2021-2022

R302
Réseaux Opérateurs

Ismail Bennis
Ismail.bennis@uha.fr
MCF, IUT de Colmar, Département Réseaux & Télécoms, bureau N°006
34 rue du Grillenbreit - 68000 Colmar Cedex

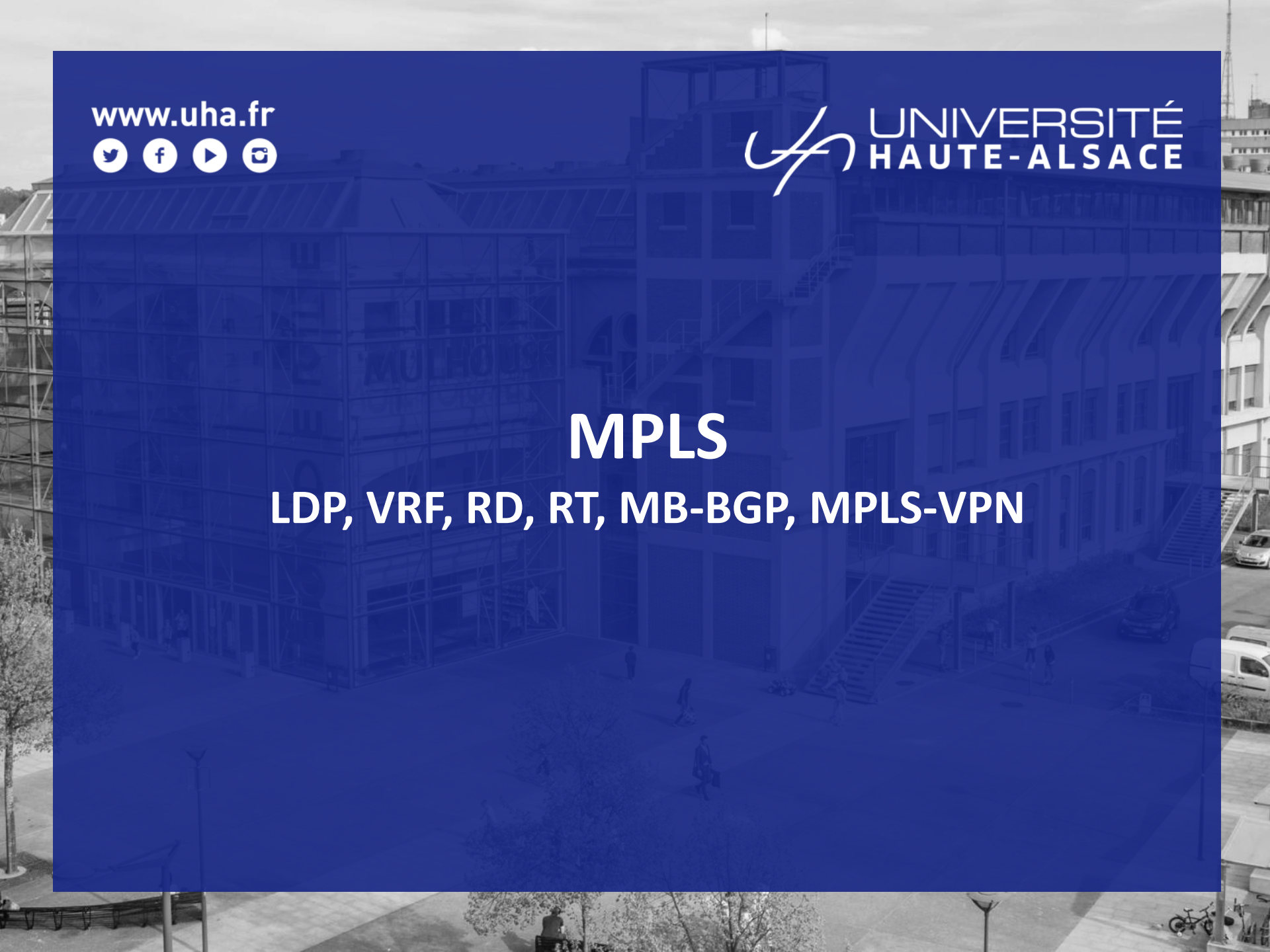
Plan de la ressource

01

- Introduction et généralité
- BGP (Border Gateway Protocol)

02

- **MPLS**
- **MPLS-VPN L3**



MPLS

LDP, VRF, RD, RT, MB-BGP, MPLS-VPN

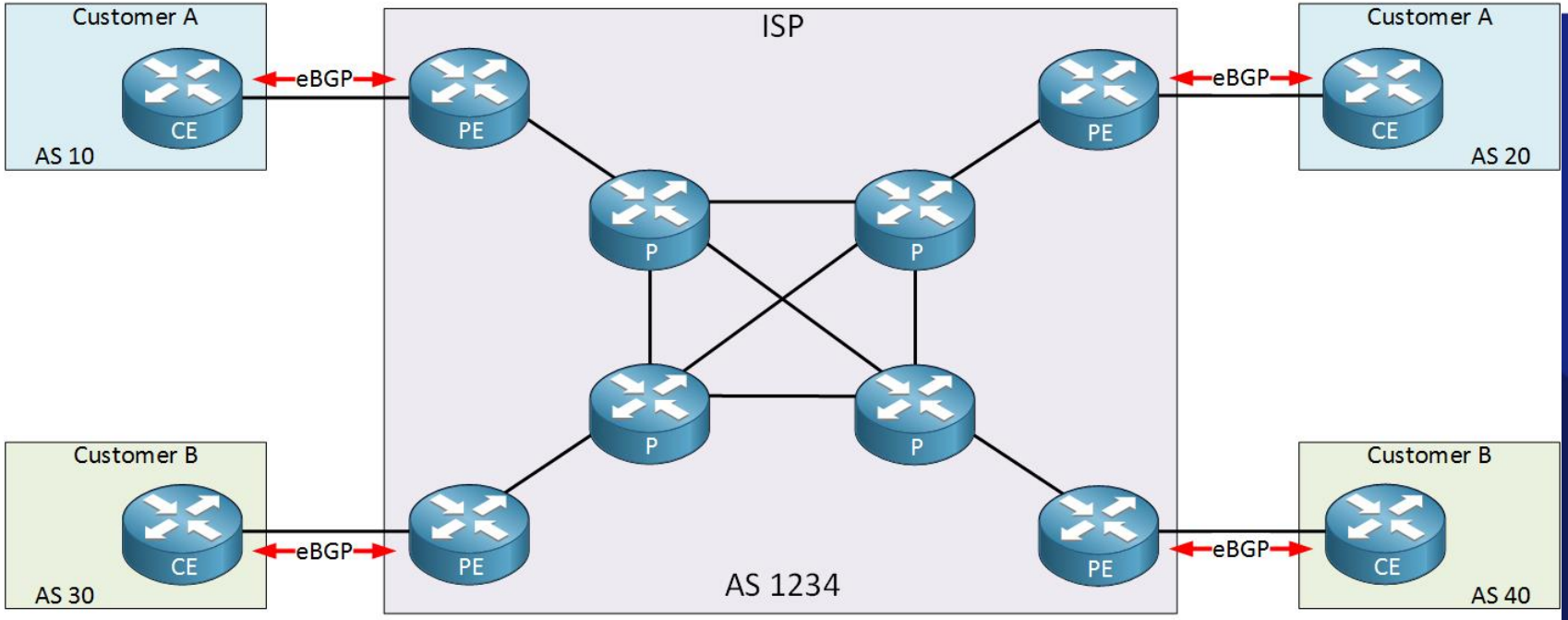
Les besoins d'un opérateur

- Les opérateurs ont besoin de plus de « certitudes » quant au routage du trafic :
  - Le routage d'un flux doit emprunter le même chemin : **mode connecté**
  - Offrir un service de connectivité dédié aux clients
  - Les décisions de routage doivent prendre en compte l'utilisation actuelle du débit des liens, afin d'optimiser la bande passante et éviter la congestion : **Traffic Engineering**
  - Un flux doit être acheminé en garantissant le respect de certaines contraintes : Quality Of Service (**QoS**)
- Parmi les solutions qui peuvent répondre à ses problèmes: **MPLS**, ATM, FR …

Pourquoi avons-nous besoin de MPLS?

- Dans les réseaux IP traditionnels, le routage des paquets s'effectue en fonction de l'adresse de destination
- Chaque routeur, pour déterminer le prochain saut consulte sa table de routage et détermine l'interface de sortie vers laquelle il faut envoyer le paquet ➔ **consommation CPU**
- Besoin d'une méthode plus efficace pour le routage des paquets
- À l'origine, l'objectif était de donner aux routeurs IP une plus grande puissance de commutation, en basant la décision de routage sur une information de label sans avoir à consulter l'entête IP et la table de routage.
- L'intérêt de MPLS n'est actuellement plus la rapidité mais l'offre de services qu'il permet : **MPLS-VPN** (réseaux privés virtuels) et le **MPLS-TE** (Traffic Engineering), qui ne sont pas réalisables sur des infrastructures IP traditionnelles.

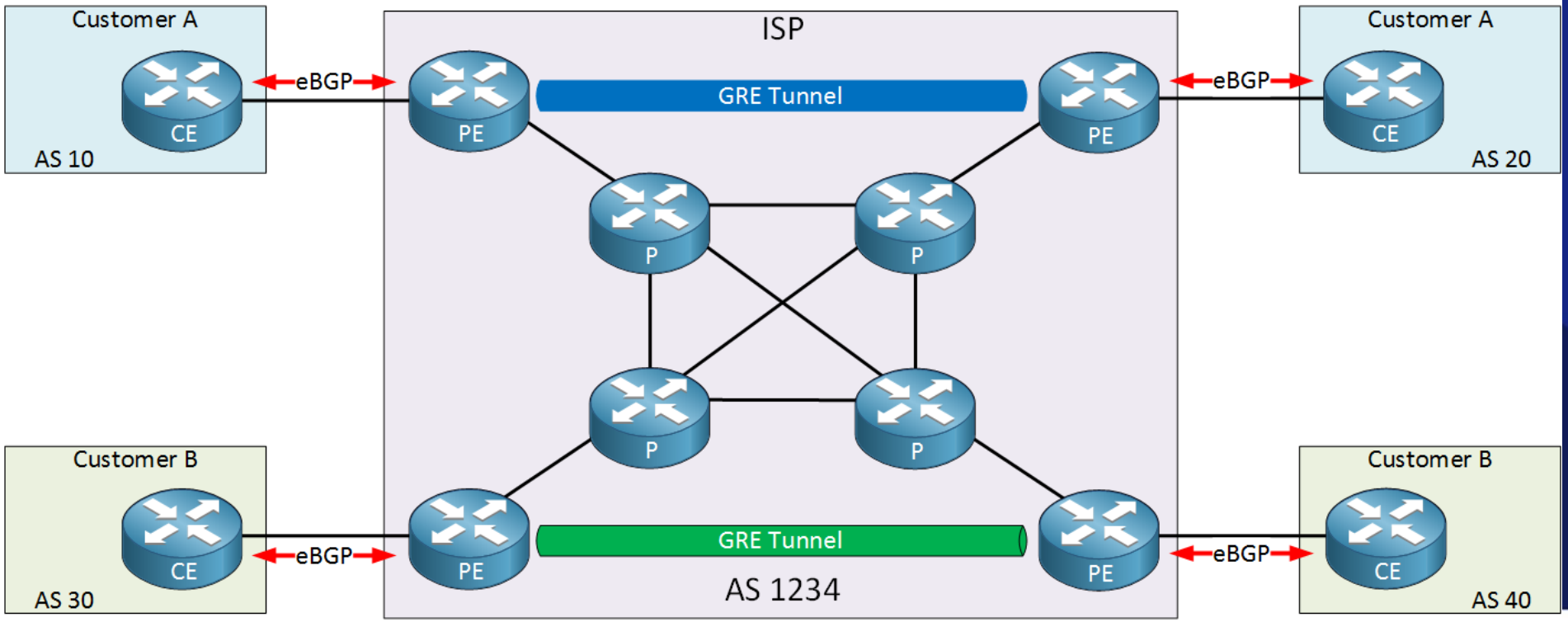
Pourquoi avons-nous besoin de MPLS?



- Comment assurer une connexion privée / isolée entre les différents sites de chaque client ?
- Comment éviter d'inonder l'ISP par des routes des clients ?
- Est-ce que l'ensemble des routeurs de l'ISP doivent prendre parti de la mise en place de la connexion entre les sites d'un client ?

Pourquoi avons-nous besoin de MPLS?

- Solution possible : création des tunnels !

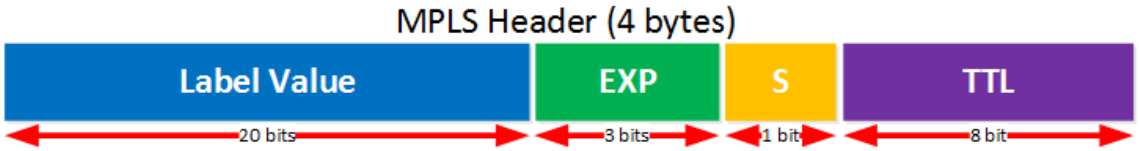


- Mais ce n'est pas scalable !

➔ Besoin d'une solution qui répond à tous ces enjeux : **MPLS**

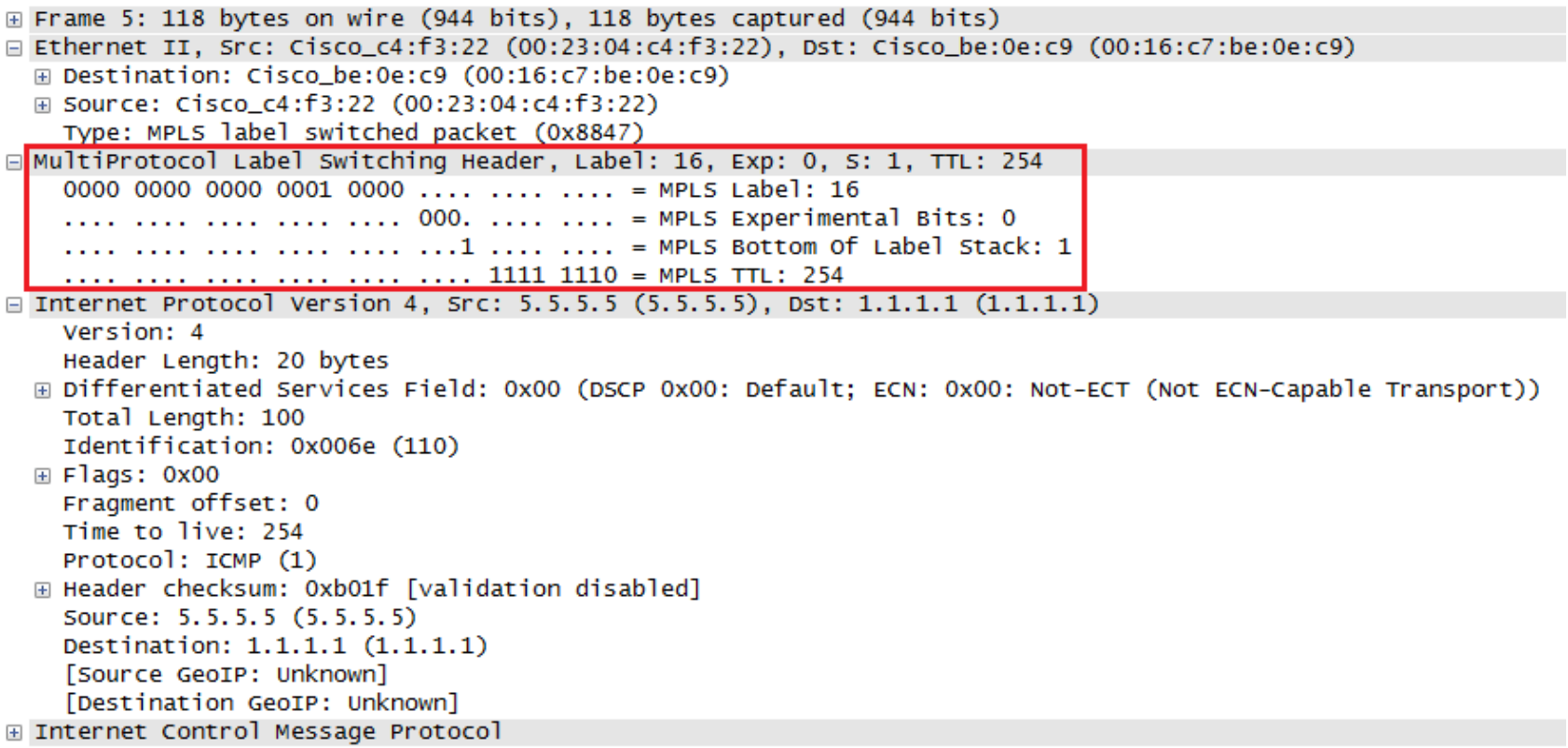
MPLS : notions de base et fonctionnement

- **MPLS** (Multi Protocol Label Switching)
  - **Multi protocol** : en plus d'IP, on peut transporter à peu près tout… IPv6, Ethernet, PPP, frame-relay, etc.
  - **Label switching** : le transfert se fait en fonction des étiquettes / labels, et non en recherchant la destination dans la table de routage.



- **Label value** : champs qui stocke la valeur du label
- **EXP**: trois bits d'expérimentation. Utilisés pour la QoS, où la valeur de priorité IP du paquet IP sera copiée ici.
- **S** : c'est le bit «bas de pile». Avec MPLS, il est possible d'ajouter plusieurs labels. Lorsque ce bit est défini à un, il s'agit du dernier en-tête MPLS. Lorsqu'il est défini à zéro, alors il reste un ou plusieurs en-têtes MPLS.
- **TTL** : chaque saut décrémente le TTL de un.
- L'en-tête MPLS est ajouté entre les en-têtes L2 et L3. C'est pourquoi nous l'appelons protocole «**layer 2.5**».

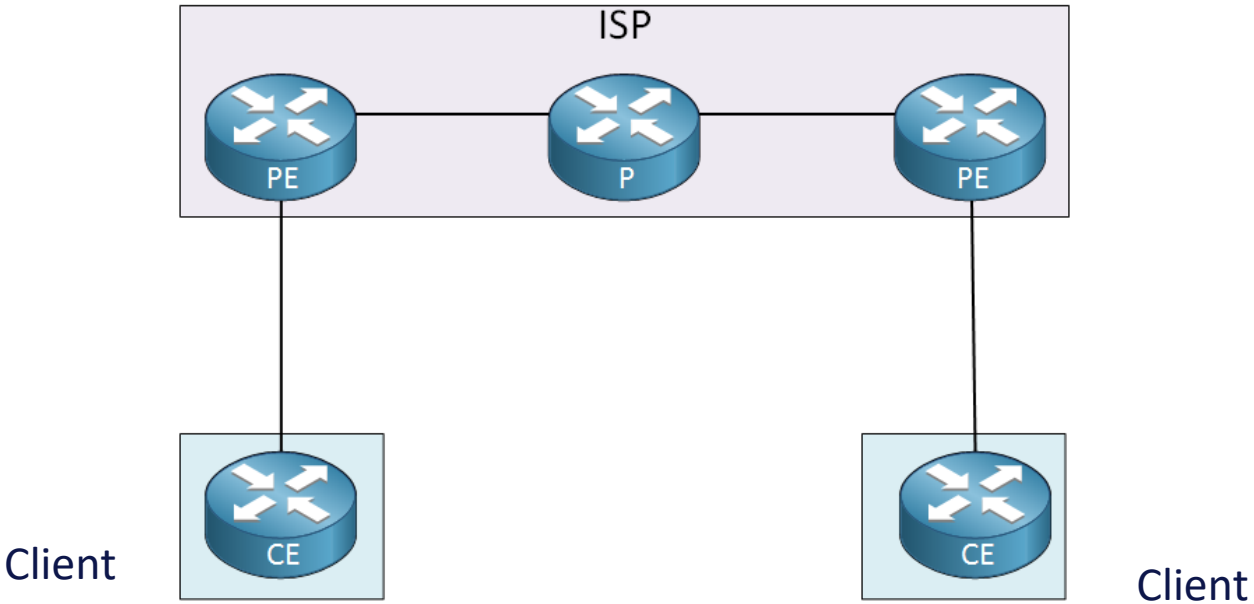
MPLS : notions de base et fonctionnement



D'où vient la valeur du label ?

➔ MPLS utilise un protocole appelé **LDP (Label Distribution Protocol)**

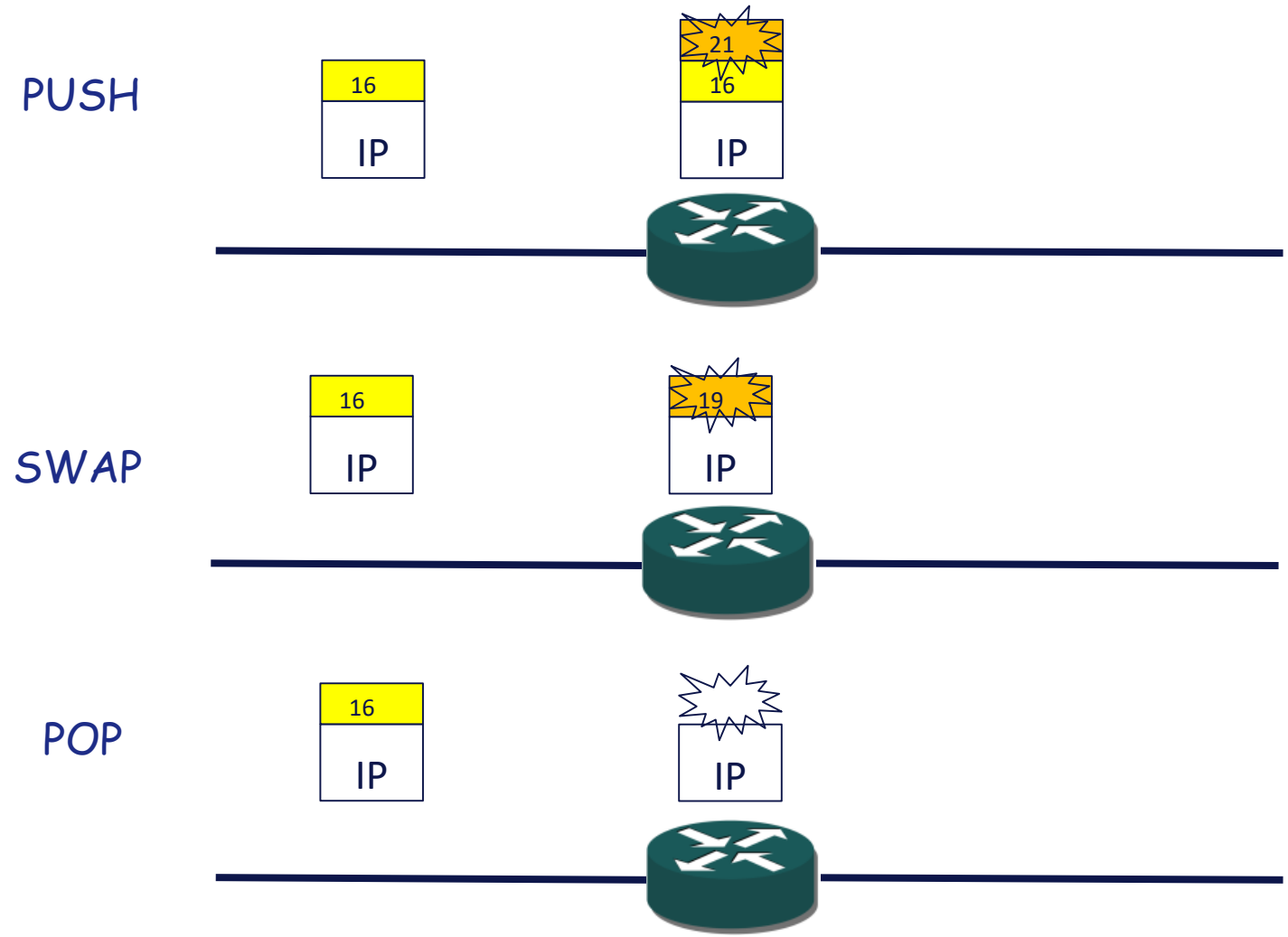
MPLS : notions de base et fonctionnement



- CE (Customer Edge) : C'est l'équipement d'interconnexion entre le réseau du client et celui du FAI. Il peut être de L2 ou L3 et il n'est pas concerné par MPLS.
- PE (Provider Edge) / **LER (Label Edge Router)**: Il se trouve à la périphérie du réseau du FAI et il a un rôle important : il reçoit et ajoute aux paquets ou trames du client une **étiquette MPLS** et les transmet vers le cœur du réseau.
- P (Provider) / **LSR (Label Switch Router) :** Connecté aux PE et aux autres routeurs P. Il a pour rôle de commuter les paquets en fonction de leurs étiquettes. Appelé aussi un routeur de transit.

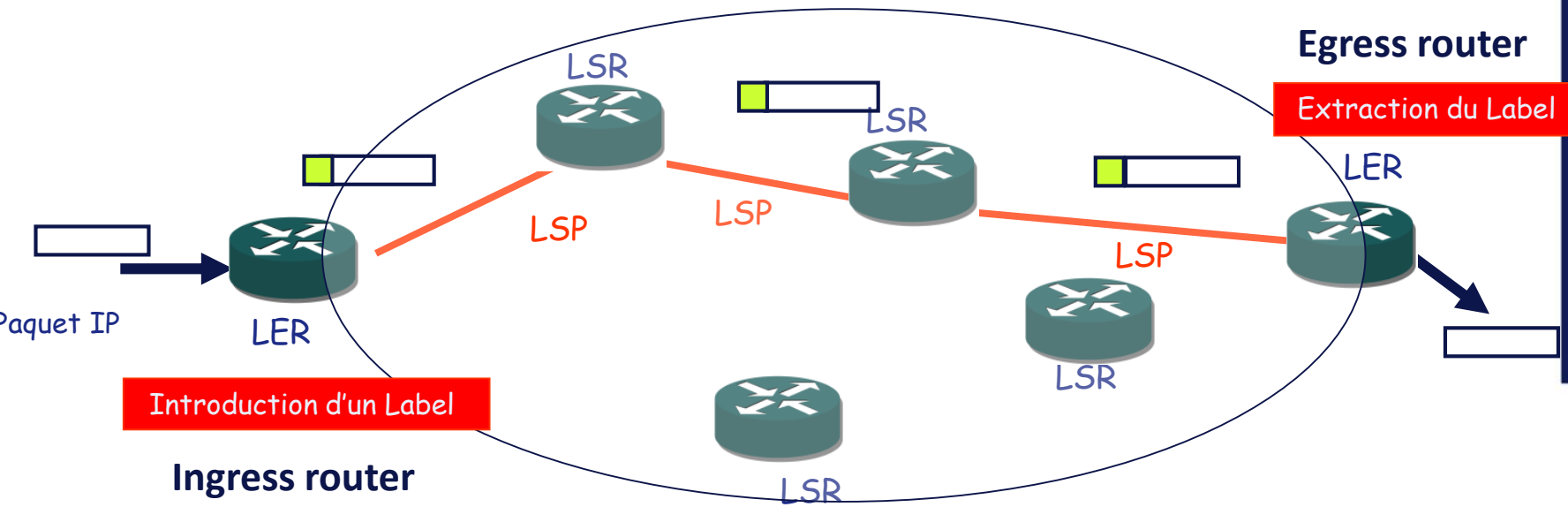
MPLS : notions de base et fonctionnement

- Trois opération de base sont réalisées par les routeurs **MPLS** :



MPLS : notions de base et fonctionnement

- Chaque paquet IP arrivant dans le réseau se voit assigner un label afin de suivre un chemin défini préalablement
- **LSP** = Label Switched Path : chemin unidirectionnel entre deux routeurs MPLS
- **PE / LER** : routeurs d'accès au domaine MPLS
- **P / LSR** : routeur de cœur de réseau



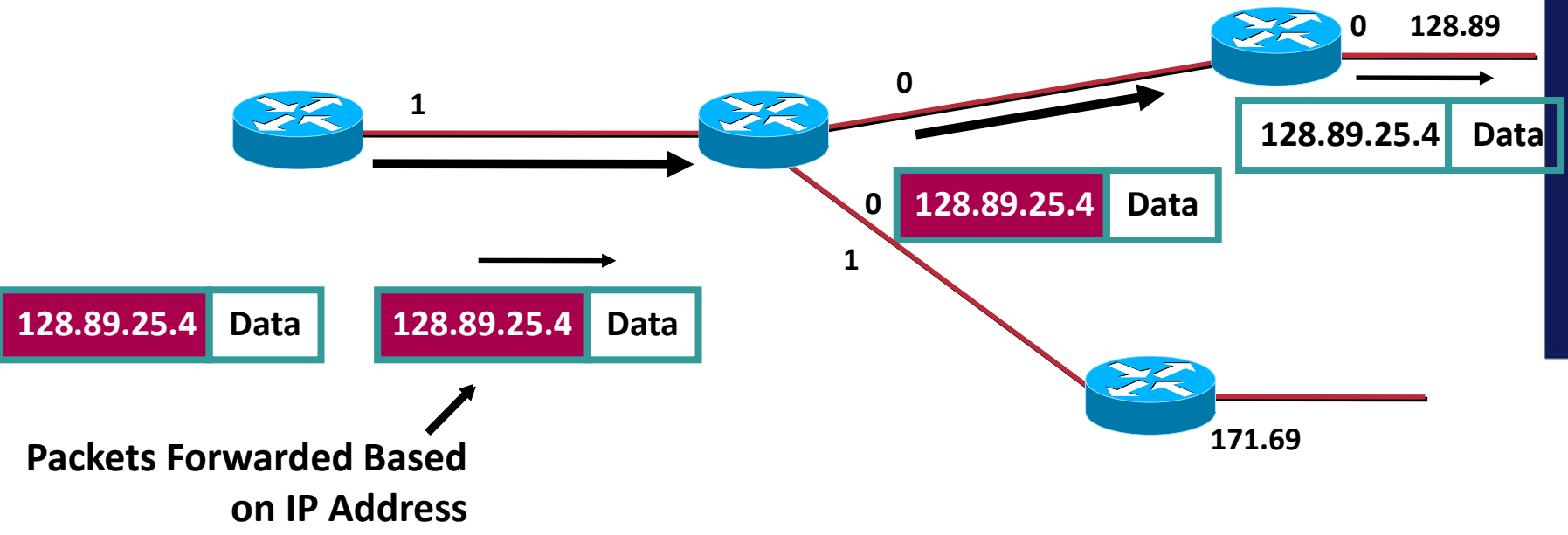
MPLS : notions de base et fonctionnement

➔ Routage de paquets VS commutation de paquets

Address Prefix	I/F
128.89	1
171.69	1
…	

Address Prefix	I/F
128.89	0
171.69	1
…	

Address Prefix	I/F
128.89	0
…	…



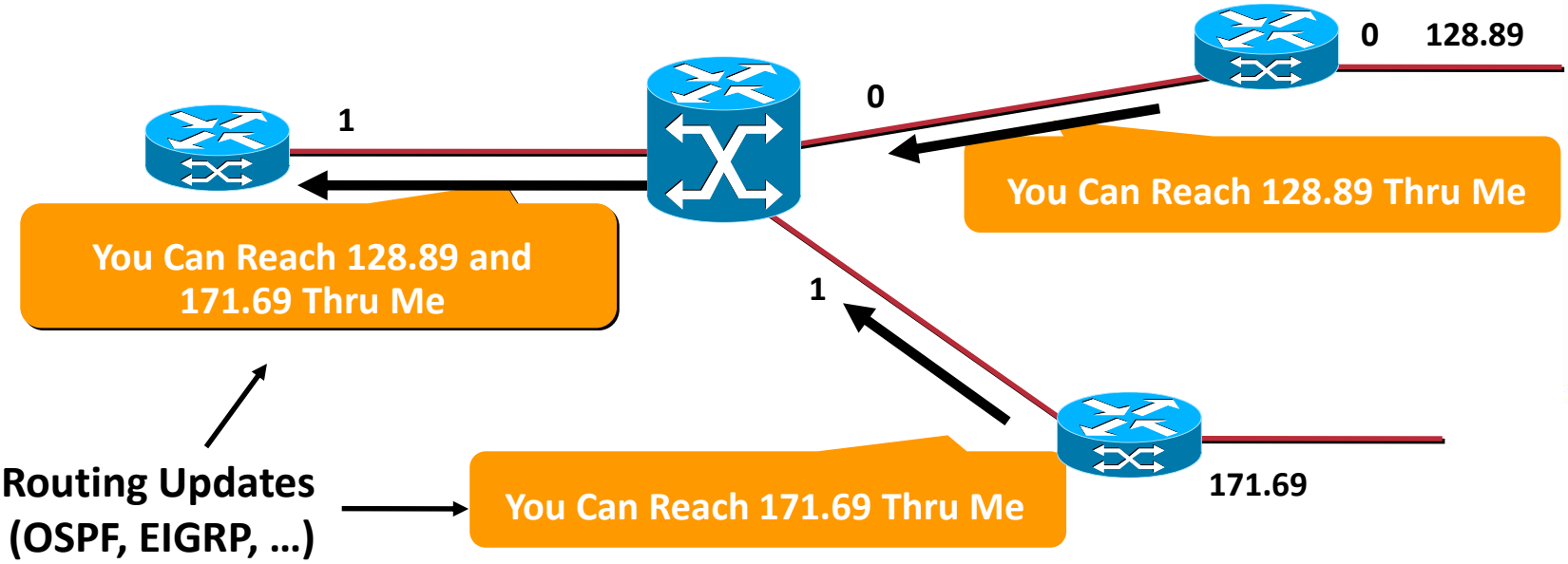
MPLS : notions de base et fonctionnement

➔ Alimentation des tables de routage (RIB)

Address Prefix	Out I'face
128.89	1
171.69	1
…	…

Address Prefix	Out I'face
128.89	0
171.69	1
…	…

Address Prefix	Out I'face
128.89	0
…	…



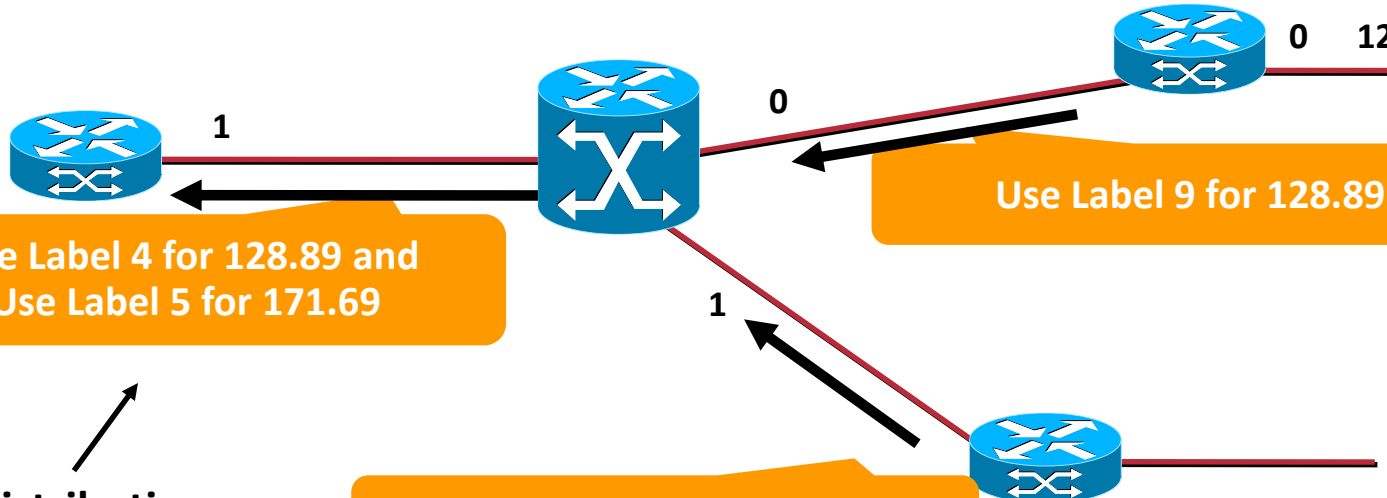
MPLS : notions de base et fonctionnement

➔ Affectation des labels (LDP)

In Label	Address Prefix	Out I'face	Out Label
-	128.89	1	4
-	171.69	1	5
…	…	…	…

In Label	Address Prefix	Out I'face	Out Label
4	128.89	0	9
5	171.69	1	7
…	…	…	…

In Label	Address Prefix	Out I'face	Out Label
9	128.89	0	-
…	…	…	…



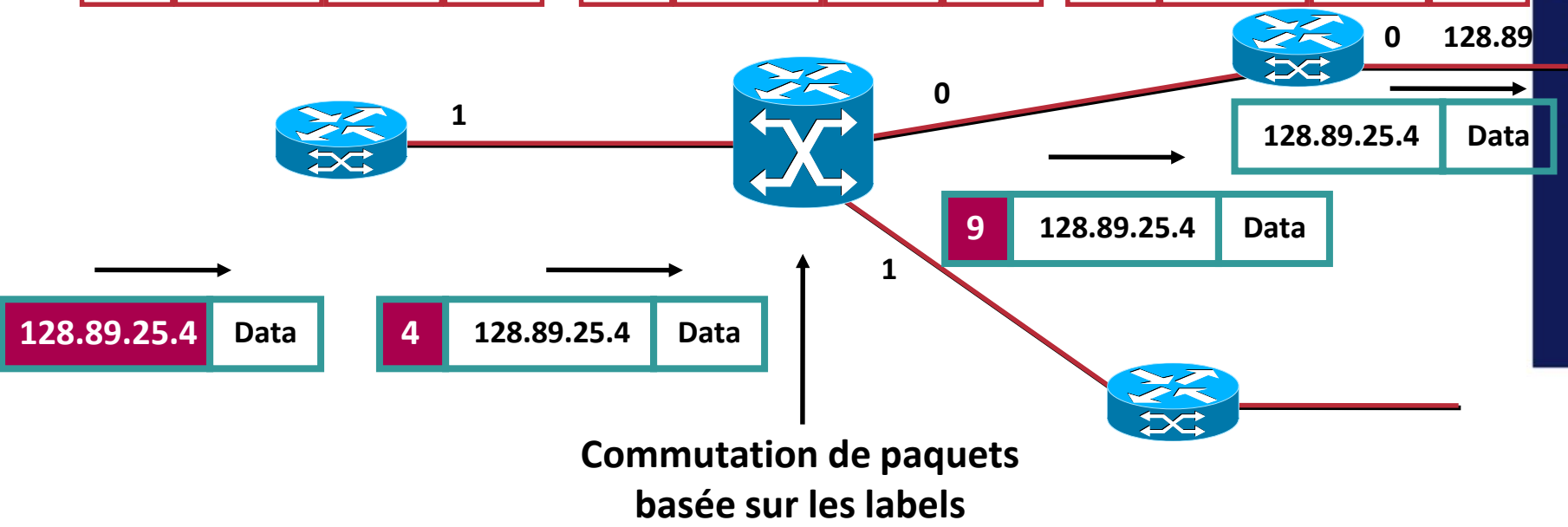
MPLS : notions de base et fonctionnement

➔ Commutation de paquets :

In Label	Address Prefix	Out I'face	Out Label
-	128.89	1	4
-	171.69	1	5
…	…	…	…

In Label	Address Prefix	Out I'face	Out Label
4	128.89	0	9
5	171.69	1	7
…	…	…	…

In Label	Address Prefix	Out I'face	Out Label
9	128.89	0	-
…	…	…	…

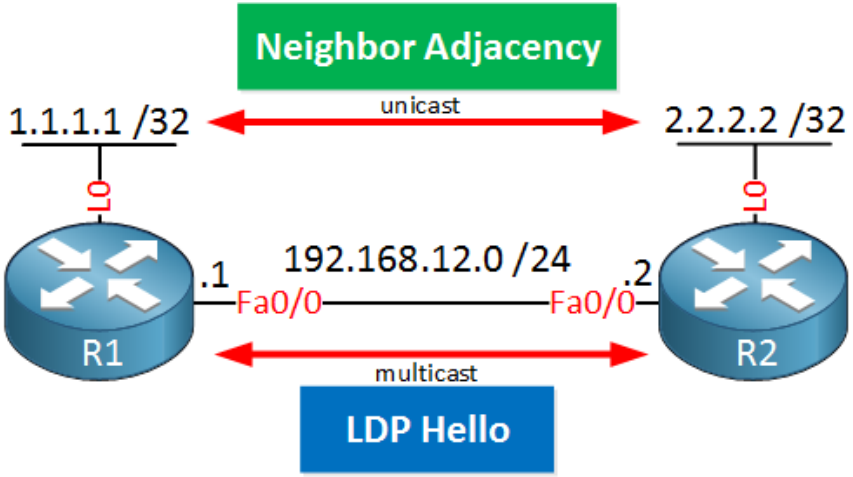


Commutation de paquets basée sur les labels

➔ Qui génère ces labels ? **LDP**

LDP (Label Distribution Protocol)

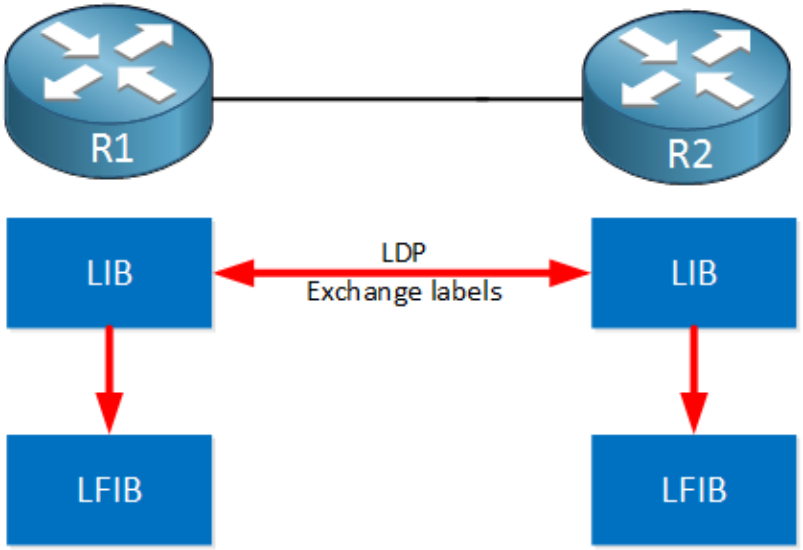
- **LDP** est un protocole qui génère et échange automatiquement des **labels** entre les routeurs.
- Chaque routeur générera localement des labels pour ses préfixes et publiera ensuite les valeurs de ces labels à ses voisins.



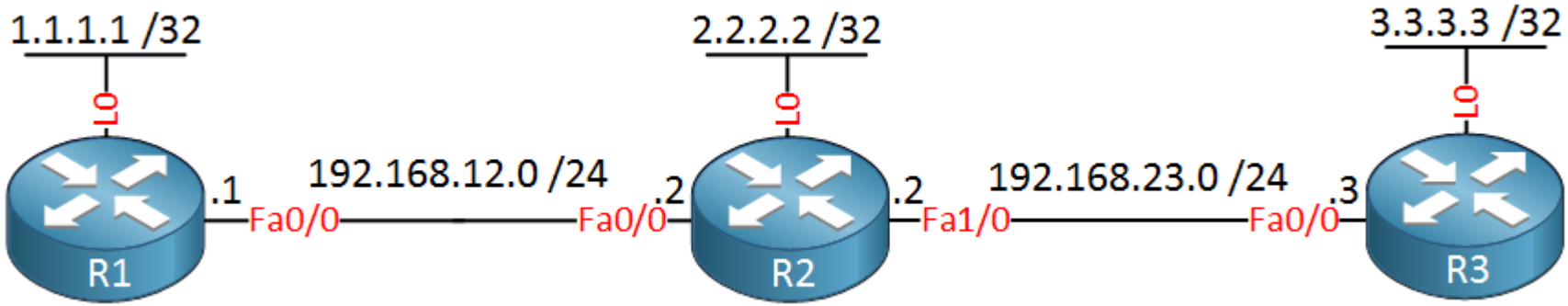
- Ici, les deux routeurs enverront des paquets Hello multicast sur leurs interfaces FastEthernet. Dans ce paquet Hello, ils publieront une adresse IP de transport qui est utilisée pour établir la connexion TCP entre eux.
- Cette connexion est utilisée pour l'échange d'informations sur les labels

LDP (Label Distribution Protocol)

- Les routeurs qui ont formé un voisinage LDP échangeront les informations des labels utilisés dans leurs LIB.
- Tous les routeurs exécutant LDP sauront désormais quelles valeurs d'étiquette à utiliser lorsqu'ils commutent un paquet MPLS vers leur voisin



LDP : Configuration et vérification




```
R1(config)#router ospf 1
R1(config-router)#network 192.168.12.0 0.0.0.255 area 0
R1(config-router)#network 1.1.1.1 0.0.0.0 area 0
```

```
R2(config)#router ospf 1
R2(config-router)#network 192.168.12.0 0.0.0.255 area 0
R2(config-router)#network 192.168.23.0 0.0.0.255 area 0
R2(config-router)#network 2.2.2.2 0.0.0.0 area 0
```

```
R3(config)#router ospf 1
R3(config-router)#network 192.168.23.0 0.0.0.255 area 0
R3(config-router)#network 3.3.3.3 0.0.0.0 area 0
```

```
RX(config)#interface FastEthernet X/X
RX(config-if)#mpls ip
```

➔ Sur les interfaces qui sont dans le domaine MPLS

LDP : Configuration et vérification

```
R2#show mpls interfaces
Interface              IP            Tunnel   BGP Static Operational
FastEthernet0/0        Yes (ldp)     No       No  No     Yes
FastEthernet0/1        Yes (ldp)     No       No  No     Yes
```

➔ Vérification des interfaces où le LDP est activé

```
R2#show mpls ldp neighbor
    Peer LDP Ident: 3.3.3.3:0; Local LDP Ident 2.2.2.2:0
        TCP connection: 3.3.3.3.40724 - 2.2.2.2.646
        State: Oper; Msgs sent/rcvd: 32/32; Downstream
        Up time: 00:20:12
        LDP discovery sources:
          FastEthernet0/1, Src IP addr: 192.168.23.3
        Addresses bound to peer LDP Ident:
          192.168.23.3    192.168.34.3    3.3.3.3
    Peer LDP Ident: 1.1.1.1:0; Local LDP Ident 2.2.2.2:0
        TCP connection: 1.1.1.1.646 - 2.2.2.2.46288
        State: Oper; Msgs sent/rcvd: 26/26; Downstream
        Up time: 00:16:37
        LDP discovery sources:
          FastEthernet0/0, Src IP addr: 192.168.12.1
        Addresses bound to peer LDP Ident:
          1.1.1.1         192.168.12.1
```

➔ Vérification du voisinage LDP

LDP : Configuration et vérification


```
R1(config)#mpls label range 100 199
R2(config)#mpls label range 200 299
R3(config)#mpls label range 300 399
```

➔ Choisir l'intervalle des valeurs possibles à utiliser pour les labels (optionnel)

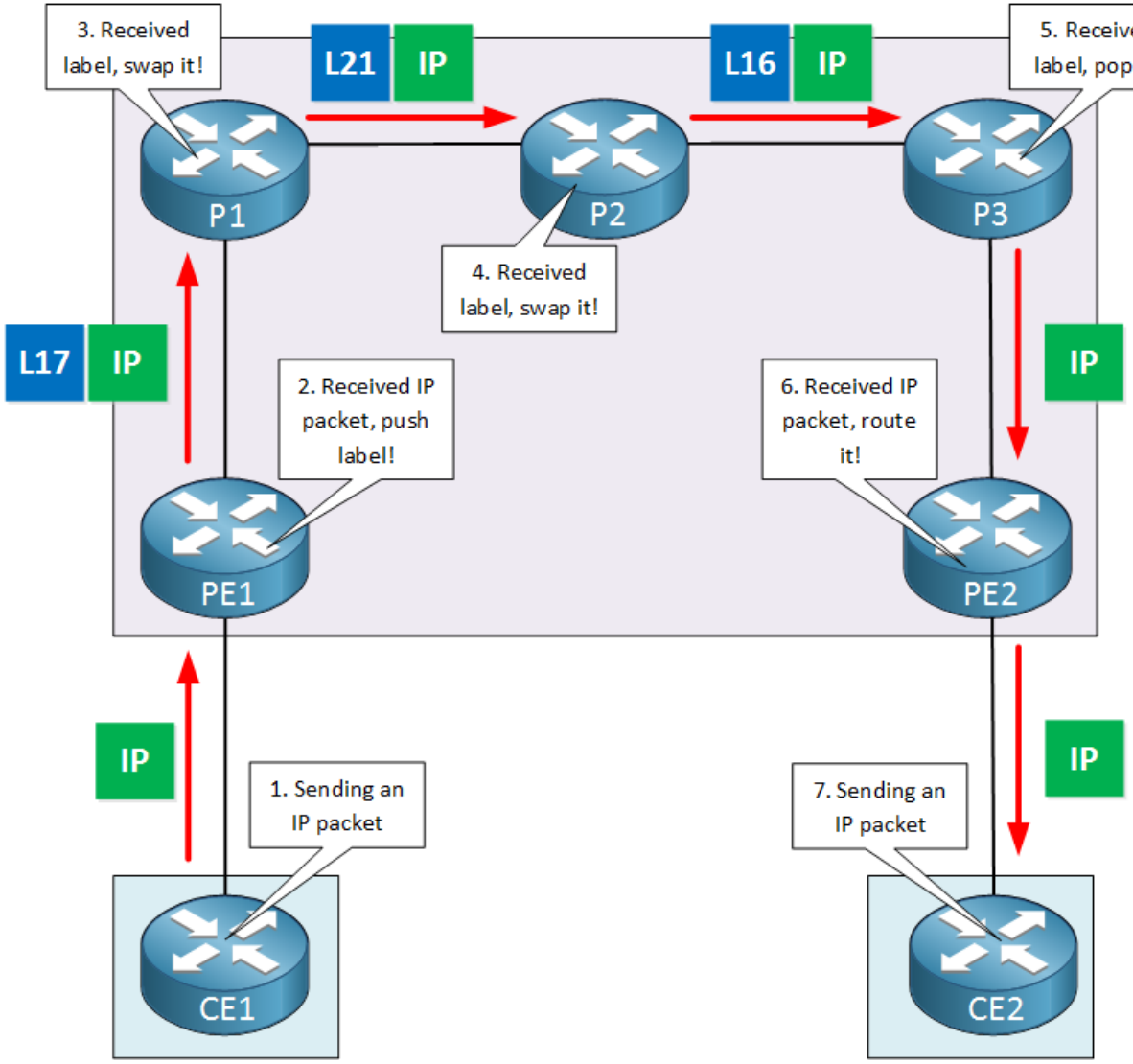
```
R1#show ip route

      1.0.0.0/32 is subnetted, 1 subnets
C        1.1.1.1 is directly connected, Loopback0
      2.0.0.0/32 is subnetted, 1 subnets
O        2.2.2.2 [110/2] via 192.168.12.2, 00:36:02,
      3.0.0.0/32 is subnetted, 1 subnets
O        3.3.3.3 [110/3] via 192.168.12.2, 00:36:02,
      192.168.12.0/24 is variably subnetted, 2 subne
C        192.168.12.0/24 is directly connected, Fast
L        192.168.12.1/32 is directly connected, Fast
O     192.168.23.0/24 [110/2] via 192.168.12.2, 00:3
```

RIB

```
R1#show mpls ldp bindings
  lib entry: 1.1.1.1/32, rev 4
        local binding:  label: imp-null
        remote binding: lsr: 2.2.2.2:0, label: 200
  lib entry: 2.2.2.2/32, rev 6
        local binding:  label: 100
        remote binding: lsr: 2.2.2.2:0,  label: imp-null
  lib entry: 3.3.3.3/32, rev 10
        local binding:  label: 102
        remote binding: lsr: 2.2.2.2:0, label: 201
  lib entry: 192.168.12.0/24, rev 2
        local binding:  label: imp-null
        remote binding: lsr: 2.2.2.2:0, label: imp-null
  lib entry: 192.168.23.0/24, rev 8
        local binding:  label: 101
        remote binding: lsr: 2.2.2.2:0, label: imp-null
```

LIB

LDP (Label Distribution Protocol) - implicit-null



➔ penultimate hop popping

- PE2 n'a pas besoin de recevoir le paquet avec un label pour les réseaux

qui lui sont directement connectés, ou bien rattachés (appris par IGP, EGP, routage statique…)

- Pour cela PE2 annonce ces réseaux avec le label « **implicit-null** » à ses voisins.

LDP : Configuration et vérification

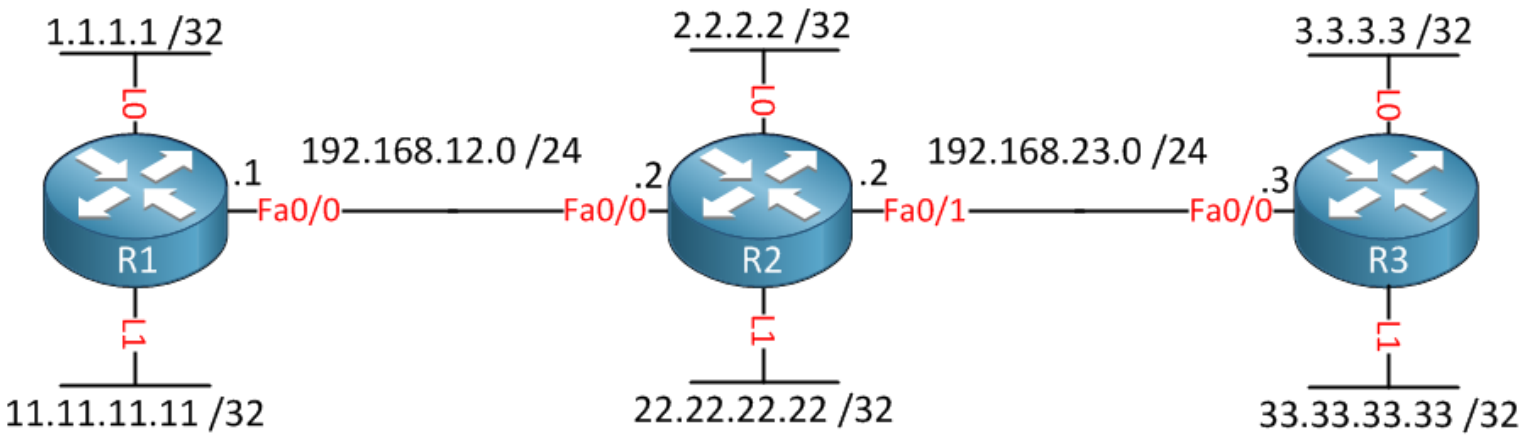
- Afficher la LFIB :

```
R1#show mpls forwarding-table
Local      Outgoing   Prefix           Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id     Switched      interface
100        Pop Label  2.2.2.2/32       0             Fa0/0      192.168.12.2
101        Pop Label  192.168.23.0/24  0             Fa0/0      192.168.12.2
102        201        3.3.3.3/32       0             Fa0/0      192.168.12.2
```

- Remarques :
  - Pas d'entrée pour 1.1.1.1/32 ou 192.168.12.0/24 car nous n'avons pas de labels pour ces préfixes.
  - Pour aller à 3.3.3.3/32, il faut rajouter le label 201 à l'en-tête MPLS avant d'envoyer le paquet à R2 (192.168.12.2)
  - Lorsque R1 reçoit quelque chose pour 2.2.2.2/32 ou 192.168.23.0/24, alors il doit 'retirer le label' avant de transmettre le paquet à R2

➔ «penultimate hop popping» : faire gagner du temps à R2 en supprimant déjà l'en-tête MPLS.

LDP : Configuration et vérification



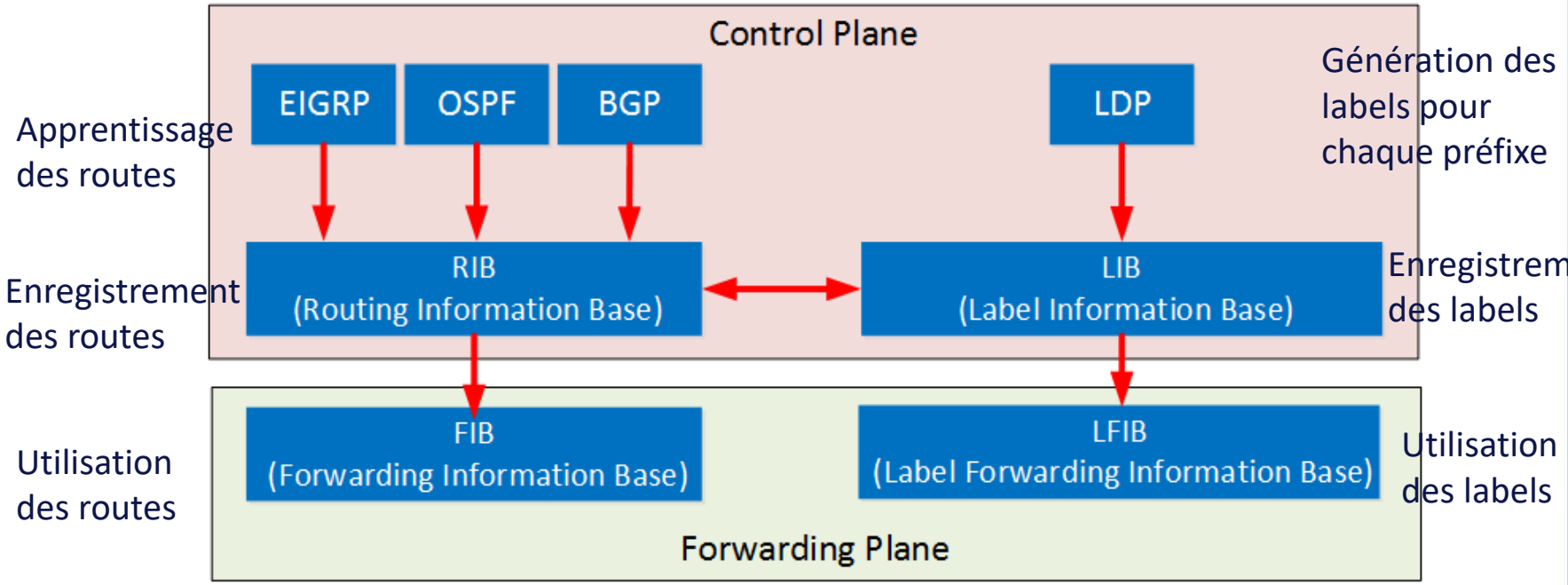

```
R1(config)#access-list 1 permit 1.1.1.1 0.0.0.0
R1(config)#no mpls ldp advertise-labels
R1(config)#mpls ldp advertise-labels for 1
```

➔ Objectif : filtrer les préfixes auxquels on va associer des labels

```
R1#show mpls forwarding-table
Local  Outgoing    Prefix           Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id     switched   interface
16     Pop tag     2.2.2.2/32       0          Fa0/0       192.168.12.2
17     Untagged    33.33.33.33/32   0          Fa0/0       192.168.12.2
18     Untagged    3.3.3.3/32       0          Fa0/0       192.168.12.2
19     Untagged    22.22.22.22/32   0          Fa0/0       192.168.12.2
20     Untagged    192.168.23.0/24  0          Fa0/0       192.168.12.2
```

LDP (Label Distribution Protocol)

- Où sont stocker les labels générés par LDP ?

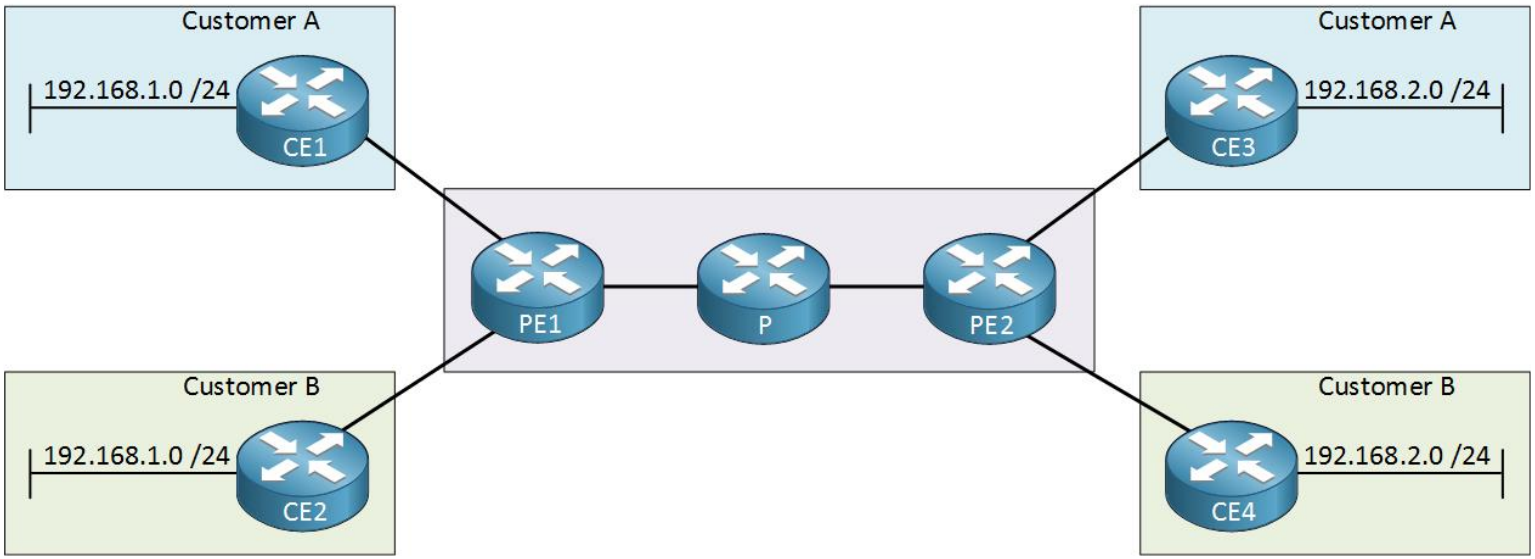


- MPLS se base sur ces labels pour orienter les paquets

MPLS-VPN

VRF, RD, RT, MB-BGP

MPLS-VPN Layer 3



- Une entreprise sur plusieurs sites géographiques aura besoin d'interconnecter les réseaux informatiques de ses sites.
- La solution de liaison spécialisée est très couteuse !
- Les FAI disposent de backbones étendus qui couvre une large portion de territoire. Donc plus simple pour une entreprise de relier ses sites aux points de présence (**POP**) des FAI et mettre en place une solution **VPN** (Virtual Private Networks).

MPLS-VPN Layer 3

- Dans l'optique **MPLS-VPN**, un VPN est un ensemble de sites placés sous la même autorité, ou groupés suivant un intérêt particulier.
- **MPLS-VPN** permet d'éviter d'inonder le cœur réseau des FAI avec les routes des clients.
- Avec **MPLS-VPN** on peut interconnecter les différents sites d'un client d'une manière complétement isolée des autres clients.
- Les clients peuvent avoir le même plan d'adressage privée !
- On parle '**Layer 3**' car on se base sur les réseaux IP
- La mise en place du service **MPLS-VPN** nécessite plusieurs notions et mécanismes :
  - **VRF** (Virtual Routing and Forwarding)
  - **RT** (Route Target)
  - **RD** (Route Distinguisher)
  - **MB-BGP** (Multi Protocol BGP)

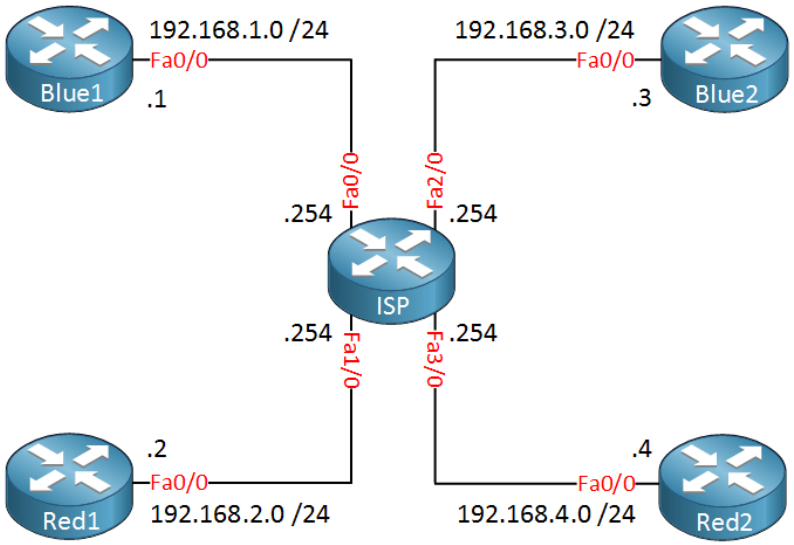
  } Control Plane

  - Le **label VPN**

  } Data Plane

MPLS-VPN – VRF (Virtual Routing and Forwarding)

- Par défaut un routeur utilise une seule table de routage pour l'ensemble des routes qu'il connait.
- Avec **VRF**, un routeur peut avoir de multiple tables de routage virtuelles qui vont être attachées aux interfaces.
- C'est l'équivalant du **VLAN** pour les commutateurs
- VRF sans MPLS est appelé **VRF lite**




```
ISP#show ip route connected
C    192.168.4.0/24 is directly connected, FastEthernet3/0
C    192.168.1.0/24 is directly connected, FastEthernet0/0
C    192.168.2.0/24 is directly connected, FastEthernet1/0
C    192.168.3.0/24 is directly connected, FastEthernet2/0
```

Table de routage par défaut

➔ Les préfixes des deux clients se trouvent dans la même table de routage !

MPLS-VPN – VRF (Virtual Routing and Forwarding)

```
ISP(config)#ip vrf Red
ISP(config-vrf)#exit
ISP(config)#ip vrf Blue
ISP(config-vrf)#exit
```

➔ Création des VRF «Red» et «Blue» pour isoler les deux clients

```
ISP(config)#interface FastEthernet 0/0
ISP(config-if)#ip vrf forwarding Blue
% Interface FastEthernet0/0 IP address 192.168.1.254 removed due to enabling VRF
Blue
ISP(config-if)#ip address 192.168.1.254 255.255.255.0
```

➔ Assigner l'interface à une table VRF

➔ Redéfinir l'adresse IP

```
ISP#show ip vrf
  Name                              Default RD          Interfaces
  Blue                                                  Fa0/0
                                                        Fa2/0
  Red                                                   Fa1/0
                                                        Fa3/0
```

➔ Vérification

MPLS-VPN – VRF (Virtual Routing and Forwarding)

```
ISP#show ip route connected
```
➔ La table de routage globale ne contient plus d'entrées

```
ISP#show ip route vrf Blue connected
C    192.168.1.0/24 is directly connected, FastEthernet0/0
C    192.168.3.0/24 is directly connected, FastEthernet2/0
```

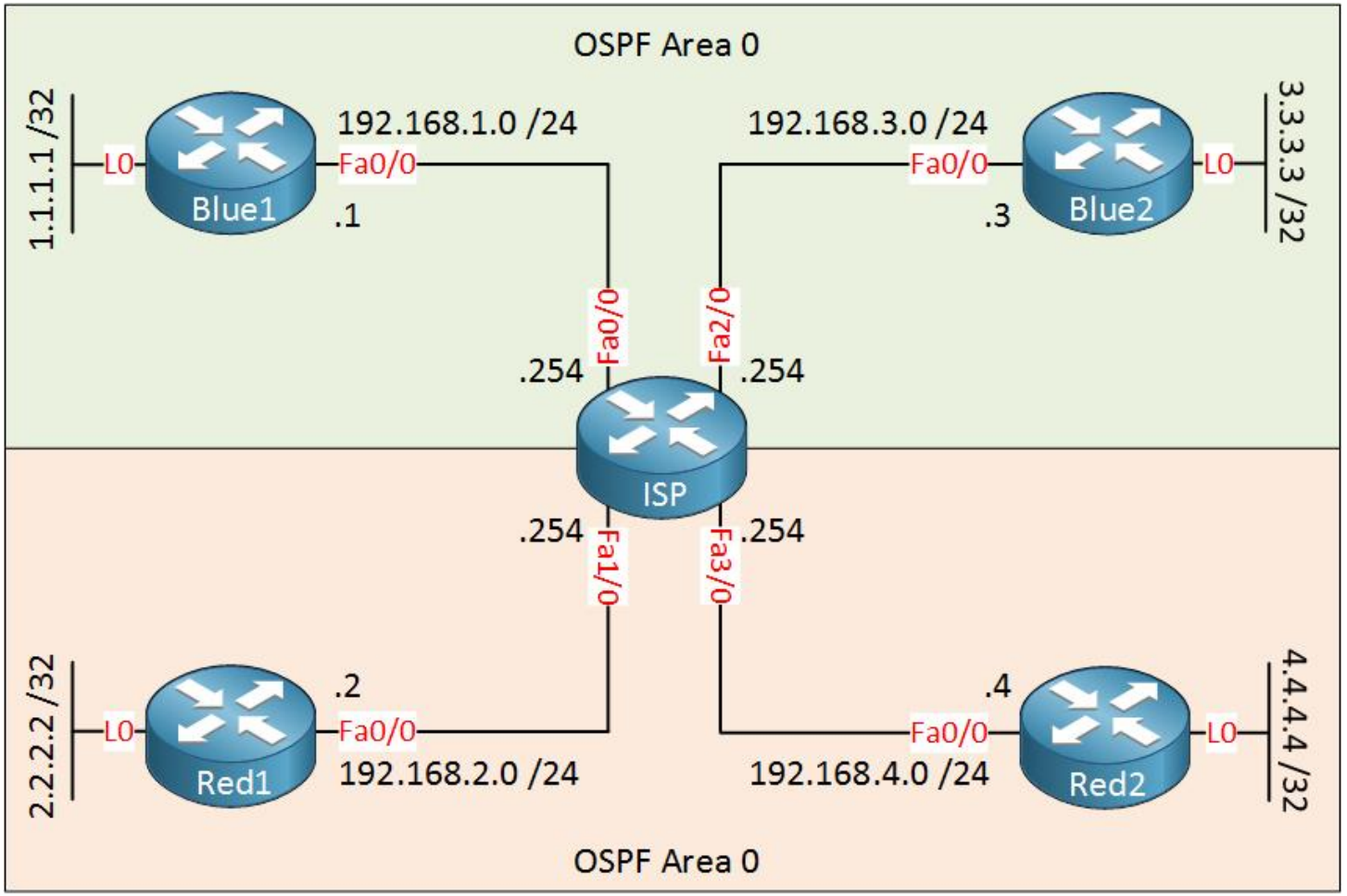
```
ISP#show ip route vrf Red connected
C    192.168.4.0/24 is directly connected, FastEthernet3/0
C    192.168.2.0/24 is directly connected, FastEthernet1/0
```

➔ Pour afficher la table VRF d'un client il faut utiliser le nom correspondant

```
ISP#ping vrf Blue 192.168.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

➔ Idem pour un ping !

MPLS-VPN – VRF (Virtual Routing and Forwarding)




```
ISP(config)#ip route vrf Blue 1.1.1.1 255.255.255.255 192.168.1.1
```

➔ Idem pour rajouter une route statique !

MPLS-VPN – VRF (Virtual Routing and Forwarding)

➔ Idem le routage dynamique !

```
Blue1(config)#router ospf 1
Blue1(config-router)#network 192.168.1.0 0.0.0.255 area 0
Blue1(config-router)#network 1.1.1.1 0.0.0.0 area 0
```

```
Blue2(config)#router ospf 1
Blue2(config-router)#network 192.168.3.0 0.0.0.255 area 0
Blue2(config-router)#network 3.3.3.3 0.0.0.0 area 0
```

```
Red1(config)#router ospf 1
Red1(config-router)#network 192.168.2.0 0.0.0.255 area 0
Red1(config-router)#network 2.2.2.2 0.0.0.0 area 0
```

```
Red2(config)#router ospf 1
Red2(config-router)#network 192.168.4.0 0.0.0.255 area 0
Red2(config-router)#network 4.4.4.4 0.0.0.0 area 0
```

```
ISP(config)#router ospf 2 vrf Red
ISP(config-router)#network 192.168.2.0 0.0.0.255 area 0
ISP(config-router)#network 192.168.4.0 0.0.0.255 area 0
```

```
ISP(config)#router ospf 1 vrf Blue
ISP(config-router)#network 192.168.1.0 0.0.0.255 area 0
ISP(config-router)#network 192.168.3.0 0.0.0.255 area 0
```

➔ Utilisation d'un autre processus OSPF !

MPLS-VPN – VRF (Virtual Routing and Forwarding)

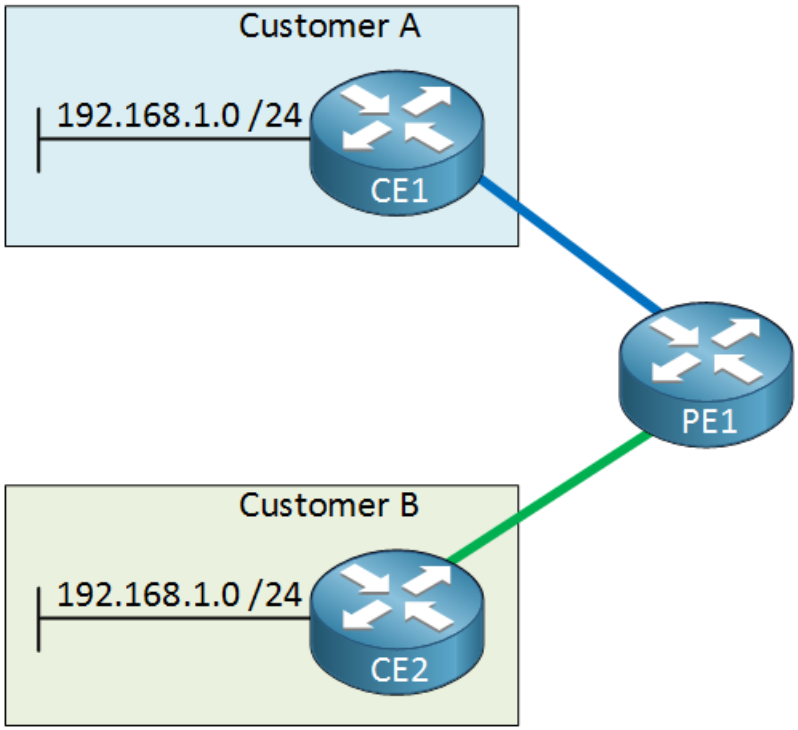
```
ISP#show ip route vrf Blue ospf

Routing Table: Blue

     1.0.0.0/32 is subnetted, 1 subnets
O       1.1.1.1 [110/2] via 192.168.1.1, 00:00:24, FastEthernet0/0
     3.0.0.0/32 is subnetted, 1 subnets
O       3.3.3.3 [110/2] via 192.168.3.3, 00:00:24, FastEthernet2/0
```

```
ISP#show ip route vrf Red ospf

Routing Table: Red

     2.0.0.0/32 is subnetted, 1 subnets
O       2.2.2.2 [110/2] via 192.168.2.2, 00:00:19, FastEthernet1/0
     4.0.0.0/32 is subnetted, 1 subnets
O       4.4.4.4 [110/2] via 192.168.4.4, 00:00:19, FastEthernet3/0
```

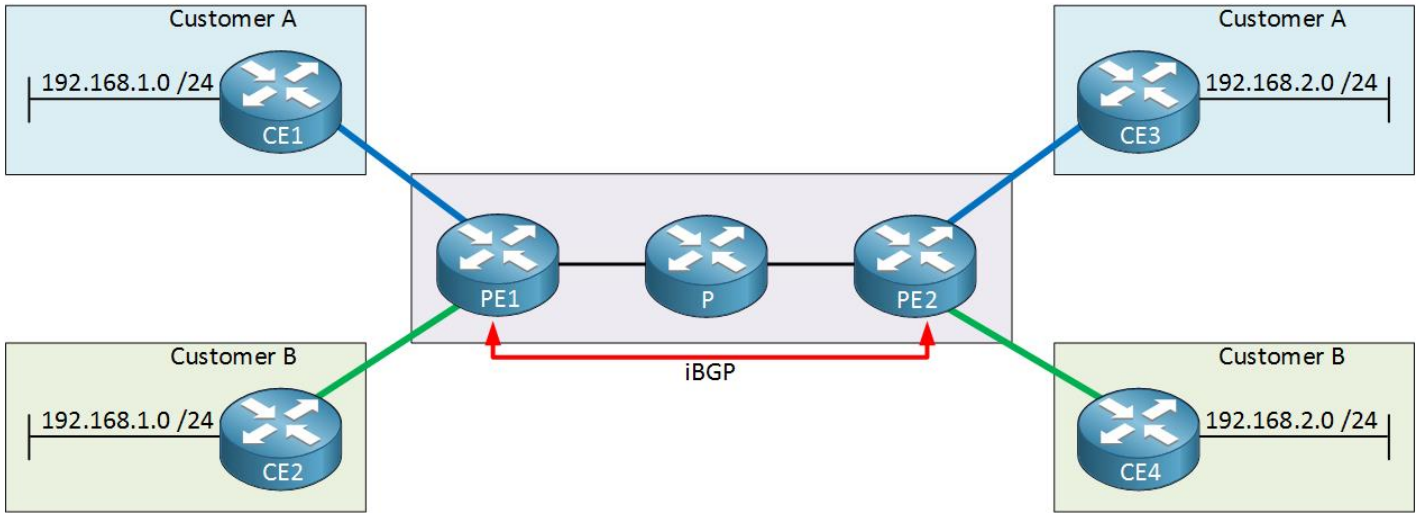
➔ Une table de routage par client !

MPLS-VPN – VRF (Virtual Routing and Forwarding)

- Donc on crée une VRF par client sur les **PE**.
- Les routeurs **P** n'ont pas besoin de **VRF** car ils vont faire que la commutation de paquet avec **MPLS**

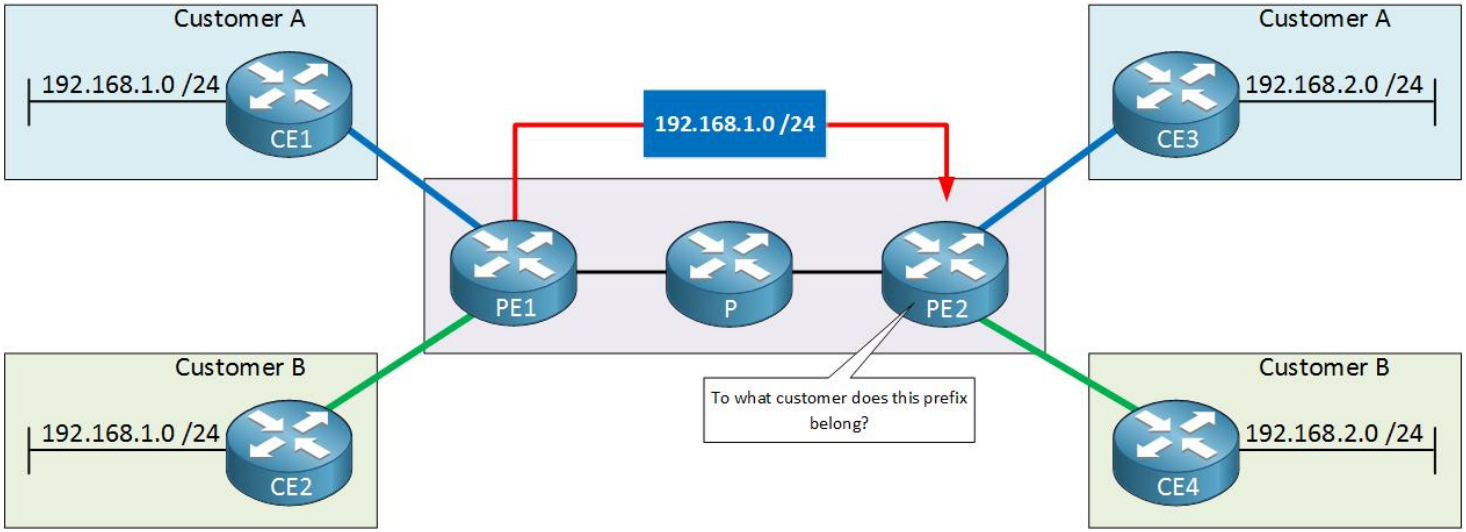


MPLS-VPN – VRF (Virtual Routing and Forwarding)



- Maintenant chaque **PE** doit informer les autres **PE** des routes qu'il a appris
  - Les **CE** informent les **PE** des préfixes qu'ils connaissent avec du **eBGP** ou du **IGP (RIP, OSPF, …)**
  - Les **PE** stockent les préfixes appris dans la **VRF** correspondante
  - Les **PE** alimentent leurs tables BGP par les préfixes de chaque **VRF**
  - Les **PE** informent les paires de leurs préfixes avec du **iBGP**

MPLS-VPN Layer 3



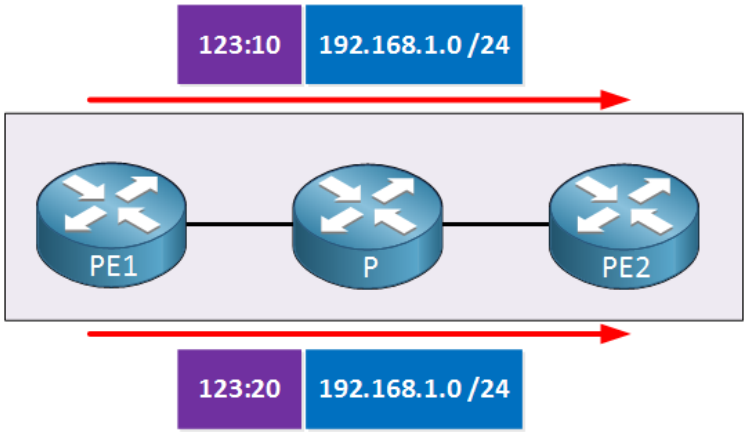
- Il y a un problème : PE1 a deux entrées pour 192.168.1.0/24 dans sa base BGP, il va donc annoncer qu'une seule entrée dans l'échange iBGP
- PE2 va recevoir l'entrée pour le préfixe 192.168.1.0/24 mais sans précision ça concerne quel client !!

➔ Il faut rendre les préfixes uniques !

➔ Solution : **RD (Route Distinguisher)**

MPLS-VPN – RD (Route Distinguisher)

- On ajoute au préfixe un champs sur 8 octets appelé **RD** :



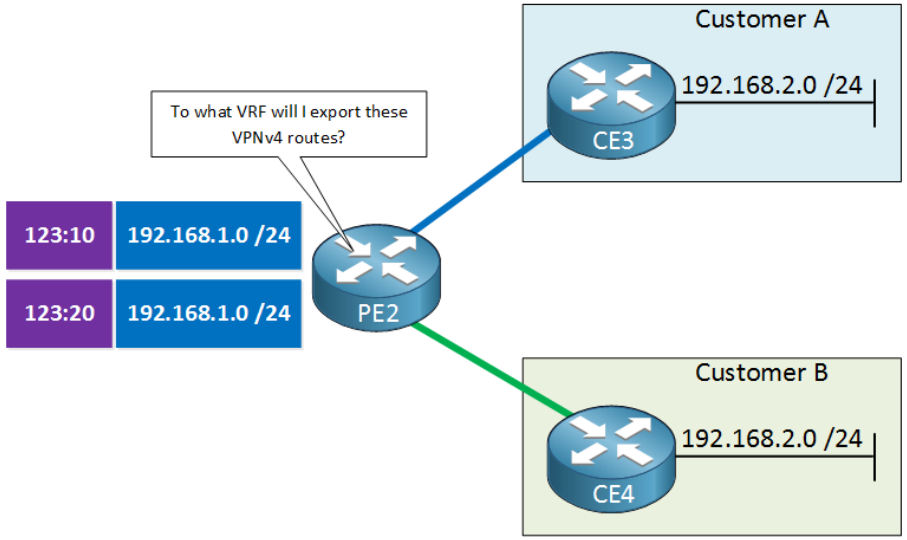
- Le **RD** a généralement la forme **ASN:NN** avec ASN c'est le N° du AS et NN un nombre qui identifie le site du client.
- Avec le **RD** on peut différencier les préfixes des différents clients
- La combinaison du **RD** avec le préfixe est appelée **VPNv4route**
- Pour pouvoir annoncer ce type de routes on a besoin du **MP-BGP**

MPLS-VPN – MP-BGP (Multi Protocol BGP)

- En effet, BGP ne sait géré que les routes IPv4 !
- Le protocole **MP-BGP** est une extension du protocole BGPv4 permettant d'échanger des routes Multicast et des routes **VPNv4**.
- **MP-BGP** adopte une terminologie similaire à BGP concernant le peering :
  - **MP-BGP** : peering entre routeurs d'un même AS ;
  - **MP-eBGP** : peering entre routeurs situés dans 2 AS différents.
- **MP-BGP** utilize un nouveau format NLRI (Network Layer Reachability Information) pour échanger les routes **VPNv4**. Ce format a les attributs suivant :
  - RD (Route Distinguisher)
  - IPv4 prefix
  - Next Hop
  - VPN Label

MPLS-VPN – RT (Route Target)

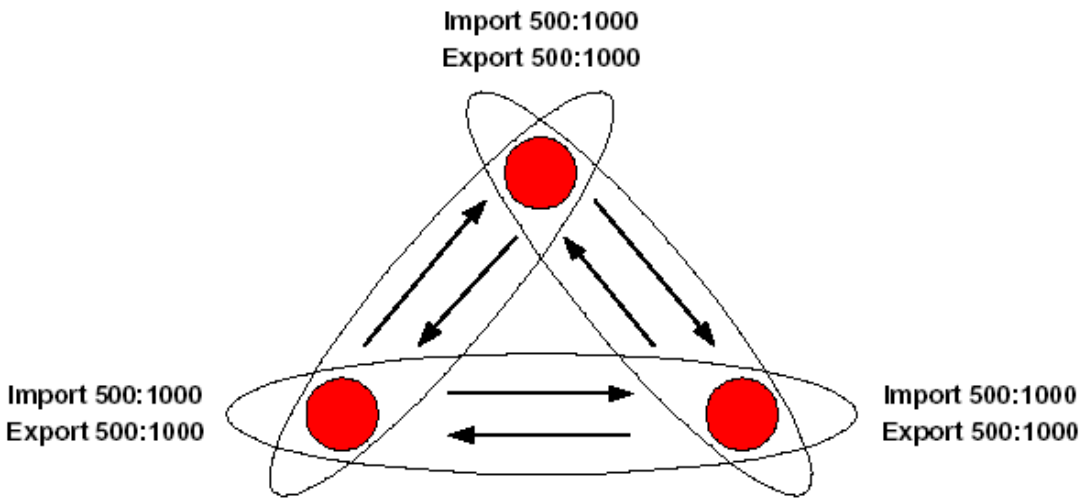
- A la réception des routes **VPNv4**, PE2 se demande il faut les exporter dans quelle **VRF** ??



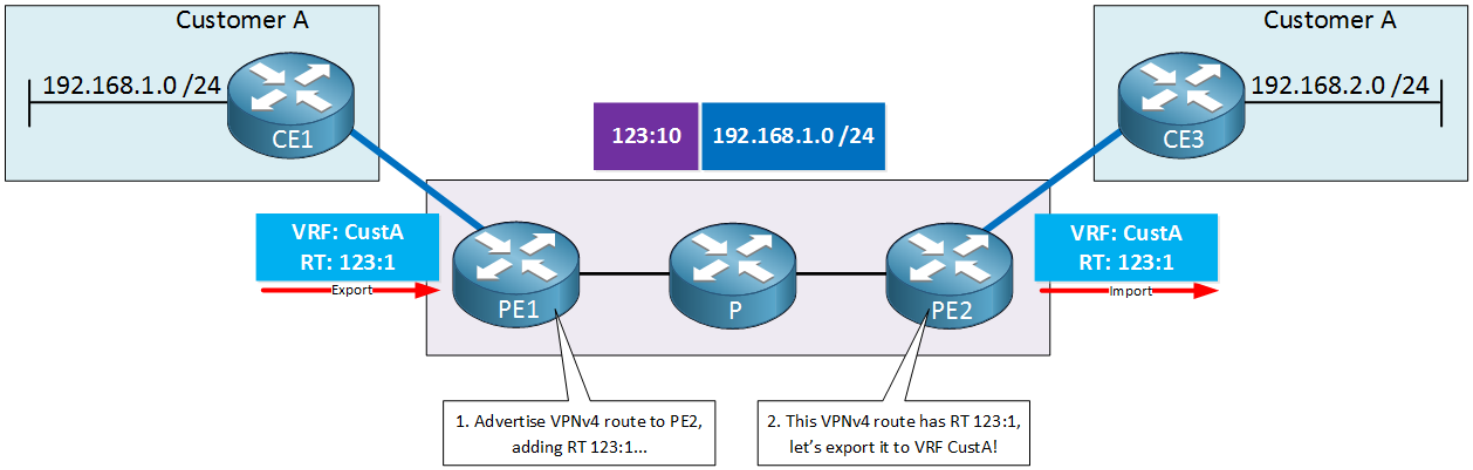
- En effet, le **RD** permet de garantir l'unicité des routes **VPNv4** échangées entre **PE**, mais ne définit pas la manière dont les routes vont être insérées dans les **VRF** de ces **PE**.
- **L'import et l'export** de routes sont gérés grâce à une communauté étendue BGP (extended community) appelée **RT (Route Target)**

MPLS-VPN – RT (Route Target)

- Les **RT** ne sont rien de plus que des sortes de filtres appliqués sur les routes **VPNv4**.
- Chaque **VRF** définie sur un **PE** est configurée pour exporter ses routes suivant un certain nombre de **RT**.
- Une route VPN exportée avec un **RT** donné sera ajoutée dans les VRF des autres PE important ce **RT**.
- Ainsi, appliquer un RT permet de savoir s'il faut exporter / importer (ou les deux) les routes VPNv4 et pour quelle VRF (client)
- Le **RT** est codé sur 8 octets du même format que le **RD** (ASN: NN).



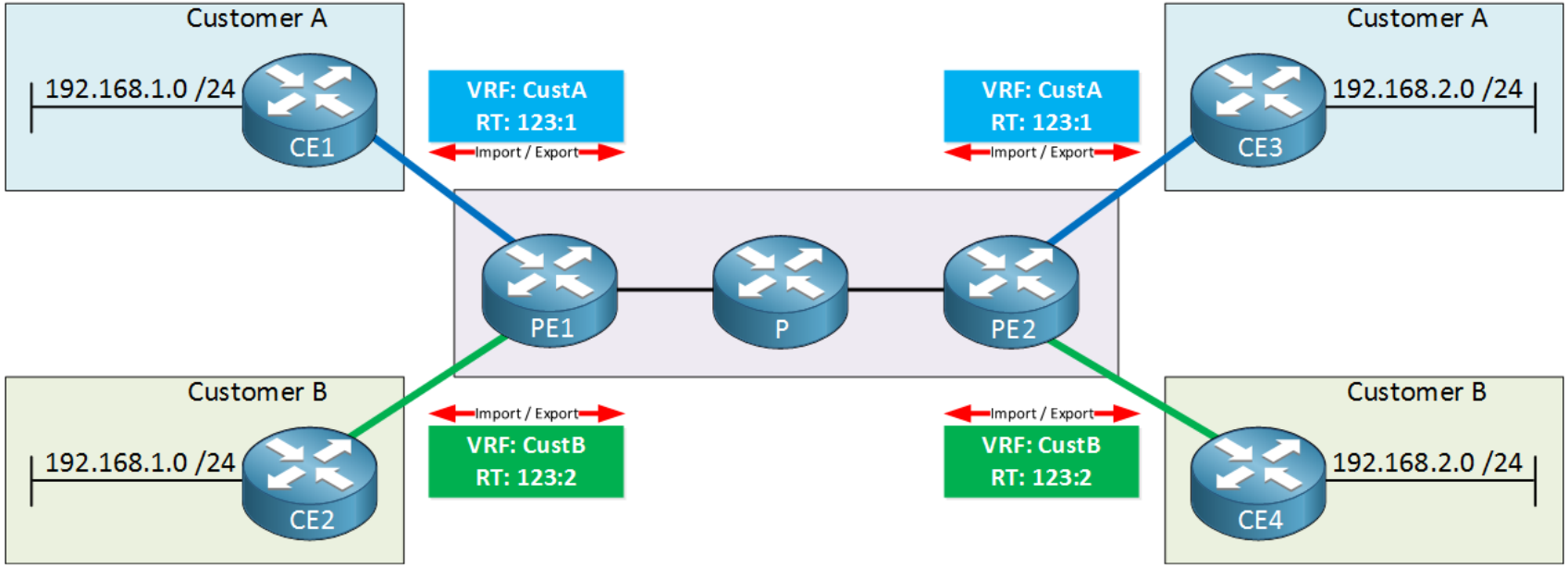
MPLS-VPN – RT (Route Target) - exemple



- Ici les deux routeurs PE sont configurés pour utiliser la VRF «CustA» pour le client A.
  - Lorsque PE1 reçoit un préfixe de CE1, il y ajoute **RD 123:10** pour créer une route VPNv4 unique.
  - PE1 est configuré pour appliquer **RT 123:1** à toutes les routes VPNv4 concernant la VRF CustA.
  - PE1 annoncera la route VPNv4 vers PE2.
  - PE2 est configuré pour importer toutes les routes VPNv4 qui utilisent **RT 123:1** dans la VRF CustA.
  - Lorsque PE2 reçoit la route VPNv4, il la redistribue dans le VRF afin que CE3 reçoit le préfixe.
  - Le résultat final sera que CE3 apprendra le préfixe 192.168.1.0 / 24 qui a été annoncé par CE1.

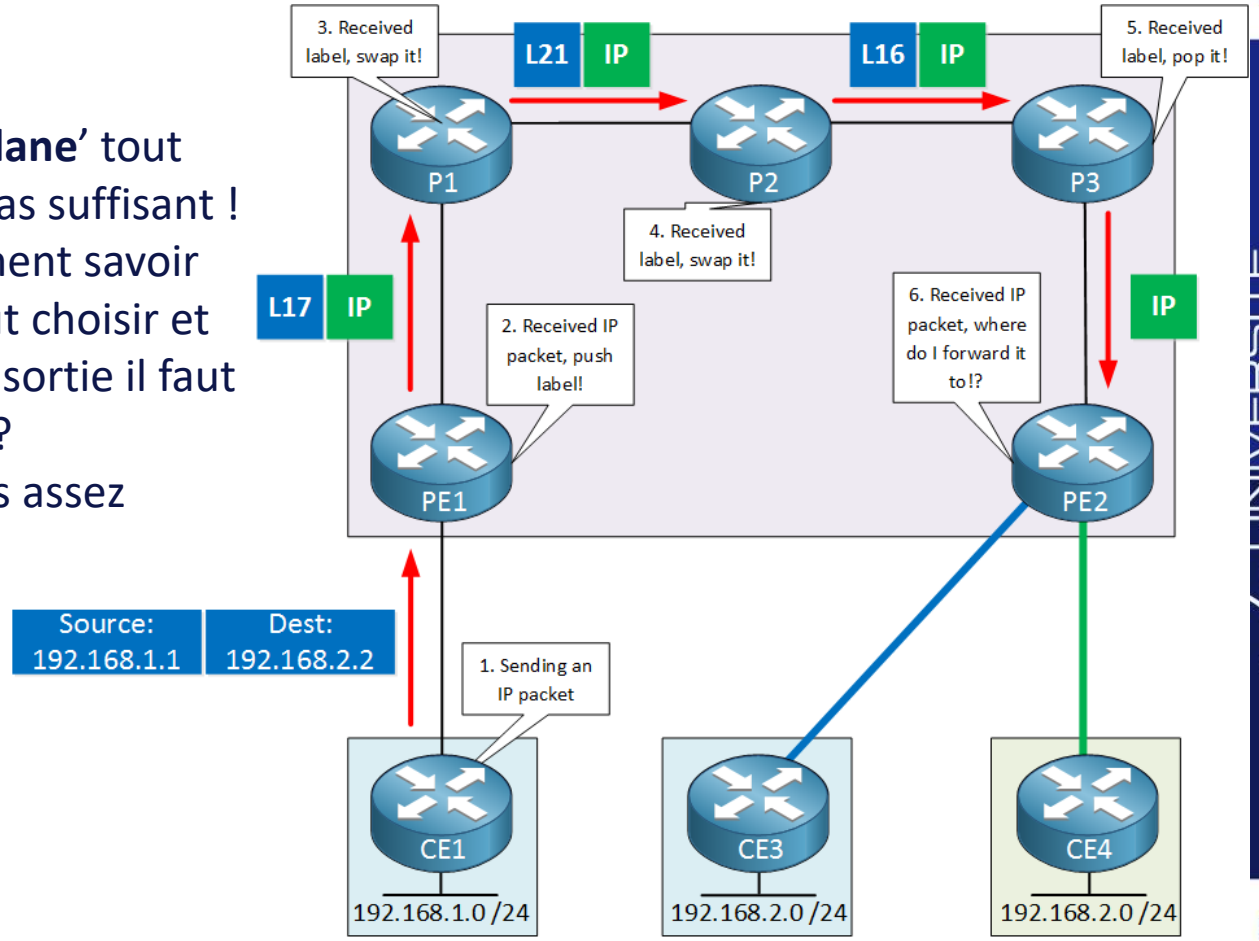
MPLS-VPN – RT (Route Target)

- Avec une configuration en **export/import** les CE d'un client sont informés de l'ensemble des préfixes des différents sites
- Le **RT** ajoute un contrôle sur les routes **VPNv4** et une meilleur flexibilité



MPLS-VPN – Le label VPN

- Au niveau du '**control plane**' tout est prêt, mais ce n'est pas suffisant !
- Au niveau du PE2 comment savoir quel prochain saut il faut choisir et vers quelle interface de sortie il faut transmettre le paquet !?
- L'entête IP ne donne pas assez d'informations.

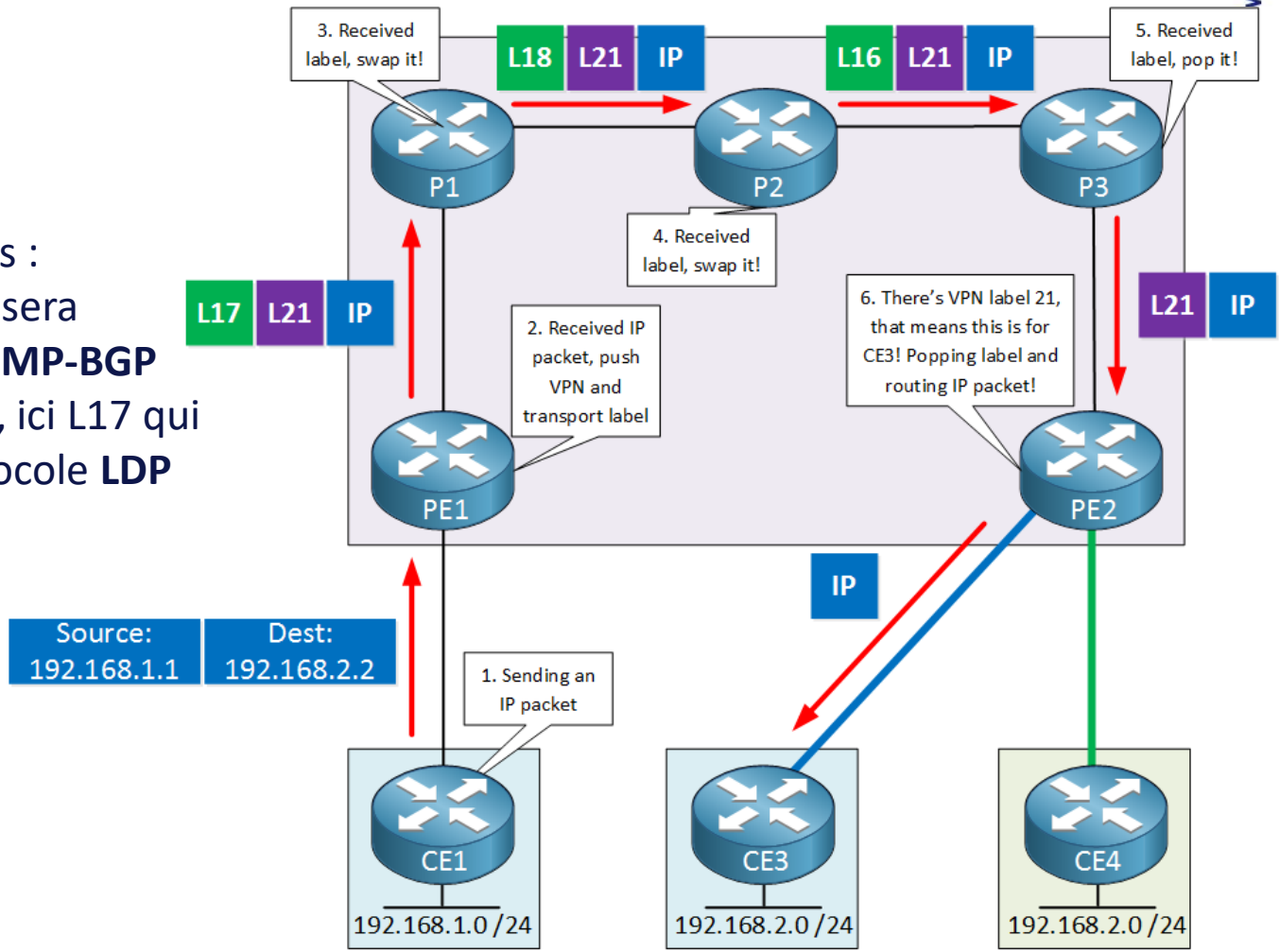


Les notions de **VRF**, **RD**, **RT** et **MP-BGP** opèrent au niveau du '**control plane**'. Au niveau '**data plane**' rien n'indique au niveau des **PE** vers où il faut regarder pour aiguiller le paquet vers la bonne sortie.

MPLS-VPN – Le label VPN

Le PE1 ajoute deux labels :

- Le **label VPN**, ici L21, sera annoncé par les MAJ **MP-BGP**
- Le **label de transport**, ici L17 qui est appris par le protocole **LDP**



➔ Solution : ajouter un 2e label (**VPN Label**) qui sera annoncé par les MAJ MP-BGP afin d'orienter le paquet vers le bon CE

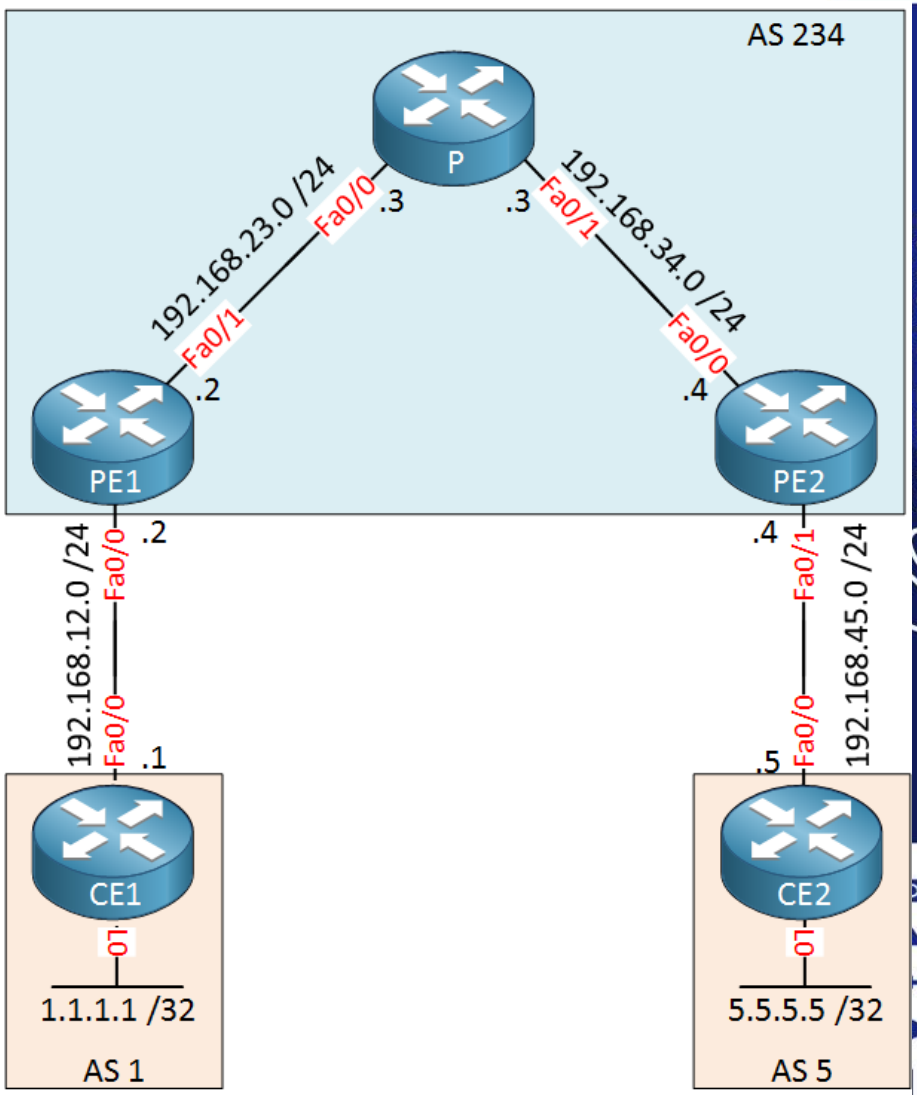
MPLS-VPN – Le label VPN

- La transmission des paquets IP provenant des CE sur le backbone **MPLS** emploie la notion de **label stacking**. Pour atteindre un site donné, le **PE** source encapsule **deux labels** : le premier sert à atteindre le PE de destination, tandis que le second détermine l'interface de sortie sur le PE, à laquelle est reliée le CE.
- Le second label est appris grâce aux updates **MP-BGP**.
- On remarque que seul le premier label a été modifié, le label VPN ayant été conservé intact pendant tout le cheminement sur le backbone.
- Pour afficher labels VPN appris par MP-BGP, on peut utiliser la commande Cisco « **show ip bgp vpnv4 vrf <vrf> tags** »

MPLS-VPN – configuration et vérification

Ici un FAI interconnecte les deux sites d'un client qui souhaite échanger les informations des réseaux présents sur un site avec le 2e site.
Pour cela il faut :

- Configurer un IGP et le LDP au sein du FAI.
- Création des VRF au niveau des PE
- Monter des sessions iBGP entre les PE.
- Configurer eBGP ou un IGP entre les PE et CE



MPLS-VPN – configuration et vérification

- Configuration : IGP and LDP

```
PE1(config)#interface loopback 0
PE1(config-if)#ip address 2.2.2.2 255.255.255.255
```

```
P(config)#interface loopback 0
P(config-if)#ip address 3.3.3.3 255.255.255.255
```

```
PE2(config)#interface loopback 0
PE2(config-if)#ip address 4.4.4.4 255.255.255.255
```

➔ On configure des loopback pour identifier les routeurs

```
PE1(config)#router ospf 1
PE1(config-router)#network 192.168.23.0 0.0.0.255 area 0
PE1(config-router)#network 2.2.2.2 0.0.0.0 area 0
```

```
P(config)#router ospf 1
P(config-router)#network 192.168.23.0 0.0.0.255 area 0
P(config-router)#network 192.168.34.0 0.0.0.255 area 0
P(config-router)#network 3.3.3.3 0.0.0.0 area 0
```

```
PE2(config)#router ospf 1
PE2(config-router)#network 192.168.34.0 0.0.0.255 area 0
PE2(config-router)#network 4.4.4.4 0.0.0.0 area 0
```

➔ Tous les préfixes au sein du FAI doivent être joignable avec un IGP

➔ On active mpls sur les PE et les P

```
PE1(config)#interface FastEthernet 0/1
PE1(config-if)#mpls ip
```

```
PE2(config)#interface FastEthernet 0/0
PE2(config-if)#mpls ip
```

```
P(config)#interface FastEthernet 0/0
P(config-if)#mpls ip
P(config)#interface FastEthernet 0/1
P(config-if)#mpls ip
```

MPLS-VPN – configuration et vérification

- Vérification :
  - show mpls interfaces
  - show mpls ldp neighbor
  - PE1#ping 4.4.4.4 source loopback 0
  - PE1#traceroute 4.4.4.4 source loopback 0

➔ Vérifier la connectivité entre les PE en utilisant les interfaces d'identification (loopback)

```
PE1#ping 4.4.4.4 source loopback 0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 4.4.4.4, timeout is 2 seconds:
Packet sent with a source address of 2.2.2.2
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

➔ Vérifier la commutation des labels entre les PE

```
PE1#traceroute 4.4.4.4 source loopback 0
Type escape sequence to abort.
Tracing the route to 4.4.4.4
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.23.3 [MPLS: Label 17 Exp 0] 0 msec 0 msec 4 msec
  2 192.168.34.4 0 msec 0 msec *
```

MPLS-VPN – configuration et vérification

- Création des VRF au niveau des PE

```
PE1(config)#ip vrf CUSTOMER
```

➔ Création de la VRF CUSTOMER

```
PE1(config-vrf)#rd 1:1
```

➔ Ajouter le RD pour rendre les routes uniques

```
PE1(config-vrf)#route-target both 1:1
```

➔ Spécifier ce qu'il va être importé et exporté dans cette VRF avec le RT

```
PE1#show run | begin vrf
ip vrf CUSTOMER
rd 1:1
route-target export 1:1
route-target import 1:1
```

```
PE1(config)#interface FastEthernet 0/0
PE1(config-if)#ip vrf forwarding CUSTOMER
 % Interface FastEthernet0/0 IPv4 disabled and address(es) removed due to enabling VRF CUSTOMER
```

➔ Affecter l'interface à la VRF

```
PE1(config-if)#ip address 192.168.12.2 255.255.255.0
```

➔ Reconfigurer l'adresse IP de l'interface

```
PE2(config)#ip vrf CUSTOMER
PE2(config-vrf)#rd 1:1
PE2(config-vrf)#route-target export 1:1
PE2(config-vrf)#route-target import 1:1
PE2(config)#interface FastEthernet 0/1
PE2(config-if)#ip vrf forwarding CUSTOMER
PE2(config-if)#ip address 192.168.45.4 255.255.255.0
```

➔ Même configuration pour PE2

MPLS-VPN – configuration et vérification

- Configuration du iBGP sur les PE

```
PE1(config)#router bgp 234
PE1(config-router)#neighbor 4.4.4.4 remote-as 234
PE1(config-router)#neighbor 4.4.4.4 update-source loopback 0
PE1(config-router)#address-family vpnv4
PE1(config-router-af)#neighbor 4.4.4.4 activate
```

➔ Afin d'échanger les routes VPNv4 il faut activer l'address-family correspondante

➔ Le routeur ajoute automatiquement la commande 'send-community extended' qui permet d'annoncer le RT

```
PE1#show run | section bgp
router bgp 234
 bgp log-neighbor-changes
 neighbor 4.4.4.4 remote-as 234
 neighbor 4.4.4.4 update-source Loopback0
 !
 address-family vpnv4
  neighbor 4.4.4.4 activate
  neighbor 4.4.4.4 send-community extended
 exit-address-family
```

```
PE2(config)#router bgp 234
PE2(config-router)#neighbor 2.2.2.2 remote-as 234
PE2(config-router)#neighbor 2.2.2.2 update-source loopback 0
PE2(config-router)#address-family vpnv4
PE2(config-router-af)#neighbor 2.2.2.2 activate
```

➔ Même configuration pour l'autre PE

MPLS-VPN – configuration et vérification

- Puisqu'on veut échanger juste les routes VPNv4 entre les PE, on peut faire en sorte que ces derniers n'échange pas les routes IPv4

```
PE1(config)#router bgp 234
PE1(config-router)#address-family ipv4
PE1(config-router-af)#no neighbor 4.4.4.4 activate
```

```
PE2(config)#router bgp 234
PE2(config-router)#address-family ipv4
PE2(config-router-af)#no neighbor 2.2.2.2 activate
```

MPLS-VPN – configuration et vérification

- La commande **show ip bgp summary** n'est pas utile dans notre cas puisqu'elle concerne que les préfixes IPv4 unicast
- **show bgp vpnv4 unicast all summary**

```
PE1#show bgp vpnv4 unicast all summary
BGP router identifier 2.2.2.2, local AS number 234
BGP table version is 1, main routing table version 1

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
4.4.4.4         4          234       7       7        1    0    0 00:03:03        0
```

```
PE2#show bgp vpnv4 unicast all summary
BGP router identifier 4.4.4.4, local AS number 234
BGP table version is 1, main routing table version 1

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
2.2.2.2         4          234       8       8        1    0    0 00:04:00        0
```

MPLS-VPN – configuration et vérification

- Mise en place du eBGP entre les PE et les CE

```
CE1(config)#interface loopback 0
CE1(config-if)#ip address 1.1.1.1 255.255.255.255

CE1(config)#router bgp 1
CE1(config-router)#neighbor 192.168.12.2 remote-as 234
CE1(config-router)#network 1.1.1.1 mask 255.255.255.255
```

```
CE2(config)#interface loopback 0
CE2(config-if)#ip address 5.5.5.5 255.255.255.255

CE2(config)#router bgp 5
CE2(config-router)#neighbor 192.168.45.4 remote-as 234
CE2(config-router)#network 5.5.5.5 mask 255.255.255.255
```

```
PE1(config)#router bgp 234
PE1(config-router)#address-family ipv4 vrf CUSTOMER
PE1(config-router-af)#neighbor 192.168.12.1 remote-as 1
```

```
PE2(config)#router bgp 234
PE2(config-router)#address-family ipv4 vrf CUSTOMER
PE2(config-router-af)#neighbor 192.168.45.5 remote-as 5
```

➔ Monter la session eBGP depuis la **VRF** qui corresponde au client

MPLS-VPN – configuration et vérification

```
PE1#show bgp vpnv4 unicast vrf CUSTOMER summary
BGP router identifier 2.2.2.2, local AS number 234
BGP table version is 2, main routing table version 2
1 network entries using 160 bytes of memory
1 path entries using 56 bytes of memory
2/1 BGP path/bestpath attribute entries using 272 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 536 total bytes of memory
BGP activity 1/0 prefixes, 1/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
192.168.12.1    4            1      13      12        2    0    0 00:07:31        1
```

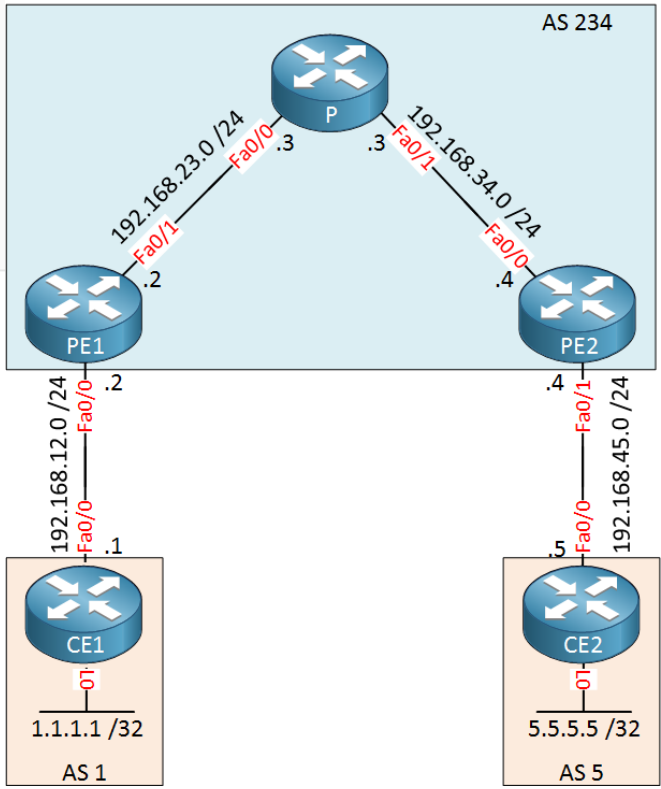
MPLS-VPN – configuration et vérification

```
PE2#show bgp vpnv4 unicast vrf CUSTOMER summary
BGP router identifier 4.4.4.4, local AS number 234
BGP table version is 4, main routing table version 4
2 network entries using 320 bytes of memory
2 path entries using 112 bytes of memory
3/2 BGP path/bestpath attribute entries using 408 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 912 total bytes of memory
BGP activity 2/0 prefixes, 2/0 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down
State/PfxRcd
192.168.45.5    4            5       5       5        4    0    0 00:00:31        1
```

MPLS-VPN – configuration et vérification

```
PE2#show bgp vpnv4 unicast vrf CUSTOMER
BGP table version is 4, local router ID is 4.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, x best-external,
f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf CUSTOMER)
*>i1.1.1.1/32       2.2.2.2                  0    100      0 1 i
*> 5.5.5.5/32       192.168.45.5             0             0 5 i
```

Avec les sessions **MP-BGP** pas besoin de changer le nexthop avec la commande **next-hop-self**, c'est fait automatiquement !



MPLS-VPN – configuration et vérification

```
CE1#show ip bgp
BGP table version is 3, local router ID is 1.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, x best-external,
f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 1.1.1.1/32       0.0.0.0                  0         32768 i
*> 5.5.5.5/32       192.168.12.2                           0 234 5 i
```

```
CE2#show ip bgp
BGP table version is 3, local router ID is 5.5.5.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, x best-external,
f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 1.1.1.1/32       192.168.45.4                           0 234 1 i
*> 5.5.5.5/32       0.0.0.0                  0         32768 i
```

➔ Vérifier que les CE s'échange bien les routes

MPLS-VPN – configuration et vérification

```
CE1#traceroute 5.5.5.5 source loopback 0
Type escape sequence to abort.
Tracing the route to 5.5.5.5
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.12.2 0 msec 0 msec 4 msec
  2 192.168.23.3 [MPLS: Labels 17/19 Exp 0] 0 msec 0 msec 4 msec
  3 192.168.45.4 [MPLS: Label 19 Exp 0] 0 msec 0 msec 4 msec
  4 192.168.45.5 0 msec 0 msec *
```

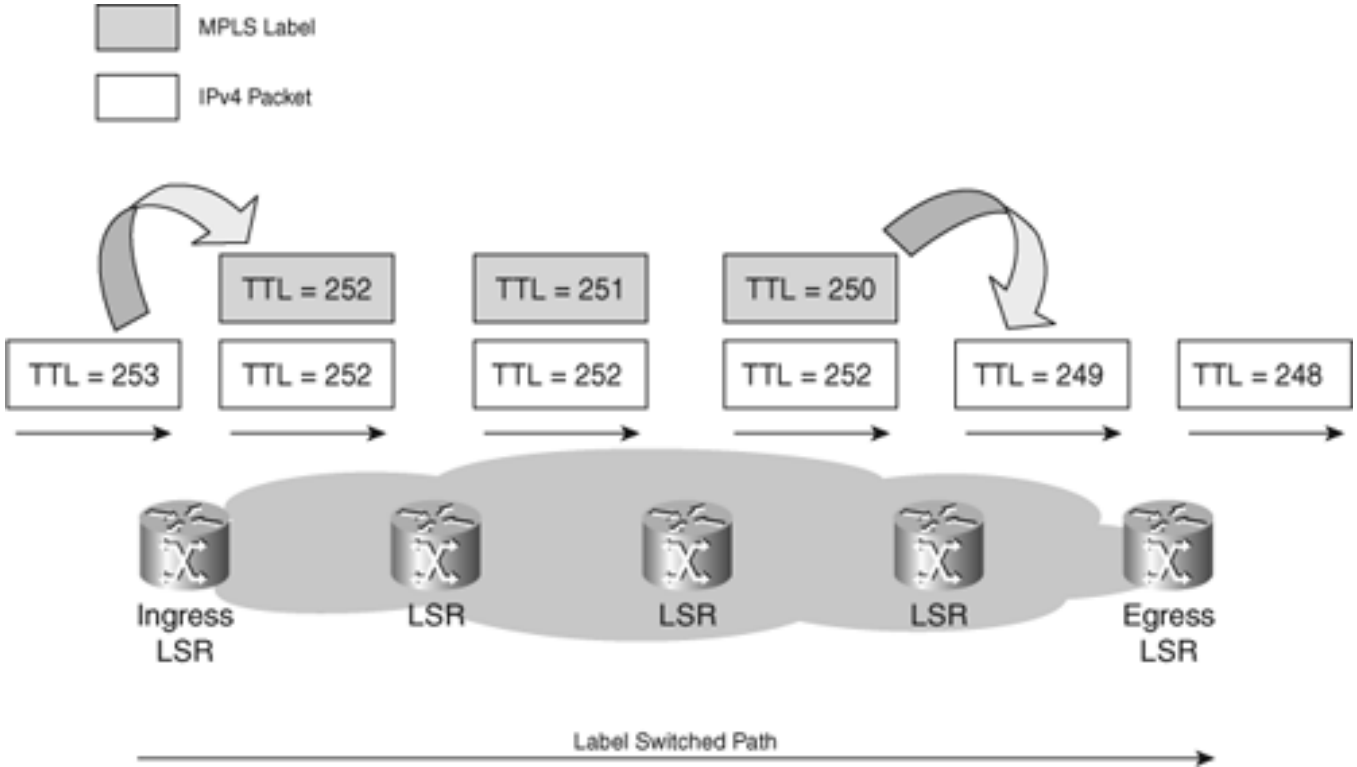
➔ Vérifier la mise en place des labels

➔ 19 est le label VPN et 17 le label de transport

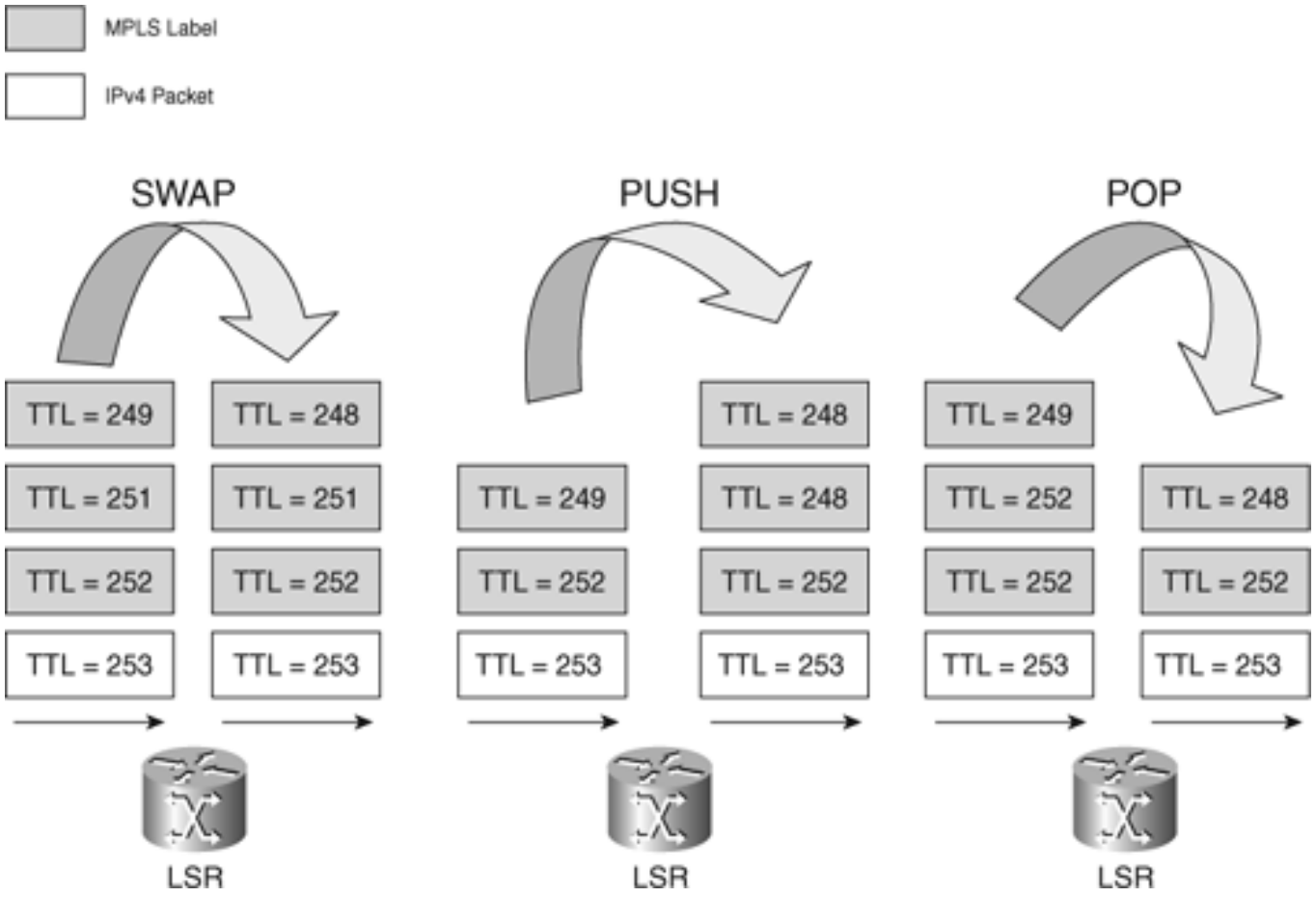
Comportement du TTL dans les réseaux MPLS

La façon dont le TTL se propage de l'en-tête IP à la pile des labels et vice versa peut être résumée avec les règles suivantes :

- Lorsqu'un paquet IP reçoit un label (ou plus) pour la première fois, la valeur **TTL -1** est copiée de l'en-tête IP vers les champs TTL de(s) label(s) ajouté(s)
- Lorsque l'opération **swap** est a opéré au niveau d'un LSR, la valeur **TTL -1** du label supérieure entrant est copié vers le label supérieure sortant
- Lorsque l'opération **pop** est a opéré, la valeur **TTL -1** du label supérieure entrant est copié sur le label supérieure nouvellement exposé.



Comportement du TTL dans les réseaux MPLS



- Dans le cas de l'opération **pop,** si le label nouvellement exposé a une valeur TTL inférieure au label supérieure entrant, alors le TTL sortant est le **TTL -1** du label nouvellement exposé

Forwarding Equivalence Class (FEC)

- Une **FEC** est la représentation d'un ensemble de paquets qui sont transmis de la même manière, qui suivent le même chemin au sein du réseau et ayant la même priorité.
- MPLS constitue les FEC selon de nombreux critères : adresse destination, adresse source, application, QoS, etc.
- Quand un paquet IP arrive à un ***ingress LER***, il sera associé à une ***FEC***. Puis, exactement comme dans le cas d'un routage IP classique, un protocole de routage sera mis en œuvre pour découvrir un chemin jusqu'à ***l'egress LER***.
- À la différence d'un routage IP classique cette opération ne se réalise qu'une seule fois. Ensuite, tous les paquets appartenant à la même ***FEC*** seront acheminés suivant ce chemin appelé ***LSP***