Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Projet Intégré

Rapport Final

Auteurs :
Marc Roten
Julien Borgognon
Bryan Perdrizat
Davide Anthony Previte

Professeur :
François Buntschu

20 septembre 2018

Table des matières

1 Introduction

Ce rapport fait suite à la conception et design réalisés en amont. Le design nous a aidé à voir à quoi devrait ressembler notre réseau final, ou dans notre cas, lors de la pr

1.1 Cahier des charges

1 Utilisation native d' IPv4 et IPv6
2 Attribution dynamique des adresses en IPv4 et IPv6
3 connexion internet haut débit avec IPv4 et IPv6 pour le client (note : Berlin n'est qu'en IPv6 et doit pouvoir se connecter sur des sites IPv4
4 Routage dynamique entre Fribourg et Berlin (Internet Service Providers (ISP))
5 Routage dynamique "privé" entre Fribourg et Berlin (Client)
6 Mise en place cloud OpenStack
7 Gestion du domaine DNS fri-learning.ch sur une VM du cloud
8 Mise en place d'un serveur web www.fri-learning.ch comme vitrine de l'entreprise
9 Spécification de la structure LAN et de l'accès sans fil au sein des bâtiments du siège principal du client
10 Spécification de l'architecture réseau de l'ISP

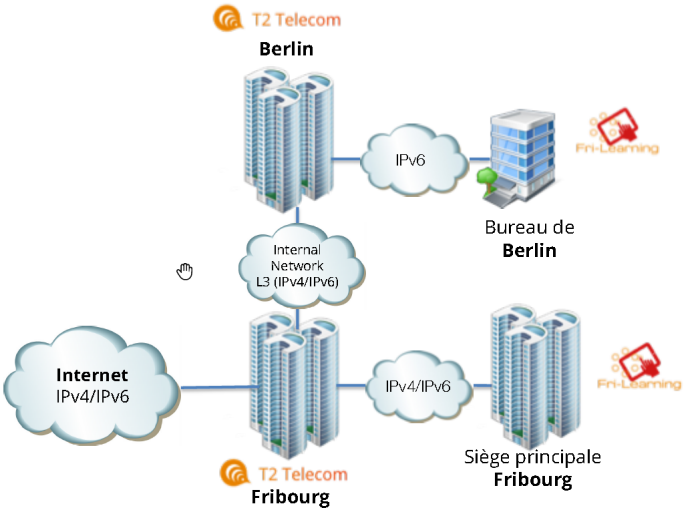

FIGURE 1: Données du projet

2 Rappel de la maquette

Depuis la conception de notre réseau à maintenant, il y a eu de petites modifications. Pour comprendre notre configuration, un petit rappel de notre infrastructure est nécessaire. Nous la mettons à disposition sur les figures 3 et 2. La figure 2 est un mélange entre la maquette logique et la maquette physique. C'est à dire que c'est la maquette physique, mais avec des adresses IP qui ne devraient pas apparaître sur celle-ci.

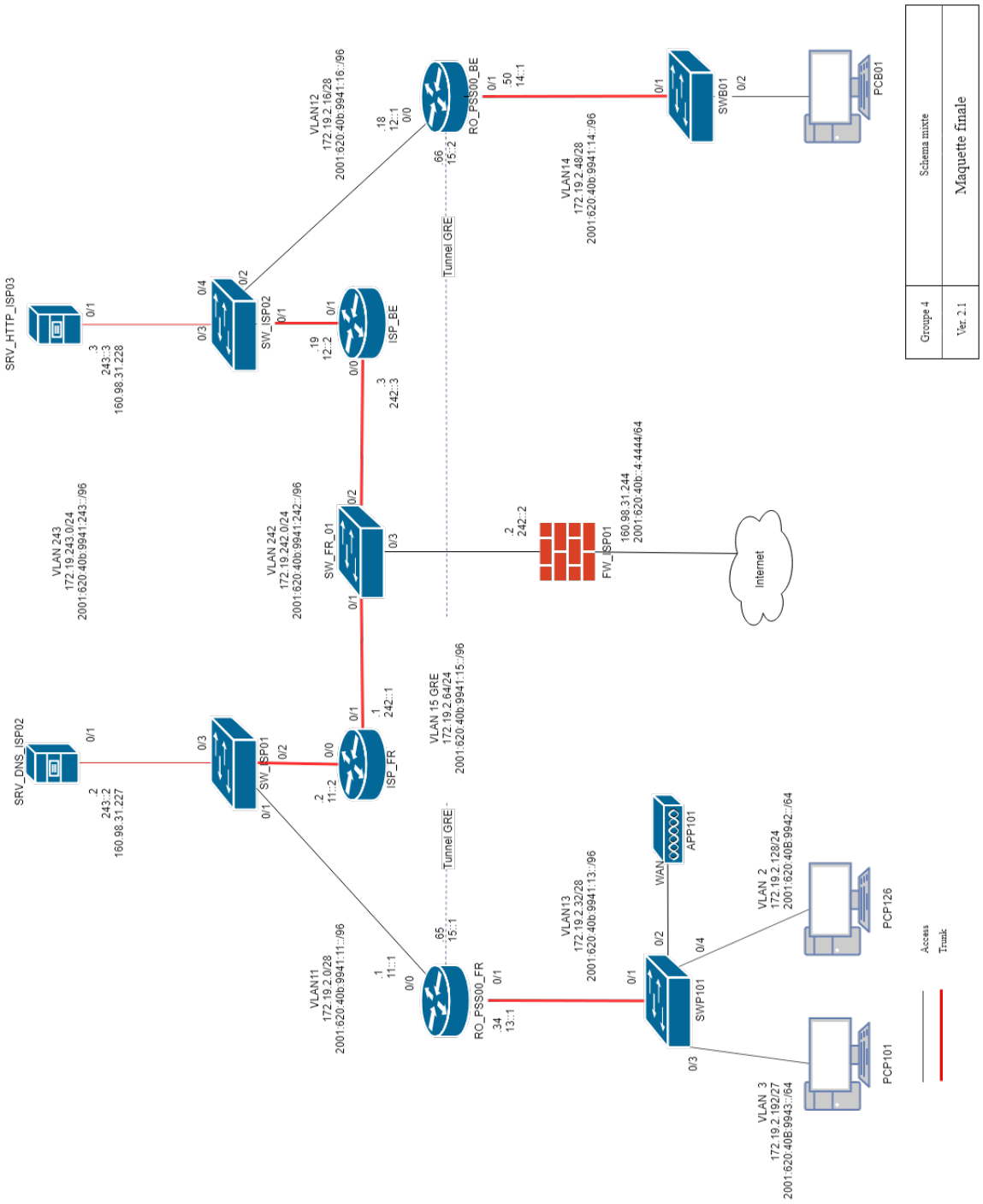

FIGURE 2: Maquette mixte entre logique et physique

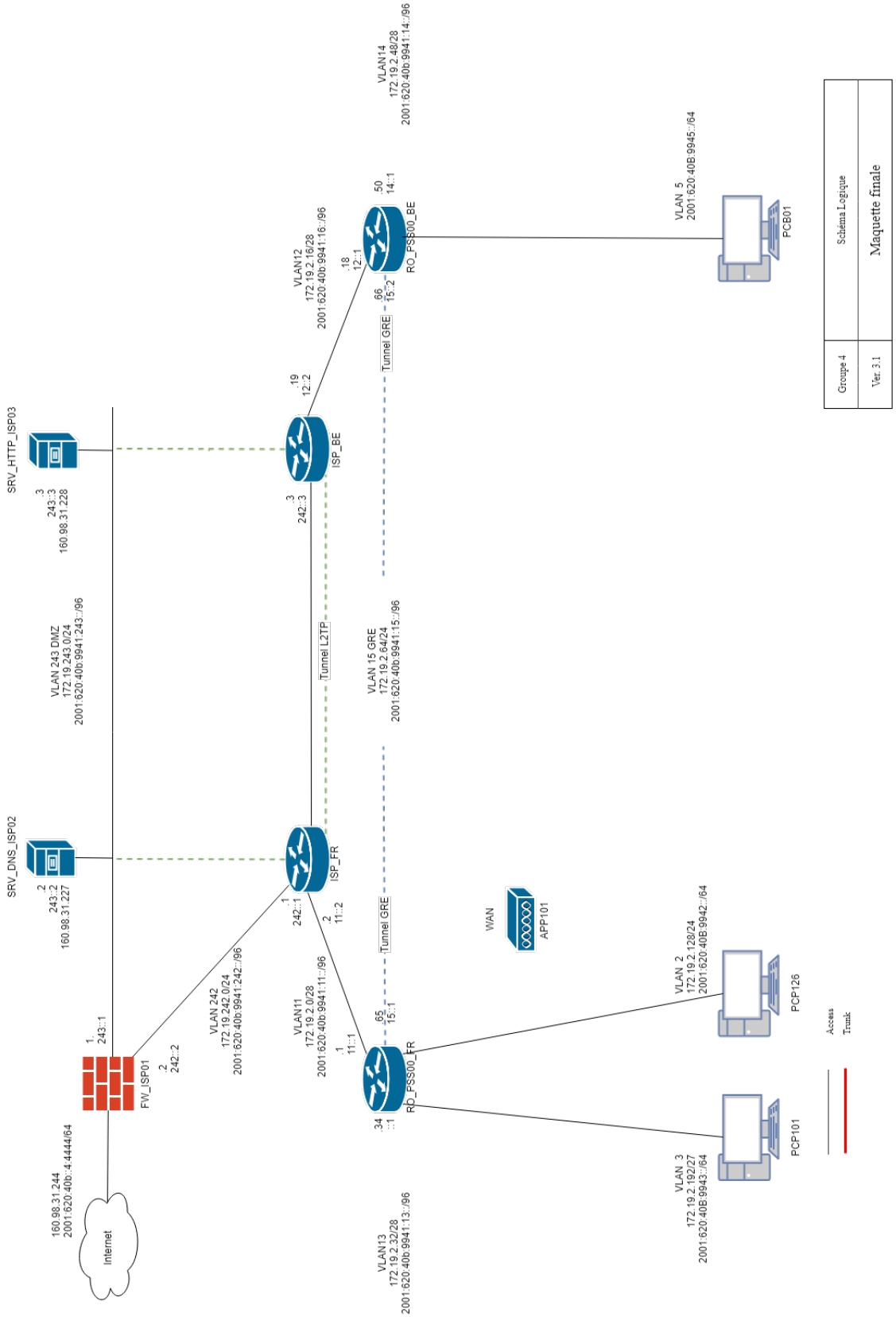

FIGURE 3: Maquette logique

3 Serveur DNS

Nous avons décidé d'installer notre serveur Domaine Name Server (DNS) sur une distribution Linux, version 16.04 LTS. Le service DNS pour les systèmes Ubuntu s'appelle Bind9. Ce chapitre expliquera le déroulement de l'installation de Bind9.

3.1 Installation

Il faut commencer par installer Bind9

Installer Bind9

```
sudo apt-get install bind9 bind9utils bind9-doc
```

Il faut, ensuite, définir la zone forward et la zone reverse. Pour ce faire il faut éditer le fichier name.conf.local

Édition named.conf

```
sudo nano /etc/bind/named.conf
```

Fichier named.conf après modification

```
zone "pri.fri-learning.ch␣" {
      type master;
      file "/etc/bind/for.fri-learning.ch";
      allow-transfer { 172.19.243.2; };
      also-notify { 172.19.243.2; };
};
zone "243.19.172.in-addr.arpa" {
      type master;
      file "/etc/bind/rev.fri-learning.ch";
      allow-transfer { 172.19.243.2; };
      also-notify { 172.19.243.2; };
};
```

Une fois le fichier sauvegarder, il faut créer les fichiers des zones que nous avons défini dans l'étape précédente.

Pour la *zone forward*, il faut éditer le fichier suivant :

Édition /etc/bind/for.fri-learning.ch

```
sudo nano /etc/bind/for.fri-learning.ch
```

Fichier /etc/bind/for.fri-learning.ch

```
$TTL 86400
@   IN  SOA     pri.fri-learning.ch. root.fri-learning.ch. (
      2011071001  ;Serial
      3600        ;Refresh
      1800        ;Retry
      604800      ;Expire
      86400       ;Minimum TTL
)
@       IN  NS          pri.fri-learning.ch.
@       IN  A           160.98.31.228
pri     IN  A           160.98.31.228
```

Pour la zone reverse, il faut éditer le fichier suivant :

Édition /etc/bind/re.fri-learning.ch

```
sudo nano /etc/bind/rev.fri-learning.ch
```

Fichier /etc/bind/for.fri-learning.ch

```
$TTL 86400
@   IN  SOA     pri.fri-learning.ch. root.fri-learning.ch. (
      2011071002  ;Serial
      3600        ;Refresh
      1800        ;Retry
      604800      ;Expire
      86400       ;Minimum TTL
)
@       IN  NS          pri.fri-learning.ch.
pri     IN  A           160.98.31.228
2       IN  PTR         pri.fri-learning.ch.
```

Ensuite, il faut définir les permissions pour le répertoire Bind9.

Définition des permissions

```
sudo chmod -R 755 /etc/bind
sudo chown -R bind:bind /etc/bind
```

Pour vérifier si la configuration du DNS est correcte, il faut exécuter les commandes suivantes :

Vérification de la configuration

```
sudo named-checkconf /etc/bind/named.conf
sudo named-checkconf /etc/bind/named.conf.local
```

Il faut également vérifier les zones :

Vérification des zones

```
sudo named-checkzone fri-learning.ch /etc/bind/for.fri-learning.ch
sudo named-checkzone fri-learning.ch /etc/bind/rev.fri-learning.ch
```

Pour finir, il faut configurer l'interface réseau pour ajouter l'adresse du DNS :

Configuration de l'interface

```
sudo vi /etc/network/interfaces
```

3.2 Problèmes rencontrés

Une fois le DNS configuré, lors des tests depuis les utilisateurs sur le WIFI, le DNS ne faisait pas le travail demandé. Après vérification, la station utilisatrice avait le bon serveur DNS configuré. Il pouvait atteindre le serveur grâce à un ping.

La commande "*dig 172.19.243.2*" confirmait que la connexion avec le serveur était fonctionnelle.

En faisant un *"dig google.ch"*, un avertissement indiquait que la récursion n'était pas disponible sur notre serveur DNS.

```
juliens-macbook:~ julienborgognon$ dig google.ch

; <<>> DiG 9.10.6 <<>> google.ch
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: REFUSED, id: 4410
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.ch.                     IN      A

;; Query time: 54 msec
;; SERVER: 172.19.243.2#53(172.19.243.2)
;; WHEN: Thu Jun 14 20:07:56 CEST 2018
;; MSG SIZE  rcvd: 38
```

FIGURE 4: dig non-fonctionnel de google.ch

Nous avons donc autorisé la récursion sur notre serveur. Il a fallu modifier le fichier /etc/named.conf et ajouter :

Configuration de la récursion

```
Recursion  yes;
allow-recursion { any; };
```

Une fois cette modification sauvegardée et le serveur redémarré. Le serveur DNS était fonctionnel comme nous le prouve la capture suivante.

```
juliens-macbook:~ julienborgognon$ dig google.ch

; <<>> DiG 9.10.6 <<>> google.ch
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64561
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.ch.                     IN      A

;; ANSWER SECTION:
google.ch.              300     IN      A       216.58.214.67

;; AUTHORITY SECTION:
google.ch.              3546    IN      NS      ns1.google.com.
google.ch.              3546    IN      NS      ns3.google.com.
google.ch.              3546    IN      NS      ns2.google.com.
google.ch.              3546    IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.         172753  IN      A       216.239.32.10
ns1.google.com.         172753  IN      AAAA    2001:4860:4802:32::a
ns2.google.com.         172753  IN      A       216.239.34.10
ns2.google.com.         172753  IN      AAAA    2001:4860:4802:34::a
ns3.google.com.         172753  IN      A       216.239.36.10
ns3.google.com.         172753  IN      AAAA    2001:4860:4802:36::a
ns4.google.com.         172753  IN      A       216.239.38.10
ns4.google.com.         172753  IN      AAAA    2001:4860:4802:38::a

;; Query time: 3271 msec
;; SERVER: 172.19.243.2#53(172.19.243.2)
;; WHEN: Thu Jun 14 20:15:02 CEST 2018
;; MSG SIZE  rcvd: 312
```

FIGURE 5: dig fonctionnel de google.ch depuis une station en WIFI

3.3 Mise en place DNS64

Pour que les stations en IPv6 puissent communiquer avec des machiner ou des sites en IPv4, il faut mettre en place sur notre serveur DNS, la fonction DNS64. Pour se faire, nous allons modifier le fichier de configuration dans Bind9

fichier à modifier /etc/bind/named.conf.options

```
options {
  # directory "/var/cache/bind";
//******************************
  # listen-on-v6 { any; };
    # allow-query { any; };
  dns64 2001:cafe::/96 {
        clients { any; };
        mapped { any; };
        suffix ::;
        recursive-only yes;
};
//******************************
  };
};
```

il nous faut ensuite réaliser les commandes suivantes :

Redémarrer ensuite Bind9

```
sudo service bind9 restart
```

On peut à présent réaliser un Dig ipv6.google.com

Dig ipv6.google.com

```
; <<>> DiG 9.10.3-P4-Ubuntu <<>> ipv6.google.com A
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62262
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;ipv6.google.com.    IN  A

;; ANSWER SECTION:
ipv6.google.com. 603225  IN  CNAME ipv6.l.google.com.

;; AUTHORITY SECTION:
l.google.com.    60  IN  SOA ns1.google.com. dns-admin.google.com. 200659738 900 900 1800 60

;; Query time: 1637 msec
;; SERVER: 172.19.243.2#53(172.19.243.2)
;; WHEN: Fri Jun 15 08:33:26 DST 2018
;; MSG SIZE  rcvd: 115
```

4 Serveur WEB

Le serveur WEB de notre infrastructure est *hosté* sur un serveur Apache2, lui-même installé sur la version 16.04 LTS d'Ubuntu.

4.1 Installation d'Apache2

L'installation est très rapide, il suffit d'installer le service Apache2.

Commandes d'installation pour le service Apache2

```
  sudo apt update
    sudo apt install apache2
```

Une fois le service installé, il suffit de modifier le fichier HTML qui se situe à /var/www/html/index.html.

Code pour la page WEB

```
<!DOCTYPE html>

<html>
<head>
  <title>RIP G4</title>
  <style>
      body: {
      display: block;
      width: 100%;
      }
          h1: {
      display: inline-block;
      margin: 0 auto;
      text-align:center;
      position: relative;
      }
  </style>
</head>
<body>
  <h1>Groupe 4, It Works</h1>
</body
</html>
```

Le site WEB est accessible via son adresse IP qui est : 160.98.31.228. La page affiche ceci :

FIGURE 6: Page WEB

Il est également possible d'atteindre notre site depuis l'adresse fri-learning.ch comme l'atteste la capture ci-dessous.

FIGURE 7: Accès au site WEB via son nom

5 WLAN

L'Access Point est un ZyXEL NBG6515. Le SSID est "Groupe4". Le WIFI à 2.4GHz a été activé contrairement au 5GHz qui a été laissé de côté. Le Channel utilisé est le 1. L'encryption se fait grâce au WPA2-PSK. La clé de sécurité est : ciscocisco Seul la version 4 de l'IP est configurée.

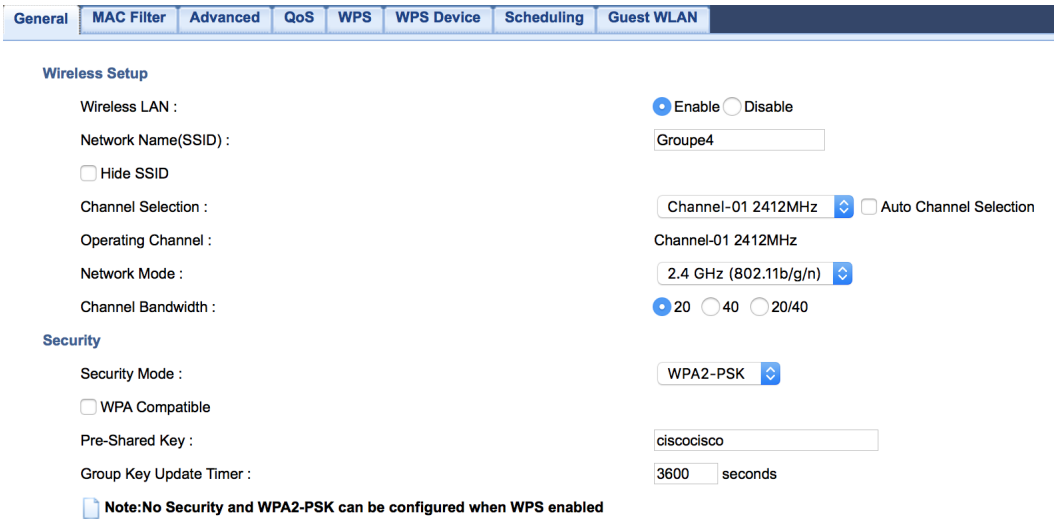

FIGURE 8: Configuration WIFI 2.4GHz

Un range Dynamic Host Configuration Protocol (DHCP) pour les stations se connectant au WIFI a été configuré. Les adresses distribuée sont 172.19.4.12 à 172.19.4.64.

FIGURE 9: Configuration DHCP

Pour la sortie WAN, nous avons défini une adresse IP statique : 172.19.1.10/24. Cette adresse IP fait partie du Virtual Local Area Network (VLAN) Open Space. L'adresse IP de notre serveur DNS y a également été configurée.

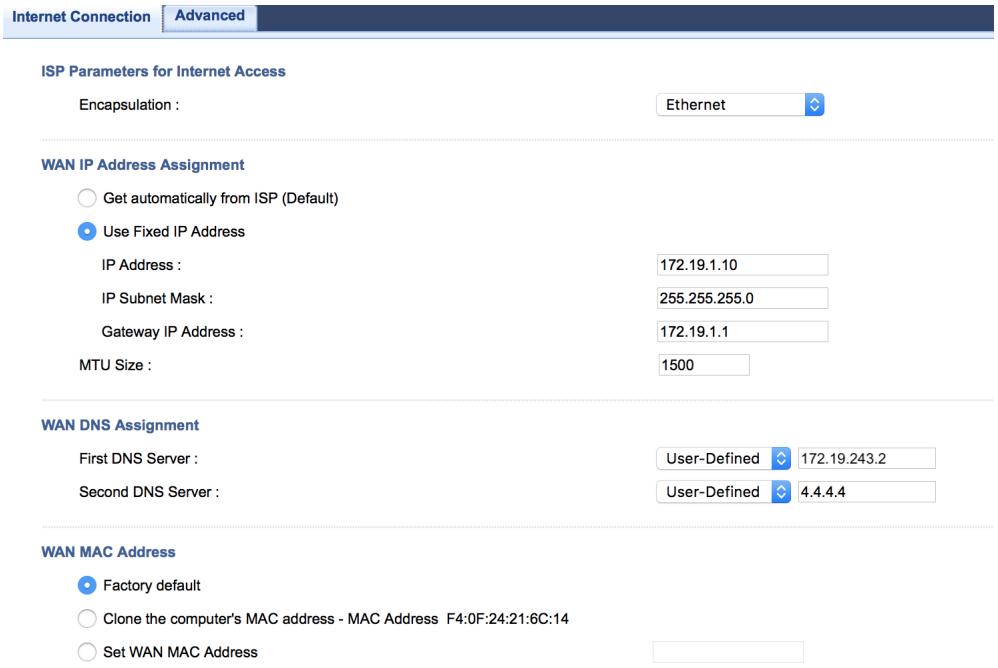

FIGURE 10: Configuration WAN

Pour finir, l'adresse IP de l'Access Point (AP) du côté du LAN WIFI est 172.19.4.10/24 Cette adresse sera la passerelle par défaut des stations connectées au WLAN.

FIGURE 11: Configuration IP LAN

6 Routage

6.1 OSPF

Pour la configuration de l'Open Shortest Path First (OSPF) du client, nous avons créé les zones comme indiquer dans la figure 12. À Berlin, les clients appartiennent à la zone 2 et les clients à Fribourg sont dans la zone 1.

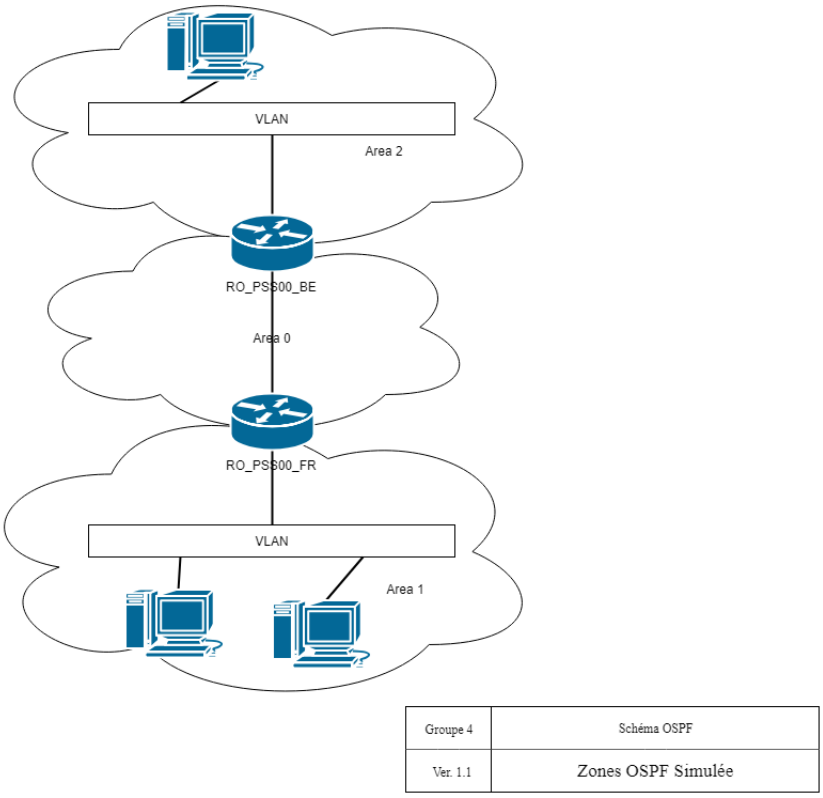

FIGURE 12: Schéma des zones OSPF simulée

Pour une meilleure compréhension de nos routeurs, nous avons décider de leur donner un router-id fixe est prédéfini par nos soins. Cela donne les IDs suivants :

Nom routeur	Router-ID ipv6	Router-ID ipv4
RO_PSS00_FR	1.1.1.1	1.1.1.2
RO_PSS00_BE	2.2.2.2	2.2.2.3
ISP_FR	3.3.3.3	3.3.3.4
ISP_BE	4.4.4.4	4.4.4.5
ASA	5.5.5.5	5.5.5.6

FIGURE 13: Base de données OSPF chez le client

La différence entre OSPFv2 et OSPFv3 se situe au niveau du router-ID qui diffère d'un chiffre.

La base de données IPv6 de nos routeurs chez le client est la suivante :

```
RO_PSSOO_FR#sh ipv6 ospf d

          OSPFv3 Router with ID (1.1.1.1) (Process ID 10)

              Router Link States (Area 0)

ADV Router      Age         Seq#       Fragment ID  Link count  Bits
1.1.1.1         547         0x80000006 0            1           B
2.2.2.2         547         0x8000000A 0            1           B

              Inter Area Prefix Link States (Area 0)

ADV Router      Age         Seq#       Prefix
1.1.1.1         1932        0x80000001 2001:620:40B:9940::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9949::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9948::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9947::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9946::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9944::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9943::/64
1.1.1.1         1932        0x80000001 2001:620:40B:9942::/64
2.2.2.2         1917        0x80000001 2001:620:40B:9945::/64
2.2.2.2         1917        0x80000001 2001:620:40B:9941:14::/96

              Link (Type-8) Link States (Area 0)

ADV Router      Age         Seq#       Link ID    Interface
1.1.1.1         1939        0x80000002 14         Tu0
2.2.2.2         1915        0x80000002 14         Tu0

              Intra Area Prefix Link States (Area 0)

ADV Router      Age         Seq#       Link ID    Ref-lstype  Ref-LSID
1.1.1.1         1939        0x80000001 0          0x2001      0
2.2.2.2         1915        0x80000001 0          0x2001      0

              Router Link States (Area 1)

ADV Router      Age         Seq#       Fragment ID  Link count  Bits
1.1.1.1         1907        0x80000003 0            0           B

              Inter Area Prefix Link States (Area 1)

ADV Router      Age         Seq#       Prefix
1.1.1.1         1935        0x80000001 2001:620:40B:9941:15::/96
1.1.1.1         535         0x80000005 2001:620:40B:9941:14::/96
1.1.1.1         535         0x80000005 2001:620:40B:9945::/64
 --More--

              Link (Type-8) Link States (Area 1)

ADV Router      Age         Seq#       Link ID    Interface
1.1.1.1         1946        0x80000002 23         Fa0/1.100
1.1.1.1         1946        0x80000002 22         Fa0/1.9
1.1.1.1         1946        0x80000002 21         Fa0/1.8
1.1.1.1         1946        0x80000002 20         Fa0/1.7
1.1.1.1         1946        0x80000002 19         Fa0/1.6
1.1.1.1         1946        0x80000002 18         Fa0/1.4
1.1.1.1         1946        0x80000002 17         Fa0/1.3
1.1.1.1         1946        0x80000002 16         Fa0/1.2

              Intra Area Prefix Link States (Area 1)

ADV Router      Age         Seq#       Link ID    Ref-lstype  Ref-LSID
1.1.1.1         1946        0x80000001 0          0x2001      0
```

FIGURE 14: Base de données OSPF chez le client

L'ISP redistribue ses routes statiques via l'OSPF, cette fonction est possible grâce à la méthode suivante :

Redistribution des routes statiques via OSPF

```
ipv6 router ospf 20
  redistribute static
router ospf 2
  redistribute static subnets
```

6.2 Statique

Pour que le routage fonctionne correctement, et que le chemin emprunté soit le bon, il faut indiquer des routes statiques à nos routeurs. Les routes statiques indiquent toujours le directement connecté. La route par défaut s'implémente de la manière suivante :

Route statique par défaut

```
RO_PSSOO_FR#ip route 0.0.0.0 0.0.0.0 172.19.2.2
```

Grâce à cette route on définit le chemin à emprunter lorsque la route n'est pas inscrite dans la table de routage.

6.3 Routage de nos routeurs

Pour voir les routes de nos routeurs, il y a deux commandes à disposition, ne pour les routes IPv4 et une pour les routes IPv6 :

Route IPv4

```
RO_PSSOO_FR#show ip route
```

Route IPv6

```
RO_PSSOO_FR#show ipv6 route
```

Sur nos routeurs chez le client, les routes ressemblent à cela :

```
RO_PSSOO_FR#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 172.19.2.2 to network 0.0.0.0

     172.19.0.0/16 is variably subnetted, 12 subnets, 4 masks
C       172.19.2.128/26 is directly connected, FastEthernet0/1.2
C       172.19.2.224/27 is directly connected, FastEthernet0/1.6
C       172.19.3.192/26 is directly connected, FastEthernet0/1.100
C       172.19.2.192/27 is directly connected, FastEthernet0/1.3
O IA    172.19.2.48/28 [110/11112] via 172.19.2.66, 00:01:00, Tunnel0
C       172.19.2.32/28 is directly connected, FastEthernet0/1
C       172.19.4.0/24 is directly connected, FastEthernet0/1.9
C       172.19.3.0/26 is directly connected, FastEthernet0/1.8
C       172.19.2.0/28 is directly connected, FastEthernet0/0
C       172.19.1.0/24 is directly connected, FastEthernet0/1.4
C       172.19.3.64/26 is directly connected, FastEthernet0/1.7
C       172.19.2.64/28 is directly connected, Tunnel0
S*   0.0.0.0/0 [1/0] via 172.19.2.2
```

FIGURE 15: Route IPv4 à Fribourg

```
RO_PSSOO_BE#sh ipv6 route
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   ::/0 [1/0]
     via 2001:620:40B:9941:12::2, FastEthernet0/0
C   2001:620:40B:9941:12::/96 [0/0]
     via ::, FastEthernet0/0
L   2001:620:40B:9941:12::1/128 [0/0]
     via ::, FastEthernet0/0
C   2001:620:40B:9941:14::/96 [0/0]
     via ::, FastEthernet0/1
L   2001:620:40B:9941:14::1/128 [0/0]
     via ::, FastEthernet0/1
C   2001:620:40B:9941:15::/96 [0/0]
     via ::, Tunnel0
L   2001:620:40B:9941:15::2/128 [0/0]
     via ::, Tunnel0
C   2001:620:40B:9945::/64 [0/0]
     via ::, FastEthernet0/1.5
L   2001:620:40B:9945::1/128 [0/0]
     via ::, FastEthernet0/1.5
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

FIGURE 16: Route IPv6 à Berlin

Sur nos routeurs chez l'ISP, les routes ressemblent à cela :

```
ISP_BE#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 172.19.242.1 to network 0.0.0.0

     172.19.0.0/16 is variably subnetted, 12 subnets, 4 masks
O E2    172.19.3.128/26 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.2.128/26 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
C       172.19.242.0/24 is directly connected, FastEthernet0/0
O E2    172.19.2.224/27 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.3.192/26 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.2.192/27 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
C       172.19.2.16/28 is directly connected, FastEthernet0/1.12
O E2    172.19.4.0/24 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.3.0/26 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O       172.19.2.0/28 [110/2] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.1.0/24 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
O E2    172.19.3.64/26 [110/20] via 172.19.242.1, 00:00:56, FastEthernet0/0
S*   0.0.0.0/0 [1/0] via 172.19.242.1
```

FIGURE 17: Route IPv4 chez l'ISP à Berlin

```
ISP_BE#sh ipv6 route
IPv6 Routing Table - 17 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   ::/0 [1/0]
     via 2001:620:40B:9941:242::1
O   2001:620:40B:9941:11::/96 [110/2]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
C   2001:620:40B:9941:12::/96 [0/0]
     via ::, FastEthernet0/1.12
L   2001:620:40B:9941:12::2/128 [0/0]
     via ::, FastEthernet0/1.12
C   2001:620:40B:9941:242::/96 [0/0]
     via ::, FastEthernet0/0
L   2001:620:40B:9941:242::3/128 [0/0]
     via ::, FastEthernet0/0
O   2001:620:40B:9941:243::/96 [110/11]
     via FE80::7E69:F6FF:FE5C:802A, FastEthernet0/0
OE2  2001:620:40B:9942::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
OE2  2001:620:40B:9943::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
OE2  2001:620:40B:9944::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
S   2001:620:40B:9945::/64 [1/0]
     via 2001:620:40B:9941:12::1
OE2  2001:620:40B:9946::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
OE2  2001:620:40B:9947::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
OE2  2001:620:40B:9948::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
OE2  2001:620:40B:9949::/64 [110/20]
     via FE80::219:30FF:FEE3:275, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

FIGURE 18: Route IPv6 chez l'ISP à Berlin

Sur la figure 18, on peut voir les adresses statiques redistribuées par l'OSPF. Celles-ci sont mentionnées par *O E2*

Pour vérifier la bonne configuration de l'OSPF, on peut vérifier les voisins de nos routeurs.

```
ISP_BE#sh ipv6 ospf ne

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
3.3.3.3           1   FULL/DROTHER    00:00:39    5               FastEthernet0/0
5.5.5.5           1   FULL/DR         00:00:36    14              FastEthernet0/0
ISP_BE#
```

FIGURE 19: Tables des voisins OSPF

Sur la figure 19 on peut voir que le routeur de l'ISP à Berlin a dans ces voisins les deux autres routeurs de l'area 0, soit le routeur avec l'ID 3.3.3.3 et celui avec l'ID 5.5.5.5 qui sont indiqués à la figure 13.

7 GRE

Le tunnel Generic Routing Encapsulation (GRE) permet de passer du site de Fribourg au site de Berlin via une interface "fictive", pour tester l'interface, il suffit d'effectuer un tracer depuis Fribourg sur l'interface de sortie sur le routeur de Berlin. Sur la figure 20, on peut voir que le chemin passe bien par le tunnel GRE.

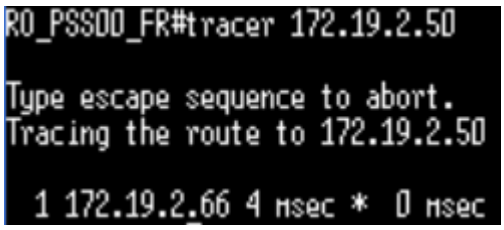

FIGURE 20: Tracer depuis Fribourg vers Berlin

Après plusieurs captures WireShark, nous n'avons pas réussi à voir le tunnel GRE sur celles-ci. Mais nous pouvons voir que les clients à Fribourg peuvent joindre les clients à Berlin, comme le montre la figure 21.

FIGURE 21: Capture WireShark en ipv6

8 ASA

L'équipement de Cisco, l'ASA 5505 est un *firewall* qui dans le cas de notre réseau permet de gérer les flux entrant et sortant du réseau. Afin de protèger les équipements internes au réseau de l'ISP. contre les accès malveillant de terminaux se trouvant de l'autre coté du *firewall*. Le réseau autour du *firewall* peut être simplifié comme sur la figure **??**

Le schéma présente trois réseaux distincts :

Inside : Représente l'entierté du réseau cacher derrière le firewall comprenant le réseau backbone de l'ISP ainsi que les sous-réseau des sièges de Fri-Learning. Il n'est pas accessible depuis le réseau **Outside** et le réseau **Inside**

Outside : Réseaux situés entres le *firewall* et n'importe quels autres ressoures sur internet. Ce 'réseau' est accesible par le réseau **Inside** et le réseau de la **DMZ**

DMZ : Ce réseau comporte les machines qui hébergent les services (DNS et HTTP) qui sont fournis à la fois au réseau **Inside** et au réseau **Outside**, les services sont accessible depuis le réseau **Outside** via une adresse IP publique statique ainsi que d'un nom de domaine dans le cas du serveur HTTP.

8.1 Création des interfaces

Avant de pouvoir créer des règles NAT/PAT ou encore des ACL, il faut définir les interfaces de sortie de l'ASA. Dans la situation illustrée à la **??** une interface correspond à un réseau. Les interfaces ont été configuré comme telle.

Code de configuration des interfaces physiques et logiques

```
interface Ethernet0/0
!
interface Ethernet0/1
 switchport access vlan 2
!
interface Ethernet0/2
 switchport access vlan 3
!
interface Vlan1
 description Reseau local interne
 nameif inside_INT
 security-level 100
 ip address 172.19.242.2 255.255.255.0
 ipv6 address 2001:620:40b:9941:242::2/96
 ipv6 enable
 ipv6 ospf 20 area 0
!
interface Vlan2
 description Reseau externe (HEIA)
 nameif outside_INT
 security-level 0
 ip address 160.98.31.244 255.255.254.0
 ipv6 address 2001:620:40b:1030::4:4444/64
 ipv6 enable
 ipv6 ospf 1 area 30
!
interface Vlan3
 description Zone demilitarisee
 nameif DMZ
 security-level 50
 ip address 172.19.243.1 255.255.255.0
 ipv6 address 2001:620:40b:9941:243::1/96
 ipv6 enable
 ipv6 ospf 20 area 0
!
```

Le code ci-dessus permet de créer le schéma de réseau suivant :

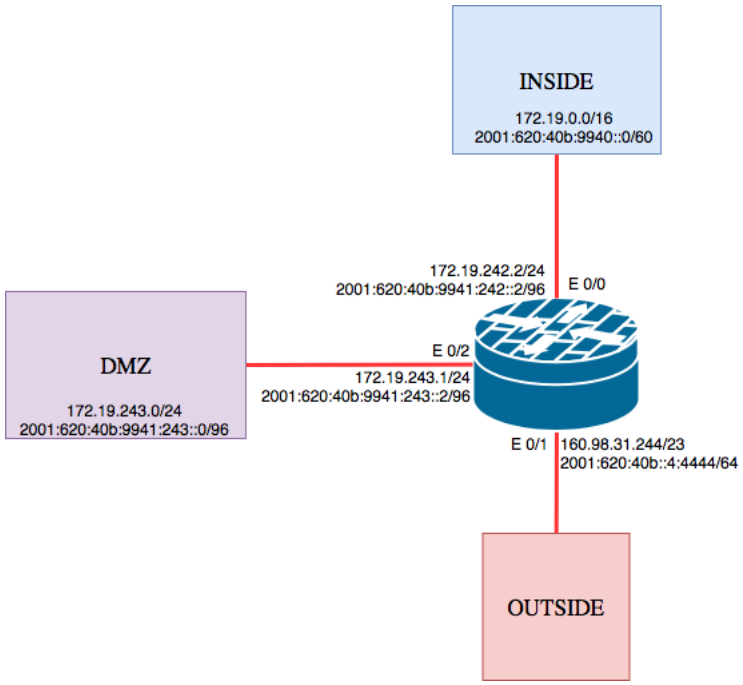

FIGURE 22: Schéma simplifié outside-inside-DMZ

8.2 NAT/PAT

8.2.1 Implémentation de la sortie du réseau en IPv4/IPv6

Afin de permettre aux sous-réseaux d'accéder à internet en IPv4 et IPv6, il est nécessaire de pourvoir l'interface du firewall coté **Outside** d'au moins une adresse publique de chaque, de plus pour s'assurer que tous les terminaux du réseau **Inside** puisse accéder à internet malgré que l'interface de sortie ne dispose que d'une adresse, un service de type NAT/PAT est indispensable, dont l'implémentation suit.

Configuration PAT pour l'accès à Internet

```
object network intern_v4
 subnet 172.19.0.0 255.255.0.0
 nat (inside_INT,outside_INT) dynamic interface
object network intern_v6
 subnet 2001:620:40b:9940::/60
 nat (inside_INT,outside_INT) dynamic interface
```

8.2.2 Implémentation de la DMZ

La DMZ est à mi-chemin entre l'intérieur et l'exérieur du réseau de l'ISP, ce réseau à donc besoin de permissions et de règles particulières. Comme les services (et par conséquence les machines qui les héberges) sont uniquement accessible via une adresse IP publique statique, il est nécessaire de référencer ces dernières dans la configuration de l'ASA, comme suit :

Configuration des adresses IP locals statiques

```
object network DMZ
 subnet 172.19.243.0 255.255.255.0
object network HTTP_global
 host 172.19.243.3
object network DNS_global
 host 172.19.243.2
```

Puis pour permettre au réseau **Outside** d'accéder aux addresses statiques, les règles NAT sont les suivantes :

Règle NAT pour les adresses IP publiques statiques

```
object network DMZ
 nat (DMZ,outside_INT) dynamic interface
object network HTTP_global
 nat (DMZ,outside_INT) static 160.98.31.228
object network DNS_global
 nat (DMZ,outside_INT) static 160.98.31.227
```

8.3 Implémentation du NAT64

Le principe du DNS64 en couplage avec le NAT64 est de permettre à des stations disposant uniquement d'une adresse IPv6 de pouvoir naviguer sur des services web ne disposant pas d'adresse IPv6 publique.

Le rôle de l'ASA est de translater les adresses fournient par le DNS64 en l'adresse IPv4

réelles du services distant. Ce service s'active simplement grâce au configuration suivante :

Configuration du NAT64

```
object network NAT64_prefix
 subnet 2001:cafe::/96
object network OUT2NAT64
 subnet 0.0.0.0 0.0.0.0
 nat (outside_INT,inside_INT) static NAT64_prefix
```

8.4 Access List

Lors de la définition des interfaces virtuelles le champ "securitiy-level" permet de définir un niveau de sécurité, le niveau 0 est le niveau le plus bas qui est assigné par défaut au réseau **Outside**, à l'inverse le niveau 100 est le plus haut, dans la plus part des cas il est donné au réseau **Inside**. Une interface virtuelle peut accéder à une autre si son niveau de sécurité est suppérieur à cette dernière. Dans le cas contraire si la première interface possède un niveau inférieur, l'accès au sous-réseau est dénié et un message ICMP est envoyé à la source.

Pour outre-passer ce genre de restriciton, les ACL sont la solution. Nous allons donc détaillés les cas qui nous intéresse et voir en détail leurs implémentations.

8.4.1 Accès à internet

Pour l'accès à internet, une ACL est enclenché pour faire transiter les paquets requetés par les stations du réseau **Inside**.

Configuration de l'ACL pour la navigation sur Internet

```
access-list IN2OUT extended permit ip any any
```

8.4.2 DMZ

Dans le cas de la DMZ, bien que l'ASA puisse translater l'adresse publique en adresse local, ceci n'autorise en rien le réseau **Outside** à accéder à la **DMZ**. C'est pour cette raison que plusieurs ACL sont nécessaires.

ACL pour permettre au station externe d'accéder à la DMZ

```
access-list OUT2DMZ extended permit ip any host 172.19.243.2
access-list OUT2DMZ extended permit ip any host 172.19.243.3
```

8.5 Access Group

Pour appliquer les ACL à une interface, il obligatoire de spécifier cette configuration dans un "access-group"

Configuration des Access Group par interface logique

```
access-group OUT2DMZ in interface outside_INT
access-group IN2OUT global
```

9 Installation d'OpenStack

Durant ce travail, nous avons du mettre en place une architecture de cloud OpenStack suivant l'architecture à la figure

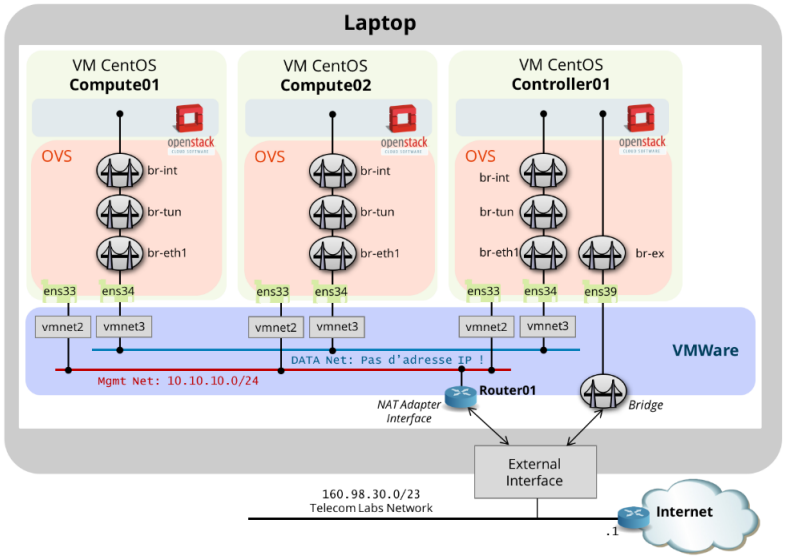

FIGURE 23: Architecture à mettre en place

L'architecture suivante a été mise en place de manière physique comme suit :

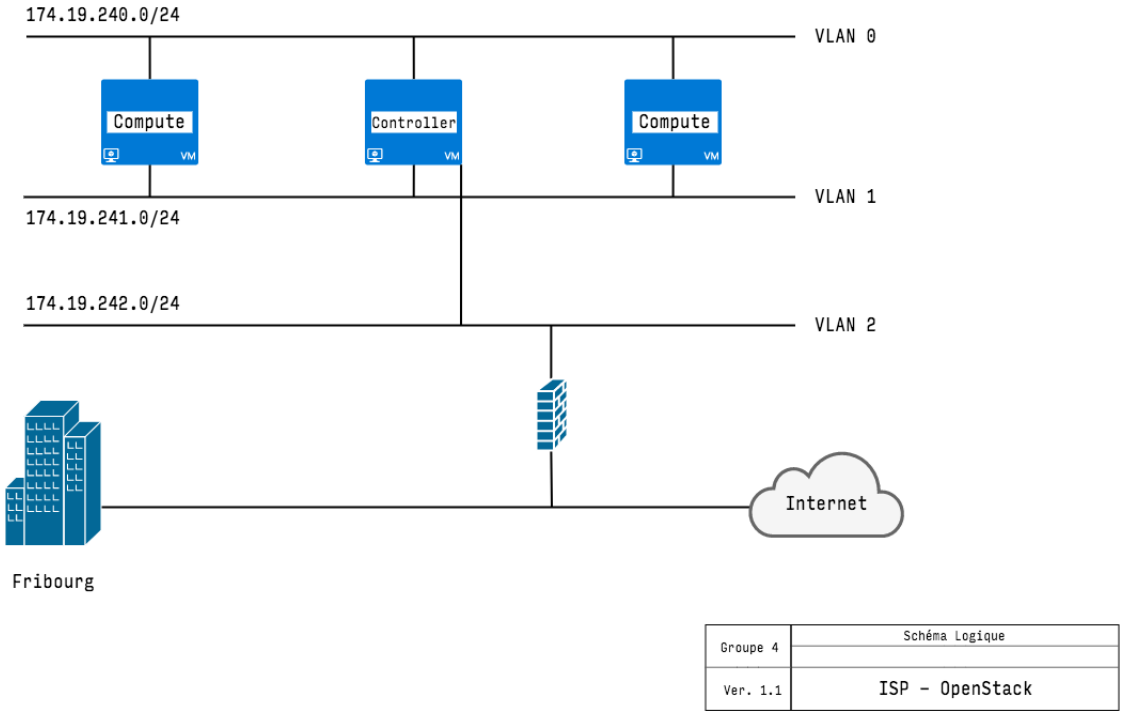

FIGURE 24: Schéma logique d'OpenStack

L'installation d'Openstack s'est passée sans problèmes sur nos trois machines : notre controller et nos deux compute.

9.1 Déploiement d'OpenStack sur nos Compute Nodes

Il faut tout d'abord que toutes nos machines aient une interface sur le même sous réseau.

Adresse de nos différentes machines de notre infrastructure

```
10.10.10.66 compute01.tic.heia-fr.ch compute01
10.10.10.67 compute02.tic.heia-fr.ch compute02
10.10.10.68 controller01.tic.heia-fr.ch controller01
```

Pour le déploiement d'OpenStack sur nos compute Nodes, il faut établir un transfert des clé RSA entre les différents éléments de notre architecture.

On obtient les clés RSA via la fonction ci-dessous :

Transfert de clé RSA du Compute01 vers les compute 01-02

```
[root@localhost ~]# ssh-keygen -t rsa
```

Transfert de clé RSA du Controller01 vers les Compute 01-02

```
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.66
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.67
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.68
```

Transfert de clé RSA du Compute01 vers le Controller01 et le Compute 02

```
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.67
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.68
```

Transfert de clé RSA du Compute02 vers le Controller01 et le compute01

```
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.66
[root@localhost ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.10.10.68
```

On vérifie ensuite que la connexion SSH fonctionne entre les différents composants.

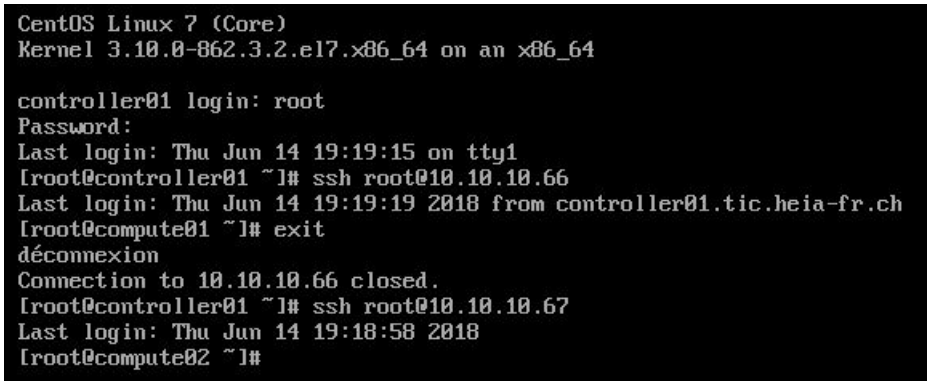

FIGURE 25: Connexion du Controller01 vers Compute01

On voit que la connexion peut s'établir entre le Controller01 et les différents Compute.

Pour Déployer Openstack, il existe des moyens facilités tels que PackStack. On installe donc PackStack sur notre controller. Pour déployer PackStack sur la version Queens, on ne doit pas suivre la notice fournie, on doit se rendre sur le site rdoproject.org

Déploiement via PackStack

```
$ yum update -y
$ yum install -y centos-release-openstack-queens
$ yum update -y
$ yum install -y openstack-packstack
$ packstack --allinone
$ yum install -y centos-release-openstack-queens
$ yum-config-manager --enable openstack-queens

$ sudo yum install -y openstack-packstack
$ sudo packstack --allinone
```

installation de PackStack

D'une fois installé PackStack sur la dernière release d'OpenStack, on modifie le fichier
`[root@controller01 ~]# vi /root/answers.txt`

installation de PackStack

`[root@controller01 ~]# packstack --answer-file=/root/answers.txt --timeout=600`

9.2 Accès au DashBoard

Via l'adresse 10.10.10.68/dashboard, On accède au DashBoard OpenStack ci-dessous, on créera nos différentes instances depuis cette interface.

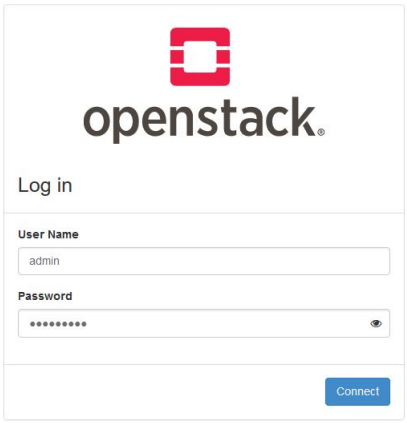

FIGURE 26: DashBoard openStack

En suivant la documentation fournie, on crée un réseau, un routeur virtuel dont voici la topologie représentée dans le dashBoard.

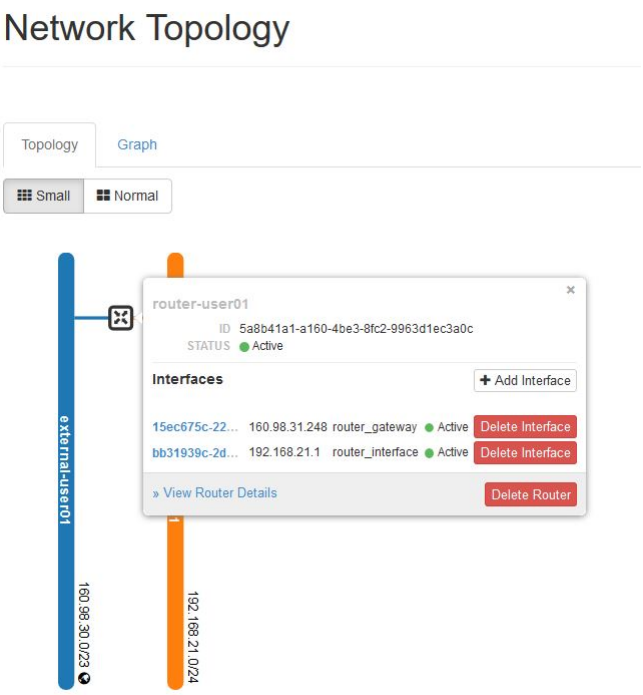

FIGURE 27: Topologie Instance Openstack

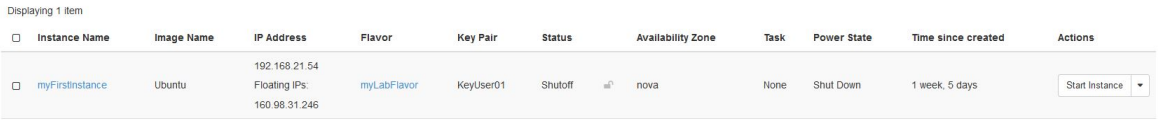

FIGURE 28: Instance crée pour ce projet

9.3 Problèmes rencontrés

Malgré la mise en place rigoureuse dont nous avons fait preuve dans la mise en place d'OpenStack, nous avons été dans l'impossibilité de nous connecter sur nos instances. En effet, depuis le dashboard, nous ne pouvons pas nous connecter sur nos différentes instances crées.

Voici les spécifications de notre instance

FIGURE 29: Instance crée pour ce projet

FIGURE 30: clé RSA keyUser1.pem

La connexion SSH via nos clés données en pairs dans le dashBoard reste tout de même infructueuse.

Nous avons par conséquent pas pu faire tourner le serveur HTTP sur une machine virtuelle dans l'OpenStack. Il en va de même pour le serveur DNS.

Le serveur DNS, et le serveur HTTP seront installé sur une distribution Linux, version 16.04 LTS Ubuntu.

10 Conclusion

Nous avons pu par le biais de ce projet intégré développer concrètement nos facultés en terme de design de réseau, de mise en place des équipements.

Ce travail a été très motivant pour nous. Bien qu'on n'aie pas pu mettre en place le Cloud OpenStack pour faire tourner nos VM qui auraient pu héberger notre serveur DNS et notre serveur HTTP. C'est pourquoi nous avons du installer le serveur HTTP et DNS sur des machines physiques et non virtuelles.

Le serveur HTTP et DNS sur les machines physique n'a posé aucun problème particulier, l'installation s'est déroulée sans encombres.

Pour la partie réseau, nous avons pu élargir nos connaissances via différentes encombres rencontrées au long de ce projet, notamment au niveau du routage entre l'ISP et le client.

Lorsque nous avons commencé avec l'OSPF, nous n'avions pas encore vu la théorie en cours. On a donc du se baser sur les analyses préliminaires réalisée plus tôt dans le projet. L'OSPF n'a pas posé de problèmes, à chaque fois on s'est occupé de résoudre nos problèmes, soit via nos propres moyens (internet, netacad.com) soit par notre Superviseur, M. Buntschu.

Au niveau de l'adressage, on a pu clairement mettre en pratique les notions abordées en cours, notamment au niveau IPv4 Ipv6, et au niveau des VLAN's.

Ce Projet est une bonne première expérience en terme de conception réseau et de respect du cahier des charges.

11 Références

Table des figures

12 Glossaire

AP Access Point.

DHCP Dynamic Host Configuration Protocol.
DNS Domaine Name Server.

GRE Generic Routing Encapsulation.

ISP Internet Service Providers.

OSPF Open Shortest Path First.

VLAN Virtual Local Area Network.

WLAN Wireless Local Area Network.