# Apache 2.4 and PHP 5/7

## Apache 2.4 Installation

1. Install apache 2.4

```
$ sudo apt update
$ sudo apt install apache2
```

2. Add one of the two commands to add the user to apache's user group:

```
$ sudo adduser <user> www-data
$ sudo usermod -a -G www-data <user>
```

3. Setup your virtual hosts
4. Create sub folders in `/var/log/apache2` if you setup log files for the virtual hosts in sub folders
5. Install and configure Let's Encrypt Certbot

## PHP Installation

1. Install packages

```
$ sudo apt update
$ sudo apt install -y curl wget gnupg2 ca-certificates lsb-release apt-transport-https software-properties-common
```

2. Add the SURY repository to your system

```
$ echo "deb https://packages.sury.org/php/ $(lsb_release -sc) main" | sudo tee /etc/apt/sources.list.d/sury-php.list
```

3. Import the repository key

```
$ wget -qO - https://packages.sury.org/php/apt.gpg | sudo apt-key add -
```

4. Install the desired PHP version, where V is the major and v is the minor version number, for example 5.6 or 7.4

```
$ sudo apt update
$ sudo apt install phpV.v
```

5. Enable modules:

```
$ sudo a2enmod ssl
$ sudo a2enmod proxy
$ sudo a2enmod proxy_http
```

```
$ sudo a2enmod rewrite
```

- How To Install PHP 7.4 on Debian 10 / Debian 9
- Supported Versions
- Expired Debian key issue

# PHP Extensions

- PHP extensions for Joomla:

```
$ sudo apt install phpV.v-{bz2,curl,gd,mbstring,mysql,xml,zip,json}
phpV.v-{bcmath}
```

- PHP extensions for Wiki:

```
$ sudo apt install phpV.v-{bz2,curl,gd,mbstring,mysql,xml,zip,json}
phpV.v-{sqlite3}
```

- Restart the service with one of the 2 commands below:

```
$ sudo service apache2 restart
$ sudo /etc/init.d/apache2 restart
```

# Set or change PHP version

1. Set the desired PHP version for Apache2 and restart the service with one of the 2 commands below:

```
sudo a2dismod phpV.v
sudo a2enmod phpV.v
sudo systemctl restart apache2
sudo service apache2 restart
```

2. Set the desired PHP version for CLI:

```
sudo update-alternatives --set php /usr/bin/phpV.v
sudo update-alternatives --set phar /usr/bin/pharV.v
sudo update-alternatives --set phar.phar /usr/bin/phar.pharV.v
```

3. Check PHP cli Version

```
php -v
```

4. Check PHP apache2 Version: call phpinfo(); in a script
5. Once you have installed a required extension, use the below command to verify it

```
php -m | grep -i mysql
```

- How to Switch between Multiple PHP Version on Debian 9

# Settings

## Harden apache

- change *ServerTokens* and *ServerSignature* in /etc/apache2/conf.d/security.conf
- add *Require all granted* to your web space, possibly exclude black listed ip addresses, and restrict access to phpmyadmin etc. Put a respective conf file into /etc/apache2/conf.d.

## Check ini files

- apache2: load a php file with the following content

```
<?php phpinfo();?>
```

- cli:

```
php --ini
```

## php.ini

- ```
  max_execution_time = 120
  max_input_vars = 2000
  memory_limit = 512M
  post_max_size = 32M
  sys_temp_dir = "/tmp"
  upload_tmp_dir = "/tmp"
  upload_max_filesize = 16M
  date.timezone = Asia/Bangkok
  ```

# Xdebug

1. Open terminal and write following command:

```
php -i > /var/www/html/php_info.txt
```

2. Copy the output from /var/www/html/php_info.txt
3. Go to the Xdebug: Installation Wizard, and paste the output inside the text box on the page. It will analyze the output and will recommend the most suited package of Xdebug.
4. Download that package, for example xdebug-2.8.4.tgz
5. Install the pre-requisites for compiling PHP extensions

```
sudo apt install phpV.v-dev autoconf automake
```
```

6. Unpack the downloaded file with `tar -xvzf xdebug-2.8.4.tgz` within a temp folder, then change to that folder, run *phpize* and check it's output:

```
cd xdebug-2.8.4
phpize
Configuring for:
...
Zend Module Api No:      20170718 (7.2), 20190902 (7.4)
Zend Extension Api No:   3220170718 (7.2), 320190902 (7.4)
```

7. If it does not, you are using the wrong phpize. Please follow this FAQ entry and skip the next step.
8. Run:

```
./configure
make
```

9. Copy the module to:

```
sudo cp modules/xdebug.so /usr/lib/php/{20170718, 20190902}
```

10. Modify the configuration in /etc/php/{7.2, 7.4}/cli/php.ini for Xdebug 2:

```
zend_extension = /usr/lib/php/{20170718, 20190902}/xdebug.so
xdebug.remote_enable=1
xdebug.remote_port=9000 (default: 9000)
xdebug.profiler_enable=0
xdebug.profiler_enable_trigger=1
xdebug.profiler_output_dir=PATH_TO_PROFILER_OUTPUT_DIR
xdebug.remote_log=PATH_TO_LOG/xdebug.log
```

    1. Change the PATH_TO_PROFILER_OUTPUT_DIR to point to the directory you want to receive profiler output. change PATH_TO_LOG to point to the directory where you want to receive xdebug.log.
    2. Make sure that zend_extension = /usr/lib/php/20190902/xdebug.so is below the line for OPcache. Please also update php.ini files in adjacent directories (*apache2* and *cli*), as your system might be configured with a separate php.ini file for the web server and command line.
11. Restart your webserver.
12. Create a PHP page that has phpinfo(). Load it in a browser and look for the info on the Xdebug module. If you see it next to the Zend logo, you have been successful!
13. On the command line, you can also `php -m`. This lists all loaded modules. Xdebug should appear twice there (once under 'PHP Modules' and once under 'Zend Modules').

# phpMyAdmin

Debian 10/11, other than Debian 9, require manual installation of phpmyadmin, phpmyadmin has been removed from Debian's repositories.

## Installation

- Download phpMyAdmin from the Downloads page, scroll down to the table with download links for the latest stable release, and copy the download link ending in tar.gz:

```
wget
https://files.phpmyadmin.net/phpMyAdmin/5.0.2/phpMyAdmin-5.0.2-english.
tar.gz
```

- Unzip the tarball, then move the folder:

```
tar xvf phpMyAdmin-5.0.2-english.tar.gz
sudo mv phpMyAdmin-5.0.2-english/ /usr/share/phpmyadmin
```

## Configuration

- To begin, make a new directory where phpMyAdmin will store its temporary files and set ownership:

```
sudo mkdir -p /var/lib/phpmyadmin/tmp
sudo chown -R www-data:www-data /var/lib/phpmyadmin
```

- In the same folder, create a file with the blowfish secret passphrase and set permissions:

```
sudo vim /var/lib/phpmyadmin/blowfish_secret.inc.php
<?php
$cfg['blowfish_secret'] = '32-CHAR-LONG-SECRET-KEY';
<ESC>:wq
sudo chown root:www-data /var/lib/phpmyadmin/blowfish_secret.inc.php
sudo chmod 640 /var/lib/phpmyadmin/blowfish_secret.inc.php
```

- Copy folder phpmyadmin from a Debian 9 installation to /etc. Check the **apache.conf** file and remove path elements in *php_admin_value open_basedir* which do not exist in your system.
- Symlink the configuration files for apache:

```
cd /etc/apache2/conf-available
ln -s ../../phpmyadmin/apache.conf phpmyadmin.conf
cd ../conf-enabled
ln -s ../conf-available/phpmyadmin.conf phpmyadmin.conf
```

- Symlink the configuration files for phpmyadmin in it's root folder:

```
cd /usr/share/phpmyadmin
ln -s /etc/phpmyadmin/config.inc.php config.inc.php
ln -s /etc/phpmyadmin/config.header.inc.php config.header.inc.php
ln -s /etc/phpmyadmin/config.footer.inc.php config.footer.inc.php
```

- Create a additional config file with the path to the **tmp** directory:

```
vim /etc/phpmyadmin/conf.d/tempdir.php
```

```
<?php
$cfg['TempDir'] = '/var/lib/phpmyadmin/tmp';
```

- Make sure the *phpmyadmin* user has been created in mysql:

```
mysql -u <my-admin-user> -p
SELECT user,host FROM mysql.user;
CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
GRANT USAGE ON phpmyadmin.* TO phpmyadmin
```

- The password is in file /etc/phpmyadmin/config-db.php. Restrict permissions of that file as it contains a password:

```
sudo chown root:www-data /etc/phpmyadmin/config-db.php
sudo chmod 640 /etc/phpmyadmin/config-db.php
```

- Create a regular MariaDB user for the purpose of managing databases through phpMyAdmin, as it's recommended that you log in using another account than the pma user. You could create a user that has privileges to all tables within the database, as well as the power to add, change, and remove user privileges, with this command. Whatever privileges you assign to this user, be sure to give it a strong password as well:

```
sudo mariadb
GRANT ALL PRIVILEGES ON *.* TO 'user'@'localhost' IDENTIFIED BY
'password' WITH GRANT OPTION;
exit
```

- phpMyAdmin Configuration

## Manual Upgrade on Debian 9

- Check latest version of phpMyAdmin
- Download and install

```
cd /usr/share/phpmyadmin/
sudo wget
https://files.phpmyadmin.net/phpMyAdmin/5.1.0/phpMyAdmin-5.1.0-english.
tar.gz
tar xzf phpMyAdmin-5.1.0-english.tar.gz
sudo mv phpMyAdmin-5.1.0-english phpmyadmin
```

- Modify the following lines in /usr/share/phpmyadmin/libraries/vendor_config.php

```
define('TEMP_DIR', '/var/lib/phpmyadmin/tmp/');
define('CONFIG_DIR', '/etc/phpmyadmin/');
```

- See How To Manually Upgrade phpMyAdmin and phpMyAdmin blowfish secret generator

## Security

- You should secure access to phpmyadmin, for example by limiting access to verified ip

addresses

# Windows Subsystem for Linux

- For Windows Subsystem for Linux, create a **Virtual Host** file with document root in */mnt/<drive>/htdocs* or similar, if you need to access it through the Windows file system.
- You should set the apache user to the one who owns the files in the document root, which helps avoiding problems with permissions on the Windows NTFS file system:

```
export APACHE_RUN_USER=<user>
export APACHE_RUN_GROUP=<user>
```

- Restart apache and remove session variables if any:

```
/etc/init.d/apache2 restart
rm /var/lib/php/sessions/*
```

- You can check the current apache user with:

```
ps -ef | egrep '(httpd|apache2|apache)' | grep -v `whoami` | grep -v
root | head -n1 | awk '{print $1}'
```

- Replace the apache default user and group permissions of *www-data* with the one of <user>. run:

```
sudo chown root:<user> /var/lib/phpmyadmin/blowfish_secret.inc.php
sudo chown -R <user>:<user> /var/lib/tmp
sudo chown root:<user> /etc/phpmyadmin/config-db.php
```

- Check permissions of the folder containing the http files according to "Failed to Enumerate Objects in the Container" Windows 10 Error. Most importantly, make sure all files are owned by the same user. Run a (windows) command shell on the windows path of that folder as administrator and run:

```
takeown /F X:\FULL_PATH_TO_FOLDER
takeown /F X:\FULL_PATH_TO_FOLDER /r /d y
icacls X:\FULL_PATH_TO_FOLDER /grant Administrators:F
icacls X:\FULL_PATH_TO_FOLDER /grant Administrators:F /t
```

# apache2 settings

## MaxRequestedWorkers

Modify /etc/apache2/mods-available/mdm-prefork.conf and restart apache2

```
$ sudo apache2ctl -V | grep MPM
vim /etc/apache2/mods-available/mdm-prefork.conf
  MaxRequestedWorkers 400
```

```
  ServerLimit 400
$ sudo service apache2 restart
```

### Links

- Correct steps to add another domain to existing certificate
- Revoking Let's Encrypt Certificates
- Correct Way to Delete a Certbot SSL Certificate
- What is the difference of certbot and certbot-auto?
- Remove a domain in Let's Encrypt

## SSL for localhost

### Ignore invalid certificates

- You can just use the default ssl conf file in /etc/apache2/sites-available which makes use of the snakeoil certificate. Modify DocumentRoot and add Directory permissions.
- Paste this in chrome, enable, and chrome will ignore invalid certificates for localhost:

```
chrome://flags/#allow-insecure-localhost
```

### Create certificate for localhost

1. Make a folder to keep your certificate files and change to that folder, for example ~/certs/ssl.
2. Generate RootCA.pem, RootCA.key & RootCA.crt:

```
openssl req -x509 -nodes -new -sha256 -days 1024 -newkey rsa:2048 -
keyout RootCA.key -out RootCA.pem -subj "/C=US/CN=Example-Root-CA"
openssl x509 -outform pem -in RootCA.pem -out RootCA.crt
```

3. Create a file domains.ext that lists all your local domains:

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment,
dataEncipherment
subjectAltName = @alt_names
[alt_names]
DNS.1 = localhost
DNS.2 = localhost.yourdomain.tld
DNS.3 = machine1.yourdomain.tld
DNS.4 = machine2.yourdomain.tld
```

4. Generate localhost.key, localhost.csr, and localhost.crt:

```
openssl req -new -nodes -newkey rsa:2048 -keyout localhost.key -out
localhost.csr -subj "/C=US/ST=YourState/L=YourCity/O=Example-
```

```
Certificates/CN=localhost.local"
openssl x509 -req -sha256 -days 1024 -in localhost.csr -CA RootCA.pem -
CAkey RootCA.key -CAcreateserial -extfile domains.ext -out
localhost.crt
```

5. Configure Apache:

```
SSLEngine on
SSLCertificateFile "/home/user/certs/ssl/localhost.crt"
SSLCertificateKeyFile "/home/user/certs/ssl/localhost.key"
```

6. Restart Apache
7. At this point, the site would load with a warning about self-signed certificates. In order to get a green lock, your new local CA has to be added to the trusted Root Certificate Authorities in your OS or browser.
    - For Windows 10 Chrome & Edge: Windows 10 recognizes .crt files, so you can right-click and open *RootCA.crt*.
    - Select *Install Certificate...*, select *Local Machine*, then select *Trusted Root Certification Authorities* and confirm.
    - You might need to clear cookies and cache for the browser to pick up the certificate from the server
8. If you want to utilitze the certificate for an Endian Firewall, do the following:
    - Rename the files *server.crt*, *server.csr*, and *server.key* in folder /etc/httpd and etc/httpd/cert
    - Copy the newly generate certificate files *localhost.crt*, *localhost.csr*, and *localhost.key* to *server.crt*, *server.csr*, and *server.key* in folder /etc/httpd
    - Copy the newly generate certificate file *localhost.crt* to *server.crt* in folder /etc/httpd/certs and append the parameters from the renamed original *server.crt* file
    - Restart httpd
    - You can check the domain names included in the original certificate:

```
openssl x509 -text < $CERT_FILE
```

#### Links

- How to create an HTTPS certificate for localhost domains (reference)
- Let's Encrypt: Certificates for localhost
- How to Get SSL HTTPS for localhost
- Troubleshooting Apache SSL Certificate Errors

# Proxy

* Setup a VirtualHost on your main apache server, which for this example is now called “proxy”. * There needs to be another (regular) VirtualHost file on the backuppc server, which for this example is now called “host”. * The SSL certificates are served from the “proxy” through access to https://sub.domain.tld * The “host” serves an unencrypted site through port 80. This assumes your local network is secure.

### VirtualHost on the "proxy" server

* sub.domain.tld: external domain name with which you access the “host” behind the “proxy” * host.yourdomain.tld: internal domain name of your “host”. You may also choose to have both names the same.

```
<VirtualHost *:80>
    ServerName sub.domain.tld
    Redirect 301 / https://sub.domain.tld
<VirtualHost>
<VirtualHost *:443>
    ServerName sub.domain.tld
    ServerAdmin you@domain.tld
    DocumentRoot /var/www/html/yoursite
    SSLEngine on
    RedirectMatch ^/$ /yourapp/ # use this if backuppc is not the default
app, or if you need to access another app on the same server
    <Location "/yourapp/">
          ProxyPass "http://host.yourdomain.tld/yourapp/"
          ProxyPassReverse "http://host.yourdomain.tld/yourapp/"
          Require all granted
    </Location>
    # add other options such as Files and Directory permissions
    Include /etc/letsencrypt/options-ssl-apache.conf
    SSLCertificateFile /etc/letsencrypt/live/sub.domain.tld/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/sub.domain.tld/privkey.pem
</VirtualHost>
```

### Links

- Use apache as a HTTPS to HTTP Proxy
- Reverse Proxy Guide
- Simple Apache reverse proxy example

## Links

- How To Install PHP 7.4 / 7.3 / 7.2 / 7.1 on Debian 10 / Debian 9
- DEB.SURY.ORG
- How To Install PHP (7.2, 7.1 & 5.6) on Debian 9 Stretch
- How to Install PHP on Debian 9
- How to Switch between Multiple PHP Version on Debian 9
- How to Install Multiple PHP Version with Apache on Debian 9
- php.net - SimpleXML Installation
- php.net - The configuration file
- How To Specify A Custom php.ini For A Web Site
- SSL Checker
- How to Setup Auto-Renew for Let’s Encrypt SSL Certificates (Apache)
- Correct Way to Delete a Certbot SSL Certificate

- How To Secure Apache with Let's Encrypt on Debian 9
- Certbot-auto
- Letsencrypt add domain to existing certificate
- Certbot-auto deployment best practices
- Multiple servers one IP address
- What is the size limit of a post request?
- Xdebug: Installation
- GitHub: Xdebug Installation
- Xdebug: all settings
- How To Install and Secure phpMyAdmin on Debian 9
- How To Install phpMyAdmin From Source on Debian 10
- How To Install phpMyAdmin with Apache on Debian 10
- Problem with phpMyAdmin and PHP 7.2
- How to Manually Upgrade phpMyAdmin
- Download phpMyAdmin
- phpMyAdmin trouble shooting
- Getting Chrome to accept self-signed localhost certificate
- How to create a self-signed certificate with OpenSSL
- How to install an SSL Certificate on Apache
- Securing Apache and blocking a list of ip addresses
- Apache keeps going down on a Plesk server: server reached MaxRequestWorkers setting

From:
https://wiki.condrau.com/ - **Bernard's Wiki**

Permanent link:
**https://wiki.condrau.com/deb9:apache**

Last update: **2024/04/20 15:05**