Cisco IOS NAT - Intégration avec MPLS VPN

Contenu

[Introduction](#)
[Avantages de NAT – Intégration MPLS](#)
[Considérations de conception](#)
[Scénarios de déploiement](#)
[Options de déploiement et détails de configuration](#)
[PE de sortie NAT](#)
[PE d'entrée NAT](#)
[Paquets arrivant au PE central après le PE d'entrée NAT](#)
[Entretenez l'exemple](#)
[Disponibilité](#)
[Conclusion](#)
[Informations connexes](#)

Introduction

Le logiciel de Traduction d'adresses de réseau (NAT) de Cisco IOS® permet l'accès aux services partagés du multiple MPLS VPN, même lorsque les périphériques dans les VPN utilisent les adresses IP qui superposent. Le Cisco IOS NAT est Vrf-averti et peut être configuré sur des Routeurs de Provider Edge dans le réseau MPLS.

Remarque: Le MPLS dans l'IOS est pris en charge seulement avec NAT existant. À ce moment, il n'y a aucun support dans le Cisco IOS pour NVI NAT avec le MPLS.

On projette que le déploiement de MPLS VPN augmente rapidement au cours des plusieurs années à venir. Les avantages d'une infrastructure réseau commune que l'extension rapide d'autorisations et les options flexibles de Connectivité piloteront assurément davantage de croissance des services qui peuvent être offerts à la communauté d'interréseau.

Cependant, les barrages au développement demeurent toujours. L'IPv6 et sa promesse d'un espace d'adresse IP qui dépasse les besoins de Connectivité d'avenir a lieu toujours pendant les phases tôt du déploiement. Les réseaux existants utilisent généralement des schémas privés d'adressage IP comme définis dans le RFC 1918 . La traduction d'adresses réseau est employée souvent pour interconnecter des réseaux quand les espaces d'adressage superposent ou la duplication existe.

Les fournisseurs de services et les entreprises qui ont des services d'application réseau qu'ils veulent offrir ou le partage avec des clients et partenaires voudront réduire n'importe quelle charge de Connectivité placée sur l'utilisateur du service. Il est desirable, même obligatoire, pour étendre l'offre à autant d'utilisateurs possibles car nécessaire pour atteindre les buts désirés ou pour retourner. Le schéma d'adressage IP en service ne doit pas être une barrière qui exclut des utilisateurs possibles.

En déployant le Cisco IOS NAT dans l'infrastructure commune MPLS VPN, les fournisseurs de services de transmissions peuvent en libérer de l'obligation de Connectivité sur des clients et accélérer leur capacité de lier des services plus partagés d'application à plus de consommateurs de ces services.

Avantages de NAT – Intégration MPLS

L'intégration NAT avec le MPLS a des avantages pour les deux fournisseurs de services et leurs clients de l'entreprise. Il offre à des fournisseurs de services plus d'options de déployer des services partagés et de permettre d'accéder à ces services. Les offres de service supplémentaire peuvent être un différentiateur au-dessus des concurrents.

Pour le fournisseur de services	Pour le VPN
Plus d'offres de services	Coûts réduits
Options accrues d'accès	Un accès plus simple
Augmentation des recettes	Adressage de la flexibilité

Les clients de l'entreprise recherchant à externaliser une partie de leur charge de travail en cours peuvent également tirer bénéfice des offres plus larges des fournisseurs de services. Le décalage de la charge d'exécuter n'importe quelle traduction d'adresses nécessaire au réseau du fournisseur de service les soulage d'une tâche administrative compliquée. Les clients peuvent continuer à utiliser l'adressage privé, pourtant mettent à jour l'accès aux services partagés et à l'Internet. La consolidation de la fonction NAT dans le réseau du fournisseur de service peut également diminuer le coût total aux clients de l'entreprise puisque les Routeurs de Customer Edge ne doivent pas remplir la fonction NAT.

Considérations de conception

En considérant les conceptions qui appelleront NAT dans le réseau MPLS, la première étape est de déterminer les besoins de service d'un point de vue d'application. Vous devrez considérer les protocoles transmission utilisée et n'importe quelle spéciale de client/serveur imposée par l'application. Assurez-vous que le soutien nécessaire des protocoles utilisés sont pris en charge et manipulés par le Cisco IOS NAT. Une liste de protocoles pris en charge est fournie dans les passerelles NAT de couche application de Cisco IOS de document.

Ensuite, il sera nécessaire de déterminer l'utilisation prévue du service partagé et le débit de trafic anticipé dans le paquet-par-deuxième. NAT est une fonction CPU-intensive de routeur. Par conséquent, les exigences de marche seront un facteur en sélectionnant une option particulière de déploiement et détermineront le nombre de périphériques NAT impliqués.

En outre, considérez tous les problèmes de sécurité et précautions qui devraient être pris. Bien que MPLS VPN, par définition, soient privé et le trafic efficacement distinct, le réseau de service partagé est généralement commun parmi beaucoup de VPN.

Scénarios de déploiement

Il y a deux options pour le déploiement NAT dans le Provider Edge MPLS :

- Centralisé avec le siège potentiel d'explosion NAT de sortie
- Distribué avec le siège potentiel d'explosion NAT d'entrée

Quelques avantages à configurer la fonction NAT au point de sortie du réseau MPLS le plus près au réseau de service partagé incluent :

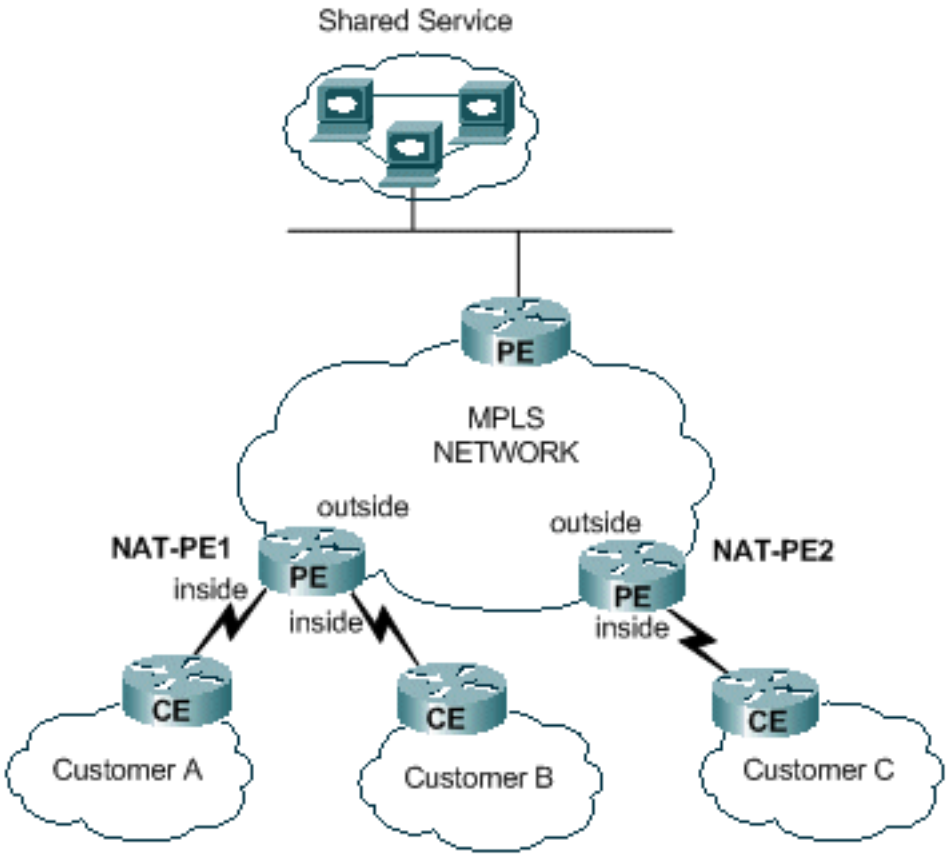
- Une configuration centralisée qui favorise une mise en service plus simple
- Dépannage simplifié
- Évolutivité opérationnelle améliorée
- Conditions requises diminuées d'allocation d'adresse IP

Cependant, les avantages sont compensés par une réduction d'évolutivité et de représentation. C'est le compromis principal qui doit être considéré. Naturellement, la fonction NAT peut également être remplie dans les réseaux client si on le détermine que l'intégration de cette caractéristique avec un réseau MPLS n'est pas desirable.

PE d'entrée NAT

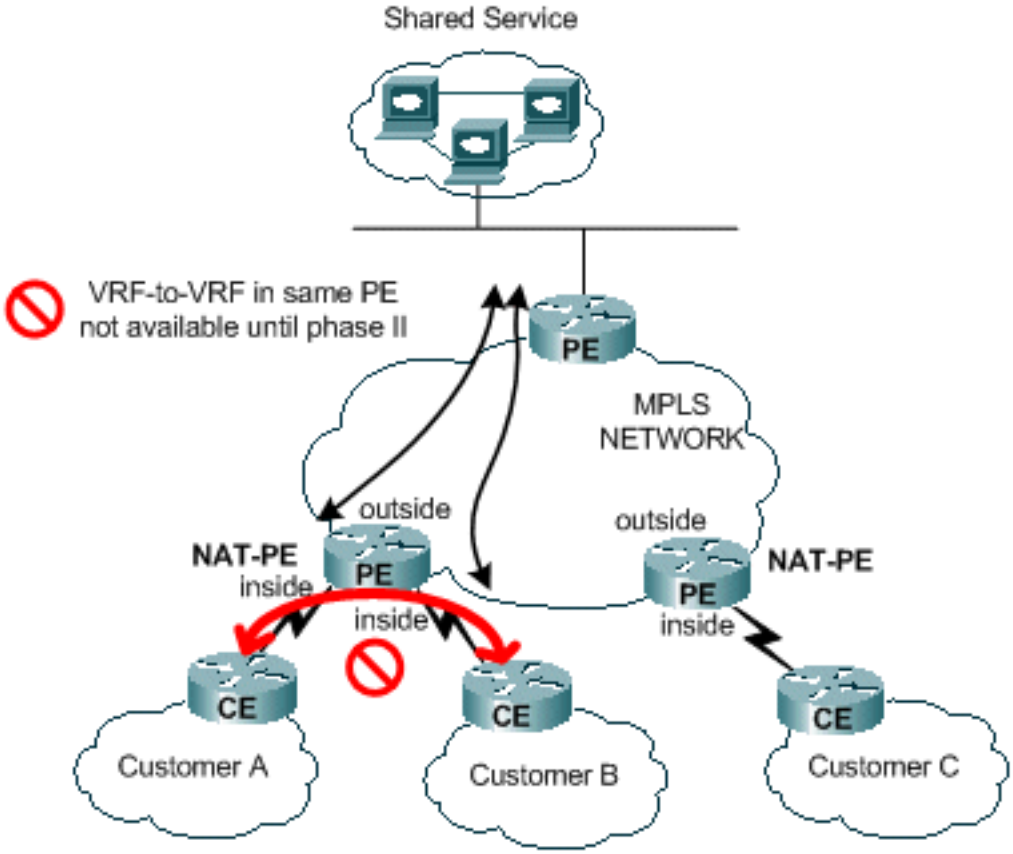
NAT peut être configuré au routeur PE d'entrée de réseau MPLS suivant les indications de la figure 1. Avec cette conception, l'évolutivité est mise à jour dans une large mesure tandis que la représentation est optimisée en distribuant la fonction NAT au-dessus de beaucoup de périphériques de périphérie. Les traitements NAT de chaque PE trafiquent pour des sites localement connectés à ce PE. Règles NAT et listes de contrôle d'accès ou contrôle de mappages de route que les paquets exigent la traduction.

Figure 1 : PE d'entrée NAT



Il y a une restriction qui empêche NAT entre deux vrf tout en également fournissante NAT à un service partagé suivant les indications de la figure 2. C'est dû à la condition requise d'indiquer des interfaces en tant qu'interfaces NAT de « intérieur » et de « extérieur ». Le soutien des connexions entre les vrf dans un PE simple est prévu pour une future release de Cisco IOS.

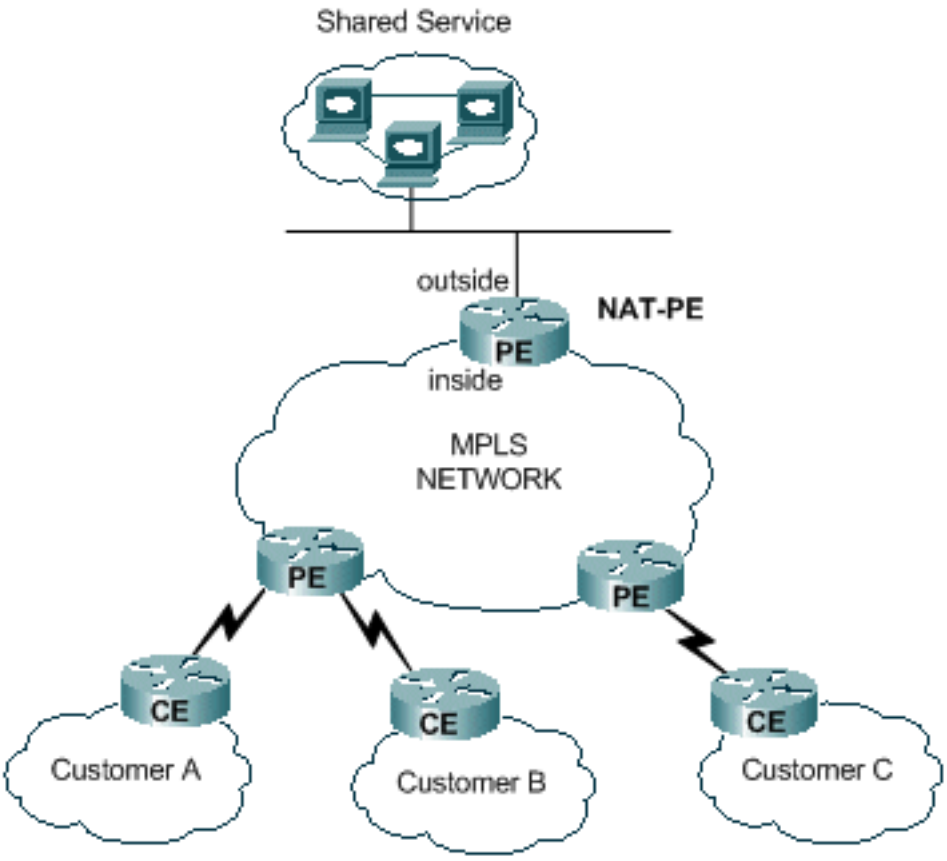
Figure 2 : D'entreprise à entreprise



PE de sortie NAT

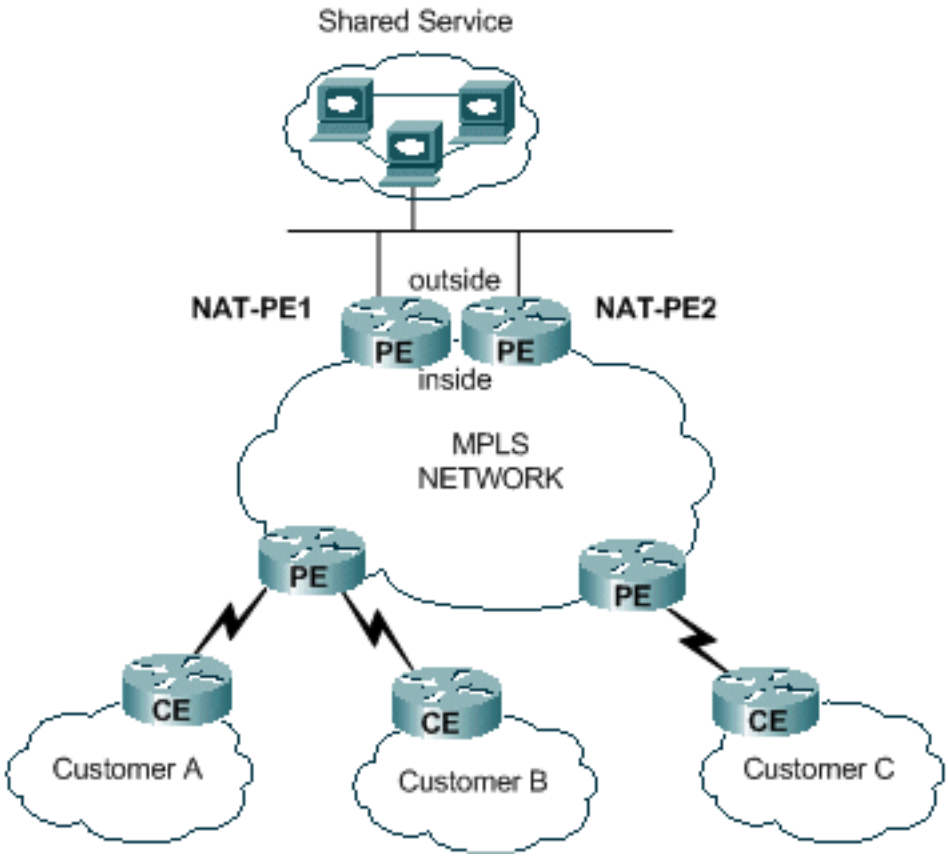
NAT peut être configuré au routeur PE de sortie de réseau MPLS suivant les indications de la figure 3. Avec cette conception, l'évolutivité est réduite à un certain degré puisque le PE central doit mettre à jour des artères pour tous les réseaux client qui accèdent au service partagé. Les exigences de performance des applications doivent également être considérées de sorte que le trafic ne surcharge pas le routeur qui doit traduire les adresses IP des paquets. Puisque NAT se produit centralement pour tous les clients à l'aide de ce chemin, des groupes d'adresse IP peuvent être partagés ; ainsi, le nombre total de sous-réseaux exigés est réduit.

Figure 3 : PE de sortie NAT



Des plusieurs routeurs pourraient être déployés pour augmenter l'évolutivité de la conception NAT de PE de sortie suivant les indications de la figure 4. Dans ce scénario, le client VPN pourrait « provisioned » sur un routeur NAT spécifique. La traduction d'adresses réseau se produirait pour l'ensemble du trafic à et du service partagé pour cela a placé des VPN. Par exemple, le trafic des VPN pour le client A et B pourrait utiliser NAT-PE1, alors que le trafic à et du VPN pour le C de client utilise NAT-PE2. Chaque PE NAT porterait le trafic seulement pour la particularité VPN définie et mettrait à jour seulement des artères de nouveau aux sites dans ces VPN. Des pools d'adresses NAT distincts pourraient être définis chez chacun des Routeurs NAT de PE de sorte que des paquets soient conduits du réseau de service partagé au PE NAT approprié pour la traduction et le routage de nouveau au client VPN.

Figure 4 : Plusieurs PE de sortie NAT



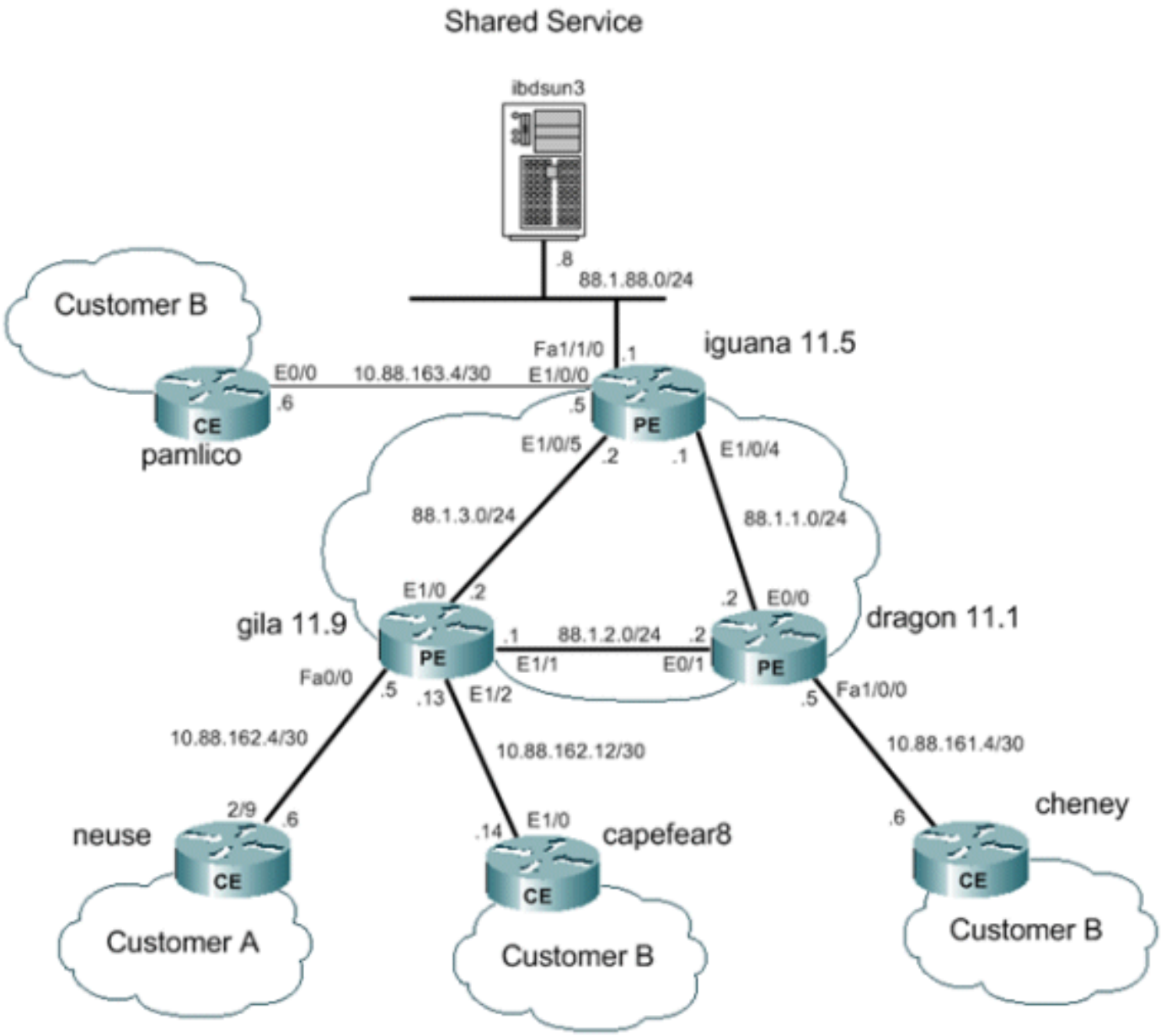
La conception centralisée impose une restriction sur la façon dont le réseau de service partagé doit être configuré. Spécifiquement, l'utilisation de l'importation/d'exportation des artères MPLS VPN entre un service partagé VPN et le client VPN n'est pas possible. C'est dû à la nature de l'exécution MPLS comme spécifié par RFC 2547 . Quand des artères sont importées et exportées utilisant les communautés étendues et les descripteurs d'artère, NAT ne peut pas déterminer la source VPN du paquet l'entrée dans du PE NAT central. Le cas habituel est de faire au réseau de service partagé une interface générique plutôt qu'une interface de VRF. Une artère au réseau de service partagé est alors ajoutée dans le PE NAT central pour chaque table de VRF associée avec un client VPN ayant besoin de l'accès au service partagé en tant qu'élément du processus d'approvisionnement. Ceci est décrit plus en détail plus tard.

Options de déploiement et détails de configuration

Cette section inclut quelques détails liés à chacune des options de déploiement. Tous les exemples sont pris du réseau représenté sur le schéma 5. se rapportent à ce diagramme pour le reste de cette section.

Remarque: Dans le réseau utilisé pour illustrer l'exécution du VRF NAT pour ce document, seulement des Routeurs de PE sont inclus. Il n'y a aucun Routeurs du noyau « P ». Cependant, les mécanismes essentiels peuvent encore être vus.

Figure 5 : Exemple NAT de configuration de VRF



PE de sortie NAT

Dans cet exemple, les Routeurs de Provider Edge le **Gila** marqué et le **dragon** sont configurés en tant que Routeurs simples de PE. Le PE central près du RÉSEAU LOCAL partagé de service (**iguane**) est configuré pour NAT. Un groupe NAT simple est partagé par chaque client VPN qui a besoin de l'accès au service partagé. Le NAT est exécuté seulement sur des paquets destinés pour l'hôte partagé de service chez 88.1.88.8.

Expédition NAT de données de PE de sortie

Avec le MPLS, chaque paquet entre dans le réseau à un PE d'entrée et quitte le réseau MPLS à un PE de sortie. Le chemin des Routeurs de commutation par étiquette traversés du d'entrée au de sortie est connu comme chemin commuté par étiquette (LSP). Le LSP est unidirectionnel. Un LSP différent est utilisé pour le trafic de retour.

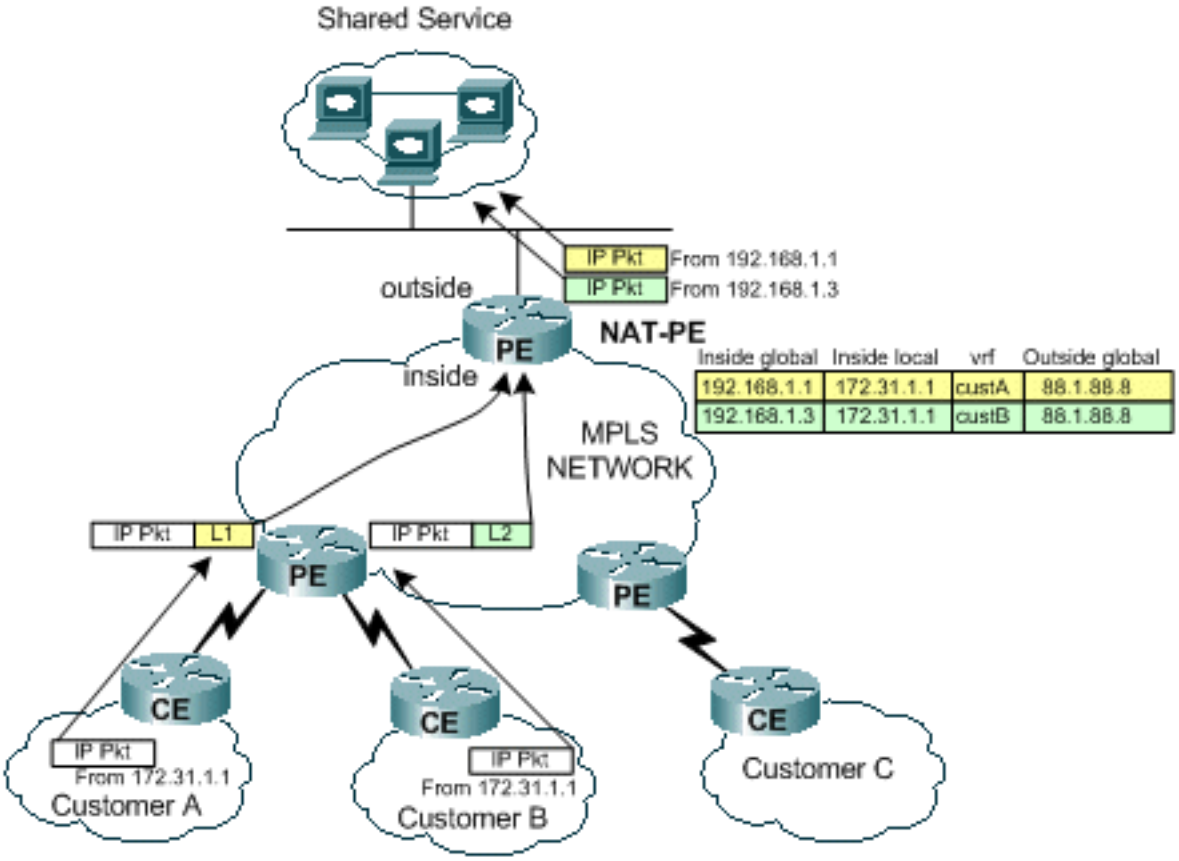
En utilisant le PE de sortie NAT, une classe d'équivalence d'expédition (FEC) est efficacement définie pour tout le trafic des utilisateurs du service partagé. En d'autres termes, tous les paquets destinés pour le RÉSEAU LOCAL partagé de service sont des membres d'une FEC commune. Un paquet est assigné à une FEC particulière juste une fois à la périphérie d'entrée du réseau et suit

le LSP au PE de sortie. La FEC est indiquée dans le paquet de données en ajoutant une étiquette particulière.

Écoulement de paquet au service partagé du VPN

Afin des périphériques dans le multiple VPN qui ont superposer l'adresse complote pour accéder à un hôte partagé de service, NAT est exigée. Si NAT est configuré au PE de sortie, les entrées de table de traduction d'adresses réseau incluront un identifiant de VRF pour différencier des adresses en double et pour assurer le routage approprié.

Figure 6 : Paquets transmis au PE de sortie NAT



La figure 6 montre les paquets destinés pour un serveur partagé de service deux du client VPNs qui ont des systèmes d'adressage d'IP en double. La figure affiche qu'un paquet commençant au client A avec une adresse source de 172.31.1.1 a destiné pour un serveur partagé chez 88.1.88.8. Un autre paquet du client B avec la même adresse IP source est également envoyé à la même chose serveur partagé. Quand les paquets atteignent le routeur PE, une consultation de la couche 3 est faite pour le réseau IP de destination dans le Forwarding Information Base (FIB).

L'entrée de FIB indique le routeur PE expédier le trafic au PE de sortie utilisant une pile d'étiquette. L'étiquette inférieure dans la pile est assignée par le routeur PE de destination, dans ce cas **iguane de** routeur.

```
iguana# show ip cef vrf custA 88.1.88.8 88.1.88.8/32, version 47, epoch 0, cached adjacency
88.1.3.2 0 packets, 0 bytes tag information set local tag: VPN-route-head fast tag rewrite with
Et1/0, 88.1.3.2, tags imposed: {24} via 88.1.11.5, 0 dependencies, recursive next hop 88.1.3.2,
Ethernet1/0 via 88.1.11.5/32 valid cached adjacency tag rewrite with Et1/0, 88.1.3.2, tags
imposed: {24} iguana# show ip cef vrf custB 88.1.88.8 88.1.88.8/32, version 77, epoch 0, cached
adjacency 88.1.3.2 0 packets, 0 bytes tag information set local tag: VPN-route-head fast tag
rewrite with Et1/0, 88.1.3.2, tags imposed: {28} via 88.1.11.5, 0 dependencies, recursive next
hop 88.1.3.2, Ethernet1/0 via 88.1.11.5/32 valid cached adjacency tag rewrite with Et1/0,
88.1.3.2, tags imposed: {28} iguana#
```

Nous pouvons voir de l'affichage que des paquets de custA de VRF aurons une valeur de balise de 24 (0x18) et les paquets du custB de VRF auront une valeur de balise de 28 (0x1C).

Dans ce cas, parce qu'il n'y a aucun Routeurs « P » dans notre réseau, là n'est aucune balise supplémentaire imposée. Il y avait eu de principaux Routeurs, une étiquette extérieure aurait été imposée et le processus normal de l'échange d'étiquette aurait eu lieu dans le principal réseau jusqu'à ce que le paquet ait atteint le PE de sortie.

Puisque le routeur du **Gila** est directement connecté au PE de sortie, nous voyons que la balise est sautée avant qu'on l'ajoute jamais :

```
gila# show tag-switching forwarding-table Local Outgoing Prefix Bytes tag Outgoing Next Hop tag
tag or VC or Tunnel Id switched interface 16 Pop tag 88.1.1.0/24 0 Et1/1 88.1.2.2 Pop tag
88.1.1.0/24 0 Et1/0 88.1.3.2 17 Pop tag 88.1.4.0/24 0 Et1/1 88.1.2.2 18 Pop tag 88.1.10.0/24 0
Et1/1 88.1.2.2 19 Pop tag 88.1.11.1/32 0 Et1/1 88.1.2.2 20 Pop tag 88.1.5.0/24 0 Et1/0 88.1.3.2
21 19 88.1.11.10/32 0 Et1/1 88.1.2.2 22 88.1.11.10/32 0 Et1/0 88.1.3.2 22 20 172.18.60.176/32 0
Et1/1 88.1.2.2 23 172.18.60.176/32 0 Et1/0 88.1.3.2 23 Untagged 172.31.1.0/24[V] 4980 Fa0/0
10.88.162.6 24 Aggregate 10.88.162.4/30[V] 1920 25 Aggregate 10.88.162.8/30[V] 137104 26
Untagged 172.31.1.0/24[V] 570 Et1/2 10.88.162.14 27 Aggregate 10.88.162.12/30[V] \ 273480 30 Pop
tag 88.1.11.5/32 0 Et1/0 88.1.3.2 31 Pop tag 88.1.88.0/24 0 Et1/0 88.1.3.2 32 16 88.1.97.0/24 0
Et1/0 88.1.3.2 33 Pop tag 88.1.99.0/24 0 Et1/0 88.1.3.2 gila# gila# show tag-switching
forwarding-table 88.1.88.0 detail Local Outgoing Prefix Bytes tag Outgoing Next Hop tag tag or
VC or Tunnel Id switched interface 31 Pop tag 88.1.88.0/24 0 Et1/0 88.1.3.2 MAC/Encaps=14/14,
MRU=1504, Tag Stack{} 005054D92A250090BF9C6C1C8847 No output feature configured Per-packet load-
sharing gila#
```

Les prochains affichages dépeignent des paquets d'écho comme reçu par le routeur NAT de PE de sortie (à interface E1/0/5 sur l'**iguane**).

```
From CustA: DLC: ----- DLC Header ----- DLC: DLC: Frame 1 arrived at 16:21:34.8415; frame size
is 118 (0076 hex) bytes. DLC: Destination = Station 005054D92A25 DLC: Source = Station
0090BF9C6C1C DLC: Ethertype = 8847 (MPLS) DLC: MPLS: ----- MPLS Label Stack ----- MPLS: MPLS:
Label Value = 00018 MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1 (Bottom of
Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4,
header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal
delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT
bit - transport protocol will ignore the CE bit IP: .... ...0 = CE bit - no congestion IP: Total
length = 100 bytes IP: Identification = 175 IP: Flags = 0X IP: .0.. .... = may fragment IP: ..0.
.... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP:
Protocol = 1 (ICMP) IP: Header checksum = 5EC0 (correct) IP: Source address = [172.31.1.1] IP:
Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP:
Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = 4AF1 (correct) ICMP: Identifier = 4713 ICMP:
Sequence number = 6957 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".]
From CustB: DLC: ----- DLC Header ----- DLC: DLC: Frame 11 arrived at 16:21:37.1558; frame size
is 118 (0076 hex) bytes. DLC: Destination = Station 005054D92A25 DLC: Source = Station
0090BF9C6C1C DLC: Ethertype = 8847 (MPLS) DLC: MPLS: ----- MPLS Label Stack ----- MPLS: MPLS:
Label Value = 0001C MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1 (Bottom of
Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4,
header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal
delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT
bit - transport protocol will ignore the CE bit IP: .... ...0 = CE bit - no congestion IP: Total
length = 100 bytes IP: Identification = 165 IP: Flags = 0X IP: .0.. .... = may fragment IP: ..0.
.... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP:
Protocol = 1 (ICMP) IP: Header checksum = 5ECA (correct) IP: Source address = [172.31.1.1] IP:
Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP:
Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = AD5E (correct) ICMP: Identifier = 3365 ICMP:
Sequence number = 7935 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".]
```

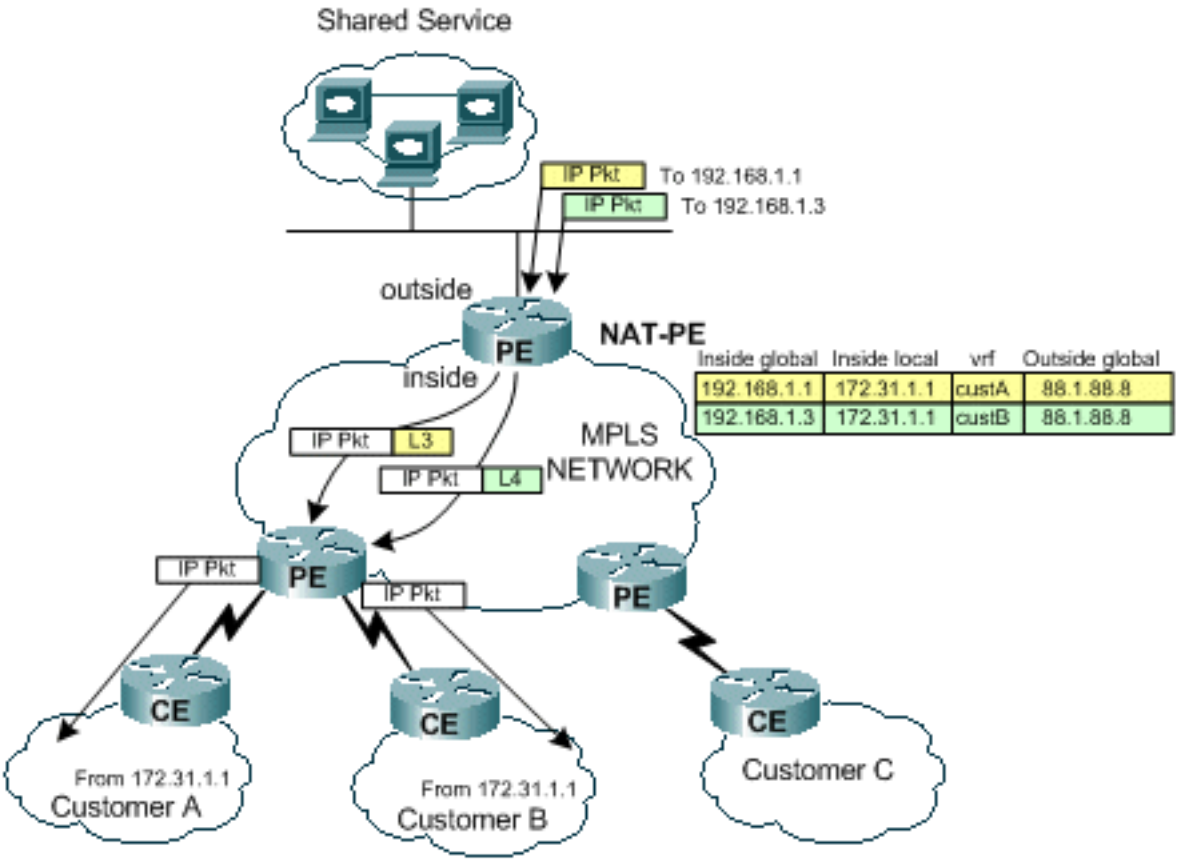
Ces pings ont comme conséquence les entrées suivantes étant créées dans la table NAT dans l'**iguane de** routeur PE de sortie. Les entrées spécifiques créées pour les paquets affichés ci-dessus peuvent être appariées par leur identifiant d'ICMP.

```
iguana# show ip nat translations Pro Inside global Inside local Outside local Outside global
icmp 192.168.1.3:3365 172.31.1.1:3365 88.1.88.8:3365 88.1.88.8:3365 icmp 192.168.1.3:3366
172.31.1.1:3366 88.1.88.8:3366 88.1.88.8:3366 icmp 192.168.1.3:3367 172.31.1.1:3367
88.1.88.8:3367 88.1.88.8:3367 icmp 192.168.1.3:3368 172.31.1.1:3368 88.1.88.8:3368
88.1.88.8:3368 icmp 192.168.1.3:3369 172.31.1.1:3369 88.1.88.8:3369 88.1.88.8:3369 icmp
192.168.1.1:4713 172.31.1.1:4713 88.1.88.8:4713 88.1.88.8:4713 icmp 192.168.1.1:4714
172.31.1.1:4714 88.1.88.8:4714 88.1.88.8:4714 icmp 192.168.1.1:4715 172.31.1.1:4715
88.1.88.8:4715 88.1.88.8:4715 icmp 192.168.1.1:4716 172.31.1.1:4716 88.1.88.8:4716
88.1.88.8:4716 icmp 192.168.1.1:4717 172.31.1.1:4717 88.1.88.8:4717 88.1.88.8:4717 iguana# show
ip nat translations verbose Pro Inside global Inside local Outside local Outside global icmp
192.168.1.3:3365 172.31.1.1:3365 88.1.88.8:3365 88.1.88.8:3365 create 00:00:34, use 00:00:34,
left 00:00:25, Map-Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp 192.168.1.3:3366
172.31.1.1:3366 88.1.88.8:3366 88.1.88.8:3366 create 00:00:34, use 00:00:34, left 00:00:25, Map-
Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp 192.168.1.3:3367 172.31.1.1:3367
88.1.88.8:3367 88.1.88.8:3367 create 00:00:34, use 00:00:34, left 00:00:25, Map-Id(In): 2,
flags: extended, use_count: 0, VRF : custB icmp 192.168.1.3:3368 172.31.1.1:3368 88.1.88.8:3368
88.1.88.8:3368 create 00:00:34, use 00:00:34, left 00:00:25, Map-Id(In): 2, flags: extended,
use_count: 0, VRF : custB icmp 192.168.1.3:3369 172.31.1.1:3369 88.1.88.8:3369 88.1.88.8:3369
create 00:00:34, use 00:00:34, left 00:00:25, Map-Id(In): 2, flags: extended, use_count: 0, VRF
: custB icmp 192.168.1.1:4713 172.31.1.1:4713 88.1.88.8:4713 88.1.88.8:4713 create 00:00:37, use
00:00:37, left 00:00:22, Map-Id(In): 1, Pro Inside global Inside local Outside local Outside
global flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:4714 172.31.1.1:4714
88.1.88.8:4714 88.1.88.8:4714 create 00:00:37, use 00:00:37, left 00:00:22, Map-Id(In): 1,
flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:4715 172.31.1.1:4715 88.1.88.8:4715
88.1.88.8:4715 create 00:00:37, use 00:00:37, left 00:00:22, Map-Id(In): 1, flags: extended,
use_count: 0, VRF : custA icmp 192.168.1.1:4716 172.31.1.1:4716 88.1.88.8:4716 88.1.88.8:4716
create 00:00:37, use 00:00:37, left 00:00:22, Map-Id(In): 1, flags: extended, use_count: 0, VRF
: custA icmp 192.168.1.1:4717 172.31.1.1:4717 88.1.88.8:4717 88.1.88.8:4717 create 00:00:37, use
00:00:37, left 00:00:22, Map-Id(In): 1, flags: extended, use_count: 0, VRF : custA iguana#
```

Écoulement de paquet de service partagé de nouveau à l'origine VPN

Pendant que les paquets circulent de nouveau aux périphériques qui ont accédé à l'hôte partagé de service, la table NAT est examinée avant le routage (paquets allant de l'interface NAT de « extérieur » à l'interface de « intérieur »). Puisque chaque seule entrée inclut l'identifiant correspondant de VRF, le paquet peut être traduit et conduit convenablement.

Figure 7 : Paquets transmis de nouveau à l'utilisateur de services partagé



Suivant les indications de la figure 7, le trafic de retour est d'abord examiné par NAT pour trouver une entrée assortie de traduction. Par exemple, un paquet est envoyé à la destination 192.168.1.1. La table NAT est recherchée. Quand la correspondance est trouvée, la traduction appropriée est faite à l'adresse de « interne local » (172.31.1.1) et alors une consultation de contiguïté est exécutée utilisant l'ID de VRF associé de l'entrée NAT.

```
iguana# show ip cef vrf custA 172.31.1.0 172.31.1.0/24, version 12, epoch 0, cached adjacency
88.1.3.1 0 packets, 0 bytes tag information set local tag: VPN-route-head fast tag rewrite with
Et1/0/5, 88.1.3.1, tags imposed: {23} via 88.1.11.9, 0 dependencies, recursive next hop
88.1.3.1, Ethernet1/0/5 via 88.1.11.9/32 valid cached adjacency tag rewrite with Et1/0/5,
88.1.3.1, tags imposed: {23} iguana# show ip cef vrf custB 172.31.1.0 172.31.1.0/24, version 18,
epoch 0, cached adjacency 88.1.3.1 0 packets, 0 bytes tag information set local tag: VPN-route-
head fast tag rewrite with Et1/0/5, 88.1.3.1, tags imposed: {26} via 88.1.11.9, 0 dependencies,
recursive next hop 88.1.3.1, Ethernet1/0/5 via 88.1.11.9/32 valid cached adjacency tag rewrite
with Et1/0/5, 88.1.3.1, tags imposed: {26} iguana#
```

L'étiquette 23 (0x17) est utilisée pour le trafic destiné pour 172.31.1.0/24 dans le custA et l'étiquette 26 (0x1A) de VRF est utilisée pour des paquets destinés pour 172.31.1.0/24 dans le custB de VRF.

Ceci est vu dans les paquets de réponse d'écho envoyés de l'**iguane de** routeur :

```
To custA: DLC: ----- DLC Header ----- DLC: DLC: Frame 2 arrived at 16:21:34.8436; frame size is
118 (0076 hex) bytes. DLC: Destination = Station 0090BF9C6C1C DLC: Source = Station 005054D92A25
DLC: Ethertype = 8847 (MPLS) DLC: MPLS: ----- MPLS Label Stack ----- MPLS: MPLS: Label Value =
00017 MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1 (Bottom of Stack) MPLS: Time
to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4, header length = 20
bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal delay IP: ....
0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT bit - transport
protocol will ignore the CE bit IP: .... ...0 = CE bit - no congestion IP: Total length = 100
bytes IP: Identification = 56893 IP: Flags = 4X IP: .1.. .... = don't fragment IP: ..0. .... =
last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP: Protocol = 1
(ICMP) IP: Header checksum = 4131 (correct) IP: Source address = [88.1.88.8] IP: Destination
```

address = [172.31.1.1] IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP: Type = 0 (Echo reply) ICMP: Code = 0 ICMP: Checksum = 52F1 (correct) ICMP: Identifier = 4713 ICMP: Sequence number = 6957 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".]

Quand le paquet atteint le routeur PE de destination, l'étiquette est utilisée pour déterminer le VRF et l'interface appropriés pour envoyer le paquet plus de.

gila# **show mpls forwarding-table** Local Outgoing Prefix Bytes tag Outgoing Next Hop tag tag or VC or Tunnel Id switched interface 16 Pop tag 88.1.1.0/24 0 Et1/1 88.1.2.2 Pop tag 88.1.1.0/24 0 Et1/0 88.1.3.2 17 Pop tag 88.1.4.0/24 0 Et1/1 88.1.2.2 18 Pop tag 88.1.10.0/24 0 Et1/1 88.1.2.2 19 Pop tag 88.1.11.1/32 0 Et1/1 88.1.2.2 20 Pop tag 88.1.5.0/24 0 Et1/0 88.1.3.2 21 19 88.1.11.10/32 0 Et1/1 88.1.2.2 22 88.1.11.10/32 0 Et1/0 88.1.3.2 22 20 172.18.60.176/32 0 Et1/1 88.1.2.2 23 172.18.60.176/32 0 Et1/0 88.1.3.2 **23 Untagged 172.31.1.0/24[V] 6306 Fa0/0 10.88.162.6** 24 Aggregate 10.88.162.4/30[V] 1920 25 Aggregate 10.88.162.8/30[V] 487120 **26 Untagged 172.31.1.0/24[V] 1896 Et1/2 10.88.162.14** 27 Aggregate 10.88.162.12/30[V] \ 972200 30 Pop tag 88.1.11.5/32 0 Et1/0 88.1.3.2 31 Pop tag 88.1.88.0/24 0 Et1/0 88.1.3.2 32 16 88.1.97.0/24 0 Et1/0 88.1.3.2 33 Pop tag 88.1.99.0/24 0 Et1/0 88.1.3.2 gila#

Configurations

Quelques informations étrangères ont été enlevées des configurations par souci de concision.

```
IGUANA:
!
ip vrf custA
 rd 65002:100
 route-target export 65002:100
 route-target import 65002:100
!
ip vrf custB
 rd 65002:200
 route-target export 65002:200
 route-target import 65002:200
!
ip cef
mpls label protocol ldp
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 88.1.11.5 255.255.255.255
 no ip route-cache
 no ip mroute-cache
!
interface Loopback11
 ip vrf forwarding custA
 ip address 172.16.1.1 255.255.255.255
!
interface Ethernet1/0/0
 ip vrf forwarding custB
 ip address 10.88.163.5 255.255.255.252
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet1/0/4
 ip address 88.1.1.1 255.255.255.0
 ip nat inside
 no ip mroute-cache
 tag-switching ip
!
interface Ethernet1/0/5
 ip address 88.1.3.2 255.255.255.0
 ip nat inside
 no ip mroute-cache
 tag-switching ip
```

```
!
!
interface FastEthernet1/1/0
 ip address 88.1.88.1 255.255.255.0
 ip nat outside
 full-duplex
!
interface FastEthernet5/0/0
 ip address 88.1.99.1 255.255.255.0
 speed 100
 full-duplex
!
router ospf 881
 log-adjacency-changes
 redistribute static subnets
 network 88.1.0.0 0.0.255.255 area 0
!
router bgp 65002
 no synchronization
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 88.1.11.1 remote-as 65002
 neighbor 88.1.11.1 update-source Loopback0
 neighbor 88.1.11.9 remote-as 65002
 neighbor 88.1.11.9 update-source Loopback0
 neighbor 88.1.11.10 remote-as 65002
 neighbor 88.1.11.10 update-source Loopback0
 no auto-summary
 !
 address-family ipv4 multicast
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family vpnv4
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.1 send-community extended
 neighbor 88.1.11.9 activate
 neighbor 88.1.11.9 send-community extended
 no auto-summary
 exit-address-family
 !
 address-family ipv4
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.9 activate
 neighbor 88.1.11.10 activate
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf custB
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf custA
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
!
ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length 24
```

```
ip nat inside source list 181 pool SSPOOL1 vrf custA overload
ip nat inside source list 181 pool SSPOOL1 vrf custB overload
ip classless
ip route 88.1.88.0 255.255.255.0 FastEthernet1/1/0
ip route 88.1.97.0 255.255.255.0 FastEthernet5/0/0 88.1.99.2
ip route 88.1.99.0 255.255.255.0 FastEthernet5/0/0 88.1.99.2
ip route 192.168.1.0 255.255.255.0 Null0
ip route vrf custA 88.1.88.8 255.255.255.255 FastEthernet1/1/0 88.1.88.8 global
ip route vrf custB 10.88.208.0 255.255.240.0 10.88.163.6
ip route vrf custB 64.102.0.0 255.255.0.0 10.88.163.6
ip route vrf custB 88.1.88.8 255.255.255.255 FastEthernet1/1/0 88.1.88.8 global
ip route vrf custB 128.0.0.0 255.0.0.0 10.88.163.6
no ip http server
!
access-list 181 permit ip any host 88.1.88.8
!
GILA:

!
ip vrf custA
 rd 65002:100
 route-target export 65002:100
 route-target import 65002:100
!
ip vrf custB
 rd 65002:200
 route-target export 65002:200
 route-target import 65002:200
!
ip cef
mpls label protocol ldp
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 88.1.11.9 255.255.255.255
!
interface FastEthernet0/0
 ip vrf forwarding custA
 ip address 10.88.162.5 255.255.255.252
 duplex full
!
interface Ethernet1/0
 ip address 88.1.3.1 255.255.255.0
 no ip mroute-cache
 duplex half
 tag-switching ip
!
interface Ethernet1/1
 ip address 88.1.2.1 255.255.255.0
 no ip mroute-cache
 duplex half
 tag-switching ip
!
interface Ethernet1/2
 ip vrf forwarding custB
 ip address 10.88.162.13 255.255.255.252
 ip ospf cost 100
 duplex half
!
interface FastEthernet2/0
 ip vrf forwarding custA
 ip address 10.88.162.9 255.255.255.252
 duplex full
!
```

```
router ospf 881
 log-adjacency-changes
 redistribute static subnets
 network 88.1.0.0 0.0.255.255 area 0
 default-metric 30
!
router bgp 65002
 no synchronization
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 88.1.11.1 remote-as 65002
 neighbor 88.1.11.1 update-source Loopback0
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.5 remote-as 65002
 neighbor 88.1.11.5 update-source Loopback0
 neighbor 88.1.11.5 activate
 no auto-summary
 !
 address-family ipv4 vrf custB
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf custA
 redistribute connected
 redistribute static
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family vpnv4
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.1 send-community extended
 neighbor 88.1.11.5 activate
 neighbor 88.1.11.5 send-community extended
 no auto-summary
 exit-address-family
!
ip classless
ip route vrf custA 172.31.1.0 255.255.255.0 FastEthernet0/0 10.88.162.6
ip route vrf custB 172.31.1.0 255.255.255.0 Ethernet1/2 10.88.162.14
!
```

Le dragon de routeur aurait une configuration très semblable au **Gila**.

Importation/exportation des cibles d'artère non permises

Quand le réseau de service partagé est configuré comme exemple de VRF lui-même, NAT central au PE de sortie n'est pas possible. C'est parce que les paquets entrant ne peuvent pas être distingués et seulement une route de retour vers le sous-réseau d'origine est présente au PE de sortie NAT.

Remarque: Les affichages qui suivent sont censés pour illustrer le résultat d'une configuration non valide.

Le réseau témoin a été configuré de sorte que le réseau de service partagé ait été défini comme exemple de VRF (nom = sserver de VRF). Maintenant, un affichage de la table CEF sur le PE d'entrée affiche ceci :

```
gila# show ip cef vrf custA 88.1.88.0 88.1.88.0/24, version 45, epoch 0, cached adjacency
88.1.3.2 0 packets, 0 bytes tag information set local tag: VPN-route-head fast tag rewrite with
Et1/0, 88.1.3.2, tags imposed: {24} via 88.1.11.5, 0 dependencies, recursive next hop 88.1.3.2,
Ethernet1/0 via 88.1.11.5/32 valid cached adjacency tag rewrite with Et1/0, 88.1.3.2, tags
imposed: {24} gila# gila# show ip cef vrf custB 88.1.88.0 88.1.88.0/24, version 71, epoch 0,
cached adjacency 88.1.3.2 0 packets, 0 bytes tag information set local tag: VPN-route-head fast
tag rewrite with Et1/0, 88.1.3.2, tags imposed: {24} via 88.1.11.5, 0 dependencies, recursive
next hop 88.1.3.2, Ethernet1/0 via 88.1.11.5/32 valid cached adjacency tag rewrite with Et1/0,
88.1.3.2, tags imposed: {24} gila# iguana# show tag-switching forwarding vrftags 24 Local
Outgoing Prefix Bytes tag Outgoing Next Hop tag tag or VC or Tunnel Id switched interface 24
Aggregate 88.1.88.0/24[V] 10988 iguana#
```

Remarque: Avis comment la valeur *24 de* balise est imposée pour le custA de VRF et le custB de VRF.

Cet affichage affiche la table de routage pour l'exemple partagé « sserver » de VRF de service :

```
iguana# show ip route vrf sserver 172.31.1.1 Routing entry for 172.31.1.0/24 Known via "bgp
65002", distance 200, metric 0, type internal Last update from 88.1.11.9 1d01h ago Routing
Descriptor Blocks: * 88.1.11.9 (Default-IP-Routing-Table), from 88.1.11.9, 1d01h ago Route
metric is 0, traffic share count is 1 AS Hops 0
```

Remarque: Seulement une artère est présente pour le réseau de destination du point de vue du routeur de PE de sortie (**iguane**).

Par conséquent, le trafic des plusieurs clients VPN ne pourrait pas être distingué et le trafic de retour ne pourrait pas atteindre le VPN approprié. **Dans le cas où le service partagé doit être défini comme exemple de VRF, la fonction NAT doit être déplacée au PE d'entrée.**

PE d'entrée NAT

Dans cet exemple, les Routeurs de Provider Edge le **Gila** marqué et le **dragon** sont configurés pour NAT. Un groupe NAT est défini pour chaque client relié VPN qui a besoin de l'accès au service partagé. Le groupe approprié est utilisé pour chacune des adresses de réseau client qui sont NATed. Le NAT est exécuté seulement sur des paquets destinés pour l'hôte partagé de service chez 88.1.88.8.

```
ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length 24 ip nat pool SSPOOL2 192.168.2.1
192.168.2.254 prefix-length 24 ip nat inside source list 181 pool SSPOOL1 vrf custA overload ip
nat inside source list 181 pool SSPOOL2 vrf custB overload
```

Remarque: Dans ce scénario, des groupes partagés ne sont pas pris en charge. Si le RÉSEAU LOCAL partagé de service (au PE de sortie) est connecté par une interface générique, alors le groupe NAT peut être partagé.

Un ping originaire d'une adresse en double (172.31.1.1) dans chacun des réseaux s'est relié au **neuse** et aux résultats **capefear8** dans ces entrées NAT :

Du Gila :

```
gila# show ip nat translations Pro Inside global Inside local Outside local Outside global icmp
192.168.1.1:2139 172.31.1.1:2139 88.1.88.8:2139 88.1.88.8:2139 icmp 192.168.1.1:2140
172.31.1.1:2140 88.1.88.8:2140 88.1.88.8:2140 icmp 192.168.1.1:2141 172.31.1.1:2141
88.1.88.8:2141 88.1.88.8:2141 icmp 192.168.1.1:2142 172.31.1.1:2142 88.1.88.8:2142
88.1.88.8:2142 icmp 192.168.1.1:2143 172.31.1.1:2143 88.1.88.8:2143 88.1.88.8:2143 icmp
192.168.2.2:676 172.31.1.1:676 88.1.88.8:676 88.1.88.8:676 icmp 192.168.2.2:677 172.31.1.1:677
88.1.88.8:677 88.1.88.8:677 icmp 192.168.2.2:678 172.31.1.1:678 88.1.88.8:678 88.1.88.8:678 icmp
192.168.2.2:679 172.31.1.1:679 88.1.88.8:679 88.1.88.8:679 icmp 192.168.2.2:680 172.31.1.1:680
88.1.88.8:680 88.1.88.8:680
```

Remarque: La même adresse d'interne local (172.31.1.1) est traduite à chacun des groupes

définis selon le VRF de source. Le VRF peut être vu dans la commande **bavarde de traduction nat de show ip** :

```
gila# show ip nat translations verbose Pro Inside global Inside local Outside local Outside
global icmp 192.168.1.1:2139 172.31.1.1:2139 88.1.88.8:2139 88.1.88.8:2139 create 00:00:08, use
00:00:08, left 00:00:51, Map-Id(In): 3, flags: extended, use_count: 0, VRF : custA icmp
192.168.1.1:2140 172.31.1.1:2140 88.1.88.8:2140 88.1.88.8:2140 create 00:00:08, use 00:00:08,
left 00:00:51, Map-Id(In): 3, flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:2141
172.31.1.1:2141 88.1.88.8:2141 88.1.88.8:2141 create 00:00:08, use 00:00:08, left 00:00:51, Map-
Id(In): 3, flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:2142 172.31.1.1:2142
88.1.88.8:2142 88.1.88.8:2142 create 00:00:08, use 00:00:08, left 00:00:51, Map-Id(In): 3,
flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:2143 172.31.1.1:2143 88.1.88.8:2143
88.1.88.8:2143 create 00:00:08, use 00:00:08, left 00:00:51, Map-Id(In): 3, flags: extended,
use_count: 0, VRF : custA icmp 192.168.2.2:676 172.31.1.1:676 88.1.88.8:676 88.1.88.8:676 create
00:00:10, use 00:00:10, left 00:00:49, Map-Id(In): 2, flags: extended, use_count: 0, VRF : custB
icmp 192.168.2.2:677 172.31.1.1:677 88.1.88.8:677 88.1.88.8:677 create 00:00:10, use 00:00:10,
left 00:00:49, Map-Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp 192.168.2.2:678
172.31.1.1:678 88.1.88.8:678 88.1.88.8:678 create 00:00:10, use 00:00:10, left 00:00:49, Map-
Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp 192.168.2.2:679 172.31.1.1:679
88.1.88.8:679 88.1.88.8:679 create 00:00:10, use 00:00:10, left 00:00:49, Map-Id(In): 2, flags:
extended, use_count: 0, VRF : custB icmp 192.168.2.2:680 172.31.1.1:680 88.1.88.8:680
88.1.88.8:680 create 00:00:10, use 00:00:10, left 00:00:49, Map-Id(In): 2, flags: extended,
use_count: 0, VRF : custB
```

Ces affichages affichent les informations de routage pour chacun des VPN localement reliés pour le client A et le client B :

```
gila# show ip route vrf custA Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP D -
EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 -
OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 I - IS-IS, L1 -
IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area * - candidate default, U - per-user
static route, o - ODR P - periodic downloaded static route Gateway of last resort is 88.1.11.1
to network 0.0.0.0     172.18.0.0/32 is subnetted, 2 subnets
B       172.18.60.179 [200/0] via 88.1.11.1, 00:03:59
B       172.18.60.176 [200/0] via 88.1.11.1, 00:03:59
     172.31.0.0/24 is subnetted, 1 subnets
S       172.31.1.0 [1/0] via 10.88.162.6, FastEthernet0/0
     10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
B       10.88.0.0/20 [200/0] via 88.1.11.1, 00:03:59
B       10.88.32.0/20 [200/0] via 88.1.11.1, 00:03:59
C       10.88.162.4/30 is directly connected, FastEthernet0/0
C       10.88.162.8/30 is directly connected, FastEthernet2/0
B       10.88.161.8/30 [200/0] via 88.1.11.1, 00:04:00
     88.0.0.0/24 is subnetted, 2 subnets
B       88.1.88.0 [200/0] via 88.1.11.5, 00:04:00
B       88.1.99.0 [200/0] via 88.1.11.5, 00:04:00
S 192.168.1.0/24 is directly connected, Null0 B* 0.0.0.0/0 [200/0] via 88.1.11.1, 00:04:00 gila#
show ip route vrf custB Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP D -
EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 -
OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 I - IS-IS, L1 -
IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area * - candidate default, U - per-user
static route, o - ODR P - periodic downloaded static route Gateway of last resort is not set
64.0.0.0/16 is subnetted, 1 subnets
B       64.102.0.0 [200/0] via 88.1.11.5, 1d21h
     172.18.0.0/32 is subnetted, 2 subnets
B       172.18.60.179 [200/0] via 88.1.11.1, 1d21h
B       172.18.60.176 [200/0] via 88.1.11.1, 1d21h
     172.31.0.0/24 is subnetted, 1 subnets
S       172.31.1.0 [1/0] via 10.88.162.14, Ethernet1/2
     10.0.0.0/8 is variably subnetted, 6 subnets, 3 masks
B       10.88.194.16/28 [200/100] via 88.1.11.1, 1d20h
B       10.88.208.0/20 [200/0] via 88.1.11.5, 1d21h
B       10.88.194.4/30 [200/100] via 88.1.11.1, 1d20h
```

```
B       10.88.163.4/30 [200/0] via 88.1.11.5, 1d21h
B       10.88.161.4/30 [200/0] via 88.1.11.1, 1d21h
C       10.88.162.12/30 is directly connected, Ethernet1/2
     11.0.0.0/24 is subnetted, 1 subnets
B       11.1.1.0 [200/100] via 88.1.11.1, 1d20h
     88.0.0.0/24 is subnetted, 2 subnets
B       88.1.88.0 [200/0] via 88.1.11.5, 1d21h
B       88.1.99.0 [200/0] via 88.1.11.5, 1d21h
S    192.168.2.0/24 is directly connected, Null0 B 128.0.0.0/8 [200/0] via 88.1.11.5, 1d21h
```

Remarque: Une artère pour chacun des groupes NAT a été ajoutée de la configuration statique. Ces sous-réseaux sont ultérieurement importés dans le VRF partagé de serveur à l'iguane de routeur PE de sortie :

```
iguana# show ip route vrf sserver Routing Table: sserver
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       I - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
Gateway of last resort is not set     64.0.0.0/16 is subnetted, 1 subnets
B       64.102.0.0 [20/0] via 10.88.163.6 (custB), 1d20h
     172.18.0.0/32 is subnetted, 2 subnets
B       172.18.60.179 [200/0] via 88.1.11.1, 1d20h
B       172.18.60.176 [200/0] via 88.1.11.1, 1d20h
     172.31.0.0/24 is subnetted, 1 subnets
B       172.31.1.0 [200/0] via 88.1.11.9, 1d05h
     10.0.0.0/8 is variably subnetted, 8 subnets, 3 masks
B       10.88.194.16/28 [200/100] via 88.1.11.1, 1d20h
B       10.88.208.0/20 [20/0] via 10.88.163.6 (custB), 1d20h
B       10.88.194.4/30 [200/100] via 88.1.11.1, 1d20h
B       10.88.162.4/30 [200/0] via 88.1.11.9, 1d20h
B       10.88.163.4/30 is directly connected, 1d20h, Ethernet1/0/0
B       10.88.161.4/30 [200/0] via 88.1.11.1, 1d20h
B       10.88.162.8/30 [200/0] via 88.1.11.9, 1d20h
B       10.88.162.12/30 [200/0] via 88.1.11.9, 1d20h
     11.0.0.0/24 is subnetted, 1 subnets
B       11.1.1.0 [200/100] via 88.1.11.1, 1d20h
     12.0.0.0/24 is subnetted, 1 subnets
S       12.12.12.0 [1/0] via 88.1.99.10
     88.0.0.0/24 is subnetted, 3 subnets
C       88.1.88.0 is directly connected, FastEthernet1/1/0
S       88.1.97.0 [1/0] via 88.1.99.10
C       88.1.99.0 is directly connected, FastEthernet5/0/0
B 192.168.1.0/24 [200/0] via 88.1.11.9, 1d20h B 192.168.2.0/24 [200/0] via 88.1.11.9, 01:59:23 B
128.0.0.0/8 [20/0] via 10.88.163.6 (custB), 1d20h
```

Configurations

Quelques informations étrangères ont été enlevées des configurations par souci de concision.

```
GILA:
ip vrf custA
 rd 65002:100
 route-target export 65002:100
 route-target export 65002:1001
 route-target import 65002:100
!
ip vrf custB
 rd 65002:200
 route-target export 65002:200
 route-target export 65002:2001
```

```
 route-target import 65002:200
 route-target import 65002:10
!
ip cef
mpls label protocol ldp
!interface Loopback0
 ip address 88.1.11.9 255.255.255.255
!
interface FastEthernet0/0
 ip vrf forwarding custA ip address 10.88.162.5 255.255.255.252 ip nat inside duplex full !
interface Ethernet1/0 ip address 88.1.3.1 255.255.255.0 ip nat outside no ip mroute-cache duplex
half tag-switching ip ! interface Ethernet1/1 ip address 88.1.2.1 255.255.255.0 ip nat outside
no ip mroute-cache duplex half tag-switching ip ! interface Ethernet1/2 ip vrf forwarding custB
ip address 10.88.162.13 255.255.255.252 ip nat inside duplex half ! router ospf 881 log-
adjacency-changes redistribute static subnets network 88.1.0.0 0.0.255.255 area 0 default-metric
30 ! router bgp 65002 no synchronization no bgp default ipv4-unicast bgp log-neighbor-changes
neighbor 88.1.11.1 remote-as 65002 neighbor 88.1.11.1 update-source Loopback0 neighbor 88.1.11.1
activate neighbor 88.1.11.5 remote-as 65002 neighbor 88.1.11.5 update-source Loopback0 neighbor
88.1.11.5 activate no auto-summary ! address-family ipv4 vrf custB redistribute connected
redistribute static no auto-summary no synchronization exit-address-family ! address-family ipv4
vrf custA redistribute connected redistribute static no auto-summary no synchronization exit-
address-family ! address-family vpnv4 neighbor 88.1.11.1 activate neighbor 88.1.11.1 send-
community extended neighbor 88.1.11.5 activate neighbor 88.1.11.5 send-community extended no
auto-summary exit-address-family ! ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length
24 ip nat pool SSPOOL2 192.168.2.1 192.168.2.254 prefix-length 24 ip nat inside source list 181
pool SSPOOL1 vrf custA overload ip nat inside source list 181 pool SSPOOL2 vrf custB overload ip
classless ip route vrf custA 172.31.1.0 255.255.255.0 FastEthernet0/0 10.88.162.6 ip route vrf
custA 192.168.1.0 255.255.255.0 Null0 ip route vrf custB 172.31.1.0 255.255.255.0 Ethernet1/2
10.88.162.14 ip route vrf custB 192.168.2.0 255.255.255.0 Null0 ! access-list 181 permit ip any
host 88.1.88.8 !
```

Remarque: Les interfaces qui font face aux réseaux client sont indiquées comme interfaces NAT de « intérieur » et interfaces MPLS sont indiquées en tant que « extérieur » NAT relie.

```
iguana:
ip vrf custB
 rd 65002:200
 route-target export 65002:200
 route-target export 65002:2001
 route-target import 65002:200
 route-target import 65002:10
!
ip vrf sserver
 rd 65002:10
 route-target export 65002:10
 route-target import 65002:2001
 route-target import 65002:1001
!
ip cef distributed
mpls label protocol ldp
!interface Loopback0
 ip address 88.1.11.5 255.255.255.255
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet1/0/0
 ip vrf forwarding custB
 ip address 10.88.163.5 255.255.255.252
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet1/0/4
 ip address 88.1.1.1 255.255.255.0
 no ip route-cache
```

```
 no ip mroute-cache
 tag-switching ip
!
interface Ethernet1/0/5
 ip address 88.1.3.2 255.255.255.0
 no ip route-cache
 no ip mroute-cache
 tag-switching ip
!
interface FastEthernet1/1/0
 ip vrf forwarding sserver
 ip address 88.1.88.1 255.255.255.0
 no ip route-cache
 no ip mroute-cache
 full-duplex
!
router ospf 881
 log-adjacency-changes
 redistribute static subnets
 network 88.1.0.0 0.0.255.255 area 0
!
router bgp 65002
 no synchronization
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 88.1.11.1 remote-as 65002
 neighbor 88.1.11.1 update-source Loopback0
 neighbor 88.1.11.9 remote-as 65002
 neighbor 88.1.11.9 update-source Loopback0
 neighbor 88.1.11.10 remote-as 65002
 neighbor 88.1.11.10 update-source Loopback0
 no auto-summary
 !
 address-family ipv4 multicast
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family vpnv4
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.1 send-community extended
 neighbor 88.1.11.9 activate
 neighbor 88.1.11.9 send-community extended
 no auto-summary
 exit-address-family
 !
 address-family ipv4
 neighbor 88.1.11.1 activate
 neighbor 88.1.11.9 activate
 neighbor 88.1.11.10 activate
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf sserver
 redistribute connected
 no auto-summary
 no synchronization
 exit-address-family
 !
 address-family ipv4 vrf custB
 redistribute connected
 redistribute static
 no auto-summary
```

```
 no synchronization
 exit-address-family
```

Le dragon de routeur aurait une configuration très semblable au **Gila**.

Paquets arrivant au PE central après le PE d'entrée NAT

Les suivis ci-dessous illustrent la condition requise pour de seuls groupes NAT quand le réseau de service partagé par destination est configuré comme exemple de VRF. De nouveau, référez-vous au diagramme dans la figure 5. Les paquets affichés ci-dessous ont été capturés pendant qu'ils entraient dans l'interface IP e1/0/5 MPLS à l'**iguane de** routeur.

Écho du client A VPN

Ici, nous voyons une requête d'écho provenir l'adresse IP source 172.31.1.1 dans le custA de VRF. L'adresse source a été traduite à 192.168.1.1 comme spécifiée par la configuration NAT :

```
ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length 24
ip nat inside source list 181 pool SSPOOL1 vrf custA overload
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 1 arrived at  09:15:29.8157; frame size is 118 (0076 hex)
            bytes.
        DLC:  Destination = Station 005054D92A25
        DLC:  Source      = Station 0090BF9C6C1C
        DLC:  Ethertype   = 8847 (MPLS)
        DLC:
  MPLS: ----- MPLS Label Stack  -----
        MPLS:
        MPLS: Label Value = 00019 MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1
(Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP:
Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0
.... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP:
.... ..0. = ECT bit - transport protocol will ignore the CE bit IP: .... ...0 = CE bit - no
congestion IP: Total length = 100 bytes IP: Identification = 0 IP: Flags = 0X IP: .0.. .... =
may fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254
seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = 4AE6 (correct) IP: Source address =
[192.168.1.1] IP: Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header -
---- ICMP: ICMP: Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = 932D (correct) ICMP: Identifier
= 3046 ICMP: Sequence number = 3245 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP
header".] ICMP:
```

Écho du client B VPN

Ici, nous voyons une requête d'écho provenir l'adresse IP source 172.31.1.1 dans le custB de VRF. L'adresse source a été traduite à 192.168.2.1 comme spécifiée par la configuration NAT :

```
ip nat pool SSPOOL2 192.168.2.1 192.168.2.254 prefix-length 24
ip nat inside source list 181 pool SSPOOL2 vrf custB overload
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 11 arrived at  09:15:49.6623; frame size is 118 (0076 hex)
            bytes.
        DLC:  Destination = Station 005054D92A25
        DLC:  Source      = Station 0090BF9C6C1C
        DLC:  Ethertype   = 8847 (MPLS)
        DLC:
  MPLS: ----- MPLS Label Stack  -----
        MPLS:
        MPLS: Label Value = 00019 MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1
```

(Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT bit - transport protocol will ignore the CE bit IP: ....0 = CE bit - no congestion IP: Total length = 100 bytes IP: Identification = 15 IP: Flags = 0X IP: .0.. .... = may fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = 49D6 (correct) **IP: Source address = [192.168.2.2]** IP: Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP: Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = AB9A (correct) ICMP: Identifier = 4173 ICMP: Sequence number = 4212 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".]

Remarque: La valeur de mpls label est *0019* dans chacun des deux paquets affichés ci-dessus.

Réponse d'écho au client A VPN

Ensuite, nous voyons une réponse d'écho allant de retour à l'adresse IP 192.168.1.1 de destination dans le custA de VRF. L'adresse de destination est traduite à 172.31.1.1 par la fonction NAT de PE d'entrée.

To VRF custA: DLC: ----- DLC Header ----- DLC: DLC: Frame 2 arrived at 09:15:29.8198; frame size is 118 (0076 hex) bytes. DLC: Destination = Station 0090BF9C6C1C DLC: Source = Station 005054D92A25 DLC: Ethertype = 8847 (MPLS) DLC: MPLS: ----- MPLS Label Stack ----- MPLS: **MPLS: Label Value = 0001A** MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1 (Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT bit - transport protocol will ignore the CE bit IP: ....0 = CE bit - no congestion IP: Total length = 100 bytes IP: Identification = 18075 IP: Flags = 4X IP: .1.. .... = don't fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = C44A (correct) IP: Source address = [88.1.88.8] **IP: Destination address = [192.168.1.1]** IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP: Type = 0 (Echo reply) ICMP: Code = 0 ICMP: Checksum = 9B2D (correct) ICMP: Identifier = 3046 ICMP: Sequence number = 3245 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".] ICMP:

Réponse d'écho au client B VPN

Ici, nous voyons une réponse d'écho allant de retour à l'adresse IP 192.168.1.1 de destination dans le custB de VRF. L'adresse de destination est traduite à 172.31.1.1 par la fonction NAT de PE d'entrée.

To VRF custB: DLC: ----- DLC Header ----- DLC: DLC: Frame 12 arrived at 09:15:49.6635; frame size is 118 (0076 hex) bytes. DLC: Destination = Station 0090BF9C6C1C DLC: Source = Station 005054D92A25 DLC: Ethertype = 8847 (MPLS) DLC: MPLS: ----- MPLS Label Stack ----- MPLS: **MPLS: Label Value = 0001D** MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1 (Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP: Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0 .... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP: .... ..0. = ECT bit - transport protocol will ignore the CE bit IP: ....0 = CE bit - no congestion IP: Total length = 100 bytes IP: Identification = 37925 IP: Flags = 4X IP: .1.. .... = don't fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254 seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = 75BF (correct) IP: Source address = [88.1.88.8] **IP: Destination address = [192.168.2.2]** IP: No options IP: ICMP: ----- ICMP header ----- ICMP: ICMP: Type = 0 (Echo reply) ICMP: Code = 0 ICMP: Checksum = B39A (correct) ICMP: Identifier = 4173 ICMP: Sequence number = 4212 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP header".]

Remarque: Dans les paquets de retour, les valeurs de mpls label sont incluses et diffèrent : *001A* pour le custA de VRF et *001D* pour le custB de VRF.

Écho du client A VPN – La destination est une interface générique

Ce prochain ensemble de paquets affichent la différence quand l'interface au RÉSEAU LOCAL partagé de service est une interface générique et pas une partie d'un exemple de VRF. Ici, la configuration a été changée pour utiliser un pool commun pour des les deux les gens du pays VPN avec les adresses IP superposantes.

```
ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length 24 ip nat inside source list 181
pool SSPOOL1 vrf custA overload ip nat inside source list 181 pool SSPOOL1 vrf custB overload
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 1 arrived at  09:39:19.6580; frame size is 118 (0076 hex)
            bytes.
        DLC:  Destination = Station 005054D92A25
        DLC:  Source      = Station 0090BF9C6C1C
        DLC:  Ethertype   = 8847 (MPLS)
        DLC:
  MPLS: ----- MPLS Label Stack  -----
        MPLS:
        MPLS: Label Value = 00019 MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1
(Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP:
Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0
.... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP:
.... ..0. = ECT bit - transport protocol will ignore the CE bit IP: .... ...0 = CE bit - no
congestion IP: Total length = 100 bytes IP: Identification = 55 IP: Flags = 0X IP: .0.. .... =
may fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254
seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = 4AAF (correct) IP: Source address =
[192.168.1.1] IP: Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header -
---- ICMP: ICMP: Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = 0905 (correct) ICMP: Identifier
= 874 ICMP: Sequence number = 3727 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP
header".]
```

Écho du client B VPN – La destination est une interface générique

Ici, nous voyons une requête d'écho provenir l'adresse IP source 172.31.1.1 dans le custB de VRF. L'adresse source a été traduite à 192.168.1.3 (de pool commun SSPOOL1) comme spécifiée par la configuration NAT :

```
ip nat pool SSPOOL1 192.168.1.1 192.168.1.254 prefix-length 24 ip nat inside source list 181
pool SSPOOL1 vrf custA overload ip nat inside source list 181 pool SSPOOL1 vrf custB overload
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 11 arrived at  09:39:26.4971; frame size is 118 (0076 hex)
            bytes.
        DLC:  Destination = Station 005054D92A25
        DLC:  Source      = Station 0090BF9C6C1C
        DLC:  Ethertype   = 8847 (MPLS)
        DLC:
  MPLS: ----- MPLS Label Stack  -----
        MPLS:
        MPLS: Label Value = 0001F MPLS: Reserved For Experimental Use = 0 MPLS: Stack Value = 1
(Bottom of Stack) MPLS: Time to Live = 254 (hops) MPLS: IP: ----- IP Header ----- IP: IP:
Version = 4, header length = 20 bytes IP: Type of service = 00 IP: 000. .... = routine IP: ...0
.... = normal delay IP: .... 0... = normal throughput IP: .... .0.. = normal reliability IP:
.... ..0. = ECT bit - transport protocol will ignore the CE bit IP: .... ...0 = CE bit - no
congestion IP: Total length = 100 bytes IP: Identification = 75 IP: Flags = 0X IP: .0.. .... =
may fragment IP: ..0. .... = last fragment IP: Fragment offset = 0 bytes IP: Time to live = 254
seconds/hops IP: Protocol = 1 (ICMP) IP: Header checksum = 4A99 (correct) IP: Source address =
[192.168.1.3] IP: Destination address = [88.1.88.8] IP: No options IP: ICMP: ----- ICMP header -
---- ICMP: ICMP: Type = 8 (Echo) ICMP: Code = 0 ICMP: Checksum = 5783 (correct) ICMP: Identifier
= 4237 ICMP: Sequence number = 977 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end of "ICMP
```

```
header".]
```

Remarque: Quand l'interface au PE de sortie est une interface générique (pas un exemple de VRF), les étiquettes imposées sont différentes. Dans ce cas, *0x19* et *0x1F*.

Réponse d'écho au client A VPN – La destination est une interface générique

Ensuite, nous voyons une réponse d'écho allant de retour à l'adresse IP 192.168.1.1 de destination dans le custA de VRF. L'adresse de destination est traduite à 172.31.1.1 par la fonction NAT de PE d'entrée.

```
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 2 arrived at  09:39:19.6621; frame size is 114 (0072 hex)
            bytes.
        DLC:  Destination = Station 0090BF9C6C1C
        DLC:  Source      = Station 005054D92A25
        DLC:  Ethertype   = 0800 (IP)
        DLC:
  IP: ----- IP Header -----
        IP:
        IP: Version = 4, header length = 20 bytes
        IP: Type of service = 00
        IP:       000. ....   = routine
        IP:       ...0 .... = normal delay
        IP:       .... 0... = normal throughput
        IP:       .... .0.. = normal reliability
        IP:       .... ..0. = ECT bit - transport protocol will ignore the CE
            bit
        IP:       .... ...0 = CE bit - no congestion
        IP: Total length    = 100 bytes
        IP: Identification  = 54387
        IP: Flags           = 4X
        IP:       .1.. .... = don't fragment
        IP:       ..0. .... = last fragment
        IP: Fragment offset = 0 bytes
        IP: Time to live    = 254 seconds/hops
        IP: Protocol        = 1 (ICMP)
        IP: Header checksum = 3672 (correct)
        IP: Source address      = [88.1.88.8]
        IP: Destination address = [192.168.1.1] IP: No options IP: ICMP: ----- ICMP header -----
ICMP: ICMP: Type = 0 (Echo reply) ICMP: Code = 0 ICMP: Checksum = 1105 (correct) ICMP:
Identifier = 874 ICMP: Sequence number = 3727 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end
of "ICMP header".]
```

Réponse d'écho au client B VPN – La destination est une interface générique

Ici, nous voyons une réponse d'écho allant de retour à l'adresse IP 192.168.1.3 de destination dans le custB de VRF. L'adresse de destination est traduite à 172.31.1.1 par la fonction NAT de PE d'entrée.

```
DLC:  ----- DLC Header -----
        DLC:
        DLC:  Frame 12 arrived at  09:39:26.4978; frame size is 114 (0072 hex)
            bytes.
        DLC:  Destination = Station 0090BF9C6C1C
        DLC:  Source      = Station 005054D92A25
        DLC:  Ethertype   = 0800 (IP)
        DLC:
  IP: ----- IP Header -----
        IP:
```

```
        IP: Version = 4, header length = 20 bytes
        IP: Type of service = 00
        IP:       000. ....   = routine
        IP:       ...0 .... = normal delay
        IP:       .... 0... = normal throughput
        IP:       .... .0.. = normal reliability
        IP:       .... ..0. = ECT bit - transport protocol will ignore the CE
            bit
        IP:       .... ...0 = CE bit - no congestion
        IP: Total length    = 100 bytes
        IP: Identification  = 61227
        IP: Flags           = 4X
        IP:       .1.. .... = don't fragment
        IP:       ..0. .... = last fragment
        IP: Fragment offset = 0 bytes
        IP: Time to live    = 254 seconds/hops
        IP: Protocol        = 1 (ICMP)
        IP: Header checksum = 1BB8 (correct)
        IP: Source address      = [88.1.88.8]
        IP: Destination address = [192.168.1.3] IP: No options IP: ICMP: ----- ICMP header -----
ICMP: ICMP: Type = 0 (Echo reply) ICMP: Code = 0 ICMP: Checksum = 5F83 (correct) ICMP:
Identifier = 4237 ICMP: Sequence number = 977 ICMP: [72 bytes of data] ICMP: ICMP: [Normal end
of "ICMP header".]
```

Remarque: Puisque les réponses sont destinées à une adresse globale, aucune étiquette de VRF n'est imposée.

Avec l'interface de sortie au segment partagé de RÉSEAU LOCAL de service défini comme interface générique, on permet un pool commun. Les pings ont comme conséquence ces entrées NAT dans le routeur le **Gila** :

```
gila# show ip nat translations Pro Inside global Inside local Outside local Outside global icmp
192.168.1.3:4237 172.31.1.1:4237 88.1.88.8:4237 88.1.88.8:4237 icmp 192.168.1.3:4238
172.31.1.1:4238 88.1.88.8:4238 88.1.88.8:4238 icmp 192.168.1.3:4239 172.31.1.1:4239
88.1.88.8:4239 88.1.88.8:4239 icmp 192.168.1.3:4240 172.31.1.1:4240 88.1.88.8:4240
88.1.88.8:4240 icmp 192.168.1.3:4241 172.31.1.1:4241 88.1.88.8:4241 88.1.88.8:4241 icmp
192.168.1.1:874 172.31.1.1:874 88.1.88.8:874 88.1.88.8:874 icmp 192.168.1.1:875 172.31.1.1:875
88.1.88.8:875 88.1.88.8:875 icmp 192.168.1.1:876 172.31.1.1:876 88.1.88.8:876 88.1.88.8:876 icmp
192.168.1.1:877 172.31.1.1:877 88.1.88.8:877 88.1.88.8:877 icmp 192.168.1.1:878 172.31.1.1:878
88.1.88.8:878 88.1.88.8:878 gila# gila# show ip nat tr ver
Pro Inside global      Inside local       Outside local      Outside global
icmp 192.168.1.3:4237  172.31.1.1:4237    88.1.88.8:4237     88.1.88.8:4237
    create 00:00:08, use 00:00:08, left 00:00:51, Map-Id(In): 2,
    flags:
extended, use_count: 0, VRF : custB icmp 192.168.1.3:4238 172.31.1.1:4238 88.1.88.8:4238
88.1.88.8:4238 create 00:00:08, use 00:00:08, left 00:00:51, Map-Id(In): 2, flags: extended,
use_count: 0, VRF : custB icmp 192.168.1.3:4239 172.31.1.1:4239 88.1.88.8:4239 88.1.88.8:4239
create 00:00:08, use 00:00:08, left 00:00:51, Map-Id(In): 2, flags: extended, use_count: 0, VRF
: custB icmp 192.168.1.3:4240 172.31.1.1:4240 88.1.88.8:4240 88.1.88.8:4240 create 00:00:08, use
00:00:08, left 00:00:51, Map-Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp
192.168.1.3:4241 172.31.1.1:4241 88.1.88.8:4241 88.1.88.8:4241 create 00:00:08, use 00:00:08,
left 00:00:51, Map-Id(In): 2, flags: extended, use_count: 0, VRF : custB icmp 192.168.1.1:874
172.31.1.1:874 88.1.88.8:874 88.1.88.8:874 create 00:00:16, use 00:00:16, left 00:00:43, Map-
Id(In): 3, Pro Inside global Inside local Outside local Outside global flags: extended,
use_count: 0, VRF : custA icmp 192.168.1.1:875 172.31.1.1:875 88.1.88.8:875 88.1.88.8:875 create
00:00:18, use 00:00:18, left 00:00:41, Map-Id(In): 3, flags: extended, use_count: 0, VRF : custA
icmp 192.168.1.1:876 172.31.1.1:876 88.1.88.8:876 88.1.88.8:876 create 00:00:18, use 00:00:18,
left 00:00:41, Map-Id(In): 3, flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:877
172.31.1.1:877 88.1.88.8:877 88.1.88.8:877 create 00:00:18, use 00:00:18, left 00:00:41, Map-
Id(In): 3, flags: extended, use_count: 0, VRF : custA icmp 192.168.1.1:878 172.31.1.1:878
88.1.88.8:878 88.1.88.8:878 create 00:00:18, use 00:00:18, left 00:00:41, Map-Id(In): 3, flags:
extended, use_count: 0, VRF : custA gila# debug ip nat vrf IP NAT VRF debugging is on gila# .Jan
2 09:34:54 EST: NAT-TAGSW(p) : Tag Pkt s=172.18.60.179, d=10.88.162.9, vrf=custA .Jan 2 09:35:02
```

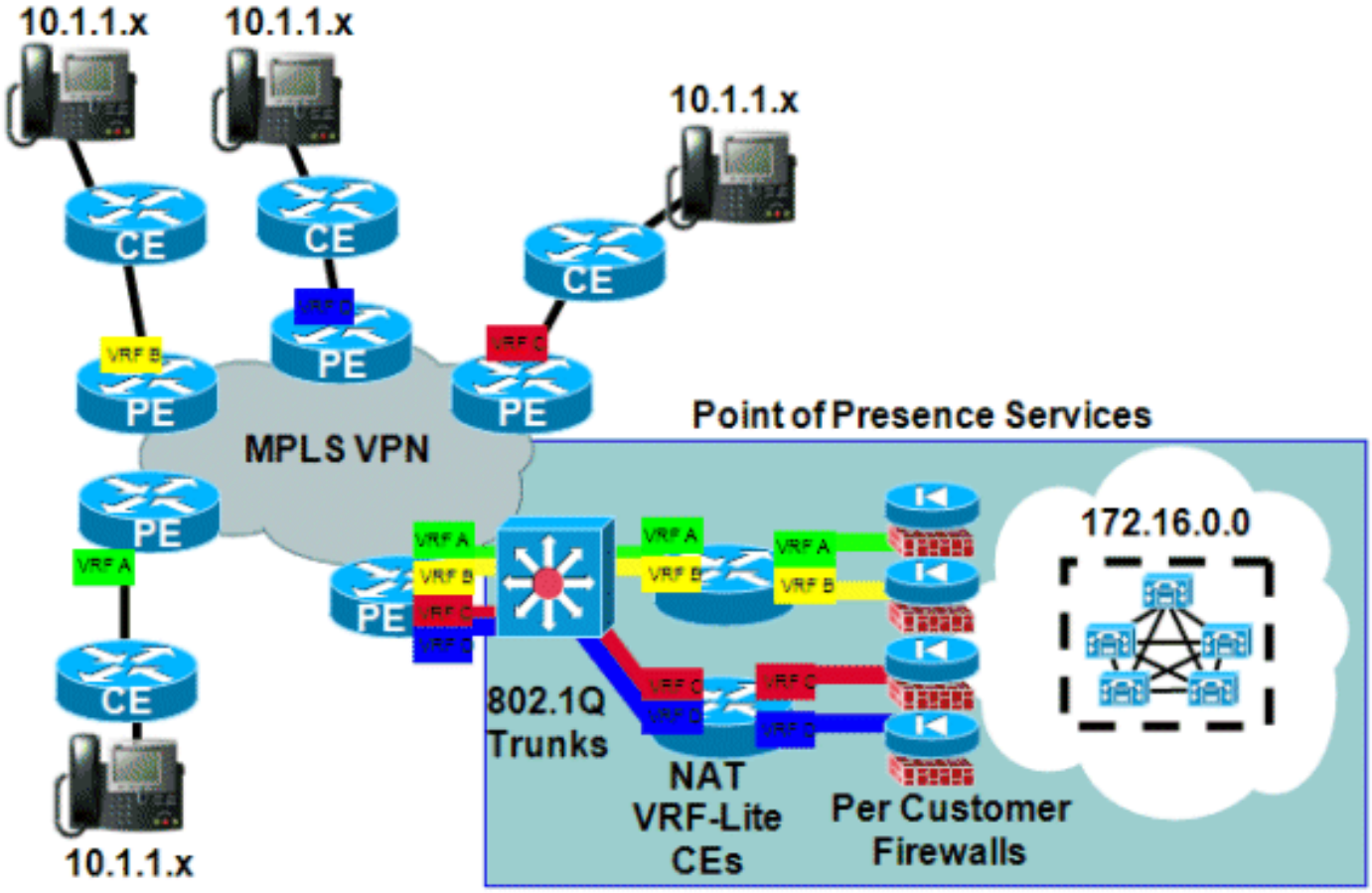
```
EST: NAT-TAGSW(p) : Tag Pkt s=172.18.60.179, d=10.88.162.13, vrf=custB .Jan 2 09:35:12 EST: NAT-
ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custA .Jan 2 09:35:12 EST: NAT-ip2tag: Punting
to process .Jan 2 09:35:12 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custA .Jan 2
09:35:12 EST: NAT-ip2tag: Punting to process .Jan 2 09:35:12 EST: NAT-ip2tag : Tag Pkt
s=172.31.1.1, d=88.1.88.8, vrf=custA .Jan 2 09:35:12 EST: NAT-ip2tag: Punting to process .Jan 2
09:35:12 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custA .Jan 2 09:35:12 EST:
NAT-ip2tag: Punting to process .Jan 2 09:35:12 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1,
d=88.1.88.8, vrf=custA .Jan 2 09:35:12 EST: NAT-ip2tag: Punting to process .Jan 2 09:35:19 EST:
NAT-ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custB .Jan 2 09:35:19 EST: NAT-ip2tag:
Punting to process .Jan 2 09:35:19 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8,
vrf=custB .Jan 2 09:35:19 EST: NAT-ip2tag: Punting to process .Jan 2 09:35:19 EST: NAT-ip2tag :
Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custB .Jan 2 09:35:19 EST: NAT-ip2tag: Punting to process
.Jan 2 09:35:19 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1, d=88.1.88.8, vrf=custB .Jan 2 09:35:19
EST: NAT-ip2tag: Punting to process .Jan 2 09:35:19 EST: NAT-ip2tag : Tag Pkt s=172.31.1.1,
d=88.1.88.8, vrf=custB .Jan 2 09:35:19 EST: NAT-ip2tag: Punting to process gila#
```

Entretenez l'exemple

Un exemple d'un service virtuel partagé IP PBX est affiché dans la figure 8. Ceci illustre une variante aux exemples d'entrée et de sortie décrits plus tôt.

Dans cette conception, le service VoIP partagé avant-est fini par un ensemble de routeurs qui remplissent la fonction NAT. Ces Routeurs ont de plusieurs interfaces de VRF utilisant une caractéristique connue sous le nom de Vrf-Lite. Le trafic circule alors à la batterie partagée de Cisco CallManager. Des services de Pare-feu sont également fournis sur une base de par-société. Les appels inter-sociétaires doivent traverser le Pare-feu, alors que des appels internes à l'entreprise sont traités à travers le client VPN utilisant le système d'adressage interne de la société.

Figure 8 : Exemple virtuel géré de service PBX



Disponibilité

Le soutien NAT de Cisco IOS de MPLS VPNs est disponible dans la Cisco IOS version 12.2(13)T et est disponible pour toutes les Plateformes qui prennent en charge le MPLS et peuvent exécuter cette série de version de déploiement anticipé (ED).

Conclusion

Le Cisco IOS NAT a des caractéristiques pour permettre le déploiement extensible des services partagés aujourd'hui. Cisco continue à développer le soutien NAT de la passerelle de niveau application (ALG) des protocoles importants pour des clients. Les améliorations des performances et l'accélération matérielle pour des fonctions de traduction s'assureront que NAT et ALGs fournissez les solutions acceptables pendant quelque temps encore. Toutes les activités de normes et actions communautaires appropriées sont surveillées par Cisco. Car d'autres normes sont développées, leur utilisation sera évaluée a basé sur des désirs, des exigences, et l'application de client.

Informations connexes

- Passerelles NAT de couche application de Cisco IOS
- Architectures MPLS et VPN
- Conception et réalisation avancée MPLS
- Support et documentation techniques - Cisco Systems