Documentation
Toutes les parties

Documentation pour différent éléments

James Benone

v. 2025.08.21 - 15:54

Thursday 11 June 2026

Table des matières

1. Gestion d'un VPS/VDS/Serveur

1.1 Création et Configuration d'une LAMP

1.1.1 Prérequis

Un serveur sous linux ou une VM sous linux

1.1.2 Création d'une connexion sécurisée

Création du nom de domaine en local

Pour modifier le fichier hosts, sur windows, c'est `C:/Windows/System32/drivers/etc/hosts` et sur linux, c'est `/etc/hosts`

Ajouter une ligne avec :

Text Only

```
<ip-lamp> lamp.mshome.net
```

Création de la configuration

On va créer un fichier de config

Text Only

```
nano config
```

Et on va ajouté le contenu suivant :

Text Only

```
Host lamp
    HostName lamp.mshome.net
    User <utilisateur>
    Port 22
```

Générer une clé SSH

Génerer une clé SSH sur windows ou linux via la commande suivante :

Bash

```
ssh-keygen -t ed25519
```

Vous pouvez juste faire enter jusqu'à la création du fichier.

Récupérer la clé public que vous venez de créer en ouvrant le fichier sur windows ou via la commande suivante sous linux :

Bash

```
cat id_ed25519.pub
```

Et copier la clé

Mettre la clé sur la machine

Ensuite, connectez-vous une première fois sur la machine de manière physique ou via le shell (si vous utilisez une VM)

aller dans `~/.ssh/` et modifier le fichier `authorized_keys` ou créer le si il n'existe pas via VI ou NANO

Bash

```
nano authorized_keys
```

et ajouter la clé public que vous aviez fait juste avant

Votre première connexion via SSH

Ouvrez votre terminal et faite la commande suivante

Bash

```
ssh lamp
```

1.1.3 Installation et configuration des éléments nécessaire

Apache2

Pour installer apache2, il suffit de faire un :

Bash

```
sudo apt install apache2 -y
```

MariaDB

Pour installer MariaDB, il suffit de faire un :

Bash

```
sudo apt install mariadb-server mariadb-client -y
```

Puis de le configurer avec :

Bash

```
mariadb_secure_installation
```

Si il vous met des trucs, mettez oui à chaque fois sauf dans le cas d'une configuration parcticulière

PHP

Pour installer PHP, on va uniquement installer les paquets nécessaire via :

Bash

```
sudo apt install php php-dev php-mysql -y
```

TEST

Vous allez dans `/var/www/html`

Et vous allez créer le fichier `info.php` avec le contenu suivant :

PHP

```php
<?php
    // Affiche les informations de PHP
    phpinfo();
?>
```

Et vous allez dans votre url lamp.mshome.net/info.php

Vous devriez avoir la page PHP

XDebug

Dans un premier temps, on va installer XDebug avec l'installeur officiel de PHP via la commande suivante :

Bash

```
sudo pecl install xdebug
```

Ensuite vous allez modifier le fichier de configuration PHP dans :

Bash

```
# Remplacer <version> par votre version de PHP (dans ce cas 8.3)
sudo nano /etc/php/<version>/php.ini
```

Et vous allez toute à la fin de la page puis vous ajoutez :

INI

```ini
; blabla du dessus

[xdebug]
zend_extension=/usr/lib/php/20230831/xdebug.so
xdebug.start_with_request=yes
xdebug.mode=debug
```

Toujours dans php.ini, il faut aussi chercher les éléments `display_errors` et `display_startup_errors`, les valeurs par défaut sont `Off`, remplacés les par `On`.

Pour en finir, il suffit de relancer apache2 avec la commande suivante :

Bash

```
sudo service apache2 restart
```

ZSH et OhMyZsh

Pour installer ZSH, il suffit de faire :

Text Only

```
sudo apt install zsh -y
```

Et installer OhMyZsh :

Text Only

```
sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"
```

Si il vous propose : `Do you want to change your default shell to zsh? [Y/n]`, vous faite simplement la touche entrer.

Il suffit de changer le thême via :

Bash

```
nano .zshrc
```

Et de modifier `ZSH_THEME` par `bira` puis de mettre à jour avec :

Bash

```
source .zshrc
```

Configurer les permissions du dossier html

Premièrement, on va aller dans le dossier `/var/www`

Deuxièmement, on va donner les droits du dossier à Apache via la commande suivante :

Bash

```
sudo chown -R www-data:www-data html
```

Et maintenant, on va donner les droits au groupe de modifier le dossier et son contenu :

Bash

```
sudo chmod -R g+w html
```

Il suffit maintenant d'ajouter l'utilisateur au groupe `www-data` avec :

Bash

```
sudo usermod -aG www-data <utilisateur>
```

Pour finir, il suffit de sortir avec la commande `exit` et de revenir de la même manière dont vous êtes venu juste avant

1.1.4 Ajout d'un projet école

Ajout du fichier config

Créer le fichier ou modifier le via la commande :

Text Only

```
nano config
```

Et ajouter le contenu suivant :

Text Only

```
Host gitlab.ictge.ch
    HostName gitlab.ictge.ch
    User <utilisateur>
    Port 22002
```

Configurer git

Il faut pour cela modifier son nom d'utilisateur et son email de façon global via la commande :

Bash

```
git config --global user.name james-bnn
git config --global user.email james.bnn@eduge.ch
```

Création d'une clé SSH sur le serveur

Pour cela, vous pouvez simplement faire la commande :

Bash

```
ssh-keygen -t ed25519
```

Et faire entrer jusqu'à la création de la clé

Maintenant, il faut récupérer la clé via :

Bash

```
cat id_ed25519.pub
```

Copier son contenu.

Ajout de la clé sur GitLab

Aller sur votre instance GitLab et connectez-vous.

Ensuite, faite `Profile/Edit Profile/SSH Keys` et cliquer sur `Add new key`

Mettez ensuite la clé et faite `Add key`

Importer un projet

Sur votre instance GitLab, aller sur le projet que vous souhaitez ajouter et faite `Code->Clone with SSH`

Retourner sur le terminal SSH et aller dans `/var/www/html` et faite la commande suivante pour importer le projet :

Bash

```
git clone git@gitlab.ictge.ch:edu-bonvinp/webeng_es1.git
```

Ouvrir VSCode sur votre serveur

Ouvrez votre Visual Studio Code, cliquer sur la petite îcone en bas à droite des 2 flèches qui se croise puis faite `Connect to Host->lamp`

Une fois cela fait, faite `Open folder` et aller dans `/var/www/html/<votre-projet>`

Et voilà, vous êtes prêt à travailler.

15 octobre 2025

1.2 Protéger votre VPS/VDS/Serveur

1.2.1 Pare-feu

Dans un premier temps, on va installer un pare-feu, sur linux, on peut installer ufw via la commande suivante :

Bash

```
sudo apt install ufw -y
```

Il faut le configurer et mettre les ports que vous souhaitez exposer, rappelez-vous que moi de ports sont ouverts, mieux s'est :

Bash

```
sudo ufw allow 22 # Port SSH
sudo ufw allow 80 # Port HTTP
sudo ufw allow 443 # Port HTTPS
sudo ufw allow 3306 # Port MySQL (non recommandé)
```

Il faut absolument que vous laissiez le port SSH ouvert au risque de ne plus avoir accès à votre serveur.

Il ne vous reste plus qu'à l'activer

Bash

```
sudo ufw enable
```

Le message suivant va apparaitre, il suffit de mettre "y" :

Text Only

```
Command may disrupt existing ssh connections. Proceed with operation (y|n)?
```

1.2.2 Configurer SSH

IMPORTANT : Il faut avoir fait les sections "Générer une clé SSH" et "Mettre la clé sur la machine".

On va aller toucher au fichier de configuration, et pour cela, il faut aller dans :

Bash

```
sudo nano /etc/ssh/sshd_config
```

Et vous allez chercher la première ligne à changer : `PermitRootLogin` et vous devez la décommenté et mettre no :

Text Only

```
PermitRootLogin no
```

Vous allez aussi chercher : `PasswordAuthentication`

Text Only

```
PasswordAuthentication no
```

Il ne vous reste plus qu'à redémarrer ssh :

Bash

```
sudo service ssh restart
```

Maintenant, essayer de vous connectez en SSH via le mot de passe et vous devriez avoir le message suivant :

Bash

```
martin@192.168.1.144: Permission denied (publickey).
```

Si le mot de passe est toujours fonctionnel

Dans le cas ou il vous propose toujours de vous connectez par mot de passe, c'est qu'il y a un fichier config dans le dossier `/etc/ssh/sshd_config.d/` qui contient une instruction opposée. Vous devez donc inspecter les différents fichiers pour modifier la valeur.

Dans mon cas, ce fichier était `50-cloud-init.conf`.

1.2.3 Fail2Ban

Le fail2ban est un outil qui s'occupe permet d'empêcher le brute-force

Bash

```
sudo apt install fail2ban
```

Si vous souhaitez configurer le fail2ban, vous pouvez en modifiant le fichier de configuration :

Bash

```
sudo nano /etc/fail2ban/jail.d/defaults-debian.conf
```

Mettez le contenu suivant et modifier selon vos envies

INI

```
[DEFAULT]
banaction = nftables
banaction_allports = nftables[type=allports]
backend = systemd
ignoreip = 127.0.0.1
bantime = 6000
findtime = 600
maxretry = 3

[sshd]
enabled = true
port = ssh
```

```
logpath = %(sshd_log)s
backend = %(sshd_backend)s
```

Et il vous suffit de redémarrer le fail2ban

Bash

```
sudo service fail2ban restart
```


15 octobre 2025

1.3 Mettre la double authentification sur SSH

1.3.1 Installer le paquet

On va installer le packet de Google Authenticator via la commande suivante :

Bash

```
sudo apt install -y libpam-google-authenticator
```

1.3.2 Configurer

Dans un premier temps, on va activer la double authentification par SSH en allant modifier le fichier suivant :

Bash

```
sudo nano /etc/pam.d/sshd
```

Puis ajouter la ligne suivante :

Bash

```
auth required pam_google_authenticator.so
```

Mais il faut également commenter une ligne :

Bash

```
# @include common-auth
```

Puis redémarrer le service via la commande :

Bash

```
sudo service ssh restart
```

Maintenant, il faut modifier la configuration SSH :

Bash

```
sudo nano /etc/ssh/sshd_config
```

Premièrement, ajouter une ligne dans le fichier :

INI

```
AuthenticationMethods publickey,keyboard-interactive
```

Et il faut modifier les lignes :

INI

```
KbdInteractiveAuthentication yes # SET TO YES
ChallengeResponseAuthentication yes # SET TO YES
```

1.3.3 Activer sur un compte

Pour cela, il faut éxecuter la commande de Google :

Bash

```
google-authenticator
```

Il va vous demandé si vous voulez un token basé sur le temps

Text Only

```
Do you want authentication tokens to be time-based (y/n) y
```

Vous allez avoir un QRCode a scanné avec votre natel et rentrez le code.

Il va vous donnez des codes d'urgences, sauvegardez-les quelques parts de sécurisé et sur une autre plateforme.

Ensuite, il va vous posez quelques questions, il va falloir les configurer :

Text Only

```
Do you want me to update your "/home/<user>/.google_authenticator" file? (y/n) y

Do you want to disallow multiple uses of the same authentication
token? This restricts you to one login about every 30s, but it increases
your chances to notice or even prevent man-in-the-middle attacks (y/n) y

By default, a new token is generated every 30 seconds by the mobile app.
In order to compensate for possible time-skew between the client and the server,
we allow an extra token before and after the current time. This allows for a
time skew of up to 30 seconds between authentication server and client. If you
experience problems with poor time synchronization, you can increase the window
from its default size of 3 permitted codes (one previous code, the current
code, the next code) to 17 permitted codes (the 8 previous codes, the current
code, and the 8 next codes). This will permit for a time skew of up to 4 minutes
between client and server.
Do you want to do so? (y/n) n

If the computer that you are logging into isn't hardened against brute-force
login attempts, you can enable rate-limiting for the authentication module.
By default, this limits attackers to no more than 3 login attempts every 30s.
Do you want to enable rate-limiting? (y/n) y
```

Vous avez terminé la configuration de votre appareil.

15 octobre 2025

2. Publier un site sur un nom de domaine

Supposons que vous voulez publier un site fait en PHP se situant dans `/var/www/html/<votre-projet>`

Il y a plusieurs choses à faire

2.1 Commander le nom de domaine

Dans un premier temps, il vous faut un nom de domaine. Je vous recommande de prendre votre nom de domaine chez Infomaniak, cependant, ce n'est pas une obligation.

Commandez un nom de domaine si vous n'en avez pas.

2.2 Rediriger le nom de domaine

Rendez vous sur le panel de Infomaniak.

Vous allez vous rendre dans la partie `Web & Domains` et cliquer sur `Domaine` .

Cliquer ensuite sur le nom de domaine que vous souhaitez utiliser.

Une fois sur ce dernier, à gauche, vous devriez avoir `Zone DNS` , aller-y.

Vous arriverez sur une page ressemblant à celle-ci :

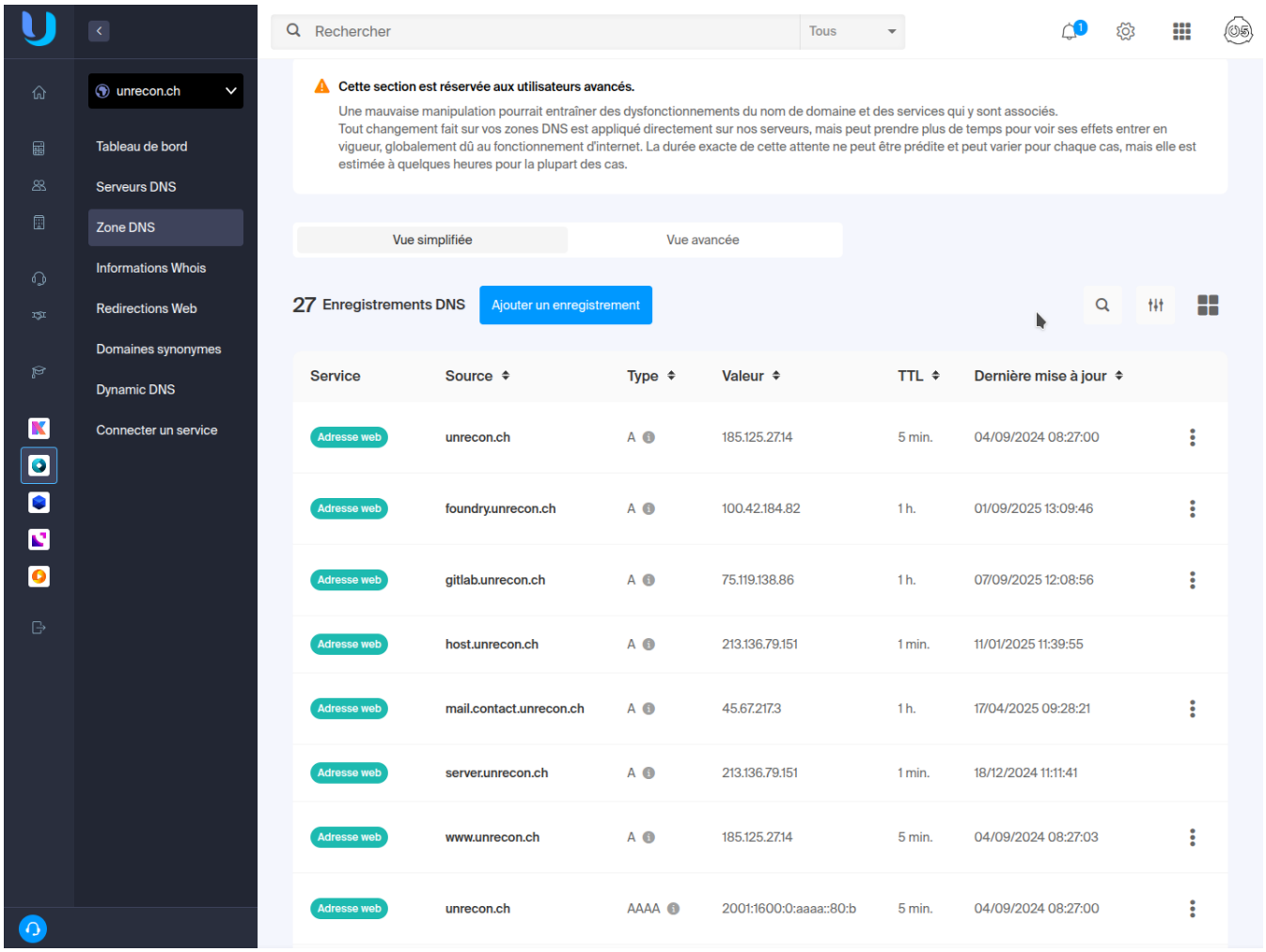

Figure 12 — Zone DNS d'Infomaniak

Cliquer sur `Ajouter un enregistrement`

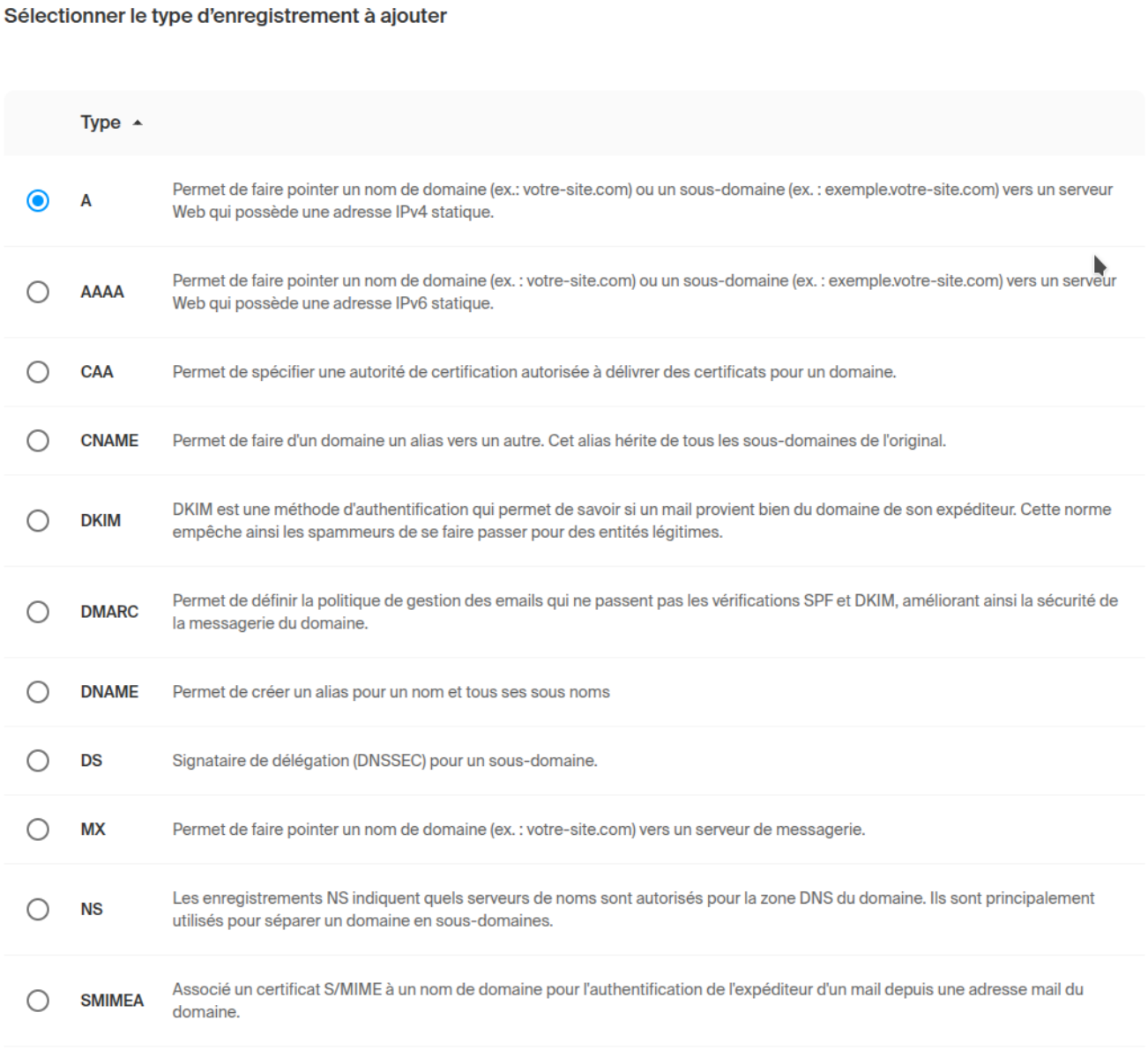

Figure 13 — Zone DNS d'Infomaniak pour l'ajout d'enregistrement

Dans notre cas, on veut un enregistrement de type **A**, il faut juste descendre et cliquer sur `continuer`.

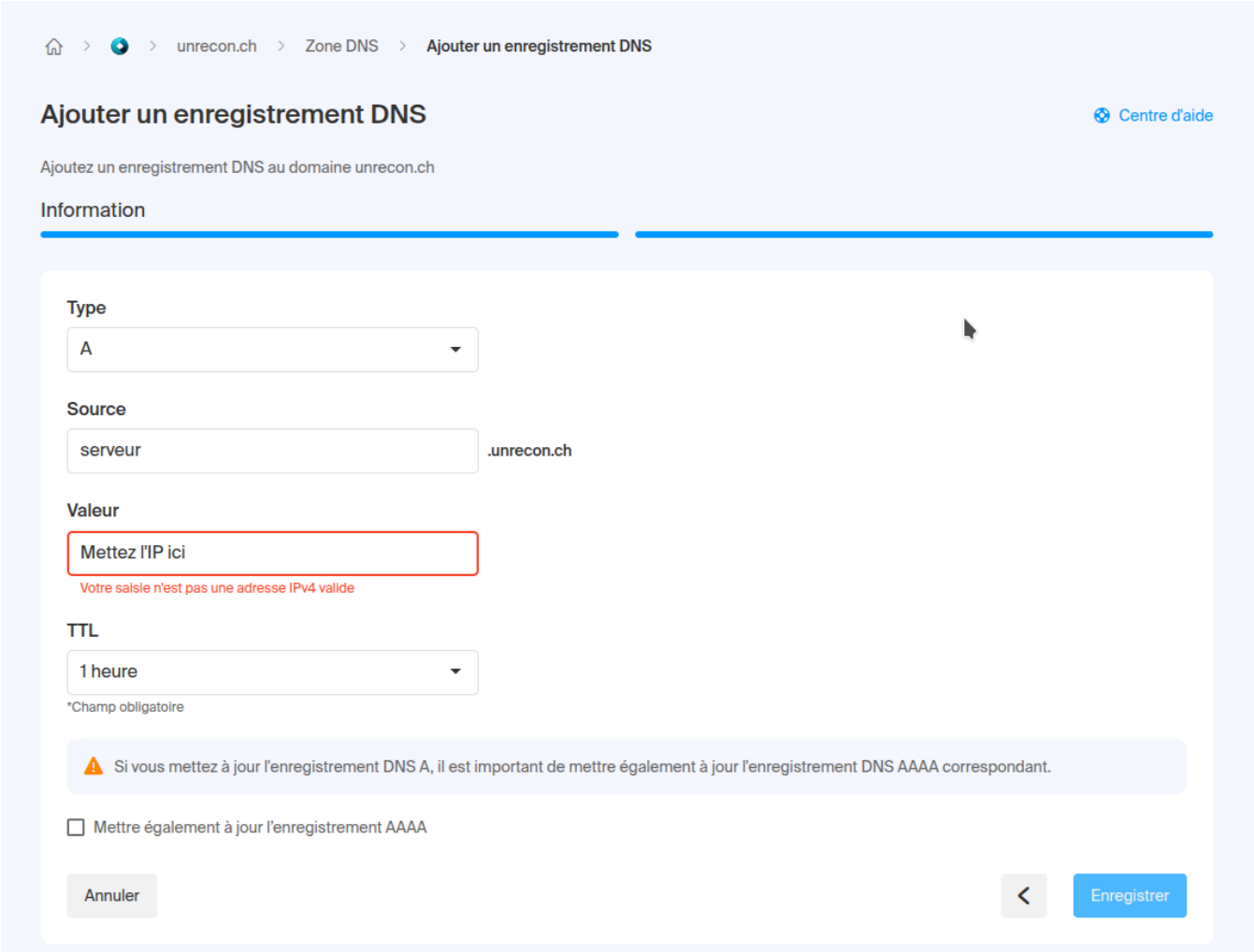

Figure 14 — Zone DNS d'Infomaniak pour l'ajout d'enregistrementd - formulaire

Une fois ici, il faut juste completer le sous nom de domaine que vous souhaitez utilisé, dans mon cas `serveur`, et l'adresse IP de votre serveur puis il ne vous reste plus qu'à cliquer sur `Enregistrer`.

IMPORTANT : Ne toucher pas au TTL à moins que vous sachiez ce que cela fait.

2.3 Installation de dépendance

Premièrement, il faut installer un autre gestionnaire de paquet pour installer la suite.

Il s'agit de `snap` et non pas snapchat, hin.

Bash

```
sudo apt update
sudo apt install snapd -y
```

Une fois ceci fait, il faut installer certbot

Bash

```
sudo snap install --classic certbot
```

Et il faut juste faire une dernière commande :

Bash

```
sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

2.4 Installation du certificat SSL/TLS

Il faut maintenant configurer le certificat SSL/TLS pour sécuriser votre site web.

IMPORTANT : Il est fortement recommandé de couper votre serveur web pendant ce processus pour éviter toute erreur.

Bash

```
sudo service apache2 stop
sudo service nginx stop
```

Maintenant, on peut faire la demande du certificat

Bash

```
sudo certbot certonly --standalone -d <votre-nom-de-domaine>
```

Figure 15 — ## Configuration

2.4.1 Apache2

Il vous faut, dans un premier temps, aller dans le dossier suivant :

Bash

```
cd /etc/apache2
```

Vous allez créer un fichier, par exemple : `website.conf` , un truc que vous comprendrez

Bash

```
sudo nano sites-available/website.conf
```

Vous allez insérer le contenu suivant :

HTML

```
<VirtualHost *:80>
    ServerName <domaine>

    RewriteEngine On
    RewriteCond %{HTTPS} !=on
    RewriteRule ^/?(.*) https://%{SERVER_NAME}/$1 [R,L]
</VirtualHost>
```

```
<VirtualHost *:443>
    ServerName <domaine>
    DocumentRoot "/var/www/html/<votre-projet>"

    AllowEncodedSlashes On

    php_value upload_max_filesize 100M
    php_value post_max_size 100M

    <Directory "/var/www/html/<votre-projet>">
        Require all granted
        AllowOverride all
    </Directory>

    SSLEngine on
    SSLCertificateFile /etc/letsencrypt/live/<domaine>/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/<domaine>/privkey.pem
</VirtualHost>
```

Remplacez `<votre-projet>` par le nom du dossier comprenant votre projet dans `/var/www/html` .

Remplacez aussi `<domaine>` par le nom de domaine que vous avez utilisé.

Faite la commande suivante :

Bash

```
sudo apache2ctl -t
```

Vous **DEVEZ** avoir la réponse : `syntax OK`

Vous devrez juste activer quelques mods via la commande suivante :

Bash

```
sudo a2enmod rewrite
sudo a2enmod ssl
```

Vous pouvez maintenant activé le site :

Bash

```
sudo a2ensite website.conf
```

Pour finir, il ne vous reste plus qu'à démarrer votre Apache2

Bash

```
sudo service apache2 start
```

2.4.2 Nginx

Il vous faut, dans un premier temps, aller dans le dossier suivant :

Bash

```
cd /etc/nginx
```

Vous allez créer un fichier, par exemple : `website.conf` , un truc que vous comprendrez

Bash

```
sudo nano sites-available/website.conf
```

Vous allez insérer le contenu suivant :

YAML

```
server {
    listen 80;
    server_name <domaine>;
    return 301 https://$server_name$request_uri;
}

server {
    # Replace the example <domain> with your domain name or IP address
    listen 443 ssl http2;
    server_name <domaine>;

    root /var/www/html/<votre-projet>;
    index index.php;

    access_log /var/log/nginx/website.app-access.log;
    error_log  /var/log/nginx/website.app-error.log error;

    client_max_body_size 100m;
    client_body_timeout 120s;

    sendfile off;
    ssl_certificate /etc/letsencrypt/live/<domaine>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<domaine>/privkey.pem;
    ssl_session_cache shared:SSL:10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384";
    ssl_prefer_server_ciphers on;

    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Robots-Tag none;
    add_header Content-Security-Policy "frame-ancestors 'self'";
    add_header X-Frame-Options DENY;
    add_header Referrer-Policy same-origin;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location ~ \.php$ {
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        fastcgi_pass unix:/run/php/php8.3-fpm.sock;
        fastcgi_index index.php;
        include fastcgi_params;
        fastcgi_param PHP_VALUE "upload_max_filesize = 100M \n post_max_size=100M";
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
```

```
        fastcgi_param HTTP_PROXY "";
        fastcgi_intercept_errors off;
        fastcgi_buffer_size 16k;
        fastcgi_buffers 4 16k;
        fastcgi_connect_timeout 300;
        fastcgi_send_timeout 300;
        fastcgi_read_timeout 300;
        include /etc/nginx/fastcgi_params;
    }

    location ~ /\.ht {
        deny all;
    }
}
```

Remplacez `<votre-projet>` par le nom du dossier comprenant votre projet dans `/var/www/html`.

Remplacez aussi `<domaine>` par le nom de domaine que vous avez utilisé.

Il faut maintenant activer le site en faisant la commande suivante :

Bash

```
sudo ln -s /etc/nginx/sites-available/website.conf /etc/nginx/sites-enabled/website.conf
```

Pour finir, il ne vous reste plus qu'à démarrer votre Nginx

Bash

```
sudo service nginx start
```

15 octobre 2025

3. GitLab

3.1 GitLab Runner

3.1.1 Description

Ce document à pour but d'expliquer comment déployer un GitLab Runner en containeur.

L'installation de GitLab ne sera pas expliqué ici, seul la partie GitLab Runner sera expliqué.

3.1.2 Installation et Configuration

Installation de Docker

Tutoriel provenant de la documentation officiel de Docker :

Installation des dépendances :

Bash

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo
apt-get remove $pkg; done
```

Ajouter le dépot de Docker à apt :

Bash

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://
download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

Installation de Docker :

Bash

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Lancement du containeur

On va créer notre containeur GitLab Runner

Bash

```
docker run -d --name gitlab-runner --restart always \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -v gitlab-runner-config:/etc/gitlab-runner \
  gitlab/gitlab-runner:latest
```

Puis on va l'enregistrer dans notre GitLab

Bash

```
docker run --rm -it -v gitlab-runner-config:/etc/gitlab-runner gitlab/gitlab-runner register
```

Vous devez aller dans vos paramètres CI/CD, ajouter un Runner Linux et vous n'avez qu'à mettre la clé, sélectionner la méthode d'éxecution Shell et donner un nom à votre Runner.

Configuration de containeur

On va rentrer dans le containeur et on va installer ce qu'il nous faut

Bash

```
sudo docker exec -it gitlab-runner /bin/bash
```

CONFIGURATION POUR PYTHON

Pour installer les éléments nécessaires à des projets Python, voici la commande :

Bash

```
apt-get update && apt-get install -y python3 python3-pip python3-venv
```

CONFIGURATION POUR NODEJS

Pour installer les éléments nécessaires à des projets NodeJS, voici la commande :

Installation des dépendances de NVM

Bash

```
apt update && apt install curl -y
```

Installation de NVM :

Bash

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.3/install.sh | bash
```

Dernière configuration :

Bash

```
export NVM_DIR="$([ -z "${XDG_CONFIG_HOME-}" ] && printf %s "${HOME}/.nvm" || printf %s "${XDG_CONFIG_HO
ME}/nvm")"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
```

3.1.3 En cas d'erreur

Erreur 1

Si vous avez une erreur qui ressemble un peu près à ceci :

Bash

```
OSError: cannot load library 'libgobject-2.0-0': libgobject-2.0-0: cannot open shared object file: No su
ch file or directory.  Additionally, ctypes.util.find_library() did not manage to locate a library calle
d 'libgobject-2.0-0'
```

Faite cette commande dans votre containeur :

Bash

```
apt-get install -y libglib2.0-0 libpangocairo-1.0-0
```

15 octobre 2025

4. Base de données

4.1 Récupérer des données

4.1.1 Commande de base

SQL

```sql
SELECT * FROM `Charts`
```

Cette commande permet de récupérer toutes les données d'une table, en l'occurence, `Charts` ici.

Voici le résultat que vous devriez obtenir avec les données de test :

id	title	content	created_at	id_user
1	test	content-test	2025-10-09	1
2	mermaid_db	content-mermaid_db	2025-10-09	1
3	mermaid_api	content-mermaid_api	2025-10-09	1
4	mermaid_infra	content-mermaid_infra	2025-10-09	1
5	mermaid_call	content-mermaid_call	2025-10-09	1
6	reseau maison	content-reseau_maison	2025-10-13	2
7	nas-infra	content-nas-infra	2025-10-13	2
8	planif-travaux	content-planif-travaux	2025-10-13	2
9	projet-git-commit	content-projet-git-commit	2025-10-26	4
10	projet-gantt	content-projet-gantt	2025-10-26	4
11	projet-diagram-classe	content-projet-diagram-classe	2025-10-26	4
12	projet-mcd	content-projet-mcd	2025-10-26	4
13	infra_projet	content-infra_projet	2025-10-30	5
14	merge	content-merge	2025-11-01	7

4.1.2 Récupérer une partie de la table

On veut récupérer tout de la table `Users` faut le mot de passe pour des raisons de sécurité. Pour se faire, on va éxecuter la commande suivante :

SQL

```sql
SELECT `id`, `email`, `code`, `created_at` FROM `Users`;
```

Voici les données que vous devriez avoir :

id	email	code	created_at
1	james@benone.ch		2025-10-01 15:04:56.000
2	martin.lavalais@ikmail.com		2025-10-03 19:45:12.000
3	michel@example.com	194012	2025-10-06 11:50:03.000
4	elton-john@yahoo.dev		2025-10-17 09:03:12.000
5	sasukedu92@gmail.com		2025-10-23 19:45:54.000
6	johnwick@gmail.com	847102	2025-10-30 05:23:23.000
7	sogo.sogot@gmail.com		2025-11-02 17:30:34.000

4.1.3 Récupérer une donnée spécifique

Supposons qu'on veut récupérer la liste des charts de l'utilisateur `1`.

On va faire la commande suivante :

SQL

```sql
SELECT * FROM `Charts` WHERE `id_user` = 1;
```

Voici les données que vous devriez avoir :

id	title	content	created_at	id_user
1	test	content-test	2025-10-09	1
2	mermaid_db	content-mermaid_db	2025-10-09	1
3	mermaid_api	content-mermaid_api	2025-10-09	1
4	mermaid_infra	content-mermaid_infra	2025-10-09	1
5	mermaid_call	content-mermaid_call	2025-10-09	1

4.1.4 Récupérer avant une date

Supposons que l'on veut récupérer tous les utilisateurs qui ont créer un compte avant novembre.

On peut récupérer cette donnée via la commande suivante :

SQL

```sql
SELECT `id`, `email`, `code`, `created_at` FROM `Users` WHERE `created_at` < '2025-11-01';
```

Voici les données récupérés :

id	email	code	created_at
1	james@benone.ch		2025-10-01 15:04:56.000
2	martin.lavalais@ikmail.com		2025-10-03 19:45:12.000
3	michel@example.com	194012	2025-10-06 11:50:03.000
4	elton-john@yahoo.dev		2025-10-17 09:03:12.000
5	sasukedu92@gmail.com		2025-10-23 19:45:54.000
6	johnwick@gmail.com	847102	2025-10-30 05:23:23.000

4.1.5 Récupérer des champs vides

On veut récupérer tous les utilisateurs qui on un code.

La commande est la suivante :

SQL

```sql
SELECT `id`, `email`, `code`, `created_at` FROM `Users` WHERE `code` IS NOT NULL;
```

Voici les données récupérés

id	email	code	created_at
3	michel@example.com	194012	2025-10-06 11:50:03.000
6	johnwick@gmail.com	847102	2025-10-30 05:23:23.000

4.1.6 Récuperer le nombre de colonne

On veut récupérer le nombre de `Charts` par `Users`. Pour cela, on va utiliser la fonction `count()`.

Voici un exemple :

SQL

```sql
SELECT `id_user`, count(`id`) FROM `Charts`;
```

Voici les données que vous devriez avoir :

id_user	count("id")
1	14

On a eu le nombre de `Charts`, mais pas comme on voulait.

Ici il y a 2 problèmes.

Dans un premier temps, on n'a pas le nombre **PAR** utilisateur et c'est marqué `count("id")`.

Mais je vous rassure, il y a moyen d'y remédier !

4.1.7 Récupérer des données groupés

Reprenons la commande d'avant mais rajoutons un petit truc et regardons ce que cela donne :

SQL

```sql
SELECT `id_user`, count(`id`) FROM `Charts` GROUP BY `id_user`;
```

Voici les données récupérés :

id_user	count("id")
1	5
2	3
4	4
5	1
7	1

On a déjà rêgler un problème, mais il en reste un...

4.1.8 Mettre des alias

Il est possible de mettre des alias pour les colonnes mais aussi pour les tables.

Voici un exemple avec les 2 :

SQL

```sql
SELECT c.id_user, count(c.id) AS nb_charts FROM `Charts` AS c GROUP BY c.id_user;
```

Voici les données récupérés :

id_user	nb_charts
1	5
2	3
4	4
5	1
7	1

4.1.9 Trier les données

Supposons que maintenant, on veut trier en fonction de celui qui a créé le plus de `Charts`

À savoir : Il y a 2 types de tri, `DESC` et `ASC`. Si vous ne mettez rien, par défaut, c'est `ASC`. `ASC` tri par ordre croissant alors que `DESC` tri en décroissant.

On peut via la commande suivante :

SQL

```
SELECT c.id_user, count(c.id) AS "nb_charts" FROM `Charts` AS c GROUP BY c.id_user ORDER BY nb_charts DE
SC;
```

Voici les données récupérés :

id_user	nb_charts
1	5
4	4
2	3
5	1
7	1

4.1.10 Limiter

On garde la même requête qu'avant mais on veut limiter le nombre de retour à, par exemple, 3.

On peut via la commande suivante :

SQL

```
SELECT c.id_user, count(c.id) AS "nb_charts" FROM `Charts` AS c GROUP BY c.id_user ORDER BY nb_charts DE
SC LIMIT 3;
```

Voici les données récupérés :

id_user	nb_charts
1	5
4	4
2	3

4.1.11 Récupérer 2 tables

On veut récupérer les 2 tables liés avec les champs suivants : `id`, `email` `created_at`, `nb_charts`,

Pour se faire, on peut le faire via la commande suivante :

SQL

```
SELECT u.id, u.email, u.created_at, count(c.id) AS "nb_charts" FROM `Users` AS u
JOIN `Charts` AS c ON c.id_user = u.id
GROUP BY u.id
ORDER BY nb_charts DESC;
```

Voici les données récupérés :

id	email	created_at	nb_charts
1	james@benone.ch	2025-10-01 15:04:56.000	5
4	elton-john@yahoo.dev	2025-10-17 09:03:12.000	4
2	martin.lavalais@ikmail.com	2025-10-03 19:45:12.000	3
5	sasukedu92@gmail.com	2025-10-23 19:45:54.000	1
7	sogo.sogot@gmail.com	2025-11-02 17:30:34.000	1

Mais il y a un problème, on n'a pas tous les utilisateurs.

Il est tout de même possible de les afficher avec la commande suivante :

SQL

```sql
SELECT u.id, u.email, u.created_at, count(c.id) AS "nb_charts" FROM `Users` AS u
LEFT JOIN `Charts` AS c ON c.id_user = u.id
GROUP BY u.id
ORDER BY nb_charts DESC;
```

Voici les données récupérés :

id	email	created_at	nb_charts
1	james@benone.ch	2025-10-01 15:04:56.000	5
4	elton-john@yahoo.dev	2025-10-17 09:03:12.000	4
2	martin.lavalais@ikmail.com	2025-10-03 19:45:12.000	3
5	sasukedu92@gmail.com	2025-10-23 19:45:54.000	1
7	sogo.sogot@gmail.com	2025-11-02 17:30:34.000	1
3	michel@example.com	2025-10-06 11:50:03.000	0
6	johnwick@gmail.com	2025-10-30 05:23:23.000	0

Il faut savoir qu'il existe plusieurs types de `JOIN`. `LEFT`, `RIGHT`, `INNER`, etc...

Par défaut, si vous ne mettez que `JOIN`, il l'exécutera comme un `INNER JOIN`.

5 novembre 2025

4.2 Gérer des données

4.2.1 INSERT

Simple

On peut insérer les données simplement.

Pour la table utilisateur, Comme ceci :

SQL

```sql
INSERT INTO `Users` VALUES (8, "pascal@gmail.com", "Super", null, now());
```

On a créer un utilisateur avec l'identifiant `8`, l'email `pascal@gmail.com`, le mot de passe `Super`, aucun code et la date de création à maintenant.

Colonnes précisés

Si on ajoute un utilisateur à la base de données, très souvent, on ne va pas donner l'identifiant, on laisse la base de données faire.

Pour cela, on peut via la commande suivante :

SQL

```sql
INSERT INTO `Users`(`email`, `password`, `code`, `created_at`) VALUES ("kevin@gmail.com", "Super",
null, now());
```

On a créer un utilisateur avec l'identifiant généré par la base de données, l'email `kevin@gmail.com`, le mot de passe `Super`, aucun code et la date de création à maintenant.

4.2.2 UPDATE

Pour modifier une donnée, on peut le faire via la commande suivante :

SQL

```sql
UPDATE `Users` SET `code` = null WHERE `id` = 3;
```

On a modifié l'utilisateur avec l'identifiant `3` et on lui a dit qu'il n'a maintenant plus de code.

4.2.3 DELETE

Pour supprimer une donnée, on peut le faire via la commande suivante :

SQL

```sql
DELETE FROM `Users` WHERE `id` = 8;
```

On a supprimé l'utilisateur avec l'identifiant `8`. Il n'apparaîtra donc plus dans la base de données.

5 novembre 2025

4.3 Gérer une base de données

4.3.1 Créer une base de données

SQL

```sql
CREATE DATABASE test_db;
```

Ceci créer une base de données nommée `test_db`.

Mais il y a un problème, supposons qu'elle existe, cela va nous créer une erreur. Nous pouvons toute fois y remédier avec un petit ajout :

SQL

```sql
CREATE DATABASE IF NOT EXISTS test_db;
```

Il va alors créer la base de données **UNIQUEMENT** si aucune base de données à ce nom.

4.3.2 Supprimer une base de données

SQL

```sql
DROP DATABASE test_db;
```

Ceci supprime une base de données nommée `test_db`.

Mais même cas, si cela n'existe pas, il y aura une erreur et comme pour avant, il y a :

SQL

```sql
DROP DATABASE IF EXISTS test_db;
```

Il va alors supprimer la base de données **UNIQUEMENT** si il y a une base de données à ce nom.

4.3.3 Créer une table

Supposons que nous avons créer la base de données `test_db`.

Nous voulons rajouter une table simple qui contient `id`, `email`, `password`, `code` et `created_at`. Rien de plus, rien de moins. On va l'appeler `Users`.

Il faut savoir plusieurs choses dans ce cas, il faut penser aux types des variables (int, varchar, date, datetime, etc...) et il faut toujours une clé primaire par table. La clé primaire est un identifiant unique d'une colonne. Dans notre cas, sa sera `id` et on va l'auto-incrémenté (augmente à chaque ajout d'une ligne).

Voici la commande pour créer une base de données :

SQL

```sql
CREATE TABLE test_db.Users (
    id int(11) auto_increment NOT NULL,
    email varchar(256) NOT NULL,
    password varchar(128) NOT NULL,
    code varchar(10) NULL DEFAULT NULL,
    created_at TIMESTAMP DEFAULT now() NOT NULL,
    CONSTRAINT Users_PK PRIMARY KEY (id),
    CONSTRAINT Users_email_UNIQUE UNIQUE KEY (email),
    CONSTRAINT Users_code_UNIQUE UNIQUE KEY (code)
)
ENGINE=InnoDB
DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_general_ci;
```

Il y a quelques particularités que je vais expliqué ici.

Déjà, vous avez remarqué qu'à quasiment toutes les lignes, il y a `NULL` ou `NOT NULL`. Il faut généralement en mettre à chaque ligne et cela détermine si la variable peut-être null ou doit avoir un contenu.

À la ligne `created_at`, vous voyez `DEFAULT now()`. Cela veut dire que la valeur par défaut de cette variable sera la date d'ajout de la ligne. Il y a aussi `DEFAULT NULL`, cela veut juste dire que par défaut, la valeur est vide, qu'elle ne contient rien.

Il y a aussi `CONSTRAINT Users_PK PRIMARY KEY (id)` qui définit la clé primaire.

Il y a aussi `CONSTRAINT Users_email_UNIQUE UNIQUE KEY (email)` qui définit que cette valeur doit être unique et ne peut pas être en doublon dans la table. **À SAVOIR**, si une colonne est unique et nullable, null est la seule "valeur" qui peut-être attribué à plusieurs endroits car il signifie qu'il n'y a rien.

`ENGINE=InnoDB` définit le moteur qui gère la table.

Les lignes ci-dessous définissent la table de caractère utilisé pour cette table. Elle peut varié en fonction des besoins linguistiques.

SQL

```sql
DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_general_ci`
```

4.3.4 Supprimer une table

Comme pour les bases de données, on peut mettre un `IF EXISTS`.

SQL

```sql
DROP TABLE IF EXISTS test_db.Users;
```

Cela supprime la table `Users` ainsi que toutes ces données.

5 novembre 2025

4.4 Gérer des utilisateurs

4.4.1 Créer un utilisateur

IMPORTANT : Si vous voulez créer un utilisateur avec des paramètres IP spécifiques, je vous recommande fortement de vous renseigner sur le fonctionnement des IP et des tables IP.

Créer un utilisateur local

Supposons que l'on veut créer un utilisateur `admin` et qu'il ne soit accessible qu'en local. On peut via la commande suivante :

SQL

```sql
CREATE USER "admin"@"localhost" IDENTIFIED BY "Super";
```

Cela créer l'utilisateur `admin` avec le mot de passe `Super` qui ne peut-être accéder qu'en local ou avec l'IP `127.0.0.1`.

Créer un utilisateur public

Supposons que vous avez une équipe de sécurité qui est externe mais qui doit avec accès à une table de type `Logs`.

On peut via la commande suivante :

SQL

```sql
CREATE USER "cyber-team"@"%" IDENTIFIED BY "Super";
```

Cela créer l'utilisateur `cyber-team` avec le mot de passe `Super` qui peut-être accéder de n'importe ou.

Créer un utilisateur interne

Supposons que vous avez une équipe qui gère les données manuellements et qui à besoin des accès, vous pouvez via la commande suivante :

SQL

```sql
CREATE USER "bi-dep"@"10.5.40.0/24" IDENTIFIED BY "Super";
```

Cela créer l'utilisateur `bi-dep` avec le mot de passe `Super` qui peut-être accéder par tous les ordinateurs ayant une IP qui commence par `10.5.40`.

4.4.2 Supprimer un utilisateur

Supposons que l'équipe de sécurité ne travail plus pour vous et vous devez supprimer l'utilisateur.

Vous pouvez via la commande suivante :

SQL

```sql
DROP USER "cyber-team"@"%";
```

Cela supprime l'utilisateur `cyber-team`.

4.4.3 Gérer les droits

Ajouter des droits

Supposons qu'on veut attribuer les droits de gérer les données montrés dans Gérer les données et Récupérer les données.

On peut via la commande suivante :

SQL

```sql
GRANT SELECT, INSERT, UPDATE, DELETE ON test_db.* TO "bi-dep"@"10.5.40.0/24";
```

Cela donne les droits de récupérer, ajouter, changer ou supprimer les données de toutes les tables de la base de données `test_db`.

Ajouter tous les droits

ATTENTION : Il est important de ne jamais faire ceci à un utilisateur publiquement accéssible.

Supposons que l'on veut donner tous les droits à notre administrateur.

Il faut pour cela, éxécuter la commande suivante :

SQL

```sql
GRANT ALL PRIVILEGES ON *.* TO "admin"@"localhost" WITH GRANT OPTION;
```

Cela donne tous les droits sur toutes les tables de toutes les bases de données. et `WITH GRANT OPTION` donne également le droit de gérer les autres utilisateurs.

Supprimer des droits

Supposons que l'on veut empécher au `bi-dep` de modifier ou supprimer les données de la table `Logs` .

On peut via la commande suivante :

SQL

```sql
REVOKE UPDATE, DELETE ON test_db.Logs TO "bi-dep"@"10.5.40.0/24";
```

Actualiser les droits

Pour actualiser les droits et les appliqués, il faut dire au logiciel qui gère les bases de données de mettre les permissions à jour via la commande suivante :

SQL

```sql
FLUSH PRIVILEGES;
```

5 novembre 2025

5. DNS

5.1 Définition

DNS ou Domain Name System (Système de nom de domaine) permet à un utilisateur de pouvoir visiter un site via un nom qui sert d'alias à une adresse IP, vous simplifiant la vie lorsque vous devez vous rendre sur plusieurs sites web et surtout pour vos grand-parents.

Un DNS en tant qu'administrateur système, vous permet de gérer vos domaines et sous-domaines lors de la configuration de site web, de serveur mail, de vpn et encore plein d'autres.

5.2 Historique

5.2.1 ARPANET

À l'époque d'ARPANET, la Stanford Research Institute (de nos jours, SRI International) utilisait le fichier `hosts.txt` pour définir une adresse mais cela devait se faire sur tous les ordinateurs et si une donnée devait être ajouté au fichier, il fallait modifier tous les ordinateurs.

Elizabeth Feinler a alors eu une idée, développé un outil nommé **Assigned Number List** qui était gérer par Jon Pastel à l'Information Science Institute de l'Université de Californie du Sud.

5.2.2 Le premier système WHOIS

Avant la mise en place du premier système WHOIS, les adresses étaient attribuées manuellement et on devait appeler le SRI Network Information Center ou SRI-NIC (qui était géré par Mme Feinler).

Un peu plus tard, Mme Feinler et son équipe on décidé de créer le premier système WHOIS pour la récupération d'informations sur les ressources, les contacts et les entités. Ils ont également développé le concept de nom de domaine et Mme Feinler à suggérer que cela soit basé sur l'adresse physique de l'ordinateur.

5.2.3 Dans les années 80

Paul Mockapetris, étant à l'Université de Californie du Sud, a créé un système de nom de domaine. L'équipe en charge des normes RFC a alors introduit la norme RFC 882 et 883 en novembre 1983 puis RFC 973 en janvier 1986.

5.2.4 Création du BIND

En 1984, quatre étudiants de l'université de Berkeley, Douglas Terry, Mark Painter, David Riggle et Songnian Zhou, ont écrit la première implémentation du serveur de noms Unix pour le Berkeley Internet Name Domain, communément appelé BIND.

Les versions 4.9.3 et suivantes de BIND ont été développées et maintenues par l'ISC, avec le soutien des sponsors de l'ISC. En tant que co-architectes/programmeurs, Bob Halley et Paul Vixie ont publié la première version prête à la production de BIND version 8 en mai 1997.

5.3 Fonctionnement

5.3.1 En tant qu'utilisateur

Imaginons que vous souhaitez aller sur youtube pour regarder une vidéo.

Vous allez dans votre navigateur et vous tapez `www.youtube.com`. Que se passe-t-il derrière ? Voici un schéma **TRÈS SIMPLIFIÉ**

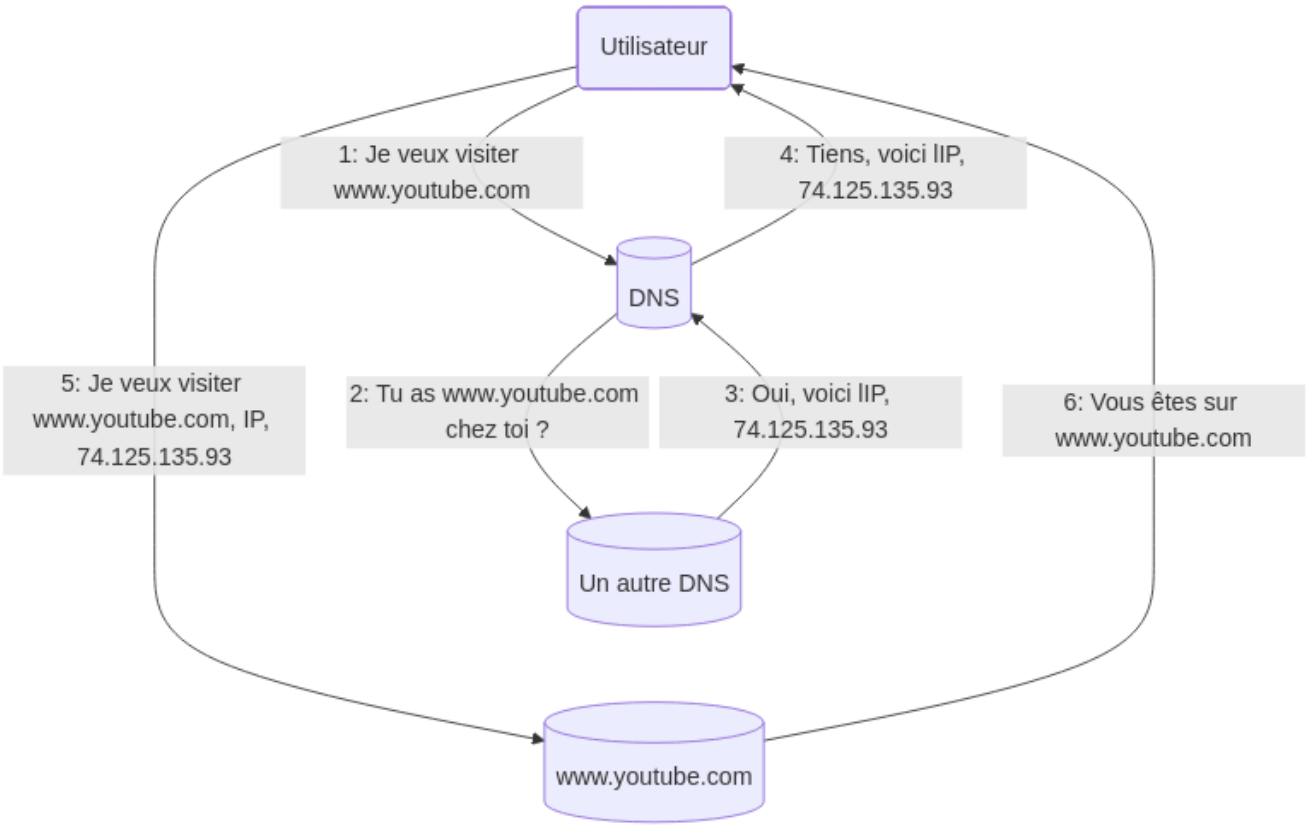

Figure 17 — Exemple du fonctionnement d'un DNS au niveau simple

1. Vous allez demandé qu'elle l'IP du serveur qui a www.youtube.com a un serveur DNS, généralement, celui de votre FAI (Fournisseur d'Accès à Internet).

Si le serveur DNS de votre FAI n'a pas `www.youtube.com` chez lui, il va demandé à un autre serveur DNS pour l'avoir.

1. Il demande à un autre serveur DNS si il a `www.youtube.com`

Il se trouve comme par chance, l'autre serveur DNS à `www.youtube.com`

1. Il renvoie donc l'IP de www.youtube.com

Le serveur DNS de votre FAI s'empresse alors de vous renvoyez la bonne nouvelle.

1. Et il le fait donc

Vous avez maintenant l'IP associé au nom de domaine.

1. Vous demandez donc au serveur ayant l'adresse IP pour avoir `www.youtube.com`
2. Il vous renvoie `www.youtube.com`

Vous pouvez maintenant regarder des vidéos youtube.

Il est possible de choisir un serveur DNS que vous pouvez définir sur votre routeur pour des raisons de performences, de sécurité ou autres.

5.3.2 En tant qu'administrateur

En tant qu'administrateur de votre serveur DNS, vous avez beaucoup de chose qui s'offre à vous dans la configuration.

- Vous pouvez ajouter **un enregistrement** à votre serveur DNS de type A
- Sur cette enregistrement, vous mettez un **TTL** de 1 jour.
- Ou alors, vous voulez rediriger un sous-domaine à vous vers votre NAS héberger chez vous mais l'IP est dynamique ? Utilisez un **DynDNS**.
- Vous souhaitez obtenir des informations sur un nom de domaine ? Utilisez un **outil**

Tous ces points, on va y revenir.

5.3.3 De manière global

Dans le monde, il y a 13 serveurs DNS majeurs dit **RootDNS** qui pour résumé, sont les serveurs DNS principaux dans le monde.

Mais pas tous les serveurs DNS s'y réfère, cela dépend de si ils sont **public** ou **privé**.

Et aussi si il est coté **client** ou coté **serveur**.

Également, il y a des serveurs DNS dit **esclave** qui se réfère au **maître de zone**.

Il ne faut pas oublier les **TLD** qui gère certain domaine.

Mais tous cela, on y reviendra plus tard.

5.4 Cache et TTL

Lorsqu'un serveur DNS demande une valeur, pour ne pas avoir à redemander la valeur, il le met dans le cache. Cela permet d'accélérer le temps de réponse et de ne pas avoir à redemander au serveur DNS. Le temps qu'il passera en cache dépend du TTL.

Le TTL ou Time To Live définit le temps qu'une valeur passera dans le cache. Après ce temps, il faut redemander au serveur DNS.

5.5 Les enregistrements

Un serveur DNS contient plusieurs types d'enregistrement listés ci-dessous.

5.5.1 Liste des types de registre

Type	RFC	Description
A	1035	Permet de pointer un nom de domaine ou sous-domaine à une adresse IPv4.
AAAA	3596	Permet de pointer un nom de domaine ou sous-domaine à une adresse IPv6.
CAA	8659	Permet de spécifier une autorité de certification autorisée à délivrer des certificats pour un domaine.
CNAME	1035	Permet de faire d'un domaine un alias vers un autre. Cet alias hérite de tous les sous-domaines de l'original.
DKIM	6376	est une méthode d'authentification qui permet de savoir si un mail provient bien du domaine de son expéditeur. Cette norme empêche ainsi les spammeurs de se faire passer pour des entités légitimes.
DMARC	7489	Permet de définir la politique de gestion des emails qui ne passent pas les vérifications SPF et DKIM, améliorant ainsi la sécurité de la messagerie du domaine.
DNAME	6672	Permet de créer un alias pour un nom et tous ses sous noms.
DS	4034	Signataire de délégation (DNSSEC) pour un sous-domaine.
MX	1035	Permet de faire pointer un nom de domaine (ex. : votre-site.com) vers un serveur de messagerie.
NS	1035	Les enregistrements NS indiquent quels serveurs de noms sont autorisés pour la zone DNS du domaine. Ils sont principalement utilisés pour séparer un domaine en sous-domaines.
SMIMEA	8162	Associé un certificat S/MIME à un nom de domaine pour l'authentification de l'expéditeur d'un mail depuis une adresse mail du domaine.
SRV	2782	Permet d'indiquer quels sont les services disponibles pour un domaine. Ils sont souvent utilisés pour les protocoles XMPP, LDAP ou pour configurer Microsoft Office 365.
SSHFP	4255	Permet d'enregistrer l'empreinte d'une clé ssh publique dans la zone du domaine afin d'identifier et sécuriser vos connexions ssh.
TLSA	6698	Permet d'enregistrer l'empreinte d'un certificat TLS ou SSL dans la zone du domaine. Il est souvent utilisé pour DANE.
TXT	1035	Permet d'insérer un texte quelconque dans un enregistrement DNS.

Chaque enregistrement fonctionnent de la manière suivante :

Type, Source, Valeur, TTL

Si on prend l'exemple de youtube, sa donnerai sa :

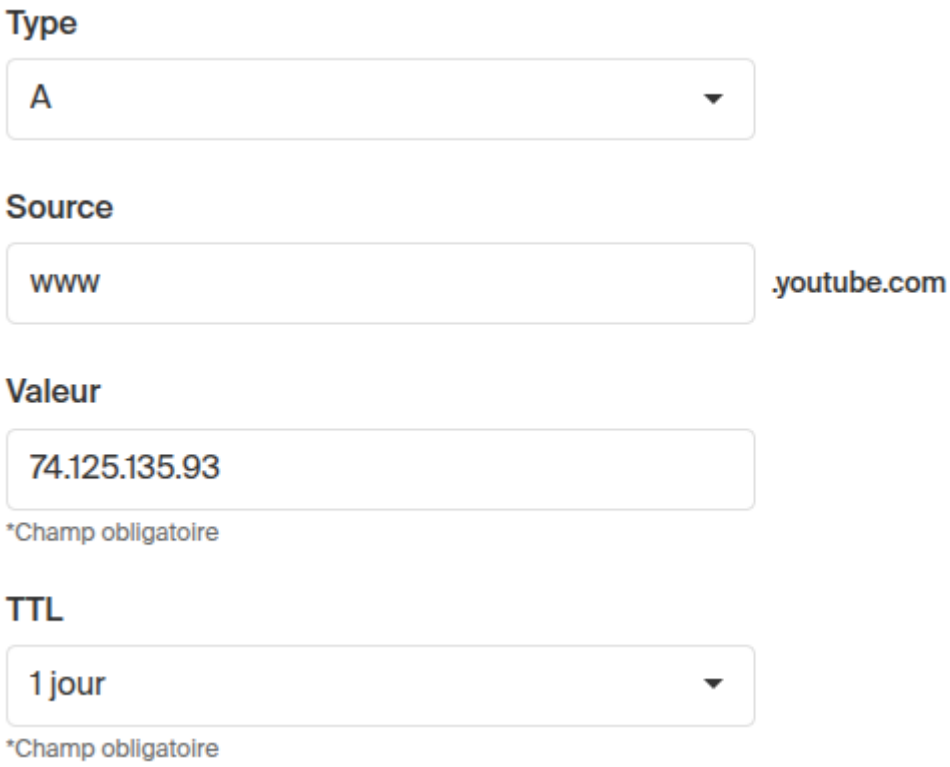

Figure 18 — Panel Infomaniak, DNS avec exemple de youtube

5.5.2 Exemple de contenu dans un serveur DNS

Text Only

```
; Domain: unrecon.ch
; Exported (y-m-d hh:mm:ss): 2025-10-14 14:24:45
; Actual version

$TTL 3600
@                          IN SOA   nsany1.infomaniak.com. hostmaster.infomaniak.ch. (2025100826 10800
3600 605800 3600)
*.gitlab               3600 IN A     75.119.138.86
                       300  IN A     185.125.27.14
                       300  IN AAAA  2001:1600:0:aaaa::80:b
                       3600 IN MX 5  mta-gw.infomaniak.ch.
                       3600 IN NS    nsany1.infomaniak.com.
                       3600 IN NS    nsany2.infomaniak.com.
                       3600 IN TXT   "v=spf1 include:spf.infomaniak.ch -all"
autoconfig             3600 IN CNAME infomaniak.com.
autodiscover           3600 IN CNAME infomaniak.com.
bot                    3600 IN CNAME host.unrecon.ch:3030.
contact                3600 IN MX 10 mail.contact.unrecon.ch.
dkim._domainkey.mail   3600 IN CNAME dkim._domainkey.alias.proton.me.
dkim02._domainkey.mail 3600 IN CNAME dkim02._domainkey.alias.proton.me.
dkim03._domainkey.mail 3600 IN CNAME dkim03._domainkey.alias.proton.me.
foundry                3600 IN A     100.42.184.82
gitlab                 3600 IN A     75.119.138.86
host                   60   IN A     213.136.79.151
mail                   3600 IN MX 10 mx1.alias.proton.me.
mail                   3600 IN MX 20 mx2.alias.proton.me.
mail                   3600 IN TXT   "pm-verification=oxkllbawdtxinqklihhehijapmdtaq"
mail                   3600 IN TXT   "v=spf1 include:alias.proton.me ~all"
mail.contact           3600 IN A     45.67.217.3
mermaid                3600 IN A     213.136.79.151
server                 60   IN A     213.136.79.151
www                    300  IN A     185.125.27.14
www                    300  IN AAAA  2001:1600:0:aaaa::80:b
_dmarc                 3600 IN TXT   "v=DMARC1; p=reject; pct=100;"
_dmarc.mail            3600 IN TXT   "v=DMARC1; p=quarantine; pct=100; adkim=s; aspf=s"
_domainkey             3600 IN NS    nsany1.infomaniak.com.
_domainkey             3600 IN NS    nsany2.infomaniak.com.
```

5.6 Dynamic DNS

un Dynamic DNS ou DynDNS vous permet de pointer un domaine ou sous-domaine à vous vers une adresse IP dynamique.

Exemple : Imaginez que vous héberger une instance de NextCloud et vous voulez qu'il soit accessible par des membres de votre famille et des amis facilement. Vous pouvez configurer un DynDNS et même si l'IP change, vous n'aurez pas à le découvrir et le changer manuellement.

5.6.1 Mise en place

Création de l'utilisateur

Il faut avoir les droits administrateurs sur votre serveur DNS.

Dans un premier temps, rendez-vous sur le panel ou vous avez votre nom de domaine. Allez ensuite dans la zone `Dynamic DNS`.

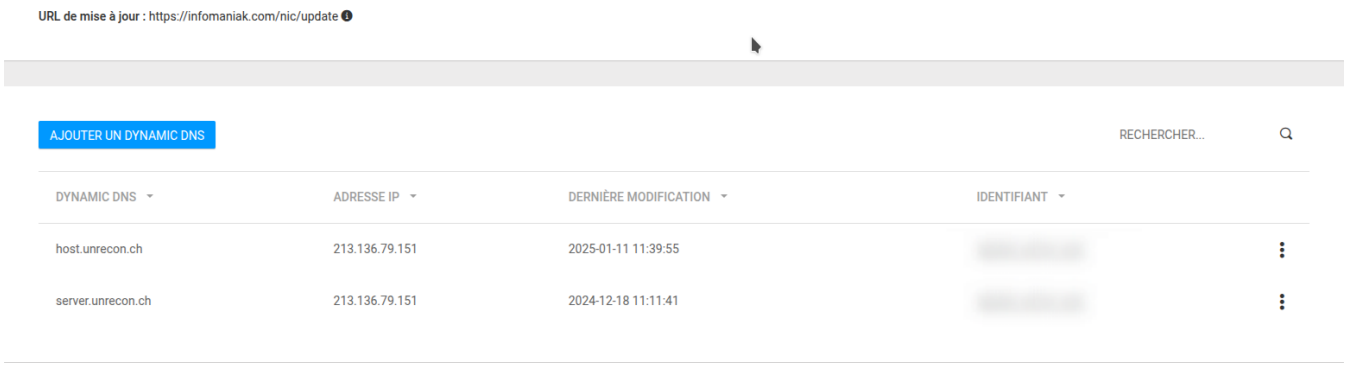

Figure 19 — Panel Infomaniak, Zone des DynDNS

Vous allez dans : `AJOUTER UN DYNAMIC DNS`.

Vous devriez avoir un formulaire qui s'est affiché :

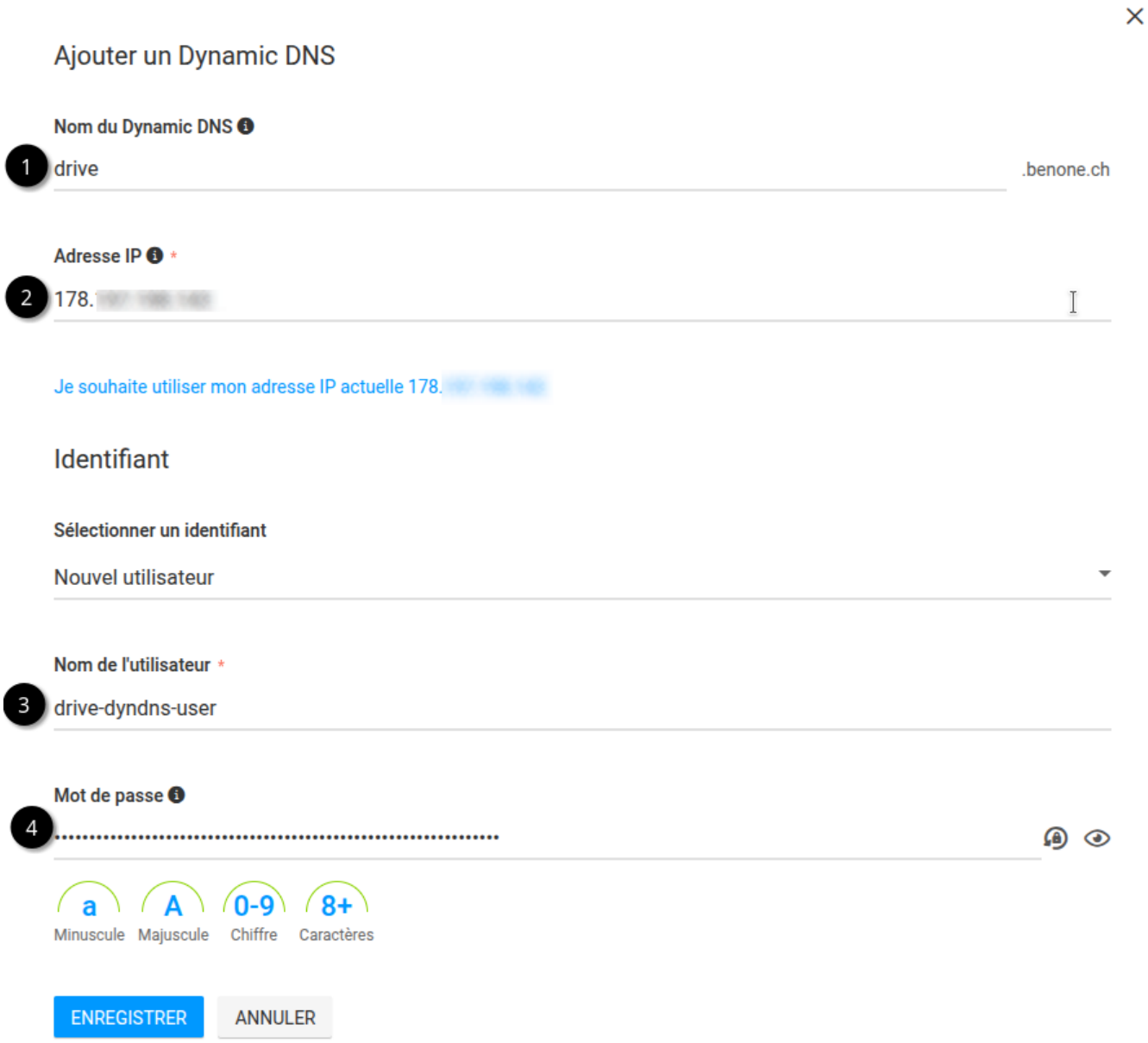

Figure 20 — Panel Infomaniak, Ajouter un DynDNS

1. Laisser vide pour le domaine ou mettez le sous-domaine concerné (dans mon exemple: drive)
2. L'adresse IPv4 public de votre NAS
3. Le nom de l'identifiant pour le DynDNS
4. Le mot de passe

À FAIRE | Configuration de votre routeur

À FAIRE | Configuration sur Linux

Dans un premier temps, on va installer le logiciel pour la gestion du DynDNS sur Linux via la commande suivante :

Bash

```
sudo apt install ddclient -y
```

Configuration sur un NAS

Sur votre NAS, vous devez vous connectez avec un compte administrateur.

Ensuite, aller dans le Panneau de configuration, puis Accès externe et enfin dans DDNS.

Voici une image :

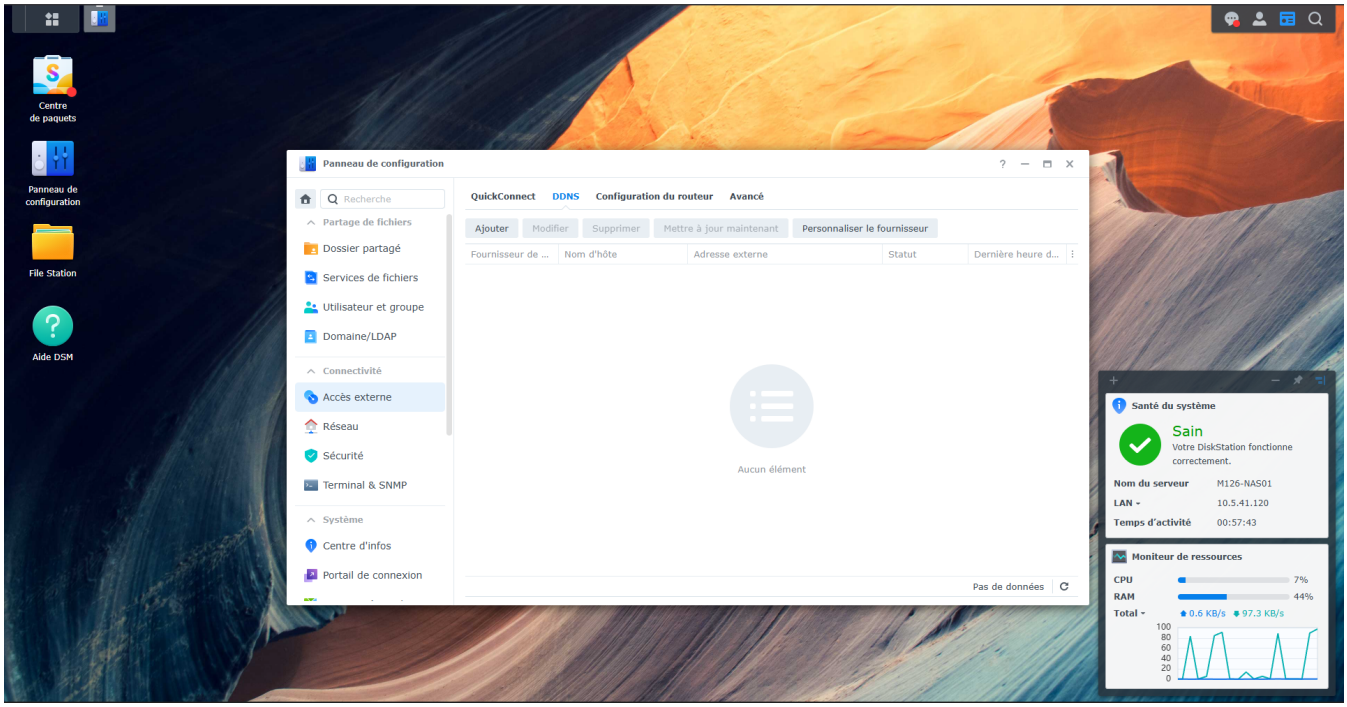

Figure 21 — Panneau de configuration du NAS

Cliquer sur `Ajouter` et vous devriez avoir cette pop-up :

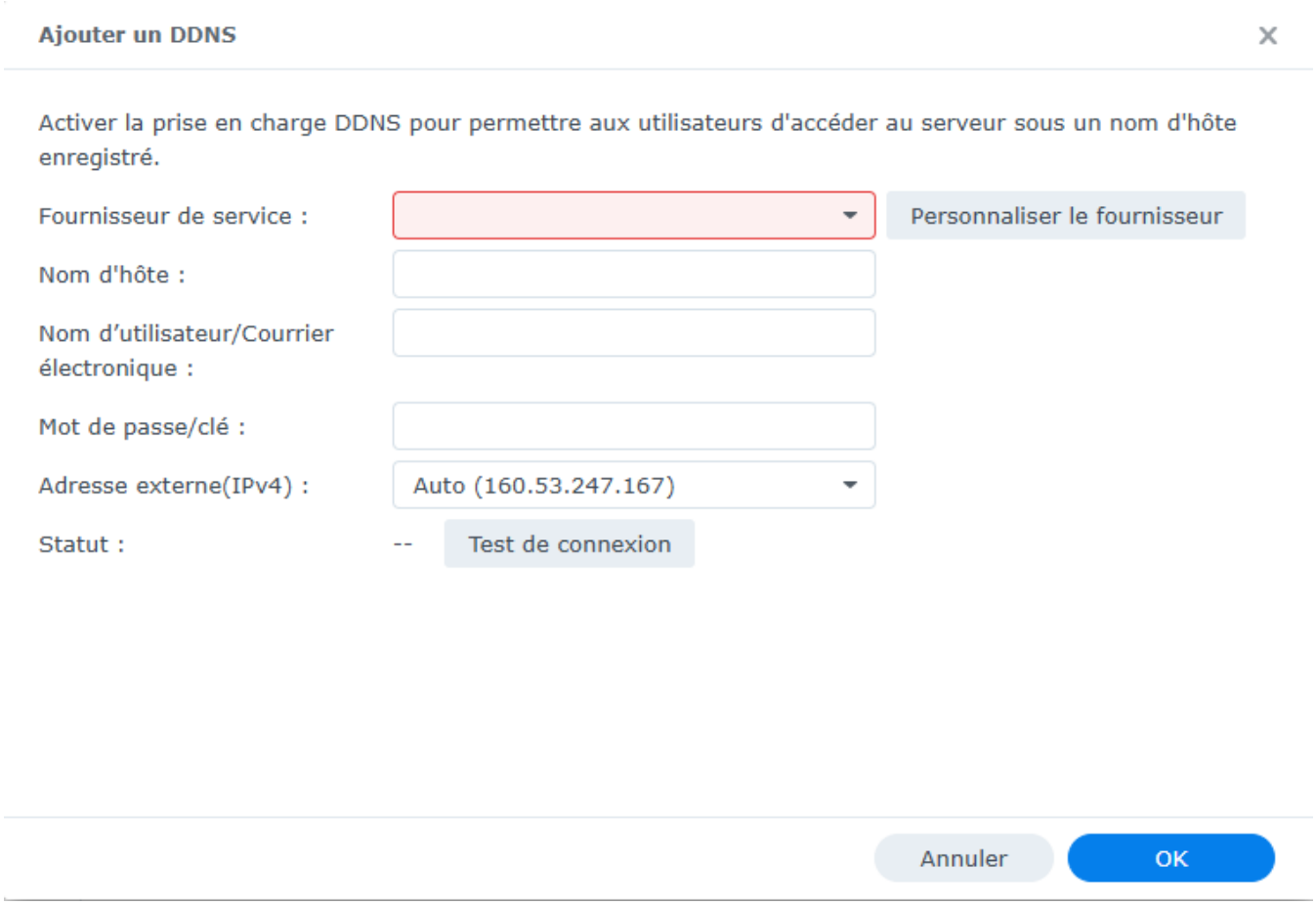

Figure 22 — Panneau de configuration du NAS, Section DDNS

Vous trouverez 4 champs à configurer :

- Le Fournisseur (votre SLD comme Infomaniak)
- Le nom d'hôte
- Votre adresse Mail
- Le mot de passe

Dans un premier temps on va faire le fournisseur. Le problème c'est que Infomaniak n'est pas dans la liste des fournisseurs par défaut, on doit le rajouter. Vous allez donc cliquer sur `Personnaliser le fournisseur`.

Ceci devrait apparaître :

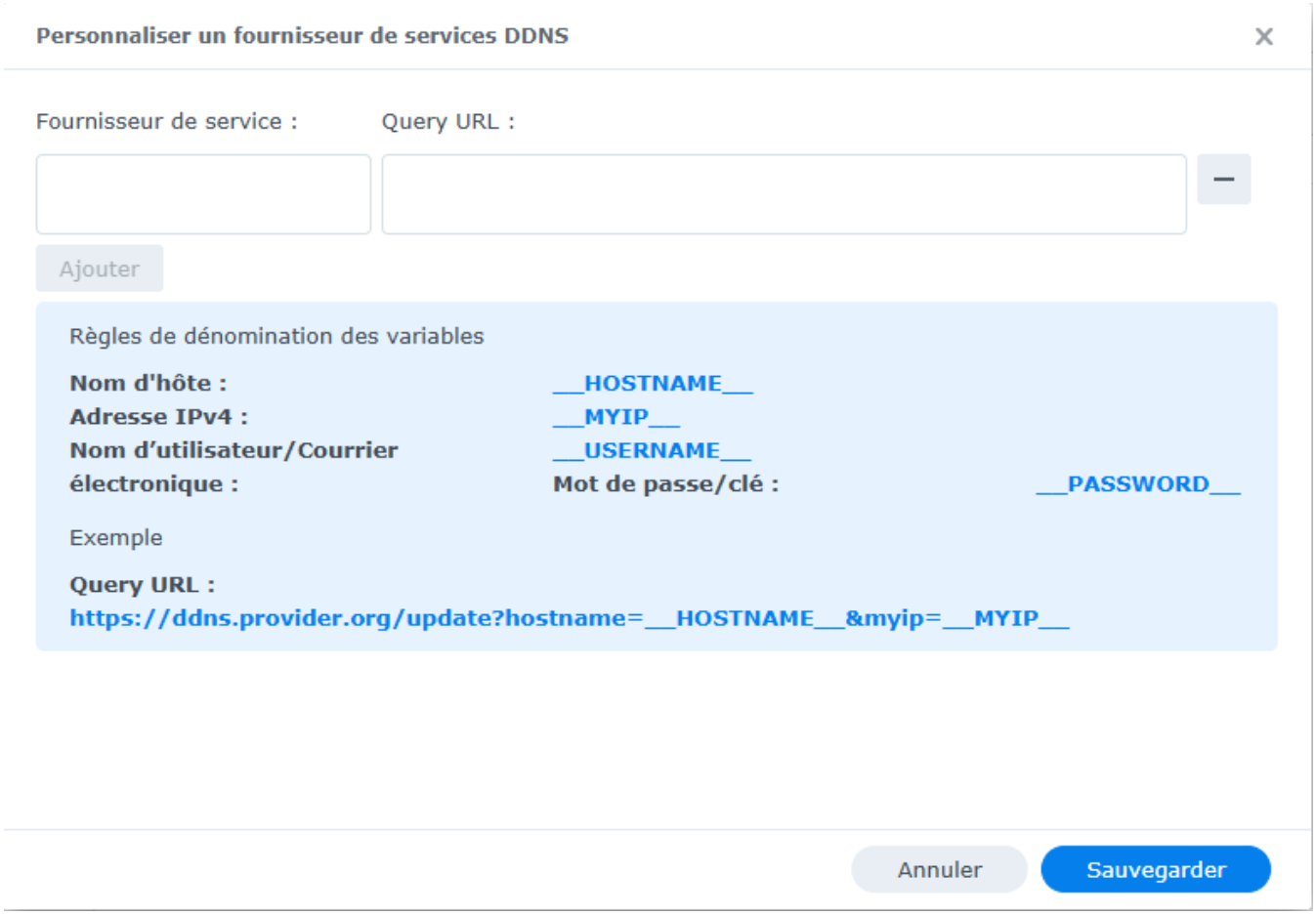

Figure 23 — Panneau de configuration du NAS, Pop-Up pour le fournisseur

Mettez le nom du fournisseur et le lien de mise-à-jour (Pour Infomaniak, voici le lien : `infomaniak.com/nic/update`). Cliquer sur `Sauvegarder` et vous n'avez plus qu'à sélectionner Infomaniak dans la liste des fournisseurs.

Ensuite, dans le nom d'hôte, mettez le sous-domaine que vous utilisez (dans mon cas, `drive.benone.ch`).

Pour le nom d'utilisateur, mettez le nom d'utilisateur que vous avez mis sur votre fournisseur (dans mon cas, `drive-dyndns-user`) et son mot de passe.

Vous n'avez plus qu'à tester via le bouton `Test de connexion` et cela devrait vous mettre `Normal`.

Aller maintenant sur le lien et vérifier que cela fonctionne !

5.7 Architecture DNS

Une architecture DNS se divise sur plusieurs niveaux.

5.7.1 Explication de chaque niveau

Niveau Racine (Root Level)

C'est le sommet de l'arbre DNS, où se trouvent les serveurs racines qui gèrent la zone racine. Ils dirigent les requêtes vers les serveurs des domaines de premier niveau (TLD) comme .ch, .fr, etc...

Domaines de premier niveau (TLD)

TLD ou `Top Level Domain` représentent les extensions de domaine, qui peuvent être génériques (.com, .org, etc...) ou géographiques (.fr, .ch, etc...). Les serveurs autoritaires des TLD gèrent les sous-domaines correspondant à leur niveau.

Domaine de second niveau (SLD)

SLD ou `Second Level Domain` sont généralement attribués à des organisations ou individus et sont administrés par eux via des registraires (exemple : Infomaniak).

Sous-Domaines

Ils permettent de subdiviser un domaine pour organiser des sections spécifiques (exemple : gitlab.unrecon.ch).

5.7.2 Exemple en image

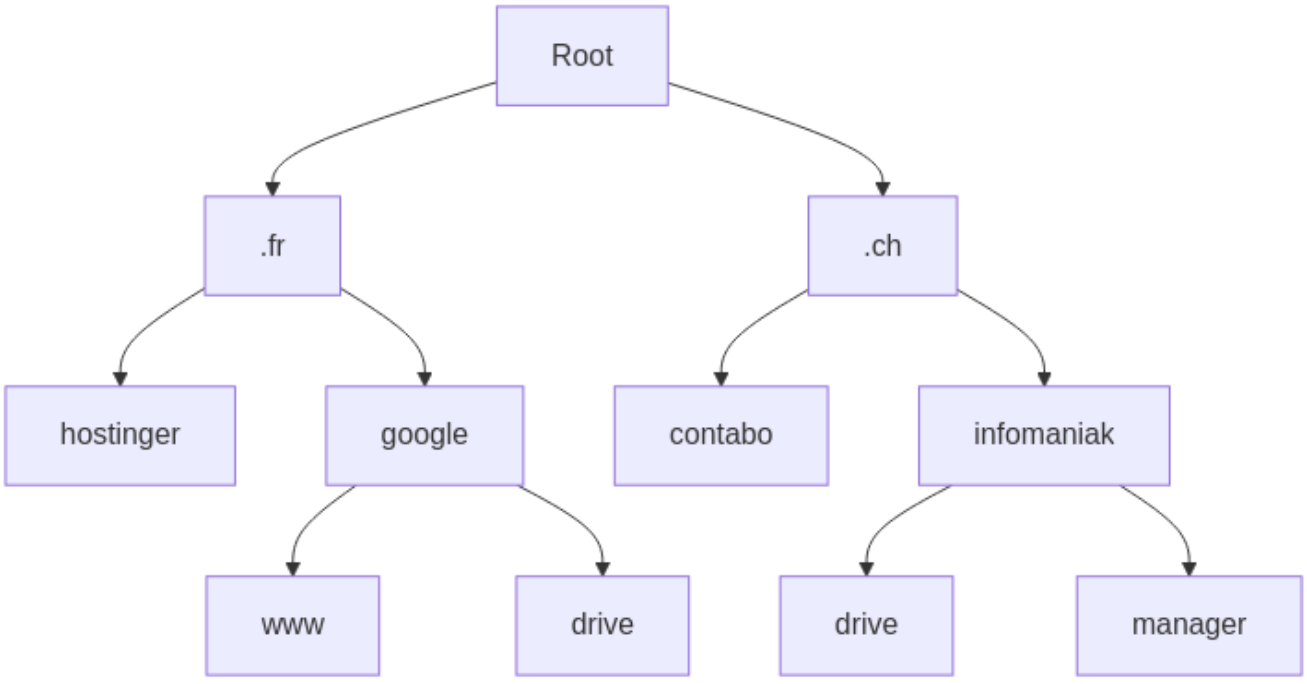

Figure 24 — Exemple d'une architecture DNS

5.7.3 Requête en image

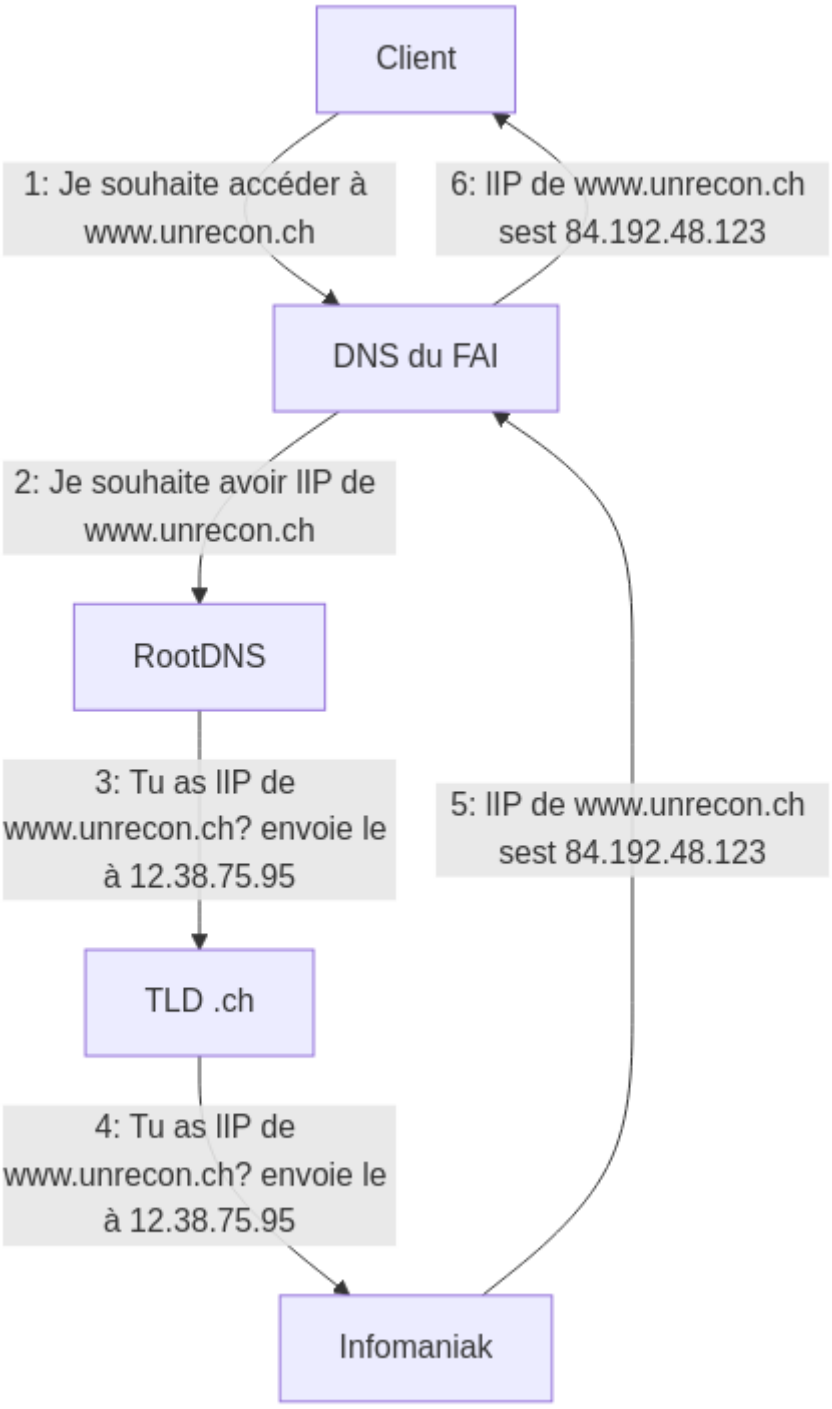

Figure 25 — Exemple du fonctionnement d'un DNS au niveau avancé

Explication

Supposons que vous souhaitez visiter le site web de `www.unrecon.ch` .

1. Le client souhaite aller sur `www.unrecon.ch`
2. Le DNS du FAI demande au serveur DNS Racine pour l'IP de `www.unrecon.ch`

Le RootDNS ne contient pas l'IP mais uniquement les informations des TLD, qu'elle serveur gère le domaine de premier niveau.

1. Le RootDNS demande alors au TLD ch pour avoir l'IP de `www.unrecon.ch` .

Le TLD ne contient pas non plus l'IP mais il sait ou est enregistré l'IP de `www.unrecon.ch` .

1. Le TLD demande au SLD l'IP de `www.unrecon.ch`

Infomaniak est un des registraires dans le monde. Il y a aussi Contabo, Google, GoDaddy et plein d'autres.

Le SLD est celui qui contient tout votre registre DNS comme vos enregistrements de type A, AAAA, etc...

1. Le SLD renvoie donc au serveur DNS du FAI l'adresse IP de `www.unrecon.ch` .
2. Le DNS du FAI vous renvoie l'adresse IP de `www.unrecon.ch` .

5.7.4 Liste des serveurs RootDNS

Nom	IPv4	IPv6	Opérateur
a.root-servers.net	198.41.0.4	2001:503:ba3e::2:30	Verisign, Inc.
b.root-servers.net	170.247.170.2	2801:1b8:10::b	University of Southern California, Information Sciences Institute
c.root-servers.net	192.33.4.12	2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13	2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10	2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241	2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4	2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53	2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17	2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30	2001:503:c27::2:30	Verisign, Inc.
k.root-servers.net	193.0.14.129	2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42	2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33	2001:dc3::35	WIDE Project

5.7.5 Liste des serveurs TLD .ch et .fr

ch

Les serveurs DNS TLD .ch sont gérés par SWITCH.

Nom	IPv4	IPv6
a.nic.ch	130.59.31.41	2001:620:0:ff:0:0:0:56
b.nic.ch	130.59.31.43	2001:620:0:ff:0:0:0:58
d.nic.ch	194.0.25.39	2001:678:20:0:0:0:0:39
e.nic.ch	194.0.17.1	2001:678:3:0:0:0:0:1
f.nic.ch	194.146.106.10	2001:67c:1010:2:0:0:0:53

fr

Nom	IPv4	IPv6
d.nic.fr	194.0.9.1	2001:678:c:0:0:0:0:1
f.ext.nic.fr	194.146.106.46	2001:67c:1010:11:0:0:0:53
g.ext.nic.fr	194.0.36.1	2001:678:4c:0:0:0:0:1

5.8 Serveur DNS privé et public

5.8.1 Privé

Un serveur DNS privé permet de gérer la résolution des noms de domaines en fonction de son registre mais peut aussi utiliser un serveur DNS public pour avoir accès à internet.

5.8.2 Public

Un serveur DNS public permet de gérer la résolution des noms de domaines au niveau mondial. Contrairement à un DNS privé, il n'est pas possible d'écraser un nom de domaine car elle suit les règles.

5.8.3 Exemple

Imaginez que vous travaillez dans une entreprise, on va la CPEG, la Caisse de Prévoyance de l'État de Genève.

La CPEG à un site public qui est sur les serveurs DNS public mais imaginons qu'elle a un site interne disponible à l'adresse `www.cpeg.int`. `www.cpeg.int` est enregistré sur le serveur DNS privé de l'entreprise et `www.cpeg.ch` est toujours accessible par les employés. Comment ça marche ?

Cas 1 : DNS Privé

L'employé souhaite accédé à `www.cpeg.int`. Le serveur DNS locale va alors voir si il a le nom de domaine dans son registre. Dans ce cas, il l'a, alors il va retourné l'IP du site web.

Cas 2 : DNS Public

L'employé souhaite accédé à `www.cpeg.ch`. Le serveur DNS locale va alors voir si il a le nom de domaine dans son registre. Dans ce cas, il ne l'a pas, alors il va demandé à un autre serveur DNS pour avoir l'IP et va le donner le renvoyer.

5.8.4 Comment ça marche

Le serveur DNS privé a enregistré le nom de domaine `www.cpeg.int` dans son registre, mais il n'a pas `www.cpeg.ch`. Pour continuer à avoir accès à l'Internet mondial, il faut dire au serveur DNS privé que si il n'a pas l'adresse, d'aller questionner un serveur DNS public.

5.8.5 Exemple d'architecture

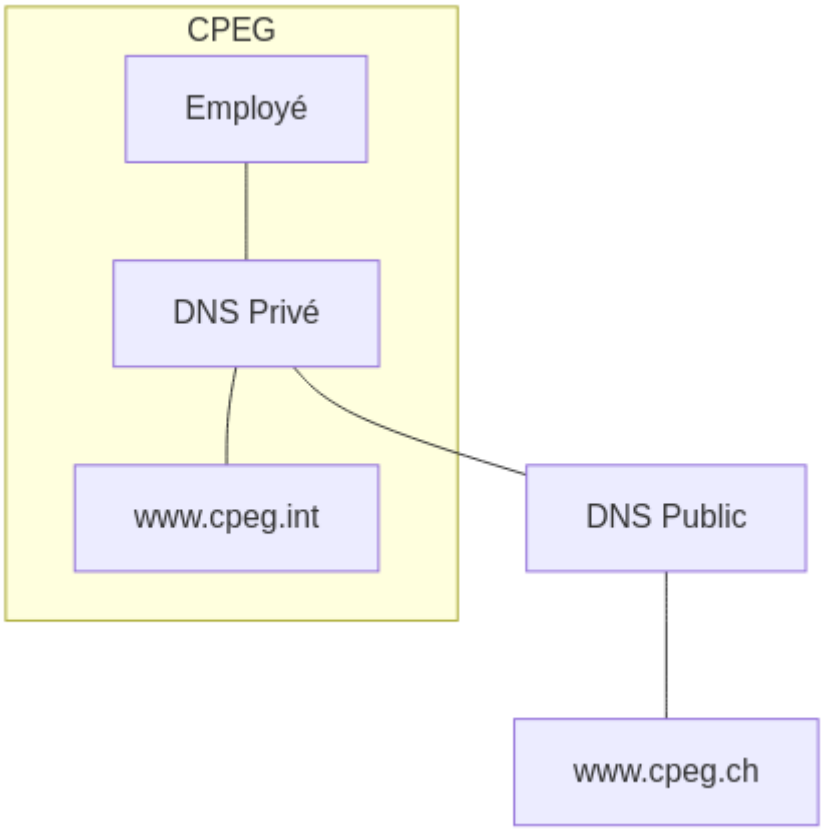

Figure 26 — Architecture fictif avec un DNS Privé

5.8.6 Liste des serveurs DNS publics

Propriétaire	IP DNS Primaire	IP DNS Secondaire
Google	8.8.8.8	8.8.4.4
Cloudflare	1.1.1.1	1.0.0.1
OpenDNS	208.67.222.222	208.67.220.220
Swisscom	195.186.4.162	195.186.1.162
Sunrise	212.98.37.132	194.230.55.100
Salt	162.159.32.11	162.159.33.85

5.9 Différence entre client et serveur

Pour comprendre la différence entre client et serveur, il faut savoir qu'il existe 3 types de serveur DNS.

5.9.1 Type de serveur

Autoritaire (Serveur)

Les serveurs dit **Autoritaire**, ils contiennent les informations de leur zone et ne s'occupe que de leur zone d'autorité.

Les RootDNS et les TLD sont des serveurs DNS autoritaires.

Récursif (Serveur)

Les serveurs dit **Récursif**, s'occupe de la résolution des noms de domaines, c'est eux qui contiennent les informations des noms de domaines ou qui vont faire le demande vers d'autres serveurs DNS récursif.

Les serveurs DNS récursifs font 2 types de requêtes, des requêtes récursives et non-récursives.

Si le serveur DNS à la donnée dans son cache ou la dans son registre, il va alors faire une requête non récursive, c'est à dire qu'il ne va pas faire d'autre requête externes et va juste renvoyer l'information.

Si le serveur DNS n'à pas la donnée dans son cache ou dans son registre, il va alors faire une requête récursive, c'est à dire qu'il va demander l'information à d'autres serveurs DNS jusqu'à avoir la réponse ou renvoyer une erreur.

Les SLD sont des serveurs DNS sont des serveurs DNS récursif.

Client

Le client DNS quant à lui, ne fait qu'une requête au serveur DNS qui le gère qui est **OBLIGATOIREMENT** un serveur DNS récursif. Le client DNS ne va jamais intérrogé un serveur DNS autoritaire car ce n'est pas son rôle.

Votre routeur est un client DNS.

5.10 Maître de zone et esclave

Le concepte de Maître de zone et d'esclave sert a déchargé le serveur DNS principale (ici, maître de zone) en plein de petit serveur DNS (ici, esclave).

L'esclave a les données via le maître de zone mais il n'a que des droits de lecture. C'est à dire que si vous consultez un site, il peut vous donnez la donnée mais elle ne peut pas la modifié, vous devez vous adressez au maître de zone.

L'esclave, pour obtenir les données, va faire ce qu'on appel un transfert de zone, il va demandé une copie au maître de zone pour se synchronisé avec lui.

5.10.1 Exemple

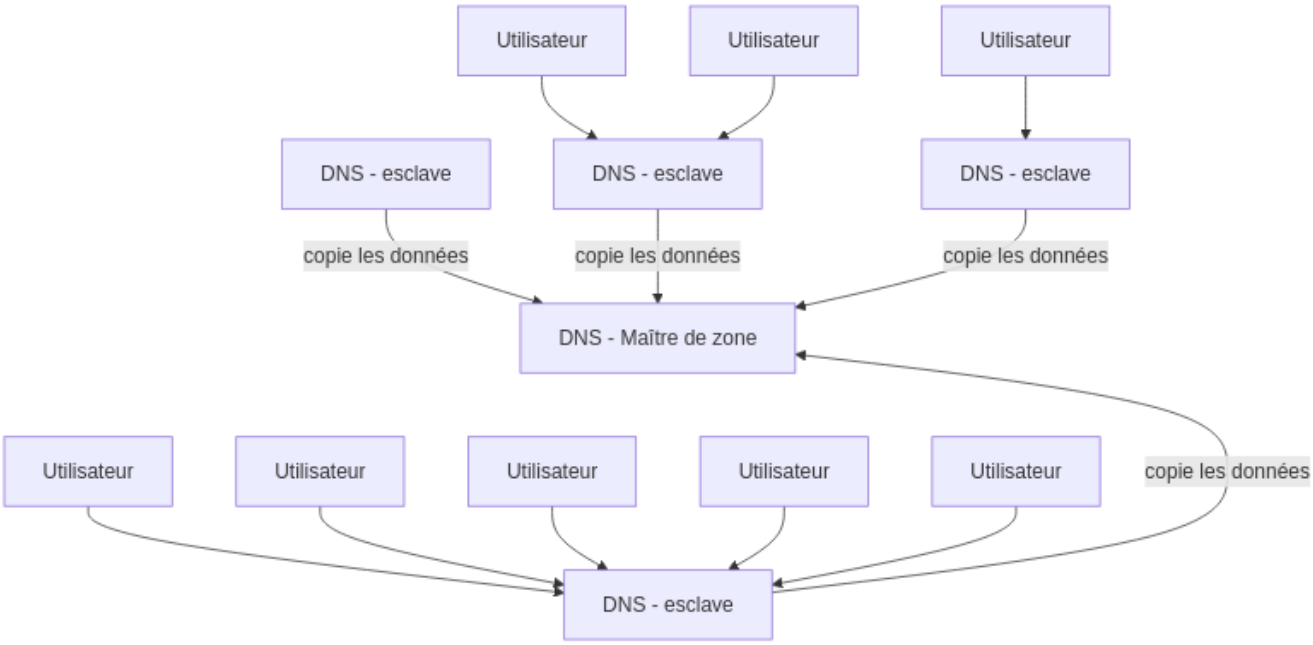

Figure 27 — Exemple d'une architecture avec Maitre de zone et des esclaves

Les utilisateurs vont seulement intéragir avec les esclaves pour obtenir les informations.

Si ils doivent modifier des données, ils vont directement voir le maître de zone.

5.11 Logiciel de Serveur DNS

5.11.1 Windows

Si vous souhaitez utiliser windows pour héberger votre serveur DNS, je vous recommande fortement d'installer une version de Windows Server (2022, 2019, 2016) qui intègre l'outil `DNS Windows Server`.

Cliquez-ici pour aller voir un tutoriel.

5.11.2 Linux

Sur Linux, il existe un logiciel nommée `BIND9` qui permet d'agir comme un serveur DNS.

Cliquez-ici pour aller voir un tutoriel.

5.11.3 MacOS

Vous pouvez aussi installer `BIND9` sur MacOS via le gestionnaire de paquets.

Cliquez-ici pour aller voir un tutoriel.

5.12 Outils

Ils existent différent outils pour examiner les données d'un serveur DNS.

5.12.1 NSLookUp

C'est un outil installable ou une version en ligne existe via ce lien.

Il permet de tester la résolution des serveurs DNS.

5.12.2 Dig

C'est un outil de google pour vérifier les informations d'un serveur DNS disponible en ligne via ce lien.

5.12.3 WhoIs

WhoIs permet, comme Dig, de vérifier les informations d'un serveur DNS et même de faire un diagnostique.

C'est un site disponible via ce lien.

5.13 Source

- Wikipedia DNS
- Devopedia
- Infomaniak
- IBM
- CloudFlare
- AWS
- IANA RootDNS
- IANA ch
- IANA fr
- Root-Servers
- Wikipedia WHOIS

5.14 Normes RFC

- IETF RFC 1035
- IETF RFC 2782
- IETF RFC 3596
- IETF RFC 4034
- IETF RFC 4255
- IETF RFC 6376
- IETF RFC 6672
- IETF RFC 6698
- IETF RFC 7489
- IETF RFC 8162
- IETF RFC 8659

5.15 Liens des outils/tutoriels

- Tutoriel Serveur Windows DNS
- Tutoriel DNS Bind9 sur Linux
- Tutoriel DNS Bind9 sur MacOS
- Outil en ligne nslookup.
- Outil en ligne Dig
- Outil en ligne WhoIs

5 novembre 2025

Tables des figures

Images

Blocs de code

