IUT ANNECY
UNIVERSITÉ SAVOIE MONT BLANC

INFO

INFORMATIQUE

B.U.T. 1

Second Semester

Year 2025-26

SAÉ 2.03 | Web server

Installation Guide

Authors:

GONZALEZ PAULO Rémy

MAUHIN Théo

Guide version: 1.00

To follow correctly this guide, you will need a Raspberry Pi 2, a SD card of at least 8GB storage, and an internet connection.

Legal Notice & Copyright

Table of contents :

1. Using this guide :

1.1. Purpose:

This guide aims to help you set up an optimized web server on a Raspberry Pi 2 for hosting WordPress sites.

1.2. Scope:

This guide covers the full process of turning a Raspberry Pi 2 into a secure, optimized web server to self-host a WordPress website.

Specifically, this guide includes:

- Installing and performing the initial setup of Raspberry Pi OS Lite.
- Securing the server environment using UFW (firewall) and Fail2ban.
- Deploying a complete web stack with Nginx, PHP 8.4, and MariaDB.
- Installing and configuring WordPress.
- Optimizing page load times and server performance using caching techniques (W3 Total Cache).
- Conducting performance testing.

1.3. System Organization:

The system is organized around a LEMP stack architecture specifically optimized for the hardware constraints of a Raspberry Pi 2. At the foundational level, the device runs Raspberry Pi OS Lite, a headless operating system chosen to minimize resource consumption and maximize available memory. Network access and system security are strictly managed using SSH for remote administration, UFW as a firewall to allow only essential web and administrative traffic, and Fail2ban to actively protect the server against brute-force attacks. The core web environment relies on Nginx to serve incoming HTTP and HTTPS requests efficiently, working in conjunction with PHP FPM to execute dynamic scripts and MariaDB to securely manage the relational database.

On top of this lightweight infrastructure, WordPress operates as the main content management system. To ensure optimal performance and fast page load times on a low-resource device, the setup integrates the W3 Total Cache plugin directly with the Nginx configuration to serve static cached pages, deliberately avoiding database caching to preserve the limited RAM available on the Raspberry Pi while also preserving the limited CPU resources.

1.4. Technical support:

If you encounter any issues that cannot be resolved using this guide, please feel free to contact us at: sae203.contact@pro.dev

2. Installation procedures:

2.1. Describing the main steps:

The installation procedure is divided into several chronological phases to ensure a logical, stable, and secure setup process. It begins with the operating system installation, which consists of flashing the lightweight Raspberry Pi OS Lite onto the SD card. Once the system is

running, the next step covers the network and SSH configuration to enable secure remote administration.

With access established, the focus shifts to security enhancements by configuring the UFW firewall and installing Fail2ban to prevent brute-force attacks.

After securing the server, the guide details the LEMP stack setup, involving the installation of core components such as Nginx, MariaDB, and PHP 8.4. This creates the necessary environment for the subsequent WordPress deployment, where the Content Management System is downloaded, configured, and launched. Finally, the procedure addresses performance optimization by integrating the W3 Total Cache plugin to drastically improve page loading times, concluding with performance testing to validate the server's capabilities under load.

2.2. Before installing: Prerequisites

Before beginning the installation process, it is essential to gather all the necessary hardware and software components. You will need a Raspberry Pi 2 board and a micro SD card with a minimum capacity of 8GB to store the operating system and the WordPress environment. Since the Raspberry Pi 2 lacks built-in wireless connectivity, an Ethernet cable and a stable internet connection are required for downloading updates and software packages. An external HDMI monitor and an USB keyboard are also necessary for the initial setup.

Additionally, you must have access to a separate computer equipped with an SD card reader to prepare the boot media. On this computer, you should install the Raspberry Pi Imager software, which is available for download from the official Raspberry Pi website. Finally, ensure that this secondary computer is connected to the same local network as the Raspberry Pi to facilitate remote administration via SSH once the initial setup is complete.

2.3. Preparing to install

Before beginning this installation, verify that your computer is fully up to date and close any resource-intensive applications. You will also need the Raspberry Pi Imager software. If it is not already installed, please download it from the official website (https://www.raspberrypi.com/software/) and install it before proceeding.

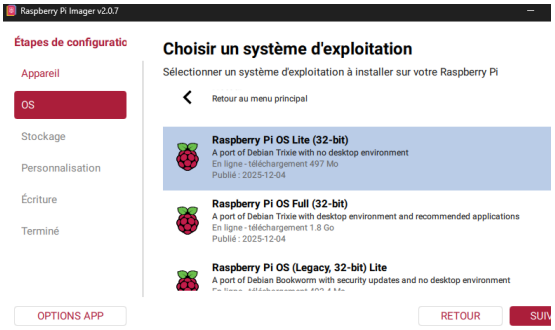
3. Installing Raspberry Pi OS Lite

3.1. Selecting the correct hardware

Open the Raspberry Pi Imager application. When prompted to select your target hardware, choose « Raspberry Pi 2 ». Then, press the continue button.

3.2. Selecting the correct OS

When prompted to select your target OS, click on “ Other ” and select “ Raspberry Pi OS Lite (32-bit) ”. Then, press the continue button.



3.3. Selecting the correct storage

On this page, select the correct storage for the installation of the OS.

WARNING: All data on the selected storage will be deleted. Ensure you select the right storage and that it contains no important files or data before continuing.

3.4. Naming the host

On the next screen, change the hostname to a name of your choice, then click "Continue".

3.5. Setting up timezone and keyboard layout

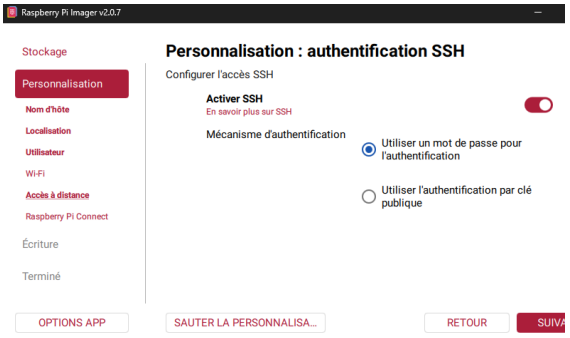
On the following screen, select your appropriate timezone, capital city, and keyboard layout, then click continue (Note: If your preferred keyboard layout is not listed, you will be able to configure it later in the system settings).

3.6. Defining username and password

On this screen, define a username and secure password for your primary system account, then click "Continue".

3.7. Activating SSH

Bypass the Wi-Fi setup screen by clicking "Continue", as the Raspberry Pi 2 relies on an Ethernet connection. On the SSH setup page, enable SSH and choose "Use password authentication". While password authentication is less secure than using SSH keys, it is simpler for this initial setup. Finally, click "Continue".



3.8. Finishing the installation

After configuring these settings, click "Continue" until you are prompted to write the image. Click the "Write" button to begin flashing the OS to your SD card. Please wait for the process to complete. Once the success message appears, you can safely remove the SD card, close the software, and proceed to the next step.

4. Configuring the network and SSH

4.1. Hardware setup and initial boot

With the operating system successfully written to the micro SD card, remove it from your computer and carefully insert it into the slot located on the underside of the Raspberry Pi 2. Before powering on the device, connect it to an external monitor using an HDMI cable and attach a USB keyboard to allow for local interaction. Additionally, ensure the Ethernet cable is securely connected to your router or switch to provide network access. Finally, plug in the power supply to turn on the Raspberry Pi. The system

will begin its initial boot sequence, and after a few moments, the command-line login prompt will appear on your screen, indicating that the device is ready for the network configuration.

4.2. Editing system configuration

Once successfully logged in, you can execute the " **sudo raspi-config** " command. This tool allows you to adjust your keyboard layout and other core system settings if necessary.

Important: Do not modify the network settings within this menu, as the network configuration will be addressed in the following step.

4.3. Configuring the network

To set up your network parameters, type the command " **sudo nmtui** " to open the NetworkManager Text User Interface. This tool provides a visual menu within your terminal where you can easily edit your connection settings.

For a web server, it is highly recommended to assign a static IP address rather than using a dynamic one, ensuring your server remains consistently accessible. Once your network configuration is applied and you have exited the tool, you can verify your connection and display your current IP address by typing the command " **ifconfig** ". Please note this IP address carefully, as you will need it to access your Raspberry Pi remotely via SSH in the following step.

4.4 Connecting remotely via SSH

Now that the Raspberry Pi is connected to the local network and its IP address is known, you can easily manage it remotely from your main computer without needing the external monitor anymore. On a Windows machine, open the Command Prompt or PowerShell. To establish the connection, type the command " **ssh username@ip_address** ", carefully replacing "username" with the account name you created during the OS installation and "ip_address" with the one you noted in the previous step. Upon your very first connection attempt, the system will prompt you to accept the server's security fingerprint, simply type "yes" to confirm and proceed. Finally, enter your user password when prompted to gain full command-line access to your server, allowing you to begin the software configuration.

5. Securing the server

5.1. Updating the operating system

Before anything else, you must update your OS. To do this, simply type the command " **sudo apt update && sudo apt upgrade -y** " in your terminal. It is recommended to reboot the device to load the updated core systems with the command " **sudo reboot** ".

5.2. Installing the firewall

With the operating system updated, the next step is to set up a firewall. The Uncomplicated Firewall (UFW) is an excellent, user-friendly tool for managing network traffic. To install it on your system, execute the command " **sudo apt install ufw** ". Then, you can verify if this is correctly installed by

typing the following command " **sudo ufw status** " (this should respond "status : inactive")

5.3. Configuring default firewall rules

For optimal security, it is highly recommended to block all incoming connections by default while allowing all outgoing ones. To apply this configuration, run the command " **sudo ufw default deny incoming** " followed by " **sudo ufw default allow outgoing** ".

5.4. Allowing essencials services

Now, you must manually allow the necessary services for your web server. Run the command " **sudo ufw allow 22** " to keep your remote SSH access active. Next, use the command " **sudo ufw allow http** " to open TCP port 80 for standard web traffic, and " **sudo ufw allow https** " to open TCP port 443 for secure connections.

5.5. Activating the firewall

Next, in order to activate the firewall, type the following command " **sudo ufw enable** "

5.6. Disabling SSH root login

As an additional security measure, it is highly recommended to disable remote SSH access for the default root account. To apply this restriction, edit the SSH configuration file by running the command " **sudo nano /etc/ssh/sshd_config.d/50-cloud-init.conf** ". Within this file, add the line " **PermitRootLogin no** " and save your changes. This configuration ensures that attackers cannot attempt to brute-force the most privileged account on your server. Before restarting the SSH server, we check if the configuration is valid using the command " **sudo sshd -t** " and then if it succeeds, we can restart the service by running the command " **sudo systemctl restart sshd** "

5.7. Installing Fail2ban

To further secure your server by blocking the IP addresses of brute-force attackers, you should install the Fail2ban software. To do this, simply execute the command " **sudo apt install fail2ban** " in your terminal.

5.8. Creating a custom Fail2ban configuration

To create a personalized configuration file, type the command " **sudo nano /etc/fail2ban/fail2ban.d/fail2ban.conf** ". Once the editor opens, you must add three specific lines to the file. First, type " **[DEFAULT]** ", followed by " **dbpurgeage = 1y** " on the next line, and finally " **action = ufw** " on the third line. Save your changes and close the file when you are done.

1.1. Securing SSH with Fail2ban jails

To actively protect your SSH service against unauthorized access, you must define specific ban rules. Create a new local jail file by executing the command " **sudo nano /etc/fail2ban/jail.local** ". Once the text editor opens, insert the configuration block provided below. This specific setup ensures that any user who enters an incorrect password four times within a ten minute window will be automatically banned for one full hour. Furthermore, it activates a dedicated filter to mitigate SSH-based DDoS

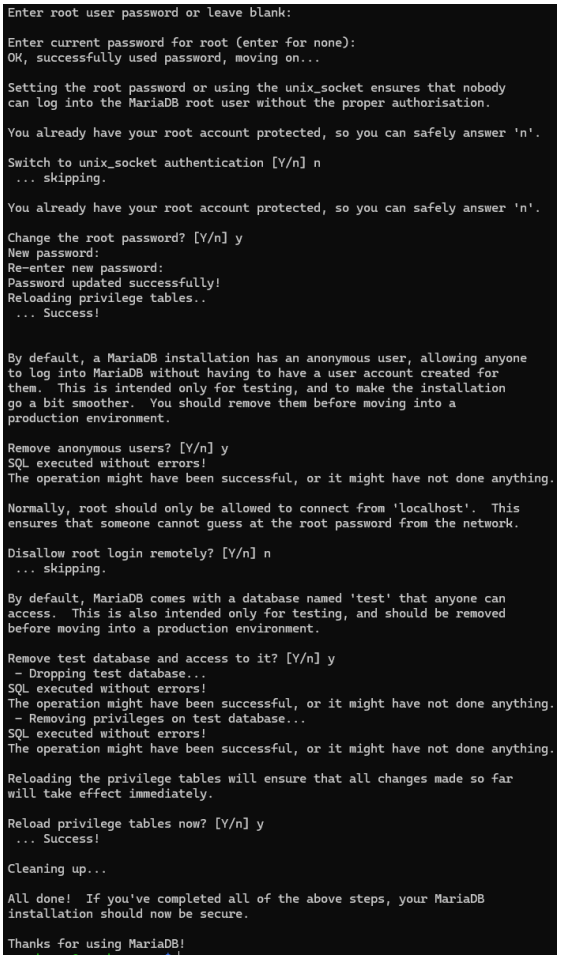
attacks. After pasting the configuration, save your changes and exit the editor.

[sshd]
enabled = true
port = 22
logpath = /var/log/auth.log
maxretry = 4
bantime = 3600
findtime = 600
[ssh-ddos]
enabled = true
port = ssh
filter = sshd
mode = ddos
logpath = %(sshd_log)s
backend = %(sshd_backend)s

6. MariaDB installation:

6.1. Installing and securing MariaDB

To install MariaDB and perform the initial setup, you must run two specific commands. First, type " **sudo apt install mariadb-server** " to download the package. Once installed, execute " **sudo mariadb-secure-installation** ". During this configuration process, answer the system prompts in this exact order: **N**, **Y**, **Y**, **N**, **Y**, **Y**, while entering the required information when asked, such as your new database root password. Upon completion, your terminal output should look similar to the following:



6.2. Enabling the MariaDB service

Once the security setup is complete, enable the service to start automatically at boot by running the command " **sudo systemctl enable --now mariadb** ". Next, apply all changes by restarting the service with the command " **sudo systemctl restart mariadb** ". The remainder of the MariaDB configuration, including the creation of databases and user accounts, will be covered later in this guide.

7. Installing Nginx and PHP 8.4

7.1. Installing Nginx dependencies

For performance and compatibility reasons, we will install the latest version of Nginx directly from the developer's repositories. First, type the following command to install the necessary dependencies: “ **sudo apt install curl gnupg2 ca-certificates lsb-release debian-archive-keyring** ”.

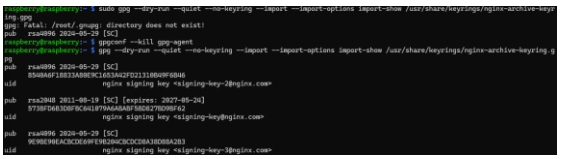
7.2. Importing the Nginx signing key

To download the packages safely, you must retrieve the signing key to verify the authenticity of the installer. Execute the first command below to download and store the key. Afterward, verify that the key has been successfully saved by running the second command.

```
curl https://nginx.org/keys/nginx_signing.key | gpg --dearmor | sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null
```

```
gpg --dry-run --quiet --no-keyring --import --import-options import-show /usr/share/keyrings/nginx-archive-keyring.gpg
```

If you receive a “ Directory does not exist ” error during this process, type the command “ **gpgconf --kill gpg-agent** ” to restart the GPG service and try again.



7.3. Adding the Nginx repository

Next, set up the apt repository for stable Nginx packages, you need to add it to your system's sources. Run the following command to do so :

```
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
https://nginx.org/packages/debian `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
```

7.4. Prioritizing the official Nginx packages

To prevent your system from accidentally installing older Nginx versions from the default OS repositories during updates, you must set up APT pinning. This configuration explicitly forces the package manager to prioritize the official developer packages you just added. Execute the command block below to apply this priority rule :

```
echo -e "Package: *\nPin: origin nginx.org\nPin: release o=nginx\nPin-Priority: 900\n" \
    | sudo tee /etc/apt/preferences.d/99nginx
```

7.5. Installing and starting Nginx

Install the package by running the following command: “ **sudo apt update && sudo apt install -y nginx** ”. Once the installation is done, enable and start the service with the command “ **sudo systemctl enable --now nginx** ”.

7.6. Removing the default configuration

To simplify the rest of the Nginx setup, we will remove the default site configuration. To do this, execute the

following command: “ **sudo rm /etc/nginx/sites-enabled/default** ”.

7.7 Installing and enabling PHP

Now that the web server is ready, you need to install PHP and its required extensions for WordPress. To do this, run the command “ **sudo apt install php-fpm php-mysql php-curl php-dom php-imagick php-mbstring php-zip php-gd php-intl** ”. Once the installation is complete, activate the service to start automatically by executing the command “ **sudo systemctl enable --now php8.4-fpm** ”.

7.8. Testing the Nginx and PHP integration

Now, we must verify that Nginx is correctly configured to process PHP files. To do this, create a new test configuration file by executing the command “ **sudo nano /etc/nginx/conf.d/test-php.conf** ”. Once the text editor opens, paste the following configuration block inside :

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

  root /var/www/html;
  index index.php index.html index.htm;

  server_name _;

  location / {
    try_files $uri $uri/ =404;
  }

  # PHP-FPM Configuration
  location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/run/php/php8.4-fpm.sock;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    include fastcgi_params;
  }

  # Block access to .htaccess files
  location ~ /\.ht {
    deny all;
  }
}
```

After saving and closing the file, it is highly recommended to check the syntax of your Nginx configuration to avoid any crashes. Run the following command : “ **sudo nginx -t** ”.

If everything is configured correctly, the system should return a message similar to this indicating success :

Finally, apply these new settings by reloading the Nginx service with the command “ **sudo systemctl reload nginx** ”.

7.9. Checking PHP info and cleaning up

Create a PHP test file showing all installed modules and the server configuration by typing the following command : “ **echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php** ”.

Now, open a web browser on your computer and navigate to the URL **http://YOUR_RASPBERRY_IP/info.php**

(make sure to replace "YOUR_RASPBERRY_IP" with your actual Raspberry Pi IP address) to verify that everything works correctly.

Once you have confirmed the configuration is functional, you must delete the test files for security reasons. Run the following commands to remove them :

```
sudo rm /etc/nginx/sites-enabled/test-php
sudo rm /var/www/html/info.php
```

8. Configuring WordPress

8.1. Creating the WordPress configuration

Create a dedicated Nginx configuration file for your WordPress installation by typing the command “ **sudo nano /etc/nginx/conf.d/wordpress.conf** ”. Once the editor opens, insert the following lines into the newly created file :

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

    root /var/www/html/wordpress;

    index index.php index.html index.htm index.nginx-debian.html;

    # catch-all is fine for now, should be changed to domain name later
    server_name _;

    # Limit max file upload to 100M for security and performance
    client_max_body_size 100M;

    # uncomment this line after the W3Cache installation
    # include /var/www/html/wordpress/nginx.conf;

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

    # pass PHP scripts to FastCGI server
    #
    location ~ \.php$ {
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
        fastcgi_pass unix:/run/php/php-fpm.sock;
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Permitted-Cross-Domain-Policies none;
```

```
    add_header X-Frame-Options "SAMEORIGIN";
  }

  # Don't log access to favicon.ico and robots.txt
  location = /favicon.ico {
    log_not_found off;
    access_log off;
  }

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

  # Deny access to hidden files
  location ~ /\.ht {
    deny all;
  }

  # Prevent PHP from running from upload directories
  location ~* /(?:uploads|files)/.*\.php$ {
    deny all;
  }

  # Caching and gzip configuration
  location ~* \.(jpg|jpeg|gif|png|webp|svg|woff|woff2|ttf|css|js|ico|xml)$ {
    expires 30d;
    log_not_found off;
  }

}
```

8.2. Activating the configuration

In order to make the configuration active, create a symbolic link to the " sites-enabled" folder using the following command: " **sudo ln -s /etc/nginx/sites-available/wordpress /etc/nginx/sites-enabled/wordpress** ". You can try the configuration with the command " **sudo nginx -t** ", and finally restart the server by typing the command " **sudo systemctl restart nginx** ".

8.3. Configuring PHP upload limits

To configure the PHP FPM upload limit, you need to modify its main configuration file. Run the following commands to automatically increase both the maximum post size and the maximum upload file size to 100 megabytes :

```
sudo sed -i "s/^post_max_size = 8M/post_max_size = 100M/g" /etc/php/8.4/fpm/php.ini
```

```
sudo sed -i "s/^upload_max_filesize = 2M/upload_max_filesize = 100M/g" /etc/php/8.4/fpm/php.ini
```

9. Configuring the database

9.1. Connecting to the MariaDB console

To configure the database for your WordPress installation, you first need to connect to the MariaDB database management system. Execute the following command to log in as the administrator: " **sudo mysql -u root -p** ". When prompted, enter the root password you created during the setup of MariaDB.

9.2. Creating the WordPress database and user

Now, you need to create a specific database for WordPress and a dedicated user with full permissions over it. Execute the following SQL commands one by one. Make sure to replace the

word 'password' with a strong, secure password of your choice :

CREATE DATABASE wordpress;
CREATE USER 'wordpress'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON wordpress.* TO 'wordpress'@'localhost';

10. Installing WordPress

10.1. Installing download and extraction tool

To install WordPress, you first need to download the software's zip archive to your server and then extract it. To prepare your system for this, install the necessary tools by running the following command " **sudo apt install curl unzip** ".

10.2. Downloading the WordPress archive

Download the WordPress archive into the Linux temporary folder using the curl command. To do this, simply execute the following command in your terminal : " **curl https://wordpress.org/latest.zip -o /tmp/wordpress.zip** "

10.3. Extracting the WordPress files

Now, you need to extract the downloaded WordPress files directly into the web server directory. To do this, simply execute the following command : " **sudo unzip -d /var/www/html /tmp/wordpress.zip** ".

10.4. Creating the WordPress configuration file

Navigate to the WordPress folder and make a copy of the sample configuration file to create your active wp-config.php. To do this, run the following command : " **cd /var/www/html/wordpress && sudo cp wp-config-sample.php wp-config.php** ".

10.5. Editing the WordPress configuration

Edit the configuration file to set the database name, username, and password. To do this, type the following command: " **sudo nano /var/www/html/wordpress/wp-config.php** ". Then, look for the database parameters within the file and replace them with your own credentials, like this :

```
// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** Database username */
define( 'DB_USER', 'wordpress' );

/** Database password */
define( 'DB_PASSWORD', 'password' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8mb4' );

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

10.6. Changing folder's owner

Finally, change the ownership of the WordPress folder to the www-data user and group to make the files accessible by Nginx. To do this, run the following command : " **sudo chown -R www-data:www-data /var/www/html/wordpress** ".

10.7. Completing the web installation

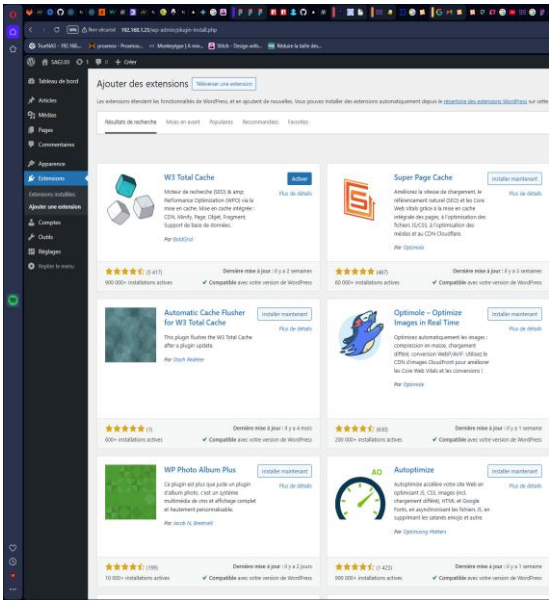
You can now open a web browser and navigate to your server's IP address via HTTP (for example, http://YOUR_RASPBERRY_IP) to continue the WordPress setup. On the first installation page, enter the required site information. Please note that the username and password you create here will be your administrator credentials to log into WordPress.

11. Optimizing WordPress

11.1. Installing the W3 Total Cache plugin

To optimize the speed of your WordPress site, you need to install the W3 Total Cache plugin. Log into your WordPress admin dashboard, navigate to the "Plugins" tab, and click "Add New". Search for "W3 Total Cache" and click "Install Now".

Important : DON'T activate the plugin yet.



11.2. Changing directory permissions

Change the permissions of the wp-content/ and wp-content/uploads/ folders temporarily for the duration of the plugin configuration. Run the following commands to do so :

sudo chmod 777 /var/www/html/wordpress/wp-content/
sudo chmod 777 /var/www/html/wordpress/wp-content/uploads/

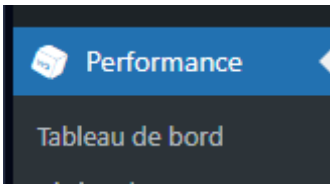
11.3. Activating the plugin and restoring permissions

Go back to your WordPress dashboard and finally activate the W3 Total Cache plugin. Once activated, you must immediately restore the directory permissions to their secure state. Run the following commands to do so :

sudo chmod 755 /var/www/html/wordpress/wp-content/
sudo chmod 755 /var/www/html/wordpress/wp-content/uploads/

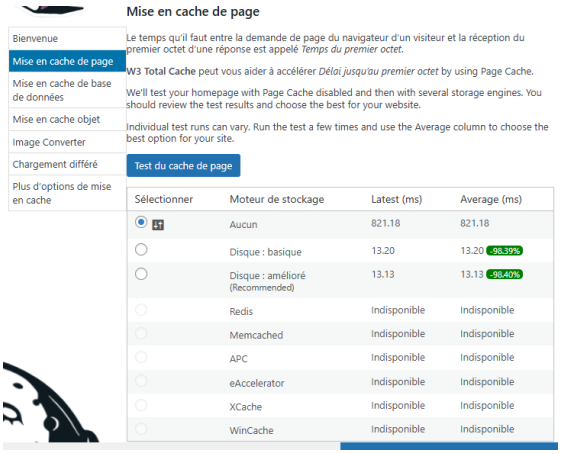
11.4. Configuring the cache via the Setup Guide

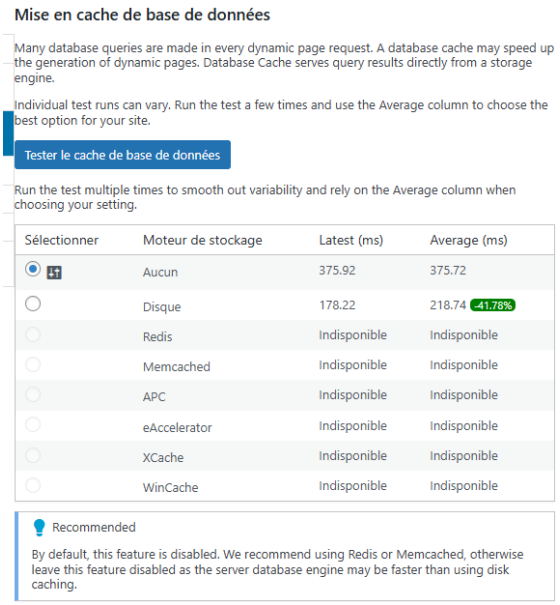
Go to the "Performance" section, then click on "Dashboard" in the navigation bar of your WordPress administrator dashboard.



Follow the instructions provided by the setup guide.

Note: For our specific system configuration, it is **NOT recommended** to enable database caching due to the lack of RAM for Redis or Memcached solutions, so only enable page caching with the “Disk: enhanced” solution.





11.5. Uncommenting the cache configuration

Open your Nginx configuration file by typing the following command: “ **sudo nano /etc/nginx/sites-available/wordpress** ”. Find the Include line for W3 Total Cache (“ **# include /var/www/html/wordpress/nginx.conf ;** ”) and remove the **#** symbol at the beginning to uncomment it. Save and exit the file.

Once this is done, restart Nginx with the command “ **sudo systemctl restart nginx** “.

11.6. Testing site navigation

Test navigating through your website; everything should now be working correctly.

Important testing notes:

- Ensure you are **logged out** of your WordPress administrator account, as the caching system is bypassed for administrator.
- The initial visit to any page will trigger the cache compilation.

Please **visit a page twice** to verify that the caching mechanism is actively working.

12. Testing the performances

12.1. Performance testing command

To test your site's performance and verify your caching efficiency, you can use the Apache Benchmark tool. Run the following command in your terminal :

```
ab -n 10000 -c 500 http://YOUR_RASPBERRY_IP/
```

- Replace the first number (10000) with the total number of requests you want to perform.
- Replace the second number (500) with the number of concurrent accounts making those requests.
- Replace the URL with your actual WordPress URL. Make sure to always include a trailing slash (/) at the end of the address.

13. Troubleshooting

13.1. "502 Bad Gateway" Error on the Website

The cause : Nginx is unable to communicate with the PHP engine. This usually happens if the PHP FPM service is down or if the socket path in the Nginx configuration is incorrect.

The solution : First, restart the PHP service by running " **sudo systemctl restart php8.4-fpm** ". If the issue persists, open your WordPress Nginx configuration and ensure the fastcgi_pass line points exactly to **unix:/run/php/php8.4-fpm.sock**.

13.2. "Error Establishing a Database Connection"

The cause : WordPress cannot connect to the MariaDB server. This is almost always caused by a typo in the database credentials.

The solution : Open your configuration file by typing " **sudo nano /var/www/html/wordpress/wp-config.php** ". Carefully double-check that the **DB_NAME, DB_USER**, and **DB_PASSWORD** exactly match the ones you created in step 9.2.

13.3. WordPress asks for FTP credentials to install plugins or updates

The cause : The web server (Nginx) does not have the correct permissions to write files to the WordPress directory.

The solution : You need to re-apply the correct ownership to the folders. Run the command " **sudo chown -R www-data:www-data /var/www/html/wordpress** " to grant Nginx full write access again.

13.4. Blank page or layout breaking after activating the cache

The cause : A conflict between the W3 Total Cache rules and the Nginx server block.

The solution : Temporarily comment out the Nginx Include line you activated in

step 12.6 by adding a **#** in front of it **(# include /var/www/html/wordpress/nginx.conf ;**). Restart Nginx using " **sudo systemctl reload nginx** " and clear your browser cache.

14. Glossary :

ab (Apache Benchmark) : A command-line tool used to measure the performance of HTTP web servers by simulating high loads of concurrent network requests to test server stability and speed under pressure.

Apt (Advanced Package Tool) : The package manager used by Debian-based systems. It allows you to securely install, update, and remove software by downloading packages from official repositories.

Caching : The process of temporarily storing frequently accessed data, such as fully rendered web pages. This drastically speeds up loading times for future visitors by preventing the server from recalculating the same information.

Chmod and Chown : Core commands used to manage system security. **Chmod** (Change Mode) modifies who is allowed to read, write, or execute a file. **Chown** (Change Owner) assigns the legal ownership of a file or directory to a specific system user or group.

Curl : A powerful command-line utility used to transfer data to or from a server over the internet. In this guide, it is utilized to securely download the WordPress installation archive directly from the official servers.

Database / SQL : A structured system for storing and organizing information. For WordPress, it acts as the site's memory, retaining every article, comment, and user setting. SQL is the language used to communicate with this database.

FastCGI and Unix Socket : Internal communication protocols designed for speed. In our configuration, the socket acts as a direct bridge allowing Nginx to instantly transmit dynamic requests to the PHP FPM engine without performance loss.

Localhost : A hostname that refers to the current machine being used. In computer networking, the standard loopback address for localhost is 127.0.0.1.

MariaDB : A highly popular, open-source Relational Database Management System (RDBMS). It acts as the digital vault for your website, securely storing and organizing all dynamic WordPress data.

Nano : A simple and intuitive text / code editor used directly within the terminal. It allows you to modify server configuration files without needing a graphical user interface.

Nginx : A high-performance, lightweight web server software. Its primary responsibility is to listen for incoming HTTP requests from users' web browsers and quickly serve the appropriate website files or route dynamic requests to PHP.

PHP FPM (FastCGI Process Manager) : The background service that processes dynamic PHP code. Because Nginx cannot natively execute PHP scripts, it passes these requests to PHP FPM,

which processes the WordPress code and returns plain, readable HTML.

RAM (Random Access Memory) : The fast working memory of your server. Since this resource can be limited on microcomputers, it is important to optimize services like MariaDB and caching mechanisms to prevent system overload.

Root (Superuser) : The supreme administrator account that possesses absolute rights over the system or the database. It is the equivalent of the "Administrator" account on Windows.

SSH (Secure Shell) : A cryptographic network protocol used to remotely control the server. This is typically how administrators type command lines from their main computer without needing a screen plugged into the server.

Sudo : A fundamental command that allows a standard user to execute programs with the elevated security privileges of the root account. It is required for making system-wide changes.

Symbolic Link (Symlink) : A special type of file that serves as an advanced shortcut pointing to another file or directory. It is traditionally used in Nginx setups to link available configurations to enabled ones.

W3 Total Cache : An optimization plugin for WordPress. It integrates with your server to implement various caching mechanisms, effectively minimizing server response times and reducing overall resource consumption.

WordPress : The most widely used Content Management System (CMS) in the world. It is the final application that allows you to create and manage your website through a simple graphical interface without writing code.

15. Index :

WordPress: Nginx configuration, 8.1; Database setup, 9.2; Download and extraction, 10.1, 10.2, 10.3; Configuration, 10.4, 10.5; Folder ownership, 10.6; Web installation completion, 10.7; Optimization, 11.1.