Cisco IOS による ESP および ISAKMP パケットでのポリシー ルーティングとその影響

内容

概要

このドキュメントでは、Cisco IOS® を使用するときに、Encapsulating Security Payload（ESP）および Internet Security Association and Key Management Protocol（ISAKMP）のパケットに適用する際のポリシー ベース ルーティング（PBR）およびローカル PBR の効果について説明します。

著者：Cisco TAC エンジニア、Michal Garcarz

前提条件

要件

次の項目に関する基本的な知識が推奨されます。

- Cisco IOS
- Cisco IOS の VPN 設定

使用するコンポーネント

このドキュメントの情報は、Cisco IOS バージョン 15.x に基づくものです。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、初期（デフォルト）設定の状態から起動しています。対象のネットワークが実稼働中である場合には、どのようなコマンドについても、その潜在的な影響について確実に理解しておく必要があります。

背景説明

IPSec トンネルを確立する前に、ルータは ISAKMP 交換を開始します。これらのパケットがルータによって生成されると、パケットはローカルで生成されたトラフィックとして扱われ、ローカル PBR の決定が適用されます。また、ルータ（Enhanced Interior Gateway Routing Protocol（EIGRP）、Next Hop Resolution Protocol（NHRP）、Border Gateway Protocol（BGP）、または Internet Control Message Protocol（ICMP））によって生成されたパケットはすべて、ローカルで生成されたトラフィックと見なされ、ローカル PBR の決定が適用されます。

ルータによって転送され、トンネルを通過して送信されるトラフィック（トランジット トラフィックと呼ばれます）は、ローカルで生成されたトラフィックとは見なされません。ルータの入力インターフェイスで必要なルーティング ポリシーを適用する必要があります。

トンネルを通過するトラフィックであるということは、次のことを意味します。つまり、ローカルで生成されたトラフィックは PBR に従いますが、トランジット トラフィックは従いません。この記事では、動作におけるこの違いの結果について説明します。

ESP でカプセル化する必要のあるトランジット トラフィックでは、ESP のカプセル化の前後に PBR がパケットの出力インターフェイスを決定するため、ルーティング エントリは必要ありません。ESP でカプセル化する必要のあるローカルで生成されたトラフィックの場合、ローカル PBR はカプセル化の前にしかパケットの出力インターフェイスを決定せず、ルーティングがカプセル化後のパケットの出力インターフェイスを決定するため、ルーティング エントリが必要になります。

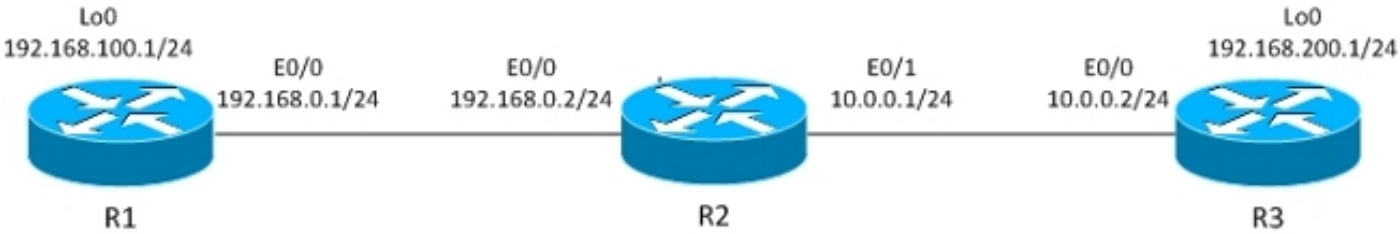
このドキュメントでは、2 つの ISP リンクが設定された 1 台のルータを使用する一般的な設定例を示します。1 つ目のリンクはインターネットにアクセスするために使用され、2 つ目は VPN 用です。いずれかのリンクで障害が発生した場合、トラフィックは別のインターネット サービス プロバイダー（ISP）リンクを使用して再ルーティングされます。欠点もあります。

PBR は Cisco Express Forwarding（CEF）で実行され、ローカル PBR はプロセススイッチングされることに注意してください。

ルータのローカルで生成されるトラフィック

このセクションでは、ルータ(R)1から開始されるトラフィックの動作について説明します。トラフィックはR1によってカプセル化されたESPです。

トポロジ



IPSec LAN-to-LAN トンネルは、R1 と R3 の間で構築されます。

対象トラフィックは、R1 Lo0（192.168.100.1）と R3 Lo0（192.168.200.1）の間です。

ルータ R3 には R2 へのデフォルト ルートがあります。

R1 にはルーティング エントリはなく、直接接続されたネットワークしかありません。

コンフィギュレーション

R1 にはすべてのトラフィック用のローカル PBR があります。

```
interface Loopback0
 ip address 192.168.100.1 255.255.255.0
!
interface Ethernet0/0
 ip address 192.168.0.1 255.255.255.0
 crypto map CM

track 10 ip sla 10
ip sla 10
 icmp-echo 192.168.0.2 source-ip 192.168.0.1

route-map LOCALPBR permit 10
 set ip next-hop verify-availability 192.168.0.2 1 track 10
ip local policy route-map LOCALPBR
```

デバッグ

R1 でローカルに生成されるすべてのトラフィックは、アップ状態になると R2 に送信されます。

トンネルを起動すると発生する事柄を確認するには、対象のトラフィックをルータ自体から送信

します。

```
R1#debug ip packet
R1#ping 192.168.200.1 source lo0
```

> **注意**：debug ip packet コマンドによって大量のデバッグが生成される場合があり、CPU 使用率に大きく影響します。注意して使用してください。

また、このデバッグでは、デバッグによって処理されるトラフィック量を制限するために、アクセス リストを使用することもできます。debug ip packet コマンドは、プロセススイッチングされるトラフィックのみを表示します。

以下は R1 のデバッグです。

```
IP: s=192.168.100.1 (local), d=192.168.200.1 (Ethernet0/0), len 100, local
feature, Policy Routing(3), rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk
FALSE
IP: s=192.168.100.1 (local), d=192.168.200.1 (Ethernet0/0), len 100, sending
IP: s=192.168.100.1, d=192.168.200.1, pak EF6E8F28 consumed in output feature,
packet consumed, IPSec output classification(30), rtype 2, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, local feature, Policy
Routing(3), rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, sending
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
IPSec output classification(30), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
IPSec: to crypto engine(64), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
Post-encryption output features(65), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, post-encap feature,
(1), rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, post-encap feature,
FastEther Channel(3), rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, sending full packet
```

次のことが起きます。

対象トラフィック（192.168.100.1 > 192.168.200.1）は、ローカル PBR によってマッチングされ、出力インターフェイスが決定されます（E0/0）。 このアクションによって暗号コードがトリガーされ、ISAKMP が開始されます。そのパケットはローカル PBR によってポリシー ルーティングされます。これにより、出力インターフェイス（E0/0）が決定されます。 ISAKMP トラフィックが送信され、トンネルがネゴシエートされます。

ping をもう一度実行すると何が起きるでしょうか。

```
R1#show crypto session
Crypto session current status

Interface: Ethernet0/0
Session status: UP-ACTIVE
Peer: 10.0.0.2 port 500
 IKEv1 SA: local 192.168.0.1/500 remote 10.0.0.2/500 Active
```

```
 IPSEC FLOW: permit ip host 192.168.100.1 host 192.168.200.1
       Active SAs: 2, origin: crypto map

R1#ping 192.168.200.1 source lo0 repeat 1

IP: s=192.168.100.1 (local), d=192.168.200.1 (Ethernet0/0), len 100, local
feature, Policy Routing(3), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.100.1 (local), d=192.168.200.1 (Ethernet0/0), len 100, sending
IP: s=192.168.100.1 (local), d=192.168.200.1 (Ethernet0/0), len 100, output
feature, IPSec output classification(30), rtype 2, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
IP: s=192.168.100.1, d=192.168.200.1, pak EEB40198 consumed in output feature,
packet consumed, IPSec: to crypto engine(64), rtype 2, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, output feature,
IPSec output classification(30), rtype 1, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, output feature,
IPSec: to crypto engine(64), rtype 1, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, output feature,
Post-encryption output features(65), rtype 1, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), g=10.0.0.2, len 172,
forward
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, post-encap
feature, (1), rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, post-encap
feature, FastEther Channel(3), rtype 0, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 172, encapsulation
failed.
Success rate is 0 percent (0/1)
```

次のことが起きます。

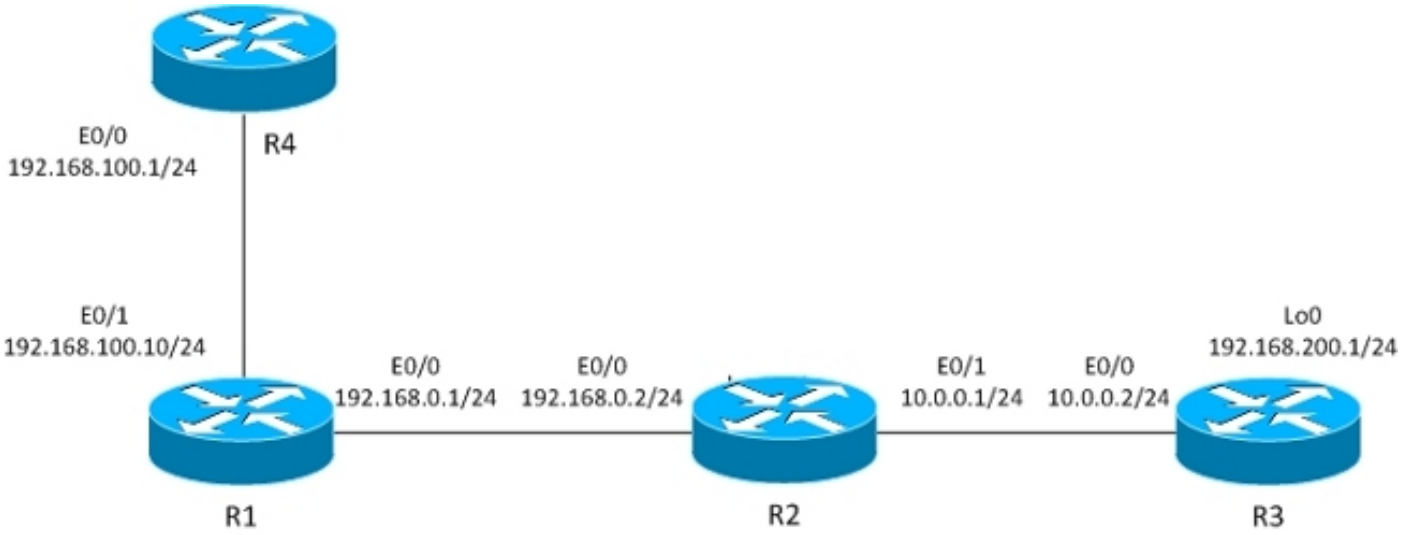
対象トラフィック（192.168.100.1 > 192.168.200.1）は、ローカル ポリシー ルーティングにより生成され、出力インターフェイスが決定されます（E0/0）。 パケットは E0/0 の IPSec 出力機能によって消費され、カプセル化されます。カプセル化パケット（192.168.0.1 から 10.0.0.2）では、出力インターフェイスを決定するためルーティングがチェックされますが、R1 のルーティング テーブルには何もありません。これは、カプセル化が失敗する理由です。

このシナリオでは、トンネルはアップ状態ですが、トラフィックは送信されません。それは、出力インターフェイスを決定するために、ESP のカプセル化の後で Cisco IOS がルーティング テーブルをチェックするためです。

ルータ経由のトランジット トラフィック

このセクションでは、ルータを経由するトランジット トラフィックの動作について説明します。このトラフィックはルータによって ESP でカプセル化されます。

トポロジ



R1 と R3 の間に L2L トンネルが構築されています。

対象トラフィックは、R4 Lo0（192.168.100.1）と R3 Lo0（192.168.200.1）の間です。

ルータ R3 には R2 へのデフォルト ルートがあります。

ルータ R4 には R1 へのデフォルト ルートがあります。

R1 にはルーティングはありません。

コンフィギュレーション

前のトポロジは、ルータが暗号化のパケット（ローカルで生成されたトラフィックではなく、トランジット トラフィック）を受信するとフローを表示するように変更されます。

これで、R4 から受信する対象トラフィックは R1 にポリシー ルーティングされます（E0/1 上の PBR による）。また、すべてのトラフィックのためのローカル ポリシー ルーティングもあります。

```
interface Ethernet0/1
 ip address 192.168.100.10 255.255.255.0
 ip policy route-map PBR

route-map LOCALPBR permit 10
 set ip next-hop verify-availability 192.168.0.2 1 track 10
!
route-map PBR permit 10
 set ip next-hop verify-availability 192.168.0.2 1 track 10

ip local policy route-map LOCALPBR
```

デバッグ

R1 でトンネルを起動すると（対象トラフィックを R4 から受信した後で）発生する事柄を確認するには、次を入力します。

```
R1#debug ip packet
```

R4#**ping 192.168.200.1**
以下は R1 のデバッグです。

IP: **s=192.168.100.1 (Ethernet0/1), d=192.168.200.1 (Ethernet0/0)**, len 100,
input feature, **Policy Routing(68)**, rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.100.1 (Ethernet0/1), d=192.168.200.1 (Ethernet0/0), len 100,
input feature, MCI Check(73), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.100.1, d=192.168.200.1, pak EEB4A9D8 consumed in output feature,
packet consumed, IPSec output classification(30), rtype 2, forus FALSE,
sendself FALSE, mtu 0, fwdchk FALSE
IP: **s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0)**, len 192, **local feature,
Policy Routing(3)**, rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, sending
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
IPSec output classification(30), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
IPSec: to crypto engine(64), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, output feature,
Post-encryption output features(65), rtype 2, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, post-encap
feature, (1), rtype 2, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0), len 192, post-encap
feature, FastEther Channel(3), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: **s=192.168.0.1 (local), d=10.0.0.2 (Ethernet0/0)**, len 192, **sending full
packet**
次のことが起きます。

対象トラフィックは E0/0 の PBR にヒットし、ISAKMP パケットを送信する暗号コードをトリガーします。その ISAKMP パケットはローカルでポリシー ルーティングされ、出力インターフェイスはローカル PBR によって決定されます。トンネルが構築されます。

次のように、もう一度 R4 から 192.168.200.1 への ping を実行します。

R4#**ping 192.168.200.1**
以下は R1 のデバッグです。

IP: **s=192.168.100.1 (Ethernet0/1), d=192.168.200.1 (Ethernet0/0)**, len 100,
input feature, Policy Routing(68), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.100.1 (Ethernet0/1), d=192.168.200.1 (Ethernet0/0), len 100,
input feature, MCI Check(73), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.100.1 (Ethernet0/1), d=192.168.200.1 (Ethernet0/0), len 100,
output feature, IPSec output classification(30), rtype 2, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
IP: **s=192.168.100.1, d=192.168.200.1**, pak EF722068 consumed in output feature,
packet consumed, IPSec: to crypto engine(64), rtype 2, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
IP: **s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0),** len 172, **input
feature, Policy Routing(68)**, rtype 2, forus FALSE, sendself FALSE, mtu 0,

```
fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, input
feature, MCI Check(73), rtype 2, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, output
feature, IPSec output classification(30), rtype 2, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, output
feature, IPSec: to crypto engine(64), rtype 2, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, output
feature, Post-encryption output features(65), rtype 2, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), g=192.168.0.2, len
172, forward
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, post-encap
feature, (1), rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172, post-encap
feature, FastEther Channel(3), rtype 0, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
IP: s=192.168.0.1 (Ethernet0/1), d=10.0.0.2 (Ethernet0/0), len 172,
sending full packet
```

次のことが起きます。

対象トラフィックは E0/0 の PBR にヒットし、その PBR は出力インターフェイス (E0/0) を決定します。 E0/0 では、パケットが IPsec によって消費され、カプセル化されます。カプセル化されたパケットが同じ PBR ルールに対してチェックされ、出力インターフェイスが決定された後に、パケットが正しく送受信されます。

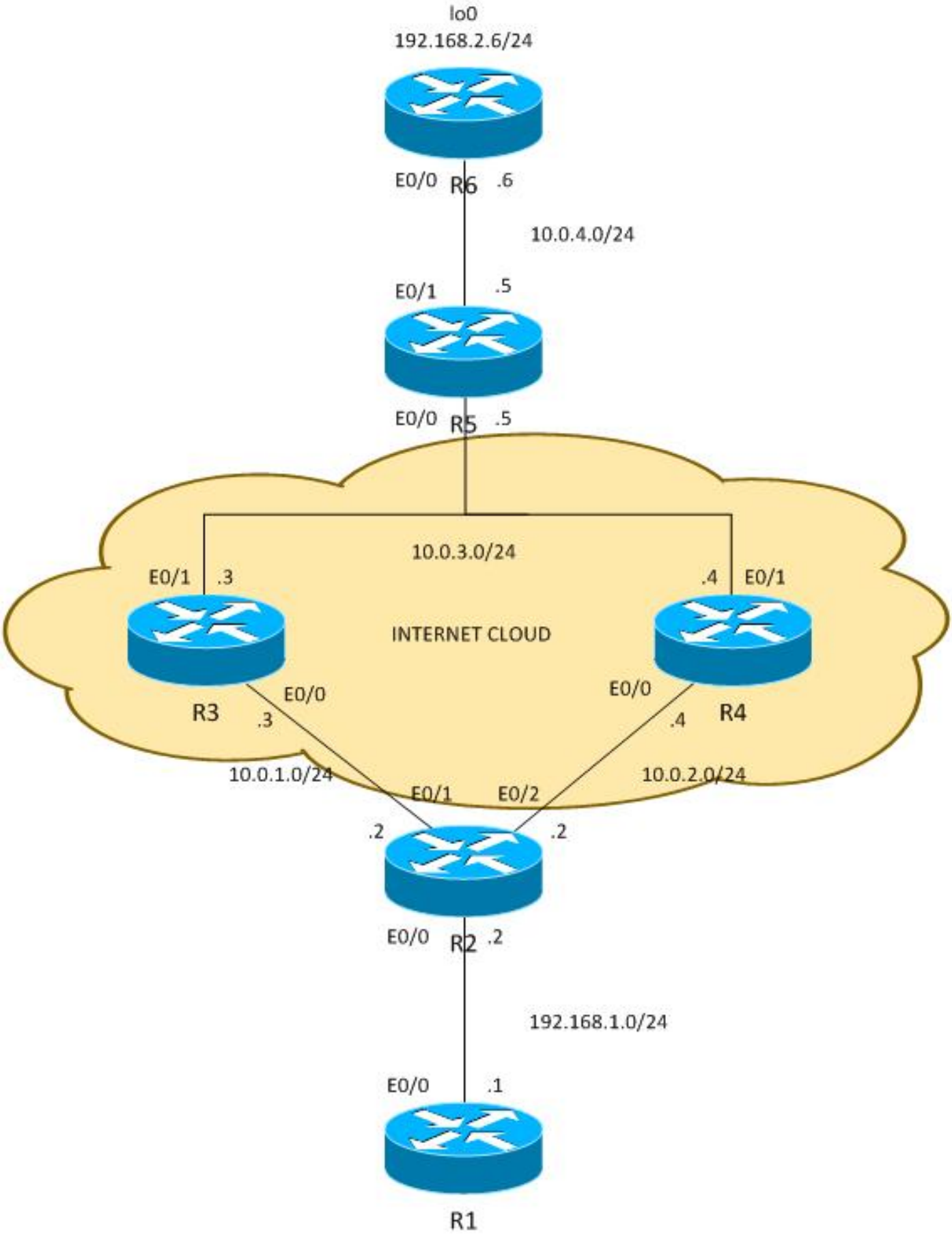
動作の違いの概要

ローカルで生成されたトラフィックの場合、カプセル化されないトラフィック (ISAKMP) の出力インターフェイスはローカル PBR によって決定されます。ローカルで生成されたトラフィックでは、カプセル化後のトラフィック (ESP) の出力インターフェイスはルーティング テーブルで決定されます (ローカル PBR はチェックされません) 。 トランジット トラフィックの場合、カプセル化後のトラフィック (ESP) の出力インターフェイスは、インターフェイス PBR によって決定されます (カプセル化の前後で 2 回) 。

設定例

以下は、PBR と VPN を使用するローカル PBR で発生する可能性のある問題を示す実際の設定例です。R2 (CE) には 2 つの ISP リンクがあります。R6 ルータにも CE と 1 つの ISP リンクがあります。R2からR3への最初のリンクは、R2のデフォルトルートとして使用されます。R4への2番目のリンクは、R6へのVPNトラフィックにのみ使用されます。ISPリンクに障害が発生した場合、トラフィックは他のリンクに再ルーティングされます。

トポロジ



コンフィギュレーション

192.168.1.0/24 と 192.168.2.0/24 間のトラフィックは保護されています。Open Shortest Path First（OSPF）は、10.0.0.0/8 アドレスをアドバタイズするためにインターネット クラウドで使

用されます。このアドレスは、カスタマーに対して ISP によって割り当てられたパブリック アドレスとして処理されます。実際には、OSPF の代わりに BGP が使用されます。

R2 と R6 の設定は暗号マップに基づいています。R2 では、アップ状態の場合は、R4 に VPN トラフィックを送信するために E0/0 の PBR が使用されます。

```
route-map PBR permit 10
 match ip address cmap
 set ip next-hop verify-availability 10.0.2.4 1 track 20

ip access-list extended cmap
 permit ip 192.168.1.0 0.0.0.255 192.168.2.0 0.0.0.255

crypto map cmap 10 ipsec-isakmp
 set peer 10.0.4.6
 set transform-set TS
 match address cmap

interface Ethernet0/0
 ip address 192.168.1.2 255.255.255.0
 ip nat inside
 ip virtual-reassembly in
 ip policy route-map PBR
```

以下では、ローカル PBR が不要であることが分かります。インターフェイスPBRは対象トラフィックを10.0.2.4にルーティングします。これにより、ルーティングがR3を経由するリモートピアを指している場合でも、暗号コードがトリガーされ、正しいインターフェイス (R4へのリンク) からISAKMPが開始されます。

R6 では、VPN の 2 つのピアが使用されます。

```
crypto map cmap 10 ipsec-isakmp
 set peer 10.0.2.2             !primary
 set peer 10.0.1.2
 set transform-set TS
 match address cmap
```

R2はIP Service Level Agreement (SLA ; サービスレベル契約) を使用してR3とR4をpingします。デフォルトルートはR3です。R3に障害が発生した場合は、R4を選択します。

```
ip sla 10
 icmp-echo 10.0.1.3
ip sla schedule 10 life forever start-time now
ip sla 20
 icmp-echo 10.0.2.4
ip sla schedule 20 life forever start-time now

track 10 ip sla 10
track 20 ip sla 20

ip route 0.0.0.0 0.0.0.0 10.0.1.3 track 10
ip route 0.0.0.0 0.0.0.0 10.0.2.4 100
```

また、R2 はすべての内部ユーザに対してインターネット アクセスを許可します。R3 への ISP がダウンした場合に冗長性を実現するには、ルート マップが必要です。これは、内部トラフィックを別の出力インターフェイスにポート アドレス変換 (PAT) します (R3 がアップ状態で、デフォルト ルートが R3 を指している場合 E0/1 インターフェイスに PAT します。また R3 がダウン状態で、R4 がデフォルト ルートとして使用されている場合 E0/2 インターフェイスに PAT しま

す）。

```
ip access-list extended pat
 deny   ip 192.168.1.0 0.0.0.255 192.168.2.0 0.0.0.255
 deny   udp any any eq isakmp
 deny   udp any eq isakmp any
 permit ip any any

route-map RMAP2 permit 10
 match ip address pat
 match interface Ethernet0/2
!
route-map RMAP1 permit 10
 match ip address pat
 match interface Ethernet0/1

ip nat inside source route-map RMAP1 interface Ethernet0/1 overload
ip nat inside source route-map RMAP2 interface Ethernet0/2 overload

interface Ethernet0/0
 ip address 192.168.1.2 255.255.255.0
 ip nat inside
 ip virtual-reassembly in
 ip policy route-map PBR

interface Ethernet0/1
 ip address 10.0.1.2 255.255.255.0
 ip nat outside
 ip virtual-reassembly in
 crypto map cmap

interface Ethernet0/2
 ip address 10.0.2.2 255.255.255.0
 ip nat outside
 ip virtual-reassembly in
 crypto map cmap
```

ISAKMP を実行する際に、VPN トラフィックは変換から除外される必要があります。ISAKMP トラフィックが変換から除外されない場合、R3 に向かう外部インターフェイスに PAT されます。

```
R2#show ip nat translation
Pro Inside global      Inside local       Outside local      Outside global
udp 10.0.1.2:500       10.0.2.2:500       10.0.4.6:500       10.0.4.6:500

*Jun  8 09:09:37.779: IP: s=10.0.2.2 (local), d=10.0.4.6, len 196, local
feature, NAT(2), rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.2.2 (local), d=10.0.4.6 (Ethernet0/1),
len 196, sending
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, Post-routing NAT Outside(24), rtype 1, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, Common Flow Table(27), rtype 1, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, Stateful Inspection(28), rtype 1, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, IPSec output classification(34), rtype 1, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
```

```
output feature, NAT ALG proxy(59), rtype 1, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, IPSec: to crypto engine(75), rtype 1, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
output feature, Post-encryption output features(76), rtype 1, forus FALSE, sendself
FALSE, mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
pre-encap feature, IPSec Output Encap(1), rtype 1, forus FALSE, sendself FALSE,
mtu 0, fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
 pre-encap feature, Crypto Engine(3), rtype 1, forus FALSE, sendself FALSE, mtu 0,
fwdchk FALSE
*Jun  8 09:09:37.779: IP: s=10.0.1.2 (local), d=10.0.4.6 (Ethernet0/1), len 196,
sending full packet
```

Testing

この設定には、完全な冗長性があります。VPNはR4リンクを使用し、残りのトラフィックはR3でルーティングされます。R4に障害が発生した場合、VPNトラフィックはR3リンクで確立されます（PBRのルートマップが一致せず、デフォルトルーティングが使用されます）。

R4 への ISP がダウンする前に、R6 はピア 10.0.2.2 からのトラフィックを確認します。

```
R6#show crypto session
Crypto session current status

Interface: Ethernet0/0
Session status: UP-ACTIVE
Peer: 10.0.2.2 port 500
 IKEv1 SA: local 10.0.4.6/500 remote 10.0.2.2/500 Active
 IPSEC FLOW: permit ip 192.168.2.0/255.255.255.0 192.168.1.0/255.255.255.0
       Active SAs: 2, origin: crypto map
```

R2 が VPN トラフィックに R3 への ISP を使用した後、R6 はピア 10.0.1.2 からのトラフィックを確認します。

```
R6#show crypto session
Crypto session current status

Interface: Ethernet0/0
Session status: UP-ACTIVE
Peer: 10.0.1.2 port 500
 IKEv1 SA: local 10.0.4.6/500 remote 10.0.1.2/500 Active
 IPSEC FLOW: permit ip 192.168.2.0/255.255.255.0 192.168.1.0/255.255.255.0
       Active SAs: 2, origin: crypto map
```

他方のシナリオでは、R3 へのリンクがダウンしても、すべて正常に動作します。VPNトラフィックは引き続きR4へのリンクを使用します。外部アドレスを適切にするため、ネットワークアドレス変換(NAT)は192.168.1.0/24からPATに対して実行されます。R3 がダウンする前に、10.0.1.2 に変換されます。

```
R2#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 10.0.1.2:1        192.168.1.1:1      10.0.4.6:1         10.0.4.6:1
```

R3 がダウンした後、R4 へのリンクを使用する新しい変換（10.0.2.2 への変換）とともに古い変

換が残ります。

```
R2#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
icmp 10.0.2.2:0        192.168.1.1:0      10.0.4.6:0         10.0.4.6:0
icmp 10.0.1.2:1        192.168.1.1:1      10.0.4.6:1         10.0.4.6:1
```

対応策

すべてが正常に動作する場合、欠点はどこにありますか。以下に詳細を示します。

ローカルで生成されるトラフィック

以下のシナリオでは、R2 自体から VPN トラフィックを開始する必要があります。このシナリオでは、R2 に ISAKMP トラフィックを R4 を介して送信させるように、またトンネルがアップ状態になるように、R2 のローカル PBR を設定する必要があります。ただし、ルーティング テーブルを使用して (デフォルトでは R3 を指しています) 出力インターフェイスが決定されます。また、VPN の転送に使用されるパケットは R4 ではなく R3 に送信されます。これを確認するには、次を入力します。

```
ip access-list extended isakmp
 permit udp any any eq isakmp
 permit udp any eq isakmp any
 permit icmp any any

route-map LOCAL-PBR permit 10
 match ip address isakmp
 set ip next-hop verify-availability 10.0.2.4 1 track 20

ip local policy route-map LOCAL-PBR
```

この例では、ローカルで生成されるインターネット制御メッセージプロトコル(ICMP)はR4を介して強制されます。そうしないと、192.168.1.2から192.168.2.5にローカルで生成されるトラフィックがルーティングテーブルを使用して処理され、R3とのトンネルが確立されます。

この設定を適用するとどうなりますか。192.168.1.2から192.168.2.5へのICMPパケットはR4に向けて送信され、トンネルはR4へのリンクを使用して開始されます。トンネルは設定されます。

```
R2#ping 192.168.2.6 source e0/0 repeat 10
Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 192.168.2.6, timeout is 2 seconds:
Packet sent with a source address of 192.168.1.2
.!!!!!!!!!
Success rate is 90 percent (9/10), round-trip min/avg/max = 4/4/5 ms

R2#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
```

```
Interface: Ethernet0/1
Session status: DOWN
Peer: 10.0.4.6 port 500 fvrf: (none) ivrf: (none)
     Desc: (none)
     Phase1_id: (none)
 IPSEC FLOW: permit ip 192.168.1.0/255.255.255.0 192.168.2.0/255.255.255.0
       Active SAs: 0, origin: crypto map
       Inbound:  #pkts dec"ed 0 drop 0 life (KB/Sec) 0/0
       Outbound: #pkts enc"ed 0 drop 0 life (KB/Sec) 0/0

Interface: Ethernet0/2
Uptime: 00:00:06
Session status: UP-ACTIVE
Peer: 10.0.4.6 port 500 fvrf: (none) ivrf: (none)
     Phase1_id: 10.0.4.6
     Desc: (none)
 IKEv1 SA: local 10.0.2.2/500 remote 10.0.4.6/500 Active
         Capabilities:(none) connid:1009 lifetime:23:59:53
 IKEv1 SA: local 10.0.2.2/500 remote 10.0.4.6/500 Inactive
         Capabilities:(none) connid:1008 lifetime:0
 IPSEC FLOW: permit ip 192.168.1.0/255.255.255.0 192.168.2.0/255.255.255.0
       Active SAs: 2, origin: crypto map
       Inbound:  #pkts dec"ed 9 drop 0 life (KB/Sec) 4298956/3593
       Outbound: #pkts enc"ed 9 drop 0 life (KB/Sec) 4298956/3593
```

すべてが正しく機能しているように見えます。トラフィックは正しいリンクE0/2を使用してR4に送信されます。R6でも、トラフィックが10.2.2.2から受信されていることがわかります。これはR4のリンクIPアドレスです。

```
R6#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation

Interface: Ethernet0/0
Uptime: 14:50:38
Session status: UP-ACTIVE
Peer: 10.0.2.2 port 500 fvrf: (none) ivrf: (none)
     Phase1_id: 10.0.2.2
     Desc: (none)
 IKEv1 SA: local 10.0.4.6/500 remote 10.0.2.2/500 Active
         Capabilities:(none) connid:1009 lifetime:23:57:13
 IPSEC FLOW: permit ip 192.168.2.0/255.255.255.0 192.168.1.0/255.255.255.0
       Active SAs: 2, origin: crypto map
       Inbound:  #pkts dec"ed 1034 drop 0 life (KB/Sec) 4360587/3433
       Outbound: #pkts enc"ed 1029 drop 0 life (KB/Sec) 4360587/3433
```

ただし実際には、**ESP パケットの非対称ルーティングが存在しています。**ESPパケットは送信元として10.0.2.2で送信されますが、R3へのリンクに配置されます。暗号化された応答はR4を通じて返されます。これは、R3とR4のカウンタをチェックすることで確認できます。

100 パケットを送信する前の E0/0 の R3 カウンタ：

```
R3#show int e0/0 | i pack
 5 minute input rate 0 bits/sec, 0 packets/sec
```

```
 5 minute output rate 0 bits/sec, 0 packets/sec
     739 packets input, 145041 bytes, 0 no buffer
    0 input packets with dribble condition detected
     1918 packets output, 243709 bytes, 0 underruns
```
100 パケットを送信した後の同じカウンタ：

```
R3#show int e0/0 | i pack
 5 minute input rate 0 bits/sec, 0 packets/sec
 5 minute output rate 0 bits/sec, 0 packets/sec
    839 packets input, 163241 bytes, 0 no buffer
    0 input packets with dribble condition detected
    1920 packets output, 243859 bytes, 0 underruns
```
着信パケットの数は（R2へのリンクで）100増加しましたが、発信パケットの数は2増加のみでした。そのため、R3は暗号化されたICMP工コーのみを確認します。

応答は、100 パケットを送信する前に、R4 に表示されます。

```
R4#show int e0/0 | i packet
 5 minute input rate 0 bits/sec, 0 packets/sec
 5 minute output rate 1000 bits/sec, 1 packets/sec
    793 packets input, 150793 bytes, 0 no buffer
    0 input packets with dribble condition detected
    1751 packets output, 209111 bytes, 0 underruns
```
100 パケットを送信した後：

```
R4#show int e0/0 | i packet
 5 minute input rate 0 bits/sec, 0 packets/sec
 5 minute output rate 0 bits/sec, 0 packets/sec
     793 packets input, 150793 bytes, 0 no buffer
    0 input packets with dribble condition detected
    1853 packets output, 227461 bytes, 0 underruns
```
R2に向けて送信されたパケットの数は102（暗号化されたICMP応答）増加し、受信されたパケットは0増加しました。したがって、R4は暗号化されたICMP応答のみを確認します。もちろん、パケット キャプチャによってこれは確認できます。

なぜ、このような現象が発生するのでしょうか。答えはこの記事の最初の部分にあります。

これらの ICMP パケットのフローを次に示します。

1. ローカル PBR のため、192.168.1.2 から 192.168.2.6 への ICMP は E0/2（R4 へのリンク）に配置されます。
2. ISAKMP セッションは 10.0.2.2 で作成され、予測どおり E0/2 リンクに配置されます。
3. カプセル化後のICMPパケットの場合、ルータは出力インターフェイスを決定する必要があります。これは、R3をポイントするルーティングテーブルを使用して行われます。このため、送信元が10.0.2.2（R4へのリンク）の暗号化パケットがR3を経由して送信されます。
4. R6 は 10.0.2.2 から ESP パケットを受信し（これは ISAKMP セッションと一致します）、パケットを復号化して、ESP 応答を 10.0.2.2 に送信します。
5. ルーティングのために、R5 は R4 を経由して 10.0.2.2 に応答を返します。
6. R2 はその応答を受信して復号化し、パケットを受け入れます。

これが、ローカルに生成されたトラフィックに普通以上の注意を払うべき理由です。

多くのネットワークでは、Unicast Reverse Path Forwarding(uRPF)が使用され、10.0.2.2を送信

元とするトラフィックがR3のE0/0でドロップされる可能性があります。その場合、pingは機能しません。

この問題の解決策はありますか。ローカルで生成されたトラフィックをトランジット トラフィックとして強制的にルータに処理させることが可能です。そのためには、ローカル PBR はトラフィックを、トランジット トラフィックのようにルーティングされる偽のループバック インターフェイスに転送する必要があります。

これは推奨されていません。

> **注**：PBR とともに NAT を使用する場合、特に注意してください（PAT アクセス リストの ISKMP トラフィックに関する前項を参照してください）。

PBR を使用しない設定例

また妥協的なもう一つの解決策があります。前の例と同じトポロジを使用して、PBR またはローカル PBR を使用せずにすべての要件を満たすことが可能です。このシナリオでは、ルーティングだけが使用されます。もう 1 つだけルーティング エントリが R2 に追加され、すべての PBR/ローカル PBR の設定が削除されます。

```
ip route 192.168.2.0 255.255.255.0 10.0.2.4 track 20
```

R2 全体で、このルーティング設定が使用されます。

```
ip route 0.0.0.0 0.0.0.0 10.0.1.3 track 10
ip route 0.0.0.0 0.0.0.0 10.0.2.4 100
ip route 192.168.2.0 255.255.255.0 10.0.2.4 track 20
```

最初のルーティング エントリは R3 へのデフォルト ルーティングになります（R3 へのリンクのアップ時）。2 番目のルーティング エントリは R4 へのデフォルト バックアップ ルートになります（R3 へのリンクのダウン時）。3 番目のエントリは、R4 リンクの状態に応じて、リモート VPN ネットワークへのトラフィックが送信される方法を決定します（R4 リンクがアップ状態の場合、リモート VPN ネットワークへのトラフィックは R4 経由で送信されます）。 この設定では、ポリシー ルーティングの必要はありません。

欠点は何ですか。PBR を使用したこれ以上の細かい制御はできません。送信元アドレスを決定することはできません。この場合、192.168.2.0/24 へのすべてのトラフィックは、送信元に関係なく R4 に向けて送信されます（アップ状態の場合）。前述の例では、これらは PBR と特定の送信元（192.168.1.0/24 が選択されていました）によって制御されていました。

どのシナリオにとって、この解決策は単純になりますか。複数の LAN ネットワーク（R2 の背後）の場合です。 ネットワークの一部がセキュアな方法（暗号化）で、それ意外のネットワークは非セキュアな方法（暗号化されていない）で 192.168.2.0/24 にアクセスする必要がある場合、セキュアではないネットワークからのトラフィックは R2 の E0/2 インターフェイスに配置され、暗号マップにはアクセスできません。したがって、暗号化されずに R4 へのリンクを経由して送信されます（主な要件は、暗号化されたトラフィックにのみ R4 を使用することです）。

このタイプのシナリオや要件はまれであるため、この解決策が広く使用されています。

要約

VPN や NAT とともに PBR および ローカル PBR を使用する場合、複雑になる可能性があるため、パケット フローの詳細を理解している必要があります。

ここで示されたようなシナリオでは、2 つの別個のルータ (各ルータに 1 つの ISP リンクを持つ) を使用することが推奨されています。ISP 障害が発生した場合、トラフィックは簡単に再ルーティングできます。PBR の場合、その必要がなく、全体の設計はよりシンプルになります。

また、PBR を使用する必要のない妥協的な解決策もありますが、代わりにスタティック フローティング ルーティングを使用します。

確認

現在、この設定に使用できる確認手順はありません。

トラブルシュート

現在、この設定に関する特定のトラブルシューティング情報はありません。

関連情報

- テクニカル サポートとドキュメント – Cisco Systems
- Cisco IOS 15.3 M&T- Cisco Systems