IP オプション、TCP フラグ、非隣接ポート、をフィルタする IP アクセス リストの作成

このモジュールは、特定の IP オプション、TCP フラグ、非隣接ポート、を含む IP パケットをフィルタする IP アクセス リストの使用方法について説明します。

IP オプション、TCP フラグ、非隣接ポートをフィルタする IP アクセス リストの作成に関する前提条件

このモジュールのいずれかのタスクを実行する前に、次のモジュールの情報を把握しておく必要があります。

- 『IP アクセス リストの概要』
- 『IP アクセス リストの作成とインターフェイスへの適用』

IP オプション、TCP フラグ、非隣接ポートをフィルタする IP アクセス リストの作成に関する情報

IP オプション

IP は、サービスを提供するときに、タイプ オブ サービス、存続可能時間、オプション、およびヘッダー チェックサムという 4 つの主要メカニズムを使用します。

オプションは一般的に IP オプションと呼ばれ、一部の状況で必要な制御機能のために用意されていますが、ほとんどの一般的な通信では不要です。IP オプションには、タイム スタンプ、セキュリティ、および特殊なルーティングに関する条件が含まれます。

IP オプションはデータグラムに含まれる場合と含まれない場合があります。IP オプションはすべての IP モジュール（ホストとゲートウェイ）で実装する必要があります。オプションというのは、実装ではなく、任意の指定したデータグラムでの送信を指します。環境によっては、セキュリティ オプションがすべてのデータグラムで必要です。

オプション フィールドは長さが可変です。オプションの個数はゼロ個以上です。IP オプションには、次の 2 つの形式のいずれかを使用できます。

- 形式 1：単一オクテットの option-type
- 形式 2：1 つの option-type オクテット、option-length オクテット、および実際の option-data オクテット

option-length オクテットは、option-type オクテット、option-length オクテット、および option-data オクテットの数をカウントします。

option-type オクテットには、1 ビットのコピー済みフラグ、2 ビットのオプション クラス、および 5 ビットのオプション番号という 3 つのフィールドがあります。これらのフィールドは、オプション タイプ フィールドの 8 ビット値を構成します。IP オプションは、一般的にその 8 ビット値で参照されます。

IP オプションの詳細な一覧と説明については、次の URL の RFC 791『*Internet Protocol*』を参照してください。http://www.faqs.org/rfcs/rfc791.html

IP オプションをフィルタする利点

- ネットワークからの IP オプションを含むパケットをフィルタすることで、ダウンストリームのデバイスとホストにかかるオプション パケットの負荷が軽減されます。
- また、この機能によって、分散型システムでルート プロセッサ（RP）処理が必要な IP オプションを含むパケットについて、RP への負荷が最小限になります。以前は、パケットは常に RP CPU でルーティングまたは処理されていました。パケットをフィルタすることで、パケットの RP への影響を回避できます。

TCP フラグに基づいてフィルタする利点

ACL TCP フラグ フィルタリング機能には、TCP フラグに基づいてフィルタする柔軟なメカニズムが用意されています。以前は、パケットのいずれかの TCP フラグがアクセス コントロールエントリ（ACE）で指定されたフラグに一致する限り、着信パケットは一致していました。すべてのフラグが設定されたパケットがアクセス コントロール リスト（ACL）を通過する可能性があるため、この動作ではセキュリティの抜け穴を考慮しています。ACL TCP フラグ フィルタリング機能では、フィルタするフラグの任意の組み合わせを選択できます。設定されているフラグ、および設定されていないフラグに基づいてマッチングする機能によって、TCP フラグに基づくフィルタリングの制御性が向上するため、セキュリティが強化されます。

TCP パケットは偽造の同期パケットとして送信され、それがリスニング ポートで受け入れられる可能性があるため、ファイアウォールデバイスの管理者は、偽造の TCP パケットをドロップするフィルタリング ルールを設定することを推奨します。

アクセス リストを構成する ACE を設定し、特定のグループの TCP フラグが設定されているパケットのみ、または設定されていないパケットのみを許可することで、不正な TCP パケットを検出およびドロップできます。ACL TCP フラグ フィルタリング機能によって、次のようにパケット フィルタリングの制御性が向上します。

- フィルタする TCP パケットについて、TCP フラグの任意の組み合わせを選択できます。
- 設定されているフラグと設定されていないフラグに基づいてマッチングできるように、ACE を設定できます。

TCP フラグ

次の表は TCP フラグの一覧です。詳細については、RFC 793『*Transmission Control Protocol*』を参照してください。

表 *1 : TCP* フラグ

TCP フラグ	目的
ACK	Acknowledge フラグ：セグメントの acknowledgment フィールドが、このセグメントの送信元が受信を予測している番号の次のシーケンス番号を指定することを示します。
FIN	Finish フラグ：接続をクリアするために使用されます。
PSH	Push フラグ：呼び出しのデータを受信ユーザーに対してただちにプッシュする必要があることを示します。
RST	Reset フラグ：受信者が以降のやり取りなしで接続を削除する必要があることを示します。

TCP フラグ	目的
SYN	Synchronize フラグ：接続の確立に使用されます。
URG	Urgent フラグ：urgent フィールドが重要で、セグメント シーケンス番号に追加する必要があることを示します。

アクセス コントロール エントリ機能での非隣接ポートに関する名前付き ACL サポートを使用する利点

この機能によって、同じ送信元アドレス、宛先アドレス、およびプロトコルに関して複数のエントリを処理するために、アクセス コントロール リストで必要なアクセス コントロール エントリ（ACE）の数が大幅に削減されます。大量の ACE を保守している場合、可能な限り、新しいアクセス リスト エントリを作成するときは、この機能を使用して既存のアクセス リスト エントリのグループを統合します。非隣接ポートを使用するアクセス リスト エントリを設定すると、保守するアクセス リスト エントリ数が少なくなります。

TTL 値のフィルタリング方法

IPは、拡張名前付きおよび番号付きアクセスリストは、インターフェイスを発着信するパケットの TTL 値でフィルタリングできます。有効な TTL 値 0 ～ 255 のパケットを許可または拒否できます（フィルタリング）。その他のフィールド（送信元または宛先アドレスなど）でのフィルタリングと同様に、**ip access-group** コマンドは **in** または **out** を指定します。これにより、アクセスリストの入力または出力が行われ、それぞれ着信または発信パケットに適用されます。TTL 値は、アクセス リスト エントリで指定したプロトコル、アプリケーション、およびその他の設定とともにチェックされ、すべての条件を満たす必要があります。

入力インターフェイスに到達した TTL 値 0 または 1 のパケットに対する特別な処理

分散型シスコ エクスプレス フォワーディング（dCEF）、CEF、ファスト スイッチング、プロセス スイッチングなどのソフトウェア スイッチング パスは、通常、アクセス リスト ステートメントに基づいてパケットを許可または廃棄します。ただし、入力インターフェイスに到達したパケットの TTL 値が 0 または 1 であるときには、特別な処理が必要です。TTL 値が 0 または 1 のパケットは、CEF、dCEF、またはファスト スイッチング パスで入力アクセス リストがチェックされる前に、プロセス レベルに送信されます。入力アクセス リストは、TTL 値が 2 ～ 255 であるパケットに適用され、許可または拒否の決定が行われます。

TTL 値が 0 または 1 のパケットは、デバイスから外部に転送されることがないため、プロセス レベルに送信されます。プロセス レベルでは、各パケットがそのデバイス宛であるかどうか、および Internet Control Message Protocol（ICMP）TTL 値期限切れメッセージを返送する必要があるかどうかをチェックする必要があります。つまり、TTL が 0 または 1 のパケットをドロップする意図で TTL 値 0 または 1 のフィルタリングを設定した ACL が入力インターフェイスで設定されている場合でも、高速なパスではパケットのドロップが発生しないということです。代わりに、プロセスが ACL を適用するときに、プロセス レベルで発生します。これはハード

ウェア スイッチング プラットフォームについてもあてはまります。TTL 値が 0 または 1 のパケットはルート プロセッサ（RP）またはマルチレイヤ スイッチ フィーチャ カード（MSFC）のプロセス レベルに送信されます。

出力インターフェイスでは、TTL 値でのアクセス リスト フィルタリングは、その他のアクセス リスト機能と同じように動作します。チェックはデバイスで有効な最も高速なスイッチング パスで行われます。これは、より高速なスイッチング パスは出力インターフェイスですべての TTL 値（0 ～ 255）を均等に処理するためです。

TTL 値 0 と 1 でフィルタリングするためのコントロール プレーン ポリシング

TTL 値が 0 または 1 のパケットに対する特別な動作によって、デバイスの CPU 使用率が高くなります。0 または 1 の TTL 値 でフィルタリングする場合は、CPU が過負荷になることを防ぐためにコントロール プレーン ポリシング（CPP）を使用してください。CPP を活用するには、TTL 値 0 および 1 をフィルタリングすることに特化したアクセス リストを設定し、CPP を通じてそのアクセス リストを適用する必要があります。このアクセス リストは、その他のインターフェイス アクセス リストとは別のアクセス リストにします。CPP は個々のインターフェイスにおいてではなくシステム全体に対して機能するため、そのようなアクセスリストはデバイス全体に対して 1 つのみ設定する必要があります。このタスクは、セクション「TTL 値 0 と 1 でフィルタリングするコントロールプレーン ポリシングの有効化」で説明しています。

TTL 値に基づいてフィルタする利点

- 存続可能時間（TTL）値でのフィルタリングは、デバイスに到達できるパケット、またはデバイスに到達できないパケットを制御する方法を提供します。ネットワーク レイアウトを確認することで、特定のデバイスからのパケットをホップ数に基づいて許可するか拒否するかを選択できます。たとえば、小規模ネットワークでは、ホップ数が 3 より大きい場所からのパケットを拒否する可能性があります。TTL 値でのフィルタリングでは、トラフィックがネイバーデバイスから発信されたかどうかを検証できます。たとえば特定プロトコルの初期 TTL 値より 1 小さい TTL 値のパケットのみを受け入れることで、1 ホップで自分に到達するパケットのみを受け入れることができます。
- 多くのコントロール プレーン プロトコルはネイバーのみと通信しますが、パケットを誰からも受信します。TTL でフィルタリングするアクセス リストを受信側ルータに適用すると、不要なパケットをブロックできます。
- Cisco ソフトウェアが送信するすべてのパケットは、プロセス レベルに対して TTL 値が 0 または 1 です。デバイスは、Internet Control Message Protocol（ICMP）TTL 値期限切れメッセージを送信元に送信する必要があります。TTL 値が 0 ～ 2 であるパケットをフィルタリングすることで、プロセス レベルでの負荷を削減できます。

IP オプション、TCP フラグ、非隣接ポートをフィルタする IP アクセス リストの作成方法

IP オプションを含むパケットのフィルタリング

アクセス リストを設定して、IP オプションを含むパケットをフィルタし、アクセス リストが適切に設定されていることを確認するには、次の手順を完了します。

（注）
- IP オプションのフィルタリングに関する ACL のサポート機能は、名前付きの拡張 ACL でのみ使用できます。
- この機能を設定する場合、リソース予約プロトコル（RSVP）マルチプロトコル ラベル スイッチング トラフィック エンジニアリング（MPLS TE）、Internet Group Management Protocol バージョン 2（IGMPV2）、および IP オプション パケットを使用するその他のプロトコルは、ドロップまたは無視モードでは機能しない可能性があります。
- ほとんどの Cisco デバイスでは、IP オプションを含むパケットはハードウェアではスイッチされませんが、処理するコントロール プレーン ソフトウェアが必要です（主に、オプションを処理し、IP ヘッダーを書き直す必要があるため）。結果として、IP オプションを含むすべての IP パケットは、ソフトウェアでフィルタとスイッチが行われます。

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

ステップ 2 **configure terminal**

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **ip access-list extended** *access-list-name*

例：

```
Device(config)# ip access-list extended mylist1
```

名前で IP アクセス リストを指定し、名前付きアクセス リストのコンフィギュレーション モードを開始します。

ステップ 4 [*sequence-number*] **deny** *protocol source source-wildcard destination destination-wildcard* [**option** *option-value*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# deny ip any any option traceroute
```

（任意）名前付き IP アクセス リスト モードで **deny** ステートメントを指定します。

- このアクセス リストでは **deny** ステートメントを最初に使用していますが、必要なステートメントの順序に応じて、**permit** ステートメントが最初に使用される可能性もあります。
- **option** キーワードおよび *option-value* 引数を使用して、特定の IP オプションを含むパケットをフィルタします。
- この例では、traceroute IP オプションを含むすべてのパケットが除外されます。
- エントリを削除するには、このコマンドの **no** *sequence-number* 形式を使用します。

ステップ 5 [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**option** *option-value*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# permit ip any any option security
```

名前付き IP アクセス リスト モードで **permit** ステートメントを指定します。

- この例では、セキュリティ IP オプションを含むすべてのパケット（まだフィルタされていないパケット）が許可されます。
- エントリを削除するには、このコマンドの **no** *sequence-number* 形式を使用します。

ステップ 6 必要に応じて、ステップ 4 またはステップ 5 を繰り返します。

アクセス リストは変更できます。

ステップ 7 **end**

例：

```
Device(config-ext-nacl)# end
```

（任意）名前付きアクセス リスト 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 **show ip access-lists** *access-list-name*

例：

```
Device# show ip access-lists mylist1
```

（任意）IP アクセス リストの内容を表示します。

次の作業

アクセス リストをインターフェイスに適用するか、アクセス リストを受け入れるコマンドから参照します。

（注） IP オプションを含むすべてのパケットを効率的に除去するには、**ip options drop** グローバル コマンドを設定することを推奨します。

TCP フラグを含むパケットのフィルタリング

この作業では、アクセス リストを設定して、TCP フラグを含むパケットをフィルタし、アクセス リストが適切に設定されていることを確認します。

（注）
- TCP フラグのフィルタリングを使用できるのは、名前付きの拡張 ACL のみです。
- ACL TCP フラグ フィルタリング機能は、Cisco ACL の場合にのみサポートされます。
- 事前に、次のコマンドライン インターフェイス（CLI）形式を使用して、TCP フラグチェック メカニズムを設定できます。

permit tcp any any rst 同じ ACE を示す次の形式を使用できるようになりました。 **permit tcp any any match-any +rst** いずれの CLI 形式も使用できますが、新しいキーワード **match-all** または **match-any** を選択する場合、プレフィックスに「+」または「-」を付けた新しいフラグを次に指定する必要があります。単一の ACL では、古い形式のみ、または新しい形式のみを使用することを推奨します。CLI の古い形式と新しい形式の混在やマッチングを行うことはできません。

注意 新しい構文形式の ACE を持つデバイスを、ACL TCP フラグ フィルタリング機能をサポートしないシスコ ソフトウェアの以前のバージョンでリロードすると、ACE は適用されないため、セキュリティの抜け穴が発生する可能性があります。

ステップ 1 enable

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

ステップ 2 configure terminal

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **ip access-list extended** *access-list-name*

例：

```
Device(config)# ip access-list extended kmd1
```

名前で IP アクセス リストを指定し、名前付きアクセス リストのコンフィギュレーション モードを開始します。

ステップ 4 [*sequence-number*] **permit tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established**|{**match-any** | **match-all**} {**+** | **-**} *flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# permit tcp any any match-any +rst
```

名前付き IP アクセス リスト モードで **permit** ステートメントを指定します。

- このアクセス リストでは **permit** ステートメントを最初に使用していますが、必要なステートメントの順序に応じて、**deny** ステートメントが最初に使用される可能性もあります。
- **permit** コマンドの TCP コマンド構文を使用します。
- RST TCP ヘッダー フラグが設定されたすべてのパケットは一致し、ステップ 3 で名前付きアクセス リスト kmd1 に合格できます。

ステップ 5 [*sequence-number*] **deny tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established**|{**match-any** | **match-all**} {**+** | **-**} *flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# deny tcp any any match-all -ack -fin
```

（任意）名前付き IP アクセス リスト モードで **deny** ステートメントを指定します。

- このアクセス リストでは **permit** ステートメントを最初に使用していますが、必要なステートメントの順序に応じて、**deny** ステートメントが最初に使用される可能性もあります。
- **deny** コマンドの TCP コマンド構文を使用します。
- ACK フラグが設定されず、FIN フラグも設定されていないパケットは、ステップ 3 で名前付きアクセス リスト kmd1 に合格しません。
- 上位層プロトコル（ICMP、IGMP、TCP、およびUDP）を許可するその他のコマンド構文については、**deny**（IP）コマンドを参照してください。

ステップ 6 必要に応じてステップ 4 またはステップ 5 を繰り返し、計画したシーケンス番号でステートメントを追加します。エントリを削除するには、**no** *sequence-number* コマンドを使用します。

アクセス リストは変更できます。

ステップ 7 **end**

例：

```
Device(config-ext-nacl)# end
```

（任意）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 **show ip access-lists** *access-list-name*

例：

```
Device# show ip access-lists kmd1
```

（任意）IP アクセス リストの内容を表示します。

- 出力を見直して、アクセス リストに新しいエントリが含まれることを確認します。

次の作業

アクセス リストをインターフェイスに適用するか、アクセス リストを受け入れるコマンドから参照します。

非隣接ポートを使用するアクセス コントロール エントリの設定

非隣接 TCP または UDP ポート番号を使用するアクセス リスト エントリを作成するには、次の作業を実行します。この作業では TCP ポートを使用しますが、**permit** および **deny** コマンドの UDP 構文を使用して、非隣接 UDP ポートをフィルタすることもできます。

この作業では **permit** コマンドを最初に使用していますが、フィルタリングの目標に合わせた順序で、**permit** および **deny** コマンドを使用できます。

（注） ACL：アクセス コントロール エントリでの非隣接ポートに関する名前付き ACL サポート機能を使用できるのは、名前付きの拡張 ACL のみです。

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

ステップ 2 **configure terminal**

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **ip access-list extended** *access-list-name*

例：

```
Device(config)# ip access-list extended acl-extd-1
```

名前で IP アクセス リストを指定し、名前付きアクセス リストのコンフィギュレーション モードを開始します。

ステップ 4 [*sequence-number*] **permit tcp** *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established** {**match-any** | **match-all**} {**+** | **-**} *flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# permit tcp any eq telnet ftp any eq 450 679
```

名前付き IP アクセス リスト コンフィギュレーション モードで **permit** ステートメントを指定します。

- 演算子には、**lt**（次の値より小さい）、**gt**（次の値より大きい）、**eq**（次の値に等しい）、**neq**（次の値に等しくない）**range**（次の範囲）があります。
- 演算子が source および source-wildcard 引数の後にある場合、送信元ポートに一致する必要があります。演算子が destination および destination-wildcard 引数の後にある場合、宛先ポートに一致する必要があります。
- **range** 演算子には 2 つのポート番号が必要です。**eq** および **neq** 演算子の後には、最大 10 個のポートを設定できます。他のすべての演算子は 1 つのポート番号が必要です。
- UDP ポートをフィルタするには、このコマンドの UDP 構文を使用します。

ステップ 5 [*sequence-number*] **deny tcp** *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established** {**match-any** | **match-all**} {**+** | **-**} *flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# deny tcp any neq 45 565 632 any
```

（任意）名前付きアクセス リスト コンフィギュレーション モードで **deny** ステートメントを指定します。

- 演算子には、**lt**（次の値より小さい）、**gt**（次の値より大きい）、**eq**（次の値に等しい）、**neq**（次の値に等しくない）**range**（次の範囲）があります。
- 演算子が *source* および *source-wildcard* 引数の後にある場合、送信元ポートに一致する必要があります。演算子が *destination* および *destination-wildcard* 引数の後にある場合、宛先ポートに一致する必要があります。

- **range** 演算子には 2 つのポート番号が必要です。**eq** および **neq** 演算子の後には、最大 10 個のポートを設定できます。他のすべての演算子は 1 つのポート番号が必要です。
- UDP ポートをフィルタするには、このコマンドの UDP 構文を使用します。

ステップ 6 必要に応じてステップ 4 またはステップ 5 を繰り返し、計画したシーケンス番号でステートメントを追加します。エントリを削除するには、**no** *sequence-number* コマンドを使用します。

アクセス リストは変更できます。

ステップ 7 **end**

例：

```
Device(config-ext-nacl)# end
```

（任意）名前付きアクセス リスト 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 **show ip access-lists** *access-list-name*

例：

```
Device# show ip access-lists kmd1
```

（任意）アクセス リストの内容を表示します。

非隣接ポートを使用する複数アクセス リスト エントリの1つのアクセス リスト エントリへの統合

非隣接ポートを使用するアクセス リスト エントリ グループを 1 つのアクセス リスト エントリに統合するには、次の作業を実行します。

この作業では TCP ポートを使用しますが、**permit** および **deny** コマンドの UDP 構文を使用して、非隣接 UDP ポートをフィルタすることもできます。

この作業では **permit** コマンドを最初に使用していますが、フィルタリングの目標に合わせた順序で、**permit** および **deny** コマンドを使用できます。

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

- パスワードを入力します（要求された場合）。

ステップ 2 **show ip access-lists** *access-list-name*

例：

```
Device# show ip access-lists mylist1
```

（任意）IP アクセス リストの内容を表示します。

- 出力を見直して、アクセス リスト エントリを統合できるかどうかを確認します。

ステップ 3 **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 4 **ip access-list extended** *access-list-name*

例：

```
Device(config)# ip access-list extended mylist1
```

名前で IP アクセス リストを指定し、名前付きアクセス リストのコンフィギュレーション モードを開始します。

ステップ 5 **no** [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard*[**option** *option-name*] [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# no 10
```

統合できる重複するアクセス リスト エントリを削除します。

- このステップを繰り返して、ポート番号のみが異なるために統合できるエントリを削除します。
- このステップを繰り返して、たとえばアクセス リスト エントリ 20、30、および 40 を削除した後は、1 つの **permit** ステートメントに統合されるため、これらのエントリは削除されます。
- *sequence-number* が指定された場合、その他のコマンド構文は任意です。

ステップ 6 [*sequence-number*] **permit** *protocol source source-wildcard*[*operator port*[*port*]] *destination destination-wildcard*[*operator port*[*port*]] [**option** *option-name*] [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

例：

```
Device(config-ext-nacl)# permit tcp any neq 45 565 632 any eq 23 45 34 43
```

名前付きアクセス リスト コンフィギュレーション モードで **permit** ステートメントを指定します。

- このインスタンスでは、非隣接ポートを使用するアクセス リスト エントリ グループは、1 つの **permit** ステートメントに統合されました。
- **eq** および **neq** 演算子の後には、最大 10 個のポートを設定できます。

ステップ 7 必要に応じてステップ 5 と 6 を繰り返し、**permit** または **deny** ステートメントを追加して、可能な場合はアクセス リスト エントリを統合します。エントリを削除するには、**no** *sequence-number* コマンドを使用します。

アクセス リストは変更できます。

ステップ 8 **end**

例：

```
Device(config-std-nacl)# end
```

（任意）名前付きアクセス リスト 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 9 **show ip access-lists** *access-list-name*

例：

```
Device# show ip access-lists mylist1
```

（任意）アクセス リストの内容を表示します。

次の作業

アクセス リストをインターフェイスに適用するか、アクセス リストを受け入れるコマンドから参照します。

TTL 値に基づいたパケットのフィルタリング

アクセス リストは柔軟性に優れているため、TTL 値に基づいてパケットをフィルタリングする **permit** と **deny** コマンドの組み合わせ 1 つだけでは定義することができません。次のタスクでは、TTL フィルタリングを実行する例を 1 つだけ示します。独自のフィルタリング プランを満たす **permit** と **deny** ステートメントを適切に設定します。

（注） デバイスで使用する Cisco のソフトウェア リリースに応じて、アクセス リストで演算子 EQ または NEQ を指定する場合、アクセス リストでは最大 10 個の TTL 値を指定できます。TTL 値の数は、シスコのソフトウェア リリースによって異なります。

手順の概要

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard*[**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]
5. **permit** または **deny** ステートメントを続けて追加し、必要なフィルタリングを実現します。
6. **exit**
7. **interface** *type number*
8. **ip access-group** *access-list-name* {**in** | **out**}

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	**ip access-list extended** *access-list-name* 例： `Device(config)# ip access-list extended ttlfilter`	IP アクセス リストを名前で定義します。 • TTL 値でフィルタリングするアクセス リストは、拡張アクセス リストである必要があります。
ステップ 4	[*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard*[**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**] 例： `Device(config-ext-nacl)# permit ip host 172.16.1.1 any ttl lt 2`	パケットが名前付き IP アクセス リストを通過できる条件を設定します。 • すべてのアクセス リストには、**permit** ステートメントが 1 つ以上必要です。 • この例では、送信元 172.16.1.1 から TTL 値が 2 未満の接続先へのパケットが許可されています。
ステップ 5	**permit** または **deny** ステートメントを続けて追加し、必要なフィルタリングを実現します。	--
ステップ 6	**exit** 例： `Device(config-ext-nacl)# exit`	コンフィギュレーションモードを終了して、コマンドライン インターフェイス（CLI）モード階層で次に高いレベルのモードを開始します。
ステップ 7	**interface** *type number* 例： `Device(config)# interface ethernet 0`	インターフェイスタイプを設定し、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 8	**ip access-group** *access-list-name* {**in** | **out**} 例： `Device(config-if)# ip access-group ttlfilter in`	アクセス リストをインターフェイスに適用します。

TTL 値 0 と 1 でフィルタリングするコントロール プレーン ポリシングの有効化

TTL 値 0 または 1 に基づいて IP パケットをフィルタリングしたり、CPU の過負荷を防止したりするには、次のタスクを実行します。このタスクでは、TTL 値 0 と 1 で分類用のアクセス リストを設定し、モジュラ QoS コマンドライン インターフェイス（CLI）（MQC）を設定して、ポリシーマップをコントロールプレーンに適用します。アクセスリストを通過するパケットはドロップされます。この特別なアクセス リストは、他のインターフェイス アクセス リストとは異なります。

アクセス リストは柔軟性に優れているため、TTL 値に基づいてパケットをフィルタリングする **permit** と **deny** コマンドの組み合わせ 1 つだけでは定義することができません。次のタスクでは、TTL フィルタリングを実行する例を 1 つだけ示します。独自のフィルタリング プランを満たす **permit** と **deny** ステートメントを適切に設定します。

手順の概要

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* **ttl** *operator* value
5. **permit** または **deny** ステートメントを続けて追加し、必要なフィルタリングを実現します。
6. **exit**
7. **class-map** *class-map-name* [**match-all** | **match-any**]
8. **match access-group** {*access-group* | **name** *access-group-name*}
9. **exit**
10. **policy-map** *policy-map-name*
11. **class** {*class-name* | **class-default**}
12. **drop**
13. **exit**
14. **exit**
15. **control-plane**
16. **service-policy** {**input** | **output**} *policy-map-name*

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）。

	コマンドまたはアクション	目的
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	**ip access-list extended** *access-list-name* 例： `Device(config)# ip access-list extended ttlfilter`	IP アクセス リストを名前で定義します。 • TTL 値でフィルタリングするアクセス リストは、拡張アクセス リストである必要があります。
ステップ 4	[*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* **ttl** *operator* value 例： `Device(config-ext-nacl)# permit ip host 172.16.1.1 any ttl lt 2`	パケットが名前付き IP アクセス リストを通過できる条件を設定します。 • すべてのアクセス リストには、**permit** ステートメントが 1 つ以上必要です。 • この例では、送信元 172.16.1.1 から TTL 値が 2 未満の接続先へのパケットが許可されています。
ステップ 5	**permit** または **deny** ステートメントを続けて追加し、必要なフィルタリングを実現します。	アクセス リストを通過するパケットはドロップされます。
ステップ 6	**exit** 例： `Device(config-ext-nacl)# exit`	コンフィギュレーション モードを終了して、CLI モード階層で次に高いレベルのモードを開始します。
ステップ 7	**class-map** *class-map-name* [**match-all** | **match-any**] 例： `Device(config)# class-map acl-filtering`	指定したクラスへのパケットのマッチングに使用するクラス マップを作成します。
ステップ 8	**match access-group** {*access-group* | **name** *access-group-name*} 例： `Device(config-cmap)# match access-group name ttlfilter`	指定したアクセス コントロール リストに基づいて、クラス マップの一致基準を設定します
ステップ 9	**exit** 例： `Device(config-cmap)# exit`	コンフィギュレーション モードを終了して、CLI モード階層で次に高いレベルのモードを開始します。

	コマンドまたはアクション	目的
ステップ 10	**policy-map** *policy-map-name* 例： `Device(config)# policy-map acl-filter`	1 つ以上のインターフェイスに付加できるポリシーマップを作成または変更し、サービス ポリシーを指定します。
ステップ 11	**class** {*class-name* | **class-default**} 例： `Device(config-pmap)# class acl-filter-class`	作成または変更するポリシーのクラス名を指定するか、ポリシーを指定する前にデフォルト クラス（一般に class-default クラスといいます）を指定します。
ステップ 12	**drop** 例： `Device(config-pmap-c)# drop`	特定のクラスに属するパケットを廃棄するトラフィック クラスを設定します。
ステップ 13	**exit** 例： `Device(config-pmap-c)# exit`	コンフィギュレーション モードを終了して、CLI モード階層で次に高いレベルのモードを開始します。
ステップ 14	**exit** 例： `Device(config-pmap)# exit`	コンフィギュレーション モードを終了して、CLI モード階層で次に高いレベルのモードを開始します。
ステップ 15	**control-plane** 例： `Device(config)# control-plane`	デバイスのコントロール プレーンに関連する属性またはパラメータを関連付けたり、変更したりします。
ステップ 16	**service-policy** {**input** | **output**} *policy-map-name* 例： `Device(config-cp)# service-policy input acl-filter`	集約コントロール プレーン サービスのためにポリシーマップをコントロールプレーンに適用します。

IP オプション、TCP フラグ、非隣接ポートのフィルタリングの設定例

例：IP オプションを含むパケットのフィルタリング

次の例は、アクセス リスト エントリ（ACE）に指定されている IP オプションが含まれる場合にのみ、TCP パケットを許可するように設定された ACE を含む、mylist2 という拡張アクセス リストを示します。

```
ip access-list extended mylist2
 10 permit ip any any option eool
 20 permit ip any any option record-route
 30 permit ip any any option zsu
 40 permit ip any any option mtup
```

一致し、それによって許可されたパケットの数を示すため、**show access-list** コマンドが入力されました。

```
Device# show ip access-list mylist2
Extended IP access list test
10 permit ip any any option eool (1 match)
20 permit ip any any option record-route (1 match)
30 permit ip any any option zsu (1 match)
40 permit ip any any option mtup (1 match)
```

例：TCP フラグを含むパケットのフィルタリング

次のアクセス リストでは、TCP フラグ ACK および SYN が設定され、FIN フラグが設定されていない場合にのみ、TCP パケットを許可します。

```
ip access-list extended aaa
 permit tcp any any match-all +ack +syn -fin
 end
```

show access-list コマンドは、ACL を表示するために入力しました。

```
Device# show access-list aaa

Extended IP access list aaa
 10 permit tcp any any match-all +ack +syn -fin
```

例：非隣接ポートを使用するアクセス リスト エントリの作成

eq および **neq** 演算子の後に最大 10 ポートを入力できるため、次のアクセス リスト エントリを作成できます。

```
ip access-list extended aaa
```

```
 permit tcp any eq telnet ftp any eq 23 45 34
 end
```

show access-lists コマンドを入力して、新しく作成されたアクセス リスト エントリを表示します。

```
Device# show access-lists aaa

Extended IP access list aaa
 10 permit tcp any eq telnet ftp any eq 23 45 34
```

例：既存の複数のアクセス リスト エントリと非隣接ポートを使用する 1 つのアクセス リスト エントリの統合

show access-lists コマンドは、abc というアクセス リストについて、アクセス リスト エントリ グループを表示するために使用されます。

```
Device# show access-lists abc
Extended IP access list abc
 10 permit tcp any eq telnet any eq 450
 20 permit tcp any eq telnet any eq 679
 30 permit tcp any eq ftp any eq 450
 40 permit tcp any eq ftp any eq 679
```

エントリはすべて同じ **permit** ステートメント用であり、ポートのみが異なるため、1 つの新しいアクセス リスト エントリに統合できます。次の例では、重複するアクセス リスト エントリを削除し、以前に表示されていたアクセス リスト エントリ グループを統合する新しいアクセス リスト エントリを作成します。

```
ip access-list extended abc
 no 10
 no 20
 no 30
 no 40
 permit tcp any eq telnet ftp any eq 450 679
 end
```

show access-lists コマンドを再入力すると、統合されたアクセス リスト エントリが表示されます。

```
Device# show access-lists abc
Extended IP access list abc
 10 permit tcp any eq telnet ftp any eq 450 679
```

例：TTL 値のフィルタリング

次のアクセス リストは、存続可能時間（TTL）の値が 10 と 20 でタイプ オブ サービス（ToS）レベルが 3 の IP パケットをフィルタリングします。また、TTL が 154 を超える IP パケットをフィルタリングし、その規則を先頭以外のフラグメントにも適用します。フラッシュの優先レベルと 1 以外の TTL 値を持つ IP パケットを許可し、そのようなパケットのログ メッセージをコンソールに送信します。他のすべてのパケットは拒否されます。

```
ip access-list extended incomingfilter
 deny ip any any tos 3 ttl eq 10 20
 deny ip any any ttl gt 154 fragments
 permit ip any any precedence flash ttl neq 1 log
!
interface ethernet 0

ip access-group incomingfilter in
```

例：TTL 値 0 と 1 でフィルタリングするコントロール プレーン ポリシング

次の例では、acl-filter と呼ばれるポリシー マップで使用するために、acl-filter-class と呼ばれるトラフィック クラスを設定します。アクセス リストは、存続可能時間（TTL）値が 0 または 1 の送信元からの IP パケットを許可します。アクセス リストに一致するパケットがドロップされます。ポリシー マップはコントロール プレーンに結合されます。

```
ip access-list extended ttlfilter

 permit ip any any ttl eq 0 1

class-map acl-filter-class

 match access-group name ttlfilter

policy-map acl-filter

 class acl-filter-class

 drop

control-plane

 service-policy input acl-filter
```

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
セキュリティ コマンド	『*Cisco IOS Security Command Reference*』
no ip optionsコマンドを使用した、IP オプションを含むパケットをドロップまたは無視するためのデバイスの設定。	*ACL IP* オプションの選択的ドロップ

関連項目	マニュアル タイトル
アクセス リストに関する概要情報	*IP* アクセス リストの概要
IP アクセス リストの作成とインターフェイスへの適用に関する情報	*IP* アクセス リストの作成とインターフェイスへの適用
QoS コマンド	『*Cisco IOS Quality of Service Solutions Command Reference*』

RFC

RFC	タイトル
RFC 791	*Internet Protocol*（インターネット プロトコル） http://www.faqs.org/rfcs/rfc791.html
RFC 793	伝送制御プロトコル（*TCP*）
RFC 1393	『*Traceroute Using an IP Option*』

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

フィルタするための IP アクセス リストの作成に関する機能情報

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。

表 *2:* フィルタするための *IP* アクセス リストの作成に関する機能情報

機能名	リリース	機能の設定情報
ACL -- アクセス コントロールエントリでの非隣接ポートに関する名前付き ACL サポート	12.3(7)T 12.2(25)S	この機能を使用すると、1 つのアクセス コントロール エントリで非隣接ポートを指定できるため、複数のエントリが同じ送信元アドレス、宛先アドレス、およびプロトコルを持ち、ポートのみが異なる場合に、アクセス コントロール リストで必要なエントリ数を大幅に減らすことができます。
IP オプションのフィルタリングに関する ACL のサポート	12.3(4)T 12.2(25)S 15.2(2)S 15.4(1)S	この機能を使用すると、IP オプションを含むパケットをフィルタできます。その結果、ルータが偽造パケットで飽和状態にならないように防ぎます。 Cisco IOS リリース 15.4(1)S では、Cisco ASR 901S ルータのサポートが追加されました。
ACL TCP フラグ フィルタリング	12.3(4)T 12.2(25)S	この機能は、TCP フラグに基づくフィルタリングに柔軟なメカニズムを提供します。Cisco IOS リリース 12.3(4)T 以前は、パケット内のいずれかの TCP フラグがアクセス コントロール エントリ（ACE）で指定されたフラグに一致する限り、着信パケットは一致していました。すべてのフラグが設定されたパケットがアクセスコントロールリスト（ACL）を通過する可能性があるため、この動作ではセキュリティの抜け穴を考慮しています。ACL TCP フラグ フィルタリング機能では、フィルタするフラグの任意の組み合わせを選択できます。設定されているフラグ、および設定されていないフラグに基づいてマッチングする機能によって、TCP フラグに基づくフィルタリングの制御性が向上するため、セキュリティが強化されます。

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。