comment-installer-lomp-stack-openlitespeed-mysql-and-php-sur-debian-12

OpenLiteSpeed is a lightweight and open-source version of the LiteSpeed Server developed by LiteSpeed Technologies. It supports Apache Rewrite rules, HTTP/2 and HTTP/3 along with TLS v1.3 and QUIC protocols. It comes with a WebGUI-based Administration panel which makes it different from other servers and easier to manage.

The LOMP Stack is an acronym for Linux, OpenLiteSpeed, MySQL/MariaDB, and PHP. Litespeed servers are known for their speed, especially with PHP which integrates using the LiteSpeed Server Application Programming Interface (LSAPI). The LiteSpeed PHP (LSPHP) interpreter serves dynamic PHP pages via LSAPI.

In this tutorial, we will learn how to install an OpenLiteSpeed Server on a Debian 12 machine.

Prerequisites

- A server running Debian 12.
- A non-root user with sudo privileges.
- A fully qualified domain name (FQDN) like `example.com` pointing to the server.
- SELinux doesn't need to be disabled or configured to work with OpenLiteSpeed.
- Make sure everything is updated.

  ```
  $ sudo apt update
  $ sudo apt upgrade
  ```

- Few packages that your system needs.

  ```
  $ sudo apt install wget curl nano ufw software-properties-common dirmngr apt-transport-https gnupg2 ca-certificates lsb-release debian-archive-keyring unzip -y
  ```

  Some of these packages may already be installed on your system.

Step 1 - Configure Firewall

The first step is to configure the firewall. Debian comes with ufw (Uncomplicated Firewall) by default.

Check if the firewall is running.

```
$ sudo ufw status
```

You will get the following output.

```
Status: inactive
```

Allow SSH port so that the firewall doesn't break the current connection upon enabling it.

```
$ sudo ufw allow OpenSSH
```

Allow HTTP and HTTPS ports as well.

```
$ sudo ufw allow http
$ sudo ufw allow https
```

Open ports related to Openlitespeed.

```
$ sudo ufw allow 7080/tcp
```

Enable the Firewall

```
$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Check the status of the firewall again.

```
$ sudo ufw status
```

You should see a similar output.

```
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
443                        ALLOW       Anywhere
7080/tcp                   ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
443 (v6)                   ALLOW       Anywhere (v6)
7080/tcp (v6)              ALLOW       Anywhere (v6)
```

Step 2 - Install OpenLiteSpeed

OpenLiteSpeed doesn't ship a package for Debian 12. Therefore, we will build our copy from the source code.

Download the OpenLiteSpeed source code file. You can get the link to the latest source code file from the OpenLiteSpeed official downloads page.

```
$ wget https://openlitespeed.org/packages/openlitespeed-1.7.18.src.tgz
```

Extract the file.

```
$ tar -zxf openlitespeed-*.tgz
```

Switch to the extracted directory.

```
$ cd openlitespeed-1.7.18
```

Build the server package.

```
$ sudo ./build.sh
```

You might have to wait a good 5-10 minutes for the process to finish. Ignore any warnings you receive during the process. You will receive the following output when it is finished.

```
[100%] Linking CXX shared library modpagespeed.so
/usr/bin/ld: warning: 140.x25519-asm-x86_64.o.o: missing .note.GNU-stack section implies executable stack
/usr/bin/ld: NOTE: This behaviour is deprecated and will be removed in a future version of the linker
[100%] Built target modpagespeed
Start to pack files.
-e Building finished, please run ./install.sh for installation.
-e You may want to update the ols.conf to change the settings before installation.
-e Enjoy.
```

Once the process is finished, open the file *ols.conf* for editing.

```
$ sudo nano ols.conf
```

Edit the file as shown below.

```
#If you want to change the default values, please update this file.
#

SERVERROOT=/usr/local/lsws
OPENLSWS_USER=nobody
OPENLSWS_GROUP=nobody
OPENLSWS_ADMIN=navjot
OPENLSWS_EMAIL=navjot@example.com
OPENLSWS_ADMINSSL=yes
OPENLSWS_ADMINPORT=7080
USE_LSPHP7=yes
DEFAULT_TMP_DIR=/tmp/lshttpd
PID_FILE=/tmp/lshttpd/lshttpd.pid
OPENLSWS_EXAMPLEPORT=8088

#You can set password here
#OPENLSWS_PASSWORD=
```

Don't add your password here. We will set the password later using the command line. Once you are finished, save the file by pressing **Ctrl + X** and entering **Y** when prompted.

Now that OpenLiteSpeed is built, let us install it.

```
$ sudo ./install.sh
```

The installer script installs and enables the *lsws* service for the server. You will receive the following output when finished.

```
Updating webcache manager, please waiting ...
Downloading latest shared code tar file...
Checking tar file md5...
Removing existing shared code directory...
Extracting downloaded shared code...
Removing local shared code tar file...
Updating lscmctl script...
Done!

-e Installation finished, Enjoy!

-e Your webAdmin password is kXjWTl5j, written to file /usr/local/lsws/adminpasswd.
```

Start the OpenLiteSpeed server.

```
$ sudo systemctl start lsws
```

Check the status of the service.

```
$ sudo systemctl status lsws
? lsws.service - LSB: lshttpd
     Loaded: loaded (/etc/init.d/lsws; generated)
     Active: active (running) since Wed 2023-09-27 15:55:13 UTC; 4h 18min ago
       Docs: man:systemd-sysv-generator(8)
      Tasks: 4 (limit: 4652)
     Memory: 79.2M
        CPU: 37.823s
     CGroup: /system.slice/lsws.service
             ??64164 "openlitespeed (lshttpd - main)"
             ??64171 "openlitespeed (lscgid)"
             ??64184 "openlitespeed (lshttpd - #01)"
             ??64185 "openlitespeed (lshttpd - #02)"

Sep 27 15:55:11 lomp systemd[1]: Starting lsws.service - LSB: lshttpd...
Sep 27 15:55:13 lomp systemd[1]: Started lsws.service - LSB: lshttpd.
```

Check the version of the server installed.

```
$ /usr/local/lsws/bin/lshttpd -v
LiteSpeed/1.7.18 Open (BUILD built: Tue Aug 29 12:59:39 UTC 2023)
        module versions:
        lsquic 3.2.0
        modgzip 1.1
        cache 1.64
        mod_security 1.4 (with libmodsecurity v3.0.9)
```

Create the Administrator Password

You can use the administrator password given during the installation process. However, you should create your own by resetting it. Run the password reset script.

```
$ sudo /usr/local/lsws/admin/misc/admpass.sh
```

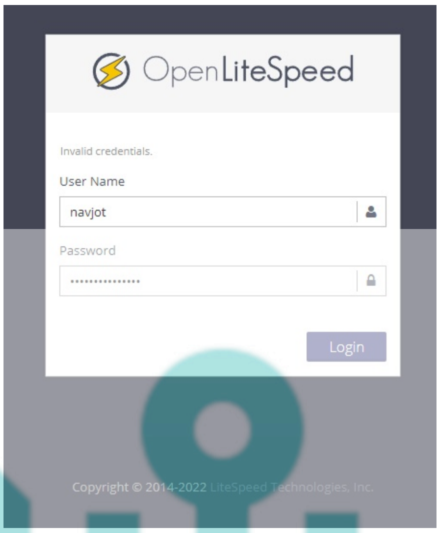
You will get the following output. Choose your username and set a strong password.

```
Please specify the user name of administrator.
This is the user name required to login the administration Web interface.

User name [admin]: navjot

Please specify the administrator's password.
This is the password required to login the administration Web interface.

Password:
Retype password:
Administrator's username/password is updated successfully!
```

You can now use the new administrator password.

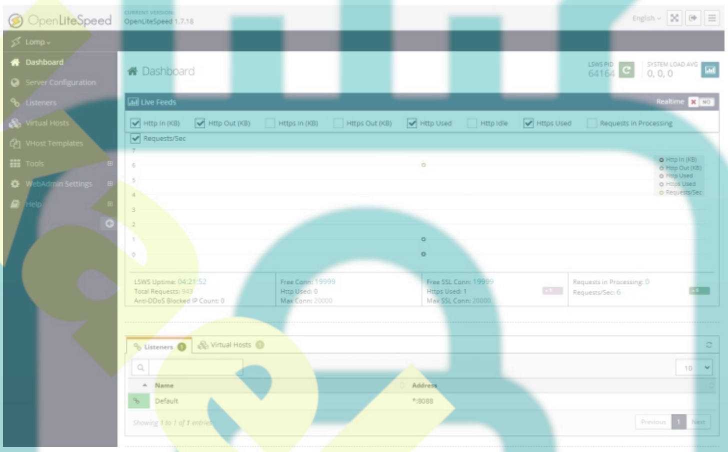
Open *http://<YOURSERVERIP>:7080* to access OpenLiteSpeed's administration panel. On your first login, your browser will warn that your connection is not private. Click Advanced and click "Accept the risk and Continue" (in the case of Firefox) or "Proceed to *<YOURSERVERIP>(unsafe)*" (in the case of Chromium-based browser). You won't see the warning again.

You should see the login page.



Enter the credentials you set earlier, and press the **Login** button to proceed.

You will get the following screen.



Step 3 - Install MariaDB

Debian doesn't ship with MySQL server anymore. Therefore, we will be using the MySQL drop-in replacement, MariaDB. But before proceeding ahead, we need to update the LiteSpeed repository. The Litespeed repository added via the installer doesn't work properly.

Open the file `/etc/apt/sources.list.d/lst_debian_repo.list` for editing.

```
$ sudo nano /etc/apt/sources.list.d/lst_debian_repo.list
```

Change the file contents by adding the Debian 11 (`bullseye`) to it. We can't use Debian 12 (`bookworm`) to it since the repository is not updated for it.

```
deb http://rpms.litespeedtech.com/debian/ bullseye main
```

Once you are finished, save the file by pressing **Ctrl + X** and entering **Y** when prompted.

Install the MariaDB server.

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

MariaDB service is automatically started and running post-install.

Check the status of the service.

```
$ sudo systemctl status mariadb
```

You will get the following output.

```
? mariadb.service - MariaDB 10.11.3 database server
     Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; preset: enabled)
     Active: active (running) since Fri 2023-09-29 01:21:09 UTC; 1min 26s ago
       Docs: man:mariadbd(8)
             https://mariadb.com/kb/en/library/systemd/
   Main PID: 361450 (mariadbd)
     Status: "Taking your SQL requests now..."
      Tasks: 10 (limit: 4652)
     Memory: 190.8M
        CPU: 411ms
     CGroup: /system.slice/mariadb.service
             ??361450 /usr/sbin/mariadbd
```

Run the MariaDB security script.

```
$ sudo mariadb-secure-installation
```

You will be asked for the root password. Press **Enter** because we haven't set any password for it.

```
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
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
```

Next, you will be asked if you want to switch to the Unix socket authentication method. The `unix_socket` plugin allows you to use your operating system credentials to connect to the MariaDB server. Since you already have a protected root account, enter `n` to proceed.

```
OK, successfully used password, moving on...

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

```
You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
```

Next, you will be asked if you want to change your root password. On Debian 12, the root password is tied closely to automated system maintenance, so it should be left alone. Type `n` to proceed further.

```
 ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
```

Next, you will be asked certain questions to improve MariaDB security. Type **Y** to remove anonymous users, disallow remote root logins, remove the test database, and reload the privilege tables.

```
 ... skipping.

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
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

You can enter the MariaDB shell by typing `sudo mysql` or `sudo mariadb` on the command line.

Step 4 - Install PHP

Since we built our package from the source, it compiles and builds an old version of PHP which is not recommended for use. You can check it via the following command.

```
$ /usr/local/lsws/fcgi-bin/lsphp -v
PHP 5.6.40 (litespeed) (built: May 10 2023 23:03:31)
Copyright (c) 1997-2016 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
```

Therefore, we need to build and install the latest version of PHP. But, before we proceed, we need to install the build tools.

```
$ sudo apt install build-essential autoconf libtool bison re2c pkg-config
```

The next step is to install the packages required by the PHP build process.

```
$ sudo apt install libssl-dev libsqlite3-dev zlib1g-dev libcurl4-openssl-dev libonig-dev libzip-dev libmemcached-dev libreadline-dev libgmp-dev libpng-dev libjpeg-dev libwebp-dev libxpm-dev libicu-dev libfreetype6-dev
```

Next, download the PHP source code. We will download the PHP 8.2.10 version which is the latest version available at the time of writing.

```
$ cd ~
$ wget https://www.php.net/distributions/php-8.2.11.tar.gz
```

Extract the files.

```
$ tar -xzf php-8.2.11.tar.gz
```

Switch to the downloaded directory.

```
$ cd php-8.2.11
```

Next, run the configure script with the following options. The `--enable-litespeed` option is essential.

```
$ sudo ./configure --prefix=/usr/local/lsws/lsphp82 --enable-bcmath --enable-calendar --enable-exif --enable-ftp --enable-gd --enable-intl --enable-mbregex --enable-mbstring --enable-mysqlnd --enable-opcache --enable-s
```

You should get the following output once the script is finished.

```
+--------------------------------------------------------------------+
| License:                                                           |
| This software is subject to the PHP License, available in this     |
| distribution in the file LICENSE. By continuing this installation  |
| process, you are bound by the terms of this license agreement.     |
| If you do not agree with the terms of this license, you must abort |
| the installation process at this point.                            |
+--------------------------------------------------------------------+

Thank you for using PHP.
```

Compile the source.

```
$ sudo make -j $(nproc)
```

You will receive a similar output once finished.

```
/bin/bash /home/navjot/php-8.2.11/libtool --silent --preserve-dup-deps --tag CC --mode=link cc -shared -I/home/navjot/php-8.2.11/include -I/home/navjot/php-8.2.11/main -I/home/navjot/php-8.2.11 -I/home/navjot/php-8.2.1
/bin/bash /home/navjot/php-8.2.11/libtool --silent --preserve-dup-deps --tag CC --mode=install cp ext/opcache/opcache.la /home/navjot/php-8.2.11/modules

Build complete.
Don't forget to run 'make test'.
```

Once you are finished, run the following command to try and check the version.

```
$ ./sapi/litespeed/php -v
PHP 8.2.11 (litespeed) (built: Sep 28 2023 18:40:08)
Copyright (c) The PHP Group
Zend Engine v4.2.11, Copyright (c) Zend Technologies
```

Install PHP.

```
$ sudo make install
```

You will get the following output.

```
Installing shared extensions:     /usr/local/lsws/lsphp82/lib/php/extensions/no-debug-non-zts-20220829/
Installing PHP CLI binary:        /usr/local/lsws/lsphp82/bin/
Installing PHP CLI man page:      /usr/local/lsws/lsphp82/php/man/man1/
Installing PHP LiteSpeed binary:  /usr/local/lsws/lsphp82/bin/
Installing phpdbg binary:         /usr/local/lsws/lsphp82/bin/
Installing phpdbg man page:       /usr/local/lsws/lsphp82/php/man/man1/
Installing PHP CGI binary:        /usr/local/lsws/lsphp82/bin/
Installing PHP CGI man page:      /usr/local/lsws/lsphp82/php/man/man1/
Installing build environment:     /usr/local/lsws/lsphp82/lib/php/build/
Installing header files:          /usr/local/lsws/lsphp82/include/php/
Installing helper programs:       /usr/local/lsws/lsphp82/bin/
  program: phpize
  program: php-config
Installing man pages:             /usr/local/lsws/lsphp82/php/man/man1/
  page: phpize.1
  page: php-config.1
Installing PEAR environment:      /usr/local/lsws/lsphp82/lib/php/
[PEAR] Archive_Tar    - installed: 1.4.14
[PEAR] Console_Getopt - installed: 1.4.3
[PEAR] Structures_Graph- installed: 1.1.1
[PEAR] XML_Util       - installed: 1.4.5
warning: pear/PEAR dependency package "pear/Archive_Tar" installed version 1.4.14 is not the recommended version 1.4.4
[PEAR] PEAR           - installed: 1.10.13
Wrote PEAR system config file at: /usr/local/lsws/lsphp82/etc/pear.conf
You may want to add: /usr/local/lsws/lsphp82/lib/php to your php.ini include_path
Installing PDO headers:           /usr/local/lsws/lsphp82/include/php/ext/pdo/
```

Verify the PHP installation. There are two PHP binaries available in the `/usr/local/lsws/lsphp82/bin` directory. One is the normal `php` which is the command-line version and the other is the Litespeed version `lsphp`. The second one is the one we will be using.

```
$ /usr/local/lsws/lsphp82/bin/lsphp -v
PHP 8.2.11 (litespeed) (built: Sep 28 2023 18:40:08)
Copyright (c) The PHP Group
Zend Engine v4.2.11, Copyright (c) Zend Technologies
```

You can check the list of enabled PHP modules.

```
$ /usr/local/lsws/lsphp82/bin/php --modules
[PHP Modules]
bcmath
calendar
Core
ctype
curl
date
dom
exif
FFI
fileinfo
filter
ftp
gd
gettext
gmp
hash
iconv
imap
intl
json
ldap
libxml
mbstring
mysqli
mysqlnd
openssl
pcre
PDO
pdo_mysql
pdo_pgsql
pdo_sqlite
Phar
posix
pspell
random
readline
Reflection
session
shmop
SimpleXML
soap
sockets
sodium
SPL
sqlite3
standard
sysvsem
sysvshm
tidy
tokenizer
xml
xmlreader
xmlwriter
xsl
zip
zlib

[Zend Modules]
```

Copy the `php.ini-production` from the install folder to the `/usr/local/lsws/lsphp82/lib` folder.

```
$ sudo cp php.ini-production /usr/local/lsws/lsphp82/lib/php.ini
```

We will configure OpenLiteSpeed to work with PHP later.

Open the `php.ini` for editing.

```
$ sudo nano /usr/local/lsws/lsphp82/lib/php.ini
```

Find the variable `include_path` and change it's value as shown below.

```
$ ;;;;;;;;;;;;;;;;;;;;;;;;;
; Paths and Directories ;
;;;;;;;;;;;;;;;;;;;;;;;;;

; UNIX: "/path1:/path2"
include_path = /usr/local/lsws/lsphp82/lib/php
```

Once you are finished, save the file by pressing **Ctrl + X** and entering **Y** when prompted.

Step 5 - Configure MariaDB

Log in to the MariaDB shell.

```
$ sudo mysql
```

Create a test database. Replace `testdb` with the appropriate database name of your choice.

```
mysql> CREATE DATABASE testdb;
```

Create a test user. Replace `testuser` with an appropriate username. Replace `Your_Password123` with a strong password.

```
mysql> CREATE USER 'testuser'@'localhost' IDENTIFIED BY 'Your_Password123';
```

Grant all privileges on the database to the user.

```
mysql> GRANT ALL PRIVILEGES ON testdb.* TO 'testuser'@'localhost';
```

Since we are not modifying the root user, you should create another SQL user for performing administrative tasks that employ password authentication. Choose a strong password for this one.

```
MariaDB> GRANT ALL ON *.* TO 'navjot'@'localhost' IDENTIFIED BY 'Yourpassword32!' WITH GRANT OPTION;
```

Flush user privileges.

```
mysql> FLUSH PRIVILEGES;
```

Exit the MySQL shell.

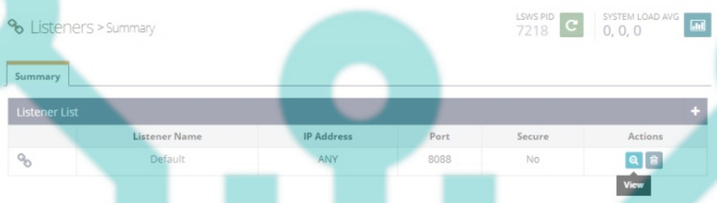
```
mysql> exit
```

Step 6 - Configure OpenLiteSpeed

Switch the HTTP port back to 80

Let us change the default HTTP port to 80. Log in to your administration panel at `http://<YOURSERVERIP>:7080` with the credentials you just created.

Visit the **Listeners** section from the left. You will see the default listeners with port `8080`.


Listeners > Summary

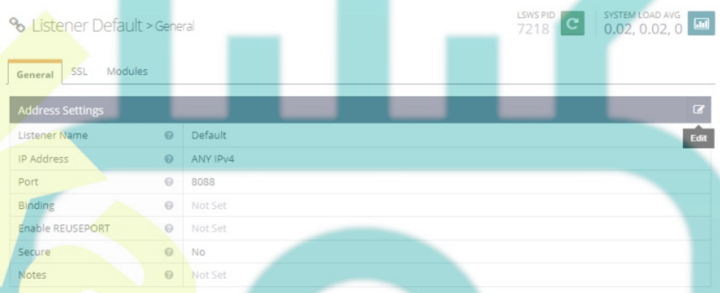
LSWS PID 7218 | SYSTEM LOAD AVG 0, 0, 0

Summary

Listener List

	Listener Name	IP Address	Port	Secure	Actions
	Default	ANY	8088	No	View

Click the **View** button to see the detailed configuration. On the next page under **Listener Default > General** Page, click on the **Edit icon** and change the port from `8080` to `80`.

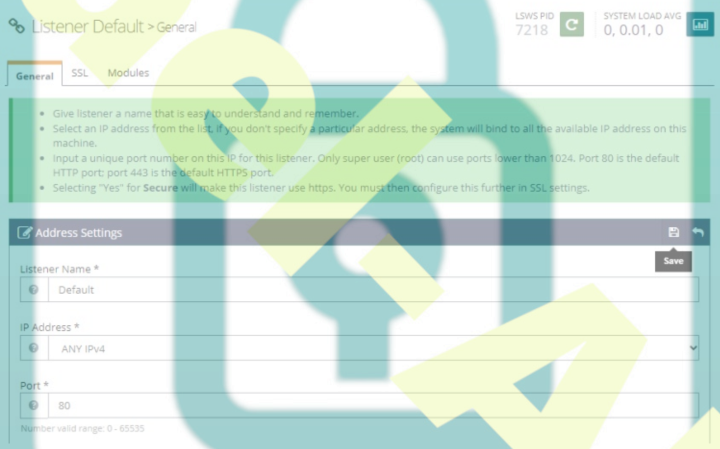

Listener Default > General

LSWS PID 7218 | SYSTEM LOAD AVG 0.02, 0.02, 0

General | SSL | Modules

Address Settings — Edit

Listener Name	Default
IP Address	ANY IPv4
Port	8088
Binding	Not Set
Enable REUSEPORT	Not Set
Secure	No
Notes	Not Set


Listener Default > General

LSWS PID 7218 | SYSTEM LOAD AVG 0, 0.01, 0

General | SSL | Modules

- Give listener a name that is easy to understand and remember.
- Select an IP address from the list. If you don't specify a particular address, the system will bind to all the available IP address on this machine.
- Input a unique port number on this IP for this listener. Only super user (root) can use ports lower than 1024. Port 80 is the default HTTP port; port 443 is the default HTTPS port.
- Selecting "Yes" for **Secure** will make this listener use https. You must then configure this further in SSL settings.

Address Settings — Save

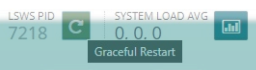
Listener Name * : Default

IP Address * : ANY IPv4

Port * : 80

Number valid range: 0 - 65535

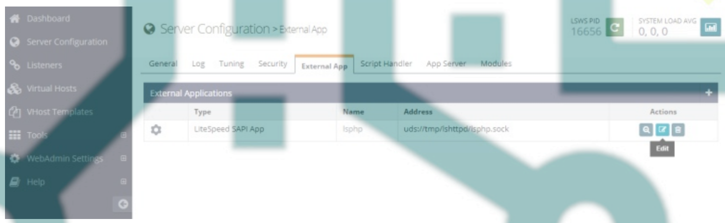
Click Save and then restart the server by clicking on the Graceful restart button.



Step 7 - Configure PHP

In this step, we need to associate our copy of PHP 8.2 with the server.

Click on the **Server Configuration** section on the left and then on the tab **External App**. You will see an existing LiteSpeed App for PHP. We will make some edits to it.



Server Configuration > External App

LSWS PID 16656 | SYSTEM LOAD AVG 0, 0, 0

General | Log | Tuning | Security | External App | Script Handler | App Server | Modules

External Applications

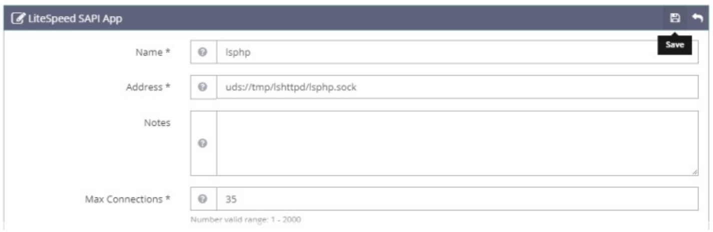
	Type	Name	Address	Actions
	LiteSpeed SAPI App	lsphp	uds://tmp/lshttpd/lsphp.sock	Edit

Click on the **Edit** button to edit the PHP app.

Next, match the configuration as shown below. Leave all the other fields blank.

```
Name: lsphp
Address: uds://tmp/lshttpd/lsphp.sock
Max Connections: 35
Environment: PHP_LSAPI_MAX_REQUESTS=500
             PHP_LSAPI_CHILDREN=35
             LSAPI_AVOID_FORK=200M
Initial Request Timeout (secs): 60
Retry Timeout : 0
Persistent Connection: Yes
Response Buffering: no
Start By Server: Yes(Through CGI Daemon)
Command: lsphp82/bin/lsphp
Back Log: 100
Instances: 1
Priority: 0
Memory Soft Limit (bytes): 2047M
Memory Hard Limit (bytes): 2047M
Process Soft Limit: 1400
Process Hard Limit: 1500
```

Click Save when finished.

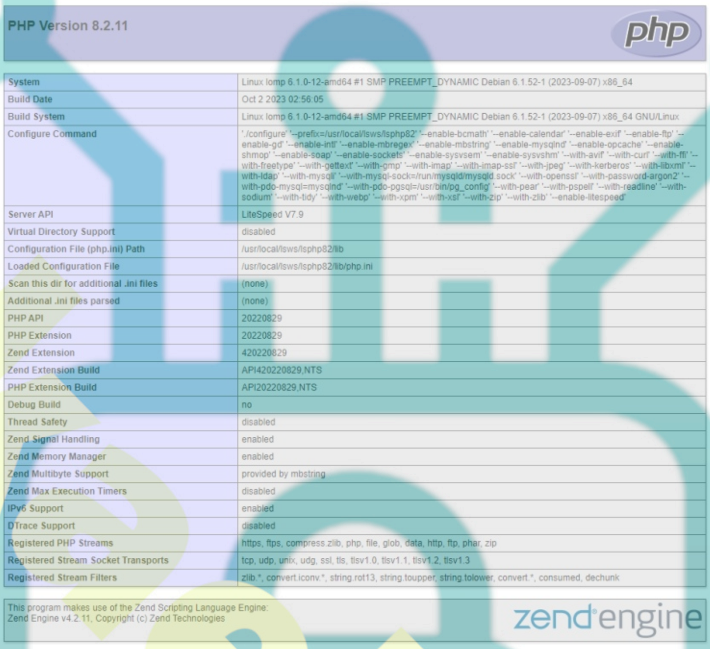


Name *	lsphp
Address *	uds://tmp/lshttpd/lsphp.sock
Notes	
Max Connections *	35

Number valid range: 1 - 2000

Now that we have created our own PHP 8.2 app, we need to tell the server to start using it. Since we edited the default listing, it is already configured. Restart the server by clicking on the **Graceful restart** button.

To test whether your PHP has been switched correctly, visit *`http://<YOURSERVERIP>/phpinfo.php`* in your browser.



PHP Version 8.2.11

System	Linux lomp 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64
Build Date	Oct 2 2023 02:56:05
Build System	Linux lomp 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
Configure Command	'./configure' '--prefix=/usr/local/lsws/lsphp82' '--enable-bcmath' '--enable-calendar' '--enable-exif' '--enable-ftp' '--enable-gd' '--enable-intl' '--enable-mbregex' '--enable-mbstring' '--enable-mysqlnd' '--enable-opcache' '--enable-shmop' '--enable-soap' '--enable-sockets' '--enable-sysvsem' '--enable-sysvshm' '--with-avif' '--with-curl' '--with-ffi' '--with-freetype' '--with-gettext' '--with-gmp' '--with-imap' '--with-imap-ssl' '--with-jpeg' '--with-kerberos' '--with-libxml' '--with-ldap' '--with-mysqli' '--with-mysql-sock=/run/mysqld/mysqld.sock' '--with-openssl' '--with-password-argon2' '--with-pdo-mysql=mysqlnd' '--with-pdo-pgsql=/usr/bin/pg_config' '--with-pear' '--with-pspell' '--with-readline' '--with-sodium' '--with-tidy' '--with-webp' '--with-xpm' '--with-xsl' '--with-zip' '--with-zlib' '--enable-litespeed'
Server API	LiteSpeed V7.9
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/lsws/lsphp82/lib
Loaded Configuration File	/usr/local/lsws/lsphp82/lib/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20220829
PHP Extension	20220829
Zend Extension	420220829
Zend Extension Build	API420220829,NTS
PHP Extension Build	API20220829,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
Zend Max Execution Timers	disabled
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk

This program makes use of the Zend Scripting Language Engine:
Zend Engine v4.2.11, Copyright (c) Zend Technologies

Restart PHP

On OpenLiteSpeed, if you edit php.ini or install a new PHP module, restarting the server won't show the changes. You will need to restart the PHP process for that. First, you will need to locate the process IDs for the *`lsphp`* process.

```
$ ps aux | grep lsphp
nobody    500747  0.9  0.8 121104 34928 ?        S    05:58   0:00 lsphp
nobody    500748  0.0  0.4 121104 16760 ?        Ss   05:58   0:00 lsphp
navjot    500751  0.0  0.0   3876  1908 pts/0    S+   05:58   0:00 grep lsphp
```

Manually kill the *`lsphp`* processes.

```
$ sudo kill -9 500747
$ sudo kill -9 500748
```

Restart the Server.

```
$ sudo systemctl restart lsws
```

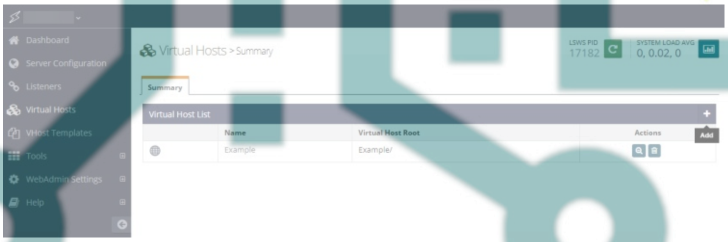
Step 8 - Create VirtualHost

First, we need to create directories for our virtual host.

```
$ sudo mkdir /usr/local/lsws/example.com/{html,logs} -p
```

The *`html`* directory will hold the public files and the *`logs`* directory will contain server logs.

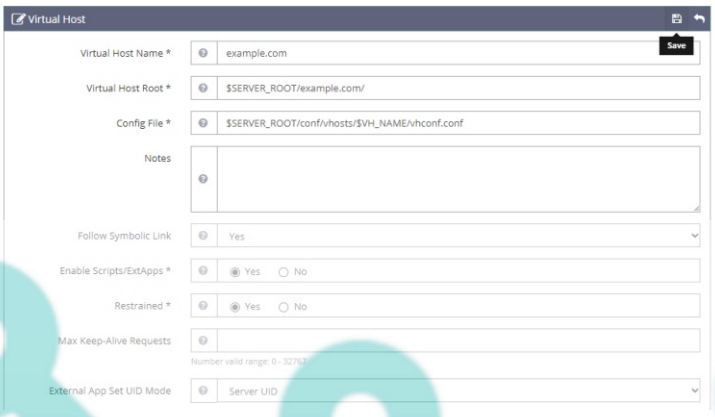
Next, open the Admin console, access the **Virtual Hosts** section from the left, and click the Add button.



Name	Virtual Host Root	Actions
Example	Example/	

Fill in the values as specified

```
Virtual Host Name: example.com
Virtual Host Root: $SERVER_ROOT/example.com/
Config File: $SERVER_ROOT/conf/vhosts/$VH_NAME/vhconf.conf
Follow Symbolic Link: Yes
Enable Scripts/ExtApps: Yes
Restrained: Yes
External App Set UID Mode: Server UID
```



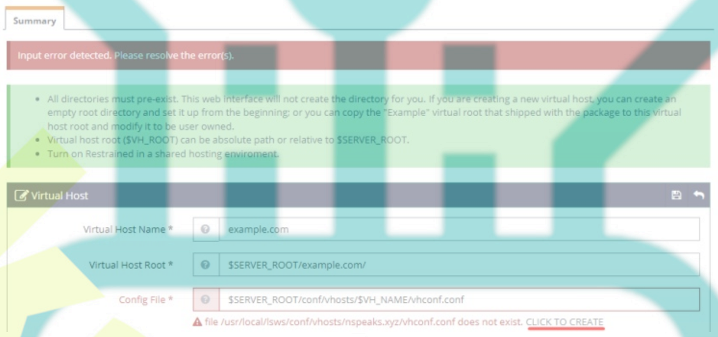
Virtual Host

Save

Virtual Host Name *	example.com
Virtual Host Root *	$SERVER_ROOT/example.com/
Config File *	$SERVER_ROOT/conf/vhosts/$VH_NAME/vhconf.conf
Notes	
Follow Symbolic Link	Yes
Enable Scripts/ExtApps *	[✓ Yes] No
Restrained *	[✓ Yes] No
Max Keep-Alive Requests	
External App Set UID Mode	Server UID

Number valid range: 0 - 32767

Click on the **Save** button when finished. You will get the following error because the configuration file doesn't exist as of now. Click on the link to create the configuration file.



Summary

Input error detected. Please resolve the error(s).

- All directories must pre-exist. This web interface will not create the directory for you. If you are creating a new virtual host, you can create an empty root directory and set it up from the beginning; or you can copy the "Example" virtual root that shipped with the package to this virtual host root and modify it to be user owned.
- Virtual host root ($VH_ROOT) can be absolute path or relative to $SERVER_ROOT.
- Turn on Restrained in a shared hosting enviroment.

Virtual Host

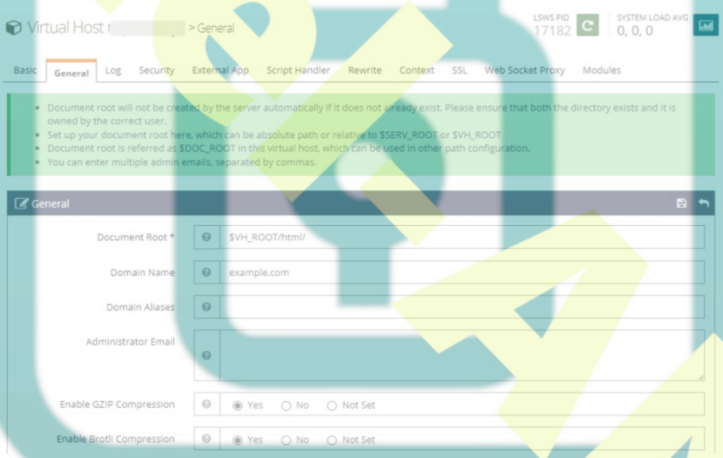
Virtual Host Name *	example.com
Virtual Host Root *	$SERVER_ROOT/example.com/
Config File *	$SERVER_ROOT/conf/vhosts/$VH_NAME/vhconf.conf

⚠ file /usr/local/lsws/conf/vhosts/nspeaks.xyz/vhconf.conf does not exist. CLICK TO CREATE

Click the **Save** button again to finish creating the Virtual Host.

Once the virtual host is created, go to **Virtual Hosts -> Choose Virtual Host(example.com) -> General** and modify the configuration as given.

```
Document Root: $VH_ROOT/html/
Domain Name: example.com
Enable GZIP Compression: Yes
Enable Brotli Compression: Yes
```



Virtual Host [redacted] > General

LSWS PID 17182 SYSTEM LOAD AVG 0, 0, 0

Basic General Log Security External App Script Handler Rewrite Context SSL Web Socket Proxy Modules

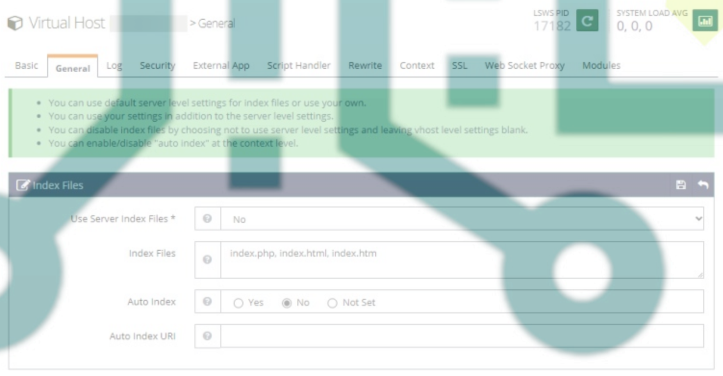
- Document root will not be created by the server automatically if it does not already exist. Please ensure that both the directory exists and it is owned by the correct user.
- Set up your document root here, which can be absolute path or relative to $SERV_ROOT or $VH_ROOT
- Document root is referred as $DOC_ROOT in this virtual host, which can be used in other path configuration.
- You can enter multiple admin emails, separated by commas.

General

Document Root *	$VH_ROOT/html/
Domain Name	example.com
Domain Aliases	
Administrator Email	
Enable GZIP Compression	[✓ Yes] No Not Set
Enable Brotli Compression	[✓ Yes] No Not Set

Click the **Save** button when finished. Next, we need to set up index files. Click the edit button against **Index files** below the **General** Section. Set the following options.

```
Use Server Index Files: No
Index files: index.php, index.html, index.htm
Auto Index: No
```



Virtual Host [redacted] > General

LSWS PID 17182 SYSTEM LOAD AVG 0, 0, 0

Basic General Log Security External App Script Handler Rewrite Context SSL Web Socket Proxy Modules

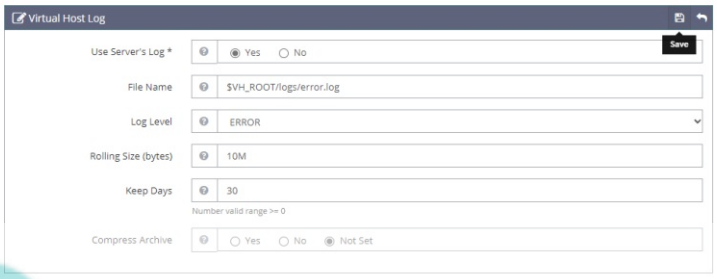
- You can use default server level settings for index files or use your own.
- You can use your settings in addition to the server level settings.
- You can disable index files by choosing not to use server level settings and leaving vhost level settings blank.
- You can enable/disable "auto index" at the context level.

Index Files

Use Server Index Files *	No
Index Files	index.php, index.html, index.htm
Auto Index	Yes [✓ No] Not Set
Auto Index URI	

Click Save when done. Next, we need to choose Log files. Go to the **Log** section, click Edit against **Virtual Host Log**, and fill in the following values. Leave other settings unchanged.

```
Use Server's Log: Yes
File Name: $VH_ROOT/logs/error.log
Log Level: ERROR
Rolling Size (bytes): 10M
Keep Days: 30
Compress Archive: Not Set
```

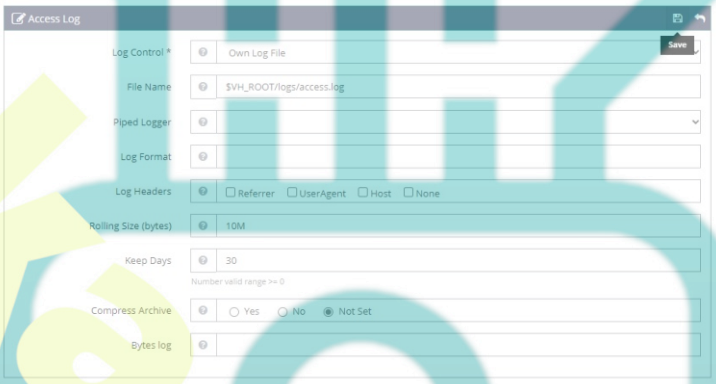


Use Server's Log * [✓ Yes] No
File Name: $VH_ROOT/logs/error.log
Log Level: ERROR
Rolling Size (bytes): 10M
Keep Days: 30
Number valid range >= 0
Compress Archive: Yes No [✓ Not Set]

You can choose the **Log Level** as `DEBUG` if you are on a development machine.

Click Save and then click the plus sign in the **Access Log** section to add a new entry. Fill in the following values.

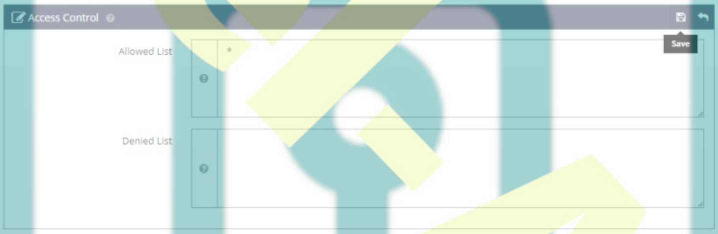
```
Log Control: Own Log File
File Name: $VH_ROOT/logs/access.log
Piped Logger: Not Set
Log Format: Not Set
Log Headers: Not Set
Rolling Size (bytes): 10M
Keep Days: 30
Compress Archive: Not Set
Bytes log: Not Set
```



Log Control *: Own Log File
File Name: $VH_ROOT/logs/access.log
Piped Logger:
Log Format:
Log Headers: ☐ Referrer ☐ UserAgent ☐ Host ☐ None
Rolling Size (bytes): 10M
Keep Days: 30
Number valid range >= 0
Compress Archive: Yes No [✓ Not Set]
Bytes log:

Click **Save** when done. Next, we need to configure **Access Control** under the **Security** section. Set the following Values.

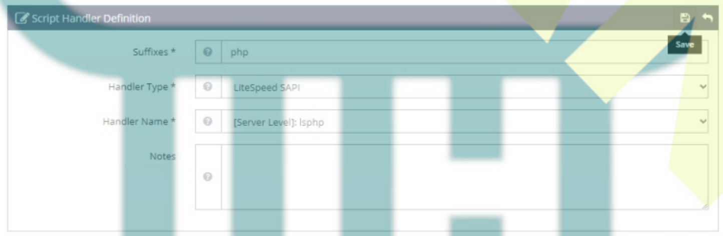
```
Allowed List: *
Denied List: Not set
```



Allowed List: *
Denied List:

Click **Save** when done. Next, we need to set the **Script Handler Definition**. Click the plus (+) sign to add a new definition. Set the following values.

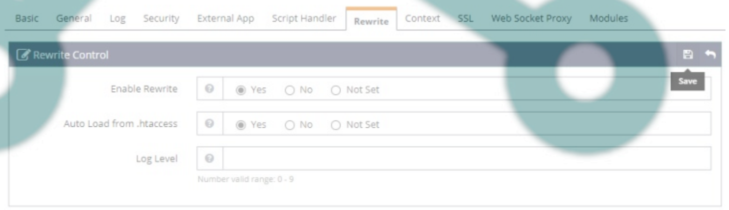
```
Suffixes: php
Handler Type: LiteSpeed SAPI
Handler Name: [Server Level]: lsphp
```



Suffixes *: php
Handler Type *: LiteSpeed SAPI
Handler Name *: [Server Level]: lsphp
Notes:

Click **Save** when done. Next, we need to set **Rewrite Control** under the **Rewrite** section. Set the following values.

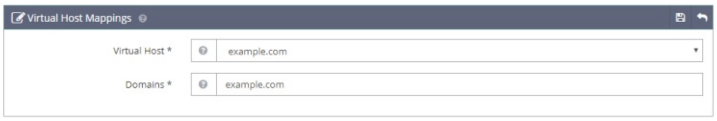
```
Enable Rewrite: Yes
Auto Load from .htaccess: Yes
Log Level: Not Set
```



Enable Rewrite: [✓ Yes] No Not Set
Auto Load from .htaccess: [✓ Yes] No Not Set
Log Level:
Number valid range: 0 - 9

Click **Save** when done. And at last, we need to set the Listeners. Go to the **Listeners** section and click on the **View** button against **Default Listener**. Then, click on the Add button against **Virtual Host Mappings** to add a new mapping and set the following values.

```
Virtual Host: example.com
Domains: example.com
```



Virtual Host * example.com

Domains * example.com

Click **Save** when done. Now, click on the **Graceful restart** button to apply all the changes above and restart the server.

Step 9 - Install SSL

Setting up SSL in OpenLiteSpeed requires us to set up two certificates. A self-signed certificate for the overall server and a Let's Encrypt site-specific server.

The administration panel already comes with a self-signed certificate pre-installed which is available in the `/usr/local/lsws/admin/conf` directory.

Let us create the Self Signed Certificate for the overall server first.

```
$ cd ~
$ openssl req -x509 -days 365 -newkey rsa:4096 -keyout key.pem -out cert.pem -nodes
```

You will get a similar output.

```
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:example.com
Email Address []:navjot@example.com
```

You can press enter through all the fields and leave them empty. Fill in your domain for the Common name and your email address.

Create the directory for the self-signed certificates.

```
$ sudo mkdir /usr/local/lsws/certs
```

Copy the certificate to the `/usr/local/lsws/certs` directory.

```
$ sudo mv *.pem /usr/local/lsws/certs
```

We need to install Certbot to generate free SSL certificates offered by Let's Encrypt.

You can either install Certbot using Debian's repository or grab the latest version using the Snapd tool. We will be using the Snapd version. Debian 12 comes doesn't come with Snapd installed.

Install Snapd package.

```
$ sudo apt install -y snapd
```

Run the following commands to ensure that your version of Snapd is up to date.

```
$ sudo snap install core
$ sudo snap refresh core
```

Issue the following command to install Certbot.

```
$ sudo snap install --classic certbot
```

Use the following command to ensure that the Certbot command can be run by creating a symbolic link to the `/usr/bin` directory.

```
$ sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

Verify the installation.

```
$ certbot --version
certbot 2.6.0
```

Run the following command to generate an SSL Certificate.

Obtain the SSL certificate. The webroot directory is set to the public HTML directory configured earlier.

```
$ sudo certbot certonly --webroot -w /usr/local/lsws/example.com/html/ --agree-tos --no-eff-email --staple-ocsp --preferred-challenges http -m name@example.com -d example.com
```

Generate a **Diffie-Hellman group** certificate.

```
$ sudo openssl dhparam -dsaparam -out /etc/ssl/certs/dhparam.pem 4096
```

Check the Certbot renewal scheduler service.

```
$ sudo systemctl list-timers
```

You will find `snap.certbot.renew.service` as one of the services scheduled to run.

```
NEXT                        LEFT          LAST                        PASSED       UNIT                      ACTIVATES
.....
Sat 2023-09-30 18:12:21 UTC 2h 59min left  Sat 2023-09-30 14:22:18 UTC 50min ago   apt-daily.timer           apt-daily.service
Sat 2023-09-30 18:54:00 UTC 3h 41min left  -                           -           snap.certbot.renew.timer  snap.certbot.renew.service
Sun 2023-10-01 00:00:00 UTC 8h left        -                           -           dpkg-db-backup.timer      dpkg-db-backup.service
```

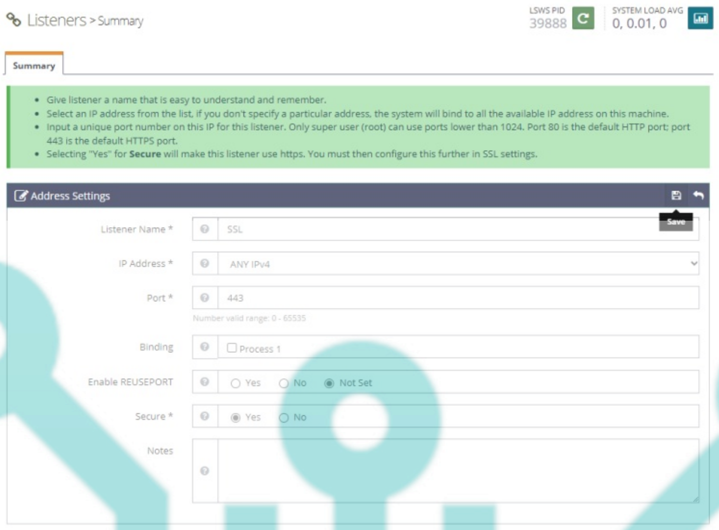
To check whether the SSL renewal is working fine, do a dry run of the process.

```
$ sudo certbot renew --dry-run
```

If you see no errors, you are all set. Your certificate will renew automatically.

Now open the Admin console, go to **Listeners >> Add New Listener,** and add the following values.

```
Listener Name: SSL
IP Address: ANY IPv4
Port: 443
Secure: Yes
```


Listeners > Summary

LSWS PID 39888 SYSTEM LOAD AVG 0, 0.01, 0

Summary

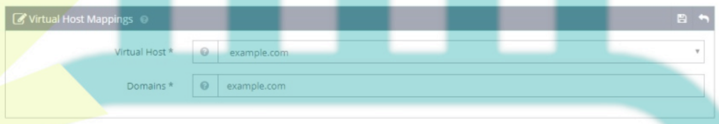
- Give listener a name that is easy to understand and remember.
- Select an IP address from the list, if you don't specify a particular address, the system will bind to all the available IP address on this machine.
- Input a unique port number on this IP for this listener. Only super user (root) can use ports lower than 1024. Port 80 is the default HTTP port; port 443 is the default HTTPS port.
- Selecting "Yes" for **Secure** will make this listener use https. You must then configure this further in SSL settings.

Address Settings

Listener Name *	SSL
IP Address *	ANY IPv4
Port *	443 (Number valid range: 0 - 65535)
Binding	Process 1
Enable REUSEPORT	Yes No [✓ Not Set]
Secure *	[✓ Yes] No
Notes	

Click **Save** when done. Next, go to the **Virtual Host Mappings** section under the SSL Listener by clicking on SSL, clicking on the Add button, and filling in the following values.

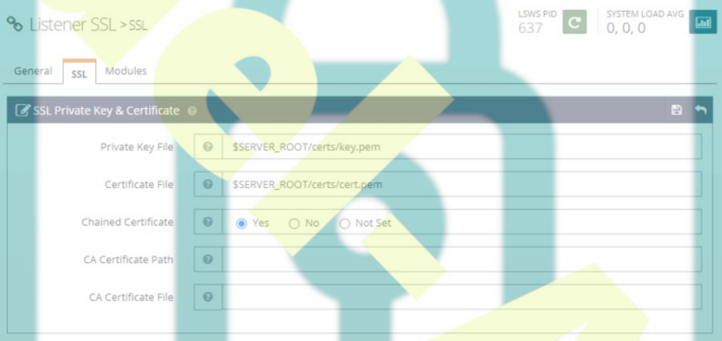
```
Virtual Host: example.com
Domains: example.com
```


Virtual Host Mappings

Virtual Host *	example.com
Domains *	example.com

Click **Save** when done. Next, go to **Listeners >> SSL Listener >> SSL Tab >>SSL Private Key & Certificate** (Edit button) and fill in the following values for the self-signed certificate we created before.

```
Private Key File: /home/user/key.pem
Certificate File: /home/user/cert.pem
Chained Certificate: Yes
```


Listener SSL > SSL

LSWS PID 637 SYSTEM LOAD AVG 0, 0, 0

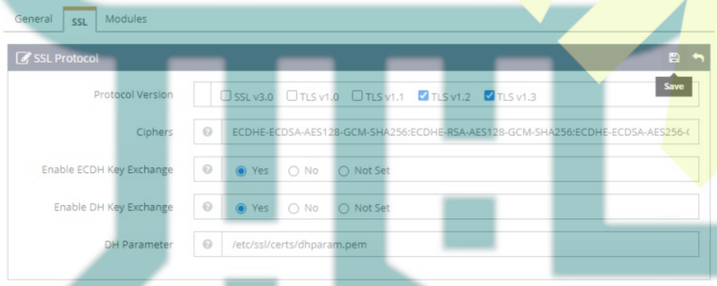
General SSL Modules

SSL Private Key & Certificate

Private Key File	$SERVER_ROOT/certs/key.pem
Certificate File	$SERVER_ROOT/certs/cert.pem
Chained Certificate	[✓ Yes] No Not Set
CA Certificate Path	
CA Certificate File	

Click **Save** when done. Next, go to **Listeners >> SSL Listener >> SSL Tab >> SSL Protocol** (Edit button) and fill in the following values for the SSL protocol and cipher details.

```
Protocol Version: TLS v1.2 TLS v1.3
Ciphers: ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-A
Enable ECDH Key Exchange: Yes
Enable DH Key Exchange: Yes
DH Parameter: /etc/ssl/certs/dhparam.pem
```

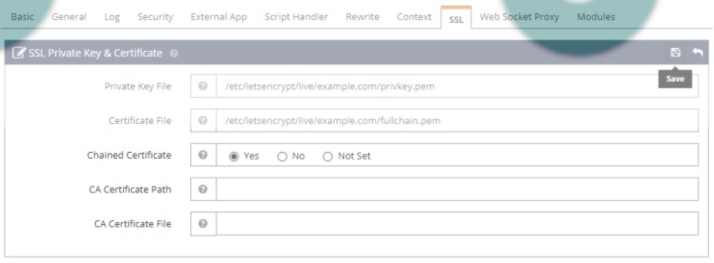

General SSL Modules

SSL Protocol

Protocol Version	SSL v3.0 TLS v1.0 TLS v1.1 [✓ TLS v1.2] [✓ TLS v1.3]
Ciphers	ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-(
Enable ECDH Key Exchange	[✓ Yes] No Not Set
Enable DH Key Exchange	[✓ Yes] No Not Set
DH Parameter	/etc/ssl/certs/dhparam.pem

Click **Save** when done. Next, go to **Virtual Hosts >> example.com >> SSL Tab >> SSL Private Key & Certificate** (Edit button) and fill in the following values with the Let's Encrypt Certificate.

```
Private Key File: /etc/letsencrypt/live/example.com/privkey.pem
Certificate File: /etc/letsencrypt/live/example.com/fullchain.pem
Chained Certificate: Yes
```


Basic General Log Security External App Script Handler Rewrite Context SSL Web Socket Proxy Modules

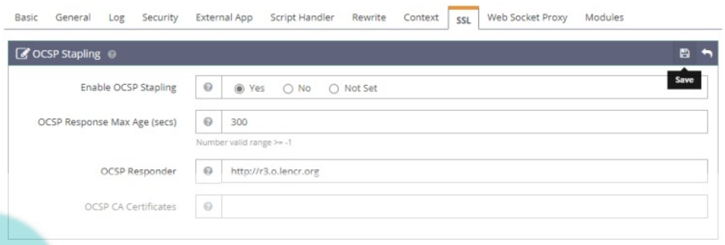
SSL Private Key & Certificate

Private Key File	/etc/letsencrypt/live/example.com/privkey.pem
Certificate File	/etc/letsencrypt/live/example.com/fullchain.pem
Chained Certificate	[✓ Yes] No Not Set
CA Certificate Path	
CA Certificate File	

Click **Save** when done. Next, go to **Virtual Hosts >> example.com >> SSL Tab >> OCSP Stapling** (Edit button) and fill in the following values to enable OCSP Stapling.

```
Enable OCSP Stapling: Yes
```

```
OCSP Response Max Age(Secs): 300
OCSP Responder: http://r3.o.lencr.org
```



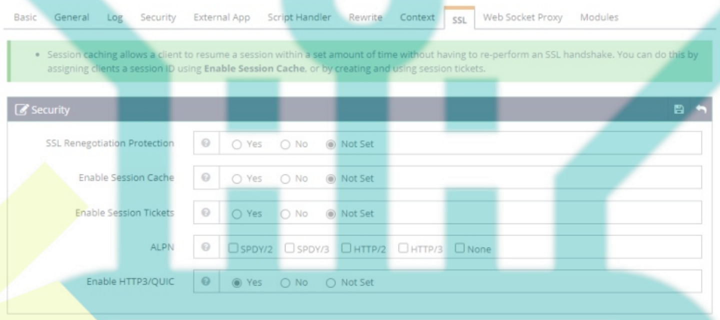
OCSP Stapling

Enable OCSP Stapling	[✓ Yes] No Not Set
OCSP Response Max Age (secs)	300 (Number valid range >= -1)
OCSP Responder	http://r3.o.lencr.org
OCSP CA Certificates	

Click **Save** when done. Next, go to **Virtual Hosts >> example.com >> SSL Tab >> Security** (Edit button) and fill in the following values to enable HTTP3/QUIC protocol.

```
Enable HTTP3/QUIC: Yes
```

We don't need to enable other options because they are on by default.



• Session caching allows a client to resume a session within a set amount of time without having to re-perform an SSL handshake. You can do this by assigning clients a session ID using **Enable Session Cache**, or by creating and using session tickets.

Security

SSL Renegotiation Protection	Yes No [✓ Not Set]
Enable Session Cache	Yes No [✓ Not Set]
Enable Session Tickets	Yes No [✓ Not Set]
ALPN	SPDY/2 SPDY/3 HTTP/2 HTTP/3 None
Enable HTTP3/QUIC	[✓ Yes] No Not Set

Click **Save** when finished.

Restart the server by clicking on the Graceful restart button.

Step 10 - Test Site

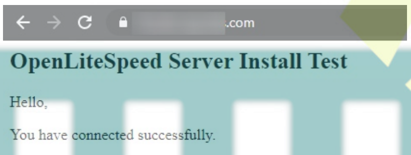
Create a Test file in your `html` directory.

```
$ sudo nano /usr/local/lsws/example.com/html/index.php
```

Paste the following code in the Nano editor.

```
<html>
<head>
    <h2>OpenLiteSpeed Server Install Test</h2>
</head>
    <body>
    <?php echo '<p>Hello,</p>';

    // Define PHP variables for the MySQL connection.
    $servername = "localhost";
    $username = "testuser";
    $password = "Your_Password123";

    // Create a MySQL connection.
    $conn = mysqli_connect($servername, $username, $password);

    // Report if the connection fails or is successful.
    if (!$conn) {
        exit('<p>Your connection has failed.<p>' .  mysqli_connect_error());
    }
    echo '<p>You have connected successfully.</p>';
    ?>
</body>
</html>
```

Save the file by pressing **Ctrl + X** and entering **Y** when prompted. Open the URL `https://example.com` in a browser and you should see the following page.



OpenLiteSpeed Server Install Test

Hello,

You have connected successfully.

The test site is fully functional. You can start using the server to serve dynamic PHP websites and applications.

Conclusion

This concludes our tutorial on installing LOMP Stack (OpenLiteSpeed, MySQL, and PHP) on a Debian 12 server. If you have any questions, post them in the comments below.