



# Collecting Image Data Using an RTSP Stream

This guide will show you how to establish a live RTSP stream with a local webcam and use it to collect image data!

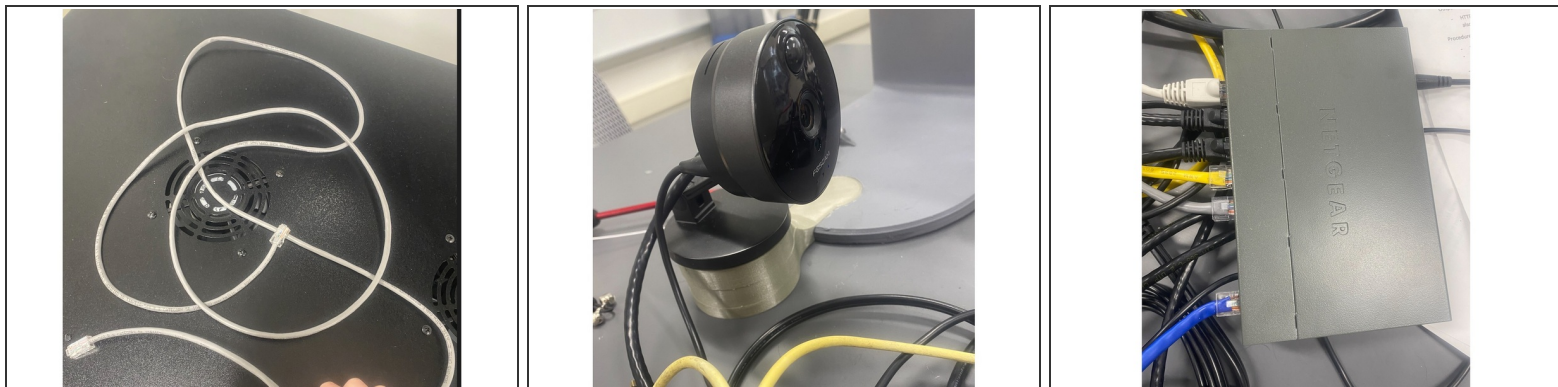
Written By: Tyler Burns



## INTRODUCTION

This guide will walk you step by step through using a Linux or Mac based device to access an RTSP (Real Time Streaming Protocol) stream of a webcam connected to the same local network as your computer and run a simple python script to begin capturing and storing images from the live RTSP feed. Perfect for creating a machine learning data set or gathering images for later analysis!

## Step 1 — Bill of Materials - Getting Started



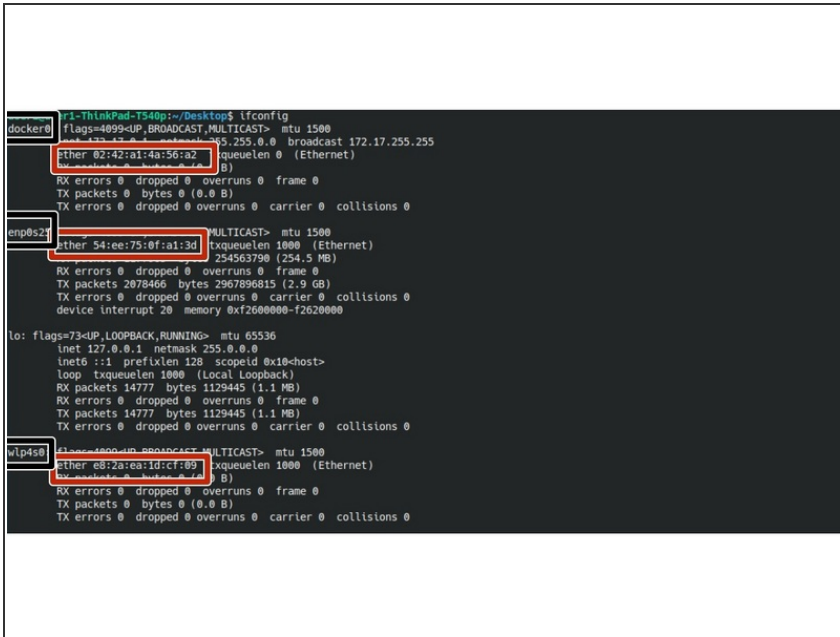
- An **RTSP** compatible **webcam**
- A Computer that has access to **Linux command line utilities** (usually a device with MacOS or a Linux/unix operating system)
- The computer must also have [Python 3](#). As well, as the the [openCV](#) Python library installed. Links to instructions to install each are embedded.
- A Network Switch
- 2x Ethernet cables (Cat 5 or higher)

## Step 2 — Connecting to the Switch and Finding your Camera



- In order to Connect to the RTSP stream of our camera to establish a live feed we first need to find the IP address (a unique address that identifies a device on a local network) of our camera
- To find the IP address we first need to determine the network protocol being used on your server switch.
- To do this first connect just your linux based terminal and camera to the switch via Ethernet and make not of your cameras MAC address
- This will be found on the camera or in the documentation accompanying it.

## Step 3 — finding network protocol and IP address



```
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
        ether 02:42:a1:4a:56:a2 (Ethernet)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 0 bytes 0 (0.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s2f: flags=4163<UP,BROADCAST,MULTICAST> mtu 1500
        ether 54:ee:75:0f:a1:3d (Ethernet)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 2078466 bytes 2967896815 (2.9 GB)
        RX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
        device interrupt 20 memory 0xf2600000-f2620000

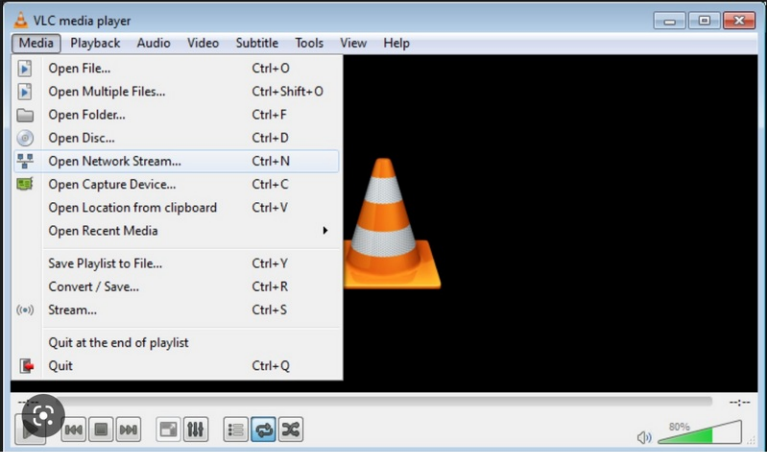
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x1<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 14777 bytes 1129445 (1.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14777 bytes 1129445 (1.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp4s8: flags=4163<UP,BROADCAST,MULTICAST> mtu 1500
        ether e8:2a:ea:1d:c7:09 (Ethernet)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 0 bytes 0 (0.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- Now open the linux terminal on your computer and run the command `ifconfig` in the terminal this should bring up a list of network protocols along with information that accompanies them
- Now look at the ether heading under each protocol and find the MAC address that corresponds your camera. The protocol that has your camera in the ether will be your network protocol!
- Next type the following command into your linux terminal: `sudo arp-scan --interface=<your_network_interface> --localnet`
- Note: Replace `<your_network_interface>` with the name of the network interface that you are using to connect to the switch (e.g., `eth0`, `wlan0`, `enp3s0`, etc.).
- after the common has finished running look for the entry that corresponds to your webcam. The entry will typically include the manufacturer's name or a model number. The IP address of your webcam will be listed in the first column of the entry.



## Step 4 — Connecting to the camera's RTSP stream

A screenshot of the VLC media player interface. The 'Media' menu is open, showing options like 'Open File...', 'Open Multiple Files...', 'Open Folder...', 'Open Disc...', 'Open Network Stream...', 'Open Capture Device...', 'Open Location from clipboard', 'Open Recent Media', 'Save Playlist to File...', 'Convert / Save...', 'Stream...', 'Quit at the end of playlist', and 'Quit'. The 'Open Network Stream...' option is highlighted. The background shows a traffic cone icon.

### Hikvision RTSP Stream (NVR)

`rtsp://<address>:<port>/Streaming/Channels/<id>`

`<address>` - DDNS or IP for the system

`<port>` - RTSP port (typically 554)

`<id>` - streaming channel - for example "101" would be channel 1 and stream #1 (mainstream). Changing the `<id>` to 102 would create channel 1 and stream #2. 201 would produce the channel #2 of the NVR and mainstream and 202 would produce channel #2 and substream.

Example: `rtsp://192.168.1.210:554/Streaming/Channels/101`

This RTSP stream would mainstream channel 1 on an NVR at 192.168.1.210.

You can also encode credentials into the URL by entering it prior to the IP address - for example.

Example URL: `rtsp://admin:12345@192.168.1.210:554/Streaming/Channels/101` will bring up mainstream for channel 1, where admin is the username and 12345 is the password.

- Congratulations you have found your IP address the hardest part of finding your Camera's RTSP URL.
- Your camera's specific RTSP URL will depend on the manufacturer of the camera however the formatting for the RTSP URL for nearly every manufacturer can be found [here](#)
- The link above will provide specific instructions for your camera based on manufacturer a couple tips that apply across the board however are that the Port field of the stream format will typically be 554 and any usernames or passwords can either be found on your camera or in its documentation unless otherwise changed.
- You can test your RTSP URL to make sure it is correct and running properly by opening your computers VLC media player going to the media tab going to the open network stream button and typing in your RTSP URL

## Step 5 — Collecting Images Using the RTSP Stream

```
#amount of time desired between each image captured from RTSP stream
imagecapturetime = 1
# enterfolder path of where images should be saved
dir_path = ''
#enter the RTSP URL here
RTSP_URL = ''
number_of_images = 100
```

```
#add timestampStr to image name if you would like timestamped images
timestampStr = realtime.strftime("%d-%b-%Y (%H:%M:%S.%f)")
#writes image and numbers it in order of images taken in folder
cv2.imwrite(os.path.join(dir_path , f'{n}.png'), frame)
```

- Now that you have your RTSP Stream URL working it is time to use your simply copy the code I wrote [here](#)
- To run the code first specify the how often you would like the code to collect an image, the file path of the directory you would like the images saved to, the RTSP URL and, the number of images you want to collect!
- The coded has the added ability to display the RTSP feed while images are being collected as well as the functionality to timestamp the images if the timestamp is added in brackets to the [fstring](#) that is the image file names
- Now run your code from the terminal and collect data! Perfect for generating data from a machine learning model or sending data to some sort of image analysis tool!