Apache Web-Server

Vorbereitung

1. Basissystem einrichten
2. Firewall einrichten & aktivieren
3. SSH-Zugang einrichten
4. MySQL-Datenbank
5. PHP einrichten

- https://httpd.apache.org/docs/2.4/
- https://wiki.ubuntuusers.de/Apache_2.4/
- https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-ubuntu-20-04

Installation

```
sudo apt install apache2
```

Apache Status abfragen

```
sudo systemctl status apache2
```

Verzeichnis für Webseiten anlegen.

```
sudo mkdir /var/www/sites
```

Default-Einstellungen für die Webseiten-Verzeichnisse mit ACL. www-data wird als User & Gruppe vom Apache-Webserver genutzt. Wenn ein <FTPUSER> Daten auf den Server hochlädt, behält die Gruppe www-data und somit der Apache-Webserver alle erforderlichen Zugriff-Rechte um in diesem Bereich handeln zu können.

```
sudo setfacl -dm g:www-data:rwx /var/www/sites/
```

Den Main-User in die Gruppe www-data aufnehmen, damit er Schreibrechte im Bereich der HTML-Seiten hat.

```
sudo usermod -aG www-data <SUDO-USER>
```

Firewall

Firewall einrichten
Freigabe für Webseiten

```
sudo ufw allow 80/tcp
```

Firewall-Freigabe für verschlüsselte Webseiten

```
sudo ufw allow 443/tcp
```

FTP-Zugang

```
sudo apt install vsftpd
```

```
sudo nano /etc/vsftpd.conf
```

```
local_enable=YES
anonymus_enable=NO
write_enable=YES
```

```
sudo systemctl reload vsftpd
```

FTP-User anlegen - Namen festlegen. Der Zugriff für den <FTPUSER> wird auf den u.a. Bereich eingeschränkt, das gilt auch für den SSH-Zugriff über die Shell. Der o.a. <SUDO-USER> sollte daher nicht hier verwendet werden.

```
sudo adduser <FTPUSER> --no-create-home
```

FTP-User in die Apache-Gruppe aufnehmen

```
sudo usermod -aG www-data <FTPUSER>
```

Zugang für FTP-User einschränken auf den Bereich der Webseiten im Verzeichnis /var/www

```
sudo nano /etc/ssh/sshd_config
```

```
Match User <FTPUSER>
  X11Forwarding no
  AllowTcpForwarding no
  PermitTTY no
  ForceCommand internal-sftp
  ChrootDirectory /var/www/
  PasswordAuthentication yes
```

```
sudo systemctl reload ssh
```

Für den FTP-Zugriff von <FTPUSER> mit o.a. Einschränkung (Match User ...) ist eine Besonderheit zu beachten, da sonst kein Zugang per FTP möglich ist.
Der Owner vom Verzeichnis /var/www, bzw. Pfad aus Parameter **ChrootDirectory**, muss root sein und Gruppe oder Sonstige dürfen keine Schreibrechte besitzen.
Diese Vorgaben gelten nur für das Hauptverzeichnis, nicht für darin enthaltene Unterverzeichnisse, daher im folgenden die Option -R nicht angewandt.
Ggf. Einstellungen anpassen::

Setze Besitzer.

```
sudo chown root:root /var/www/
```

Entferne Schreibrechte (w) für Gruppe (g)

```
sudo chmod g-w /var/www/
```

Entferne Schreibrechte für Sonstige (o)

```
sudo chmod o-w /var/www/
```

Anschließend hat der FTP-User Lese-Zugriff auf das Verzeichnis /var/www
Schreibrechte müssen dann in weiteren Unterverzeichnissen (var/www/html, /var/www/sites, etc.) für die Gruppe www-data erteilt werden bzw. müssten eigentlich bereits für den Apache-Webserver vorhanden sein - siehe auch oben unter Installation.

Webseiten & virtueller Host

Verzeichnis für die HTML-Daten über die Shell anlegen

```
sudo mkdir /var/www/sites/BLUEGNU.DE
```

```
sudo chmod -R 775 /var/www/sites/BLUEGNU.DE
```

Die Programm-Dateien können dann mit einem FTP-Programm übertragen werden.

Je nach Anwendungsfall ist es sinnvoll sensible Daten außerhalb des direkten Webseitenzugriffs und in andere Verzeichnisse zu platzieren. Der Apache-Webbrowser (User www-data) muss Zugriff darauf haben → Z.B. im Verzeichnis /var/www/data/…. Das muss über die Konfiguration der jeweilige Webseite definiert werden. Z.B. bei Nextcloud über die …config/config.php

```
sudo mkdir /var/www/data/
```

Auch hier kann man wie bereits im Verzeichniss /var/www/sites/ den Bereich für die Gruppe www-data per Default mit Rechten versehen.

```
sudo setfacl -dm g:www-data:rwx /var/www/data/
```

Alternativ:

```
sudo chown -R www-data:www-data /var/www/data/
```

und

```
sudo chmod -R 775 /var/www/data/
```

Apache-Konfiguration virtueller Host der Domain

```
sudo nano /etc/apache2/sites-available/BLUEGNU.DE.conf
```

```
<VirtualHost *:80>
  ServerName BLUEGNU.DE
  ServerAdmin email@BLUEGNU.DE
  DocumentRoot /var/www/sites/BLUEGNU.DE
  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
  <Directory /var/www/sites/BLUEGNU.DE>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    Order allow,deny
    allow from all
  </Directory>
</VirtualHost>
```

Beim Erstellen der Let's-Encrypt-Zertifikate gab es Probleme mit dem ServerAlias „www.…“. Daher wurde hier auf die Einstellung des ServerAlias verzichtet. Es ist möglich, die www.<DOMAIN> als eigenständige Subdomain zu verarbeiten (s.u.) und dafür eine eigenes Zertifikat zu erstellen. Für Documentroot wird das selbe Verzeichnis angegeben.

Virtuellen Host im Apache-Web-Server registrieren

```
sudo a2ensite BLUEGNU.DE.conf
```

Defaultseite deaktivieren. Dadurch wird erste Domain (nach Alphabet) zur Defaultseite, sollte z.B. nur die Server-IP-Adresse über einen Webbrowser aufgerufen werden, oder die per DNS hierhin umgeleitete Domain nicht zu finden sein.

```
sudo a2dissite 000-default.conf
```

Alternativ kann eine permanente Umleitung in diese Datei geschrieben werden, um eine bestimmte Domain aufzurufen.

```
<VirtualHost *:80>
  RedirectPermanent / https://duckduckgo.de/
</VirtualHost>
```

Hostname und Fully-Qualified Host Name (FQHN) festlegen

```
sudo hostnamectl set-hostname server.BLUEGNU.DE
```

Einstellungen für den Apacheserver testen

```
sudo apache2ctl configtest
```

Sollte es Probleme mit dem FQDM[1)] geben, dann ggf. die Datei hosts anpassen. Reihenfolge beachten.

```
sudo nano /etc/hosts
```

```
127.0.1.1 server.BLUEGNU.DE server
127.0.0.1 localhost
```

Wenn Test ok, dann alles aktivieren

```
sudo systemctl reload apache2
```

Bei Apache registrierte Hosts

```
sudo apache2ctl -S
```

Bei Apache registrierte Module

```
sudo apache2ctl -M
```

Mit FTP-Zugang die Struktur anlegen und Dateien übertragen

Simple HTML-Datei

```
sudo nano /var/www/sites/BLUEGNU.DE/index.html
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>TEST-TITEL</title>
  </head>
  <body>
  <center>
  <h1>TESTSEITE auf Apache-Server erreichbar</h2>
  </center>
  </body>
</html>
```

Proxy-Server

Erweiterungen installieren

```
sudo a2enmod proxy proxy_ajp proxy_http rewrite deflate headers
proxy_balancer proxy_connect proxy_html ssl
```

Konfiguration mit SSL, hier für eine Subdomain, siehe auch Apache Web-Server:

```
sudo nano  /etc/apache2/sites-available/<SUBDOMAIN>.conf
```

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
```

```
  ServerName calibre.<DOMAIN>.de
  ...
  ...
  ProxyPass        / http://localhost:8080/
  ProxyPassReverse / http://localhost:8080/
  ...
  ...
</VirtualHost>
</IfModule>
```

<DOMAIN> ersetzen
DocumentRoot kann entfernt oder deaktiviert (#) werden

```
sudo apachectl configtest
```

```
sudo systemctl restart apache2
```

Anschließend kann die Firewall für den Port 8080 (sofern verwendet) wieder deaktiviert werden, da der Aufruf nun über den Domainnamen bzw. Subdomain erfolgt. Siehe auch Firewall.

SSL-Zertifikate

Einrichtung: Ubuntu -> Apache
SSL-Zertifikat testen

DNS-Einstellungen

Siehe auch DNS-Einstellungen

Die Web-Adressen müssen zuvor beim Provider auf den Server umgeleitet werden.
Das erfolgt über die DNS-Zeiger der Domain(s):

- A → @ → IPV4-Adresse des Servers
- A → www → IPV4-Adresse des Servers (sofern www gewünscht)

Sollen zusätzlich Subdomains eingerichtet werden:

- A → * → IPV4-Adresse des Servers

In diesem Beispiel werden durch das (zweite) * alle Subdomains umgeleitet.
Es ist natürlich auch möglich, einzelne Subdomains gezielt auf einen Server umzuleiten. So können dann verschiedene Subdomains auf unterschiedliche Servern geleitet werden.
Beispiel für die Subdomain „sub“:

- A → sub → IPV4-Adresse des Servers

Let's-Encrypt

```
sudo apt install certbot python3-certbot-apache
```

Zertifikat(e) erstellen

```
sudo certbot --apache
```

Durch das Erstellen eines Zertifikats wird automatisch zusätzlich eine Datei für den virtuellen Host der SSL-Verbindung angelegt. Z.B. für die Hostdatei „BLUEGNU.DE.conf“, zusätzlich hinzu „/etc/apache2/sites-available/BLUEGNU.DE-le-ssl.conf“[2)].

Timer-Einstellungen für Autorenew

```
sudo systemctl status certbot.timer
```

Erneuerung Testen

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

Zertifikate anzeigen

```
sudo certbot certificates
```

Zertifikat löschen

```
sudo certbot delete
```

Weitere Domains

Hier als Beispiel die Subdomain für das wiki > https://wiki.BLUEGNU.DE
Verzeichnis für die Daten anlegen

```
sudo mkdir /var/www/sites/wiki.BLUEGNU.DE
```

Die Programm-Dateien können dann mit einem FTP-Programm übertragen werden.

Apache-Konfiguration virtueller Host der Domain

```
sudo nano /etc/apache2/sites-available/wiki.BLUEGNU.DE.conf
```

```
<VirtualHost *:80>
  ServerName wiki.BLUEGNU.DE
  ServerAdmin email@BLUEGNU.DE
  DocumentRoot /var/www/sites/wiki.BLUEGNU.DE
  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
  <Directory /var/www/sites/wiki.BLUEGNU.DE>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    Order allow,deny
    allow from all
  </Directory>
</VirtualHost>
```

Virtuellen Host im Apache-Web-Server registrieren

```
sudo a2ensite wiki.BLUEGNU.DE.conf
```

Einstellungen für den Apacheserver testen

```
sudo apache2ctl configtest
```

Wenn Test ok, dann alles aktivieren

```
sudo systemctl reload apache2
```

SSL-Zertifikat registrieren

```
sudo certbot --apache
```

1)
Fully qualified domain name
2)
le für Let's-Encrypt