DSP Monthly Project 01

Xu, Minhuan

October 12, 2022

Contents

Abstract

I will use my Raspberry Pi to launch a server of MJPG-streamer to broadcast the stream captured by the CSI camera and in help of fast reverse proxy to broadcast the live-stream to Public Network.

1 Introduction

1.1 Raspberry Pi

Raspberry Pi is a card micro computer which has almost everything a normal PC has, such as a 1000M Ethernet port, USB 3.0 and 2.0, HDMI output, 3.5 mm earphone port and the most important in my project, the CSI camera module.

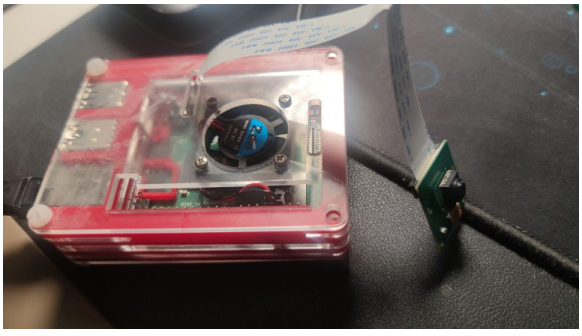

Figure 1: My Raspberry Pi

1.2 MJPG-streamer

MJPG-streamer is a command line application that copies JPEG frames from one or more input plugins to multiple output plugins. It can be used to stream JPEG files over an IP-based network from a webcam to various types of viewers such as Chrome, Firefox, VLC, and other software capable of receiving MJPG streams. [1]

1.3 FRP

frp is a fast reverse proxy to help you expose a local server behind a NAT or firewall to the Internet. As of now, it supports TCP and UDP, as well as HTTP and HTTPS protocols, where requests can be forwarded to internal services by domain name. [2]

2 Basic Setting

2.1 Imager

First, download the Raspberry Pi Imager. In this Imager, we can write the OS we want into the SD card and initialize the configure of the OS easily, like setting username (and password), enable the ssh (by which I can log in with the terminal in my computer).

2.2 Log-in

Second, put the SD card into my Raspberry Pi, plug in the Ethernet port (I have a router in dormitory). Then, plug in the 5 V power. My Raspberry Pi is turning on! Access my router with browser and find local IP of my Raspberry Pi which is 192.168.31.189. Then, open my terminal (in windows, PowerShell), and typing

```
PS C:\Users\xmh> ssh -l xmh 192.168.31.189
```

Since I'm going to use `apt-get` and `pip` to manage the programs, I want to change the source of them. It is easy to do that follow the instructions on TUNA [3].

2.3 Enable the Pi Camera

Raspberry PI's camera is turned off by default. So I need to enable it. Use `sudo raspi-config` to enter the system configure.

Here's the result.

(a) System Configure (b) Enable the Camera

Figure 2: raspi-config

3 Implement

3.1 Run MJPG-streamer server

It is not such easy to run a server of MJPG-streamer, so I found a repository in GitHub which has Instructions and helper scripts for running mjpg-streamer on Raspberry Pi [4].

It don't take too much time to implement that. After that I can access the server by browser using the IP mentioned in Section 2, soon I have the *first* photo. See Fig. 3.

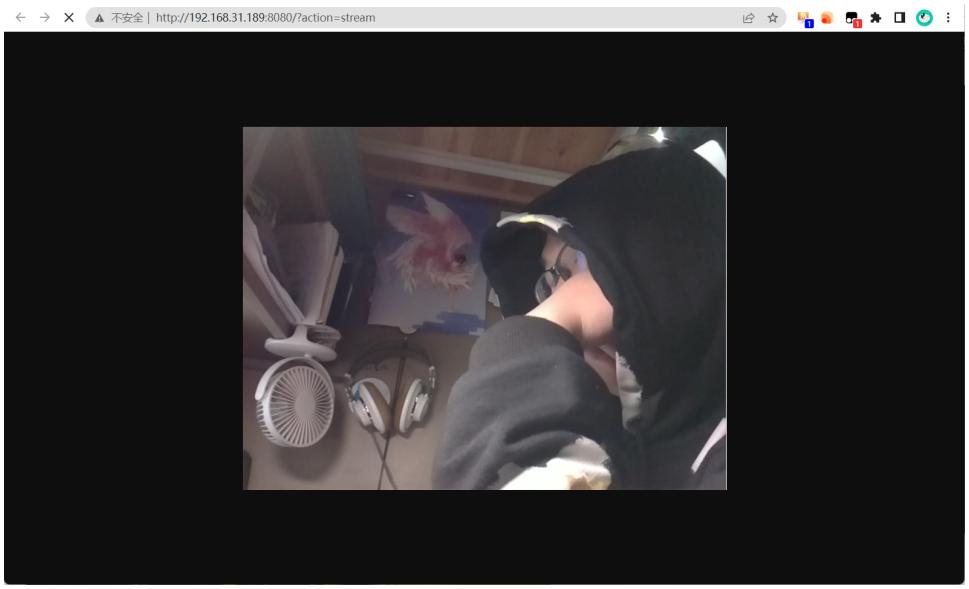

Figure 3: My First Photo by Pi Camera

3.2 Broadcast in School

The next problem is to Mirroring it to the Public Network. At the beginning, I try to use the iLZU to do that. According to my test before, If we can know the local IP we are using to access the iLZU (campus network), 2 devices connected to the iLZU can *find* each other. By referring the webpage `http://10.10.0.166:8800/home`, I knew the local IP is `172.23.xxx.xxx` and it changes in regular basis. I try to ask other students to have a try to access my Raspberry Pi, and it was a big success.

3.3 Fast Reverse Proxy

But I still want to access the stream server without connecting the campus network. So, I use fast reverse proxy [2]. For other reasons, I have a VPS which has its own IPv4. I deployed the server of frp on that machine and release the port 7000 as the proxy port and release the 80 as the service port. And deployed the client side on my Raspberry Pi. I will show my configure files in Code Listing section and that will be more clear.

So, I can now firstly access the 80 port of the VPS and then the VPS relay the data to my Raspberry Pi. See Fig. 4

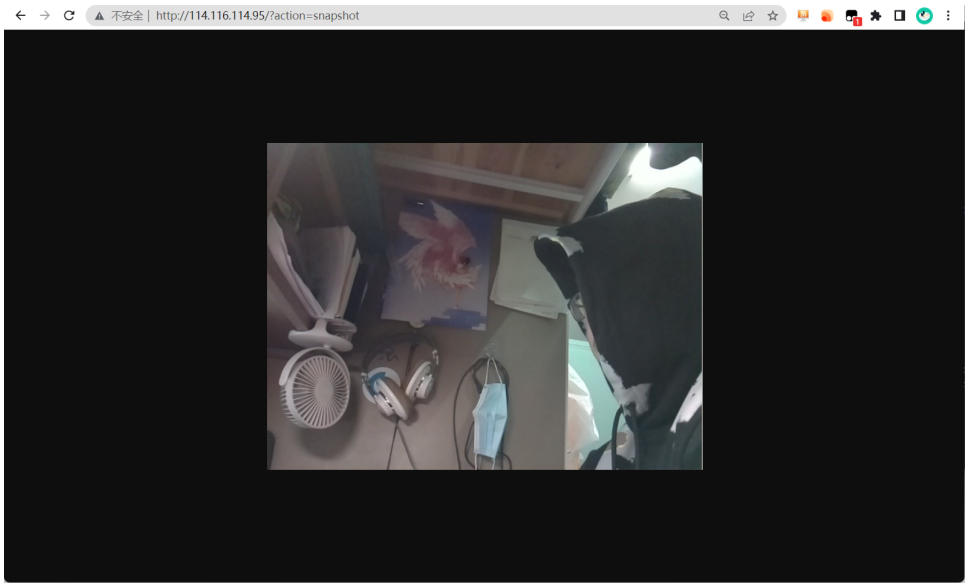

Figure 4: My Second Photo by Pi Camera

For safety's sake, I want to change the default web application username and password into `xmh` and `xmhxmhxmh`.

4 Preparation For OpenCV

The future goal is to server a face-recognition back end, so it is important to use python to control the camera. Then, deploying an environment of OpenCV is what I need to do.

`sudo apt-get install python3-opencv`

`pip3 install numpy --upgrade`

After installing OpenCV and numpy, I shouldn't forget to install the packets below, otherwise python will tell me to install it.

`sudo apt-get install libatlas-base-dev`

I write the code to let python to control camera and write the image into file. See in the Code Listing section.

References

[1] https://github.com/jacksonliam/mjpg-streamer/.

[2] https://github.com/fatedier/frp/.

[3] https://mirrors.tuna.tsinghua.edu.cn/.

[4] https://github.com/meinside/rpi-mjpg-streamer.

Appendix A Code Listing

```python
#use OpenCV to take a photo and save it
import cv2
import numpy

#init
camera = cv2.VideoCapture(1)

#read the camera
ret,img = camera.read()

#save to snapshot.jpg
cv2.imwrite('./img.jpg',img)

#release the camera
camera.release()
cv2.destroyAllWindows()
```

```python
#use http request to download the snapshot generated by MJPG-streamer
import requests

od = {'action':'snapshot'}

authe = ('xmh','xmhxmhxmh')

response = requests.get("http://mc.xwxstudio.com/?",params=od, auth=authe)
response.raise_for_status()
with open('./snapshot.jpg','wb') as f:
    f.write(response.content)
```

```
//the configure of frp server
[common]
bind_port = 7000
```

```
// the configure of frp client
[common]
server_addr = 114.116.114.95
server_port = 7000

[webcam]
type = tcp
local_ip = 127.0.0.1
local_port = 8080
remote_port = 80
```