

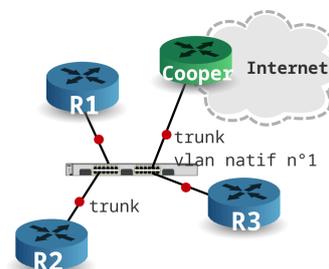
# Introduction au routage dynamique OSPF avec Bird

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 le logiciel Bird.



## Table des matières

1. Copyright et Licence .....	2
1.1. Méta-information .....	2
1.2. Conventions typographiques .....	2
2. Préparer les systèmes pour le routage IPv4 et IPv6 .....	3
3. Valider les communications entre routeurs .....	5
4. Configurer les démons OSPF Bird .....	7
5. Échanger les routes entre Bird et le système .....	15
6. Publier les routes par défaut via OSPF .....	17
7. Consulter les documents de référence .....	21

## 1. Copyright et Licence

---

Copyright (c) 2000,2024 Philippe Latu.  
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Copyright (c) 2000,2024 Philippe Latu.  
Permission est accordée de copier, distribuer et/ou modifier ce document selon les termes de la Licence de Documentation Libre GNU (GNU Free Documentation License), version 1.3 ou toute version ultérieure publiée par la Free Software Foundation ; sans Sections Invariables ; sans Texte de Première de Couverture, et sans Texte de Quatrième de Couverture. Une copie de la présente Licence est incluse dans la section intitulée « Licence de Documentation Libre GNU ».

### 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-bird.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
- bird - Internet Routing Daemon

### 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. 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 le paquet bird avant de brasser les postes sur les commutateurs attribués avec le plan d'adressage de la salle de travaux pratiques.

```
$ aptitude search ~ibird
i   bird                - démon de routage internet
```

Sans configuration particulière, les services bird et bird6 sont lancés.

```
R1:~# systemctl status bird
# bird.service - BIRD Internet Routing Daemon (IPv4)
  Loaded: loaded (/lib/systemd/system/bird.service; enabled; vendor preset: enabled)
  Active: active (running) since Sat 2018-10-20 12:58:52 UTC; 2min 23s ago
  Main PID: 751 (bird)
  Memory: 632.0K
  CGroup: /system.slice/bird.service
          └─751 /usr/sbin/bird -f -u bird -g bird

oct. 20 12:58:52 R1 systemd[1]: Starting BIRD Internet Routing Daemon (IPv4)...
oct. 20 12:58:52 R1 systemd[1]: Started BIRD Internet Routing Daemon (IPv4).
oct. 20 12:58:52 R1 bird[751]: Started
```

```
R1:~# systemctl status bird6
# bird6.service - BIRD Internet Routing Daemon (IPv6)
  Loaded: loaded (/lib/systemd/system/bird6.service; enabled; vendor preset: enabled)
  Active: active (running) since Sat 2018-10-20 17:18:50 UTC; 3h 9min ago
  Process: 1728 ExecStartPre=/usr/sbin/bird6 -p (code=exited, status=0/SUCCESS)
  Process: 1722 ExecStartPre=/usr/lib/bird/prepare-environment (code=exited, status=0/SUCCESS)
  Main PID: 1729 (bird6)
  Memory: 788.0K
  CGroup: /system.slice/bird6.service
          └─1729 /usr/sbin/bird6 -f -u bird -g bird

oct. 20 17:18:50 R1 systemd[1]: Starting BIRD Internet Routing Daemon (IPv6)...
oct. 20 17:18:50 R1 systemd[1]: Started BIRD Internet Routing Daemon (IPv6).
oct. 20 17:18:50 R1 bird6[1729]: Started
```

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.

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.

Il est aussi possible d'éditer le fichier `/etc/network/interfaces` de façon à rendre cette configuration permanente. Voici une copie de ce fichier pour le routeur R1.

```
# The loopback network interface
auto lo
iface lo inet loopback
```

```
# The primary network interface
auto enp0s1
iface enp0s1 inet manual
  up ip link set dev $IFACE up
  down ip link set dev $IFACE down

# uplink
auto enp0s1.360
iface enp0s1.360 inet static
  address 192.168.104.130/27
  gateway 192.168.104.129
  dns-nameserver 172.16.0.2

iface enp0s1.360 inet6 static
  address 2001:678:3fc:168::2/64
  gateway fe80:168::1

# R1 -> R2
auto enp0s1.480
iface enp0s1.480 inet static
  address 10.48.0.1/29

# R1 -> R3
auto enp0s1.481
iface enp0s1.481 inet static
  address 10.48.1.1/29

# R1 -> lxd
auto asw-host
iface asw-host inet manual
  ovs_type OVSBridge
  ovs_ports sw-vlan10
  up ip link set dev $IFACE up
  down ip link set dev $IFACE down

allow-asw-host sw-vlan10
iface sw-vlan10 inet static
  ovs_type OVSBridge
  ovs_bridge asw-host
  ovs_options asw-host 10
  address 10.10.0.1/24

iface sw-vlan10 inet6 static
  ovs_type OVSBridge
  ovs_bridge asw-host
  ovs_options asw-host 10
  address fd14:ca46:3864:a::1/64
```

### 3. 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
```

```
R1:~# ping -qc2 2001:678:3fc:c::2
PING 2001:678:3fc:c::2(2001:678:3fc:c::2) 56 data bytes

--- 2001:678:3fc:c::2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 19ms
rtt min/avg/max/mdev = 0.070/0.295/0.521/0.226 ms
```

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

- Q3. Comment visualiser la table de routage au niveau système ?

Utiliser la commande `ip` pour visualiser la table de routage

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 - 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 192.0.2.7
```

```
R1:~# ip -6 route ls
2001:678:3fc:a::/64 dev eth0 proto kernel metric 256 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
```

#### Routeur R2 - niveau système

```
R2:~# ip route ls
10.1.12.0/26 dev eth0.12 proto kernel scope link src 10.1.12.2
10.1.23.0/26 dev eth0.23 proto kernel scope link src 10.1.23.2
```

```
R2:~# ip -6 route ls
2001:678:3fc:c::/64 dev eth0.12 proto kernel metric 256 pref medium
2001:678:3fc:17::/64 dev eth0.23 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 eth0.23 proto kernel metric 256 pref medium
```

#### Routeur R3 - niveau système

```
R3:~# ip route ls
10.1.13.0/26 dev eth0.13 proto kernel scope link src 10.1.13.3
10.1.23.0/26 dev eth0.23 proto kernel scope link src 10.1.23.3
```

```
R3:~# ip -6 route ls
2001:678:3fc:d::/64 dev eth0.13 proto kernel metric 256 pref medium
2001:678:3fc:17::/64 dev eth0.23 proto kernel metric 256 pref medium
fe80::/64 dev eth0 proto kernel metric 256 pref medium
fe80::/64 dev eth0.13 proto kernel metric 256 pref medium
fe80::/64 dev eth0.23 proto kernel metric 256 pref medium
```

#### 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 le protocole de routage dynamique puisse fonctionner normalement.

Si cette fonction n'est pas active dans le noyau Linux, aucune décision d'acheminement 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](#).

## 4. Configurer les démons OSPF Bird

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. Quels sont les fichiers de configuration à éditer pour activer les protocoles OSPFv2 et OSPFv3 sur le routeur ?

Une fois le paquet `bird` installé, deux démons distincts sont lancés : `bird` pour IPv4 et `bird6` pour IPv6. Rechercher dans la liste des fichiers fournis avec le paquet, les exemples de fichiers de configuration.

Les fichiers de configuration sont au nombre de deux. Ils sont placés dans le dossier `/etc/bird/`. Deux exemples de ces fichiers sont données dans le dossier de documentation du paquet.

```
R1:~# dpkg -L bird | grep example
/usr/share/doc/bird/examples
/usr/share/doc/bird/examples/bird.conf.gz
/usr/share/doc/bird/examples/bird6.conf.gz
```

- Q6. Comment accéder à l'état des différents protocoles actifs pour chaque démon ?

À chaque édition d'un fichier de configuration, il faut relancer le démon correspondant. C'est à nouveau dans la liste des fichiers du paquet que l'on identifie les outils d'accès à la configuration active des deux démons.

Il faut consulter la section **Remote control** de la documentation Bird. Les commandes utiles pour cette question sont les suivantes.

```
show status
show protocols
```

Chaque démon dispose d'une console propre avec les outils `birdc` et `birdc6`. Ce sont ces deux consoles qui permettent de connaître le statut du démon, la liste des protocoles actifs et les informations relatives au fonctionnement de ces protocoles.

```
R1:~# birdc
BIRD 1.6.4 ready.
bird> sh status
BIRD 1.6.4
Router ID is 0.0.4.1
Current server time is 2018-10-20 20:59:04
Last reboot on 2018-10-20 16:54:04
Last reconfiguration on 2018-10-20 20:58:58
Daemon is up and running
```

```
R1:~# birdc6
BIRD 1.6.4 ready.
bird> sh status
BIRD 1.6.4
Router ID is 0.0.6.1
Current server time is 2018-10-20 21:00:22
Last reboot on 2018-10-20 17:18:49
Last reconfiguration on 2018-10-20 17:18:49
Daemon is up and running
```

De la même façon, on peut connaître la liste des protocoles actifs de chaque démon.

```
bird> sh protocols
name  proto  table  state  since      info
device1 Device  master up      16:54:04
kernel1 Kernel  master up      20:58:58
```

- Q7. Comment activer le protocole de routage OSPF et attribuer l'identifiant du routeur ?

Consulter le document **BIRD User's Guide** à la section OSPF pour activer le protocole. Consulter les tableaux des plans d'adressage pour obtenir la valeur de l'identifiant du routeur à configurer.

On édite les fichiers `/etc/bird.conf` et `/etc/bird6.conf` avec les paramètres suivants.

```
router id IPv4 address
protocol ospf <name>
area <id>
```

Voici une copie des fichiers du routeur R1.

```
R1:~# grep -v ^# /etc/bird/bird.conf
```

```
router id 0.0.1.4;

protocol kernel {
    scan time 10;
    import none;
}

protocol device {
    scan time 10;
}

protocol ospf OSPFv2R1 {
    area 0 {
        };
    }
}
```

```
R1:~# grep -v ^# /etc/bird/bird6.conf
```

```
router id 0.0.1.6;

protocol kernel {
    scan time 10;
    import none;
}

protocol device {
    scan time 10;
}

protocol ospf OSPFv3R1 {
    area 0 {
        };
    }
}
```

Une fois les deux services relancés, on peut vérifier que les éléments demandés sont bien présents dans la configuration des démons de routage OSPF.

```
R1:~# systemctl restart bird
```

```
R1:~# birdc sh protocols
```

```
BIRD 1.6.4 ready.
```

name	proto	table	state	since	info
kernel1	Kernel	master	up	09:38:34	
device1	Device	master	up	09:38:34	
<b>OSPFv2R1</b>	<b>OSPF</b>	<b>master</b>	<b>up</b>	<b>09:38:34</b>	<b>Alone</b>

```
R1:~# birdc sh ospf state
```

```
BIRD 1.6.4 ready.
```

```
area 0.0.0.0
```

```
    router 0.0.1.4
        distance 0
```

```

R1:~# systemctl restart bird6

R1:~# birdc6 sh protocols
BIRD 1.6.4 ready.
name      proto  table  state  since      info
kernel1  Kernel  master  up     09:42:33
device1   Device  master  up     09:42:33
OSPFv3R1 OSPF   master  up     09:42:33  Alone

R1:~# birdc6 sh ospf state
BIRD 1.6.4 ready.

area 0.0.0.0

    router 0.0.1.6
        distance 0

```

- Q8. Comment activer et valider le protocole de routage OSPF pour les réseaux IPv4 et IPv6 connus de chaque routeur ?

Consulter la section OSPF de la documentation [BIRD User's Guide](#) ainsi que l'exemple [OSPF example](#). Il suffit d'adapter les exemples avec les noms d'interfaces en fonction du contexte.

On édite les fichiers `/etc/bird.conf` et `/etc/bird6.conf` avec les paramètres suivants.

```

interface <interface pattern>
authentication none|simple|cryptographic;

```

On vérifie au niveau console (Voir [Remote control](#)) l'état de la base de connaissance des deux processus OSPF avec la commande suivante.

```
show ospf state
```

#### Routeur R1 : OSPFv2 & interfaces

```

R1:~# grep -v ^# /etc/bird/bird.conf

router id 0.0.1.4;

protocol kernel {
    scan time 10;
    import none;
}

protocol device {
    scan time 10;
}

protocol ospf OSPFv2R1 {
    rfc1583compat yes;
    area 0 {
        interface "eth0.12", "eth0.13" {
            authentication none;
        };
    };
}

```

#### Routeur R1 : base de connaissance OSPFv2

Dans la copie d'écran ci-dessous, on relève les deux routeurs voisins de R1 ainsi que le réseau distant 10.1.23.0/26.

```

R1:~# birdc sh ospf state
BIRD 1.6.4 ready.

area 0.0.0.0

    router 0.0.1.4
        distance 0
        network 10.1.12.0/26 metric 10
        network 10.1.13.0/26 metric 10

```

```

router 0.0.2.4
  distance 10
  network 10.1.12.0/26 metric 10
  network 10.1.23.0/26 metric 10

router 0.0.3.4
  distance 10
  network 10.1.13.0/26 metric 10
  network 10.1.23.0/26 metric 10

network 10.1.12.0/26
  dr 0.0.2.4
  distance 10
  router 0.0.2.4
  router 0.0.1.4

network 10.1.13.0/26
  dr 0.0.3.4
  distance 10
  router 0.0.3.4
  router 0.0.1.4

network 10.1.23.0/26
  dr 0.0.2.4
  distance 20
  router 0.0.2.4
  router 0.0.3.4

```

### Routeur R1 : OSPFv3 & interfaces

```

R1:~# grep -v ^# /etc/bird/bird6.conf

router id 0.0.1.6;

protocol kernel {
  scan time 10;
  import none;
}

protocol device {
  scan time 10;
}

protocol ospf OSPFv3R1 {
  area 0 {
    interface "eth0.12", "eth0.13" {
      authentication none;
    };
  };
}

```

### Routeur R1 : base de connaissance OSPFv3

Dans la copie d'écran ci-dessous, on relève les deux routeurs voisins de R1 ainsi que le réseau distant 2001:678:3fc:17::/64.

```

R1:~# birdc6 sh ospf state
BIRD 1.6.4 ready.

area 0.0.0.0

  router 0.0.1.6
    distance 0
    network [0.0.1.6-2] metric 10
    network [0.0.1.6-3] metric 10

  router 0.0.2.6
    distance 10
    network [0.0.1.6-2] metric 10
    network [0.0.3.6-3] metric 10

```

```

router 0.0.3.6
  distance 10
  network [0.0.1.6-3] metric 10
  network [0.0.3.6-3] metric 10

network [0.0.1.6-2]
  distance 10
  router 0.0.1.6
  router 0.0.2.6
  address 2001:678:3fc:c::/64

network [0.0.1.6-3]
  distance 10
  router 0.0.1.6
  router 0.0.3.6
  address 2001:678:3fc:d::/64

network [0.0.3.6-3]
  distance 20
  router 0.0.3.6
  router 0.0.2.6
  address 2001:678:3fc:17::/64

```

- Q9. Comment identifier le type de réseau des interfaces actives d'un routeur pour chaque version du protocole de routage OSPF ?

La question précédente montre que la configuration des deux processus `bird` et `bird6` est basée sur l'activation du protocole par interface. Il faut donc rechercher dans la section **Remote control** l'instruction qui donne l'état des interfaces actives.

```
show ospf interface
```

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

#### Routeur R1 : OSPFv2 & interfaces

```

R1:~# birdc sh ospf interface
BIRD 1.6.4 ready.
OSPFv2R1:
Interface eth0.12 (10.1.12.0/26)
  Type: broadcast
  Area: 0.0.0.0 (0)
  State: Backup
  Priority: 1
  Cost: 10
  Hello timer: 10
  Wait timer: 40
  Dead timer: 40
  Retransmit timer: 5
  Designated router (ID): 0.0.2.4
  Designated router (IP): 10.1.12.2
  Backup designated router (ID): 0.0.1.4
  Backup designated router (IP): 10.1.12.1
Interface eth0.13 (10.1.13.0/26)
  Type: broadcast
  Area: 0.0.0.0 (0)
  State: Backup
  Priority: 1
  Cost: 10
  Hello timer: 10
  Wait timer: 40
  Dead timer: 40
  Retransmit timer: 5
  Designated router (ID): 0.0.3.4
  Designated router (IP): 10.1.13.3
  Backup designated router (ID): 0.0.1.4
  Backup designated router (IP): 10.1.13.1

```

## Routeur R1 : OSPFv3 &amp; interfaces

```

R1:~# birdc6 sh ospf interface
BIRD 1.6.4 ready.
OSPFv3R1:
Interface eth0.12 (IID 0)
  Type: broadcast
  Area: 0.0.0.0 (0)
  State: DR
  Priority: 1
  Cost: 10
  Hello timer: 10
  Wait timer: 40
  Dead timer: 40
  Retransmit timer: 5
  Designated router (ID): 0.0.1.6
  Designated router (IP): fe80::70de:4fff:fe1d:68b4
  Backup designated router (ID): 0.0.2.6
  Backup designated router (IP): fe80::943a:41ff:fe65:7307
Interface eth0.13 (IID 0)
  Type: broadcast
  Area: 0.0.0.0 (0)
  State: DR
  Priority: 1
  Cost: 10
  Hello timer: 10
  Wait timer: 40
  Dead timer: 40
  Retransmit timer: 5
  Designated router (ID): 0.0.1.6
  Designated router (IP): fe80::70de:4fff:fe1d:68b4
  Backup designated router (ID): 0.0.3.6
  Backup designated router (IP): fe80::3032:e9ff:fe73:6322

```

Q10. 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 commande utile de la section **Remote control** est la suivante.

```
show ospf neighbors
```

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:~# birdc sh ospf neighbors
BIRD 1.6.4 ready.
OSPFv2R1:
Router ID      Pri      State      DTime  Interface  Router IP
0.0.2.4       1       Full/DR    00:31  eth0.12   10.1.12.2
0.0.3.4       1       Full/DR    00:34  eth0.13   10.1.13.3

```

```

R1:~# birdc6 sh ospf neighbors
BIRD 1.6.4 ready.
OSPFv3R1:
Router ID      Pri      State      DTime  Interface  Router IP
0.0.2.6       1       Full/BDR   00:33  eth0.12   fe80::943a:41ff:fe65:7307
0.0.3.6       1       Full/BDR   00:36  eth0.13   fe80::3032:e9ff:fe73:6322

```

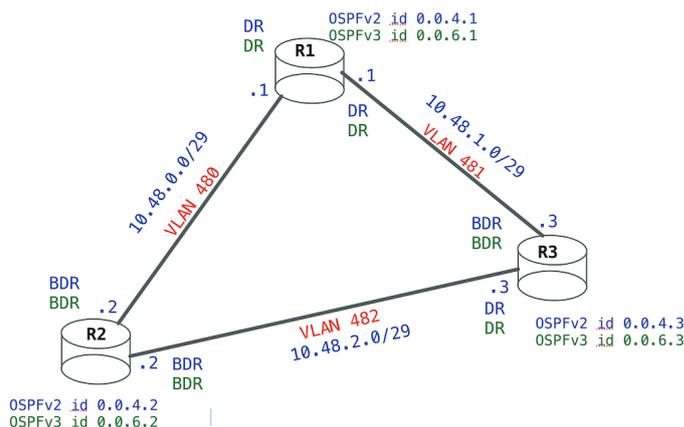
Q11. Comment identifier le rôle des différentes interfaces des routeurs pour chacun des liens du triangle de la topologie logique ?

### ! 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.

Q12. Quelles sont les réseaux IPv4 et IPv6 présents dans la base calcul du protocole OSPF ?

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

La commande utile dans les deux consoles est la suivante.

```
show 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:~# birdc sh route
BIRD 1.6.4 ready.
10.1.12.0/26      dev eth0.12 [OSPFv2R1 16:47:15] * I (150/10) [0.0.2.4]
10.1.13.0/26     dev eth0.13 [OSPFv2R1 16:47:10] * I (150/10) [0.0.3.4]
10.1.23.0/26     via 10.1.13.3 on eth0.13 [OSPFv2R1 16:47:10] * I (150/20) [0.0.2.4]
```

Les valeurs notées entre parenthèses 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 du coût de lien par défaut pour chaque interface active. La valeur par défaut est 10

Les deux premiers réseaux de la table sont joignable via un lien direct ; soit une métrique de 10. Le troisième réseau est joignable via deux liens Ethernet ; d'où la métrique de 20.

Pour les réseaux IPv6, on retrouve les mêmes métriques puisque la topologie est identique pour les deux version du protocole IP.

```
R1:~# birdc6 sh route
BIRD 1.6.4 ready.
2001:678:3fc:d::/64 dev eth0.13 [OSPFv3R1 10:01:30] * I (150/10) [0.0.1.6]
2001:678:3fc:c::/64 dev eth0.12 [OSPFv3R1 10:05:36] * I (150/10) [0.0.1.6]
2001:678:3fc:17::/64 via fe80::3032:e9ff:fe73:6322 on eth0.13 [OSPFv3R1 10:05:38] * I (150/20) [0
```

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

- Q13. Comment utiliser toutes les solutions disponibles pour joindre le réseau distant depuis chacun des sommets de la topologie triangle ?

Avec la topologie logique triangle, le réseau du côté opposé à un sommet (au routeur) doit être joignable depuis les deux réseaux locaux raccordés à ce routeur. Nous sommes donc dans un contexte multi chemins.

Consulter la section OSPF de la documentation [BIRD User's Guide](#) et rechercher l'instruction qui permet l'utilisation de plusieurs chemins à coût égal.

```
ecmp switch [limit number]
```

La fonction réseau du noyau Linux recherchée est connue sous le nom Equal Cost Multi Path ou ECMP.

Sur le routeur R1, la configuration du bloc d'instructions OSPF de chaque processus est la suivante.

```
R1:~# sed -n '/protocol ospf/,//p' /etc/bird/bird.conf
protocol ospf OSPFv2R1 {
    rfc1583compat yes;
    ecmp yes;
    area 0 {
        interface "eth0.12", "eth0.13" {
            authentication none;
        };
    };
}
```

```
R1:~# sed -n '/protocol ospf/,//p' /etc/bird/bird6.conf
protocol ospf OSPFv3R1 {
    ecmp yes;
    area 0 {
        interface "eth0.12", "eth0.13" {
            authentication none;
        };
    };
}
```

Relativement aux questions précédentes, les tables de routage proposées par les processus `bird` et `bird6` font apparaître les deux chemins disponibles pour joindre le réseau distant du sommet de la topologie triangle.

```
R1:~# birdc sh route
BIRD 1.6.4 ready.
10.1.12.0/26 dev eth0.12 [OSPFv2R1 19:49:42] * I (150/10) [0.0.1.4]
10.1.13.0/26 dev eth0.13 [OSPFv2R1 19:50:37] * I (150/10) [0.0.1.4]
10.1.23.0/26 multipath [OSPFv2R1 19:50:48] * I (150/20) [0.0.2.4]
via 10.1.12.2 on eth0.12 weight 1
via 10.1.13.3 on eth0.13 weight 1
```

```
R1:~# birdc6 sh route
BIRD 1.6.4 ready.
2001:678:3fc:d::/64 dev eth0.13 [OSPFv3R1 19:50:38] * I (150/10) [0.0.1.6]
2001:678:3fc:c::/64 dev eth0.12 [OSPFv3R1 19:49:44] * I (150/10) [0.0.1.6]
2001:678:3fc:17::/64 multipath [OSPFv3R1 19:50:50] * I (150/20) [0.0.2.6]
  via fe80::3032:e9ff:fe73:6322 on eth0.13 weight 1
  via fe80::943a:41ff:fe65:7307 on eth0.12 weight 1
```

## 5. Échanger les routes entre Bird et le système

Dans la section précédente, tous les échanges de préfixes réseau IPv4 et IPv6 se font entre les démons Bird installés sur les trois routeurs de la topologie étudiée. Il faut maintenant être capable d'échanger les résultats des traitements OSPFv2 et OSPFv3 avec le sous-système réseau du noyau de chaque routeur.

- Sur R1, les deux démons Bird doivent importer la route par défaut déjà connue au niveau système. De plus, les routes vers les réseaux fictifs de R2 et R3 apprises via OSPF doivent être exportées vers le sous-système réseau du noyau de R1.
- Sur les routeurs R2 et R3, les routes par défaut apprises via OSPF doivent être exportées vers le sous-système réseau du noyau. De plus les routes des réseaux fictifs doivent être importées dans les démons Bird pour être publiées via OSPF.



### Avertissement

Les résultats des questions de cette section ne sont visibles que si les routes sont déjà présentes, soit dans les démons de routage Bird, soit au niveau système.

Q14. Comment faire pour que les routes connues du sous-système réseau du noyau Linux soient importées dans les deux démons `bird` et `bird6` ?

Consulter la section kernel de la documentation [BIRD User's Guide](#) à la recherche des paramètres d'importation. Les commandes utiles pour cette question sont les suivantes.

```
import
learn switch
```

Pour cette question, les configurations des démons `bird` et `bird6` sont identiques quel que soit le routeur considéré. Voici un extrait de fichier de configuration.

```
R1:~# sed -n '/protocol kernel/,/^}/p' /etc/bird/bird.conf
protocol kernel {
    scan time 10;
    import all;
    learn yes;
#    export all;    # Actually insert routes into the kernel routing table
}
```

Tant que les questions sur l'ajout de réseaux fictifs ne sont pas traitées, seul les démons du routeur R1 ont un résultat observable. Les routes par défaut sont importées dans les démons Bird.

```
R1:~# birdc show route 0.0.0.0/0
BIRD 1.6.4 ready.
0.0.0.0/0      via 192.0.2.1 on eth0 [kernel1 13:47:04] * (10)
```

```
R1:~# birdc6 show route ::/0
BIRD 1.6.4 ready.
::/0          via fe80::dc02:44ff:fe64:4834 on eth0 [kernel1 2018-10-21] * (10)
```

Q15. Comment faire pour que les routes calculées par les processus `bird` et `bird6` soient soumises au sous-système réseau du noyau Linux ?

Consulter la section kernel de la documentation [BIRD User's Guide](#) à la recherche des paramètres d'exportation. La commande utile pour cette question est la suivante.

## export

En reprenant l'exemple du routeur R1, la configuration des blocs d'instructions `kernel` de chaque processus devient :

```
R1:~# sed -n '/protocol kernel/,/^}/p' /etc/bird/bird.conf
protocol kernel {
    scan time 10;
    import all;
    learn yes;
    export all; # Actually insert routes into the kernel routing table
}
```

Avec la commande `ip` au niveau système, on voit apparaître les «sources» d'alimentation de la table de routage du système en question : `kernel` et `bird`.

```
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 bird
    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 192.0.2.7
```

```
R1:~# ip -6 route ls
2001:678:3fc:a::/64 dev eth0 proto kernel metric 256 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 bird metric 1024
    nexthop via fe80::3032:e9ff:fe73:6322 dev eth0.13 weight 1
    nexthop via fe80::943a:41ff:fe65:7307 dev eth0.12 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 metric 1024 onlink pref medium
```

## 6. Publier les routes par défaut via OSPF

Dans la topologie logique étudiée, le routeur R1 dispose d'un lien vers l'Internet. On considère ce lien comme 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 deux routeurs R2 et R3.

Avant publication de la route par défaut depuis le routeur R1, les démons OSPF n'utilisent que des annonces LSA (Link State Advertisement) de type 1 et 2. Voici un tableau de référence pour le codage des LSAs.

Tableau 1. Codage des annonces OSPF

OSPFv2	Description	OSPFv3	Description
1	Router LSA	0x2001	Router LSA
2	Network LSA	0x2002	Network LSA
3	Network Summary LSA	0x2003	Inter-Area Prefix LSA for ABRs
4	ASBR Summary LSA	0x2004	Inter-Area Router LSA for ABRs
5	AS-external LSA	0x4005	AS-external LSA
6	Group Membership LSA	0x2006	Group Membership LSA
7	Not So Stubby Area LSA	0x2007	Type-7 LSA
8		0x2008	Link LSA
9		0x2009	Intra-Area Prefix LSA

- ABR : Area Border Router
- ASBR : Autonomous System Border Router

Les listes des annonces connues du routeur R1 avant publication des routes par défaut sont données dans les copies d'écran ci-dessous.

```
R1:~# birdc show ospf lsadb
BIRD 1.6.4 ready.
```

```
Area 0.0.0.0
```

```

Type  LS ID           Router           Sequence  Age  Checksum
0001  0.0.1.4           0.0.1.4         80000148  1465  88f0
0001  0.0.2.4           0.0.2.4         80000146  1466  70f2
0002  10.1.12.2         0.0.2.4         80000003  1466  3439
0001  0.0.3.4           0.0.3.4         80000147  1469  a4b6
0002  10.1.13.3         0.0.3.4         80000003  1470  2148
0002  10.1.23.3         0.0.3.4         80000003   375  bf9e
```

```
R1:~# birdc6 show ospf lsadb
BIRD 1.6.4 ready.
```

```
Area 0.0.0.0
```

```

Type  LS ID           Router           Sequence  Age  Checksum
2001  0.0.0.0           0.0.1.6         8000015c  1643  116b
2009  0.0.0.0           0.0.1.6         8000015c  1643  7e2b
2001  0.0.0.0           0.0.2.6         80000151  1647  4243
2009  0.0.0.0           0.0.2.6         80000151  1647  9a18
2002  0.0.0.2           0.0.2.6         80000001  1647  9976
2009  0.0.0.2           0.0.2.6         80000001  1647  a95c
2001  0.0.0.0           0.0.3.6         80000153  1646  4c35
```

```

2009 0.0.0.0      0.0.3.6      80000153 1646 9c12
2002 0.0.0.2      0.0.3.6      80000001 1646 9b72
2009 0.0.0.2      0.0.3.6      80000001 1646 c939
2002 0.0.0.3      0.0.3.6      80000003 575 9a6f
2009 0.0.0.3      0.0.3.6      80000003 575 ca2a

```

Link eth0.12

```

Type  LS ID      Router      Sequence  Age  Checksum
0008  0.0.0.2    0.0.1.6    80000137 1648 c5df
0008  0.0.0.2    0.0.2.6    8000012e 575 5640

```

Link eth0.13

```

Type  LS ID      Router      Sequence  Age  Checksum
0008  0.0.0.3    0.0.1.6    80000137 1646 e1c1
0008  0.0.0.2    0.0.3.6    8000012e 610 62dc

```

Q16. Quelle est la condition préalable à respecter pour que le routeur R1 soit en mesure de publier une route par défaut via OSPF ?

Avant de procéder à l'importation de route dans les démons Bird, on doit s'assurer de la présence des deux routes par défaut IPv4 et IPv6 dans les tables de routage au niveau système.

Sur le routeur R1 uniquement, on valide la présence des routes par défaut.

```

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 metric 1024 onlink pref medium

```

Q17. Comment valider l'importation des routes par défaut dans les deux démons bird et bird6 ?

L'importation des routes depuis le niveau système dans les démons Bird a été traitée à la [Section 5, « Échanger les routes entre Bird et le système »](#). Ici, on se contente de vérifier la présence des routes par défaut au niveau des consoles de chaque démon.

On peut spécifier le préfixe réseau directement dans l'affichage de la table de routage de chaque démon.

```

R1:~# birdc show route 0.0.0.0/0
BIRD 1.6.4 ready.
0.0.0.0/0      via 192.0.2.1 on eth0 [kernel1 2018-10-27] * (10)

R1:~# birdc6 show route ::/0
BIRD 1.6.4 ready.
::/0          via fe80::dc02:44ff:fe64:4834 on eth0 [kernel1 2018-10-27] * (10)

```

Q18. Comment créer les filtres qui serviront à exporter les routes par défaut dans la configuration de chaque démon pour les protocoles OSPFv2 et OSPFv3 ?

Il faut consulter la documentation [BIRD User's Guide](#) aux sections Filters et OSPF pour trouver des exemples de syntaxe.

Voici une copie d'écran pour chaque démon.

```

R1:~# sed -n '/^filter export_OSPF/,/^}/p' /etc/bird/bird.conf
filter export_OSPF {
    if ( net = 0.0.0.0/0 ) then {
        ospf_metric2 = 100;
        accept;
    }
    reject;
}

```

```
R1:~# sed -n '/^filter export_OSPF/,/^}/p' /etc/bird/bird6.conf
filter export_OSPF {
    if ( net = ::/0 ) then {
        ospf_metric2 = 100;
        accept;
    }
    reject;
}
```

- Q19. Comment appliquer les filtres de la question précédente pour que les routes par défaut soient exportées via OSPF à destination des autres routeurs ?

Il faut consulter la documentation [BIRD User's Guide](#) à la section OSPF et rechercher un exemple de la directive suivante.

export

Voici une copie d'écran pour chaque démon avec l'exportation dans le processus OSPF en fonction du filtre défini préalablement.

```
R1:~# sed -n '/^protocol ospf/,/^}/p' /etc/bird/bird.conf
protocol ospf OSPFv2R1 {
    rfc1583compat yes;
    ecmp yes;
    export filter export_OSPF;
    area 0 {
        interface "eth0.12", "eth0.13" {
            authentication none;
        }
    }
}
```

```
R1:~# sed -n '/^protocol ospf/,/^}/p' /etc/bird/bird6.conf
protocol ospf OSPFv3R1 {
    ecmp yes;
    export filter export_OSPF;
    area 0 {
        interface "eth0.12", "eth0.13" {
            authentication none;
        }
    }
}
```

- Q20. Quelles sont les nouvelles annonces LSA apparues après exportation des routes par défaut depuis R1 vers les deux autres routeurs de la topologie triangle ?

À partir du [Tableau 1, « Codage des annonces OSPF »](#) donné en début de section, donner le nouveau rôle du routeur R1.

Une fois que l'exportation des routes par défaut dans OSPF est effective sur R1, ce routeur devient Autonomous System Border Router ou ASBR. Dès lors, il émet des annonces de type 5 que l'on peut identifier dans les bases de chacun des trois routeurs de l'aire OSPF.

Par exemple, on obtient les résultats suivants sur le routeur R2.

```
R2:~# birdc show ospf lsadb type 5
BIRD 1.6.4 ready.

Global

Type  LS ID          Router           Sequence  Age  Checksum
0005  0.0.0.0            0.0.1.4         80000001  1570  afb9
```

```
R2:~# birdc6 show ospf lsadb type 5
BIRD 1.6.4 ready.

Global

Type  LS ID          Router           Sequence  Age  Checksum
4005  0.0.0.0            0.0.1.6         80000001  1574  e4c8
```

- Q21. Comment valider l'exportation des routes par défaut depuis les deux démons `bird` et `bird6` vers le niveau système sur les routeurs R2 et R3 ?

L'exportation des routes depuis les démons Bird vers le système a été traitée à la [Section 5, « Échanger les routes entre Bird et le système »](#). Ici, on se contente de vérifier la présence des routes par défaut au niveau système sur R2 et R3.

Voici une copie d'écran pour le routeur R2 qui caractérise le fait que les routes par défaut ont été apprises via Bird.

```
R2:~# ip route ls default
default via 10.1.12.1 dev eth0.12 proto bird

R2:~# ip -6 route ls default
default via fe80::70de:4fff:fe1d:68b4 dev eth0.12 proto bird metric 1024 pref medium
```

Si la table de routage du routeur d'accès à Internet contient les routes statiques vers les réseaux de l'aire OSPF, il est possible de lancer les tests ICMP classiques. Voici deux exemples depuis le routeur R2.

```
R2:~# ping -qc2 9.9.9.9
PING 9.9.9.9 (9.9.9.9) 56(84) bytes of data.

--- 9.9.9.9 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 2ms
rtt min/avg/max/mdev = 13.796/13.910/14.024/0.114 ms

R2:~# ping -qc2 2620:fe::fe
PING 2620:fe::fe(2620:fe::fe) 56 data bytes

--- 2620:fe::fe ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 3ms
rtt min/avg/max/mdev = 43.986/44.802/45.618/0.816 ms
```

## 7. Consulter les documents de référence

---

### 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.