Beleidsrouting en de invloed daarvan op ESP- en ISAKMP-pakketten met Cisco IOS

Inhoud

Inleiding

Dit document beschrijft het effect van Op beleid gebaseerde routing (PBR) en lokale PBR bij toepassing op ingesloten security payload-pakketten (ESP) en Internet Security Association en Key Management Protocol (ISAKMP) wanneer u Cisco IOS® gebruikt.

Bijgedragen door Michal Garcarz, Cisco TAC Engineer.

Voorwaarden

Vereisten

Cisco raadt u aan basiskennis van deze onderwerpen te hebben:

- Cisco IOS-Cisco
- VPN-configuratie op Cisco IOS

Gebruikte componenten

De informatie in dit document is gebaseerd op Cisco IOS versie 15.x.

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u de potentiële impact van elke opdracht begrijpen.

Achtergrondinformatie

Vóór de tunnelvestiging van IPsec start de router een ISAKMP-uitwisseling. Aangezien die pakketten door de router worden gegenereerd, worden de pakketten behandeld zoals lokaal gegenereerd verkeer en om het even welke lokale besluiten van PBR worden toegepast. Daarnaast worden alle pakketten die door de router (Enhanced Interior Gateway Routing Protocol (DHCP), Next Hopprotocol (NHRP), Border Gateway Protocol (BGP) of Internet Control Message Protocol (ICMP) worden gegenereerd door indelingen), ook beschouwd als lokaal gegenereerd verkeer en hebben de lokale PBR-beslissing van toepassing.

Het verkeer dat door de router wordt doorgestuurd en door de tunnel wordt verstuurd, dat transitoverkeer wordt genoemd, wordt niet beschouwd als lokaal gegenereerd verkeer, en elk gewenst routingbeleid moet worden toegepast op de ingangsinterface van de router.

Dit heeft gevolgen voor het verkeer dat door de tunnel rijdt, omdat het lokaal gegenereerde verkeer PBR volgt, maar het transitoverkeer niet. Dit artikel legt de consequenties van dit verschil in gedrag uit.

Voor transitverkeer dat ESP moet worden ingekapseld, hoeft er geen routing items te zijn, omdat PBR de spanning-interface voor het pakket voor en na ESP-insluiting bepaalt. Voor lokaal gegenereerd verkeer dat ESP ingekapseld moet zijn, is het noodzakelijk om het verzenden van ingangen te hebben, omdat lokale PBR de spanning slechts voor het pakket vóór insluiting bepaalt en het routing de spanning interface voor het post-ingekapselde pakket bepaalt.

Dit document bevat een typisch configuratievoorbeeld waar een router met twee ISP-koppelingen wordt gebruikt. Eén link wordt gebruikt om toegang tot het internet te krijgen en de tweede is voor VPN. In het geval van een storing van de link wordt het verkeer via een andere link tussen Internet Service Provider (ISP) uitgevoerd. Ook valkuilen worden gepresenteerd.

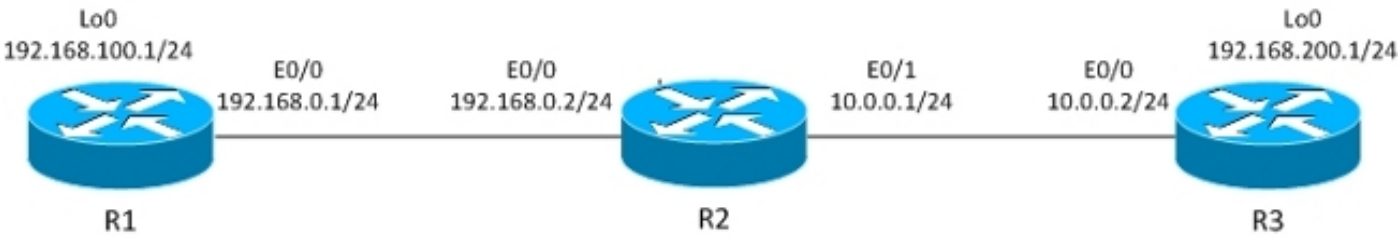
Merk op dat PBR wordt uitgevoerd in Cisco Express Forwarding (CEF), terwijl lokale PBR proces-

switched is.

Lokaal gegenereerd verkeer op de router

In dit deel wordt het gedrag beschreven van verkeer dat van router (R)1 is geïnitieerd. Dat verkeer is ESP ingekapseld door R1.

Topologie



De IPsec LAN-to-LAN tunnel is gemaakt tussen R1 en R3.

Het interessante verkeer ligt tussen R1 Lo0 (192.168.100.1) en R3 Lo0 (192.168.200.1).

De R3 router heeft een standaardroute naar R2.

R1 heeft geen routingangen, alleen rechtstreeks verbonden netwerken.

Configuratie

R1 heeft lokale PBR voor al het verkeer:

```
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
```

Debugs

Alle lokaal gegenereerd verkeer op R1 wordt naar R2 verzonden wanneer het UP is.

Om te verifiëren wat voorkomt wanneer u de tunnel omhoog brengt, stuur het interessante verkeer van de router zelf:

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

> Voorzichtig: De opdracht ip-pakketten debug kan een grote hoeveelheid defecten opleveren en heeft grote impact op het CPU-gebruik. Gebruik het voorzichtig.

Dit debug laat ook de toegang-lijst toe om de hoeveelheid verkeer te beperken die door insecten wordt verwerkt. Het debug IP-pakketopdracht geeft alleen verkeer weer dat naar een ander proces is overgeschakeld.

Hier zijn de uiteinden op 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
```

Dit is wat er gebeurt:

Het interessante verkeer (192.168.100.1 > 192.168.2001) wordt afgesloten met een lokale PBR en de spanning-interface wordt bepaald (E0/0). Deze actie zet de cryptocode in werking om ISAKMP te initiëren. Dat pakket is ook beleid-routed door lokale PBR, dat de spanning interface (E0/0) bepaalt. Het ISAKMP-verkeer wordt verstuurd en de tunnel wordt onderhandeld

Wat gebeurt er als je weer pingelt?

```
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)
```

Dit is wat er gebeurt:

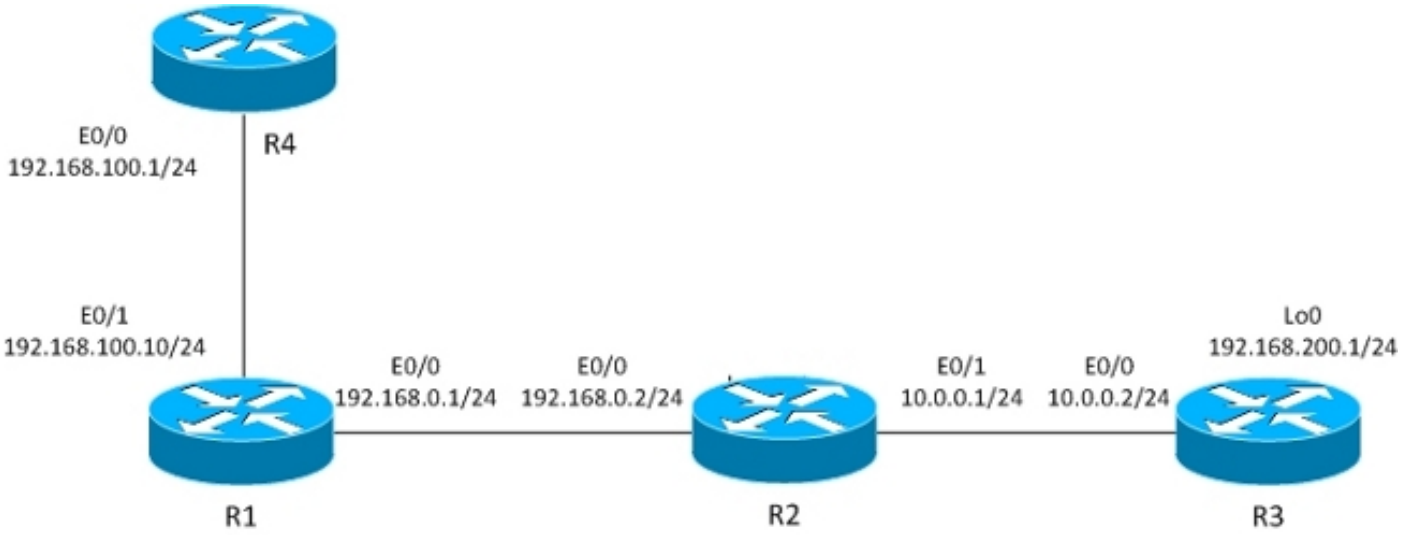
Het lokaal gegenereerde interessante verkeer, 192.168.100.1 > 192.168.2001, is lokaal beleidsgestuurd en de spanning-interface wordt bepaald (E0/0). Het pakket wordt geconsumeerd door de uitvoeroptie IPsec op E0/0 en ingekapseld. Het ingekapselde pakket (van 192.168.0.1 tot 10.0.0.2) wordt gecontroleerd voor het routing om de spanning interface te bepalen, maar er is niets in de routingtabellen van R1, wat de reden is dat de insluiting mislukt.

In dit scenario, is de tunnel UP, maar het verkeer wordt niet verzonden omdat, na ESP insluiting, Cisco IOS de routingtabellen controleert om de spanning interface te bepalen.

Doorsturen van verkeer door de router

In dit gedeelte wordt het gedrag beschreven voor transitovervoer dat door de router komt, wat ESP is ingekapseld door die router.

Topologie



De L2L-tunnel is gebouwd tussen R1 en R3.

Het interessante verkeer ligt tussen R4 (192.168.100.1) en R3 lo0 (192.168.2001).

De R3 router heeft een standaardroute naar R2.

De R4 router heeft een standaardroute naar R1.

R1 heeft geen routing.

Configuratie

De vorige topologie wordt gewijzigd om de stroom te tonen wanneer de router pakketten voor encryptie (transitverkeer in plaats van lokaal gegenereerd verkeer) ontvangt.

Op dit moment is het interessante verkeer dat van R4 wordt ontvangen beleid-routed op R1 (door PBR op E0/1), en er is ook lokale beleidsrouting voor al verkeer:

```
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
```

Debugs

Om te verifiëren wat er gebeurt wanneer u de tunnel op R1 brengt (nadat u het interessante verkeer van R4 ontvangt), ga in:

```
R1#debug ip packet
```

```
R4#ping 192.168.200.1
```

Hier zijn de uiteinden op 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
```

Dit is wat er gebeurt:

Het interessante verkeer bereikt PBR op E0/0 en zet crypto code op om het ISAKMP-pakket te verzenden. Dat ISAKMP-pakket lokaal is routeerd, en de accu-interface wordt bepaald door de lokale PBR. Er wordt een tunnel gebouwd.

Hier is nog een ping van 192.168.2001 van R4:

```
R4#ping 192.168.200.1
```

Hier zijn de uiteinden op 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
```

Dit is wat er gebeurt:

Het interessante verkeer bereikt PBR op E0/0 en dat PBR de spanning-interface (E0/0) bepaalt. Op E0/0 wordt het pakket door IPSec geconsumeerd en ingekapseld. Nadat het ingekapselde pakket tegen de zelfde PBR regel wordt gecontroleerd en de spanning interface wordt bepaald, wordt het pakket verzonden en ontvangen correct.

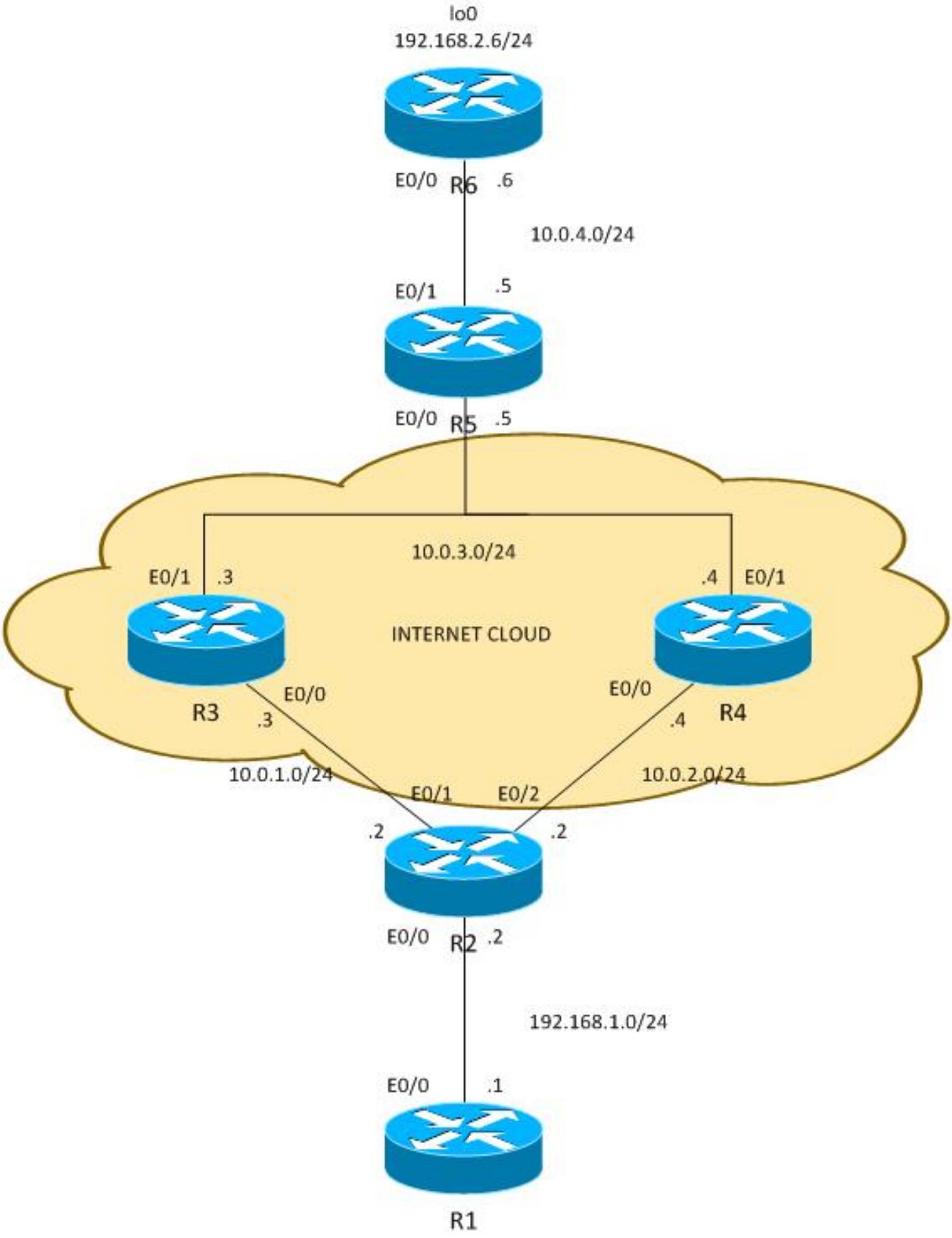
Samenvatting voor gedragsverschillen

Voor lokaal gegenereerd verkeer, wordt de spanning interface voor niet-ingekapseld verkeer (ISAKMP) bepaald door lokale PBR. Voor lokaal gegenereerd verkeer, wordt de spanning interface voor post-ingekapseld verkeer (ESP) bepaald door de routingtabellen (de lokale PBR wordt niet gecontroleerd). Voor transitverkeer wordt de spanning-interface voor post-ingekapseld verkeer (ESP) bepaald door de interface PBR (tweemaal, voor en na insluiting).

Configuratievoorbeeld

Dit is een praktisch configuratievoorbeeld dat de kwesties presenteert die u met PBR en lokale PBR met VPN kunt tegenkomen. R2 (CE) heeft twee ISP-koppelingen. De R6 router heeft ook CE en één ISP verbinding. De eerste verbinding van R2 naar R3 wordt gebruikt als standaardroute voor R2. De tweede link naar R4 wordt alleen gebruikt voor VPN-verkeer naar R6. In het geval van een storing van de ISP wordt het verkeer terugverwezen naar de andere link.

Topologie



Configuratie

Het verkeer tussen 192.168.1.0/24 en 192.168.2.0/24 wordt beschermd. Open Shortest Path First (OSPF) wordt in de Internet-cloud gebruikt om de 10.0.0.0/8 adressen te adverteren, die worden

behandeld als openbare adressen die door ISP aan de klant zijn toegewezen. In de echte wereld wordt BGP gebruikt in plaats van OSPF.

De configuratie op R2 en R6 is gebaseerd op de crypto-kaart. Op R2 wordt PBR gebruikt op E0/0 om VPN-verkeer naar R4 te sturen als het UP is:

```
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
```

Hier zie je dat lokale PBR niet nodig is. De interface PBR routeert interessant verkeer naar 10.0.2.4. Dat veroorzaakt de crypto code om ISAKMP van de juiste interface (verbinding met R4) te initiëren, zelfs wanneer de routing naar verre peer points door R3 is.

Op R6 worden twee peers voor VPN gebruikt:

```
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 gebruikt een IP Service Level Agreement (SLA) om R3 en R4 te pingelen. De standaardroute is R3. In het geval van een R3-storing kiest het R4:

```
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
```

Ook R2 maakt internettoegang mogelijk voor alle binnengebruikers. Om overtolligheid te bereiken in het geval waar ISP aan R3 is, is een route-kaart nodig. Het is Port Address Translations (PATs) binnenin traffic to a other egress interface (PAT tot E0/1 interface wanneer R3 omhoog is en de standaardroute wijst naar R3, en PAT naar interface E0/2 wanneer R3 omlaag is en R4 als standaardroute wordt gebruikt).

```
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
```

VPN-verkeer moet, net als ISAKMP, van vertaling worden uitgesloten. Als ISAKMP-verkeer niet van vertaling is uitgesloten, wordt PATed toegevoegd aan de externe interface die naar R3 gaat:

```
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
```

Testen

Met deze configuratie is er een volledige redundantie. VPN gebruikt de R4 link, en de rest van het verkeer wordt met R3 routeerd. In het geval van een R4-fout, wordt het VPN-verkeer met de R3-verbinding tot stand gebracht (de route-map voor PBR komt niet overeen en de standaardrouting wordt gebruikt).

Voordat de ISP naar R4 is uitgeschakeld, ziet R6 het verkeer vanaf peer 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
```

Nadat R2 ISP aan R3 voor VPN-verkeer gebruikt, ziet R6 verkeer vanaf peer 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
```

Voor het tegenovergestelde scenario, wanneer de link naar R3 omlaag gaat, werkt alles nog steeds prima. VPN-verkeer gebruikt nog steeds de link naar R4. Network Address Translation (NAT) wordt uitgevoerd voor 192.168.1.0/24 naar PAT om het externe adres aan te passen. Voordat de R3 naar beneden gaat, is er een vertaling naar 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
```

Nadat het R3-nummer is gedaald, zijn er nog steeds de oude vertaling samen met de nieuwe vertaling (tot 10.0.2.2) die de link naar R4 gebruikt:

```
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
```

valkuilen

Als alles goed werkt, waar zijn dan de valkuilen? Ze staan in de details.

Lokaal gegenereerd verkeer

Hier is een scenario dat VPN verkeer van de R2 zelf moet initiëren. Dit scenario vereist dat u lokale PBR op R2 configureren om R2 te dwingen om ISAKMP-verkeer via R4 te verzenden en de tunnel naar boven te laten gaan. Maar de graafinterface wordt bepaald met het gebruik van routingtabellen, met de standaardinstelling op R3, en dat pakket wordt naar R3 verzonden in plaats van R4, dat wordt gebruikt voor doorgeleiding voor VPN. Voer om dit te verifiëren de volgende gegevens in:

```
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
```

In dit voorbeeld wordt het Internet Control Message Protocol (ICMP), dat lokaal wordt gegenereerd, via R4 afgedwongen. Zonder dat protocol wordt het verkeer dat lokaal wordt gegenereerd van 192.168.1.2 tot 192.168.2.5, verwerkt met het gebruik van routingtabellen en wordt een tunnel met R3 tot stand gebracht.

Wat gebeurt er nadat je deze configuratie toepast? Het ICMP-pakket van 192.168.1.2 tot 192.168.2.5 wordt naar R4 gezet en een tunnel wordt gestart met de link naar R4. De tunnel is opgezet:

```
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
```

Alles lijkt correct te werken. Het verkeer wordt verstuurd met de juiste link E0/2 naar R4. Zelfs R6 laat zien dat het verkeer wordt ontvangen vanaf 10.2.2.2, dat is R4's link IP-adres:

```
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
```

Maar eigenlijk is er **asymmetrische routing voor ESP-pakketten** hier. ESP-pakketten worden verstuurd met 10.0.2.2 als bron, maar worden op de link naar R3 gezet. Een versleuteld antwoord wordt teruggestuurd door R4. Dit kan worden geverifieerd door tellers op R3 en R4 te controleren:

R3 tellers van E0/0 alvorens 100 pakketten te verzenden:

```
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
```

En dezelfde tellers, na het verzenden van 100 pakketten:

```
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
```

Het aantal binnenkomende pakketten werd met 100 verhoogd (op de link naar R2), maar de uitgaande pakketten werden slechts met 2 verhoogd. Dus R3 ziet alleen de gecodeerde ICMP echo.

De reactie wordt op R4 gezien voordat er 100 pakketten worden verzonden:

```
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
```

Nadat u 100 pakketten hebt verzonden:

```
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
```

Het aantal pakketten dat naar R2 wordt verzonden steeg met 102 (gecodeerd ICMP antwoord), terwijl de ontvangen pakketten met 0 werden verhoogd. Dus R4 ziet enkel het gecodeerde ICMP antwoord. Een pakketvastlegging bevestigt dit uiteraard.

Waarom gebeurt dit? Het antwoord luidt in het eerste deel van het artikel.

Hier is de stroom van die ICMP-pakketten:

1. Het ICMP wordt van 192.168.1.2 tot 192.168.2.6 op E0/2 (link naar R4) gezet vanwege lokale PBR.
2. De ISAKMP-sessie is gebouwd met 10.0.2.2 en maakt de E0/2 link aan zoals verwacht.
3. Voor ICMP pakketten na insluiting, moet de router de spanning interface bepalen, die wordt uitgevoerd met het gebruik van routeringstabellen die wijzen naar R3. Dit is waarom het gecodeerde pakket met bron 10.0.2.2 (link naar R4) via R3 wordt verzonden.
4. R6 ontvangt een ESP-pakket van 10.0.2.2, dat in overeenstemming is met de ISAKMP-sessie, decrypteert het pakket en stuurt de ESP-respons naar 10.0.2.2.
5. Vanwege de routing stuurt R5 een reactie naar 10.0.2.2 via R4.
6. R2 ontvangt het en decrypteert, en het pakje wordt geaccepteerd.

Daarom is het belangrijk om extra voorzichtig te zijn met het lokaal gegenereerde verkeer.

In veel netwerken wordt Unicast Reverse Path Forwarding (uRPF) gebruikt en kan het verkeer dat uit 10.0.2.2 komt worden gedropt op E0/0 van R3. In dat geval werkt ping niet.

Is er een oplossing voor dit probleem? Het is mogelijk om de router te dwingen om lokaal gegenereerd verkeer als transitoverkeer te behandelen. Daarvoor is het nodig dat lokale PBR het verkeer richt op een enorme loopback-interface waarvan het als transitoverkeer wordt routeerd.

Dit wordt niet geadviseerd.

> Opmerking: Het is belangrijk om extra voorzichtig te zijn wanneer u NAT samen met PBR gebruikt (raadpleeg de vorige sectie over ISKMP-verkeer in PAT-toegangslijst).

Voorbeelden zonder PBR

Er is ook nog een andere oplossing die een compromis is. Met dezelfde topologie als het vorige voorbeeld, is het mogelijk om aan alle vereisten te voldoen zonder het gebruik van PBR of lokale PBR. Voor dit scenario, wordt slechts de routing gebruikt. Er wordt slechts één routingstoegang toegevoegd op R2 en alle PBR-/plaatselijke PBR-configuraties worden verwijderd:

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

In totaal heeft R2 deze routingconfiguratie:

```
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
```

De eerste routingingang is een standaard routing naar R3, wanneer de link naar R3 omhoog is. De tweede routingingang is een back-upstandaardroute naar R4, wanneer de link naar R3 is weggevallen. De derde ingang beslist welke manier het verkeer naar het verre VPN netwerk wordt verzonden, afhankelijk van de staat van de R4 verbinding (als R4 verbinding UP is, wordt het verkeer naar het verre VPN netwerk verzonden via R4). Met deze configuratie is er geen behoefte aan beleidsrouting.

Wat is het nadeel? Er is geen korrelcontrole meer waarbij PBR wordt gebruikt. Het bronadres kan niet worden bepaald. In dit geval wordt al het verkeer naar 192.168.2.0/24 naar R4 verstuurd wanneer het omhoog is, ongeacht de bron. In het vorige voorbeeld werd dit door PBR en de specifieke bron gecontroleerd: 192.168.1.0/24 is geselecteerd.

Voor welk scenario is deze oplossing te eenvoudig? Voor meerdere LAN-netwerken (achter R2). Wanneer sommige van die netwerken 192.168.2.0/24 op een veilige manier (versleuteld) en op andere onveilige manieren (niet versleuteld) moeten bereiken, wordt het verkeer vanaf onveilige netwerken nog steeds op de E0/2-interface van R2 gezet en raakt het niet op de crypto-kaart. Het wordt dus niet versleuteld via een link naar R4 (en de primaire eis was om R4 alleen te gebruiken voor versleuteld verkeer).

Dit soort scenario's en de eisen ervan zijn zeldzaam, en daarom wordt deze oplossing vrij vaak gebruikt.

Samenvatting

Het gebruik van PBR- en lokale PBR-functies samen met VPN’s en NAT kan complex zijn en

vereist een diepgaand begrip van de pakketstroom.

Voor scenario's zoals die hier worden gepresenteerd, wordt het geadviseerd om twee afzonderlijke routers te gebruiken - elke router met één ISP-link. Wanneer een ISP faalt, kan het verkeer eenvoudig worden teruggedraaid. Er is geen behoefte aan PBR en het algemene ontwerp is veel eenvoudiger.

Er is ook een compromisoplossing die niet het gebruik van PBR vereist, maar de statische drijvende routing in plaats daarvan gebruikt.

Verifiëren

Er is momenteel geen verificatieprocedure beschikbaar voor deze configuratie.

Problemen oplossen

Er is momenteel geen specifieke troubleshooting-informatie beschikbaar voor deze configuratie.

Gerelateerde informatie

- Technische ondersteuning en documentatie – Cisco Systems
- Cisco IOS 15.3 M&T-E Cisco-systemen