Andrianandrasana-Dina Vagnona
Nathanaël Derousseaux

Université de Strasbourg

Rapport de projet VPN

I - Configuration des CE

A - Adresses de loopback

On commence par ajouter leurs adresses de loopback aux CE. Afin de mettre en évidence l'isolation des deux VPN entre eux, on va utiliser les mêmes adresses pour le VPNX et le VPNY.
Voiçi les adresses de loopback utilisées :

- **CE1:** 192.168.4.1/32
- **CE2:** 192.168.4.1/32
- **CE3:** 192.168.4.2/32
- **CE4:** 192.168.4.2/32
- **CE6A:** 192.168.4.3/32
- **CE6B:** 192.168.4.3/32

Ainsi, on s'assure qu'il n'y a pas deux adresses identiques au sein d'un même vpn, mais qu'il y a des adresses identiques entre les deux VPN.

B - Liaison routeur-hôte

La prochaine étape est de configurer la liaison routeur-hôte. De la même manière que pour les adresses de loopback, on va faire en sorte que les adresses de la liaison ne se superposent pas au sein d'un même VPN, mais qu'il y ai des adresses identiques entre les deux VPNs. On montrera ainsi la capacité d'isolation du réseau des VPNs.
Voici les adresses utilisées :

- **Router CE1:** 172.16.4.2/24
- **Hôte CE1:** 172.16.4.1/24
- **Router CE2:** 172.16.4.2/24
- **Hôte CE2:** 172.16.4.1/24
- **Router CE3:** 172.17.4.2/24
- **Hôte CE3:** 172.17.4.1/24
- **Router CE4:** 172.17.4.2/24
- **Hôte CE4:** 172.17.4.1/24
- **Router CE6A:** 172.18.4.2/24
- **Hôte CE6A:** 172.18.4.1/24
- **Router CE6B:** 172.18.4.2/24
- **Hôte CE6B:** 172.18.4.1/24

Rappel des commandes à effectuer sur l'hôte pour ajouter l'adresse de la liaison vers le router :

```
CEX_host:~# ip address add <LINK-ADDRESS> dev CEXrouter
```

Au niveau de l'hôte on ajoute ensuite un route statique par défaut pour lui permettre de joindre toutes les adresses inconnues.

Voici par exemple la commande entrée pour CE

```
CE1_host:~# ip route add default via 172.16.4.2
```

La commande :
`CEX_host:~# ip route add default via <gateway_address>`

II - Configuration du VPNX

A - Activation de VPN

On crée une session iBGP entre tous les PE d'un même VPN afin qu'ils puissent se partager des adresses de type ipv4 VPN.

```
PEX_router# conf t
PEX_router(config)# router bgp 4
PEX_router(config-router)# address-family ipv4 vpn
PEX_router(config-router-af)# neighbor <Loopback du voisin VPN>
activate

On a créé par la suite le VRF, correspondant à notre vpn. (ici
nous nous concentrons sur le vpn X puisque les commandes sont
similaire pour le vpn Y)

./goto PE1 container
root@PE1_router ~> ip link add VRF_X type vrf table 10
root@PE1_router ~> ip link set VRF_X up
root@PE1_router ~> ip link set port_CE1 master VRF_X

Ces commandes créées indiquent sur quelle interface se trouve le
client du PE afin de savoir d'où est ce que la VRF va apprendre va
apprendre le sous réseau privé du client.

On peut constater le changement ici :
Interface       Status  VRF           Addresses
---------       ------  ---           ---------
VRF_X           up    VRF_X
port_CE1        up    VRF_X           4.0.13.2/24

Interface       Status  VRF           Addresses
---------       ------  ---           ---------
host            up    default         4.106.0.2/24
lo              up    default         4.156.0.1/32
matrix_4        up    default         4.0.198.1/24
port_P1         up    default         4.0.10.2/24
port_PE2        up    default         4.0.8.2/24
```

```
ssh             up    default         158.4.15.1/16
```

Où l'on voit bien la création d'une deuxième catégorie d'interface pour VRF_X.

B - Liaison CE-PE

Il est maintenant temps de configurer la liaison entre les CE et leur PE correspondant. Pour le VPNX, nous allons utiliser 3 méthodes distinctes :

CE1 et PE1 communiqueront à travers **une route statique**,
CE6A et PE6 communiqueront à l'aide de **OSPF**,
et pour finir, CE3 et PE3 utiliseront **RIP**.

Route statique sur CE1<->PE1 :
On définit une route statique sur CE1 afin qu'il puisse joindre toutes les adresses qu'il ne connaît pas via les le routeur PE1 :

- `ip route 172.16.0.0/12 4.0.13.2`

Puis on met en place une route statique sur le routeur PE1 pour que l'extérieur sache que le préfixe 172.16.4.0/24 se situe ici

- `ip route 172.16.4.0/24 4.0.13.1 vrf VRF_X`

On peut visualiser les routes apprises à l'aide de la commande suivante :

```
PE1_router# sh ip route vrf VRF_X
     [...]

VRF VRF_X:
C>* 4.0.13.0/24 is directly connected, port_CE1, 1d04h41m
S>* 172.16.4.0/24 [1/0] via 4.0.13.1, port_CE1, weight 1, 1d04h40m
```

Où l'on voit bien grâce au "S>" que la préfixe privée du CE à été appris grâce au route statique mises en place précédemment.

Configuration OSPF CE6A <-> PE6
Sur PE6 on indique le sous-réseau 4.0.17.0/24 dans la session ospf du VRF_X puis on indique une redistribution dans la session bgp afin que les routes apprises via ospf ici soient partagées à la session BGP afin que le sous réseau privé de CE6A puisse être indiqué dans la VRF et distribué dans tous le VPN

```
router ospf vrf VRF_X
 redistribute bgp
 network 4.0.17.0/24 area 0
```

Quant à CE6A : on indique dans la session ospf 4.0.17.0/24 pour le lien avec PE6, 172.18.4.0/24 pour permettre la communication avec le sous-réseau privé correspondant et l'addresse de loopback 192.168.4.3/32.

```
router ospf
 network 4.0.17.0/24 area 0
 network 172.18.4.0/24 area 0
 network 192.168.4.3/32 area 0
```

Regardons l'état des routes :
PE6_router# sh ip route vrf VRF_X
[...]

VRF VRF_X:
O 4.0.17.0/24 [110/10] is directly connected, port_CE6A, weight 1, 03:51:49
C>* 4.0.17.0/24 is directly connected, port_CE6A, 03:51:49
B> 172.16.4.0/24 [200/0] via 4.156.0.1 (vrf default) (recursive), label 16, weight 1, 03:41:55
 * via 4.0.11.1, port_PE5 (vrf default), label 37/16, weight 1, 03:41:55
B> 172.17.4.0/24 [200/2] via 4.155.0.1 (vrf default) (recursive), label 16, weight 1, 03:41:55
 * via 4.0.11.1, port_PE5 (vrf default), label 36/16, weight 1, 03:41:55
O>* 172.18.4.0/24 [110/20] via 4.0.17.1, port_CE6A, weight 1, 03:51:38

On remarque 172.16.4.0/24 et 172.17.4.0/24 (que l'on verra juste après) on été apprise via bgp et 172.18.4.0/24 via OSPF.

Configuration RIP CE3 <-> PE3

Pour le protocole rip la logique est la même, il faut juste adapter les commandes au protocole.
Pour PE3 :

```
router rip vrf VRF_X
 network 4.0.15.0/24
 redistribute bgp
```

et CE3 :

```
router rip
 network 4.0.15.0/24
 network 172.17.4.0/24
 network 192.168.4.2/32
```

On peut visualiser les routes de la même façon que les 2 exemples précédents.

*Pour le VPNY, on n'utilisera que **des routes statiques** pour plus de simplicité.*

C - Configuration de distribution des routes

On cherche à configurer le VPN X sous une topologie en full mesh.

C'est pourquoi on utilise la même route target pour pour les 3 routeurs soir *10:100.*

```
rt vpn both 10:100
```

Comment les PE vont-ils apprendre les préfixes privées des CE?
Chaque PE connaît le préfixe privé du CE avec lequel il est directement connecté. Il va stocker le préfixe privé dans sa VRF et va par la suite la distribuer à tous les autres PE.
Par exemple l'activation de la distribution de route se voit dans la suite de commande :

(Par exemple pour des routes statique)

```
router bgp 4 vrf VRF_Y
 address-family ipv4 unicast
 redistribute static (ospf ou rip)
```

Une fois BGP et MPLS de mis en place on remarque que VRF_X contient tous les préfixes privés de CE1, CE3 et CE6A.

```
PE1_router# sh ip route vrf V
VFR_X  VRF_X
PE1_router# sh ip route vrf VRF_X

[...]

VRF VRF_X:
C>* 4.0.13.0/24 is directly connected, port_CE1, 1d01h06m
S>* 172.16.4.0/24 [1/0] via 4.0.13.1, port_CE1, weight 1, 1d01h05m
B>  172.17.4.0/24 [200/2] via 4.155.0.1 (vrf default) (recursive),
label 16, weight 1, 1d01h04m
  *                       via 4.0.8.1, port_PE2 (vrf default),
label 22/16, weight 1, 1d01h04m
B>  172.18.4.0/24 [200/20] via 4.158.0.1 (vrf default)
(recursive), label 80, weight 1, 00:04:15
  *                       via 4.0.10.1, port_P1 (vrf default),
label 41/80, weight 1, 00:04:15
B>  192.168.4.2/32 [200/2] via 4.155.0.1 (vrf default)
(recursive), label 16, weight 1, 1d01h04m
  *                       via 4.0.8.1, port_PE2 (vrf default),
label 22/16, weight 1, 1d01h04m
B>  192.168.4.3/32 [200/10] via 4.158.0.1 (vrf default)
(recursive), label 80, weight 1, 00:04:15
  *                       via 4.0.10.1, port_P1 (vrf default),
label 41/80, weight 1, 00:04:15
```

En effet, BGP sert principalement à la redistribution des routes pour que chaque PE ait la même VRF et MPLS sert à la commutation de paquet avec des label à 2 niveaux. Parce qu'en effet, pour un vpn, on utilise 2 labels pour communiquer un paquet dans un tunnel : 1 pour identifier le vpn de manière unique et l'autre pour la commutation de paquet.
Pour une distribution automatique des label MPLS on utilise la commande :

```
     label vpn export auto
```

Essayons de **traceroute** un paquet de l'hôte CE1 vers l'hôte CE3 :

```
CE3_host:~# traceroute 172.16.4.1
traceroute to 172.16.4.1 (172.16.4.1), 30 hops max, 46 byte
packets
 1  172.17.4.2 (172.17.4.2)  0.016 ms  0.017 ms  0.006 ms
 2  4.0.15.2 (4.0.15.2)  1.292 ms  0.197 ms  0.207 ms
 3  *  *  *
 4  4.0.13.1 (4.0.13.1)  2.986 ms  1.437 ms  1.317 ms
 5  172.16.4.1 (172.16.4.1)  1.513 ms  1.457 ms  1.082 ms
```

On remarque l'on n'a pas pas connaissance de ce que se passe entre les 2 router CE et c'est normal en raison de MPLS.

III - Mise en place du VPN Y

Ici nous cherchons à mettre en place une topologie hub and spoke qui consiste à mettre en place un hub qui va recevoir et accepter les annonces de route et tous ses spokes qui quant à eux ne pourront pas communiquer entre eux, ils ne pourront communiquer que via le hub.

A - configuration faites

La configuration est la même que celle du VPN en full mesh à quelques détails près.
Tout d'abord, les spoke n'ont qu'un seul voisin dans leur session iBGP puisqu'ils ne peuvent communiquer qu'avec le hub.
Puis la configuration des routes target est complètement différente étant donnée que les routeur bgp spoke ne sont sensé n'accepter que les annonces de routes venant de sont hub donc il importera les routes target 20:100 et exportera 20:200

B - Redistribution de route de la part du hub.

Dans la configuration que l'on propose les spoke partagent leur route au hub mais le hub ne partage pas ses routes à ses spokes ce qui fait que les VRF de chacun des spokes n'ont pas la même VRF et par conséquent ils ne peuvent pas communiquer entre eux.
Il y a deux méthodes pour que le hub partage ses routes à ses spokes.

./goto.sh PE5 router

Hello, this is FRRouting (version 7.5.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.
Vue depuis PE5 :

PE5_router# sh ip bgp vrf VRF_Y

[...]

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 172.16.4.0/24	4.152.0.1@0<	0	100	0	?
*> 172.17.4.0/24	4.157.0.1@0<	0	100	0	?
*> 172.18.4.0/24	4.158.0.1@0<	0	100	0	?

Displayed 3 routes and 3 total paths

Vue depuis PE4 :
PE4_router# sh ip bgp vrf VRF_Y
[...]

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 172.17.4.0/24	4.0.16.1	0		32768	?

Displayed 1 routes and 1 total paths

1- Utiliser le CE du hub

En effet, une méthode courante est de faire en sorte que le hub communique ses routes à son CE, le CE recommande ses routes au hub qui partage alors sa VRF aux spoke. Cependant, ici n'ayant pas de CE connecter à PE5 nous avons essayé cette méthode en vain.
Faire en sorte que le CE soit le PE lui-même aurait été une solution.

2- Route reflector

On peut changer le routeur PE5 en un réflecteur de route pour que les spokes partagent leur routes au hub et le hub partage ses routes aux spokes.

Dans la configuration qu'il y a en annexe nous avons fait des tests entre les routeurs CE2 et CE4 et dans cette configuration, il faut que les PE puissent continuer à communiquer entre eux donc la session iBGP en full est toujours n'est pas enlever.
Cependant la configuration ici n'est pas fonctionnelle puisque le ping d'un hôte à l'autre n'est pas fonctionnel.