**MySQL, Apache, Wordpress and PHP configuration and security**

**Guides and resources used for creating these notes:**

**LAMP Setup:** <https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mariadb-php-lamp-stack-debian9>

**Wordpress Setup:**

<https://linuxconfig.org/how-to-install-wordpress-on-debian-9-stretch-linux>

**Wordpress security:**

<https://www.wordfence.com/learn/>

**MySQL guides**

<https://dev.mysql.com/doc/refman/8.0/en/security.html>

<https://www.upguard.com/blog/top-11-ways-to-improve-mysql-security>

https://www.tecmint.com/mysql-mariadb-security-best-practices-for-linux/

**Apache2 Documentation:**

<http://httpd.apache.org/docs/current/>

<https://geekflare.com/apache-web-server-hardening-security/>

**Wordpress**

**Wordpress Config:**

* **Plugins**
  + <https://wordpress.org/plugins/> - Link of where to search for plugins OR *Plugins > Add New* on the side of the dashboard
* **Users**
  + *Users > Add New* on the side to add a new user.
  + Keep register self off by default (which it usually already is) as people could spam the site.

**Wordpress Security:**

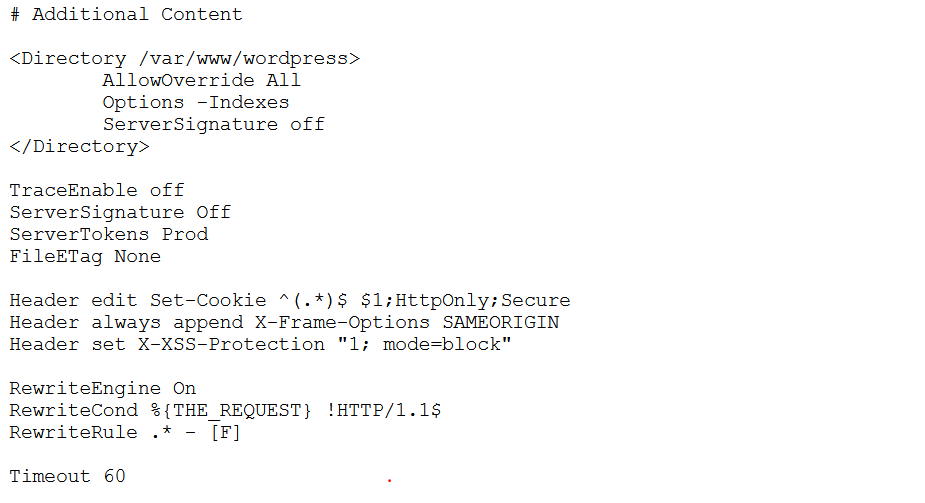
* **Updates**
  + *Dashboard > Home > Updates* - Update plugins, themes, etc ASAP as these could be patches to vulnerabilities.
* **Passwords**
  + DO NOT just use english words. Make an annoying password that’s harder to guess for cracking programs. (Ex: Don’t do this: Osprey123 DO this: #0$pr3y!23)
* **Editing website content**
  + DO NOT edit website files directly on the server with editors like vim. It becomes *accessible with the web.* ( A .swp file is created, that is accessible to the web)
* **HTTPS**
  + Use this to avoid data being tapped over the network.
  + Get familiar with LetsEncrypt
* **Download a Web-App Firewall / Monitor logs**
  + Wordfence is a reputable web-app firewall for wordpress.
  + The website *access* log and website *error* log are the two most important.
    - */var/log/apache2/access.log* and */var/log/apache2/error.log* are the locations. Use an editor like nano to view them if commands like tail and cat are too clunky to use.
* **Install fail2ban**
  + Stops bruteforce attacks on SSH by blocking IPs that fail too many times at authentication.
* **Use a malware and source code scanner to search the site for malware.**
  + Wordfence is capable of doing these things.

**MySQL**

* **Use the *mysql\_secure\_installation* command immediately**
  + Change the password for root to something secure
  + Remove the anonymous users
  + Disable remote login (only localhost should be allowed in here unless specified otherwise)
  + Remove the test database
  + Reload the privilege tables
* **Root access**
  + Make sure no one can access the database with *mysql -u root* without a password.
* **Port 3306 blocked for outside**
  + Make sure Port 3306 (The default port for MySQL / Mariadb databases) is inaccessible from the outside.
  + Use *nmap* to scan ports or use *telnet* on port 3306 (*telnet server\_host 3306)* to check if that port is open.
* **DO NOT run mysql with root**
  + Use a separate account to run it.
* **Disable SHOW DATABASES**
  + Add *skip-show-database* under *[mysqld]* in the */etc/mysql/my.cnf/* configuration file.
  + This helps prevent some information-gathering for hackers.
* **Prevent reading and accessing local files from MySQL** 
  + Adding *local-infile=0* to */etc/mysql/my.cnf* will prevent hackers from reading or accessing files on the OS should they successfully use methods like SQL injection.
* **Secure the MySQL config file**
  + *chmod 644 /etc/mysql/my.cnf*
  + Making the file only writable to root
* **Deleting MySQL history**
  + ***cat /dev/null > ~/.mysql\_history***
  + This will permanently cleanup the file that tracks the creation of user accounts, passwords, etc.

**Apache2**

**Finished product for additional configuration for the */etc/apache2/apache2.conf* file (BE SURE TO RESTART APACHE AFTER ALL CHANGES)**

****

* **Remove server and OS info**
  + Add *ServerTokens Prod* and *ServerSignature Off* somewhere in the file.
  + This will prevent the OS and Server type from being displayed in the GET requests.
* **Disable directory browser listing**
  + Add either *Options -Indexes* or *Options None* in the */var/www/wordpress* bracket.
  + Should prevent attackers from traversing directories from the browser.
* **Disabling Etags:**
  + Add *FileETag None* somewhere in the file.
  + Prevents attackers from obtaining information about inode numbers, MIME boundaries and child processes.
* **Protecting System Settings**
  + In the root directory of the config file (*<Directory />)* add *Options -Indexes* and *AllowOverride None*
  + This will prevent users from changing apache settings. (Only allow root to make changes)
* **Disable TRACE HTTP requests**
  + Adding *TraceEnable off*
  + Can prevent theft of cookie information and cross site tracing attacks
* **Set Cookies with HttpOnly and Secure flags**
  + Add this line to the config file: *Header edit Set-Cookie ^(.\*)$ $1;HttpOnly;Secure*
  + Prevents common cross site scripting attacks.
* **Prevent Clickjacking Attacks**
  + Add *Header always append X-Frame-Options SAMEORIGIN*
* **Prevent bowser XSS bypassing**
  + Prevents attackers using browser tricks to disable XSS protects.
  + Add *Header set X-XSS-Protection “1; mode=block”*
* **Disable HTTP 1.0 Protocol** 
  + Add the following lines:
    - *RewriteEngine On*
    - *RewriteCond %{THE\_REQUEST} !HTTP/1.1$*
    - *RewriteRule .\* - [F]*
  + HTTP 1.0 has weaknesses related to session hijacking.
* **Timeout value config**
  + Add *Timeout 60* to the file
  + This lowers the timeout threshold of the server to 60 seconds and can mitigate a lighter DoS attack. Not THAT important but worth throwing in.