

# Debian Dienste

- [LAMP Installation \(MySQL, Apache, PHP\)](#)
- [E-Mail bei Linux Updates](#)
- [SSH](#)
  - [Root Login per SSH erlauben](#)
  - [Gotify Benachrichtigung bei SSH-Login](#)
- [FTP](#)
  - [FTP-Server auf einem Debian Server einrichten](#)
- [iSCSI](#)
  - [Initiatornamen auf Debian-System anzeigen lassen](#)
- [Samba](#)
  - [Samba Netzwerkfreigabe erstellen](#)

# LAMP Installation (MySQL, Apache, PHP)

## Einleitung

In dieser Anleitung beschreibe ich kurz wie wir einen **LAMP Webserver** installieren.

Ein **LAMP Webserver** ist eine Installationskombination von **Apache, MySQL / MariaDB** und **PHP**.

---

## Voraussetzungen

Um diese Installation durchführen zu können, sind folgende Voraussetzungen gegeben:

- Zugriff auf die Konsole über SSH / Telnet / Lokal
- Root Rechte / sudo Rechte
- Debian 11
- Internet Anbindung des Servers

---

## Apache Webserver Installation

Um den Apache Webserver zu installieren, aktualisierst du zuerst die Paketquellen und installierst ggf. Updates und entfernst nicht mehr benötigte Pakete.

```
sudo apt-get update && apt-get upgrade -y && apt-get autoremove -y
```

Als Nächstes installierst du den Apache2 Webserver.

```
sudo apt install apache2 -y
```

Du kannst jetzt die Installation mit dem Befehl überprüfen.

Wenn du eine UFW Firewall mit installiert und aktiviert hast, musst du die Ports auf der Firewall freigeben!

```
sudo ufw allow 80/tcp && \  
sudo ufw allow 443/tcp && \  
sudo ufw reload
```

---

# MariaDB Datenbank Installation

Um jetzt die MariaDB Datenbank zu installieren, aktualisierst du zuerst wieder die Paketquellen, installierst Updates und entfernst nicht mehr benötigte Pakete.

```
sudo apt-get update && apt-get upgrade -y && apt-get autoremove -y
```

Als Nächstes installierst du jetzt die MariaDB Datenbank.

```
sudo apt install mariadb-server -y
```

Und jetzt führen wir ein Skript aus. Dieses lässt uns Sicherheitseinstellungen für unseren Datenbankserver einstellen. Dieses öffnen wir mit folgendem Befehl

```
mysql_secure_installation
```

Hier werden einige Dinge abgefragt. Diese Einstellungen werden wir jetzt gemeinsam setzen.

- Zuerst Enter drücken (Wir haben bisher kein root Kennwort gesetzt für den Datenbankbenutzer)
- Debian 11: Die Unix Authentifizierung lehnen wir mit **n** ab.
- Jetzt **Y** eingeben. Wir setzen nun ein Kennwort für den **Root** Datenbank Benutzer.
- Als nächstes **Y** eingeben. Wir wollen alle unbekannten Benutzer löschen.
- Dann geben wir wieder **Y** ein. Wir unterbinden damit eine Anmeldung des **Root Benutzers** außerhalb unseres Servers.
- Und wieder geben wir **Y** ein. Damit wird die Testdatenbank und die Rechte dorthin gelöscht.
- Als Letztes geben wir wieder ein **Y** ein. Damit werden die Berechtigungen einmal neu geladen.

Jetzt kannst du dich mit folgendem Befehl auf dem SQL Server einloggen. Du wirst nach der Eingabe nach dem Kennwort des Root Benutzers gefragt. Sobald du dieses eingegeben hast, kannst du SQL Befehle absetzen.

```
mysql -u root -p
```

Wenn kein Passwort für Root angegeben wurde, loggt sich der Benutzer automatisch auf dem Server ein.

---

## Installation von PHP 7.4

Um unsere Installation abzuschließen, installieren wir jetzt PHP7.4

PHP ist eine serverseitige Programmiersprache. Damit können Befehle direkt auf dem Server ausgeführt werden, z.B. werden Datenbankabfragen häufig über PHP durchgeführt.

```
sudo apt install php7.4 php7.4-cli php7.4-common php7.4-curl php7.4-gd php7.4-intl php7.4-json  
php7.4-mbstring php7.4-mysql php7.4-opcache php7.4-readline php7.4-xml php7.4-xsl php7.4-zip  
php7.4-bz2 libapache2-mod-php7.4 -y
```

Du hast jetzt erfolgreich PHP 7.4 mit Modulen installiert!

---

## Installation von PHP 8.0

Um unsere Installation abzuschließen, installieren wir jetzt PHP7.4

PHP ist eine serverseitige Programmiersprache. Damit können Befehle direkt auf dem Server ausgeführt werden, z.B. werden Datenbankabfragen häufig über PHP durchgeführt.

```
sudo apt install php8.0 php8.0-cli php8.0-common php8.0-curl php8.0-gd php8.0-intl php8.0-  
mbstring php8.0-mysql php8.0-opcache php8.0-readline php8.0-xml php8.0-xsl php8.0-zip php8.0-  
bz2 libapache2-mod-php8.0 -y
```

Du hast jetzt erfolgreich PHP 8.0 mit Modulen installiert!

---

## Optional: Installation von phpMyAdmin

Du kannst optional auch phpMyAdmin installieren, um deine Datenbank über eine Weboberfläche zu verwalten.

Als Erstes wechselst du in das Verzeichnis, in dem du phpMyAdmin ablegen möchtest.

```
cd /var/www
```

Nun lädst du das verpackte Archiv mit den Dateien für phpMyAdmin herunter.

```
wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.zip -O  
phpmyadmin.zip
```

Als Nächstes entpackst du das ZIP Archiv und löschst das alte Verzeichnis.

```
sudo unzip phpmyadmin.zip && sudo rm phpmyadmin.zip
```

Jetzt veränderst du den Namen des Ordners in phpMyAdmin.

```
mv phpMyAdmin-*-all-languages phpmyadmin
```

Jetzt veränderst du die Berechtigungen auf das Verzeichnis.

```
sudo chmod -R 0755 phpmyadmin
```

Und damit wir dann über den Webbrowser auf phpMyAdmin zugreifen können, erstellen wir eine Konfigurationsdatei für den Apache2 Webserver.

```
sudo nano /etc/apache2/conf-available/phpmyadmin.conf
```

Dort fügst du folgende Konfiguration ein.

```
Alias /phpmyadmin /var/www/phpmyadmin

<Directory /var/www/phpmyadmin>
    Options SymLinksIfOwnerMatch
    DirectoryIndex index.php
</Directory>

<Directory /var/www/phpmyadmin/templates>
    Require all denied
</Directory>

<Directory /var/www/phpmyadmin/libraries>
    Require all denied
</Directory>

<Directory /var/www/phpmyadmin/setup/lib>
    Require all denied
</Directory>
```

Sobald du mit der Tastenkombination **STRG + X** und danach **Y** die Datei gespeichert hast, aktivieren wir jetzt die Konfigurationsdatei.

```
sudo a2enconf phpmyadmin && sudo systemctl reload apache2
```

Um die Installation abzuschließen, erstellen wir temporäres Verzeichnis im **phpMyAdmin Verzeichnis** und vergeben für dieses die entsprechende Berechtigung.

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

Du kannst dich jetzt mit den entsprechenden Datenbankbenutzern anmelden. Wenn sich **phpMyAdmin** auf demselben Server wie die Datenbank befindet, musst du keinen Server angeben und du kannst dich mit einem Benutzer anmelden, mit dem Host **localhost**. Du kannst dich also auch als **root** anmelden.

# E-Mail bei Linux Updates

## Einleitung

In Linux kannst du mithilfe von **Apticron** automatisiert E-Mail Benachrichtigungen versenden, wenn Updates verfügbar sind. Du kannst dort auch einstellen, dass du informiert werden möchtest, selbst wenn keine Updates verfügbar sind.

## Einrichtung

Im ersten Schritt verbindest du dich mit deinem Server, damit du Konsolenzugriff hast. Dort aktualisierst du einmal die Paketquellen und installierst mögliche Updates.

```
sudo apt update && sudo apt upgrade -y
```

Im nächsten Schritt installierst du **Apticron** und **sendmail** auf deinem Rechner.

```
sudo apt install apticron sendmail -y
```

Als Nächstes kopierst du die Konfigurationsdatei in das Apticron Verzeichnis, damit du diese modifizieren kannst.

```
sudo cp /usr/lib/apticron/apticron.conf /etc/apticron/apticron.conf
```

Diese Konfigurationsdatei öffnest du in einem Editor deiner Wahl, ich verwende dazu **nano**.

```
nano /etc/apticron/apticron.conf
```

In dieser Datei gibst du Parameter an, wie die E-Mail aussehen soll, welche Absender Adresse er verwenden soll und wer der Empfänger ist.

```
# apticron.conf
#
# The values set in /etc/apticron/apticron.conf will override the settings
# in this file.

#
# Set EMAIL to a space separated list of addresses which will be notified of
# impending updates. By default the root account will be notified.
#
EMAIL="<empfänger>@<domain>"
```

```
#
# Set DIFF_ONLY to "1" to only output the difference of the current run
# compared to the last run (ie. only new upgrades since the last run). If there
# are no differences, no output/email will be generated. By default, apticron
# will output everything that needs to be upgraded.
#
# DIFF_ONLY="1"
#
DIFF_ONLY="0"

#
# Set LISTCHANGES_PROFILE if you would like apticron to invoke apt-listchanges
# with the --profile option. You should add a corresponding profile to
# /etc/apt/listchanges.conf
#
# LISTCHANGES_PROFILE="apticron"
#
# By default apt-listchanges is run with no profile
#
LISTCHANGES_PROFILE=""

#
# From hostname manpage: "Displays all FQDNs of the machine. This option
# enumerates all configured network addresses on all configured network inter-
# faces, and translates them to DNS domain names. Addresses that cannot be
# translated (i.e. because they do not have an appropriate reverse DNS
# entry) are skipped. Note that different addresses may resolve to the same
# name, therefore the output may contain duplicate entries. Do not make any
# assumptions about the order of the output."
#
# By default only the first FQDN is used
#
# ALL_FQDNS="1"
ALL_FQDNS="0"

#
# Set SYSTEM if you would like apticron to use something other than the output
# of "hostname -f" for the system name in the mails it generates. This option
# overrides the ALL_FQDNS above.
```



```
#
# SYSTEM="foobar.example.com"
#
SYSTEM="<FQDN>"

#
# Set IPADDRESSNUM if you would like to configure the maximal number of IP
# addresses apticron displays. The default is to display 1 address of each
# family type (inet, inet6), if available.
#
IPADDRESSNUM="1"

#
# Set IPADDRESSES to a whitespace separated list of reachable addresses for
# this system. If unset or empty, apticron will try to work these out using
# the "ip" command.
#
# IPADDRESSES="192.0.2.1 2001:db8:1:2:3::1"
#
IPADDRESSES=""

#
# Set NOTIFY_HOLDS="0" if you don't want to be notified about new versions of
# packages on hold in your system. The default behavior is downloading and
# listing them as any other package.
#
# NOTIFY_HOLDS="0"
#
NOTIFY_HOLDS="1"

#
# Set NOTIFY_NEW="0" if you don't want to be notified about packages which
# are not installed in your system. Yes, it's possible! There are some issues
# related to systems which have mixed stable/unstable sources. In these cases
# apt-get will consider for example that packages with "Priority:
# required"/"Essential: yes" in unstable but not in stable should be installed,
# so they will be listed in dist-upgrade output. Please take a look at
# http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=531002#44
#
# NOTIFY_NEW="0"
```

```
#
NOTIFY_NEW="1"

#
# Set NOTIFY_NO_UPDATES="1" if you want to be notified when there are no
# new versions. This is useful to assure you that apticron works well.
# By default notifications will be sent only when new versions are available.
#
# NOTIFY_NO_UPDATES="1"
#
NOTIFY_NO_UPDATES="1"

#
# Set CUSTOM_SUBJECT if you want to replace the default subject used in
# the notification e-mails. This may help filtering/sorting client-side e-mail.
# If you want to use internal vars please use single quotes here. Ex:
CUSTOM_SUBJECT='[apticron] $SYSTEM: $NUM_PACKAGES package update(s)'
#
# CUSTOM_SUBJECT=""

#
# Set CUSTOM_NO_UPDATES_SUBJECT if you want to replace the default subject used
# in the no update notification e-mails. This may help filtering/sorting
# client-side e-mail.
# If you want to use internal vars please use single quotes here. Ex:
CUSTOM_NO_UPDATES_SUBJECT='[apticron] $SYSTEM: no updates'
#
# CUSTOM_NO_UPDATES_SUBJECT=""

#
# Set CUSTOM_FROM if you want to replace the default sender by changing the
# 'From:' field used in the notification e-mails.
#
CUSTOM_FROM="<sender>@<domain>"

# Set GPG_ENCRYPT="1" if you want to encrypt the mail being send to
# $EMAIL. apticron will use gpg and the public key of the recipient to encrypt
# the mail. Please note that the $EMAIL value above can't be an alias, since
# gpg will trust it to encrypt the message.
#
```

```
GPG_ENCRYPT=""
```

Apticron verwendet immer die Datei im **/etc/apticron** Verzeichnis, wenn diese existiert. Sonst nutzt er die aus dem **/usr/lib/apticron** Verzeichnis.

Wenn du überprüfen möchtest, ob **Apticron** funktioniert, setze folgenden Befehl ab.

```
sudo apticron
```

SSH

# Root Login per SSH erlauben

## Einleitung

In diesem kleinen Beitrag erläutere ich kurz, wie wir auf unserem *Debian Server* den *SSH-Login* für den Benutzer *root* aktivieren. Dazu müssen wir die Konfiguration anpassen und den Dienst vom SSH-Server neu starten.

**Achtung:** Den Login für den Benutzer *root* zu aktivieren, stellt eine Sicherheitslücke dar! Setze die folgenden Schritte nur um wenn es sich hier um kein kritisches System handelt.

## Root Login erlauben

Um die Konfiguration anzupassen, müssen wir im ersten Schritt die *SSH-Dienst-Konfiguration* öffnen.

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

Und dort ändern wir den Eintrag `PermitRootLogin` auf `yes`. Falls wir den Login wieder deaktivieren wollen, ändern wir den Eintrag einfach wieder auf `no`.

```
PermitRootLogin yes
```

Und zum Schluss starten wir einmal den *SSH-Dienst* neu.

```
sudo systemctl restart ssh
```

Um die Änderung schnell durchzuführen, kann man auch das unten stehende Skript ausführen. Dann werden Schritte automatisch durchgeführt.

```
sudo sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config
sudo service ssh restart
```

# Gotify Benachrichtigung bei SSH-Login

## Einleitung

In diesem kleinen Artikel wird kurz beschrieben, wie wir über einen erfolgreichen SSH Login auf unserem **Debian Server** über **Gotify** informiert werden können.

## Installation

Damit das Skript funktioniert, muss im ersten Schritt sichergestellt werden, dass `curl` installiert ist. Dies können wir einfach mit dem unten stehenden Befehl installieren.

```
sudo apt install curl -y
```

Im Anschluss erstellen wir eine Datei, in dem wir das Skript hinzufügen können.

```
sudo nano /usr/local/bin/ssh-notify-gotify.sh
```

In die Datei fügen wir den folgenden Inhalt ein:

```
exec &> /dev/null

Gotify_URL="https://<gotify-domain>"
Gotify_Token="<Token>"

notify()
{
    now=$date
    title="SSH Login auf $(/bin/hostname -f)"
    message="Erfolgreicher SSH-Login auf $(/bin/hostname -f) mit dem Benutzer $(/usr/bin/who | grep pts)"
    curl -k "https://gotify.m-juergensen.de/message?token=${Gotify_Token}" -F "title=${title}" -F "message=${message}"
}

notify
```

## Anpassungen

Im nächsten Schritt müssen wir kleine Anpassungen vornehmen. Zum Beispiel müssen wir das Skript ausführbar machen, dazu führen wir den folgenden Befehl aus:

```
chmod +x /usr/local/bin/ssh-notify-gotify.sh
```

Jetzt müssen wir im Home-Verzeichnis des SSH-Benutzers in der `.bashrc` Datei den Pfad zum Skript angeben. Dieses wird dann ausgeführt, sobald sich ein Benutzer anmeldet.

FTP



# FTP-Server auf einem Debian Server einrichten

## Einleitung

In diesem kleinen Artikel beschreibe ich kurz, wie wir auf einem Debian Server einen FTP-Server einrichten können. Dazu verwenden wir den **ProFTPD-Dienst** welchen wir einfach herunterladen können.

## Durchführung

### Dienst Installation

Im ersten Schritt müssen wir natürlich den FTP-Dienst installieren. Dazu stellen wir im ersten Schritt eine SSH-Verbindung mit unserem Debian Server her und geben den folgenden Befehl ein:

```
sudo apt install proftpd-basic
```

### Dienst Konfiguration

Im zweiten Schritt müssen wir den Dienst konfigurieren. Wir legen hier die Konfigurationsdatei im **conf.d Verzeichnis** an, damit bei Paketupdates unsere Konfiguration nicht überschrieben werden kann.

```
nano /etc/proftpd/conf.d/config.conf
```

In diese Datei fügen wir den folgenden Inhalt ein:

```
# Ftp Benutzer benötigt keine Shell
<Global>
    RequireValidShell off
</Global>
# Deaktiviere IPv6
UseIPv6 off
# Standard Verzeichnis für ftpuser
DefaultRoot ~ ftpuser
# Erlaube FTP Login nur für eine Gruppe
<Limit LOGIN>
```

```
DenyGroup !ftpuser  
</Limit>
```

Im letzten Schritt müssen wir den FTP-Dienst einmal neu starten.

```
sudo systemctl restart proftpd.service
```

## FTP-Gruppe anlegen

Im nächsten Schritt muss die FTP-Gruppe **ftpuser** angelegt werden, damit wir die Benutzer welche eine FTP Verbindung herstellen sollen, die Berechtigung dazu erhalten. Dazu führen wir den folgenden Befehl aus:

```
sudo groupadd ftpuser
```

## FTP-Benutzer anlegen

Jetzt legen wir einen Benutzer an, welcher dazu verwendet wird, eine FTP-Verbindung herzustellen. Dieser bekommt dann auch ein Home-Verzeichnis zugewiesen, in dem sich der Benutzer automatisch befindet, wenn der Benutzer eine Verbindung herstellt.

Um den Benutzer anzulegen, führen wir den folgenden Befehl durch:

```
sudo adduser <benutzername> --shell /bin/false --home <pfad-zum-home-verzeichnis>
```

# Beispiel

```
sudo adduser transfer --shell /bin/false --home /home/transfer
```

**Info:** Wir werden bei Ausführung des Befehls nach einem Kennwort gefragt, hier wird empfohlen ein möglichst komplexes Kennwort zu verwenden.

Jetzt fügen wir den Benutzer nur noch der entsprechenden Gruppe hinzu:

```
sudo usermod -aG <gruppe> <benutzername>
```

# Beispiel

```
sudo usermod -aG ftpuser transfer
```

Jetzt können wir eine Verbindung mit unserem FTP-Server herstellen.

# iSCSI

# Initiatornamen auf Debian-System anzeigen lassen

Um den **Initiatornamen** auf einem Debian-System anzeigen zu lassen, müssen wir nur den folgenden Befehl ausführen:

```
cat /etc/iscsi/initiatorname.iscsi
```

Dieser zeigt in der letzten Zeile den **Initiatornamen** unseres Servers an. Dieser könnte wie folgt aussehen:

```
## DO NOT EDIT OR REMOVE THIS FILE!  
## If you remove this file, the iSCSI daemon will not start.  
## If you change the InitiatorName, existing access control lists  
## may reject this initiator. The InitiatorName must be unique  
## for each iSCSI initiator. Do NOT duplicate iSCSI InitiatorNames.  
InitiatorName=iqn.1993-08.org.debian:01:13ba30508417
```

# Samba

# Samba Netzwerkfreigabe erstellen

## Einleitung

Du kannst mit Samba einen Server erstellen, auf dem du deine Dokumente in einem Netzwerkfreigabe-Ordner ablegen kannst. Diesen kannst du dann unter Linux, Windows, Mac OS integrieren und so von jedem Gerät Netzwerk weit auf deine Dokumente zugreifen.

**Achtung:** Samba 1.0 zählt als veraltet und sollte daher nur in lokalen abgesicherten Netzwerken installiert werden.

## Installation

Um die Installation durchführen zu können, gibt es folgende Voraussetzungen:

- Debian 10 / 11
- root oder sudo Rechte
- Konsolenzugriff per SSH / Telnet / Lokal
- Internetanbindung des Servers

Zuerst installieren wir das Paket **samba**

```
sudo apt-get install samba
```

Als zweiten Schritt sichern wir die derzeitige Samba-Konfiguration. Dieses Backup dient zum eventuellen Zurückspielen auf den Ursprungszustand.

```
sudo mv /etc/samba/smb.conf /etc/samba/smb.backup
```

Und nun konfigurieren wir den Samba Server. Du erstellst und öffnest die neue Konfigurationsdatei im nächsten Schritt.

```
sudo nano /etc/samba/smb.conf
```

Dort fügst du die Konfiguration ein und speicherst die Datei mit der Tastenkombination **STRG + X** und danach **Y**.

```
[global]
workgroup = smb
security = user
map to guest = Bad Password

[homes]
comment = Home Directories
browsable = no
read only = no
create mode = 0750

[share]
path = /var/share/
public = yes
writable = yes
comment = smb share
printable = no
guest ok = yes
```

In der Konfigurationsdatei kannst du dann noch den Pfad zur Dateiablage verändern oder auch den Namen der Freigabe von **share** auf einen anderen beliebigen setzen.

**Info:** Du verbindest das Netzlaufwerk dann über den UNC Namen mit dem Freigabe Namen dahinter.

**Beispiel:** \\192.168.1.13\share

Um dann Daten abzulegen, muss der Ordner ggf. erst erstellt werden und dann mit Schreibe und Lese Berechtigungen für Public versehen werden.

Die Berechtigungen für die Benutzer werden dann über die Samba-Freigabe gesteuert.

```
sudo mkdir /var/share sudo chmod -R 777 /var/share
```

Und als Letztes starten wir den Samba Service neu. Samba liest dann die neue Konfigurationsdatei ein, und die Freigabe ist dann erreichbar.

```
sudo systemctl restart smbd.service
```

Du kannst den Status des Samba-Service auch überprüfen. Setze dazu den folgenden Befehl ab:

```
sudo systemctl status smbd.service
```