求のために送信されない SIGTERM 信号または SIGKILL 信号の受信のため、プロセスが強制終了されることがあります。
term	（必須）プロセスの正常な終了。
start	（必須）プロセスの開始。
user_restart	（必須）CLI コマンドからのプロセスの再起動要求が原因でプロセスを終了。
user_shutdown	（必須）CLI コマンドからのプロセスの終了要求が原因でプロセスを終了。
sub_system	（任意）プロセス イベントをパブリッシュした EEM ポリシーに割り当てられる番号。他のすべての番号は Cisco での使用のために予約されているため、番号は 798 に設定されます。
version	（任意）バージョンマネージャによって割り当てられるプロセスのバージョン番号。major_number.minor_number.level の形式である必要があります。指定される場合、バージョン番号の各コンポーネントは、1 ～ 4294967295 の範囲の整数である必要があります。
instance	（任意）プロセス インスタンス ID。指定される場合、この引数は、1 ～ 4294967295 の範囲の整数である必要があります。
path	（任意）プロセス パス名（正規表現文字列）。process-name 引数の値に空白文字が含まれている場合、二重引用符で囲む必要があります。すべてのプロセスを照合するには、パス「.*」を使用します。
ノード	（任意）ノード名は、「node」という語句と、それに続く、次の形式を使用してスラッシュ文字で区切られた 2 つのフィールドで構成される、文字列です。 node<slot-number>/<cpu-number> slot-number は、ハードウェア スロット番号です。cpu-number は、ハードウェア CPU 番号です。たとえば、スロット 0 にある Cisco Catalyst 6500 シリーズ スイッチのスーパーバイザ カードの SP CPU は、node0/0 と指定されます。たとえば、スロット 0 にある Cisco Catalyst 6500 シリーズ スイッチのスーパーバイザ カードの RP CPU は、node0/1 と指定されます。node 引数が指定されない場合、デフォルトのノード指定は、常に、すべての該当するノードを表す正規表現パターン マッチ * です。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

任意の引数が指定されない場合、イベントは、引数のすべての可能な値に対して照会されます。複数の引数が存在する場合、すべての条件が一致したときに、プロセスイベントが発生します。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"sub_system 0x%x instance %u process_name {%s} path {%s} exit_status 0x%x"
```

```
"respawn_count %u last_respawn_sec %ld last_respawn_msec %ld fail_count %u"
"dump_count %u node_name {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
sub_system	アプリケーション固有のイベントをパブリッシュしたEEMポリシーに割り当てられる番号。他のすべての番号はCiscoでの使用のために予約されているため、番号は 798 に設定されます。
instance	プロセス インスタンス ID。
process_name	プロセス名。
path	パスを含むプロセスの絶対名。
exit_status	プロセスの最後の終了ステータス。
respawn_count	プロセスが再起動された回数。
last_respawn_sec **last_respawn_msec**	最後の再起動が発生したカレンダー時間。
fail_count	失敗したプロセスの再起動試行の回数。プロセスが正常に再起動されると、0 にリセットされます。
dump_count	プロセスで取られたコア ダンプの数。
node_name	プロセスが存在するノードの名前。ノード名は、「node」という語句と、それに続く、次の形式を使用してスラッシュ文字で区切られた 2 つのフィールドで構成される、文字列です。 **node** *slot-number* / *cpu-number* slot-number は、ハードウェア スロット番号です。cpu-number は、ハードウェア CPU 番号です。

event_register_resource

Embedded Resource Manager（ERM）イベントの登録を行います。この Tcl コマンド拡張を使用すると、指定されたポリシーの ERM イベント レポートに基づいて、ポリシーが実行されます。ERM イベントは、EEM リソース イベントによってスクリーニングされ、これによって、指定された ERM ポリシーへの一致が発生したときに、EEM ポリシーを実行できます。

構文

```
event_register_resource policy policy-name [queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

ポリシー	（必須）ポリシーの使用を指定します。
policy-name	（必須）ERM ポリシーの名前。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"owner_id %lld user_id %lld" time_sent %llu dampen_time %d notify_data_flags %u"
"level {%s} direction {%s} configured_threshold %u current_value %u"
"policy_violation_flag {%s} policy_id %d"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベントタイプのイベントの名前を表すASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
owner_id	Embedded Resource Manager（ERM）オーナー ID。
user_id	ERM ユーザー ID。
time_sent	ERM イベント時間、ナノ秒単位。
dampen_time	ERM 減衰時間、ナノ秒単位。
notify_data_flags	ERM 通知データ フラグ。
level	ERM イベント レベル。イベント レベルは、Normal、Minor、Major、および Critical の 4 つです。
direction	ERM イベント方向。イベント方向は、アップ、ダウン、または、変更なしのうちのいずれかです。
configured_threshold	設定されている ERM しきい値。
current_value	ERM によって報告された、現在の値。

イベント タイプ	説明
policy_violation_flag	ERM ポリシー違反フラグ（False または True）。
policy_id	ERM ポリシー ID。

event_register_rf

冗長ファシリティ（RF）イベントの登録を行います。この Tcl コマンド拡張を使用すると、RF の進行またはステータス イベントの通知が発生したときに、ポリシーが実行されます。

構文

```
event_register_rf [tag ?] event ?
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

event	（必須）RF の進行またはステータスイベントの名前。有効な値は次のとおりです。 • RF_PROG_ACTIVE • RF_PROG_ACTIVE_DRAIN • RF_PROG_ACTIVE_FAST = 200 • RF_PROG_ACTIVE_PRECONFIG • RF_PROG_ACTIVE_POSTCONFIG • RF_PROG_EXTRALOAD • RF_PROG_HANDBACK • RF_PROG_INITIALIZATION • RF_PROG_PLATFORM_SYNC • RF_PROG_STANDBY_BULK • RF_PROG_STANDBY_COLD • RF_PROG_STANDBY_CONFIG • RF_PROG_STANDBY_FILESYS • RF_PROG_STANDBY_HOT • RF_PROG_STANDBY_OIR_SYNC_DONE • RF_REGISTRATION_STATUS • RF_STATUS_MAINTENANCE_ENABLE • RF_STATUS_MANUAL_SWACT • RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE • RF_STATUS_PEER_COMM • RF_STATUS_PEER_PRESENCE • RF_STATUS_REDUNDANCY_MODE_CHANGE • RF_STATUS_SWACT_INHIBIT

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event	このイベントが発生する原因となる RF の進行またはステータス イベント通知。

event_register_routing

event routing コマンドによってトリガーされるイベントの登録を行います。これらのイベントは、ルート エントリが Routing Information Base（RIB）インフラストラクチャで変更されるときに、ルーティング イベント ディテクタによって処理され、イベントがパブリッシュされます。この Tcl コマンド拡張を使用すると、このスクリプトのルーティング ポリシーが実行されます。監視されるルートのネットワーク IP アドレスを指定する必要があります。

構文

```
event_register_routing [tag ?] network ? length [ge|le|ne] [type add|remove|modify|all]
 [protocol ?] [queue_priority normal|low|high|last] [maxrun ?] [nice {0 | 1}]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
network	ネットワーク IP アドレスを指定します。ネットワーク番号には、任意の有効な IP アドレスまたはプレフィックスを指定できます。

length	ネットワーク マスクの長さをビット単位で指定します。ビット マスクは 0 から 32 までの番号を使用できます。 • **ge**（任意）照合されるプレフィックスの最小の長さを指定します。**ge** キーワードは、演算子の「以上」を表します。 • **le**（任意）照合されるプレフィックスの最大の長さを指定します。**le** キーワードは、演算子の「以下」を表します。 • **ne**（任意）プレフィックスの長さを照合しない指定をします。**ne** キーワードは、演算子の「等しくない」を表します。 **ge** キーワード、**le** キーワード、および **ne** キーワードが設定されない場合、ネットワーク長の完全一致が処理されます。
type	（任意）必要なポリシーのトリガーを指定します。タイプ オプションは、**add**、**remove**、**modify**、および **all** です。デフォルトは **all** です。
プロトコル	（任意）監視されているネットワークのプロトコルの値を指定します。 次のプロトコルのいずれかを使用できます：**all**、**bgp**、**connected**、**eigrp**、**isis**。**iso-igrp**、**mobile**、**odr**、**ospf**、**rip**、**static** デフォルトは **all** です。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u" "event_severity {%s} %u network %u mask %u protocol %u lastgateway %u distance %u" "time_sec %u time_msec %u metric %u lastinterface %u"

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベントタイプのイベントの名前を表すASCII文字列。
event_pub_sec event_pub_msec	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
network	IP アドレス形式のネットワーク プレフィックス。
mask	IP アドレス形式のネットワーク マスク。
protocol	ネットワーク プロトコルのタイプ。
type	追加、削除、または変更するイベントのタイプ。
lastgateway	最後に認識されたゲートウェイ。
distance	アドミニストレーティブ ディスタンス。
time_sec time_msec	イベントが EEM にパブリッシュされたときの、秒単位およびミリ秒単位でのイベントの時間。
metric	パス メトリック。
lastinterface	最後に認識されたインターフェイス。

event_register_rpc

EEM SSH リモート プロシージャ コール（RPC）コマンドによってトリガーされるイベントの登録を行います。これらのイベントは、このイベントをスクリーニングするRPCイベントディテクタによって処理されます。このTclコマンド拡張を使用すると、このスクリプトのRPCポリシーが実行されます。

構文

```
event_register_rpc [queue_priority {normal | low | high | last}] [maxrun <sec.msec>]
[nice {0 | 1}] [default <sec.msec>]
```

引数

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

デフォルト	（任意）CLI イベント ディテクタがポリシーの終了を待つ時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。ポリシーが終了する前にデフォルトの時間の期限が切れると、デフォルト アクションが実行されます。デフォルト アクションによって、コマンドが実行されます。この引数が指定されない場合、デフォルトの時間は30秒に設定されます。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u
arg %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec　event_pub_msec	EEMに対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。

argc **arg0** **arg1** **arg2** **arg3** **arg4** **arg6** **arg7** **arg8** **arg9** **arg10** **arg11** **arg12** **arg13** **arg14**	Extensible Markup Language（XML）Simple Object Access Protocol（SOAP）コマンドからスクリプトに渡されるパラメータ。

event_register_snmp

簡易ネットワーク管理プロトコル（SNMP）統計イベントの登録を行います。この Tcl コマンド拡張を使用すると、SNMP オブジェクト ID（OID）によって指定されたカウンタが、定義されたしきい値に近くなったときに、ポリシーが実行されます。

構文

```
event_register_snmp [tag ?] oid ? get_type exact|next
entry_op gt|ge|eq|ne|lt|le entry_val ?
entry_type value|increment|rate
[exit_comb or|and]
[exit_op gt|ge|eq|ne|lt|le] [exit_val ?]
[exit_type value|increment|rate]
[exit_time ?] poll_interval ? [average_factor ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

oid	（必須）SNMP ドット付き表記でのデータ エレメントの OID 番号（たとえば、1.3.6.1.2.1.2.1.0）。次のタイプの OID を使用できます。 • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
entry_op	（必須）現在の OID データの値を開始値と比較するために使用される開始比較演算子。真の場合、イベントが発生し、終了基準を満たすまでイベント モニターリングがディセーブルにされます。
get_type	（必須）指定された OID に適用する必要がある SNMP 取得操作のタイプ。get_type 引数が「exact」の場合、指定された OID の値が取得されます。get_type 引数が「next」の場合、指定された OID の辞書順での後続値が取得されます。
entry_val	（必須）SNMP イベントを発生させる必要があるかどうかを判断するために、現在の OID データの値を比較する必要がある値。
entry-type	entry-val 引数によって指定されたオブジェクト ID に適用される操作のタイプを指定します。 値は、entry-val 引数の実際の値として定義されます。 増分では、entry-val フィールドは増分差異として使用され、entry-val は、現在のカウンタの値と、イベントが最後に真であったとき（これが新しいイベントの場合は最初にポーリングされたサンプル）の値との間の差と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。 レートは、ある期間の変更の平均レートとして定義されます。期間は、average-factor の値に、poll-interval の値を乗じたものです。ポーリング間隔ごとに、現在のサンプルと前のサンプルとの間の差が取得され、絶対値として記録されます。前の average-factor 値サンプルの平均は、変更のレートとして取得されます。
exit_comb	（任意）イベント モニターリングが再度イネーブルにされるよう、終了基準が満たされているかどうかを判断するために必要な、終了条件テストの組み合わせを示す、終了組み合わせ演算子を使用します。「and」の場合は、終了基準を満たすために、終了値と終了時間テストの両方を渡す必要があります。「or」の場合は、終了基準を満たすために、終了値または終了時間テストのいずれかを渡します。exit_comb が「and」の場合、exit_op と exit_val（exit_time）が存在する必要があります。exit_comb が「or」の場合、（exit_op と exit_val）または（exit_time）が存在する必要があります。

exit_op	（任意）現在の OID データの値を終了値と比較するために使用される終了比較演算子。真の場合、このイベントのイベント モニターリングが再度イネーブルにされます。
exit_val	（任意）終了基準を満たすかどうかを判断するために、現在の OID データの値を比較する必要がある値。
exit-type	（任意）exit-val 引数によって指定されたオブジェクト ID に適用される操作のタイプを指定します。指定されない場合、値が仮定されます。 値は、exit-val 引数の実際の値として定義されます。 増分では、exit-val フィールドは増分差異として使用され、exit-val は、現在のカウンタの値と、イベントが最後に真であったとき（これが新しいイベントの場合は最初にポーリングされたサンプル）の値との間の差と、比較されます。負の値によって、減少しているカウンタの増分差異がチェックされます。 レートは、ある期間の変更の平均レートとして定義されます。期間は、average-factor の値に、poll-interval の値を乗じたものです。ポーリング間隔ごとに、現在のサンプルと前のサンプルとの間の差が取得され、絶対値として記録されます。前の average-factor 値サンプルの平均は、変更のレートとして取得されます。
exit_time	（任意）イベント モニターリングが再度イネーブルにされるときに発生するイベントの後の、POSIX タイマー ユニットの数。SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数である必要があります。MMM はミリ秒を表し、0 ～ 999 の整数である必要があります。
poll_interval	（必須）POSIX タイマー ユニットの連続的なポーリング間の間隔。間隔は、現在、最小で 1 秒に設定されます（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。
average-factor	（任意）レート ベースの計算に使用される期間の計算に使用される 1 から 64 の範囲の数。average-factor の値は、poll-interval の値を乗じた値で、ミリ秒単位で導き出される期間です。最少平均係数値は 1 です。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"event_severity {%s} oid {%s} val {%s} delta_val {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	SNMP イベントの重大度で、次のいずれかの値を指定できます。 • normal • minor • major
oid	SNMP ドット付き表記での、データ エレメントのオブジェクト ID。
val	データ エレメントの値。
delta_val	ポリシーの値間のデルタ値。

event_register_snmp_notification

簡易ネットワーク管理プロトコル（SNMP）通知トラップ イベントの登録を行います。この Tcl コマンド拡張を使用すると、特定のインターフェイスまたはアドレスで、指定された SNMP オブジェクト ID（OID）で SNMP トラップが検出されるときに、ポリシーが実行されます。SNMP 通知が Tcl ポリシーを使用して動作するようにするには、**snmp-server manager** CLI コマンドを有効にする必要があります。

構文

```
event_register_snmp_notification [tag ?] oid ? oid_val ?
op {gt|ge|eq|ne|lt|le}
[maxrun ?]
[src_ip_address ?]
[dest_ip_address ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
[default ?]
[direction {incoming|outgoing}]
[msg_op {drop|send}]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
oid	（必須）SNMP ドット付き表記でのデータ エレメントの OID 番号（たとえば、1.3.6.1.2.1.2.1.0）。指定された OID がドット（.）で終わっている場合、ドットの前の OID 番号で始まっているすべての OID が、照会されます。次のタイプの OID を使用できます。 • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
oid_val	（必須）SNMP イベントを発生させる必要があるかどうかを判断するために、現在の OID データの値を比較する必要がある OID 値。
op	（必須）現在の OID データの値を、SNMP プロトコル データ ユニット（PDU）の OID データ値と比較するために使用される、比較演算子。真の場合、イベントが発生します。
maxrun	（任意）スクリプトの最大実行時間（ssssssss[.mmm] 形式で指定します。ssssssss は、0 ～ 31536000 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
src_ip_address	（任意）SNMP 通知トラップが発信される発信元 IP アドレス。デフォルトは all です。すべての IP アドレスから SNMP 通知トラップを受信するよう、設定されます。
dest_ip_address	（任意）SNMP 通知トラップが送信される宛先 IP アドレス。デフォルトは all です。すべての宛先 IP アドレスから SNMP トラップを受信するよう、設定されます。

queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 queue_priority_last 引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
デフォルト	（任意）SNMP 通知イベント ディテクタがポリシーの終了を待つ、秒単位での時間を指定します。time 時間は、ssssssssss[.mmm] 形式で指定します。ssssssssss は、0 ～ 4294967295 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です
direction	（任意）発着信 SNMP トラップまたは通知 PDU がフィルタリングする方向。デフォルトは incoming です。
msg_op	（任意）イベントが一度トリガーされると、SNMP PDU（廃棄または送信）で行われるアクション。デフォルトは send です。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} oid_val {%s} src_ip_addr {%s} dest_ip_addr {%s} x_x_x_x_x
 (varbinds) {%s} trunc_vb_buf {%s} trap_oid {%s} enterprise_oid {%s} generic_trap %u
specific_trap %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
oid	ユーザー指定オブジェクト ID。
oid_val	ユーザー指定オブジェクト ID 値。
src_ip_addr	SNMP プロトコル データ ユニット（PDU）の発信元 IP アドレス。
dest_ip_addr	SNMP PDU の宛先の IP アドレス。
x_x_x_x_x (varbinds)	SNMP PDU varbind 情報。
trap_oid	トラップ OID 値を示します。
enterprise_oid	エンタープライズ OID 値を示します。
generic_trap	汎用トラップ タイプの番号の 1 つを示します。0 から 6 の、7 つの汎用トラップ タイプがあります。
specific_trap	指定されたトラップ コードの番号の 1 つを示します。

event_register_snmp_object

簡易ネットワーク管理プロトコル（SNMP）オブジェクト イベントの登録を行います。この Tcl コマンド拡張を使用すると、特定のインターフェイスまたはアドレスで、指定された SNMP オブジェクト ID（OID）で SNMP が検出されるときに、値が置き換えられます。

構文

```
event_register_snmp_object oid ?
```

```
type {int|uint|counter|counter64|gauge|ipv4||oid|string}
sync {yes|no}
skip {yes|no}
[istable {yes|no}]
[default ?]
[queue_priority {normal|low|high|last}]
[maxrun ?]
[nice {0|1}]
```

引数

oid	（必須）SNMP ドット付き表記でのデータ エレメントの OID 番号（たとえば、1.3.6.1.2.1.2.1.0）。指定された OID がドット（.）で終わっている場合、ドットの前の OID 番号で始まっているすべての OID が、照会されます。次のタイプの OID を使用できます。 • COUNTER_TYPE • COUNTER_64_TYPE • GAUGE_TYPE • INTEGER_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE • TIME_TICKS_TYPE
type	（必須）OID 値のタイプ。
sync	（必須）「yes」は、EEM ポリシーが通知されることを意味します。アプレット set_exit_status または Tcl 戻り値が 0 の場合、SNMP によって、要求が処理されます。戻り値が 1 の場合、SNMP によって、get 要求のポリシーで指定された値が使用され、set 要求は処理されません。「no」は、EEM は通知されず、SNMP によって要求が処理されることを意味します。 1 つの OID のみが、同期ポリシーに関連付けられます。ただし、複数の同期ポリシーが、同じ OID に登録できます。
skip	sync 引数が no の場合は必須で、sync 引数が yes の場合は不要ですskip 引数が「yes」の場合、SNMP によって要求が処理されることを意味します。skip 引数が「no」の場合、SNMP は、オブジェクトが存在しないかのように動作することを意味します。
istable	（任意）値「no」は、OIDがスカラーオブジェクトであることを意味し、「yes」は、OID がテーブル オブジェクトであることを意味します。

デフォルト	（任意）SNMP オブジェクト イベント ディテクタがポリシーの終了を待つ時間（sssssssss[.mmm] 形式で指定します。sssssssss は、0 ～ 4294967295 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります）。ポリシーが終了する前にデフォルトの時間の期限が切れると、デフォルト アクションが実行されます。デフォルト アクションは、通常、SNMP サブシステムによって set 要求または get 要求を処理することです。この引数が指定されない場合、デフォルトの時間は 30 秒に設定されます。
maxrun	（任意）スクリプトの最大実行時間（ssssssss[.mmm] 形式で指定します。ssssssss は、0 ～ 31536000 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 queue_priority_last 引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u
event_severity {%s}" "oid {%s} request {%s} request_type {%s} value %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
event_severity	イベントの重大度。
oid	受信した get 要求または set 要求の SNMP オブジェクトの ID。
request	get または set の要求タイプ。
request_type	要求のタイプ（現在または次の）。
value	set 要求のみ。オブジェクトに設定される値。

event_register_syslog

Syslog イベントの登録を行います。この Tcl コマンド拡張を使用すると、一定の時間内に一定回数の発生後、特定パターンの Syslog メッセージが記録されるときに、ポリシーがトリガーされます。

構文

```
event_register_syslog [tag ?] [occurs ?] [period ?] pattern ?
[priority all|emergencies|alerts|critical|errors|warnings|notifications|
informational|debugging|0|1|2|3|4|5|6|7]
[queue_priority low|normal|high|last]
[severity_fatal] [severity_critical] [severity_major]
[severity_minor] [severity_warning] [severity_notification]
[severity_normal] [severity_debugging]
[maxrun ?] [nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。

occurs	（任意）イベントが発生する前の発生回数。この引数が指定されない場合、イベントは1回目から発生します。指定される場合、0より大きい値を指定する必要があります。
period	（任意）イベントを発生させるために取る必要がある 1 つまたは複数のイベントの間の、秒単位およびミリ秒単位の時間の間隔（SSSSSSSSSS[.MMM]形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0～999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、期間チェックは適用されません。
pattern	（必須）Syslog メッセージ パターン マッチの実行に使用される正規表現。この引数は、記録された Syslog メッセージを指定するためにポリシーによって使用されます。
priority	（任意）スクリーニングされるメッセージのプライオリティ。この引数が指定される場合、指定されたロギング プライオリティ レベルまたはそれ以下メッセージのみがスクリーニングされます。この引数が指定されない場合、デフォルトのプライオリティは 0 です。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。

nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です
severity_xxx	（任意）スクリーニングされるイベントの重大度。この引数が指定される場合、指定された重大度のメッセージのみがスクリーニングされます。syslog イベントの重大度レベルのマッピングについては、「Syslog イベントの重大度のマッピング」というタイトルの表を参照してください。

複数の条件が存在する場合、すべての条件が一致したときに、Syslogイベントが発生します。

***Table 75: Syslog* イベントの重大度のマッピング**

重大度のキーワード	Syslogのプライオリティ	説明
severity_fatal	LOG_EMERG (0)	システムが使用不可能な状態。
severity_critical	LOG_ALERT (1)	クリティカル条件で、即時対応が必要であることを示す
severity_major	LOG_CRIT (2)	重大な状態。
severity_minor	LOG_ERR (3)	軽微な状態。
severity_warning	LOG_WARNING (4)	警告状態。
severity_notification	LOG_NOTICE (5)	基本的な通知、情報メッセージ
severity_normal	LOG_INFO (6)	正常なイベント、正常な状態に戻ったことを伝える
severity_debugging	LOG_DEBUG (7)	デバッグ メッセージ。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"msg {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントのIDを示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
msg	パターンと一致する最後の Syslog メッセージ。

event_register_timer

パブリッシャとサブスクライバの両方として、タイマーを作成し、タイマーイベントの登録を行います。時間特有または時間に基づいたポリシーをトリガーする必要があるときに、この Tcl コマンド拡張を使用します。このイベント タイマーは、イベントのパブリッシャとサブスクライバの両方です。パブリッシャの部分は、名前付きタイマーがオフになるという条件を示します。サブスクライバの部分は、イベントが登録されているタイマーの名前を示します。

Note CRON および絶対時間の指定は、現地時間で動作します。

構文

```
event_register_timer [tag ?] watchdog|countdown|absolute|cron
[name ?] [cron_entry ?]
[time ?]
[queue_priority low|normal|high|last] [maxrun ?]
[nice 0|1]
```

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
watchdog	（必須）ウォッチドッグ タイマー。
countdown	（必須）カウントダウン タイマー。
絶対	（必須）絶対タイマー。
cron	（必須）CRON タイマー。

name	（任意）タイマーの名前。

cron_entry	

	（任意）CRON タイマー タイプが指定される場合に、指定する必要があります。他のいずれかのタイマー タイプが指定される場合には、指定しないでください。cron_entry は、UNIX CRON デーモンで使用される部分的な UNIX Crontab エントリ（最初の 5 つのフィールド）です。 cron_entry の指定は、5 つのフィールドが使用されるテキスト文字列で構成されます。フィールドは、空白文字で区切られます。フィールドは、CRON タイマーイベントがトリガーされるときの時刻と日付を表します。フィールドの説明については、「CRON イベントがトリガーされるときの時刻と日付」というタイトルの表を参照してください。 番号の範囲を使用できます。範囲は、ハイフンで区切られる 2 つの数字で表示されます。範囲には、2つの数字自身も含まれます。たとえば、時刻に入力される 8-11 は、8 時、9 時、10 時、および 11 時での実行を示します。 フィールドはアスタリスク記号（*）も使用でき、これは常に「first-last」を表します。 リストを使用できます。リストは、カンマで区切られた番号のセット（または範囲）です。例："1,2,5,9" および "0-4,8-12"。 手順の値は、範囲の組み合わせで使用できます。範囲に続く「/<number>」によって、範囲内での省略値を指定します。たとえば、2時間ごとにイベントのトリガーを指定する場合、「0-23/2」を hour フィールドに使用できます。アスタリスク記号後にも手順を使用でき、「2 時間ごと」と指定する場合は、「*/2」を使用します。 month フィールドと day of week フィールドには、名前も使用できます。特定の日または月の最初の 3 文字を使用します（ケースは問題ではありません）。名前の範囲またはリストは使用できません。 タイマー イベントがトリガーされる日は、day of month と day of week の 2 つのフィールドで指定できます。両方のフィールドが制限される（つまり * ではない）場合、いずれかのフィールドが現在の時刻と一致すると、イベントがトリガーされます。たとえば、「30 4 1,15 * 5」の場合、各月の 1 日と 15 日に加え、金曜日の午前 4:30 にイベントがトリガーされます。 最初の 5 つのフィールドの代わりに、7 つの特殊文字列の 1 つが表示されることがあります。これらの 7 つの特殊文字の説明については、「cron_entry の特殊文字列」というタイトルの表を参照してください。 例 1：「0 0 1,15 * 1」では、各月の 1 日と 15 日、および月曜日ごとに、真夜中の 0 時に、イベントがトリガーされます。1 つのフィールドによってのみ日を指定する場合、他のフィールドは * に設定する必要があります。「0 0 * * 1」では、月曜日にのみ、真夜中の 0 時に、イベントがトリガーされます。 例 2：「15 16 1 * *」では、各月の 1 日の午後 4:15 にイベントがトリガーされます。 例 3：「0 12 * * 1-5」では、各週の月曜日から金曜日まで、正午に、イベントがトリガーされます。

	例 4：「@weekly」では、1 週間に一度、日曜日の真夜中の 0 時に、イベントがトリガーされます。
time	（任意）CRON 以外のタイマー タイプが指定される場合に、指定する必要があります。CRON タイマー タイプが指定される場合には、指定しないでください。ウォッチドッグ タイマーとカウントダウン タイマーでは、タイマーの期限が切れるまでの秒およびミリ秒の単位での数です。絶対タイマーでは、期限切れ時刻のカレンダー時間です。時間は、SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります。期限の絶対日付は、1970 年 1 月 1 日以降の秒およびミリ秒の単位での数です。指定された日付がすでに過ぎた場合、タイマーの期限はただちに切れます。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

Table 76: CRON イベントがトリガーされるときの時刻と日付

フィールド	使用可能な値
minute	0 ～ 59
hour	0 ～ 23
day of month	1 ～ 31
month	1 ～ 12（または名前、下記を参照）
day of week	0 ～ 7（0 または 7 が日曜日または名前。「cron_entry の特殊文字列」というタイトルの表を参照）

Table 77: cron_entry の特殊文字列

文字列	意味
@yearly	1 年に 1 回トリガーする、「0 0 1 1 *」。
@annually	@yearly と同じ。
@monthly	1 か月に 1 回トリガーする、「0 0 1 * *」。
@weekly	1 週間に 1 回トリガーする、「0 0 * * 0」。
@daily	1 日に 1 回トリガーする、「0 0 * * *」。
@midnight	@daily と同じ。
@hourly	1 時間に 1 回トリガーする、「0 * * * *」。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec　event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
timer_type	タイマーのタイプ。次のいずれかです。 • watchdog • countdown • 絶対
timer_time_sec　timer_time_msec	タイマーの期限が切れる時間。
timer_remain_sec **timer_remain_msec**	次の期限切れ前の残りの時間。

関連項目

event_register_timer_subscriber

event_register_timer_subscriber

サブスクライバとしてタイマー イベントの登録を行います。この Tcl コマンド拡張を使用すると、サブスクライバとして、登録するイベント タイマーの名前が指定されます。イベント タイマーは、別のポリシーまたは別のプロセスに依存して、カウンタが実際に操作されます。たとえば、policyB はタイマー加入者ポリシーとして動作しますが、policyA（タイマー ポリシーは不要ですが）では、register_counter、timer_arm、または timer_cancel の各 Tcl コマンド拡張を使用して、policyB で参照されているカウンタが操作されます。

構文

```
event_register_timer_subscriber watchdog|countdown|absolute|cron
name ? [queue_priority low|normal|high|last] [maxrun ?] [nice 0|1]
```

引数

watchdog	（必須）ウォッチドッグ タイマー。

countdown	（必須）カウントダウン タイマー。
絶対	（必須）絶対タイマー。
cron	（必須）CRON タイマー。
name	（必須）タイマーの名前。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

Note タイマーイベントまたはカウンタイベントの登録を行う EEM ポリシーは、パブリッシャとサブスクライバの両方として動作できます。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} event_pub_sec %u event_pub_msec %u"
"timer_type %s timer_time_sec %ld timer_time_msec %ld"
"timer_remain_sec %ld timer_remain_msec %ld"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec　event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
timer_type	タイマーのタイプ。次のいずれかです。 • watchdog • countdown • 絶対
timer_time_sec　timer_time_msec	タイマーの期限が切れる時間。
timer_remain_sec **timer_remain_msec**	次の期限切れ前の残りの時間。

関連項目

event_register_timer

event_register_track

Cisco IOS Object Tracking サブシステムからのレポート イベントの登録を行います。この Tcl コマンド拡張を使用すると、指定されたオブジェクト番号の Cisco IOS Object Tracking サブシステム レポートに基づいて、ポリシーがトリガーされます。

構文

```
event_register_track ? [tag ?] [state up|down|any] [queue_priority low|normal|high|last]
```

```
 [maxrun ?]
[nice 0|1]
```

引数

?（番号を表す）	（必須）1 から 500 の範囲でトラックされるオブジェクト番号。
tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
state	（任意）トラックされるオブジェクトの状態遷移によってイベントが発生するよう、指定します。**up** が指定されている場合、トラックされるオブジェクトが down 状態から up 状態に遷移するときにイベントが発生します。**down** が指定されている場合、トラックされるオブジェクトが up 状態から down 状態に遷移するときにイベントが発生します。**any** が指定されている場合、トラックされるオブジェクトがある状態から別の状態に遷移するときにイベントが発生します。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

任意の引数が指定されない場合、イベントは、引数のすべての可能な値に対して照会されます。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"track_number {%u} track_state {%s}"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一のイベント ID を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec **event_pub_msec**	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
track_number	イベントがトリガーされる原因となるトラックされるオブジェクトの番号。
track_state	イベントがトリガーされたときのトラックされるオブジェクトの状態。有効な値は up または down です。

event_register_wdsysmon

Watchdog System Monitor イベントの登録を行います。この Tcl コマンド拡張を使用すると、いくつかのサブイベントまたは条件の組み合わせである複合イベントの登録が行われます。たとえば、特定処理の CPU の使用率が 80% を超える場合で、かつ処理に使用されるメモリが初期割り当て容量の 50% よりも大きい場合といった条件を組み合わせて登録できます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
event_register_wdsysmon [tag ?] [timewin ?]
[sub12_op and|or|andnot]
[sub23_op and|or|andnot]
```

```
[sub34_op and|or|andnot]
[sub1 subevent-description]
[sub2 subevent-description]
[sub3 subevent-description]
[sub4 subevent-description] [node ?]
[queue_priority low|normal|high|last]
[maxrun ?] [nice 0|1]
```

各引数は、位置に依存しません。

Note 演算子の定義は、and（論理 And 操作）or（論理 Or 操作）、andnot（論理 And Not 操作）です。たとえば、「sub12_op and」では、サブイベント 1 およびサブイベント 2 が真であるときにイベントが発生するよう定義されます。「sub23_op or」では、sub12_op で定義された条件が真で、サブイベント 3 が真であるときに、イベントが発生するよう定義されます。ロジックは、次のようにダイアグラム化できます。(((sub1 sub12_op sub2) sub23_op sub3) sub34_op sub4) が真の場合、イベントが発生

引数

tag	（任意）Tcl スクリプト内で複数のイベント文をサポートするため、Tcl コマンド拡張のトリガーとともに使用できるタグを指定する文字列。
timewin	（任意）イベントが生成されるようにするために、すべてのサブイベントが発生する必要がある時間ウィンドウ（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。
sub12_op	（任意）サブイベント 1 とサブイベント 2 とを比較する組み合わせ演算子。
sub23_op	（任意）サブイベント 1、2 とサブイベント 3 とを比較する組み合わせ演算子。
sub34_op	（任意）サブイベント 1、2、サブイベント 3、サブイベント 4 とを比較する組み合わせ演算子。
sub1	（任意）サブイベント 1 の指定を意味します。
subevent-description	（任意）サブイベントの構文。
sub2	（任意）サブイベント 2 の指定を意味します。
sub3	（任意）サブイベント 3 の指定を意味します。
sub4	（任意）サブイベント 4 の指定を意味します。

ノード	（任意）デッドロック条件が監視されるノード名は、「node」という語句と、それに続く、次の形式を使用してスラッシュ文字で区切られた 2 つのフィールドで構成される文字列です。 node<slot-number>/<cpu-number> slot-number は、ハードウェア スロット番号です。cpu-number は、ハードウェア CPU 番号です。たとえば、スロット 0 にある Cisco Catalyst 6500 シリーズ スイッチのスーパーバイザ カードの SP CPU は、node0/0 と指定されます。たとえば、スロット 0 にある Cisco Catalyst 6500 シリーズ スイッチのスーパーバイザ カードの RP CPU は、node0/1 と指定されます。node 引数が指定されない場合、デフォルトのノード指定は、登録が行われているローカル ノードです。
queue_priority	（任意）次のような、スクリプトがキューに入れられる優先度レベル。 • queue_priority low：3 つの優先度レベルの最も低いレベルでスクリプトがキューに入れられるよう指定します。 • queue_priority normal：low プライオリティよりも高く、high プライオリティよりも低い優先度レベルでスクリプトがキューに入れられるよう指定します。 • queue_priority high：3 つの優先度レベルの最も高いレベルで、スクリプトがキューに入れられるよう指定します。 • queue_priority last：最も低い優先度レベルでスクリプトがキューに入れられるよう指定します。 「queue_priority_last」引数が設定された状態で複数のスクリプトが登録されている場合、これらのスクリプトは、イベントのパブリッシュ順に実行されます。 **Note** queue_priority 引数によって、登録されているスクリプトの実行優先度ではなく、キューイングの優先度が指定されます。 この引数が指定されない場合、デフォルトのキューイング プライオリティは normal です。
maxrun	（任意）スクリプトの最大ランタイム（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合、デフォルトの 20 秒ランタイム制限が使用されます。
nice	（任意）ポリシーの実行時間優先度の設定。nice 引数が 1 に設定されている場合、ポリシーは、デフォルトの優先度よりも低い実行時間優先度で実行されます。デフォルト値は 0 です

サブイベント

subevent description の構文は、7 つのケースのうちの 1 つを使用できます。

subevent descriptions の引数では、number 引数の値に次の制約事項が適用されます。

- dispatch_mgr では、val は、0 ～ 4294967295 の範囲の整数である必要があります。
- cpu_proc および cpu_tot では、val は、0 ～ 100 の整数である必要があります。
- mem_proc、mem_tot_avail、および mem_tot_used では、is_percent が偽の場合、val は、0 ～ 4294967295 の範囲の整数である必要があります。

```
1. deadlock procname ?
```

引数

procname	（必須）デッドロック条件をモニターするプロセス名を指定する正規表現。指定された場合、サブイベントによって、時間ウィンドウは無視されます。

```
2. dispatch_mgr [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

引数

procname	（任意）dispatch_manager ステータスをモニターするプロセス名を指定する正規表現。
op	（任意）収集されたイベント数を指定された値と比較するために使用される比較演算子。真の場合、このイベントが発生します。
val	（任意）発生したイベント数の値を比較する必要があります。
period	（任意）発生したイベント数の時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。

```
3. cpu_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

引数

procname	（任意）CPU の使用条件をモニターするプロセス名を指定する正規表現。
op	（任意）収集された CPU 使用率サンプル パーセンテージを、指定されたパーセント値と比較するために使用される、比較演算子。真の場合、このイベントが発生します。
val	（任意）サンプル期間の平均 CPU 使用率のパーセント値を比較する必要があります。

period	（任意）サンプルの収集の平均の時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。

```
4. cpu_tot [op gt|ge|eq|ne|lt|le] [val ?] [period ?]
```

引数

op	（任意）収集された合計システム CPU 使用率サンプル パーセンテージを、指定されたパーセント値と比較するために使用される、比較演算子。真の場合、このイベントが発生します。
val	（任意）サンプル期間の平均 CPU 使用率のパーセント値を比較する必要があります。
period	（任意）サンプルの収集の平均の時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。

```
5. mem_proc [procname ?] [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period
 ?]
```

引数

procname	（任意）メモリ使用状況をモニターするプロセス名を指定する正規表現。
op	（任意）収集された使用メモリを、指定された値と比較するために使用される比較演算子。真の場合、このイベントが発生します。
val	（任意）キロバイト単位で指定される、パーセンテージまたは絶対値。パーセンテージは、指定された時間内で最も古いサンプルと、最新のサンプルとの違いを表します。メモリ使用量が時間内で 150 KB から 300 KB に増えた場合、増加パーセンテージは 100 です。これは、測定された値を比較する必要がある値です。
is_percent	（任意）真の場合、パーセンテージの値が収集され、比較されます。これ以外の場合、絶対値が収集され、比較されます。
period	（任意）is_percent が真に設定される場合、時間のパーセンテージが計算されます。これ以外の場合、収集されるサンプルの平均が計算される時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。

```
6. mem_tot_avail [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

引数

op	（任意）使用可能な収集されたメモリを指定された値と比較するために使用される比較演算子。真の場合、このイベントが発生します。
val	（任意）キロバイト単位で指定される、パーセンテージまたは絶対値。パーセンテージは、指定された時間内で最も古いサンプルと、最新のサンプルとの違いを表します。使用可能なメモリ使用量が時間内で 300 KB から 150 KB に減った場合、減少パーセンテージは 50 です。これは、測定された値を比較する必要がある値です。
is_percent	（任意）真の場合、パーセンテージの値が収集され、比較されます。これ以外の場合、絶対値が収集され、比較されます。
period	（任意）is_percent が真に設定される場合、時間のパーセンテージが計算されます。これ以外の場合、収集されるサンプルの平均が計算される時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。

```
7. mem_tot_used [op gt|ge|eq|ne|lt|le] [val ?] [is_percent TRUE|FALSE] [period ?]
```

引数

op	（任意）収集された使用メモリを、指定された値と比較するために使用される比較演算子。真の場合、このイベントが発生します。
val	（任意）キロバイト単位で指定される、パーセンテージまたは絶対値。パーセンテージは、指定された時間内で最も古いサンプルと、最新のサンプルとの違いを表します。メモリ使用量が時間内で 150 KB から 300 KB に増えた場合、増加パーセンテージは 100 です。これは、測定された値を比較する必要がある値です。
is_percent	（任意）真の場合、パーセンテージの値が収集され、比較されます。これ以外の場合、絶対値が収集され、比較されます。
period	（任意）is_percent が真に設定される場合、時間のパーセンテージが計算されます。これ以外の場合、収集されるサンプルの平均が計算される時間（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。この引数が指定されない場合は、最新のサンプルが使用されます。 **Note** is_percent が真に設定されている場合、この引数は必須です。これ以外の場合、この引数は任意です。

結果文字列

なし

_cerrno を設定

なし

Event_reqinfo

```
"event_id %u event_type %u event_type_string {%s} %u event_pub_sec %u event_pub_msec %u"
"num_subs %u"
```

イベント タイプ	説明
event_id	パブリッシュされた該当イベントの ID を示す一意の番号。同一のイベントで複数のポリシーを実行可能であり、その場合、各ポリシーは同一の event_id を保持します。
event_type	イベントのタイプ。
event_type_string	このイベント タイプのイベントの名前を表す ASCII 文字列。
event_pub_sec event_pub_msec	EEM に対してイベントがパブリッシュされた、秒単位およびミリ秒単位の時間。
num_subs	サブイベント番号。

サブイベント情報文字列は、次のような、デッドロック サブイベント用です。

```
"{type %s num_entries %u entries {entry 1, entry 2, ...}}"
```

サブイベントタイプ	説明
type	Wdsysmon サブイベントのタイプ。
num_entries	デッドロックのプロセスおよびスレッドの番号。
entries	デッドロックのプロセスおよびスレッドの情報。

各エントリは次のとおりです。

```
"{node {%s} procname {%s} pid %u tid %u state %s b_node %s b_procname %s b_pid %u
b_tid %u}"
```

このエントリでは、プロセス A のスレッド m によって、プロセス B のスレッド n でブロックされるシナリオが記述されているとすると、次のようになります。

サブイベントタイプ	説明
node	プロセス A のスレッド m があるノードの名前。
procname	プロセス A の名前。
pid	プロセス A のプロセス ID。

サブイベントタイプ	説明
tid	プロセス A のスレッド m のスレッド ID。
state	プロセス A のスレッド m のスレッド状態。次のいずれかになります。 • STATE_CONDVAR • STATE_DEAD • STATE_INTR • STATE_JOIN • STATE_MUTEX • STATE_NANOSLEEP • STATE_READY • STATE_RECEIVE • STATE_REPLY • STATE_RUNNING • STATE_SEM • STATE_SEND • STATE_SIGSUSPEND • STATE_SIGWAITINFO • STATE_STACK • STATE_STOPPED • STATE_WAITPAGE • STATE_WAITTHREAD
b_node	プロセス B のスレッドがあるノードの名前。
b_procname	プロセス B の名前。
b_pid	プロセス B のプロセス ID。
b_tid	プロセス B のスレッド n のスレッド ID。0 は、プロセス A のスレッド m は、プロセス B のすべてのスレッド上でブロックされることを意味します。

dispatch_mgr サブイベントについて

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

サブイベント タイプ	説明
type	Wdsysmon サブイベントのタイプ。
node	POSIX プロセスが存在するノードの名前。
procname	このサブイベントの POSIX プロセス名。
pid	このサブイベントの POSIX プロセス ID。 **Note** 前述の 3 つのフィールドは、このディスパッチ マネージャのオーナー プロセスについて説明します。
value	**sec** 変数と **msec** 変数が、0 に指定されている、またはイベント登録 Tcl コマンド拡張で指定されていない場合、ディスパッチ マネージャによって処理されたイベント数は、最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、このディスパッチ マネージャによって処理されるイベントの合計数は、該当する時間ウィンドウにあります。
sec msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。登録 Tcl コマンド拡張で時間ウィンドウが指定され、かつその時間ウィンドウがゼロよりも大きい場合、**sec** 変数および **msec** 変数は、この時間ウィンドウの最も古いサンプルと最新のサンプルとの実際の時間の差分です。

cpu_proc サブイベントについて

```
"{type %s node {%s} procname {%s} pid %u value %u sec %ld msec %ld}"
```

サブイベント タイプ	説明
type	Wdsysmon サブイベントのタイプ。
node	POSIX プロセスが存在するノードの名前。
procname	このサブイベントの POSIX プロセス名。
pid	このサブイベントの POSIX プロセス ID。 **Note** 前述の 3 つのフィールドは、その CPU 使用率がモニターされているプロセスについて説明します。

サブイベント タイプ	説明
value	**sec** 変数と **msec** 変数が 0 に指定されている、またはイベント登録 Tcl コマンド拡張で指定されていない場合、プロセスの CPU 使用率は最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、プロセス CPU 使用率の平均は、該当する時間ウィンドウにあります。
sec　msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。登録 Tcl コマンド拡張で時間ウィンドウが指定され、かつその時間ウィンドウがゼロよりも大きい場合、**sec** 変数および **msec** 変数は、この時間ウィンドウの最も古いサンプルと最新のサンプルとの実際の時間の差分です。

cpu_tot サブイベントについて

```
"{type %s node {%s} value %u sec %ld msec %ld}"
```

サブイベント タイプ	説明
type	Wdsysmon サブイベントのタイプ。
node	CPU 使用率の合計がモニターされているノードの名前。
value	**sec** 変数と **msec** 変数が 0 に指定されている、またはイベント登録 Tcl コマンド拡張で指定されていない場合、合計 CPU 使用率は最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、合計 CPU 使用率の平均は、該当する時間ウィンドウにあります。
sec　msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。登録 Tcl コマンド拡張で時間ウィンドウが指定され、かつその時間ウィンドウがゼロよりも大きい場合、**sec** 変数および **msec** 変数は、この時間ウィンドウの最も古いサンプルと最新のサンプルとの実際の時間の差分です。

mem_proc サブイベントについて

```
"{type %s node {%s} procname {%s} pid %u is_percent %s value %u diff %d sec %ld msec
%ld}"
```

サブイベント タイプ	説明
type	Wdsysmon サブイベントのタイプ。

サブイベントタイプ	説明
node	POSIX プロセスが存在するノードの名前。
procname	このサブイベントの POSIX プロセス名。
pid	このサブイベントの POSIX プロセス ID。 **Note** 前述の 3 つのフィールドは、そのメモリ使用率がモニターされているプロセスについて説明します。
is_percent	TRUE または FALSE のいずれかです。TRUE は、値がパーセント値であることを示します。FALSE は、値が絶対値であることを示します（平均値の場合もあります）。
value	**sec** 変数と **msec** 変数が 0 に指定されている、またはイベント登録 Tcl コマンド拡張で指定されていない場合、プロセスで使用されたメモリは最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、プロセスで使用されたメモリ使用率の平均は、該当する時間ウィンドウにあります。
サブイベントタイプ	説明
diff	**sec** 変数と **msec** 変数が 0 に指定されている、またはイベント登録 Tcl コマンド拡張で指定されていない場合、**diff** は、今まで収集されたプロセスで使用されたメモリの最初のサンプルと、プロセスで使用されたメモリの最新サンプルのパーセンテージの差分です。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、**diff** は、プロセスで使用されたメモリの使用状況のうち、指定された時間ウィンドウで最も古い値と最新の値のパーセンテージの差分です。
sec　msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。登録 Tcl コマンド拡張で時間ウィンドウが指定され、かつその時間ウィンドウがゼロよりも大きい場合、**sec** 変数および **msec** 変数は、この時間ウィンドウの最も古いサンプルと最新のサンプルとの実際の時間の差分です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で **sec** 引数と **msec** 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **value** は最新のサンプルでプロセスによって使用されたメモリです。
- **diff** は 0 です。
- **sec** と **msec** は両方とも 0 です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

- **value** は指定された時間ウィンドウでプロセスによって使用されたメモリ サンプル値の平均です。
- **diff** は 0 です。
- **sec** および **msec** は、両方とも、この時間ウィンドウ内の最も古いサンプルのタイムスタンプと最新のサンプルのタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

- **value** は 0 です。
- **diff** は指定された時間ウィンドウの、最も古いプロセスで使用されたメモリ サンプルと最新のプロセスで使用されたメモリ サンプルとのパーセンテージによる差分です。
- **sec** および **msec** は、プロセスで使用されたメモリ サンプルの、この時間ウィンドウ内の最も古いタイムスタンプと最新のプタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で **sec** 引数と **msec** 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **value** は 0 です。
- **diff** は今まで収集された、最初のプロセスで使用されたメモリ サンプルと、最新のプロセスで使用されたメモリ サンプルとのパーセンテージによる差分です。
- **sec** および **msec** は、今まで収集されたプロセスで使用されたメモリの最初のサンプルのタイムスタンプと、プロセスで使用されたメモリの最新のサンプルのタイムスタンプの実際の時間の差分です。

mem_tot_avail サブイベントについて

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

サブイベント タイプ	説明
type	Wdsysmon サブイベントのタイプ。
node	使用可能なメモリの合計がモニターされているノードの名前。
is_percent	TRUE または FALSE のいずれかです。TRUE は、値がパーセント値であることを示します。FALSE は、値が絶対値であることを示します（平均値の場合もあります）。

サブイベントタイプ	説明
used	**sec** 変数と **msec** 変数が、0 に指定されるか、または、イベント登録 Tcl コマンド拡張で指定されない場合、使用されたメモリの合計は、最新のサンプルにあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、使用された合計メモリ使用率の平均は、該当する時間ウィンドウにあります。
avail	**sec** 変数と **msec** 変数が 0 に指定されている、または、イベント登録 Tcl コマンド拡張で指定されていない場合、**avail** は使用可能な総メモリの最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、**avail** は、指定された時間ウィンドウ内での使用可能な総メモリの使用率です。
diff	**sec** 変数と **msec** 変数が 0 に指定されている、または、イベント登録 Tcl コマンド拡張で指定されていない場合、**diff** は、今まで収集された使用可能な総メモリの最初のサンプルと、使用可能な総メモリの最新サンプルのパーセンテージの差分です。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、**diff** は、使用可能な総メモリの使用率のうち、指定された時間ウィンドウで最も古い値と最新の値のパーセンテージの差分です。
sec msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、これらの変数は、この時間ウィンドウの、最も古いサンプルと最新のサンプルとの実際の時間の差分です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で **sec** 引数と **msec** 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **used** は最新のサンプルで使用されたメモリの合計です。
- **avail** は最新のサンプルで使用可能なメモリの合計です。
- **diff** は 0 です。
- **sec** と **msec** は両方とも 0 です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

- **used** は 0 です。
- **avail** は指定された時間ウィンドウで使用可能な合計メモリ サンプル値の平均です。
- **diff** は 0 です。
- **sec** および **msec** は、両方とも、この時間ウィンドウ内の使用可能な総メモリの最も古いサンプルのタイムスタンプと最新サンプルのタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

- **used** は 0 です。
- **avail** は 0 です。
- **diff** 指定された時間ウィンドウの、最も古い使用可能なメモリ サンプルの合計と最新の可能なメモリ サンプルの合計とのパーセンテージによる差分です。
- **sec** および **msec** は、両方とも、この時間ウィンドウ内の使用可能な総メモリの最も古いサンプルのタイムスタンプと最新サンプルのタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で **sec** 引数と **msec** 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **used** は 0 です。
- **avail** は 0 です。
- **diff** 今まで収集された、最初の使用可能なメモリ サンプルの合計と、最新の使用可能なメモリ サンプルの合計との間の、パーセンテージによる差です。
- **sec** および **msec** は、今まで収集された使用可能な総メモリの最初のサンプルのタイムスタンプと、使用可能な総メモリの最新サンプルのタイムスタンプ間の実際の時間の差です。

mem_tot_used サブイベントについて

```
"{type %s node {%s} is_percent %s used %u avail %u diff %d sec %ld msec %ld}"
```

サブイベントタイプ	説明
type	Wdsysmon サブイベントのタイプ。
node	使用されているメモリの合計がモニターされているノードの名前。
is_percent	TRUE または FALSE のいずれかです。TRUE は、値がパーセント値であることを示します。FALSE は、値が絶対値であることを示します（平均値の場合もあります）。
used	**sec** 変数と **msec** 変数が、0 に指定されるか、または、イベント登録 Tcl コマンド拡張で指定されない場合、使用されたメモリの合計は、最新のサンプルにあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、使用された合計メモリ使用率の平均は、該当する時間ウィンドウにあります。

サブイベントタイプ	説明
avail	**sec** 変数と **msec** 変数が、0 に指定されている、または、イベント登録 Tcl コマンド拡張で指定されていない場合、**avail** は使用されたメモリ合計の最新サンプル内にあります。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、**avail** は、指定された時間ウィンドウ内での使用されたメモリ合計の使用状況です。
diff	**sec** 変数と **msec** 変数が 0 に指定されている、または、イベント登録 Tcl コマンド拡張で指定されていない場合、**diff** は、今まで収集された使用されたメモリ合計の最初のサンプルと、使用されたメモリ合計の最新サンプルのパーセンテージの差分です。時間ウィンドウが指定され、登録 Tcl コマンド拡張でゼロよりも大きい場合、**diff** は、使用されたメモリ合計の使用状況のうち、指定された時間ウィンドウで最も古い値と最新の値のパーセンテージの差分です。
sec msec	イベント登録 Tcl コマンド拡張で、**sec** 変数と **msec** 変数が 0 に指定されているか、または指定されていない場合、両方とも 0 です。登録 Tcl コマンド拡張で時間ウィンドウが指定され、かつその時間ウィンドウがゼロよりも大きい場合、**sec** 変数および **msec** 変数は、この時間ウィンドウの最も古いサンプルと最新のサンプルとの実際の時間の差分です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で sec 引数と msec 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **used** は最新のサンプルで使用されたメモリの合計です。
- **avail** は最新のサンプルで使用可能なメモリの合計です。
- **diff** は 0 です。
- **sec** と **msec** は両方とも 0 です。

is_percent 引数が FALSE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

- **used** は指定された時間ウィンドウで使用された合計メモリ サンプル値の平均です。
- **avail** は 0 です。
- **diff** は 0 です。
- **sec** および **msec** は、両方とも、この時間ウィンドウ内の使用されたメモリ合計の最も古いサンプルのタイムスタンプと最新サンプルのタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で時間ウィンドウがゼロよりも大きい値に指定されている場合は、次のようになります。

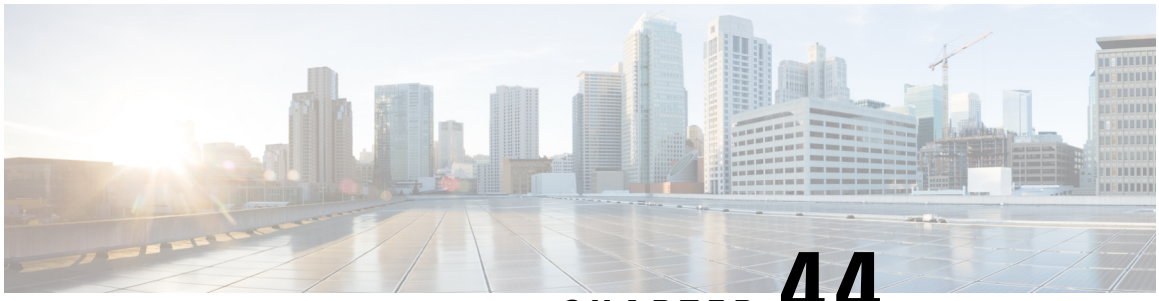
- **used** は 0 です。

- **avail** は 0 です。
- **diff** 指定された時間ウィンドウの、使用された最も古いメモリ サンプルの合計と使用された最新のメモリ サンプルの合計とのパーセンテージによる差分です。
- **sec** および **msec** は、両方とも、この時間ウィンドウ内の使用されたメモリ合計の最も古いサンプルのタイムスタンプと最新サンプルのタイムスタンプの実際の時間の差分です。

is_percent 引数が TRUE であり、イベント登録 Tcl コマンド拡張で sec 引数と msec 引数が 0 に指定されているか、または指定されていない場合は、次のようになります。

- **used** は 0 です。
- **avail** は 0 です。
- **diff** は今まで収集された、使用された最初のメモリ サンプルの合計と、使用された最新のメモリ サンプルの合計との間のパーセンテージによる差です。
- **sec** および **msec** は、今まで収集された使用されたメモリ合計の最初のサンプルのタイムスタンプと、使用されたメモリ合計の最新サンプルのタイムスタンプ間の実際の時間の差です。

Note サブイベントの説明内部では、各引数は、位置に依存しません。



CHAPTER 44

EEM イベントの Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

event_completion

トリガーしたイベントのサービスが行われている EEM サーバーに、通知を送信します。イベントでは、このイベント インスタンスの **return_code** である 1 つの引数のみが使用されます。

構文

```
event_completion status ?
```

引数

status	（必須）このイベントインスタンスの終了ステータス（return_code）。ゼロの値によって、エラーがないことが示され、他のすべての整数によって、エラーが示されます。

結果文字列

なし

_cerrno を設定

非対応

event_completion_with_wait

event_completion_with_wait コマンドは、2 つのコマンド、**event_completion** と **event_wait** を使いやすいように 1 つのコマンドに組み合わせたものです。

event_completion コマンドによって、ポリシーをトリガーしたイベントに対してポリシーがサービスを実行したことが EEM サーバーに通知されます。イベントでは、このイベント インスタンスの **return_code** である 1 つの引数のみが使用されます。

event_wait ポリシーがスリープ状態になります。Tcl ポリシーで、新しいイベントを通知する新しい信号を受信すると、ポリシーは使用状態になり、再度スリープ状態に戻ります。このループが継続されます。**event_wait** ポリシーは、**event_completed** ポリシーの前に起動され、エラーが発生して、ポリシーが終了します。

構文

```
event_completion_with_wait status ? [refresh_vars]
```

引数

status	（必須）このイベント インスタンスの exit_status（return_code）。ゼロの値は、エラーがないことを示します。他のすべての整数は、エラーを示します。
refresh_vars	（任意）組み込み変数と環境変数は、このイベントインスタンス中に EEM Policy Director からアップデート（リフレッシュ）する必要があるかどうかを示します。

結果文字列

なし

_cerrno を設定

対応

使用例

この 1 つのコマンドを使用した前述の例の類似例を示します。

```
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set i 1
while {1 == 1} {  # Start high performance policy loop
   array set arr_einfo [event_reqinfo]
   if {$_cerrno != 0} {
       set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
       $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
       error $result
   }
   action_syslog msg "event $i serviced" priority info
   if {$i == 5} {
       action_syslog msg "Exiting after servicing 5 events" priority info
       exit 0
   }
   incr i
   array set _event_state_arr [event_completion_with_wait status 0 refresh_vars 1]
   if {$_event_state_arr(event_state) != 0} {
       action_syslog msg "Exiting: failed event_state " \
                         " $event_state_arr(event_state)" priority info
       exit 0
   }
}
```

Note 実行される設定の出力は、event_publish Tcl コマンドと同じです。

event_publish

アプリケーション固有のイベントをパブリッシュします。

構文

```
event_publish sub_system ? type ? [arg1 ?] [arg2 ?] [arg3 ?] [arg4 ?]
```

引数

sub_system	（必須）アプリケーション固有のイベントをパブリッシュしたEEMポリシーに割り当てられる番号。他のすべての番号はCiscoでの使用のために予約されているため、番号は 798 に設定されます。
type	（必須）指定されたコンポーネント内のイベント サブタイプ。sub_system 引数および type 引数によって、アプリケーション イベントが一意に識別されます。1 ～ 4294967295 の範囲の整数である必要があります。
[arg1 ?]-[arg4 ?]	（任意）4 つのアプリケーション イベントのパブリッシャの文字列データ。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

使用例

次に、ある機能（Tcl ステートメントの所定のグループによって CPU 時間の長さを測定するなど）を実行するため、**event_publish** Tcl コマンド拡張を使用してスクリプトを *n* 回、反復して実行する例を示します。この例では、2 つの Tcl スクリプトが使用されます。

Script1 によって、タイプ 9999 EEM イベントがパブリッシュされ、Script2 の 1 回目の実行が行われます。Script1 は、none イベントとして登録され、Cisco IOS CLI **event manager run** コマンドを使用して実行されます。Script2 は、タイプ 9999 の EEM アプリケーション イベントとして登録され、このスクリプトによって、アプリケーションによってパブリッシュされた arg1 データ（繰り返し回数）が、EEM 環境変数 test_iterations の値を超過したかどうかがチェックされます。test_iterations の値が超えた場合、スクリプトによってメッセージが書き込まれ、終了します。これ以外の場合、スクリプトによって残りの文が実行され、別の実行が再スケジュールされます。Script2 の CPU 使用率を測定するには、10 の倍数である test_iterations の値を使用して、Script2 によって使用される CPU 時間の平均の長さを計算します。

Tcl スクリプトを実行するには、次の Cisco IOS コマンドを使用します。

```
configure terminal
 event manager environment test_iterations 100
 event manager policy script1.tcl
 event manager policy script2.tcl
 end
event manager run script1.tcl
```

Tcl スクリプト Script2 によって、100 回実行されます。余分な処理なしてスクリプトを実行し、CPU 使用率の平均を導き出し、次に余分な処理を追加して、テストを繰り返す場合、以降の CPU 使用率から前の CPU 使用率を差し引き、余分な処理の平均を調べることができます。

Script1（script1.tcl）

```
::cisco::eem::event_register_none
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
# Query the event info.
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
```

```
}

action_syslog priority info msg "EEM application_publish test start"
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}

# Cause the first iteration to run.
event_publish sub_system 798 type 9999 arg1 0
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

Script2（script2.tcl）

```
::cisco::eem::event_register_appl sub_system 798 type 9999

# Check if all the required environment variables exist.
# If any required environment variable does not exist, print out an error msg and quit.
if {![info exists test_iterations]} {
    set result \
      "Policy cannot be run: variable test_iterations has not been set"
    error $result $errorInfo
}

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

# Query the event info.
array set arr_einfo [event_reqinfo]
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
# Data1 contains the arg1 value used to publish this event.
set iter $arr_einfo(data1)

# Use the arg1 info from the previous run to determine when to end.
if {$iter >= $test_iterations} {
    # Log a message.
    action_syslog priority info msg "EEM application_publish test end"
    if {$_cerrno != 0} {
      set result [format \
            "component=%s; subsys err=%s; posix err=%s;\n%s" \
            $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
      error $result
    }
    exit 0
}
set iter [expr $iter + 1]

# Log a message.
set msg [format "EEM application_publish test iteration %s" $iter]
action_syslog priority info msg $msg
if {$_cerrno != 0} {
    set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
      $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
```

```
    error $result
}

# Do whatever processing that you want to measure here.

# Cause the next iteration to run. Note that the iteration is passed to the
# next operation as arg1.
event_publish sub_system 798 type 9999 arg1 $iter
if {$_cerrno != 0} {
    set result [format \
          "component=%s; subsys err=%s; posix err=%s;\n%s" \
          $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
    error $result
}
```

event_wait

Tclポリシーがスリープ状態になります。Tclポリシーで、新しいイベントを通知する新しい信号を受信すると、ポリシーは使用状態になり、再度スリープ状態に戻ります。このループが継続されます。**event_wait** ポリシーは、**event_completed** ポリシーの前に起動され、エラーが発生して、ポリシーが終了します。

構文

```
event_wait [refresh_vars]
```

引数

refresh_vars	（任意）組み込み変数と環境変数は、このイベントインスタンス中にEEM Policy Directorからアップデート（リフレッシュ）する必要があるかどうかを示します。

結果文字列

なし

_cerrno を設定

なし

使用例

event_wait イベント ディテクタは、**event_state** という名前の単一要素でアレイ タイプ値を返します。Event_state は、イベントの処理中にエラーが発生したかどうかを示す EEM サーバーから戻される値です。この場合のエラーの例は、ユーザーがイベントインスタンスを処理するときに、**event_completion** を設定する前に **event_wait** を設定した場合のエラーを示しています。

次に、**event_completion** Tcl コマンドと**event_wait** コマンドの両方を使用した出力例を示します。

```
::cisco::eem::event_register_syslog tag e1 occurs 1 pattern CLEAR maxrun 0
namespace import ::cisco::eem::*
namespace import ::cisco::lib::*
set i 1
while {1 == 1} {  # Start high performance policy loop
   array set arr_einfo [event_reqinfo]
   if {$_cerrno != 0} {
       set result [format "component=%s; subsys err=%s; posix err=%s;\n%s" \
       $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str]
       error $result
   }
   action_syslog msg "event $i serviced" priority info
   if {$i == 5} {
       action_syslog msg "Exiting after servicing 5 events" priority info
       exit 0
   }
   incr i
   event_completion status 0
   array set _event_state_arr [event_wait refresh_vars 0]
   if {$_event_state_arr(event_state) != 0} {
       action_syslog msg "Exiting: failed event_state " \
                         " $event_state_arr(event_state)" priority info
       exit 0
   }
}
```

次に、実行コンフィギュレーションの例を示します。

```
Device#
01:00:44: %SYS-5-CONFIG_I: Configured from console by consoleclear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:00:49: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:49: %HA_EM-6-LOG: high_perf_example.tcl: event 1 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:00:53: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:53: %HA_EM-6-LOG: high_perf_example.tcl: event 2 serviced
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:00:56: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:00:56: %HA_EM-6-LOG: high_perf_example.tcl: event 3 serviced
Device#
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:00:59: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
01:00:59: %HA_EM-6-LOG: high_perf_example.tcl: event 4 serviced
01:00:59: %HA_EM-6-LOG: high_perf_example.tcl: Exiting after servicing 5 events
Device#
Device#
Device#copy tftp disk1:
Address or name of remote host [dirt]?
Source filename [user/eem_scripts/high_perf_example.tcl]?
Destination filename [high_perf_example.tcl]?
%Warning:There is a file already existing with this name
```

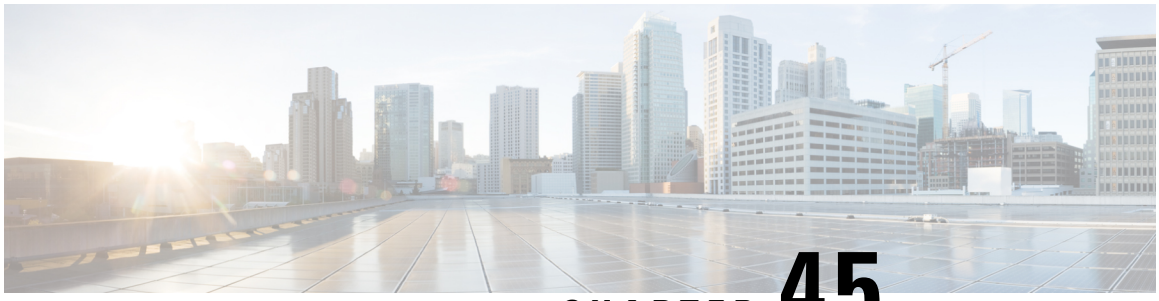
```
Do you want to over write? [confirm]
Accessing tftp://dirt/user/eem_scripts/high_perf_example.tcl...
Loading user/eem_scripts/high_perf_example.tcl from 192.0.2.19 (via FastEthernet0/0): !
[OK - 909 bytes]
909 bytes copied in 0.360 secs (2525 bytes/sec)
Device#
Device#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)#no event manager policy high_perf_example.tcl
Device(config)#event manager po high_perf_example.tcl
Device(config)#end
Device#
Device#
Device#
Device#
01:02:19: %SYS-5-CONFIG_I: Configured from console by consoleclear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
01:02:23: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
Device#
01:02:23: %HA_EM-6-LOG: high_perf_example.tcl: event 1 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:26: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:26: %HA_EM-6-LOG: high_perf_example.tcl: event 2 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:29: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:29: %HA_EM-6-LOG: high_perf_example.tcl: event 3 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
01:02:33: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
Device#
01:02:33: %HA_EM-6-LOG: high_perf_example.tcl: event 4 serviced
Device#
Device#clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#
Device#
Device#
01:02:36: %CLEAR-5-COUNTERS: Clear counter on all interfaces by console
01:02:36: %HA_EM-6-LOG: high_perf_example.tcl: event 5 serviced
01:02:36: %HA_EM-6-LOG: high_perf_example.tcl: Exiting after servicing 5 events
Device#
```

また、イベントがサービスされ、次のイベントの到着を待っている間に、**show event manager policy active** コマンドによって、次の出力が表示されます。

```
Device#show event manager policy active
Key: p - Priority        :L - Low, H - High, N - Normal, Z - Last
     s - Scheduling node :A - Active, S - Standby
default class - 1 script event
 no.  job id      p s status  time of event             event type          name
```

```
 1    11          N A wait    Mon Oct20 14:15:24 2008   syslog
high_perf_example.tcl
```

前述の例では、ステータスは待ち状態です。これは、ポリシーが次のイベントの到着を待っていることを示します。



CHAPTER 45

EEM ライブラリのデバッグ コマンド拡張

cli_debug

コマンドライン インターフェイス（CLI）のデバッグ文を、Syslog に出力します。**debug event manager tcl cli_library** Cisco IOS コマンドが有効な場合に、この Tcl コマンド拡張を使用すると、CLI デバッグステートメントが syslog に出力されます。

構文

```
cli_debug spec_string debug_string
```

引数

spec_string	（必須）spec_string 引数が使用され、デバッグ文のタイプを示します。
debug_string	（必須）debug_string 引数が使用され、デバッグ テキストを示します。

結果文字列

なし

_cerrno を設定

非対応

smtp_debug

シンプル メール転送プロトコル（SMTP）のデバッグ文を、Syslog に出力します。**debug event manager tcl smtp_library** のコマンドライン インターフェイス（CLI）コマンドが有効な場合に、この Tcl コマンド拡張によって、SMTP デバッグ文が Syslog に出力されます。

構文

```
smtp_debug spec_string debug_string
```

引数

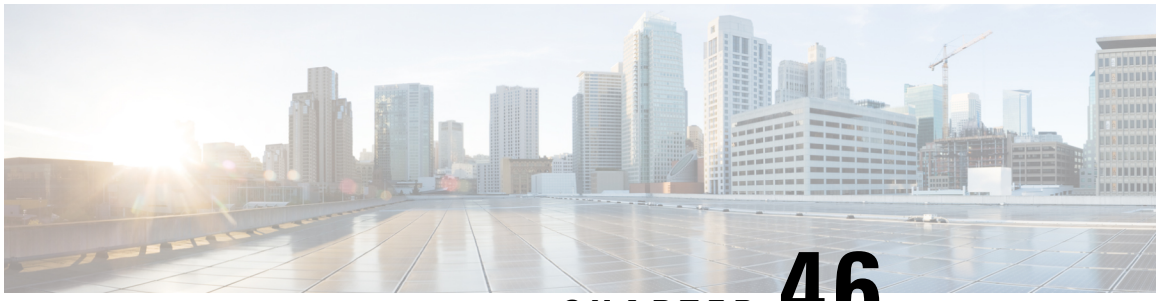
spec_string	（必須）spec_string 引数が使用され、デバッグ文のタイプを示します。
debug_string	（必須）debug_string 引数が使用され、デバッグ テキストを示します。

結果文字列

なし

_cerrno を設定

非対応



CHAPTER 46

EEM 複数イベントサポートの Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

attribute

複雑なイベントを指定します。

構文

```
attribute tag ? [occurs ?]
```

引数

tag	イベントを関連付けるために **attribute** コマンドで使用できる *event-tag* 引数を使用して、タグを指定します。
occurs	（任意）EEM イベントがトリガーされる前の発生数を指定します。指定されない場合、EEM イベントは 1 回目から発生します。範囲は 1 ～ 4294967295 です。

結果文字列

なし

_cerrno を設定

非対応

correlate

イベントおよびトラックされるオブジェクトに関連する、1つの複雑なイベントを構築し、ブール値のロジックを使用します。

構文

```
correlate event ? track ? [andnot | and | or] event ? track ?
```

引数

event	スクリプト内で複数のイベント文をサポートするために、**trigger** コマンドで使用できるイベントを指定します。 *event-tag* 引数に関連付けられているイベントが、**trigger** コマンドによって指定されて何度も発生する場合、結果は真です。これ以外の場合、結果は偽です。
track	トラックするイベント オブジェクト番号を指定します。指定できる範囲は 1 ～ 500 です。 トラックされるオブジェクトが設定されている場合、評価の結果は真です。トラックされるオブジェクトが未設定または未定義の場合、評価の結果は偽です。この結果は、オブジェクトの状態には関係ありません。
andnot	（任意）イベント 1 が発生した場合にアクションが実行され、さらに、イベント 2 およびイベント 3 が一緒に発生した場合にはアクションが実行されないよう、指定します。

および	（任意）イベント 1 が発生した場合にアクションが実行され、さらに、イベント 2 およびイベント 3 が一緒に発生した場合にアクションが実行されるよう、指定します。 **Note** 「and」を使用して、トラップや syslog メッセージなどのイベントをグループ化した場合、デフォルトのトリガー発生時間枠は 3 分です。
または	（任意）イベント 1 が発生した場合にアクションが実行されるか、または、イベント 2 およびイベント 3 が一緒に発生した場合にアクションが実行されるよう、指定します。

結果文字列

なし

_cerrno を設定

非対応

trigger

Embedded Event Manager（EEM）イベントの複数イベントの設定機能を指定します。複数イベントは、1つまたは複数のイベント発生、1つまたは複数のトラックされるオブジェクト状態、および発生するイベントの時間を起動できるイベントです。イベントは指定されたパラメータに基づいて発生します。

構文

```
trigger [occurs ?] [period ?] [period-start ?] [delay ?]
```

引数

occurs	（任意）EEMイベントが発生する前に発生した合計相関回数を指定します。数が指定されない場合、EEM イベントは 1 回目から発生します。範囲は 1 ～ 4294967295 です。
period	（任意）1 つまたは複数が発生する必要がある間の、秒単位、および、任意でミリ秒単位での、時間の間隔。これは、sssssssss[.mmm] 形式で指定します。sssssssss は、0 ～ 4294967295 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります。
period-start	（任意）イベント相関ウィンドウの開始を指定します。指定されない場合、最初の CRON 期間の発生後、イベント監視はイネーブルにされます。

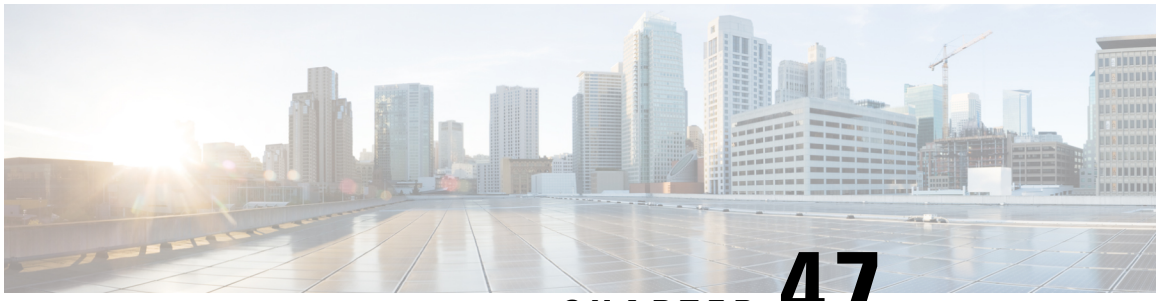
delay	（任意）すべての条件が真の場合にイベントの発生後の秒数とミリ秒数（任意）を指定します（sssssssss[.mmm]形式で指定します。sssssssss は、0 ～ 4294967295 の秒数を表す整数で、mmm は 0 ～ 999 のミリ秒数を表す整数である必要があります）。

結果文字列

なし

_cerrno を設定

非対応



CHAPTER 47

EEM SMTP ライブラリのコマンド拡張

すべてのシンプル メール転送プロトコル（SMTP）ライブラリ コマンドは、::cisco::lib 名前空間に属します。

このライブラリを使用するには、ユーザーは、電子メール テンプレート ファイルを用意する必要があります。テンプレートファイルに Tcl グローバル変数を含めると、**event manager environment Cisco IOS** コマンドライン インターフェイス（CLI）コンフィギュレーション コマンドを使用して電子メールサービスと電子メールテキストを設定できるようになります。電子メールテンプレートファイルでグローバル変数を置き換え、設定された電子メールサーバーを使用して、設定された To アドレス、CC アドレス、From アドレス、および Subject 行プロパティで必要な電子メールコンテキストを送信するには、このライブラリにあるコマンドを使用します。

電子メール テンプレート

電子メール テンプレート ファイルの形式は、次のとおりです。

Note RFC 2554 に基づき、SMTP 電子メール サーバー名 Mailservername には、username:password@host、username@host、または host のテンプレート形式のいずれか 1 つを使用できます。

```
Mailservername:<space><the list of candidate SMTP server addresses>
From:<space><the e-mail address of sender>
To:<space><the list of e-mail addresses of recipients>
Cc:<space><the list of e-mail addresses that the e-mail will be copied to>
Sourceaddr:<space><the IP addresses of the recipients>
Subject:<subject line>
<a blank line>
<body>
```

Note テンプレートには、通常、設定のための Tcl グローバル変数が含まれていることに注意してください。

Tcl ポリシーでは、電子メール テンプレートの「Port」行でポート番号を指定できます。ポートを指定しなかった場合、デフォルトのポート 25 が使用されます。

次に、サンプル E メール テンプレート ファイルを挙げます。

```
Mailservername: $_email_server
From: $_email_from
To: $_email_to
Cc: $_email_cc
Sourceaddr: $_email_ipaddr
Port: <port number>
Subject: From router $routername: Process terminated
process name: $process_name
subsystem: $sub_system
exit status: $exit_status
respawn count: $respawn_count
```


- smtp_send_email, on page 920
- smtp_subst, on page 921

smtp_send_email

電子メール テンプレート ファイルのテキストが、すべてのグローバル変数ですでに置き換えられている場合、シンプル メール転送プロトコル（SMTP）を使用して電子メールを送信します。電子メール テンプレートによって、候補メール サーバーのアドレス、To アドレス、CC アドレス、From アドレス、件名の行、および電子メールの本文が指定されます。

Note ライブラリが、リストにあるサーバーの 1 つに接続できるまで、サーバーへの接続が、1 つ 1 つ試行されるよう、候補電子メール サーバーのリストを用意できます。

構文

```
smtp_send_email text
```

引数

text	（必須）すべてのグローバル変数ですでに置き換えられた、E メール テンプレート ファイルのテキスト。

結果文字列

なし

_cerrno を設定

- 1 行目の形式が間違っている：Mailservername：サーバー名のリスト。
- 2 行目の形式が間違っている：From：送信元アドレス。

- 3 行目の形式が間違っている：To：送信先アドレスのリスト。
- 4 行目の形式が間違っている：CC：コピー送信先アドレスのリスト。
- メール サーバーへの接続エラー：リモート サーバーによって $sock が閉じられている（$sock はメール サーバーに開かれているソケットの名前）。
- メール サーバーへの接続エラー：$sock 応答コードが service ready greeting ではなく $k である（$sock はメール サーバーに開かれているソケットの名前、$k は $sock の応答コード）。
- メール サーバーへの接続エラー：すべてのメール サーバー候補に接続できない。
- メール サーバーからの接続解除エラー：リモート サーバーによって $sock が閉じられている（$sock はメール サーバーに開かれているソケットの名前）。

サンプル スクリプト

電子メールテンプレートですべての必要なグローバル変数が定義された後には、次のようになります。

```
if [catch {smtp_subst [file join $tcl_library email_template_sm]} result] {
    puts stderr $result
    exit 1
}
if [catch {smtp_send_email $result} result] {
    puts stderr $result
    exit 1
}
```

smtp_subst

電子メール テンプレート ファイル e-mail_template の場合、ファイルにある各グローバル変数を、そのユーザー定義値によって置き換えます。置換後に、ファイルのテキストを返します。

構文

```
smtp_subst e-mail_template
```

引数

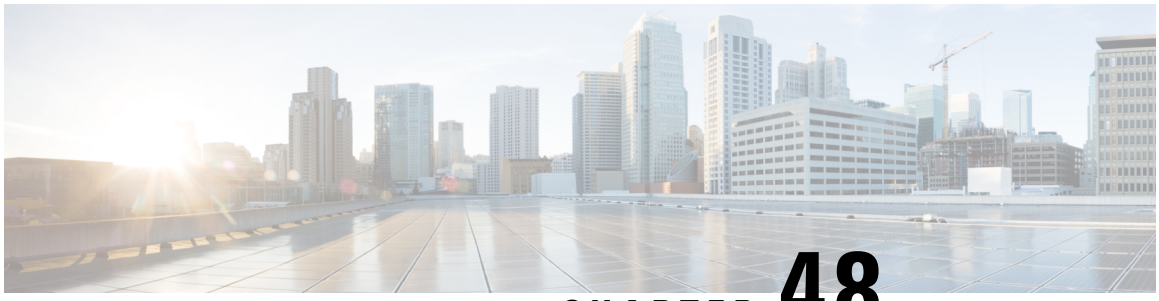
e-mail_template	（必須）グローバル変数が、ユーザー定義値によって置き換えられる必要がある、電子メール テンプレート ファイルの名前。ファイル名の例は /disk0://example.template で、スロット 0 の ATA フラッシュ ディスクの上位レベルディレクトリにある example.template という名前のファイルを表します。

結果文字列

すべてのグローバル変数で置き換えられた、電子メール テンプレート ファイルのテキスト。

_cerrno を設定

- 電子メール テンプレート ファイルを開けられない。
- 電子メール テンプレート ファイルを閉じられない。



CHAPTER 48

EEM システム情報の Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM システム情報コマンド（**sys_reqinfo_xxx**）では、Set _cerrno セクションが yes に設定されています。

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

• sys_reqinfo_syslog_freq, on page 932
• sys_reqinfo_syslog_history, on page 934

sys_reqinfo_cli_freq

すべてのコマンドライン インターフェイス（CLI）イベントの頻度情報を問い合わせます。

構文

```
sys_reqinfo_cli_freq
```

引数

なし

結果文字列

```
rec_list {{CLI frequency string 0},{CLI frequency str 1}, ...}
```

各 CLI の頻度の文字列は、次のとおりです。

time_sec %ld time_msec %ld match_count %u raise_count %u occurs %u period_sec %ld period_msec %ld pattern {%s}

rec_list	CLI イベント頻度リストの開始をマークします。
time_sec time_msec	この CLI イベントが発生した最後の時刻。
match count	CLI イベントによって指定されたパターンが、CLI コマンドによって照会される回数。
raise_count	この CLI イベントが発生した回数。次のフィールドは、CLI イベント指定に関する情報です。 • sync：「yes」は、イベント パブリッシュが同期的に実行される必要があることを意味します。Event Manager Server がイベントのパブリッシュを完了したときに、イベントディテクタが通知されます。Event Manager Server は、CLI コマンドが実行される必要があるかどうかを示すコードを返します。 • skip：「yes」は、sync フラグが設定されているときに、CLI コマンドは実行してはいけないことを意味します。
occurs	イベントが発生する前の発生回数。この引数が指定されない場合、イベントは 1 回目から発生します。
period_sec period_msec	イベントを発生させるには、発生回数が POSIX タイマー ユニットのこの数以内である必要があります。この引数が指定されない場合は、適用されません。

pattern	CLI コマンドのパターン マッチの実行に使用される正規表現。

_cerrno を設定

対応

sys_reqinfo_cli_history

コマンドライン インターフェイス（CLI）コマンドの履歴を問い合わせます。

構文

```
sys_reqinfo_cli_history
```

引数

なし

結果文字列

```
rec_list {{CLI history string 0}, {CLI history str 1},...}
```

各 CLI の履歴の文字列は、次のとおりです。

```
time_sec %ld time_msec %ld cmd {%s}
```

rec_list	CLI コマンド履歴リストの開始をマークします。
time_sec time_msec	CLI コマンドが実行された時刻。
cmd	CLI コマンドのテキスト。

_cerrno を設定

対応

sys_reqinfo_cpu_all

指定された期間で、指定された順序で、上位プロセスの CPU 使用率（POSIX プロセスと IOS プロセスの両方）を問い合わせます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
sys_reqinfo_cpu_all order cpu_used [sec ?] [msec ?] [num ?]
```

引数

order	（必須）プロセスの CPU 使用率のソートに使用される順序。
cpu_used	（必須）指定されたウィンドウの、CPU 使用率の平均が、降順でソートされるよう、指定します。
sec msec	（任意）CPU 使用率の平均が計算される、秒単位およびミリ秒単位での時間。0 から 4294967295 の範囲の整数である必要があります。指定されない場合か、または、sec と msec の両方が 0 と指定される場合、最新の CPU サンプルが使用されます。
num	（任意）表示される、ソートされたプロセスのリストの上位からのエントリの数。1 ～ 4294967295 の範囲の整数である必要があります。デフォルト値は 5 です。

結果文字列

```
rec_list {{process CPU info string 0},{process CPU info string 1}, ...}
```

各プロセスの CPU 情報文字列は、次のとおりです。

```
pid %u name {%s} cpu_used %u
```

rec_list	プロセス CPU 情報リストの開始をマークします。
pid	プロセス ID。
name	プロセス名。
cpu_used	sec と msec が、ゼロよりも大きい数で指定される場合、平均パーセンテージは、指定された時間のプロセス CPU 使用率から計算されるよう、指定します。sec と msec が、両方ともゼロか、または指定されない場合。平均パーセンテージは、最新のサンプルのプロセス CPU 使用率から計算されます。

_cerrno を設定

対応

sys_reqinfo_crash_history

クラッシュしたすべてのプロセスのプロセス情報を問い合わせます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
sys_reqinfo_crash_history
```

引数

なし

結果文字列

```
rec_list {{crash info string 0},{crash info string 1}, ...}
Where each crash info string is:
job_id %u name {%s} respawn_count %u fail_count %u dump_count %u
inst_id %d exit_status 0x%x exit_type %d proc_state {%s} component_id 0x%x
crash_time_sec %ld crash_time_msec %ld
```

job_id	システム マネージャによってプロセスに割り当てられるジョブ ID。1 ～ 4294967295 の整数。
name	プロセス名。
respawn_count	プロセスの再起動の合計回数。
fail_count	プロセスの再起動試行の回数。プロセスが正常に再起動されると、このカウントはゼロにリセットされます。
dump_count	実行されたコア ダンプの回数。
inst_id	プロセス インスタンス ID。
exit_status	プロセスの最後の終了ステータス。
exit_type	最後の終了タイプ。
proc_state	Sysmgr プロセスの状態。error、forced_stop、hold、init、ready_to_run、run、run_rnode、stop、waitEOltimer、wait_rnode、wait_spawntimer、wait_tpl の 1 つです。
component_id	プロセスが属するコンポーネントのコンポーネント ID に割り当てられているバージョン マネージャ。
crash_time_sec crash_time_msec	1970 年 1 月 1 日以降の秒およびミリ秒の単位で、プロセスがクラッシュした最後の時刻を表します。

_cerrno を設定

対応

sys_reqinfo_mem_all

指定された期間で、指定された順序で、上位プロセスのメモリの使用状況（POSIX と IOS の両方）を問い合わせます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
sys_reqinfo_mem_all order allocates|increase|used [sec ?] [msec ?] [num ?]
```

引数

order	（必須）プロセスのメモリの使用状況のソートに使用される順序。
allocates	（必須）指定された時間ウィンドウの期間に、メモリの使用状況が、プロセス割り当ての数によって降順でソートされるよう、指定します。
increase	（必須）指定された時間ウィンドウの期間に、メモリの使用状況が、プロセスで増えたメモリのパーセンテージによって降順でソートされるよう、指定します。
used	（必須）メモリが、プロセスによって使用される現在のメモリによってソートされるよう、指定します。
sec msec	（任意）プロセスでのメモリの使用状況が計算される、秒単位およびミリ秒単位での時間。0 から 4294967295 の範囲の整数である必要があります。sec と msec の両方が指定され、ゼロではない場合、割り当て数は、該当する時間で収集された最も古いサンプルと最新のサンプルでの、割り当て数の差です。パーセンテージは、該当する時間で収集された最も古いサンプルと最新のサンプルとの、パーセンテージの差分として計算されます。指定されない場合か、または、sec と msec の両方が 0 と指定される場合、収集された最初のサンプルが、最も古いサンプルとして使用されます。つまり、時間は、起動から現在時までの時間で設定されます。
num	（任意）表示される、ソートされたプロセスのリストの上位からのエントリの数。1 ～ 4294967295 の範囲の整数である必要があります。デフォルト値は 5 です。

結果文字列

```
rec_list {{process mem info string 0},{process mem info string 1}, ...}
```

各プロセスのメモリ情報文字列は、次のとおりです。

```
pid %u name {%s} delta_allocs %d initial_alloc %u current_alloc %u percent_increase %d
```

rec_list	プロセスのメモリの使用状況情報リストの開始をマークします。
pid	プロセス ID。
name	プロセス名。
delta_allocs	該当する期間で収集された、最も古いサンプルと最新のサンプルでの、割り当て数の差として、差を指定します。
initial_alloc	時間の開始時にプロセスによって使用される、キロバイト単位での、メモリの容量を指定します。

current_alloc	プロセスによって使用される、キロバイト単位での、メモリの容量を指定します。
percent_increase	該当する時間で収集された最も古いサンプルと最新のサンプルとの、使用メモリのパーセンテージの差分を指定します。パーセンテージの差は、current_alloc から initial_alloc の 100 を差し引いた数として、および、initial_alloc で割った数として、表すことができます。

_cerrno を設定

対応

sys_reqinfo_proc

1 つの POSIX プロセスに関する情報を問い合わせます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
sys_reqinfo_proc job_id ?
```

引数

job_id	（必須）システム マネージャによってプロセスに割り当てられるジョブ ID。1 ～ 4294967295 の範囲の整数である必要があります。

結果文字列

```
job_id %u component_id 0x%x name {%s} helper_name {%s} helper_path {%s} path {%s}
node_name {%s} is_respawn %u is_mandatory %u is_hold %u dump_option %d
max_dump_count %u respawn_count %u fail_count %u dump_count %u
last_respawn_sec %ld last_respawn_msec %ld inst_id %u proc_state %s
level %d exit_status 0x%x exit_type %d
```

job_id	システム マネージャによってプロセスに割り当てられるジョブ ID。1 ～ 4294967295 の整数。
component_id	プロセスが属するコンポーネントのコンポーネント ID に割り当てられているバージョン マネージャ。
name	プロセス名。
helper_name	ヘルパー プロセスの名前。
helper_path	ヘルパー プロセスの実行可能パス。
path	プロセスの実行可能パス。

node_name	プロセスが属するノードのノード名に割り当てられているシステム マネージャ。
is_respawn	プロセスが再生成できることを指定するフラグ。
is_mandatory	プロセスが実行され続ける必要があることを指定するフラグ。
is_hold	APIによって呼び出されるまでプロセスが再生成されることを指定するフラグ。
dump_option	コア ダンプのオプション。
max_dump_count	許可されるコア ダンプの最大数。
respawn_count	プロセスの再起動の合計回数。
fail_count	プロセスの再起動試行の回数。プロセスが正常に再起動されると、このカウントはゼロにリセットされます。
dump_count	実行されたコア ダンプの回数。
last_respawn_sec last_respawn_msec	1970 年 1 月 1 日以降の POSIX タイマー ユニットでの秒およびミリ秒の単位で、プロセスが開始された最後の時刻を表します。
inst_id	プロセス インスタンス ID。
proc_state	Sysmgr プロセスの状態。error、forced_stop、hold、init、ready_to_run、run、run_rnode、stop、waitEOltimer、wait_rnode、wait_spawntimer、wait_tpl の 1 つです。
level	プロセス実行レベル。
exit_status	プロセスの最後の終了ステータス。
exit_type	最後の終了タイプ。

_cerrno を設定

対応

sys_reqinfo_proc_all

すべての POSIX プロセスの情報を問い合わせます。この Tcl コマンド拡張は、ソフトウェア モジュール方式イメージでのみサポートされます。

構文

```
sys_reqinfo_proc_all
```

引数

なし

結果文字列

```
rec_list {{process info string 0}, {process info string 1},...}
```

各プロセスの情報文字列は、**sysreq_info_proc** Tcl コマンド拡張の結果文字列と同じです。

_cerrno を設定

対応

sys_reqinfo_routername

デバイス名を問い合わせます。

構文

```
sys_reqinfo_routername
```

引数

なし

結果文字列

```
routername %s
```

この場合、routername がデバイスの名前です。

_cerrno を設定

対応

sys_reqinfo_snmp

簡易ネットワーク管理プロトコル（SNMP）オブジェクト ID によって指定されたエンティティの値を問い合わせます。

構文

```
sys_reqinfo_snmp oid ? get_type exact|next
```

引数

oid	（必須）ドット付き表記での SNMP OID（たとえば、1.3.6.1.2.1.2.1.0）。
get_type	（必須）指定された OID に適用する必要がある SNMP 取得操作のタイプ。get_type が「exact」の場合、指定された OID の値が取得されます。get_type が「next」の場合、指定された OID の辞書順での後続値が取得されます。

結果文字列

```
oid {%s} value {%s}
```

oid	SNMP OID。
value	割り当てられた SNMP データ エレメントの値文字列。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 37)    FH_ENOSNMPDATA  (can't retrieve data from SNMP)
```

このエラーは、SNMP オブジェクト タイプのデータがなかったことを意味します。

```
(_cerr_sub_err = 51)    FH_ESTATSTYP  (invalid statistics data type)
```

このエラーは、SNMP 統計データ タイプが無効であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

sys_reqinfo_syslog_freq

すべての Syslog イベントの頻度情報を問い合わせます。

構文

```
sys_reqinfo_syslog_freq
```

引数

なし

結果文字列

```
rec_list {{event frequency string 0}, {log freq str 1}, ...}
```

各イベントの頻度の文字列は、次のとおりです。

```
time_sec %ld time_msec %ld match_count %u raise_count %u occurs %u
period_sec %ld period_msec %ld pattern {%s}
```

time_sec time_msec	1970 年 1 月 1 日以降の POSIX タイマー ユニットでの秒およびミリ秒の単位で、最後のイベントが発生した時刻を表します。
match_count	イベントの登録以降、この Syslog イベント指定によって指定されたパターンが、Syslog メッセージによって照会される回数。
raise_count	この Syslog イベントが発生した回数。
occurs	イベントを発生させるために必要な発生回数。指定されない場合、イベントは 1 回目から発生します。
period_sec period_msec	イベントを発生させるには、発生回数が POSIX タイマーユニットのこの数以内である必要があります。この引数が指定されない場合、時間のチェックは適用されません。
pattern	Syslog メッセージのパターン マッチの実行に使用される正規表現。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 45)    FH_ESEQNUM  (sequence or workset number out of sync)
```

このエラーは、イベント ディテクタ シーケンスまたは作業セット番号が無効であったことを意味します。

```
(_cerr_sub_err = 46)    FH_EREGEMPTY  (registration list is empty)
```

このエラーは、イベント ディテクタ登録リストが空であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

sys_reqinfo_syslog_history

指定された Syslog メッセージの履歴を問い合わせます。

構文

```
sys_reqinfo_syslog_history
```

引数

なし

結果文字列

```
rec_list {{log hist string 0}, {log hist str 1}, ...}
```

各記録の履歴の文字列は、次のとおりです。

```
time_sec %ld time_msec %ld msg {%s}
```

time_sec time_msec	1970 年 1 月 1 日以降の秒およびミリ秒の単位で、メッセージが記録された時刻を表します。
msg	Syslog メッセージ。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 44)    FH_EHISTEMPTY  (history list is empty)
```

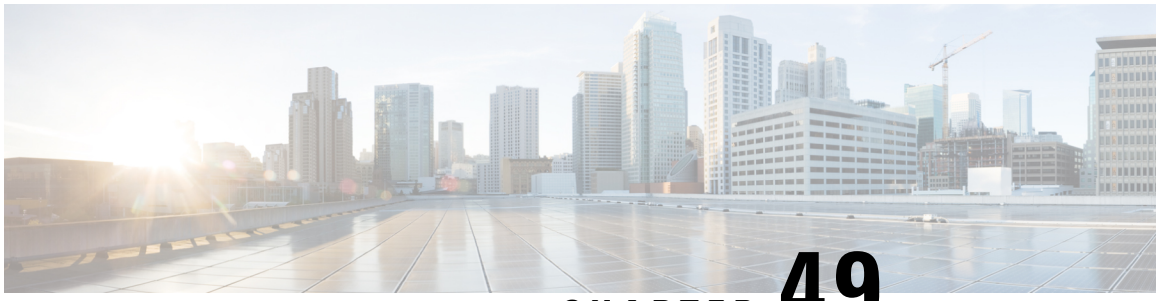
このエラーは、履歴のリストが空であったことを意味します。

```
(_cerr_sub_err = 45)    FH_ESEQNUM  (sequence or workset number out of sync)
```

このエラーは、イベント ディテクタ シーケンスまたは作業セット番号が無効であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。



CHAPTER 49

EEM ユーティリティの Tcl コマンド拡張

次の表記法が、Tcl コマンド拡張ページで説明されている構文に使用されます。

- 任意の引数は、たとえば次の例のように、角カッコ内に示されます。

[type ?]

- 疑問符（?）は入力する変数を表します。
- 引数間の選択肢は、たとえば次の例のように、パイプ文字で示されます。

priority low|normal|high

Note すべての EEM Tcl コマンド拡張について、エラーがあった場合、戻される Tcl 結果文字列には、エラー情報が含まれます。

Note 数値範囲が指定されていない引数は、–2147483648 から 2147483647 までの整数から取得されます。

appl_read

Embedded Event Manager（EEM）アプリケーションの揮発性データを読み取ります。この Tcl コマンド拡張では、EEM アプリケーションの揮発性データの読み取りがサポートされます。EEM アプリケーションの揮発性データは、API をパブリッシュする EEM アプリケーションが使用される Cisco ソフトウェア プロセスによってパブリッシュすることができます。EEM アプリケーションの揮発性データは、EEM ポリシーによってパブリッシュできません。

Note 現在、アプリケーション揮発性データをパブリッシュする Cisco ソフトウェアはありません。

構文

```
appl_read name ? length ?
```

引数

name	（必須）アプリケーションによってパブリッシュされる文字列データの名前。
length	（必須）読み取る文字列データの長さ。1 ～ 4294967295 の範囲の整数である必要があります。

結果文字列

```
data %s
```

data は、読み取られる、アプリケーションによってパブリッシュされた文字列データです。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

このエラーは、アプリケーション イベント ディテクタ情報キーまたはその他の ID が見つからなかったことを意味します。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

appl_reqinfo

Embedded Event Manager（EEM）から、前に保存された情報が取得されます。この Tcl コマンド拡張によって、一意のキーで前に保存された EEM からの情報の取得がサポートされます。これは、情報を取得するために指定する必要があります。情報の取得によって、その情報が EEM から削除されることに、注意してください。再度取得できるようにするには、再保存する必要があります。

構文

```
appl_reqinfo key ?
```

引数

キー	（必須）データの文字列キー。

結果文字列

```
data %s
```

data は、取得されるアプリケーション文字列データです。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

このエラーは、アプリケーション イベント ディテクタ情報キーまたはその他の ID が見つからなかったことを意味します。

appl_setinfo

Embedded Event Manager（EEM）に情報を保存します。この Tcl コマンド拡張によって、同じポリシーまたは別のポリシーによって、後で取得できる Embedded Event Manager への情報の保存がサポートされます。一意のキーを指定する必要があります。このキーを使用すると、情報を後で取得することができます。

構文

```
appl_setinfo key ? data ?
```

引数

キー	（必須）データの文字列キー。
data	（必須）保存するアプリケーション文字列データ。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  8)    FH_EDUPLICATEKEY  (duplicate appl info key)
```

このエラーは、アプリケーション イベント ディテクタ情報キーまたはその他の ID が重複していたことを意味します。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 34)    FH_EMAXLEN  (maximum length exceeded)
```

このエラーは、オブジェクト長またはオブジェクト数が、最大値を超えたことを意味します。

```
(_cerr_sub_err = 43)    FH_EBADLENGTH  (bad API length)
```

このエラーは、API メッセージ長が無効であったことを意味します。

counter_modify

カウンタの値を変更します。

構文

```
counter_modify event_id ? val ? op nop|set|inc|dec
```

引数

event_id	（必須）**register_counter** Tcl コマンド拡張によって返されるカウンタイベント ID。0 ～ 4294967295 の範囲の整数である必要があります。
val	（必須） **Note** op nop 引数値の組み合わせが指定されている以外は必須です。 • op が設定されている場合、この引数は、設定されるカウンタ値を表します。 • op が inc の場合、この引数は、カウンタを増やすために使用される値です。 • op が dec の場合、この引数は、カウンタを減らすために使用される値です。
op	（必須） • nop：現在のカウンタの値を取得します。 • set：カウンタの値を指定値に設定します。 • inc：カウンタの値を指定値分増やします。 • dec：カウンタの値を指定値分減らします。

結果文字列

```
val_remain %d
```

val_remain は、カウンタの現在の値です。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 30)    FH_ECTBADOPER  (bad counter threshold operator)
```

このエラーは、カウンタ イベント ディテクタの設定演算子または変更演算子が、無効であったことを意味します。

description

記録されたポリシーの簡単な説明を記述します。

構文

```
description ?
```

引数

line	（任意）1 文字から 240 文字で構成されるポリシーの簡単な説明。

結果文字列

なし

_cerrno を設定

対応

使用例

説明文は、ポリシーの作成者によって入力されます。Tcl のイベント登録文の前または後に表示できます。ポリシーには、1 つの説明のみ使用できます。

Note 1 つのポリシーに複数の説明文を登録した場合、障害が発生します。

次に、**event_register_syslog** ポリシーに簡単な説明が指定される例を示します。

```
::cisco::eem::description "This Tcl command looks for the word count in syslog messages."
::cisco::eem::event_register_syslog tag 1 ...
::cisco::eem::event_register_snmp_object tag 2 ...
::cisco::eem::trigger {
       ::cisco::eem::correlate event 1 and event 2
       ::cisco::eem::attribute tag 1 occurs 1
       ::cisco::eem::attribute tag 2 occurs 1
}
```

fts_get_stamp

最後にソフトウェアがブートされて以来の経過時間を返します。この Tcl コマンド拡張を使用すると、配列「nsec nnnn」に、ブート以降のナノ秒数が返されます。ここで、nnnn はナノ秒数です。

構文

```
fts_get_stamp
```

引数

なし

結果文字列

```
nsec %d
```

nsec は、ブート以降のナノ秒数です。

_cerrno を設定

非対応

register_counter

カウンタを登録し、カウンタ イベント ID を返します。この Tcl コマンド拡張は、カウンタのパブリッシャによって使用され、イベント ID を使用してカウンタを操作する前に、この登録が実行されます。

構文

```
register_counter name ?
```

引数

name	（必須）操作されるカウンタの名前。

結果文字列

```
event_id %d
event_spec_id %d
```

event_id は、指定されたカウンタのカウンタイベント ID です。**unregister_counter** または **counter_modify** Tcl コマンド拡張によって、カウンタの操作に使用されます。event_spec_id 引数は、指定されたカウンタのイベント指定 ID です。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  4)    FH_EINITONCE  (Init() is not yet done, or done twice.)
```

このエラーは、EEM イベント ディテクタがその初期化を完了する前に、特定のイベントを登録する要求が行われたことを意味します。

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

このエラーは、内部イベント指定で指定されたイベント タイプが無効であったことを意味します。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 10)    FH_ECORRUPT  (internal EEM API context is corrupt)
```

このエラーは、内部 EEM API コンテキスト構造が破損したことを意味します。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 16)    FH_EBADFMPPTR  (bad ptr to fh_p data structure)
```

このエラーは、各 EEM API コールで使用されるコンテキスト ポインタが不正確であったことを意味します。

```
(_cerr_sub_err = 17)    FH_EBADADDRESS  (bad API control block address)
```

このエラーは、EEM API に渡された制御ブロック アドレスが不正確であったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 25)    FH_ESUBSEXCEED  (number of subscribers exceeded)
```

このエラーは、タイマーまたはカウンタのサブスクライバの数が、最大値を超えたことを意味します。

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

これは、サブスクライバの索引が無効であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

このエラーは、この要求を処理する EEM イベント ディテクタは使用できないことを意味します。

register_timer

タイマーを登録し、タイマー イベント ID を返します。この Tcl コマンド拡張は、カウンタのパブリッシャによって使用され、パブリッシャまたはサブスクライバとしての登録に、**event_register_timer** コマンド拡張が使用されなかった場合に、イベント ID を使用してタイマーを操作する前に、この登録が実行されます。

構文

```
register_timer watchdog|countdown|absolute|cron name ?
```

引数

name	（必須）操作されるタイマーの名前。

結果文字列

```
event_id %u
```

event_id は指定したタイマーのタイマーイベント ID です（これを使用して、**timer_arm** または **timer_cancel** コマンド拡張によってタイマーを操作するために使用されます）。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  4)    FH_EINITONCE  (Init() is not yet done, or done twice.)
```

このエラーは、EEM イベント ディテクタがその初期化を完了する前に、特定のイベントを登録する要求が行われたことを意味します。

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

このエラーは、内部イベント指定で指定されたイベント タイプが無効であったことを意味します。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 10)    FH_ECORRUPT  (internal EEM API context is corrupt)
```

このエラーは、内部 EEM API コンテキスト構造が破損したことを意味します。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 16)    FH_EBADFMPPTR  (bad ptr to fh_p data structure)
```

このエラーは、各 EEM API コールで使用されるコンテキスト ポインタが不正確であったことを意味します。

```
(_cerr_sub_err = 17)    FH_EBADADDRESS  (bad API control block address)
```

このエラーは、EEM API に渡された制御ブロック アドレスが不正確であったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 25)    FH_ESUBSEXCEED  (number of subscribers exceeded)
```

このエラーは、タイマーまたはカウンタのサブスクライバの数が、最大値を超えたことを意味します。

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

これは、サブスクライバの索引が無効であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

このエラーは、この要求を処理する EEM イベント ディテクタは使用できないことを意味します。

timer_arm

タイマーを搭載します。タイプは、CRON、ウォッチドッグ、カウントダウン、または絶対の場合があります。

構文

```
timer_arm event_id ? cron_entry ?|time ?
```

引数

event_id	（必須）**register_timer** Tcl コマンド拡張によって返されるタイマー イベント ID。0 ～ 4294967295 の範囲の整数である必要があります。
cron_entry	（必須）タイマー タイプが CRON の場合に存在する必要があります。他のタイプのタイマーの場合には、存在させることはできません。CRON タイマー指定によって、CRON テーブル エントリの形式が使用されます。
time	（必須）タイマー タイプが CRON ではない場合に存在する必要があります。タイマー タイプが CRON の場合には、存在できません。ウォッチドッグ タイマーおよびカウントダウン タイマーでは、タイマーの期限が切れるまでの秒数およびミリ秒数です。絶対タイマーでは、期限切れ時刻のカレンダー時間です（SSSSSSSSSS[.MMM] 形式で指定します。SSSSSSSSSS は、0 ～ 4294967295 の秒数を表す整数で、MMM は 0 ～ 999 のミリ秒数を表す整数である必要があります）。期限の絶対日付は、1970 年 1 月 1 日以降の秒およびミリ秒の単位での数です。指定された日付がすでに過ぎた場合、タイマーの期限はただちに切れます。

結果文字列

```
sec_remain %ld msec_remain %ld
```

sec_remain および msec_remain は、タイマーの次の期限切れまでの残り時間です。

Note タイマー タイプが CRON の場合、sec_remain 引数および msec_remain 引数には 0 が返されます。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

このエラーは、内部イベント指定で指定されたイベント タイプが無効であったことを意味します。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベント ディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

このエラーは、イベントが登録されたときか、またはイベント ディテクタの内部イベント構造が破損したときに、イベント ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 27)    FH_ETMDELAYZR  (zero delay time)
```

このエラーは、タイマーの搭載に指定された時間がゼロであったことを意味します。

```
(_cerr_sub_err = 42)    FH_ENOTREGISTERED  (request for event spec that is unregistered)
```

このエラーは、イベント検出が登録できなかったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

このエラーは、この要求を処理する EEM イベント ディテクタは使用できないことを意味します。

timer_cancel

タイマーを取り消します。

構文

```
timer_cancel event_id ?
```

引数

event_id	（必須）**register_timer** Tcl コマンド拡張によって返されるタイマー イベント ID。0 ～ 4294967295 の範囲の整数である必要があります。

結果文字列

```
sec_remain %ld msec_remain %ld
```

sec_remain および msec_remain は、タイマーの次の期限切れまでの残り時間です。

Note タイマー タイプが CRON の場合、sec_remain および msec_remain には 0 が返されます。

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  6)    FH_EBADEVENTTYPE  (unknown EEM event type)
```

このエラーは、内部イベント指定で指定されたイベント タイプが無効であったことを意味します。

```
(_cerr_sub_err =  7)    FH_ENOSUCHKEY  (could not find key)
```

このエラーは、アプリケーション イベント ディテクタ情報キーまたはその他の ID が見つからなかったことを意味します。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベント ディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 12)    FH_ENOSUCHEID  (unknown event ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベントディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

このエラーは、イベント ディテクタが使用できなかったことを意味します。

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

このエラーは、この要求を処理する EEM イベント ディテクタは使用できないことを意味します。

unregister_counter

カウンタの登録を解除します。この Tcl コマンド拡張は、以前に **register_counter** Tcl コマンド拡張に登録されていたカウンタの登録を解除するために、カウンタパブリッシャによって使用されます。

構文

```
unregister_counter event_id ? event_spec_id ?
```

引数

event_id	（必須）**register_counter** コマンド拡張によって返されるカウンタイベント ID。0 ～ 4294967295 の範囲の整数である必要があります。
event_spec_id	（必須）**register_counter** コマンド拡張によって返された、指定されたカウンタのカウンタイベント指定 ID。0 ～ 4294967295 の範囲の整数である必要があります。

結果文字列

なし

_cerrno を設定

対応

```
(_cerr_sub_err =  2)    FH_ESYSERR  (generic/unknown error from OS/system)
```

このエラーは、オペレーティング システムによってレポートされたエラーを意味します。エラーとともにレポートされる POSIX errno 値を使用して、オペレーティング システム エラーの原因を調べます。

```
(_cerr_sub_err =  9)    FH_EMEMORY  (insufficient memory for request)
```

このエラーは、メモリの内部 EEM 要求に障害が発生したことを意味します。

```
(_cerr_sub_err = 11)    FH_ENOSUCHESID  (unknown event specification ID)
```

このエラーは、イベントが登録されたときか、またはイベントディテクタの内部イベント構造が破損したときに、イベント指定 ID を照会できなかったことを意味します。

```
(_cerr_sub_err = 22)    FH_ENULLPTR  (event detector internal error - ptr is null)
```

このエラーは、内部 EEM イベント ディテクタ ポインタに値が含まれている必要があったときに、ヌルであったことを意味します。

```
(_cerr_sub_err = 26)    FH_ESUBSIDXINV  (invalid subscriber index)
```

これは、サブスクライバの索引が無効であったことを意味します。

```
(_cerr_sub_err = 54)    FH_EFDUNAVAIL  (connection to event detector unavailable)
```

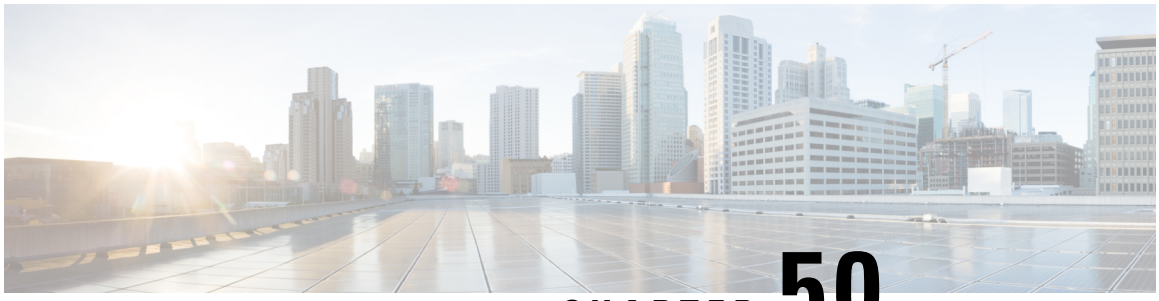
このエラーは、イベント ディテクタが使用できなかったことを意味します。

```
(_cerr_sub_err = 56)    FH_EFDCONNERR  (event detector connection error)
```

このエラーは、この要求を処理する EEM イベント ディテクタは使用できないことを意味します。

第 IX 部

Embedded Syslog Manager



CHAPTER 50

Embedded Syslog Manager（ESM）

Embedded Syslog Manager（ESM）機能は、システム メッセージ ロガーによって伝送される前にシステム ロギング メッセージをフィルタリング、拡大、相互関連付け、ルーティング、カスタマイズできるようにするプログラマブル フレームワークを提供します。

Embedded Syslog Manager の制約事項

Embedded Syslog Manager（ESM）フィルタは Tool Command Language（Tcl）で記述されているため、Embedded Syslog Manager（ESM）は Tcl 8.3.4 Cisco IOS XE サブシステムに依存します。ESM は、Tcl バージョン 8.3.4 以降をサポートするイメージでのみ使用可能です。Tcl 8.3.4 サポートが追加されるかどうかはリリースによって異なります。

ESM フィルタは Tcl で書かれています。

ESM フィルタリングは、SNMP「履歴」ロギングには適用できません。したがって、**logging history** および **snmp-server enable traps syslog** コマンドを使ってロギングされるメッセージには ESM フィルタリングが適用されません。

Embedded Syslog Manager について

システム メッセージ ロギング

Embedded Syslog Manager を導入すると、システム メッセージを標準メッセージ、XML 形式のメッセージ、または ESM でフィルタリングされたメッセージとして、別個にロギングできる

ようになります。これらの出力は、従来のあらゆるsyslogターゲットに送信できます。たとえば、コンソール接続への標準ロギング、XML 形式のメッセージのバッファへのロギング、およびESMでフィルタリングされたメッセージのモニタへのロギングをイネーブルにできます。同様に、各タイプの出力は異なるリモート ホストに送信できます。別個のロギング プロセスの利点は、たとえばESM フィルタ モジュールに問題がある場合に標準のロギングが影響を受けないことです。

システム ロギング メッセージの形式

システム ロギング メッセージは、次の形式で表示されます。

%<ファシリティ>-<シビラティ（重大度）>-<ニーモニック>: <メッセージ テキスト>

以下に、システム ロギング メッセージの例を示します。

```
%LINK-5-CHANGED: Interface Serial3/3, changed state to administratively down
```

通常は、これらのメッセージの前にエラー シーケンス番号やタイムスタンプなどの追加のテキストが存在します。

<sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

エラー シーケンス番号とタイムスタンプの後に続くシステム ロギング メッセージの例を以下に示します。

```
000013: Mar 18 14:52:10.039:%LINK-5-CHANGED: Interface Serial3/3, changed state
to administratively down
```

Note システムロギングメッセージで使用されるタイムスタンプの形式は、**service timestamps** グローバル コンフィギュレーション モード コマンドによって決まります。**service sequence-numbers** グローバル コンフィギュレーション コマンドは、先頭のシーケンス番号をイネーブルまたはディセーブルにします。時刻の前のアスタリスク（*）は、システム クロックが信頼できる時刻源と同期していないため、時刻が不正確である可能性があることを示しています。

Embedded Syslog Manager の利点

Cisco ソフトウェアの組み込み機能である Embedded Syslog Manager（ESM）を使用すると、送信元でのシステム メッセージ ロギングの完全な制御が可能になります。ESM に備わっているプログラマティック インターフェイスを使用すると、システム ロギングに関連する特定のニーズを満たすカスタム フィルタを作成できます。この機能の利点は次のとおりです。

- カスタマイズ：システム ロギング メッセージを扱う完全にカスタマイズ可能な処理。相互に連携する複数の syslog コレクタをサポートします。

- 主要なメッセージのシビラティ（重大度）のエスカレーション：システム定義のシビラティ（重大度）レベルを使用する代わりに、syslog メッセージに関する独自のシビラティ（重大度）レベルを設定できます。
- 特定のメッセージに対象を絞る：ファシリティのタイプやシビラティ（重大度）のタイプに基づいて、特定のメッセージまたはメッセージ タイプをさまざまな syslog コレクタにルーティングできます。
- SMTP ベースの E メール アラート：TCP ベースの syslog コレクタまたはシンプル メール転送プロトコル（SMTP）サーバなどの外部サーバに TCP を使用して通知する機能。
- メッセージの制限：デバイスレベルのイベントを関連付けることにより、syslog の「メッセージ ストーム」を制限し、管理することができます。

ESM は UDP ベースの syslog メカニズムに代わるものではなく、現在のシステム ロギング プロセスと並行して動作可能なオプションのサブシステムです。たとえば、元の syslog メッセージ ストリームをサーバ A によって収集し続けるのと同時に、フィルタリング、相互関連、その他の方法でカスタマイズされた ESM ロギング ストリームをサーバ B に送信することができます。元の syslog ストリームまたは ESM ストリームのいずれかを受信するよう、syslog メッセージの現在のすべてのターゲット（コンソール、モニタ、バッファ、および syslog ホストリスト）を設定することができます。ESM ストリームは、それに応じて、ユーザ定義のストリームにさらに分割し、コレクタにルーティングできます。

syslog フィルタ モジュール

Embedded Syslog Manager（ESM）は syslog フィルタ モジュールを使用してシステム ロギング メッセージを処理します。syslog フィルタ モジュールは、ローカル システム メモリまたはリモート ファイル サーバに保存される Tool Command Language（Tcl）で書かれたスクリプトです。ESM は、独自のスクリプトを書いて参照できるため、カスタマイズ可能です。

syslog フィルタ モジュールは、プレーンテキスト ファイルまたはコンパイル済みファイルとして書いて、保存できます。Tcl スクリプトの事前コンパイルは、TclPro などのツールを使用して実行できます。コンパイル済みスクリプトは編集できないため、セキュリティおよび管理された一貫性の尺度になります。

Note Tcl スクリプト モジュールには実行可能コマンドが含まれているため、コンフィギュレーション ファイルを管理するのと同じ方法で、これらのファイルのセキュリティを管理する必要があります。

Embedded Syslog Manager の使用方法

ESM syslog フィルタ モジュールの書き込み

Embedded Syslog Manager（ESM）設定で syslog フィルタ モジュールを参照する前に、システム ロギング メッセージへの適用対象となるモジュールを書き込むか、取得する必要があります。syslog フィルタ モジュールは、ローカル システム メモリまたはリモート ファイル サーバに保存できます。syslogフィルタモジュールを書き込む前に、次の概念を理解しておく必要があります。

ESM フィルタ プロセス

ESM がイネーブルの場合、すべてのシステム ロギング メッセージは、参照された syslog フィルタ モジュールを通して処理されます。syslog フィルタ モジュールは、フィルタ チェーンで順番に処理されます。フィルタチェーン内の syslog フィルタ モジュールの位置は、**logging filter** グローバル コンフィギュレーション モード コマンドで適用された位置タグによって決定されます。位置が指定されていない場合、モジュールは、設定に追加された順番で処理されます。

各フィルタ モジュールの出力は、チェーン内の次のフィルタ モジュールの入力として使用されます。したがって、元の syslog メッセージ（::orig_msg）を含む Tcl グローバル変数は、チェーン内の次のフィルタを呼び出す前に、各フィルタの戻り値に設定されます。したがって、フィルタが NULL を戻した場合、メッセージは ESM ストリームに送信されません。すべてのフィルタがメッセージを処理した後、メッセージは、ロガーによる配信のためのキューに入れられます。

コンソール、バッファ、モニタ、および syslog ホストは、特定のメッセージ ストリーム（通常、XML、または ESM）を受信するように設定できます。syslog ホストは、ユーザ定義の番号が付いたストリームを受信するように、さらに限定できます。各ターゲットは各メッセージを検査し、そのストリーム タグに基づいてメッセージを受け入れるか、または拒否します。ESM フィルタは、Tcl グローバル変数「::stream」を変更することによってメッセージのストリーム タグを変更し、宛先ストリームを変更できます。

syslog フィルタ モジュールの入力

Embedded Syslog Manager（ESM）が有効になっている場合、システム ロギング メッセージがロギング プロセスに送信されます。システム ロギング メッセージの中、およびフォーマットされた syslog メッセージ全体に含まれる各データ エレメントが、Tcl グローバル変数として記録されます。syslog メッセージのデータ エレメントの形式は、次のとおりです。

<sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

メッセージ テキストには、しばしばメッセージ引数が含まれます。

また、メッセージが syslog ホストで受信されるときに「syslog カウント」番号が追加されます。

<syslog-count>: <sequence-number>: <time stamp>:%<facility>-<severity>-<mnemonic>: <message-text>

次の例は、シーケンスの最初に含まれる syslog カウント番号を示しています。

以下の表に、syslog フィルタ モジュールで使用される Tcl スクリプト入力変数を示します。フィルタで扱う必要のある syslog メッセージ データは、Tcl グローバル名前空間変数として渡されます。したがって、スクリプト モジュール内で変数の前に二重のコロンを付ける必要があります。

標準的な ESM フィルタ処理

システム ロギング メッセージが生成されるたびに、syslog フィルタ モジュールがシリーズで呼び出されます。このシリーズは ::module_position 変数によって決定されますが、これは一般的にシステム設定内でモジュールが参照される順番（モジュールが設定される順番）です。

あるフィルタ モジュールの出力が、次のフィルタ モジュールへの入力になります。フィルタへの入力は Tcl グローバル名前空間変数であるため、それぞれのフィルタは、フィルタの目的に応じてこれらの変数のいずれかまたはすべてを変更できます。

後続の複数のフィルタ実行の間で Embedded Syslog Manager（ESM）フレームワークによって自動的に更新される Tcl グローバル変数は、::orig_msg 変数と ::cli_args 変数だけです。フレームワークは、自動的に ::orig_msg の値をフィルタ モジュールの戻り値に設定します。したがって、元のメッセージを変更するかフィルタリングするように設定されているフィルタでは、::orig_msg 変数の値を手動で設定しないでください。フィルタは目的の値を返すことだけが必要です。たとえば、次の 1 行の ESM フィルタ

```
return “This is my new syslog message.”
```

は、受け取ったメッセージをすべて無視して出力を「This is my new syslog message」という定数文字列に常に変更します。このモジュールがチェーン内の最後のフィルタである場合、すべての ESM ターゲットはこの文字列を最終的な syslog メッセージとして受け取ります。

1 行の ESM フィルタは、

```
return “”
```

ESM ストリームへのすべての syslog メッセージをブロックします。たとえば、

```
return $::orig_msg
```

という行は、メッセージをチェーンの次のフィルタに渡すだけです。したがって、不要なメッセージを抑制するように設定されている ESM フィルタは次のように見えます。

```
if { [my_procedure_to_check_this_message] == 1 } {
     return $::orig_msg
} else {
     return “”
}
```

その設計によって、一部のフィルタは ::orig_msg 変数をまったく使用せず、そのデータ エレメントから syslog メッセージを再構築する（::format_string、::msg_args、::timestamp などを使用）場合があります。たとえば、XML タグ付きフィルタは個々のデータ エレメントにタグを付け、

元のフォーマットされたメッセージを無視します。このようなモジュールが Tcl スクリプトの最初で ::orig_msg 変数を検査することは重要です。以前のフィルタで「メッセージを送信してはならない」と指示されている場合（つまり ::orig_msg が NULL の場合）、メッセージは処理されず、NULL を返します。

また、**exec** および **config** Tcl コマンドを使用して syslog フィルタモジュールにコマンドを追加することもできます。たとえば、syslog メッセージに送信元 IP アドレスを追加する必要があり、さらに（**logging source-interface** コマンドを使って）イーサネット 2/0 インターフェイスから送信されるよう syslog メッセージが設定される場合には、スクリプト内で **exec** Tcl コマンドを次のように使用することで、モジュールの初期化中に **show interface Ethernet 2/0** コマンドを発行できます。

```
set source_ip_string [exec show ip int E2/0 | inc Internet]
puts $source_ip_string
"  Internet address is 10.4.2.63/24"
```

バックグラウンド ESM フィルタの処理

Tcl では、後の時点で処理するためにコマンドをキューに入れることができます。それを行うには **after** Tcl コマンドを使用します。このコマンドの最も一般的な使用方法は、一定の時間間隔（「相関期間」と呼ばれる）にわたって、複数のイベントを相互に関連付ける（収集して要約する）ことです。該当する期間が満了すると、フィルタは「ウェイクアップ」し、その期間に発生したイベントを計算または要約する必要があります。また、多くの場合、イベント報告用の新しい syslog メッセージを送信する必要があります。このバックグラウンドプロセスは、特定の時間の経過後、Tcl インタープリタがキューに入れられたコマンドを実行できるようにする ESM イベント ループ プロセスによって処理されます。

syslog フィルタモジュールが相関期間を利用する必要がある場合、相関期間が満了した時点で **after** Tcl コマンドを使用して要約手順を呼び出す必要があります（「Embedded Syslog Manager の設定例」セクションにある例を参照）。バックグラウンドプロセスの実行中は通常のフィルタチェーンの処理が行われないため、出力を生成するために、これらのフィルタで 2 つの ESM Tcl 拡張機能（**errmsg** または **esm_errmsg**）のいずれかを使用する必要があります。

バックグラウンド処理中に、**after** コマンドによってキューに入れられたコマンドは、（通常の処理のように）フィルタチェーンのコンテキストでは実行されず、Tcl インタープリタによって連続して実行される自律的な手順となります。したがって、これらのバックグラウンド手順では通常の Tcl グローバル名前空間変数を扱うべきではありません（ただし **esm_errmsg** の使用時に次のフィルタ用にグローバル名前空間変数を設定する場合を除く）。代わりに、独自の名前空間に保存される変数を扱う必要があります。これらの変数が手順の定義の外側で宣言される場合、すべてのコールで永続的です。

errmsg Tcl コマンドの目的は、新しいメッセージを作成して配信用に送り出し、その際に他の syslog フィルタモジュールをすべて回避することです。**errmsg** コマンドの構文は次のとおりです。

```
errmsg <severity> <stream> <message_string>
```

esm_errmsg Tcl コマンドの目的は、新しいメッセージを作成し、フィルタチェーン内のそれ以降にある syslog フィルタモジュールでメッセージを処理して、配信用に送り出すことです。**esm_errmsg** コマンドの構文は次のとおりです。

```
esm_errmsg <module_position>
```

errmsg() Tcl 関数と esm_errmsg() Tcl 関数の主な相違点として、**errmsg** はフィルタを無視してメッセージを配信用にキューに直接入れます。一方、**esm_errmsg** はフィルタチェーンの後続部分に syslog メッセージを送ります。

次の例では、新しい syslog メッセージが作成され、アラートシビラティ（重大度）1 のタグを付けられて、設定された ESM ロギング ターゲット（ストリーム 2）に送信されます。このフィルタの目的は、30 分の相関期間中に個々の SYS-5-CONFIG メッセージを抑制し、ウィンドウ終了時に要約メッセージを送信することです。

```
errmsg 1 2 "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
```

（それ以降にある残りのフィルタを呼び出す）**esm_errmsg** を使用するために、このバックグラウンドプロセスは **esm_errmsg** を呼び出す前に、必要な Tool Command Language（Tcl）グローバル名前空間変数を設定する必要があります。::module_position を渡すと、開始するフィルタが ESM フレームワークに通知されます。したがって、**esm_errmsg** コマンドを使用するフィルタは、バックグラウンド処理用に、（通常の処理中にグローバル名前空間変数で渡される）::module_position を独自の名前空間変数に保存する必要があります。次に例を示します。

```
proc ::my_filter_namespace::my_summary_procedure{}
{
 set ::orig_msg "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
 set ::timestamp "*Jan 24 09:34:02.539"
 set ::severity 1
 set ::stream 2
 set ::traceback ""
 set ::pid ""
set ::process ""
set ::format_string "There have been %d configuration changes to the router
between %s and %s"
set ::msg_args {12 "Jan 24 09:04:01.539" "Jan 24 09:34:01.324"}
esm_errmsg $::my_filter_namespace::my_module_position
}
```

esm_errmsg コマンド用にすべてのグローバル名前空間変数を設定することの利点は、フィルタがモジュール式であり、ESM フレームワークでのフィルタの使用順序が重要でないことです。たとえば、ESM を宛先とするすべてのメッセージの後ろにメッセージ発信者のホスト名を付ける必要がある場合、次のように 1 行の「hostname」フィルタを作成し、フィルタ チェーンの最後に配置できます。

```
return "$::orig_msg -- $::hostname"
```

この例では、バックグラウンド処理中にいずれかのフィルタが新しいメッセージを生成し、**errmsg** ではなく **esm_errmsg** を使用する場合、これらのメッセージの後ろに明確にホスト名が付けられます。

次の作業

syslogフィルタモジュールを作成した後、デバイスからアクセス可能な場所にファイルを保存する必要があります。ファイルは、ローカル システム メモリにコピーするか、ネットワーク ファイル サーバに保存できます。

Embedded Syslog Manager の設定

Embedded Syslog Manager（ESM）を設定するには、生成された syslog メッセージに適用する 1 つまたは複数のフィルタを指定し、syslog メッセージのターゲットを指定します。

Before you begin

デバイスで 1 つまたは複数の syslog フィルタ モジュールを使用できなければなりません。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging filter** *filter-url* [*position*] [**args** *filter-arguments*]
4. システム ロギング出力に適用する必要がある各syslogフィルタモジュールに対して、ステップ 3 を繰り返します。
5. 次のいずれか 1 つを入力します。
   - **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*]
   - または
   - **logging host** {*ip-address* | *hostname*} **filtered** [**stream** *stream-id*]
6. 希望する各システム ロギング宛先に対して、ステップ 5 を繰り返します。
7. **logging source-interface** *type number*
8. **logging origin-id** {**hostname** | **ip** | **ipv6** |**string** *user-defined-id*}
9. **end**
10. **show logging**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
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	**logging filter** *filter-url* [*position*] [**args** *filter-arguments*] **Example:** `Device(config)# logging filter slot0:/escalate.tcl 1 args CONFIG_I 1`	生成されたシステム ロギング メッセージに適用する 1 つまたは複数の syslog フィルタ モジュールを指定します。 • 使用する必要がある各 syslog フィルタ モジュールに対して、このコマンドを繰り返します。 • *filter-url* 引数は、syslog フィルタ モジュール（スクリプト）を示す Cisco IOS ファイル システムの場所です。場所としてローカルメモリまたはリモートサーバーが可能です（**tftp:**、**ftp:** または **rcp:** を使用）。 • オプションの *position* 引数は、syslog フィルタ モジュールを実行する順序を指定します。この引数を省略した場合、指定されたモジュールは、チェーンの最後のモジュールとして配置されます。 • 再び **logging filter** コマンドを入力して異なる位置（position）を指定することにより、フィルタの順序をすばやく再設定できます。 • オプションの **args** *filter-arguments* 構文を追加し、引数を指定されたフィルタに渡すことができます。複数の引数を指定できます。引数の数とタイプは、syslog フィルタ モジュールで定義する必要があります。たとえば、特定の E メールアドレスを引数として受け入れるように syslog フィルタモジュールが設計されている場合、**args user@host.com** 構文を使用して E メールアドレスを渡すことができます。複数の引数は、通常、スペースで区切ります。 • 実行するモジュールのリストからモジュールを削除するには、このコマンドの **no** 形式を使用します。
ステップ 4	システム ロギング出力に適用する必要がある各 syslog フィルタ モジュールに対して、ステップ 3 を繰り返します。	--

	Command or Action	Purpose

ステップ 5	次のいずれか 1 つを入力します。
	• **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*]
	• または
	• **logging host** {*ip-address* | *hostname*} **filtered** [**stream** *stream-id*]
	Example:
	`Device(config)# logging console filtered informational`
	Example:
	`Device(config)# logging host 209.165.200.225 filtered stream 20`
		ESM のフィルタリングされた syslog 出力のターゲットを指定します。
		• ESM フィルタリングされた syslog メッセージの送信先として、コンソール、モニタ（TTY および Telnet 接続）、システム バッファ、またはリモート ホストが可能です。
		• オプションの *level* 引数は、メッセージの送信を、指定された値またはそれより低い値のメッセージに限定します。たとえば、レベル **1** が指定されている場合、レベル 1（アラート）またはレベル 0（緊急事態）のメッセージだけが、指定されたターゲットに送信されます。レベルはキーワードまたは数字として指定できます。
		• コンソール、モニタ接続、またはシステムバッファにロギングする場合、*level* 引数によって指定されたシビラティ（重大度）のしきい値が ESM フィルタリングよりも優先されます。ESM ターゲットに送信されるべきメッセージが ESM フィルタから返された場合でも、シビラティ（重大度）が設定済みしきい値を満たさない（つまりレベル値より高い）場合には、メッセージが送信されません。
		• リモートホストにロギングする場合、ストリーム タグを使用すると、メッセージ タイプに基づいて宛先を指定できます。**stream** *stream-id* 構文を使用すると、指定されたストリーム値を持つメッセージだけを特定のホストに送信するよう ESM を設定できます。
		• ストリーム値は、設定された syslog フィルタ モジュールによって、メッセージに適用されます。たとえば、シビラティ（重大度） 5 のすべてのメッセージでストリーム タグ「20」を設定できます。さらに、ストリーム タグ「20」を持つすべてのメッセージが 209.165.200.225 のホストに送信されるように指定できます。

ステップ 6	希望する各システム ロギング宛先に対して、ステップ 5 を繰り返します。	• ロギング ホスト コマンドを複数回発行することによって、異なるシステム ロギング ストリームに異なるターゲットを指定できます。

	Command or Action	Purpose
		• コンソール、モニタ接続、またはシステムバッファに向けて、異なるシビラティ（重大度）のメッセージを送信するように設定できます。たとえば、ネットワーク オペレーションセンター（NOC）の画面に重要なメッセージだけを表示できます（モニタまたはコンソール接続を使用）。
ステップ 7	**logging source-interface** *type number* **Example:** `Device(config)# logging source-interface Gigabitethernet 0/0`	（任意）リモート syslog ホストに送信する syslog メッセージの送信元インターフェイスを指定します。 • 通常では、リモート ホストに送信される syslog メッセージは、メッセージ生成の時点で使用できるあらゆるインターフェイスを使用します。このコマンドを使用すると、デバイスは、指定されたインターフェイスだけから syslog メッセージをリモート ホストに送信します。
ステップ 8	**logging origin-id** {**hostname** | **ip** | **ipv6** |**string** *user-defined-id*} **Example:** `Device(config)# logging origin-id string "Domain 2, Router 5"`	（任意）リモート ホストに送信される syslog メッセージに発信元 ID を追加できます。 • 発信元 ID は、リモート ホストに送信されるすべての syslog メッセージの最初に追加されます。ID はホスト名、IP アドレス、または指定する任意のテキストです。 • 発信元 ID は、syslog 出力を複数のデバイスから 1 つの syslog ホストに送信する場合、システム ロギング メッセージの送信元を識別するのに役立ちます。
ステップ 9	**end** **Example:** `Device(config)# end`	現在のコンフィギュレーション セッションを終了し、CLI を特権 EXEC モードに返します。
ステップ 10	**show logging** **Example:** `Device# show logging`	（任意）ESM のフィルタリングされたロギングのステータスを含む、システム ロギングのステータスを表示します。 • バッファへのフィルタリングされたロギングがイネーブルの場合、このコマンドは、バッファに保存されたデータも表示します。

	Command or Action	Purpose
		• syslog フィルタ モジュールをこのコマンドの出力に示す順番は、フィルタ モジュールが実行される順番です。

Embedded Syslog Manager の設定例

例：Embedded Syslog Manager の設定例

次の例では、コンソール接続の Embedded Syslog Manager（ESM）フィルタ ロギングがイネーブル、モニタ接続とバッファでの標準ロギングがイネーブル、ホスト 209.165.200.225 での XML 形式のロギングがイネーブルです。

```
Device(config)# logging filter tftp://209.165.200.225/ESM/escalate.tcl
Device(config)# logging filter slot0:/email.tcl user@example.com
Device(config)# logging filter slot0:/email_guts.tcl
Device(config)# logging console filtered
Device(config)# logging monitor 4
Device(config)# logging buffered debugging
Device(config)# logging host 209.165.200.225 xml
Device(config)# end

Device# show logging
Syslog logging: enabled (0 messages dropped, 8 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering enabled)
    Console logging: level debugging, 21 messages logged, xml disabled,
                     filtering enabled
    Monitor logging: level warnings , 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 30 messages logged, xml disabled,
                     filtering disabled
    Logging Exception size (8192 bytes)
    Count and timestamp logging messages: disabled

Filter modules:
    tftp://209.165.200.225/ESM/escalate.tcl
    slot0:/email.tcl user@example.com

    Trap logging: level informational, 0 message lines logged
        Logging to 209.165.200.225, 0 message lines logged, xml enabled,
              filtering disabled

Log Buffer (8192 bytes):

*Jan 24 09:34:28.431: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:34:51.555: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:49:44.295: %SYS-5-CONFIG_I: Configured from console by console
Device#
```

例：syslog フィルタ モジュール

syslog スクリプト モジュールは Tcl スクリプトです。独自の syslog スクリプト モジュールの開発に役立つよう、次の例を示します。

Note これらのスクリプト モジュールは単に例として提供されており、シスコによるサポートの対象外です。これらのスクリプトの機能または影響に関して、明示的または黙示的に保証することはありません。

例：シビラティ（重大度）のエスカレーション

この ESM syslog フィルタ モジュール例では、1 つのニーモニック（最初の CLI 引数経由で供給される）を監視し、メッセージのシビラティ（重大度）を 2 つめの CLI 引数によって指定されるシビラティ（重大度）に拡大します。

```
#  ========================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Severity Escalation Filter                          ||||      ||||
#                                                  ..:||||||:..:||||||:..
#                                                 ------------------------
#                                                 C i s c o  S y s t e m s
#  ========================================================================
#
# Usage: Set CLI Args to "mnemonic new_severity"
#
# Namespace: global
# Check for null message
if { [string length $::orig_msg] == 0} {
   return ""
}

if { [info exists ::cli_args] } {
    set args [split $::cli_args]
    if { [ string compare -nocase [lindex $args 0] $::mnemonic ] == 0 } {
        set ::severity [lindex $args 1]
        set sev_index [ string first [lindex $args 0] $::orig_msg ]
        if {  $sev_index >= 2 } {
           incr sev_index -2
           return [string replace $::orig_msg $sev_index $sev_index \
              [lindex $args 1]]
        }
    }
}
return $::orig_msg
```

例：メッセージのカウント

この ESM syslog フィルタ モジュール例は、読みやすくするために 2 つのファイルに分割されています。最初のファイルで、ユーザは msg_to_watch アレイを設定することにより、カウント対象のメッセージおよび要約する頻度（相関期間）を設定できます。実際の手順は counting_guts.tcl ファイルに含まれています。バックグラウンド処理を行う可能性がある他の

ESM フィルタとの競合を避けるために、別個の名前空間「counting」を使用していることに注意してください。

```
#  ================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Message Counting Filter                         ||||      ||||
#                                                ..:||||||:..:||||||:..
#                                               -----------------------
#                                               C i s c o  S y s t e m s
# ================================================================================


#
# Usage:
# 1) Define the location for the counting_guts.tcl script
#
# 2) Define message categories to count and how often to dump them (sec)
#    by populating the "msg_to_watch" array below.
#    Here we define category as facility-severity-mnemonic
#    Change dump time to 0 to disable counting for that category
#
# Namespace: counting
namespace eval ::counting {
    set sub_script_url tftp://172.16.0.0/12/ESM/counting_guts.tcl
    array set msg_to_watch {
        SYS-5-CONFIG_I            5
    }
#  ====================== End User Setup ==============================
# Initialize processes for counting
    if { [info exists init] == 0 } {
       source $sub_script_url
       set position $module_position
    }
# Process the message
process_category
} ;# end namespace counting
```

メッセージ カウント サポート モジュール（**counting_guts.tcl**）

```
#  ================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Message Counting Support Module                 ||||      ||||
#                                                ..:||||||:..:||||||:..
#  (No User Modification)                       -----------------------
#                                               C i s c o  S y s t e m s
# ================================================================================



namespace eval ::counting {

# namespace variables

array set cat_msg_sev {}
array set cat_msg_traceback {}
array set cat_msg_pid {}
array set cat_msg_proc {}
```

```
array set cat_msg_ts {}
array set cat_msg_buginfseq {}
array set cat_msg_name {}
array set cat_msg_fac {}
array set cat_msg_format {}
array set cat_msg_args {}
array set cat_msg_count {}
array set cat_msg_dump_ts {}

# Should I count this message ?
   proc query_category {cat} {
        variable msg_to_watch
        if { [info exists msg_to_watch($cat)] } {
            return $msg_to_watch($cat)
        } else {
            return 0
        }
    }
   proc clear_category {index} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        unset cat_msg_sev($index) cat_msg_traceback($index) cat_msg_pid($index)\
              cat_msg_proc($index) cat_msg_ts($index) \
              cat_msg_buginfseq($index)  cat_msg_name($index) \
              cat_msg_fac($index) cat_msg_format($index) cat_msg_args($index)\
              cat_msg_count($index) cat_msg_dump_ts($index)
    }
# send out the counted messages
   proc dump_category {category} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        variable poll_interval
        set dump_timestamp [cisco_service_timestamp]
foreach index [array names cat_msg_count $category] {
      set fsm "$cat_msg_fac($index)-$cat_msg_sev($index)-$cat_msg_name($index)"
      set ::orig_msg \
        [format "%s%s: %%%s: %s %s %s %s - (%d occurence(s) between %s and %s)"\
        $cat_msg_buginfseq($index)\
          $dump_timestamp\
          $fsm \
          [uplevel 1 [linsert $cat_msg_args($index) 0 ::format
$cat_msg_format($index) ]] \
          $cat_msg_pid($index) \
          $cat_msg_proc($index) \
```

```
          $cat_msg_traceback($index) \
          $cat_msg_count($index) \
          $cat_msg_ts($index) \
          $dump_timestamp]
# Prepare for remaining ESM filters
               set ::severity $cat_msg_sev($index)
               set ::traceback $cat_msg_traceback($index)
               set ::pid $cat_msg_pid($index)
               set ::process $cat_msg_proc($index)
               set ::timestamp $cat_msg_ts($index)
               set ::buginfseq $cat_msg_buginfseq($index)
               set ::mnemonic $cat_msg_name($index)
               set ::facility $cat_msg_fac($index)
               set ::format_string $cat_msg_format($index)
               set ::msg_args [split $cat_msg_args($index)]
               esm_errmsg $counting::position
               clear_category $index
        }
  }
# See if this message already has come through since the last dump.
# If so, increment the count, otherwise store it.
   proc process_category {} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        if { [string length $::orig_msg] == 0} {
           return ""
        }
        set category "$::facility-$::severity-$::mnemonic"
        set correlation_window [expr [ query_category $category ] * 1000]
        if { $correlation_window == 0 } {
            return $::orig_msg
        }
        set message_args [join $::msg_args]
        set index "$category,[lindex $::msg_args 0]"
        if { [info exists cat_msg_count($index)] } {
                 incr cat_msg_count($index)
         } else {
              set cat_msg_sev($index) $::severity
              set cat_msg_traceback($index) $::traceback
              set cat_msg_pid($index) $::pid
              set cat_msg_proc($index) $::process
              set cat_msg_ts($index) $::timestamp
              set cat_msg_buginfseq($index) $::buginfseq
              set cat_msg_name($index) $::mnemonic
              set cat_msg_fac($index) $::facility
              set cat_msg_format($index) $::format_string
              set cat_msg_args($index) $message_args
              set cat_msg_count($index) 1
              set cat_msg_dump_ts($index) [clock seconds]
              catch [after $correlation_window counting::dump_category $index]
         }
         return ""
   }
# Initialized
```

```
set init 1
} ;#end namespace counting
```

例：XML タギング

この ESM syslog フィルタ モジュールは、ユーザ定義の XML タグを syslog メッセージに適用します。

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  XML Tagging Filter                                  ||||      ||||
#                                                  ..:||||||:..:||||||:..
#                                                 -------------------------
#                                                C i s c o  S y s t e m s
#  ====================================================================
#
# Usage: Define desired tags below.
#
# Namespace: xml
# Check for null message
    if { [string length $::orig_msg] == 0} {
       return ""
    }
namespace eval xml {
####  define tags ####
set MSG_OPEN "<ios-log-msg>"
set MSG_CLOSE "</ios-log-msg>"
set FAC_OPEN   "<facility>"
set FAC_CLOSE  "</facility>"
set SEV_OPEN   "<severity>"
set SEV_CLOSE  "</severity>"
set MNE_OPEN   "<msg-id>"
set MNE_CLOSE  "</msg-id>"
set SEQ_OPEN   "<seq>"
set SEQ_CLOSE  "</seq>"
set TIME_OPEN  "<time>"
set TIME_CLOSE "</time>"
set ARGS_OPEN  "<args>"
set ARGS_CLOSE "</args>"
set ARG_ID_OPEN "<arg id="
set ARG_ID_CLOSE "</arg>"
set PROC_OPEN "<proc>"
set PROC_CLOSE "</proc>"
set PID_OPEN "<pid>"
set PID_CLOSE "</pid>"
set TRACE_OPEN "<trace>"
set TRACE_CLOSE "</trace>"
#  ======================= End User Setup ==============================
#### clear result ####
set result ""
#### message opening, facility, severity, and name ####
append result $MSG_OPEN $FAC_OPEN $::facility $FAC_CLOSE $SEV_OPEN $::severity
$SEV_CLOSE $MNE_OPEN $::mnemonic $MNE_CLOSE
#### buginf sequence numbers ####
if { [string length $::buginfseq ] > 0 } {
    append result $SEQ_OPEN $::buginfseq $SEQ_CLOSE
}
#### timestamps ####
if { [string length $::timestamp ] > 0 } {
    append result $TIME_OPEN $::timestamp $TIME_CLOSE
}
```

```
#### message args ####
if { [info exists ::msg_args] } {
    if { [llength ::msg_args] > 0 } {
        set i 0
        append result $ARGS_OPEN
        foreach arg $::msg_args {
            append result $ARG_ID_OPEN $i ">" $arg $ARG_ID_CLOSE
                incr i
        }
        append result $ARGS_CLOSE
    }
}
#### traceback ####
if { [string length $::traceback ] > 0 } {
    append result $TRACE_OPEN $::traceback $TRACE_CLOSE
}
#### process ####
if { [string length $::process ] > 0 } {
    append result $PROC_OPEN $::process $PROC_CLOSE
}
#### pid ####
if { [string length $::pid ] > 0 } {
    append result $PID_OPEN $::pid $PID_CLOSE
}
#### message close ####
append result $MSG_CLOSE
return "$result"
} ;# end namespace xml
```

例：SMTP ベースの電子メール アラート

この ESM syslog フィルタ モジュール例では、コンフィギュレーション メッセージを監視し、CLI 引数として供給される E メール アドレスに送信します。このフィルタは 2 つのファイルに分割されています。最初のファイルはフィルタを実装し、2 番目のファイルは Simple Mail Transfer Protocol（SMTP）クライアントを実装します。

```
#  ====================================================================
#  Embedded Syslog Manager                            ||        ||
#                                                     ||        ||
#  Email Filter                                      ||||      ||||
#  (Configuration Change Warning)                 ..:||||||:..:||||||:..
#                                                ------------------------
#                                                 C i s c o  S y s t e m s
#   ====================================================================
# Usage:  Provide email address as CLI argument.  Set email server IP in
#         email_guts.tcl
#
# Namespace: email
if { [info exists email::init] == 0 } {
   source tftp://123.123.123.123/ESM/email_guts.tcl
}
# Check for null message
if { [string length $::orig_msg] == 0} {
      return ""
   }
if { [info exists ::msg_args] } {
    if { [string compare -nocase CONFIG_I $::mnemonic ] == 0 } {
                email::sendmessage $::cli_args $::mnemonic \
                [string trim $::orig_msg]
    }
```

```
}
return $::orig_msg
```

E メール サポート モジュール（email_guts.tcl）

```
#  ====================================================================
#  Embedded Syslog Manager                              ||        ||
#                                                       ||        ||
#  Email Support Module                                 ||||      ||||
#                                                  ..:||||||:..:||||||:..
#                                                 -----------------------
#                                                  C i s c o  S y s t e m s
#   ====================================================================
#
# Usage: Set email host IP, from, and friendly strings below.
#
namespace eval email {
    set sendmail(smtphost)172.16.0.1
    set sendmail(from) $::hostname
    set sendmail(friendly) $::hostname
    proc sendmessage {toList subject body} {
        variable sendmail
        set smtphost $sendmail(smtphost)
        set from $sendmail(from)
        set friendly $sendmail(friendly)
        set sockid [socket $smtphost 25]
## DEBUG
set status [catch {
        puts $sockid "HELO $smtphost"
        flush $sockid
        set result [gets $sockid]
        puts $sockid "MAIL From:<$from>"
        flush $sockid
        set result [gets $sockid]
        foreach to $toList {
            puts $sockid "RCPT To:<$to>"
            flush $sockid
        }
        set result [gets $sockid]
        puts  $sockid "DATA "
        flush $sockid
        set result [gets  $sockid]
        puts  $sockid "From: $friendly <$from>"
        foreach to $toList {
            puts $sockid "To:<$to>"
        }
        puts  $sockid "Subject: $subject"
        puts  $sockid "\n"
        foreach line [split $body "\n"] {
            puts  $sockid " $line"
        }
        puts  $sockid "."
        puts  $sockid "QUIT"
        flush $sockid
        set result [gets  $sockid]
} result]
        catch {close $sockid }
    if {$status} then {
        return -code error $result
    }
}
} ;# end namespace email
set email::init 1
```

例：ストリーム

この ESM syslog フィルタ モジュールの例では、特定のファシリティを監視し（最初の CLI 引数）、これらのメッセージを特定のストリームにルーティングします（2 番目の CLI 引数）。

```
#  ===================================================================
#  Embedded Syslog Manager                                 ||        ||
#                                                          ||        ||
#  Stream Filter (Facility)                               ||||      ||||
#                                                      ..:||||||:..:||||||:..
#                                                     ------------------------
#                                                      C i s c o  S y s t e m s
#   ===================================================================
# Usage:  Provide facility and stream as CLI arguments.
#
# Namespace: global
# Check for null message
#  ====================== End User Setup ==============================
set args [split $::cli_args]
if { [info exists ::msg_args] } {
    if { $::facility == [lindex $args 0] } {
                set ::stream [lindex $args 1]
    }
}
return $::orig_msg}
```

例：送信元 IP タギング

logging source-interface CLI コマンドを使用すると、デバイスから送信されるすべての syslog パケットで送信元 IP アドレスを指定できます。次の syslog フィルタモジュールの例では、フィルタモジュール内で **show** CLI コマンド（この場合は **show running-config** および **show ip interface**）を使用して、syslog メッセージに送信元 IP アドレスを追加する方法を示します。スクリプトは、最初に「source_ip::init」というローカル名前空間変数を検索します。処理される最初の syslog メッセージで変数が定義されていない場合、フィルタは **show** コマンドを実行し、正規表現を使用して、送信元インターフェイスとその IP アドレスを取得します。

このスクリプトでは、**show** コマンドが 1 回だけ実行されることに注意してください。送信元インターフェイスまたはその IP アドレスが変更される場合、新しい情報を取得するためにフィルタを再び初期化する必要があります（show コマンドをすべての syslog メッセージで実行できますが、これはうまく拡張できません）。

```
#  ===================================================================
#  Embedded Syslog Manager                                 ||        ||
#                                                          ||        ||
#  Source IP Module                                       ||||      ||||
#                                                      ..:||||||:..:||||||:..
#                                                     ------------------------
#                                                      C i s c o  S y s t e m s
#   ===================================================================
# Usage: Adds Logging Source Interface IP address to all messages.
#
# Namespace:source_ip
#
#  ====================== End User Setup ==============================
namespace eval ::source_ip {
    if { [info exists init] == 0 } {
       if { [catch {regexp {^logging source-interface (.*$)} [exec show
```

```
run | inc logging source-interface] match source_int}]} {
           set suffix "No source interface specified"
       } elseif { [catch {regexp {Internet address is (.*)/.*$} [exec
show ip int $source_int | inc Internet] match ip_addr}]} {
           set suffix "No IP address configured for source interface"
       } else {
           set suffix $ip_addr
       }
       set init 1
    }

    if { [string length $::orig_msg] == 0} {
      return ""
    }
    return "$::orig_msg - $suffix"
} ;# end namespace source_ip
```

Embedded Syslog Manager に関する追加情報

関連資料

関連項目	マニュアル タイトル
Cisco IOS XE コマンド	『Command Lookup Tool』
システム メッセージ ロギング	「Troubleshooting and Fault Management」モジュール
XML 形式のシステム メッセージ ロギング	「XML Interface to Syslog Messages」モジュール
シスコ ソフトウェアでの Tcl 8.3.4 サポート	*Cisco IOS Scripting with Tcl* のモジュール
ネットワーク管理コマンド（logging コマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンドモード、コマンド履歴、使用上のガイドライン、および例	『*Cisco IOS Network Management Command Reference*』

標準および RFC

標準/RFC	タイトル
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	--

標準/RFC	タイトル
RFC-3164	*The BSD Syslog Protocol* この RFC は、情報提供のために記載しています。シスコによる syslog の実装では、この RFC で言及されているプロトコル ガイドラインとの完全な準拠性を要求していません。 サポートされている RFC がすべて記載されているわけではありません。

MIB

MIB	MIB のリンク
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Embedded Syslog Manager の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 78: Embedded Syslog Manager の機能情報

機能名	リリース	機能情報
Embedded Syslog Manager		Embedded Syslog Manager（ESM）機能は、Cisco IOS システム メッセージ ロガーによる伝送前に、システム ロギング メッセージのフィルタリング、拡大、相互の関連付け、ルーティング、およびカスタマイズを可能にするプログラマブル フレームワークを提供します。

用語集

Note この用語集に記載されていない用語については、『Internetworking Terms and Acronyms』を参照してください。

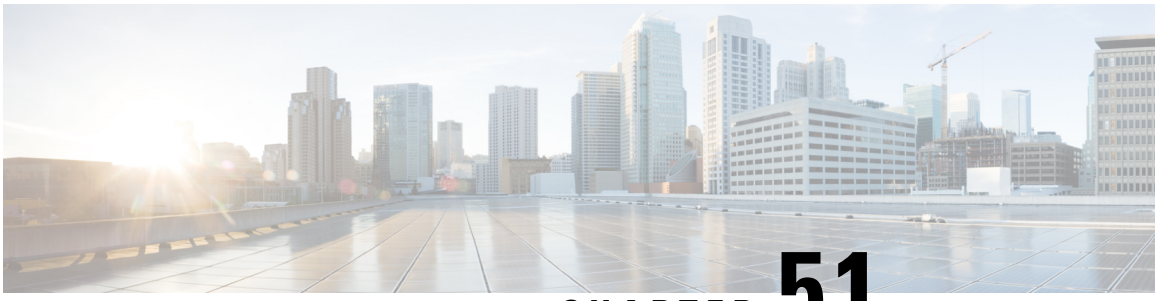
console：デバイスのコンソールポートへの接続（CTY またはコンソール回線）を示します。一般的にこれはコンソール ポートに直接接続された端末または端末エミュレーション プログラムを備えた PC です。**show terminal** コマンドに対応します。

monitor：ラインポートでの TTY（TeleTYpe 端末）ライン接続を示します。つまり、「モニタ」キーワードは、端末回線接続または Telnet（端末エミュレーション）接続に対応します。TTY ライン（ポートとも呼ばれます）は、端末、モデム、シリアル プリンタなどの周辺装置と通信します。TTY 接続の例として、ダイヤルアップ モデムを使用してデバイスに接続する端末エミュレーション プログラムを備えた PC があります。

SEMs：システムエラーメッセージの略。システムロギング（syslog）プロセスによって生成されるメッセージを指す用語として、「システム エラー メッセージ」が以前に使用されていました。syslog メッセージは標準化された形式を使用し、「緊急」（レベル 0）から「デバッグ」（レベル 7）までの 8 つのシビラティ（重大度）があります。これらのメッセージには、エラー以外のデバイス動作の通知（情報の通知など）が含まれる場合もあるため、「システムエラーメッセージ」という用語は、実際に誤解を招くおそれがあります。

syslog：シスコソフトウェアでのシステム メッセージ ロギング プロセスの略。また、「syslog メッセージ」のように、生成されたメッセージを指す場合もあります。専門用語としての「syslog」はリモート ホストへのメッセージ ロギング プロセスのみを指しますが、一般的には、すべてのシスコ システム ロギング プロセスを示すために使用されます。

trap：エラーメッセージを送信するためのシステムソフトウェア内のトリガー。「トラップ ロギング」とは、リモート ホストへのメッセージのロギングを意味します。リモート ホストはトラップ メッセージを送信するデバイスから見ると実際には syslog ホストですが、受信側デバイスは、収集された syslog データを他のデバイスに提供することが多いため、受信側デバイスもまた「syslog サーバ」と呼ばれます。



CHAPTER 51

ローカル不揮発性ストレージへのロギング

ローカル不揮発性ストレージへのロギング機能では、システム ロギング メッセージを Advanced Technology Attachment フラッシュ ディスクに保存できます。デバイスが再起動しても、ブートフラッシュまたはハードディスクに保存されたメッセージは消去されません。

ローカル不揮発性ストレージへのロギングの前提条件

logging buffered コマンドをイネーブルにする

logging persistent コマンドを使用して、ローカル不揮発性ストレージへのロギング機能をイネーブルにする前に、**logging buffered** コマンドを使用して、内部バッファへのメッセージのロギングをイネーブルにする必要があります。詳細については、「ブートフラッシュまたはハードディスクへのロギング メッセージの書き込み」セクションを参照してください。

ローカル不揮発性ストレージへのロギングの制約事項

使用できるブートフラッシュまたはハードディスクの容量によって、保存されるログ ファイルのサイズと数が制限される

システム ロギング メッセージに割り当てられるブートフラッシュまたはハードディスクの容量によって、保存できるロギング ファイルの数が制限されます。割り当てしきい値を超えると、ディレクトリ内の最も古いログ ファイルが削除され、新しいシステム ロギング メッセージ用の容量を用意します。システム ロギング メッセージを恒久的に保存するには、外部デバ

イスにアーカイブする必要があります。詳細については、「外部ディスクへのロギング メッセージのコピー」セクションを参照してください。

Note ローカル不揮発性ストレージへのロギングは、最大2GBの記憶域を使用する場合があります。

ローカル不揮発性ストレージへのロギングに関する情報

システム ロギング メッセージ

システム ロギング メッセージには、デバイスのアプリケーション プログラミング インターフェイス（API）によって生成されたエラーおよびデバッグ メッセージが含まれます。一般的に、ロギング メッセージはデバイスのメモリ バッファに保存され、バッファが満杯になると古いメッセージが新しいメッセージで上書きされます。デバイスが再起動すると、すべてのロギング メッセージがメモリ バッファから消去されます。

ローカル不揮発性ストレージへのロギングの設定方法

ブートフラッシュまたはハードディスクへのロギングメッセージの書き込み

ローカル不揮発性ストレージへのロギング機能をイネーブルにし、ブートフラッシュまたはハードディスクにロギング メッセージを書き込むには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging buffered** [*buffer-size* | *severity-level*]
4. **logging persistent** [**url harddisk:**/*directory*] [**size** *filesystem-size*] [**filesize** *logging-file-size*]

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
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 モードをイネーブルにします。
ステップ 3	**logging buffered** [*buffer-size* | *severity-level*] **Example:** `Device(config)# logging buffered`	ローカル バッファへのシステム メッセージ ロギングをイネーブルにし、バッファにロギングするメッセージをシビラティ（重大度）に基づいて制限します。 • オプションの *buffer-size* の引数は、バッファのサイズを指定します。指定できる値の範囲は 4096 ～ 4294967295 です。デフォルトのサイズは、プラットフォームによって異なります。 • オプションの *severity-level* 引数は、バッファへのメッセージのロギングを、指定されたシビラティ（重大度）またはそれ以上のレベルのメッセージに制限します。
ステップ 4	**logging persistent** [**url harddisk:**/*directory*] [**size** *filesystem-size*] [**filesize** *logging-file-size*] **Example:** `Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384` **Note** デフォルト値は次のとおりです。url: bootflash:/syslog filesystem-size: 10% of total disk space logging-file-size: 262144	メモリ バッファ内のロギング メッセージを、デバイスのブートフラッシュまたはハードディスク上の指定のディレクトリに書き込みます。 • ロギング メッセージをブートフラッシュまたはハードディスク上のファイルに書き込む前に、シスコ ソフトウェアは、十分なディスク領域があるかどうかをチェックします。十分なディスク スペースがない場合、ロギング メッセージの最も古いファイル（タイムスタンプによる）が削除され、現在のファイルが保存されます。 • ログ ファイルのファイル名フォーマットは log_MM:DD:YYYY::hh:mm:ss です（例：log_11:26:2012::01:01:41）。 **Note** この機能は、そのファイル名の形式（秒レベルまでのタイムスタンプ サフィックスが含まれている）により、1 秒あたり 1 つのログ ファイルだけをサポートします。

	Command or Action	Purpose
		Note このコマンドのデフォルトは次のとおりです。 • **url: bootflash:/syslog** Filesystem-size: 10% of total disk space. Logging-file-size: 262144

外部ディスクへのロギング メッセージのコピー

ロギング メッセージを、ブートフラッシュまたはハードディスクから外部ディスクにコピーするには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **copy** *source-url destination-url*

DETAILED STEPS

	Command or Action	Purpose
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	**copy** *source-url destination-url* **Example:** `Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog`	ブートフラッシュまたはハードディスク上の指定のファイルまたはディレクトリを、FTP 経由で指定の URL にコピーします。

ローカル不揮発性ストレージへのロギングの設定例

例：ブートフラッシュまたはハードディスクへのロギングメッセージの書き込み

次に、最大 134217728 バイト（128 MB）のロギング メッセージをディスク 0 の syslog ディレクトリに書き込み、16384 バイトのファイル サイズを指定する例を示します。

```
Device(config)# logging buffered
Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384
```

例：外部ディスクへのロギング メッセージのコピー

次に、ロギング メッセージをデバイスのブートフラッシュまたはハードディスクから外部ディスクにコピーする例を示します。

```
Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
copy コマンド	Cisco IOS Configuration Fundamentals Command Reference
ネットワーク管理コマンド（logging コマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンド モード、コマンド履歴、使用上のガイドライン、および例	『Cisco IOS Network Management Command Reference』

MIB

MIB	MIB のリンク
• この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存MIB のサポートに変更はありません。	選択したプラットフォーム、Cisco IOS XE リリース、およびフィーチャ セットの MIB を検索してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

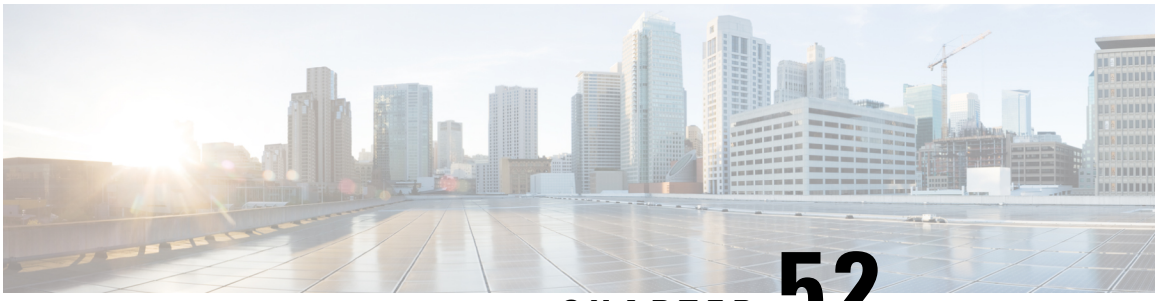
ローカル不揮発性ストレージへのロギングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Table 79: ローカル不揮発性ストレージへのロギングの機能情報

機能名	リリース	機能情報
ローカル不揮発性ストレージへのロギング	Cisco IOS XE Release 2.1	ローカル不揮発性ストレージへのロギング機能では、システム ロギング メッセージを Advanced Technology Attachment フラッシュ ディスクに保存できます。デバイスが再起動しても、ブートフラッシュまたはハードディスクに保存されたメッセージは消去されません。 次のコマンドが導入または変更されました。**logging persistent**。



CHAPTER 52

syslogの信頼性の高い伝送およびフィルタリング

syslog の信頼性の高い伝送およびフィルタリング機能によって、デバイスを syslog メッセージの受信用にカスタマイズできます。この機能は、ブロック拡張可能交換プロトコル（BEEP）を使用したsyslogメッセージの信頼性の高いセキュアな伝送を提供します。さらに、基盤となる転送方式にかかわらず、1つのロギング ホストへの複数のセッションを可能にし、メッセージ ディスクリミネータと呼ばれるフィルタリング メカニズムを提供します。

この章では、syslog 機能のための信頼性の高い伝送およびフィルタリングの機能と、それらのネットワーク内での設定方法について説明します。

syslog の信頼性の高い伝送およびフィルタリングの前提条件

- デバイス レベルのレート制限を、ビジネス要件、ネットワーク トラフィック要件、またはパフォーマンス要件を満たすように設定します。
- 各 BEEP セッションには、RFC 3195 準拠の syslog-RAW 交換プロファイルが含まれている必要があります。
- 暗号イメージを使用する場合、プロビジョニング サービスに「DIGEST-MD5」を指定する Simple Authentication and Security Layer（SASL）プロファイルを確立する必要があります。

- syslog サーバは BEEP と互換性がある必要があります。
- syslog の信頼性の高い伝送およびフィルタリング機能の複数セッション機能を使用するには、syslogサーバアプリケーションは、複数のセッションを処理できる必要があります。

syslog の信頼性の高い伝送およびフィルタリングの制約事項

- syslog-RAW、SASL、およびトランスポート層セキュリティ（TLS）プロファイルだけがサポートされています。
- syslog セッションの両端で同じ転送方式を使用する必要があります。
- メッセージ ディスクリミネータを特定の syslog セッションに関連付けるには、事前に定義する必要があります。
- syslog セッションは、1 つのメッセージ ディスクリミネータとだけ関連付けることができます。
- ユーザ データグラム プロトコル（UDP）によるメッセージ伝送は、TCP または BEEP による伝送よりも速くなります。

syslog の信頼性の高い伝送およびフィルタリングに関する情報

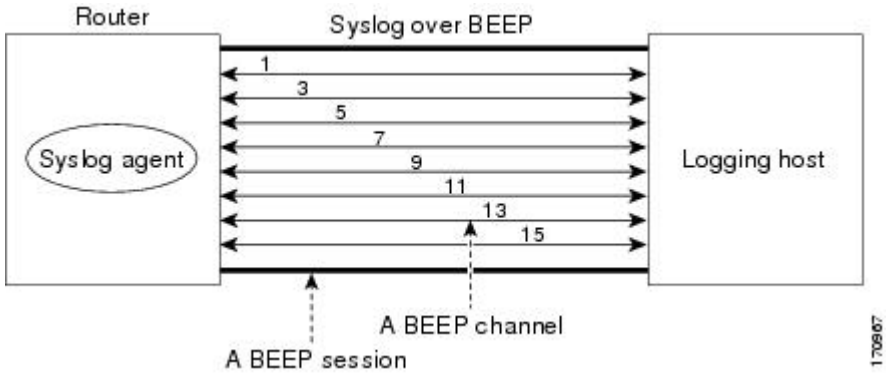
BEEP 転送のサポート

BEEPは、コネクション型非同期相互作用のための汎用アプリケーションプロトコルフレームワークです。これは、従来さまざまなプロトコルの実装で何度も利用されてきた機能を提供することを目的としています。BEEP は一般的に TCP 上で動作し、メッセージの交換が可能です。HTTP および同様のプロトコルとは異なり、接続の両端でいつでもメッセージを送信できます。BEEP には暗号化と認証のファシリティも含まれており、高い拡張性があります。

syslog メッセージの転送プロトコルとしての BEEP は、複数のチャネルを提供します。各チャネルは、同じホストへの異なるセッションに設定できます。BEEP は信頼性の高い転送を提供します。BEEP 接続を介して送信される syslog メッセージは、順序どおりに伝送されることが保証されます。

syslog の信頼性の高い伝送およびフィルタリング機能に導入されたコマンドライン インターフェイス（CLI）によって、新しい BEEP セッションが最大 8 つのチャネルを持つように設定できます。

次の図は、8 つの異なる syslog セッションを実現する、8 つのチャネルによる BEEP セッションを示しています。



チャネルは1、3、5、7、9、11、13、および15と識別されます。使用できるチャネルの数（8）は、従来の RFC-3164 syslog メッセージ（0 ～ 7）のシビラティ（重大度）の番号に対応して設計されています。メッセージディスクリミネータは、シビラティ（重大度）がBEEPチャネルにマッピングされるように使用できます。インテリジェントな BEEP syslog サーバ（使用する BEEP スタックによって異なる）は、このマッピングを使用して、シビラティ（重大度）の高いメッセージを優先できます（RFC 3081、セクション 3.1.4 を参照）。メッセージディスクリミネータと関連付けられている場合を除いて、すべてのsyslogセッション（チャネル）は、すべての syslog メッセージを受信します。

syslog メッセージ

syslog メッセージには、ホストが番号をメッセージの識別子として使用し、受信されたメッセージにギャップがあるかどうか検出できるようにするシーケンス番号があります。syslog メッセージには連続した番号が付けられます。BEEP の信頼性は、シーケンス番号（次の理由によって必要である）の必要性に代わるものではありません。

- シーケンス番号は、syslog メッセージを識別する簡単な方法を提供します。信頼性に関する問題に関係なく、シーケンス番号はメッセージ識別子として機能します。
- BEEP セッションは、syslog メッセージを送信するデバイスがアップ状態の間ずっと機能しているとは限りません。シーケンス番号は、BEEP セッションの間にメッセージが失われたかどうかを管理アプリケーションが評価する方法を提供します。
- BEEP はいくつかの転送のうちの 1 つにすぎません。信頼性の低い転送も使用されるので、syslog プロトコルは、常に提供されている信頼性の高い転送に依存すべきではありません。

syslog メッセージの既存の番号付け方式は、高度なメッセージの識別機能および複数のホストに対応するために、syslog の拡張で制限されます。メッセージの識別によって、シーケンス番号にギャップが発生します。つまり、ホストは、メッセージを見逃したかどうかを検出する能力を失います。シーケンス番号にギャップが発生しないよう、各セッションでsyslogメッセージに連番が付けられた場合は、シーケンス番号でメッセージが一意に識別されなくなるため、どのメッセージが同じで、どのメッセージが異なるかを簡単に関連付けできなくなります。

識別をシーケンスおよび信頼性から分離するために、syslog メッセージに対して、次の変更が行われました。

- シーケンス番号は、メッセージの識別子として保持されます。低い番号を持つメッセージは、高い番号を持つメッセージよりも優先されますが、連続するように保証されていません。
- シーケンスを保証するために、syslog メッセージの本文の部分に追加フィールドが追加されます。このフィールドの内容には、特定のセッションのシーケンス番号が含まれています。複数のセッションで送信される同じメッセージに、それぞれ異なるシーケンス番号が付けられる可能性があります。

syslog セッション

syslog セッションは、デバイス上の syslog エージェントから syslog メッセージの受信者への論理リンクです。たとえば、syslog エージェントと次のいずれかとの間で syslog セッションを確立できます。

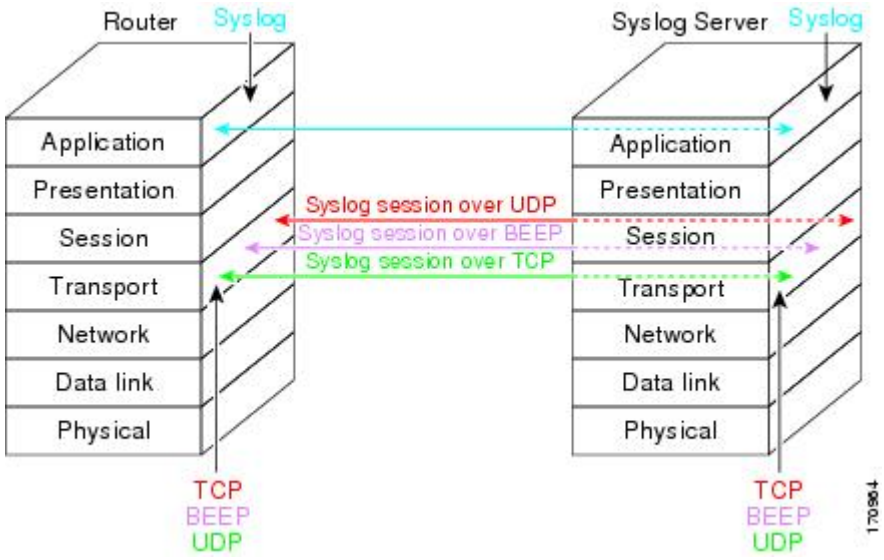
- デバイス コンソール
- デバイスのロギング バッファ
- デバイス モニタ
- 外部 syslog サーバ

syslog セッションは、syslog の送信元と syslog の宛先の間の転送接続で動作します。転送接続では次の任意のプロトコルを使用できます。

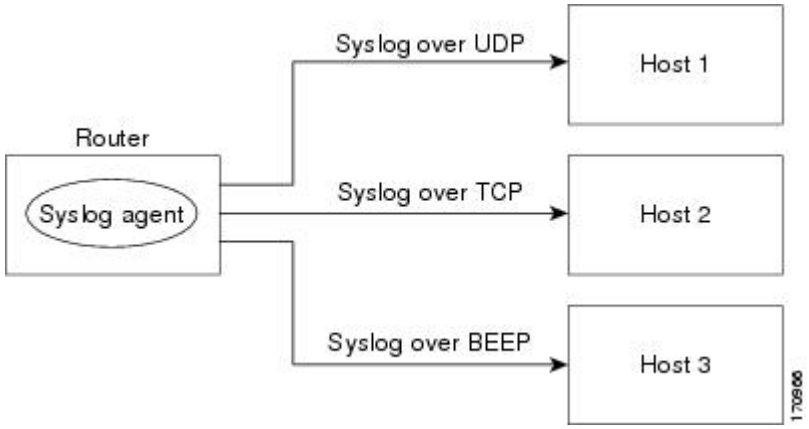
- TCP
- UDP（1 つのリモート アドレスおよびポートとの関連付け）
- BEEP（BEEP セッション内のチャネル）

次の図に、オープン システム相互接続（OSI）モデルを使用したデバイスと syslog サーバ間の syslog セッションおよび転送プロトコルのマッピングを示します。

Note 次の図は Internet Explorer を使用すると最適に表示されます。



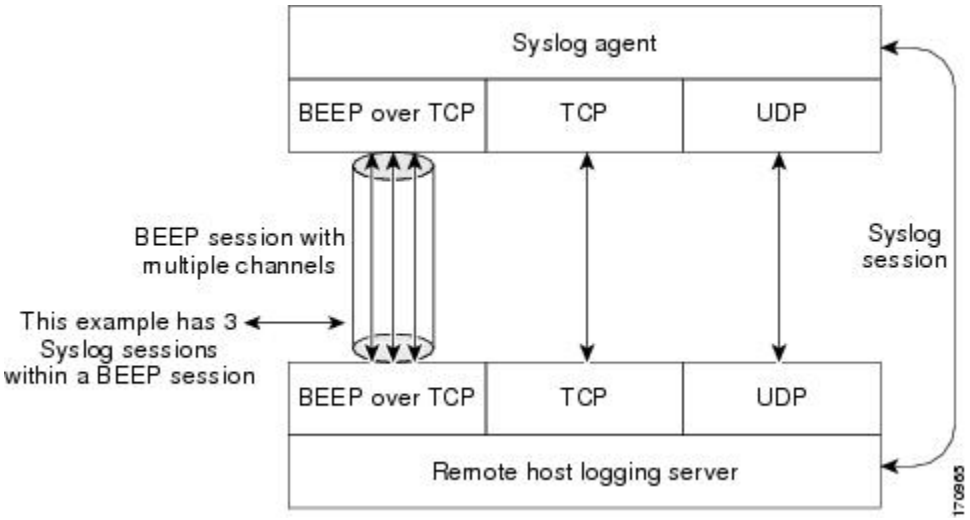
次の図に、UDP、TCP、および BEEP を使用した 1 つの syslog エージェントから複数のホストへの複数の syslog セッションを示します。



複数の syslog セッション

syslogセッションは、転送接続に依存しません。シスコデバイスは、それぞれが独自の転送接続で動作する複数の syslog セッションをサポートできます。複数の syslog セッションで同じ転送接続を共有できませんが、それぞれ独自の転送接続で動作している複数の syslog セッションを、同じリモートホストで終端することはできます。1 つの例として、複数のチャネルが使用される BEEP セッションがあります。

次の図は、エンドツーエンドの syslog セッションを示しています。1 つの BEEP セッションに 3 つの syslog セッションがあることに注目してください。



TCP プロトコルと UDP プロトコルには多重チャネルはありませんが、これらのプロトコルでは複数のポートを使用して、syslog ホストへの複数の syslog セッションを確立できます。UDP および TCP 転送方式が BEEP の複数チャネル機能と同様の機能を持つようにするために、syslog の信頼性の高い伝送およびフィルタリング機能では、UDP および TCP 転送方式によって、同じロギング ホストへの複数の syslog セッションを確立できます。BEEP セッションを経由する syslog セッションもサポートされています。

メッセージ ディスクリミネータ

メッセージ ディスクリミネータは syslog プロセッサです。メッセージ ディスクリミネータは syslog セッションに関連付けられ、セッションを転送接続にバインドします。

メッセージ送信の前に、メッセージは、ユーザが指定した基準リストを持つメッセージディスクリミネータの影響を受けます。最初のフィルタリング基準によってメッセージがブロックされると、フィルタリング チェックが停止します。

Note CLI の基準の順序は、基準がチェックされる順番には影響しません。

- フィルタリング基準は次のとおりです。これらの基準は、次にリストされている順序でチェックされます。
  - 指定されたシビラティ（重大度）
  - 正規表現と一致するメッセージ本文内のファシリティ
  - 正規表現と一致するニーモニック
  - 正規表現と一致するメッセージ本文の部分

メッセージ ディスクリミネータは、次の機能を提供します。

- オプションのレート制限：超えてはならない、時間間隔あたりのメッセージの転送レートの指定。レート制限を超えた場合、メッセージは、デバイスの判断で遅延するか廃棄され

ます。レート制限の適用は、その syslog セッションでの syslog メッセージの信頼性の高い伝送が保証されなくなったことを意味します。レート制限の目的は、受信者の syslog サーバで、syslog の保証された伝送を必要としないアプリケーションの「フラッディング」が発生する可能性を回避することにあります。

- 相互関連付け：候補となるイベントメッセージを検査し、イベント全体の情報を可能な限り集約して、集約された情報を含む新しいイベントを作成。関連する機能は次のとおりです。
  - メッセージカウントを維持し、特定のタイプの最初のメッセージの送信とそのタイプの次のメッセージの送信の間の特定の時間待機することによる、重複メッセージの除去。
  - 変動するメッセージの除去
  - 単純なメッセージの相互関連付け。たとえば、あるメッセージが別のメッセージによって報告された原因の症状である場合、1 つの統合されたメッセージが報告されます。

メッセージディスクリミネータは、特定の宛先および転送と関連付けることができます。つまり、フィルタはホストに依存する可能性があります。このため、メッセージディスクリミネータは、それぞれ異なるディスクリミネータに適用できる複数のセッション、転送、またはチャネルに対して可能なデバイスサポートによって、syslogセッション、転送、またはチャネルに適用されます。

メッセージディスクリミネータの確立は、syslogセッションの確立から分離している必要があります。メッセージディスクリミネータは、適用される syslogセッション、転送、またはチャネルを参照する必要があります。分離の理由は次のとおりです。

- メッセージディスクリミネータは接続から個別に管理でき、メッセージディスクリミネータの設定に使用できる機能の調整を syslog セッションの設定方法に反映させる必要はなく、その逆も同様です。
- 複数の接続を同じメッセージ ディスクリミネータに適用でき、さまざまな syslog 冗長性トポロジが可能です。

明示的なメッセージ ディスクリミネータが syslog セッションと関連付けられていない場合、デバイス全体のグローバル設定から汎用メッセージディスクリミネータが使用されます。属性値を指定せずに、「空の」メッセージディスクリミネータを作成できます（レート制限もフィルタも設定されません）。

レート制限

Cisco IOS XE syslog でのデバイス全体のレート制限機能は、syslog の信頼性の高い伝送およびフィルタリング機能に保存され、「グローバル レート制限」と呼ばれています。グローバル レート制限を使用しない場合、システム リソースがその量をサポートできる場合には、すべてのイベント メッセージがリモート syslog ホストに送信されます。グローバル レート制限が設定されると、すべての宛先に適用されます。この値は、「汎用メッセージ ディスクリミネータ」のレート制限属性（設定されている場合）に設定されます。グローバル レート制限の欠点

は、最も性能の低いリモート syslog ホストのレート制限によって、デバイスが syslog メッセージを送信する速度が設定されることです。

syslog の信頼性の高い伝送およびフィルタリング機能は、syslog セッションベースのレート制限を提供し、グローバル レート制限の影響を回避します。このセッションベースのレート制限は、特定のメッセージ ディスクリミネータと関連付けられており、各 syslog セッションに対して、レート受け入れレベルを別々に設定できます。

セッションベースのレート制限が行われている場合、グローバル レート制限の使用は推奨されません。メッセージディスクリミネータのレート制限は、syslog メッセージの超えてはならないレートを指定しますが、このレートに到達することを保証しません。設定されたグローバル レート制限によって、セッションのレート制限に到達していない場合でも、そのセッションのメッセージが廃棄されることがあります。グローバル レート制限とセッションベースのレート制限が同時に使用される場合、これらの動作について理解することが重要です。

syslog の信頼性の高い伝送およびフィルタリングの利点

- BEEP の認証機能および暗号化機能では、syslog メッセージの信頼性の高いセキュアな伝送が提供されます。
- 基盤となる転送方式に依存しない 1 つのロギング ホストへの複数のセッション
- セッションベースのメッセージ フィルタリングおよびレート制限
- 複数の接続を同じメッセージ ディスクリミネータに適用でき、さまざまな syslog 冗長性トポロジが可能です。
- デフォルトの syslog カウントを無効にする新しい CLI コマンド
- レート制限によって廃棄される syslog メッセージの関連部分の識別に役立つ新しい CLI コマンド

syslog の信頼性の高い伝送およびフィルタリングの設定方法

メッセージ ディスクリミネータの作成

syslog メッセージのメッセージディスクリミネータを作成するには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]

4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr1 facility includes facl357`	ファシリティ サブフィルタを持つメッセージ ディスクリミネータを作成します。 この例では、ファシリティ フィールドに「facl357」があるすべてのメッセージが伝送されます。
ステップ 4	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータのロギング バッファとの関連付け

メッセージディスクリミネータを特定のバッファと関連付けるには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*\| **includes** *string*}] [**severity** {**drops** *sev-num* \| **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging buffered** [**discriminator** *discr-name* \| **xml**] [*buffer-size*] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:**	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）。

	Command or Action	Purpose
	`Device> enable`	
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	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr2`	メッセージ ディスクリミネータを作成します。
ステップ 4	**logging buffered** [**discriminator** *discr-name* | **xml**] [*buffer-size*] [*severity-level*] **Example:** `Device(config)# logging buffered discriminator pacfltr2 5`	ローカルバッファへのロギングをイネーブルにし、メッセージ ディスクリミネータを指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータのコンソール端末との関連付け

メッセージディスクリミネータをコンソール端末と関連付けるには、次の作業を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging console** [**discriminator** *discr-name* | **xml**] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	**enable** **Example:**	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）。

	Command or Action	Purpose
	`Device> enable`	
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	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr3`	メッセージ ディスクリミネータを作成します。
ステップ 4	**logging console** [**discriminator** *discr-name* | **xml**] [*severity-level*] **Example:** `Device(config)# logging console discriminator pacfltr3 1`	コンソールへのロギングをイネーブルにして、特定のシビラティ（重大度）のメッセージをフィルタリングするメッセージディスクリミネータを指定します。
ステップ 5	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータの端末回線との関連付け

メッセージディスクリミネータを端末回線に関連付け、モニタにメッセージを表示するには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*\| **includes** *string*}] [**severity** {**drops** *sev-num* \| **includes** *sev-num*}] [**rate-limit** *msglimit*]
4. **logging monitor** [**discriminator** *discr-name*\| **xml**] [*severity-level*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**logging discriminator** *discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops** *string*| **includes** *string*}] [**severity** {**drops** *sev-num* | **includes** *sev-num*}] [**rate-limit** *msglimit*] **Example:** `Device(config)# logging discriminator pacfltr4`	メッセージ ディスクリミネータを作成します。
ステップ 4	**logging monitor** [**discriminator** *discr-name*| **xml**] [*severity-level*] **Example:** `Device(config)# logging monitor discriminator pacfltr4 2`	pacfltr4 という名前のメッセージ ディスクリミネータを指定し、シビラティ（重大度） 2 以下で端末回線へのメッセージのロギングをイネーブルにします。
ステップ 5	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ カウンタのイネーブル化

デバッグ、ログ、またはsyslogメッセージのロギングをイネーブルにするには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging message-counter** {**debug** | **log** | **syslog**}
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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	**logging message-counter** {**debug** | **log** | **syslog**} **Example:** `Device(config)# logging message-counter syslog`	syslog メッセージのロギングをイネーブルにします。
ステップ 4	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

BEEP セッションの追加と削除

BEEP セッションを追加および削除するには、次の手順を実行します。

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging host** {{*ip-address* | *hostname*} [**vrf** *vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator** *discr-name* | [[**filtered** [**stream** *stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel** *chnl-number*] [**sasl** *profile-name*] [**tls cipher** [*cipher-num*] **trustpoint** *trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port** *port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string** *custom-string*}]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
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）。

	Command or Action	Purpose
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	**logging host** {{*ip-address* | *hostname*} [**vrf** *vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator** *discr-name* | [[**filtered** [**stream** *stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel** *chnl-number*] [**sasl** *profile-name*] [**tls cipher** [*cipher-num*] **trustpoint** *trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port** *port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string** *custom-string*}] **Example:** `Device(config)# logging host host3 transport beep port 600 channel 3`	ロギング ホストを指定し、メッセージのロギングのための転送プロトコル、ポート、およびチャネルを指定します。
ステップ 4	**end** **Example:** `Device(config)# end`	CLI を特権 EXEC モードに戻します。

syslog の信頼性の高い伝送およびフィルタリングの設定例

転送とロギングの設定例

```
Device(config)# do show running-config
 | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
Device(config)# logging host 209.165.201.1 transport beep port 600 channel 3
Device(config)# logging host 209.165.201.1 transport tcp port 602

Device(config)# show running-config | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
logging host 209.165.201.1 transport beep port 600 channel 3
logging host 209.165.201.1 transport tcp port 602
Device(config)#
```

syslog トランザクションの VRF 対応送信元インターフェイスに関する追加情報

関連資料

関連項目	マニュアル タイトル
ネットワーク管理コマンド（logging コマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンド モード、コマンド履歴、使用上のガイドライン、および例	『*Cisco IOS Network Management Command Reference*』
Syslog ロギング	*Troubleshooting and Fault Management module*

標準および RFC

標準/RFC	タイトル
この機能によりサポートされる新規または変更された標準/RFC はありません。またこの機能による既存の標準/RFC のサポートに変更はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャ セットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

syslog の信頼性の高い伝送およびフィルタリングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

***Table 80: syslog* の信頼性の高い伝送およびフィルタリングの機能情報**

機能名	リリース	機能情報
syslog の信頼性の高い伝送およびフィルタリング	Cisco IOS XE Release 2.1	syslog の信頼性の高い伝送およびフィルタリング機能によって、デバイスを syslog メッセージの受信用にカスタマイズできます。この機能は、BEEP を使用した syslog メッセージの信頼性の高いセキュアな伝送を提供します。さらに、基盤となる転送方式にかかわらず、1 つのロギング ホストへの複数のセッションを可能にし、メッセージディスクリミネータと呼ばれるフィルタリング メカニズムを提供します。 Cisco IOS XE リリース 2.1 では、この機能は Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに導入されました。 次のコマンドが導入または変更されました。**logging buffered**、**logging console**、 **logging discriminator**、 **logging host**、 **logging message-counter**、 **logging monitor**、 **show logging**。

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。