

Dokumentation Deployment einer Rest API

Autor: Gerrit Koppe

Ausbildungsberuf: Fachinformatiker für Anwendungsentwicklung

Klasse: IFA12

Lernfeld 9: Netzwerke und Dienste bereitstellen

18. Juni 2023

Anmerkungen

Aus Gründen der Leserlichkeit wird in dieser Dokumentation das Wort „Server“ verwendet, wann immer vom Raspberry Pi die Rede ist.

Alle ausgeführten Befehle werden im laufenden Text der Dokumentation angegeben. Während des gesamten Prozesses, welcher in dieser Dokumentation beschrieben wird, wurden des Weiteren Screenshots gemacht, welche in den Anlagen in Kapitel 9.1 beigelegt sind. Auf diese wird an den entsprechenden Stellen der Dokumentation verwiesen.

Wann immer in einer Terminal Abbildung „#. . .“ steht, zeigt dies an, dass die Datei, welche jeweils bearbeitet wurde, weitere Zeilen hat, welche aber nicht relevant für den jeweils aktuellen Arbeitsschritt sind und dementsprechend ausgelassen wurden.

Inhaltsverzeichnis

1	Einleitung	1
2	Vorbereitungen	1
3	Konfiguration der User	1
3.1	Anlage neuer Benutzer	1
3.2	Konfiguration des administrativen Benutzers	2
4	Konfiguration des Netzwerks	2
5	Einrichten der Firewall	3
5.1	Installation	3
5.2	Konfiguration	4
6	Einrichtung Webserver	6
6.1	Installation	7
6.2	Konfiguration Webserver	7
7	Einrichtung Rest API auf Betriebssystem	7
8	Veröffentlichung der Swagger Dokumentation auf dem Webserver	8
9	Anlagen	9
9.1	Bilder	9
9.1.1	Konfiguration User	9
9.1.2	Konfiguration Netzwerk	10
9.1.3	Konfiguration Firewall	11
9.1.4	Konfiguration Webserver	12
9.2	Quellen	12
9.2.1	Internetquellen	12

1 Einleitung

In dieser Dokumentation wird die Konfiguration eines Raspberry Pi¹ als Host einer Rest API, sowie das Deployment besagter Rest API beschrieben.

Es wird zunächst auf allgemeine Vorbereitungen eingegangen. Anschließend wird beschrieben, wie die User und das Netzwerk, sowie die Firewall des Servers konfiguriert werden müssen. Abschließend wird die Installation des Webserver, sowie das Deployment der Rest API auf dem Server beschrieben.

Sämtliche Arbeitsschritte sind sowohl durch die jeweiligen Terminal Befehle, als auch durch Screenshots die während der Umsetzung entstanden sind, dokumentiert.

2 Vorbereitungen

3 Konfiguration der User

Nachdem das Betriebssystem des Servers installiert, der Server in das Netzwerk eingebunden und überprüft wurde, ob eine SSH Verbindung zum Server möglich ist, wurden zwei neue User angelegt, um den Server abzusichern, da der User „Pi“ der Standarduser des Betriebssystems ist und somit allgemein bekannt.

Es wurden insgesamt zwei neue User angelegt. Ein User „benutzer72“ mit grundlegenden Nutzungsrechten und ein Benutzer „fernzugriff“ mit administrativen Rechten. Des Weiteren kann der Benutzer „fernzugriff“ verwendet werden, um eine SSH-Verbindung zum Server aufzubauen.

3.1 Anlage neuer Benutzer

Zunächst wurde der user „benutzer72“ mit folgenden Befehlen angelegt:

```
pi@raspberrypi:~ $ sudo useradd -m benutzer72
pi@raspberrypi:~ $ sudo passwd benutzer72
New password:
Retype new password:
passwd: password updated successfully
```

Der Befehl `useradd` dient dazu, den neuen Benutzer anzulegen. Mit dem flag `-m` wird außerdem automatisch ein Home-Verzeichnis für den neuen Benutzer erzeugt. Des Weiteren wurde dem neuen Benutzer mittels des `passwd` Befehls ein neues Passwort zugewiesen².

Nachdem der Benutzer `benutzer72` konfiguriert wurde, wurde ein neuer administrativer Nutzer „fernzugriff“ angelegt. Die Vorgehensweise war hier zunächst identisch zu der der Neuanlage von `benutzer72`³:

¹Fortan „Server“

²Vgl. Abbildung 1 in Kapitel 9.1.1

³Vgl. Abbildung 2 in Kapitel 9.1.1

```
pi@raspberrypi:~$ sudo useradd -m fernzugriff
pi@raspberrypi:~$ sudo passwd fernzugriff
New password:
Retype new password:
passwd: password updated successfully
```

3.2 Konfiguration des administrativen Benutzers

Da der Benutzer „fernzugriff“ administrative Rechte auf dem Server erhalten sollte, wurde er anschließend in die Gruppe „sudo“ aufgenommen⁴:

```
pi@raspberrypi:~$ sudo usermod -aG sudo fernzugriff
```

Hier dient der Befehl `usermod` allgemein dazu, einen User zu modifizieren. Der Flag `-aG` gibt an, dass der User einer Gruppe hinzugefügt werden soll, welche wiederum direkt hinter dem Flag definiert ist (in diesem Fall „sudo“).

Abschließend wurde dem Benutzer „fernzugriff“ noch das Recht gewährt, sich per SSH mit dem Server zu verbinden. Dafür wurde die Einstellung `AllowUsers` in der Datei `/etc/ssh/sshd_config` angepasst⁵:

```
pi@raspberrypi:~$ sudo nano /etc/ssh/sshd_config
```

```
$OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $

#Port 22
#AddressFamily any
AllowUsers    fernzugriff
#...
```

Nun besitzt der Benutzer „fernzugriff“ alle notwendigen Rechte, um ihn für administrative Tätigkeiten zu verwenden. Fortan wird der Benutzer „pi“ nicht mehr verwendet und alle Umsetzungen werden mit dem Benutzer „fernzugriff“ durchgeführt.

4 Konfiguration des Netzwerks

Bislang nutzte der Server die IP-Adresse, welche ihm vom DHCP-Server des Netzwerks zugewiesen wurde. Fortan sollen aber statische Einstellungen für IP-Adresse, DNS-Server und Router verwendet

⁴Vgl. Abbildung 3 in Kapitel 9.1.1.

⁵Vgl. Abbildung 4 in Kapitel 9.1.1

werden.

Um dies zu konfigurieren, musste die Datei `/etc/dhcpd.conf` angepasst werden⁶:

```
fernzugriff@raspberrypi:~ $ sudo nano /etc/dhcpd.conf
```

```
# Example static IP configuration
#interface eth0
static ip_address=192.168.24.113/24
#static ip6_address=fd51:42f8:caae:d92e::ff/64
static routers=192.168.24.254
static domain_name_servers=192.168.24.254
#...
```

Um diese Änderungen in Kraft zu setzen, gab es zwei Möglichkeiten. Entweder musste der `dhcpd` Dienst über folgenden Befehl neugestartet werden:

```
fernzugriff@raspberrypi:~ $ sudo systemctl restart dhcpd.service
```

Da durch den Neustart des Dienstes aber gleichzeitig auch die IP-Adresse geändert würde und somit die SSH Verbindung abgebrochen wäre, wurde sich dafür entschieden, den Server komplett neuzustarten:

```
fernzugriff@raspberrypi:~ $ sudo reboot
```

Somit ist der Server vollständig in das Netzwerk eingebunden.

5 Einrichten der Firewall

5.1 Installation

Im Standard ist auf Linux-basierten Systemen die Firewall `IPTables` installiert. Diese wurde zunächst um das Tool „UFW“⁷ erweitert, welches es erlaubt, `IPTables` über simple Befehle zu konfigurieren. Um UFW zu installieren, wurde zunächst der `apt` Package Index aktualisiert, um die neueste Version von UFW im Zugriff zu haben⁸:

⁶Vgl. Abbildung 5 in Kapitel 9.1.2

⁷Vgl. Quelle 1 in Kapitel 9.2.1

⁸Vgl. Abbildung 6 in Kapitel 9.1.3

```
fernzugriff@raspberrypi:~ $ sudo apt update
```

Anschließend wurde UFW installiert⁹:

```
fernzugriff@raspberrypi:~ $ sudo apt install ufw
```

5.2 Konfiguration

Nachdem UFW erfolgreich installiert wurde, wurden die notwendigen Firewall Regeln für den Server eingerichtet.

Zunächst musste sichergestellt werden, dass SSH Verbindungen zum Server auch nach Aktivierung der Firewall weiterhin möglich sein würden, allerdings nur aus dem Netzwerk, in dem der Server sich befindet. Da dem Server zuvor, wie in Kapitel 4 beschrieben, eine statische IP-Adresse und somit statisch ein Netzwerk zugewiesen wurden, muss hierbei nicht darauf geachtet werden, dass das Netzwerk sich in der Zukunft ändern könnte.

Um SSH weiterhin zuzulassen, musste der Port 22 für alle eingehenden Pakete aus dem Netzwerk 192.168.24.0/24 freigeschaltet werden¹⁰:

```
fernzugriff@raspberrypi:~ $ sudo ufw allow from 192.168.24.0/24  
proto tcp to any port 22  
  
Rules updated
```

Da im Standard bereits Regeln für den Port 22 definiert sind, mussten diese gelöscht werden. Der Grund dafür, dass zuerst eine neue Regel eingerichtet wurde, ist, dass ansonsten die Gefahr bestanden hätte, das System nicht mehr per SSH zu erreichen.

Um die alten SSH Regeln zu löschen, wurden ihre IDs benötigt. Um diese sehen zu können, musste UFW zunächst aktiviert werden, um anschließend alle Regeln anzeigen lassen zu können:

⁹Vgl. Abbildung 7 in Kapitel 9.1.3

¹⁰Vgl. Abbildung 8 in Kapitel 9.1.3

```
fernzugriff@raspberrypi:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
fernzugriff@raspberrypi:~$ sudo ufw status numbered
Status: active
```

	To	Action	From
	--	-----	----
[1]	22/tcp	ALLOW IN	192.168.24.0/24
[2]	22	ALLOW IN	Anywhere
[3]	22 (v6)	ALLOW IN	Anywhere (v6)

Die Regeln mit den IDs 2 und 3 mussten nun gelöscht werden, damit die selbst erstellte Regel für SSH Verbindungen korrekt arbeitet:

```
fernzugriff@raspberrypi:~$ sudo ufw delete 3
Deleting:
    allow 22 (v6)
Proceed with operation (y|n)? y
Rule deleted
fernzugriff@raspberrypi:~$ sudo ufw delete 2
Deleting:
    allow 22
Proceed with operation (y|n)? y
Rule deleted
```

Des Weiteren sollten ausgehende DNS Anfragen erlaubt werden. Das DNS Protokoll arbeitet über Port 53, entsprechend musste dieser für ausgehende Pakete freigegeben werden¹¹:

```
fernzugriff@raspberrypi:~$ sudo ufw allow out to any port 53
Rule added
Rule added (v6)
```

Der Port, über den die Rest API, nachdem sie deployed sein würde, erreichbar sein soll, wurde ebenfalls freigegeben¹²:

¹¹Vgl. Abbildung 10 in Kapitel 9.1.3

¹²Vgl. Abbildung 9 in Kapitel 9.1.3


```
fernzugriff@raspberrypi:~ $ sudo ufw allow 13376
Rule added
Rule added (v6)
```

Außerdem soll der Webserver aus allen Netzwerken erreichbar sein. Da wir die HTTP Kommunikation nicht mit SSL verschlüsseln werden, reicht es hier, den Port 80 für alle Netzwerke freizugeben:

```
fernzugriff@raspberrypi:~ $ sudo ufw allow 80
Rule added
Rule added (v6)
```

Alle Pakete, die keiner dieser Regeln entsprechen, sollten abgelehnt werden. Um dies zu erreichen, wurde die Standard Regel für eingehende Pakete angepasst¹³:

```
fernzugriff@raspberrypi:~ $ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
```

Nachdem alle Regeln eingerichtet wurden, wurde noch einmal über den UFW Status überprüft, ob alle Regeln existierten:

```
fernzugriff@raspberrypi:~ $ sudo ufw status numbered
Status: active
```

	To	Action	From	
	--	-----	----	
[1]	13376	ALLOW IN	Anywhere	
[2]	22/tcp	ALLOW IN	192.168.24.0/24	
[3]	53	ALLOW OUT	Anywhere	(out)
[4]	80	ALLOW OUT	Anywhere	
[5]	13376 (v6)	ALLOW IN	Anywhere (v6)	
[6]	53 (v6)	ALLOW OUT	Anywhere (v6)	(out)
[7]	80 (v6)	ALLOW OUT	Anywhere (v6)	

6 Einrichtung Webserver

Nachdem die Firewall Regeln eingerichtet wurden, konnte der Webserver installiert und konfiguriert werden. Die Wahl fiel hier auf den Webserver Apache2.

¹³Vgl. Abbildung 11 in Kapitel 9.1.3

6.1 Installation

Da der Package Index bereits aktualisiert wurde, konnte der Webserver direkt installiert werden:

```
fernzugriff@raspberrypi:~ $ sudo apt install apache2
```

6.2 Konfiguration Webserver

Da der Apache2 Server im Standard bereits die Ports für HTTP abhört, mussten diese nicht mehr konfiguriert werden.

Nach der Installation wurde zunächst eine einfache Startseite für den Webserver eingerichtet. Dafür musste die Datei `/var/www/html/index.html`, welche bei der Installation des Apache2 angelegt wurde, angepasst werden:

```
fernzugriff@raspberrypi:~ $ cd /var/www/html
fernzugriff@raspberrypi:/var/www/html $ sudo nano index.html
```

```
<html>
  <body>
    <h1>Todo-Listen Verwaltung</h1>
    <h3>Gerrit Koppe</h3>
  </body>
</html>
```

Diese konnte nun erreicht werden, sobald die IP-Adresse des Servers im Browser aufgerufen wurde¹⁴.

7 Einrichtung Rest API auf Betriebssystem

Um die Rest API auf dem Betriebssystem einzurichten, mussten zunächst die Skripte auf dem Server abgelegt werden. Zunächst wurde ein neuer Ordner `api` im Verzeichnis `/var` angelegt:

```
fernzugriff@raspberrypi:~ $ cd /var
fernzugriff@raspberrypi:/var $ sudo mkdir todo-api
```

Da die Rest API sich in einem Git Repository befindet, konnte dieses in den neu erstellten Ordner geklont werden:

¹⁴Vgl. Abbildung 12 in Kapitel 9.1.4

```
fernzugriff@raspberrypi:/var $ cd todo-api
fernzugriff@raspberrypi:/var/todo-api $ sudo git clone
"https://github.com/LeichtMatrosee/ToDoRestServer.git"
```

Nachdem die Rest API auf den Server geklont wurde, mussten noch alle Python Libraries nachinstalliert werden, welche noch nicht auf dem Server existierten (in diesem Fall nur `flask_cors`, dies kann sich bei anderen Installationen unterscheiden).

Anschließend wurde überprüft, ob die API lauffähig war:

8 Veröffentlichung der Swagger Dokumentation auf dem Webserver

Damit die veröffentlichte Rest API bedient werden kann, sollte die Swagger Dokumentation auf dem Webserver einzusehen sein.

Um dies zu erreichen, wurde zunächst die Datei `/var/www/html/api/index.html` angelegt, welche als Homepage für die Dokumentation dienen sollte:

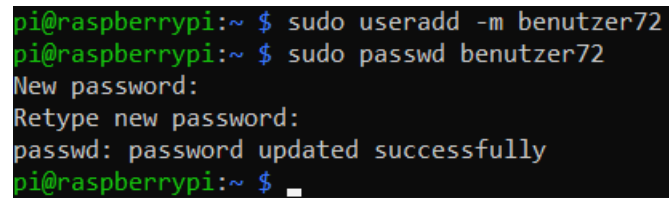
```
fernzugriff@raspberrypi:~ $ cd /var/www/html/api
fernzugriff@raspberrypi:/var/www/html/api $ sudo nano index.html
```

```
<html>
  <body>
    <h1>Projekt Todo-Listen Verwaltung</h1>
    <br />
    <h2>Client Dokumentation</h2><br />
    <a href='./documentation/Client/API-Dokumentation/index.html'>
      API-Dokumentation des Rest Clients
    </a>
    <br />
    <h2>Client Dokumentation</h2><br />
    <a href='https://github.com/LeichtMatrosee/RestClient'>
      Repo des Rest Clients
    </a>
  </body>
</html>
```

9 Anlagen

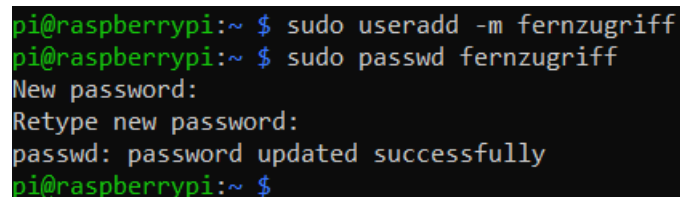
9.1 Bilder

9.1.1 Konfiguration User



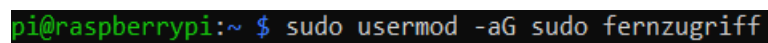
```
pi@raspberrypi:~ $ sudo useradd -m benutzer72
pi@raspberrypi:~ $ sudo passwd benutzer72
New password:
Retype new password:
passwd: password updated successfully
pi@raspberrypi:~ $
```

Abbildung 1: Anlage und Konfiguration des Benutzers „benutzer72“



```
pi@raspberrypi:~ $ sudo useradd -m fernzugriff
pi@raspberrypi:~ $ sudo passwd fernzugriff
New password:
Retype new password:
passwd: password updated successfully
pi@raspberrypi:~ $
```

Abbildung 2: Anlage und Konfiguration des Benutzers „fernzugriff“



```
pi@raspberrypi:~ $ sudo usermod -aG sudo fernzugriff
```

Abbildung 3: Benutzer „fernzugriff“ in die Gruppe sudo aufnehmen

```
# $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
AllowUsers      fernzugriff

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none
```

Abbildung 4: Benutzer „fernzugriff“ SSH-Rechte gewähren

9.1.2 Konfiguration Netzwerk

```
# Example static IP configuration:
#interface eth0
static ip_address=192.168.24.113/24
#static ip6_address=fd51:42f8:caae:d92e::ff/64
static routers=192.168.24.254
static domain_name_servers=192.168.24.254
```

Abbildung 5: Angepasste dhcpd.conf Datei

9.1.3 Konfiguration Firewall

```
pi@raspberrypi:~$ sudo apt update
Get:1 http://raspbian.raspberrypi.org/raspbian bullseye InRelease [15.0 kB]
Get:2 http://archive.raspberrypi.org/debian bullseye InRelease [23.6 kB]
Get:3 http://raspbian.raspberrypi.org/raspbian bullseye/main armhf Packages [13.2 MB]
Get:4 http://archive.raspberrypi.org/debian bullseye/main armhf Packages [316 kB]
Fetched 13.6 MB in 15s (887 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
31 packages can be upgraded. Run 'apt list --upgradable' to see them.
pi@raspberrypi:~$
```

Abbildung 6: Update APT

```
fernzugriff@raspberrypi:~$ sudo apt install ufw
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done: 100%
The following package was automatically installed and is no longer required:
  libfuse2
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  ufw
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 167 kB of archives.
After this operation, 857 kB of additional disk space will be used.
Get:1 http://mirror.netzwerke.de/raspbian/raspbian bullseye/main armhf ufw all 0.36-7.1 [167 kB]
Fetched 167 kB in 0s (497 kB/s)
Preconfiguring packages ...
Selecting previously unselected package ufw.
(Reading database ... 109573 files and directories currently installed.)
Preparing to unpack .../archives/ufw_0.36-7.1_all.deb ...
Unpacking ufw (0.36-7.1) ...
Setting up ufw (0.36-7.1) ...

Creating config file /etc/ufw/before.rules with new version

Creating config file /etc/ufw/before6.rules with new version

Creating config file /etc/ufw/after.rules with new version

Creating config file /etc/ufw/after6.rules with new version
Created symlink /etc/systemd/system/multi-user.target.wants/ufw.service → /lib/systemd/system/ufw.service.
Processing triggers for rsyslog (8.2102.0-2+deb11u1) ...
Processing triggers for man-db (2.9.4-2) ...
```

Abbildung 7: Installation der UFW Firewall

```
fernzugriff@raspberrypi:/etc/ufw$ sudo ufw allow from 192.168.24.0/24 proto tcp to any port 22
Rules updated
```

Abbildung 8: SSH Port für gleiches Netzwerk öffnen

```
fernzugriff@raspberrypi:/etc/ufw$ sudo ufw allow 13376
Rules updated
Rules updated (v6)
```

Abbildung 9: Port der API für alle Netzwerke freischalten

```
fernzugriff@raspberrypi:/etc/ufw $ sudo ufw allow out to any port 53
Rule added
Rule added (v6)
```

Abbildung 10: Ausgehende DNS Anfragen erlauben

```
fernzugriff@raspberrypi:/etc/ufw $ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
```

Abbildung 11: Alle eingehenden Pakete ohne Regel verbieten

9.1.4 Konfiguration Webserver

Todo-Listen Verwaltung

Gerrit Koppe

Abbildung 12: Alle eingehenden Pakete ohne Regel verbieten

9.2 Quellen

9.2.1 Internetquellen

1. Ubuntu Wiki: UncomplicatedFirewall. 2022 - Online unter:
<https://wiki.ubuntu.com/UncomplicatedFirewall> [11.06.2023]