

BTS Services informatiques aux organisations - SISR	
Session 2022	
E4 – Support et mise à disposition de services informatiques	
Coefficient 4	
DESCRIPTION DE LA REALISATION PROFESSIONNELLE	
NOM et prénom du candidat : LEDUC Quentin	N° candidat : 02145639104
Contexte de la réalisation professionnelle Avec les membres de Scani, Actifs, nous voulons lancer un LAB pour nos tests, et nos pré-productions de nos logiciels internes et systèmes de l'extérieur. Pour ce faire nous devons installer un proxy.	
Intitulé de la réalisation professionnelle Installer un serveur proxy avec nginx	
Période de réalisation : DU 15/08/21 AU 18/08/21 Modalité : En équipe	Lieu : Joigny
Principale(s) activité(s) concernée(s) : <ul style="list-style-type: none"> - Développer son projet professionnel - Mettre en place son environnement d'apprentissage personnel 	
Conditions de réalisation <ul style="list-style-type: none"> • Ressources présentes Aucune • Résultats attendus Pouvoir parler à toutes nos machines depuis l'extérieur • Durée de réalisation 	
Modalités d'accès à cette réalisation professionnelle. Site internet : www.netwaze.fr Aller dans « Réalisations Professionnelles » Mot de passe : Mr.Robot	

PROBLÉMATIQUE

Pour faciliter la vie de mes projets j'ai besoin de deux serveurs web. Le notre actuel est basé sur un Rocky avec NGINX et d'un autre basé sur un Ubuntu avec Apache. Ubuntu étant le système Debian le plus utiliser il y a l'ensemble de mes projets compatible avec ce système. Comparer à maintenant ou je remodifie et bricole pour que la plupart fonctionne avec NGINX

SOLUTION

Pour que nos 2 serveurs WEB fonctionne nous allons faire en sorte d'installer un Proxy qui s'occupera de rediriger les noms de domaine / sous domaine au bon serveur en HTTP. Ce serveur produira les certificat SSL pour l'utilisateur. Voulant apprendre d'avantage à me servir de distribution Debian, je vais choisir un Ubuntu pour créer ce serveur Proxy.

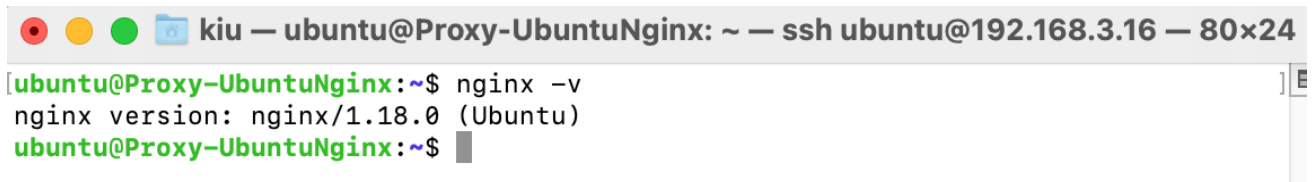
PRÉREQUIS MATÉRIEL

- Un serveur Ubuntu
- Connaitre les adresses IP de son réseau
- Avoir de la patience

PRÉREQUIS

Tout d'abord nous mettons à jour notre serveur et on installe NGINX.

```
$ sudo apt update
$ sudo apt upgrade
$ sudo apt Install nginx
```

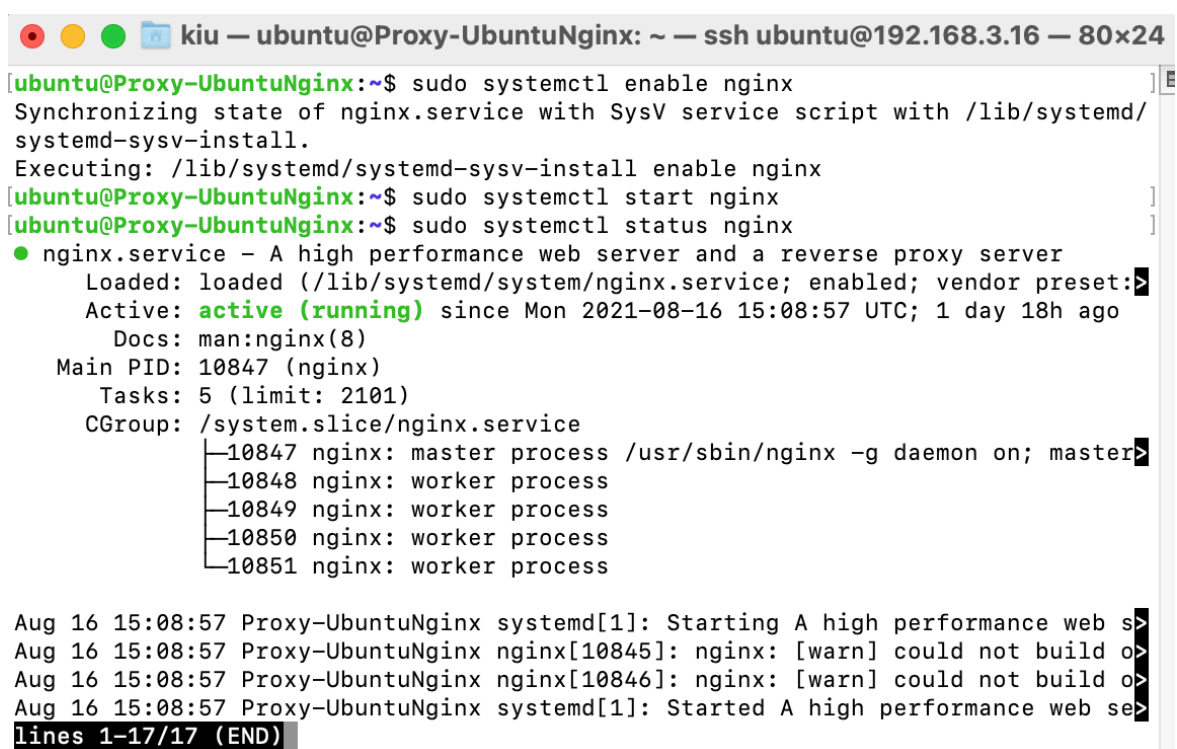


```
kiu — ubuntu@Proxy-UbuntuNginx: ~ — ssh ubuntu@192.168.3.16 — 80x24
[ubuntu@Proxy-UbuntuNginx:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
ubuntu@Proxy-UbuntuNginx:~$ █
```

Une fois installé on vérifie la version de nginx : `nginx -v`

Puis nous activons le démarrage de NGINX au démarrage du système et nous démarrons notre serveur NGINX et on vérifie qu'il fonctionne bien.

```
$ sudo systemctl enable nginx
$ sudo systemctl start nginx
$ sudo systemctl status nginx
```



```
kiu — ubuntu@Proxy-UbuntuNginx: ~ — ssh ubuntu@192.168.3.16 — 80x24
[ubuntu@Proxy-UbuntuNginx:~$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /lib/systemd/
systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
[ubuntu@Proxy-UbuntuNginx:~$ sudo systemctl start nginx
[ubuntu@Proxy-UbuntuNginx:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:
   Active: active (running) since Mon 2021-08-16 15:08:57 UTC; 1 day 18h ago
     Docs: man:nginx(8)
    Main PID: 10847 (nginx)
      Tasks: 5 (limit: 2101)
   CGroup: /system.slice/nginx.service
           └─10847 nginx: master process /usr/sbin/nginx -g daemon on; master
             └─10848 nginx: worker process
               └─10849 nginx: worker process
                 └─10850 nginx: worker process
                   └─10851 nginx: worker process

Aug 16 15:08:57 Proxy-UbuntuNginx systemd[1]: Starting A high performance web s
Aug 16 15:08:57 Proxy-UbuntuNginx nginx[10845]: nginx: [warn] could not build o
Aug 16 15:08:57 Proxy-UbuntuNginx nginx[10846]: nginx: [warn] could not build o
Aug 16 15:08:57 Proxy-UbuntuNginx systemd[1]: Started A high performance web se
lines 1-17/17 (END)
```

CONFIGURATION DU PROXY NGINX

Puis nous nous rendons dans notre `/etc/nginx/sites-available/` et on crée un nouveau fichier, pour mon exemple je dois mettre netwaze.fr derrière ce proxy.

```
[ubuntu@Proxy-UbuntuNginx:/etc/nginx/sites-available]$ ls  
default netwaze.fr.conf  
ubuntu@Proxy-UbuntuNginx:/etc/nginx/sites-available$ █
```

Nous devons aussi créer un lien symbolique dans le dossier `/etc/nginx/sites-enabled/` avec la commande : `$ ln -s /etc/nginx/sites-available/netwaze.fr.conf /etc/nginx/sites-enabled/netwaze.fr.conf` et on redémarre

Puis nous allons appliquer une configuration pour qu'il redirige toutes les requêtes vers mon serveur netwaze.fr et une configuration pour WordPress qui est légèrement différente d'une configuration basic. On installe ensuite certbot : `$ sudo apt install certbot python3-certbot-nginx` ATTENTION il faut ouvrir les ports 80 et 443 sur son routeur vers le proxy.

Il faut par la même occasion ouvrir les firewall dans Ubuntu pour laisser les ports rentrer dans notre machine.

```
$ ufw allow 80
```

```
$ ufw allow 443
```

CRÉATION DU CERTIFICAT SSL

Nous passons donc à la configuration de certbot avec : `$ sudo certbot --nginx -d netwaze.fr -d www.netwaze.fr`

Une fois la configuration de certbot terminée nous pouvons constater que certbot a intégré nos certificats directement dans notre configuration de netwaze.fr.conf.

```
upstream netwaze {
    server 192.168.3.183;
}

server {
    server_name netwaze.fr;

    location / {
        rewrite ^([\^?#]*)/([\^?#\.\.]+)([\^?#].*)?$ $1$2/$3 permanent;
        proxy_set_header X-Forwarded-Host $host;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_pass http://netwaze;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_connect_timeout 90;
        proxy_read_timeout 90;
        proxy_send_timeout 90;
        proxy_set_header X-Is-Reverse-Proxy "true";
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_buffering on;
        proxy_buffers 12 12k;
        proxy_redirect off;
        proxy_http_version 1.1;
    }

    # global gzip on
    gzip on;
    gzip_min_length 10240;
    gzip_types text/plain text/css text/xml text/javascript application/x-javascript application/xml;
    gzip_disable "MSIE [1-6]\.";
    add_header Cache-Control public;

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/netwaze.fr/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/netwaze.fr/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

server {
    if ($host = netwaze.fr) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    server_name netwaze.fr;
    listen 80;
    return 404; # managed by Certbot
}
```

Fichier de configuration de netwaze.fr.conf + certbot

CONFIGURATION DE NOTRE SERVEUR WEB + WORDPRESS

Maintenant que nous avons installé notre certificat il faut aller le retirer de notre serveur web qui se trouve dans notre configuration de netwaze.fr dans `/etc/nginx/conf.d/netwaze.fr.conf`. Une fois retiré nous avons cette configuration.

```
GNU nano 2.9.8 netwaze.fr.conf

server {
    listen 80;
    server_name netwaze.fr;
    root /var/www/netwaze/;
    index index.php;

    access_log /var/log/nginx/netwaze.fr.access.log;
    error_log /var/log/nginx/netwaze.fr.error.log;

    # Don't allow pages to be rendered in an iframe on external domains.
    add_header X-Frame-Options "SAMEORIGIN";

    # MIME sniffing prevention
    add_header X-Content-Type-Options "nosniff";

    # Enable cross-site scripting filter in supported browsers.
    add_header X-Xss-Protection "1; mode=block";

    location = /favicon.ico {
        log_not_found off;
        access_log off;
    }

    location = /robots.txt {
        allow all;
        log_not_found off;
        access_log off;
    }

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

    location ~ \.php$ {
        try_files $uri =404;
        fastcgi_pass unix:/run/php-fpm/www.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
    #location ~* \.(js|css|png|jpg|jpeg|gif|ico|svg)$ {
    #    expires max;
    #    log_not_found off;
    # }
    # Prevent access to hidden files
    location ~* /\.(!well-known\/) {
        deny all;
    }
    # Prevent access to certain file extensions
    location ~\.(ini|log|conf)$ {
        deny all;
    }
}
```

Maintenant que notre fichier de configuration est ok il ne reste plus qu'à rajouter un petit bout de code dans notre fichier wp-config.php de notre wordpress qui ainsi saura qu'il est derrière un proxy et qu'il ne soit pas surpris d'avoir des paquets pour lui venant du proxy.

<?php

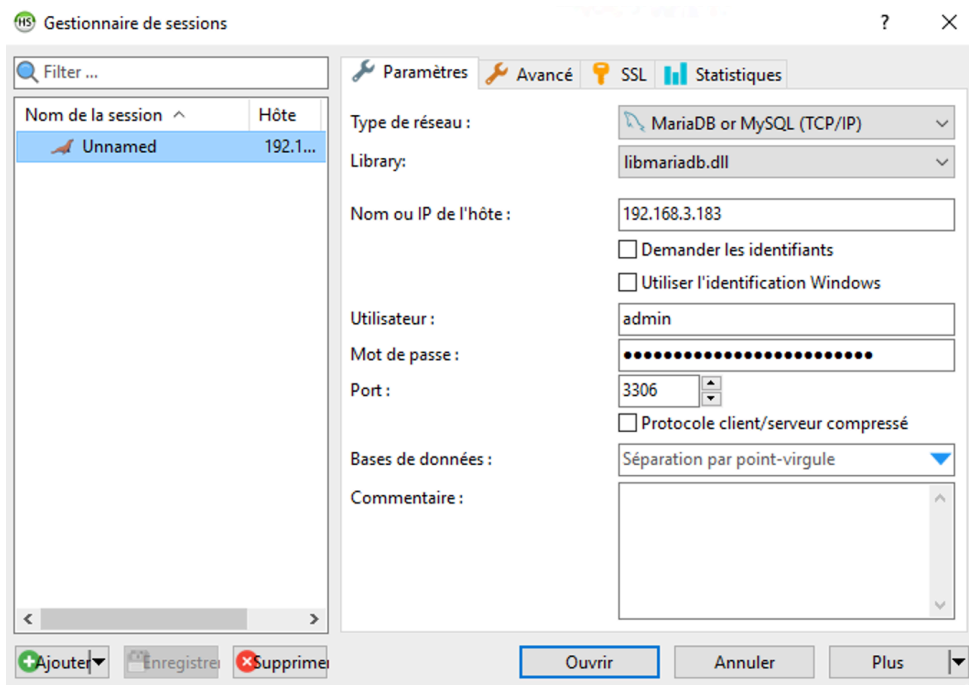
```
define('FORCE_SSL_ADMIN', true) ; if ($_SERVER['HTTP_X_FORWARDED_PROTO'] == 'https') $_SERVER['HTTPS']='on' ; if (isset($_SERVER['HTTP_X_FORWARDED_HOST'])) { $_SERVER['HTTP_HOST'] = $_SERVER['HTTP_X_FO$
```

Il faut impérativement rentrer ce bout de code directement après la variable <?php.

CONFIGURATION DE LA BASE DE DONNÉES

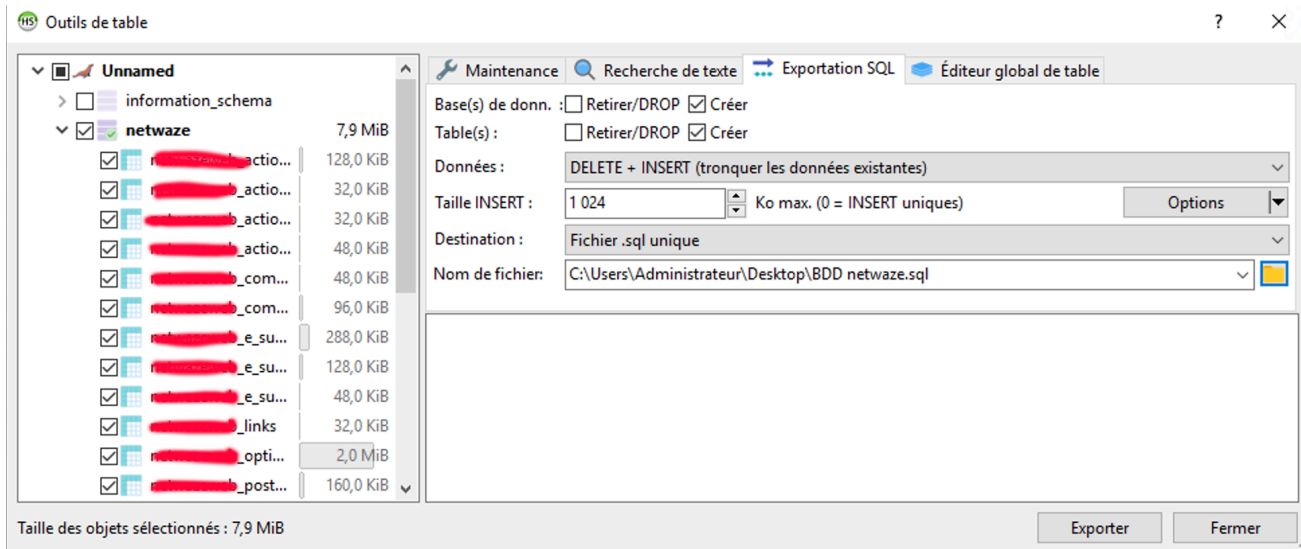
A l'heure actuelle notre configuration fonctionne ... mais nous avons un petit problème HTTPS. Avant notre serveur répondait directement sur le port 443, hors maintenant le Proxy s'occupe du certificat et par conséquent écoute sur le port 443. Maintenant notre Wordpress en local doit être sur du HTTP car le Proxy et le serveur web fonctionne dans ce sens. Pour ce faire nous allons retirer totalement les https:// et les remplacer par http:// dans la base de données.

Pour nous connecter dessus nous allons utiliser une application Gratuite et Opensource qui s'appel HeidiSQL et Notepad++.

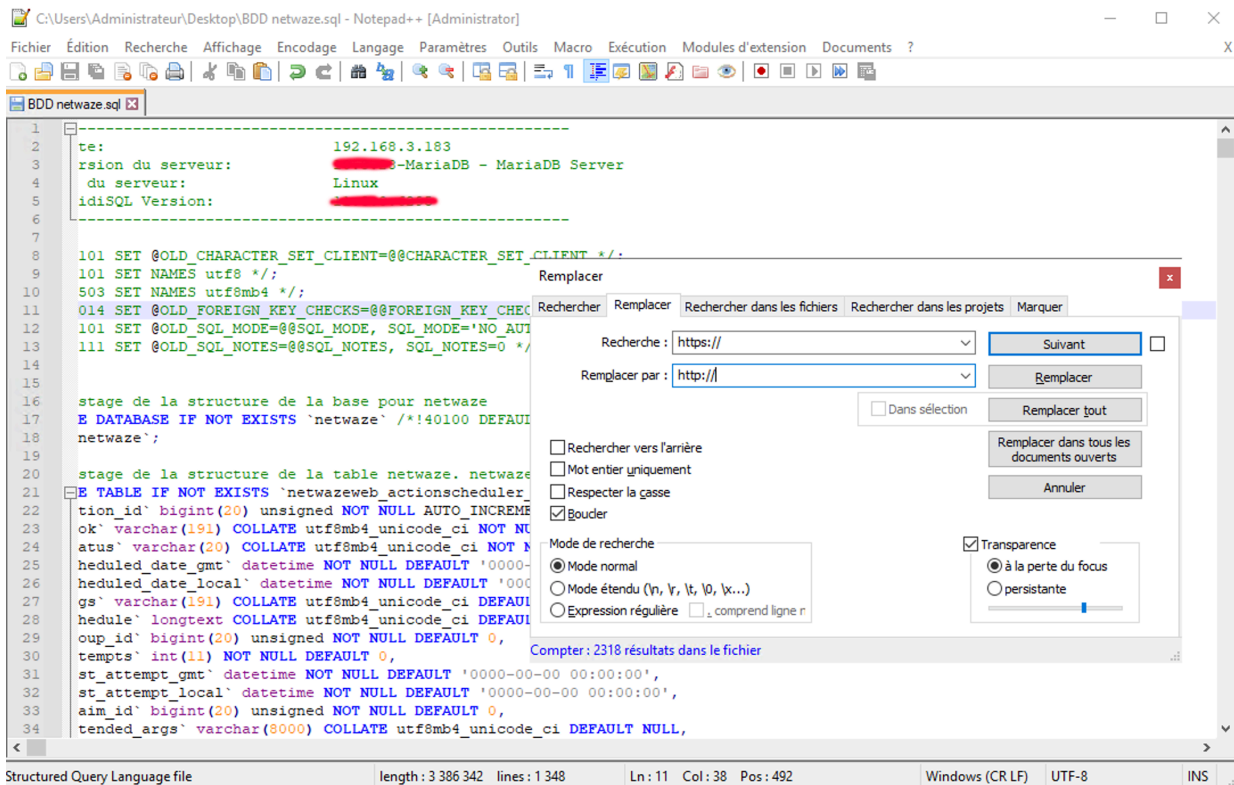


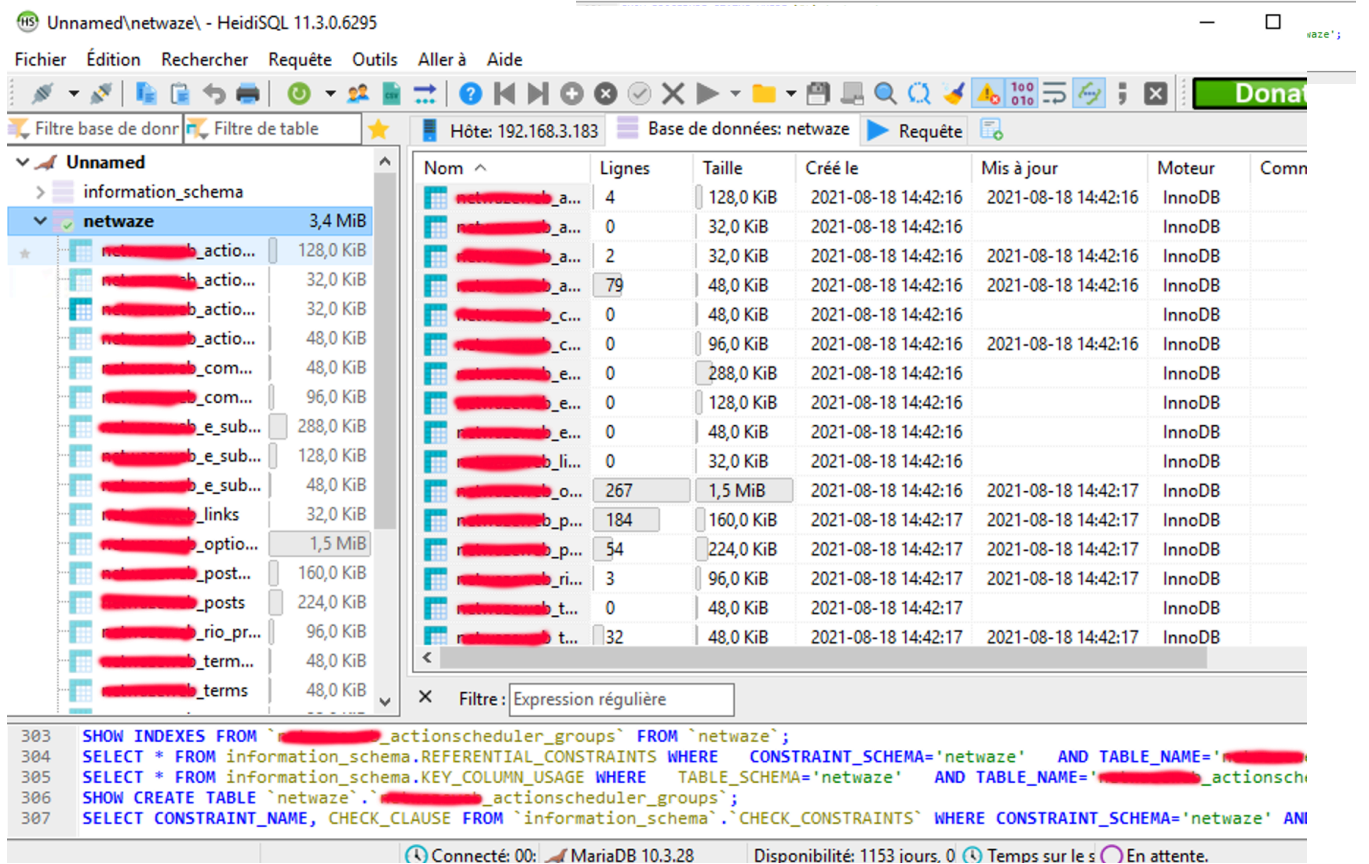
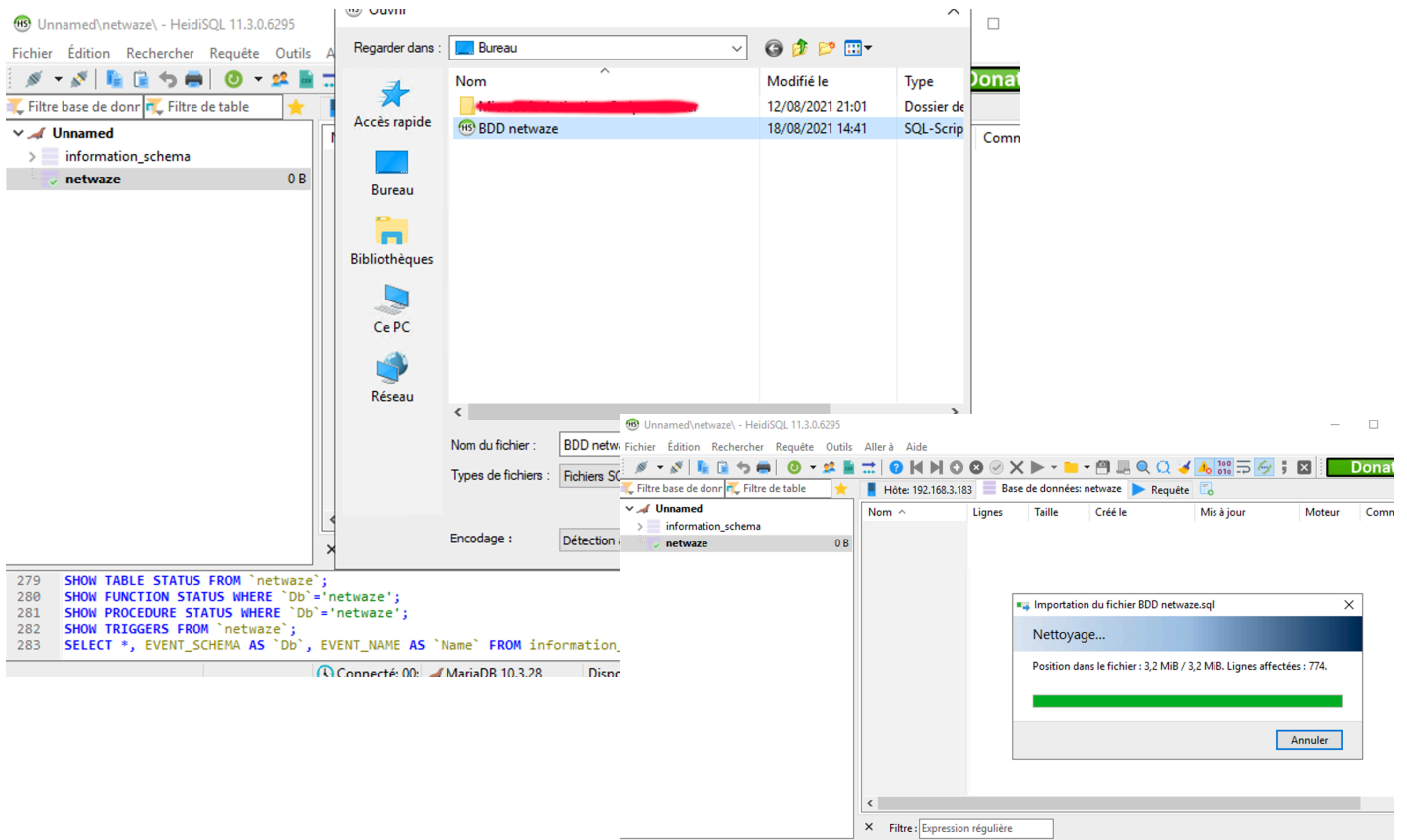
Tout d'abord nous nous connectons à notre base de données. Automatiquement l'application va l'enregistrer. Bien évidemment cet outil fonctionne avec MariaDB / Mysql, PostreSQL, SQLite et plein d'autre ...

Une fois connecter nous sauvegardons notre base de données, pour ce faire nous devons aller dans outils puis exporter la base de données en SQL puis nous prenons toutes les tables SQL et on choisi dans données « DELETE + INSERT » et on fini par lui donner un chemin de destination final et on peut exporter.



Nous ouvrons Notepad++ et recherchons tout les termes https:// pour les remplacer par http://. On peut sauvegarder et réimporter la base de données avec HeidiSQL.





Notre base de données correctement réimporter à partir de HeidiSQL

Nous pouvons maintenant utiliser notre outil après ces vérifications. Nous tapons la commande : \$ `https netwaze.fr` qui devrait nous donner notre serveur proxy.

```
[kiu@MacBook-Pro-de-kiu ~ % https netwaze.fr
HTTP/1.1 200 OK
Cache-Control: public
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html; charset=UTF-8
Date: Wed, 18 Aug 2021 13:08:42 GMT
Last-Modified: Wed, 18 Aug 2021 12:44:43 GMT
Server: nginx/1.18.0 (Ubuntu)
Transfer-Encoding: chunked
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Powered-By: PHP/7.4.6
X-Xss-Protection: 1; mode=block
```

Ceci nous donne bien notre proxy nginx 1.18.0 qui tourne bien sur notre Ubuntu.

On peut aussi le vérifier d'une autre manière, en éteignant notre serveur web. Ceci donnera une erreur quand on voudra aller sur notre site internet.



Afin d'avoir un peu plus de sécurité on va enlever le NGINX/1.18.0 (Ubuntu). Pour le faire nous devons installer un nouveau paquet \$ `sudo apt install nginx-extras` et on va modifier dans notre configuration `nginx.conf` `server_token` en `off` et rajouter une ligne `more_set_headers 'Server: Ce site est un serveur Apache :)';`

```
sendfile on;
tcp_nopush on;
tcp_nodelay on;
keepalive_timeout 65;
types_hash_max_size 2048;
server_tokens off;
more_set_headers 'Server: Ce site est un serveur Apache :)';
# server_names_hash_bucket_size 64;
# server_name_in_redirect off;

include /etc/nginx/mime.types;
default_type application/octet-stream;

##
```

On redémarre NGINX : \$ systemctl restart nginx
et on regarde notre erreur changer.

Nous voyons que notre erreur avec le numéro de version et le système d'exploitation a disparu mais il reste nginx (pas très important) et quand on refait un \$ https netwaze.fr on remarque que notre message apparait.

502 Bad Gateway

nginx

```
Cache-Control: public
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html; charset=UTF-8
Date: Wed, 18 Aug 2021 13:53:28 GMT
Last-Modified: Wed, 18 Aug 2021 12:44:43 GMT
Server: Ce site est un serveur Apache :)
Transfer-Encoding: chunked
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Powered-By: PHP/7.4.6
X-Xss-Protection: 1; mode=block

<!DOCTYPE html>
```

Partie 3 – Veille technologique.

Pour faire un reverse proxy nous aurions pu utiliser Nginx Proxy Manager sous forme de Docker, HAProxy pour rajouter plus de sécurité avec une notion de haute disponibilité ou utiliser Squid.