

Cisco IOS NAT - Intégration à MPLS VPN

Contenu

[Introduction](#)

[Avantages de l'intégration NAT - MPLS](#)

[Considérations de conception](#)

[Scénarios de déploiement](#)

[Options de déploiement et détails de configuration](#)

[NAT PE de sortie](#)

[NAT PE en entrée](#)

[Paquets arrivant au niveau de l'équipement d'abonné central après la NAT de l'équipement d'abonné d'entrée](#)

[Exemple de service](#)

[Disponibilité](#)

[Conclusion](#)

[Informations connexes](#)

Introduction

Le logiciel Cisco IOS[®] Network Address Translation (NAT) permet d'accéder aux services partagés à partir de plusieurs VPN MPLS, même lorsque les périphériques des VPN utilisent des adresses IP qui se chevauchent. La fonction NAT de Cisco IOS est compatible VRF et peut être configurée sur les routeurs de périphérie du fournisseur au sein du réseau MPLS.

Remarque : MPLS dans IOS est pris en charge uniquement avec NAT hérité. Actuellement, Cisco IOS ne prend pas en charge NAT NVI avec MPLS.

Le déploiement des VPN MPLS devrait augmenter rapidement au cours des prochaines années. Les avantages d'une infrastructure réseau commune qui permet une expansion rapide et des options de connectivité flexibles vont indubitablement stimuler la croissance des services qui peuvent être offerts à la communauté interréseau.

Cependant, les obstacles à la croissance demeurent. L'IPv6 et sa promesse d'un espace d'adressage IP qui dépasse les besoins de connectivité dans un avenir prévisible sont encore dans les premières phases du déploiement. Les réseaux existants utilisent généralement des systèmes d'adressage IP privés tels que définis dans [RFC 1918](#). La traduction d'adresses réseau est souvent utilisée pour interconnecter des réseaux lorsque des espaces d'adresses se chevauchent ou font double emploi.

Les fournisseurs de services et les entreprises qui disposent de services d'applications réseau qu'ils souhaitent proposer ou partager avec leurs clients et partenaires voudront réduire au minimum toute charge de connectivité imposée à l'utilisateur du service. Il est souhaitable, voire obligatoire, d'étendre l'offre à autant d'utilisateurs potentiels que nécessaire pour atteindre les objectifs ou le retour désirés. Le schéma d'adressage IP utilisé ne doit pas constituer un obstacle

excluant les utilisateurs potentiels.

En déployant Cisco IOS NAT dans l'infrastructure VPN MPLS commune, les fournisseurs de services de communication peuvent alléger une partie de la charge de connectivité des clients et accélérer leur capacité à lier davantage de services d'applications partagés à un plus grand nombre de consommateurs de ces services.

Avantages de l'intégration NAT - MPLS

L'intégration NAT avec MPLS présente des avantages tant pour les fournisseurs de services que pour leurs clients d'entreprise. Il offre aux fournisseurs de services davantage d'options pour déployer des services partagés et fournir un accès à ces services. Les offres de services supplémentaires peuvent être un facteur de différenciation par rapport à la concurrence.

Pour le fournisseur de services	Pour VPN
Plus d'offres de services	Coûts réduits
Options d'accès améliorées	Accès simplifié
Augmentation des revenus	Souplesse d'adressage

Les entreprises qui souhaitent externaliser une partie de leur charge de travail actuelle peuvent également bénéficier d'offres plus larges de la part des fournisseurs de services. Le transfert de la charge liée à la traduction des adresses nécessaires vers le réseau du fournisseur de services les soulage d'une tâche administrative compliquée. Les clients peuvent continuer à utiliser l'adressage privé tout en conservant l'accès aux services partagés et à Internet. La consolidation de la fonction NAT au sein du réseau du fournisseur de services peut également réduire le coût total pour les entreprises clientes puisque les routeurs de périphérie du client n'ont pas à exécuter la fonction NAT.

Considérations de conception

Lors de la prise en compte des conceptions qui appelleront NAT au sein du réseau MPLS, la première étape consiste à déterminer les besoins en services du point de vue des applications. Vous devez tenir compte des protocoles utilisés et de toute communication client/serveur spéciale imposée par l'application. Assurez-vous que la prise en charge nécessaire des protocoles utilisés est prise en charge et gérée par Cisco IOS NAT. Une liste des protocoles pris en charge est fournie dans le document [Passerelles de couche application NAT Cisco IOS](#).

Ensuite, il sera nécessaire de déterminer l'utilisation prévue du service partagé et le débit de trafic prévu en paquets par seconde. NAT est une fonction gourmande en CPU du routeur. Par conséquent, les exigences de performances seront un facteur dans la sélection d'une option de déploiement particulière et pour déterminer le nombre de périphériques NAT concernés.

Considérez également les problèmes de sécurité et les précautions à prendre. Bien que les VPN MPLS, par définition, soient un trafic privé et effectivement distinct, le réseau de service partagé est généralement commun à de nombreux VPN.

Scénarios de déploiement

Il existe deux options pour le déploiement NAT à la périphérie du fournisseur MPLS :

- Centralisé avec les PE NAT de sortie
- Distribué avec des PE NAT d'entrée

La configuration de la fonction NAT au point de sortie du réseau MPLS le plus proche du réseau à services partagés présente les avantages suivants :

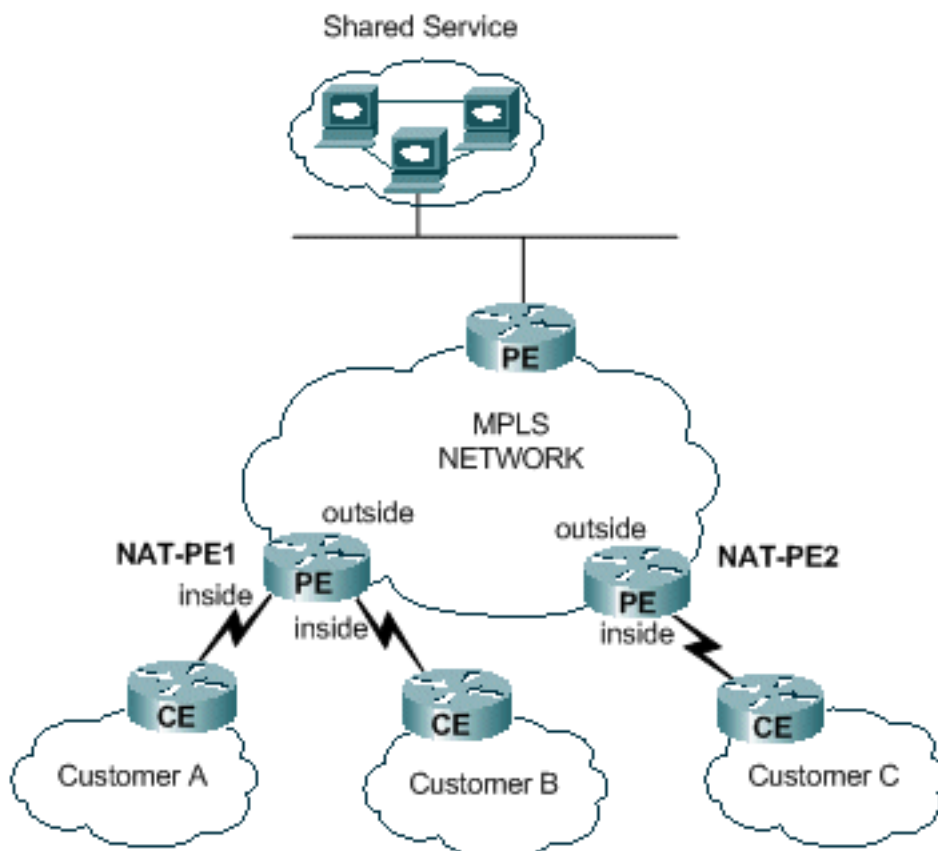
- Une configuration centralisée qui facilite le provisionnement des services
- Dépannage simplifié
- Évolutivité opérationnelle améliorée
- Diminution des besoins d'allocation d'adresses IP

Cependant, les avantages sont compensés par une réduction de l'évolutivité et des performances. Il s'agit du principal compromis à prendre en considération. Bien sûr, la fonction NAT peut également être exécutée au sein des réseaux du client s'il est déterminé que l'intégration de cette fonctionnalité à un réseau MPLS n'est pas souhaitable.

NAT PE en entrée

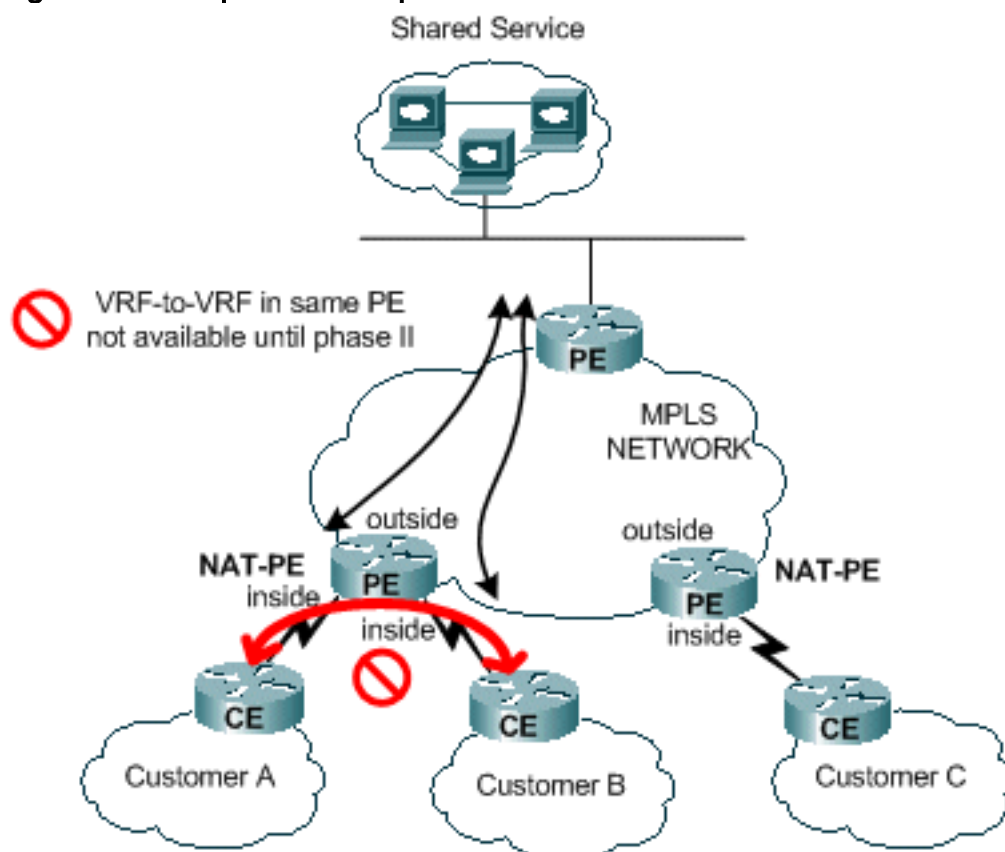
La NAT peut être configurée au niveau du routeur PE d'entrée réseau MPLS, comme illustré à la [Figure 1](#). Grâce à cette conception, l'évolutivité est maintenue dans une large mesure, tandis que les performances sont optimisées en répartissant la fonction NAT sur de nombreux périphériques. Chaque NAT PE gère le trafic des sites connectés localement à ce PE. Les règles NAT et les listes de contrôle d'accès ou les mappages de route contrôlent quels paquets nécessitent une traduction.

Figure 1 : NAT PE en entrée



Il existe une restriction qui empêche la NAT entre deux VRF tout en fournissant la NAT à un service partagé, comme illustré à la [Figure 2](#). Cela est dû à la nécessité de désigner les interfaces comme " NAT à l'intérieur des " et " à l'extérieur des interfaces ". La prise en charge des connexions entre VRF dans un seul PE est prévue pour une prochaine version de Cisco IOS.

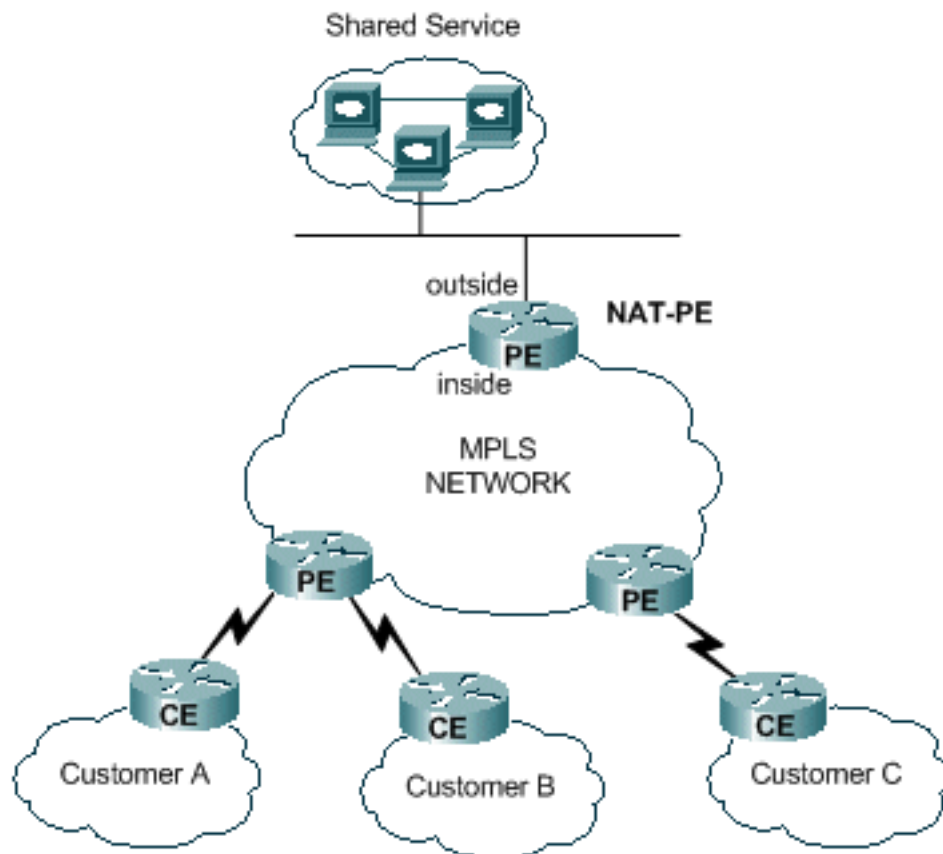
Figure 2 : Entreprise à entreprise



NAT PE de sortie

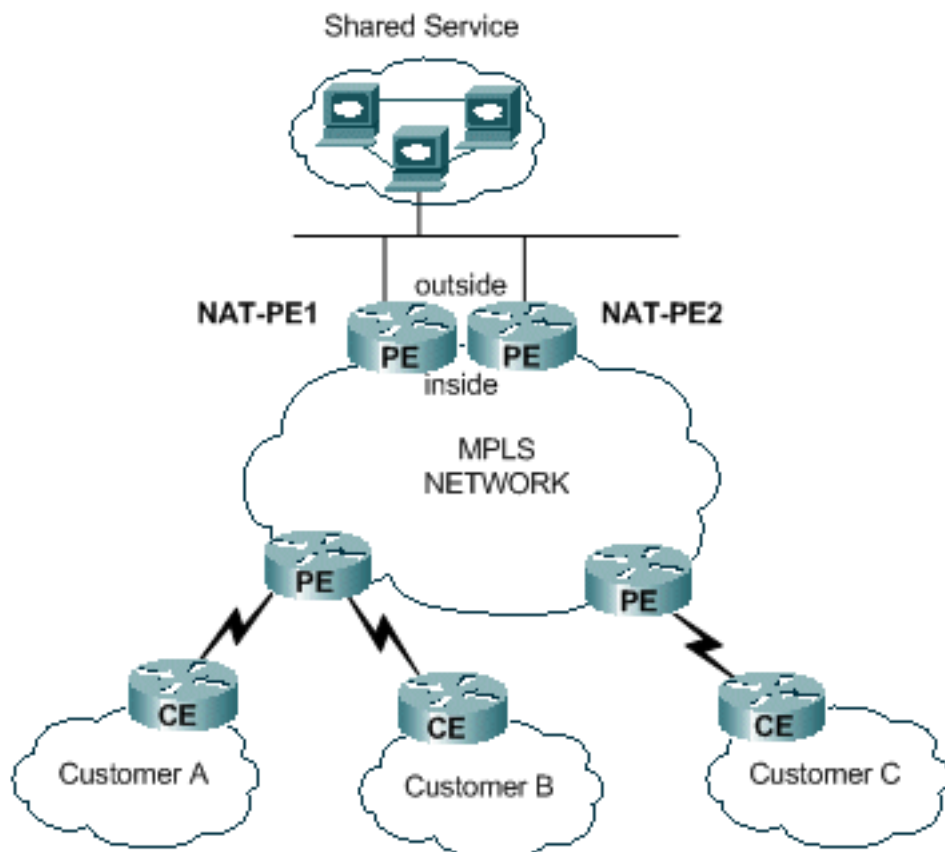
La NAT peut être configurée au niveau du routeur PE de sortie du réseau MPLS, comme illustré à la [Figure 3](#). Grâce à cette conception, l'évolutivité est réduite dans une certaine mesure, car l'équipement d'abonné central doit gérer des routes pour tous les réseaux de clients qui accèdent au service partagé. Les exigences de performances des applications doivent également être prises en compte afin que le trafic ne surcharge pas le routeur qui doit traduire les adresses IP des paquets. Comme la NAT se produit de manière centralisée pour tous les clients utilisant ce chemin, les pools d'adresses IP peuvent être partagés ; ainsi, le nombre total de sous-réseaux requis est réduit.

Figure 3 : NAT PE de sortie



Plusieurs routeurs peuvent être déployés pour accroître l'évolutivité de la conception NAT de l'équipement de périphérie de sortie, comme illustré à la [Figure 4](#). Dans ce scénario, les réseaux privés virtuels des clients peuvent être " provisionnés " sur un routeur NAT spécifique. La traduction d'adresses réseau se produirait pour le trafic agrégé en provenance et à destination du service partagé pour cet ensemble de VPN. Par exemple, le trafic en provenance des VPN des clients A et B peut utiliser NAT-PE1, tandis que le trafic en provenance et à destination du VPN du client C utilise NAT-PE2. Chaque NAT PE transporte le trafic uniquement pour les VPN spécifiques définis et ne conserve que les routes vers les sites de ces VPN. Des pools d'adresses NAT distincts peuvent être définis au sein de chacun des routeurs PE NAT de sorte que les paquets soient routés du réseau de service partagé vers le PE NAT approprié pour traduction et routage vers le VPN client.

Figure 4 : NAT PE de sortie multiple



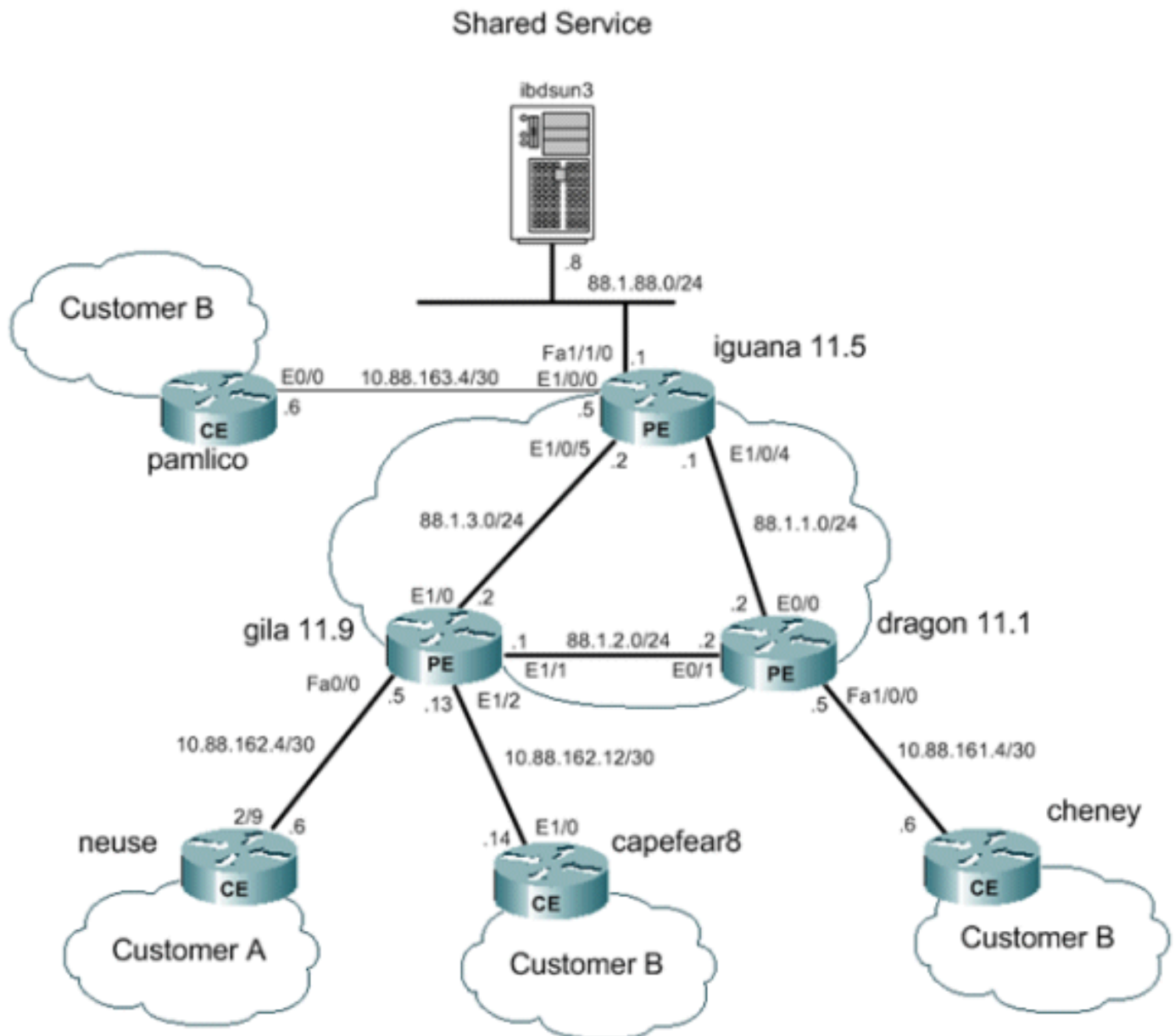
La conception centralisée impose une restriction sur la configuration du réseau de services partagés. Plus précisément, l'importation/exportation de routes VPN MPLS entre un VPN de service partagé et des VPN client n'est pas possible. Ceci est dû à la nature du fonctionnement MPLS tel que spécifié par [RFC 2547](#). Lorsque des routes sont importées et exportées à l'aide des communautés étendues et des descripteurs de route, la NAT ne peut pas déterminer le VPN source du paquet entrant dans le port d'accès NAT central. Le cas habituel est de faire du réseau de service partagé une interface générique plutôt qu'une interface VRF. Une route vers le réseau de service partagé est ensuite ajoutée dans l'équipement de pontage NAT central pour chaque table VRF associée à un VPN client nécessitant un accès au service partagé dans le cadre du processus de provisionnement. Ceci est décrit plus en détail plus loin.

[Options de déploiement et détails de configuration](#)

Cette section contient des détails relatifs à chacune des options de déploiement. Tous les exemples proviennent du réseau représenté à la [Figure 5](#). Reportez-vous à ce diagramme pour le reste de cette section.

Remarque : dans le réseau utilisé pour illustrer le fonctionnement de la NAT VRF pour ce document, seuls les routeurs PE sont inclus. Il n'existe aucun routeur " P " principal. Cependant, les mécanismes essentiels restent à voir.

Figure 5 : Exemple de configuration de la NAT VRF



[NAT PE de sortie](#)

Dans cet exemple, les routeurs de périphérie du fournisseur marqués **gila** et **dragon** sont configurés comme des routeurs PE simples. L'équipement d'abonné central près du réseau local de service partagé (**iguana**) est configuré pour NAT. Un pool NAT unique est partagé par chaque VPN client qui a besoin d'un accès au service partagé. La NAT est exécutée uniquement sur les paquets destinés à l'hôte de service partagé à l'adresse 88.1.88.8.

[Transfert de données NAT PE de sortie](#)

Avec MPLS, chaque paquet entre dans le réseau au niveau d'un PE d'entrée et quitte le réseau MPLS au niveau d'un PE de sortie. Le chemin des routeurs de commutation d'étiquette traversés d'entrée en sortie est appelé chemin commuté d'étiquette (LSP). Le LSP est unidirectionnel. Un autre LSP est utilisé pour le trafic de retour.

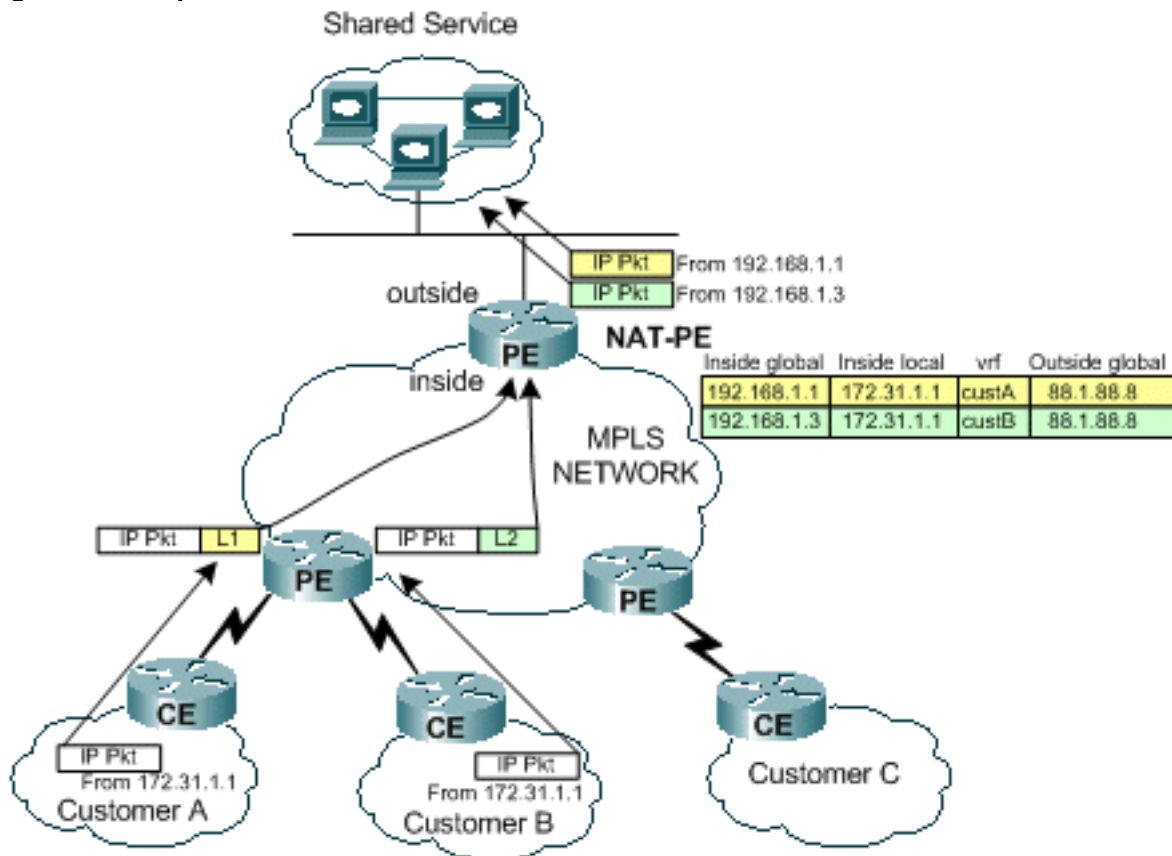
Lors de l'utilisation de la fonction NAT PE de sortie, une classe d'équivalence de transfert (FEC) est effectivement définie pour tout le trafic provenant des utilisateurs du service partagé. En d'autres termes, tous les paquets destinés au réseau local de service partagé sont membres d'une FEC commune. Un paquet est attribué à un FEC particulier une seule fois à la périphérie d'entrée

du réseau et suit le LSP vers le PE de sortie. La FEC est désignée dans le paquet de données en ajoutant une étiquette particulière.

Flux de paquets vers le service partagé à partir du VPN

Pour que les périphériques de plusieurs VPN qui ont des schémas d'adresses qui se chevauchent puissent accéder à un hôte de service partagé, NAT est requis. Lorsque la NAT est configurée au niveau du PE de sortie, les entrées de la table de traduction d'adresses réseau incluent un identificateur VRF pour différencier les adresses en double et assurer un routage approprié.

Figure 6 : Paquets transmis à la NAT PE de sortie



La Figure 6 illustre les paquets destinés à un hôte de service partagé provenant de deux réseaux privés virtuels clients qui ont des schémas d'adressage IP en double. La figure montre un paquet provenant du client A avec l'adresse source 172.31.1.1 destinée à un serveur partagé à l'adresse 88.1.88.8. Un autre paquet du client B avec la même adresse IP source est également envoyé au même serveur partagé. Lorsque les paquets atteignent le routeur PE, une recherche de couche 3 est effectuée pour le réseau IP de destination dans la base d'informations de transfert (FIB).

L'entrée FIB indique au routeur PE de transférer le trafic vers le PE de sortie à l'aide d'une pile d'étiquettes. L'étiquette inférieure de la pile est attribuée par le routeur PE de destination, dans ce cas le routeur **iguana**.

```
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#

```

L'affichage montre que les paquets de la fonction VRF CustA ont une valeur de balise de 24 (0x18) et les paquets de la fonction VRF CustB ont une valeur de balise de 28 (0x1C).

Dans ce cas, comme il n'y a aucun routeur "P" dans notre réseau, aucune balise supplémentaire n'est imposée. S'il y avait eu des routeurs principaux, une étiquette externe aurait été imposée et le processus normal d'échange d'étiquettes aurait eu lieu dans le réseau principal jusqu'à ce que le paquet atteigne le PE de sortie.

Puisque le routeur **gila** est directement connecté au périphérique PE de sortie, nous voyons que la balise est poppée avant d'être ajoutée :

```

gila#
show tag-switching forwarding-table

```

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing interface	Next Hop
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 tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing interface	Next Hop
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 affichages suivants décrivent les paquets d'écho reçus par le routeur NAT PE de sortie (à l'interface E1/0/5 sur **iguana**).

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 requêtes ping entraînent la création des entrées suivantes dans la table NAT du routeur PE de sortie **iguana**. Les entrées spécifiques créées pour les paquets indiqués ci-dessus peuvent être associées par leur identificateur 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#

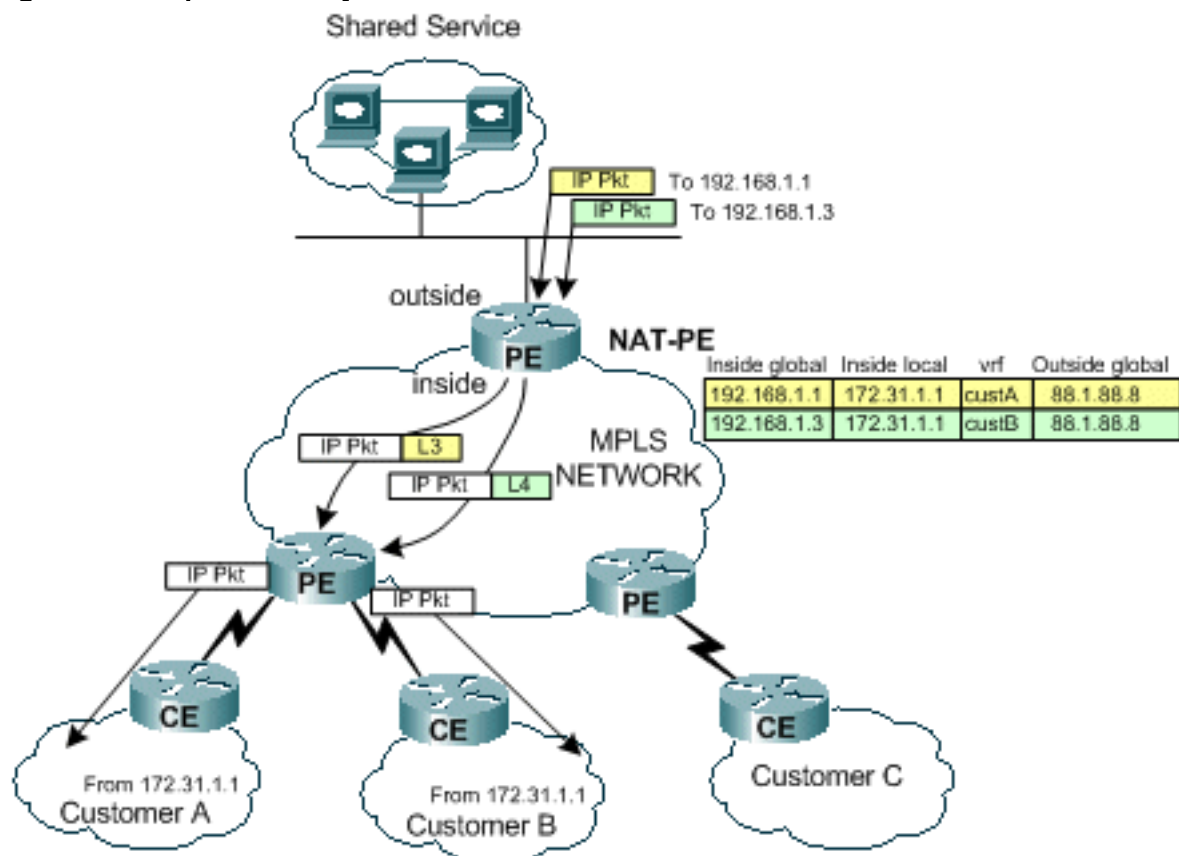
```

Flux de paquets du service partagé vers le VPN d'origine

Au fur et à mesure que les paquets reviennent aux périphériques qui ont accédé à l'hôte de service partagé, la table NAT est examinée avant le routage (les paquets allant de la NAT " de

l'interface " externe à " l'interface " interne). Comme chaque entrée unique inclut l'identificateur VRF correspondant, le paquet peut être traduit et routé de manière appropriée.

Figure 7 : Paquets renvoyés à l'utilisateur Shared Service



Comme le montre la [Figure 7](#), le trafic de retour est d'abord examiné par NAT pour trouver une entrée de traduction correspondante. Par exemple, un paquet est envoyé à la destination 192.168.1.1. La table NAT est recherchée. Lorsque la correspondance est trouvée, la traduction appropriée est effectuée vers l'adresse " locale interne " (172.31.1.1), puis une recherche de contiguïté est effectuée à l'aide de l'ID 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 à 172.31.1.0/24 dans VRF custA et l'étiquette 26 (0x1A) est utilisée pour les paquets destinés à 172.31.1.0/24 dans VRF custB.

Ceci est visible dans les paquets de réponse d'écho envoyés par le routeur **iguana** :

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".]
```

Lorsque le paquet atteint le routeur PE de destination, l'étiquette est utilisée pour déterminer le

VRF et l'interface appropriés pour envoyer le paquet.

gila#

show mpls forwarding-table

Local tag	Outgoing tag or VC	Prefix or Tunnel Id	Bytes tag switched	Outgoing interface	Next Hop
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

Certaines informations superflues ont été supprimées des configurations pour plus 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 vpv4
 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 routeur **dragon** aurait une configuration très similaire à **gila**.

Importation/exportation des cibles de routage non autorisée

Lorsque le réseau de service partagé est configuré en tant qu'instance VRF elle-même, la NAT centrale au niveau du PE de sortie n'est pas possible. Ceci est dû au fait que les paquets entrants ne peuvent pas être distingués et qu'une seule route vers le sous-réseau d'origine est présente au niveau de la NAT PE de sortie.

Remarque : Les affichages ci-dessous sont destinés à illustrer le résultat d'une configuration non valide.

L'exemple de réseau a été configuré de sorte que le réseau de service partagé ait été défini comme une instance VRF (nom VRF = serveur). Maintenant, un affichage de la table CEF sur le PE d'entrée montre 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 : Notez comment la valeur de balise 24 est imposée pour le client VRF CustA et le client VRF CustB.

Cet affichage montre la table de routage pour l'instance VRF de service partagé " " serveur :

```
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 : une seule route est présente pour le réseau de destination du point de vue du routeur PE de sortie (*iguana*).

Par conséquent, le trafic provenant de plusieurs VPN clients n'a pas pu être distingué et le trafic de retour n'a pas pu atteindre le VPN approprié. **Dans le cas où le service partagé doit être défini comme une instance VRF, la fonction NAT doit être déplacée vers le PE d'entrée.**

NAT PE en entrée

Dans cet exemple, les routeurs de périphérie du fournisseur marqués *gila* et *dragon* sont configurés pour NAT. Un pool NAT est défini pour chaque VPN client connecté qui a besoin d'un accès au service partagé. Le pool approprié est utilisé pour chacune des adresses réseau du client qui sont de type NAT. La NAT est exécutée uniquement sur les paquets destinés à l'hôte de service partagé à l'adresse 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, les pools partagés ne sont pas pris en charge. Si le réseau local de service partagé (au niveau du PE de sortie) est connecté via une interface générique, le pool NAT peut être partagé.

Une requête ping provenant d'une adresse dupliquée (172.31.1.1) dans chacun des réseaux connectés à *neuse* et *capefear8* génère les entrées NAT suivantes :

De *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 locale interne (172.31.1.1) est traduite dans chacun des pools définis en fonction du VRF source. Le VRF est visible dans la commande **show ip nat translation verbose** :

```

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 indiquent les informations de routage pour chacun des VPN locaux 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 route pour chacun des pools NAT a été ajoutée à partir de la configuration statique. Ces sous-réseaux sont ensuite importés dans le VRF du serveur partagé au niveau du routeur PE de sortie **iguana** :

```
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

Certaines informations superflues ont été supprimées des configurations pour plus 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 vpv4
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 des clients sont désignées comme “ NAT à l'intérieur des interfaces ” et les interfaces MPLS sont désignées comme “ NAT à l'extérieur des interfaces ”.

```

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 routeur **dragon** aurait une configuration très similaire à **gila**.

[Paquets arrivant au niveau de l'équipement d'abonné central après la NAT de l'équipement d'abonné d'entrée](#)

Les traces ci-dessous illustrent la nécessité de pools NAT uniques lorsque le réseau de service

partagé de destination est configuré en tant qu'instance VRF. Reportez-vous au diagramme de la [figure 5](#). Les paquets indiqués ci-dessous ont été capturés lors de leur entrée dans l'interface IP MPLS e1/0/5 au niveau du routeur **iguana**.

Écho du VPN du client A

Ici, nous voyons une requête d'écho provenant de l'adresse IP source 172.31.1.1 dans le client VRFa. L'adresse source a été traduite en 192.168.1.1 comme spécifié 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 VPN du client B

Ici, nous voyons une requête d'écho provenant de l'adresse IP source 172.31.1.1 dans le clientB VRF. L'adresse source a été traduite en 192.168.2.1 comme spécifié 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 l'étiquette MPLS est *0019* dans les deux paquets indiqués ci-dessus.

Réponse d'écho au VPN du client A

Ensuite, une réponse d'écho revenant à l'adresse IP de destination 192.168.1.1 dans le client VRFa s'affiche. L'adresse de destination est traduite en 172.31.1.1 par la fonction NAT 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 VPN du client B

Ici, nous voyons une réponse d'écho revenant à l'adresse IP de destination 192.168.1.1 dans le clientB VRF. L'adresse de destination est traduite en 172.31.1.1 par la fonction NAT 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 d'étiquette MPLS sont incluses et diffèrent :

001A pour la gamme VRF CustA et 001D pour la gamme VRF CustB.

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

Cet ensemble de paquets suivant montre la différence lorsque l'interface vers le réseau local de service partagé est une interface générique et ne fait pas partie d'une instance VRF. Ici, la configuration a été modifiée pour utiliser un pool commun pour les deux VPN locaux avec des adresses IP qui se chevauchent.

```
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 VPN du client B - La destination est une interface générique

Ici, nous voyons une requête d'écho provenant de l'adresse IP source 172.31.1.1 dans le clientB VRF. L'adresse source a été traduite en 192.168.1.3 (à partir du pool commun SSPOOL1) comme spécifié 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 : lorsque l'interface du PE de sortie est une interface générique (et non une instance 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, une réponse d'écho revenant à l'adresse IP de destination 192.168.1.1 dans le client VRFa s'affiche. L'adresse de destination est traduite en 172.31.1.1 par la fonction NAT 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 VPN du client B - La destination est une interface générique

Ici, nous voyons une réponse d'écho revenant à l'adresse IP de destination 192.168.1.3 dans le clientB VRF. L'adresse de destination est traduite en 172.31.1.1 par la fonction NAT 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 VRF n'est imposée.

Avec l'interface de sortie vers le segment LAN de service partagé défini comme interface générique, un pool commun est autorisé. Les requêtes ping aboutissent aux entrées NAT suivantes dans le routeur **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
.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#

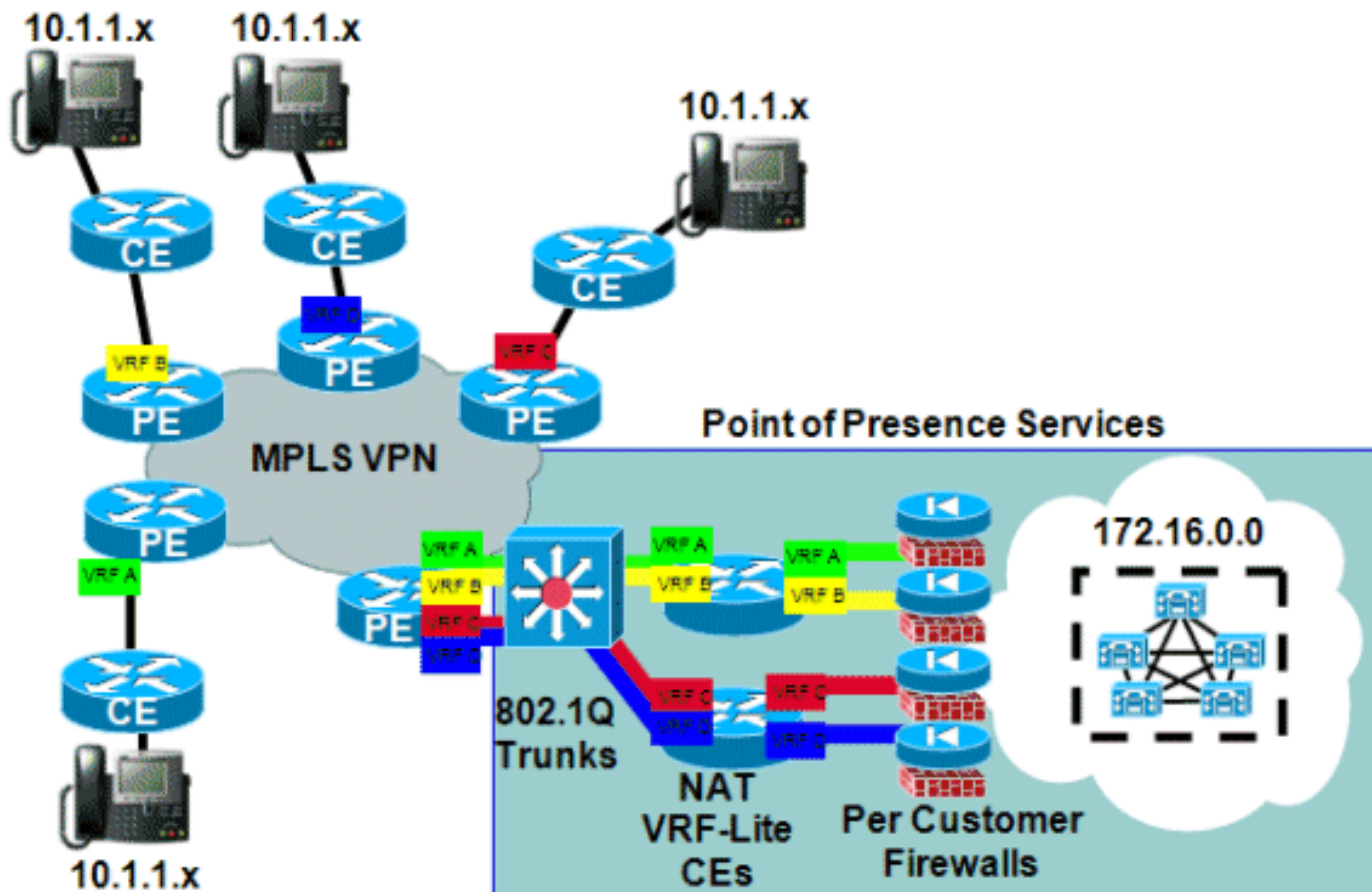
```

Exemple de service

La [Figure 8](#) illustre un exemple de service PBX IP virtuel partagé. Ceci illustre une variante des exemples d'entrée et de sortie décrits précédemment.

Dans cette conception, le service VoIP partagé est frontal par un ensemble de routeurs qui exécutent la fonction NAT. Ces routeurs disposent de plusieurs interfaces VRF utilisant une fonction appelée VRF-Lite. Le trafic est ensuite acheminé vers le cluster Cisco CallManager partagé. Les services de pare-feu sont également fournis par entreprise. Les appels interentreprises doivent passer par le pare-feu, tandis que les appels intra-entreprise sont traités sur le VPN du client à l'aide du schéma d'adressage interne de l'entreprise.

Figure 8 : Exemple de service PBX virtuel géré



Disponibilité

La prise en charge NAT de Cisco IOS pour les VPN MPLS est disponible dans la version 12.2(13)T de Cisco IOS et est disponible pour toutes les plates-formes qui prennent en charge MPLS et peuvent exécuter cette version de déploiement précoce.

Conclusion

La fonction NAT de Cisco IOS offre aujourd'hui des fonctionnalités permettant un déploiement évolutif des services partagés. Cisco continue à développer la prise en charge de la passerelle de niveau application (ALG) NAT pour les protocoles importants pour les clients. L'amélioration des performances et l'accélération matérielle des fonctions de traduction garantissent que NAT et ALG fournissent des solutions acceptables pour un certain temps. Toutes les activités de normalisation pertinentes et les actions communautaires sont surveillées par Cisco. À mesure que d'autres normes sont élaborées, leur utilisation sera évaluée en fonction des souhaits, des exigences et des applications des clients.

Informations connexes

- [Passerelles de couche application NAT Cisco IOS](#)
- [Architectures MPLS et VPN](#)
- [Conception et mise en oeuvre MPLS avancées](#)
- [Support et documentation techniques - Cisco Systems](#)