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
```

Es sollten hier keine ACL-Rechte vergeben sein. ggf. löschen (kein + am Ende > ls -l):

```
sudo setfacl -bR /var/www/sites
```

Eine Gruppe für alle User der Webseiten erstellen

```
sudo groupadd web
```

Die der Apache-User muss in dieses Gruppe aufgenommen werden

```
sudo usermod -aG web www-data
```

Betroffene User in die Gruppe aufnehmen

```
sudo usermod -aG web <USER>
```

Rechte vergeben

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

```
sudo find /var/www/sites -type d -exec chmod 2775 {} \;
```

```
sudo find /var/www/sites -type f -exec chmod 664 {} \;
```

Testen, ob nur root Rechte auf alle Ebenen hat (sollte sein)

```
namei -l /var/www/sites/
```

Firewall

Firewall einrichten
Freigabe für Webseiten

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

Firewall-Freigabe für verschlüsselte Webseiten

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

FTP-Zugang

Der FTP-Zugang sollte nur als SFTP zugelassen werden und wird konfiguriert über SSH-Verbindungen.

Eingeschränkter Zugang

Soll ein Kunde/Anwender eingeschränkte Rechte für eine oder mehrere Webseiten erhalten, kann dies wie folgt eingerichtet werden.

Anwender Y ⇒ <ANY>

User anlegen

```
sudo useradd -M -s /usr/sbin/nologin <ANY>
```

-s = Shell entfernen, kein Zugriff über SSH.

Passwort vergeben

```
sudo passwd <ANY>
```

Web-Gruppe zuordnen

```
sudo usermod -aG web <ANY>
```

Alternativ kann der Zugriff auch mit einem SSH-Schlüssel erfolgen. Dazu den öffentlichen Schlüssel in der Datei /home/<ANY>/.ssh/authorized_keys registrieren bzw. hinterlegen. Wird der User wie hier

beschrieben angelegt, wird sein Home-Verzeichnis nicht automatisch mit angelegt. Das muss dann manuell erfolgen. Erforderlich, auch wenn der User später keinen Zugriff darauf haben wird.
Rechte für das Verzeichnis setzen:

```
sudo chown -R <ANY>:<ANY> /home/<ANY>/.ssh
```

```
sudo chmod 700 /home/<ANY>/.ssh
```

```
sudo chmod 600 /home/<ANY>/.ssh/authorized_keys
```

Hier wird der User eingesperrt:
Chroot-Verzeichnis anlegen

```
sudo mkdir -p /sftp-chroot/<ANY>/webseite_1
```

```
sudo chown root:root /sftp-chroot
```

```
sudo chmod 755 /sftp-chroot
```

```
sudo chown root:root /sftp-chroot/<ANY>
```

```
sudo chmod 755 /sftp-chroot/<ANY>
```

Einbinden

```
sudo mount --bind /var/www/sites/<WEBSEITENDATEIEN> /sftp-
chroot/<ANY>/webseite1
```

Dauerhaft machen

```
sudo nano /etc/fstab
```

```
/var/www/sites/<WEBSEITENDATEIEN> /sftp-chroot/<ANY>/webseite1 none bind 0 0
```

SSH für jeden User mit eingeschränkten Rechten extra konfigurieren

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

Einrichtung mit Password

```
Match User <ANY>
  PasswordAuthentication yes
  PubkeyAuthentication no
  ChrootDirectory /sftp-chroot/<ANY>
  ForceCommand internal-sftp -d /web
  AllowTcpForwarding no
  X11Forwarding no
```

Einrichtung mit Key-Zugriff

```
Match User <ANY>
  PasswordAuthentication no
  PubkeyAuthentication yes
  ChrootDirectory /sftp-chroot/<ANY>
  ForceCommand internal-sftp -d /web
  AllowTcpForwarding no
  X11Forwarding no
```

Syntax des Ports überprüfen

```
sudo sshd -t
```

Restart

```
sudo systemctl restart ssh
```

Webseiten & virtueller Host

Verzeichnis für die HTML-Daten anlegen

```
sudo mkdir /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/data/…. Das muss über die Konfiguration der jeweilige Webseite definiert werden. Z.B. bei Nextcloud über die …config/config.php

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

Für dieses Verzeichnis sollten die gleichen Zugriffsrechte konfiguriert sein, wie für das Verzeichnis /var/www/sites

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>
```

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
```

1)

Fully qualified domain name

2)

le für Let's-Encrypt

From:
https://wiki.bluegnu.de/ - **wiki**

Permanent link:
https://wiki.bluegnu.de/doku.php/open:it:apache?rev=1777230267

Last update: **2026/04/26 21:04**