Inhaltsverzeichnis

Nextcloud-Server

Ein Beispiel einer reelen Konfiguration. Hier wird nur die Installation und Konfiguration in Kurzform aufgelistet. Eine Installationsanleitung findet sich unter folgenden Links: Nextcloud einrichten. Was ist eine Cloud? Allgemeine erklärt.

Webserver einrichten (nginx)

Damit Ihr Rechner und damit auch Ihre Cloud PHP-Skripte ausführen kann und über das Internet erreichbar ist, müssen Sie zunächst einen Web- und Datenbankserver einrichten. Wir verwenden hierfür **nginx**.

```
sudo apt-get install nginx php7.3-fpm php7.3-mysql mariadb-server mariadb-
client
```

Sobald Sie gefragt werden, ob Sie fortfahren möchten, bestätigen Sie durch Drücken der **[Enter]**-Taste.

Richten Sie nun den Datenbank-Server ein:

```
sudo mysql_secure_installation
```

Falls Sie nach Ihrem Passwort gefragt werden, überspringen Sie die Abfrage mit der **[Enter]**-Taste, da aktuell noch keines festgelegt ist. Die nachfolgenden Abfragen bestätigen Sie jeweils durch Eingabe von **Y** und **[Enter]**. Sobald Sie nach einem neuen Passwort gefragt werden, geben Sie ein sicheres Passwort für den Zugang zur Datenbank ein und notieren Sie es sich. Das Passwort wird während der Eingabe nicht angezeigt, lassen Sie sich davon nicht irritieren.

```
Set root password? [Y/n] Y
Remove anonymous users? [Y/n] Y
Disallow root login remotely? [Y/n] Y
Remove test database and access to it? [Y/n] Y
Reload privilege tables now? [Y/n] Y
```

Anschließend installieren Sie durch Eingabe dieses Befehls noch einige benötigte Pakete. Darunter ist auch der Editor vim oder nano, mit dem Sie Dateien bearbeiten können. Wenn erforderlich, bestätigen Sie auch hier eventuelle Nachfragen mit Hilfe der **[Enter]**-Taste.

```
# Anstelle von vim können Sie auch nano als Editor installieren.
sudo apt-get install php-pear php7.3-xml php7.3-cli php7.3-gd php7.3-intl
php7.3-curl php7.3-zip php7.3-mbstring vim openssl
```

Öffnen Sie die PHP-Konfigurationsdatei mit dem folgenden Befehl, damit auch bei zukünftigen Updates weniger Änderungen notwendig sind:

```
sudo vi /etc/php/7.3/fpm/pool.d/www.conf
```

Sie sollten durch Scrollen mit den Pfeiltasten einen „listen“-Zeile ohne „;“ davor finden. Ändern Sie die Zeile gegebenenfalls ab (zB von „listen = /var/run/php7.3-fpm.sock“), sodass sie wie folgt aussieht:

```
listen = 127.0.0.1:9000
```

Durch **[Einfg]** können Sie die Zeile entsprechend bearbeiten. Drücken Sie anschließend **[Esc]**, geben Sie „:wq“ (=Schließen und Speichern) ein und drücken Sie **[Enter]** um die Datei zu speichern.

Port-Freigabe & Dynamisches DNS

Damit sichergestellt ist, dass Sie von außen auf Ihren Server zugreifen können, müssen Sie in Ihrem Router eine sogenannte Port-Freigabe einrichten. Dies unterscheidet sich von Router zu Router. Zunächst müssen Sie jedoch Ihre IP-Adresse kennen. Geben Sie dafür auf dem Raspberry Pi folgendes ein:

```
ifconfig
```

Irgendwo in der Ausgabe wird so ein ähnlicher Eintrag zu finden sein: „inet Adresse:192.168.178.47“. Das Unterstrichene, in Ihrem Fall sieht das vermutlich anders aus, ist Ihre lokale IP-Adresse.

Richten Sie jetzt auf Ihrem Router eine (HTTPS)-Port-Freigabe (TCP) für den Eingangsport 443, Ausgangsport 443 und den Rechner mit Ihrer lokalen IP-Adresse ein. Je nach Router-Modell ist dies unterschiedlich, hier Anleitungen für einige Router:

Fritz!Box Port-Freigabe

TP-Link-Router

Video: Port-Freigabe bei einem T-Online-Router

Video: Port-Freigabe bei einer Easy Box

Andere Modelle: Suchen Sie nach „Port-Freigabe“ und dem Namen Ihres Routers, um eine passende Anleitung zu finden

In der Regel werden Sie nach außen - gemeint ist als nicht die interne IP-Adresse Ihres Rechners - eine dynamische IP-Adresse haben. Ihre IP wird sich also regelmäßig ändern. Deshalb müssen Sie einen sogenannten dynamischen DNS-Anbieter auswählen. Wählen Sie einfach einen von diesen Anbietern aus:

dynv6

TwoDNS

SPDNS

DNSDynamic.org

GOIP

ClickIP.de

Melden Sie sich nun bei einer dieser Seiten für einen kostenlosen Tarif an. Dort werden Sie einen sogenannten Hostnamen anlegen, z.B. „muellercloud.twodns.de“. Über diesen Namen ist Ihre Cloud später erreichbar. Auf der Seite des dynamischen DNS-Anbieters werden Sie häufig auch Informationen vorfinden, um diesen Anbieter in Ihrem Router einzurichten.

Dort stehen keine Informationen dazu? Dann probieren Sie es mal über das Administrationsmenü Ihres Routers (bei der Fritz!Box „fritz.box“ im Browser eingeben) und suchen dort nach „Dynamisches DNS“. Haben Sie einen solchen Menüpunkt gefunden, geben Sie dort ggf. Ihre Login-Daten und Ihren Host-Namen ein.

Sobald Sie sich sowohl bei einem Anbieter registriert als auch diesen korrekt in Ihrem Router eingerichtet haben, können wir mit dem nächsten Schritt fortfahren.

SSL-Verbindung einrichten

Nun sind Sie fast am Ende dieses Tutorials angekommen. Nach den nächsten zwei Schritten werden Sie einen korrekt konfigurierten, kleinen Server eingerichtet haben. Zunächst aber müssen wir noch dafür sorgen, dass Sie Ihre Dateien via SSL verschlüsselt synchronisieren können.

Nachfolgend wird ein sicheres, aber nicht-vertrauenswürdiges (=Bestätigung im Browser erforderlich) Zertifikat generiert. Wenn Sie Ihre Cloud für andere Nutzer öffnen möchten, können Sie mit entsprechenden Kenntnisen diesem Let's Encrypt-Tutorial folgen.

Führen Sie den folgenden Befehl aus, um ein neues Verzeichnis anzulegen und in dieses zu wechseln:

```
sudo mkdir -p /var/www/{ssl,cloud} && cd /var/www/ssl
```

Damit auch der Benutzer Pi später Zugriff auf die Dateien hat, legen wir eine neue Gruppe an und fügen Pi hinzu:

```
# Anstelle von Pi kann jeder beliebiger Gruppenname oder Username stehen.
sudo addgroup pi && sudo usermod -aG pi pi
```

Anschließend ändern wir die Rechte an den Ordnern:

```
sudo chown -R www-data:pi /var/www/cloud && sudo chown -R pi:pi /var/www/ssl
&& sudo chmod -R 775 /var/www/cloud && sudo chmod -R 770 /var/www/ssl
```

Mit dem folgenden Befehle erstellen Sie nun ein SSL-Zertfikat, welches Sie selbst signieren:

```
sudo openssl req -x509 -nodes -days 9999 -newkey rsa:2048 -keyout
cloudssl.key -out cloudssl.crt
```

Bei Nachfragen bezüglich eines „Challenge-Passwords“ können Sie sich ein Passwort ausdenken und mit **[Enter]** bestätigen. Als „Common Name“ können Sie Ihren Hostnamen (z.B. meinecloud.twodns.de) angeben. Sonstige Felder können Sie per einfachen Druck auf **[Enter]** leer lassen.

Besondere Sicherheit bekommen wir, indem wir den sogenannten Diffie-Hellman-Schlüsselaustausch ermöglichen. Machen Sie sich einen Kaffee oder Tee und führen Sie vorher diesen Befehl aus. Das Ausführen des Befehls kann einige Minuten (in meinem Test ca. 15-20 Minuten) dauern:

```
openssl dhparam 2048 > /var/www/ssl/dhparam.pem
```

Anschließend nehmen wir die richtige Konfiguration für PHP vor, um sicherzustellen, dass auch große Dateien problemlos hochgeladen werden können. Führen Sie dafür den folgenden Befehl aus, der eine für Sie vorbereitete Datei von unserer Website zu Ihnen herunterlädt:

```
sudo wget -q -O /etc/php/7.3/fpm/conf.d/user.ini - https://eigene-cloud-
einrichten.de/getPHPINI
```

```
# Der Inhalt der Datei sieht wie folgt aus:

### DEFAULT INI FILE ###
memory_limit=256M
upload_max_filesize=1G
post_max_size=2G
session.save_path=/tmp register_globals=off
date.timezone = "Europe/Berlin"
display_errors=off
error_reporting=
log_errors=off
max_file_uploads=25
max_input_time=-1
output_buffering=0
request_order=
serialize_precision=50
expose_php=off
cgi.fix_pathinfo=0
```

Starten Sie nun den Webserver und PHP neu:

```
sudo service nginx restart && sudo service php7.3-fpm restart
```

MySQL-Datenbank einrichten

Wir werden nun noch eine MySQL/MariaDB-Datenbank einrichten.

a. Falls Sie bereits oben ein Datenbank-Passwort gesetzt haben, geben Sie in der Kommando-Zeile Ihres Rechners den nachfolgenden Befehl ein, und setzen Sie an Stelle der eckigen Klammern das von Ihnen zuvor festgelegte Datenbank-Passwort ein. Wichtig: Die eckigen Klammern bitte in allen folgenden Beispielen auch ersetzen bzw. entfernen!

```
sudo mysql --user=root --password="[IHR DATENBANK-PASSWORT]" mysql
```

Nehmen Sie nun noch einige Anpassungen vor:

```
MariaDB [mysql]> update user set plugin='' where User='root';
MariaDB [mysql]> flush privileges;
```

Nun erstellen wir eine neue Datenbank, den Namen können Sie aus Buchstaben, Zahlen und Unterstrichen beliebig wählen, merken Sie ihn sich jedoch. Folgender Befehl ist dazu notwendig, bestätigt wird mit **[Enter]**:

```
CREATE database [DATENBANK-NAME, z.B. cloud_db];
```

Anschließend erstellen wir einen neuen MySQL-Benutzer und gewähren ihm alle Rechte auf die Datenbank. Dafür müssen Sie sich einen Benutzernamen und im nächsten Schritt ein MySQL-Benutzer-Passwort ausdenken. Ergänzen Sie diese in den Befehlen:

```
GRANT ALL PRIVILEGES ON [DATENBANK-NAME, WIE IM KOMMANDO ZUVOR FESTGELEGT].*
TO '[MYSQL-BENUTZER-NAME]'@'localhost'
```

[Enter] drücken und anschließend folgendes eingeben und mit **[Enter]** bestätigen:

```
1. IDENTIFIED BY '[MYSQL-BENUTZER-Passwort]';
```

Kommt jetzt die Meldung „Query OK“ haben Sie die Datenbank und den Benutzer korrekt eingerichtet. Bitte merken Sie sich alle Daten, da Sie sie später noch einmal benötigen werden. Mit **[Strg]+[C]** schließen Sie das Programm.

Installationsschritte bei/mit Apache2 WEB-Server

```
1. sudo -s                                                    #
   Root-System-Rechte erlangen. Root-Passwort notwendig.
2. apt-get update                                             #
   Systemsoftware updaten.
3. apt-get upgrade                                            #
   Systemsoftware auf den neusten Stand bringen.
4. apt-get install mariadb-server                             #
   Maria-Datenbank installieren.
5. sudo mysql_secure_installation                             #
   [01] Installation und Konfiguration der Maria-DB.
6. sudo mysql -u root                                         #
```

```
    Als root einloggen.
 7.  CREATE DATABASE nextcloud;                                          #
    Neue DB mit den Namen nextcloud anlegen.
 8.  CREATE USER 'nextcloud'@'localhost' IDENTIFIED BY 'password';       #
    Neuen Benutzer "nextcloud" in Maria-DB->nextcloud anlegen.
 9.                                                                      #
    Datenbankbenutzer die Rechte anlegen.
10.  GRANT ALL ON nextcloud.* TO 'nextcloud'@'localhost' IDENTIFIED BY
    'password' WITH GRANT OPTION;
11.  FLUSH PRIVILEGES;                                                  #
    Einstellungen speichern.
12.  EXIT;                                                              #
    Aus Maria-DB ausloggen.
13. sudo apt install apache2 libapache2-mod-php7.2                      #
    Apache2 Web-Server installieren & alle nötigen php Module
14. sudo apt install php7.2-gd php7.2-json php7.2-mysql php7.2-curl php7.2-
    mbstring php7.2-intl php-imagick php7.2-xml php7.2-zip
15. # Hier erfolgt die Installation von open-ssh
```

01

Der Dialog und die Antworten **(fett)** bei Ausführung des Terminal-Befehls.

1. Enter current password for root (enter for none): **Enter**
2. Set root password? [Y/n]: **N**
3. Remove anonymous users? [Y/n]: **Y**
4. Disallow root login remotely? [Y/n]: **Y**
5. Remove test database and access to it? [Y/n]: **Y**
6. Reload privilege tables now? [Y/n]: **Y**

02

xxx

03

xxx

04

xxx

```
[Video]
comment = Videos
path = /mnt/storage/shares/video
write list = heimnetz
valid users = heimnetz,heimgast
force user = heimnetz

[Audio]
comment = Audio
path = /mnt/storage/shares/audio
write list = heimnetz
valid users = heimnetz,heimgast
force user = heimnetz

[Dokumente]
comment = Dokumente
path = /mnt/storage/shares/dokumente
write list = heimnetz
valid users = heimnetz,heimgast
force user = heimnetz

[Public]
comment = Public
path = /mnt/storage/shares/public
writeable = yes
guest ok = yes
force user = heimnetz

[Private]
comment = Private
path = /mnt/storage/shares/private
write list = heimnetz
valid users = heimnetz
force user = heimnetz
```