BunkerWeb

Guide de Déploiement et Configuration

Web Application Firewall (WAF) avec Docker

Auteur : Mamadou CAMARA

Février 2026

SOMMAIRE :

1. Introduction

Ce guide vous accompagne dans le déploiement de BunkerWeb, une solution de **Web Application Firewall** (**WAF**) open-source qui sécurise vos applications Web. BunkerWeb est basé sur Nginx et intègre de nombreux mécanismes de protection avancés.

2. Qu'est-ce que BunkerWeb ?

BunkerWeb est une solution WAF (Web Application Firewall) complète qui permet de publier vos applications Web sur Internet en les protégeant efficacement. Il s'agit d'un reverse proxy basé sur Nginx qui embarque de nombreux composants de sécurité.

• Fonctionnalités principales

- **ModSecurity** : couche WAF détectant les requêtes malveillantes
- **DNSBL** : vérification des adresses IP sur liste noire
- **Bad Behavior** : détection des comportements suspects (erreurs 400/404)
- **Antibot :** système de Captcha pour valider les utilisateurs humains
- **GeoIP :** filtrage géographique
- **HTTPS** : intégration Let's Encrypt ou certificats personnalisés
- **Blacklist / Greylist / Whitelist** : gestion des listes d'accès

• Avantages

- Interface Web intuitive pour la configuration
- Solution française et open-source
- Support de plugins (CrowdSec, ClamAV, etc.)
- Intégration Docker et Kubernetes

3. Prérequis

• Environnement technique

- Serveur Linux (Ubuntu 22.04 ou supérieur recommandé)
- Docker et Docker Compose installés
- Enregistrement DNS pointant vers votre serveur

• Ressources minimales

Ressource	Minimum requis
CPU	1 cœur
RAM	1 Go
Disque	20 Go

Note : Ces ressources sont suffisantes pour un environnement de test. En production, les besoins augmenteront en fonction du trafic et du nombre de services.

4. Architecture de BunkerWeb

BunkerWeb s'appuie sur plusieurs services Docker pour fonctionner :

Service	Rôle
bunkerweb	Service frontal basé sur Nginx qui traite les requêtes HTTP/HTTPS
bw-scheduler	Orchestrateur qui gère le stockage et la génération des configurations
bw-ui	Interface Web de gestion et configuration
bw-db	Base de données MariaDB pour stocker les configurations

5. Déploiement avec Docker

• Création des répertoires

Créez la structure de répertoires nécessaire :

```
mkdir -p /containers/bunkerweb/

cd /containers/bunkerweb

chmod 700 bw-data
```

• Configuration Docker Compose

IMPORTANT : Avant de déployer, vous devez personnaliser les paramètres suivants dans le fichier **docker-compose.yml** :

Paramètre	Action requise
API_WHITELIST_IP	Ajouter **192.168.X.0/24** pour autoriser votre réseau (Réseau interne)
changeme_mysql	Remplacer par un mot de passe fort (3 occurrences)
changeme_adminpassword	Remplacer par un mot de passe administrateur fort pour accéder à l'interface Web
SERVER_NAME	Remplacer **bounkerweb.mamadou.local** par votre domaine

```
services:
  bunkerweb:
    image: bunkerity/bunkerweb:1.6.0
    restart: always
    ports:
      - 80:8080
      - 443:8443
    environment:
      - DATABASE_URI=mariadb+pymysql://bunkerweb:changeme_mysql@bw-
db:3306/db # Remember to set a stronger password for the database
      - API_WHITELIST_IP=127.0.0.0/8 10.150.250.0/24   192.168.X.0/24
    volumes:
      - ./ssl:/ssl
    networks:
      - bw-universe
      - bw-services
```

```
  bw-scheduler:
    image: bunkerity/bunkerweb-scheduler:1.6.0
    restart: always
    depends_on:
      - bunkerweb
    environment:
      - DATABASE_URI=mariadb+pymysql://bunkerweb:changeme_mysql@bw-
db:3306/db # Remember to set a stronger password for the database
      - API_WHITELIST_IP=127.0.0.0/8 10.150.250.0/24
      - BUNKERWEB_INSTANCES="bunkerweb"
      - MULTISITE=yes
      - SERVE_FILES="no"
      - USE_CLIENT_CACHE=yes
      - USE_GZIP=yes
      - SERVER_NAME=bunkerweb.mamadou.local
      - bunkerweb.mamadou.local_USE_UI=yes
      - bunkerweb.mamadou.local_USE_REVERSE_PROXY=yes
      - bunkerweb.mamadou.local_REVERSE_PROXY_URL=/
      - bunkerweb.mamadou.local_REVERSE_PROXY_HOST=http://bw-ui:7000
      - bunkerweb.mamadou.local_INTERCEPTED_ERROR_CODES=400 404 405 413
429 500 501 502 503 504
      - bunkerweb.mamadou.local_MAX_CLIENT_SIZE=50m
    volumes:
      - ./ssl:/ssl
    networks:
      - bw-universe
      - bw-docker

  bw-ui:
    image: bunkerity/bunkerweb-ui:1.6.0
    restart: always
    environment:
      - DATABASE_URI=mariadb+pymysql://bunkerweb:changeme_mysql@bw-
db:3306/db # Remember to set a stronger password for the database
      - ADMIN_USERNAME=admin
      - ADMIN_PASSWORD=changeme_adminpassword # Remember to set a stronger
password for the changeme user
    networks:
      - bw-universe
      - bw-docker

  bw-db:
    image: mariadb:11
    restart: always
    environment:
```

```
      - MYSQL_RANDOM_ROOT_PASSWORD=yes
      - MYSQL_DATABASE=db
      - MYSQL_USER=bunkerweb
      - MYSQL_PASSWORD=changeme_mysql # Remember to set a stronger password
for the database
    volumes:
      - ./bw-data:/var/lib/mysql
    networks:
      - bw-docker

networks:
  bw-universe:
    name: bw-universe
    ipam:
      driver: default
      config:
        - subnet: 10.150.250.0/24
  bw-services:
    name: bw-services
  bw-docker:
    name: bw-docker
```

Paramètres de sécurité critiques

1. Whitelist API (CRITIQUE)

Par défaut, l'API n'autorise que :

- 127.0.0.0/8 (localhost)
- 10.150.250.0/24 (réseau Docker interne)

⚠ **Pour accéder à l'interface depuis votre IP 192.168.X.Y, vous DEVEZ ajouter votre réseau à la whitelist :**

```
API_WHITELIST_IP=127.0.0.0/8 10.150.250.0/24 192.168.X.0/24
```

2. Mots de passe sécurisés

🚨 **DANGER : Ne jamais utiliser les mots de passe par défaut en production !**

Générer des mots de passe forts :

```
openssl rand -base64 32
```

6. Démarrage et vérification

- ## Téléchargement des images Docker

```
sudo docker compose pull
```

- ## Lancement des conteneurs

```
sudo docker compose up -d && sudo docker compose logs -f
```

Le premier démarrage prend quelques minutes. Surveillez les logs pour détecter d'éventuelles erreurs. Appuyez sur Ctrl+C pour quitter l'affichage des logs.

- ## Vérification de l'état des services

```
sudo docker  ps
```

```
root@debian:/home/mamadou/bunkerweb-project# sudo docker  ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED       STATUS                 PORTS
                                    NAMES
638294bc0015   bunkerity/bunkerweb-scheduler:1.6.0   "./entrypoint.sh"        3 weeks ago   Up 3 weeks (healthy)
                                    bunkerweb-project-bw-scheduler-1
a4debbc4ee5c   bunkerity/bunkerweb:1.6.0            "./entrypoint.sh"        3 weeks ago   Up 3 weeks (healthy)   80/tcp, 8443/udp, 0.0.0.0:80->8080/tcp, [::]:80->8080/tcp
0.0.0.0:443->8443/tcp, [::]:443->8443/tcp   bunkerweb-project-bunkerweb-1
009675a3fcf9   bunkerity/bunkerweb-ui:1.6.0         "./entrypoint.sh"        3 weeks ago   Up 3 weeks (healthy)   7000/tcp
                                    bunkerweb-project-bw-ui-1
54b148875bec   mariadb:11                           "docker-entrypoint.s…"   3 weeks ago   Up 3 weeks             3306/tcp
                                    bunkerweb-project-bw-db-1
root@debian:/home/mamadou/bunkerweb-project#
```

Tous les conteneurs doivent afficher l'état "Up" (running).

- ## Accès à l'interface Web

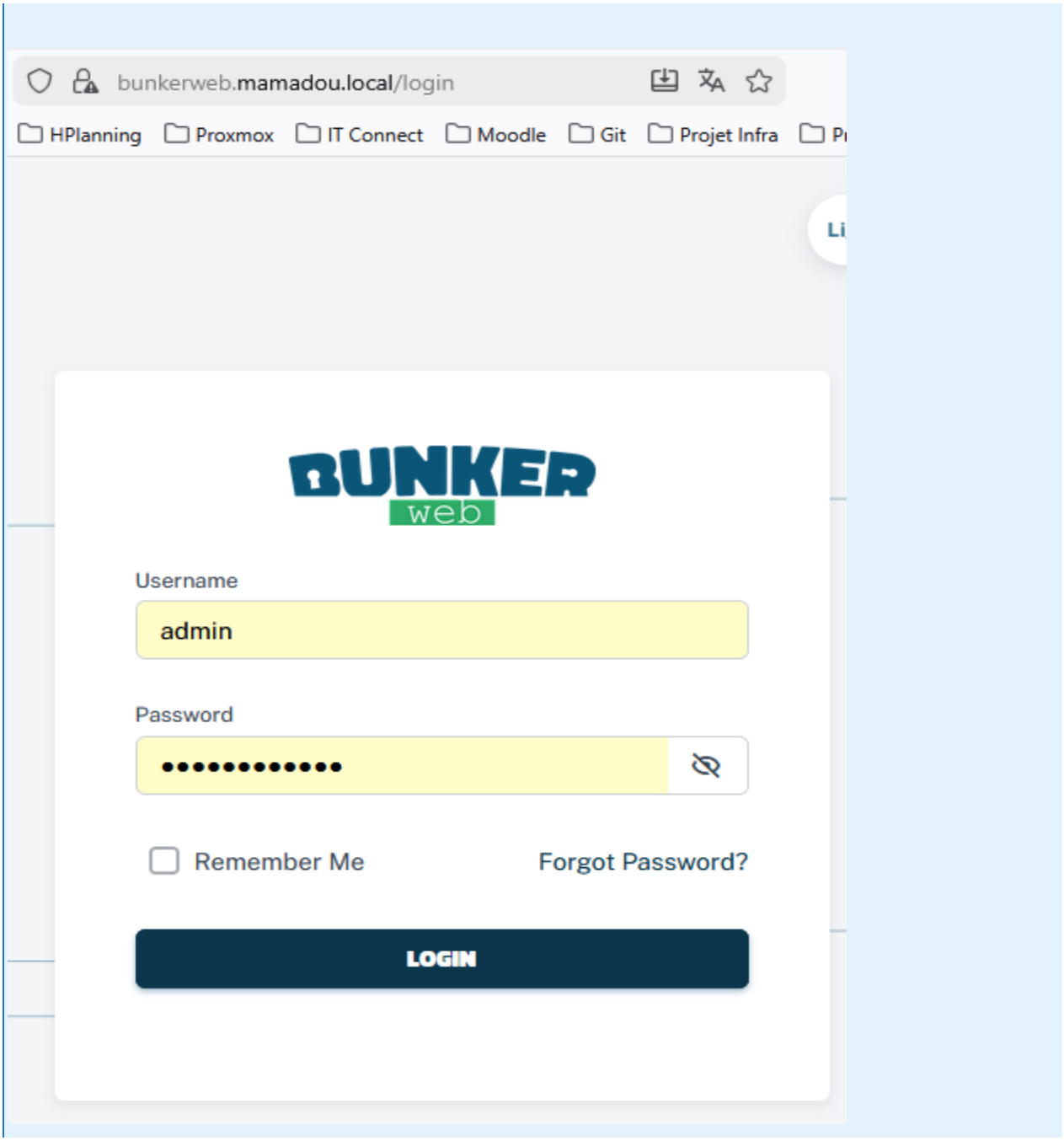
Une fois les conteneurs démarrés, accédez à l'interface Web via :

- http://bunkerweb.mamadou.local (ou votre nom de domaine personnalisé)
- http://192.168.X.Y (si le DNS n'est pas configuré)

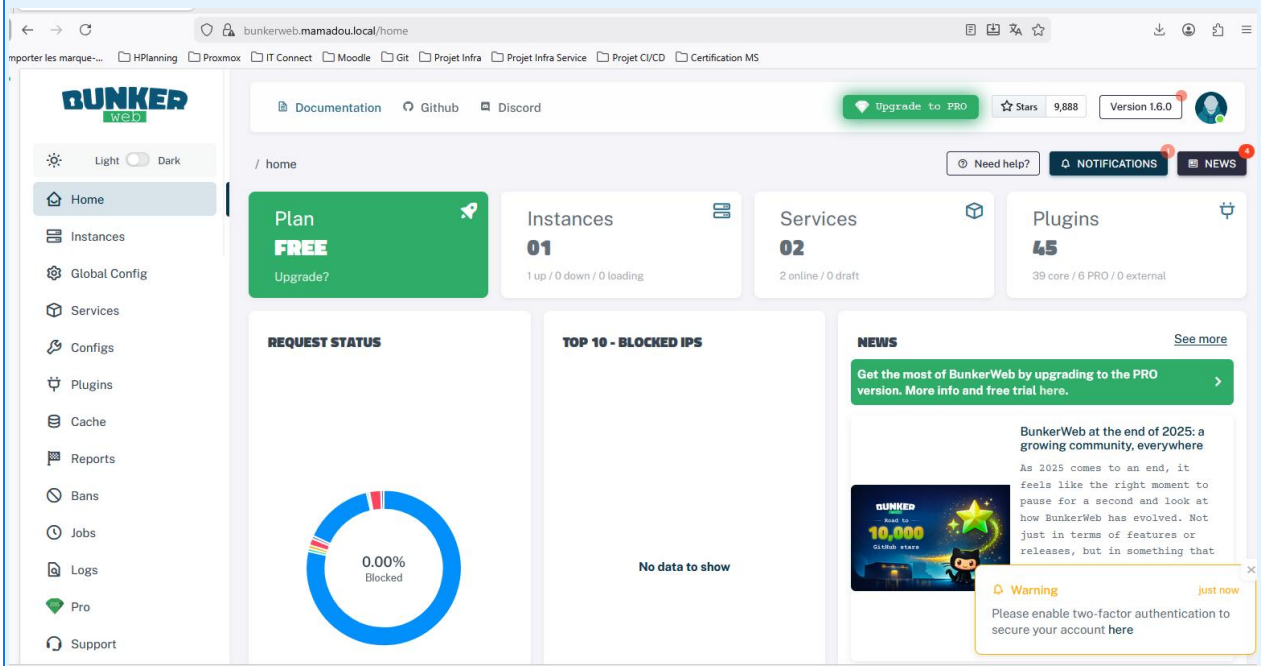
ⓘ Note : Lors de la première connexion, vous recevrez un avertissement de certificat car BunkerWeb utilise un certificat auto-signé par défaut. Ceci est normal.

Connectez-vous avec les identifiants définis dans docker-compose.yml :

- Nom d'utilisateur : **admin**
- Mot de passe : *celui que vous avez défini*



Une fois connecté, vous arrivez sur la page d'accueil de l'interface Web.

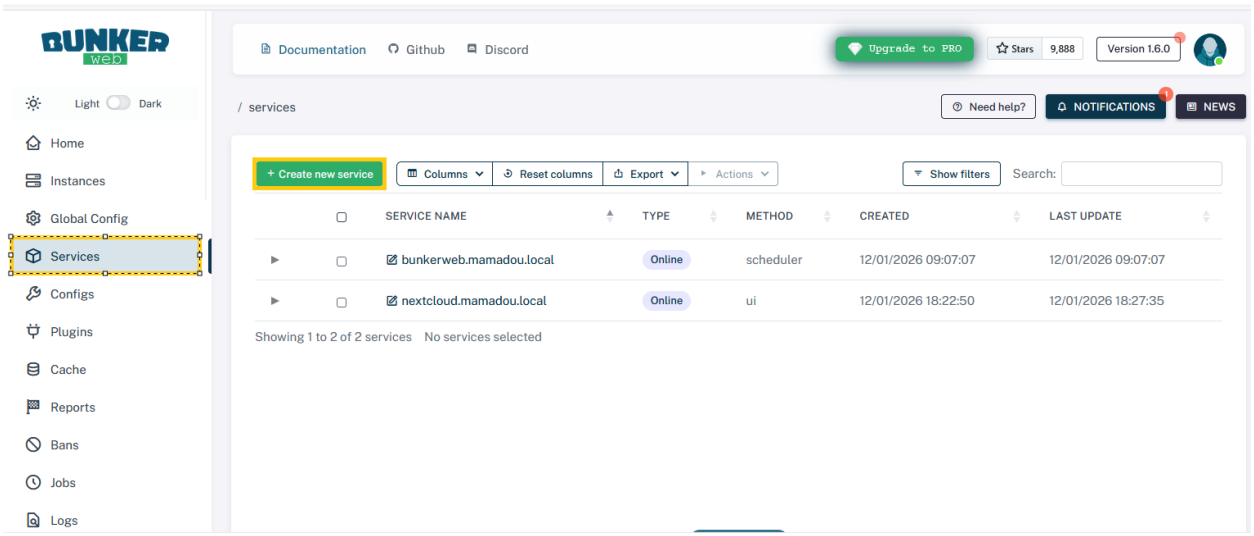


7. Configuration d'un service

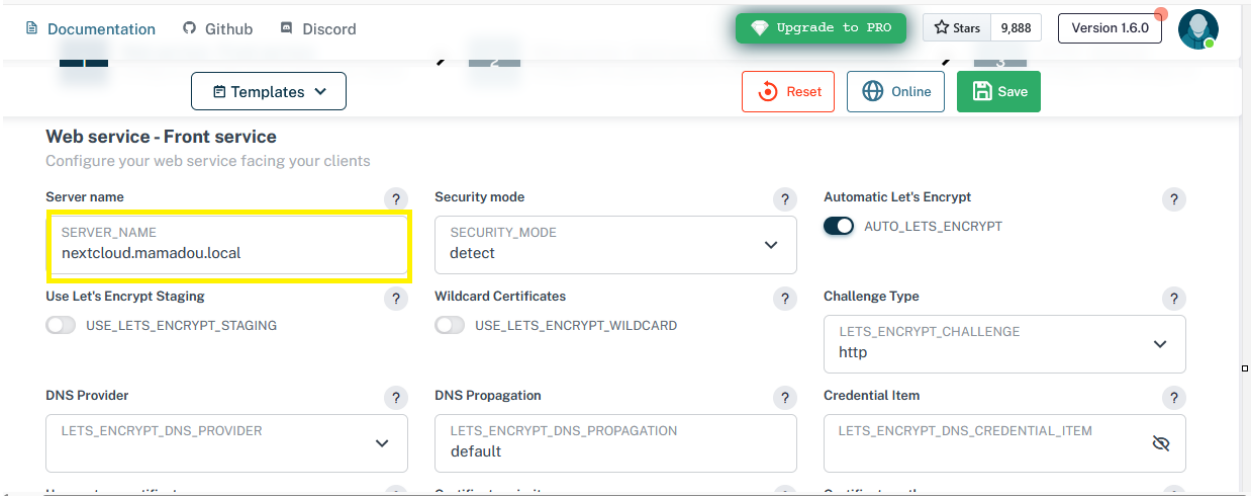
Un "service" dans **BunkerWeb** représente une application Web que vous souhaitez publier et protéger. Voici comment configurer votre premier service en tant que reverse proxy.

• Créer un nouveau service

- Dans le menu de navigation, cliquez sur "Services"
- Cliquez sur le bouton "**NEW SERVICE**"

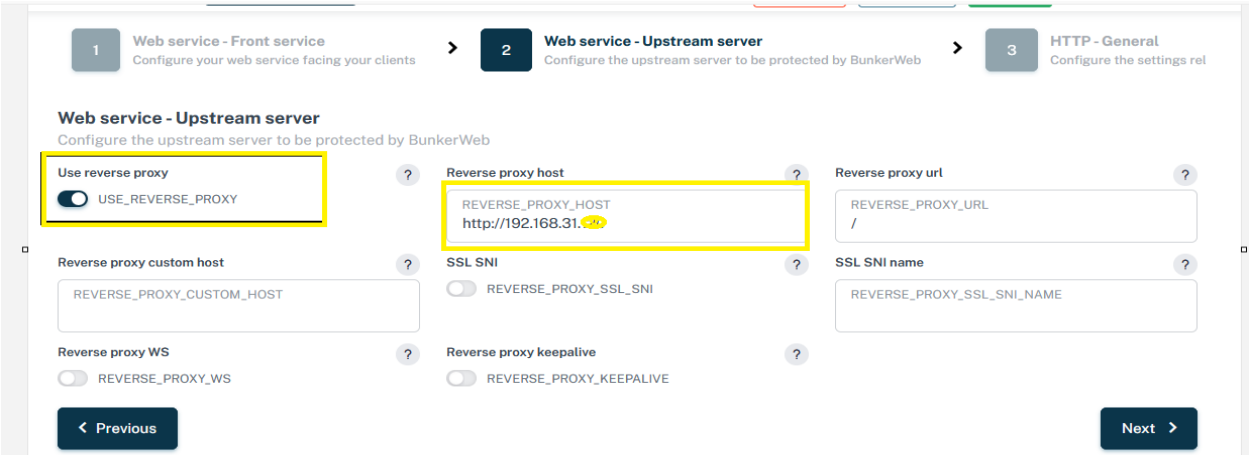


- Configurez le champ "Server name" avec le nom de domaine de votre application (ex: **nextcloud.mamadou.local**)



- ## Configurer le reverse proxy

- Cliquez sur l'onglet "GENERAL"
- Recherchez le plugin "Reverse proxy" et cliquez dessus
- Cochez la case "Use reverse proxy"
- Indiquez l'URL du serveur de destination (ex: http://192.168.X.Z:8080)

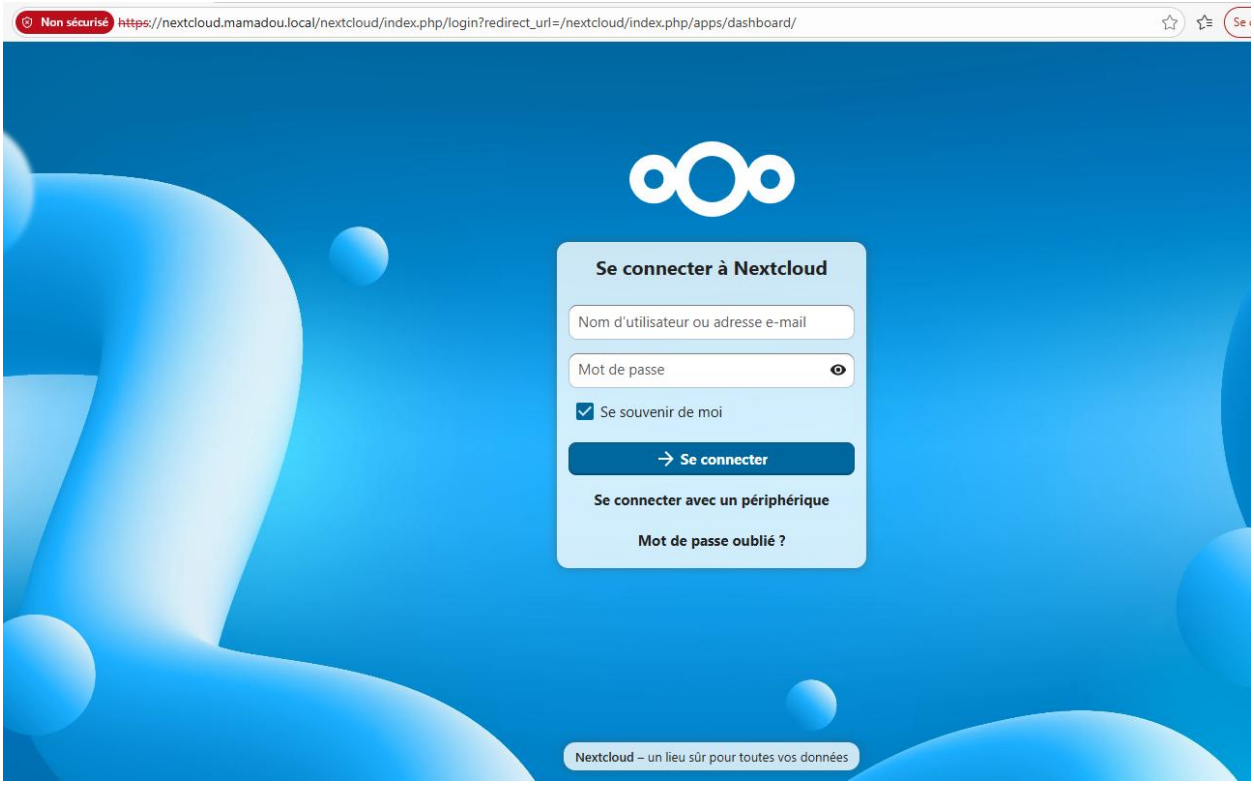


- ## Enregistrer la configuration

Cliquez sur le bouton "SAVE" en bas de la page. Votre service est maintenant créé et protégé par les plugins de sécurité activés par défaut (ModSecurity, Bad Behavior, etc.).

On peut maintenant tester notre service, si vous faites un test avec un site en production, avant de modifier l'enregistrement DNS je vous conseille d'utiliser le fichier hosts de votre ordinateur.

Depuis un navigateur accéder à votre application, il est possible que vous ayez une erreur de certificat car par défaut il va utiliser un certificat autosigné.


Non sécurisé
https://nextcloud.mamadou.local/nextcloud/index.php/login?redirect_url=/nextcloud/index.php/apps/dashboard/
Se connecter à Nextcloud
Nom d'utilisateur ou adresse e-mail
Mot de passe
Se souvenir de moi
Se connecter
Se connecter avec un périphérique
Mot de passe oublié ?
Nextcloud – un lieu sûr pour toutes vos données

8. Plugins de sécurité activés par défaut

Plugin	Protection assurée
ModSecurity	WAF qui analyse et bloque les requêtes malveillantes (injections SQL, XSS, etc.)
Bad Behavior	Détecte et bloque les bots malveillants et comportements suspects
Antibot	Challenge Captcha pour vérifier que l'utilisateur est humain
DNSBL	Vérifie les IP des clients contre des listes noires publiques
Limit	Limite le taux de requêtes pour prévenir les attaques par déni de service (DoS)

⚠ Note : La configuration par défaut de BunkerWeb est assez stricte. Si vous rencontrez des blocages légitimes, consultez les logs et ajustez les paramètres des plugins LIMIT, BAD BEHAVIOR et MODSECURITY.

9. Astuces et dépannage

• Afficher les logs Web (access_log)

Pour consulter les logs Web en temps réel :

```
sudo docker compose logs -f bunkerweb
```

• Désactiver la redirection HTTP vers HTTPS

Si vous souhaitez temporairement désactiver la redirection automatique :

- Accédez à la configuration du service concerné
- Allez dans le plugin "MISCELLANEOUS"
- Décochez les cases "Redirect HTTP to HTTPS" et "Auto redirect HTTP to HTTPS"

• Configurer un certificat Let's Encrypt

Pour obtenir automatiquement un certificat SSL gratuit :

- Dans la configuration du service, accédez au plugin "LET'S ENCRYPT"
- Cochez la case "Automatic Let's Encrypt"
- Indiquez votre adresse email

• Accès au service bloqué

Si un accès légitime est bloqué :

- Consultez les logs pour identifier la cause du blocage
- Ajustez les paramètres des plugins LIMIT, BAD BEHAVIOR et MODSECURITY
- Envisagez d'ajouter l'adresse IP concernée à la whitelist si nécessaire

• Commandes Docker utiles

Commande	Description
docker compose ps	Afficher l'état des conteneurs
docker compose logs -f	Suivre les logs en temps réel
docker compose restart	Redémarrer tous les services
docker compose down	Arrêter et supprimer les conteneurs
docker compose pull	Mettre à jour les images

10. Checklist de sécurité avant production

• Configuration de base

- ✓ Mots de passe MySQL modifiés (3 occurrences dans docker-compose.yml)
- ✓ Mot de passe administrateur BunkerWeb modifié
- ✓ API_WHITELIST_IP configuré avec votre réseau (192.168.31.0/24)
- ✓ SERVER_NAME configuré avec votre nom de domaine

• Certificats SSL/TLS

- ✓ Certificats SSL configurés (Let's Encrypt ou personnalisés)
- ✓ Redirection HTTP vers HTTPS activée

• Sauvegardes et maintenance

- ✓ Sauvegardes du répertoire ./bw-data/ planifiées
- ✓ Sauvegarde de la configuration **docker-compose.yml**
- ✓ Mise à jour régulière des images Docker planifiée

• Sécurité réseau

- ✓ Firewall système configuré (UFW, iptables, etc.)
- ✓ Ports 80 et 443 ouverts uniquement pour les réseaux autorisés
- ✓ SSH sécurisé (clés, port non-standard, etc.)

• Surveillance

- ✓ Logs de sécurité activés et surveillés
- ✓ Alertes configurées pour les événements de sécurité

11. Ressources et aller plus loin

• Documentation officielle

- Documentation BunkerWeb : https://docs.bunkerweb.io/
- GitHub officiel : https://github.com/bunkerity/bunkerweb
- Exemples de déploiement : https://github.com/bunkerity/bunkerweb/tree/master/examples

• Tutoriels avancés à venir

- Configuration de certificats personnalisés
- Intégration avec CrowdSec pour une protection collaborative
- Configuration avancée des paramètres de sécurité
- Monitoring et alertes avec Prometheus/Grafana

• Mise à jour de BunkerWeb

Pour mettre à jour BunkerWeb vers la dernière version :

```
cd /containers/bunkerweb sudo docker compose pull sudo docker compose up -d
```

12. Conclusion

BunkerWeb est une solution puissante et complète pour sécuriser vos applications Web. Ce guide nous a permis de déployer une instance fonctionnelle avec Docker et de configurer nos premiers services protégés.

N'oubliez pas de :

- Modifier tous les mots de passe par défaut
- Configurer correctement **l'API_WHITELIST_IP**
- Mettre en place des certificats SSL/TLS
- Surveiller les logs régulièrement
- Maintenir le système à jour

Bonne protection avec BunkerWeb ! 🌓