

The Wayback Machine - <https://web.archive.org/web/20220508112821/https://www.c...>



- ACCUEIL
- CATÉGORIES
 - [PROGRAMMATION EN C](#)
 - [INFORMATIQUE](#)
 - [OUTILS DE PROGRAMMATION](#)
 - [PROJETS 42](#)
- À PROPOS
- CONTACT
-

-
-

Born2beroot 03 : installer WordPress sur un serveur Debian

Par [Mia Combeau](#)

Dans [Projets 42](#)

13 mars 2022

11 min de lecture

[Laisser un commentaire](#)

B

Pour la partie bonus du projet Born2beroot, il nous faut installer WordPress sur notre serveur virtuel, ainsi que

mettre en place un autre service de notre choix. Afin de créer un site WordPress fonctionnel, nous devons installer un serveur HTTP, un gestionnaire de bases de données et le PHP. Pour le service de notre choix, nous allons installer et configurer Fail2ban par mesure de sécurité.

Dans cette série d'articles, notre serveur virtuel tourne sous Debian 11.2 (bullseye) avec VirtualBox 6.1.

Born2beroot : [Installation](#) | [Configuration](#) | [Bonus](#) | [Sujet \[pdf\]](#)

Table des matières

[Qu'est-ce que WordPress ?](#)

[Installer PHP pour WordPress](#)

[Installer lighttpd, un serveur web pour WordPress](#)

[Tester le serveur lighttpd](#)

[Activer FastCGI](#)

[Installer MariaDB, un gestionnaire de bases de données pour WordPress](#)

[Installer WordPress](#)

[Installer Fail2ban](#)

[Sources et lectures complémentaires](#)

Qu'est-ce que WordPress ?

WordPress est un système de gestion de contenu (SGC ou CMS, *content management system* en anglais) qui permet à n'importe qui, n'importe où de créer et d'entretenir un site internet fonctionnel, sans prérequis techniques. Sa simplicité d'usage, sa nature open source et gratuite, ainsi que sa capacité d'adaptation selon les besoins de ses utilisateurs en a fait un grand succès : 43% de tous les sites internet l'utilisent [\[source\]](#).

Pour que notre serveur Born2beroot puisse héberger un site WordPress, il nous faudra trois choses : un logiciel de serveur HTTP, un gestionnaire de bases de données, et le PHP. On devra de plus apporter quelques modifications à notre pare-feu pour permettre la communication via certains ports.

Installer PHP pour WordPress

Le **PHP** (*PHP: Hypertext Preprocessor*), c'est un langage de programmation libre très populaire pour la création de pages web dynamiques via un serveur HTTP. Il est indispensable au bon fonctionnement de WordPress.

Un paquet ne suffit pas pour l'installation de PHP. Au minimum, on va devoir en installer quatre : `php-common`, `php-cgi`, `php-cli` et `php-mysql`. Le problème c'est que APT et Aptitude n'ont que les dépôts de PHP version 7.4. Or, on va évidemment vouloir la dernière version de PHP (8.1 au moment de la rédaction). Il nous faudra donc indiquer un autre dépôt à notre gestionnaire de paquets.


C'est le dépôt de Sury qu'il nous faut. Pour le récupérer, nous allons devoir installer `cURL`, un simple outil en ligne de commande qui permet d'accéder à un URL sans avoir à passer par un navigateur web :

```
$ sudo apt update
$ sudo apt install curl
$ sudo curl -sSL https://packages.sury.org/php
$ sudo apt update
```



Maintenant qu'APT a le dépôt, il suffit de lancer l'installation de PHP, ici la version 8.1, et puis des paquets qu'il nous faut :

```
$ sudo apt install php8.1
$ sudo apt install php-common php-cgi php-cli
```



Pour vérifier la version PHP sur le système Born2beroot, faisons la commande suivante :

```
$ php -v
```

Installer lighttpd, un serveur web pour WordPress

A présent, il nous faut installer un logiciel de serveur web qui répond aux requêtes HTTP, le protocole réseau créé pour distribuer du contenu web.

Le serveur HTTP open source que nous devons choisir ici est **lighttpd** (prononcé « **lighty** »). Avec une plus petite empreinte mémoire que les autres serveurs web (comme Apache) et une gestion intelligente de la charge CPU, lighttpd est optimisé pour la rapidité, tout en restant sécurisé, conforme et flexible.

Cependant, il est fort possible qu'Apache a été installé sur le serveur comme dépendance pour l'un des modules PHP. Pour éviter les conflits entre notre serveur web lighttpd et Apache, on va tout d'abord voir si Apache est installé, et si c'est le cas, le désinstaller :

```
$ systemctl status apache2
$ sudo apt purge apache2
```

Une fois Apache désinstallé, nous pouvons installer lighttpd :

```
$ sudo apt install lighttpd
```

Ensuite, il nous faut le lancer, l'activer et vérifier sa version et son statut avec les commandes suivantes :

```
$ sudo lighttpd -v
$ sudo systemctl start lighttpd
$ sudo systemctl enable lighttpd
$ sudo systemctl status lighttpd
```

Son statut devait montrer qu'il est actif. Tout ce qu'il nous reste à faire, c'est d'autoriser le trafic HTTP dans notre pare-feu :

```
$ sudo ufw allow http
$ sudo ufw status
```

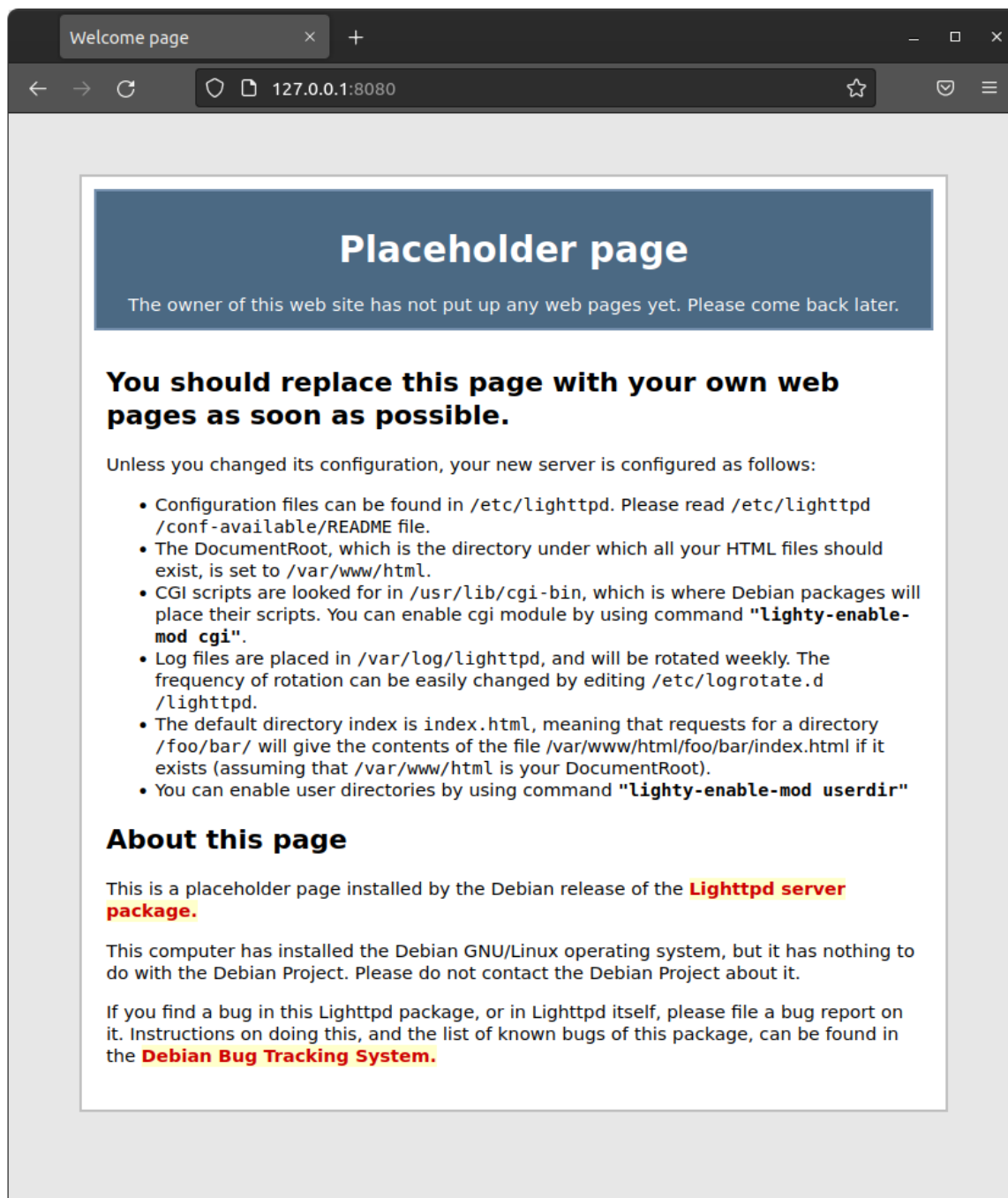
En vérifiant que la règle a bien été prise en compte, on devrait voir le port 80 autorisé. Le port 80, c'est le port HTTP par défaut. Il faut aussi faire une redirection de port dans VirtualBox pour pouvoir accéder au port 80 de la machine virtuelle depuis l'extérieur, comme on l'a fait précédemment pour le port 4242 :

- **Configuration >> réseau >> Adapter 1 >> Avancé >> Redirection de ports**
- Ajouter la règle **Port hôte: 8080, port invité: 80**, vu qu'on ne veut pas affecter le port 80 de l'hôte.



Tester le serveur lighttpd

Enfin, on peut faire un test pour voir si le serveur lighttpd marche correctement. Dans un navigateur dans la machine hôte, on peut se connecter à l'adresse et au port suivants : **http://127.0.0.1:8080** (ou **http://localhost:8080**). On devrait voir la page d'accueil de lighttpd, comme ceci :



On remplacera cette page avec un site WordPress sous peu !

Faisons un deuxième test rapide. Dans la machine virtuelle, créons un fichier nommé **info.php** dans le répertoire **/var/www/html** comme ceci :

```
$ sudo nano /var/www/html/info.php
```

Dans ce fichier, on va écrire un petit script pour montrer les informations de PHP sur le serveur :

```
1 | <?php
```

```
2 | phpinfo();  
3 | ?>
```

Maintenant, dans notre navigateur côté hôte, on peut aller voir ce fichier à l'adresse : **http://127.0.0.1/info.php**.

Sauf qu'on se retrouve avec une erreur « 403 Forbidden »... Alors que ce passe-t-il?

Activer FastCGI

A l'aube du World Wide Web, un serveur HTTP envoyait directement des pages pré-écrites en HTML à chaque requête. Aujourd'hui avec la technologie CGI et les pages web dynamiques, le serveur web ne répond pas directement mais transfère les données de la requête HTTP à une application externe. L'application traite la demande et renvoie le code HTML généré au serveur web qui peut ensuite répondre à la requête.

FastCGI (***Fast Common Gateway Interface*** ou « interface passerelle commune rapide ») est un protocole binaire qui permet à un serveur HTTP d'interagir avec des applications externes, comme dans notre cas, le PHP. C'est ce protocole entre lighttpd et PHP que nous devons mettre en place pour pouvoir accéder à notre page `info.php` depuis un navigateur web.

On va donc activer les modules FastCGI de lighttpd avec les commandes suivantes :

```
$ sudo lighty-enable-mod fastcgi  
$ sudo lighty-enable-mod fastcgi-php  
$ sudo service lighttpd force-reload
```

Maintenant, on devrait voir une page comme ceci à l'adresse **http://127.0.0.1:8080/info.php** :



PHP Version 8.1.3

System	Linux mcombeau42 5.10.0-12-amd64 #1 SMP Debian 5.10.103-1 (2022-03-07) x86_64
Build Date	Feb 23 2022 16:07:16
Build System	Linux
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/cgi
Loaded Configuration File	/etc/php/8.1/cgi/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/cgi/conf.d
Additional .ini files parsed	/etc/php/8.1/cgi/conf.d/10-mysqlnd.ini, /etc/php/8.1/cgi/conf.d/10-opcache.ini, /etc/php/8.1/cgi/conf.d/10-pdo.ini, /etc/php/8.1/cgi/conf.d/20-calendar.ini, /etc/php/8.1/cgi/conf.d/20-ctype.ini, /etc/php/8.1/cgi/conf.d/20-exif.ini, /etc/php/8.1/cgi/conf.d/20-ffi.ini, /etc/php/8.1/cgi/conf.d/20-fileinfo.ini, /etc/php/8.1/cgi/conf.d/20-ftp.ini, /etc/php/8.1/cgi/conf.d/20-gettext.ini, /etc/php/8.1/cgi/conf.d/20-iconv.ini, /etc/php/8.1/cgi/conf.d/20-mysqli.ini, /etc/php/8.1/cgi/conf.d/20-pdo_mysql.ini, /etc/php/8.1/cgi/conf.d/20-phar.ini, /etc/php/8.1/cgi/conf.d/20-posix.ini, /etc/php/8.1/cgi/conf.d/20-readline.ini, /etc/php/8.1/cgi/conf.d/20-shmop.ini, /etc/php/8.1/cgi/conf.d/20-sockets.ini, /etc/php/8.1/cgi/conf.d/20-sysvmsg.ini, /etc/php/8.1/cgi/conf.d/20-sysvsem.ini, /etc/php/8.1/cgi/conf.d/20-sysvshm.ini, /etc/php/8.1/cgi/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902.NTS
PHP Extension Build	API20210902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
 Zend Engine v4.1.3, Copyright (c) Zend Technologies
 with Zend OPcache v8.1.3, Copyright (c), by Zend Technologies

zend®engine

Installer MariaDB, un gestionnaire de bases de données pour WordPress

WordPress stocke le contenu d'un site internet dans une base de données. **MariaDB** est un gestionnaire de base de données libre et gratuit, fondé sur MySQL. Pour l'installer, il suffit de faire :

```
$ sudo apt install mariadb-server
```


Ensuite, on va lancer MariaDB, l'activer au démarrage du système et vérifier son statut :

```
$ sudo systemctl start mariadb
$ sudo systemctl enable mariadb
$ systemctl status mariadb
```

On devrait voir que MariaDB est actif. Il faut encore sécuriser son installation avec la commande :

```
$ sudo mysql_secure_installation
```

On nous posera alors plusieurs questions pour configurer les paramètres de sécurité de MariaDB (ici, root ne désigne pas l'utilisateur root de notre machine mais celui du gestionnaire de !) :

```
Enter current password for root (enter for non
Switch to unix_socket authentication [Y/n]: Y
Set root password? [Y/n]: Y
New password: 101Asterix!
Re-enter new password: 101Asterix!
Remove anonymous users? [Y/n]: Y
Disallow root login remotely? [Y/n]: Y
Remove test database and access to it? [Y/n]:
Reload privilege tables now? [Y/n]: Y
```

Ensuite, on doit redémarrer le service MariaDB :

```
$ sudo systemctl restart mariadb
```

Maintenant que MariaDB est correctement installé, il nous faut configurer une nouvelle base de données pour notre site WordPress.

```
$ mysql -u root -p
```

On nous demandera alors notre mot de passe root pour MariaDB (pas le mot de passe root de la machine virtuelle !). Enfin, on peut créer notre base de données pour WordPress avec des commandes en SQL :

```
MariaDB [(none)]> CREATE DATABASE wordpress_db
MariaDB [(none)]> CREATE USER 'admin'@'localhost'
```

```
MariaDB [(none)]> GRANT ALL ON wordpress_db.*  
MariaDB [(none)]> FLUSH PRIVILEGES;  
MariaDB [(none)]> EXIT;
```

Maintenant, si on retourne dans MariaDB avec la commande précédente **mysql -u root -p** et qu'on fait :

```
MariaDB [(none)]> show databases;
```

On devrait voir quelque chose comme ceci :

```
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| wordpress_db |  
+-----+
```

Notre base de données **wordpress_db** est bien présente.

Installer WordPress

Avant de pouvoir installer WordPress sur notre serveur virtuel Born2beroot, il nous faut avoir les paquets **wget** pour télécharger depuis un serveur web et **tar** pour décompresser un fichier.

```
$ sudo apt install wget  
$ sudo apt install tar
```

Ensuite, on va télécharger l'archive de la dernière version de WordPress depuis le site officiel pour l'extraire et mettre son contenu dans le répertoire **/var/www/html**. Puis on va supprimer l'archive et le dossier d'extraction.

```
$ wget http://wordpress.org/latest.tar.gz  
$ tar -xzf latest.tar.gz  
$ sudo mv wordpress/* /var/www/html/  
$ rm -rf latest.tar.gz wordpress/
```

Il nous faut un fichier de configuration pour WordPress.
Un exemple est inclus dans nos fichiers, renommons-le et modifions-le :

```
$ sudo mv /var/www/html/wp-config-sample.php /  
$ sudo nano /var/www/html/wordpress/wp-config.
```

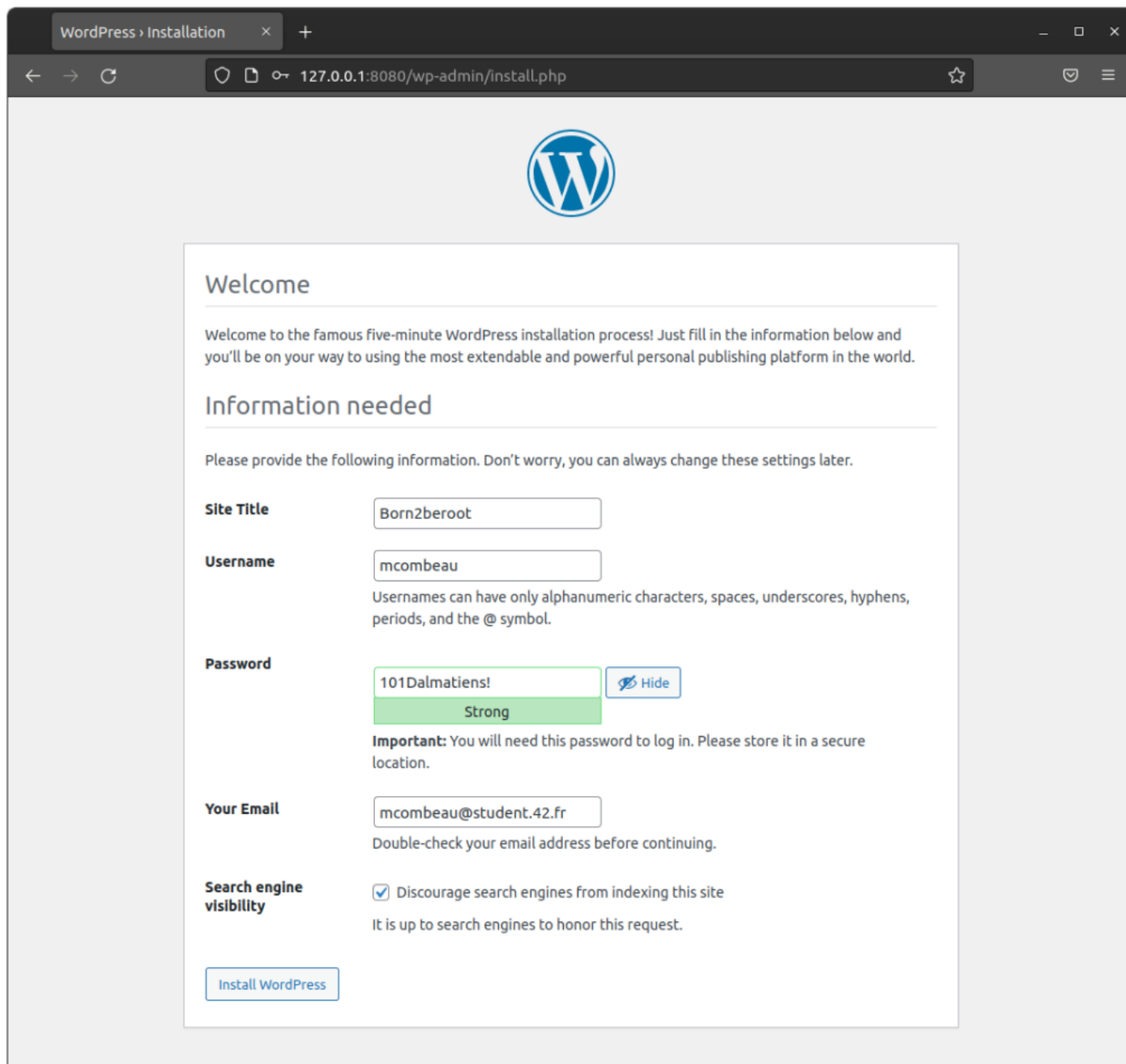
Ici, on veut changer les paramètres de la base de données pour diriger WordPress vers celle qu'on a créée avec MariaDB.

```
<?php  
/* ... */  
/** The name of the database for WordPress */  
define( 'DB_NAME', 'wordpress_db' );  
  
/** Database username */  
define( 'DB_USER', 'admin' );  
  
/** Database password */  
define( 'DB_PASSWORD', 'WPpassw0rd' );  
  
/** Database host */  
define( 'DB_HOST', 'localhost' );
```

On doit aussi changer les permissions des répertoires WordPress pour l'utilisateur www-data (le serveur HTTP) et redémarrer lighttpd :

```
$ sudo chown -R www-data:www-data /var/www/htm  
$ sudo chmod -R 755 /var/www/html/  
$ sudo systemctl restart lighttpd
```

Enfin, on peut se connecter dans le navigateur de l'hôte à l'adresse **http://127.0.0.1:8080** pour accéder au menu de configuration de notre site WordPress.



WordPress » Installation

127.0.0.1:8080/wp-admin/install.php

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title

Username
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password
Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email
Double-check your email address before continuing.

Search engine visibility ☒ Discourage search engines from indexing this site
It is up to search engines to honor this request.

Et voilà ! Une fois l'installation terminée, on peut se connecter et customiser notre nouveau site comme on veut. Tout est possible !

Installer Fail2ban

Fail2ban est un logiciel qui analyse les journaux d'un serveur pour identifier et interdire les adresses IP suspectes. S'il trouve de multiples tentatives de connexion échouées ou des attaques automatisées provenant d'une adresse IP, il peut bloquer celle-ci avec le pare-feu, soit temporairement soit définitivement.

C'est ce service qu'on va installer ici pour le second bonus de Born2beroot. On va ensuite lancer le programme et l'activer au démarrage, ainsi que vérifier son statut.

```
$ sudo apt install fail2ban
$ sudo systemctl start fail2ban
$ sudo systemctl enable fail2ban
$ sudo systemctl status fail2ban
```

Il va ensuite falloir créer et modifier le fichier **/etc/fail2ban/jail.local** pour configurer ses paramètres.

```
$ sudo cp /etc/fail2ban/jail.conf /etc/fail2ba
$ sudo nano /etc/fail2ban/jail.local
```

Pour appliquer Fail2ban aux connexions SSH, il faut ajouter quelques lignes au fichier, dans la section « SSH servers » qui commence à la ligne 279 du fichier :

```
#
# SSH servers
#

[sshd]

# To use more aggressive sshd modes set filter
# normal (default), ddos, extra or aggressive
# See "tests/files/logs/sshd" or "filter.d/ssh
# mode    = normal
enabled   = true
maxretry  = 3
findtime  = 10m
bantime   = 1d
port      = 4242
logpath   = %(sshd_log)s
backend   = %(sshd_backend)s
```

Afin de voir les tentatives de connexion échouées et les adresses IP interdites, il suffit de lancer les commandes suivantes :

```
$ sudo fail2ban-client status
$ sudo fail2ban-client status sshd
```

```
$ sudo tail -f /var/log/fail2ban.log
```

Pour tester le fait que Fail2ban interdit bien les adresses IP, on peut changer le bantime à une valeur plus permissive, comme 30m, dans le fichier de configuration `/etc/fail2ban/jail.local`. Ensuite on peut tenter de se connecter plusieurs fois via SSH depuis le terminal de la machine hôte, mais avec le mauvais mot de passe. Après quelques essais, la connexion devrait se voir refusée et la commande `fail2ban-client status sshd` devrait nous montrer l'adresse IP interdite.

Et voilà tout pour les bonus de Born2beroot !

Sources et lectures complémentaires

- WordPress, *Before You Install* [[WordPress.org](https://wordpress.org)]
- Lighttpd [[site officiel](#)]
- MariaDB [[site officiel](#)]
 - *mysql_secure_installation* [[MariaDB](#)]
 - *MariaDB Server Documentation* [[MariaDB](#)]
- Richard, *Creating New MySQL User and Database for WordPress* [[Website for Students](#)]
- PHP [[site officiel](#)]
- Ondřej Surý, *deb.sury.org: Frequently Asked Questions* [[GitHub](#)]
- WordPress, *How to Install WordPress* [[WordPress.org](https://wordpress.org)]
- SuperHosting, *What is CGI, FastCGI?* [[SuperHosting.bg](#)]
- Abhishek Prakash, *Secure Your Linux Server With Fail2Ban [Beginner's Guide]* [[Linux Handbook](#)]

[base de données cursus 42](#) [Debian Fail2ban guide](#)
[machine virtuelle ordinateur](#) [PHP serveur web](#) [WordPress](#)

A PROPOS DE L'AUTEUR

Mia Combeau

Étudiante à 42Paris, exploratrice du monde numérique. Je code ici à la norme de l'école 42, donc les boucles for, les switch, les opérateurs ternaires et plein d'autres choses me sont hors de portée... pour l'instant !

[AUTRES ARTICLES](#)

LAISSER UN COMMENTAIRE

Commentaire

Nom *

E-mail *

Site web

☐ Enregistrer mon nom, mon e-mail et mon site dans le navigateur pour mon prochain commentaire.

[ENVOYER LE COMMENTAIRE](#)

LIRE LA SUITE

Binaire 010 : l'utilité des opérations bitwise et du bit shifting

7 mai 2022

B

Pipex : reproduire l'opérateur pipe « | » en C

2 avril 2022

P

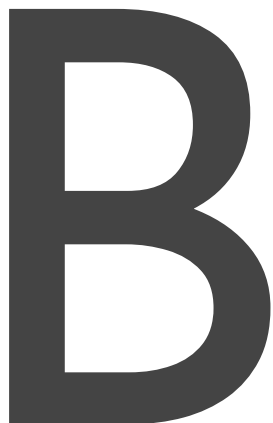
Born2beroot 02 : configurer un serveur virtuel Debian

11 mars 2022

B

Born2beroot 01 : créer une machine virtuelle Debian

9 mars 2022



Par Mia Combeau 13 mars 2022

Remonter

Pipex : reproduire l'opérateur pipe « | » en C

Remonter

Born2beroot 02 : configurer un serveur virtuel Debian

MENU

- Mentions légales
- Contact

CATÉGORIES

- Informatique
- Programmation en C
- Projets 42



Cette œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International.



- ACCUEIL
- CATÉGORIES
 - PROGRAMMATION EN C
 - INFORMATIQUE
 - OUTILS DE PROGRAMMATION
 - PROJETS 42
- À PROPOS
- CONTACT
-

Tapez ici pour rechercher...

RECHERCHER

CATÉGORIES

- [Informatique](#)
- [Programmation en C](#)
- [Projets 42](#)

MIA COMBEAU

Étudiante à 42Paris, exploratrice du monde numérique. Je code ici à la norme de l'école 42, donc les boucles for, les switch, les opérateurs ternaires et plein d'autres choses me sont hors de portée... pour l'instant !

- [github](#)
- [linkedin](#)