Installer, configurer et gérer le sous-domaine "cms"

- **Notes** : ce sous-domaine hébergera le Wordpress temporairement. Il sera à terme basculé sur le domaine principal <domaine-sinp> et le sous-domaine cms.<domaine-sinp> redirigera vers l'interface d'administration de Wordpress.
- Ressources :
 - Wordpress wp-config.php
 - How To Install WordPress With Docker Compose
 - Docker Hub Wordpress : paramètres et configuration.
 - Docker Wordpress Inject configuration using environment variable #142

Installer le domaine

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

```
server {
    listen 80;
    listen [::]:80;
    server name cms.<domaine-sinp>;
    location / {
        # ATTENTION : si le header HOST n'est pas renvoyé
Wordpress utilise l'url 127.0.0.1:50080 pour les fichiers CSS,
JS...
        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://127.0.0.1:50080/;# ATTENTION : bien
mettre un slash final! Sinon => erreur 404
```

- Créer un lien depuis les sites actifs : cd /etc/nginx/sites-enabled/; ln -s ../sites-available/cms.conf cms.conf
 - Tester la config et relancer Nginx si OK : nginx-reload ou nginx -t && nginx -s reload
 - Tester l'URL http://cms.<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 "docker-paca-sinp" récupéré

précédemment et si nécessaire resynchroniser le dossier web-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 :
 - o 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 cms.silene.eu : cd ~/docker/cms.silene.eu
 - o exécuter la commande : docker-compose up
 - vérifier que tout fonctionne à l'adresse : http://cms.<domaine-sinp> (l'installateur Wordpress doit s'afficher)
 - o 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

- Installer un certificat SSL via Certbot (Letsencrypt): certbot --nginx -d cms.<domainesinp>
 - Répondre : 2
 - \circ Tester ensuite la redirection auto de HTTP vers HTTPS : http://cms.<domaine-sinp>/ \rightarrow doit redirigé vers HTTPS automatiquement
- Tester la configuration SSL: https://www.ssllabs.com/ssltest/analyze.html?d=cms.<domainesinp>
- Tester I'URL https://cms.<domaine-sinp>/
- La configuration finale :

```
server {
    listen 443 ssl http2; # managed by Certbot
    listen [::]:443 ssl http2; # managed by Certbot
    server name cms.<domaine-sinp>;
    location / {
        # ATTENTION : si le header HOST n'est pas renvoyé Wordpress
utilise l'url 127.0.0.1:50080 pour les fichiers CSS, JS...
        proxy set header Host $host;
        proxy_set_header X-Real-IP $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://127.0.0.1:50080/;# ATTENTION : bien mettre un
slash final! Sinon => erreur 404
    }
    ssl certificate /etc/letsencrypt/live/cms.<domaine-</pre>
sinp>/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/cms.<domaine-</pre>
```

```
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 cms.<domaine-sinp>;
   return 302 https://$host$request_uri;
}
```

Basculer le domaine cms.<domaine-sinp> vers <domaine-sinp>

Par exemple, pour basculer le domaine *cms.silene.eu* vers *silene.eu* et en redirigeant *www.silene.eu*// *vers* silene.eu. Les étapes de la procédure à suivre sont : * au préalable, il est nécessaire de mettre à jour le certificat SSL pour inclure les nouveaux domaines www.silene.eu// et silene.eu. Ces domaines sont déjà attribué à un serveur en production. Il est donc nécessaire d'installer un plugin de *Certbot* : *certbot-dns-ovh*. Ce plugin, spécifique à OVH, va nous permettre de générer un certificat même si ces domaines ne pointent pas vers l'IP du nouveau serveur dans la zone DNS du domaine *silene.eu*.

- modifier le fichier de configuration de nginx
- modifier l'URL enregistré dans la configuration de Wordpress via l'interface d'administration

Installation du plugin certbot-dns-ovh

• Ressources :

- https://buzut.net/certbot-challenge-dns-ovh-wildcard/
- https://www.deltasight.fr/differentes-options-certbot/
- https://www.digitalocean.com/community/tutorials/how-to-acquire-a-let-s-encrypt-certifica te-using-dns-validation-with-acme-dns-certbot-on-ubuntu-18-04
- Sur "web-srv", installer le plugin avec la même méthode que Certbot : aptitude install python3-certbot-dns-ovh
- Créer une nouvelle clé d'API ici : https://eu.api.ovh.com/createToken/
 - Sélectionner 1 jour ou 30 jours pour la validité suivant le besoin
 - Pour les droits mettre ceci en remplaçant <domaine-sinp> par le nom de domaine principal (Ex. : "silene.eu") :

```
GET /domain/zone/
GET: /domain/zone/<domaine-sinp>/
GET /domain/zone/<domaine-sinp>/status
GET /domain/zone/<domaine-sinp>/record
GET /domain/zone/<domaine-sinp>/record/*
```

```
POST /domain/zone/<domaine-sinp>/record
POST /domain/zone/<domaine-sinp>/refresh
DELETE /domain/zone/<domaine-sinp>/record/*
```

- o Après validation, les informations qui s'affichent vont permettre de créer le fichier .ovhapi : vi /root/.ovhapi
 - Y placer le contenu suivant où les valeurs <...> seront remplacées par celles obtenues précédemment :

```
dns ovh endpoint = ovh-eu
dns ovh application_key = <application-key>
dns ovh application secret = <application-secret>
dns ovh consumer key = <consumer-key>
```

- Restreindre les droits sur ce fichier : chmod 600 /root/.ovhapi
- La tentative de mise à jour du certificat avec l'option --certname a échouée : certbot certonly --dns-ovh --dns-ovh-credentials ~/.ovhapi --cert-name cms.silene.eu -d silene.eu,www.silene.eu,cms.silene.eu
- C'est le renouvellement du certificat qui a permis le bon fonctionnement : certbot certonly --dns-ovh --dns-ovh-credentials ~/.ovhapi -d silene.eu, www.silene.eu, cms.silene.eu
 - L'option Renew & replace the cert (limit ~5 per 7 days) a été sélectionnée
- Ce mécanisme est aussi intéressant pour générer des certificats "étoiles" ou pour des serveurs non accessibles depuis internet.
- Pour le renouvellement du nom de domaine, si vous savez que les entrées DNS pointeront vers le bon serveur dans 60 jours alors vous pouvez modifier le fichier

/etc/letsencrypt/renewal/cms.<domaine-sinp>. Sous la section [renewalparams]:

- Supprimer la ligne : dns ovh credentials
- Remplacer la valeur de authenticator par nginx
- ∘ Ajouter une ligne : installer = nginx

Modification du fichier de configuration Nginx

- Modifier les lignes contenant server name ainsi : server name <domaine-sinp> www.<domaine-sinp> cms.<domaine-sinp>;
 - Ex.:server name silene.eu www.silene.eu cms.silene.eu;
- Recharger la configuration de Nginx : nginx reload
- La configuration finale :

```
server {
    listen 443 ssl http2; # managed by Certbot
    listen [::]:443 ssl http2; # managed by Certbot
    server name <domaine-sinp> www.<domaine-sinp> cms.<domaine-sinp>;
    location / {
        # ATTENTION : si le header HOST n'est pas renvoyé Wordpress
utilise l'url 127.0.0.1:50080 pour les fichiers CSS, JS...
        proxy_set_header Host $host;
```

```
proxy set header X-Real-IP $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 $scheme;
        proxy pass http://127.0.0.1:50080/;# ATTENTION : bien mettre un
slash final ! Sinon => erreur 404
    ssl certificate /etc/letsencrypt/live/cms.<domaine-</pre>
sinp>/fullchain.pem; # managed by Certbot
    ssl certificate key /etc/letsencrypt/live/cms.<domaine-</pre>
sinp>/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by
Certbot
    ssl dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
server {
    listen 80;
    listen [::]:80;
    server name <domaine-sinp> www.<domaine-sinp> cms.<domaine-sinp>;
    return 302 https://<domaine-sinp>$request uri;
```

Mise à jour de l'URL dans Wordpress

La dernière étapes consiste à mettre à jour l'URL enregistrée pour Wordpress :

- Se connecter à l'interface d'admin du wordpress : https://cms.<domaine-sinp>/wp-admin
- Se rendre à l'aide du menu gauche dans : Réglages > Général
- Remplacer l'ancienne URL (cms.<domaine-sinp>, par exemple cms.silene.eu) par la nouvelle (<domaine-sinp>, par exemple silene.eu) dans les champs :
 - Adresse web de Wordpress (URL)
 - Adresse web du site (URL)
- Enregistrer
- Vérifier la bonne redirection des domaines. Par exemple pour Silene, tous les domaines suivant doivent rediriger vers *silene.eu* en HTTPS : www.silene.eu, cms.silene.eu
- Il peut-être nécessaire de modifier les URL saisies dans les pages du CMS si elles ne sont pas correctes. Plusieurs solutions :
 - Modifier manuellement les URL (si peu de page et de références)
 - Passer par Adminer (ou PhpMyAdmin) pour faire les changements à l'aide de requêtes SQL : https://techsch.com/tutorials/change-domain-wordpress-docker-phpmyadmin
 - Utiliser Wordpress CLI avec la commande search-replace : https://developer.wordpress.org/cli/commands/search-replace/#examples

Modification temporaire de son fichier /etc/hosts

Si vous souhaitez depuis votre ordinateur accéder à un site dont les entrées DNS publiques ne pointent pas vers le bon serveur, il est nécessaire de modifier son fichier /etc/hosts. Voici les manipulations à réaliser suivant votre système d'exploitation :

Linux

- Éditer en root (ou avec sudo) le fichier /etc/hosts : vi /etc/hosts
- Ajouter une ligne contenant, par exemple : 51.91.137.130 www.silene.eu cms.silene.eu

Windows 10 and 8

- Appuyer sur la touche Windows
- Tapez Notepad dans le champ de recherche
- Dans les résultats, clic droit sur Notepad et sélectionner : "Exécuter en tant qu'administrateur"
- Depuis Notepad, ouvrir le fichier suivant: c:\Windows\System32\Drivers\etc\hosts
- Ajouter une ligne contenant: 51.91.137.130 silene.eu www.silene.eu cms.silene.eu
- Cliquer sur le menu : Fichier > Sauvegarder vos changements

Windows 7 and Vista

- Cliquez sur Démarrer > Tous les programmes > Accessoires.
- Clic droit sur Notepad et sélectionner "Exécuter en tant qu'administrateur"
- Cliquez sur Continuer dans la fenêtre qui demande votre permission.
- Depuis Notepad, cliquez sur : Fichier > Ouvrir.
- Dans le champ du nom du fichier, tapez : C:\Windows\System32\Drivers\etc\hosts
- Cliquer sur *Ouvrir*.
- Ajouter une ligne contenant : 51.91.137.130 silene.eu www.silene.eu cms.silene.eu
- Cliquer sur le menu : Fichier > Sauvegarder vos changements

MacOs

- Suivre ce tutoriel : https://pourron.com/aides-tutos/changer-le-ficher-hosts-sur-mac-os-x/
- La ligne à ajouter au fichier /etc/hosts est : 51.91.137.130 www.silene.eu cms.silene.eu

Note sur Docker et Wordpress

Ressources :

- Doc PhpMailer
- Discussion Github sur Docker Wordpress et la config d'envoie d'email
- Site d'exemple pour l'utilisation du Hook phpmailer init
- Par défaut, le docker Wordpress n'est pas configuré pour envoyé des emails. Pour que l'envoie d'email fonctionne aussi bien dans les extensions que dans le coeur de Wordpress, il a été nécessaire d'installer l'extension WP Mail SMTP.
 - Les tentatives suivantes ont été réalisés :
 - configuration de sendmail (installation, lancement auto, modif php.ini) dans l'image via un DockerFile ⇒ fonctionne dans le système du container mais pas dans Wordpress.
 - utilisation du hook phpmailer_init dans un plugin pour forcer l'utilisation du SMTP Google ⇒ fonctionne dans les plugins mais pas dans le coeur de Wordpress (les notifications de nouveau utilisateur par exemple ne sont pas envoyée).
 - utilisation d'autres plugin de config SMTP plus simple que WP Mail SMTP ⇒ ne fonctionne pas... A priori, WP Mail SMTP n'utilise pas le hook mais surcharge le variable globale \$phpmailer avec son propre code.
 - Les fichiers utilisés pour les différentes tentatives sont dans le dépôt Git SINP-PACA.

Configuration du Wordpress sur le sous-domaine "cms"



ATTENTION : activer l'indexation par les moteurs de recherche juste avant la mise en production

- Installateur Wordpress :
 - Choisir la langue : Français
 - Titre du site : SILENE
 - *Identifiant* : votre identifiant (ne pas utiliser de compte standard ex. admin pour éviter les piratages).
 - Mot de passe : voir Keepass
 - Votre adresse de messagerie : votre email pro
 - Visibilité pour les moteurs de recherche : cocher la case pour éviter l'indexation pendant la phase de mise au point...
- "Extensions" > "Extensions installées"
 - Supprimer toutes les extensions par défaut
- "Extensions" > "Ajouter"
 - Ajouter l'extension WP Mail SMTP et l'activer
 - Cliquer sur "Réglages" :
 - Adresse e-mail d'envoi : mailer@cbn-alpin.fr
 - Cocher Forcer l'e-mail d'expédition
 - Nom de l'expéditeur : Adminsys
 - Autre SMTP (OVH):
 - Hébergeur SMTP : ssl0.ovh.net
 - Cryptage : TLS Port SMTP : 587
 - o Identifiant SMTP: mailer@silene.eu
 - Mot de passe SMTP: utiliser le mot de passe du compte mailer@silene.eu. À modifier dans le fichier wp-config.php du container cms-wordpress (Voir comment modifier ce fichier). Modifier

également la valeur dans le fichier . env du dossier contenant le docker-compose.yml.

- Autre SMTP (Ancienne config avec Gmail) :
 - Hébergeur SMTP : smtp-relay.gmail.com
 - Cryptage : TLS o Port SMTP: 587
 - Identifiant SMTP : mailer@cbn-alpin.fr
 - Mot de passe SMTP : générer une mot de passe d'application sur le compte mailer@cbn-alpin.fr et l'utiliser ici.
- Tester l'envoie d'email avec l'onglet correspondant
- Ajouter l'extension WP Matomo et l'activer
 - Cliquer sur "Réglages" :
 - Se connecter à Matomo :
 - Mode de Matomo : Auto-hébergé (API HTTP par défaut)
 - URL de Matomo: https://analytics.<domaine-sinp>/
 - Jeton d'authentification à l'API : récupérer le jeton sur l'outil Matomo installé, menu "Paramètres > Personnel > Sécurité > Jetons d'authentification".
 - Cocher Auto configuration
 - Site déterminé : cms-<region>-sinp (https://cms.<domaine-sinp>)
 - Afficher les statistiques :
 - Date par défaut de Matomo : Aujourd'hui
 - Vue d'ensemble Matomo sur le Tableau de bord : Aujourd'hui
 - o Cocher Graphique Matomo sur le Tableau de bord
 - Afficher les statistiques pour : Administrator, Editor.
 - Cocher Afficher les statistiques par article
 - Cocher Raccourci Matomo
 - Nom affiché de WP-Matomo : WP Matomo
 - Cocher Activer les codes courts
 - Activer le suivi :
 - Ajouter le code de suivi : code de suivi par défaut
 - Position du code JavaScript : entête
 - ATTENTION : si WP-Matomo n'arrive pas à se connecter au domaine analytics.silene.eu c'est peut être du l'erreur "No route to host" du container Docker. Tester en se connectant au container et en tapant : curl https://analytics.silene.eu . Si Curl affiche "no route to host", le problème vient de Firewalld qui bloque l'accès depuis le container Docker.
- "Réglages" > "Général" :
 - Slogan : Système d'Information et de Localisation des Espèces Natives et Envahissantes
 - Adresse web de Wordpress (URL): d'abord mettre https://cms.<domaine-sinp> pendant la phase de test puis pour la production https:<domaine-sinp>. * Adresse web du site (URL): d'abord mettre https://cms. < domaine-sinp > pendant la phase de test puis pour la production https:<domaine-sinp>.
 - Adresse e-mail d'administration : adminsys@<domaine-sinp>
 - o Rôle par défaut de tout nouvel utilisateur : contributeur
- "Réglages" > "Discussions" :
 - Décocher la case Autoriser les notification de lien en provenance d'autres blogs (pings et rétroliens) sur les nouvelles publications
 - Décocher la case Autoriser les commentaires sur les nouvelles publications

- "Réglages" > "Lecture" :
 - Visibilité pour les moteurs de recherche : décocher la case "Demander aux moteurs de recherche de ne pas indexer ce site" lors de la mise en PRODUCTION.
- "Apparence" > "Personnaliser" > "Identité du site" :
 - Icône du site : charger le fichier favicon.png
- "Apparence" > "Thèmes" :
 - Supprimer tous les thèmes inutilisés (passer par "Détails" pour accéder à l'action "Supprimer")
- "Outils" > "Santé du site" :
 - Doit afficher: Bon travail! Tout fonctionne parfaitement ici.

Sauvegarde et restauration de Wordpress



Le container blacklabelops/volumerize n'est plus maintenu... voir à terme pour son remplacement si nécessaire

- La sauvegarde et la restauration des fichiers et base de données de Wordpress utilise le container blacklabelops/volumerize.
- Via Docker-Compose le lancement du container ne semble pas générer les fichiers nécessaire pour mettre en place le système Cron utilisant Jobber. Pour l'activer, il faut :
 - Se connecter au container : docker exec -it cms-volumerize /bin/bash
 - Se placer dans le dossier suivant : cd /opt/volumerize
 - Ajouter la ligne : CUR_DIR='/opt/volumerize après la première ligne du fichier : vi /otp/volumerize/create_jobber.sh
 - o Lancer le script : ./create jobber.sh
 - Vérifier la présence du fichier : vi /root/.jobber
 - Recharger les jobs de tous les utilisateurs : jobber reload -a
 - Vérifier la présence des jobs : jobber list
 - ∘ Tester un job : jobber test <nom-du-job>
- Les sauvegardes sont :
 - o exécutées automatiquement chaque jour à 2 heures du matin
 - glissantes sur 7 jours
 - incrémentales sur 6 jours
 - o complètes tous les 7 jours
 - stockées dans le dossier /home/admin/docker/cms.<domaine-sinp>/backup
- Backend de sauvegarde Dropbox :
 - https://github.com/blacklabelops/volumerize/tree/master/backends/Dropbox
- Le fichier docker-compose.yml principal contient le service cms volumerize. Ce service lance le container blacklabelops/volumerize qui contient l'outil Duplicity. Ce dernier se charge des sauvegardes et restauration.
- Pour réaliser une restauration, il faut utiliser le fichier *docker-compose.restore.yml* en suivant les étapes suivante :
 - Arrêter le service cms-volumerize : docker-compose stop cms-volumerize
 - Vérifier les différences entre les données actuelles et le dernier point de restauration : docker-compose -f docker-compose.restore.yml run cms-volumerizerestore-from-local verify
 - Pour vérifier avec un point plus ancien, utiliser l'option t et un temps
 - Exécuter une restauration avec le dernier point de restauration local : docker-compose

- -f docker-compose.restore.yml run cms-volumerize-restore-from-local restore
 - Pour restaurer à un point plus ancien, utiliser l'option t et un temps. Exemple pour restaurer l'état 3 jours avant : docker-compose -f dockercompose.restore.yml run cms-volumerize-restore-from-local restore -t 3D
- Si nécessaire, redémarrer les services restaurer : docker-compose restart cmsnginx cms-wordpress cms-mariadb cms-adminer cms-volumerize
- Vérifier que la restauration est bonne
- Voir la documentation pour restaurer depuis Dropbox
 - Ex.: docker-compose -f docker-compose.restore.yml run cmsvolumerize-restore-from-dropbox restore

Mise à jour du Docker Wordpress

- En local:
 - Mettre à jour les versions des outils dans le fichier docker-compose.yml
 - Mettre à jour la version de l'image Docker Wordpress dans le fichier /wordpress/build/Dockerfile .
 - Synchroniser le serveur avec ces modifications : rsync -av ./ admin@web-<region>-sinp:~/docker/<dossier-docker-cms>/ --dry-run
- Sur le serveur "web-srv":
 - Se placer dans le dossier contenant le fichier docker-compose.yml du CMS. Ex.: cd ~/docker/cms.silene.eu/
 - Arrêter les containers : docker-compose down
 - Redémarrer les container en mode "daemon" : docker-compose up -d
 - Notes : cela va automatiquement télécharger les nouvelles images et les démarrer.
- Sur le web :
 - Une fois les containers démarrés, se rendre sur le site du CMS
 - Se connecter à l'administration du Wordpress
 - Mettre à jour à nouveau le CMS avec le panneau de contrôle.
 - Notes : En effet, l'ensemble du contenu du CMS présent dans le container dans le dossier /var/www/html/ a été placé dans un volume docker. Ainsi, la mise à jour de l'image ne met pas à jour le CMS... Cette seconde mise à jour met donc bien à jour le CMS et sa base de données.

Modification de la configuration du Wordpress

S'il est nécessaire de mettre à jour la configuration du Docker Wordpress, il faut :

- Se connecter à l'instance "web-srv" en tant qu'admin : ssh admin@<sinp-web>
- Accéder à un shell du container du Wordpress: docker exec -it cms-wordpress
- Modifier le fichier de config : vi /var/www/html/wp-config.php
- Sortir du container : exit

Commandes utiles

- Pour accéder au container Nginx en tant que root : docker exec -it --user root cmsnginx /bin/bash
- Pour accéder au container Wordpress en tant que root : docker exec -it cms-wordpress --user root /bin/bash
- Effacer tous les volumes (ATTENTION : supprime toutes les données !) : docker-compose down --volumes
- Pour voir si tout vos paramètres sont correctement pris en compte par Docker Compose : docker-compose -f docker-compose.yml -f docker-compose.dev.yml config

From:

https://sinp-wiki.cbn-alpin.fr/ - CBNA SINP

Permanent link:

https://sinp-wiki.cbn-alpin.fr/serveurs/installation/web-srv/docker-wordpress?rev=1614891911

Last update: 2021/03/04 21:05

