

Installer le domaine "web-log-analyser"

- **Notes** : ce domaine hébergera l'outil *GoAccess* (équivalant à *AwStats*) permettant d'analyser et de visualiser les logs web.
- **Ressources** :
 - [GoAccess - Get Started](#)
 - [GoAccess - Man page](#)
 - [strftime - Man page](#)
 - [Nginx - Variables list](#)
 - [Nginx - WebSocket proxying](#) et [Nginx Reverse Proxy WebSocket Timeout](#)
 - [config/goaccess.conf - master - Github](#)
 - [Docker Hub - allinurl/goaccess](#)
 - [Script nginx2goaccess](#) : script transformant le format de log Nginx en valeur pour les variables de config GoAccess.

Installer le domaine

- Créer un fichier de configuration : `vi /etc/nginx/sites-available/web-log-analyser.conf`
 - Y placer le contenu suivant :

```
upstream websocket {
    server 127.0.0.1:57890;
}

server {
    listen 80;
    listen [::]:80;

    server_name web-log-analyser.<domaine-sinp>;

    auth_basic "Zone restreinte";
    auth_basic_user_file /etc/nginx/.htpasswd;

    location ^~ /ws {
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        proxy_pass http://websocket;
        proxy_http_version 1.1;
        proxy_read_timeout 1d;
    }

    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $realip_remote_addr;
        proxy_set_header X-Forwarded-Host $host:$server_port;
        proxy_set_header X-Forwarded-Server $host;
```

```
    proxy_set_header X-Forwarded-For
    $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;

    proxy_pass http://127.0.0.1:50082/;# ATTENTION : bien
mettre un slash final ! Sinon => erreur 404
}
}
```

- Voir la section [Configurer Nginx](#) pour la création et la gestion du fichier `.htpasswd`.
- Créer un lien depuis les sites actifs : `cd /etc/nginx/sites-enabled/ ; ln -s ../sites-available/web-log-analyser.conf web-log-analyser.conf`
 - Tester la config et relancer Nginx si OK : `nginx-reload` ou `nginx -t && nginx -s reload`
 - Tester l'URL `http://web-log-analyser.<domaine-sinp>/` qui doit afficher une erreur 502 car nous n'avons pas encore lancé le container Docker.
- En local, sur votre machine, se placer dans le dépôt Github "`sinp-<region>-srv`" récupéré précédemment et si nécessaire resynchroniser le dossier `bkp-srv` avec le serveur de destination en exécutant la commande `Rsync` indiquée dans le fichier `README.md`.
- Sur le serveur dans le dossier `docker` de l'utilisateur `admin` :
 - vérifier la présence du réseau Docker spécifique à notre utilisation de type `bridge` nommé `nginx-proxy` (voir fichier `.env`) : `docker network ls`
 - se placer dans le dossier `web-log-analyser.<domaine-sinp>` : `cd ~/docker/web-log-analyser`
 - exécuter la commande : `docker compose up`
 - vérifier que tout fonctionne à l'adresse : `http://web-log-analyser.<domaine-sinp>` (les graphiques doivent être remplis et l'icône de paramétrage du menu doit avoir une pastille verte ⇒ connexion WebSocket OK)
 - arrêter le container : `CTRL+C`
 - relancer le container en tant que service : `docker compose up -d`
 - si besoin de l'arrêter utiliser : `docker compose down`

Activer le SSL et HTTP2 sur le domaine

- Installer un certificat SSL via Certbot (Letsencrypt) :
 - Pour SINP PACA : `certbot --nginx -d web-log-analyser.silene.eu`
 - Pour SINP AURA : `certbot --nginx -d web-log-analyser.biodiversite-aura.net`
 - Répondre : 2
 - Tester ensuite la redirection auto de HTTP vers HTTPS :
`http://web-log-analyser.<domaine-sinp>/` → doit redirigé vers HTTPS automatiquement
- **ATTENTION** : penser à modifier les propriétés suivantes du fichier de conf `GoAccess` :
 - `ws-url` : remplacer `ws` : *par* `wss` : et remplacer le port 80 par 443
 - `origine` : remplacer le `scheme` de l'URL `http` par `https`
- Tester la configuration SSL :
<https://www.ssllabs.com/ssltest/analyze.html?d=web-log-analyser.silene.eu>
- Tester l'URL `https://web-log-analyser.<domaine-sinp>/`
- La config finale :

```
upstream websocket {
    server 127.0.0.1:57890;
}

server {
    listen 443 ssl http2; # managed by Certbot
    listen [::]:443 ssl http2; # managed by Certbot

    server_name web-log-analyser.<domaine-sinp>;

    auth_basic "Zone restreinte";
    auth_basic_user_file /etc/nginx/.htpasswd;

    location ^~ /ws {
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        proxy_pass http://websocket;
        proxy_http_version 1.1;
        proxy_read_timeout 1d;
    }

    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $realip_remote_addr;

        proxy_set_header X-Forwarded-Host $host:$server_port;
        proxy_set_header X-Forwarded-Server $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        proxy_pass http://127.0.0.1:50082/;# ATTENTION : bien mettre un
slash final ! Sinon => erreur 404
    }

    ssl_certificate /etc/letsencrypt/live/web-log-analyser.<domaine-
sinp>/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/web-log-
analyser.<domaine-sinp>/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 {
    listen 80;
    listen [::]:80;

    server_name web-log-analyser.<domaine-sinp>;

    return 302 https://web-log-analyser.<domaine-sinp>$request_uri;
```

}

Configurer la génération des rapports GoAccess

- **Notes** : *Logrotate* pour *Nginx* doit avoir été configuré pour utiliser un format de date %Y-%m-%d
- Le [dépôt Git du projet SINP-PACA](#) contient les scripts de génération de rapport journalier, hebdomadaire et mensuel pour *GoAccess* dans : `/web-srv/opt/goaccess-reports`
 - Les transférer sur le serveur :
 - Tester : `rsync -av ./web-srv/opt/ admin@web-<region>-sinp:/home/admin/dwL/ --dry-run`
 - Lancer le transfert (supprimer l'option `--dry-run`) : `rsync -av ./web-srv/opt/ admin@web-<region>-sinp:/home/admin/dwL/`
 - Copier le contenu de ce dossier dans `/opt/goaccess-reports/` : `cp -r /home/admin/dwL/goaccess-reports /opt/`
 - Donner les droits d'exécution aux scripts Bash : `chmod +x *-report`
 - Copier le fichier `goaccess-reports.cron` dans `/etc/cron.d/` en supprimant son extension `.cron` : `cp /opt/goaccess-reports/goaccess_reports.cron /etc/cron.d/goaccess_reports`
 - Recharger le service *Cron* pour prendre en compte les changements : `service cron reload`
- S'il est nécessaire de générer manuellement un rapport utiliser les commandes :
 - Rapport journalier : `/opt/goaccess-reports/daily-report <date-au-format-YYYY-MM-DD>` . Ex. : `/opt/goaccess-reports/daily-report 2020-01-01`
 - Le rapport est accessible ensuite à une adresse de la forme : <https://web-log-analyser.silene.eu/2023/06/05//<année>/<mois>/<jour>>
 - Ex. : <https://web-log-analyser.silene.eu/2020/01/01>
 - Rapport hebdomadaire : `/opt/goaccess-reports/weekly-report <date-d-un-lundi-au-format-YYYY-MM-DD>` . Ex. : `/opt/goaccess-reports/daily-report 2020-01-13`
 - Le rapport est accessible ensuite à une adresse de la forme : [/<année>/weeks/<numéro-semaine>](https://web-log-analyser.silene.eu/2020/weeks/02)
 - Ex. : <https://web-log-analyser.silene.eu/2020/weeks/02>
 - Rapport mensuel : `/opt/goaccess-reports/monthly-report <date-d-un-jour-du-mois-au-format-YYYY-MM-DD>` . Ex. : `/opt/goaccess-reports/monthly-report 2020-01-01`
 - Le rapport est accessible ensuite à une adresse de la forme : [/<année>/<mois>](https://web-log-analyser.silene.eu/2020/01)
 - Ex. : <https://web-log-analyser.silene.eu/2020/01>

Congifurer la geolocation

- Télécharger un base "IP to City Lite" depuis le site : <https://db-ip.com/db/lite.php>
- La placer dans le dossier `~/docker/web-log-analyser/goaccess/`
- Modifier le fichier `docker-compose.yml` et ajouter le volume suivant au container `web-log-analyser-goaccess` :

```
lite-2023-06.mmdb:/var/lib/GeoIP/GeoLite2-City.mmdb
```

- Créer un fichier goaccess/custom.css avec le contenu :

```
.wrap-panels:after {
  content: "IP Geolocation by DB-IP : https://db-ip.com";
  color: grey;
}
```

- Ajouter le *volume* suivant au 2 container de la stack web-log-analyser dans le fichier docker-compose.yml : - ./goaccess/custom.css:/goaccess/reports/custom.css:ro
- Ajouter les paramètres suivant à tous les fichiers de configuration de goaccess :

```
# User interfaces Options
html-custom-css /custom.css
# Geolocation Options
geoip-database /var/lib/GeoIP/GeoLite2-City.mmdb
```

- Redémarrer la stack : docker compose restart

Interface Temps Réel

- GoAccess propose une interface permettant d'afficher en temps réel les logs via l'utilisation de web-sockets.
 - Elle est accessible à la racine du domaine : <https://web-log-analyser.<domaine-sinp>/>
 - **TODO** : L'interface semble ne pas prendre en compte correctement les paramètres d'utilisation de la base de données sur disque. A voir sur le long terme (2023-06-06 - jpmilcent).
 - Nous utiliserons cette interface pour visualiser les derniers logs en temps réel.

Analyse des logs compressés .gz

- **Contexte** : GoAccess ne propose pas de solution native pour décompresser les logs. Il faut passer par *zcat* et l'utilisation des pipes.
- **Problème** : GoAccess gère mal la persistance en base des données de fichiers de log issu de l'utilisation d'un pipe avec *zcat*. Voir [la documentation de GoAccess](#).
- **Solution** : pour l'interface GoAccess temps réel, nous utilisons seulement le plus récent fichier de log généré par Nginx, le fichier *access.log*, avec l'utilisation des paramètres `--persist`, `--restore` et `--db-path=/goaccess/database`. Normalement, cela devrait nous permettre de retrouver toutes logs analysés dans l'interface en temps réel...

From:
<http://sinp-wiki.cbn-alpin.fr/> - **CBNA SINP**

Permanent link:
<http://sinp-wiki.cbn-alpin.fr/serveurs/installation/web-srv/docker-goaccess?rev=1686065686>

Last update: **2023/06/06 15:34**

