Policy Routing y su impacto en el ESP y paquetes ISAKMP con el Cisco IOS

Contenido

Introducción

Este documento describe el efecto del Routing basado en políticas (PBR) y del PBR local cuando está aplicado a los paquetes del Encapsulating Security Payload (ESP) y del Internet Security Association and Key Management Protocol (ISAKMP) cuando usted utiliza el ® del Cisco IOS.

Contribuido por Michal Garcarz, ingeniero de Cisco TAC.

Prerrequisitos

Requisitos

Cisco recomienda que usted tiene conocimiento básico de estos temas:

- IOS de Cisco
- Configuración VPN en el Cisco IOS

Componentes Utilizados

La información en este documento se basa en la versión deL Cisco IOS 15.x.

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si la red está funcionando, asegúrese de haber comprendido el impacto que puede tener cualquier comando.

Antecedentes

Antes del establecimiento del túnel IPsec, el router inicia un intercambio ISAKMP. Mientras que esos paquetes son generados por el router, se tratan los paquetes mientras que el tráfico generado a nivel local y cualquier decisión local PBR son aplicados. Además, cualquier paquete generado por el router (los ping del Enhanced Interior Gateway Routing Protocol (EIGRP), del Next Hop Resolution Protocol (NHRP), del Border Gateway Protocol (BGP), o del Internet Control Message Protocol (ICMP)) también se considera como tráfico generado a nivel local y tiene la decisión local PBR aplicada.

Trafique que es remitido por el router y enviado a través del túnel, que se llama tráfico de tránsito, no se considera tráfico generado a nivel local, y cualquier política de ruteo deseado se debe aplicar en la interfaz de ingreso del router.

Las implicaciones que esto tiene en el tráfico que atraviesa el túnel es que el tráfico generado a nivel local sigue el PBR, pero el tráfico de tránsito no hace. Este artículo explica las consecuencias de esta diferencia en el comportamiento.

Para el tráfico de tránsito que necesita ser ESP encapsulado, no hay necesidad de tener ningunas entradas de ruteo porque el PBR determina la interfaz de egreso para el paquete antes y después de la encapsulación ESP. Para el tráfico generado a nivel local que necesita ser ESP encapsulado, es necesario tener entradas de ruteo, porque el PBR local determina la interfaz de egreso solamente para el paquete antes de la encapsulación y la encaminamiento determina la interfaz de egreso para el paquete poste-encapsulado.

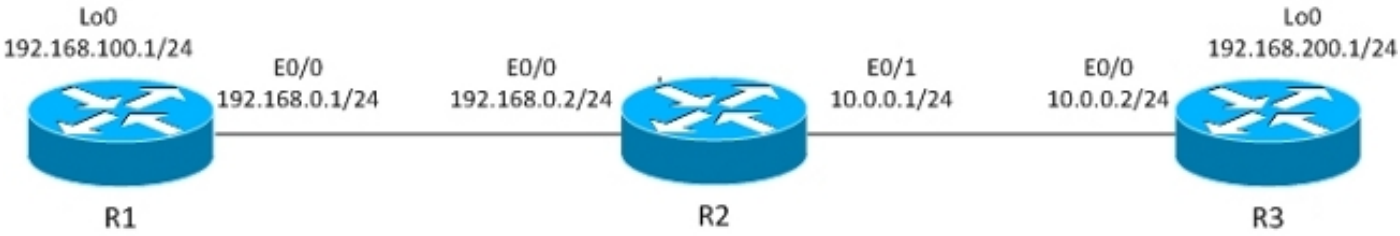
Este documento contiene un ejemplo de la configuración típica donde utilizan a un router con dos links ISP. Un link se utiliza para acceder Internet y el segundo está para el VPN. En caso de cualquier falla de link, el tráfico se rerrutea con un diverso link de Proveedor de servicios de Internet (ISP). Las trampas también se presentan.

Note por favor que el PBR está realizado en el Cisco Express Forwarding (CEF), mientras que el PBR local es process-switched.

Trafique generado localmente en el router

Esta sección describe el comportamiento del tráfico iniciado del router (R)1. Que el tráfico es ESP encapsuló por el r1.

Topología



El túnel ipsec de LAN a LAN se construye entre el r1 y el R3.

El tráfico interesante está entre el r1 Lo0 (192.168.100.1) y R3 Lo0 (192.168.200.1).

El router R3 tiene una ruta predeterminado al r2.

El r1 no tiene ninguna entrada de ruteo, solamente directamente las redes conectadas.

Configuración

El r1 tiene PBR local para todo el tráfico:

```
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
```

Depuraciones

Todo el tráfico generado a nivel local en el r1 se envía al r2 cuando está PARA ARRIBA.

Para verificar qué ocurre cuando usted trae para arriba el túnel, envíe el tráfico interesante del router sí mismo:

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

> Precaución: El comando debug ip packet pudo generar una gran cantidad de debugs y tiene impacto enorme en el USO de la CPU. Utilícelo con cautela.

Este debug también permite que la lista de acceso sea utilizada para limitar la cantidad de tráfico procesada por los debugs. El comando debug ip packet visualiza solamente el tráfico que es process-switched.

Aquí están los debugs en el r1:

```
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
```

Aquí es qué sucede:

Se determina el tráfico interesante (192.168.100.1 > 192.168.200.1) es correspondido con por el PBR local, y la interfaz de egreso (E0/0). Esta acción acciona el código crypto para iniciar el ISAKMP. Ese paquete también directiva-es ruteado por el PBR local, que determina la interfaz de egreso (E0/0). Se envía el tráfico ISAKMP, y se negocia el túnel

¿Qué sucede cuando usted hace ping otra vez?

```
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)
```

Aquí es qué sucede:

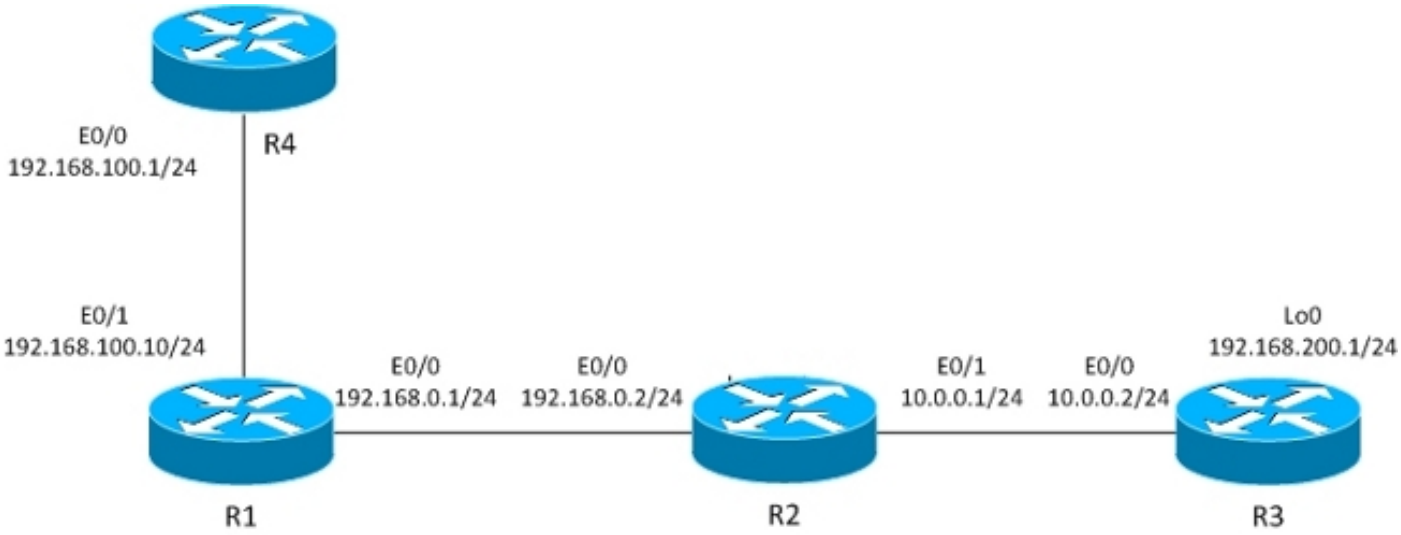
El tráfico interesante localmente generado, 192.168.100.1 > 192.168.200.1, localmente directiva-se rutea, y se determina la interfaz de egreso (E0/0). El paquete es consumido por la función de resultados del IPSec en el E0/0 y encapsulado. El paquete encapsulado (de 192.168.0.1 a 10.0.0.2) se marca para saber si hay rutear para determinar la interfaz de egreso, pero allí no es nada en las tablas de ruteo de r1, que es porqué la encapsulación falla.

En este escenario, el túnel está PARA ARRIBA, pero el tráfico no se envía porque, después de la encapsulación ESP, el Cisco IOS marca las tablas de ruteo para determinar la interfaz de egreso.

Tráfico de tránsito a través del router

Esta sección describe el comportamiento para el tráfico de tránsito que viene a través del router, que es ESP encapsulado por ese router.

Topología



El túnel L2L se construye entre el r1 y el R3.

El tráfico interesante está entre R4 (192.168.100.1) y R3 lo0 (192.168.200.1).

El router R3 tiene una ruta predeterminado al r2.

El router R4 tiene una ruta predeterminado al r1.

El r1 no tiene ninguna encaminamiento.

Configuración

La topología anterior se modifica para mostrar el flujo cuando el router recibe los paquetes para el cifrado (tráfico de tránsito en vez del tráfico generado a nivel local).

Ahora, el tráfico interesante recibido del R4 directiva-se rutea en el r1 (por el PBR en E0/1), y hay también encaminamiento de la política local para todo el tráfico:

```
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)
```

Depuraciones

Para verificar qué sucede cuando usted trae para arriba el túnel en el r1 (después de que usted recibe el tráfico interesante del R4), ingresa:

```
R1#debug ip packetR4#ping 192.168.200.1
```

Aquí están los debugs en el r1:

```
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
```

Aquí es qué sucede:

El tráfico interesante golpea el PBR en el E0/0 y el código crypto de los activadores para enviar el paquete ISAKMP. Que el paquete ISAKMP localmente directiva-está ruteado, y la interfaz de egreso es determinado por el PBR local. Se construye un túnel.

Aquí está un más ping a 192.168.200.1 del R4:

```
R4#ping 192.168.200.1
```

Aquí están los debugs en el r1:

```
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
```

Aquí es qué sucede:

El tráfico interesante golpea el PBR en el E0/0, y ese PBR determina la interfaz de egreso (E0/0). En el E0/0, el paquete es consumido por el IPSec y encapsulado. Después de que el paquete encapsulado se marque contra la misma regla PBR y se determina la interfaz de egreso, el paquete se envía y se recibe correctamente.

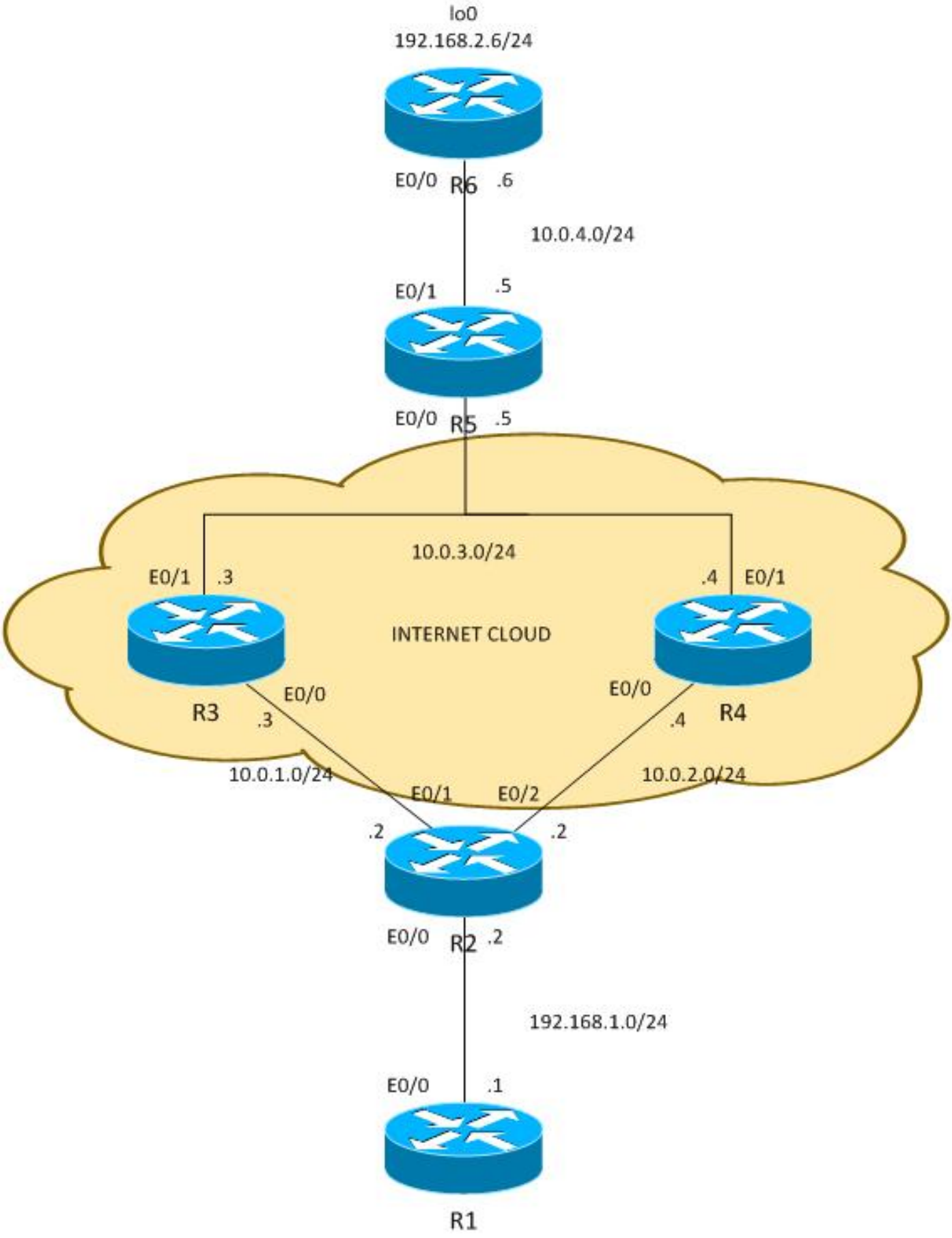
Resumen para las diferencias del comportamiento

Para el tráfico generado a nivel local, la interfaz de egreso para el tráfico NON-encapsulado (ISAKMP) es determinada por el PBR local. Para el tráfico generado a nivel local, la interfaz de egreso para el tráfico poste-encapsulado (ESP) es determinada por las tablas de ruteo (el PBR local no se marca). Para el tráfico de tránsito, la interfaz de egreso para el tráfico poste-encapsulado (ESP) es determinada por la interfaz PBR (dos veces, antes y después de la encapsulación).

Ejemplo de configuración

Éste es un ejemplo de configuración práctico que presenta los problemas que usted puede ser que haga frente con el PBR y el PBR local con el VPN. El r2 (CE) tiene dos links ISP. El router R6 también tiene el CE y un link ISP. El primer link del r2 al R3 se utiliza como ruta predeterminado para el r2. El segundo link al R4 se utiliza solamente para el tráfico VPN al R6. En caso de cualquier falla de link ISP, el tráfico se rerrutea al otro link.

Topología



Configuración

El tráfico entre 192.168.1.0/24 y 192.168.2.0/24 se protege. El Open Shortest Path First (OSPF) se utiliza en la nube de Internet para hacer publicidad de los 10.0.0.0/8 direccionamientos, que se

tratan como direcciones públicas asignadas por el ISP al cliente. En el mundo real, el BGP se utiliza en vez del OSPF.

La configuración en el r2 y el R6 se basa en el crypto-mapa. En el r2, el PBR se utiliza en el E0/0 para dirigir el tráfico VPN al R4 si está PARA ARRIBA:

```
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
```

Aquí usted ve que el PBR local no es necesario. La interfaz PBR rutea el tráfico interesante a 10.0.2.4. Eso acciona el código crypto para iniciar el ISAKMP de la interfaz correcta (link al R4), incluso cuando la encaminamiento está a las puntas del peer remoto con el R3.

En el R6, utilizan a dos pares para el VPN:

```
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
```

El r2 utiliza un acuerdo del nivel del servicio del IP (SLA) para hacer ping el R3 y el R4. La ruta predeterminado es R3. En caso del error R3, elige el R4:

```
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
```

También el r2 no prohibe a acceso a internet para todos los usuarios del interior. Para alcanzar la Redundancia en el caso donde está el ISP al R3 abajo, un route-map es necesario. Él tráfico del interior de las traducciones de dirección de puerto (palmaditas) a una diversa interfaz de egreso (la PALMADITA a la interfaz E0/1 cuando el R3 es ASCENDENTE y la ruta predeterminado señala al R3, y a la PALMADITA para interconectar el E0/2 cuando el R3 está abajo y el R4 se utiliza como ruta predeterminado).

```
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
```

El tráfico VPN necesita ser excluido de la traducción al igual que ISAKMP. Si el tráfico ISAKMP no se excluye de la traducción, es PATed a la interfaz exterior que va hacia el R3:

```
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
```

Prueba

Con esta configuración, hay una redundancia completa. El VPN utiliza el link R4, y el resto del tráfico se rutea con el R3. En caso del error R4, el tráfico VPN se establece con el link R3 (el route-map para el PBR no hace juego y se utiliza el ruteo predeterminado).

Antes de que el ISP al R4 esté abajo, el R6 ve el tráfico del par 10.0.2.2:

```
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
```

Después de que el r2 utilice el ISP al R3 para el tráfico VPN, el R6 ve el tráfico del par 10.0.1.2:

```
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
```

Para el escenario opuesto, cuando va el link al R3 abajo, todo todavía trabaja muy bien. El tráfico VPN todavía utiliza el link al R4. El Network Address Translation (NAT) se realiza para que 192.168.1.0/24 ACARICIEN para apropiarse de la dirección externa. Antes de que vaya el R3 abajo, hay una traducción a 10.0.1.2:

```
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
```

Después de que vaya el R3 abajo, todavía hay la vieja traducción junto con la nueva traducción (a 10.0.2.2) esa las aplicaciones el link hacia el R4:

```
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
```

Peligros

¿Si todo trabaja muy bien, dónde están las trampas? Están en los detalles.

Tráfico generado localmente

Aquí está un escenario que necesita iniciar el tráfico del r2 sí mismo VPN. Este escenario le requiere configurar el PBR local en el r2 para forzar el r2 para enviar el tráfico ISAKMP vía el R4 y para hacer el túnel subir. Pero la interfaz de egreso se determina con el uso de las tablas de

ruteo, con el valor por defecto señalando al R3, y ese paquete se envía al R3, en vez del R4, para el cual se utiliza transitan para el VPN. Para verificar eso, ingrese:

```
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
```

En este ejemplo, el Internet Control Message Protocol (ICMP) que se genere localmente es forzado vía el R4. Sin ese, el tráfico generado localmente de 192.168.1.2 a 192.168.2.5 se procesa con el uso de las tablas de ruteo y un túnel se establece con el R3.

¿Qué sucede después de que usted aplique esta configuración? Los paquetes icmp de 192.168.1.2 a 192.168.2.5 se ponen hacia el R4, y un túnel se inicia con el link al R4. Se configura el túnel:

```
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
```

Todo parece trabajar correctamente. El tráfico se envía con el link correcto E0/2 hacia el R4. Incluso el R6 muestra que el tráfico está recibido de 10.2.2.2, que es dirección IP del link R4:

```
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
```

Pero realmente, hay **Asymmetric Routing para los paquetes ESP** aquí. Los paquetes ESP se envían con 10.0.2.2 como fuente, pero se ponen en el link hacia el R3. Una respuesta cifrada se vuelve con el R4. Esto puede ser verificada marcando los contadores en el R3 y el R4:

Contadores R3 del E0/0 antes de enviar 100 paquetes:

```
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
```

Y los mismos contadores, después de enviar 100 paquetes:

```
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
```

El número de paquetes entrantes crecientes en 100 (en el link hacia el r2), solamente los paquetes de salida crecientes solamente en 2. El R3 ve tan solamente el eco ICMP cifrado.

La respuesta se considera en el R4, antes de enviar 100 paquetes:

```
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
```

Después de enviar 100 paquetes:

```
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
```

El número de paquetes enviados hacia el r2 creciente en 102 (respuesta de ICMP cifrada), mientras que los paquetes recibidos aumentaron en 0. El R4 ve tan solamente la respuesta de ICMP cifrada. Por supuesto, una captura de paquetes confirma esto.

¿Por qué ocurre esto? La respuesta es en la primera parte del artículo.

Aquí está el flujo de esos paquetes icmp:

1. El ICMP de 192.168.1.2 a 192.168.2.6 se pone en E0/2 (link hacia el R4) debido al PBR local.
2. Construyen a la sesión ISAKMP con 10.0.2.2 y como se esperaba puesto el link E0/2.
3. Para los paquetes icmp después de la encapsulación, el router necesita determinar la interfaz de egreso, que se hace con el uso de las tablas de ruteo que señalan al R3. Esta es la razón por la cual el paquete encriptado con la fuente 10.0.2.2 (link hacia el R4) se envía vía el R3.
4. El R6 recibe un paquete ESP de 10.0.2.2, que es constante con la sesión ISAKMP, desencripta el paquete, y envía la respuesta ESP a 10.0.2.2.
5. Debido a la encaminamiento, el R5 devuelve una respuesta a 10.0.2.2 con el R4.
6. El r2 recibe lo y los decrypts, y se valida el paquete.

Esta es la razón por la cual es importante ser extraordinariamente prudente con el tráfico generado a nivel local.

En muchas redes, se utiliza el Unicast Reverse Path Forwarding (uRPF) y el tráfico originado de 10.0.2.2 se podría caer en el E0/0 del R3. En ese caso, el ping no trabaja.

¿Hay solución para este problema? Es posible forzar al router a tratar el tráfico generado a nivel local como tráfico de tránsito. Para ese, el PBR local necesita el tráfico directo a un Loopback Interface falso de las cuales se rutee como el tráfico de tránsito.

Esto no se aconseja.

> Nota: Es importante tener extraordinariamente cuidado cuando usted utiliza el NAT junto con el PBR (refiera a la sección anterior sobre el tráfico ISKMP en la lista de acceso de la PALMADITA).

Ejemplo de configuración sin el PBR

Hay también otra solución que es un compromiso. Con la misma topología que el ejemplo anterior, es posible satisfacer todos los requisitos sin el uso del PBR o del PBR local. Para este scenarrio, solamente se utiliza el rutear. Solamente una más entrada de ruteo se agrega en el r2, y se quitan todas las configuraciones PBR/local PBR:

```
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
```

En el total, el r2 tiene esta configuración de ruteo:

```
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
```

La primera entrada de ruteo es un ruteo predeterminado hacia el R3, cuando el link al R3 está PARA ARRIBA. La segunda entrada de ruteo es una ruta predeterminado de reserva hacia el R4, cuando el link al R3 está abajo. La tercera entrada decide a qué tráfico de la manera a la red VPN remota se envía, dependiendo del estado del link R4 (si el link R4 está PARA ARRIBA, el tráfico a la red VPN remota se envía vía el R4). Con esta configuración, no hay necesidad del Policy Routing.

¿Cuál es la desventaja? No hay control granular usando el PBR más. No es posible determinar a la dirección de origen. En este caso, todo el tráfico a 192.168.2.0/24 se envía hacia el R4 cuando está PARA ARRIBA, sin importar la fuente. En el ejemplo anterior, eso fue controlada por el PBR y la fuente específica: se selecciona 192.168.1.0/24.

¿Para qué escenario es esta solución demasiado simple? Para las redes del LAN múltiple (detrás del r2). Cuando algunas de esas redes necesitan alcanzar 192.168.2.0/24 de una manera segura (cifrada) y de otras maneras inseguras (unencrypted), el tráfico de las redes inseguras todavía se pone en la interfaz E0/2 del r2 y no golpea el crypto-mapa. Tan se envía unencrypted vía un link al R4 (y al requisito principal era utilizar el R4 solamente para el tráfico encriptado).

Este tipo de escenario y sus requisitos son raros, que es porqué esta solución se utiliza bastante a menudo.

Resumen

Usando el PBR y las características locales PBR junto con los VPN y el NAT pudo ser complejo y requiere una comprensión en detalle del flujo de paquetes.

Para los escenarios tales como ésos presentados aquí, se aconseja para utilizar a dos routeres separados - cada router con un link ISP. En caso de un error ISP, el tráfico se puede rerrutear fácilmente. No hay necesidad del PBR, y el diseño total es mucho más simple.

Hay también una solución de compromiso que no requiere el uso del PBR, solamente el rutear flotante estático de las aplicaciones en lugar de otro.

Verificación

Actualmente, no hay un procedimiento de verificación disponible para esta configuración.

Troubleshooting

Actualmente, no hay información específica de troubleshooting disponible para esta configuración.

Información Relacionada

- Soporte Técnico y Documentación - Cisco Systems
- Cisco IOS 15.3 M&T- Cisco Systems