Introduction au routage dynamique OSPF avec Quagga

Philippe Latu
philippe.latu(at)inetdoc.net

https://www.inetdoc.net

Résumé

L'objectif de ce support de travaux pratiques est d'étudier le protocole de routage dynamique OSPF. Cette illustration s'appuie sur une topologie minimale très classique : le triangle. L'originalité consiste à utiliser les VLANs pour distinguer la topologie physique (l'étoile) de la topologie logique (le triangle). Cette version du support utilise la suite de démons de routage Quagga.

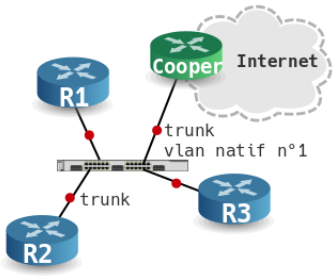


Table des matières

1. Copyright et Licence

1.1. Méta-information

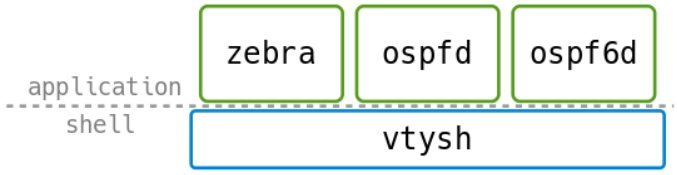
Ce document est écrit avec DocBook XML sur un système Debian GNU/Linux. Il est disponible en version imprimable au format PDF : `interco.ospf-quagga.qa.pdf`.

Toutes les commandes utilisées dans ce document ne sont pas spécifiques à une version particulière des systèmes UNIX ou GNU/Linux. C'est la distribution Debian GNU/Linux qui est utilisée pour les tests présentés. Voici une liste des paquets contenant les commandes :

- procps - Utilitaires pour le système de fichiers /proc
- iproute2 - Outils de contrôle du trafic et du réseau
- ifupdown - Outils de haut niveau pour configurer les interfaces réseau
- iputils-ping - Outils pour tester l'accessibilité de noeuds réseaux
- quagga - BGP/OSPF/RIP routing daemon

La suite de démons de routage Quagga couvre la totalité des protocoles de routage dynamiques. Ici, on se concentre sur le protocole OSPF. C'est la raison pour laquelle, seuls trois paquets sont utiles.

- quagga-core : démon de routage statique zebra et console unifiée vtysh.
- quagga-ospfd : démon du protocole de routage OSPFv2 pour IPv4.
- quagga-ospf6d : démon du protocole de routage OSPFv3 pour IPv6.



1.2. Conventions typographiques

Tous les exemples d'exécution des commandes sont précédés d'une invite utilisateur ou prompt spécifique au niveau des droits utilisateurs nécessaires sur le système.

- Toute commande précédée de l'invite `$` ne nécessite aucun privilège particulier et peut être utilisée au niveau utilisateur simple.
- Toute commande précédée de l'invite `#` nécessite les privilèges du super-utilisateur.

2. Topologie réseau étudiée

La topologie réseau étudiée peut être présentée sous deux formes distinctes : logique et physique.

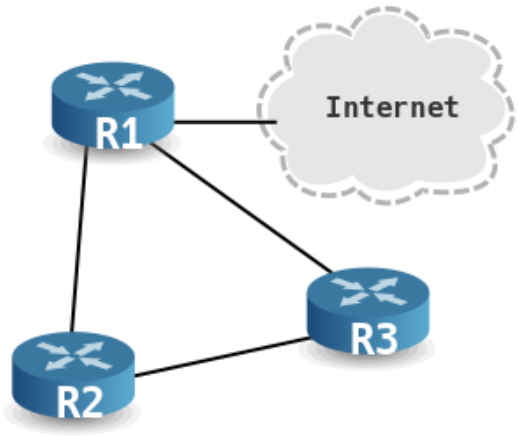
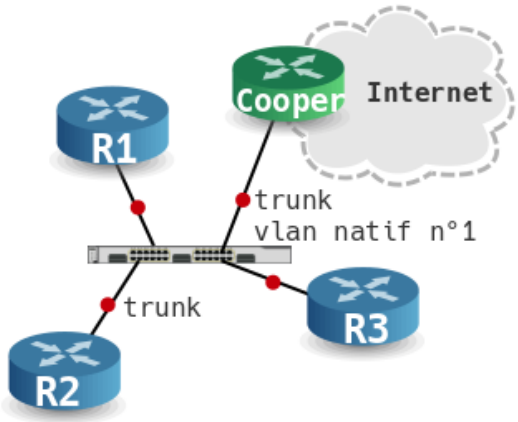
Topologie logique

On retrouve un grand classique dans l'introduction aux protocoles de routage dynamiques : le triangle. Tous les liens sont de type LAN.

Topologie physique

On s'appuie sur le support Routage Inter-VLAN pour constituer une topologie physique à base de réseaux locaux virtuels ou VLANs. On fait correspondre à chaque lien de la topologie logique en triangle un numéro de VLAN défini.

Tableau 1. Topologies type

Topologie logique	Topologie physique
	

Après avoir mis en œuvre la topologie physique en s'appuyant sur le support de la séance de travaux pratiques précédente : Routage Inter-VLAN, on implante les démons de routage OSPF sur les trois routeurs R1, R2 et R3.

Cette séance se limite à l'étude du routage dynamique à l'intérieur d'une aire unique. La seule «frontière» de communication inter-aires visible est constituée par le lien vers l'Internet. Cette route par défaut sera redistribuée via OSPF par le routeur R1 aux autres routeurs. On verra alors un exemple de route externe dans les bases de données OSPF.

On profite aussi de cette introduction pour employer une technique très répandue pour ajouter «artificiellement» des entrées de tables de routage en s'appuyant sur des interfaces virtuelles de type dummy équivalentes à des interfaces de boucle locale.

Pour les besoins de rédaction des questions et réponses de ce support, la topologie a été mise en œuvre sur des machines virtuelles KVM avec le commutateur Open vSwitch. Les éléments de correction des questions dépendent donc de cette mise en œuvre. Pour la séance de travaux pratiques «réelle», il faut se conformer strictement au plan d'adressage fourni ci-après.

Dans les manipulations à suivre, le raccordement au réseau d'interconnexion avec les routeurs de l'infrastructure de travaux pratiques est imposé. Le raccordement au commutateur de la couche distribution de la salle utilise le port «de numéro le plus élevé» de chaque commutateur de couche accès. Ces liens montants doivent être configurés en modetrunk en utilisant le VLAN natif numéro 1.

Point important, la lecture de la section «Plan d'adressage» du document Architecture réseau des travaux pratiques donne l'adresse des deux routeurs connectés à l'Internet.

- Routeur `cooper.stri` : `172.16.16.1/20` et `2001:678:3fc:4::1/64`
- Routeur `casper.stri` : `172.16.16.2/20` et `2001:678:3fc:4::2/64`

2.1. Plan d'adressage salle 211

Tableau 2. Affectation d'un commutateur par groupe de 3 routeurs - salle 211

Groupe	Commutateur
1	asw04-211.infra.stri
2	asw05-211.infra.stri
3	asw06-211.infra.stri
4	asw07-211.infra.stri

Voici, pour chaque groupe de trois routeurs, le tableau d'affectation des rôles, des numéros de VLANs et des adresses IP.

Tableau 3. Salle 211 - Groupe 1

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
christophsis	R1	OSPFv2 : 0.211.4.11 OSPFv3 : 0.211.6.11	4	eth0.4	172.16.17.10/20
					2001:678:3fc:4::6ae
			401	eth0.401	10.1.201.1/26
					2001:678:3fc:191::1
			405	eth0.405	10.1.205.1/26
					2001:678:3fc:195::1
			-	dummy0	10.1.200.1/29
					2001:678:3fc:190::1
corellia	R2	OSPFv2 : 0.211.4.12 OSPFv3 : 0.211.6.12	401	eth0.401	10.1.201.2/26
					2001:678:3fc:191::2
			403	eth0.403	10.1.203.2/26
					2001:678:3fc:193::2
			-	veth0:veth1	10.1.202.1/30 : 10.1.202.2/30
					2001:678:3fc:192::1 -> 2001:678:3fc:192::2
delaya	R3	OSPFv2 : 0.211.4.13 OSPFv3 : 0.211.6.13	405	eth0.405	10.1.205.3/26
					2001:678:3fc:195::3
			403	eth0.403	10.1.203.3/26
					2001:678:3fc:193::3
			-	veth0:veth1	10.1.204.1/30 : 10.1.204.2/30
					2001:678:3fc:194::1 -> 2001:678:3fc:194::2

Tableau 4. Salle 211 - Groupe 2

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
kashyyyk	R1	OSPFv2 : 0.211.4.21 OSPFv3 : 0.211.6.21	4	eth0.4	172.16.17.11/20
					2001:678:3fc:4::6af
			411	eth0.411	10.2.211.1/26
					2001:678:3fc:19b::1
			415	eth0.415	10.2.215.1/26
					2001:678:3fc:19f::1
			-	dummy0	10.2.210.1/29
					2001:678:3fc:19a::1
korriban	R2	OSPFv2 : 0.211.4.22 OSPFv3 : 0.211.6.22	411	eth0.411	10.2.211.2/26
					2001:678:3fc:19b::2
			413	eth0.413	10.2.213.2/26
					2001:678:3fc:19d::2
			-	veth0:veth1	10.2.212.1/30 : 10.2.212.2/30
					2001:678:3fc:19c::1 -> 2001:678:3fc:19c::2
kessel	R3	OSPFv2 : 0.211.4.23 OSPFv3 : 0.211.6.23	415	eth0.415	10.2.215.3/26
					2001:678:3fc:19f::3
			413	eth0.413	10.2.213.3/26
					2001:678:3fc:19d::3
			-	veth0:veth1	10.2.214.1/30 : 10.2.214.2/30
					2001:678:3fc:19e::1 -> 2001:678:3fc:19e::2

Tableau 5. Salle 211 - Groupe 3

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
mygeeto	R1	OSPFv2 : 0.211.4.31 OSPFv3 : 0.211.6.31	4	eth0.4	172.16.17.12/20
					2001:678:3fc:4::6b0
			421	eth0.421	10.3.221.1/26
					2001:678:3fc:1a5::1
			425	eth0.425	10.3.225.1/26
					2001:678:3fc:1a9::1
			-	dummy0	10.3.220.1/29
					2001:678:3fc:1a4::1
nelvaan	R2	OSPFv2 : 0.211.4.32 OSPFv3 : 0.211.6.32	421	eth0.421	10.3.221.2/26
					2001:678:3fc:1a5::2
			423	eth0.423	10.3.223.2/26
					2001:678:3fc:1a7::2
			-	veth0:veth1	10.3.222.1/30 : 10.3.222.2/30
					2001:678:3fc:1a6::1 -> 2001:678:3fc:1a6::2
rattatak	R3	OSPFv2 : 0.211.4.33 OSPFv3 : 0.211.6.33	425	eth0.425	10.3.225.3/26
					2001:678:3fc:1a9::3
			423	eth0.423	10.3.223.3/26
					2001:678:3fc:1a7::3
			-	veth0:veth1	10.3.224.1/30 : 10.3.224.2/30
					2001:678:3fc:1a8::1 -> 2001:678:3fc:1a8::2

Tableau 6. Salle 211 - Groupe 4

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
saleucami	R1	OSPFv2 : 0.211.4.41 OSPFv3 : 0.211.6.41	4	eth0.4	172.16.17.13/20
					2001:678:3fc:4::6b1
			431	eth0.431	10.4.231.1/26
					2001:678:3fc:1af::1
			435	eth0.435	10.4.235.1/26
					2001:678:3fc:1b3::1
			-	dummy0	10.4.230.1/29
					2001:678:3fc:1ae::1
taris	R2	OSPFv2 : 0.211.4.42 OSPFv3 : 0.211.6.42	431	eth0.431	10.4.231.2/26
					2001:678:3fc:1af::2
			433	eth0.433	10.4.233.2/26
					2001:678:3fc:1b1::2
			-	veth0:veth1	10.4.232.1/30 : 10.4.232.2/30
					2001:678:3fc:1b0::1 -> 2001:678:3fc:1b0::2
teth	R3	OSPFv2 : 0.211.4.43 OSPFv3 : 0.211.6.43	435	eth0.435	10.4.235.3/26
					2001:678:3fc:1b3::3
			433	eth0.433	10.4.233.3/26
					2001:678:3fc:1b1::3
			-	veth0:veth1	10.4.234.1/30 : 10.4.234.2/30
					2001:678:3fc:1b2::1 -> 2001:678:3fc:1b2::2

2.2. Plan d'adressage salle 213

Tableau 7. Affectation d'un commutateur par groupe de 3 routeurs - salle 213

Groupe	Commutateur
1	asw04-213.infra.stri
2	asw05-213.infra.stri
3	asw06-213.infra.stri
4	asw07-213.infra.stri

Voici, pour chaque groupe de trois routeurs, le tableau d'affectation des rôles, des numéros de VLANs et des adresses IP.

Tableau 8. Salle 213 - Groupe 1

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
alderaan	R1	OSPFv2 : 0.213.4.11 OSPFv3 : 0.213.6.11	4	eth0.4	172.16.17.1/20
					2001:678:3fc:4::6a5
			221	eth0.221	10.1.21.1/26
					2001:678:3fc:dd::1
			231	eth0.231	10.1.31.1/26
					2001:678:3fc:e7::1
			-	dummy0	10.1.10.1/29
					2001:678:3fc:da::1
bespin	R2	OSPFv2 : 0.213.4.12 OSPFv3 : 0.213.6.12	221	eth0.221	10.1.21.2/26
					2001:678:3fc:dd::2
			232	eth0.232	10.1.32.2/26
					2001:678:3fc:e8::2
			-	veth0:veth1	10.1.20.1/30 10.1.20.2/30
					2001:678:3fc:db::1 -> 2001:678:3fc:db::2
centares	R3	OSPFv2 : 0.213.4.13 OSPFv3 : 0.213.6.13	231	eth0.231	10.1.31.3/26
					2001:678:3fc:e7::3
			232	eth0.232	10.1.32.3/26
					2001:678:3fc:e8::3
			-	veth0:veth1	10.1.30.1/30 10.1.30.2/30
					2001:678:3fc:dc::1 -> 2001:678:3fc:dc::2

Tableau 9. Salle 213 - Groupe 2

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
coruscant	R1	OSPFv2 : 0.213.4.21 OSPFv3 : 0.213.6.21	4	eth0.4	172.16.17.2/20
					2001:678:3fc:4::6a6
			241	eth0.241	10.2.41.1/26
					2001:678:3fc:f1::1
			251	eth0.251	10.2.51.1/26
					2001:678:3fc:fb::1
			-	dummy0	10.2.40.1/29
					2001:678:3fc:ee::1
dagobah	R2	OSPFv2 : 0.0.4.22 OSPFv3 : 0.0.6.22	241	eth0.241	10.2.41.2/26
					2001:678:3fc:f1::2
			254	eth0.254	10.2.54.2/26
					2001:678:3fc:fe::2
			-	veth0:veth1	10.2.50.1/30 10.2.50.2/30
					2001:678:3fc:ef::1 -> 2001:678:3fc:ef::2
endor	R3	OSPFv2 : 0.213.4.23 OSPFv3 : 0.213.6.23	251	eth0.251	10.2.51.3/26
					2001:678:3fc:fb::3
			254	eth0.254	10.2.54.3/26
					2001:678:3fc:fe::3
			-	veth0:veth1	10.2.60.1/30 10.2.60.2/30
					2001:678:3fc:f0::1 -> 2001:678:3fc:f0::2

Tableau 10. Salle 213 - Groupe 3

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
felucia	R1	OSPFv2 : 0.213.4.31 OSPFv3 : 0.213.6.31	4	eth0.4	172.16.17.3/20
					2001:678:3fc:4::6a7
			261	eth0.261	10.3.61.1/26
					2001:678:3fc:105::1
			271	eth0.271	10.3.71.1/26
					2001:678:3fc:10f::1
			-	dummy0	10.3.70.1/29
					2001:678:3fc:102::1
geonosis	R2	OSPFv2 : 0.213.4.32 OSPFv3 : 0.213.6.32	261	eth0.261	10.3.61.2/26
					2001:678:3fc:105::2
			276	eth0.276	10.3.76.2/26
					2001:678:3fc:114::2
			-	veth0:veth1	10.3.80.1/30 10.3.80.2/30
					2001:678:3fc:103::1 -> 2001:678:3fc:103::2
hoth	R3	OSPFv2 : 0.213.4.33 OSPFv3 : 0.213.6.33	271	eth0.271	10.3.71.3/26
					2001:678:3fc:10f::3
			276	eth0.276	10.3.76.3/26
					2001:678:3fc:114::3
			-	veth0:veth1	10.3.90.1/30 10.3.90.2/30
					2001:678:3fc:104::1 -> 2001:678:3fc:104::2

Tableau 11. Salle 213 - Groupe 4

Poste	Rôle	OSPF router-id	VLAN	Interface	Réseau
mustafar	R1	OSPFv2 : 0.213.4.41 OSPFv3 : 0.213.6.41	4	eth0.4	172.16.17.4/20
					2001:678:3fc:4::6a8
			281	eth0.281	10.4.81.1/26
					2001:678:3fc:119::1
			291	eth0.291	10.4.91.1/26
					2001:678:3fc:123::1
			-	dummy0	10.4.100.1/29
					2001:678:3fc:116::1
naboo	R2	OSPFv2 : 0.213.4.42 OSPFv3 : 0.213.6.42	281	eth0.281	10.4.81.2/26
					2001:678:3fc:119::2
			298	eth0.298	10.4.98.2/26
					2001:678:3fc:12a::2
			-	veth0:veth1	10.4.110.1/30 10.4.110.2/30
					2001:678:3fc:117::1 -> 2001:678:3fc:117::2
tatooine	R3	OSPFv2 : 0.213.4.43 OSPFv3 : 0.213.6.43	291	eth0.291	10.4.91.3/26
					2001:678:3fc:123::3
			298	eth0.298	10.4.98.3/26
					2001:678:3fc:12a::3
			-	veth0:veth1	10.4.120.1/30 10.4.120.2/30
					2001:678:3fc:118::1 -> 2001:678:3fc:118::2

3. Préparer les systèmes pour le routage IPv4 et IPv6

La première étape consiste à installer les outils sur les trois routeurs, à appliquer une configuration commune et à mettre en place la topologie physique.

1. Installer les paquets quagga-core, quagga-ospfd et quagga-ospf6d avant de brasser les postes sur les commutateurs désignés à la Section 2, « Topologie réseau étudiée ».

```
$ aptitude search ~iquagga
i   quagga-core          - network routing daemons (core abstraction layer)
i   quagga-ospf6d        - OSPF6 routing daemon
i   quagga-ospfd         - OSPF routing daemon
```

   Sans configuration particulière, les trois services correspondant aux paquets installés ne sont pas lancés.

```
$ systemctl status zebra
systemctl status zebra
● zebra.service - GNU Zebra routing manager
   Loaded: loaded (/lib/systemd/system/zebra.service; enabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:zebra

$ systemctl status ospfd
● ospfd.service - OSPF routing daemon
   Loaded: loaded (/lib/systemd/system/ospfd.service; enabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:ospfd

$ systemctl status ospf6d
● ospf6d.service - OSPF routing daemon for IPv6
   Loaded: loaded (/lib/systemd/system/ospf6d.service; enabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:ospf6d
```

2. Activer le routage IPv4 et IPv6 au niveau noyau.

   Il faut éditer le fichier `/etc/sysctl.conf` pour fixer les valeurs des paramètres de configuration du routage. Voir la section Fonctions réseau d'une interface du support Configuration d'une interface de réseau local.

```
# sysctl -p
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
net.ipv4.conf.all.secure_redirects = 1
net.ipv4.conf.all.log_martians = 1
```

3. Créer les sous-interfaces associées aux VLANs sur chacun des routeurs `R1`, `R2` et `R3` à l'aide du script suivant :

```
#!/bin/bash

for vlan in $*
do
 ip link add link eth0 name eth0.$vlan type vlan id $vlan
 ip link set dev eth0.$vlan up
done
```

   Sur le routeur `R1`, on utilise le script avec les numéros de VLANs 12 et 13 par exemple.

```
R1:~# sh ./subinterfaces.sh 12 13
```

   On adapte l'utilisation du même script aux routeurs `R2` et `R3` avec les numéros de VLANs concernés.

4. Créer les fichiers de configuration de base pour les différents démons et outils sur chaque routeur en utilisant les patrons livrés avec le paquet quagga-core.

   Créer aussi le dossier dédié aux journaux des trois démons de routage : `/var/log/quagga`.

   La présence d'un fichier de configuration par démon est une condition indispensable pour le lancement des services via systemd.

```
# cp /usr/share/doc/quagga-core/examples/zebra.conf.sample /etc/quagga/zebra.conf
# cp /usr/share/doc/quagga-core/examples/ospfd.conf.sample /etc/quagga/ospfd.conf
# cp /usr/share/doc/quagga-core/examples/ospf6d.conf.sample /etc/quagga/ospf6d.conf
# cp /usr/share/doc/quagga-core/examples/vtysh.conf.sample /etc/quagga/vtysh.conf
# chown quagga.quagga /etc/quagga/*.conf
# mkdir /var/log/quagga
# chown quagga.adm /var/log/quagga
# chmod 2750 /var/log/quagga
```

5. Éditer le patron du fichier de configuration de l'outil vtysh.

```
# cat /etc/quagga/vtysh.conf
!
! Configuration file for vtysh.
!
service integrated-vtysh-config
hostname R1
username root nopassword
```

Cet outil offre une console unifiée pour les trois démons : zebra, ospfd et ospf6d. Dès que l'instruction `service integrated-vtysh-config` est active, le fichier `/etc/quagga/Quagga.conf` devient de fichier de configuration commun à l'ensemble des démons actifs. Pour autant, les fichiers de configuration individuels proposés ci-dessous sont toujours nécessaires ; notamment pour distinguer les fichiers de journalisations spécifiques à chaque protocole.

6. Éditer le patron du fichier de configuration du démon `zebra` en fixant les paramètres de connexion à utiliser pour y accéder séparément de la console unifiée.

```
# cat zebra.conf
! -*- zebra -*-
!
hostname R1-zebra
password zebra
enable password zebra
!
log file /var/log/quagga/zebra.log
```

7. Éditer le patron du fichier de configuration du démon `ospfd` en fixant les paramètres de connexion à utiliser pour y accéder séparément de la console unifiée.

```
# cat ospfd.conf
! -*- ospfv2 -*-
!
hostname R1-ospfd
password zebra
enable password zebra
!
log file /var/log/quagga/ospfd.log
```

8. Éditer le patron du fichier de configuration du démon `ospf6d` en fixant les paramètres de connexion à utiliser pour y accéder séparément de la console unifiée.

```
# cat ospf6d.conf
! -*- ospfv3 -*-
!
hostname R1-ospf6d
password zebra
enable password zebra
!
log file /var/log/quagga/ospf6d.log
```

9. Activer et lancer les services correspondant aux trois démons : zebra, ospfd et ospf6d.

```
# systemctl enable zebra

# systemctl enable ospfd

# systemctl enable ospf6d

# systemctl start zebra

# systemctl start ospfd

# systemctl start ospf6d
```

10. Vérifier que les services sont correctement lancés.

```
# systemctl status zebra
● zebra.service - GNU Zebra routing manager
   Loaded: loaded (/lib/systemd/system/zebra.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2018-10-12 08:55:58 UTC; 3min 12s ago
     Docs: man:zebra
  Process: 2521 ExecStart=/usr/sbin/zebra -d -A 127.0.0.1 -f /etc/quagga/zebra.conf (code=exited, status=0/SUCCES
  Process: 2520 ExecStartPre=/bin/chown -f quagga:quaggavty /etc/quagga/vtysh.conf (code=exited, status=0/SUCCESS
  Process: 2519 ExecStartPre=/bin/chown -f quagga:quagga /etc/quagga/zebra.conf (code=exited, status=0/SUCCESS)
  Process: 2518 ExecStartPre=/bin/chmod -f 640 /etc/quagga/vtysh.conf /etc/quagga/zebra.conf (code=exited, status
  Process: 2517 ExecStartPre=/sbin/ip route flush proto zebra (code=exited, status=0/SUCCESS)
 Main PID: 2522 (zebra)
   Memory: 1.3M
   CGroup: /system.slice/zebra.service
           └─2522 /usr/sbin/zebra -d -A 127.0.0.1 -f /etc/quagga/zebra.conf

oct. 12 08:55:58 R3 systemd[1]: Starting GNU Zebra routing manager...
oct. 12 08:55:58 R3 systemd[1]: Started GNU Zebra routing manager.

# systemctl status ospfd
● ospfd.service - OSPF routing daemon
   Loaded: loaded (/lib/systemd/system/ospfd.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2018-10-12 08:56:01 UTC; 4min 20s ago
     Docs: man:ospfd
  Process: 2527 ExecStart=/usr/sbin/ospfd -d -A 127.0.0.1 -f /etc/quagga/ospfd.conf (code=exited, status=0/SUCCES
  Process: 2526 ExecStartPre=/bin/chown -f quagga:quagga /etc/quagga/ospfd.conf (code=exited, status=0/SUCCESS)
  Process: 2525 ExecStartPre=/bin/chmod -f 640 /etc/quagga/ospfd.conf (code=exited, status=0/SUCCESS)
 Main PID: 2528 (ospfd)
   Memory: 1.1M
   CGroup: /system.slice/ospfd.service
           └─2528 /usr/sbin/ospfd -d -A 127.0.0.1 -f /etc/quagga/ospfd.conf

oct. 12 08:56:01 R3 systemd[1]: Starting OSPF routing daemon...
oct. 12 08:56:01 R3 systemd[1]: Started OSPF routing daemon.

# systemctl status ospf6d
● ospf6d.service - OSPF routing daemon for IPv6
   Loaded: loaded (/lib/systemd/system/ospf6d.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2018-10-12 08:56:05 UTC; 5min ago
     Docs: man:ospf6d
  Process: 2533 ExecStart=/usr/sbin/ospf6d -d -A ::1 -f /etc/quagga/ospf6d.conf (code=exited, status=0/SUCCESS)
  Process: 2532 ExecStartPre=/bin/chown -f quagga:quagga /etc/quagga/ospf6d.conf (code=exited, status=0/SUCCESS)
  Process: 2531 ExecStartPre=/bin/chmod -f 640 /etc/quagga/ospf6d.conf (code=exited, status=0/SUCCESS)
 Main PID: 2534 (ospf6d)
   Memory: 1.0M
   CGroup: /system.slice/ospf6d.service
           └─2534 /usr/sbin/ospf6d -d -A ::1 -f /etc/quagga/ospf6d.conf

oct. 12 08:56:05 R3 systemd[1]: Starting OSPF routing daemon for IPv6...
oct. 12 08:56:05 R3 systemd[1]: Started OSPF routing daemon for IPv6.
```

11. Compléter la configuration des interfaces dans la console unifiée vtysh de façon à fixer la bande passante de chaque interface active à 1Gbps.

    Voici un exemple de séquence d'instructions pour configurer une interface.

```
# vtysh

Hello, this is Quagga (version 1.2.4).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

R1# conf t
R1(config)# int eth0
R1(config-if)# bandwidth 1000000
R1(config-if)# ^Z
R1# copy run start
Building Configuration...
Integrated configuration saved to /etc/quagga/Quagga.conf
[OK]
```

    Une fois revenu au système d'exploitation, on doit obtenir un résultat équivalent à celui présenté ci-dessous. Les noms d'interfaces dépendent du routeur concerné.

```
# grep -1 bandwidth /etc/quagga/Quagga.conf
interface eth0
 bandwidth 1000000
!
interface eth0.12
 bandwidth 1000000
!
interface eth0.13
 bandwidth 1000000
!
```

Note

Contrairement à un routeur «intégré» avec un système d'exploitation dédié, le démon de routage statique n'a pas directement accès aux interfaces matérielles. Or, sur un système GNU/Linux, le débit d'une interface Ethernet filaire peut varier en fonction du débit proposé par le port du commutateur. Sans information spécifique du noyau, l'application «service de routage» n'a aucun moyen de connaître le débit exact de l'interface. C'est la raison pour laquelle il est nécessaire de paramétrer manuellement les débits de chaque interface dans la configuration du démon `zebra`.

Avertissement

Sans configuration manuelle du coût de lien d'une interface, ce paramétrage de la bande passante est essentiel dans le calcul des métriques et le fonctionnement du protocole de routage OSPF. Si les calculs de métriques pour les liens actifs sont erronés, le choix des routes à emprunter pour faire transiter le trafic utilisateur entre deux routeurs peut lui aussi être erroné.

Une fois l'ensemble des opérations de cette section réalisées, chaque routeur dispose des outils pour mettre en œuvre la topologie physique et ensuite les protocoles de routage dynamique OSPF.

4. Valider les communications entre routeurs

Avant d'aborder le déploiement du protocole de routage dynamique, il est nécessaire de valider le raccordement des routeurs aux commutateurs désignés, les communications entre chaque routeur et la visualisation des tables de routage pour les interfaces réseau configurées.

Q1. Quelles sont les opérations à effectuer pour implanter les adresses IPv4 et IPv6 des interfaces correspondant à chacun des VLANs routés ?

Au niveau liaison, les sous-interfaces ont déjà été configurées avec le script subinterfaces.sh. Il reste à paramétrer les adresses de ces sous-interfaces.

Routeur R1

```
R1:~# ip addr add 10.1.12.1/26 brd + dev eth0.12
R1:~# ip -6 addr add 2001:678:3fc:c::1/64 dev eth0.12
R1:~# ip addr add 10.1.13.1/26 brd + dev eth0.13
R1:~# ip -6 addr add 2001:678:3fc:d::1/64 dev eth0.13
```

Routeur R2

```
R2:~# ip addr add 10.1.12.2/26 brd + dev eth0.12
R2:~# ip -6 addr add 2001:678:3fc:c::2/64 dev eth0.12
R2:~# ip addr add 10.1.23.2/26 brd + dev eth0.23
R2:~# ip -6 addr add 2001:678:3fc:17::2/64 dev eth0.23
```

Routeur R3

```
R3:~# ip addr add 10.1.13.3/26 brd + dev eth0.13
R3:~# ip -6 addr add 2001:678:3fc:d::3/64 dev eth0.13
R3:~# ip addr add 10.1.23.3/26 brd + dev eth0.23
R3:~# ip -6 addr add 2001:678:3fc:17::3/64 dev eth0.23
```

Q2. Quelles sont les opérations à effectuer pour valider les communications IP entre routeurs ?

Lancer les tests ICMP usuels entre chaque routeur sur chaque lien actif.

Exemple entre R1 et R2

```
R1:~# ping -qc2 10.1.12.2
PING 10.1.12.2 (10.1.12.2) 56(84) bytes of data.

--- 10.1.12.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 6ms
rtt min/avg/max/mdev = 0.068/0.309/0.551/0.242 ms
```

L'opération est à répéter sur chaque lien entre deux routeurs reliés sur le même VLAN.

Q3. Quelles sont les opérations à effectuer pour visualiser la table de routage existante d'un routeur au niveau système et au niveau de la console unifiée vtysh ?

Utiliser la commande ip pour visualiser les tables de routage puis afficher les mêmes tables de routage à partir de la console vtysh avec les commandes du système Cisco™ IOS show ip route et show ipv6 route.

Routeur R1 - niveau système

```
R1:~# ip route ls
default via 192.0.2.1 dev eth0 onlink
10.1.12.0/26 dev eth0.12 proto kernel scope link src 10.1.12.1
10.1.13.0/26 dev eth0.13 proto kernel scope link src 10.1.13.1
192.0.2.0/27 dev eth0 proto kernel scope link src 192.0.2.9
```

```
R1:~# ip -6 route ls
2001:678:3fc:a::/64 dev eth0 proto kernel metric 256 expires 86123sec pref medium
2001:678:3fc:c::/64 dev eth0.12 proto kernel metric 256 pref medium
2001:678:3fc:d::/64 dev eth0.13 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0.12 proto kernel metric 256 pref medium
fe80::/64 dev eth0.13 proto kernel metric 256 pref medium
default via fe80::dc02:44ff:fe64:4834 dev eth0 metric 1024 onlink pref medium
```

Toutes les routes affichées correspondent à des réseaux IPv4 et IPv6 sur lesquels le routeur est directement connecté via une interface active correctement configurée.

Routeur R1 - console vtysh

```
R1# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* 0.0.0.0/0 via 192.0.2.1, eth0
C>* 10.1.12.0/26 is directly connected, eth0.12
C>* 10.1.13.0/26 is directly connected, eth0.13
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.0.2.0/27 is directly connected, eth0
```

```
R1# sh ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv6, I - IS-IS, B - BGP, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* ::/0 via fe80::dc02:44ff:fe64:4834, eth0
C>* ::1/128 is directly connected, lo
C>* 2001:678:3fc:a::/64 is directly connected, eth0
C>* 2001:678:3fc:c::/64 is directly connected, eth0.12
C>* 2001:678:3fc:d::/64 is directly connected, eth0.13
C * fe80::/64 is directly connected, eth0
C * fe80::/64 is directly connected, eth0.13
C>* fe80::/64 is directly connected, eth0.12
```

On retrouve les mêmes informations qu'au niveau système. Une distinction apparaît entre les routes par défaut héritées du niveau système qui sont repérées avec l'indicateur `K` et les autres routes qui correspondent aux réseaux IPv4 et IPv6 sur lesquels le routeur est directement connecté.

Q4. Comment activer la fonction routage du noyau Linux ?

Reprendre l'instruction présentée dans le document Configuration d'une interface de réseau local : activation du routage.

L'opération doit être répétée sur chacun des trois routeurs pour que les paquets soient acheminés normalement.

Si cette fonction n'est pas active dans le noyau Linux, aucune décision de routage d'un paquet d'une interface vers l'autre ne sera prise. Les paquets à router sont simplement jetés.

Les instructions d'activation de la fonction de routage sont données dans la section Préparation des routeurs.

5. Configurer les démons OSPF Quagga

Dans cette section, on introduit les premières commandes de configuration du protocole de routage dynamique OSPF qui permettent d'activer le protocole puis d'ajouter des entrées de réseau dans la base de données de ce protocole.

Q5. Comment peut-on contrôler que le protocole OSPF est actif ou non sur un routeur ?

Une fois la console vtysh ouverte, lancer les commandes de visualisation de l'état du protocole listées ci-dessous. Ces commandes peuvent être lancées sur chacun des trois routeurs.

show ip ospf
show ipv6 ospf6

Dans les informations données dans la copie d'écran ci-dessous, il apparaît que l'algorithme de calcul de topologie n'a pas encore été activé.

```
# vtysh

Hello, this is Quagga (version 1.2.4).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

R1# show ip ospf
 OSPF Routing Process not enabled
R1# show ipv6 ospf6
OSPFv3 is not running
```

Q6. Quelles sont les opérations à effectuer pour activer les protocoles de routage OSPFv2 et OSPFv3 ? Comment affecter manuellement l'identifiant du routeur ?

Avertissement

Les identifiants à utiliser lors de la séance de travaux pratiques sont donnés dans les tableaux des plans d'adressage. Voir Section 2, « Topologie réseau étudiée ».

La liste des commandes utiles en mode configuration dans la console vtysh est la suivante.

router ospf
router ospf6
router-id X.X.X.X
log-adjacency-changes

Toujours à partir de la console vtysh, on accède au mode configuration à l'aide de la commande conf t. Voici un exemple de séquence sur le premier routeur.

```
R1# conf t
R1(config)# router ospf
R1(config-router)#  ospf router-id 0.0.1.4
R1(config-router)#  log-adjacency-changes
R1(config-router)# exit
R1(config)# router ospf6
R1(config-ospf6)#  router-id 0.0.1.6
R1(config-ospf6)#  log-adjacency-changes
R1(config-ospf6)# ^Z
```

Le choix de codage des identifiants OSPF a pour but d'éviter une confusion avec les adresses des réseaux actifs sur chaque routeur.

Si on reprend l'instruction de la question précédente, on obtient l'état de chacun des démons de protocole de routage dynamique.

```
R1# show ip ospf
 OSPF Routing Process, Router ID: 0.0.1.4
 Supports only single TOS (TOS0) routes
 This implementation conforms to RFC2328
 RFC1583Compatibility flag is disabled
 OpaqueCapability flag is disabled
 Initial SPF scheduling delay 0 millisec(s)
 Minimum hold time between consecutive SPFs 50 millisec(s)
 Maximum hold time between consecutive SPFs 5000 millisec(s)
 Hold time multiplier is currently 1
 SPF algorithm has not been run
 SPF timer is inactive
 Refresh timer 10 secs
 Number of external LSA 0. Checksum Sum 0x00000000
 Number of opaque AS LSA 0. Checksum Sum 0x00000000
 Number of areas attached to this router: 0
 Adjacency changes are logged
```

```
R1# show ipv6 ospf6
 OSPFv3 Routing Process (0) with Router-ID 0.0.1.6
 Running 00:12:11
 Initial SPF scheduling delay 0 millisec(s)
 Minimum hold time between consecutive SPFs 50 millsecond(s)
 Maximum hold time between consecutive SPFs 5000 millsecond(s)
 Hold time multiplier is currently 1
 SPF algorithm has not been run$
 SPF timer is inactive
 Number of AS scoped LSAs is 0
 Number of areas in this router is 0
 Adjacency changes are logged
```

Q7. Quelles sont les opérations à effectuer pour activer le protocole de routage OSPFv2 pour les réseaux IPv4 connus de chaque routeur ?

Il faut ajouter une entrée pour chaque préfixe réseau IPv4 de la topologie logique connu du routeur. On peut lister les entrées marquées `c` de la table de routage.

La liste des commandes utiles en mode console et en mode configuration dans vtysh est la suivante.

show ip route connected
router ospf
network A.B.C.D/MM area 0

Voici une exemple de séquence d'instructions pour le routeur `R1`.

```
R1# show ip route connected
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel, N - NHRP,
       > - selected route, * - FIB route

C>* 10.1.12.0/26 is directly connected, eth0.12
C>* 10.1.13.0/26 is directly connected, eth0.13
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.0.2.0/27 is directly connected, eth0
```

```
R1# conf t
R1(config)# router ospf
R1(config-router)# network 10.1.12.0/26 area 0
R1(config-router)# network 10.1.13.0/26 area 0
R1(config-router)# ^Z
```

Q8. Quelles sont les opérations à effectuer pour activer le protocole de routage OSPFv3 pour les réseaux IPv6 connus de chaque routeur ?

Il faut ajouter une entrée pour chaque interface active de la topologie logique connue du routeur.

La liste des commandes utiles en mode console et en mode configuration dans vtysh est la suivante.

show ipv6 route connected
router ospf6
interface eth0.XX area 0.0.0.0

Voici une exemple de séquence d'instructions pour le routeur `R1`.

```
R1# sh ipv6 route connected
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv6, I - IS-IS, B - BGP, A - Babel, N - NHRP,
       > - selected route, * - FIB route

C>* ::1/128 is directly connected, lo
C>* 2001:678:3fc:a::/64 is directly connected, eth0
C>* 2001:678:3fc:c::/64 is directly connected, eth0.12
C>* 2001:678:3fc:d::/64 is directly connected, eth0.13
C * fe80::/64 is directly connected, eth0
C * fe80::/64 is directly connected, eth0.13
C>* fe80::/64 is directly connected, eth0.12
```

```
R1# conf t
R1(config)# router ospf6
R1(config-ospf6)# interface eth0.12 area 0.0.0.0
R1(config-ospf6)# interface eth0.13 area 0.0.0.0
R1(config-ospf6)# ^Z
```

Q9. Comment visualiser l'état des interfaces actives pour chaque processus de protocole de routage dynamique OSPFv2 ou OSPFv3 ?

Les interfaces sont dites actives pour OSPFv2 dès qu'une entrée de réseau correspondant à l'adresse IPv4 d'interface est présente dans la configuration du protocole de routage.

Pour OSPFv3, c'est plus simple. Les interfaces sont explicitement activées dans la configuration du protocole de routage.

La liste des commandes utiles dans la console vtysh est la suivante.

show ip ospf interface
show ipv6 ospf6 interface

En reprenant l'exemple du routeur R1, on obtient les résultats suivants.

```
R1# show ip ospf interface
eth0 is up
  ifindex 19, MTU 1500 bytes, BW 1000000 Kbit <UP,BROADCAST,RUNNING,MULTICAST>
  OSPF not enabled on this interface
eth0.12 is up
  ifindex 2, MTU 1500 bytes, BW 1000000 Kbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.1.12.1/26, Broadcast 10.1.12.63, Area 0.0.0.0
  MTU mismatch detection:enabled
  Router ID 0.0.1.4, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State Backup, Priority 1
  Designated Router (ID) 0.0.2.4, Interface Address 10.1.12.2
  Backup Designated Router (ID) 0.0.1.4, Interface Address 10.1.12.1
  Saved Network-LSA sequence number 0x80000039
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5
    Hello due in 8.858s
  Neighbor Count is 1, Adjacent neighbor count is 1
eth0.13 is up
  ifindex 3, MTU 1500 bytes, BW 1000000 Kbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.1.13.1/26, Broadcast 10.1.13.63, Area 0.0.0.0
  MTU mismatch detection:enabled
  Router ID 0.0.1.4, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State Backup, Priority 1
  Designated Router (ID) 0.0.3.4, Interface Address 10.1.13.3
  Backup Designated Router (ID) 0.0.1.4, Interface Address 10.1.13.1
  Saved Network-LSA sequence number 0x80000029
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5
    Hello due in 1.767s
  Neighbor Count is 1, Adjacent neighbor count is 1
lo is up
  ifindex 1, MTU 65536 bytes, BW 0 Kbit <UP,LOOPBACK,RUNNING>
  OSPF not enabled on this interface
```

```
R1# show ipv6 ospf6 interface
eth0 is up, type BROADCAST
  Interface ID: 19
   OSPF not enabled on this interface
eth0.12 is up, type BROADCAST
  Interface ID: 2
  Internet Address:
    inet : 10.1.12.1/26
    inet6: fe80::c828:72ff:fedf:3608/64
    inet6: 2001:678:3fc:c::1/64
  Instance ID 0, Interface MTU 1500 (autodetect: 1500)
  MTU mismatch detection: enabled
  Area ID 0.0.0.0, Cost 1
  State BDR, Transmit Delay 1 sec, Priority 1
  Timer intervals configured:
   Hello 10, Dead 40, Retransmit 5
  DR: 0.0.2.6 BDR: 0.0.1.6
  Number of I/F scoped LSAs is 2
    0 Pending LSAs for LSUpdate in Time 00:00:00 [thread off]
    0 Pending LSAs for LSAck in Time 00:00:00 [thread off]
eth0.13 is up, type BROADCAST
  Interface ID: 3
  Internet Address:
    inet : 10.1.13.1/26
    inet6: fe80::c828:72ff:fedf:3608/64
    inet6: 2001:678:3fc:d::1/64
  Instance ID 0, Interface MTU 1500 (autodetect: 1500)
  MTU mismatch detection: enabled
  Area ID 0.0.0.0, Cost 1
  State BDR, Transmit Delay 1 sec, Priority 1
  Timer intervals configured:
   Hello 10, Dead 40, Retransmit 5
  DR: 0.0.3.6 BDR: 0.0.1.6
  Number of I/F scoped LSAs is 2
    0 Pending LSAs for LSUpdate in Time 00:00:00 [thread off]
    0 Pending LSAs for LSAck in Time 00:00:00 [thread off]
lo is up, type LOOPBACK
  Interface ID: 1
   OSPF not enabled on this interface
```

Q10. Comment retrouver l'identifiant de routeur ?

À partir des résultats des questions précédentes, rechercher l'information demandée.

Quelque soit la version du protocole OSPF, l'identifiant de routeur est toujours codé sous la forme d'une adresse IPv4.

```
R1# sh ip ospf
 OSPF Routing Process, Router ID: 0.0.1.4
```

```
R1# sh ipv6 ospf6
 OSPFv3 Routing Process (0) with Router-ID 0.0.1.6
```

Q11. Comment identifier le type de réseau d'une interface ?

À partir des résultats des questions précédentes, rechercher l'information demandée.

Comme on utilise uniquement des liens Ethernet dans ce contexte de travaux pratiques, le type de réseau est nécessairement diffusion.

```
R1# show ip ospf interface eth0.12
eth0.12 is up
  ifindex 2, MTU 1500 bytes, BW 1000000 Kbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.1.12.1/26, Broadcast 10.1.12.63, Area 0.0.0.0
  MTU mismatch detection:enabled
  Router ID 0.0.1.4, Network Type BROADCAST, Cost: 1
```

```
R1# show ipv6 ospf6 interface eth0.12
eth0.12 is up, type BROADCAST
```

Q12. Comment obtenir la liste du ou des routeurs voisins pour chaque processus de protocole de routage dynamique OSPFv2 ou OSPFv3 ?

Dès qu'une interface est active, il y a émission de paquets HELLO et si un routeur avec un démon OSPF envoie aussi des paquets HELLO dans le même VLAN, les deux routeurs cherchent à former une adjacence.

La liste des commandes utiles dans la console vtysh est la suivante.

show ip ospf neighbor

show ipv6 ospf6 neighbor

À nouveau sur le routeur `R1`, voici un exemple de liste de routeurs OSPF voisins dans laquelle on reconnaît les identifiants des routeurs `R2` et `R3`.

```
R1# show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface            RXmtL RqstL DBsmL
0.0.2.4           1 Full/DR           32.395s 10.1.12.2       eth0.12:10.1.12.1        0     0     0
0.0.3.4           1 Full/DR           38.317s 10.1.13.3       eth0.13:10.1.13.1        0     0     0
```

```
R1# show ipv6 ospf6 neighbor
Neighbor ID     Pri    DeadTime  State/IfState         Duration I/F[State]
0.0.2.6           1    00:00:31   Full/DR              02:04:11 eth0.12[BDR]
0.0.3.6           1    00:00:31   Full/DR              02:04:01 eth0.13[BDR]
```

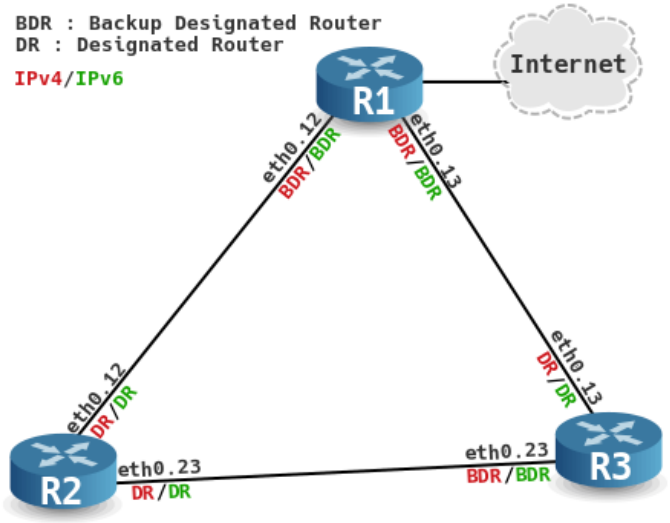
Q13. Comment identifier le rôle des différentes interfaces des routeurs pour chacun des liens du triangle de la topologie logique ?

Avertissement

La réponse à cette question suppose que les démons OSPF des trois routeurs de la topologie logique en triangle aient convergé. On doit repérer l'état `Full` pour les listes de routeurs voisins.

De plus, la réponse varie en fonction de l'ordre d'activation des démons OSPF des différents routeurs. En effet, un routeur peut être élu routeur désigné (DR) en l'absence de routeurs voisins. Cette élection n'est pas remise en cause tant qu'il n'y pas de changement d'état de lien.

À partir des résultats des questions précédentes sur les interfaces actives, il est possible de compléter le schéma de la topologie étudiée avec l'état des interfaces pour chacun des trois liens.



Sur un même réseau de diffusion, il est possible de trouver plusieurs routeurs OSPF. Établir une relation de voisinage et procéder aux échanges de bases de données topologiques entre chaque routeur revient à constituer un réseau de relations complètement maillé. À chaque recalcul de topologie, ce réseau complètement maillé est inefficace. C'est la raison pour laquelle la notion de routeur référent ou Designated Router a été introduite. Lors d'un recalcul de topologie, tous les routeurs s'adressent au référent qui correspond au cœur d'un réseau en topologie étoile.

Dans le contexte de la topologie triangle étudiée, il y a bien élection d'un routeur référent et d'un routeur référent de secours. Cependant, comme il n'y a que deux routeurs par domaine de diffusion ou VLAN, on ne peut pas caractériser l'utilité de cette élection.

Q14. Quelles sont les réseaux IPv4 et IPv6 présents dans la base calcul du protocole OSPF ?

On cherche a visualiser la liste des préfixes des réseaux connus des deux démons OSPF.

La liste des commandes utiles dans la console vtysh est la suivante.

show ip ospf route

show ipv6 ospf6 route

Une fois que les trois routeurs de la topologie ont convergé, chaque démon connaît les trois préfixes qui correspondent aux trois côtés du triangle. Un routeur correspond à un sommet du triangle et il doit apprendre le préfixe réseau du côté opposé via ses deux routeurs voisins.

Voici la vue depuis le routeur `R1`.

```
R1# sh ip ospf route
============ OSPF network routing table ============
N    10.1.12.0/26          [1] area: 0.0.0.0
                           directly attached to eth0.12
N    10.1.13.0/26          [1] area: 0.0.0.0
                           directly attached to eth0.13
N    10.1.23.0/26          [2] area: 0.0.0.0
                           via 10.1.12.2, eth0.12
                           via 10.1.13.3, eth0.13

============ OSPF router routing table =============

============ OSPF external routing table ===========
```

Les valeurs notées entre crochets correspondent à la métrique du lien pour joindre le réseau noté à gauche. Pour le protocole OSPF, le calcul de métrique se fait à partir de l'expression : 10^8 / Bande_Passante_du_lien.

La valeur du numérateur (10^8) correspond à un débit de 100Mbps. À l'époque de la rédaction du standard OSPFv2, ce débit a servi de référence. Aujourd'hui, cette valeur est complètement dépassée. C'est la raison pour laquelle on adapte le calcul de métrique en changeant le coefficient du numérateur. Voir la Section 10, « Adapter de la métrique de lien au débit ».

Les deux premiers réseaux de la table sont joignable via un lien Ethernet à 1000Mbps ; soit une métrique de `1`. Le troisième réseau est joignable via deux liens Ethernet à 1000Mbps ; d'où la métrique de `2`.

Pour les préfixes IPv6, les métriques n'apparaissent pas.

```
R1# sh ipv6 ospf6 route
*N IA 2001:678:3fc:c::/64           ::                        eth0.12 18:41:10
*N IA 2001:678:3fc:d::/64           ::                        eth0.13 18:41:00
*N IA 2001:678:3fc:17::/64          fe80::4c8c:2aff:fe42:b075 eth0.12 18:41:00
                                    fe80::440:39ff:fe30:710c  eth0.13
```

Avec OSPFv3, les relations de voisinage entre routeurs utilisent nécessairement les adresses de lien local appartenant au préfixe `fe80::/10`.

Q15. Comment visualiser les tables de routage depuis la console vtysh ?

L'affichage demandé illustre les mécanismes de choix entre différentes solutions pour une même destination. Cet affichage est à comparer avec celui demandé à la question suivante.

La liste des commandes utiles dans la console vtysh est la suivante.

show ip route
show ipv6 route

```
R1# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* 0.0.0.0/0 via 192.0.2.1, eth0
O   10.1.12.0/26 [110/1] is directly connected, eth0.12, 18:57:52
C>* 10.1.12.0/26 is directly connected, eth0.12
O   10.1.13.0/26 [110/1] is directly connected, eth0.13, 18:57:36
C>* 10.1.13.0/26 is directly connected, eth0.13
O>* 10.1.23.0/26 [110/2] via 10.1.12.2, eth0.12, 18:57:36
  *                      via 10.1.13.3, eth0.13, 18:57:36
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.0.2.0/27 is directly connected, eth0
```

```
R1# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv6, I - IS-IS, B - BGP, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* ::/0 via fe80::dc02:44ff:fe64:4834, eth0
C>* ::1/128 is directly connected, lo
C>* 2001:678:3fc:a::/64 is directly connected, eth0
O   2001:678:3fc:c::/64 [110/1] is directly connected, eth0.12, 18:50:54
C>* 2001:678:3fc:c::/64 is directly connected, eth0.12
O   2001:678:3fc:d::/64 [110/1] is directly connected, eth0.13, 18:50:44
C>* 2001:678:3fc:d::/64 is directly connected, eth0.13
O>* 2001:678:3fc:17::/64 [110/2] via fe80::4c8c:2aff:fe42:b075, eth0.12, 18:50:44
  *                              via fe80::440:39ff:fe30:710c, eth0.13, 18:50:44
C * fe80::/64 is directly connected, eth0
C * fe80::/64 is directly connected, eth0.13
C>* fe80::/64 is directly connected, eth0.12
```

- Les entrées avec le caractère `*` correspondent aux routes proposées par la suite Quagga au sous-système réseau du noyau. Les autres entrées sont placées en réserve au cas où la solution initialement proposée viendrait à faire défaut.

- L'entrée notée `K` correspond à une route apprise via la configuration système.

- Les entrées notées `C` correspondent à des routes pour lesquelles il existe une interface sur le routeur. Les métriques de ses routes ont la valeur `0`. Ce sont les routes les plus prioritaires.

- Les entrées notées `O` correspondent aux routes apprises via le protocole OSPF. La métrique de ces routes se décompose en deux parties. La valeur figée à `110` définit le niveau de priorité du protocole OSPF (Administrative Distance) relativement aux autres protocoles de routage. Les valeurs notées après le `/` sont les métriques de liens calculées comme indiqué ci-dessus.

Q16. Comment visualiser les tables de routage au niveau système ?

Utiliser une commande usuelle de visualisation de la table de routage.

ip route ls
ip -6 route ls

Avec la commande ip, on voit apparaître les «sources» d'alimentation de la table de routage finale du système.

- `kernel` pour les entrées connues du sous-système réseau du noyau. Ce sont les entrées avec le caractère `C` dans la console vtysh.

- `zebra` pour les entrées apprises via la suite Quagga. Le réseau correspondant au côté opposé au sommet du triangle est appris via OSPF puisque le sous-système réseau du noyau ne le connaît pas.

- `ra` pour la route par défaut IPv6. Le routeur `R1` se distingue des deux autres routeurs par le fait qu'il dispose d'un accès vers l'Internet. Ici, la route par défaut a été apprise par auto configuration SLAAC.

```
R1:~# ip route ls
default via 192.0.2.1 dev eth0 onlink
10.1.12.0/26 dev eth0.12 proto kernel scope link src 10.1.12.1
10.1.13.0/26 dev eth0.13 proto kernel scope link src 10.1.13.1
10.1.23.0/26 proto zebra metric 20
        nexthop via 10.1.12.2 dev eth0.12 weight 1
        nexthop via 10.1.13.3 dev eth0.13 weight 1
192.0.2.0/27 dev eth0 proto kernel scope link src 192.0.2.9
```

```
R1:~# ip -6 route ls
2001:678:3fc:a::/64 dev eth0 proto kernel metric 256 expires 86285sec pref medium
2001:678:3fc:c::/64 dev eth0.12 proto kernel metric 256 pref medium
2001:678:3fc:d::/64 dev eth0.13 proto kernel metric 256 pref medium
2001:678:3fc:17::/64 proto zebra metric 20
        nexthop via fe80::4c8c:2aff:fe42:b075 dev eth0.12 weight 1
        nexthop via fe80::440:39ff:fe30:710c dev eth0.13 weight 1 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0.12 proto kernel metric 256 pref medium
fe80::/64 dev eth0.13 proto kernel metric 256 pref medium
default via fe80::dc02:44ff:fe64:4834 dev eth0 proto ra metric 1024 expires 1685sec hoplimit 64 pref medium
```

6. Publier les routes par défaut via OSPF

Dans la topologie logique étudiée, le routeur `R1` dispose d'un lien montant vers l'Internet. On peut donc considérer que ce lien est la route par défaut vers tous les réseaux non connus de l'aire OSPF contenant les trois routeurs.

Il est possible de publier une route par défaut via le protocole OSPF depuis le routeur `R1` vers les routeurs `R2` et `R3`.

Voici, pour mémoire, une copie des bases de données OSPFv2 et OSPFv3 avant la mise en place de la publication de route par défaut. On reconnaît les LSAs (Link State Advertisement)) de type 1 et 2 qui correspondent respectivement aux annonces de routeurs et de réseaux.

```
R1# sh ip ospf database

       OSPF Router with ID (0.0.1.4)

                Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
0.0.1.4         0.0.1.4          546 0x80000035 0xff9f 2
0.0.2.4         0.0.2.4          745 0x8000008d 0x1d13 2
0.0.3.4         0.0.3.4         1157 0x80000061 0xaba9 2

                Net Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum
10.1.12.2       0.0.2.4          565 0x80000032 0xd964
10.1.13.3       0.0.3.4          867 0x80000031 0xcc6e
10.1.23.2       0.0.2.4          505 0x8000005a 0x26e2
```

```
R1# show ipv6 ospf6 database

        Area Scoped Link State Database (Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Rtr  0.0.0.0         0.0.1.6         540 80000032                0.0.2.6/0.0.0.2
Rtr  0.0.0.0         0.0.1.6         540 80000032                0.0.3.6/0.0.0.2
Rtr  0.0.0.0         0.0.2.6         551 80000086                0.0.2.6/0.0.0.2
Rtr  0.0.0.0         0.0.2.6         551 80000086                0.0.2.6/0.0.0.3
Rtr  0.0.0.0         0.0.3.6         541 8000005d                0.0.3.6/0.0.0.2
Rtr  0.0.0.0         0.0.3.6         541 8000005d                0.0.2.6/0.0.0.3
Net  0.0.0.2         0.0.2.6         551 80000030                       0.0.2.6
Net  0.0.0.2         0.0.2.6         551 80000030                       0.0.1.6
Net  0.0.0.3         0.0.2.6        1485 8000005a                       0.0.2.6
Net  0.0.0.3         0.0.2.6        1485 8000005a                       0.0.3.6
Net  0.0.0.2         0.0.3.6         541 80000030                       0.0.3.6
Net  0.0.0.2         0.0.3.6         541 80000030                       0.0.1.6
INP  0.0.0.2         0.0.2.6         551 80000030            2001:678:3fc:c::/64
INP  0.0.0.3         0.0.2.6        1485 8000005a           2001:678:3fc:17::/64
INP  0.0.0.2         0.0.3.6         541 80000030            2001:678:3fc:d::/64

        I/F Scoped Link State Database (I/F eth0.12 in Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Lnk  0.0.0.2         0.0.1.6         553 80000030      fe80::c828:72ff:fedf:3608
Lnk  0.0.0.2         0.0.2.6        1191 80000084      fe80::4c8c:2aff:fe42:b075

        I/F Scoped Link State Database (I/F eth0.13 in Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Lnk  0.0.0.3         0.0.1.6         550 80000030      fe80::c828:72ff:fedf:3608
Lnk  0.0.0.2         0.0.3.6        1492 8000005a       fe80::440:39ff:fe30:710c

        AS Scoped Link State Database

Type LSId            AdvRouter       Age   SeqNum                       Payload
```

Q17. Quelle est la condition préalable à respecter pour que le routeur `R1` soit en mesure de publier une route par défaut via son démon OSPF ?

À partir des tables de routage relevées dans la Section 5, « Configurer les démons OSPF Quagga », repérer le routeur qui dispose d'un accès vers un réseau qui n'appartient pas à la topologie triangle.

ip route ls default
ip -6 route ls default

Une route par défaut doit exister avant d'être injectée dans une aire OSPF. Dans notre cas, une route statique par défaut suffit à respecter la condition préalable.

Sur la maquette, on valide la présence des routes par défaut à l'aide la commande ip au niveau système.

```
R1:~# ip route ls default
default via 192.0.2.1 dev eth0 onlink
```

```
R1:~# ip -6 route ls default
default via fe80::dc02:44ff:fe64:4834 dev eth0 proto ra metric 1024 \
    expires 1463sec hoplimit 64 pref medium
```

Au niveau de la console vtysh, ces même routes correspondent aux entrées marquées `K` pour kernel.

```
R1# sh ip route kernel
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* 0.0.0.0/0 via 192.0.2.1, eth0
```

```
R1# sh ipv6 route kernel
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv6, I - IS-IS, B - BGP, A - Babel, N - NHRP,
       > - selected route, * - FIB route

K>* ::/0 via fe80::dc02:44ff:fe64:4834, eth0
```

Q18. Quelle est l'instruction à utiliser pour publier une route par défaut via le protocole de routage OSPFv2 pour IPv4 ?

Parmi toutes les méthodes de redistribution de routes disponibles avec protocole OSPFv2, il en existe une dédiée à l'injection de route par défaut dans une aire normale : Voir Index des commandes Quagga.

Rechercher le mot clé `default` dans l'index.

L'instruction qui correspond le plus simplement à la redistribution de route par défaut est la suivante.

default-information originate

On doit l'appliquer au démon de routage OSPFv2 uniquement sur le routeur `R1`.

```
R1# conf t
R1(config)# router ospf
R1(config-router)# default-information originate
R1(config-router)# ^Z
```

Une fois cette instruction exécutée, la base de données OSPFv2 est complétée avec une nouvelle entrée.

```
R1# sh ip ospf database

       OSPF Router with ID (0.0.1.4)

                Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
0.0.1.4         0.0.1.4         1203 0x80000039 0xfd9b 2
0.0.2.4         0.0.2.4          791 0x80000091 0x1517 2
0.0.3.4         0.0.3.4         1324 0x80000065 0xa3ad 2

                Net Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum
10.1.12.2       0.0.2.4          511 0x80000036 0xd168
10.1.13.3       0.0.3.4          884 0x80000035 0xc472
10.1.23.2       0.0.2.4          611 0x8000005e 0x1ee6

                AS External Link States

Link ID         ADV Router      Age  Seq#       CkSum  Route
0.0.0.0         0.0.1.4          133 0x80000004 0x229e E2 0.0.0.0/0 [0x0]
```

On voit apparaître une nouvelle rubrique baptisée AS External Link States. Ce nouveau rôle pour le routeur `R1` apparaît aussi lorsque l'on affiche l'état de l'instance de routage OSPFv2.

```
R1# show ip ospf
 OSPF Routing Process, Router ID: 0.0.1.4
 Supports only single TOS (TOS0) routes
 This implementation conforms to RFC2328
 RFC1583Compatibility flag is disabled
 OpaqueCapability flag is disabled
 Initial SPF scheduling delay 0 millisec(s)
 Minimum hold time between consecutive SPFs 50 millisec(s)
 Maximum hold time between consecutive SPFs 5000 millisec(s)
 Hold time multiplier is currently 1
 SPF algorithm last executed 1h21m25s ago
 Last SPF duration 90 usecs
 SPF timer is inactive
 Refresh timer 10 secs
 This router is an ASBR (injecting external routing information)
 Number of external LSA 1. Checksum Sum 0x0000229e
 Number of opaque AS LSA 0. Checksum Sum 0x00000000
 Number of areas attached to this router: 1
 Adjacency changes are logged

 Area ID: 0.0.0.0 (Backbone)
   Number of interfaces in this area: Total: 2, Active: 2
   Number of fully adjacent neighbors in this area: 2
   Area has no authentication
   SPF algorithm executed 8 times
   Number of LSA 6
   Number of router LSA 3. Checksum Sum 0x0001b65f
   Number of network LSA 3. Checksum Sum 0x0001b4c0
   Number of summary LSA 0. Checksum Sum 0x00000000
   Number of ASBR summary LSA 0. Checksum Sum 0x00000000
   Number of NSSA LSA 0. Checksum Sum 0x00000000
   Number of opaque link LSA 0. Checksum Sum 0x00000000
   Number of opaque area LSA 0. Checksum Sum 0x00000000
```

Le routeur R1, est maintenant à la frontière entre deux systèmes autonomes. Il a le rôle ASBR (Autonmous System Border Router) et il est responsable de l'émission des LSAs de type 5 à destination des autres routeurs de l'aire.

Ici, R1 utilise OSPF et possède une route statique définie au niveau système vers l'Internet. L'instruction donnée ci-dessus assure la redistribution de la route statique vers R2 et R3. Cette route apparaît comme une entrée de type E2 dans les tables de routage de ces routeurs.

L'indicateur E2 correspond au type par défaut des routes apprises par le biais de la redistribution. La métrique est un point important à considérer avec les routes de type E2. Ces routes ne présentent que le coût du chemin allant du routeur ASBR vers le réseau de destination ; ce qui ne correspond pas au coût réel du chemin à l'intérieur de l'aire OSPF.

Q19. Quelles sont les instructions à utiliser pour publier une route par défaut via le protocole de routage OSPFv3 pour IPv6 ?

Parmi toutes les méthodes de redistribution de routes disponibles avec le protocole OSPFv3, celle qui nous intéresse est relative aux entrées apprises via le noyau. En effet, dans la première question de cette section sur la présence d'une route par défaut, nous avons identifié les entrées marquées K dans les tables de routage.

Pour traiter cette question, il faut isoler le préfixe correspondant à la route par défaut, créer une carte de routage qui établit la correspondance avec ce préfixe et enfin redistribuer les routes apprises du noyau dans OSPFv3 à partir de la carte.

Les instrcutions à utiliser pour traiter cette question entrent dans la liste suivante.

ipv6 prefix-list
route-map
match ipv6 address prefix-list
redistribute kernel route-map

Voici la séquence des instructions qui permet de redistribuer uniquement la route par défaut apprise via le noyau dans OSPFv3.

```
R1# conf t
R1(config)# ipv6 prefix-list IPv6-DEFAULT-PLIST seq 5 permit ::/0
R1(config)# route-map IPv6-DEFAULT-RMAP permit 1
R1(config-route-map)# match ipv6 address prefix-list IPv6-DEFAULT-PLIST
R1(config-route-map)# exit
R1(config)# router ospf6
R1(config-ospf6)# redistribute kernel route-map IPv6-DEFAULT-RMAP
R1(config-ospf6)# ^Z
```

Une fois cette instruction exécutée, la base de données OSPFv3 est complétée avec une nouvelle entrée.

```
R1# show ipv6 ospf6 database

        Area Scoped Link State Database (Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Rtr  0.0.0.0         0.0.1.6         208 80000039               0.0.2.6/0.0.0.2
Rtr  0.0.0.0         0.0.1.6         208 80000039               0.0.3.6/0.0.0.2
Rtr  0.0.0.0         0.0.2.6         216 8000008b               0.0.2.6/0.0.0.2
Rtr  0.0.0.0         0.0.2.6         216 8000008b               0.0.2.6/0.0.0.3
Rtr  0.0.0.0         0.0.3.6         206 80000062               0.0.3.6/0.0.0.2
Rtr  0.0.0.0         0.0.3.6         206 80000062               0.0.2.6/0.0.0.3
Net  0.0.0.2         0.0.2.6         216 80000035                       0.0.2.6
Net  0.0.0.2         0.0.2.6         216 80000035                       0.0.1.6
Net  0.0.0.3         0.0.2.6        1150 8000005f                       0.0.2.6
Net  0.0.0.3         0.0.2.6        1150 8000005f                       0.0.3.6
Net  0.0.0.2         0.0.3.6         206 80000035                       0.0.3.6
Net  0.0.0.2         0.0.3.6         206 80000035                       0.0.1.6
INP  0.0.0.2         0.0.2.6         216 80000035            2001:678:3fc:c::/64
INP  0.0.0.3         0.0.2.6        1150 8000005f           2001:678:3fc:17::/64
INP  0.0.0.2         0.0.3.6         206 80000035            2001:678:3fc:d::/64

        I/F Scoped Link State Database (I/F eth0.12 in Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Lnk  0.0.0.2         0.0.1.6         218 80000035      fe80::c828:72ff:fedf:3608
Lnk  0.0.0.2         0.0.2.6         856 80000089      fe80::4c8c:2aff:fe42:b075

        I/F Scoped Link State Database (I/F eth0.13 in Area 0.0.0.0)

Type LSId            AdvRouter       Age   SeqNum                       Payload
Lnk  0.0.0.3         0.0.1.6         215 80000035      fe80::c828:72ff:fedf:3608
Lnk  0.0.0.2         0.0.3.6        1157 8000005f       fe80::440:39ff:fe30:710c

        AS Scoped Link State Database

Type LSId            AdvRouter       Age   SeqNum                       Payload
ASE  0.0.0.1         0.0.1.6         208 80000001                            ::
```

On voit apparaître une nouvelle rubrique baptisée AS Scoped Link State Database. Ce nouveau rôle pour le routeur `R1` apparaît aussi lorsque l'on affiche l'état de l'instance de routage OSPFv3.

```
R1# show ipv6 ospf6
 OSPFv3 Routing Process (0) with Router-ID 0.0.1.6
 Running 1d02:39:46
 Initial SPF scheduling delay 0 millisec(s)
 Minimum hold time between consecutive SPFs 50 millsecond(s)
 Maximum hold time between consecutive SPFs 5000 millsecond(s)
 Hold time multiplier is currently 1
 SPF algorithm has not been run$
 SPF timer is inactive
 Number of AS scoped LSAs is 1
 Number of areas in this router is 1
 Adjacency changes are logged

 Area 0.0.0.0
     Number of Area scoped LSAs is 9
     Interface attached to this area: eth0.12 eth0.13
```

Q20. Comment la publication de route par défaut apparaît-elle sur les autres routeurs de la topologie triangle ?

Relevez, dans la console vtysh, la métrique de la route par défaut sur les routeurs qui n'ont pas une connexion directe vers l'Internet.

Les instrcutions à utiliser pour traiter cette question entrent dans la liste suivante.

show ip route A.B.C.D/MM
show ipv6 route A:B:C::D/MM
show ip ospf route
show ipv6 ospf6 route

En prenant l'exemple du routeur `R2`, on retrouve les informations suivantes.

- Vue de la table de routage :

```
R2# sh ip route 0.0.0.0/0
Routing entry for 0.0.0.0/0
  Known via "ospf", distance 110, metric 10, tag 0, vrf 0, best, fib
  Last update 01:54:46 ago
  >* 10.1.12.1, via eth0.12
```

```
R2# sh ipv6 route ::/0
Routing entry for ::/0
  Known via "ospf6", distance 110, metric 1, tag 0, vrf 0, best, fib
  Last update 00:12:24 ago
  >* fe80::c828:72ff:fedf:3608, via eth0.12
```

- Vue de la base de topologie OSPF :

```
R2# sh ip ospf route
============ OSPF network routing table ============
N    10.1.12.0/26          [1] area: 0.0.0.0
                           directly attached to eth0.12
N    10.1.13.0/26          [2] area: 0.0.0.0
                           via 10.1.12.1, eth0.12
                           via 10.1.23.3, eth0.23
N    10.1.23.0/26          [1] area: 0.0.0.0
                           directly attached to eth0.23

============ OSPF router routing table =============
R    0.0.1.4               [1] area: 0.0.0.0, ASBR
                           via 10.1.12.1, eth0.12

============ OSPF external routing table ===========
N E2 0.0.0.0/0             [1/10] tag: 0
                           via 10.1.12.1, eth0.12
```

```
R2# show ipv6 ospf6 route
*N E1 ::/0                            fe80::c828:72ff:fedf:3608 eth0.12 00:18:25
*N IA 2001:678:3fc:c::/64             ::                        eth0.12 1d02:18:38
*N IA 2001:678:3fc:d::/64             fe80::c828:72ff:fedf:3608 eth0.12 1d02:18:28
                                      fe80::440:39ff:fe30:710c  eth0.23
*N IA 2001:678:3fc:17::/64            ::                        eth0.23 1d23:34:11
```

Avec les copies d'écran ci-dessus, on vérifie bien que les routes par défaut ont été apprises via le protocole OSPF.

7. Ajouter des réseaux fictifs

L'introduction de nouvelles entrées fictives dans les tables de routage est une pratique très répandue. Elle permet de qualifier le bon fonctionnement du filtrage réseau ou d'un service Internet sans ajouter de matériel. Cette section décrit justement la mise en place d'un service Web de test.

Pour réaliser ces manipulations, on dispose de deux techniques distinctes.

- Historiquement, la première solution consiste à utiliser des interfaces de boucles locales comme sur les équipements Cisco™ par exemple. Sur un système GNU/Linux les interfaces de type dummy correspondent à cet usage.

  Cette solution est satisfaisante du point de vue protocole de routage, mais ces interfaces ne participent pas au routage sur le système sur lequel elles sont implantées. Elles servent aux annonces émises par les protocoles de routage dynamique.

  Ce type d'interface est utilisé sur le routeur `R1` dans la Section 8, « Créer des interfaces de type dummy ».

- Avec le recours de plus en plus fréquent à la virtualisation et aux conteneurs, de nouveaux types d'interfaces virtuelles ont été introduits : `tap` et `veth`. Pour simplifier au maximum, les interfaces `tap` sont des cordons de brassage et les interfaces `veth` sont des liens point à point qui peuvent être isolés dans un espace de noms particulier.

  Dans notre contexte, chaque paire d'interfaces `veth` nous sert à introduire nouveau réseau fictif sans ajouter un seul équipement matériel supplémentaire.

  Relativement aux interface de type `dummy`, l'utilisation d'un lien point à point provoque effectivement une décision de routage au sein du noyau sur lequel ce lien est implanté.

  Pour compléter un peu plus le scénario du réseau factice, on peut utiliser les espaces de noms réseau (network namespaces). Un espace de nom réseau désigne un ensemble d'interfaces associées à une table de routage indépendante. Dans notre contexte, on place une extrémité du lien point à point, c'est à dire une interface `veth`, dans un espace de nom différent.

  Cet espace de nom réseau et le type d'interface `veth` sont utilisés sur les routeurs `R2` et `R3` dans la Section 9, « Créer des interfaces de type veth ».

8. Créer des interfaces de type dummy

Dans cette section, on créé une interface de type `dummy` et on installe un service Web uniquement en écoute sur les adresses IPv4 et IPv6 de cette interface. Enfin, on publie ce nouveau réseau factice via le protocole de routage dynamique OSPF.

Q21. Comment créer et configurer une interface réseau virtuelle de type dummy ?

Consulter les pages de manuels de la commande ip au niveau liaison man ip-link et rechercher les information sur le type `dummy`.

Vérifier la présence d'une nouvelle entrée dans la table de routage au niveau système.

Sur le routeur `R1`, on créé une nouvelle interface avec des adresses appartenant à deux nouveaux préfixes réseau définis les tableaux du plan d'adressage.

```
# ip link add dummy0 type dummy
# ip link set dev dummy0 up
# ip addr add 10.1.1.1/29 brd + dev dummy0
# ip -6 addr add 2001:678:3fc:b::1/64 dev dummy0
```

```
R1:~# ip route ls dev dummy0
10.1.1.0/29 proto kernel scope link src 10.1.1.1
```

```
R1:~# ip -6 route ls dev dummy0
2001:678:3fc:b::/64 proto kernel metric 256 pref medium
fe80::/64 proto kernel metric 256 pref medium
```

Cette nouvelle interface apparaît automatiquement dans la console unifiée vtysh des démons de routage de la suite Quagga.

```
R1# sh interface dummy0
Interface dummy0 is up, line protocol is up
  Link ups:       1  last: Mon, 15 Oct 2018 12:01:14 +0000
  Link downs:     0  last: (never)
  vrf: 0
  index 4 metric 0 mtu 1500
  flags: <UP,BROADCAST,RUNNING,NOARP>
  Type: Unknown
  HWaddr: d2:57:b1:cb:be:32
  inet 10.1.1.1/29 broadcast 10.1.1.7
  inet6 fe80::d057:b1ff:fecb:be32/64
  inet6 2001:678:3fc:b::1/64
```

Q22. Comment publier les préfixes réseau correspondant à l'interface `dummy0` via OSPF avec un débit binaire de lien à 100Mbps ?

Une fois la console vtysh ouverte, accéder en mode configuration de l'interface puis des deux démons de routage.

interface dummyX
bandwidth
router ospf
network A.B.C.D/MM area 0
router ospf6
interface dummyX area 0.0.0.0

On commence par la définition du débit binaire au niveau de l'interface.

```
R1# conf t
R1(config)# int dummy0
R1(config-if)# bandwidth 100000
R1(config-if)# ^Z
```

On peut ensuite publier les préfixes réseau IPv4 et IPv6 au niveau de chaque instance de protocole de routage OSPF.

```
R1# conf t
R1(config)# router ospf
R1(config-router)# network 10.1.1.0/29 area 0
R1(config-router)# ^Z
```

```
R1# conf t
R1(config)# router ospf6
R1(config-ospf6)# interface dummy0 area 0.0.0.0
R1(config-ospf6)# ^Z
```

Enfin, on vérifie la présence des deux préfixes réseau dans les bases OSPF.

```
R1# sh ip route 10.1.1.0/29
Routing entry for 10.1.1.0/29
  Known via "ospf", distance 110, metric 10, tag 0, vrf 0
  Last update 00:03:45 ago
  >  directly connected, dummy0

Routing entry for 10.1.1.0/29
  Known via "connected", distance 0, metric 0, tag 0, vrf 0, best, fib
  >* directly connected, dummy0
```

```
R1# sh ipv6 route 2001:678:3fc:b::/64
Routing entry for 2001:678:3fc:b::/64
  Known via "ospf6", distance 110, metric 10, tag 0, vrf 0
  Last update 00:04:20 ago
  >  directly connected, dummy0

Routing entry for 2001:678:3fc:b::/64
  Known via "connected", distance 0, metric 0, tag 0, vrf 0, best, fib
  >* directly connected, dummy0
```

Q23. Comment installer un service Web en écoute exclusivement sur les adresses IPv4 et IPv6 de l'interface `dummy0` ?

Installer le paquet `lighttpd` et modifier sa configuration pour que le service ne soit accessible que sur les adresses de l'interface `dummy0`.

```
# aptitude install lighttpd
<snipped/>
```

On modifie ensuite le fichier de configuration `/etc/lighttpd/lighttpd.conf` de façon à limiter l'accès aux adresses voulues.

```
--- lighttpd.conf.orig  2018-10-15 15:42:12.345480008 +0000
+++ lighttpd.conf       2018-10-15 15:43:50.091036311 +0000
@@ -22,6 +22,9 @@
 compress.filetype           = ( "application/javascript", "text/css", "text/html", "text/plain" )

 # default listening port for IPv6 falls back to the IPv4 port
-include_shell "/usr/share/lighttpd/use-ipv6.pl " + server.port
+#include_shell "/usr/share/lighttpd/use-ipv6.pl " + server.port
 include_shell "/usr/share/lighttpd/create-mime.assign.pl"
 include_shell "/usr/share/lighttpd/include-conf-enabled.pl"
+
+server.bind = "10.1.1.1"
+$SERVER["socket"] == "[2001:678:3fc:b::1]:80" { server.use-ipv6 = "enable" }
```

Après avoir redémarré l'instance de serveur Web, on vérifie que la nouvelle configuration est bien en place.

```
# systemctl restart lighttpd
<snipped>
# ss -nlt '( src :80 )'
State       Recv-Q Send-Q       Local Address:Port      Peer Address:Port
LISTEN      0      128              10.1.1.1:80             0.0.0.0:*
LISTEN      0      128    [2001:678:3fc:b::1]:80               [::]:*
```

Q24. Comment valider l'accès à ce service Web depuis les autres réseaux ?

En respectant l'ordre des protocoles de la pile TCP/IP, on commence par valider la connectivité au niveau réseau avant de passer à la couche application.

À partir du routeur R2, on utilise la séquence suivante :

- Test ICMP au niveau réseau :

```
R2:~# ping -c2 10.1.1.1
PING 10.1.1.1 (10.1.1.1) 56(84) bytes of data.
64 bytes from 10.1.1.1: icmp_seq=1 ttl=64 time=0.648 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=64 time=0.090 ms

--- 10.1.1.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 26ms
rtt min/avg/max/mdev = 0.090/0.369/0.648/0.279 ms
```

```
R2:~# ping -c2 2001:678:3fc:b::1
PING 2001:678:3fc:b::1(2001:678:3fc:b::1) 56 data bytes
64 bytes from 2001:678:3fc:b::1: icmp_seq=1 ttl=64 time=0.567 ms
64 bytes from 2001:678:3fc:b::1: icmp_seq=2 ttl=64 time=0.086 ms

--- 2001:678:3fc:b::1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 24ms
rtt min/avg/max/mdev = 0.086/0.326/0.567/0.241 ms
```

- Test HTTP au niveau application :

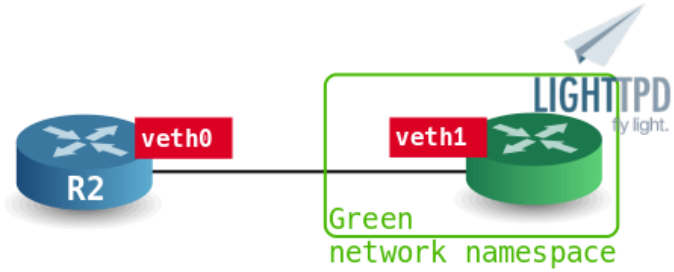
```
R2:~# wget -O /dev/null http://10.1.1.1
--2018-10-15 17:00:47--  http://10.1.1.1/
Connexion à 10.1.1.1:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 3378 (3,3K) [text/html]
Sauvegarde en : « /dev/null »

/dev/null    100%[======================>]   3,30K  --.-KB/s    ds 0s

2018-10-15 17:00:47 (298 MB/s) – « /dev/null » sauvegardé [3378/3378]
```

```
R2:~# wget -O /dev/null http://[2001:678:3fc:b::1]
--2018-10-15 17:02:17--  http://[2001:678:3fc:b::1]/
Connexion à [2001:678:3fc:b::1]:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 3378 (3,3K) [text/html]
Sauvegarde en : « /dev/null »

/dev/null    100%[======================>]   3,30K  --.-KB/s    ds 0s

2018-10-15 17:02:17 (154 MB/s) – « /dev/null » sauvegardé [3378/3378]
```

9. Créer des interfaces de type veth

L'introduction d'un nouvel espace de nom réseau revient à introduire une nouvelle table de routage dans un même système. Dans la vue ci-dessous, le cadre vert illustre les éléments ajoutés au routeur `R2` : un lien vers un espace de noms baptisé green avec un serveur web hébergé à l'intérieur.

Les manipulations présentées dans cette section doivent aussi être réalisées sur le routeur `R3` en suivant le plan d'adressage présenté à la Section 2, « Topologie réseau étudiée ».



Q25. Comment créer une paire d'interfaces réseau virtuelles de type veth sur un système GNU/Linux ?

Rechercher dans les pages de manuels la commande ip link la syntaxe qui permet de créer la paire d'interfaces `veth0` et `veth1`.

```
# ip link add veth1 type veth peer name veth0

# ip link ls dev veth0
5: veth0@veth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN \
    mode DEFAULT group default qlen 1000
    link/ether 86:22:d2:14:05:b4 brd ff:ff:ff:ff:ff:ff

# ip link ls dev veth1
6: veth1@veth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN \
    mode DEFAULT group default qlen 1000
    link/ether 9e:c8:44:e6:fa:70 brd ff:ff:ff:ff:ff:ff
```

Q26. Comment créer un nouvel espace de noms réseau appelé green et affecter l'interface `veth1` à cet espace de noms ?

Rechercher dans les pages de manuels `ip-netns` la syntaxe qui permet de créer un nouvel espace de noms réseau.

```
# ip netns add green
# ip netns ls
green
```

L'affectation d'une interface à l'espace de noms se fait avec la commande suivante.

```
# ip link set veth1 netns green
```

Q27. Comment visualiser et modifier l'état des interfaces de type `veth` dans les deux espaces de noms réseau ?

Rechercher dans les pages de manuels `ip-netns` la syntaxe qui permet d'exécuter une commande dans un nouvel espace de noms réseau.

On active les deux interfaces et on affiche les informations d'état.

```
# ip link set dev veth0 up

# ip link ls dev veth0
5: veth0@if6: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue \
   state LOWERLAYERDOWN mode DEFAULT group default qlen 1000
    link/ether 86:22:d2:14:05:b4 brd ff:ff:ff:ff:ff:ff link-netns green
```

```
# ip netns exec green ip link set dev veth1 up

# ip netns exec green ip link ls dev veth1
6: veth1@if5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue \
   state UP mode DEFAULT group default qlen 1000
    link/ether 9e:c8:44:e6:fa:70 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

Q28. Comment valider les communications entre les deux espaces de noms réseau ?

Utiliser les adresses de lien local IPv6 pour faire un test ICMP avec l'adresse multicast de sollicitation d'hôte.

Le test avec la commande ping doit désigner l'interface du lien à utiliser.

```
# ping -c2 ff02::1%veth0
PING ff02::1%veth0(ff02::1%veth0) 56 data bytes
64 bytes from fe80::8422:d2ff:fe14:5b4%veth0: icmp_seq=1 ttl=64 time=0.080 ms
64 bytes from fe80::9cc8:44ff:fee6:fa70%veth0: icmp_seq=1 ttl=64 time=0.168 ms (DUP!)
64 bytes from fe80::8422:d2ff:fe14:5b4%veth0: icmp_seq=2 ttl=64 time=0.046 ms

--- ff02::1%veth0 ping statistics ---
2 packets transmitted, 2 received, +1 duplicates, 0% packet loss, time 20ms
rtt min/avg/max/mdev = 0.046/0.098/0.168/0.051 ms
```

Le voisinage réseau de l'interface `veth0` désigne l'adresse MAC de l'interface `veth1` dans l'espace de noms réseau green.

```
# ip nei ls dev veth0
fe80::9cc8:44ff:fee6:fa70 lladdr 9e:c8:44:e6:fa:70 STALE
```

Réciproquement, le voisinage réseau de l'interface `veth1` désigne l'adresse MAC de l'interface `veth0`.

```
# ip netns exec green ip nei ls dev veth1
fe80::8422:d2ff:fe14:5b4 lladdr 86:22:d2:14:05:b4 router STALE
```

On peut donc conclure que le lien point à point entre les deux espaces de noms réseau est fonctionnel.

Q29. Comment ajouter deux préfixes réseau IPv4 et IPv6 sur la liaison point à point entre les deux interfaces de type `veth` ?

Comme lors des étapes précédentes, on reprend les commandes ip avec la désignation de l'espace de noms réseau pour l'interface à l'extrémité du lien point à point.

Dans notre exemple sur le routeur `R2`, on utilise les préfixes `10.1.2.0/30` et `2001:678:3fc:d::/64`.

- Pour l'interface `veth0` :

```
# ip addr add 10.1.2.1/30 brd + dev veth0
# ip -6 addr add 2001:678:3fc:d::1/64 dev veth0
```

- Pour l'interface `veth1` :

```
# ip netns exec green ip addr add 10.1.2.2/30 brd + dev veth1
# ip netns exec green ip -6 addr add 2001:678:3fc:d::2/64 dev veth1
```

Comme dans la question précédente, on peut qualifier la communication entre les deux interfaces au niveau réseau à l'aide de la commande ping.

Voici une copie d'écran des tests effectués depuis l'espace de noms `green`.

```
# ip netns exec green ping -qc2 10.1.2.1
PING 10.1.2.1 (10.1.2.1) 56(84) bytes of data.

--- 10.1.2.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 33ms
rtt min/avg/max/mdev = 0.043/0.055/0.067/0.012 ms

# ip netns exec green ping -qc2 2001:678:3fc:d::1
PING 2001:678:3fc:d::1(2001:678:3fc:d::1) 56 data bytes

--- 2001:678:3fc:d::1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 18ms
rtt min/avg/max/mdev = 0.050/0.081/0.113/0.032 ms
```

Q30. Comment publier les préfixes réseau correspondant à la nouvelle liaison point à point via OSPF avec un débit binaire de lien à 1Gbps ?

Une fois la console vtysh ouverte, accéder en mode configuration de l'interface puis des deux démons de routage.

interface vethX
bandwidth
router ospf
network A.B.C.D/MM area 0

router ospf6
interface vethX area 0.0.0.0

On commence par la définition du débit binaire au niveau de l'interface.

```
R1# conf t
R1(config)# int veth0
R1(config-if)# bandwidth 1000000
R1(config-if)# ^Z
```

On peut ensuite publier les préfixes réseau IPv4 et IPv6 au niveau de chaque instance de protocole de routage OSPF.

```
R1# conf t
R1(config)# router ospf
R1(config-router)# network 10.1.2.0/30 area 0
R1(config-router)# ^Z
```

```
R1# conf t
R1(config)# router ospf6
R1(config-ospf6)# interface veth0 area 0.0.0.0
R1(config-ospf6)# ^Z
```

Enfin, on vérifie la présence des deux préfixes réseau dans les bases OSPF.

```
R2# sh ip route 10.1.2.0/30
Routing entry for 10.1.2.0/30
  Known via "ospf", distance 110, metric 1, tag 0, vrf 0
  Last update 00:02:44 ago
  >  directly connected, veth0

Routing entry for 10.1.2.0/30
  Known via "connected", distance 0, metric 0, tag 0, vrf 0, best, fib
  >* directly connected, veth0
```

```
R2# sh ipv6 route 2001:678:3fc:d::/64
Routing entry for 2001:678:3fc:d::/64
  Known via "ospf6", distance 110, metric 1, tag 0, vrf 0
  Last update 00:04:44 ago
  >  directly connected, veth0

Routing entry for 2001:678:3fc:d::/64
  Known via "connected", distance 0, metric 0, tag 0, vrf 0, best, fib
  >* directly connected, veth0
```

Q31. Est-ce que les adresses IPv4 et IPv6 situées dans l'espace de noms green joiganbles depuis les réseaux distants ?

Le fait d'avoir placé l'interface `veth1` dans un espace de noms réseau distinct, impose l'utilisation d'une table de routage propre à cet espace. Sur la base des opérations déjà effectuées, on s'interroge sur le fait que ces opérations suffisent pour que les communications soient fonctionnelles.

On fait des tests ICMP et ICMP6 depuis un autre routeur pour constater que si les flux parviennent bien à destination, il n'existe pas de solution retour.

Dans l'exemple ci-dessous, on teste les communications entre le routeur `R1` et l'interface `veth1` située au delà du routeur `R3`

On lance une requête ICMP depuis `R1`.

```
R1:~# ping -c3 10.1.3.2
PING 10.1.3.2 (10.1.3.2) 56(84) bytes of data.

--- 10.1.3.2 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 40ms
```

On lance l'analyse réseau dans l'espace de noms réseau green.

```
R3:~# ip netns exec green /usr/bin/tshark -i veth1 -f icmp
<snipped/>
Capturing on 'veth1'
 1 0.000000000    10.1.13.1 → 10.1.3.2     ICMP 98 Echo (ping) request  id=0x1309, seq=1/256, ttl=63
 2 1.013794602    10.1.13.1 → 10.1.3.2     ICMP 98 Echo (ping) request  id=0x1309, seq=2/512, ttl=63
 3 2.037727585    10.1.13.1 → 10.1.3.2     ICMP 98 Echo (ping) request  id=0x1309, seq=3/768, ttl=63
^C3 packets captured
```

On relève bien l'arrivée des requêtes à destination. Il n'y a cependant aucune réponse à ces requêtes.

En reprenant le même test avec ICMP6, on relève le même problème.

```
R1:~# ping -c3 2001:678:3fc:f::2
PING 2001:678:3fc:f::2(2001:678:3fc:f::2) 56 data bytes

--- 2001:678:3fc:f::2 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 27ms
```

L'analyse réseau ne montre que l'arrivée des requêtes à destination.

```
R3:~# ip netns exec green /usr/bin/tshark -i veth1 -f icmp6
<snipped/>
Capturing on 'veth1'
 1 0.000000000 2001:678:3fc:d::1 → 2001:678:3fc:f::2 ICMPv6 118 Echo (ping) request id=0x130a, seq=1, hop limit
 2 1.001038537 2001:678:3fc:d::1 → 2001:678:3fc:f::2 ICMPv6 118 Echo (ping) request id=0x130a, seq=2, hop limit
 3 2.025046012 2001:678:3fc:d::1 → 2001:678:3fc:f::2 ICMPv6 118 Echo (ping) request id=0x130a, seq=3, hop limit
```

Q32. Comment compléter les tables de routage IPv4 et IPv6 situées dans l'espace de noms green pour que le trafic sortant par l'interface `veth1` soit émis correctement ?

On complète les tables de routage avec une route par défaut pour joindre les réseaux distants depuis l'espace de noms green.

La lecture des tables de routage dans l'espace de noms réseau green explique tout. Ces tables ne contiennent que l'entrée relative au lien point à point configuré entre les interfaces `veth0` et `veth1`.

```
R3:~# ip netns exec green ip route ls
10.1.3.0/30 dev veth1 proto kernel scope link src 10.1.3.2

R3:~# ip netns exec green ip -6 route ls
2001:678:3fc:f::/64 dev veth1 proto kernel metric 256 pref medium
fe80::/64 dev veth1 proto kernel metric 256 pref medium
```

Il faut donc compléter ces tables de routage avec une route par défaut vers l'espace de noms réseau par défaut dans lequel est située l'interface `veth0`.

```
R3:~# ip netns exec green ip route add default via 10.1.3.1

R3:~# ip netns exec green ping -c1 ff02::2%veth1
PING ff02::2%veth1(ff02::2%veth1) 56 data bytes
64 bytes from fe80::8422:d2ff:fe14:5b4%veth1: icmp_seq=1 ttl=64 time=0.072 ms

--- ff02::2%veth1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.072/0.072/0.072/0.000 ms

R3:~# ip netns exec green ip -6 route add default via fe80::8422:d2ff:fe14:5b4 dev veth1
```

Les nouvelles tables de routage de l'espace de noms green font apparaître les routes par défaut

```
R3:~# ip netns exec green ip route ls
default via 10.1.3.1 dev veth1
10.1.3.0/30 dev veth1 proto kernel scope link src 10.1.3.2
```

```
R3:~# ip netns exec green ip -6 route ls
2001:678:3fc:f::/64 dev veth1 proto kernel metric 256 pref medium
fe80::/64 dev veth1 proto kernel metric 256 pref medium
default via fe80::8422:d2ff:fe14:5b4 dev veth1 metric 1024 pref medium
```

Les tests ICMP et ICMP6 de la question précédente montrent maintenant que les communications sont bien fonctionnelles. Les requêtes sont à nouveau émises depuis le routeur `R1`

```
R1:~# ping -c3 10.1.3.2
PING 10.1.3.2 (10.1.3.2) 56(84) bytes of data.
64 bytes from 10.1.3.2: icmp_seq=1 ttl=63 time=0.484 ms
64 bytes from 10.1.3.2: icmp_seq=2 ttl=63 time=0.068 ms
64 bytes from 10.1.3.2: icmp_seq=3 ttl=63 time=0.066 ms

--- 10.1.3.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 50ms
```

```
R1:~# ping -c3 2001:678:3fc:f::2
PING 2001:678:3fc:f::2(2001:678:3fc:f::2) 56 data bytes
64 bytes from 2001:678:3fc:f::2: icmp_seq=1 ttl=63 time=0.591 ms
64 bytes from 2001:678:3fc:f::2: icmp_seq=2 ttl=63 time=0.083 ms
64 bytes from 2001:678:3fc:f::2: icmp_seq=3 ttl=63 time=0.081 ms

--- 2001:678:3fc:f::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 48ms
```

Q33. Comment installer un service Web en écoute exclusivement sur les adresses IPv4 et IPv6 situées dans l'espace de noms réseau green ?

Pour aller au plus court, on installe le paquet lighttpd et on édite la configuration du service de façon à limiter l'accès aux adresses IP voulues.

```
R3:~# aptitude install lighttpd
```

On modifie ensuite le fichier de configuration `/etc/lighttpd/lighttpd.conf` de façon à limiter l'accès aux adresses IPv4 et IPv6 de l'interface `veth1`.

Voici une copie du correctif correspondant au routeur `R3`.

```
--- lighttpd.conf.orig  2018-10-15 15:42:12.345480008 +0000
+++ lighttpd.conf       2018-10-15 15:43:50.091036311 +0000
@@ -22,6 +22,9 @@
 compress.filetype           = ( "application/javascript", "text/css", "text/html", "text/plain" )

 # default listening port for IPv6 falls back to the IPv4 port
-include_shell "/usr/share/lighttpd/use-ipv6.pl " + server.port
+#include_shell "/usr/share/lighttpd/use-ipv6.pl " + server.port
 include_shell "/usr/share/lighttpd/create-mime.assign.pl"
 include_shell "/usr/share/lighttpd/include-conf-enabled.pl"
+
+server.bind = "10.1.3.2"
+$SERVER["socket"] == "[2001:678:3fc:f::2]:80" { server.use-ipv6 = "enable" }
```

Relativement à une configuration classique, il faut lancer le serveur Web dans le contexte du nouvel espace de noms réseau. On commence donc par arrêter le démon lancé lors de l'installation du paquet.

```
R3:~# systemctl stop lighttpd
```

Là encore, pour aller au plus court on lance directement le démon dans le contexte de l'espace de noms réseau green.

```
R3:~# ip netns exec green lighttpd -f /etc/lighttpd/lighttpd.conf

R3:~# ip netns exec green ss -nlt '( src :80 )'
State      Recv-Q  Send-Q       Local Address:Port  Peer Address:Port
LISTEN     0       128               10.1.3.2:80         0.0.0.0:*
LISTEN     0       128    [2001:678:3fc:f::2]:80            [::]:*
```

Q34. Comment valider l'accès au service Web situé dans l'espace de noms réseau green depuis un réseau distant ?

Comme la partie routage a été validée dans les questions précédentes, on passe directement à la couche application avec le chargement de la page Web par défaut du serveur.

Depuis le routeur `R1`, on obtient les résultats suivants.

```
R1:~# wget -O /dev/null http://10.1.3.2
--2018-10-16 14:32:06--  http://10.1.3.2/
Connexion à 10.1.3.2:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 3378 (3,3K) [text/html]
Sauvegarde en : « /dev/null »

/dev/null       100%[===================>]   3,30K  --.-KB/s    ds 0s

2018-10-16 14:32:06 (209 MB/s) – « /dev/null » sauvegardé [3378/3378]
```

```
R1:~# wget -O /dev/null http://[2001:678:3fc:f::2]
--2018-10-16 14:34:39--  http://[2001:678:3fc:f::2]/
Connexion à [2001:678:3fc:f::2]:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 3378 (3,3K) [text/html]
Sauvegarde en : « /dev/null »

/dev/null       100%[===================>]   3,30K  --.-KB/s    ds 0s

2018-10-16 14:34:39 (268 MB/s) – « /dev/null » sauvegardé [3378/3378]
```

10. Adapter de la métrique de lien au débit

Par défaut, le calcul de métrique du le protocole OSPF se fait à partir de l'expression : 10^8 / Bande_Passante_du_lien.

Cette règle a été établie à une époque où l'utilisation d'un lien à 100Mbps devait être considéré comme une situation d'exploitation futuriste. Aujourd'hui, les liens à 100Mbps sont dépassés.

Cette section traite de la configuration des instances de protocole de routage OSPF utilisant des liens avec un débit binaire supérieur à 100Mbps.

Q35. Quelle est l'instruction à utiliser pour que le calcul de métrique OSPF se fasse sur la base d'un débit de lien à 10Gbps ?

Rechercher le mot clé reference dans l'index des instructions de configuration. Voir Index des commandes Quagga.

C'est l'instruction auto-cost reference-bandwidth qui permet de fixer une nouvelle référence de coût de lien en Mbps.

```
R2-ospfd(config-router)# auto-cost reference-bandwidth
  <1-4294967>  The reference bandwidth in terms of Mbits per second
```

Voici un extrait de la configuration du routeur `R2` après application d'une nouvelle référence à 10000Mbps soit 10Gbps.

```
R3# conf t
R3(config)# router ospf
R3(config-router)# auto-cost reference-bandwidth
  <1-4294967>  The reference bandwidth in terms of Mbits per second
R3(config-router)# auto-cost reference-bandwidth 10000
R3(config-router)# ^Z
```

Il est indispensable que tous les routeurs de l'aire OSPF aient la même référence de calcul de métrique. La même instruction doit donc être implantée dans la configuration des trois routeurs.

Q36. Comment modifier le débit binaire d'un lien à 1Gbps ?

Normalement, le débit d'un lien est directement extrait des paramètres du composant de l'interface connectée au lien. Dans le cas d'interfaces qui n'ont aucune réalité physique, ce débit peut être attribué arbitrairement par configuration. On doit rechercher dans les options de la console vtysh le moyen d'attribuer une indication de débit aux sous-interfaces de VLANs.

Comme indiqué dans la Section 3, « Préparer les systèmes pour le routage IPv4 et IPv6 », c'est dans la configuration des interfaces que l'on attribue le débit binaire d'une interface réseau. Cette opération est nécessaire compte tenu du fait que le débit d'une même interface Ethernet `eth0` peut varier de 10Mbps à 10Gbps.

```
interface eth0
 bandwidth 1000000
!
interface eth0.12
 bandwidth 1000000
!
interface eth0.23
 bandwidth 1000000
```

Q37. Comment peut-on identifier le débit d'un lien dans la configuration OSPF ?

Visualiser les paramètres des interfaces réseau depuis la console vtysh. Les commandes suivantes peuvent être lancées sur chacun des trois routeurs.

show ip ospf interface
show ipv6 ospf6 interface

Voici le résultat obtenu sur le routeur `R2`.

```
R2# sh ip ospf interface eth0.12
eth0.12 is up
  ifindex 2, MTU 1500 bytes, BW 1000000 Kbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.1.12.2/26, Broadcast 10.1.12.63, Area 0.0.0.0
  MTU mismatch detection:enabled
  Router ID 0.0.2.4, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 0.0.2.4, Interface Address 10.1.12.2
  Backup Designated Router (ID) 0.0.1.4, Interface Address 10.1.12.1
  Saved Network-LSA sequence number 0x8000009a
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5
    Hello due in 7.154s
  Neighbor Count is 1, Adjacent neighbor count is 1
```

Q38. Quel est le coût d'accès au service Web installé sur `R3` depuis `R2` ? Justifier la valeur de métrique obtenue.

À partir des informations de routage sur `R2`, faire la somme des métriques de chaque lien entre les deux extrémités en communication.

Il est possible d'obtenir les informations de calcul de métrique à partir de l'affichage d'une route particulière à la console vtysh.

```
R2# sh ip route 10.1.3.0/30
Routing entry for 10.1.3.0/30
  Known via "ospf", distance 110, metric 2, tag 0, vrf 0, best, fib
  Last update 00:10:37 ago
  >* 10.1.23.3, via eth0.23
```

Dans cet exemple, la métrique d'accès au réseau `10.1.3.0/30` correspond à la somme des coûts de deux liens à 1Gbps soit 2

11. Effectuer les manipulations sur machines virtuelles

Il est possible de réaliser l'ensemble des manipulations de ce support à l'aide de trois machines virtuelles et du commutateur Open vSwitch. Dans la figure ci-dessous, le routeur baptisé ISP correspond au système hôte sur lequel les systèmes virtuels sont exécutés.

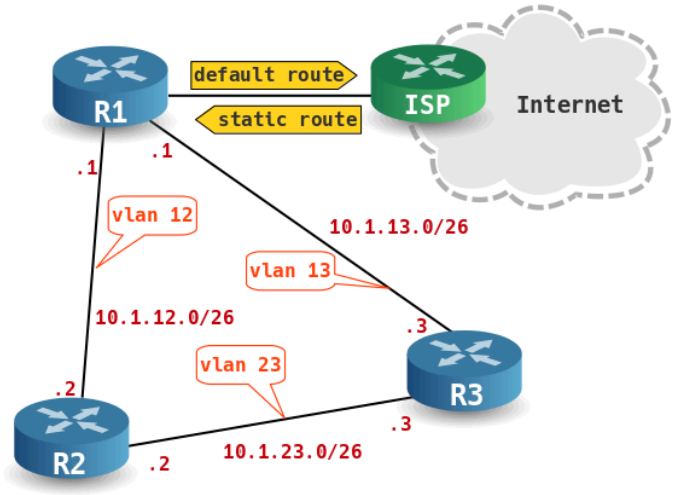
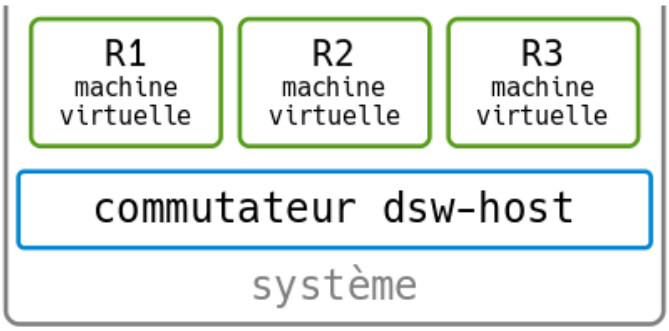


Tableau 12. Affectation des rôles, des numéros de VLANs et des adresses IP

Routeur	OSPF router-id	VLAN	Interface	Réseau
R1	OSPFv2 -> 0.0.1.4 OSPFv3 -> 0.0.1.6	0	eth0	192.0.2.9/27
				2001:678:3fc:a::9/64
		12	eth0.12	10.0.12.1/26
				2001:678:3fc:c::1/64
		13	eth0.13	10.0.13.1/26
				2001:678:3fc:d::1/64
		-	dummy0	10.1.1.1/29
				2001:678:3fc:b::1/64
R2	OSPFv2 -> 0.0.2.4 OSPFv3 -> 0.0.2.6	12	eth0.12	10.1.21.2/26
				2001:678:3fc:c::2/64
		23	eth0.23	10.1.23.2/26
				2001:678:3fc:17::2/64
		-	veth0:veth1	10.1.2.1/30 -> 10.1.2.2/30
				2001:678:3fc:e::1/64 -> 2001:678:3fc:e::2/64
R3	OSPFv2 -> 0.0.3.4 OSPFv3 -> 0.0.3.6	13	eth0.13	10.1.13.3/26
				2001:678:3fc:d::3/64
		23	eth0.23	10.1.23.3/26
				2001:678:3fc:17::3/64
		-	veth0:veth1	10.1.3.1/30 -> 10.1.3.2/30
				2001:678:3fc:f::1/64 -> 2001:678:3fc:f::2/64

11.1. Préparation du commutateur Open vSwitch

Vu du système hôte, on met en place la structure du graphique suivant.



On se réfère au guide Virtualisation système et enseignement dans lequel le lancement du commutateur virtuel Open vSwitch est intégré à la configuration du système hôte. Voici un extrait du fichier `/etc/network/interfaces` relatif à la configuration du commutateur `dsw-host`.

```
# The primary network interface
allow-ovs dsw-host
iface dsw-host inet manual
 ovs_type OVSBridge
 ovs_ports eth0 vlan1

allow-dsw-host vlan1
iface vlan1 inet static
 address 192.0.2.11/26
 gateway 192.0.2.1
 ovs_type OVSIntPort
 ovs_bridge dsw-host
 dns-nameservers 192.0.2.1

allow-dsw-host eth0
iface eth0 inet manual
 up ip link set dev $IFACE up
 down ip link set dev $IFACE down
 ovs_bridge dsw-host
 ovs_type OVSPort
 ovs_options vlan_mode=access
```

Dans cette configuration, `vlan1` est l'interface principale du système hôte. Elle sert à la fois au routage et à l'accès aux instances de machines virtuelles depuis les autres réseaux.

La configuration de départ du commutateur sur le système hôte est la suivante :

```
$ sudo ovs-vsctl show
0fb1e80b-37cf-4dce-9a19-17b9fb989610
    Bridge dsw-host
        Port "eth0"
            Interface "eth0"
        Port "vlan1"
            Interface "vlan1"
                type: internal
        Port dsw-host
            Interface dsw-host
                type: internal
    ovs_version: "2.3.0"
```

Partant de cette configuration de départ, on créé 3 cordons de brassage ; un par routeur. On dispose d'un script pour répéter les opérations de base.

```
#!/bin/bash

for patchtap in $*
do
        if [ -z "$(ip link ls | grep $patchtap)" ]; then
                echo setting $patchtap up
                sudo ip tuntap add mode tap dev $patchtap group kvm multi_queue
                sudo ip link set dev $patchtap up
        else
                echo $patchtap is already there!
        fi
done
```

Avec cet outil, la création des cordons s'effectue comme suit :

```
$ sh ./patchcables.sh tap1 tap2 tap3
```

Maintenant que les cordons sont disponibles, il faut les brasser sur le commutateur `dsw-host`. L'opération ci-dessous créé trois nouveaux ports sur le commutateur.

```
$ for port in tap1 tap2 tap3; do sudo ovs-vsctl add-port dsw-host $port; done
```

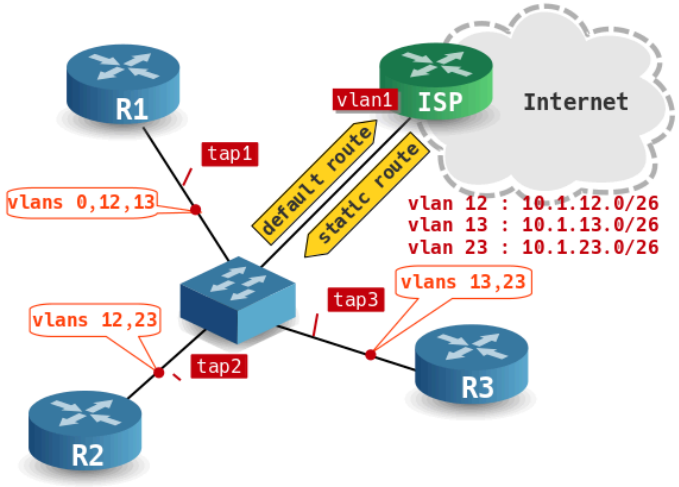
Pour respecter la topologie physique proposée dans l'énoncé, ces trois ports doivent être configurés en mode trunkde façon à ce que chaque routeur puisse router les VLANs qui le concernent. Avec Open vSwitch, les paramètres de configuration sont les suivants :

```
$ sudo ovs-vsctl set port tap1 vlan_mode=trunk trunks=0,12,13
$ sudo ovs-vsctl set port tap2 vlan_mode=trunk trunks=12,23
$ sudo ovs-vsctl set port tap3 vlan_mode=trunk trunks=13,23
```

Note

La particularité du commutateur Open vSwitch est d'utiliser le numéro 0 comme VLAN par défaut. Dans notre cas, ce VLAN 0 devient le VLAN natif dans lequel circulent les trames sans étiquettes IEEE802.1Q.

On aboutit à la représentation graphique suivante :



La table de correspondance entre VLAN et liaison devient :

VLAN numéro 0	Liaison R1 <-> ISP
VLAN numéro 12	Liaison R1 <-> R2
VLAN numéro 13	Liaison R1 <-> R3
VLAN numéro 23	Liaison R2 <-> R3

Une fois que les opérations des sections suivantes auront été effectuées, la consultation de la table CAM du commutateur donnera les informations suivantes :

```
$ sudo ovs-appctl fdb/show dsw-host
[sudo] password for latu:
 port  VLAN  MAC                Age
   11    12  ba:ad:ca:fe:00:02   10
   12    23  ba:ad:ca:fe:00:03    9
   12    13  ba:ad:ca:fe:00:03    9
   10    13  ba:ad:ca:fe:0d:01    9
   10    12  ba:ad:ca:fe:0c:01    4
   11    23  ba:ad:ca:fe:00:02    1
    1     0  de:56:80:40:1d:ed    0
    2     0  42:d9:5d:6c:89:d8    0
```

11.2. Préparation des routeurs

Sachant que l'on dispose d'une image master sur laquelle est installée un système Debian GNU/Linux de base, on créé 3 images différentielles qui vont correspondre aux instances de routeurs. Voici un exemple de suite de commandes pour la mise en place des images.

```
:~$ mkdir -p ~/vm/ospf
:~$ cd vm
:~/vm$ wget http://www.stri/vm/vm0-debian-testing-i386-base.qed
<snipped/>
:~/vm$ cd ospf
:~/vm/ospf$ ../scripts/diff-img.sh ../vm0-debian-testing-i386-base.qed r1.qed
:~/vm/ospf$ ../scripts/diff-img.sh ../vm0-debian-testing-i386-base.qed r2.qed
:~/vm/ospf$ ../scripts/diff-img.sh ../vm0-debian-testing-i386-base.qed r3.qed
```

Le code du script de création de fichier image différentiel est fourni dans le guide Virtualisation système et enseignement.

Ensuite, on créé un nouveau script shell de lancement des instances de «routeurs» dans lequel on fixe les paramètres d'initialisation de ces mêmes «routeurs».

Avertissement

Ce script ne doit être lancé qu'après l'initialisation du commutateur virtuel pour que le brassage des routeurs sur les ports du commutateur puisse se faire correctement.

```
$ cd ~/vm/ospf
$ cat ospf-startup.sh
#!/bin/bash

../scripts/ovs-startup.sh r1.qed 512 1

../scripts/ovs-startup.sh r2.qed 512 2

../scripts/ovs-startup.sh r3.qed 512 3
```

11.3. Table de routage du système hôte

Pour que les réseaux de l'aire OSPF puissent communiquer avec le système hôte et l'Internet, il est nécessaire de compléter la table de routage du système hôte. La technique usuelle employée pour répondre au besoin consiste à implanter une route statique avec un «super réseau» qui rassemble tous les réseaux de l'aire en une seule entrée.

L'aire OSPF étudiée contient tous les réseaux du Tableau 12, « Affectation des rôles, des numéros de VLANs et des adresses IP ». L'utilisation de l'outil ipcalc permet de vérifier qu'un masque de 19 bits permet d'englober ces réseaux en une seule entrée.

```
$ ipcalc 10.1.0.0/19
Address:   10.1.0.0             00001010.00000001.000 00000.00000000
Netmask:   255.255.224.0 = 19   11111111.11111111.111 00000.00000000
Wildcard:  0.0.31.255           00000000.00000000.000 11111.11111111
=>
Network:   10.1.0.0/19          00001010.00000001.000 00000.00000000
HostMin:   10.1.0.1             00001010.00000001.000 00000.00000001
HostMax:   10.1.31.254          00001010.00000001.000 11111.11111110
Broadcast: 10.1.31.255          00001010.00000001.000 11111.11111111
Hosts/Net: 8190                  Class A, Private Internet
```

On complète donc la table de routage du système hôte avec l'instruction suivante :

```
# ip route add 10.1.0.0/19 via 192.0.2.10
```

12. Consulter les documents de référence

Architecture réseau des travaux pratiques

Le support Architecture réseau des travaux pratiques présente la topologie physique des salles de travaux pratiques avec la Disposition des équipements dans l'armoire de brassage ainsi que les configurations par défaut des équipements. On y trouve aussi le plan d'adressage IP utilisé avec les autres supports de travaux pratiques, le plan de numérotations des VLANs et les affectations des groupes de ports des commutateurs.

Configuration d'une interface réseau

Le support Configuration d'une interface de réseau local présente les opérations de configuration d'une interface réseau et propose quelques manipulations sur les protocoles de la pile TCP/IP

Introduction au routage inter-VLAN

Le support Routage Inter-VLAN introduit le principe du routage inter-VLAN ainsi que ses conditions d'utilisation. C'est aussi un support de travaux pratiques dans lequel on n'utilise que du routage statique et de la traduction d'adresses sources (S-NAT) pour acheminer le trafic utilisateur entre les différents réseaux.

Index des commandes Quagga

La page à l'adresse : Quagga Command Index liste toutes les instructions utilisables dans console unifiée vtysh.

Ces commandes sont largement inspirées du système d'exploitation IOS de Cisco™.

Virtualisation système et enseignement

Le support Virtualisation système et enseignement présente la solution de virtualisation intégrée au noyau Linux : KVM. Associée au commutateur Open vSwitch, cette solution permet de construire des maquettes de travaux pratiques très complètes en offrant de nombreuses fonctions réseau «réelles» dont une table Content-addressable memory.