FTP-Server unter Linux installieren und betreiben

Vorwort zur Anleitung:

Diese Anleitung hat einen mittleren bis erhöhten Schwierigkeitsgrad. Installation von Linux und Kenntnisse in der Befehlszeile sind vorausgesetzt.

Was ist ein FTP-Server?

Ein **FTP-Server** (File Transfer Protocol Server) ist ein Server, der das **FTP-Protokoll** verwendet, um Dateien über ein Netzwerk zu übertragen.

Hauptfunktionen eines FTP-Servers:

- **Dateiübertragung:** Ermöglicht das Hochladen und Herunterladen von Dateien zwischen dem Server und einem Client
- **Zugangskontrolle:** Der FTP-Server kann so konfiguriert werden, dass nur autorisierte Benutzer Zugriff haben.
- **Verzeichnisstruktur:** Der Server speichert Dateien in Verzeichnissen und ermöglicht es, diese zu organisieren und zu navigieren.
- **Sicherheit:** Für sicheren Dateitransfer gibt es auch erweiterte Varianten wie **FTPS** (FTP Secure) oder **SFTP** (SSH File Transfer Protocol), die eine verschlüsselte Übertragung bieten.

Wie funktioniert ein FTP-Server?

1. **Verbindung:** Ein FTP-Client (z.B. ein Browser, ein spezielles FTP-Programm oder ein Betriebssystem) stellt eine Verbindung zum FTP-Server her, indem er die IP-Adresse oder den Hostnamen des Servers sowie die erforderlichen Anmeldedaten angibt.
2. **Datenübertragung:** Nach erfolgreicher Authentifizierung kann der Client Dateien auf den Server hochladen oder vom Server herunterladen.
3. **Befehlssatz:** FTP verwendet eine Reihe von Befehlen, um die Interaktion zu steuern, z.B. "GET" für das Herunterladen einer Datei, "PUT" für das Hochladen und "DELETE" für das Löschen von Dateien.

Vielseitige Anwendungsbeispiele:

- **Webhosting**
- **Backup-Lösungen**
- **Datenarchivierung**
- **Drucker und Scan im Netzwerk**
- **...**

Vor- und Nachteile:

Vorteile:
- Einfach und weit verbreitet.
- Ermöglicht die Übertragung großer Dateien.
- Viele FTP-Clients und -Server sind kostenlos und leicht zugänglich.

Nachteile:
- Standard-FTP überträgt Daten unverschlüsselt, was ein Sicherheitsrisiko darstellt.
- Benutzer müssen den richtigen FTP-Client verwenden und den Server korrekt konfigurieren.
- Ein sicherer Betrieb des FTP-Servers bedingt einer Reihe von weiteren Einstellungen, welche tieferes technisches Wissen voraussetzt.

Zusammengefasst: Ein FTP-Server ist eine praktische Lösung für die Dateiübertragung und -verwaltung über Netzwerke.

Vorbereitungen / Anforderungen:

Anforderungen • **Schwierigkeitsgrad-Installation:** Mittel • **Schwierigkeitsgrad-Anwendung:** Einfach • **Erforderliche Kenntnisse:** Linux-Befehlszeile **Systemvoraussetzungen:** • **Basis:** Debian • **Betriebssystem:** Ubuntu Server 22.04 (Headless) • **Festplattenspeicher:** Mindestens 12 GB • **CPU:** Mindestens 1 CPU • **RAM:** Mindestens 512 MB • **Internetverbindung:** Verfügbar	Die Systemanforderungen wurden in einer virtualisierten Umgebung getestet. **Tipp:** Wenn du mit Virtualisierung, z.B. ProxMox, vertraut bist, kannst du deine eigenen Server effizient und stromsparend betreiben! **Getestet auf:** Ubuntu-Server 22.04 Ubuntu-Server 24.10

Installationsablauf: 1. Durchführung von System-Upgrade 2. Statische IP-Adresse festlegen und anwenden 3. (Optional) Hostnamen festlegen 4. Installation und Einstellungen 5. Systembedienung 6. Störungsanalyse 7. Erweiterte Serverentwicklung und Impulse	Optionale Schritte können weggelassen werden, da sie lediglich unterstützende Funktionen bieten, wie zum Beispiel die Zusammenfassung von Servern in einem Rechenzentrum. Für den Betrieb in einem Rechenzentrum sollten die optionalen Abläufe jedoch aus Gründen des beruflichen Stolzes in Betracht gezogen werden.

1. Durchführung von System-Upgrade

Vor jeder Installation ist ein System-Update erforderlich:

$> ssh DEINUSER@192.168.1.X	Stelle eine SSH-Verbindung zum Server her, um Remote-Operationen durchzuführen.
$> sudo apt update	Aktualisiere die Paketquellen, um sicherzustellen, dass du die neuesten Versionen der Pakete erhältst
$> sudo apt upgrade -y	Starte das System-Upgrade und verwende dabei die Option `--yes-to-all`, um alle Bestätigungsabfragen automatisch zu beantworten.
$> sudo apt autoclean	Entferne Pakete, die nicht mehr benötigt werden, um Speicherplatz zu sparen und das System zu optimieren.
$> sudo apt autoremove	Bereinige das System von überflüssigen Abhängigkeiten, die nach Paket-Deinstallationen übrig geblieben sind.

2. Statische IP-Adresse festlegen und anwenden

Ein FTP-Server benötigt eine statische IP-Adresse. Alternativ kann auch eine per DHCP zugewiesene feste Adresse verwendet werden, jedoch sollte diese niemals geändert werden.

Es besteht die Möglichkeit, sich selbst vom Server auszuschließen, wenn die IP-Adresse nicht korrekt konfiguriert ist.

$> ip addr	Verwende den Befehl *ip addr*, um alle Netzwerk-Adapter und deren Namen anzuzeigen.
$> sudo su	Stelle sicher, dass du über Root-Rechte verfügst, um Änderungen vorzunehmen.
$> cd /etc/netplan	Gehe in das Verzeichnis, in dem sich die Netzwerk-Konfigurationsdateien befinden.
$> ls	Zeige mit dem Befehl ls alle relevanten YAML-Dateien im Verzeichnis an.
$> for i in $(ls); do mv $i $i.bak; done	Sichere alle bestehenden Konfigurationsdateien, bevor du Änderungen vornimmst.
$> touch /etc/netplan/01_static_ip.yaml	Erstelle eine neue Konfigurationsdatei
$> nano /etc/netplan/01_static_ip.yaml	Öffne die Datei mit einem Texteditor

network:
  version: 2
  renderer: networkd
  ethernets:
  ens18: #Edit this line according to your network interface name.
  dhcp4: no
  addresses:
  - 192.168.1.150/24
  gateway4: 192.168.1.1
  nameservers:
  addresses:
  - 8.8.8.8
  - 8.8.4.4
	(Copy-Paste)
	
	Ersetze **ens18** durch den Namen deines Netzwerk-Adapters (siehe Ausgabe von *ip addr*).
	
	Ändere die **IP-Adresse** und die **Netzmaske** nach Bedarf.
	
	Passen die **DNS-Adressen** an, falls erforderlich.
	
	Überprüfe und passe ggf. die **Routen (Gateway)** an.
	
	Speichern und Verlassen
	• Speichern der Datei: Ctrl + O
	• Editor verlassen: Ctrl + X

$> netplan generate && netplan apply	Wende die neuen Netzwerkeinstellungen an
--- Verbindungsunterbruch – IP wird neu gesetzt	

	Der Server wechselt nun zu einer neuen IP-Adresse, was dazu führt, dass deine aktuelle SSH-Sitzung unterbrochen wird.

3. (Optional) Hostnamen festlegen

Durch das Ändern des Hostnamens der Server-Station vergibst du einen eindeutigen Namen für das System.

In unserem Fall könnte der Hostname beispielsweise „S3-FTP-Server“ lauten.

Die Hosts-Datei wird verwendet, um auf Server-Ebene IP-Adressen Namen zuzuordnen, die intern genutzt werden. Wenn du den Hostnamen änderst, muss auch der entsprechende Eintrag in der Hosts-Datei aktualisiert werden. Bei produktiven Systemen sollte diese Kleinigkeit unbedingt beachtet werden, da sie ein Zeichen von Berufsethos und Professionalität ist. Für Tests und Experimente ist dies jedoch nicht zwingend erforderlich.

Der neue Hostname muss zwingen der gleiche sein wie auch im Hosts-Files.

$> ssh DEINUSER@192.168.1.X	Stelle eine SSH-Verbindung zum Server über die neue IP-Adresse her.
$> sudo hostnamectl set-hostname DEINHOSTNAME	Setze einen neuen Hostnamen.
$> sudo nano /etc/hosts	Bearbeite die Datei /etc/hosts oder die entsprechende Datei, in der der Hostname definiert ist.
127.0.0.1 localhost 127.0.1.1 DEINHOSTNAME # The following lines are desirable for IPv6 capable hosts ::1 ip6-localhost ip6-loopback fe00::0 ip6-localnet ff00::0 ip6-mcastprefix ff02::1 ip6-allnodes ff02::2 ip6-allrouters	Ändere ausschließlich den markierten Bereich, der den Hostnamen enthält. Achte darauf, dass du keine anderen Einträge unbeabsichtigt veränderst. **Speichern und Verlassen** • Speichern der Datei: Ctrl + O • Editor verlassen: Ctrl + X
$> sudo reboot	Starte den Server neu, damit die Änderungen wirksam werden.

4. Installation und Einstellungen

$> ssh DEINUSER@192.168.1.X	Stelle eine SSH-Verbindung zum Server über die neue IP-Adresse her.
$> sudo apt install proftpd-basic -y	Installiere den FTP-Server-Daemon
$> sudo systemctl enable proftpd.service	Aktiviere den FTP-Dienst, damit er beim Systemstart automatisch gestartet wird.

Die Standard-Einstellungen eines FTP-Servers ermöglichen es, den Server direkt nach der Installation zu nutzen. Für das Login kann jedes existierende SSH-Nutzerkonto verwendet werden. Allerdings gibt es einige wichtige Sicherheitsaspekte, die beachtet werden sollten:

- Passwörter werden unverschlüsselt (im Klartext) über das Netzwerk übertragen und sind somit anfällig für das Abfangen durch Dritte.
- Mit diesem Nutzerkonto hat man grundsätzlich Zugriff auf fast alle Dateien des Servers.
- In diesem unsicheren Zustand sollte der FTP-Server **nicht** öffentlich zugänglich gemacht werden.

Fazit: Wenn die Nutzung des FTP-Servers über die Standard-Einstellungen hinausgeht, sollten unbedingt zusätzliche Sicherheitsmaßnahmen getroffen werden!

```
# Erstellen eines Benutzers für das FTP und setzen eines Passworte
$> sudo useradd --create-home deinftpuser --shell /bin/rbash
$> sudo passwd deinftpuser
```

Erstelle einen Benutzer mit einem eigenen Home-Verzeichnis für die spätere Datenablage. Als Shell wird eine **Restricted Bash** (»rbash«) eingerichtet. Diese Shell ist per SSH zugänglich, erlaubt jedoch nur den Zugriff auf das eigene Home-Verzeichnis.

Die **rbash**-Shell kommt ohne die üblichen Komfortfunktionen, die man von einer regulären Bash kennt. Der Zugriff auf höher privilegierte Benutzer ist nur über den Befehl „**su USER**“ möglich, vorausgesetzt, der Benutzer wurde nicht der sudo-Gruppe zugewiesen. Die **rbash**-Shell ist eine gute Wahl, wenn man dem Benutzer zusätzlich ein SSH-Login ermöglichen möchte.

Wichtig: Wählen Sie ein **starkes, einzigartiges Passwort** für diesen Benutzer, das sich deutlich von üblichen Passwörtern unterscheidet.

Fazit: Diese Konfiguration bietet einen zusätzlichen Sicherheits-Puffer, jedoch bleibt **FTP** auch in dieser Form eine unsichere Lösung für den Transfer sensibler Daten.

```
$> sudo nano /etc/proftpd/proftpd.conf    # Editieren der Einstellungen
# Includes DSO modules
Include /etc/proftpd/modules.conf

# Set off to disable IPv6 support which is annoying on IPv4 only boxes.
UseIPv6 off

# If set on you can experience a longer connection delay in many cases.
<IfModule mod_ident.c>
  IdentLookups off
</IfModule>

ServerName FTP-Server
# Set to inetd only if you would run proftpd by inetd/xinetd/socket.
ServerType standalone
DeferWelcome off

# Disable MultilineRFC2228 per https://github.com/proftpd/proftpd/issues/1085
# MultilineRFC2228on

DefaultServer on
ShowSymlinks on

TimeoutNoTransfer 600
TimeoutStalled 600
TimeoutIdle 1200

DisplayLogin welcome.msg
DisplayChdir .message true
ListOptions -l
DenyFilter \*.*/

# Use this to jail all users in their homes
#DefaultRoot ~
DefaultRoot /home/deinftpuser deinftpuser

# Users require a valid shell listed in /etc/shells to login.
# RequireValidShelloff
```

```
# Port 21 is the standard FTP port.
Port 21

# Prevent DoS attacks, set the maximum number of child processes
MaxInstances 5

# Set the user and group that the server normally runs at.
User proftpd
Group nogroup

# Umask 022 is a good standard umask to prevent new files and dirs
Umask 022 022

# Normally, we want files to be overwriteable.
AllowOverwrite on

# Log-Files
TransferLog /var/log/proftpd/xferlog
SystemLog /var/log/proftpd/proftpd.log

# Logging onto /var/log/lastlog is enabled but set to off by default
#UseLastlog on

# In order to keep log file dates consistent after chroot, use timezone info.
#SetEnv TZ :/etc/localtime

<IfModule mod_quotatab.c>
        QuotaEngine off
</IfModule>
<IfModule mod_ratio.c>
        Ratios off
</IfModule>

# Delay engine reduces impact of the so-called Timing Attack
<IfModule mod_delay.c>
        DelayEngine on
</IfModule>

<IfModule mod_ctrls.c>
        ControlsEngine off
        ControlsMaxClients 2
        ControlsLog /var/log/proftpd/controls.log
        ControlsInterval 5
        ControlsSocket /var/run/proftpd/proftpd.sock
</IfModule>

<IfModule mod_ctrls_admin.c>
        AdminControlsEngine off
</IfModule>

# This is used for FTPS connections
#Include /etc/proftpd/tls.conf

# This is used for SFTP connections
#Include /etc/proftpd/sftp.conf

# This is used for other add-on modules
#Include /etc/proftpd/dnsbl.conf
#Include /etc/proftpd/geoip.conf
#Include /etc/proftpd/snmp.conf
```

# Useful to keep VirtualHost/VirtualRoot directives separated #Include /etc/proftpd/virtuals.conf # Include other custom configuration files Include /etc/proftpd/conf.d/	
$> sudo systemctl restart proftpd.service	# Laden der neuen Einstellungen

Kurze Beschreibung zu den Optionen:

#DefaultRoot ~	Die Standard-Default-Root würde den Zugang zum gesamten Verzeichnis ermöglich. Daher ist diese auszukommetieren
DefaultRoot /home/DEINUSER DEINUSER	Die neue Default-Root «sperrt» den Benutzer DEINUSER in sein Verzeichnis ein. Darüber hinaus kann dieser Benutzer nicht hinausgehen.
Port 21	Setzt den öffentlichen Port auf den gewünschen Port. Im klassischen Fall Port 21.
MaxInstances 5	Sichert den Zugang gegen DDOS Attacken ab. Nach fünf versuchen wird abgebrochen.
AllowOverwrite on	Erlaubt das Überschreiben von Daten

5. Systembedienung

$> sudo systemctl start restart proftpd.service	Startet den FTP-Daemon, um die Zeit-Synchronisation zu aktivieren.
$> sudo systemctl restart proftpd.service	Starte den FTP-Daemon neu, um Änderungen an der Konfiguration wirksam werden zu lassen.
$> sudo systemctl stop proftpd.service	Stoppt den FTP-Daemon, wenn du ihn vorübergehend deaktivieren möchtest.
$> sudo nano /etc/proftpd/proftpd.conf	Bearbeitet die FTP-Konfigurationsdatei
$> cat /var/log/proftpd/proftpd.log $> cat /var/log/proftpd/xferlog	Zeige die Logdateien des FTP-Daemons an.
$> sudo apt install nmap -y $> nmap localhost	Listet alle offenen Ports auf, um sicherzustellen, dass der FTP-Daemon ordnungsgemäß kommunizieren kann.
$> man proftpd $> man proftpd.conf	Rufe das Handbuch oder die Hilfe für den FTP-Daemon auf, um detaillierte Informationen und Befehlsoptionen zu erhalten.
$> sudo proftpd -t	Überprüfe die Konfiguration, um sicherzustellen, dass alle Einstellungen korrekt sind.
$> ftp ftp> open (to) 192.168.x.x Connected to 192.168.x.x 220 ProFTPD Server (FTP-Server) [192.168.x.x] Name (192.168.x.x:user): deinftpuser	Teste den FTP-Zugang direkt über die Kommandozeile, um sicherzustellen, dass der Server ordnungsgemäß funktioniert.

331 Password required for deinftpuser Password: ********** 230 User deinftpuser logged in Remote system type is UNIX. Using binary mode to transfer files. ftp> ls 229 Entering Extended Passive Mode (|||11820|) 150 Opening ASCII mode data connection for file list 226 Transfer complete ftp> bye 221 Goodbye.	

6. Störungsanalyse

Statische IP-Adresse wird nicht gesetzt:

Beachte, dass die Netzwerk-Konfiguration im YAML-Stil erfolgt. Die „Incidents“ (Einschübe des Textes) sind essenziell.

Einige CLI-Befehle benötigen ungewöhnlich lange…

Wenn du den optionalen Teil dieser Anleitung befolgt hast, überprüfe, ob der Hostname des Servers mit der Hosts-Datei übereinstimmt!

Der FTP-Server funktioniert generell nicht...

Die **Basis-Installation** von ProFTP ist relativ einfach, doch die eigentliche Herausforderung besteht in der Konfiguration des FTP-Servers. Die meisten Fehler resultieren aus fehlerhaften Einstellungen, was besonders deutlich wird, wenn man sich eingehender mit der Konfiguration beschäftigt. Daher sollte der **Zweck des Servers** von vornherein klar definiert sein. Die folgende Anleitung bezieht sich auf den Einsatz des Servers zum Empfangen gescannter Dokumente von einem Drucker. Die empfohlenen Sicherheitsmaßnahmen sind in einer **lokalen Umgebung** ausreichend.

Viele Probleme entstehen durch die Vielfalt der **Client-Systeme**, die verwendet werden – eine Vielzahl an Systemen, die hier nicht alle aufgezählt werden können. Häufig liegen die Ursachen für Probleme in **Credentials**, **Firewalls** oder **Proxy-Servern**, die den Zugang blockieren. Tatsächlich handelt es sich bei diesen Problemen weniger um "Fehler" im engeren Sinne, sondern um Sicherheitsvorkehrungen, die speziell den Zugriff auf **Port 21** betreffen. Es sei nochmals betont: **FTP gilt als unsicher**. Viele Geräte, insbesondere **Drucker**, verwenden auch heute noch FTP anstelle des sichereren **SFTP-Protokolls**. Der Grund hierfür liegt oft in der Hardware der Geräte, die SFTP nur selten unterstützen.

Das eigentliche **Troubleshooting** wird daher besser in spezialisierten, weiterführenden Anleitungen behandelt.

7. Erweiterte Serverentwicklung und Impulse

FTP-Ordner neigen dazu, schnell vollzulaufen. Daher sollten die Home-Verzeichnisse regelmäßig bereinigt werden. Am einfachsten lässt sich dies mit einem **CronJob** erreichen, der als **Root**-Benutzer ausgeführt wird.

In diesem Fall sind die FTP-Benutzer echte **SSH-Konten**, können aber auch durch virtuelle Konten ersetzt werden. Bei Verwendung virtueller Konten gehen jedoch einige der komfortablen Funktionen verloren, die Linux auf Basis von Kommandozeilenbefehlen bietet.

Für einen stärkeren Sicherheitsansatz sollte das **SSL-Protokoll** verwendet werden. Allerdings erhöht dies die Komplexität der Konfiguration. Zudem verliert der FTP-Server dadurch teilweise seinen ursprünglichen Zweck, da es sicherere Alternativen für den Dateitransfer gibt.

Fazit: In den mehr als 40 Jahren der Computergeschichte ist es nicht gelungen, **FTP** wirklich sicher zu machen, da es ursprünglich nicht für diese Anforderungen entwickelt wurde. Wer auf mehr Sicherheit angewiesen ist, sollte auf andere, sicherere Protokolle ausweichen oder FTP nur lokal verwenden. Wenn FTP unverzichtbar ist, empfiehlt es sich, zusätzliche Sicherheitsmaßnahmen zu kombinieren.