Apache 2.4 and PHP 7/8

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
6. If you are migrating from an old server, follow the How to migrate a (web) server guide

Apache 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.

MaxRequestedWorkers

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

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

Check configuration

- `$ sudo apachectl configtest`

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 7.4 or 8.1

```
$ 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 8.0 on Debian 11/10/9
- How To Install PHP 7.4 on Debian 10 / Debian 9
- How To Install PHP (8.1, 7.4 & 5.6) on Debian 11
- Supported Versions
- Expired Debian key issue

PHP Extensions

- PHP extensions for Joomla:

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

- PHP extensions for Wiki:

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

- PHP extensions for phpMyAdmin:

```
$ sudo apt install phpV.v-mbstring
```

- PHP extensions for vps:

```
$ sudo apt install phpV.v-{curl xml zip}
```

- Required for PHP7:

```
$ sudo apt install phpV.v-json
```

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

```
$ sudo systemctl restart apache2
$ sudo service 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

PHP Settings

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-3.1.5.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-3.1.5.tgz` within a temp folder, then change to that folder, run *phpize* and check it's output:

```
cd xdebug-3.1.5
phpize
Configuring for:
...
Zend Module Api No:      20200930 (8.0), 20210902 (8.1)
Zend Extension Api No:   420200930 (8.0), 420210902 (8.1)
```

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/{20200930, 20210902}
```

10. Create `/etc/php/{8.0, 8.1}/apache2/conf.d/99-xdebug.ini` for Xdebug 3 and add lines:

```
zend_extension     = xdebug
xdebug.remote_port = 9000 (default: 9003)
xdebug.mode        = debug
```

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').

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
```

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

Follow the guide Apache Server with Proxy to setup access to proxied servers.

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/deb11:apache

Last update: **2023/01/16 19:59**