Sinusbot auf einem Raspberry Pi installieren

Description

In dieser Anleitung geht es darum, wie wir den Sinusbot auf einem Raspberry Pi installieren und zum laufen bekommen. Aktuell unterstÃ¼tzt der Sinusbot nur eine AMD64 Architektur und keine MÃ¶glichkeit diesen unter ARM laufen zu lassen.

Es gibt einen Inoffiziellen Weg den ich hier beschreiben werde. Ich Ã¼bernehme aber keine Haftung fÃ¼r etwaige SchÃ¤den oder UnzuverlÃ¤ssigkeiten!

Sinusbot installieren

Im ersten Schritt mÃ¼ssen wir Ã¼berprÃ¼fen welche Pagesize unser Raspberry PI hat. Die Pagesize muss 4k betragen. Welche Pagesize aktuell eingestellt ist, kÃ¶nnen wir mit dem folgenden Befehl Ã¼berprÃ¼fen:

```
getconf PAGESIZE
```

Wir erhalten in der Regel auf einem Raspberry Pi dann eine Ausgabe mit **16384**. Da dies leider zu hoch ist, mÃ¼ssen wir dies Ã¼ber die Boot-Config auf unserem Raspberry Pi Ã¤ndern. Dazu Ã¶ffnen wir die folgende Datei und fÃ¼gen an unterster Stelle unter **[all]** den folgenden Inhalt ein:

```
sudo nano /boot/firmware/config.txt
```

```
kernel=kernel8.img
```

Im Anschluss mÃ¼ssen wir den Raspberry Pi einmal neustarten.

```
systemctl reboot
```

Sobald der Raspberry Pi wieder hochgefahren ist, mÃ¼ssen wir einmal Ã¼berprÃ¼fen ob die Pagesize jetzt **4096** ist.

```
getconf PAGESIZE
```

Docker installieren

Als nÃ¤chstes mÃ¼ssen wir jetzt auf unserem Raspberry Pi **Docker** und **Docker-Compose** installieren. Dazu fÃ¼hren wir die unten stehenden Befehle aus.

```
# Add Docker's official GPG key:
sudo apt update
sudo apt install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
sudo tee /etc/apt/sources.list.d/docker.sources <<EOF
Types: deb
URIs: https://download.docker.com/linux/debian
Suites: $(. /etc/os-release && echo "$VERSION_CODENAME")
Components: stable
Signed-By: /etc/apt/keyrings/docker.asc
EOF

sudo apt update
```

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

Im Anschluss kÃ¶nnen wir einmal Ã¼berprÃ¼fen ob die Installation erfolgreich geklappt hat, in dem wir die beiden Befehle ausfÃ¼hren. Wir sollten dann die Versionsnummern unserer Docker-Installationen sehen.

```
docker version
docker-compose version
```

QEMU/binfmt fÃ¼r amd64 aktivieren

Im nÃ¤chsten Schritt aktivieren wir die AMD64 Emulation auf unserem Raspberry Pi. Dazu fÃ¼hren wir den folgenden Befehl aus um den Emulator zu aktivieren.

```
docker run --privileged --rm tonistiigi/binfmt --install amd64
```

Wenn der Befehl durchgelaufen ist, sollten wir jetzt AMD64 Container ausfÃ¼hren kÃ¶nnen. Um dies einmal zu testen kÃ¶nnen wir einmal den folgenden Container einmal starten.

```
docker run --rm --platform linux/amd64 alpine uname -m
```

Benutzer & Verzeichnis vorbereiten

Um den Sinusbot jetzt laufen zu lassen, mÃ¼ssen wir dafÃ¼r einen dedizierten Benutzer fÃ¼r den Dienst erstellen. Dazu fÃ¼hren wir den folgenden Befehl aus:

```
sudo useradd sinusbot
```

Im Anschluss erstellen wir ein Verzeichnis indem unser Sinusbot spÃ¤ter seine eigenen Dateien ablegen kann. Ich erstelle diese gerne immer unter **/opt**.

```
mkdir -p /opt/sinusbot
```

Container vorbereiten und starten

Im ersten Schritt legen wir die Verzeichnisse an, in denen Sinusbot seine Daten ablegt. Dazu kÃ¶nnen wir den folgenden Befehl ausfÃ¼hren:

```
mkdir -p /opt/sinusbot/data
mkdir -p /opt/sinusbot/scripts
mkdir -p /opt/sinusbot/init
```

Dann legen wir jetzt die **Docker-Compose.yml** Datei an die mit dem folgenden Inhalt gefÃ¼llt werden muss:

```
sudo nano /opt/sinusbot/docker-compose.yml
```

```
services:
  sinusbot:
    image: sinusbot/docker:latest
    container_name: sinusbot
    platform: linux/amd64
    restart: unless-stopped
    ports:
      - "8087:8087"
    environment:
      TZ: Europe/Berlin
      UID: "1001"
      GID: "1001"
    volumes:
      - ./scripts:/opt/sinusbot/scripts
      - ./data:/opt/sinusbot/data
      - ./init:/opt/init:ro
    entrypoint: ["/bin/bash","-lc",
```

```
"/opt/init/bootstrap.sh; exec /opt/sinusbot/entrypoint.sh"]
```

> **Wichtig:** Wir mÃ¼ssen hier die UID und GID auf die IDâ??s des Sinusbot Benutzers einstellen. Dazu kÃ¶nnen wir den Befehl **`id sinusbot`** eingeben. Diese angezeigten IDâ??s tragen wir dann unter UID und GID ein.

Jetzt legen wir das Start-Skript unseres Sinusbots an und fÃ¼llen dieses auch mit Inhalt:

```
sudo nano /opt/sinusbot/init/bootstrap.sh
```

```
#!/usr/bin/env bash
set -euo pipefail

STAMP="/opt/sinusbot/data/.bootstrap_done"
if [[ -f "$STAMP" ]]; then
  exit 0
fi

# Debian buster -> archive + valid-until off
sed -i 's|http://deb.debian.org/debian|http://archive.debian.org/debian|g'
/etc/apt/sources.list || true
sed -i
's|http://deb.debian.org/debian-security|http://archive.debian.org/debian-security|g'
/etc/apt/sources.list || true
sed -i 's|security.debian.org|archive.debian.org|g' /etc/apt/sources.list || true
echo 'Acquire::Check-Valid-Until "false";' > /etc/apt/apt.conf.d/99no-check-valid-until

# Achtung: kann bei dir wieder bei libc-bin segfaulten
apt-get -o Acquire::Check-Valid-Until=false update
DEBIAN_FRONTEND=noninteractive apt-get install -y --no-install-recommends libpulse0
youtube-dl ca-certificates || true

touch "$STAMP"
```

Zum Schluss setzen wir noch die Berechtigungen fÃ¼r die Verzeichnisse und das Skript.

```
chown -R sinusbot:sinusbot /opt/sinusbot/
chmod +x /opt/sinusbot/init/bootstrap.sh
```

Jetzt kÃ¶nnen wir den Container starten und warten das unsere Sinusbot-Instanz erfolgreich hochfÃ¤hrt.

```
docker compose up -d -f /opt/sinusbot/docker-compose.yml
```

Wenn der Container erfolgreich hochgefahren ist, kÃ¶nnen wir mit der IP und dem Port 8087 das Webinterface unseres Sinusbotâ??s Ã¶ffnen.

Category

1. Linux
2. Raspberry Pi
3. Sinusbot

Date Created
19.02.2026
Author
administrator