Raspberry Pi - Zero 2 W RTSP Server

Introduction

While setting up my smart home I was looking into creating a cost effective NVR that would allow for a high degree of privacy and control. The choice of NVR software fell on the excellent Frigate NVR, it integrates well with Home Assistant and supports hardware accelerated object detection via the little AI chip I had lying around.

The other part was having affordable, power-efficient camera hardware. There are many network-attached cameras on the market, but most of them require external services to set up/function and software wise they are black-boxes which raises considerable privacy concerns.

The choice fell on the Raspberry Pi Zero 2 running the default Raspberry Pi OS. The original Raspberry Pi Zero (v1) boards unfortunately are not suitable for streaming video in Full-HD, the CPUs were running at the limit and got too hot started heavily throttling. Fortunately the Raspberry Pi Zero 2 alleviated these issues at a similar price point (once they became available again in 2023/24).

The video to Frigate has to be an RTSP stream, to facilitate this we will be running MediaMTX via docker.

OS Setup

To set up the boot media for the Pi Zero we run Raspberry Pi imager

1. Click `Choose device` -> `Raspberry Pi Zero 2 W`.
2. Click `Choose OS` -> `Raspberry Pi OS (other)` -> `Raspberry Pi OS (Legacy, 32 bit) Lite`.
3. Click `Choose Storage` -> Select the SD Card.
4. Click `Next` -> `Edit Settings`.
5. Under `General` set desired hostname, credentials, etc.
6. Under `Services` -> `Enable SSH`. 7 Click `Yes` to create the boot media for the Pi Zero.

Now we transfer the Micro-SD card to the Raspberry Pi Zero with a connected camera.

Hardware check

We will do this part locally on the Raspberry Pi, so we need to connect a keyboard and display to it, for the Pi Zero we'll need two adapters to connect a keyboard and a monitor:

- USB-A -> micro-USB
- HDMI -> micro-HDMI

Boot up the Pi Zero and wait for the initial file system resize to complete.

Once it reboots we can log in as the picam user and do a short test of whether the camera is recognized by the OS:

Terminal window

```
libcamera-hello
```

This should open a window on top of the terminal screen for a few seconds that shows the a camera feed.

Network Setup

We want to set a static IP that will persist on the SD Card in case I want to switch it over to another Raspberry Pi.

For this we edit the dhcp daemon config

Terminal window

```
sudo nano /etc/dhcpcd.conf
```

Towards the bottom we should see some commented out entries we can use. We just uncomment them and change the `ip_address` to our desired IP:

/etc/dhcpcd.conf

```
static ip_address=192.168.1.100/24
```

```
static routers=192.168.1.1
```

```
static domain_name_server=192.168.1.1
```

If there are any Raspberry Pi configurations we still want to change (e.g. enabling SSH in case we haven’t yet or setting the hostname)

Once we reboot we should be able to SSH into the machine

Terminal window

```
reboot
```

Upgrade/Install dependencies

We will install docker from the Debian repo after updating the OS packages:

Terminal window

```
sudo apt-get update
```

```
sudo apt-get upgrade -y
```

```
sudo apt-get install -y docker.io
```

Now we need to add the user to the docker group so it will have the right permissions

Terminal window

```
sudo usermod -aG docker admin
```

```
newgrp docker
```

RSTP Server setup

With docker set up we can now use it to run MediaMTX as the RTSP server, lets start it up:

Terminal window

```
docker run --rm -it --network=host --privileged --tmpfs /dev/shm:exec -v
/run/udev:/run/udev:ro -e MTX-PATHS-CAM-SOURCE=rpiCamera
bluenviron/mediamtx:latest-ffmpeg-rpi
```

Once we verified it runs correctly we can set up the service. Create a script to run it:

Terminal window

```
mkdir -p /home/admin/src/scripts
```

```
nano /home/admin/src/scripts/mediamtx.sh
```

with the following content (note, we're removing `-it` from the above command for this script, as we don't run an interactive shell for the service):

/home/admin/src/scripts/mediamtx.sh

```
#!/bin/bash
```

```
# Will be exposed at rtsp://[ip]:8554/cam
```

```
docker run --rm --network=host --privileged --tmpfs /dev/shm:exec -v
/run/udev:/run/udev:ro -e MTX_PATHS_CAM_SOURCE=rpiCamera
```

```
bluenviron/mediamtx:latest-ffmpeg-rpi
```

then make it executable:

```
Terminal window
```

```
chmod +x /home/admin/src/scripts/mediamtx.sh
```

Now we add the service:

```
Terminal window
```

```
sudo nano /etc/systemd/system/mediamtx.service
```

with:

```
/etc/systemd/system/mediamtx.service
```

```
[Unit]
```

```
Description=MediaTX
```

```
After=network.target
```

```

```

```
[Service]
```

```
ExecStart=/home/admin/src/scripts/mediamtx.sh
```

```
Restart=always
```

```
RestartSec=3
```

```
User=admin
```

```
Group=docker
```

```
[Install]
```

```
WantedBy=default.target
```

Then we start it and activate startup on system start.

```
Terminal window
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl start mediamtx
```

```
# Check logs to see if it starts up correctly
```

```
journalctl -u mediamtx -f
```

```
sudo systemctl enable mediamtx
```

take from here

Revision #1
Created 8 May 2026 22:59:35 by Steve Ling
Updated 8 May 2026 23:03:47 by Steve Ling