Florent CAGNARD,
L3 Informatique,
Avignon Université

Rapport AMS Réseau : Développement d'un FAI (Fournisseur d'Accès à *Internet*)

Sommaire des étapes du développement

Contexte

En troisième année de licence informatique à l'université d'*Avignon*, j'ai choisi l'AMS (Activité de Mise en Situation) Réseau, dont l'objectif est de **développer un fournisseur d'accès à *Internet***.

Le but du projet est de permettre à n'importe quel utilisateur de **configurer sa box *Internet*** selon ses besoins, sans avoir nécessairement de connaissances en informatique.

Une interface web permet de **paramétrer les services proposés** de manière **intuitive** (son utilisation ne demande pas de connaissances techniques).

Ce travail m'a permis de mettre en pratique toutes les connaissances acquises en **réseaux informatiques**, en **scripting** (***Bash***), et en développement web (***HTML, CSS, JavaScript, PHP***).

Formulaire IP

La première étape fut de créer un script permettant de **modifier l'adresse IP de la box *Internet***, avec **la génération automatique d'un backup**.

J'ai trouvé deux moyens qui permettent modifier l'adresse IP :

- La première solution que j'avais trouvé était d'utiliser la commande ***ifconfig*** qui permet en lui donnant des arguments, de modifier l'adresse IP (et le masque de sous-réseau si besoin) d'une interface. La commande qui fonctionnait dans mon script était ***sudo ifconfig eth0 $1***, où $1 est l'argument fourni lors de l'exécution du script (c'est à dire, l'adresse IP choisie par l'utilisateur). Cependant, j'ai abandonné cette solution qui me semblait trop compliqué au moment de générer un backup, car je me suis aperçu que lorsque nous entrons la commande ***ifconfig***, il s'agit là d'une restitution de 3 fichiers (voir photo ci-dessous qui est extraite du ***man ifconfig***).

- J'ai donc opté pour la deuxième solution qui est d'utiliser le fichier ***/etc/network/interfaces*** que j'avais déjà utilisé dans le TP1 du cours *Internet et les Services Réseaux*, ce fichier résume comment sont configurées les interfaces de la machine. Il est possible de donner une adresse IP de manière **statique** à une interface, et non de manière **automatique via DHCP**, c'est ce que j'ai mis en place sur la capture d'écran ci-dessous pour l'interface ***eth1***.

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# eth0 : NAT
allow-hotplug eth0
iface eth0 inet dhcp

# eth1 : réseau interne
allow-hotplug eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
gateway 192.168.1.1
```

Étapes d'exécution de mon script IP

Voici les étapes d'exécution de mon script ***ip.sh*** que j'ai implémenté en *Bash*, qui **modifie l'adresse IP de la box *Internet*** :

- Tout d'abord, il vérifie que le script a bien été **exécuté avec un seul argument** : l'adresse IP choisie par l'utilisateur. En faisant ça, j'ajoute une **couche de sécurité** à mon script pour ne pas qu'il soit utilisé de la mauvaise manière.
- Ensuite, il récupère l'adresse IP de la **configuration actuelle**, présente dans le fichier ***/etc/network/interfaces***.
- Après ça, il s'assure que l'adresse IP choisie par l'utilisateur n'est pas déjà celle de la configuration actuelle, pour **éviter de faire des changements qui n'auront aucun impact**.
- Puis, il remplace l'adresse IP **par la nouvelle**, grâce à la commande **sed**.
- Enfin, il **crée un backup** en veillant tout d'abord à ce que le dossier contenant toutes les sauvegardes ait bien été crée au préalable, s'il n'existe pas déjà (il peut s'agir de la première utilisation du script), il le crée. Une fois ceci vérifié, il **copie le fichier *****/etc/network/interfaces*** dans ce dossier en le **nommant de manière horodatée** avec la **date d'exécution du script**.

Redémarrage de l'interface

Une fois le fichier ***/etc/network/interfaces*** modifié, il faut redémarrer l'interface au sein du script (sans redémarrer la box *Internet*), pour appliquer les changements effectués. Néanmoins, les commandes ***sudo ifdown eth1*** puis ***sudo ifup eth1*** ne suffisent pas. D'ailleurs, avec cette version de

mon script (les deux commandes données), l'adresse n'était pas remplacée mais ajoutée ce qui fait que je n'avais pas 1 mais 2 adresses IP possibles (ce qui n'est pas tolérable):

- l'adresse IPv4 de base, l'actuelle, celle à remplacer,

```
stud@ubuntu:~/FAI/IP$ ip addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:81:19:a5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth1
       valid_lft forever preferred_lft forever
    inet 192.168.1.2/24 brd 192.168.1.255 scope global secondary eth1
       valid_lft forever preferred_lft forever
```

```
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fd00::a00:27ff:fef5:74f3/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fef5:74f3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1329 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1044 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:483126 (483.1 KB)  TX bytes:161650 (161.6 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:81:19:a5
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe81:19a5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:648 (648.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:328 (328.0 B)  TX bytes:328 (328.0 B)
```

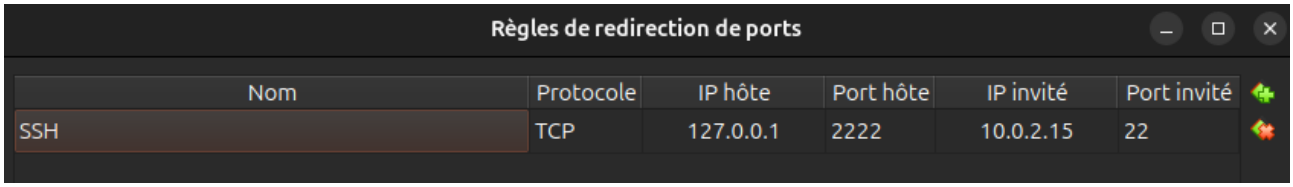
- l'adresse IPv4 qui devait remplacer l'ancienne.

Pour corriger ce problème, j'ai trouvé la commande ***sudo ip addr flush dev eth1*** qui permet de supprimer toutes les adresses d'une interface et complète donc les deux commandes précédentes.

```
stud@ubuntu:~/FAI/IP$ ./ip.sh 192.168.1.30
ifdown: interface eth1 not configured
RTNETLINK answers: File exists
Failed to bring up eth1.
stud@ubuntu:~/FAI/IP$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:f5:74:f3
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fd00::a00:27ff:fef5:74f3/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fef5:74f3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2054 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1614 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:538283 (538.2 KB)  TX bytes:223893 (223.8 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:81:19:a5
          inet addr:192.168.1.30  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:828 (828.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:328 (328.0 B)  TX bytes:328 (328.0 B)

stud@ubuntu:~/FAI/IP$ ip addr show eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:81:19:a5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.30/24 brd 192.168.1.255 scope global eth1
       valid_lft forever preferred_lft forever
stud@ubuntu:~/FAI/IP$
```

Installation de *SSH* et redirection de ports

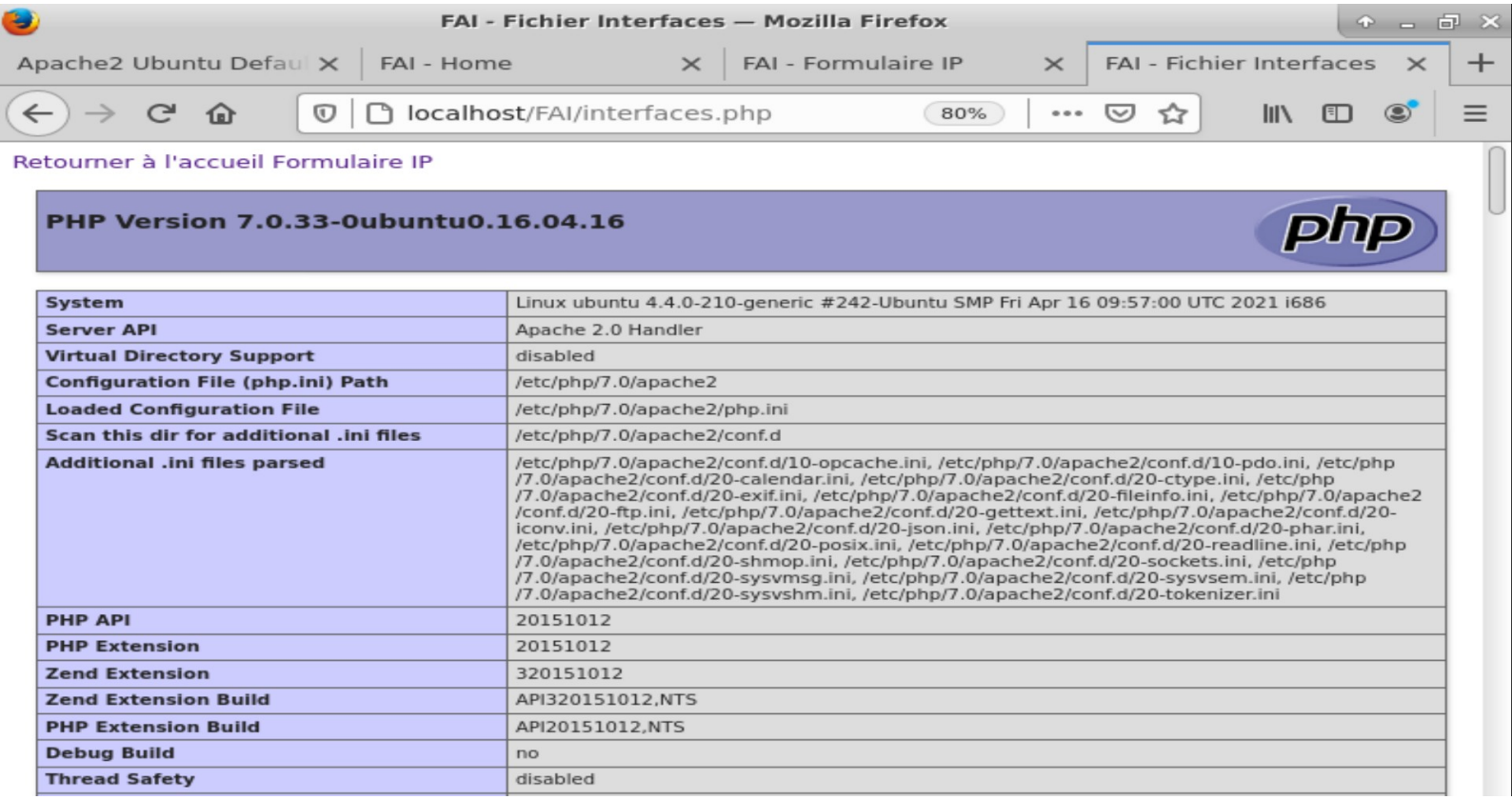
Après avoir installé le package ***openssh-server***, j'ai ajouté une **redirection de ports** afin de pouvoir me connecter en SSH à la box *Internet* depuis un terminal ouvert sur ma machine personnelle. Cette implémentation m'a également permis d'effectuer des copies de fichiers entre les machines, notamment pour alimenter mon repository *Github*, mais aussi pour apporter des modifications aux scripts implémentés.



Pour faire simple, cette redirection de port (port forwarding) dit que si quelqu'un se connecte à l'adresse 127.0.0.1 (adresse loopback = moi), sur le port 2222 (port quelconque), on envoie ce trafic vers le port 22 (SSH) de la machine virtuelle.

Installation de PHP

Apache était déjà installé avec l'image utilisée pour créer la machine virtuelle. En revanche, *PHP* n'était pas reconnu ce qui fait que le code de mes balises *PHP* ne s'exécutaient pas. Je m'en suis aperçu au moment d'afficher le fichier ***/etc/network/interfaces*** à partir de la fonction ***file_get_contents()*** qui n'affichait rien. C'est pourquoi j'ai donc entré la commande ***sudo apt install libapache2-mod-php*** que j'ai trouvé sur la documentation du langage PHP.



Pour vérifier que *PHP* était désormais bien reconnu, j'ai utilisé la commande ***phpinfo()***.

Visudo

Le script que j'ai crée, permettant de modifier l'adresse IP de la box *Internet*, doit pouvoir être exécuté depuis le serveur web, ce qui veut dire que le retour de la commande ***whoami*** ne sera pas le même selon si le script est exécuté dans le terminal ou sur l'interface web. Ce qui fait que j'ai dû faire en sorte que le script soit exécuté, sur l'interface web, sans que l'on lui demande de mot de passe (dû aux commandes ***sudo***). Pour cela, j'ai été dans le fichier des ***sudoers*** grâce à la commande ***sudo visudo*** qui m'a permis d'ajouter une ligne pour que ***www-data*** (le serveur web), puisse exécuter ce script et uniquement ce script, sans mot de passe. Pour vérifier le changement effectué, je me suis connecté en tant que ***www-data*** et j'ai testé la commande qui sera exécuté par le serveur web, en PHP, qui est ***sudo /home/stud/FAI/ip.sh*** *192.168.1.33*. On observe bien que l'on ne me demande pas de mot de passe quand je souhaite exécuter ce script en tant que ***www-data***.

```
 GNU nano 2.5.3                                                File: /etc/sudoers.tmp

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Permet à www-data (serveur web) d'exécuter ce script et seulement ce script sans demander de mot de passe.
www-data ALL=(ALL) NOPASSWD: /home/stud/FAI/IP/ip.sh

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

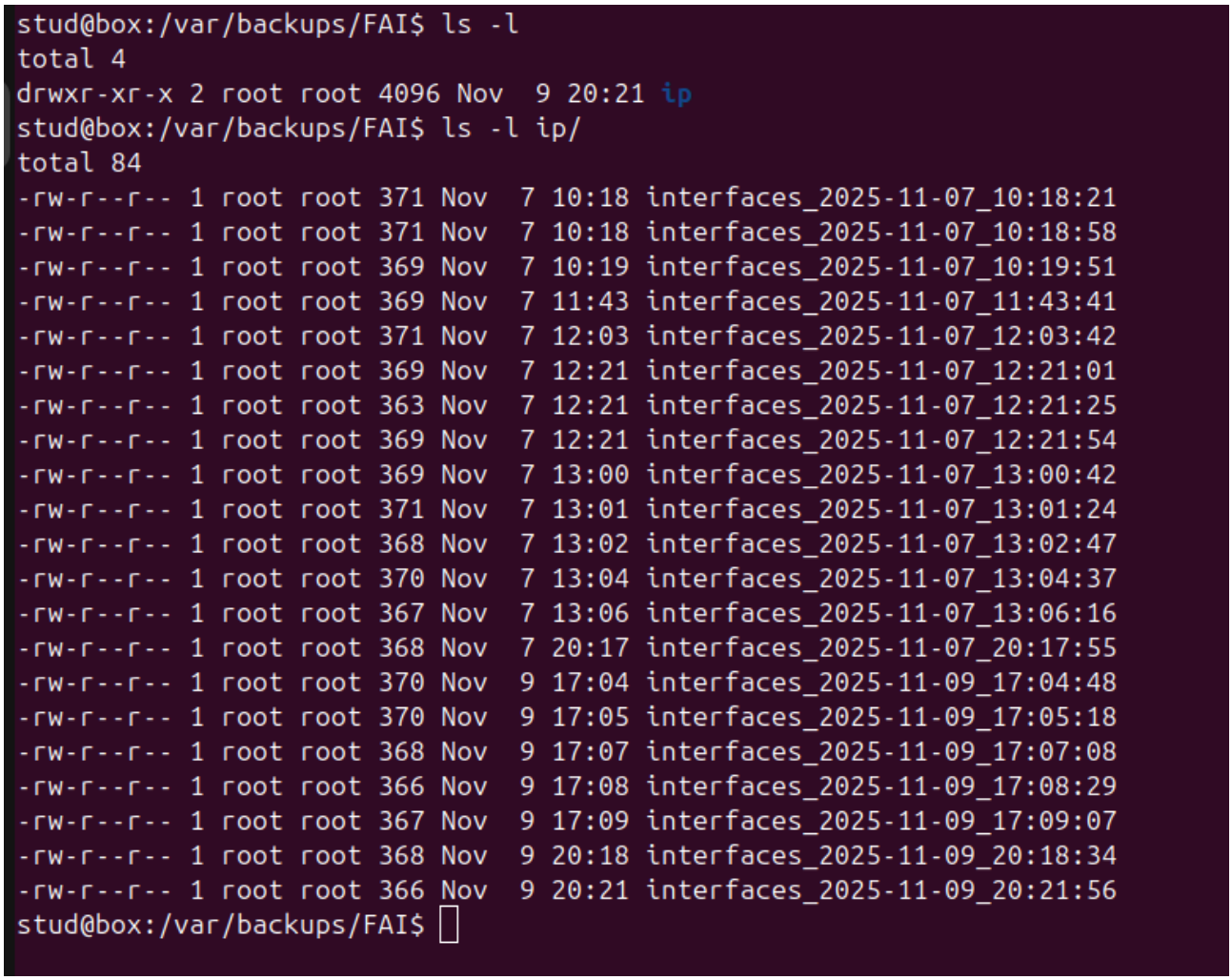
```
stud@ubuntu:~/FAI/IP$ sudo -u www-data -s
www-data@ubuntu:~/FAI/IP$ sudo /home/stud/FAI/IP/ip.sh
www-data@ubuntu:~/FAI/IP$ echo $?
1
www-data@ubuntu:~/FAI/IP$ sudo /home/stud/FAI/IP/ip.sh 192.168.1.33
RTNETLINK answers: Cannot assign requested address
www-data@ubuntu:~/FAI/IP$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:f5:74:f3
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fd00::a00:27ff:fef5:74f3/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fef5:74f3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24831 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15828 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2298606 (2.2 MB)  TX bytes:1904796 (1.9 MB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:81:19:a5
          inet addr:192.168.1.33  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe81:19a5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2527 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:153152 (153.1 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:4545 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4545 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:526887 (526.8 KB)  TX bytes:526887 (526.8 KB)

www-data@ubuntu:~/FAI/IP$
```

Ajout de l’adresse IP comme argument du script

J’ai utilisé la fonction ***escapeshellarg($arg)*** qui met automatiquement des «*quotes*» autour de l’argument *$arg* et échappe les caractères spéciaux pour qu’**ils ne soient pas interprétés par le shell**. Ceci rajoute une couche de sécurité à mon projet et permet d’ajouter l’adresse IP choisie par l’utilisateur comme argument à l’exécution du script.

Mise en place des backups dans le dossier /var/backups

Le dossier **/var** sous *Linux* contient des fichiers de données variables. Cela inclut les données administratives et de **journalisation**, ainsi que les fichiers transitoires et temporaires. Ainsi, au lieu de créer un dossier backup dans mon dossier personnel contenant toutes les backups des différents scripts, j'ai décidé de mettre ce dossier dans **/var/backups**.



Formulaire DHCP

La deuxième étape de ce projet a consisté à créer, sur l'interface web, un formulaire permettant de modifier la plage d'adresses attribuées par le serveur DHCP.

Choix du masque de sous-réseau

Au moment de créer mon formulaire IP, je n'avais pas pensé à **gérer la cohérence avec le masque de sous-réseau**. Et je me suis rendu compte de ça au moment de d**évelopper le script DHCP**, qui a besoin de **définir** au moment de déclarer, l'**adresse réseau**, qui **est définie à partir du masque de sous-réseau**. Donc, pour pouvoir faire ceci, **j'ai donc ajouté au formulaire IP, un champ supplémentaire au formulaire, pour que l'utilisateur puisse aussi modifier le masque de sous-réseau**. Cependant, cela nécessitait aussi une concordance avec l'adresse IP, en effet, si on modifie le masque de /24 à /16, l'adresse IP n'a plus 1 mais 2 octets de disponibles, en revanche, l'inverse est pareil, si on passe d'un masque /16 à /24, il ne faut plus permettre au client de modifier le troisième octet de l'adresse IP. Pour cela, j'ai donc utilisé ***JavaScript***, qui **récupère les valeurs des deux inputs et met à jour en temps réel, la disponibilité des octets de l'adresse IP en fonction de la valeur du masque de sous-réseau**. Cette implémentation **ajoute une couche de sécurité** et a été mise en place de la même manière que la modification de l'adresse IP de la box *Internet*, en ajoutant ici un **netmask** quelconque.

```
stud@box:~$ cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# eth0 : NAT
allow-hotplug eth0
iface eth0 inet dhcp

# eth1 : réseau interne
allow-hotplug eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.0.0
stud@box:~$
```

Calcul de l'adresse réseau

Pour rappel, **le masque de sous-réseau détermine quelle partie de l'adresse IP est fixe (définie au réseau) et quelle partie est disponible (adresse possible pour un hôte)**. Cela m'a rappelé mes cours de *Fondement des Réseaux*, en L2 Informatique, où on a apprit à **calculer l'adresse réseau à partir de l'adresse IP et de l'adresse du masque de sous-réseau**, après **les avoir convertis en binaire**, **on met à 1 quand en comparant les bits des deux adresses, ils sont à 1 tous les deux**. J'ai pensé que c'était plus simple de définir l'adresse réseau en *PHP* que directement dans le script *Bash* avec l'utilisation des commandes ***array_map()*** et ***sprintf()***. Une fois celle-ci calculée, elle est fournie au script DHCP, ses étapes d'exécution seront présentées par la suite.

```php
// Récupère l'adresse IP
$get_ip_command = 'cat /etc/network/interfaces | grep "address" | cut -d" " -f2';
$current_ip_address = trim(shell_exec($get_ip_command));
// Sépare l'IP en 4 octets
$ip_address_octets = array_map('intval', explode('.', $current_ip_address));

// Récupère le masque de sous-réseau
$get_subnet_mask_command = 'cat /etc/network/interfaces | grep "netmask" | cut -d" " -f2';
$current_subnet_mask = trim(shell_exec($get_subnet_mask_command));
// Sépare le masque de sous-réseau en 4 octets
$subnet_mask_octets = array_map('intval', explode('.', $current_subnet_mask));

// Calcule l'adresse réseau
$network_address = sprintf(
    "%d.%d.%d.%d",
    ($ip_address_octets[0] & $subnet_mask_octets[0]),
    ($ip_address_octets[1] & $subnet_mask_octets[1]),
    ($ip_address_octets[2] & $subnet_mask_octets[2]),
    ($ip_address_octets[3] & $subnet_mask_octets[3])
);
```

Définition de la plage d’adresses

Pour définir la plage d’adresses configurées par *DHCP*, je ne savais pas si il fallait inclure ou non la box Internet dans la plage d’adresses à configurer. Seulement, et cela paraît logique, on a implémenté précédemment un script qui modifie l’adresse IP de la box *Internet*, il ne fallait pas tout confondre et ici le serveur *DHCP* ne va traiter que les clients qui seront connectés à la box Internet. Ainsi, il faut **toujours commencer sa plage d’adresses *DHCP* à au moins +1 par rapport à sa propre adresse IP et ne jamais inclure l’adresse IP du serveur *DHCP* dans la plage d’adresses.**

```
stud@box:~$ cat /etc/dhcp/dhcpd.conf
ddns-update-style none;
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
default-lease-time 600;
max-lease-time 7200;
log-facility local7;

# Nombre de machines configurées : 10
subnet 192.168.0.0 netmask 255.255.0.0 {
        range 192.168.1.2 192.168.1.11;
}
```

Étapes de mon script DHCP

Voici les étapes qui composent mon script DHCP et permettent de configurer sa propre plage d’adresses qui seront attribuées.

- Vérifie que le script a bien été appelé avec deux arguments :
  - le nombre de machines à inclure dans la plage d’adresses (champ entré par l’utilisateur),
  - l’adresse réseau, définit grâce à la méthode expliquée au dessus, en *PHP*.
- Stockage des arguments dans des variables pour éviter de faire $1 mais $devices_number par exemple.

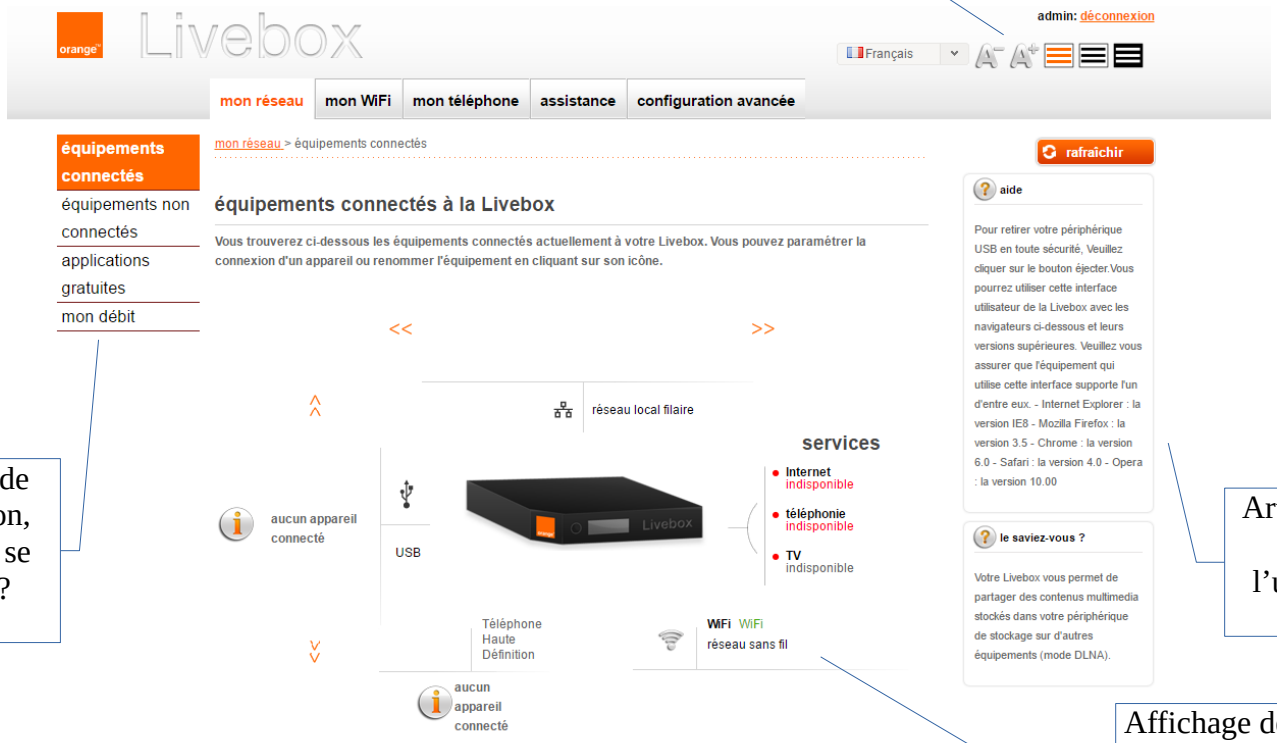
- Division de l’adresse réseau, de l’adresse IP et du masque de sous-réseau en 4 octets pour chaque.
- Calcul du *CIDR* (format /24 équivalent à 255.255.255.0 quand on parle de masque de sous-réseau, il permet aussi de définir le nombre de bits à 0 en partant de la gauche pour définir le binaire du masque de sous-réseau).
    - Le script parcourt les 4 octets du masque de sous-réseau.
    - Convertie l’octet en question, en binaire.
    - Compte le nombre de bits qui sont à 1.
    - Incrémente le *CIDR*.
- Refuse un *CIDR* = 31 ou 32, car cela voudrait dire qu’il faudrait créer respectivement une plage d’adresses pour 2 et 1 machine ce qui n’est pas possible car il faut compter l’adresse réseau et l’adresse de diffusion ce qui ne laisse plus aucune adresse hôte disponible.
- Calcule le nombre d’hôtes maximum configurable à partir du *CIDR*.
- Vérifie que le nombre d’adresses à configurer avec *DHCP* est inférieur ou égal au nombre d’hôtes maximum calculé juste au dessus.
- Incrémente de 1 la valeur du dernier octet de l’adresse IP pour justement ne pas inclure l’adresse de la box Internet dans cette plage d’adresses.
- Additionne la valeur du dernier octet de l’adresse IP avec le nombre d’hôtes choisi par l’utilisateur.
- Réécriture du fichier de configuration *DHCP* **/etc/dhcp/dhcpd.conf** à partir du TP2 en *Internet et les Services Réseaux* en y mettant cette fois-ci les bonnes variables au bon endroit.
- Redémarre le serveur DHCP.

Comparatif de interfaces web des *FAI*

Voici une liste de FAI avec pour chaque un tableau, résumant ce que j’ai bien aimé / moins aimé chez ces derniers. Ceci m’a donné des idées de fonctionnalités, parfois très utile et pourtant semble simple à mettre en place. Mais aussi des idées pour rendre l’interface web ergonomique, facile à prendre en main et donc intuitive, un critère indispensable pour un utilisateur qui n’a pas suffisamment de connaissances en informatique pour comprendre certains termes.

Mettre ça dans un onglet « Paramètre » ?

Orange



2 barres de navigation, facile de se perdre ?

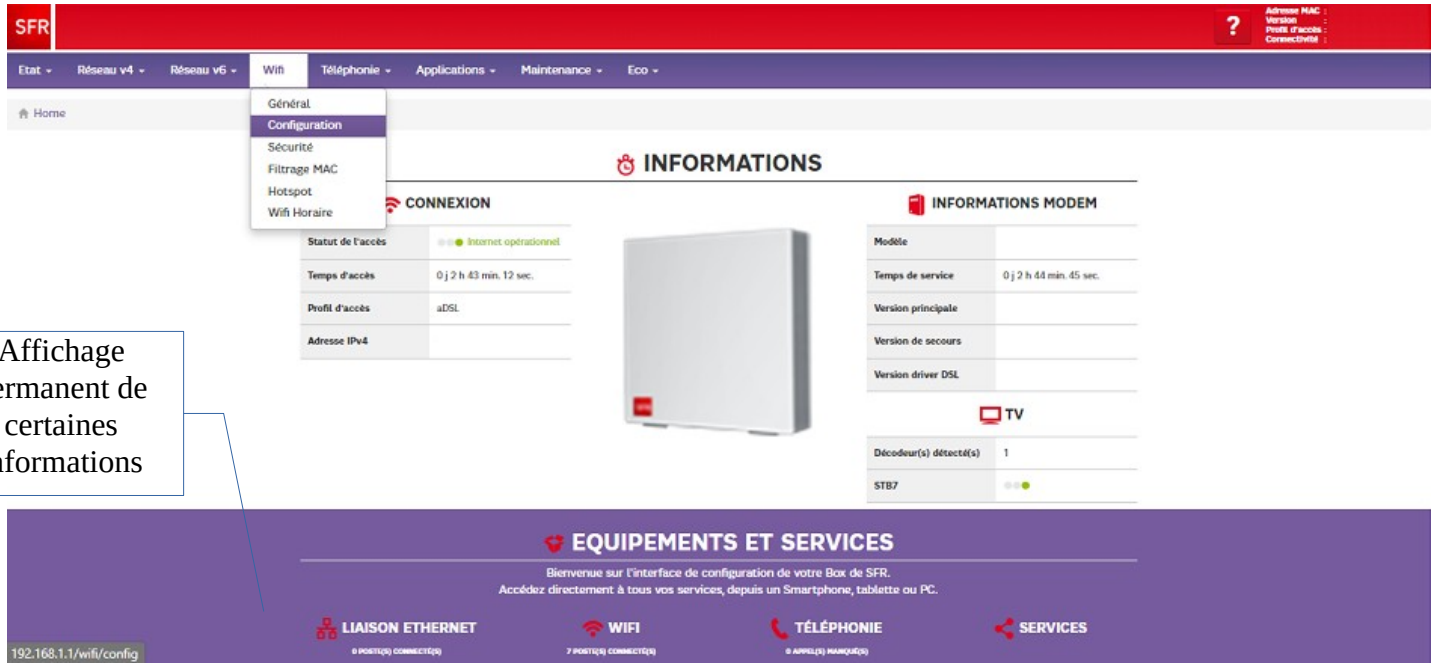
Articles pour aider l'utilisateur

Affichage de l'état des services disponibles

Points positifs	Points négatifs
Affiche l'état des services (par exemple, *sudo systemctl status isc-dhcp-server*) pour savoir si un service (serveur) est disponible (tourne) ou pas.	2 barres de navigation, une à l'horizontale et une à la verticale. Possibilité de se perdre ?
Propose des articles sur le côté ***aide*** et ***le saviez-vous*** pour **aider l'utilisateur à mieux prendre en main l'interface web**.	En terme d'implémentation, elle me semble moins intuitive et plus difficile que celle de SFR à mettre en place.
Liste les appareils connectés à la box Internet.	Les configurations de la page en haut à droite mériteraient d'être dans un onglet « Paramètres » pour aussi pouvoir modifier les informations de l'utilisateur ?
Traduire la page en plusieurs langues pour rendre le fournisseur d'accès à Internet utilisable à l'international.	
Propose de désactiver l'interface Internet / le Wi-Fi : ce qui revient à désactiver l'interface eth0. **Permettre de désactiver une interface spécifique**.	
Propose de mettre à jour, redémarrer, … (sudo apt update && sudo apt upgrade, sudo reboot, …)	
Affiche le chemin dans l'arborescence où l'on se situe.	

« Adresse MAC » n’est pas forcément un terme qui parle à tout le monde… Certaines informations ont-elles vraiment besoin d’être affichées ?

SFR

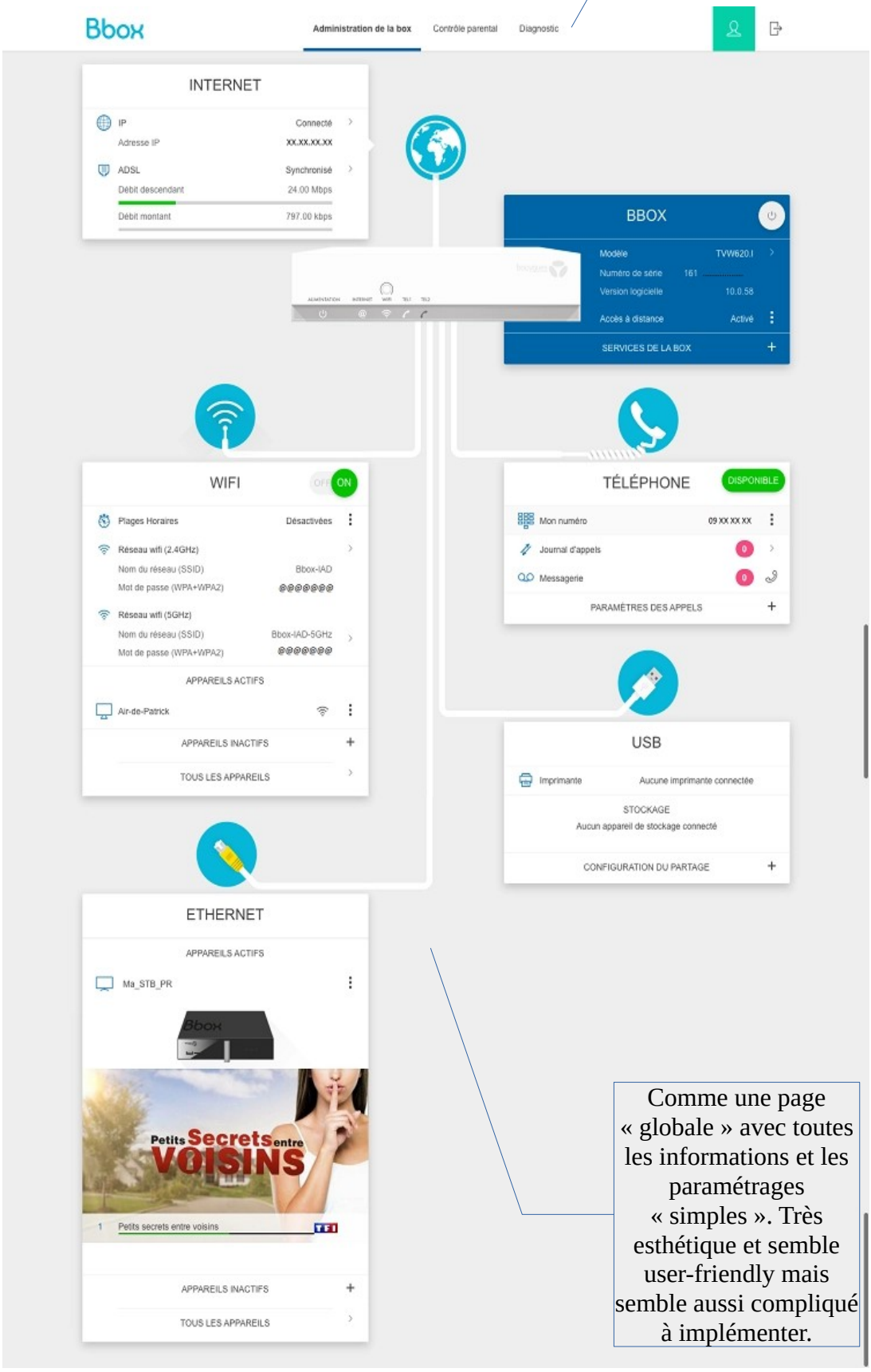


Affichage permanent de certaines informations

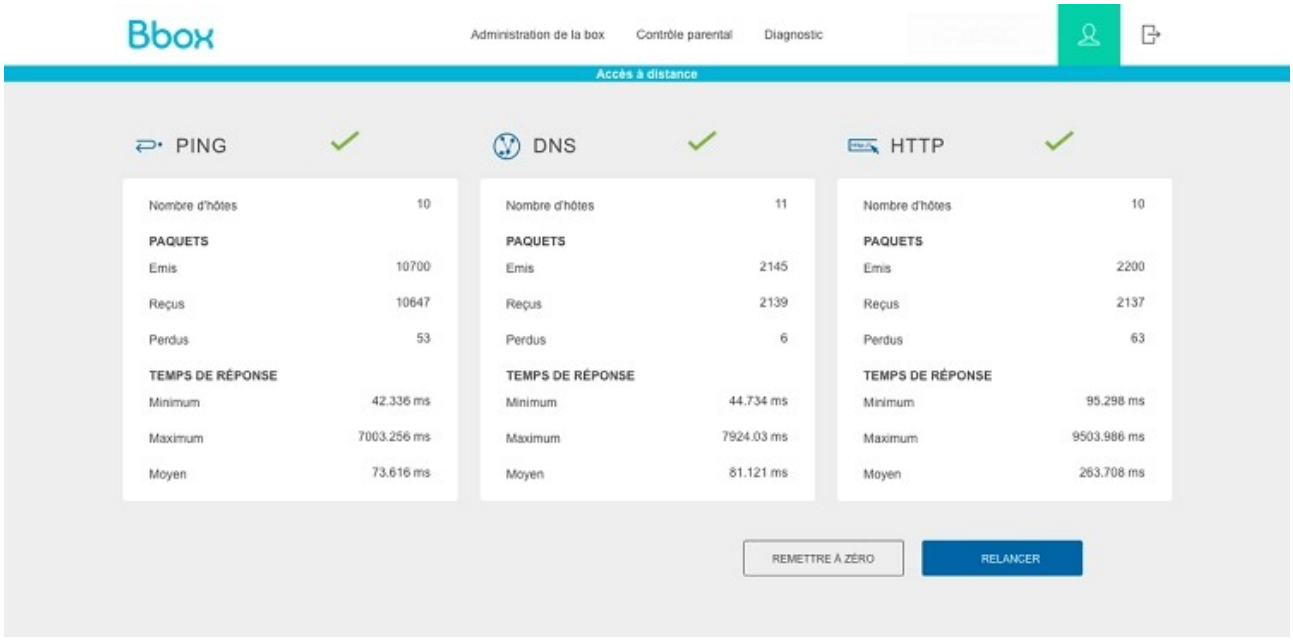
Points positifs	Points négatifs
Un affichage permanent, dynamique et automatique du nombre d’appareils connectés, appels manqués, services en cours d’utilisation, etc. (footer violet)	Certaines informations ne sont peut-être pas nécessaires
Semble simple à implémenter. Plusieurs bandeaux.	Certains termes ne parlent pas à tout le monde «*Adresse MAC*» ? C’est ce qu’il faudra implémenter à un moment du projet : proposer une version **débutante** vs une version **avancée** et utiliser en fonction des termes plus appropriés, des fonctionnalités plus avancées selon les cas d’utilisation de la box *Internet*.
	Où se fait le paramétrage du compte ? Elle me semble moins **user-friendly**
	Beaucoup d’onglets, facile de se perdre ?
	Mauvais choix des couleurs, le rouge et le violet ne vont pas très bien ensemble.

Bouygues Telecom

Barre de navigation simple avec peu d'onglets et des icônes pour les fonctionnalités liées à l'authentification



Comme une page « globale » avec toutes les informations et les paramétrages « simples ». Très esthétique et semble user-friendly mais semble aussi compliqué à implémenter.



Points positifs	Points négatifs
L'interface est **ergonomique**, **semble facile à prendre en mains** (très peu d'onglets) = **user-friendly**.	Comment configurer les services proposés dans la page « Administration de la Box » ?
Onglet «Diagnostic» pour **tester la connectivité de son réseau.**	Le design semble compliqué à implémenter.
Accès à distance activé / désactivé : autoriser ou désactiver SSH ?	
Modifier le hostname de la box Internet directement **via l'interface web**.	

Après l'étude de ces 3 FAI, je compte faire des onglets pour les configurations plus avancées (DNS, DHCP, ..) et garder un onglet pour résumer l'état des services, le test du débit, les informations simples à résumer pour ne pas se perdre et avoir des onglets jugés « inutiles » car ils ne présentent pas énormément d'informations et ne nécessiteraient pas un onglet à part entière.

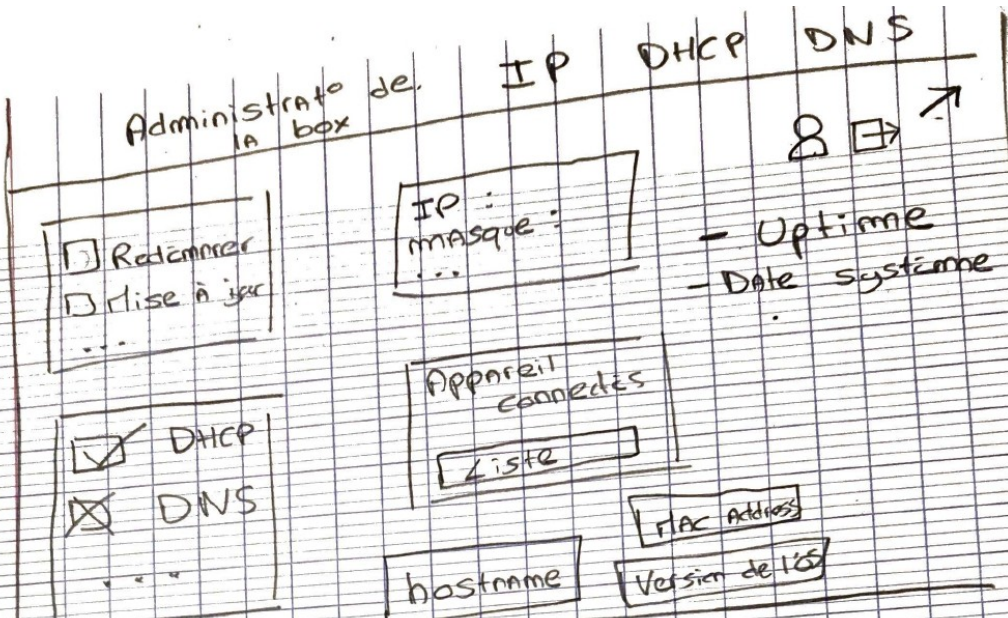
De plus, je pense qu'il faut rester simple pour faire en sorte que l'application soit facile à prendre en main, ergonomique et intuitive.

Premier visuel brouillon de mon application

Je souhaite donc faire un menu de navigation avec peu d'onglets. 1 onglet sera utilisé comme page d'accueil mais aussi pour visualiser l'état de la box Internet et de ses services proposés (DHCP, DNS, …) et proposera des fonctionnalités simples qui ne nécessitent pas d'onglets à part entière (modification du hostname, mise à jour de la box Internet, …).

Cela m'a fait aussi penser à ne pas oublier d'intégrer à un stade du projet une gestion de l'authentification à l'interface web pour ne pas permettre à n'importe qui d'avoir accès à la box Internet.

Enfin, elle sera responsive, un design simple, joli, professionnel, pour permettre à « n'importe qui » de facilement la prendre en main.



Barre de navigato reprise de Bouygues

L'onglet Administrato fait office d'accueil proposant pleins d'informato simples et utiles par rapport à la box Internet + offre des fonctionalités simples

Appareils connectés = appareils ayant obtenus une adresse IP grâce au serveur DHCP de la box Internet

* Couleurs : Bleu
  - vert, orange, rouge pour l'état de DHCP, DNS, Apache, ...

* Utilisation de W3-CSS, framework CSS utilisé en I1 informatique, simple à prendre en main, pour une interface web jolie, professionnelle et ergonomique

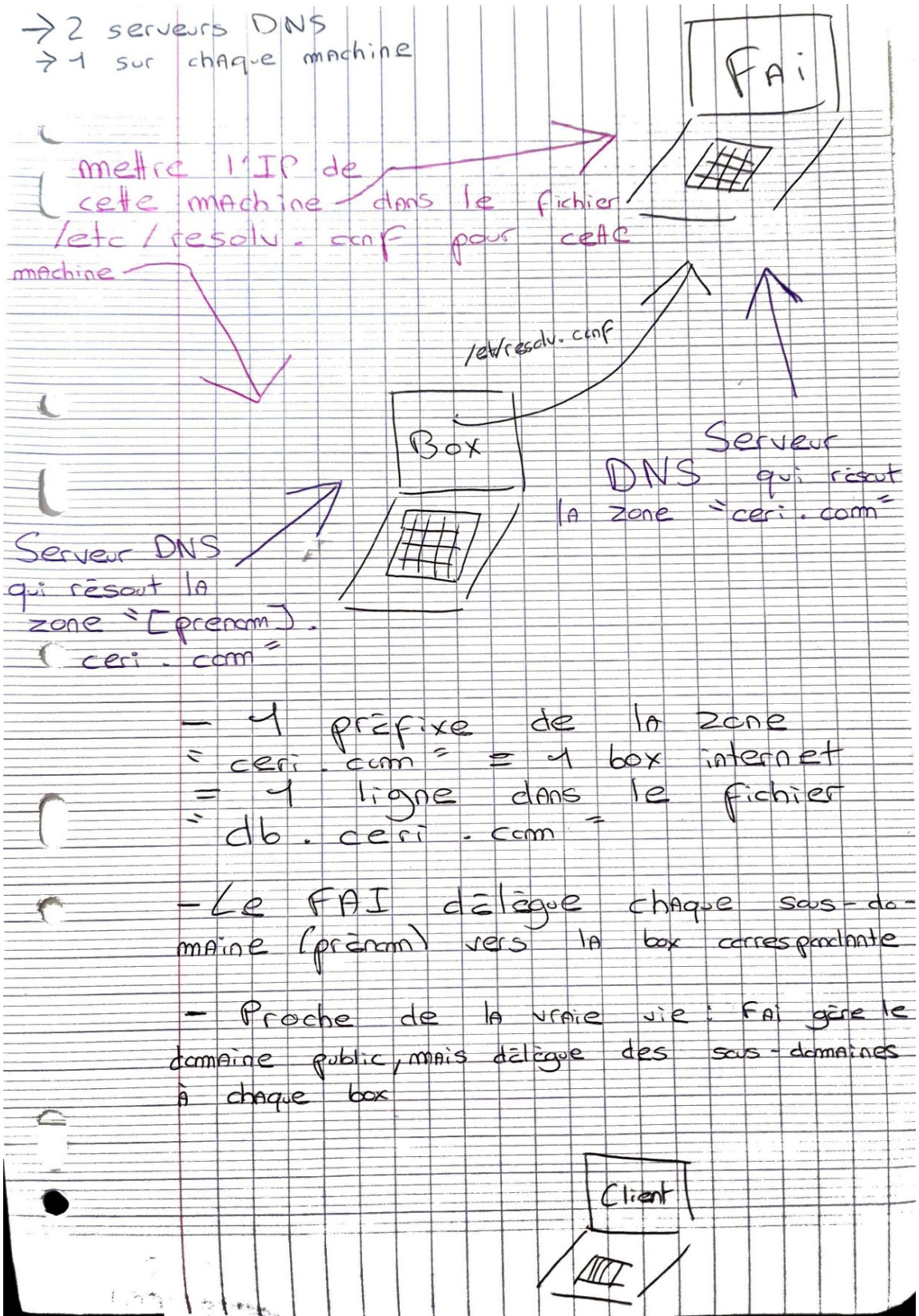
* doit être responsive

* penser à implémenter l'authentification sur l'interface - web pour ne pas laisser n'importe qui paramétrer la box.

Formulaire DNS

La troisième étape de ce projet a consisté à créer, sur l'interface web, un formulaire permettant de modifier le nom de domaine de la box Internet.

Brainstorming



Pour comprendre comme implémenter la fonctionnalité DNS à mon projet. Il fallait que je comprenne c**omment mes serveurs DNS allaient fonctionner, collaborer, dans le cadre de mon projet**. Déjà, il fallait implémenter 2 serveurs DNS :

- **Sur la Box *Internet***, en théorie, **chaque box Internet possède son serveur DNS et résout le domaine prénom.ceri.com**, où prénom est configurable sur l'interface web, et où ceri.com correspond à l'identité du FAI, c'est comme pour free.fr chez *Free*, bbox.fr chez *Bouygues*, orange.fr chez *Orange*, …
- **Sur le Fournisseur d'Accès à Internet**, une machine indépendante, avec une adresse IP fixe, qui se charge de résoudre le domaine ceri.com et délègue à la bonne box Internet en fonction du préfixe (prénom). En effet, il y a N lignes où N est le nombre de box Internet.

Comment les informations communiquent-elles ?

La box *Internet* a dans son fichier */etc/resolv.conf* le *nameserver* du FAI ainsi que search *ceri.com* pour ne mettre uniquement le préfixe lorsque l'on cherche à ping. Par exemple, au lieu de ping florent.ceri.com, **on ne ping que *florent*** (le prénom configuré) et le système va se charger de concaténer mon paramètre avec le domaine *ceri.com*.

```
stud@box:/etc/bind$ ping florent
PING florent.ceri.com (192.168.1.18) 56(84) bytes of data.
64 bytes from 192.168.1.18 (192.168.1.18): icmp_seq=1 ttl=64 time=0.075 ms
64 bytes from 192.168.1.18 (192.168.1.18): icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from 192.168.1.18 (192.168.1.18): icmp_seq=3 ttl=64 time=0.050 ms
64 bytes from 192.168.1.18 (192.168.1.18): icmp_seq=4 ttl=64 time=0.075 ms
64 bytes from 192.168.1.18 (192.168.1.18): icmp_seq=5 ttl=64 time=0.079 ms
^C
--- florent.ceri.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4075ms
rtt min/avg/max/mdev = 0.050/0.066/0.079/0.014 ms
stud@box:/etc/bind$ ping fai
PING fai.ceri.com (192.168.1.22) 56(84) bytes of data.
64 bytes from 192.168.1.22 (192.168.1.22): icmp_seq=1 ttl=64 time=0.492 ms
64 bytes from 192.168.1.22 (192.168.1.22): icmp_seq=2 ttl=64 time=0.657 ms
64 bytes from 192.168.1.22 (192.168.1.22): icmp_seq=3 ttl=64 time=0.490 ms
64 bytes from 192.168.1.22 (192.168.1.22): icmp_seq=4 ttl=64 time=0.400 ms
^C
--- fai.ceri.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3027ms
rtt min/avg/max/mdev = 0.400/0.509/0.657/0.096 ms
stud@box:/etc/bind$ cat /etc/resolv.conf
nameserver 192.168.1.22
search ceri.com
stud@box:/etc/bind$
```

Sur la machine *Client*, on y met le *nameserver* de la Box *Internet*, en effet, le client qui peut être n'importe quelle machine par exemple d'un foyer, n'a pas besoin de résoudre les adresses IP des autres box Internet du pays, voir du monde. C'est pourquoi, le *Client* ne pourra résoudre que les domaines configurés sur sa box *Internet*.

Paramétrage du fichier resolv.conf

Le fichier ***resolv.conf*** **sert à décrire quel serveur DNS sera utilisé pour résoudre les noms de domaines**.

Il faut savoir que *DHCP* permet avec l'attribut ***option domain-name servers*** de donner par défaut dans le fichier /etc/resolv.conf des machines qui auront reçus une adresse IP par ce serveur DHCP, l'adresse IP d'un serveur DNS. C'est ce que **j'ai mis en place pour que chaque *Client* qui reçoit une adresse IP via *DHCP*, ait par défaut de configuré le serveur *DNS***.

```
root@box:/etc/bind# cat /etc/dhcp/dhcpd.conf
ddns-update-style none;
option domain-name "example.org";
default-lease-time 600;
max-lease-time 7200;
log-facility local7;

# Nombre de machines configurées : 6
subnet 192.168.1.0 netmask 255.255.255.0 {
        range 192.168.1.19 192.168.1.24;
        option domain-name-servers 192.168.1.18;
}

root@box:/etc/bind#
```

Concernant la box Internet, à chaque lancement du système, elle met par défaut l'adresse IP du serveur DNS du FAI grâce au fichier ***/etc/systemd/resolved.conf*** **qui permet de configurer le fichier *./etc/resolv.conf*.** Puisque la box Internet ne reçoit pas son adresse IP avec *DHCP*, mais de manière statique, c'est avec cette méthode que j'obtiens automatiquement, à chaque lancement de la box Internet, **le bon serveur *DNS* de configuré.**

```
root@box:/etc/bind# cat /etc/systemd/resolved.conf
#  This file is part of systemd.
#
#  systemd is free software; you can redistribute it and/or modify it
#  under the terms of the GNU Lesser General Public License as published by
#  the Free Software Foundation; either version 2.1 of the License, or
#  (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
#
# See resolved.conf(5) for details

[Resolve]
DNS=192.168.1.22
FallbackDNS=
Domains=ceri.com
#LLMNR=no
#MulticastDNS=no
#DNSSEC=no
#Cache=yes
#DNSStubListener=yes
root@box:/etc/bind#
```

Étapes de mon script DNS

- Vérifie que le script a bien été appelé avec un seul argument (le prénom, préfixe du domaine « ceri.com ») : florent.ceri.com
- Stockage de l'argument dans une variable pour pouvoir plus intuitivement le manipuler.
- Attribue, si il est vide (cas où le script est utilisé en ligne de commandes, car une vérification de la validité de l'argument est déjà effectuée en PHP avant d'exécuter le script), le hostname comme valeur de l'argument du script.
- Récupère l'adresse IP et le masque de sous-réseau pour pouvoir calculer l'adresse réseau de la configuration actuelle.

```bash
#!/bin/bash

# Vérifie que le script a bien été appelé avec un seul argument :
# - Le prénom : préfixe du domaine "ceri.com" (ex: florent.ceri.com)
if [ "$#" -ne 1 ]
then
  exit 1
fi

# Stockage du prénom choisi par l'utilisateur dans une variable
newFirstName="$1"

# Si le prénom choisi est nul, on met le hostname comme prénom choisi
if [ -z "$newFirstName" ]
then
  newFirstName=$(hostname)
fi

# Récupère le prénom de la configuration actuelle
activeFirstName=$(cat /etc/bind/named.conf.local | grep "zone" | grep "ceri.com" | cut -d " " -f 2 | cut -d "." -f 1 | cut -d '"' -f 2)

# Récupère l'adresse IP de la configuration actuelle.
current_ip_address=$(cat /etc/network/interfaces | grep "address" | cut -d" " -f 2)
# Sépare l'adresse IP en 4 octets
IFS=. read -r ip_octet1 ip_octet2 ip_octet3 ip_octet4 <<< "$current_ip_address"

# Récupère le masque de sous-réseau de la configuration actuelle.
current_subnet_mask=$(cat /etc/network/interfaces | grep "netmask" | cut -d" " -f 2)
# Sépare le masque de sous-réseau en 4 octets
IFS=. read -r subnet_mask_octet1 subnet_mask_octet2 subnet_mask_octet3 subnet_mask_octet4 <<< "$current_subnet_mask"

# Calcule l'adresse réseau
network_address=$(printf "%d.%d.%d.%d\n" "$((ip_octet1 & subnet_mask_octet1))" "$((ip_octet2 & subnet_mask_octet2))" "$((ip_octet3 & subnet_mask_octet3))" "$((ip_octet4 & subn
IFS=. read -r network_address_octet1 network_address_octet2 network_address_octet3 network_address_octet4 <<< "$network_address"

```

- Calcule le CIDR qui permet en fonction de sa valeur, de définir la bonne partie réseau ainsi que la partie hôte pour la configuration des fichiers DNS.

```bash
# Calcule le CIDR (nombre de bits à 1 dans le masque de sous-réseau)
cidr=0
for octet in $subnet_mask_octet1 $subnet_mask_octet2 $subnet_mask_octet3 $subnet_mask_octet4
do
  # Convertie l'octet en binaire
  binary=$(printf "%08d" "$(bc <<< "obase=2;$octet")")
  # Compte combien de bits sont à 1
  count=$(grep -o "1" <<< "$binary" | wc -l)
  # Incrémente le CIDR
  cidr=$((cidr + count))
done

# Détermine quel(s) octet(s) définie/définissent le réseau
if [ "$cidr" -le 8 ]
then
  network_part=$(echo "$network_address_octet"1)
  ip_address_part=$(echo "$ip_octet2.$ip_octet3.$ip_octet4")
  reversed_network_part=$(echo "$network_address_octet1")
elif [ "$cidr" -le 16 ]
then
  network_part=$(echo "$network_address_octet1.$network_address_octet2")
  ip_address_part=$(echo "$ip_octet3.$ip_octet4")
  reversed_network_part=$(echo "$network_address_octet2.$network_address_octet1")
elif [ "$cidr" -le 24 ]
then
  network_part=$(echo "$network_address_octet1.$network_address_octet2.$network_address_octet3")
  ip_address_part=$(echo "$ip_octet4")
  reversed_network_part=$(echo "$network_address_octet3.$network_address_octet2.$network_address_octet1")
else
  network_part=$(echo "$network_address_octet1.$network_address_octet2.$network_address_octet3.$network_address_octet4")
  reversed_network_part=$(echo "$network_address_octet4.$network_address_octet3.$network_address_octet2.$network_address_octet1")
fi
```

- Réécrit le fichier ***named.conf.local*** de la box Internet situé dans le dossier ***/etc/bind*** pour redéfinir les zones en fonction du nouveau préfixe à configurer.
- S'assure que ce fichier **ne retourne pas une erreur de syntaxe** avant de modifier les autres fichiers grâce à la commande ***named-checkconf***.
- Récupère le numéro *SERIAL* (cela m'a fait pensé à l'attribut que l'on peut donner à une clé primaire dans une base de données, elle est incrémentée de 1 à chaque fois, ici c'est pareil : à chaque modification, il faut incrémenter ce nombre, pour que l'on puisse avoir une trace des modifications).
- Supprime l'ancien fichier db.prénom.ceri.com, où « prénom » était l'ancien préfixe configuré.

```
69  # Réecriture du fichier named.conf.local
70  {
71      # Définition de la zone de recherche directe pour déclarer la zone [prénom].ceri.com
72      echo "zone \"$newFirstName.ceri.com\" {"
73      echo "  type master;"
74      echo "  file \"/etc/bind/db.$newFirstName.ceri.com\";"
75      echo "};"
76      echo ""
77      # Définition de la zone de recherche inverse
78      echo "zone \"$reversed_network_part.in-addr.arpa\" {"
79      echo "  type master;"
80      echo "  file \"/etc/bind/reverse.$network_part.db\";"
81      echo "};"
82  } | sudo tee /etc/bind/named.conf.local > /dev/null
83
84  # Vérifie la syntaxe du fichier named.conf.local
85  if ! named-checkconf /etc/bind/named.conf.local
86  then
87    exit 3
88  fi
89
90  # Récupération du SERIAL qui permet de savoir combien de fois a été modifié le fichier forward
91  nb_serial_db_file=$(cat db.$activeFirstName.ceri.com | grep "Serial" | cut -d ";" -f 1 | tr -d '[:blank:]')
92  # L'incrémente
93  (( nb_serial_db_file++ ))
94
95  # Récupération du SERIAL qui permet de savoir combien de fois a été modifié le fichier reverse
96  nb_serial_reverse_file=$(cat reverse.$network_part.db | grep "Serial" | cut -d ";" -f 1 | tr -d '[:blank:]')
97  # L'incrémente
98  (( nb_serial_reverse_file++ ))
99
100 # Supprime le fichier db.[ancienPrénom].ceri.com
101 if [ -f "/etc/bind/db.${activeFirstName}.ceri.com" ]
102 then
103   sudo rm "/etc/bind/db.${activeFirstName}.ceri.com"
104 fi
```

- Fait une **copie** du fichier ***db.local*** en ***db.$newFirstName.ceri.com***, où ***$newFirstName*** est l'argument fourni au script.
- Modifie ce fichier en remplaçant les valeurs clés par les variables définies tout au long du script.
- Fait une copie du fichier ***db.127*** en *reverse.**$network_part.db***, où ***$network_part*** est la partie réseau calculée avec le ***CIDR***.
- Modifie ce fichier en remplaçant les valeurs clés par les variables définies tout au long du script.

```
# Copie du fichier db.local en db.[nouveauPrénom].ceri.com
sudo cp db.local db.$newFirstName.ceri.com

# Modifie le contenu du fichier db.[nouveauPrénom].ceri.com
{
    echo "\$TTL    604800"
    echo "@         IN      SOA     $newFirstName.ceri.com. admin.$newFirstName.ceri.com. ("
    echo "                              $nb_serial_db_file         ; Serial"
    echo "                           604800         ; Refresh"
    echo "                            86400         ; Retry"
    echo "                          2419200         ; Expire"
    echo "                           604800 )       ; Negative Cache TTL"
    echo ""
    echo "@                        IN      NS      $newFirstName.ceri.com."
    echo "${newFirstName}.ceri.com.         IN      A       $current_ip_address"
} | sudo tee /etc/bind/db.$newFirstName.ceri.com > /dev/null

# Supprime les fichiers reverse
sudo rm /etc/bind/reverse*

# Copie du fichier db.127 en reverse.[partieRéseau].db
sudo cp db.127 reverse.$network_part.db

# Modifie le contenu du fichier reverse.[partieRéseau].db
{
    echo "\$TTL    604800"
    echo "@       IN      SOA     $newFirstName.ceri.com. admin.$newFirstName.ceri.com. ("
    echo "                              $nb_serial_reverse_file         ; Serial"
    echo "                           604800         ; Refresh"
    echo "                            86400         ; Retry"
    echo "                          2419200         ; Expire"
    echo "                           604800 )       ; Negative Cache TTL"
    echo ""
    echo "@                     IN      NS      $newFirstName.ceri.com."
    echo "$ip_address_part      IN      PTR     $newFirstName.ceri.com."
} | sudo tee /etc/bind/reverse.$network_part.db > /dev/null
```

- Réalise un backup des fichiers qui viennent d'être configurés dans le dossier /var/backups/FAI.
- Redémarre le serveur DNS de la box Internet.
- Crée un fichier de commandes **nsupdate** (une partie à nsupdate est dédiée juste après), concrètement ***nsupdate*** **permet de modifier des informations sur un serveur DNS à distance grâce à une clé** (même principe qu'une clé SSH) **pour vérifier son identité.**
- Écriture dans ce fichier de commandes crée, au format attendu par *nsupdate*.
- Exécution de la commande, avec comme paramètre ma clé commune et connue par la **box** Internet et par le *FAI*.

```
143  # Crée le dossier où seront stockés les backups du projet s'il n'existe pas déjà
144  if [ ! -d "/var/backups/FAI" ]
145  then
146    sudo mkdir /var/backups/FAI
147  fi
148
149  # Crée le dossier "dns" dans le dossier /var/backups/FAI s'il n'existe pas déjà pour y stocker les backups des fichiers DNS configurés
150  if [ ! -d "/var/backups/FAI/dns" ]
151  then
152    sudo mkdir /var/backups/FAI/dns
153  fi
154
155  # Récupère la date pour horodater les backups
156  date=$(date '+%Y-%m-%d_%H:%M:%S')
157  sudo mkdir /var/backups/FAI/dns/$date
158
159  # Copie du fichier named.conf.local
160  sudo cp /etc/bind/named.conf.local /var/backups/FAI/dns/$date/named.conf.local
161  # Copie du fichier db.[Prénom].ceri.com
162  sudo cp /etc/bind/db.$newFirstName.ceri.com /var/backups/FAI/dns/$date/db.$newFirstName.ceri.com
163  # Copie du fichier reverse.[partieRéseau].db
164  sudo cp /etc/bind/reverse.$network_part.db /var/backups/FAI/dns/$date/reverse.$network_part.db
165
166  # Redémarre le serveur DNS pour appliquer les changements
167  sudo systemctl restart bind9
168
169  # Création du fichier update_on_isp.txt s'il n'existe pas déjà
170  if [ ! -f "/etc/bind/update_on_isp.txt" ]
171  then
172    sudo touch /etc/bind/update_on_isp.txt
173  fi
174
175  # Mise en place du fichier de commandes nsupdate pour actualiser la configuration de la zone sur le FAI
176  {
177      echo "server 192.168.1.22"  # IP du serveur DNS du FAI (ISP)
178      echo "zone ceri.com"
179      echo ""
180      # Suppression des deux anciens records
181      echo "update delete $activeFirstName.ceri.com. NS"
182      echo "update delete $activeFirstName.ceri.com. A"
183      # Ajout des nouvelles lignes de la nouvelle configuration
184      echo "update add $newFirstName.ceri.com. 86400 NS $newFirstName.ceri.com."
185      echo "update add $newFirstName.ceri.com. 86400 A $current_ip_address"
186      echo ""
187      echo "send"
188  } | sudo tee /etc/bind/update_on_isp.txt > /dev/null
189
190  # Envoie le fichier de commandes au FAI en utilisant la clé TSIG-KEYGEN générée
191  sudo nsupdate -k /etc/bind/remote-key.key /etc/bind/update_on_isp.txt
```

Mise en place de nsupdate

Sans nsupdate, le fichier DNS de la box Internet et la configuration fonctionne bien uniquement si l'on remplace le serveur DNS qui traduit les noms de domaine par celui du serveur DNS de la box Internet. Ce que l'on veut, c'est pouvoir demander au serveur DNS se trouvant sur la machine jouant le rôle du FAI, et en fonction du préfixe, il délègue la résolution à la bonne box *Internet*.

Pour cela, il faut pouvoir, depuis la box *Internet*, modifier les informations du fichier db.ceri.com, se trouvant sur la machine du FAI. Et c'est avec **nsupdate** que j'ai mis ça en place. J'avais aussi pensé à utiliser *SSH* pour écraser l'ancienne version du fichier, comme quand j'utilise *scp* pour transférer la nouvelle version d'un fichier sur une machine. Mais **nsupdate** est la solution qui **s'adapte le mieux** puisqu'elle concerne exclusivement *DNS*. Voici la mise en place de cette solution :

- Génération d'une clé ***TSIG-KEYGEN*** avec l'algorithme ***HMAC-SHA256***.

```
stud@fai:/var/lib/bind$ cat /etc/bind/remote-key.key
key "remote-key" {
        algorithm hmac-sha256;
        secret
};
stud@fai:/var/lib/bind$ tsig-keygen
```

- Déplace le fichier ***db.ceri.com*** dans le dossier ***/var/lib/bind*** qui est le répertoire conçu pour la génération des journaux pour des modifications dynamiques. Par conséquent, j'ai donc modifié la localisation de mon fichier *db.ceri.com* dans le fichier **named.conf.local** pour décrire la zone « *ceri.com* »
- Ajout d'un bloc pour y décrire ma clé généré avec **TSIG-KEYGEN** et l'utilisé pour la modification de ma zone **ceri.com**.

```
stud@fai:/var/lib/bind$ ls -l
total 12
-rw-r--r-- 1 bind bind   53 Nov 21 08:47 bind9-default.md5sum
-rw-r--r-- 1 bind bind  360 Nov 22 12:04 db.ceri.com
-rw-r--r-- 1 bind bind 1882 Nov 22 11:28 db.ceri.com.jnl
stud@fai:/var/lib/bind$ cat /etc/bind/named.conf.local
zone "ceri.com" {
        type master;
        file "/var/lib/bind/db.ceri.com";
        update-policy {
                grant remote-key zonesub any;
        };
};

key "remote-key" {
        algorithm hmac-sha256;
        secret "Zl25Sou4/5at5s7dCcySBbIR9Iu23aCgVTF/+VgYf4E=";
};

stud@fai:/var/lib/bind$
```

À noté que j'ai eu quelques problèmes à générer cette fameuse clé en utilisant **TSIG-KEYGEN**. J'ai découvert la notion d'**entropy** d'un système que l'on peut connaître en utilisant la commande **cat** sur ce fichier ***cat /proc/sys/kernel/random/entropy_avail***

```
stud@fai:/var/lib/bind$ cat /proc/sys/kernel/random/entropy_avail
1134
stud@fai:/var/lib/bind$
```

Si elle est désormais aussi haute c'est parce que j'ai installé le package ***haveged*** qui m'a permis de l'augmenter. De ce que j'ai compris, cette valeur est liée à la génération de valeurs aléatoires. Ce que j'ai constaté c'est qu'après avoir **installé ce package** et après l'avoir **enable**, au lancement de la même commande **TSIG-KEYGEN**, la clé s'est générée instantanément.

Mes sources

Voici plusieurs forums, documentations, etc. qui m'ont aidées à résoudre des problèmes que j'avais, répondre à certaines de mes questions, pour ne pas rester bloquer et avancer sur le développement de ma box *Internet*.

- https://www.ibm.com/docs/en/power9/0009-ESS?topic=notebook-setting-ip-address-in-linux
- https://unix.stackexchange.com/questions/100588/using-ip-addr-instead-of-ifconfig-reports-rtnetlink-answers-file-exists-on-de
- https://www.php.net/manual/en/install.unix.debian.php
- *https://assistancepro.orange.fr/internet_livebox/utiliser/interface_de_configuration/livebox_pro_v3/livebox_pro_v__acceder_a_linterface_de_votre_livebox_en_mode_utilisateur-393685*
- *https://wiki.archlinux.org/title/Haveged*
- *https://www.ibm.com/docs/en/aix/7.3.0?topic=n-nsupdate-command*
- *https://stackoverflow.com/questions/409351/post-vs-serverrequest-method-post*
- *https://www.malekal.com/le-fichier-etc-resolv-conf-linux/*
- *https://cloudcone.com/docs/article/how-to-check-uptime-for-a-linux-server/*
- *https://www.hivelocity.net/kb/check-linux-version/*
- *https://labex.io/fr/tutorials/linux-linux-duplicate-filtering-271417*