アクセス リストの概要

アクセス リストは、パケット フィルタリングを実行して、ネットワークを介して移動するパケットとその場所を制御します。この処理は、ネットワーク トラフィックを制限したり、ユーザやデバイスによるネットワークへのアクセスを制限したりするのに役立ちます。アクセス リストの用途は多様なので、多くのコマンドの構文でアクセス リストが参照されます。アクセス リストを使用して、次のようなことを実行できます。

アクセス コントロール リスト（ACL）は、ネットワーク トラフィック プロファイルをまとめて定義する 1 つ以上のアクセス コントロール エントリ（ACE）です。このプロファイルは、トラフィック フィルタリング、ルート フィルタリング、QoS 分類、アクセス コントロールなど、Cisco IOS XR ソフトウェアの機能で参照できます。次の 2 種類の ACL があります。

- 標準的な ACL：パケットの送信元 IP アドレスのみを確認します。トラフィックは、ACL に設定されたアドレスまたはプレフィックスの比較と、パケットにある送信元アドレスで制御されます。
- 拡張ACL：パケットの発信元アドレス以外の属性も確認します。確認する属性は、送信先アドレス、特定の IP プロトコル、UDP または TCP ポート番号、DSCP などです。ACL に記載されている属性と、着信パケットまたは発信パケット内の属性を比較することで、トラフィックが制御されます。

Cisco IOS XR は標準アクセス リストと拡張アクセス リストとを区別しません。標準アクセス リストをサポートしているのは、下位互換性を確保するためです。

IP アクセス リストの目的

- インターフェイスで着信パケットまたは発信パケットをフィルタリングします。
- ミラーリングのためにパケットをフィルタリングします。
- 必要に応じて、トラフィックをリダイレクトします。
- ルーティング アップデートの内容の制限
- アドレスまたはプロトコルに基づくデバッグ出力の制限
- vty へのアクセスの制御

- 輻輳回避、輻輳管理、プライオリティ キューイング、カスタム キューイングなどの高度な機能に使用されるトラフィックの特定または分類

IP アクセス リストの機能

アクセス リストは、permit ステートメントと deny ステートメントで構成される順次リストです。これらのステートメントは、IP アドレス、場合によっては上位層 IP プロトコルに適用されます。アクセス リストには、参照に使用される名前があります。多くのソフトウェア コマンドは、構文の一部としてアクセス リストを受け取ります。

アクセス リストを設定して名前を付けることは可能ですが、アクセス リストを受け取るコマンドによってアクセスリストが参照されるまで、有効にはなりません。複数のコマンドから同じアクセス リストを参照できます。アクセス リストで、ルータに到達するトラフィック、またはルータ経由で送信されるトラフィックは制御できますが、ルータが送信元のトラフィックは制御できません。

送信元アドレスと宛先アドレスは、IP パケットの最も一般的な 2 つのフィールドで、アクセスリストの基礎となります。送信元アドレスを指定して、特定のネットワーキング デバイスまたはホストからのパケットを制御します。宛先アドレスを指定して、特定のネットワーキング デバイスまたはホストに送信されるパケットを制御します。

また、トランスポート層の情報（パケットが TCP、UDP、ICMP、IGMP のいずれであるかなどの情報）に基づいてパケットをフィルタリングすることもできます。

ACL のワークフロー

次の図に、ACL のワークフローを示します。

IP アクセス リストのプロセスとルール

IP アクセス リストを設定するときは、次のプロセスとルールを使用してください。

- アクセスリストの条件に対してフィルタリングされる各パケットの送信元アドレスや宛先アドレス、またはプロトコルがテストされます。一度に 1 つの条件（permit ステートメントまたは deny ステートメント）がテストされます。
- パケットがアクセスリストのステートメントに一致しないと、そのパケットはリスト内の次のステートメントに対してテストされます。
- パケットとアクセスリストのステートメントが一致すると、リスト内の残りのステートメントはスキップされ、パケットは一致したステートメントに指定されたとおりに許可または拒否されます。パケットが許可されるか拒否されるかは、パケットが一致する最初のエントリによって決まります。つまり、一致すると、それ以降のエントリは考慮されません。
- アクセスリストでアドレスまたはプロトコルが拒否されると、パケットは廃棄され、インターネット制御メッセージプロトコル（ICMP）ホスト到達不能メッセージが返されます。ICMP は、Cisco IOS XR ソフトウェアで設定できます。

- 各アクセス リストの最後には暗黙の deny ステートメントがあるため、一致する条件がない場合は、パケットはドロップされます。つまり、各ステートメントに対してテストするときまでにパケットを許可または拒否しないと、パケットは拒否されます。
- アクセス リストには permit ステートメントを 1 つ以上含める必要があります。そうしないと、パケットはすべて拒否されます。
- 最初に一致が見つかった後は条件のテストが終了するため、条件の順序は重要です。同じ permit ステートメントまたは deny ステートメントでも、順序が異なる場合、ある状況では通過し、別の状況では拒否されるパケットが生じる可能性があります。
- 1 つのインターフェイス、1 つのプロトコル、1 つの方向につき、許可されるアクセス リストは 1 つだけです。
- インバウンド アクセス リストは、ルータに到達するパケットを処理します。着信パケットの処理後に、アウトバウンドインターフェイスへのルーティングが行われます。インバウンドアクセスリストが効率的なのは、フィルタリングテストで拒否されたことでパケットが廃棄される場合、ルーティング検索のオーバーヘッドが抑えられるためです。パケットがテストで許可されると、そのパケットに対してルーティングの処理が実施されます。インバウンドリストの場合、許可とは、インバウンドインターフェイスで受信したパケットを引き続き処理することを意味します。**deny**とは、パケットを破棄することです。
- アウトバウンド アクセス リストの場合、パケットの処理後にルータから送信されます。着信パケットはアウトバウンドインターフェイスにルーティングされてから、アウトバウンドアクセスリストで処理されます。アウトバウンドリストの場合、許可とは、出力バッファに対して送信されることを示し、拒否とは、パケットが廃棄されることを示します。
- アクセス リストは、使用中のアクセス グループによって適用されている場合には削除できません。アクセス リストを削除するには、まずアクセス リストを参照しているアクセス グループを削除してから、アクセス リストを削除します。
- 特定のトラフィックを拒否するACLで設定されているインターフェイスを削除する前に、そのACLを削除し、設定をコミットする必要があります。これを実行しないと、**no interface <interface-name>** コマンドが設定され、コミットされるとすぐにインターフェイスを通じて一部のパケットがリークします。
- **ipv4 access group** コマンドを使用するには、アクセス リストが必要です。

ワイルドカード マスクと暗黙的なワイルドカード マスクを使用した ACL フィルタリング

アドレス フィルタリングでは、アクセス リスト エントリ内のアドレス ビットとアクセス リストに送信されるパケットを比較するときに、ワイルドカード マスクを使用して、対応する IP アドレス ビットを確認するか無視するかを指定します。管理者は、ワイルドカード マスクを慎重に設定することにより、許可または拒否のテストに 1 つまたは複数の IP アドレスを選択できます。

IP アドレス ビット用のワイルドカード マスクでは、数値 1 と数値 0 を使用して、対応する IP アドレス ビットをどのように扱うかを指定します。1 と 0 は、サブネット（ネットワーク）マスクで意味する内容が逆になるため、ワイルドカード マスクは逆マスクとも呼ばれます。

- ワイルドカード マスク ビット 0 は、対応するビット値を確認することを示します。
- ワイルドカード マスクのビット 1 は、対応するビット値を無視することを意味します。

アクセス リスト ステートメントでは、送信元アドレスまたは宛先アドレスにワイルドカード マスクを指定する必要はありません。**host** キーワードを使用すると、ワイルドカード マスクとして 0.0.0.0 を指定したものと見なされます。

サブネットマスクでは、ネットワークとサブネットを示す隣接ビットをマスクにする必要がありますが、それとは異なり、ワイルドカード マスクではマスクに非隣接ビットを使用できます。

ワイルドカード ビットの代わりに、CIDR 形式（/x）を使用することもできます。たとえば、IPv4 アドレス 1.2.3.4 0.255.255.255 は 1.2.3.4/8 に相当し、IPv6 アドレスの場合、2001:db8:abcd:0012:0000:0000:0000:0000 は 2001:db8:abcd:0012::0/64 に相当します。

アクセス リストのコメントの組み入れ

remark アクセス リスト コンフィギュレーション コマンドを使用すると、名前付き IP アクセス リストにエントリに関するコメント（注釈）を含めることができます。コメントを含めると、ネットワーク管理者がアクセス リストを理解し、精査しやすくなります。1 つのコメント行の最大長は 255 文字です。

コメントは、**permit** ステートメントまたは **deny** ステートメントの前後どちらにでも配置できます。コメントがどの **permit** ステートメントまたは **deny** ステートメントの説明であるのかが明確になるように、コメントの位置に関して一貫性を保つようにしてください。たとえば、一部のコメントが **permit** または **deny** ステートメントの前にあり、他のコメントがステートメントの後ろにあると、混乱を招きます。コメントに順番を付けることができます。

アクセス リストの作成後、アクセス リストをインターフェイスまたは端末回線に適用することを忘れないでください。

- IPv4 ACL の設定 （5 ページ）
- IPv6 ACL の設定 （8 ページ）
- ACL の変更 （12 ページ）
- ACL ベースの転送の設定 （13 ページ）
- ブリッジ仮想インターフェイスの ACL （15 ページ）
- フラグメント制御を使用した ACL の設定 （19 ページ）
- IP パケット長による ACL フィルタリングの設定 （24 ページ）
- オブジェクトグループ ACL の概要 （28 ページ）
- IPv4 ACL での TTL の照合の設定 （32 ページ）
- IPv6 ACL の TTL の照合の設定 （34 ページ）
- IP アクセス リスト ロギング メッセージの概要 （35 ページ）
- プレフィックス リストの概要 （36 ページ）
- プレフィックス リストの設定 （37 ページ）
- プレフィックス リスト エントリの順序付けとプレフィックス リストの変更 （38 ページ）

IPv4 ACL の設定

この項では、IPv4 の入力 ACL と出力 ACL の基本的な設定について説明します。

IPv4 入力 ACL を設定するための注意事項と制約事項

IPv4 入力 ACL は、次の動作を特徴としています。

- 入力 IPv4 ACL はすべてのインターフェイスでサポートされています。
- ACL ベースの転送（ABF）は、入力方向でのみサポートされています。
- NPU ごとにデフォルトで許可されている ACL の総数は 31 です。
- ライン カードごとに許可されている付加された ACE の数は 4,000 です。
- パケット長（**pkt-length** キーワードを使用）は入力 IPv4 ACL のみでサポートされています。**pkt-length** フィルタリング値は、デフォルトでは 16 バイトの増分のみで指定できます。
- 入力インターフェイスによる ACL ロギング（**log-input** キーワードを使用）はサポートされていません。

IPv4 出力 ACL を設定するための注意事項と制約事項

IPv4 出力 ACL は、次の動作を特徴としています。

- 出力 IPv4 ACL は、メインの物理インターフェイスとバンドル インターフェイスでサポートされています。

（注） 出力 ACL は、サブインターフェイスでは直接サポートされていません。ただし、サブインターフェイスがあるメインのインターフェイス上に出力 ACL を設定した場合、ACL アクションはサブインターフェイス トラフィックにも適用されます。この出力ACLの動作は、メインのインターフェイスに ACL を適用した後でサブインターフェイスを設定した場合も同じです。

- NPU ごとに許可された出力 ACL の総数は 255 です。
- ライン カードごとに許可されている付加された ACE の数は 4,000 です。
- ACL ロギング（**log** コマンドを使用）と入力インターフェイスによる ACL ロギング（**log-input** コマンドを使用）はサポートされていません。

ギガビット イーサネット インターフェイス上での入力 IPv4 ACL の設定

GigE インターフェイス上で入力 IPv4 ACL を設定するには、次の設定を使用します。

```
/* Configure a GigE interface with an IPv4 address */
Router(config)# interface gigabitEthernet 0/0/0/0
Router(config-if)# ipv4 address 10.1.1.1 255.255.255.0
Router(config-if)# no shut
Router(config-if)# commit
Thu Jan 25 10:07:54.700 IST
Router(config-if)# exit

/* Verify if the interface is up */
Router(config)# do show ipv4 interface brief
Thu Jan 25 10:08:49.087 IST

Interface                      IP-Address      Status          Protocol Vrf-Name
GigabitEthernet0/0/0/0         10.1.1.1        Up              Up       default

/* Configure an IPv4 ingress ACL */
Router(config)# ipv4 access-list V4-ACL-INGRESS
Router(config-ipv4-acl)# 10 permit tcp 10.2.1.1 0.0.0.255 any
Router(config-ipv4-acl)# 20 deny udp any any
Router(config-ipv4-acl)# 30 permit ipv4 10.2.0.0 0.255.255.255 any
Router(config-ipv4-acl)# commit
Thu Jan 25 10:16:11.473 IST

/* Verify the ingress ACL creation */
Router(config)# do show access-lists ipv4
Thu Jan 25 10:25:19.896 IST
...
ipv4 access-list V4-ACL-INGRESS
 10 permit tcp 10.2.1.0 0.0.0.255 any
 20 deny udp any any
 30 permit ipv4 10.0.0.0 0.255.255.255 any


/* Apply the ingress ACL to the GigE interface */
Router(config)# interface GigabitEthernet 0/0/0/0
Router(config-if)# ipv4 access-group V4-ACL-INGRESS ingress
Router(config-if)# commit
Thu Jan 25 10:28:19.671 IST
Router(config-if)# exit

/* Verify if the ingress ACL has been successfully applied to the interface */
Router(config)# do show ipv4 interface
Thu Jan 25 10:29:44.944 IST
GigabitEthernet0/0/0/0 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is 10.1.1.1/24
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  common access list is not set, access list is V4-ACL-INGRESS
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000
```

ギガビット イーサネット インターフェイス上に IPv4 入力 ACL を正常に設定しました。

ギガビット イーサネット インターフェイス上での出力 IPv4 ACL の設定

GigE インターフェイス上で出力 IPv4 ACL を設定するには、次の設定を使用します。

```
/* Configure a GigE interface with an IPv4 address */
Router(config)# interface gigabitEthernet 0/0/0/1
Router(config-if)# ipv4 address 20.1.1.1 255.255.255.0
Router(config-if)# no shut
Router(config-if)# commit
Thu Jan 25 10:08:38.767 IST
Router(config-if)# exit

/* Verify if the interface is up */
Router(config)# do show ipv4 interface brief
Thu Jan 25 10:08:49.087 IST

Interface                      IP-Address      Status          Protocol Vrf-Name
GigabitEthernet0/0/0/0         10.1.1.1        Up              Up       default
GigabitEthernet0/0/0/1         20.1.1.1        Up              Up       default

/* Configure an IPv4 egress ACL */
Router(config)# ipv4 access-list V4-ACL-EGRESS
Router(config-ipv4-acl)# 10 permit ipv4 10.2.0.0 0.255.255.255 20.2.0.0 0.255.255.255
Router(config-ipv4-acl)# 20 deny ipv4 any any
Router(config-ipv4-acl)# commit
Thu Jan 25 10:25:04.655 IST

/* Verify the egress ACL creation */
Router(config)# do show access-lists ipv4
Thu Jan 25 10:25:19.896 IST
ipv4 access-list V4-ACL-EGRESS
 10 permit ipv4 10.0.0.0 0.255.255.255 20.0.0.0 0.255.255.255
 20 deny ipv4 any any
...

/* Apply the egress ACL to the GigE interface */
Router(config)# interface GigabitEthernet 0/0/0/1
Router(config-if)# ipv4 access-group V4-ACL-EGRESS egress
Router(config-if)# commit
Thu Jan 25 10:28:45.937 IST
Router(config-if)# exit

/* Verify if the egress ACL has been successfully applied to the interface */
Router(config)# do show ipv4 interface
Thu Jan 25 10:29:44.944 IST
GigabitEthernet0/0/0/1 is Up, ipv4 protocol is Up
  Vrf is default (vrfid 0x60000000)
  Internet address is 20.1.1.1/24
  MTU is 1514 (1500 is available to IP)
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is V4-ACL-EGRESS
  Inbound  common access list is not set, access list is not set
  Proxy ARP is disabled
  ICMP redirects are never sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  Table Id is 0xe0000000
...
```

ギガビット イーサネット インターフェイス上に IPv4 出力 ACL を正常に設定しました。

IPv6 ACL の設定

この項では、ギガビット イーサネットとバンドル イーサネットを介して入力 IPv6 ACL と出力 IPv6 ACL を設定するステップについて説明します。

IPv6 入力 ACL を設定するための注意事項と制約事項

IPv6 入力 ACL は、次の動作を特徴としています。

- 入力 IPv6 ACL はすべてのインターフェイスでサポートされています。
- ACL ベースの転送（ABF）は、入力方向でのみサポートされています。
- NPU ごとに許可された入力 ACL の総数は 31 です。
- ライン カードごとに許可されている付加された ACE の数は 2047 です。
- 入力インターフェイスによる ACL ロギング（**log-input** キーワードを使用）はサポートされていません。
- パケット長（**pkt-length** キーワードを使用）はサポートされていません。

ギガビット イーサネット インターフェイス上での入力 IPv6 ACL の設定

GigE インターフェイス上で入力 IPv6 ACL を設定するには、次の設定を使用します。

```
/* Configure a GigE interface with an IPv6 address */
Router(config)# interface gigabitEthernet 0/0/0/0
Router(config-if)# ipv6 address 1001::1/64
Router(config-if)# no shut
Router(config-if)# commit
Thu Jan 25 10:07:54.700 IST
Router(config-if)# exit

/* Verify if the interface is up */
Router(config)# do show ipv6 interface brief
Thu Jan 25 12:38:35.742 IST
GigabitEthernet0/0/0/0 [Up/Up]
    fe80::bd:b9ff:fea9:5606
    1001::1
…

/* Configure an IPv6 ingress ACL */
Router(config)# ipv6 access-list V6-INGRESS-ACL
Router(config-ipv6-acl)# 10 permit ipv6 any any
Router(config-ipv6-acl)# 20 deny udp any any
Router(config-ipv6-acl)# commit
Thu Jan 25 11:31:24.488 IST
Router(config-ipv6-acl)# exit


/* Verify the ingress ACL creation */
Router(config)# do show access-lists ipv6
Thu Jan 25 11:34:56.911 IST
ipv6 access-list V6-INGRESS-ACL
 10 permit ipv6 any any
```

```
 20 deny udp any any


/* Apply the ingress ACL to the GigE interface */
Router(config)# interface gigabitEthernet 0/0/0/0
Router(config-if)# ipv6 access-group V6-INGRESS-ACL ingress
Router(config-if)# commit
Thu Jan 25 11:32:55.194 IST
Router(config-if)# exit

/* Verify if the ingress ACL has been successfully applied to the interface */
Router(config)# do show ipv6 interface
Thu Jan 25 11:34:08.028 IST
GigabitEthernet0/0/0/0 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::bd:b9ff:fea9:5606
  Global unicast address(es):
    1001::1, subnet is 1001::/64
  Joined group address(es): ff02::1:ff00:1 ff02::1:ffa9:5606 ff02::2
      ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is not set
  Inbound  common access list is not set, access list is V6-INGRESS-ACL
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
…
```

ギガビット イーサネット インターフェイス上に IPv6 入力 ACL を正常に設定しました。

ギガビット イーサネット インターフェイス上での出力 IPv6 ACL の設定

GigE インターフェイス上で出力 IPv6 ACL を設定するには、次の設定を使用します。

```
/* Configure a GigE interface with an IPv6 address */
Router(config)# interface GigabitEthernet 0/0/0/1
Router(config-if)# ipv6 address 2001::1/64
Router(config-if)# no shut
Router(config-if)# commit
Thu Jan 25 11:41:25.778 IST
Router(config-if)# exit

/* Verify if the interface is up */
Router(config)# do show ipv6 interface brief
Thu Jan 25 12:38:35.742 IST
GigabitEthernet0/0/0/0 [Up/Up]
    fe80::bd:b9ff:fea9:5606
    1001::1
GigabitEthernet0/0/0/1 [Up/Up]
    fe80::23:e9ff:fea8:a44e
    2001::1
```

```
/* Configure an IPv6 egress ACL */
Router(config)# ipv6 access-list V6-EGRESS-ACL
Router(config-ipv6-acl)# 10 permit ipv6 any any
Router(config-ipv6-acl)# 20 deny udp any any
Router(config-ipv6-acl)# commit
Thu Jan 25 11:44:03.969 IST
Router(config-ipv6-acl)# exit

/* Verify the egress ACL creation */
Router(config)# do show access-lists ipv6
Thu Jan 25 11:45:53.823 IST
ipv6 access-list V6-EGRESS-ACL
 10 permit ipv6 any any
 20 deny udp any any
…

/* Apply the egress ACL to the GigE interface */
Router(config)# interface gigabitEthernet 0/0/0/1
Router(config-if)# ipv6 access-group V6-EGRESS-ACL egress
Router(config-if)# commit
Thu Jan 25 11:45:12.682 IST
Router(config-if)# exit

/* Verify if the egress ACL has been successfully applied to the interface */
Router(config)# do show ipv6 interface
Thu Jan 25 11:46:43.234 IST
…
GigabitEthernet0/0/0/1 is Up, ipv6 protocol is Up, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::23:e9ff:fea8:a44e
  Global unicast address(es):
    2001::1, subnet is 2001::/64
  Joined group address(es): ff02::1:ff00:1 ff02::1:ffa8:a44e ff02::2
      ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
   Hosts use stateless autoconfig for addresses.
  Outgoing access list is V6-EGRESS-ACL
  Inbound  common access list is not set, access list is not set
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
...
```

ギガビット イーサネット インターフェイス上に IPv6 出力 ACL を正常に設定しました。

バンドル インターフェイス上での入力 IPv6 ACL と出力 IPv6 ACL の設定

バンドル インターフェイス上で入力 IPv6 ACL と出力 IPv6 ACL を設定するには、次の設定を使用します。

```
/* Configure a bundle interface with an IPv6 address */
Router(config)# interface Bundle-Ether 1
```

```
Router(config-if)# ipv6 address 3001::1/64
Router(config-if)# no shut
Router(config-if)# commit
Thu Jan 25 13:53:47.435 IST
Router(config-if)# exit

/* Configure an IPv6 egress ACL */
Router(config)# ipv6 access-list V6-EGRESS-ACL-bundle interface
Router(config-ipv6-acl)# 10 permit tcp any any range 3000 4000
Router(config-ipv6-acl)# 20 permit ipv6 any any
Router(config-ipv6-acl)# commit
Thu Jan 25 13:57:14.960 IST
Router(config-ipv6-acl)# exit

/* Configure an IPv6 ingress ACL to deny ingress traffic on the bundle interface */
Router(config)# ipv6 access-list V6-DENY-INGRESS-ACL
Router(config-ipv6-acl)# 10 deny ipv6 any any
Router(config-ipv6-acl)# commit
Thu Jan 25 13:59:23.198 IST
Router(config-ipv6-acl)# exit

/* Verify the egress and ingress ACL creation */
Router(config)# do show access-lists ipv6
Thu Jan 25 14:00:24.055 IST
ipv6 access-list V6-DENY-INGRESS-ACL
 10 deny ipv6 any any
ipv6 access-list V6-EGRESS-ACL-BI
 10 permit tcp any any range 3000 4000
 20 permit ipv6 any any
...

/* Apply the egress and ingress ACLs to the bundle interface */
Router(config)# interface Bundle-Ether 1
Router(config-if)# ipv6 access-group V6-EGRESS-ACL-BI egress
Router(config-if)# ipv6 access-group V6-DENY-INGRESS-ACL ingress
Router(config-if)# commit
Thu Jan 25 14:04:19.536 IST
Router(config-if)# exit

/* Verify if the ACLs have been successfully applied to the interface */
Router(config)# do show ipv6 interface
Thu Jan 25 11:46:43.234 IST
…
Thu Jan 25 14:04:51.322 IST
Bundle-Ether1 is Down, ipv6 protocol is Down, Vrfid is default (0x60000000)
  IPv6 is enabled, link-local address is fe80::1:10ff:fe87:8d04 [TENTATIVE]
  Global unicast address(es):
    3001::1, subnet is 3001::/64 [TENTATIVE]
  Joined group address(es): ff02::2 ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND cache entry limit is 1000000000
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 160 to 240 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is V6-EGRESS-ACL-BI
  Inbound  common access list is not set, access list is V6-DENY-INGRESS-ACL
  Table Id is 0xe0800000
  Complete protocol adjacency: 0
```

```
  Complete glean adjacency: 0
  Incomplete protocol adjacency: 0
  Incomplete glean adjacency: 0
  Dropped protocol request: 0
  Dropped glean request: 0
```

バンドル インターフェイス上に入力 IPv6 ACL と出力 IPv6 ACLを正常に設定しました。

ACL の変更

この項では、ACL を変更するための設定の例について説明します。

```
*/ Create an Access List*/
Router(config)#ipv4 access-list acl_1

*/Add entries (ACEs) to the ACL*/
Router(config-ipv4-acl)#10 permit ip host 10.3.3.3 host 172.16.5.34
Router(config-ipv4-acl)#20 permit icmp any any
Router(config-ipv4-acl)#30 permit tcp any host 10.3.3.3
Router(config-ipv4-acl)#end

*/Verify the entries of the ACL*/:
Router#show access-lists ipv4 acl_1
ipv4 access-list acl_1
10 permit ip host 10.3.3.3 host 172.16.5.34
20 permit icmp any any
30 permit tcp any host 10.3.3.3

*/Add new entries, one with a sequence number "15" and another without a sequence number
 to the ACL. Delete an entry with the sequence number "30":*/
Router(config)#ipv4 access-list acl_1
Router(config-ipv4-acl)# 15 permit 10.5.5.5 0.0.0.255
Router(config-ipv4-acl)# no 30
Router(config-ipv4-acl)# permit 10.4.4.4 0.0.0.255
Router(config-ipv4-acl)# commit


*/When an entry is added without a sequence number, it is automatically given a sequence
 number
that puts it at the end of the access list. Because the default increment is 10, the
entry will have a sequence
number 10 higher than the last entry in the existing access list*/

*/Verify the entries of the ACL:*/
Router(config)#show access-lists ipv4 acl_1
ipv4 access-list acl_1
 10 permit ipv4 host 10.3.3.3 host 172.16.5.34

15 permit 10.5.5.5 0.0.0.255---*/newly added ACE (with the sequence number)*/
20 permit icmp any any
30 permit ipv4 10.4.4.0 0.0.0.255 any ---*/newly added ACE (without the sequence number)*/

*/The entry with the sequence number 30, that is, "30 permit tcp any host 10.3.3.3" is
deleted from the ACL*/
```

機能中の ACL を正常に変更しました。

ACL ベースの転送の設定

統合ネットワークは、音声、ビデオ、およびデータを伝送します。トラフィックによっては、ルーティング プロトコルが算出したパスを使用するのではなく、特定のパスにルーティングすることが必要になる場合があります。これを実現するには、ACL 設定にネクストホップ アドレスを指定します。これで、パケットベースで宛先アドレスをルックアップするのではなく、ACL に設定したネクストホップ アドレスを使用して指定の宛先にパケットを転送できるようになります。ACL 設定でネクストホップを使用して転送するというこの機能は、ACL ベース転送（ABF）と呼ばれます。

ACL ベース転送を使用すると、ブロードキャスト TV over IP、IP テレフォニー、データなどを対象としたサービスを複数のプロバイダーから選択することが可能になり、カフェテリア形式でインターネットにアクセスできます。サービス プロバイダーは、ユーザ トラフィックをさまざまなコンテンツ プロバイダーに迂回させることができます。

（注） リリース 6.2.2 では、GRE インターフェイスを介してルーティングされた IPv4 ABF のネクストホップがサポートされています。

（注） リリース 6.3.2 では、入力方向に適用された場合、IPv6 ABF ACL はバンドル インターフェイスで動作しません。

機能概要

- ABF は入力 ACL でのみサポートされています。
- ABF はネクストホップの変更をサポートしています。ネクストホップの変更、ネクストホップの削除、または既存のネクストホップ間での変更が可能です。

  （注） ネクストホップがデフォルトの VRF 内にある場合を除き、ACE ルールの定義時にすべてのネクストホップの VRF を指定する必要があります。これにより、パケットがネクストホップへの適切なパスを取得するようになります。

- VRF 認識型 ABF は、最大 3 つのネクストホップがある IPv4 と IPv6 でサポートされています。
- ABF は ACL ベースであるため、ACL 内の既存のルール（ACE）に一致しないパケットはデフォルトの ACL ルール（すべてドロップ）に従います。ACL が（セキュリティ上の理由ではなく）ABF リダイレクトにのみ使用されている場合は、ACL の末尾（最下位のユーザ プライオリティ）に明示的な ACL ルールを組み込んで、すべてのトラフィックを照合

して「許可」します。これにより、ABFルールに一致しないすべてのトラフィックが許可され、通常通りに転送されるようになります。

- ABF は、許可ルールのみでサポートされています。
- VRF-select（ネクストホップに対してVRFのみが設定されている場合）はサポートされていません。
- ABF のデフォルトのルートはサポートされていません。
- 低速パスではABFがサポートされないため、NPUからラインカードCPUへと入力方向にパントされたパケットはABF では処理されません。通常、これらのパケットは、ソフトウェアデータプレーンによる送信先アドレスのルックアップに基づいて転送されます。これらのタイプのパケットには、IPv4 オプション、IPv6 拡張ヘッダー、および収集（未解決/不完全）隣接関係宛のパケットなどがありますが、これらに限定されません。
- ローカルIPインターフェイス宛のパケット（「for-us」パケット）は、ABFアクションが含まれているルールに一致した場合はリダイレクトの対象になります。これは、「for-us」パケットへの一致を避けるために十分な具体的なルールを作成するか、またはABFルールの照合よりも前に（高いプライオリティの）明確な許可ACLルールをACLに配置することで防ぐことができます。

設定例

ACL ベースの転送を設定するには、次のタスクを実行します。

```
/* Enter IPv4 access list configuration mode and configure an ACL: */
router# configure
router(config)# ipv4 access-list abf-acl

/* Set the conditions for the ACL and configure ABF: */
/* The next hop for this entry is specified. */
router(config-ipv4-acl)# 10 permit ipv4 192.168.18.0 0.255.255.255 any nexthop1 ipv4
192.168.20.2
router(config-ipv4-acl)# 15 permit ipv4 192.168.21.0 0.0.0.255 any
router(config-ipv4-acl)# 20 permit ipv4 192.168.22.0 0.0.255.255 any nexthop1 ipv4
192.168.23.2
/* More than two nexthops */
router(config-ipv4-acl)# 25 permit tcp any range 2000 3000 any range 4000 5000 nexthop1
 ipv4 192.168.23.1 nexthop2 ipv4 192.168.24.1 nexthop3 ipv4 192.168.25.1

/* VRF support on ABF */
router(config-ipv4-acl)# 30 permit tcp any eq www host 192.168.12.2 precedence immediate
 nexthop1 vrf vrf1_ipv4 ipv4 192.168.13.2 nexthop2 vrf vrf1_ipv4 ipv4 192.168.14.2

router(config-ipv4-acl)# 35 permit ipv4 any any

router(config-ipv4-acl)# commit

/* (Optional) Display ACL information: */
router# show access-lists ipv4 abf-acl
```

実行コンフィギュレーション

```
ipv4 access-list abf-acl
```

```
10 permit ipv4 192.168.18.0 0.255.255.255 any nexthop1 192.168.20.2
15 permit ipv4 192.168.21.0 0.0.0.255 any
20 permit ipv4 192.168.22.0 0.0.255.255 any nexthop1 192.168.23.2
25 permit tcp any range 2000 3000 any range 4000 5000 nexthop1 ipv4 192.168.23.1 nexthop2
 ipv4 192.168.24.1 nexthop3 ipv4 192.168.25.1
30 permit tcp any eq www host 192.168.12.2 precedence immediate nexthop1 vrf vrf1_ipv4
ipv4 192.168.13.2 nexthop2 vrf vrf1_ipv4 ipv4 192.168.14.2
35 permit ipv4 any any
commit
!
```

確認

ABF 内の IP ネクストホップの状態を確認し、その予想されるネクストホップが起動するようにするには、次のコマンドを使用します。

```
Router# show access-lists ipv4 abf nexthops client pfilter_ea location 0/0/CPU0
Wed Jan 24 14:18:58.667 UTC

ACL name : abf-acl
ACE seq.   NH-1   NH-2   NH-3
 --------- --------------- --------------- ---------------
 10        192.168.13.2
 status          UP
 at status  Not Present
 exist      No
 vrf        default
 track
 pd     ctx  Present
 25   192.168.14.2    192.168.11.1  192.168.12.1
 status   UP   Down   Down
 at status  Not Present  Not Present  Not Present
 exist   No   Yes   Yes
 vrf   default  default  default
 track
 pd ctx   Present  Not present  Not present
 30   192.168.15.1  192.168.12.7
 status   Unknown  Unknown
 at status  Not Present  Not Present
 exist   No   Yes
 vrf   vrf1_ipv4  vrf1_ipv4
 track
 pd ctx   Not present  Not present
```

ABFがラインカードのインターフェイスに現在付加されているかどうかを確認するには、次のコマンドを使用します。

```
show access-lists usage pfilter location all
```

ブリッジ仮想インターフェイスの ACL

ブリッジ仮想インターフェイス（BVI）は、ルータ上のルーティング ドメインとブリッジング ドメイン間にブリッジを提供します。BVI は IP アドレスで設定され、通常のルーテッド インターフェイスとして動作します。BVI 上に ACL を設定して、そのインターフェイスを使用するネットワークに対するトラフィックをフィルタリングできます。

（注） BVIインターフェイスがブリッジドメインに含まれていない場合は、BVIインターフェイスに付加されている ACL を削除しないでください。後で BVI インターフェイスをブリッジ ドメインに追加した場合は、トラフィックがトロップされます。

BVI 上での ACL の設定による TCAM 消費の増加

BVI に ACL が設定されている場合、TCAM リソースの消費は次のように影響されます。

- ACL が BVI インターフェイスに付加されていると、TCAM エントリは物理インターフェイスのメンバーシップとは関係なく、すべてのライン カード上でプログラミングされます。これにより、BVI メンバ インターフェイスがないライン カードでも TCAM リソースの消費が増加することになります。
- ACL が BVI インターフェイスに付加されていると、TCAM エントリは物理インターフェイスのメンバーシップとは関係なく、ライン カードのすべての NPU 上でプログラミングされます。これにより、BVI メンバ インターフェイスがない NPU でも TCAM リソースの消費が増加することになります。
- 入力 ACL では、同じ ACL の TCAM エントリが同じ NPU 上のインターフェイス間で共有されます。
- 出力 ACL では、同じ ACL の TCAM エントリはすべてのインターフェイスに一意です。これにより、TCAM リソースの消費が増加します。

BVI 上での ACL 設定の制約事項

BVI 上での ACL の設定に進む前に、次の制約事項を認識しておく必要があります。

- BVI では、出力 IPv6 ACL はサポートされていません。
- **hw-module** コマンドを使用して BVI 上で出力 IPv4 ACL を有効にすると、その ACL では他のインターフェイス タイプがサポートされません（このモードの ACL では、非 BVI インターフェイスはサポートされません）。

BVI での IPv4 出力 ACL 設定の前提条件

デフォルトでは、BVI 上の IPv4 出力 ACL は無効になっており、ACL が BVI に付加されていても ACL のフィルタリングは実行されません。そのため、**hw-module** コマンドを使用して、ライン カードのリロード時に ACL を有効にします。

（注） IPv4 と IPv6 の入力 ACL はこの設定を必要としません。

次の設定を使用して、ハードウェア 上の BVI で IPv4 出力 ACL を有効にし、ライン カードをリロードします。

```
/* Enable an IPv4 egress ACL on BVI */
RP/0/RP0/CPU0:router(config)# hw-module profile acl egress layer3 interface-based
/* Enable permit statistics for the egress ACL (by default, only deny statistics are
shown)*/
RP/0/RP0/CPU0:router(config)# hw-module profile stats acl-permit
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# end
RP/0/RP0/CPU0:router# reload location all
Wed Apr 5 23:05:46.193 UTC
Proceed with reload? [confirm]
```

設定

次の項では、BVI 上で IPv4 の入力 ACL と出力 ACL を設定する手順について説明します。

BVI で IPv4 の入力 ACL と出力 ACL を設定するには、次の設定例を使用した手順を実行します。

1. グローバル コンフィギュレーション モードを開始し、IPv4 入力 ACL を設定します。

```
RP/0/RP0/CPU0:router(config)# ipv4 access-list v4-acl-ingress
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit tcp any 10.1.1.0/24 dscp cs6
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny udp any any eq ssh
RP/0/RP0/CPU0:router(config-ipv4-acl)# 30 permit ipv4 any any
RP/0/RP0/CPU0:router(config-ipv4-acl)# commit
RP/0/RP0/CPU0:router(config-ipv4-acl)# exit
```

2. IPv4 出力 ACL を設定します。

```
RP/0/RP0/CPU0:router(config)# ipv4 access-list v4-acl-egress
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 deny ipv4 any any fragments log
RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny tcp any any ack
RP/0/RP0/CPU0:router(config-ipv4-acl)# 30 permit ipv4 any any
RP/0/RP0/CPU0:router(config-ipv4-acl)# commit
RP/0/RP0/CPU0:router(config-ipv4-acl)# exit
```

3. BVI にマップする必要があるギガビット イーサネット インターフェイスを設定し、レイヤ 2 トランスポートに対して有効にします。

```
RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 0/0/0/0
RP/0/RP0/CPU0:router(config-if)# l2transport
RP/0/RP0/CPU0:router(config-if-l2)# commit
```

4. 入力 ACL と出力 ACL を BVI に付加します。

```
RP/0/RP0/CPU0:router(config)# interface BVI1
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group v4-acl-ingress ingress
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group v4-acl-egress egress
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:router(config-if)# exit
```

5. ギガビット イーサネット インターフェイスと BVI でブリッジ ドメインを設定します。

```
RP/0/RP0/CPU0:router(config)# l2vpn
RP/0/RP0/CPU0:router(config-l2vpn)# bridge group BG1
RP/0/RP0/CPU0:router(config-l2vpn-bg)# bridge-domain B1
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/0/0/0
```

```
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd-ac)# routed interface BVI1

RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)# commit
RP/0/RP0/CPU0:router(config-l2vpn-bg-bd)# exit
RP/0/RP0/CPU0:router(config-l2vpn-bg)# exit
RP/0/RP0/CPU0:router(config-l2vpn)# exit
```

6. 設定が正常にコミットされていることを確認します。

```
RP/0/RP0/CPU0:router(config)# show run
...
!
ipv4 access-list v4-acl-egress
 10 deny ipv4 any any fragments log
 20 deny tcp any any ack
 30 permit ipv4 any any
!
ipv4 access-list v4-acl-ingress
 10 permit tcp any 10.1.1.0/24 dscp cs6
 20 deny udp any any eq ssh
 30 permit ipv4 any any
!
interface GigabitEthernet0/0/0/0
 l2transport
 !
!
interface BVI1
 ipv4 access-group v4-acl-ingress ingress
 ipv4 access-group v4-acl-egress egress

!
l2vpn
 bridge group BG1
  bridge-domain B1
   interface GigabitEthernet0/0/0/0
   !
   routed interface BVI1
  !
 !
!
end
```

7. エグゼクティブ特権モードに移行して、ACL が機能していることを確認します。

```
RP/0/RP0/CPU0:router# show access-lists interface bvi1
Tue May 9 10:01:25.732 EDT
Input ACL (common): GigabitEthernet 0/0/0/0 (interface): v4-acl-ingress
Output ACL: v4-acl-egress

RP/0/RP0/CPU0:router# show access-lists summary
Tue May 9 10:02:01.167 EDT
ACL Summary:
Total ACLs configured: 2
Total ACEs configured: 6


RP/0/RP0/CPU0:router# show access-lists ipv4 v4-acl-egress hardware egress location
 0/0/CPU0
ipv4 access-list v4-acl-egress
10 deny ipv4 any any fragments log (15214 matches)
20 deny tcp any any ack (15214 matches)
```

```
30 permit ipv4 any any (15214 matches)
```

設定した ACL、ACE の総数（ACL ごとに 3 つ）、およびハードウェア内の ACE の一致も出力に明確に表示されます。

IPv4 の入力および出力 ACL が BVI 上に正常に設定されました。

フラグメント制御を使用した ACL の設定

非フラグメント パケットとパケットの先頭フラグメントは、IP 拡張アクセス リストで処理されていましたが（このアクセスリストを適用した場合）、先頭以外のフラグメントはデフォルトで許可されていました。しかし、現在では、フラグメント制御機能付きの IP 拡張アクセスリストによって、パケットの先頭以外のフラグメントもさらにきめ細かく制御できるようになりました。この機能を使用すると、IP 拡張アクセス リストを適用するときに、パケットの先頭以外の IP フラグメントを調べるかどうかを指定できます。

先頭以外のフラグメントにはレイヤ 3 情報だけしか含まれていないので、レイヤ 3 情報のみを含んでいるアクセスリスト エントリを先頭以外のフラグメントに適用できるようになりました。フラグメントにはフィルタリングに必要な情報がすべて含まれているため、アクセスリスト エントリをパケットのフラグメントに適用できるのです。

この機能により、オプションの **fragments** キーワードが、IP アクセス リスト コマンドの **deny** と **permit** に追加されます。アクセスリスト エントリに **fragments** キーワードを指定することにより、その特定のアクセスリストエントリは、パケットの先頭以外のフラグメントにのみ適用されます。フラグメントは指定に応じて許可または拒否されます。

fragments キーワードの有無に応じたアクセスリスト エントリの動作の概要は、以下のとおりです。

アクセス リスト エントリの状態	結果

fragments キーワードがなく、すべてのアクセス リスト エントリ情報が一致する	アクセス リスト エントリにレイヤ 3 情報のみが含まれている場合：
	• エントリは、非フラグメント パケット、先頭フラグメント、先頭以外のフラグメントに適用されます。
	アクセス リスト エントリにレイヤ 3 情報とレイヤ 4 情報が含まれている場合：
	• エントリは、非フラグメント パケットと先頭フラグメントに適用されます。
	  • エントリが一致し、かつ **permit** ステートメントである場合、パケットまたはフラグメントは許可されます。
	  • エントリが一致し、かつ **deny** ステートメントである場合、パケットまたはフラグメントは拒否されます。
	• エントリは、次の方法で先頭以外のフラグメントにも適用されます。先頭以外のフラグメントにはレイヤ 3 情報のみが含まれているため、アクセス リスト エントリのレイヤ 3 部分のみが適用されます。アクセス リスト エントリのレイヤ 3 の部分が一致し、
	  • エントリが **permit** ステートメントである場合は、先頭以外のフラグメントが許可されます。
	  • エントリが **deny** ステートメントの場合は、次のアクセス リスト エントリが処理されます。
	（注） 先頭以外のフラグメントと、非フラグメントまたは先頭フラグメントとでは、deny ステートメントの処理方法が異なります。

fragments キーワードがあり、すべてのアクセス リスト エントリ情報が一致する	アクセス リスト エントリは、先頭以外のフラグメントにのみ適用されます。 （注） レイヤ 4 情報を含むアクセス リスト エントリに **fragments** キーワードは設定できません。

すべてのアクセス リスト エントリに **fragments** キーワードを追加しないでください。IP パケットの先頭フラグメントは非フラグメントと見なされ、それ以降のフラグメントとは独立して扱われるためです。先頭フラグメントは、**fragments** キーワードが含まれているアクセス リストの permit または deny エントリとは一致しないため、パケットは次のアクセス リスト エントリと比較されます。この比較は、**fragments** キーワードが含まれていないアクセス リスト エントリによってパケットが許可または拒否されるまで続きます。したがって、deny エントリごと

に、2 つのアクセス リスト エントリが必要になる場合があります。ペアの最初の deny エントリは **fragments** キーワードを含んでおらず、先頭フラグメントに適用されます。ペアの 2 番目の deny エントリは **fragments** キーワードを含んでおり、以降のフラグメントに適用されます。同じホストに対して複数の **deny** アクセスリスト エントリがあり、それぞれのレイヤ 4 ポートが異なる場合は、そのホスト用として、**fragments** キーワードを含む deny アクセスリスト エントリを 1 つだけ追加する必要があります。このように、パケットのすべてのフラグメントは、アクセス リストによって同様に扱われます。

IP データグラムのパケット フラグメントは個々のパケットと見なされ、各フラグメントはアクセス リスト アカウンティングとアクセス リスト違反カウントの 1 つのパケットとして個別にカウントされます。

（注） アクセス リストおよび IP フラグメントに関するあらゆるケースを **fragments** キーワードで解決できるわけではありません。

（注） ACL 処理の範囲内で、レイヤ 3 情報は IPv4 ヘッダー内のフィールド（送信元、宛先、プロトコルなど）を参照します。レイヤ 4 情報は、TCP または UDP の送信元ポートと宛先ポート、TCP のフラグ、ICMP のタイプとコードなど、IPv4 ヘッダーの後ろに含まれるその他のデータを参照します。

フラグメント タイプでの照合を実行するための IPv4 ACL の設定

ほとんどの DoS（サービス妨害）攻撃は、フラグメント化されたパケットでネットワークをフラッディングさせることで機能します。ネットワーク内のパケットの着信フラグメントをフィルタリングすることで、このような攻撃に対する保護レイヤを余分に追加できます。

フラグメント タイプを照合する IPv4 ACL を設定し、適切なアクションを実行できます。次の設定例ではさまざまなフラグメント オプションで使用できます。

```
/* Enter the global configuraton mode and configure an IPv4 access list */
Router# config
Router(config)# ipv4 access-list TEST
Router(config-ipv4-acl)# 10 permit tcp any any

/* Configure an ACE to match on the dont-fragment flag (indicates a non-fragmented packet)

 and forward the packet to the default (pre-configured) next hop  */
Router(config-ipv4-acl)# 20 permit tcp any any fragment-type dont-fragment default

/* Configure an ACE to match on the is-fragment flag (indicates a fragmented packet)
 and forward the packet to a next hop of 10.10.10.1  */
Router(config-ipv4-acl)# 30 permit udp any any fragment-type is-fragment nexthop1 ipv4
10.10.10.1

/* Configure an ACE to match on the first-fragment flag (indicates the first fragment
of a fragmented packet)
 and forward the packet to a next hop of 20.20.20.1  */
```

```
Router(config-ipv4-acl)# 40 permit ospf any any fragment-type first-fragment nexthop1
ipv4 20.20.20.1


/* Configure an ACE to match on the last-fragment flag (indicates the last fragment of
a fragmented packet)
 and forward the packet to a next hop of 30.30.30.1  */
Router(config-ipv4-acl)# 50 permit icmp any any fragment-type last-fragment nexthop1
ipv4 30.30.30.1
Router(config-ipv4-acl)# commit
```

使用例：最初のフラグメントと最後のフラグメントを照合する IPv4 ACL の設定

この項では、パケットの最初のフラグメントの場合はそのフラグメントを転送し、パケットの最後のフラグメントの場合はそのフラグメントを破棄するように ACL を設定する使用例について説明します。

この設定では、ACLがフラグメント オフセット値（最初のフラグメントの場合は「0」）を確認します。そのフラグメントがパケットの最初のフラグメントの場合は、パケットが転送されます。そのフラグメントがパケットの最後のフラグメントの場合は、インターフェイスでドロップされます。

```
/* Enter the global configuraton mode and configure an IPv4 access list */
Router# config
Thu Jan 11 11:56:27.221 IST
Router(config)# ipv4 access-list ACLFIRSTFRAG

/* Configure an ACE to match on the first fragment.
If the fragment offset value equals 0, the fragment is forwarded to the 192.168.1.2 next
 hop */
Router(config-ipv4-acl)# 10 permit tcp any any fragment-type first-fragment nexthop1
ipv4 192.168.1.2

/* Configure an ACE to match on the last fragment, and drop the fragment at the interface.
 */
Router(config-ipv4-acl)# 20 deny tcp any any fragment-type last-fragment
Router(config-ipv4-acl)# commit
Thu Jan 11 12:01:33.297 IST

/* Validate the configuration */
Router(config-ipv4-acl)# do show access-lists
Thu Jan 11 12:05:23.646 IST
ipv4 access-list ACLFIRSTFRAG
 10 permit tcp any any fragment-type first-fragment nexthop1 ipv4 192.168.1.20
 20 deny tcp any any fragment-type last-fragment
```

フラグメント タイプを照合する IPv4 ACL を正常に設定しました。

ACL でのフラグメント オフセットによる一致

フラグメント オフセット値でパケットをフィルタリングするようにアクセス コントロール リスト（ACL）のルールを作成できます。パケットが permit ステートメントまたは deny ステートメントの条件に一致するかどうかにより、パケットはインターフェイスでそれぞれ処理されるか、またはドロップされます。フラグメント オフセット フィルタリングは、圧縮モードの ACL を使用して入力方向でのみサポートされています。

この機能の詳細については、『*IP Addresses and Services Configuration Guide for Cisco NCS 540 Series Routers*』の「*Implementing Access Lists and Prefix Lists*」の章を参照してください。詳細なコマンドリファレンスについては、『』の「*Access List Commands*」の章を参照してください。

フラグメント オフセットによる ACL 照合の設定

ACL でフラグメント オフセット一致を設定するには、IPv4 または IPv6 アクセス リスト コンフィギュレーション モードで **permit** コマンドまたは **deny** コマンドに **fragment-offset** オプションを使用します。

（注） フラグメント オフセット フィルタリングの場合は、特定の ACL を圧縮レベルが 3 のインターフェイスに付加する必要があります。そうしないと、設定が拒否されます。

設定

次に、IPv4 ヘッダーごとのフラグメント オフセットに基づいて ACL ルールを設定する例を示します。ここでは、パケットの IPv4 ヘッダー内のフラグメント オフセットが 300 ～ 400 の範囲内にある場合にのみ、パケットが許可されます。値 *300* ～ *400* は 8 バイトの単位に基づいており、これは *2400* ～ *3200* バイトのフラグメント オフセットと同じです。

```
/* Configure ACL */
Router# configure
Router(config)# ipv4 access-list fragment-offset-acl
Router(config-ipv4-acl)# 10 permit ipv4 any any fragment-offset range 300 400
Router# commit

/* Attach the ACL to the interface */
Router# configure
Router(config)# interface Bundle-Ether70
Router(config-if)# ipv4 access-group fragment-offset-acl ingress compress level 3
Router# commit
```

実行コンフィギュレーション

```
ipv4 access-list fragment-offset-acl
 10 permit ipv4 any any fragment-offset range 300 400
!

interface Bundle-Ether70
 ipv4 address 192.0.2.1 255.255.255.0
 ipv6 address 2001:DB8::1:1::1/48
 ipv4 access-group fragment-offset-acl ingress compress level 3
!
```

ACL でのフラグメント オフセットの一致の確認

```
Router#
show access-lists ipv4 fragment-offset-acl usage pfilter loc 0/0/CPU0
```

```
Wed Apr 12 19:49:54.457 UTC
Interface : Bundle-Ether70
    Input  ACL : Common-ACL : N/A  ACL : fragment-offset-acl  (comp-lvl 3)
    Output ACL : N/A



Router#
show access-lists ipv4 fragment-offset-acl hardware ing int Bundle-Ether70 loc 0/0/CPU0

Wed Apr 12 19:51:07.837 UTC
ipv4 access-list fragment-offset-acl
 10 permit ipv4 any any fragment-offset range 300 400
```

関連コマンド

- ipv4 access-list
- ipv6 access-list
- deny（IPv4）
- deny（IPv6）
- fragment-offset
- permit（IPv4）
- permit（IPv6）

IP パケット長による ACL フィルタリングの設定

入力インターフェイスでパケット長を使用してパケットをフィルタリングするようにアクセス コントロール リストを設定できます。パケットが permit ステートメントまたは deny ステートメントのパケット長条件と一致するかどうかに応じて、パケットはインターフェイスでそれぞれ処理またはドロップされます。

ACL でパケット長のフィルタリングを設定するには、IPv4 または IPv6 アクセス リスト コンフィギュレーション モードで **permit** コマンドまたは **deny** コマンドに **packet-length** オプションを使用します。

制約事項

ACL でのパケット長フィルタ リング機能には次の制約事項があります。

- パケット長フィルタリングは、シンプル（非圧縮）ACL とハイブリッド（圧縮）ACL の両方で入力方向のみがサポートされています。
- IPv6 のパケット長フィルタ リングは、ハイブリッド ACL でのみサポートされています。シンプル ACL ではサポートされていません。

- IPv4 のシンプル ACL でサポートされているのは、量子化された（16 で割り切れる値）パケット長フィルタリングのみです。

パケット長を使用してフィルタリングするためのシンプルな IPv4 ACL の設定

シンプルな ACL を設定して IPv4 ネットワーク内のパケット長でフィルタリングするには、次のステップを実行します。

1. グローバル コンフィギュレーション モードを開始し、パケット長値でパケットをフィルタリングするシンプルな IPv4 アクセス リストを設定します。

   次の特定の例では、指定したパケット長の条件に一致するパケットのみを処理する一連のステートメントを設定します。その他のパケットは、この ACL が入力インターフェイスに適用された時点でドロップされます。

   ```
   RP/0/RP0/CPU0:router# config
   RP/0/RP0/CPU0:router(config)# ipv4 access-list pktlen-v4
   RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit tcp any any packet-length eq 1664
   RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit udp any any packet-length range 1600
    2000
   RP/0/RP0/CPU0:router(config-ipv4-acl)# 30 deny ipv4 any any
   ```

2. ACL をコミットし、IPv4 ACL コンフィギュレーション モードを終了します。

   ```
   RP/0/RP0/CPU0:router(config-ipv4-acl)# commit
   RP/0/RP0/CPU0:router(config-ipv4-acl)# end
   ```

3. 必要なイーサネット インターフェイスに ACL を適用します。

   ```
   RP/0/RP0/CPU0:router(config)# interface Te0/0/0/0
   RP/0/RP0/CPU0:router(config-if)# ipv4 access-group pktlen-v4 ingress
   ```

4. 設定をコミットし、インターフェイス コンフィギュレーション モードを終了します。

   ```
   RP/0/RP0/CPU0:router(config-if)# commit
   RP/0/RP0/CPU0:router(config-if)# end
   ```

5. 設定を確認します。

   ```
   RP/0/RP0/CPU0:router# show access-lists pktlen-v4


   ipv4 access-list pktlen-v4
   10 permit ipv4 host 10.0.0.10 any packet-length lt 1008
   20 permit ipv4 host 10.0.0.9 any packet-length gt 992
   ```

6. ハードウェアで ACL の一致を確認します。

   ```
   RP/0/RP0/CPU0:router# show access-lists pktlen-v4 hardware ingress location 0/0/CPU0

   ipv4 access-list pklen-v4
   10 permit ipv4 host 10.0.0.10 any packet-length lt 1008
   20 permit ipv4 host 10.0.0.9 any packet-length gt 992
   ```

パケット長でフィルタリングするためのシンプルな IPv4 ACL を正常に設定しました。

パケット長を使用してフィルタリングするための拡張 IPv4 ACL の設定

拡張 ACL を設定して IPv4 ネットワーク内のパケット長でフィルタリングするには、次のステップを実行します。

1. グローバル コンフィギュレーション モードを開始し、拡張 ACL を設定するオブジェクト グループを作成します。

```
RP/0/RP0/CPU0:router(config)# object-group network ipv4 netobject1
RP/0/RP0/CPU0:router(config-object-group-ipv4)# 50.0.0.0/24
RP/0/RP0/CPU0:router(config-object-group-ipv4)# commit
```

2. グローバル コンフィギュレーション モードから、パケット長値でパケットをフィルタリングする IPv4 アクセス リストを設定します。

   次の特定の例では、指定したパケット長の条件に一致するパケットのみを処理するステートメントを設定します。その他のパケットは、この ACL が入力インターフェイスに適用された時点でドロップされます。

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list scaled_acl1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 net-group netobject1 any
packet-length eq 1000
```

3. ACL をコミットし、IPv4 ACL コンフィギュレーション モードを終了します。

```
RP/0/RP0/CPU0:router(config-ipv4-acl)# commit
RP/0/RP0/CPU0:router(config-ipv4-acl)# end
```

4. 必要なギガビット イーサネット インターフェイスに ACL を適用します。

```
RP/0/RP0/CPU0:router(config)# interface Te0/0/0/3
RP/0/RP0/CPU0:router(config-if)#ipv4 access-group scaled_acl1 ingress
```

5. 設定をコミットし、インターフェイス コンフィギュレーション モードを終了します。

```
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:router(config-if)# end
```

6. 設定を確認します。

```
RP/0/RP0/CPU0:router# show access-lists scaled_acl1
ipv4 access-list scaled_acl1
10 permit ipv4 net-group netobject1 any packet-length eq 1000
```

7. ハードウェアで ACL の一致を確認します。

```
RP/0/RP0/CPU0:router# show access-lists scaled_acl1 hardware ingress location 0/0/CPU0
ipv4 access-list scaled_acl1
10 permit ipv4 net-group netobject1 any packet-length eq 1000 (1500 hw matches)
```

パケット長でフィルタリングするための拡張 IPv4 ACL を正常に設定しました。

パケット長を使用してフィルタリングするための拡張 IPv6 ACL の設定

拡張 ACL を設定して IPv6 ネットワーク内のパケット長でフィルタリングするには、次のステップを実行します。

1. グローバル コンフィギュレーション モードを開始し、拡張 ACL を設定するオブジェクト グループを作成します。

```
RP/0/RP0/CPU0:router(config)# object-group  network ipv6 netobject2
RP/0/RP0/CPU0:router(config-object-group-ipv6)# 2001::0/128
RP/0/RP0/CPU0:router(config-object-group-ipv6)# commit
```

2. グローバル コンフィギュレーション モードから、パケット長値でパケットをフィルタリングする拡張 IPv6 アクセス リストを設定します。

   次の特定の例では、指定したパケット長の条件に一致するパケットのみを処理するステートメントを設定します。その他のパケットは、この ACL が入力インターフェイスに適用された時点でドロップされます。

```
RP/0/RP0/CPU0:router(config)# ipv6 access-list scaled_acl2
RP/0/RP0/CPU0:router(config-ipv6-acl)# 10 permit ipv6 net-group netobject2 any
packet-length eq 1000
RP/0/RP0/CPU0:router(config-ipv6-acl)# commit
```

3. ACL をコミットし、IPv6 ACL コンフィギュレーション モードを終了します。

```
RP/0/RP0/CPU0:router(config-ipv6-acl)# commit
RP/0/RP0/CPU0:router(config-ipv6-acl)# end
```

4. 必要なギガビット イーサネット インターフェイスに ACL を適用します。

```
RP/0/RP0/CPU0:router# config
RP/0/RP0/CPU0:router(config)# interface Te/0/0/0/3
RP/0/RP0/CPU0:router(config-if)# ipv6 access-group scaled_acl2 ingress
```

5. 設定をコミットし、インターフェイス コンフィギュレーション モードを終了します。

```
RP/0/RP0/CPU0:router(config-if)# commit
RP/0/RP0/CPU0:router(config-if)# end
```

6. 設定を確認します。

```
RP/0/RP0/CPU0:router# show access-lists ipv6 scaled_acl2
ipv6 access-list scaled_acl2
10 permit ipv6 net-group netobject2 any packet-length eq 1000
```

7. ハードウェアで ACL の一致を確認します。

```
RP/0/RP0/CPU0:router# show access-lists ipv6 scaled_acl2 hardware ingress location
0/0/CPU0
ipv6 access-list scaled_acl2
10 permit ipv6 net-group netobject2 any packet-length eq 1000 (2000 hw matches)
```

パケット長でフィルタリングするための拡張 IPv6 ACL を正常に設定しました。

オブジェクトグループ ACL の概要

オブジェクトグループ ACL を使用すると、ユーザ、デバイス、またはプロトコルをグループに分類できるため、グループ レベルのアクセス コントロール ポリシーを設定できます。個別の IP アドレス、プロトコル、およびポート番号を複数の ACE に指定する代わりに、単一の ACL にオブジェクト グループを指定できます。

この機能は、現在、数百ものACLが含まれている大規模なネットワークには非常に有益です。オブジェクトグループACL機能を使用することで、ACLあたりのACEの数が大幅に削減します。また、オブジェクトグループ ACL は判読性が高く、従来の ACL よりも簡単に管理できます。従来の ACL の代わりに オブジェクトグループ ACL を使用することで、TCAM に必要なストレージが最適化されます。

オブジェクトグループ ACL のタイプ

Cisco IOS XR では 2 つのタイプの オブジェクトグループ ACL を作成できます。

- **ネットワーク オブジェクトグループ ACL**：ホスト IP アドレス、ネットワーク IP アドレス、または別のオブジェクト グループ内にネストされているオブジェクト グループから構成されます。
- **ポート オブジェクトグループ ACL**：ポートとサポートしているレイヤ 3/レイヤ 4 プロトコルのグループから構成されます。

ACL の圧縮

オブジェクトグループACLは圧縮を使用して多数のACEに対応します。圧縮は、ACEの次の3つのフィールドを圧縮することで実行されます。

- 送信元 IP プレフィックス
- 送信先 IP プレフィックス
- 送信元ポート番号

Cisco NCS 540 シリーズ ルータは、入力インターフェイス上の ACL のアクセスグループ設定で 2 つの圧縮レベルのみをサポートしています。

- **圧縮レベル 0**：圧縮は ACE フィールドでは実行されません。

  このモードでは、オブジェクトグループ ACL は従来の ACL のように動作します。内部 TCAM リソースを利用するため、ACL の処理に必要なシステム リソースと時間に膨大な影響を与えます。

- **圧縮レベル 3**：ACE の 3 つフィールド（送信元 IP、送信先 IP、および送信元ポート）すべてが圧縮されます。

このモードでは、プレフィックスのルックアップには外部 TCAM 、ACE のルックアップには内部 TCAM を使用します。このモードは、16 ビット ベースのパケット長フィルタリングとフラグメント オフセット フィルタリングをサポートしています。

オブジェクトグループ ACL の設定

はじめる前に

オブジェクトグループ ACL に適用される次の情報および制約事項を認識しておく必要があります。

- 従来の ACL とオブジェクトグループ ACL の両方を含む ACL を設定できます。DHCP、QoS、Cisco IOS ファイアウォールなどの拡張 ACL を使用するほとんどの機能がオブジェクトグループ ACL をサポートしています。
- オブジェクト グループや、そのオブジェクト グループを参照する ACE を定義し直すことなく、オブジェクト グループ内のオブジェクトをダイナミックに変更できます。
- オブジェクトグループ ACL は、送信元グループまたは送信先グループ、あるいは送信元と送信先のグループの両方を使用して複数回設定できます。
- オブジェクト グループを削除するには、最初にすべての ACL と CPL からそのグループを削除する必要があります。

ネットワーク オブジェクトグループ ACL の設定

ネットワーク オブジェクト グループには単一または複数のネットワーク オブジェクトを含めることができます。

設定

次の一連の設定ステートメントを使用して、ネットワーク オブジェクトグループ ACL を設定します。

```
/* From the global configuration mode, create a network object group. */
RP/0/RP0/CPU0:router(config)# object-group network ipv4 netobj1
RP/0/RP0/CPU0:router(config-object-group-ipv4)# description my-network-object
RP/0/RP0/CPU0:router(config-object-group-ipv4)# host 10.1.1.1
RP/0/RP0/CPU0:router(config-object-group-ipv4)# 10.2.1.0 255.255.255.0
RP/0/RP0/CPU0:router(config-object-group-ipv4)# range 10.3.1.10 10.3.1.50

/* Create an access list referencing the object group. */
RP/0/RP0/CPU0:router(config)# ipv4 access-list network-object-acl permit ipv4 net-group
 netobj1 any

/* Apply the access list containing the object group to the desired interface and commit
 your configuration. */
RP/0/RP0/CPU0:router(config)# interface Te/0/0/0/3
RP/0/RP0/CPU0:router(config-if)# ipv4 address 1.1.1.1/24
RP/0/RP0/CPU0:router(config-if)# no shut
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group network-object-acl ingress compress
```

```
level 3
RP/0/RP0/CPU0:router(config-if)# commit
Tue Mar 28 10:23:34.106 IST

RP/0/0/CPU0:Mar 28 10:37:48.570 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/10/3 , changed state to Down
RP/0/0/CPU0:Mar 28 10:37:48.608 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/10/3 , changed state to Up

RP/0/RP0/CPU0:router(config-if)# exit
```

実行コンフィギュレーション

設定を確認します。

```
RP/0/RP0/CPU0:router(config)# show run
Tue Mar 28 10:37:55.737 IST

Building configuration...
!! IOS XR Configuration 0.0.0
...

!
object-group network ipv4 netobj1
 10.2.1.0/24
 host 10.1.1.1
 range  10.3.1.10 10.3.1.50
 description my-network-object
!
!
ipv4 access-list network-object-acl
 10 permit ipv4 net-group netobj1 any
!
interface Te0/0/0/0/3
 ipv4 address 1.1.1.1 255.255.255.0
 ipv4 access-group network-object-acl ingress compress level 3
!
```

ネットワーク オブジェクトグループ ACL は正常に設定されました。

ポート オブジェクトグループ ACL の設定

ポート オブジェクト グループには単一または複数のポート オブジェクトを含めることができます。

設定

次の一連の設定ステートメントを使用して、ポートオブジェクトグループACLを設定します。

```
/* From the global configuration mode, create a port object group, and commit your
configuration. */
RP/0/RP0/CPU0:router(config)# object-group port portobj1
RP/0/RP0/CPU0:router(config-object-group-ipv4)# description my-port-object
RP/0/RP0/CPU0:router(config-object-group-ipv4)# eq bgp
RP/0/RP0/CPU0:router(config-object-group-ipv4)# range 100 200
RP/0/RP0/CPU0:router(config-object-group-ipv4)# commit
RP/0/RP0/CPU0:router(config-object-group-ipv4)# exit
```

```
/* Create an access list referencing the object group. */
RP/0/RP0/CPU0:router(config)# ipv4 access-list port-object-acl permit ipv4 net-group
portobj1

/* Apply the access list containing the object group to the desired interface and commit
 your configuration. */
RP/0/RP0/CPU0:router(config)# interface Te0/0/0/3
RP/0/RP0/CPU0:router(config-if)# ipv4 address 2.2.2.2/24
RP/0/RP0/CPU0:router(config-if)# ipv4 access-group port-obj-acl ingress compress level
3
RP/0/RP0/CPU0:router(config-if)# no shut
RP/0/RP0/CPU0:router(config-if)# commit
Tue Mar 28 10:23:34.106 IST

RP/0/0/CPU0:Mar 28 10:37:48.570 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/10/3 , changed state to Down
RP/0/0/CPU0:Mar 28 10:37:48.608 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/10/3 , changed state to Up

RP/0/RP0/CPU0:router(config-if)# exit
```

実行コンフィギュレーション

設定を確認します。

```
RP/0/RP0/CPU0:router(config)# show run
Tue Mar 28 10:37:55.737 IST

Building configuration...
!! IOS XR Configuration 0.0.0
...
object-group port portobj1
 eq bgp
 range 100 200
!

ipv4 access-list port-object-acl
 10 permit tcp net-group portobj1
!
interface Te/0/0/0/3
 ipv4 access-group port-obj-acl ingress compress level 3
!
end
!
```

ポート オブジェクトグループ ACL が正常に設定されました。

オブジェクトグループ ACL 圧縮の確認

動作中の設定済みオブジェクトグループ ACL と ACL 内の AEC の圧縮を確認するには、この項で説明するコマンドを使用します。

（注） この項に示す出力はこの項のみのサンプルであり、前の項で示した設定に関連するものではありません。

確認

オブジェクトグループ ACL の圧縮を確認するには、次の一連の verification コマンドを使用します。

```
/* Verify the entries of the ACL in operation. */

RP/0/RP0/CPU0:router# show access-lists ipv4 network-object-acl hardware ingress location
 0/0/CPU0
ipv4 access-list network-object-acl
40 permit ospf net-group n_192.168.0.0_16 any (20898463272 matches)
70 permit tcp any net-group CORP_ALL_V4 established
100 permit udp net-group INTERNAL port-group KERBEROS_UDP net-group CORP_ALL_V4
130 permit udp net-group INTERNAL port-group DNS_UDP net-group CORP_ALL_V4
160 permit udp net-group INTERNAL port-group NTP net-group CORP_ALL_V4
190 permit udp net-group INTERNAL port-group LDAP_UDP net-group CORP_ALL_V4
...
1500 permit udp net-group VLAN60_SECURITY net-group h_192.168.77.242 port-group
UDP_50000-50100
1530 deny ipv4 net-group VLAN60_SECURITY any log (20891956640 matches)
...


/* Verify the ACE compression in the ACL. */
RP/0/RP0/CPU0:router# show access-lists ipv4 network-object-acl hardware ingress verify
 location 0/0/CPU0
Verifying TCAM entries for network-object-acl
Please wait...



    INTF    NPU lookup  ACL # intf Total  compression Total   result failed(Entry) TCAM
 entries
                type    ID  shared ACES   prefix-type Entries        ACE SEQ #
verified
---------- --- ------- --- ------ ------ ----------- ------- ------ -------------
------------

TenGigE0_0_0_10_3 (ifhandle: 0x1c8)

             1 IPV4      2      1    247 COMPRESSED      810 passed
     810
                                         SRC IP         2746 passed
    2746
                                         DEST IP        3413 passed
    3413
                                         SRC PORT        340 passed
     340
```

ACL 内の ACE の圧縮を正常に確認しました。

IPv4 ACL での TTL の照合の設定

IPv4 ヘッダーに指定されている TTL 値で照合するように ACL を設定できます。TTL 一致条件を単一の値か、または複数の値に基づくように指定できます。また、**set ttl** コマンドを使用して、IPv4 ヘッダー内の TTL 値を書き換えることもできます。

IPv4 ACL での TTL の照合の使用に関する制限

IPv4 ACL に TTL の照合を使用するには、次の制限があります。

- TTL の照合は、入力 ACL でのみサポートされています。
- **set ttl** コマンドを使用した TTL の書き換えは、ACL ロギングと併用できません。
- TTL の書き換えを IP-in-IP ヘッダーの外側の IPv4 ヘッダーに適用する場合は、GRE カプセル化解除により外側の IPv4 ヘッダーのカプセル化が解除されると、内部 IPv4 ヘッダーにも TTL の書き換えが適用されます。

設定

ACL に TTL の照合を設定するには、次のステップを実行します。

```
/* Enable TTL matching in the global configuration mode by using the hw-module command
*/
Router(config)# hw-module profile tcam format access-list ipv4 dst-addr dst-port proto
port-range enable-set-ttl ttl-match

/* Configure an IPv4 ACL with the TTL parameters */
Router(config)# ipv4 access-list acl-v4
Router(config-ipv4-acl)# 10 deny tcp any any ttl eq 100
Router(config-ipv4-acl)# 20 permit tcp any any ttl range 1 50 set ttl 200
Router(config-ipv4-acl)# 30 permit tcp any any ttl neq 100 set ttl 255
Router(config-ipv4-acl)# commit
Thu Nov  2 12:22:58.948 IST

/* Attach the IPv4 ACL to the GigE interface */
Router(config)# interface Te0/0/0/0
Router(config-if)# ipv4 address 15.1.1.1 255.255.255.0
Router(config-if)# ipv4 access-group acl-v4 ingress
Router(config-if)# commit
```

実行コンフィギュレーション

show run コマンドを使用して設定を検証します。

```
Router(config)# show run
Thu Nov  2 14:01:53.376 IST
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Thu Nov  2 12:22:59 2017 by annseque
!
hw-module profile tcam format access-list ipv4 dst-addr dst-port proto port-range
enable-set-ttl ttl-match
!
ipv4 access-list acl-v4
 10 deny tcp any any ttl eq 100
 20 permit tcp any any ttl range 1 50 set ttl 200
 30 permit tcp any any ttl neq 100 set ttl 255
!
interface Te0/0/0/0
 ipv4 address 15.1.1.1 255.255.255.0
 ipv4 access-group acl-v4 ingress
!
```

インターフェイス ベースの一意の IPv4 ACL の設定

インターフェイス間で共有され、同じ TCAM スペースを使用する ACL は共有 ACL と呼ばれています。ただし、設定できる一意の共有 ACL は 31 個のみです。それ以上の ACL を設定するには、**interface-based** コマンドを使用して ACL 共有を無効にする必要があります。ACL をインターフェイスに一意にすることによって、31 個を超える ACL を作成できます。

```
/* Enable interface-based, unique IPv4 ACLs */
Router(config)# hw-module profile tcam format access-list ipv4 src-addr src-port dst-addr
 dst-port interface-based
```

残りの ACL 設定は、「設定」の項の説明と同じです。

IPv6 ACL の TTL の照合の設定

IPv6 ヘッダーに指定されている TTL 値で照合するように ACL を設定できます。TTL 一致条件を単一の値か、または複数の値に基づくように指定できます。また、**set ttl** コマンドを使用して、IPv6 ヘッダー内の TTL 値を書き換えることもできます。

IPv6 ACL での TTL の照合の使用に関する制限

IPv6 ACL に TTL の照合を使用するには、次の制限があります。

- TTL の照合は、入力 ACL でのみサポートされています。
- **set ttl** コマンドを使用した TTL の書き換えは、ACL ロギングと併用できません。
- TTL の書き換えを IP-in-IP ヘッダーの外側の IPv6 ヘッダーに適用する場合は、GRE カプセル化解除により外側の IPv6 ヘッダーのカプセル化が解除されると、内部 IPv6 ヘッダーにも TTL の書き換えが適用されます。

設定

IPv6 ACL に TTL の照合を設定するには、次のステップを実行します。

```
/* Enable TTL matching in the global configuration mode by using the hw-module command
*/
Router(config)# hw-module profile tcam format access-list ipv6 dst-addr dst-port
enable-set-ttl ttl-match

/* Configure an IPv6 ACL with the TTL parameters */
Router(config)# ipv4 access-list acl-v6
Router(config-ipv4-acl)# 10 deny tcp any any ttl eq 50
Router(config-ipv4-acl)# 20 permit tcp any any ttl lt 50 set ttl 255
Router(config-ipv4-acl)# 30 permit tcp any any ttl gt 50 set ttl 200
Router(config-ipv4-acl)# commit
Thu Nov  2 12:22:58.948 IST

/* Attach the IPv6 ACL to the GigE interface */
Router(config)# interface Te0/0/0/0
Router(config-if)# ipv6 address 2001:2:1::1/64
Router(config-if)# ipv6 access-group acl-v6 ingress
Router(config-if)# commit
```

実行コンフィギュレーション

show run コマンドを使用して設定を検証します。

```
Router(config)# show run
Thu Nov  2 14:01:53.376 IST
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Thu Nov  2 12:22:59 2017 by annseque
!
hw-module profile tcam format access-list ipv6 dst-addr dst-port enable-set-ttl ttl-match
!
ipv4 access-list acl-v6
 10 deny tcp any any ttl eq 50
20 permit tcp any any ttl lt 50 set ttl 255
30 permit tcp any any ttl gt 50 set ttl 200
!
interface Te0/0/0/0
 ipv6 address 2001:2:1::1/64
 ipv4 access-group acl-v6 ingress
!
```

インターフェイス ベースの一意の IPv6 ACL の設定

インターフェイス間で共有され、同じ TCAM スペースを使用する ACL は共有 ACL と呼ばれています。ただし、設定できる一意の共有 ACL は 31 個のみです。それ以上の ACL を設定するには、**interface-based** コマンドを使用して ACL 共有を無効にする必要があります。ACL をインターフェイスに一意にすることによって、31 個を超える ACL を作成できます。

```
/* Enable interface-based, unique IPv6 ACLs */
Router(config)# hw-module profile tcam format access-list ipv6 src-addr dst-addr dst-port
 interface-based
```

残りの ACL 設定は、「設定」の項の説明と同じです。

IP アクセス リスト ロギング メッセージの概要

Cisco IOS XR ソフトウェアでは、標準 IP アクセス リストで許可または拒否されたパケットに関するログ メッセージが表示されます。つまり、パケットがアクセス リストに一致すると、そのパケットに関するログメッセージ情報がコンソールに送信されます。ログをコンソールに送信するメッセージのレベルは、グローバル コンフィギュレーション モードの **logging console** コマンドで制御します。

最初にパケットがアクセスリストをトリガーすると、すぐにログメッセージが生成されます。その後、5分間隔でパケットが収集されて表示または記録されます。ログメッセージにはアクセス リスト番号、パケットの許可または拒否に関する状況、パケットの送信元 IP アドレス、および直前の 5 分間に許可または拒否された送信元からのパケット数が示されます。

ただし、{**ipv4 | ipv6**} **access-list log-update threshold** コマンドを使用して、アクセス リストに一致する場合（さらに許可または拒否される場合）に、システムでログメッセージを生成するパケットの数を設定できます。この手順は、5分間隔よりも短い頻度でログメッセージを受信する場合に実行することを推奨します。

注意 *number-of-matches* 引数を 1 に設定すると、ログ メッセージはキャッシュされずにただちに送信されます。この場合、アクセス リストに一致するすべてのパケットについてログ メッセージが生成されます。大量のログ メッセージでシステムが過負荷になる可能性があるため、1 に設定することは推奨されません。

{ipv4 | ipv6} access-list log-update threshold コマンドを使用する場合でも、5 分タイマーは有効なままなので、各キャッシュのメッセージ数に関係なく、5 分が経過すると各キャッシュは空になります。ログメッセージを送信するタイミングに関係なく、しきい値が指定されていない場合と同様に、ログメッセージのキャッシュは消去され、カウントは0にリセットされます。

（注） ログ メッセージが多すぎて処理できない場合や、1 秒以内に 2 つ以上のログ メッセージを処理した場合には、ログ メッセージ パケットの一部がドロップされることがあります。この動作により、ログを生成するパケットの数が多くなっても、ルータが CPU サイクルを過度に使用することはありません。したがって、ロギング機能は課金ツールや、アクセス リストとの一致数を正確に把握するための情報源として使用しないでください。

プレフィックス リストの概要

プレフィックス リストはルート マップおよびルート フィルタリング操作に使用されるほか、ボーダー ゲートウェイ プロトコル（BGP）の多くのルート フィルタリング コマンドではアクセス リストの代わりに使用できます。プレフィックスは IP アドレスの一部であり、左端のオクテットの左端のビットから始まります。アドレスの何ビットがプレフィックスに属するかを正確に指定すると、プレフィックスを使用してアドレスを集約し、そのアドレスに対して再配布（フィルタ ルーティング アップデート）などの機能を実行できるようになります。

プレフィックス リストを使用した BGP フィルタリング

プレフィックス リストは、BGP ルート フィルタリング コマンドの多くでアクセス リストの代わりに使用できます。これは BGP プロトコルのグローバル コンフィギュレーションで設定されます。プレフィックス リストを使用した場合の利点は次のとおりです。

- サイズの大きなリストをロードしてルート ルックアップを実施する場合のパフォーマンスが大幅に向上します。
- 差分更新がサポートされます。
- CLI の使い勝手が向上します。アクセス リストを使用して BGP 更新をフィルタリングするための CLI は、パケット フィルタリング形式を使用しているため、わかりにくく使い勝手もよくありません。
- 柔軟性が高まります。

コマンドでプレフィックス リストを使用するには、あらかじめプレフィックス リストをセットアップしておく必要があります。プレフィックスリストのエントリには、シーケンス番号を割り当ててください。

プレフィックス リストでトラフィックをフィルタリングする仕組み

プレフィックスリストによるフィルタリングでは、ルートのプレフィックスが、プレフィックスリストに記載されているプレフィックスと照合されます。一致すると、一致したルートが使用されます。具体的には、プレフィックスを許可するか、拒否するかは次のルールに基づきます。

- 空のプレフィックス リストはすべてのプレフィックスを許可します。
- 特定のプレフィックスがプレフィックスリストのどのエントリとも一致しなかった場合、暗黙の deny が適用されます。
- プレフィックスリストの複数のエントリが特定のプレフィックスと一致したときは、最も長く、最も具体的な一致が選択されます。

シーケンス番号は自動的に生成されます。ただし、この自動生成をディセーブルにしている場合を除きます。シーケンス番号の自動生成をディセーブルにしている場合は、IPv4のプレフィックス リスト コンフィギュレーション コマンドで、**permit** と **deny** コマンドの *sequence-number* 引数を使用して、各エントリのシーケンス番号を指定する必要があります。プレフィックスリストのエントリを削除するには、permit または deny コマンドの **no** 形式を使用して、*sequence-number* 引数を指定します。

show コマンドの出力には、シーケンス番号が含まれます。

プレフィックス リストの設定

設定例

「Deny all routes with a prefix of 10/8」というコメントが付いたプレフィックス リスト「pfx_2」を作成します。このプレフィックス リストは、128.0.0.0/8 の /24 に一致するプレフィックスをすべて拒否します。

```
Router#configure
Router(config)#ipv4 prefix-list pfx_2


Router(config-ipv4_pfx)#10 remark Deny all routes with a prefix of 10/8
Router(config-ipv4_pfx)#20 deny 128.0.0.0/8 eq 24
/* Repeat the above step as necessary. Use the no sequence-number command to delete an
entry. */

Router(config-ipv4_pfx)#commit
```

実行コンフィギュレーション

```
Router#show running-config ipv4 prefix-list pfx_2
ipv4 prefix-list pfx_2
```

```
 10 remark Deny all routes with a prefix of 10/8
 20 deny 128.0.0.0/8 eq 24
!
```

確認

許可とコメントの設定が設定されているコンフィギュレーションに合致していることを確認します。

```
Router# show prefix-list pfx_2
ipv4 prefix-list pfx_2
 10 remark Deny all routes with a prefix of 10/8
 20 deny 128.0.0.0/8 eq 24
RP/0/RP0/CPU0:ios#
```

関連コマンド

プレフィックスリストエントリの順序付けとプレフィックス リストの変更

設定例

名前付きプレフィックス リストのエントリにシーケンス番号を割り当て、プレフィックス リストに対してエントリを追加または削除する方法を示します。プレフィックス リストを変更することを前提に説明します。プレフィックス リストの並べ替えは任意です。

```
Router#config
Router(config)#ipv4 prefix-list cl_1


Router(config)#10 permit 172.16.0.0 0.0.255.255
/* Repeat the above step as necessary adding statements by sequence number where you
planned; use the no sequence-number command to delete an entry */

Router(config)#commit
end
Router#resequence prefix-list ipv4 cl_1 20 15
```

実行コンフィギュレーション

```
/*Before resequencing/*
Router#show running-config ipv4 prefix-list cl_1
ipv4 prefix-list cl_1
 10 permit 172.16.0.0/16
!
/* After resequencing using the resequence prefix-list ipv4 cl_1 20 15 command: */
Router#show running-config ipv4 prefix-list cl_1
ipv4 prefix-list cl_1
 35 permit 172.16.0.0/16
!
```

確認

プレフィックス リストが並べ替えられたことを確認します。

```
Router#show prefix-list cl_1
ipv4 prefix-list cl_1
 35 permit 172.16.0.0/16
```

関連コマンド