La Table de Routage

La Structure d'une Table de Routage

La structure d'une table de routage semble évidente

Connaître le fonctionnement d'une table de routage peut être très utile lors d'un dépannage ou pour vérifier une configuration

```
Router# show ip route

     172.16.0.0/24 is subnetted, 4 subnets
S       172.16.4.0 is directly connected, Serial0/0/1
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Serial0/0/1
C    192.168.1.0/24 is directly connected, Serial0/0/1
S    192.168.100.0/24 is directly connected, Serial0/0/1
```

Quelques exemples

Différents types de route selon leurs **sources**

- Réseaux directement connectés
- Routes statiques
- Protocoles de routage dynamique

La source des routes n'affecte pas la structure de la table de routage

```
Router# show ip route

     172.16.0.0/24 is subnetted, 4 subnets
S       172.16.4.0 is directly connected, Serial0/0/1
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Serial0/0/1
C    192.168.1.0/24 is directly connected, Serial0/0/1
S    192.168.100.0/24 is directly connected, Serial0/0/1
```

Quelques exemples

L'hiérarchie des tables de routage IOS a été initialement **implémentée avec un schéma de routage classful**

La table de routage incorpore les deux mécanismes, classful et classless

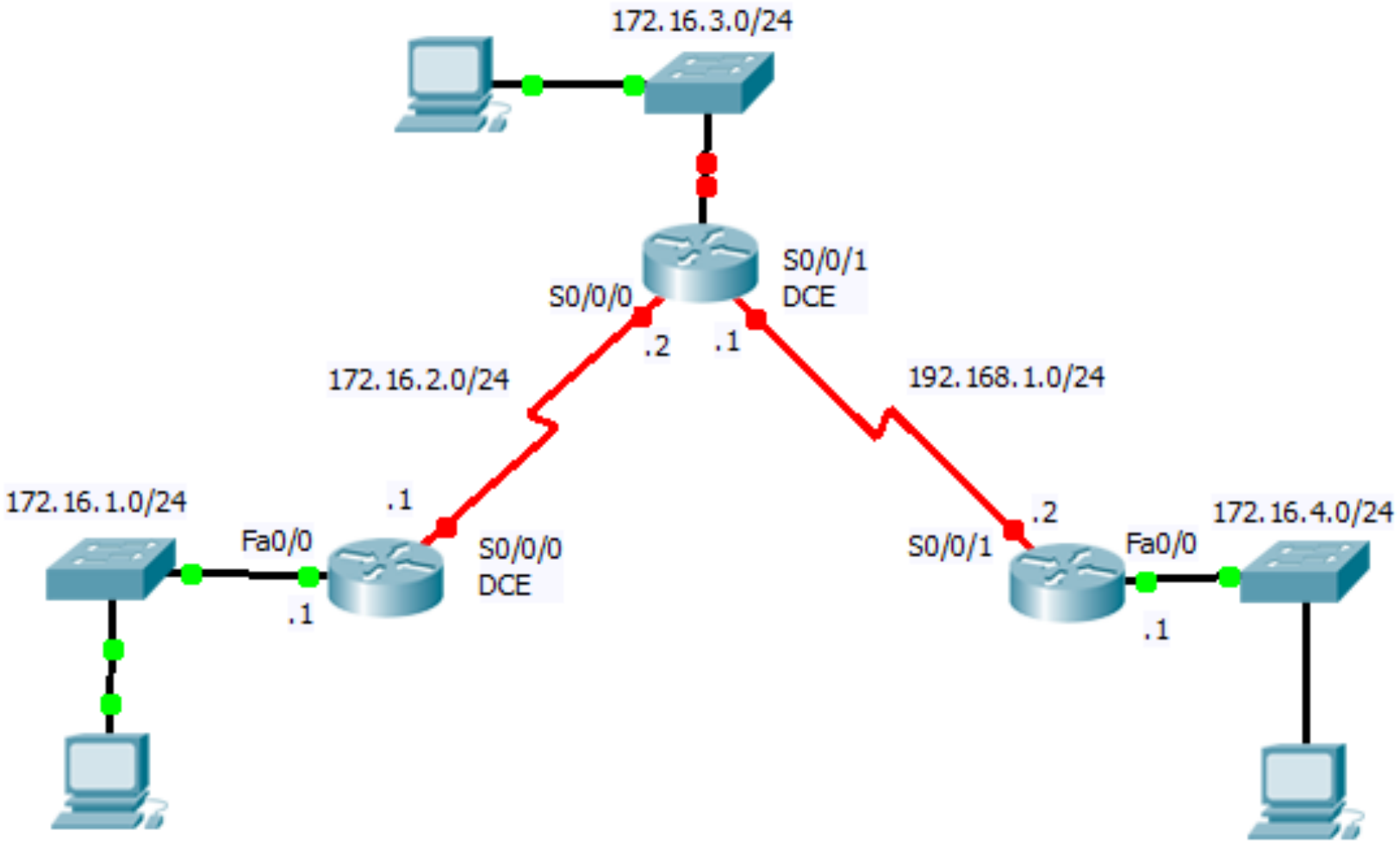
```
Router# show ip route

     172.16.0.0/24 is subnetted, 4 subnets
S       172.16.4.0 is directly connected, Serial0/0/1
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Serial0/0/1
C    192.168.1.0/24 is directly connected, Serial0/0/1
S    192.168.100.0/24 is directly connected, Serial0/0/1
```

Adresse Classful

Sous-réseaux 172.16.0.0

Topologie pour l'exemple à venir



Configurations des interfaces de R1 et R3

Pour l'instant on ne configure pas les interfaces de R2

```
R1(config)# interface FastEthernet0/0
R1(config-if)# ip address 172.16.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface Serial0/0/0
R1(config-if)# ip address 172.16.2.1 255.255.255.0
R1(config-if)# clock rate 64000
R1(config-if)# no shutdown
```

```
R3(config)# interface FastEthernet0/0
R3(config-if)# ip address 172.16.4.1 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface Serial0/0/1
R3(config-if)# ip address 192.168.1.2 255.255.255.0
R3(config-if)# no shutdown
```

Routes de Niveau 1

Dès qu'on entre « **no shutdown** », la sortie de **debug ip routing** indique que cette route a été rajoutée à la table de routage

```
R2# debug ip routing
IP routing debugging is on
R2# conf t
R2(config)# interface serial 0/0/1
R2(config-if)# ip address 192.168.1.1 255.255.255.0
R2(config-if)# clock rate 64000
R2(config-if)# no shutdown
R2(config-if)#
00:11:06: %LINK-3-UPDOWN: Interface Serial0/0/1, changed state to up
R2(config-if)#
RT: add 192.168.1.0/24 via 0.0.0.0, connected metric [0/0]
RT: interface Serial0/0/1 added to routing table
R2(config-if)# end
R2# undebug all
All possible debugging has been turned off
```

Routes de Niveau 1

La table de routage est une structure hiérarchique qui est censée accélérer la recherche d'une route lors de la transmission de paquets

Dans cette hiérarchie, on y trouve plusieurs niveaux

Pour ***simplicité***, nous étudions **toutes les routes en seulement deux niveaux**

```
RT: add 192.168.1.0/24 via 0.0.0.0, connected metric [0/0]
RT: interface Serial0/0/1 added to routing table
R2(config-if)# end

R2# show ip route
<output omitted>

C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Routes de Niveau 1

Une ***route de niveau 1*** est une route avec un **masque de sous-réseau égal ou inférieur au masque par défaut de la classe de l'adresse**

```
R2# show ip route
<output omitted>

C       192.168.1.0/24 is directly connected, Serial0/0/1
```

Celle-ci est une route niveau 1

L'adresse appartient à la classe C

Le masque est égal ou inférieur à celui de la classe C

Routes de Niveau 1

Une **route de niveau 1** peut être utilisée pour :

Route par défaut:

Une route par défaut est une route statique avec l'adresse 0.0.0.0/0

Route agrégée (supernet) :

Une route avec un masque inférieur à celui de la classe du réseau

Route pour un réseau :

Une route avec un masque égal à celui de la classe du réseau

Routes de Niveau 1

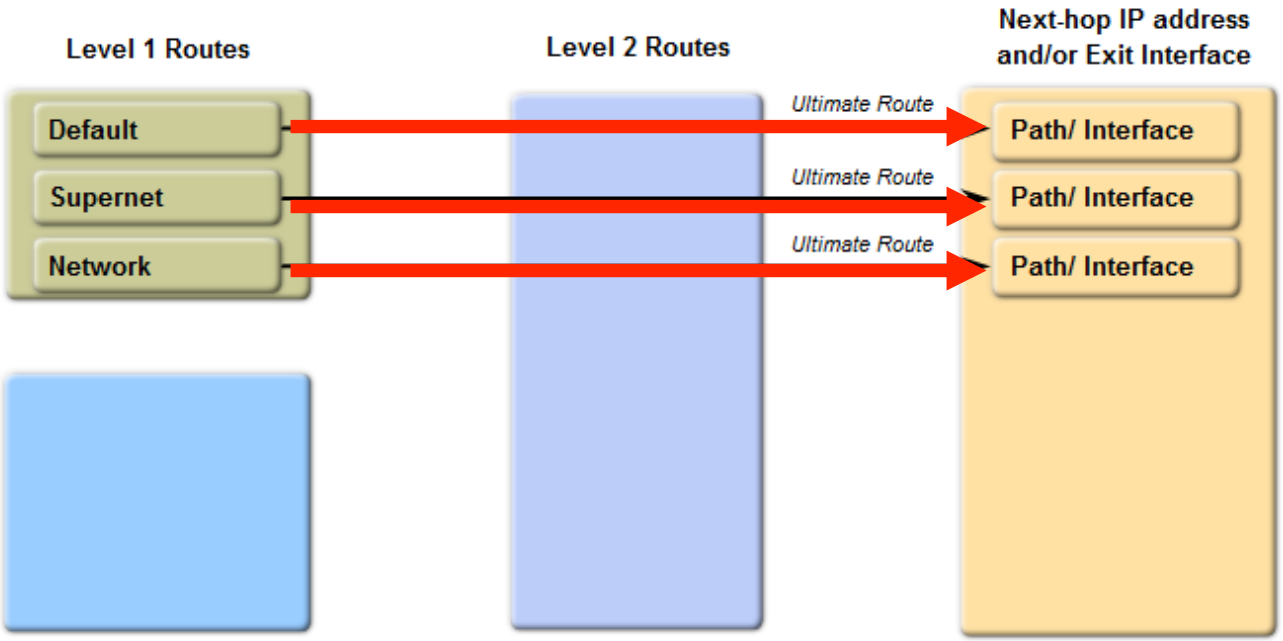
La **route de niveau 1** 192.168.1.0/24 peut être aussi appelée « route définitive » (*ultimate route*)

Une **route définitive** est une route qui inclut **une ou toutes** caractéristiques :

Une adresse de prochain saut (next-hop)

Une interface de sortie

```
C       192.168.1.0/24 is directly connected, Serial0/0/1
```



Routes de Niveau 1

Comment on sait que la route directement connectée **192.168.1.0/24** :

Est une route de niveau 1 ?

Le masque du réseau est égal à celui de la classe

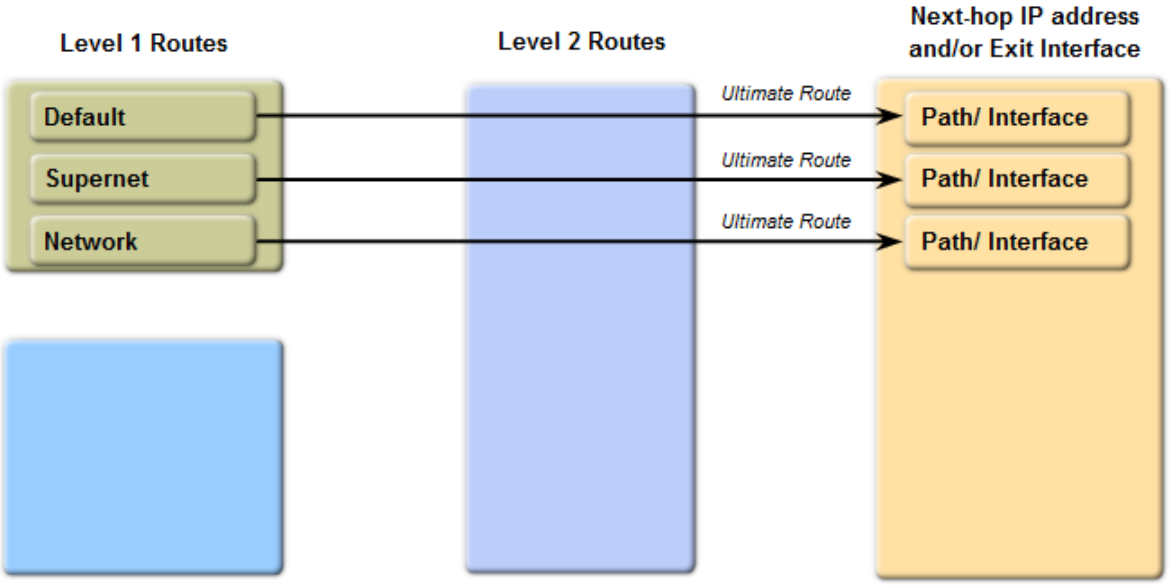
Est une route définitive ?

Contient l'interface de sortie Serial 0/0/1.

Classe C

Interface de Sortie

```
C       192.168.1.0/24 is directly connected, Serial0/0/1
```



Routes Parent et Enfant : Réseaux Classful

Un autre type de route **niveau 1** est la **route parent**

Lorsque le réseau 172.16.3.0 a été rajouté, une entrée supplémentaire a été créée

172.16.0.0

Que peut-on dire des routes *parent de niveau 1 ?*

Aucune adresse de prochain saut ou interface de sortie sont indiquées

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ip address 172.16.3.1 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# end
R2# show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile,
<text omitted>
Gateway of last resort is not set
     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Route Parent de Niveau 1

Routes Parent et Enfant : Réseaux Classful

Une **route parent** est une entête qui :

Indique la présence de routes niveau 2, aussi connues comme ***routes enfant***

Une **route parent de niveau 1** est créée automatiquement lorsqu'on rajoute un sous-réseau à la table de routage

Une **route parent** est créée à chaque fois qu'une route avec un masque plus grand que celui de la classe est entrée dans la table de routage

```
R2# show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile,
<text omitted>
Gateway of last resort is not set
     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Route Parent niveau 1

Route enfant niveau 2

Routes Parent et Enfant : Réseaux Classful

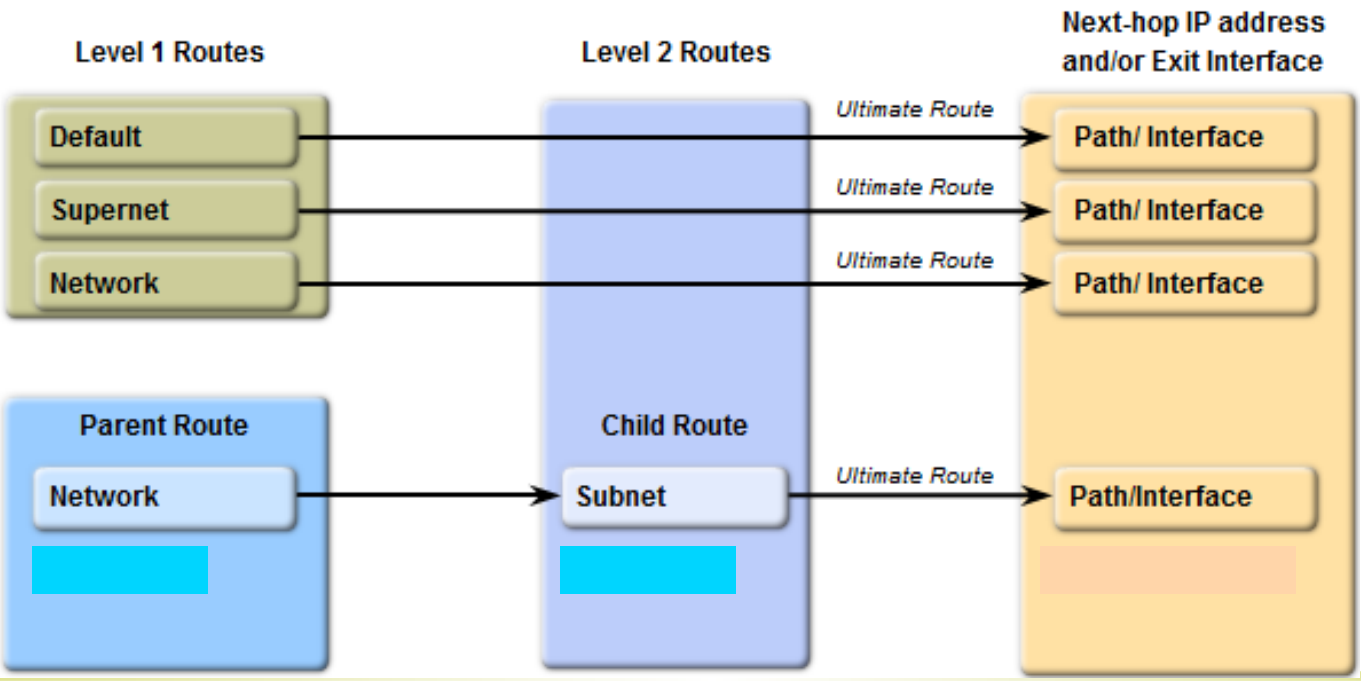
Comme une route niveau 1, la **source** d'une route niveau 2 peut être :

- Un réseau directement connecté
- Une route statique
- Une route dynamique (avec un protocole classful ou classless)

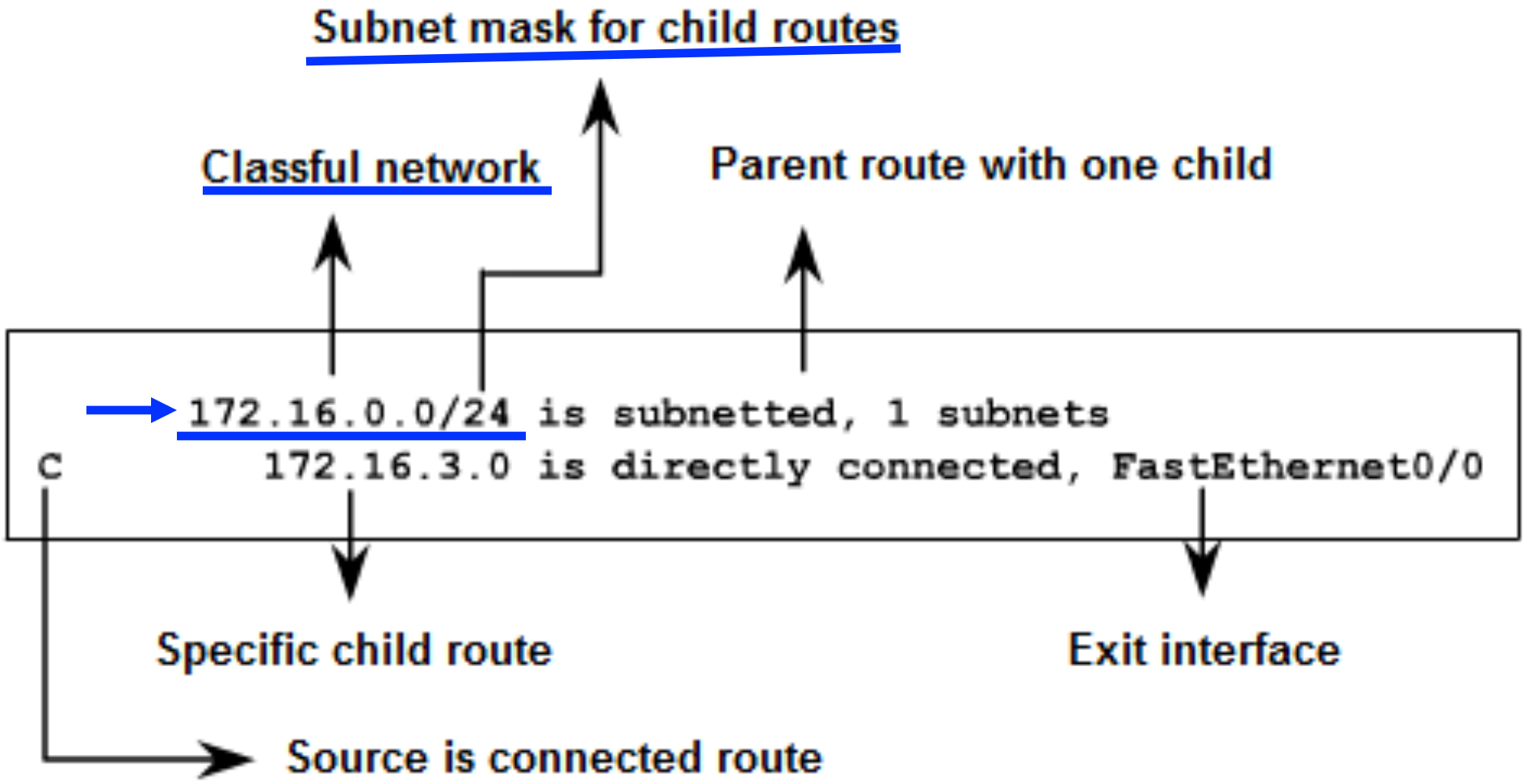
```
     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.3.0 is directly connected, FastEthernet0/0
```

Route parent niveau 1

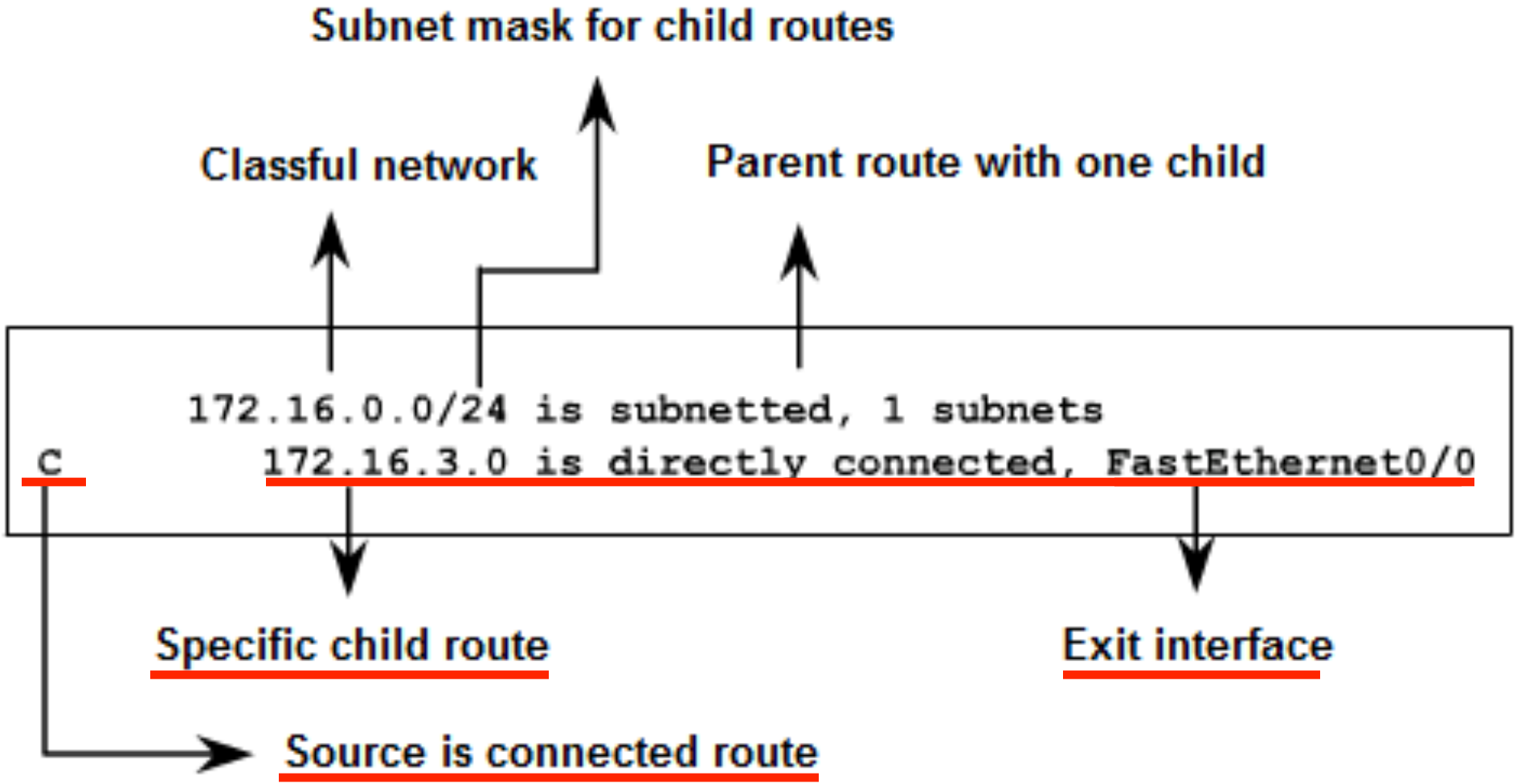
Route enfant niveau 2



Routes Parent et Enfant : Réseaux Classful



Route Enfant



Rajouter une autre route enfant

Nous allons configure une autre interface

Attention à l'endroit où cette route sera insérée

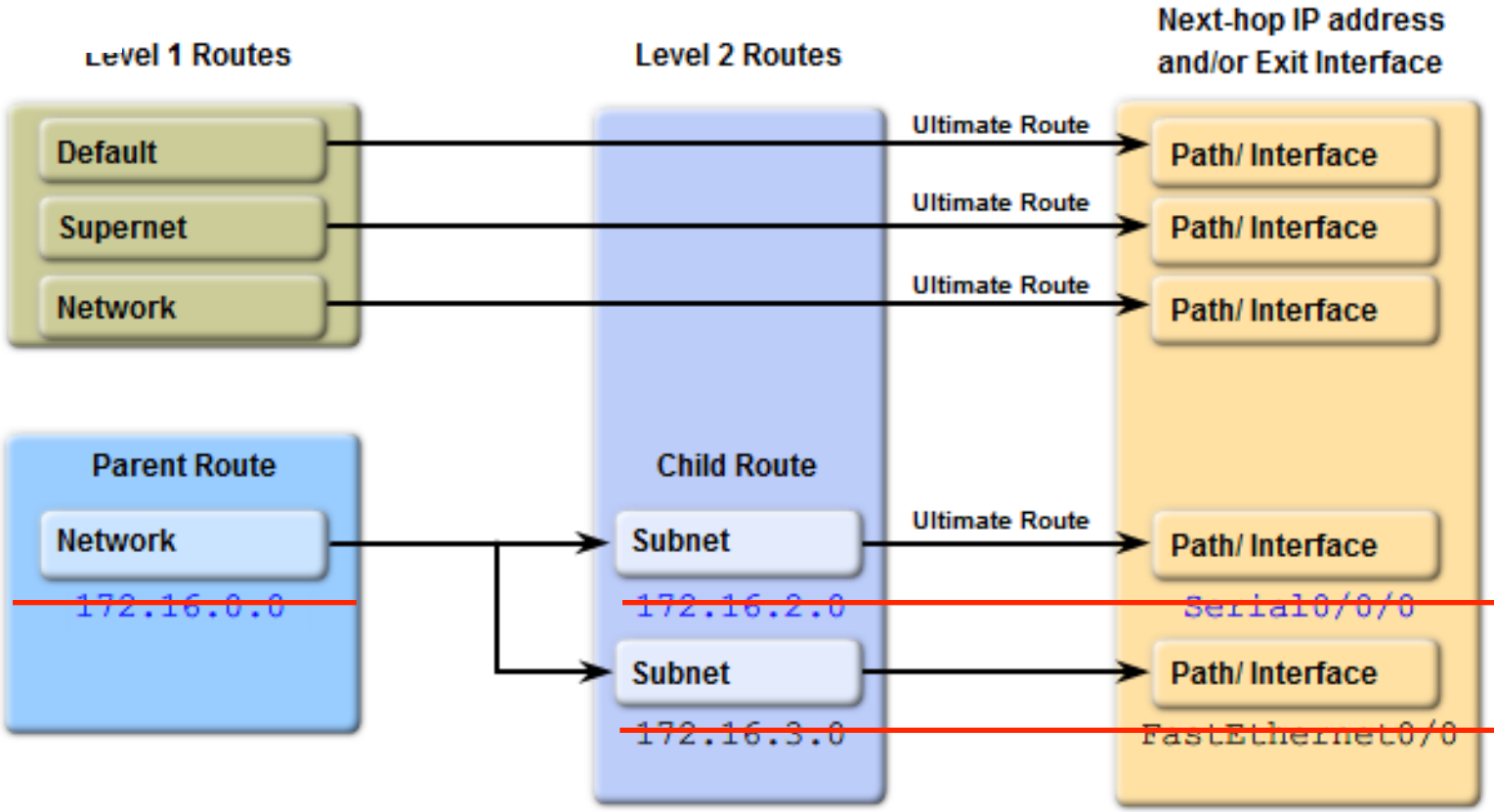
Comme toutes les routes enfant ont le même masque, la route parent gardera le masque /24 mais indique désormais deux routes

```
R2(config)# interface serial 0/0/0
R2(config-if)# ip address 172.16.2.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# end
R2# show ip route
     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Suppression de toutes les routes enfant

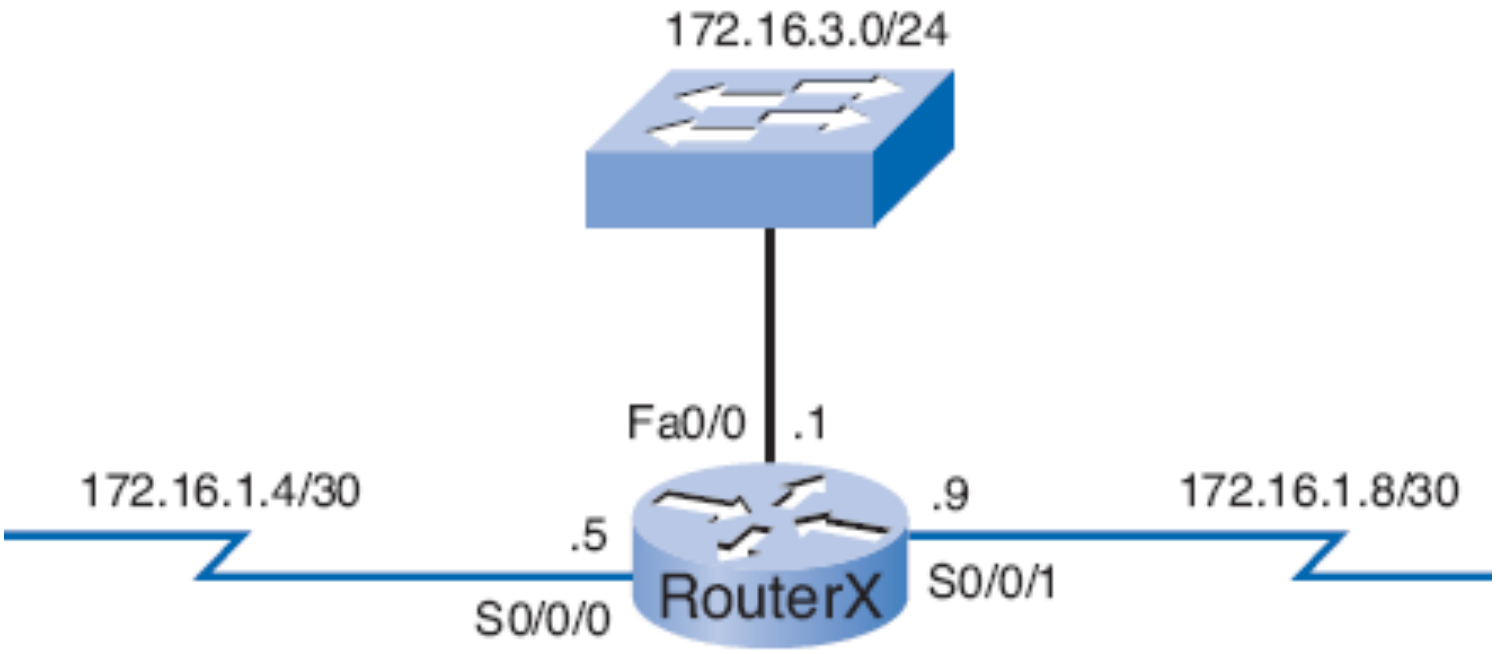
Si toutes les routes enfant sont effacées alors la route parent le sera aussi

```
172.16.0.0/24 is subnetted, 1 subnets
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
```



Routes Parent et Enfant : Réseaux Classless

Pour illustrer ce cas, on change temporairement de topologie



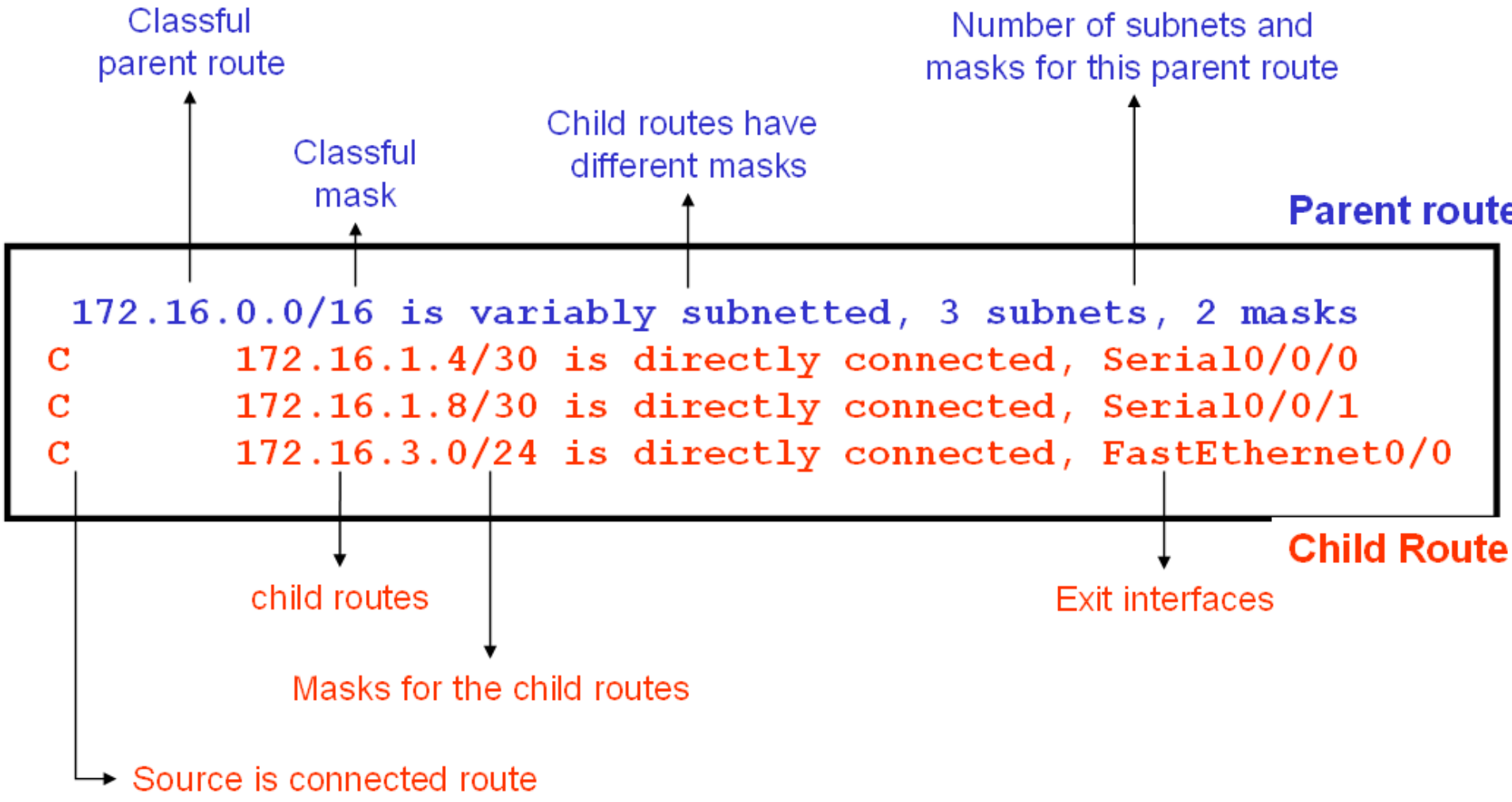
Routes Parent et Enfant : Réseaux Classless

On observe que les routes enfant ne partagent pas le même masque, comme c'était le cas dans les réseaux classful

```
RouterX# show ip route

      172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
C        172.16.1.4/30 is directly connected, Serial0/0/0
C        172.16.1.8/30 is directly connected, Serial0/0/1
C        172.16.3.0/24 is directly connected, FastEthernet0/0
```

Routes Parent et Enfant : Réseaux Classless



Fonctionnement de la table de routage

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Paquet IP

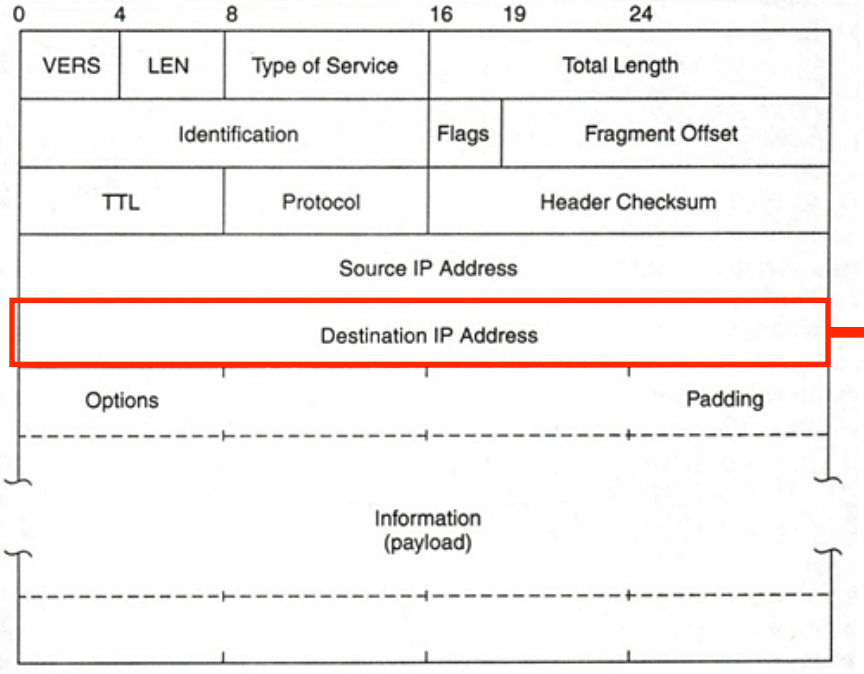


Table de Routage

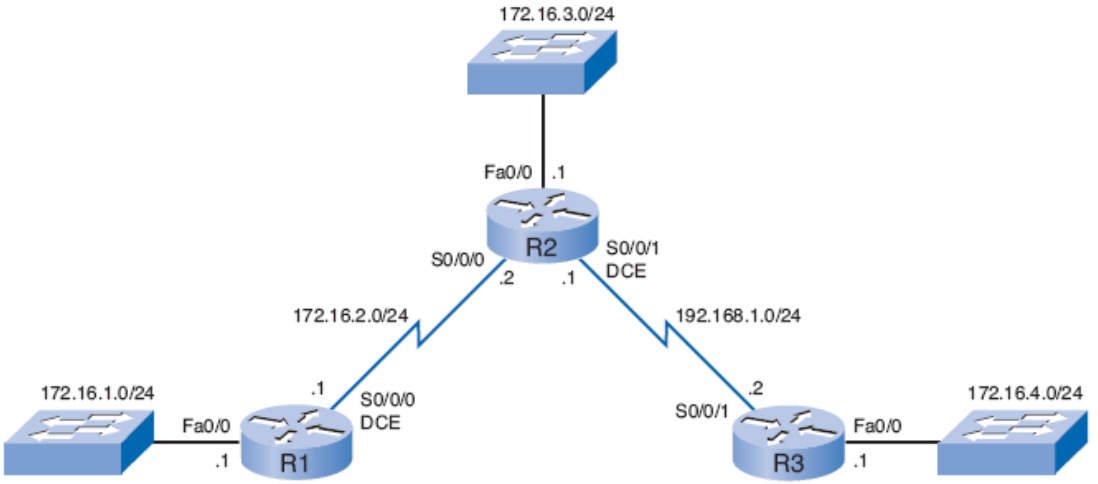
```
Router# show ip route

     172.16.0.0/24 is subnetted, 4 subnets
S       172.16.4.0 is directly connected, Serial0/0/1
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:08, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Serial0/0/1
C    192.168.1.0/24 is directly connected, Serial0/0/1
S    192.168.100.0/24 is directly connected, Serial0/0/1
```

Trouver la **meilleure correspondance**

Configuration de RIPv1

Pour illustration, nous choisissons un protocole classful qui ne supportera pas le réseau non-contigu




```
R1(config)# router rip
R1(config-router)# network 172.16.0.0
```

```
R2(config)# router rip
R2(config-router)# network 172.16.0.0
R2(config-router)# network 192.168.1.0
```

```
R3(config)# router rip
R3(config-router)# network 172.16.0.0
R3(config-router)# network 192.168.1.0
```

Étapes du processus de routage

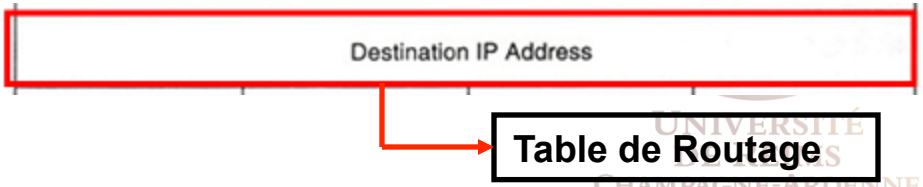
Ni R1 ni R2 ont des routes pour 172.16.4.0.

```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

```
R2# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:07, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

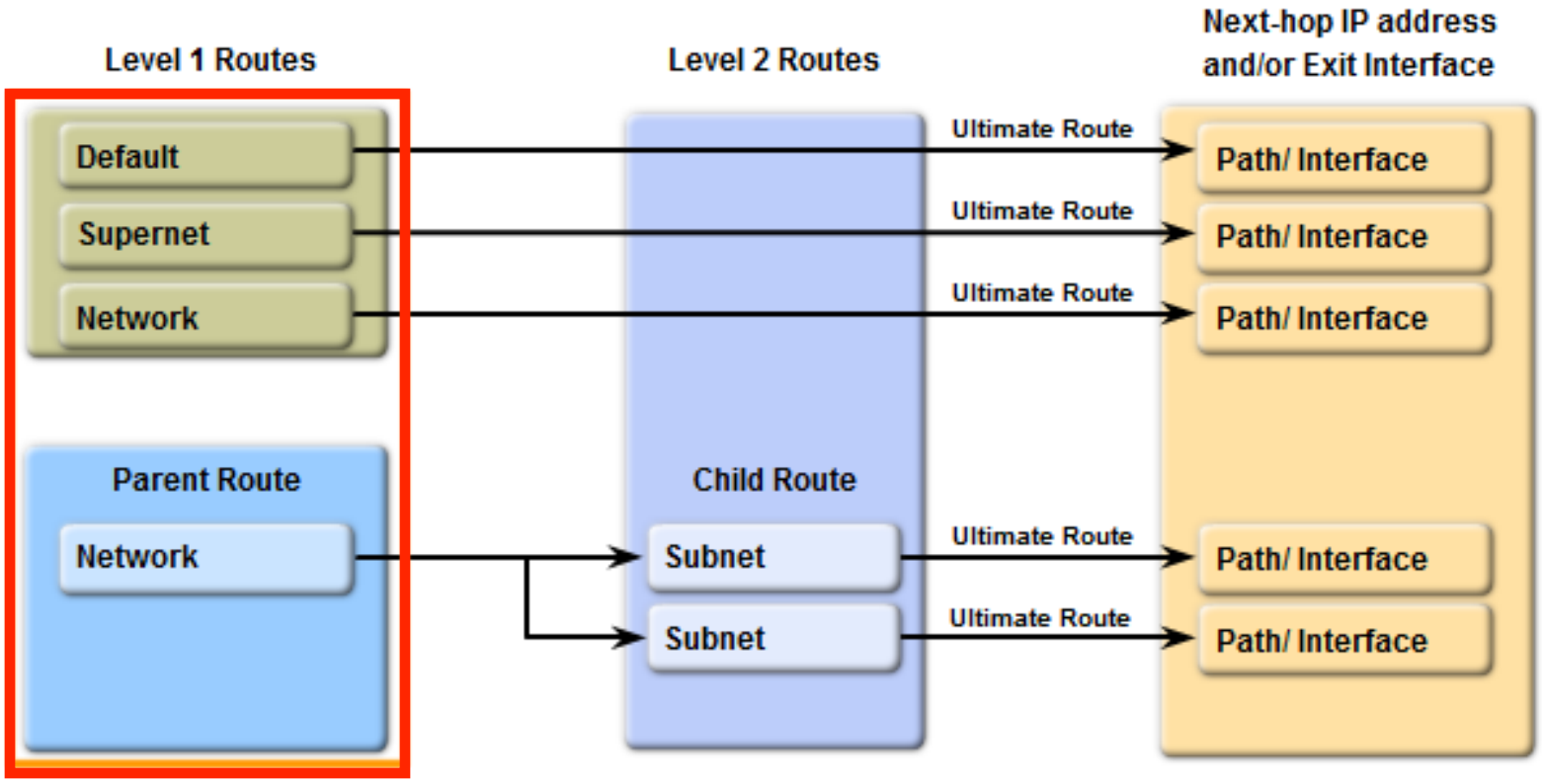
```
R3# show ip route
     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.4.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Le processus du routage

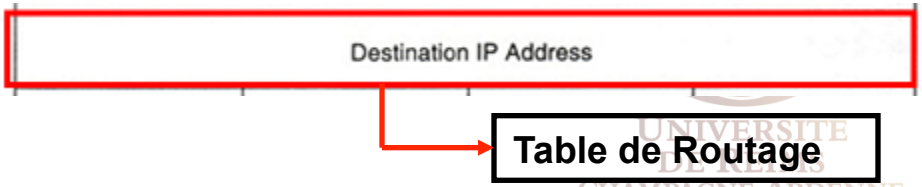


Étape 1.

Le routeur examine les routes de niveau 1

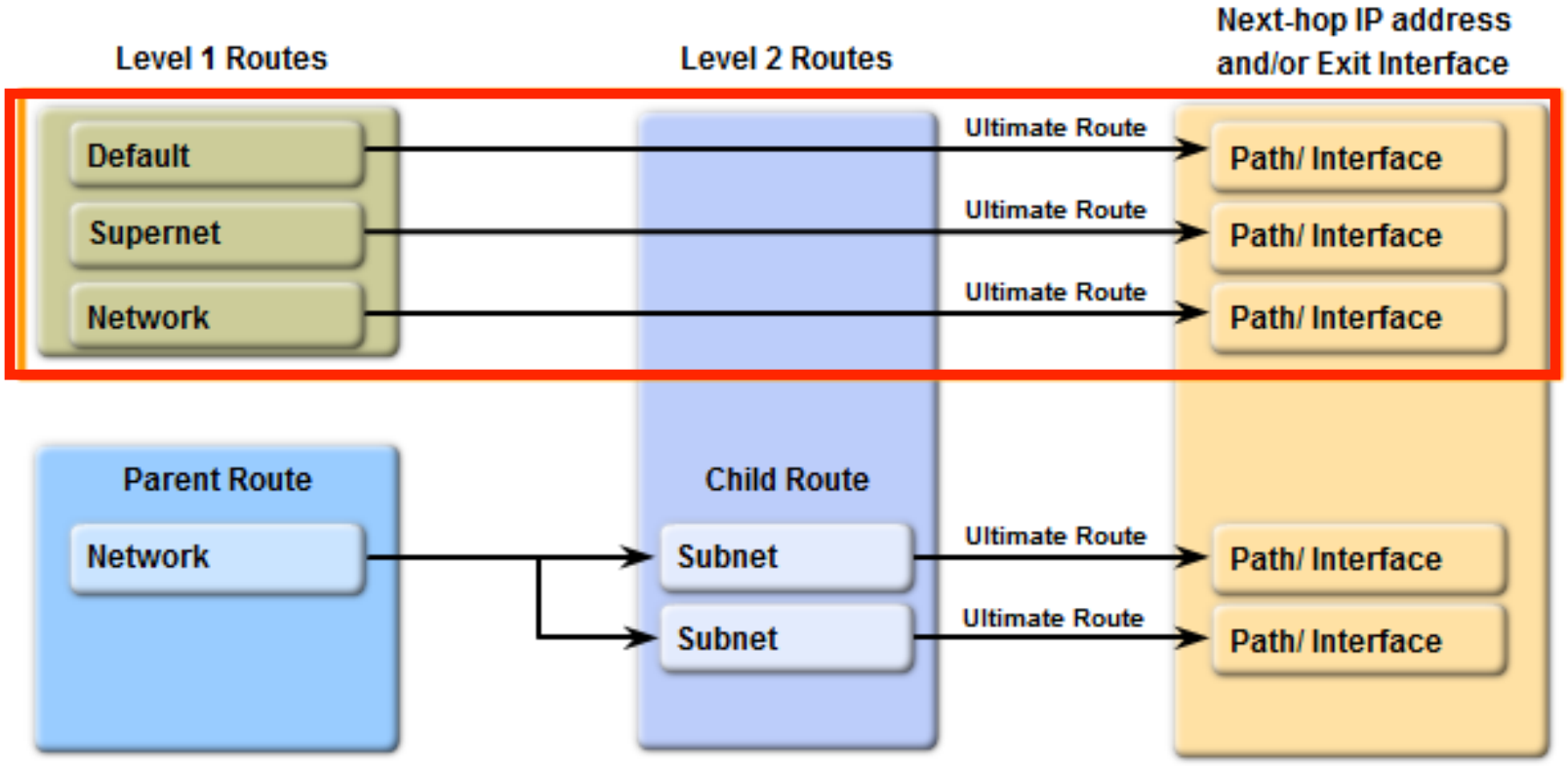


Le processus du routage

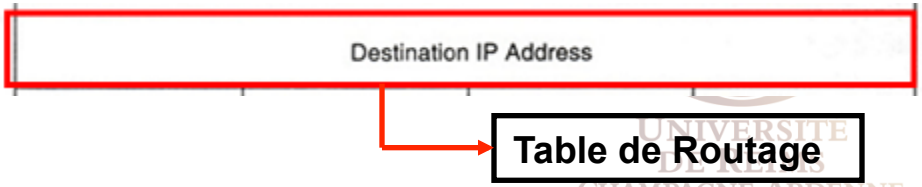


Étape 1a.

Si la meilleure correspondance est une route définitive de niveau 1 – un réseau classful, un supernet ou une route par défaut – cette route sera choisie



Le processus du routage

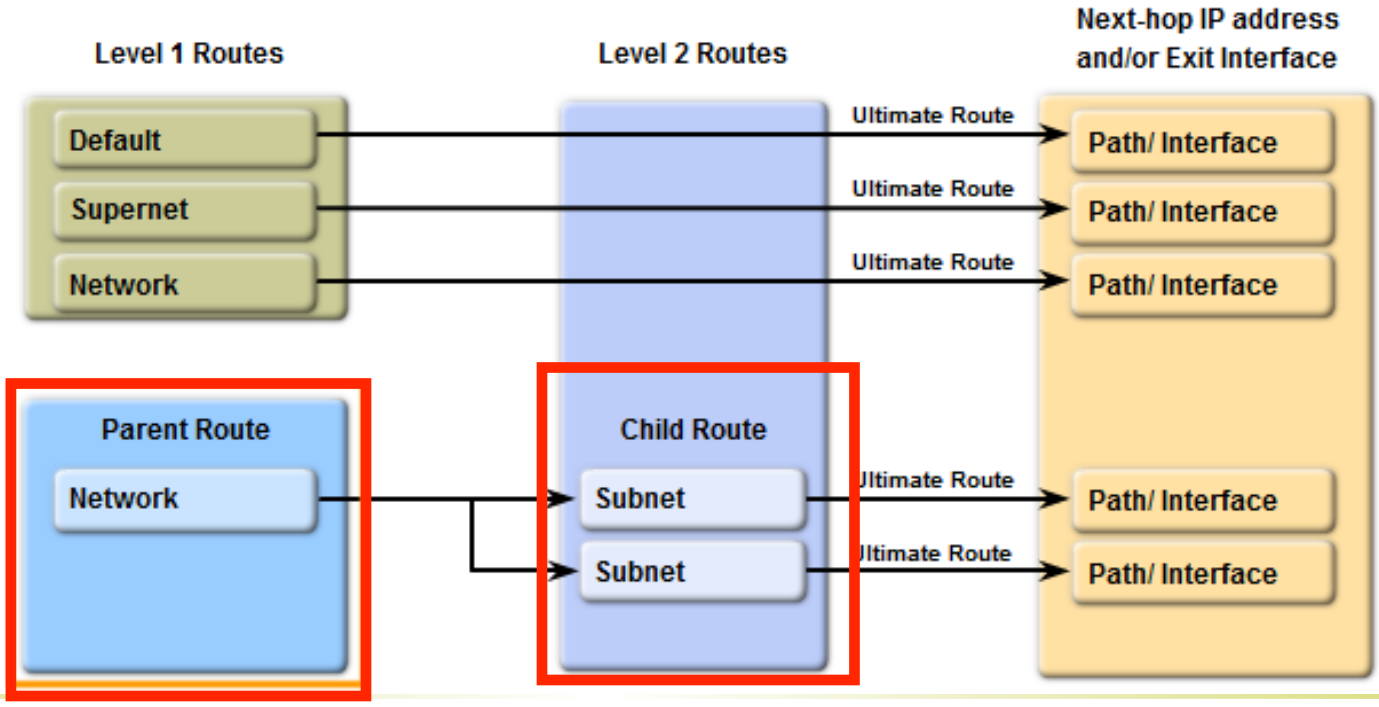


Étape 1b.

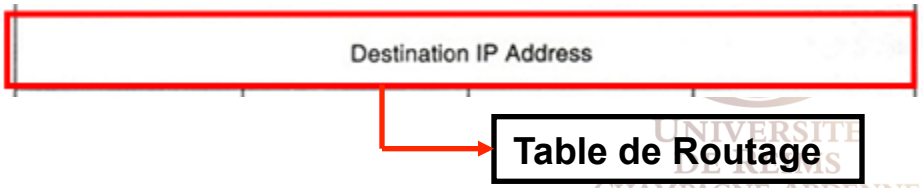
Si la meilleure correspondance est une route parent de niveau 1, avancer à l'étape 2

Étape 2.

Le route examine les routes enfant (les sous-réseaux) de cette route parent à la recherche d'une correspondance

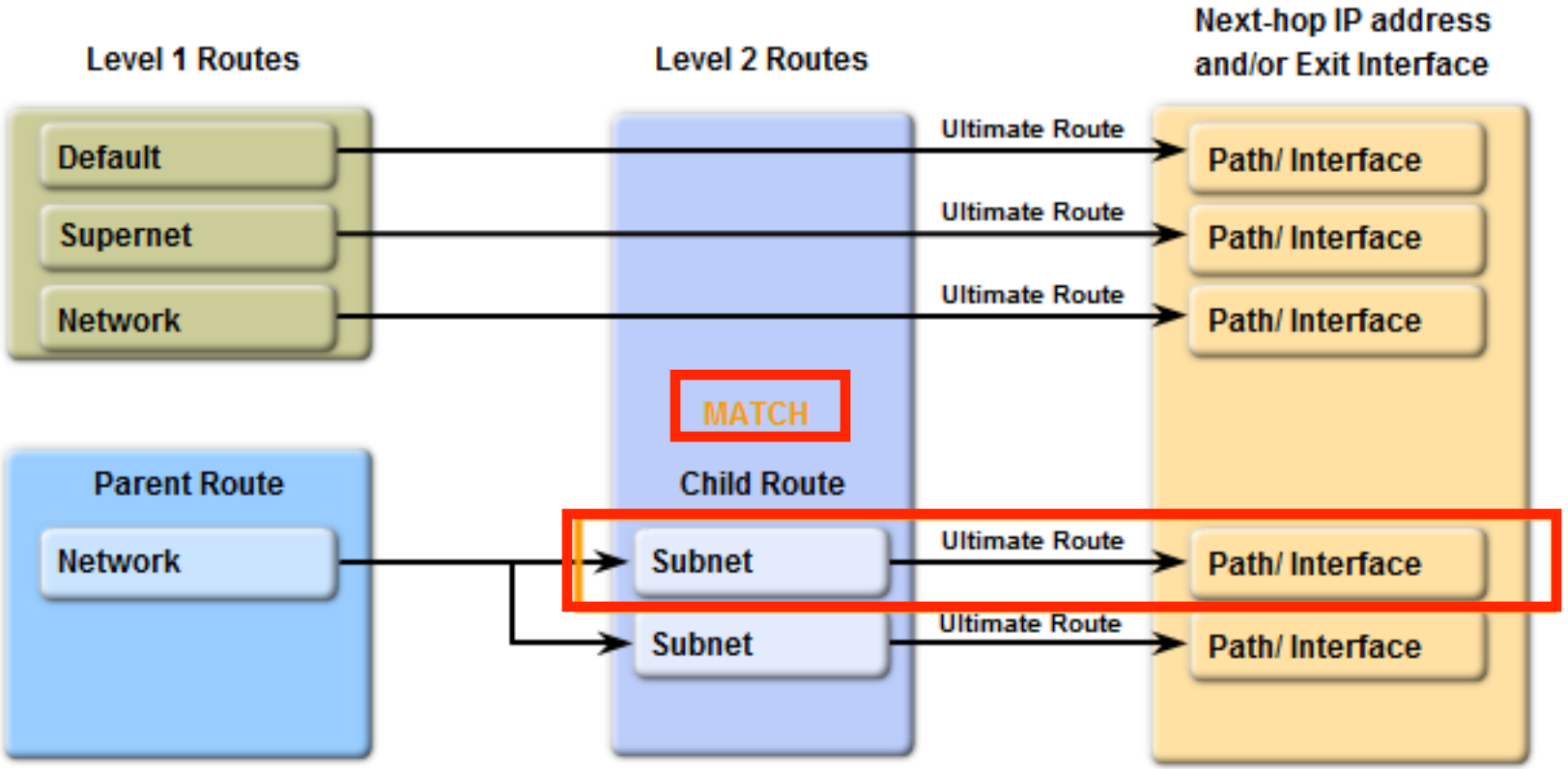


Le processus du routage

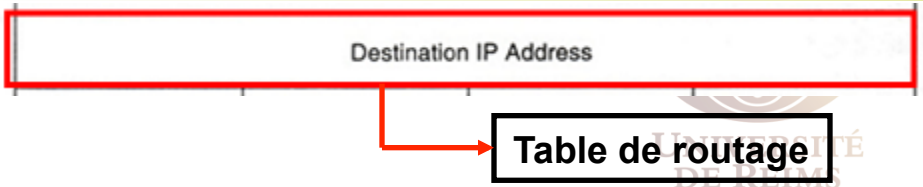


Étape 2a.

Si une correspondance avec une route niveau 2 est trouvée, celle-ci sera utilisée

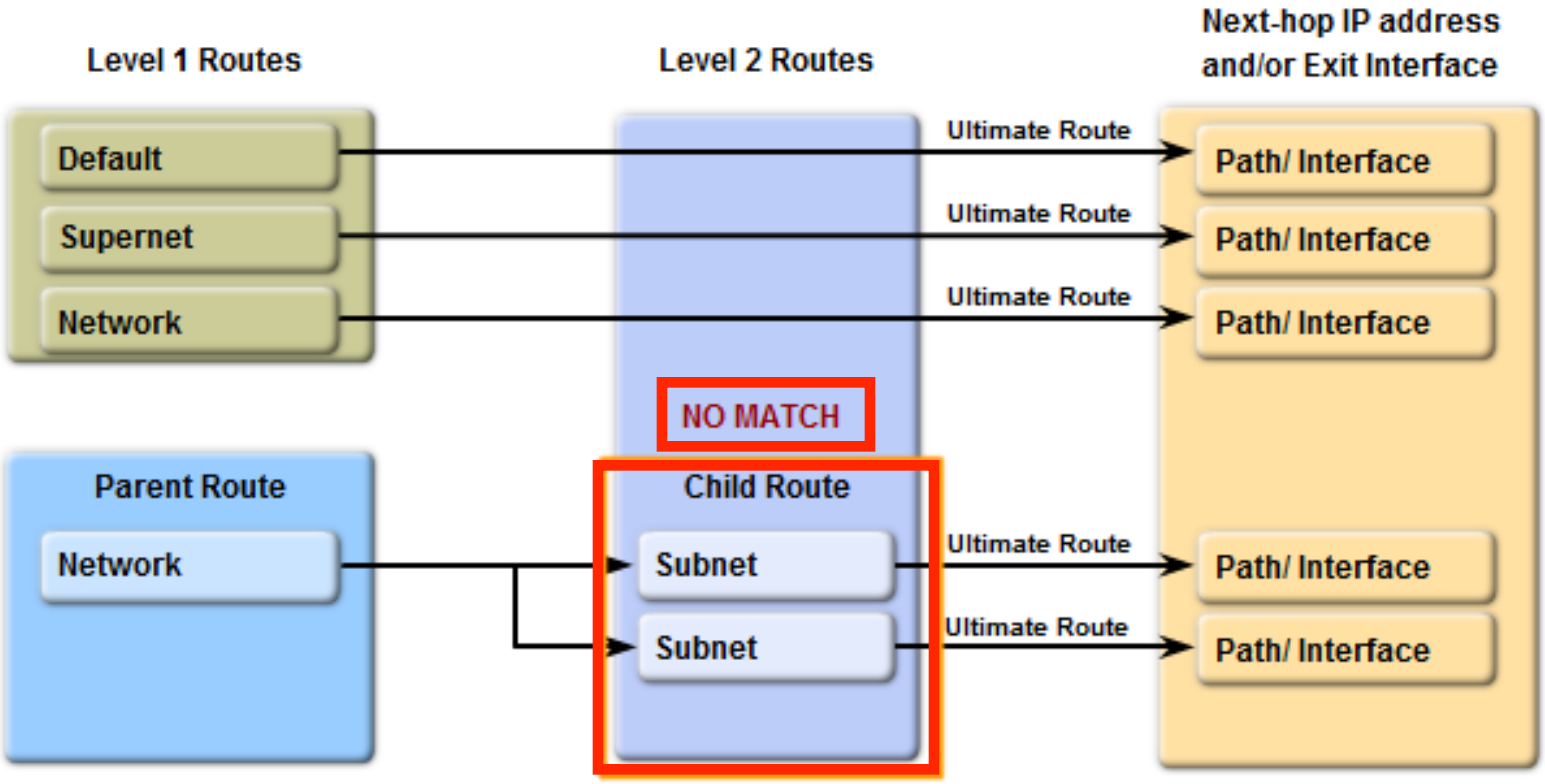


Le processus du routage

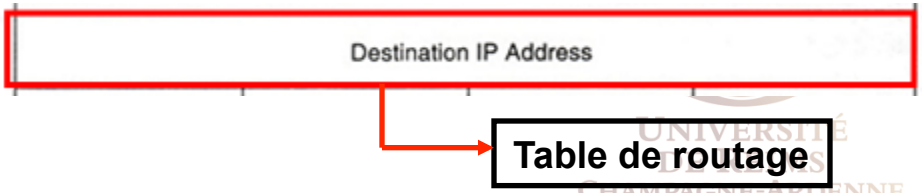


Étape 2b.

Si aucune correspondance avec une route niveau 2 est trouvée, procéder à l'étape 3



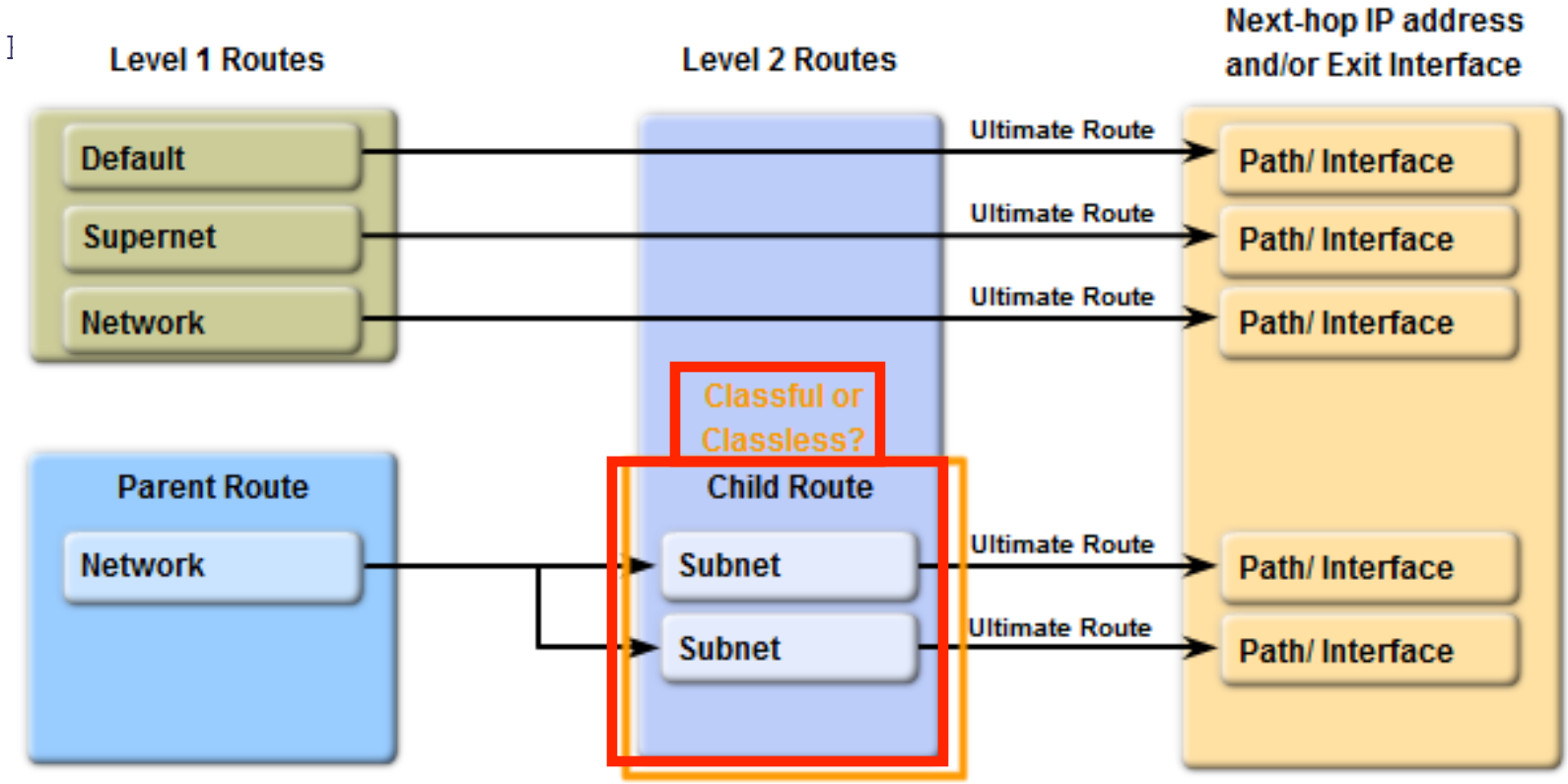
Le processus du routage



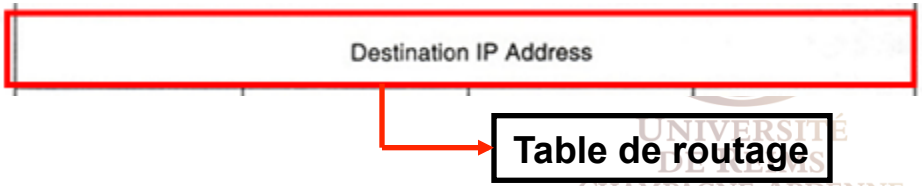
Étape 3.

Quel est le comportement du routeur ? (classful ou classless)

```
Router(config)# no ip classless
```



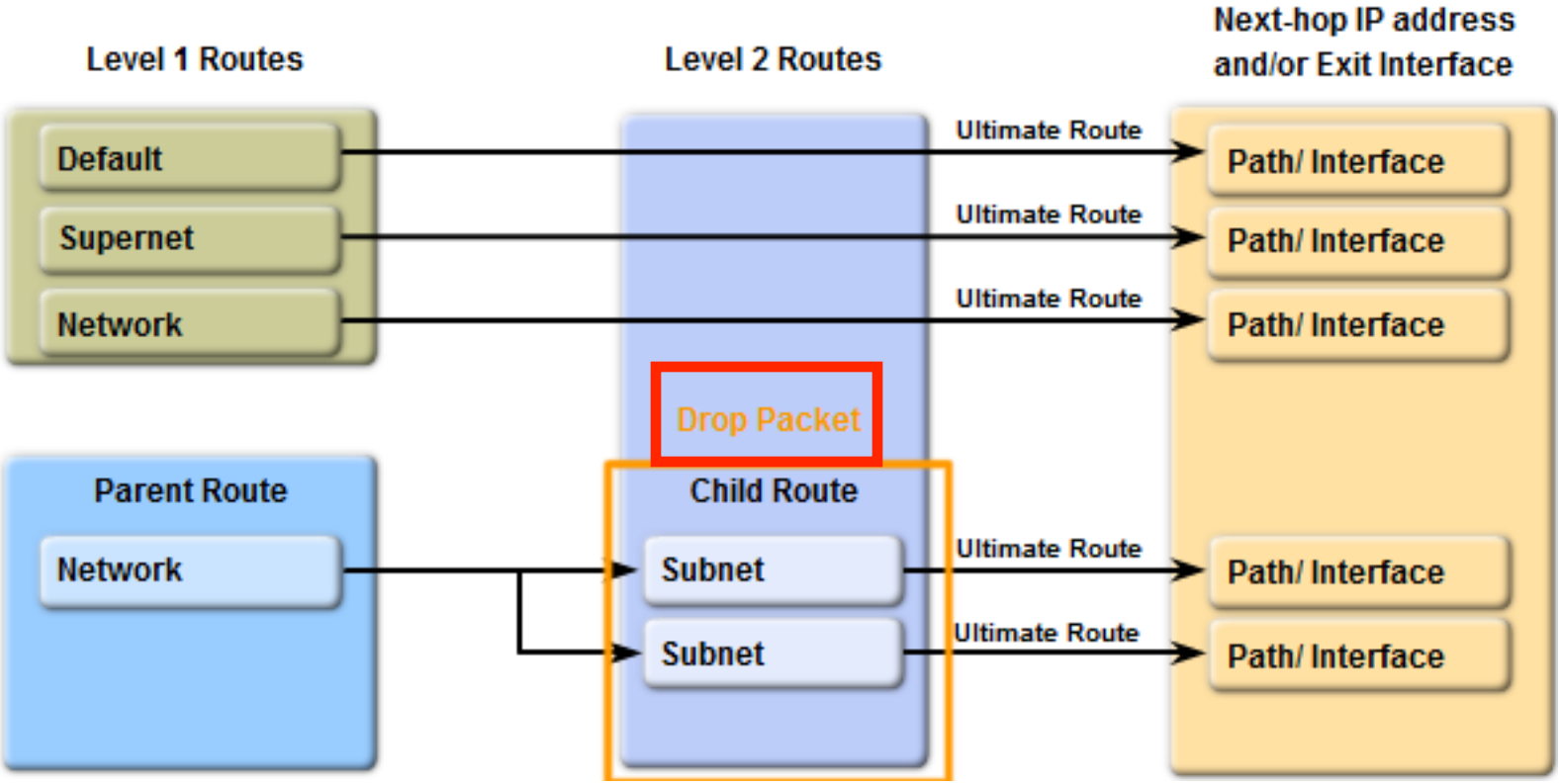
Le processus du routage



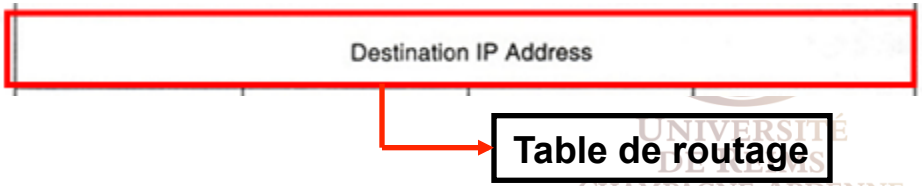
Étape 3a.

Si le **comportement classful** est actif, terminer le processus et jeter le paquet

```
Router(config)# no ip classless
```



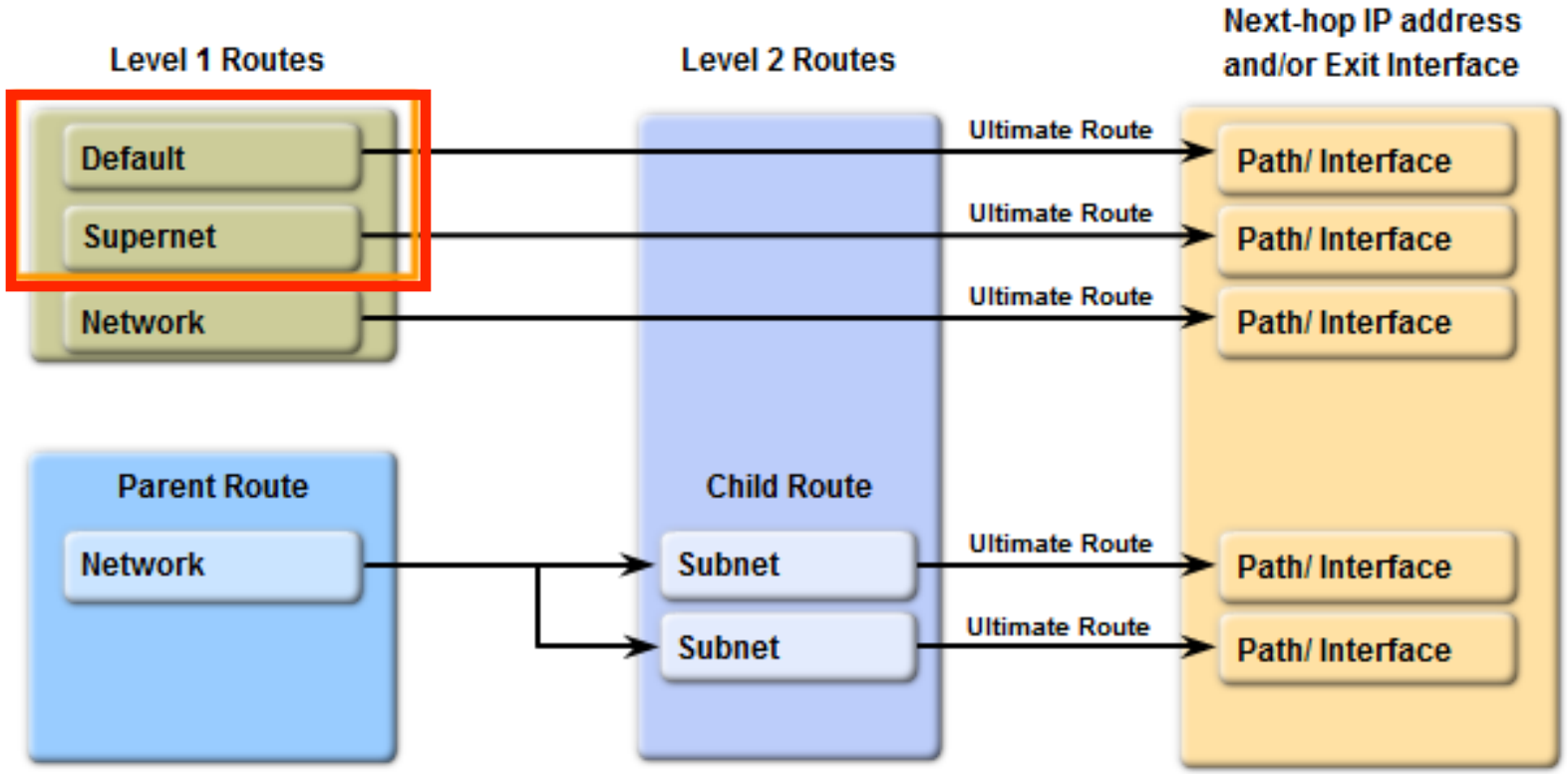
Le processus du routage



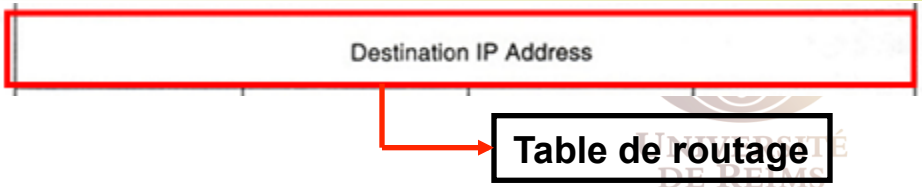
Étape 3b.

Si le comportement ***classless*** est actif, continuer à chercher un supernet dans les routes niveau 1, voir une route par défaut si disponible

```
Router(config)# ip classless
```

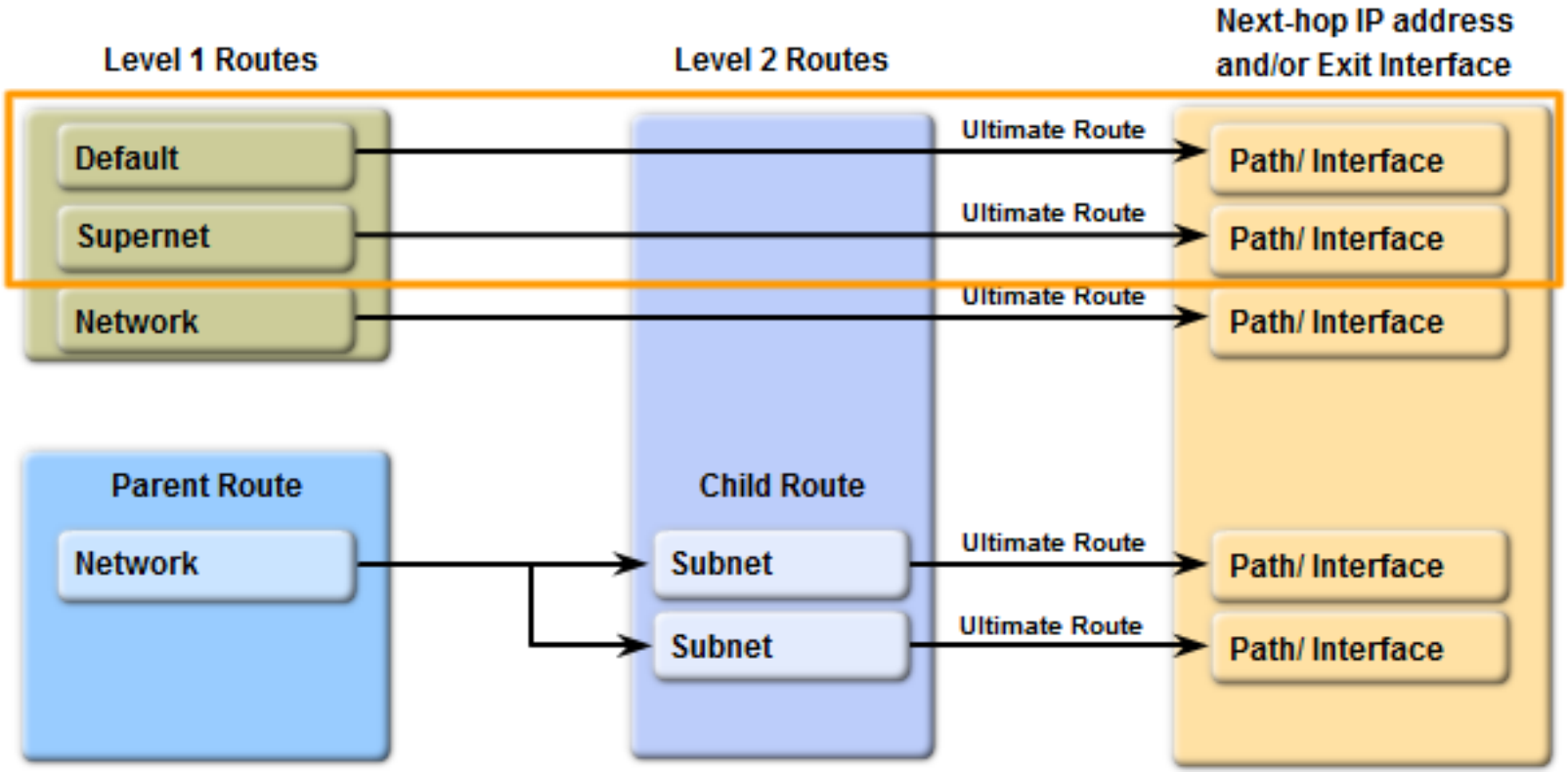


Le processus du routage



Étape 4. Correspondance – transmission du paquet

Étape 5. Aucune correspondance – jeter le paquet



Note : Modifier le comportement IOS

Si un protocole classless tel que OSPF est en utilisation, les autres routes seront examinées même si aucune correspondance est trouvée au niveau des routes enfant

Ceci arrive indépendamment d'utiliser **ip classless** ou **no ip classless**

Tourner un protocole classless est une manière de dépasser le **no ip classless**

Exemple : classful

```
RTC(config)# no ip classless
RTC#show ip route
Gateway of last resort is 192.168.1.1 to network 0.0.0.0
R    192.168.10.0/24 [120/1] via 192.168.1.1, 00:00:02, Serial1
C    192.168.1.0/24 is directly connected, Serial1
C    192.168.2.0/24 is directly connected, Ethernet0
     172.30.0.0/24 is subnetted, 3 subnets
C       172.30.1.0 is directly connected, Loopback1
C       172.30.2.0 is directly connected, Loopback2
C       172.30.3.0 is directly connected, Loopback3
S*   0.0.0.0/0 [1/0] via 192.168.1.1
```

```
RTC#ping 172.30.4.1
```

Destination du paquet : 172.30.4.1

```
Sending 5, 100-byte ICMP Echoes to 172.30.4.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Exemple : classless

```
RTC(config)# ip classless
RTC# show ip route
Gateway of last resort is 192.168.1.1 to network 0.0.0.0
R    192.168.10.0/24 [120/1] via 192.168.1.1, 00:00:02, Serial1
C    192.168.1.0/24 is directly connected, Serial1
C    192.168.2.0/24 is directly connected, Ethernet0
     172.30.0.0/24 is subnetted, 3 subnets
C       172.30.1.0 is directly connected, Loopback1
C       172.30.2.0 is directly connected, Loopback2
C       172.30.3.0 is directly connected, Loopback3
S*   0.0.0.0/0 [1/0] via 192.168.1.1
```

```
RTC#ping 172.30.4.1
```

Destination du paquet : 172.30.4.1

```
Sending 5, 100-byte ICMP Echoes to 172.30.4.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5),round-trip min/avg/max=56/57/60 ms
```

Correspondance la plus grande

Le **masque** d'une route dans la table de routage est utilisé pour déterminer le nombre minimum de bits qui doivent correspondre

IP Packet Destination	172.16.0.10	10101100.00010000.00000000.00001010
Route 1	172.16.0.0/12	10101100.00010000.00000000.00000000
Route 2	172.16.0.0/18	10101100.00010000.00000000.00000000
Route 3	172.16.0.0/26	10101100.00010000.00000000.00000000

Longest Match to IP Packet Destination

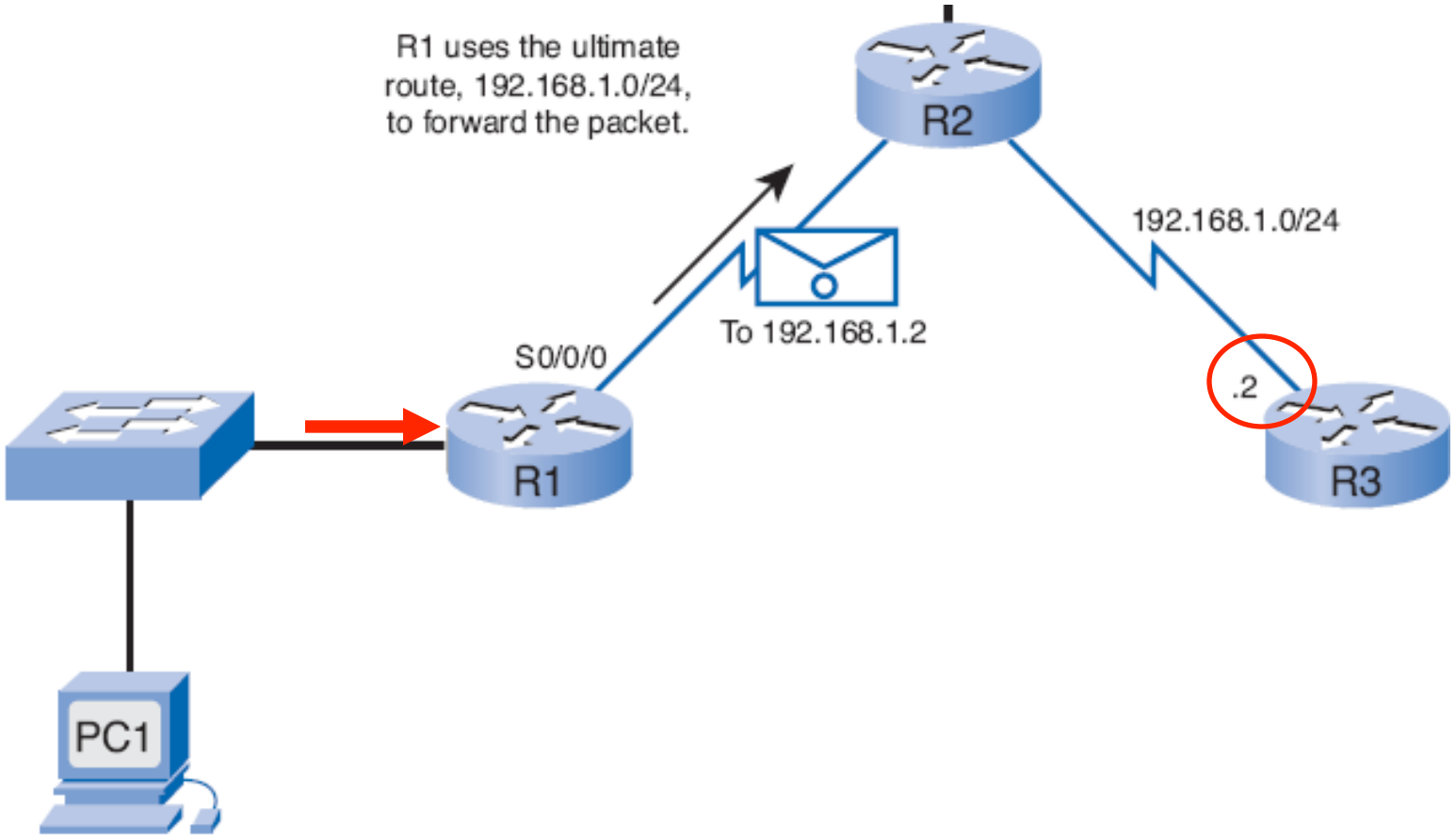
Meilleure correspondance

La meilleure correspondance est la route dans la table de routage qui contient le plus grand nombre de bits en commun avec l'adresse IP du paquet

Quelle est la meilleure route ?

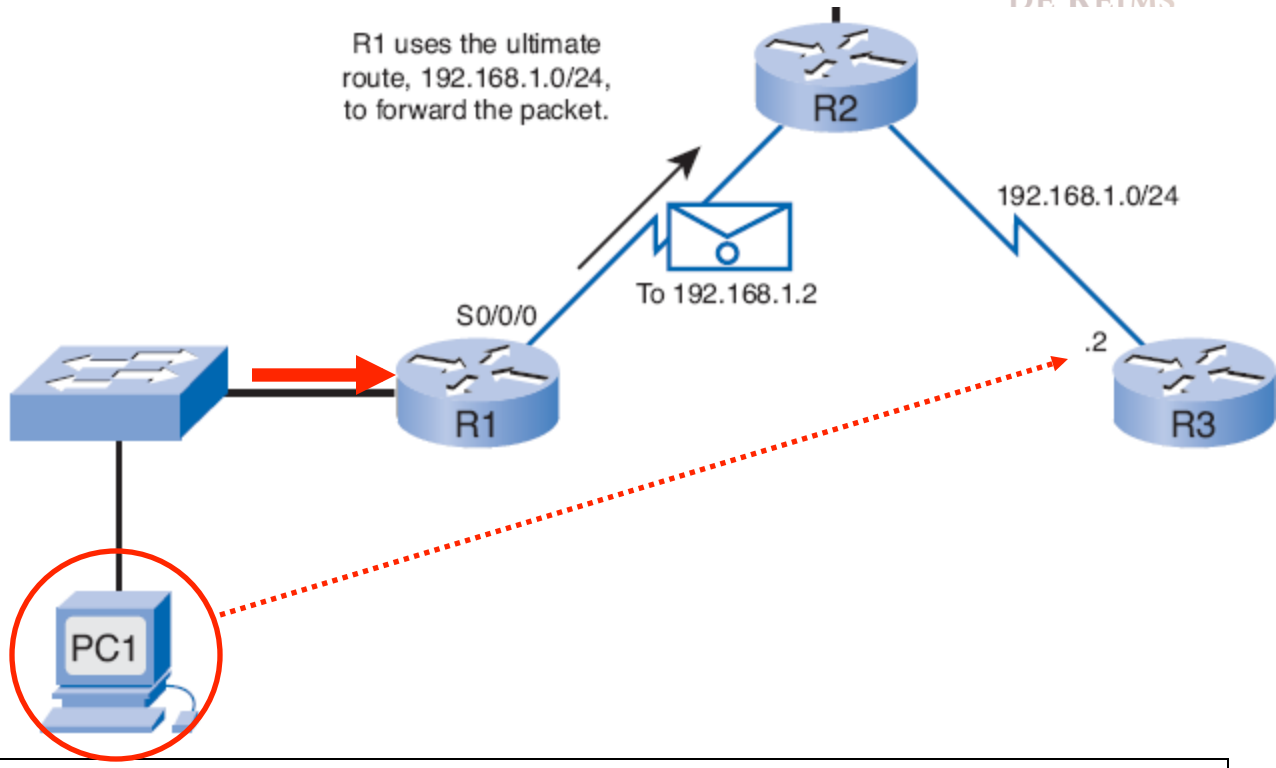
IP Packet Destination	172.16.0.10	10101100.00010000.00000000.00001010
Route 1	172.16.0.0/12	10101100.00010000.00000000.00000000
Route 2	172.16.0.0/18	10101100.00010000.00000000.00000000
Route 3	172.16.0.0/26	10101100.00010000.00000000.00000000

Exemple : Route définitive de niveau 1



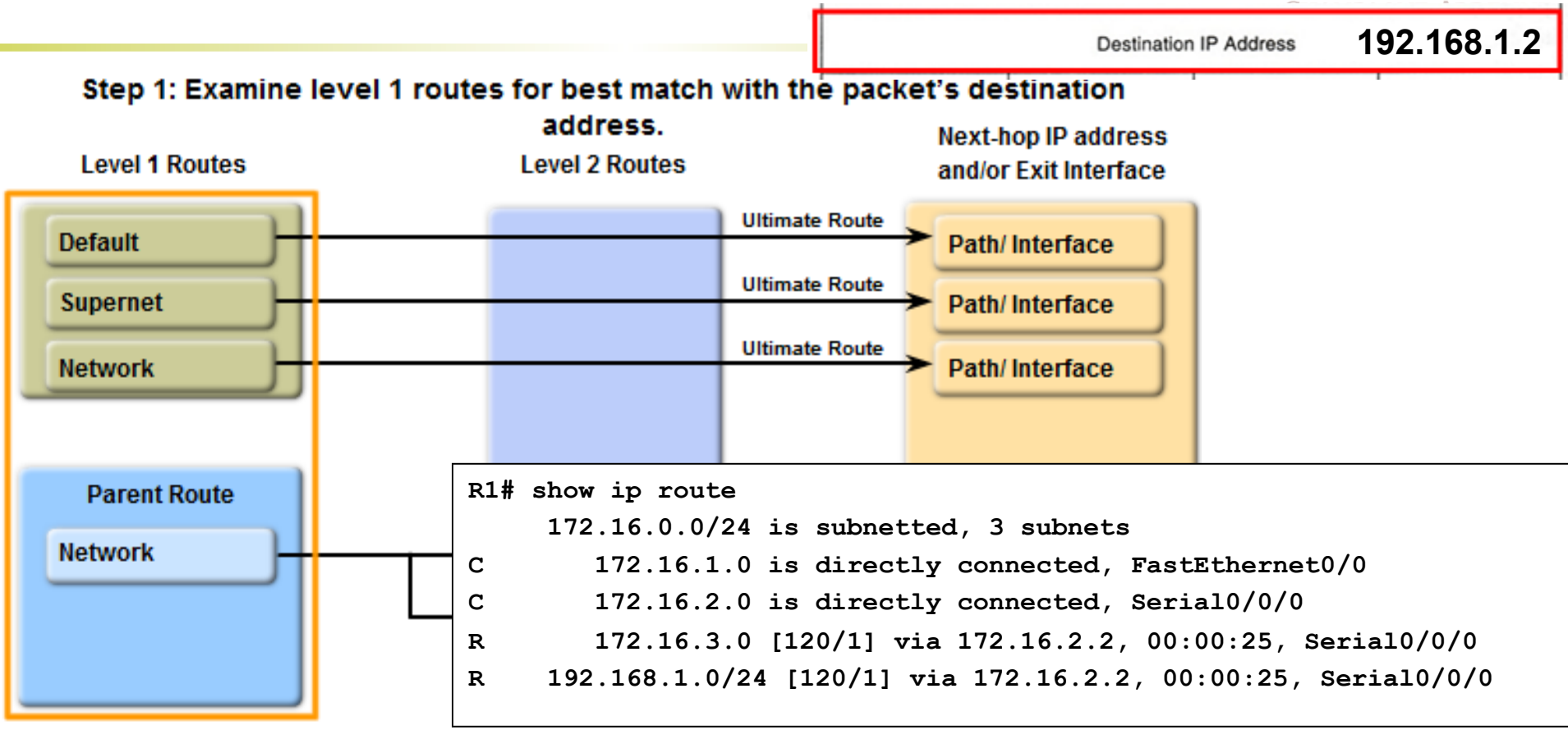
Exemple : Route définitive de niveau 1

PC1 envoie un ping à 192.168.1.2, l'interface série de R3




```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

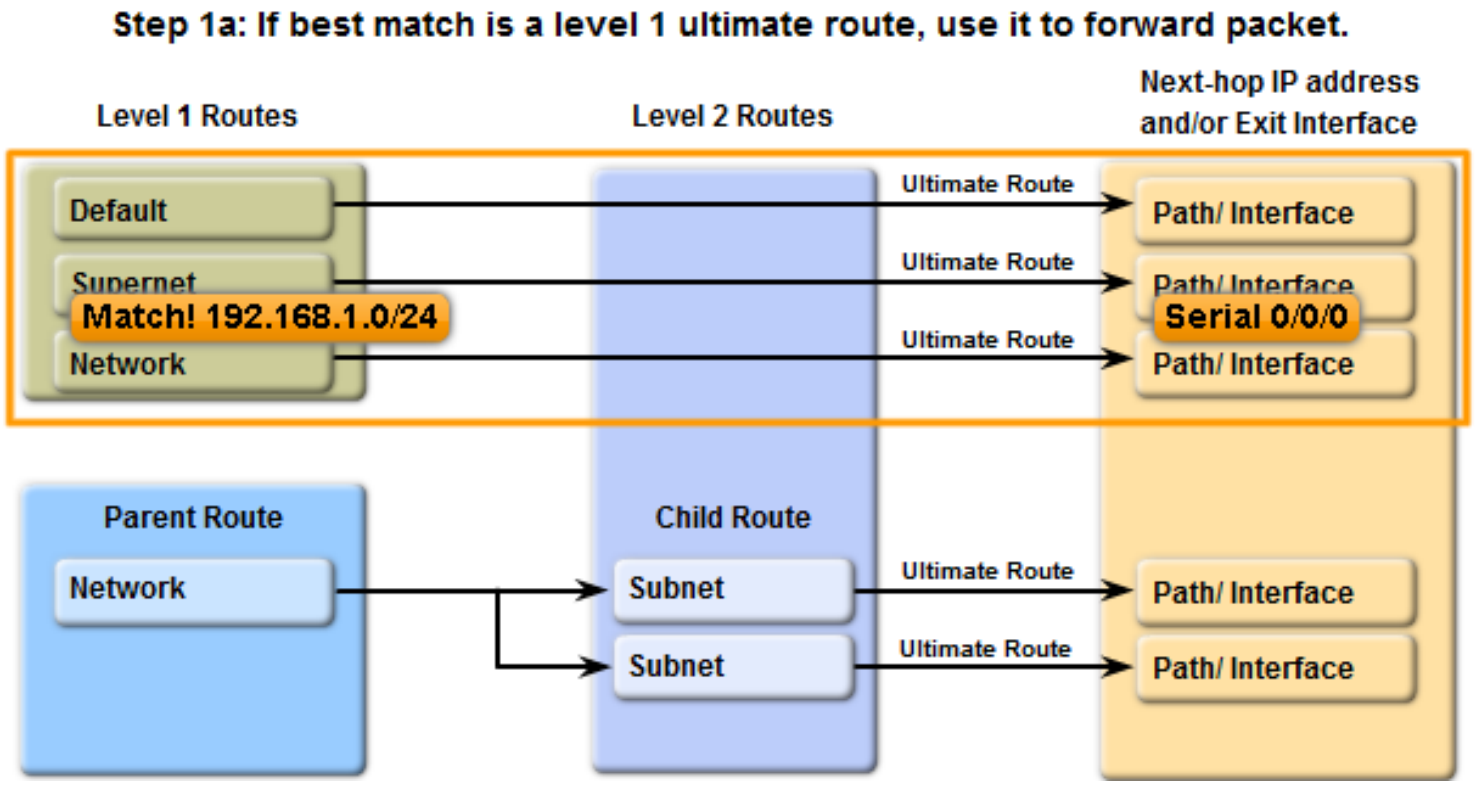
Le routeur examine d'abord les routes niveau 1 pour trouver une correspondance




```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

Une correspondance est trouvée entre **l'adresse destination 192.168.1.2** et la route définitive de 192.168.1.0/24

Destination IP Address **192.168.1.2**

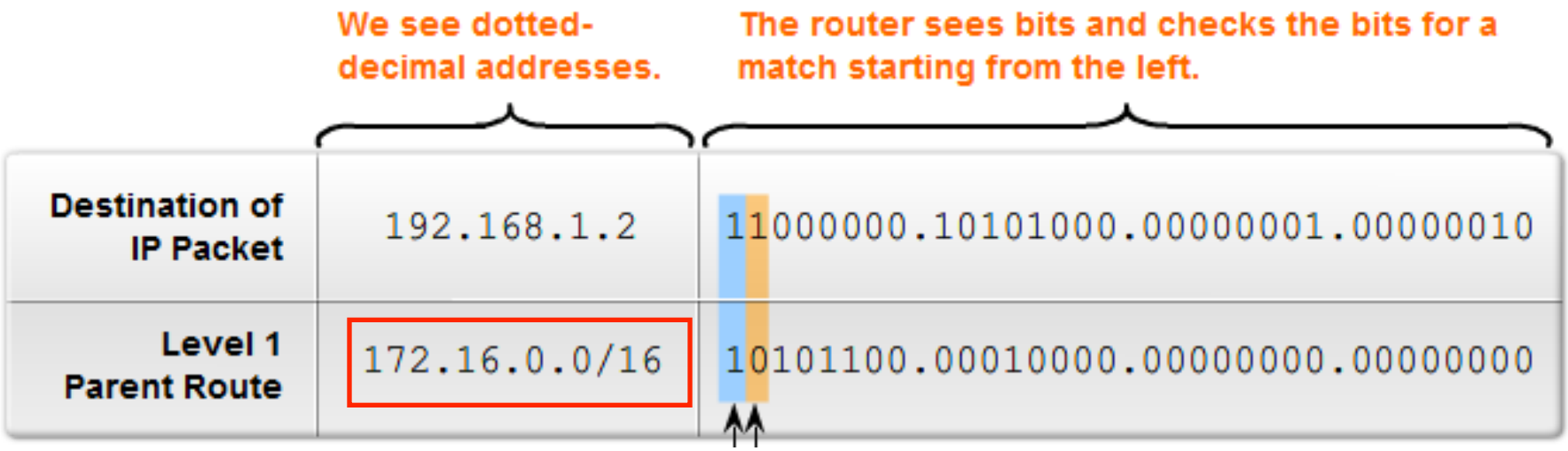



```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

Exemple : Route définitive de niveau 1

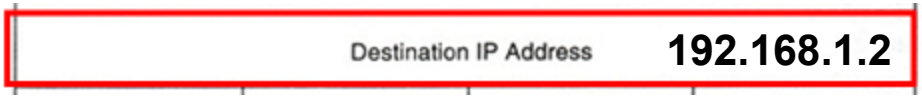
Pourquoi avoir choisi la route niveau 1 192.168.1.0/24 au lieu de regarder les sous-réseaux de 172.16.0.0 ? **Car il faut un minimum de correspondance !**

Destination IP Address	**192.168.1.2**

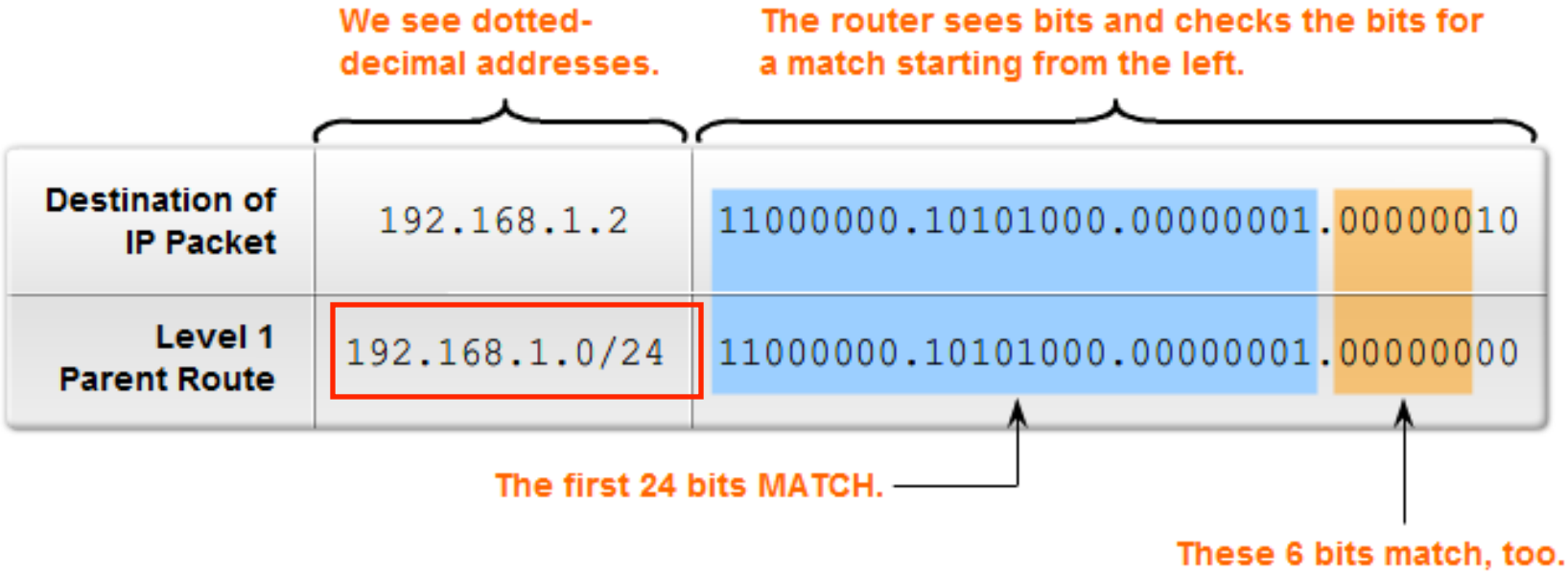


Exemple : Route définitive de niveau 1

La route choisie respecte le **minimum de 24 bits**. Il y a même **30 bits en commun !**

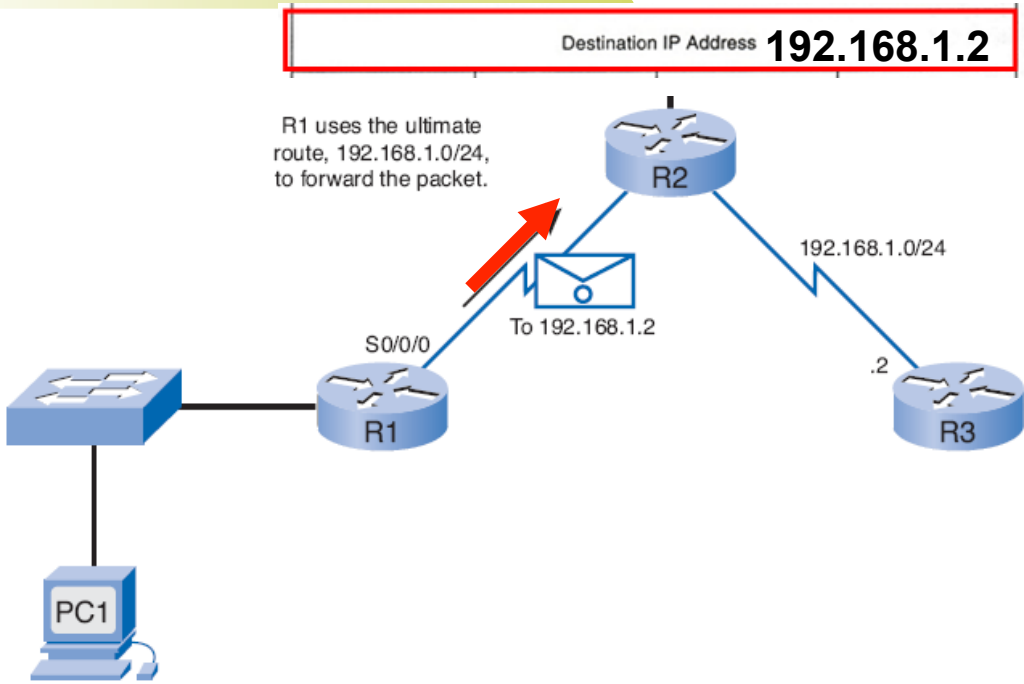



```
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```



Exemple : Route définitive de niveau 1

R1 transfère le paquet via Serial0/0/0.

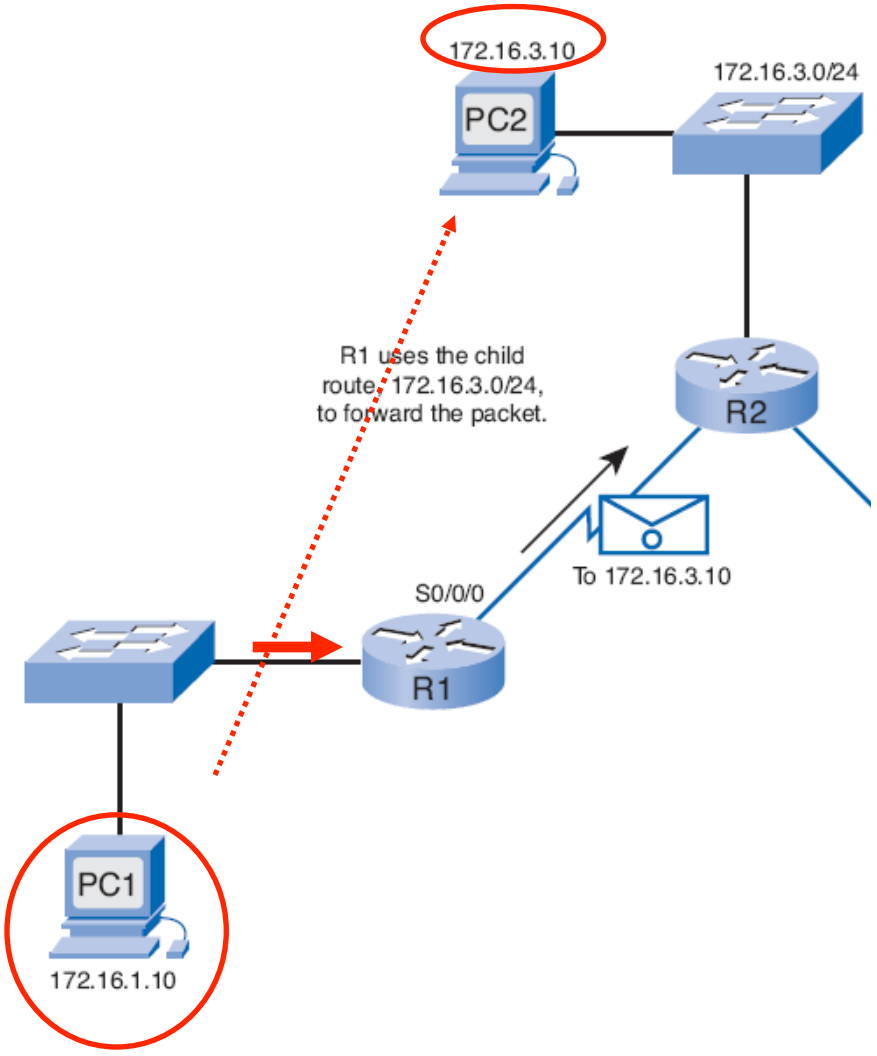


nop

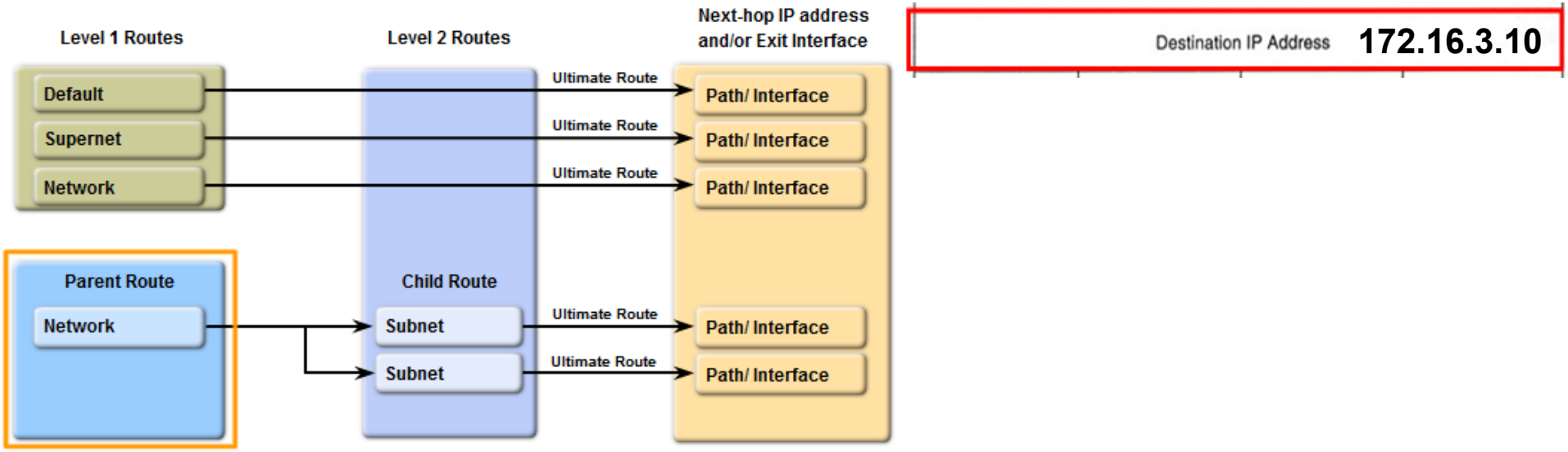
```
R1# show ip route
        172.16.0.0/24 is subnetted, 3 subnets
C          172.16.1.0 is directly connected, FastEthernet0/0
C          172.16.2.0 is directly connected, Serial0/0/0
R          172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R       192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

ici

Meilleure correspondance : routes niveau 1 et 2



La **première correspondance** trouvée est celle de la route parent niveau 1 172.16.0.0

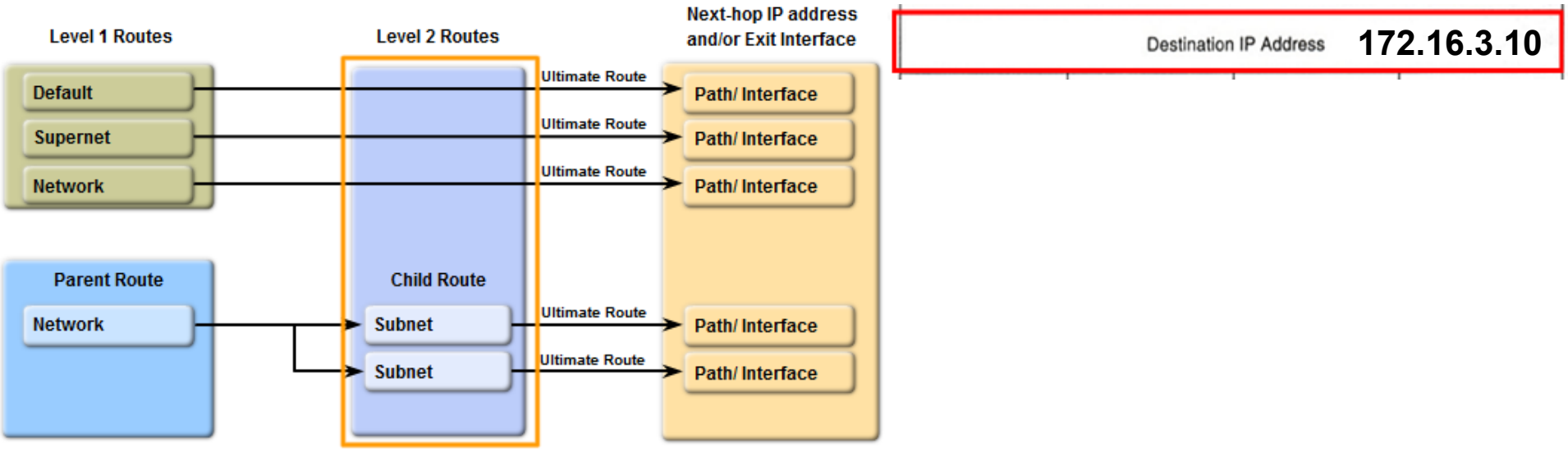


Ok

```
R1# show ip route
172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

Comme on a une correspondance avec la route parent, les **routes enfant** seront examinées à la recherche d'une correspondance

Le masque **/24** indique le nombre minimum de bits qui doivent correspondre



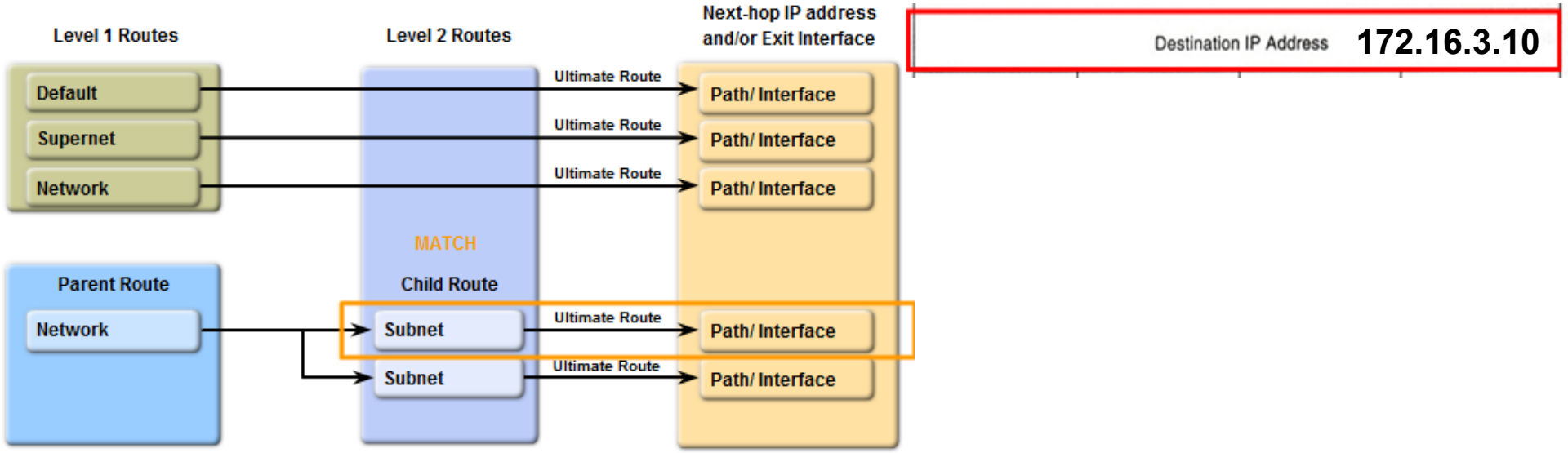
Vérifie les routes enfant

```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

Le processus de recherche regarde les routes enfant.

Des trois sous-réseaux, seulement un a une correspondance à 24 bits

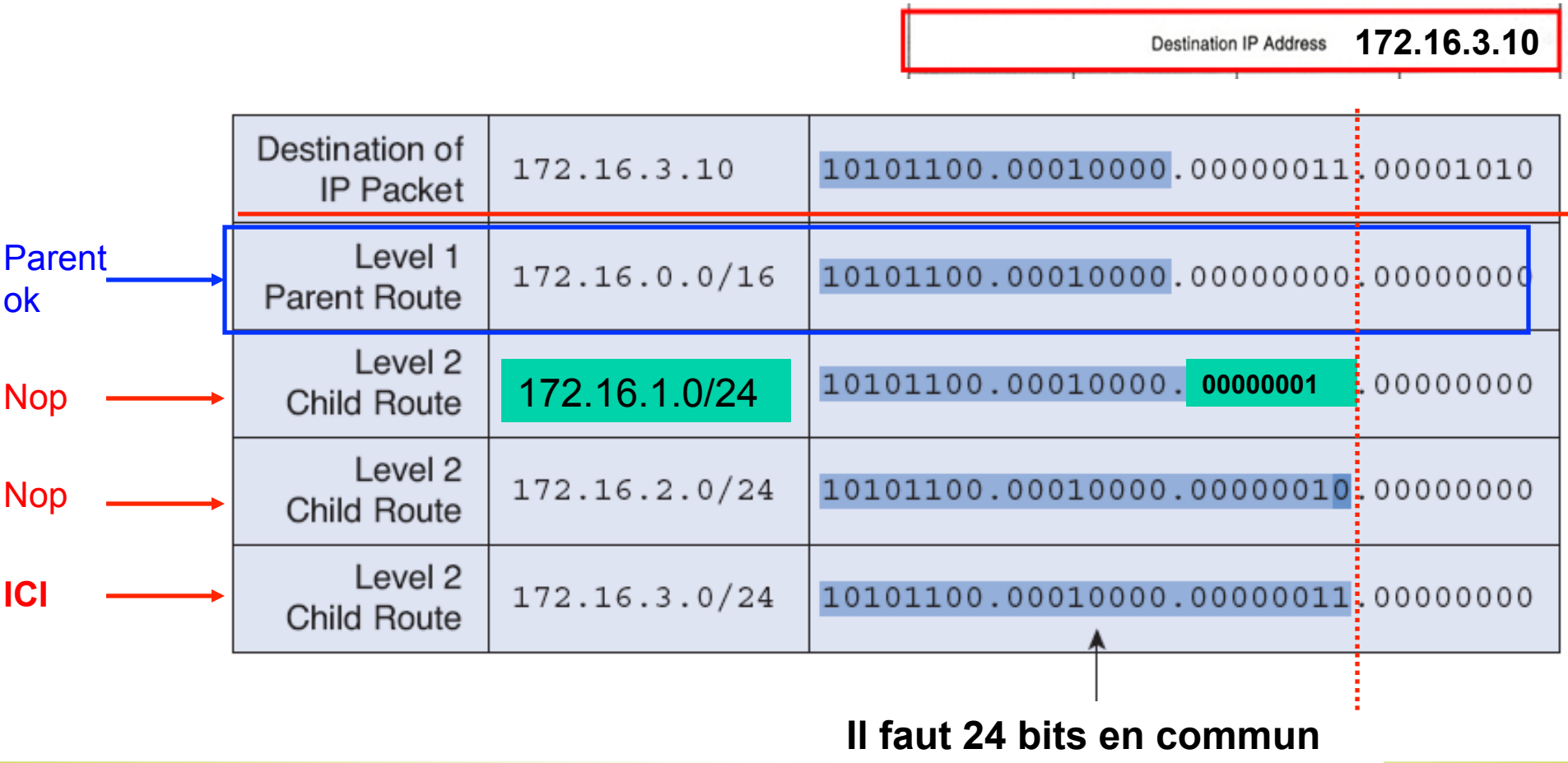
172.16.3.0




```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
R    192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25, Serial0/0/0
```

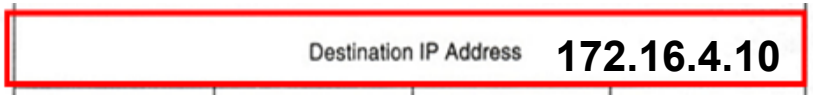
Ici !

Le routeur vérifie la route enfant 172.16.3.0/24 et trouve une correspondance



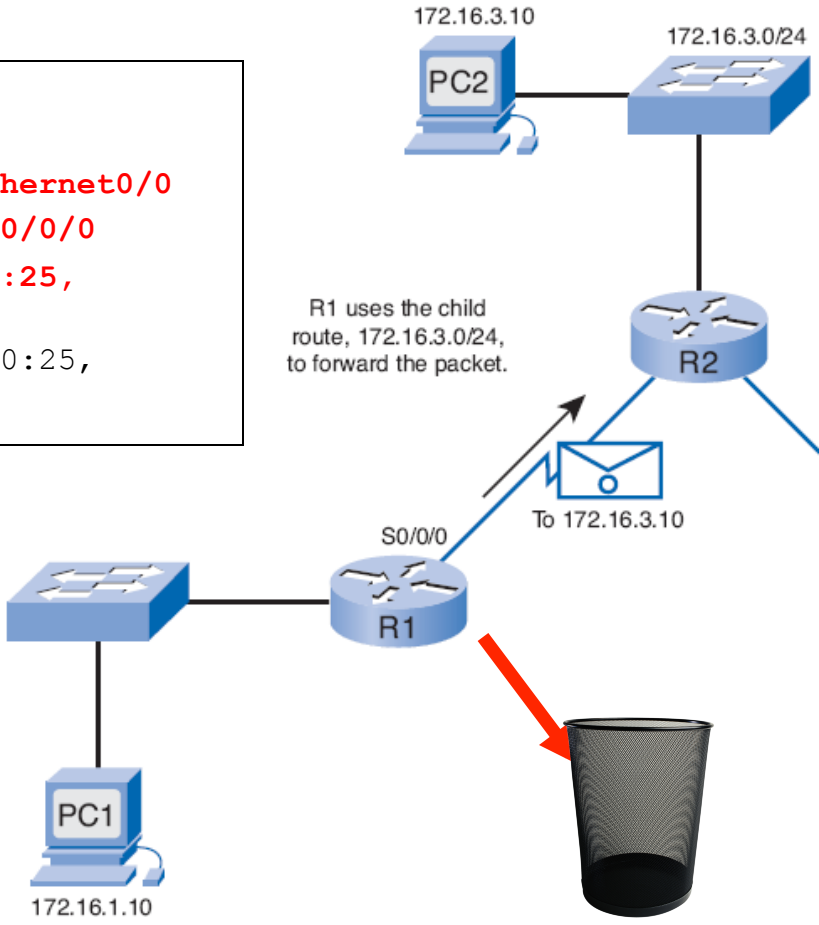
Et si le routeur ne trouve pas une route

Dans ce scénario, il jette le paquet

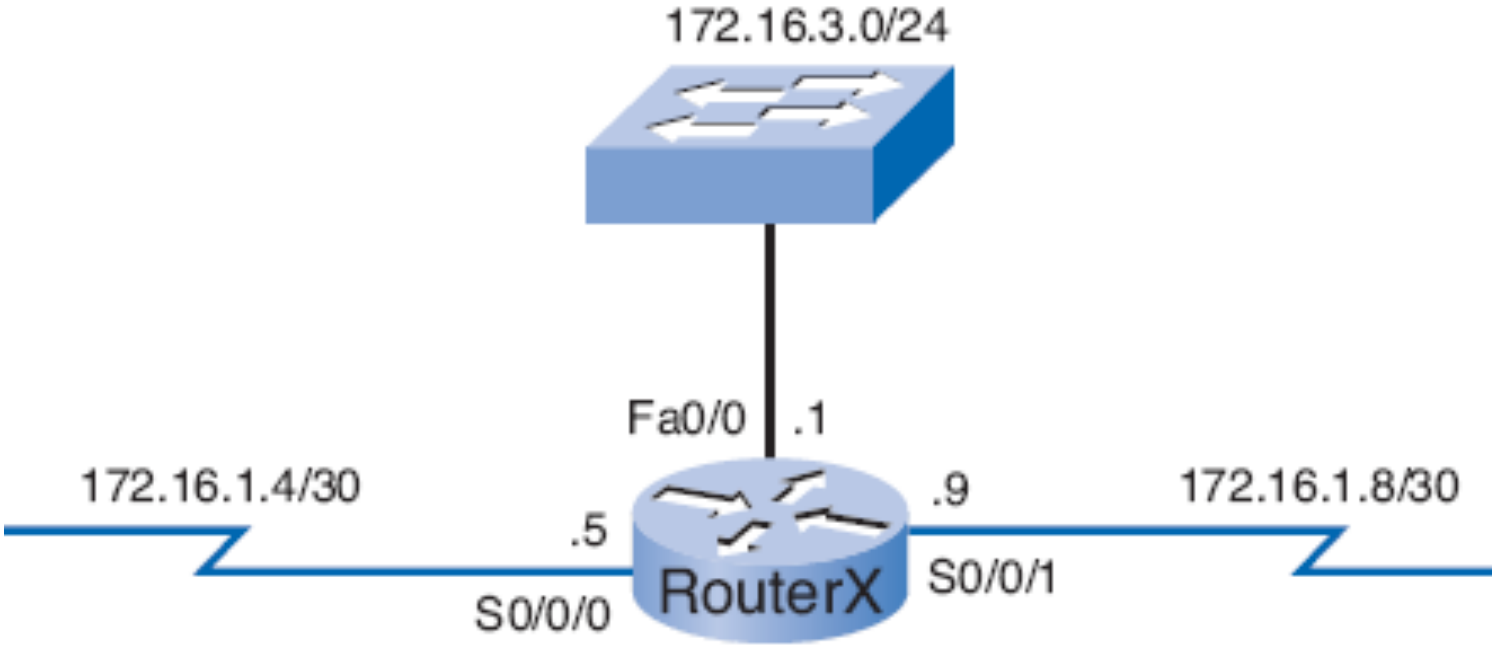



```
R1# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
C       172.16.1.0 is directly connected, FastEthernet0/0
C       172.16.2.0 is directly connected, Serial0/0/0
R       172.16.3.0 [120/1] via 172.16.2.2, 00:00:25,
        Serial0/0/0
R     192.168.1.0/24 [120/1] via 172.16.2.2, 00:00:25,
        Serial0/0/0
```

Parent ok
Nop
Nop
Nop



Exemple : Processus de routage avec VLSM

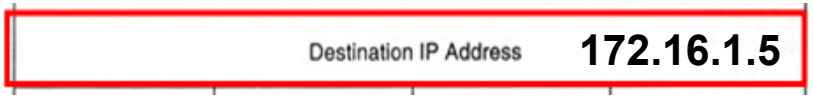


Exemple : Processus de routage avec VLSM

Destination du paquet **: 172.16.1.5**

L'utilisation de ***VSLM ne modifie pas le processus de routage***

La seule différence est que les routes enfant affichent leurs propres masques



Match Parent

Match child

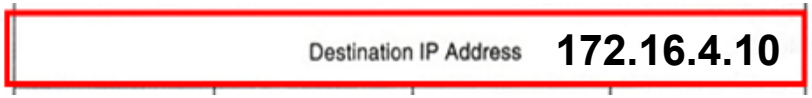
```
RouterX# show ip route

      172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
C        172.16.1.4/30 is directly connected, Serial0/0/0
C        172.16.1.8/30 is directly connected, Serial0/0/1
C        172.16.3.0/24 is directly connected, FastEthernet0/0
```

Comportement de routage

Comportement de routage classful et classless

Que arrive-t-il s'il y a une correspondance avec une route parent niveau 1, mais aucune correspondance avec le routes enfant niveau 2 ?



Parent ok

Nop

Nop

Nop

Et alors?

```
R2# show ip route

     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
S*    0.0.0.0/0 is directly connected, Serial0/0/1
```

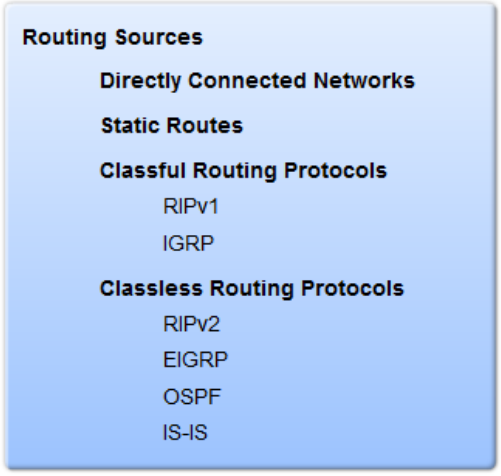
Comportement de routage classful et classless

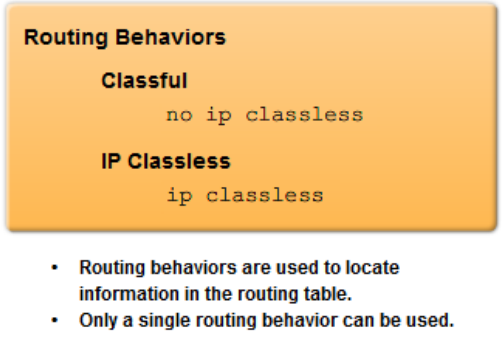
Avoir un comportement classful ou classless n'est pas la même chose qu'être un protocole classful ou classless

Pas si vrai, les protocoles classless forcent le comportement classless

Être un protocole classful ou classless *affecte comment la table sera remplie*

Avoir un comportement classful ou classless détermine comment chercher dans la table de routage

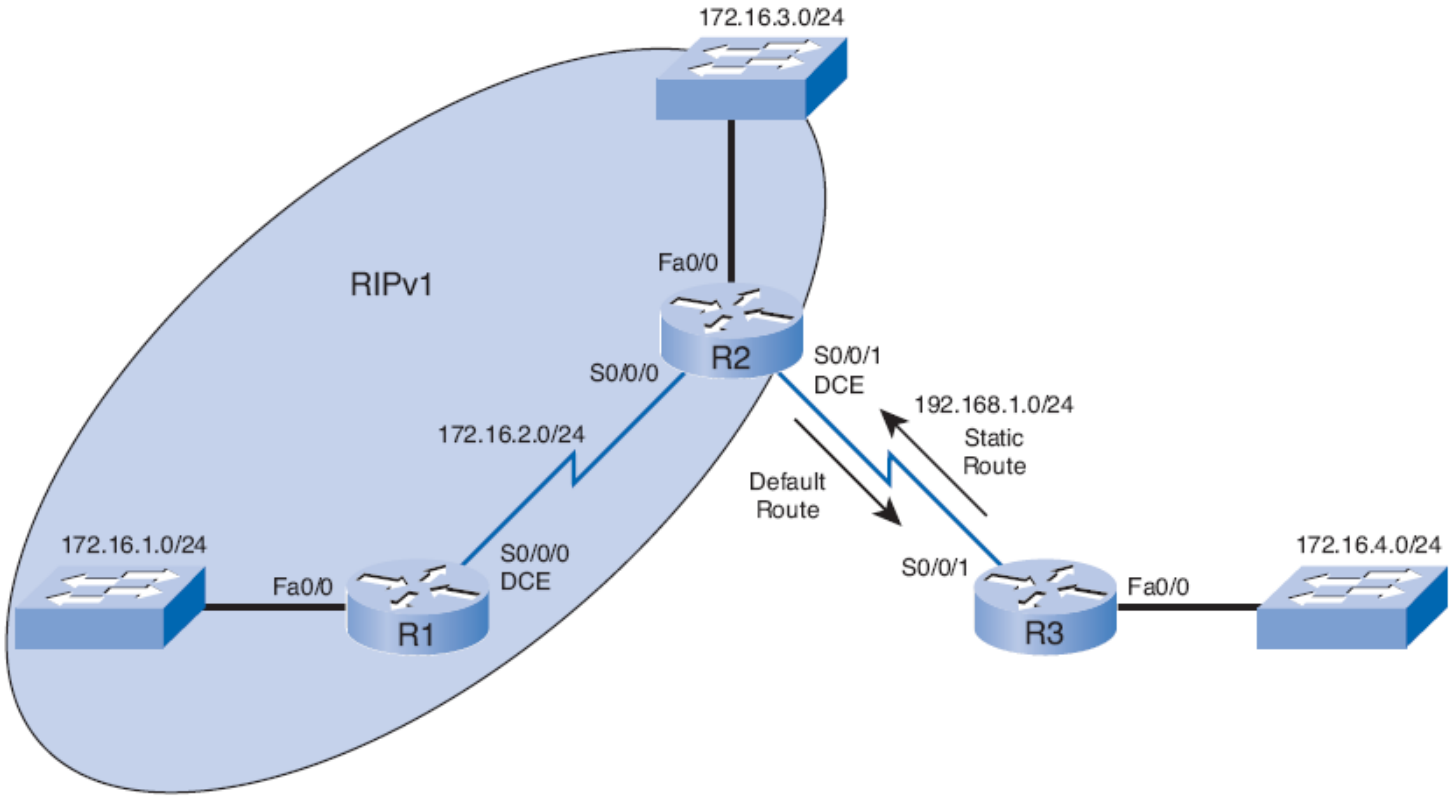




Encore une topologie

Lrs protocoles classful comme RIPv1 ne supportent pas les réseaux non contigus

Cela n'empêche pas qu'on force son fonctionnement grâce à des routes statiques



Encore une topologie

```
R2(config)# ip route 0.0.0.0 0.0.0.0 s0/0/1
R2(config)# router rip
R2(config-router)# default-information originate
R2(config-router)# no network 192.168.1.0
R2(config-router)# end
R2# show ip route

     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
S*   0.0.0.0/0 is directly connected, Serial0/0/1
```

Encore une topologie

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

On voit que la route statique pour le même réseau devient une route enfant dans la table de routage

```
R3(config)# ip route 172.16.0.0 255.255.0.0 s0/0/1
R3(config)# no router rip
R3(config-router)# end
R3# show ip route

     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C       172.16.4.0/24 is directly connected, FastEthernet0/0
S       172.16.0.0/16 is directly connected, Serial0/0/1
C    192.168.1.0/24 is directly connected, Serial0/0/1
```

Comportement classful : no ip classless

Avant l' IOS 11.3, **`no ip classless`** était le comportement par **défaut**

La commande **`no ip classless`** indique au routeur qu'il faut utiliser la recherche selon le comportement classful

Match Parent

No match

No match

No match

Et alors ?

```
R2# show ip route
     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
S*    0.0.0.0/0 is directly connected, Serial0/0/1
```

```
R1(config)# no ip classless
```

```
R2(config)# no ip classless
```

```
R3(config)# no ip classless
```

Destination IP Address **172.16.4.10**

Comportement classful : no ip classless

Et maintenant ? On a un paquet qui arrive en direction de 172.16.4.10

Le routeur R2 jette le paquet

La route par défaut n'est jamais atteinte et utilisée !!!!

```
R2# show ip route

     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
S*    0.0.0.0/0 is directly connected, Serial0/0/1
```

Match Parent → 172.16.0.0/24 is subnetted, 3 subnets

No match → R 172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0

No match → C 172.16.2.0 is directly connected, Serial0/0/0

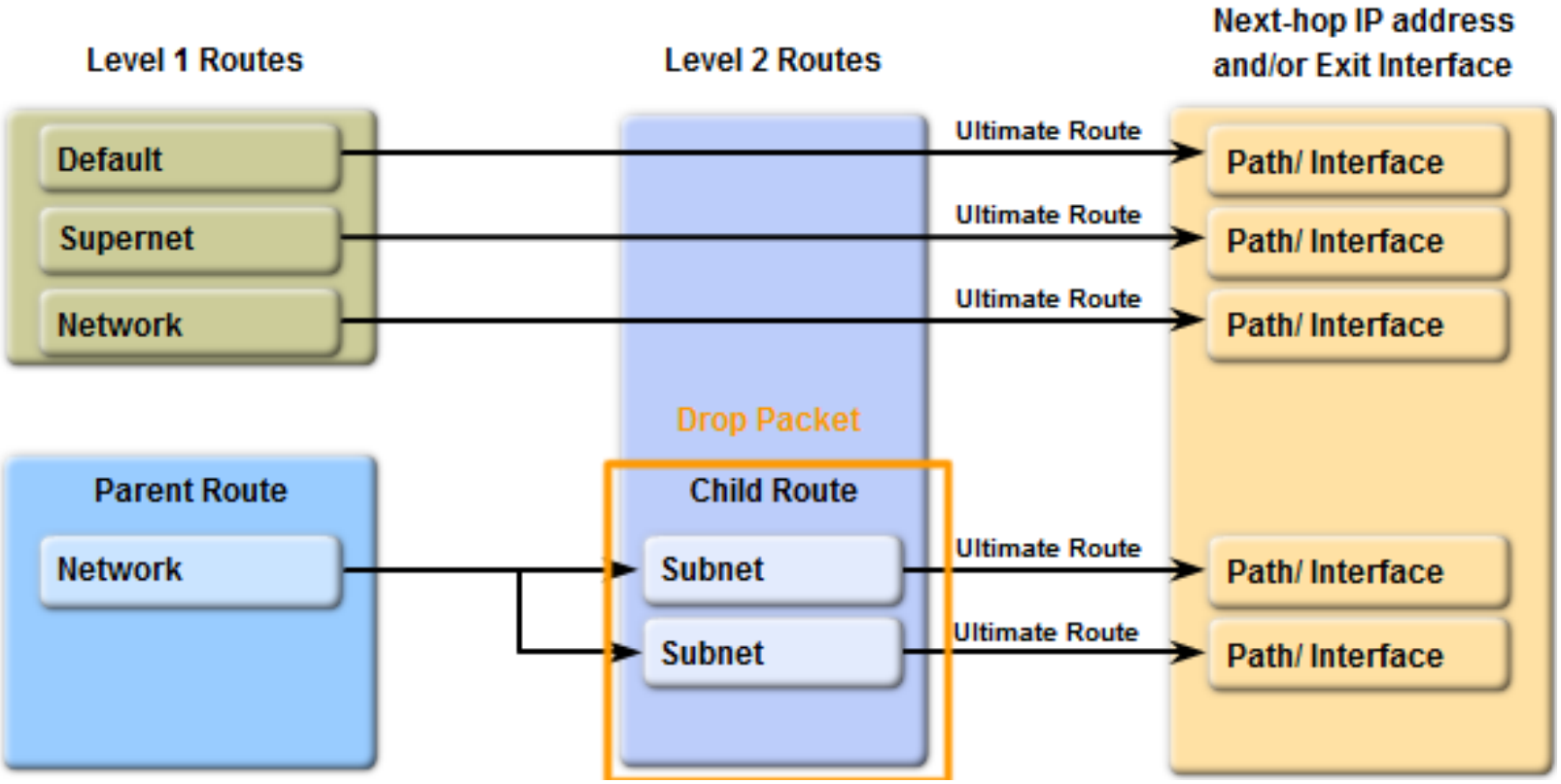
No match → C 172.16.3.0 is directly connected, FastEthernet0/0

Et alors?

→ **Jeter le paquet**

Destination IP Address 172.16.4.10

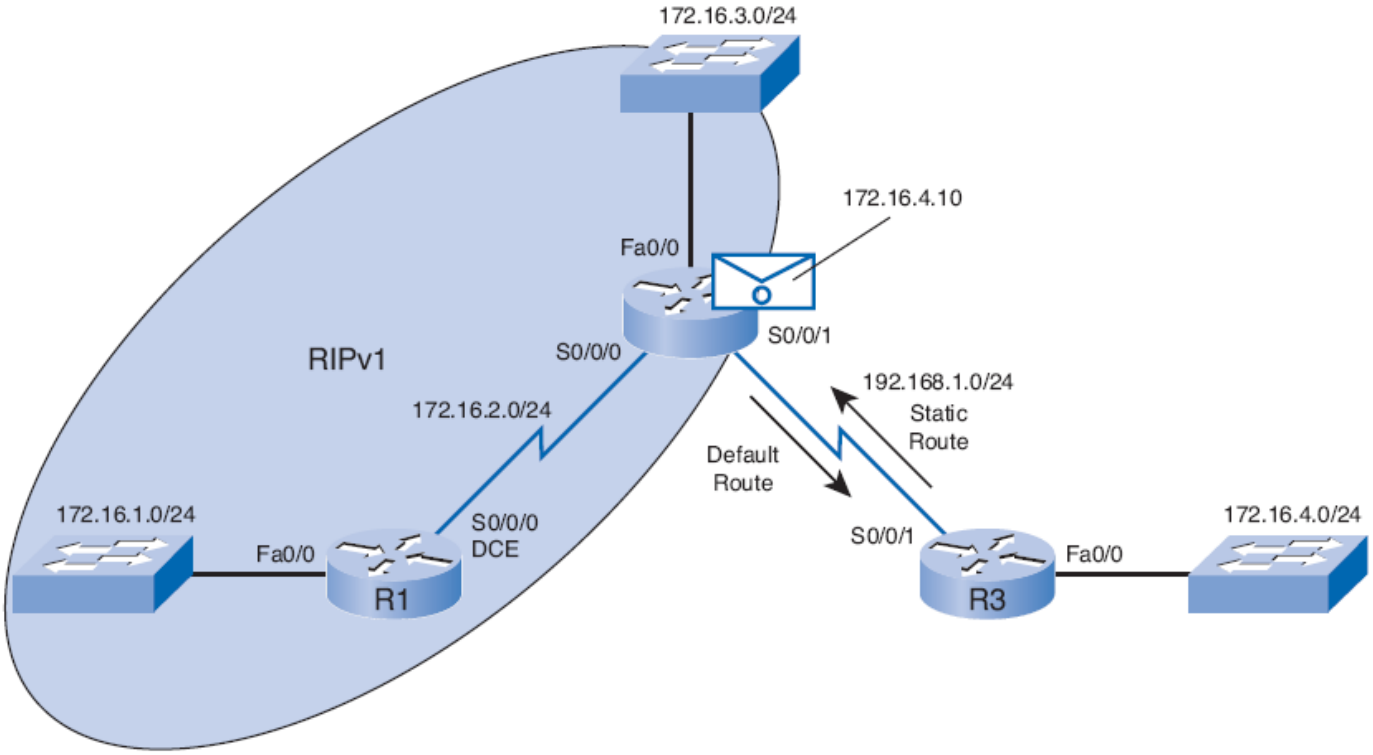
Comportement classful : no ip classless



Destination IP Address 172.16.4.10

Comportement classless : ip classless

R2 reçoit un paquet destiné à PC3 (172.16.4.10)



Destination IP Address 172.16.4.10

Comportement classless : ip classless

Match Parent →
No match →
No match →
No match →

Now what?

```
R2# show ip route

     172.16.0.0/24 is subnetted, 3 subnets
R       172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C       172.16.2.0 is directly connected, Serial0/0/0
C       172.16.3.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0/1
S*    0.0.0.0/0 is directly connected, Serial0/0/1
```

```
R1(config)# ip classless
```

```
R2(config)# ip classless
```

```
R3(config)# ip classless
```

Destination IP Address 172.16.4.10

Comportement classless : ip classless


```
R2# show ip route

      172.16.0.0/24 is subnetted, 3 subnets
R        172.16.1.0 [120/1] via 172.16.2.1, 00:00:00, Serial0/0/0
C        172.16.2.0 is directly connected, Serial0/0/0
C        172.16.3.0 is directly connected, FastEthernet0/0
C     192.168.1.0/24 is directly connected, Serial0/0/1
S*    0.0.0.0/0 is directly connected, Serial0/0/1
```

Match Parent

No match

No match

No match

Now what?

Le routeur continue la recherche des routes parent niveau 1. La route par défaut correspond, alors on l'utilise pour transférer le paquet

Protocoles de Routage à État des Liens

Introduction aux Protocoles à État des Liens

Université de Reims Champagne-Ardenne

Protocoles de routage à état des liens

- Aussi connus sous le nom de protocoles shortest path first
- Utilisent l'algorithme ***shortest path first (SPF)*** de Edsger Dijkstra

Ont la réputation d'être plus complexes que les protocoles à vecteur de distances

- Son opération et configuration **ne sont pas** complexes
- L'algorithm est simple à comprendre

	Interior Gateway Protocols				Exterior Gateway Protocols
	Distance Vector Routing Protocols		Link State Routing Protocols		Path Vector
Classful	**RIP**	IGRP			EGP
Classless	**RIPv2**	**EIGRP**	**OSPFv2**	IS-IS	BGPv4
IPv6	RIPng	EIGRP for IPv6	OSPFv3	IS-IS for IPv6	BGPv4 for IPv6

Rappel

Protocoles à vecteur de distances - panneau sur une route

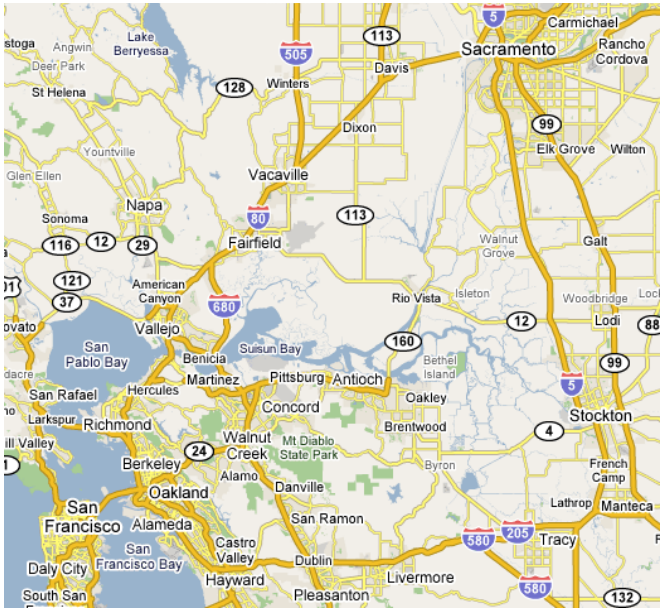
Distance et vecteur

Protocoles à état des liens - carte routière

Carte topologique

Chaque routeur détermine le meilleur chemin

Vecteur de distances

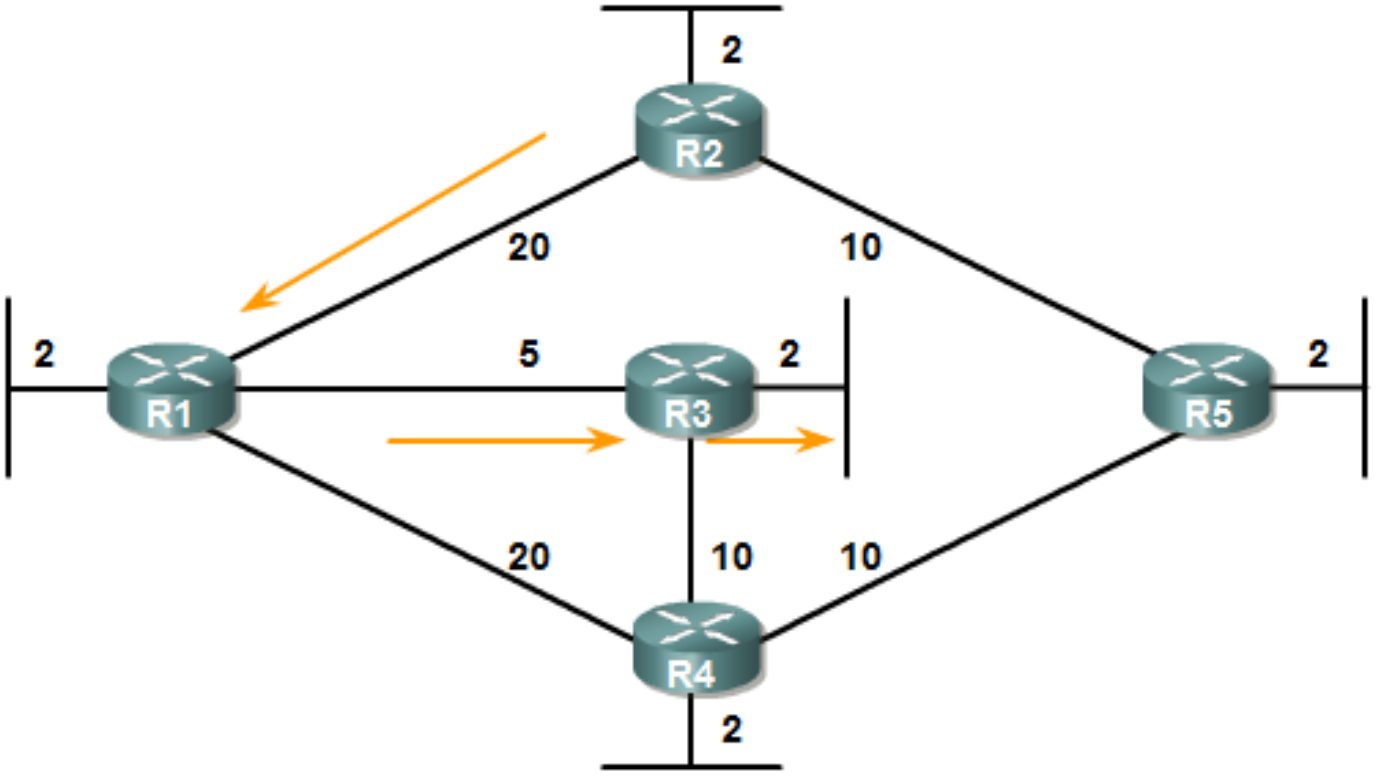


État des liens

Introduction à l'Algorithme SPF

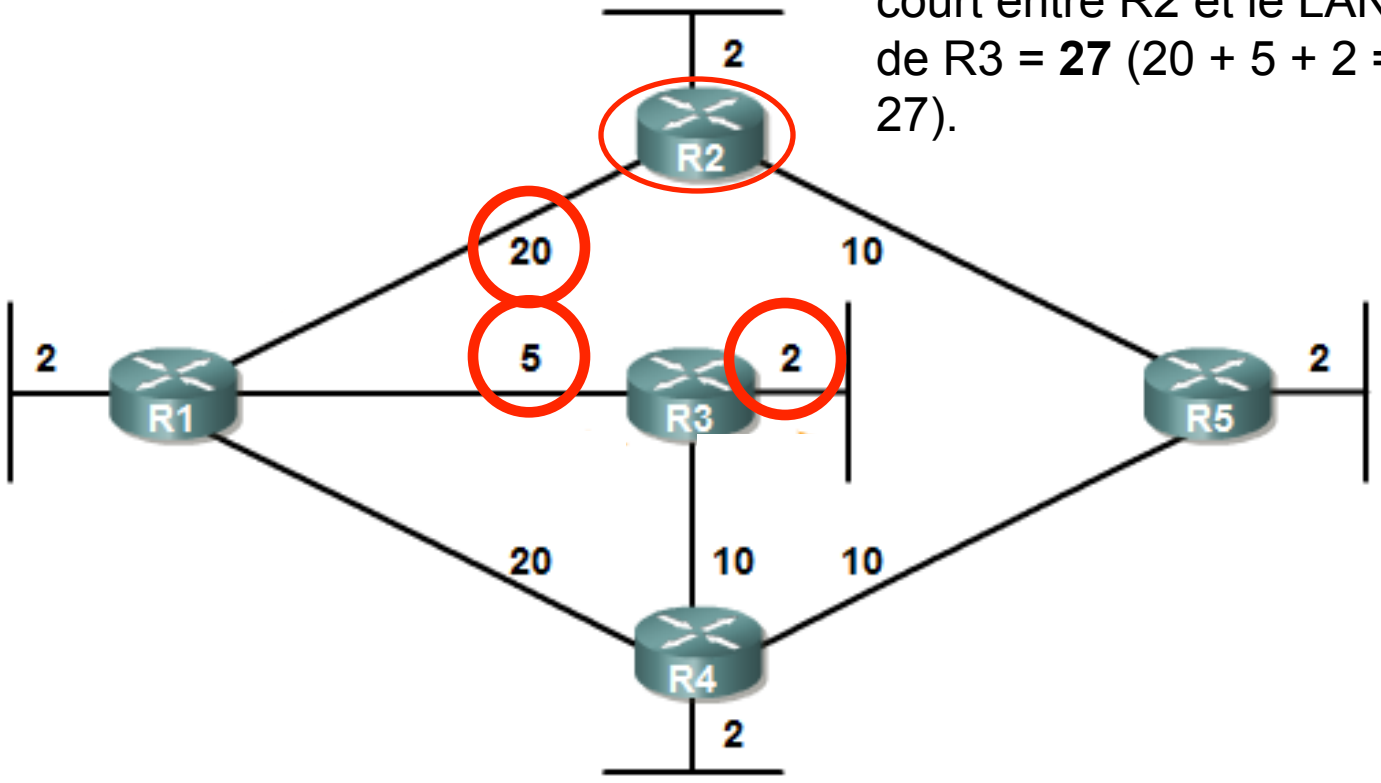
L'algorithme de Dijkstra est souvent appelé le shortest path first (SPF)

Trouver le chemin le plus court (shortest path) est l'objectif de l'algorithme



Introduction à l'Algorithme SPF

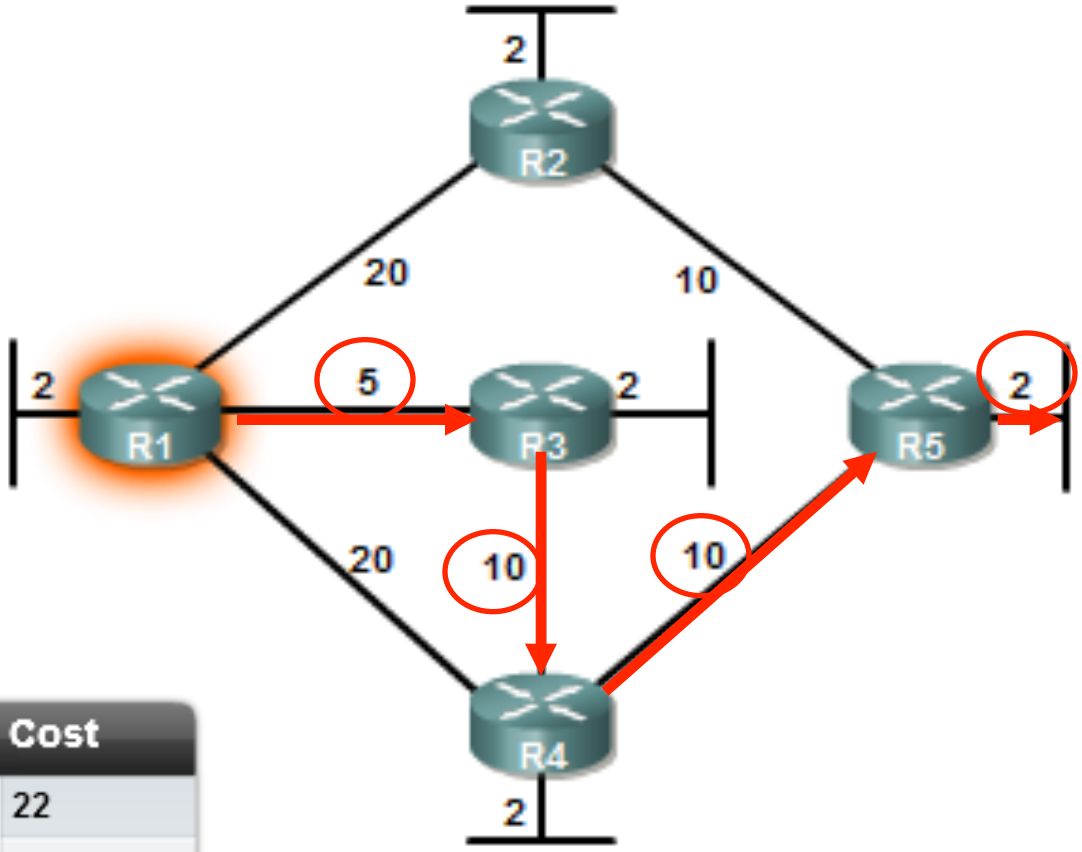
Chaque routeur exécute l'algorithme SPF et détermine le coût des liens selon son point de vue



Le coût du chemin plus court entre R2 et le LAN de R3 = **27** (20 + 5 + 2 = 27).

Introduction à l'Algorithme SPF

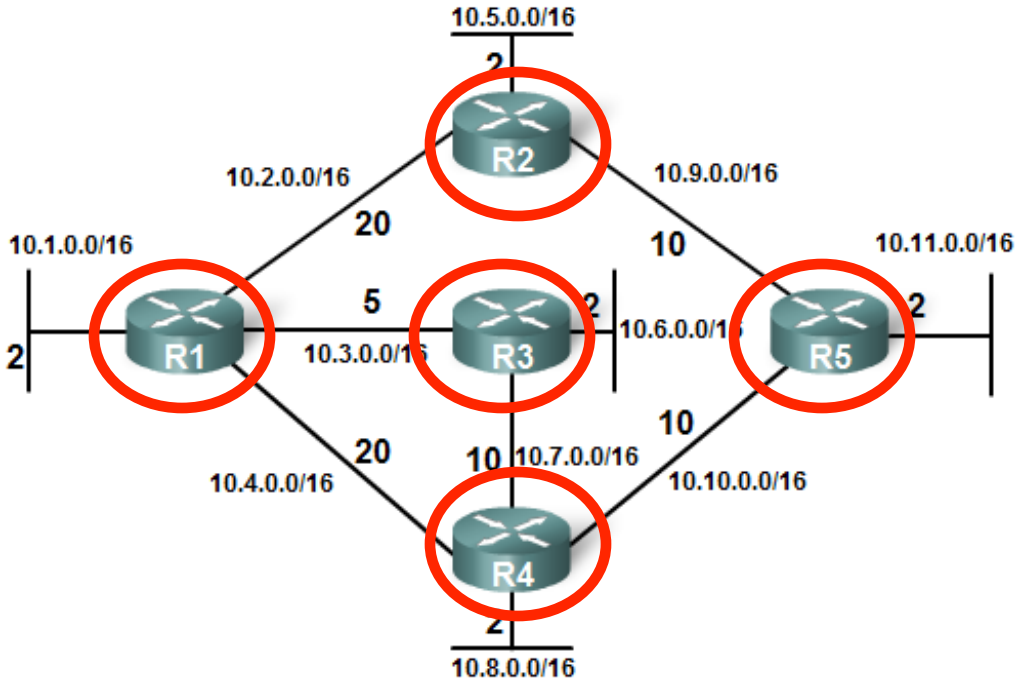
Le chemin le plus court n'est pas nécessairement le chemin avec moins de sauts



SPF pour R1

Destination	Shortest Path	Cost
R2 LAN	R1 to R2	22
R3 LAN	R1 to R3	7
R4 LAN	R1 to R3 to R4	17
R5 LAN	R1 to R3 to R4 to R5	27

Les chemins les plus courts pour chaque routeur



SPF pour R1

Destination	Shortest Path	Cost
R2 LAN	R1 to R2	22
R3 LAN	R1 to R3	7
R4 LAN	R1 to R3 to R4	17
R5 LAN	R1 to R3 to R4 to R5	27

SPF pour R2

Destination	Shortest Path	Cost
R1 LAN	R2 to R1	22
R3 LAN	R2 to R1 to R3	27
R4 LAN	R2 to R5 to R4	22
R5 LAN	R2 to R5	12

SPF pour R3

Destination	Shortest Path	Cost
R1 LAN	R3 to R1	7
R2 LAN	R3 to R1 to R2	27
R4 LAN	R3 to R4	12
R5 LAN	R3 to R4 to R5	22

SPF pour R4

Destination	Shortest Path	Cost
R1 LAN	R4 to R3 to R1	17
R2 LAN	R4 to R5 to R2	22
R3 LAN	R4 to R3	12
R5 LAN	R4 to R5	12

SPF pour R5

Destination	Shortest Path	Cost
R1 LAN	R5 to R4 to R3 to R1	27
R2 LAN	R5 to R2	12
R3 LAN	R5 to R4 to R3	22
R4 LAN	R5 to R4	12

Le processus de détermination de l'état des liens

1. Chaque routeur connaît les routes directement connectées. *(L'interface est "up")*
2. Chaque routeur se charge de trouver les voisins dans les réseaux directement connectés. *(Paquets Hello OSPF)*
3. Chaque routeur construit un ***link-state packet (LSP)*** qui contient l'état de chaque lien directement connecté *(ID des voisins, type de lien et débit)*
4. Chaque routeur diffuse son LSP à tous ses voisins, qui stockent les LSPs reçus dans une base de données

   Les voisins retransmettent les LSPs reçus à leurs propres vosins jusqu'à ce que tous les routeurs dans la zone aient reçu tous les LSPs
5. Chaque routeur utilise la base de données pour construire une carte complète de la topologie; à la suite, il calcule le meilleur chemin vers chaque réseau

   L'algorithme SPF est utilisé

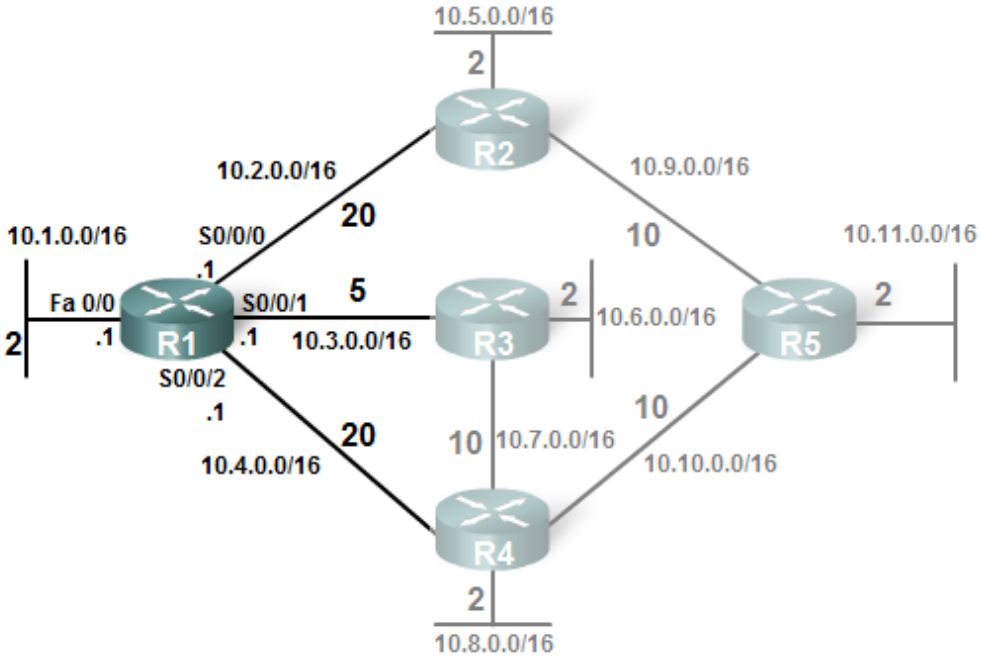
   Tous les routeurs auront une carte commune (ou arbre de topologie),mais chaque routeur choisira indépendamment le meilleur chemin

Étape 1 : Apprendre sur les réseaux directement connectés

Étape 1 : Chaque routeur apprend sur ses propres liens, grâce aux réseaux directement connectés

Les interfaces sont configurées avec une adresse et un masque

Les réseaux directement connectés sont automatiquement inclus dans la table de routage

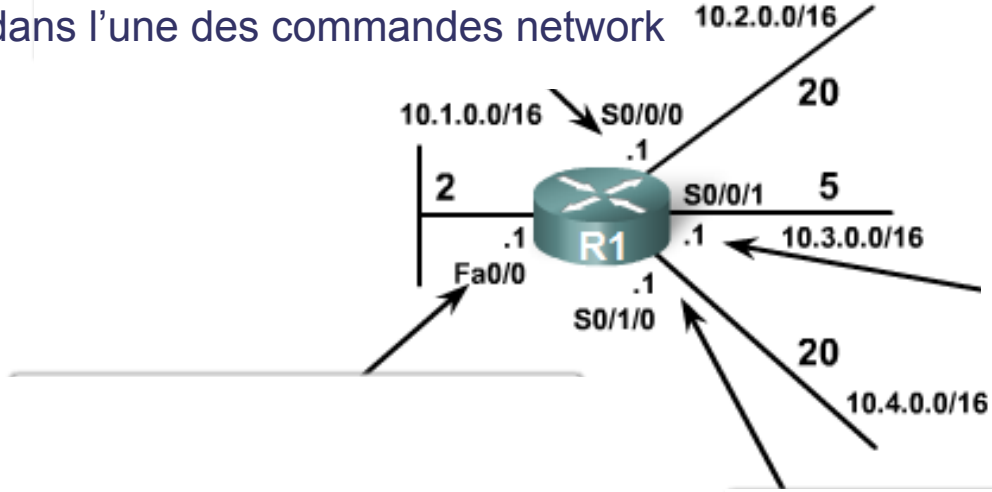


Étape 1

Un lien est une interface dans le routeur

Pour participer au processus de routage, le lien doit :

- Être actif (up)
- Être inclus dans l'une des commandes network

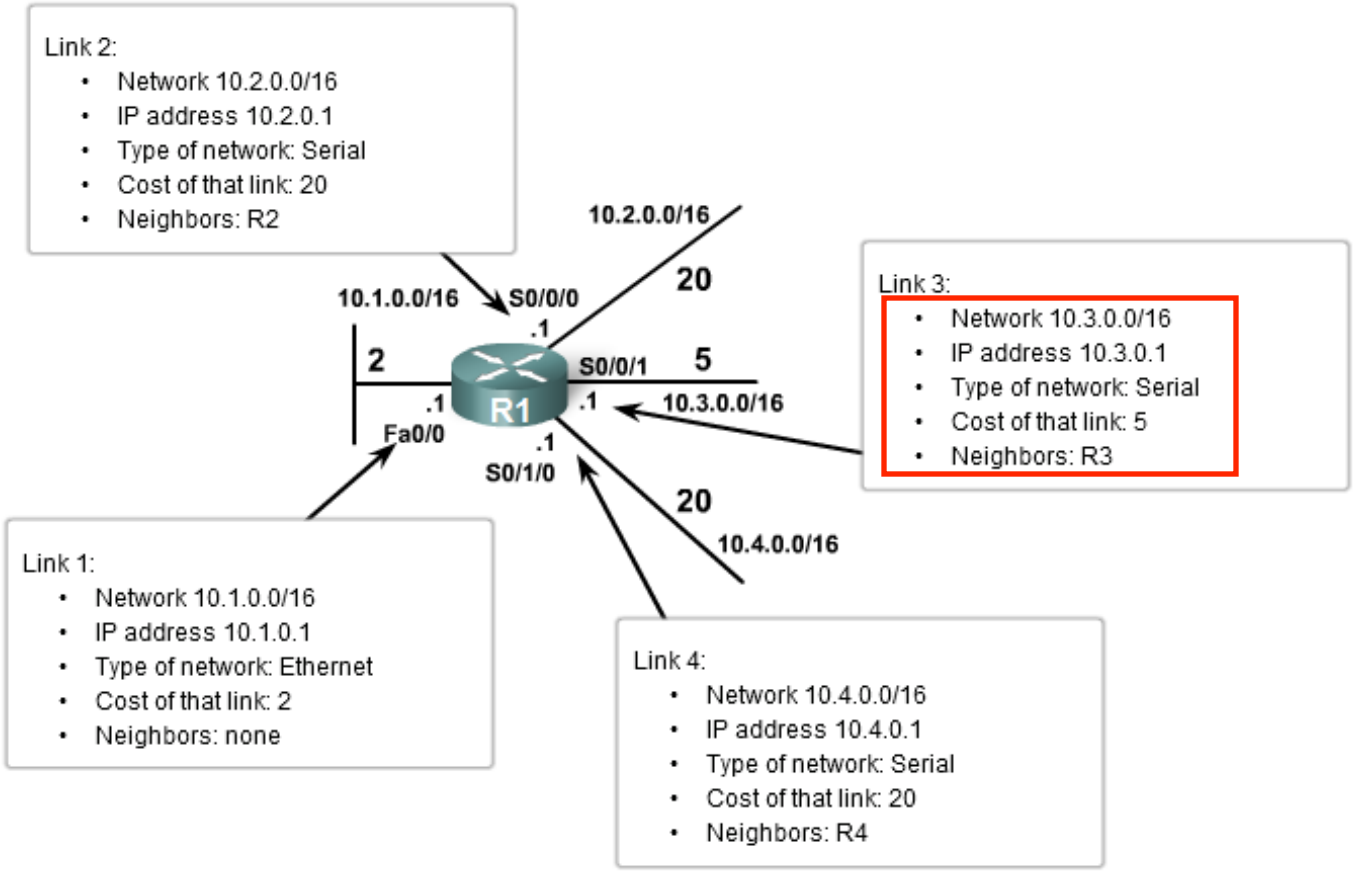


Étape 1

État des liens - Informations sur l'état des liens d'un routeur

Ces informations incluent :

- Adresse/masque IP
- Type de réseau
  - Ethernet (broadcast) ou serial point-to-point
- Coût de ce lien
- Les voisins trouvés dans ce lien

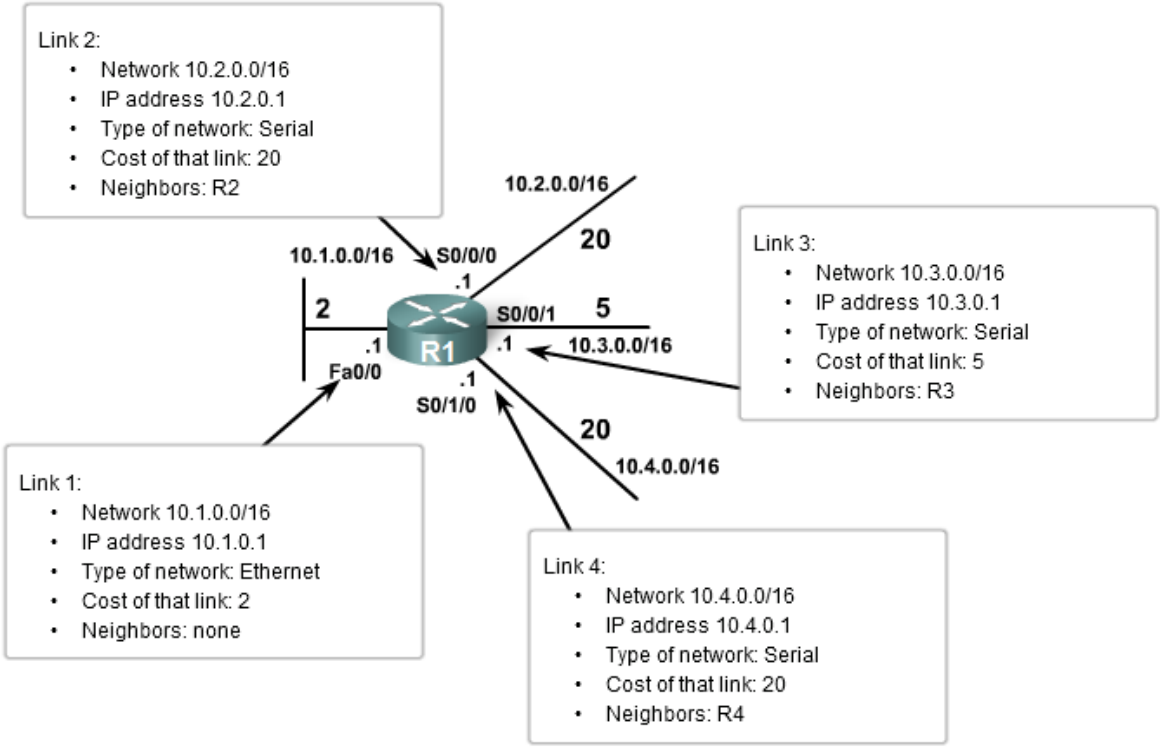


Étape 1

Initialement :

Le routeur ne connaît aucun voisin sur le lien

Il apprend sur les voisins lorsqu'il reçoit un message Hello



Étape 2 : Envoyer des Hello aux Voisins

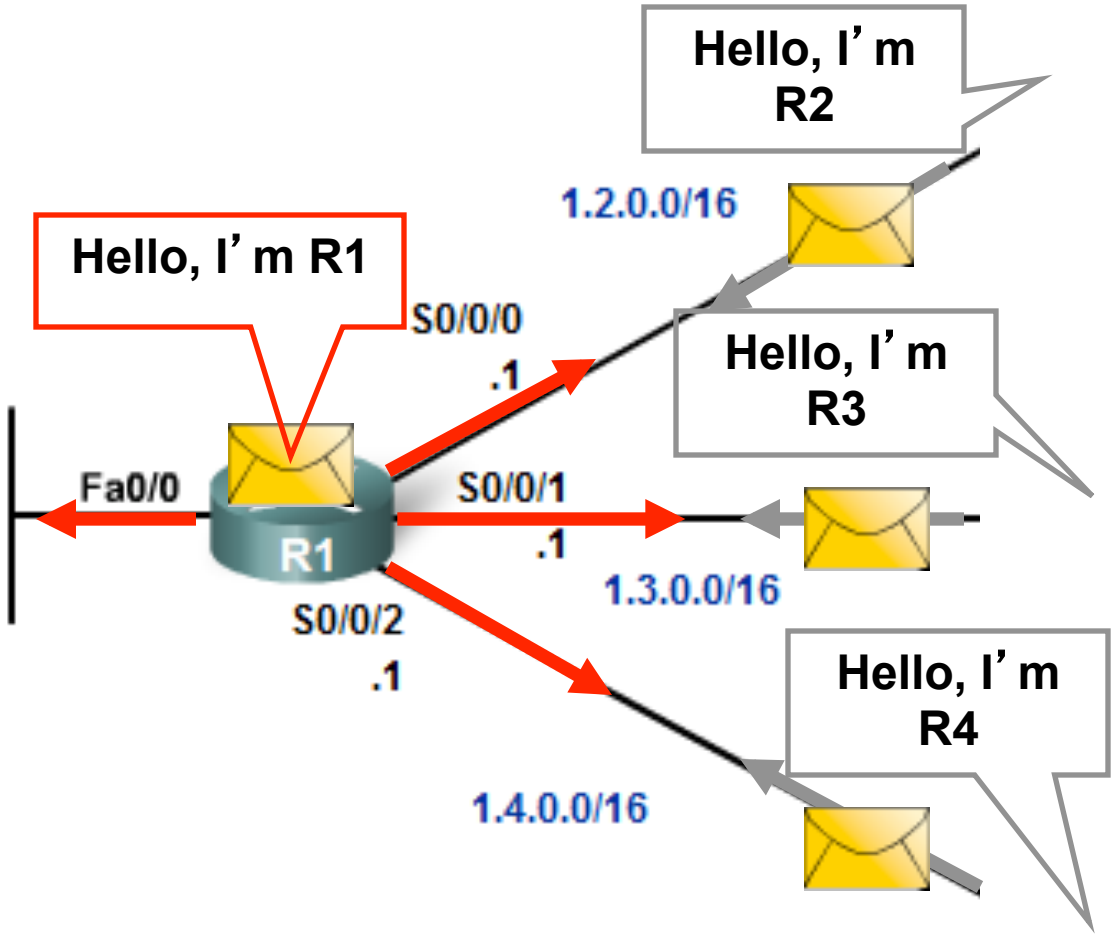
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Étape 2 : ***Chaque routeur est responsable par la découverte des voisins sur un lien***

Utilisation d'un protocole de Hello pour découvrir les voisins sur un lien

Un **voisin** est un routeur qui tourne le même type de protocole de routage à état de liens

Étape 2: Envoi des paquets Hellp



Étape 2: Envoi des paquets Hello

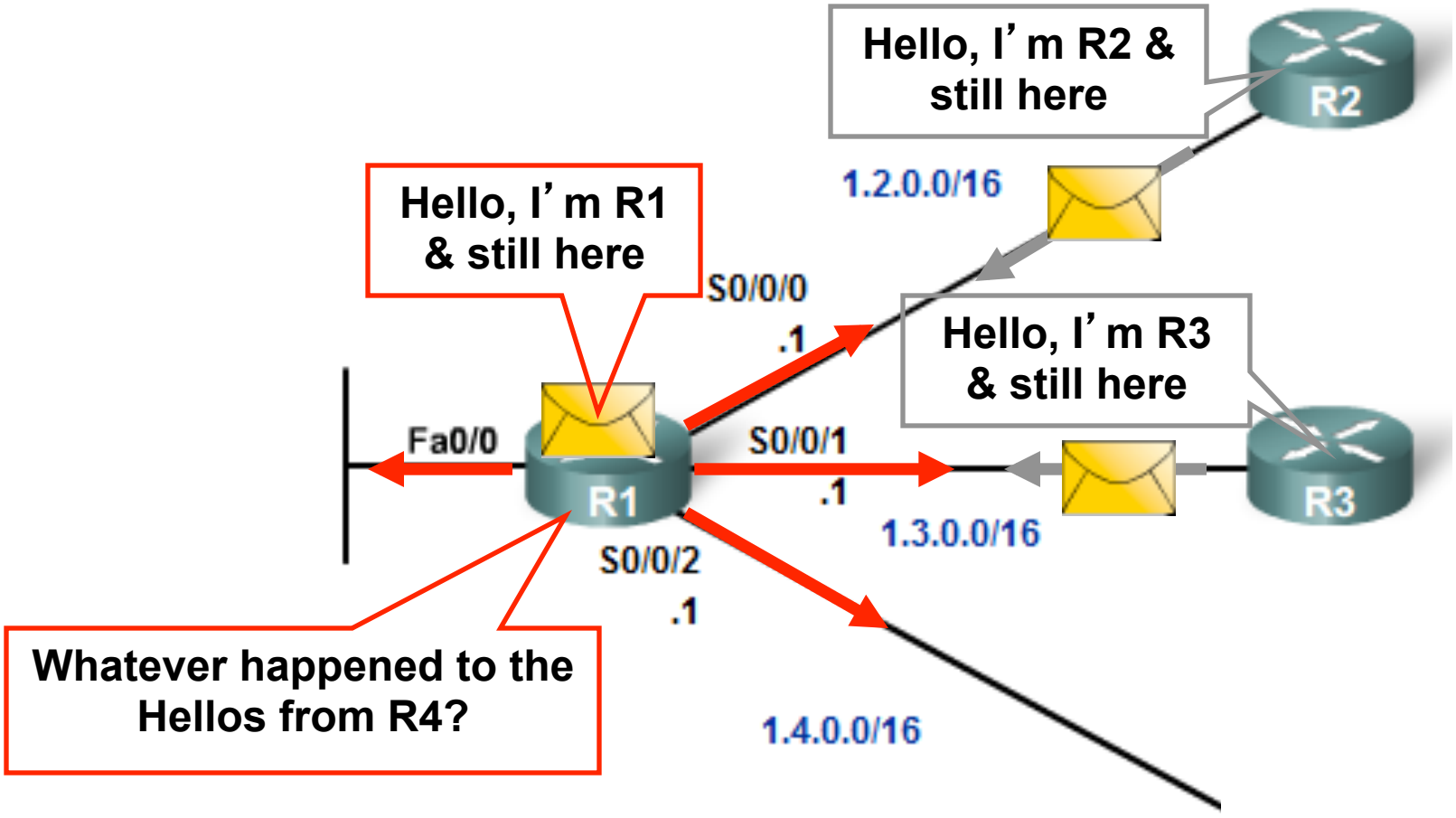
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Paquet Hello

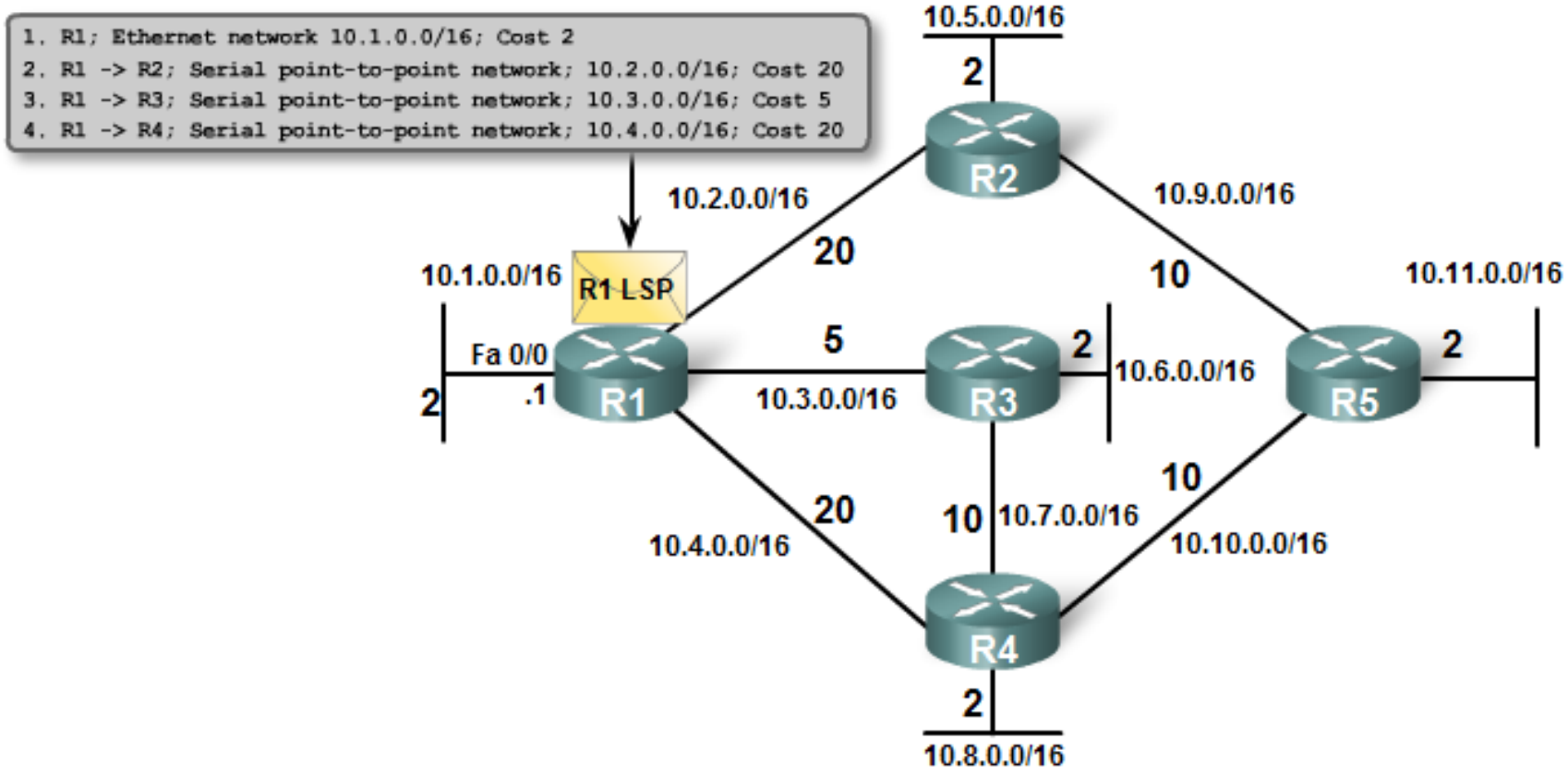
Les paquets Hello ont aussi une fonction “Keepalive”

Si on arrête de recevoir les paquets Hello d’un voisin, ce voisin est considéré comme innateignable et l’adjacence est coupée

Étape 2: Envoi de paquets Hello



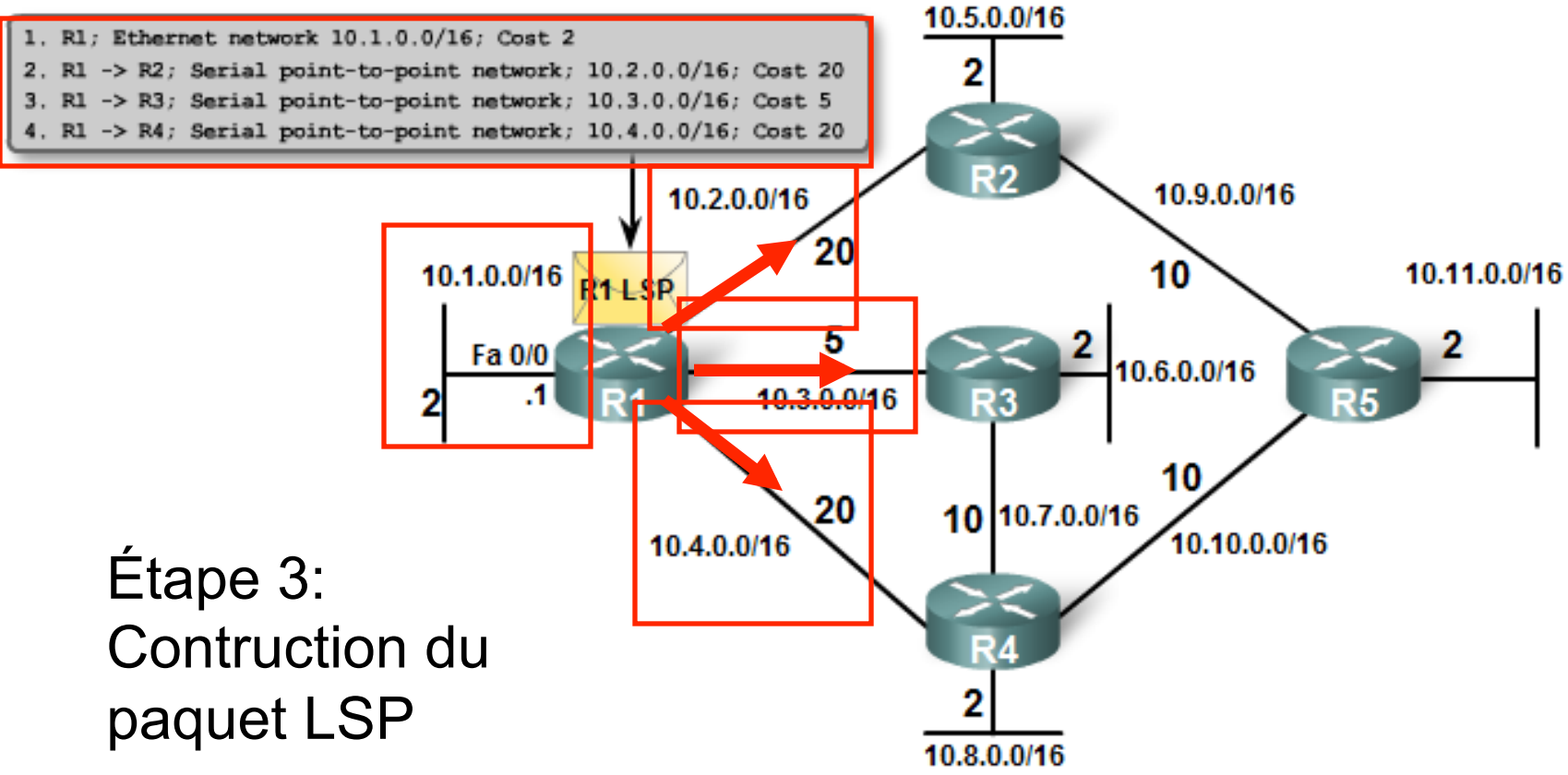
Étape 3 : *Chaque routeur construit son LSP*



Étape 3 : Construction d'une table d'états des liens

Après avoir établi les adjacences (voisinage) le routeur

- Construit ses paquets LSP
  - Inclut les informations sur l'état des liens
- Envoie les LSPs sur toute interface où une adjacence a été établie
  - Dans l'exemple, R1 n'envoie pas des LSPs par l'interface Ethernet



Étape 3:
Contruction du
paquet LSP

Étape 4 : Diffusion des paquets LSP aux voisins

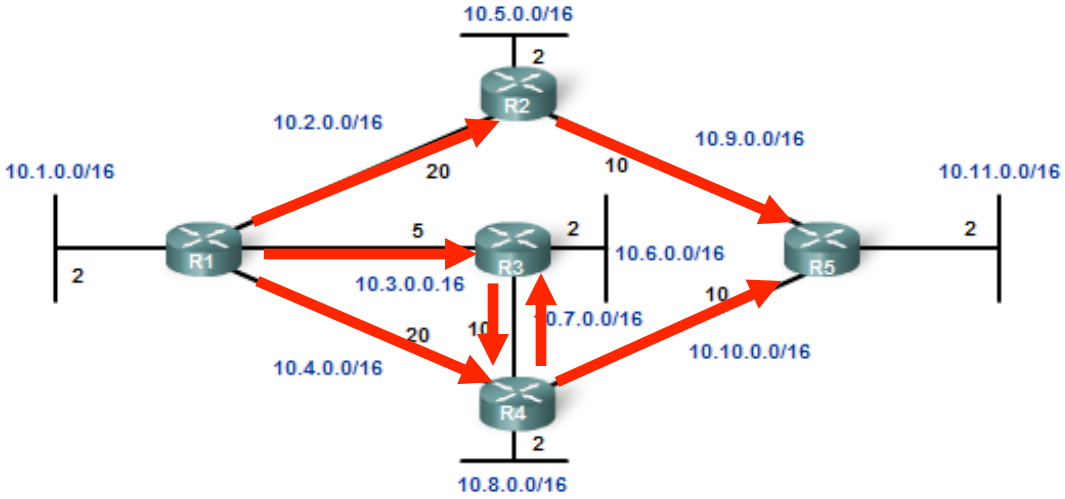
Étape 4 : ***Chaque routeur envoie ses LSPs à tous ses voisins, qui les stockent dans une base de données***

Chaque routeur inonde ses voisins avec l'information LSP

Lorsqu'un routeur reçoit un paquet LSP d'un voisin, il retransmet ce paquet LSP sur toutes ses interfaces, à l'exception de celle d'où le paquet est venu

Effet de inondation (flooding)

Les protocoles à état des liens exécutent l'algorithme SPFaprès la fin de l'inondation



Étape 4 : Diffusion des LSPs aux voisins

Un LSP doit être envoyé seulement quand :

- Pendant le démarrage du routeur ou du protocole de routage
- Toujours où il y a un changement dans la topologie
  - Lien qui tombe
  - Lien activé
  - Nouveau voisin trouvé
  - Déconnexion d'un voisin

Étape 5 : Construction d'une base de données

Après la propagation des LSPs

Chaque routeur a la totalité des LSPs du réseau

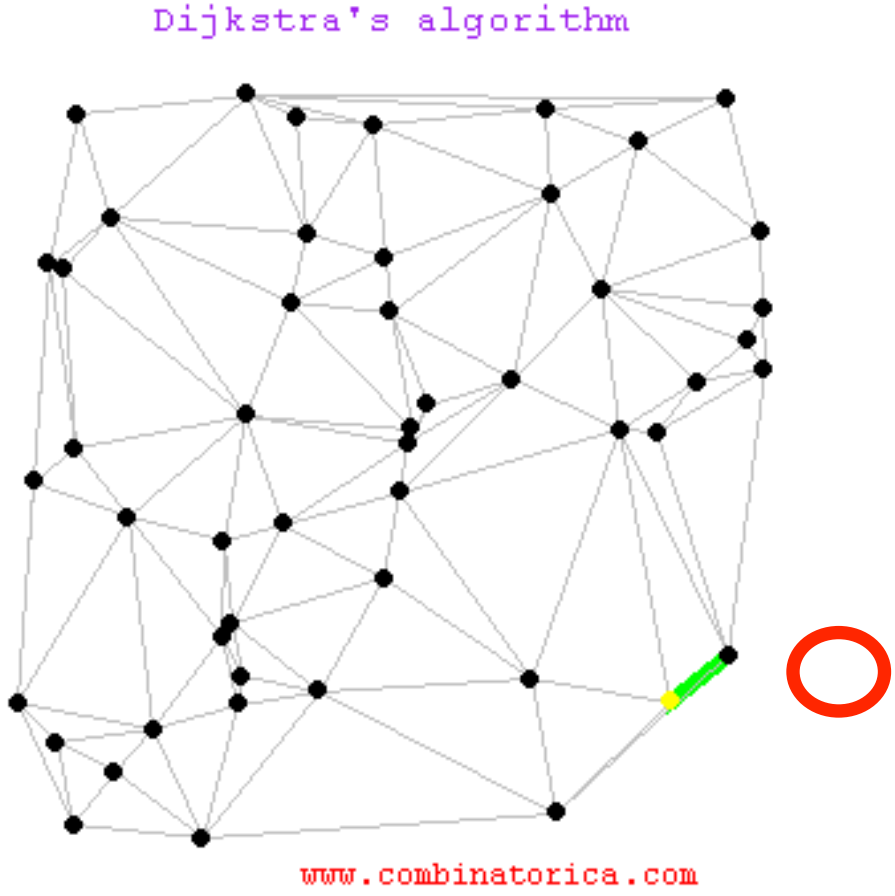
Les LSPs sont stockés dans une base de données d'état des liens

- **Étape 5 (dernière étape) :** ***Chaque routeur utilise la base de données pour calculer le meilleur chemin vers chaque destination***

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2

Exécution de l'algorithme SPF

Chaque routeur dans la zone peut utiliser l'algorithme SPF pour construire l'arbre des chemins utilisée dans le choix des routes



Étape 5 : Construction de la base de données

Avec une base de données complète, R1 peut utiliser l'algorithme SPF pour calculer l'arbre des routes

Arbre SPF pour R1

Destination	Shortest Path	Cost
R2 LAN	R1 to R2	22
R3 LAN	R1 to R3	7
R4 LAN	R1 to R3 to R4	17
R5 LAN	R1 to R3 to R4 to R5	27

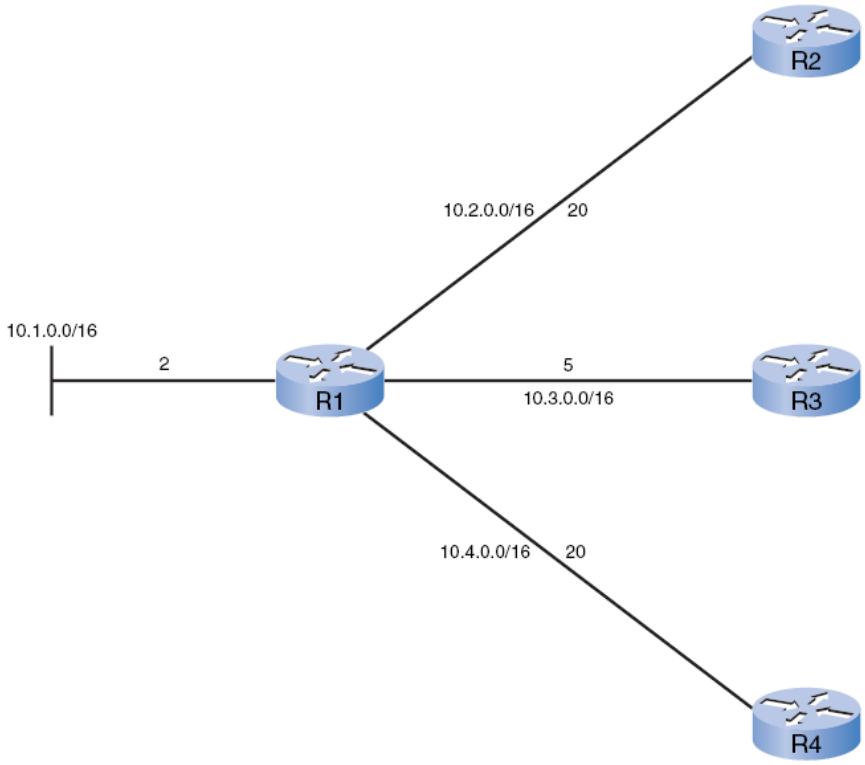
Construction de l'arbre SPF

Au début, l'arbre contient seulement les voisins directement connectés

Grâces aux LSPs des autres routeurs, R1 peut maintenant commencer à construire l'arbre SPF (R1 étant la racine)

Link State Database for R1

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2



R1 analyse les LSPs de R2

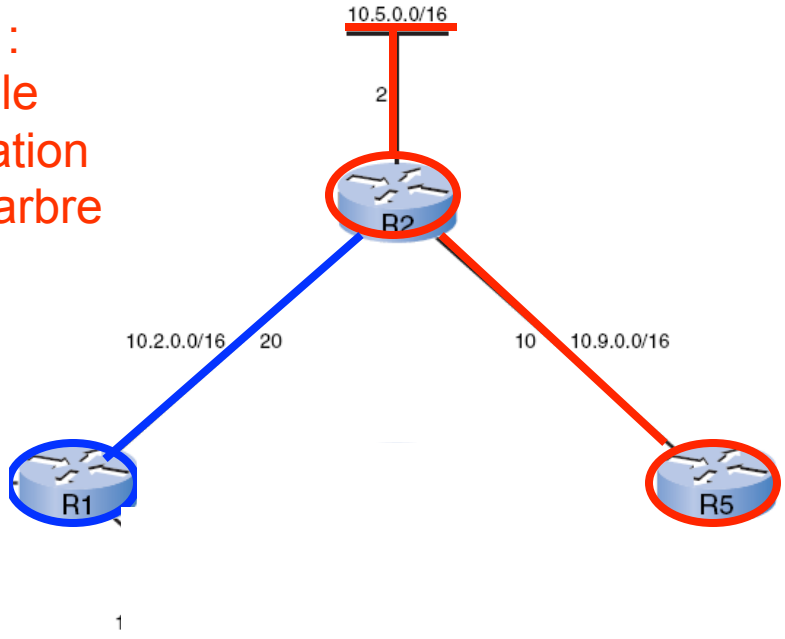
L'algorithme SPF commence par l'analyse des LSPs reçus de R2

- Connecté au voisin R1 sur le réseau 10.2.0.0/16, coût 20
- Connecté au voisin R5 sur le réseau 10.9.0.0/16, coût 10
- A un réseau 10.5.0.0/16, cout 2

Link State Database for R1

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2

Rouge : Nouvelle information dans l'arbre



R1 analyse les LSPs de R3

L'algorithme SPF analyse les LSPs reçus de R3

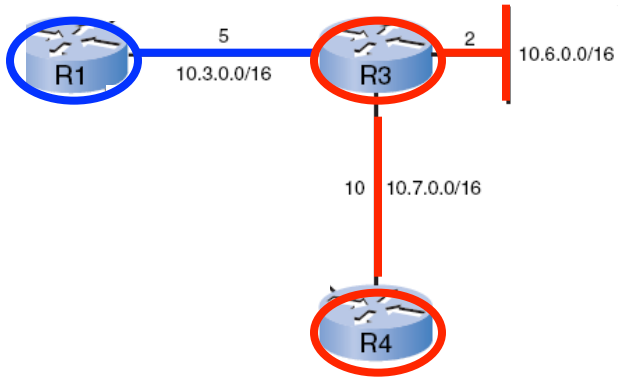
Connecté au voisin R1 sur le réseau 10.3.0.0/16, coût 5

Connecté au voisin R4 sur le réseau 10.7.0.0/16, coût 10

A un réseau 10.6.0.0/16, coût 2

Link State Database for R1

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2



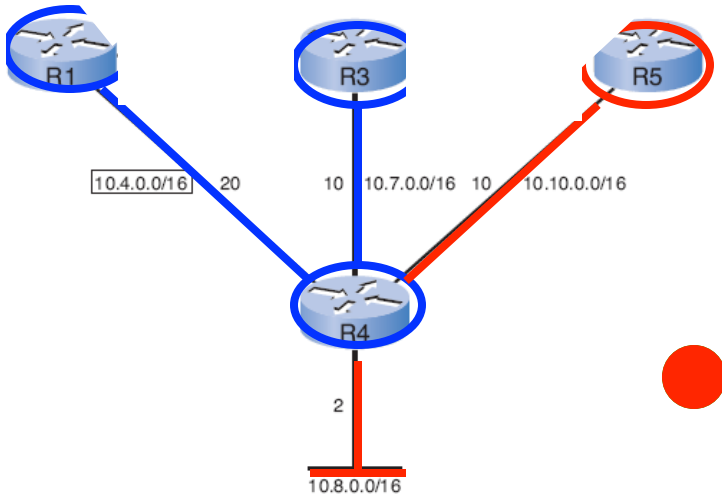
R1 analyse les LSPs de R4

L'algorithme SPF analyse les LSPs reçus de R4

- Connecté au voisin R1 sur le réseau 10.4.0.0/16, coût 20
- Connecté à R3 sur 10.7.0.0/16, côut 10
- Connecté à R5 sur 10.10.0.0/16, coût 10
- A un réseau 10.8.0.0/16, coût 2

Link State Database for R1

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2



R1 analyse les LSPs de R5

L'algorithme SPF analyse les LSPs reçus de R5

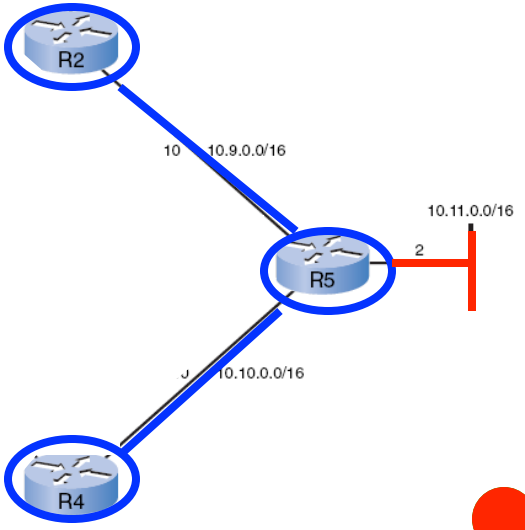
Connecté à R2 sur 10.9.0.0/16, coût 10

Connecté à R4 sur 10.10.0.0/16, coût 10

A un réseau 10.11.0.0/16, coût 2

Link State Database for R1

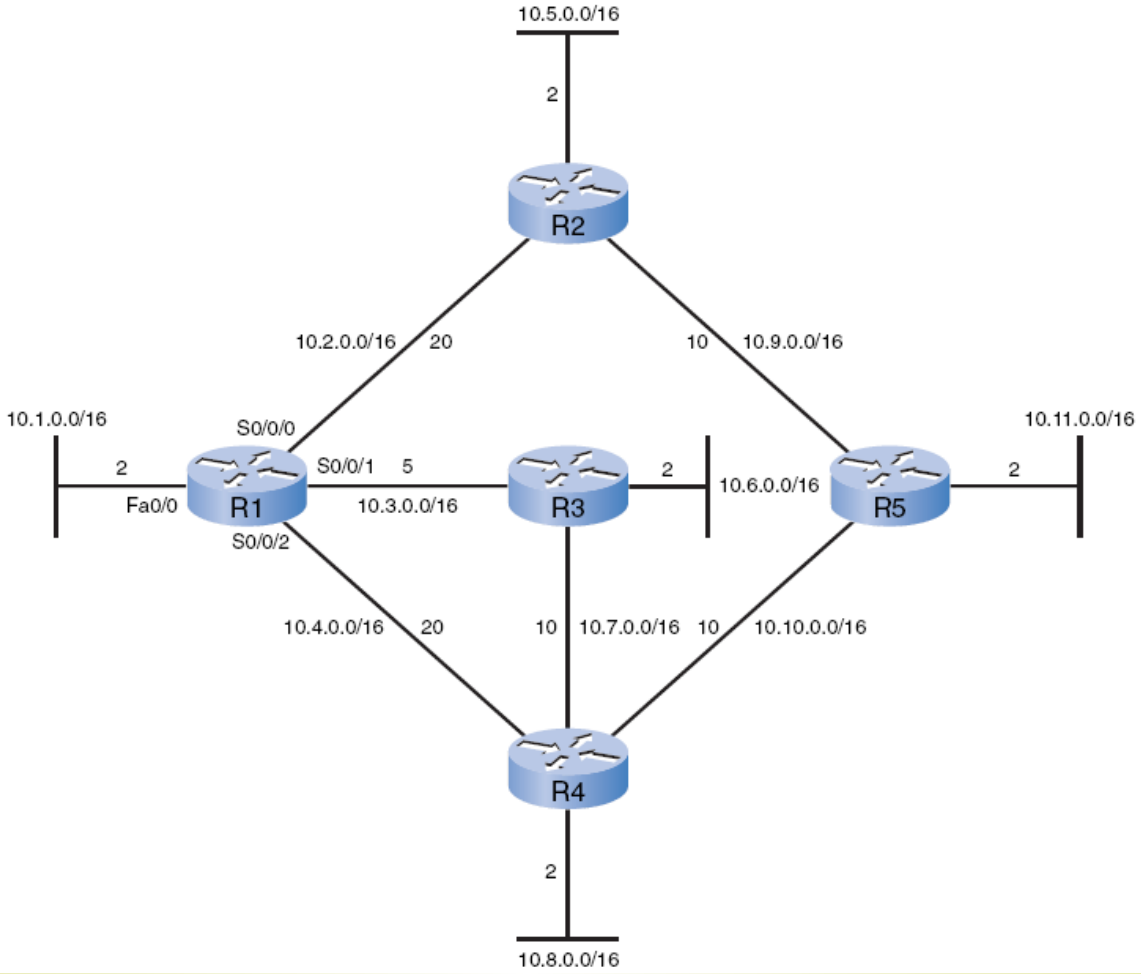
LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2



Arbre SPF

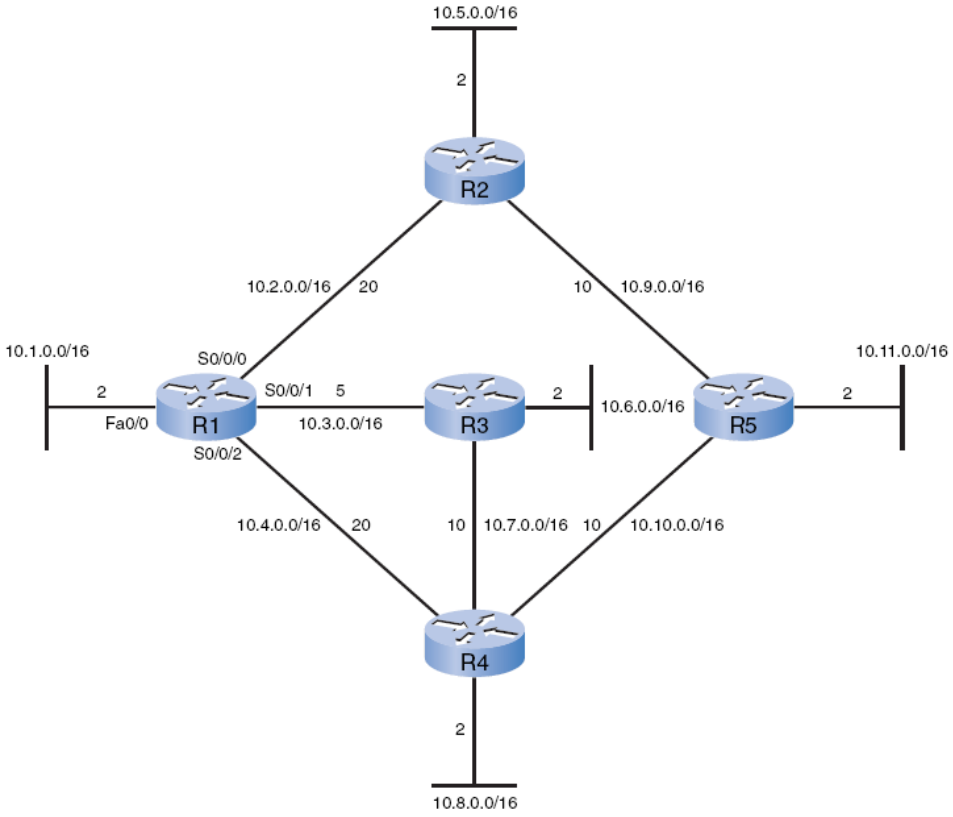
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

R1 a finalement terminé la construction de son arbre SPF



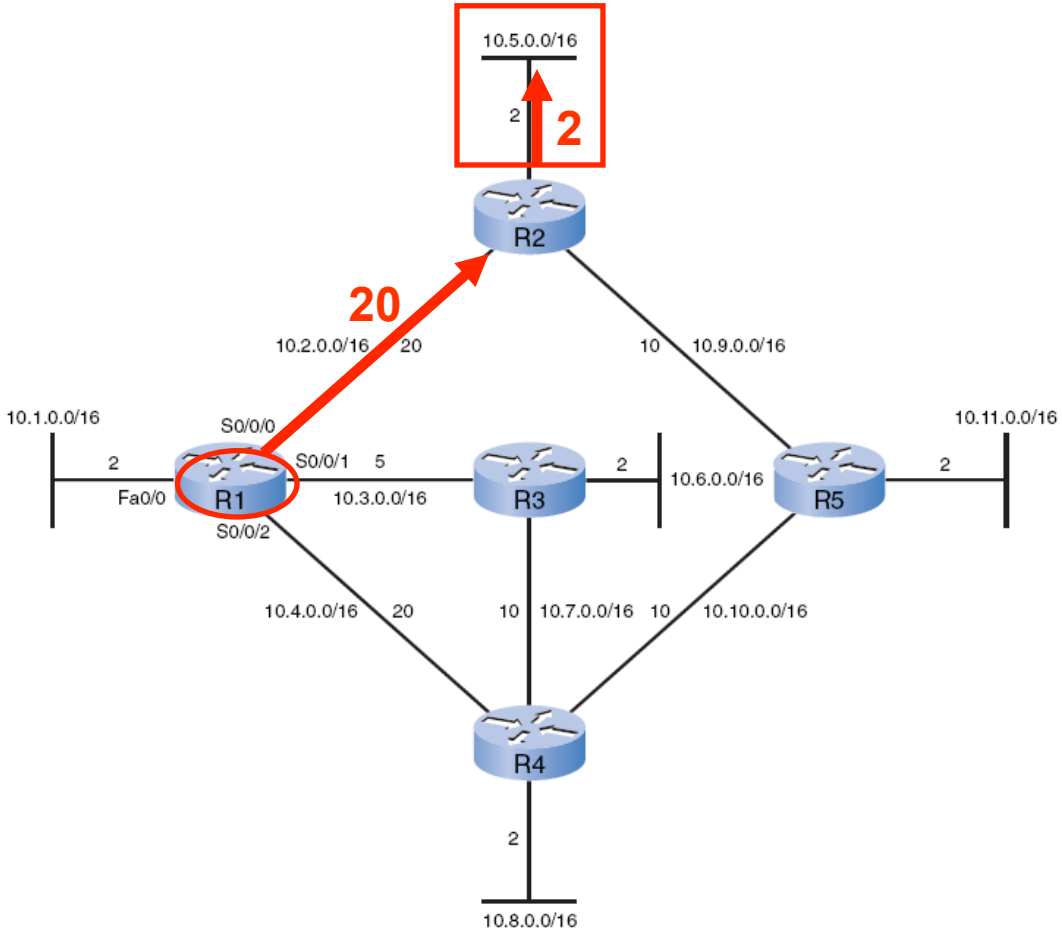
Détermination du Chemin le Plus Court

Avec l'arbre SPF, l'algorithme SPF peut trouver les chemins les plus courts dans le réseau



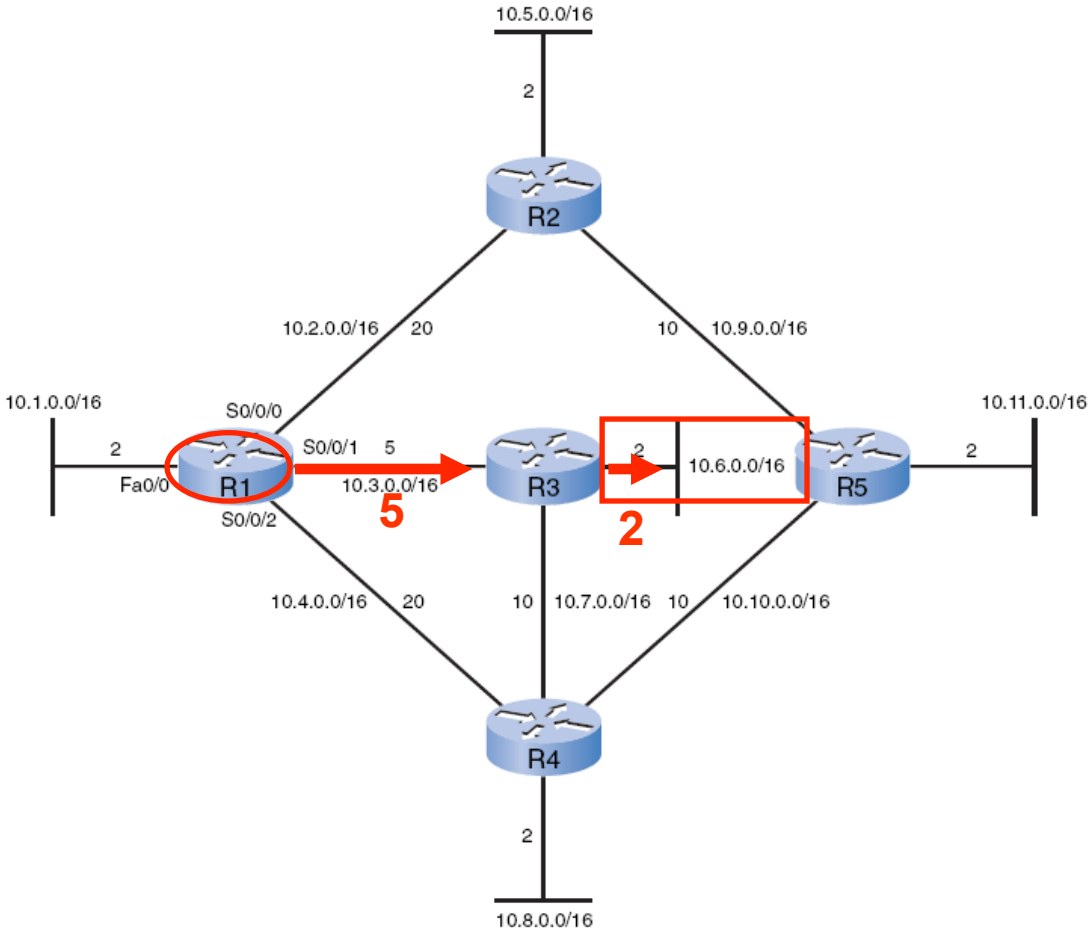
Détermination du Chemin le Plus Court

Réseau 10.5.0.0/16 via R2 Serial 0/0/0 avec un coût 22



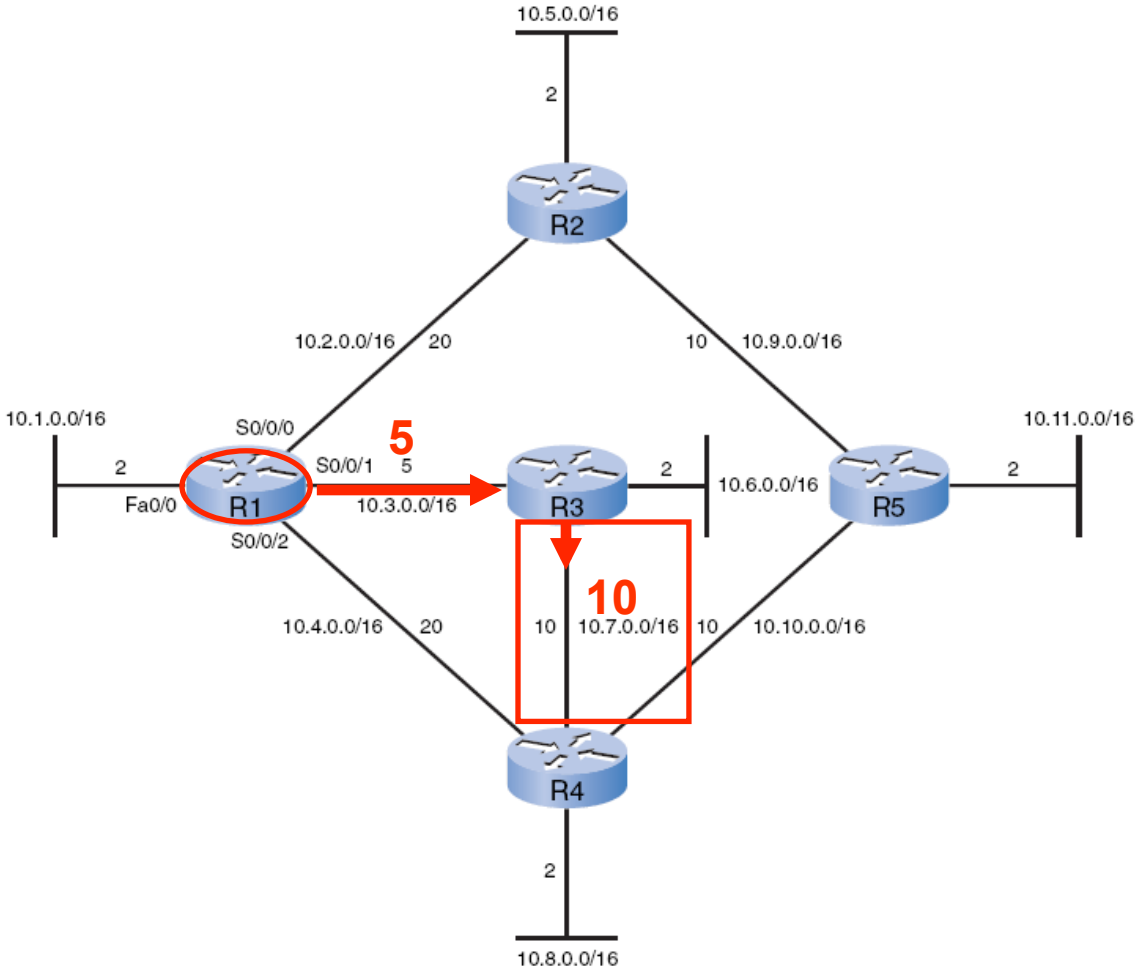
Détermination du Chemin le Plus Court

Réseau 10.6.0.0/16 via R3 Serial 0/0/1 avec un coût 7



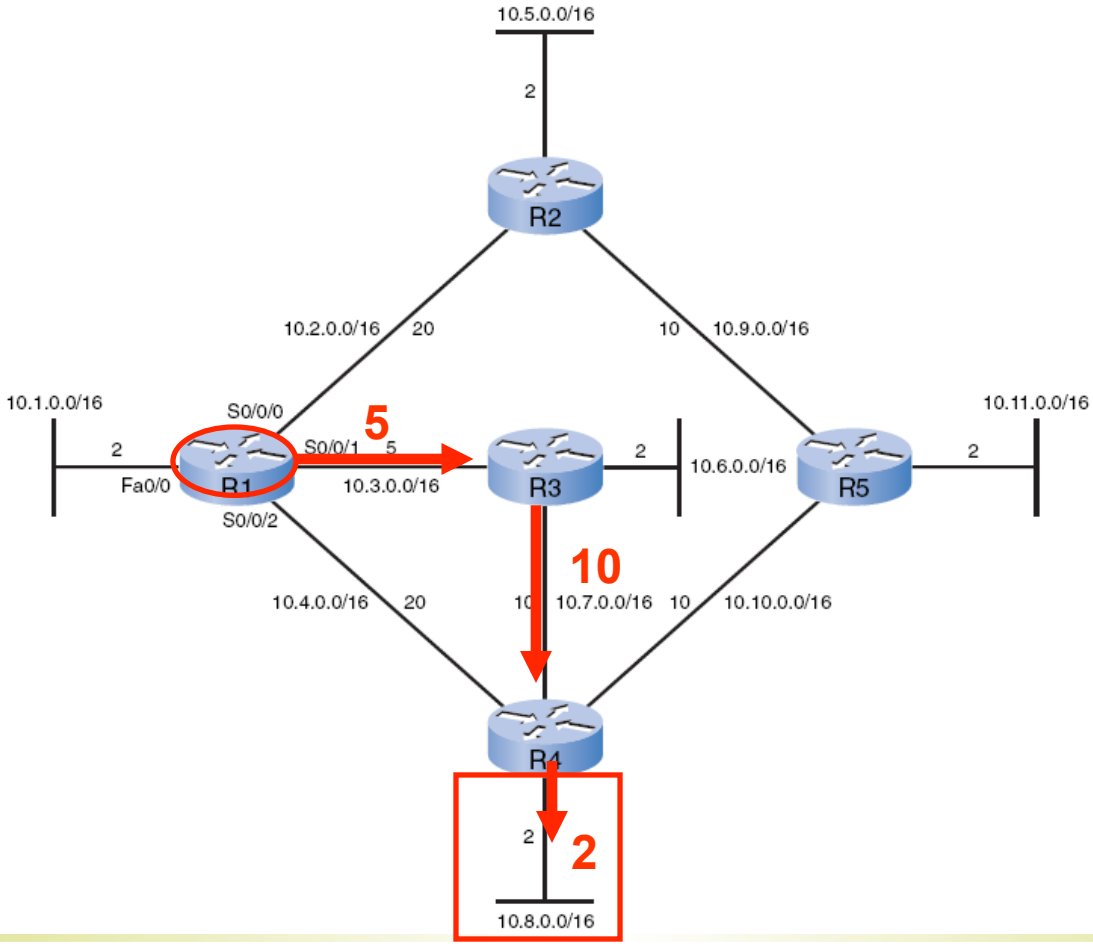
Détermination du Chemin le Plus Court

Réseau 10.7.0.0/16 via R3 Serial 0/0/1 avec un coût 15



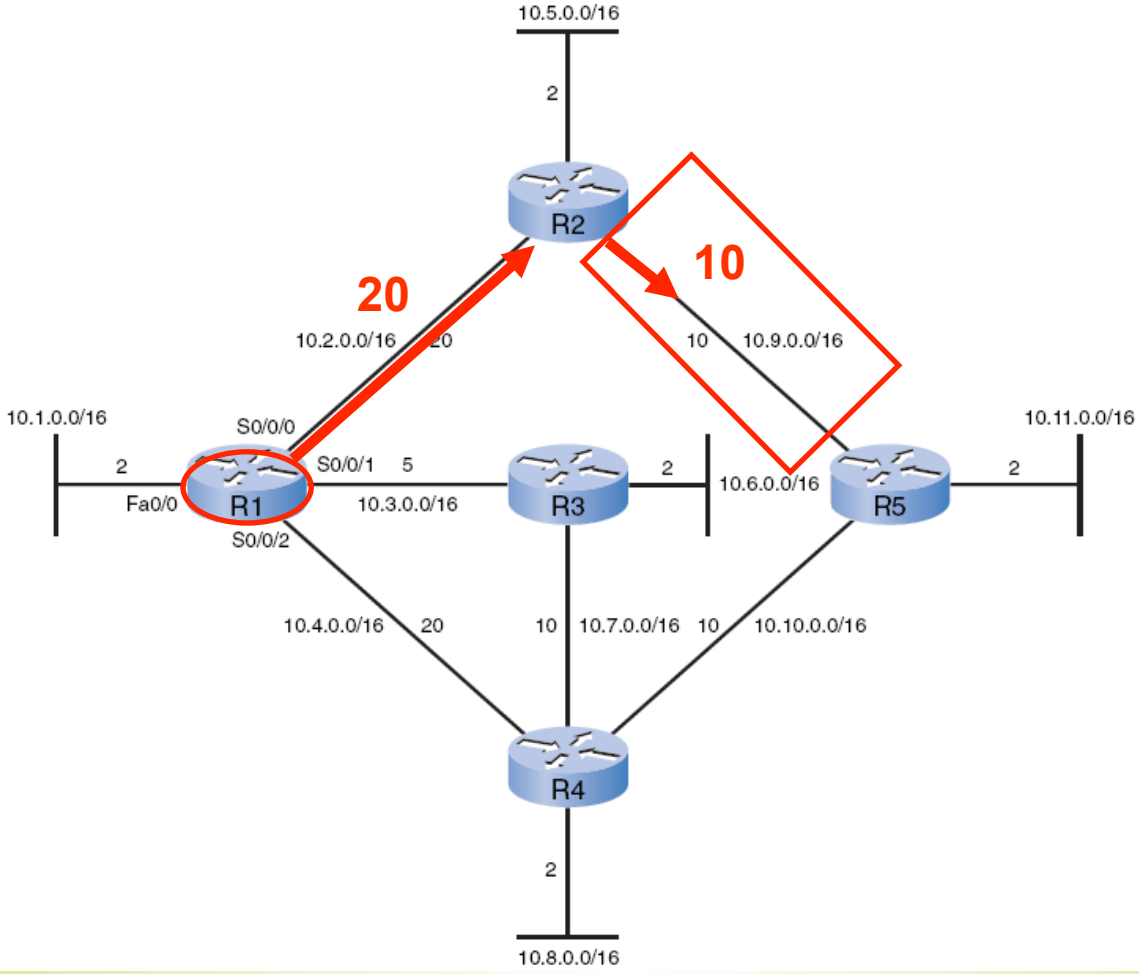
Détermination du Chemin le Plus Court

Réseau 10.8.0.0/16 via R3 Serial 0/0/1 avec un coût 17



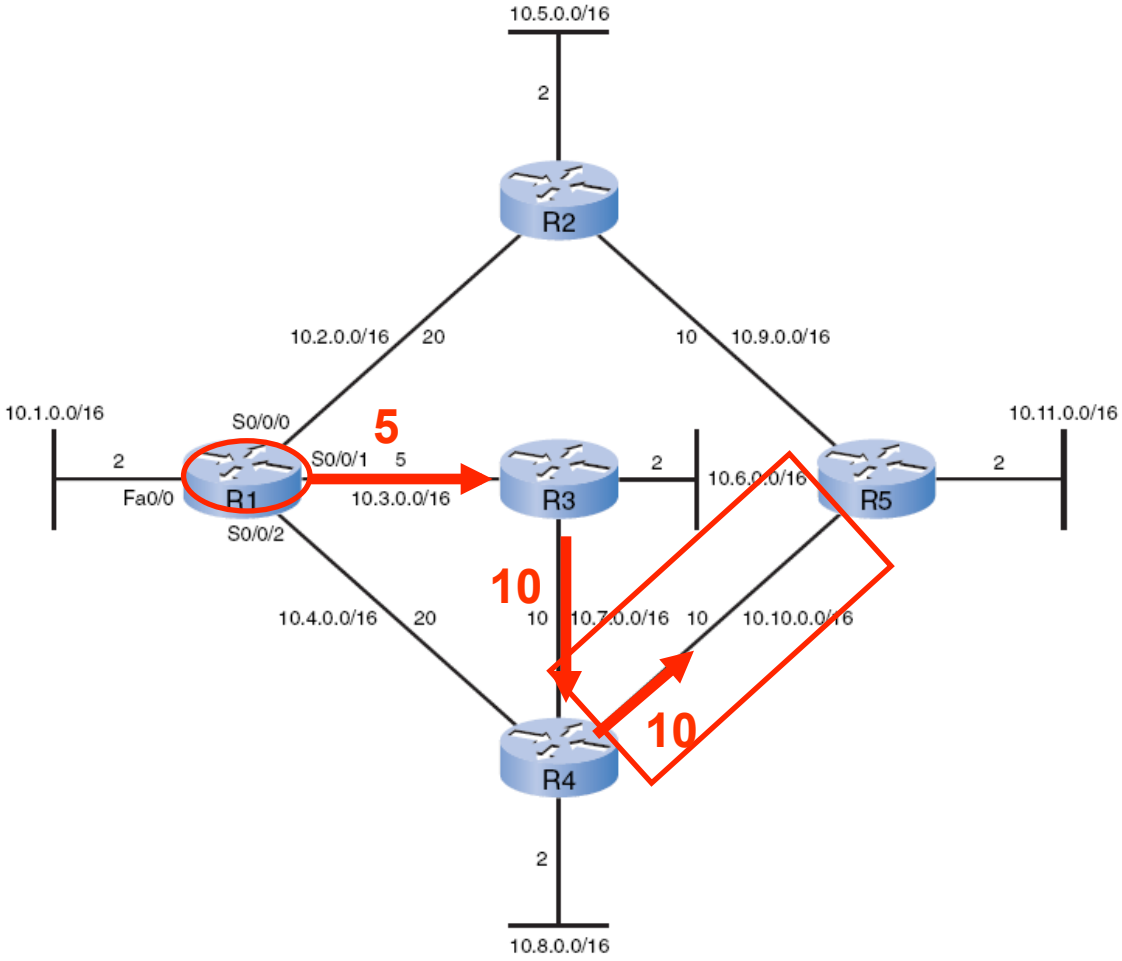
Détermination du Chemin le Plus Court

Réseau 10.9.0.0/16 via R2 Serial 0/0/0 avec un coût 30



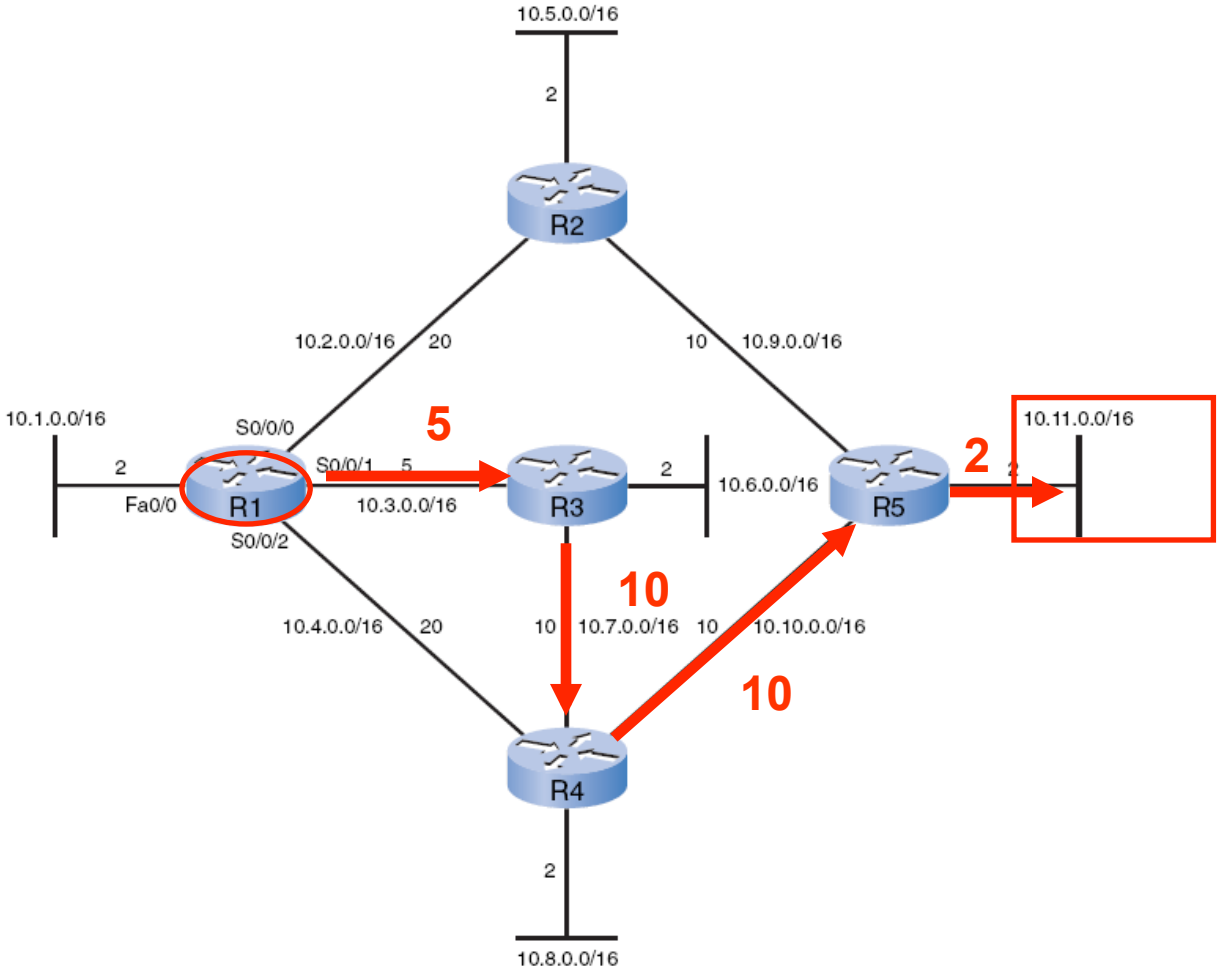
Détermination du Chemin le Plus Court

Réseau 10.10.0.0/16 via R3 Serial 0/0/1 avec un coût 25



Détermination du Chemin le Plus Court

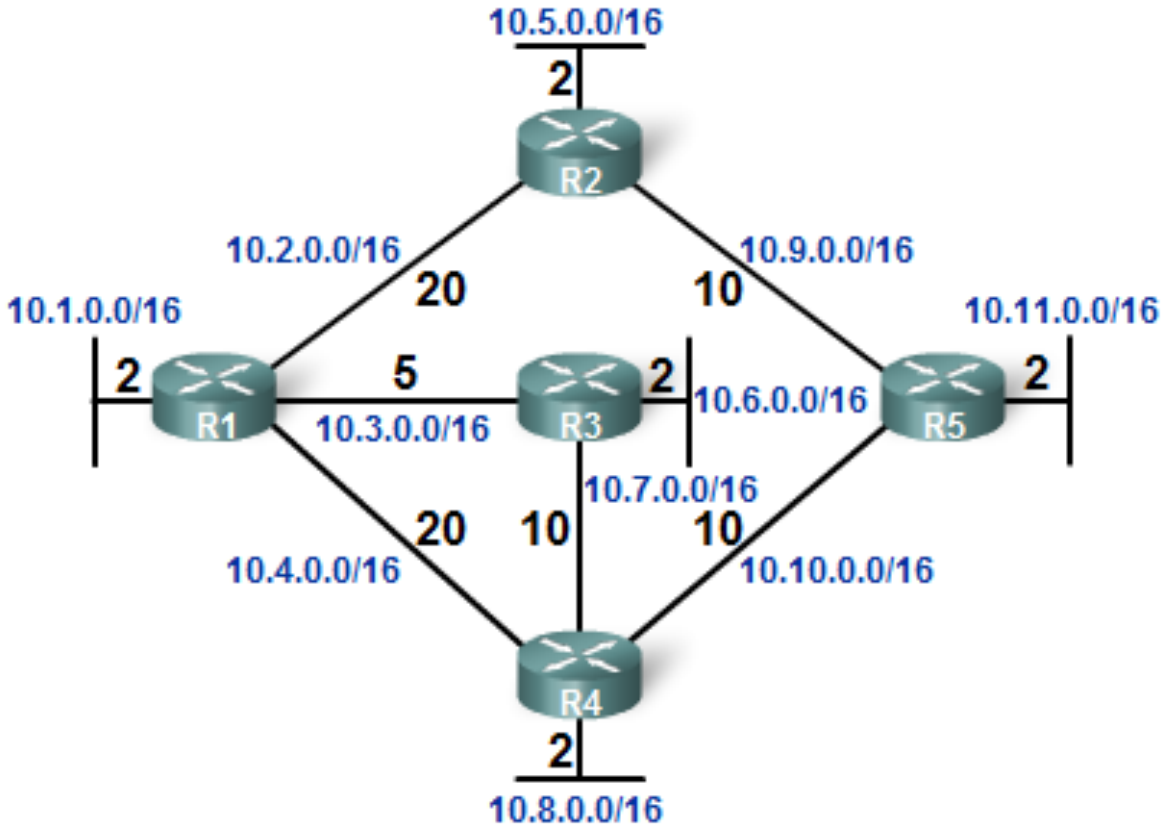
Réseau 10.11.0.0/16 via R3 Serial 0/0/1 avec un coût 27



Détermination du Chemin le Plus Court

Chaque routeur construit son arbre SPF indépendamment des autres routeurs

Les bases de données LSP doivent être identiques partout !



De l'arbre SPF à la table de routage

Les chemins établis peuvent être insérés dans la table de routage

La table de routage indiquera aussi

Les réseaux directement connectés

Les routes des autres sources telles que les routes statiques

Les paquets peuvent être transférés selon ces entrées dans la table de routage

SPF Tree for R1

SPF Information

- Network 10.5.0.0/16 via R2 serial 0/0/0 at a cost of 22
- Network 10.6.0.0/16 via R3 serial 0/0/1 at a cost of 7
- Network 10.7.0.0/16 via R3 serial 0/0/1 at a cost of 15
- Network 10.8.0.0/16 via R3 serial 0/0/1 at a cost of 17
- Network 10.9.0.0/16 via R2 serial 0/0/0 at a cost of 30
- Network 10.10.0.0/16 via R3 serial 0/0/1 at a cost of 25
- Network 10.11.0.0/16 via R3 serial 0/0/1 at a cost of 27

R1 Routing Table

Directly Connected Networks

- 10.1.0.0/16 Directly Connected Network
- 10.2.0.0/16 Directly Connected Network
- 10.3.0.0/16 Directly Connected Network
- 10.4.0.0/16 Directly Connected Network

Remote Networks

- 10.5.0.0/16 via R2 serial 0/0/0, cost = 22
- 10.6.0.0/16 via R3 serial 0/0/1, cost = 7
- 10.7.0.0/16 via R3 serial 0/0/1, cost = 15
- 10.8.0.0/16 via R3 serial 0/0/1, cost = 17
- 10.9.0.0/16 via R2 serial 0/0/0, cost = 30
- 10.10.0.0/16 via R3 serial 0/0/1, cost = 25
- 10.11.0.0/16 via R3 serial 0/0/1, cost = 27

Implémentation d'un protocole à État des Liens

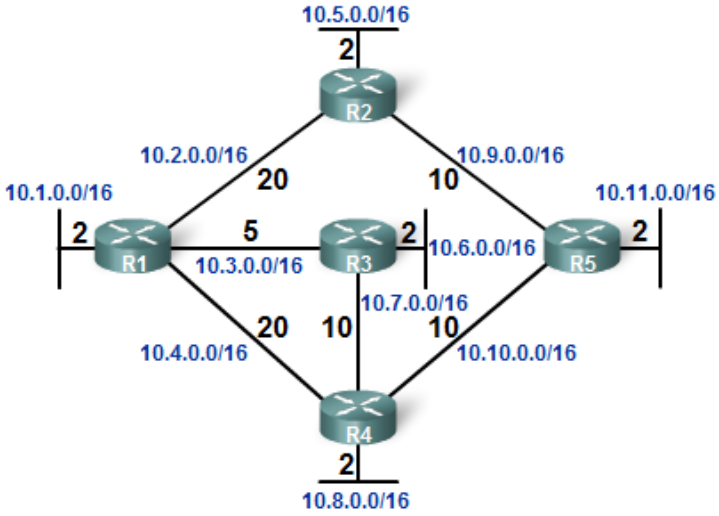
- Avantages d'un protocole à État des Liens
- Besoins d'un protocole à État des Liens
- Comparaison des protocoles à État des Liens

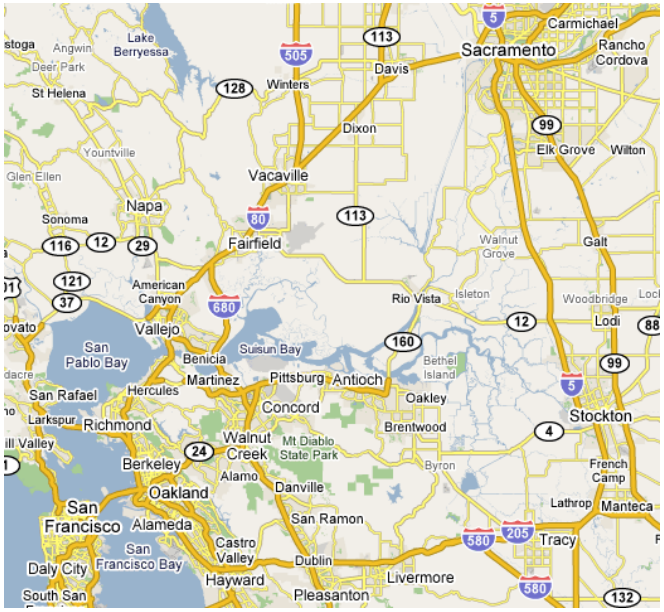
Avantages des protocoles à États des Liens

Construction d'une Carte Topologique

Les protocoles à vecteur de distance n'ont pas une vue globale du réseau

Grâce à l'arbre SPF, les algorithmes à états des liens peuvent calculer indépendamment le meilleur chemin sur tout le réseau





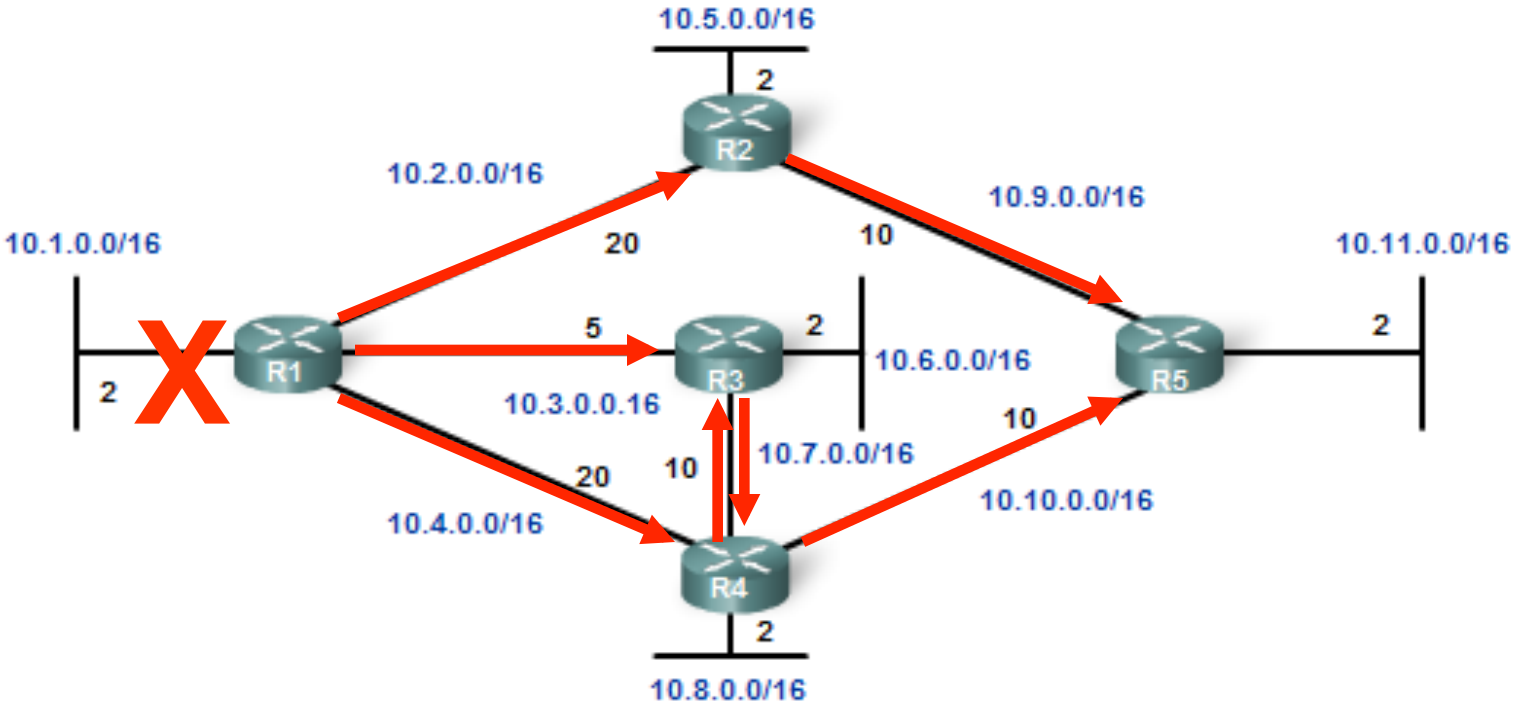
Avantages des protocoles à États des Liens

Convergence Rapide

Les paques LSP sont immédiatement diffusés sur toutes les interfaces (sauf celle d'où venait le LSP)

les hold-down timers ne sont pas nécessaires (technique des protocoles à vecteur de distance pour permettre la stabilisation du réseau)

Les modifications sont diffusées immédiatement grâce aux LSPs



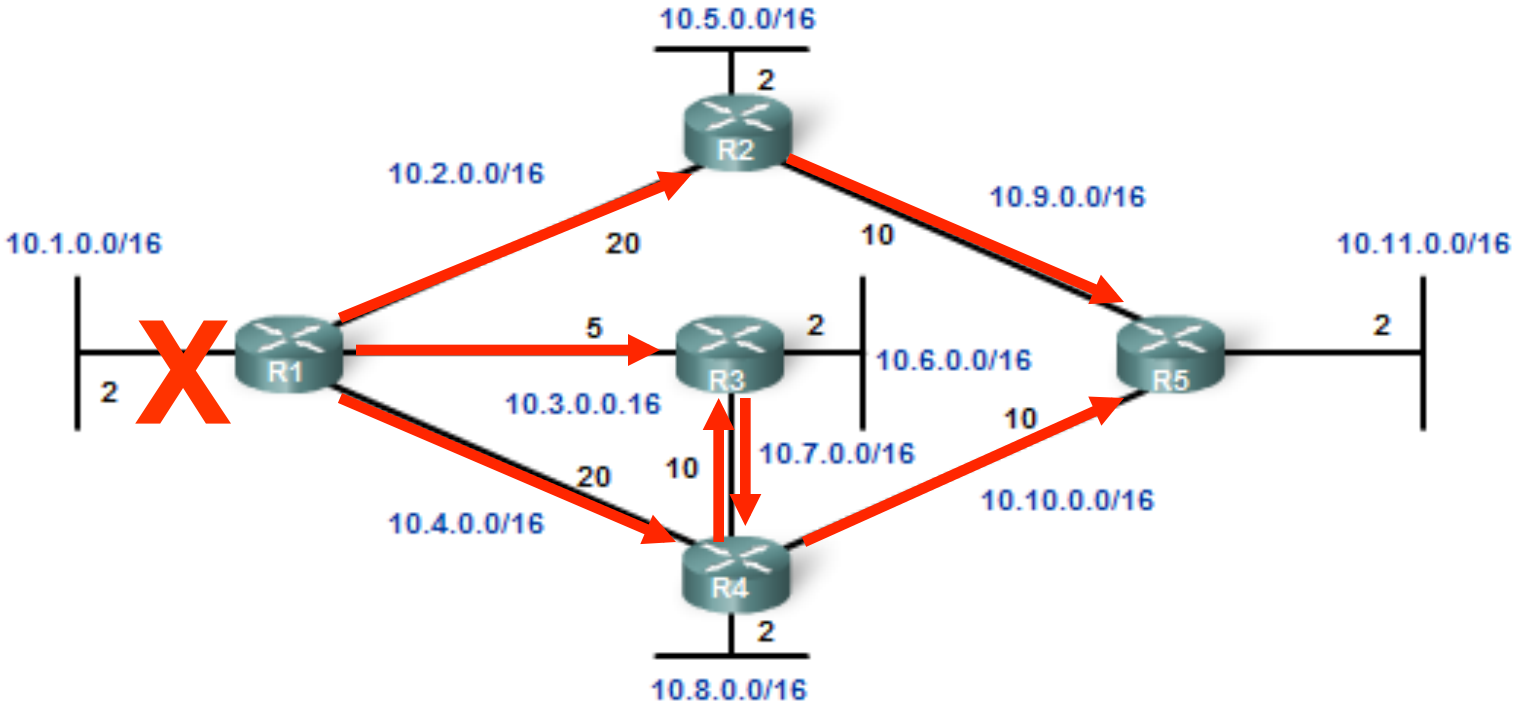
Mises à Jour dirigées par les Événements

Un LSP ne sera renvoyé que si une modification de topologie a lieu

Uniquement les informations concernant le lien

Aucune mise à jour périodique (théorie)

Dans la pratique, les routeurs OSPF renvoyent leurs LSP à chaque 30 minutes



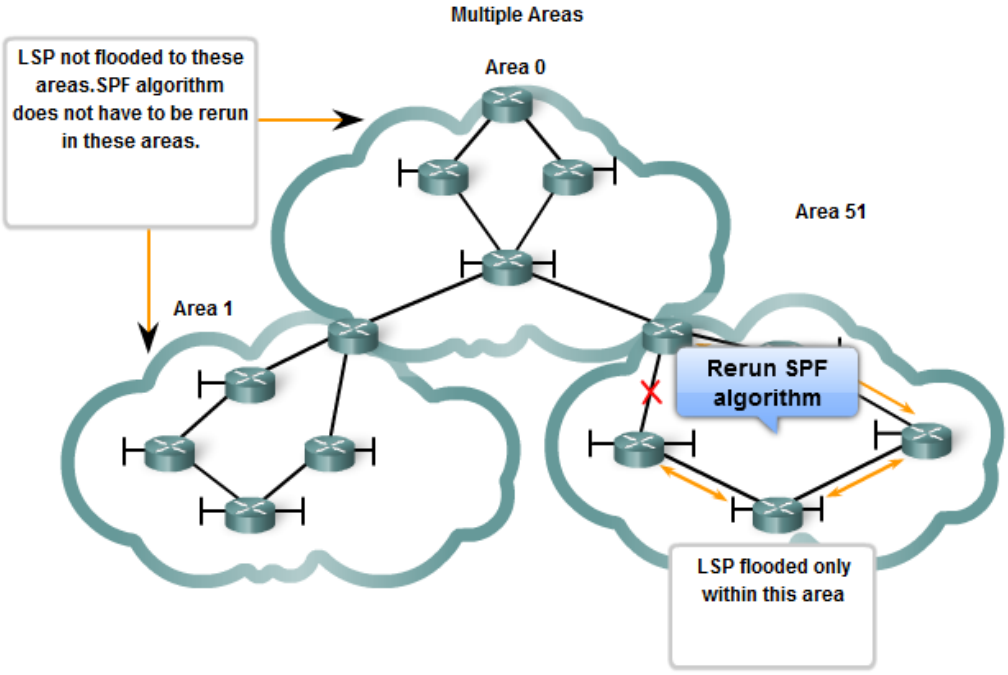
Design Hiérarchique

Les protocoles à états des liens tels que **OSPF** et **IS-IS** utilisent le concept de zones

Meilleur aggrégation des routes

Isolation des problèmes de routage à l'intérieur d'une zone

OSPF Multi-area et IS-IS seront discuté dans la partie CCNP



Besoins d'un protocole à État des Liens

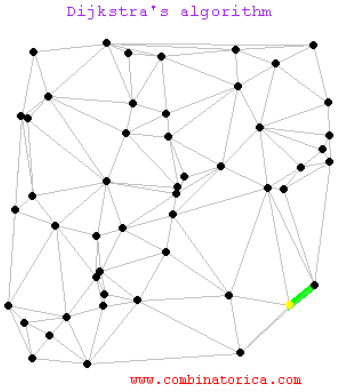
Comparés aux protocoles à Vecteur de Distance, les protocoles à États des Liens requirent plus :

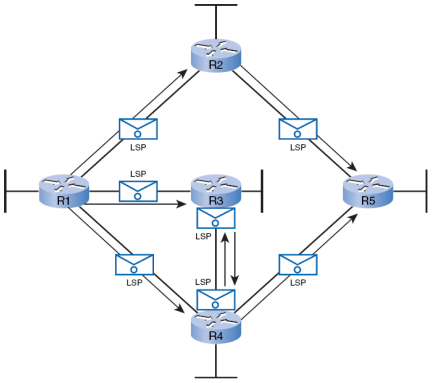
- Mémoire
  - Databases
  - Arbre SPF
- CPU
  - Algorithme SPF
- Bande passante (lors d'un flood)

Mémoire

LSPs from R2	Connected to neighbor R1 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R5 on network 10.9.0.0/16, cost of 10
	Has a network 10.5.0.0/16, cost of 2
LSPs from R3	Connected to neighbor R1 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.7.0.0/16, cost of 10
	Has a network 10.6.0.0/16, cost of 2
LSPs from R4	Connected to neighbor R1 on network 10.4.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.7.0.0/16, cost of 10
	Connected to neighbor R5 on network 10.10.0.0/16, cost of 10
	Has a network 10.8.0.0/16, cost of 2
LSPs from R5	Connected to neighbor R2 on network 10.9.0.0/16, cost of 10
	Connected to neighbor R4 on network 10.10.0.0/16, cost of 10
	Has a network 10.11.0.0/16, cost of 2
R1 link states	Connected to neighbor R2 on network 10.2.0.0/16, cost of 20
	Connected to neighbor R3 on network 10.3.0.0/16, cost of 5
	Connected to neighbor R4 on network 10.4.0.0/16, cost of 20
	Has a network 10.1.0.0/16, cost of 2

CPU





Bande Passante

Comparaison des Protocoles à États des Liens

Deux protocoles sont utilisés couramment

- **Open Shortest Path First (OSPF)**
- **Intermediate System–to–Intermediate System (IS-IS)**

Open Shortest Path First (OSPF)

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

OSPF a été crée par l' IETF (Internet Engineering Task Force) OSPF Working Group, qui est toujours en activité.

Le développement d'OSPF commence en 1987, et deux version sont en utilisation :

OSPFv2: OSPF pour les réseaux IPv4 (RFC 1247 and RFC 2328)

OSPFv3: OSPF pour les réseaux IPv6 (RFC 2740)

Intermediate System–to–Intermediate System (IS-IS)

IS-IS a été crée par l' ISO (International Organization for Standardization) et est décrit dans la norme ISO 10589.

La première version a été implémentée par DEC (Digital Equipment Corporation) et est connu comme DECnet Phase V.

IS-IS a été crée pour la pile de protocoles OSI, pas pour TCP/IP

Plus tard, Integrated IS-IS, (Dual IS-IS) a rajouté le support à TCP/IP