

Project2\_Report (2181105).docx

Date: 2018-05-05 09:03 UTC

\* All sources 53 | Internet sources 21

✓ [42]	https://www.quora.com/How-can-I-control-...s-not-locally-hosted	2.0% 1 matches
✓ [43]	https://www.reddit.com/user/its_bananas	1.9% 4 matches
✓ [44]	https://www.raspberrypi.org/forums/viewtopic.php?p=1193375	2.0% 2 matches
✓ [45]	https://www.reddit.com/r/raspberry_pi/co...om_anywhere_without/	1.8% 4 matches
✓ [46]	https://yotekai.com/q_vnc server/	1.6% 4 matches
✓ [47]	https://gist.github.com/Lauszus/c8ce73f3177d6455c27c	1.9% 7 matches
✓ [48]	https://github.com/jhmoon1211/CATMINT/blob/master/install UV4L.txt	1.5% 4 matches
✓ [49]	https://www.raspberrypi.org/forums/viewtopic.php?t=187358	0.9% 4 matches
✓ [50]	https://www.linux-projects.org/uv4l/tutorials/desktop-streaming/	0.7% 3 matches
✓ [51]	comlogins.net/vnc-viewer	0.7% 1 matches
✓ [52]	raspberry-lp2maray.blogspot.com/2016/11/tentang-vnc-di-raspberry-pi.html	0.7% 2 matches
✓ [53]	https://github.com/Microsoft/Windows-iot...xpander/CS/README.md	0.4% 2 matches
✓ [54]	https://support.realvnc.com/Knowledgebas...n-realvnc-viewer-app	0.5% 1 matches
✓ [55]	https://www.youtube.com/watch?v=EjUvur6DMBQ	0.4% 1 matches 2 documents with identical matches
✓ [58]	https://www.youtube.com/watch?v=A0pXZHAj3w	0.4% 1 matches
✓ [59]	https://www.reddit.com/r/raspberry_pi/co...e_a_project_idea_in/	0.3% 1 matches
✓ [60]	https://www.raspberrypi.org/forums/viewtopic.php?p=899866	0.3% 1 matches
✓ [61]	https://www.raspberrypi.org/products/raspberry-pi-zero/	0.3% 1 matches
✓ [62]	https://www.ibm.com/developerworks/bpm/l...oy/1206_mcelroy.html	0.2% 1 matches
✓ [63]	https://superuser.com/questions/194090/h...am-on-the-start-menu	0.2% 1 matches
✓ [64]	https://superuser.com/a/1151028	0.2% 1 matches

15 pages, 2711 words

⚠ A very light text-color was detected that might conceal letters used to merge words.

PlagLevel: selected / overall

142 matches from 65 sources, of which 65 are online sources.

Settings

Data policy: Compare with web sources, Check against my documents

Sensitivity: Medium

Bibliography: *Consider text*

Citation detection: *Reduce PlagLevel*

Whitelist: --

# INTRODUCTION

**Abstract:**

The aim of this project is to automate everyday tasks with the help of a Pico-projector module.

# Classroom Automation

## Abstract:

The aim of this project is to automate everyday tasks with the help of a Pico-projector module.

## Tasks:

Remote Desktop Access (SSH/VNC)

Remote Desktop Sharing

Remote Desktop Mirroring (UV4L)

PDF Presenter #1 (Manual)

- \* PDF presentation with interactive support. To be used by teacher manually.

PDF Presenter #2 (Automatic)

- \* Added Voice-over using TTS support. Subtitles (lecture notes) have to be provided.

PDF Presenter #3 (Lab Assistant Support)

- \* Equipment identification & content retrieval using QR codes.

PDF Presenter #4 (Programming Support)

- \* Using Jupyter Notebooks & Firefox Browser addons

PDF Presenter #5 (Classroom Support)

- \* Student attendance logging, multi-user interaction, test paper evaluation verification.

## Remote Desktop Access

Sometimes you need to access a Raspberry Pi without connecting it to a monitor. Perhaps the Pi is embedded in something like a robot, or you may want to view some information from it from elsewhere. Maybe you simply don't have a spare monitor! <sup>[55]</sup> You can find more information on accessing your Raspberry Pi remotely <sup>[42]</sup> here.

- [IP address](#)
  - How to find your Raspberry Pi's IP address in order to connect to it
- [Access over Internet](#)
  - Remote access to the Pi over the internet using Weaved or Dataplicity
- [VNC](#)
  - Remote access to the Pi's graphical interface, viewed in a window on another computer
- [SSH](#)
  - Access the command line of the Pi from another computer
- [SFTP](#)
  - Copy files between your Pi and another computer using SFTP (Secure File Transfer Protocol)
- [SCP](#)
  - Copy files between your Pi and another computer using SCP (Secure Copy Protocol)
- [SSHFS](#)
  - Copy files between your Pi and another computer using SSHFS (Secure Shell Filesystem)
- [rsync](#)
  - Synchronise folders between the Pi and another computer using rsync over SSH
- [FTP](#)
  - Copy files between your Pi and another computer using FTP (File Transfer Protocol)
- [Web Server](#)
  - Set up a website or a web page to display some information about the Pi, using a web browser on another machine, on the network or on the internet.

## VNC (Virtual Network Computing)

Sometimes it is not convenient to work directly on the Raspberry Pi. Maybe you would like to work on it from another device by remote control.

<sup>[46]</sup> VNC is a graphical desktop sharing system that allows you to remotely control the desktop interface of one computer (running VNC Server) from another computer or mobile device (running VNC Viewer). <sup>[46]</sup> VNC Viewer transmits the keyboard and either mouse or touch events to VNC Server, and receives updates to the screen in return.

You will see the desktop of the Raspberry Pi inside a window on your computer or mobile device. You'll be able to control it as though you were working on the Raspberry Pi itself.



VNC Connect from RealVNC is included with Raspbian. It consists of both VNC Server, which allows you to control your Raspberry Pi remotely, and VNC Viewer, which allows you to control desktop computers remotely from your Raspberry Pi should you want to.

You must enable VNC Server before you can use it: instructions for this are given below. <sup>[52]</sup> By default, VNC Server gives you remote access to the graphical desktop that is running on your Raspberry Pi, as though you were sitting in front of it.

<sup>[52]</sup> However, you can also use VNC Server to gain graphical remote access to your Raspberry Pi if it is headless or not running a graphical desktop. For more information on this, see Creating a virtual desktop, further below.

#### Enabling VNC Server

On your Raspberry Pi, run the following commands to make sure you have the latest version of VNC Connect:

<sup>[47]</sup> `sudo apt-get update`  
`sudo apt-get install realvnc-vnc-server realvnc-vnc-viewer`

Now enable VNC Server. You can do this graphically or at the command line.

Enabling VNC Server graphically

On your Raspberry Pi, boot into the graphical desktop.

Select Menu Preferences Raspberry Pi Configuration Interfaces.

Ensure VNC is Enabled.

Enabling VNC Server at the command line

You can enable VNC Server at the command line using raspi-config:

`sudo raspi-config`

Now, enable VNC Server by doing the following:

Navigate to Interfacing Options.

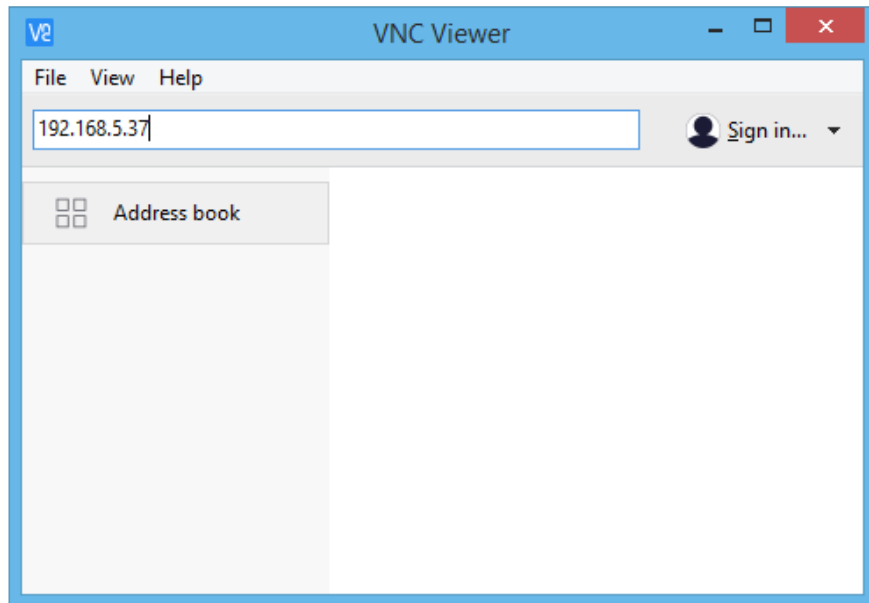
Scroll down and select VNC Yes.

#### Connecting to your Raspberry Pi with VNC Viewer

There are two ways to connect to your Raspberry Pi. You can use either or both, depending on what works best for you.

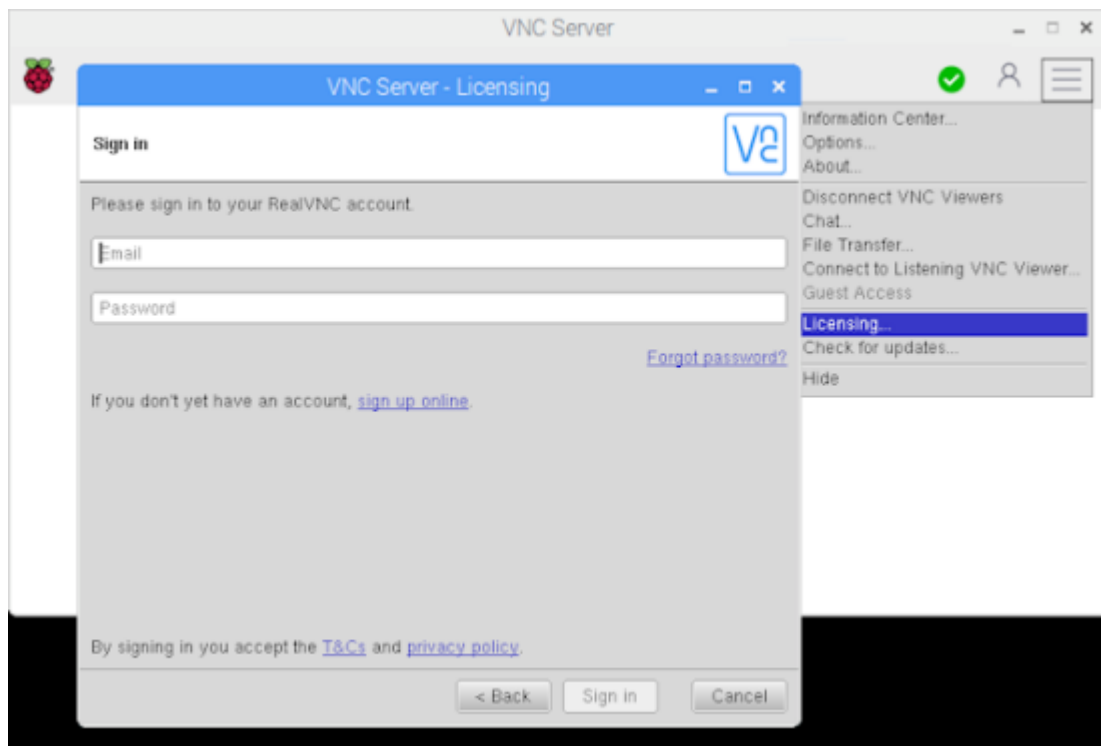
#### Establishing a direct connection

1. Direct connections are quick and simple providing you're joined to the same private local network as your Raspberry Pi. For example, this might be a wired or wireless network at home, at school, or in the office).
2. On your Raspberry Pi (using a terminal window or via SSH) use these instructions or run `ifconfig` to discover your private IP address.
3. **On the device you'll use to take control, download VNC Viewer.** For best results, use the compatible app from RealVNC.
4. Enter your Raspberry Pi's private IP address into VNC Viewer:



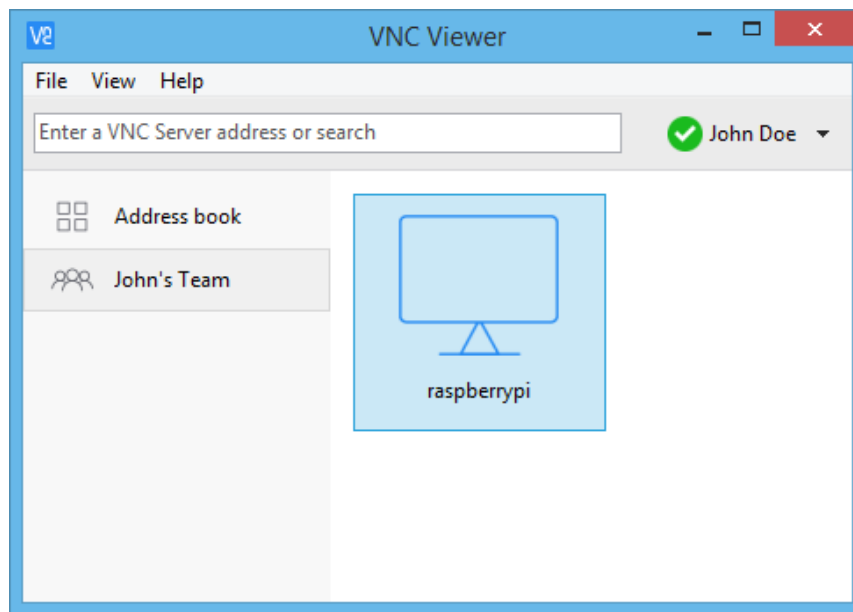
#### Establishing a cloud connection

1. You are entitled to use RealVNC's cloud service for free, provided that remote access is for educational or non-commercial purposes only.
2. **Cloud connections are convenient and encrypted end-to-end.** They are highly recommended for connecting to your Raspberry Pi over the internet. There's no firewall or router reconfiguration, and you don't need to know the IP address of your Raspberry Pi, or provide a static one.
3. **Sign up for a RealVNC account here: it's free and it only takes a few seconds.**
4. **On your Raspberry Pi, sign in to VNC Server using your new RealVNC account credentials:**



5. On the device you'll use to take control, download VNC Viewer. You must use the compatible app from RealVNC.
6. Sign in to VNC Viewer using the same RealVNC account credentials, and then either tap or click to connect to your Raspberry Pi:





#### Authenticating to VNC Server

1. To complete either a direct or cloud connection, you must authenticate to VNC Server.
2. If you're connecting from the compatible VNC Viewer app from RealVNC, enter the user name and password you normally use to log in to your user account on the Raspberry Pi. By default, these credentials are pi and raspberry.
3. If you're connecting from a non-RealVNC Viewer app, you'll first need to downgrade VNC Server's authentication scheme, specify a password unique to VNC Server, and then enter that instead.
4. If you are in front of your Raspberry Pi and can see its screen, open the VNC Server dialog on your Raspberry Pi, select Menu Options Security, and choose VNC password from the Authentication dropdown.
5. Or if you're configuring your Raspberry Pi remotely from the command line, then to make the changes for Service Mode (the default configuration for the Raspberry Pi):
  - Open the /root/.vnc/config.d/vncserver-x11 config file.
  - Replace Authentication=SystemAuth with Authentication=VncAuth and save the file.
  - In the command line, run `sudo vncpasswd -service`. This will prompt you to set a password, and will insert it for you in the right config file for VNC Server running in Service Mode.
  - Restart VNC Server.

The core of the UV4L software falls in the middleware category. It consists of a series of highly configurable drivers, an optional Streaming Server component providing a [RESTful API](#) for custom development and various extensions for the server that cooperate together. The Streaming Server also provides the necessary web UI for the end-users to try or use all the key functionalities directly. For maximum efficiency, each instance of UV4L runs as a single system process which exploits the underlying hardware natively whenever possible. Here is a more detailed [list of features](#).

Below we will see how to install all the components to get the best from UV4L, with particular focus on the driver for the Raspberry Pi camera boards for the sake of explanation, although all other drivers can be optionally installed in a similar way.

This same driver has been extended to support the TC358743 HDMI to MIPI chipset converter on all Raspberry Pi boards (this chipset is present on the [B101 HDMI to CSI-2 Bridge](#), for example). However, instructions on how to enable the TC358743 on boards different from Raspberry Pi 3 (e.g. Zero, ZeroW, CM3, etc...) will be provided [upon request](#) only.

If you are running Raspbian Wheezy or Raspbian Jessie, open a terminal and type the following commands:

```
$ curl http://www.linux-projects.[47]org/listing/uv4l_repo/lrkey.asc | sudo apt-key add -
```

On Raspbian Jessie add this line:

```
deb http://www.linux-projects.[47]org/listing/uv4l_repo/raspbian/ jessie main
```

Finally, we are ready to update the system and to fetch and install the packages:

```
[47]$ sudo apt-get update
$ sudo apt-get install uv4l uv4l-raspicam
```

The above two commands will upgrade UV4L to the most recent version, if it's already installed.

If you want the driver to be loaded at boot, also install this optional package:

```
$ sudo apt-get install uv4l-raspicam-extras
```

As a convenience, the above package will install a service script for starting, stopping or restarting the driver at any time, for example:

```
[48]$ sudo service uv4l_raspicam restart
```

When (re)starting the service, uv4l will be instructed to parse the configuration file `/etc/uv4l/uv4l-raspicam.conf` to get the default values for the driver and the server options. You can edit that file to add, remove or change the default values. This same service is started at boot.

If you are using the TC358743, the `uv4l-tc358743-extras` package has to be installed for it to work:

```
$ sudo apt-get install uv4l-tc358743-extras
```

The above package will automatically install the `uv4l-raspicam-extras` package and some other helper programs. Before using the TC358743 make sure that both the Camera interface and the I2C bus are enabled in the `raspi-config` system tool and that the line `tc358743=yes` is present or uncommented in the configuration file `/etc/uv4l/uv4l-raspicam.conf`.

Now the UV4L core component and the Video4Linux2 driver for the CSI Camera Board are installed. If you occasionally get unexpected errors from the driver, make sure the camera is enabled and enough memory is reserved for the GPU (not less than 128MB is suggested) from this menu:

```
$ sudo raspi-config
```

Also consider updating the firmware with the following command:

```
$ sudo rpi-update
```

For detailed informations, options, etc... about the modules installed type accordingly:

```
$ man uv4l
$ man uv4l-raspicam
```

To get the list of all available options:

```
[47] $ uv4l --help --driver raspicam --driver-help
```

If you have not installed the optional `uv4l-raspicam-extras` package (which provides a convenient script for starting uv4l with the settings taken from a configuration file) and want to quickly test uv4l, load it manually:

```
[49] $ uv4l --driver raspicam --auto-video_nr --width 640 --height 480 --encoding
jpeg
```

and take a JPEG snapshot from the camera:

```
[49] $ dd if=/dev/video0 of=snapshot.jpeg bs=11M count=1
```

For a list of other use cases click [here](#).

To manually terminate a running driver, close all the applications accessing the device and kill the corresponding uv4l process:

```
$ pkill uv4l
```

Apart from the driver for the Raspberry Pi Camera Board, the following Streaming Server front-end and drivers can be optionally installed:

```
$ sudo apt-get install uv4l-server uv4l-uvc uv4l-xscreen uv4l-mjpegstream
uv4l-dummy uv4l-raspisp
```

for which the manual pages are available:

```
[58] $ man uv4l-server
$ man uv4l-uvc
$ man uv4l-xscreen
$ man uv4l-mjpegstream
$ man uv4l-dummy
$ man uv4l-raspisp
```

The WebRTC extension for the Streaming Server is also available with two alternative packages depending on the Raspberry Pi model in use. If you have a Raspberry Pi 1, Compute Module 1, Zero or Zero W (Wireless), type:

```
$ sudo apt-get install uv4l-webrtc-armv6
```

Otherwise, if you have any other model (e.g. <sup>[53]</sup> **Raspberry Pi 2 or 3**), type:

```
$ sudo apt-get install uv4l-webrtc
```

As the Streaming Server is able to serve and run any custom web applications, an optional package containing the source code of some of the demos mentioned in the examples (e.g. [this one](#)) is available. The files will be installed in `/usr/share/uv4l/demos/`:

```
$ sudo apt-get install uv4l-demos
```

Note that some browsers may no longer exploit many of the WebRTC functionalities over HTTP for security reasons. You will need to configure secure HTTPS in the Streaming Server instead. To do this, you must provide a password-less private key and a valid certificate via the `--ssl-private-key-file` and the `--ssl-certificate-file` server options. A private key and a self-signed certificate can be generated as follows:

```
$ openssl genrsa -out selfsign.key 2048 && openssl req -new -x509 -key
selfsign.key -out selfsign.crt -sha256
```

Once you have installed and eventually configured the HTTP(S) Streaming Server module as shown above, make sure to reload uv4l for it to notice and start the server. Afterwards you can

access the server with the browser at the default address and port `https://raspberrypi:8080/`<sup>[50]</sup> (where `raspberrypi` has to be replaced with the actual hostname or IP address of your RaspberryPi and the protocol can be either `http` or `https`).

## Screen and audio mirroring to the Raspberry Pi

In this example we will see how to mirror the PC or laptop desktop's screen and audio-in (e.g. microphone) to the Raspberry Pi display and speakers with Firefox or Chrome. No special software or browser plug-in needs to be installed on the PC or laptop as a web application with the basic functionalities is provided by the UV4L Streaming Server and gets downloaded by the browser by simply visiting the appropriate URL. As opposite to many other screen sharing applications, no X Server is required on the Raspberry Pi, as UV4L makes use of the display natively.

It's of course possible for you to customize or install your own screen sharing web application, but this is out of scope now (see [this](#) example if you are interested).

### Requirements:

1. download and install the UV4L software on your Raspberry Pi. <sup>[62]</sup> For the purposes of this simple example, you will only really need the following packages: uv4l, uv4l-dummy, uv4l-server, uv4l-webrtc. Please refer to the [installation instructions](#).
2. HTTPS must be enabled in the UV4L Streaming Server for the browser to share the desktop. To do this, you will need a SSL private key and a valid SSL certificate. If you do not have them already, you can generate them by running the following command on the Raspberry Pi:

```
sudo bash -c "openssl genrsa -out /etc/ssl/private/selfsign.key 2048 &&
openssl req -new -x509 -key /etc/ssl/private/selfsign.key -out
/etc/ssl/private/selfsign.crt -sha256"
```

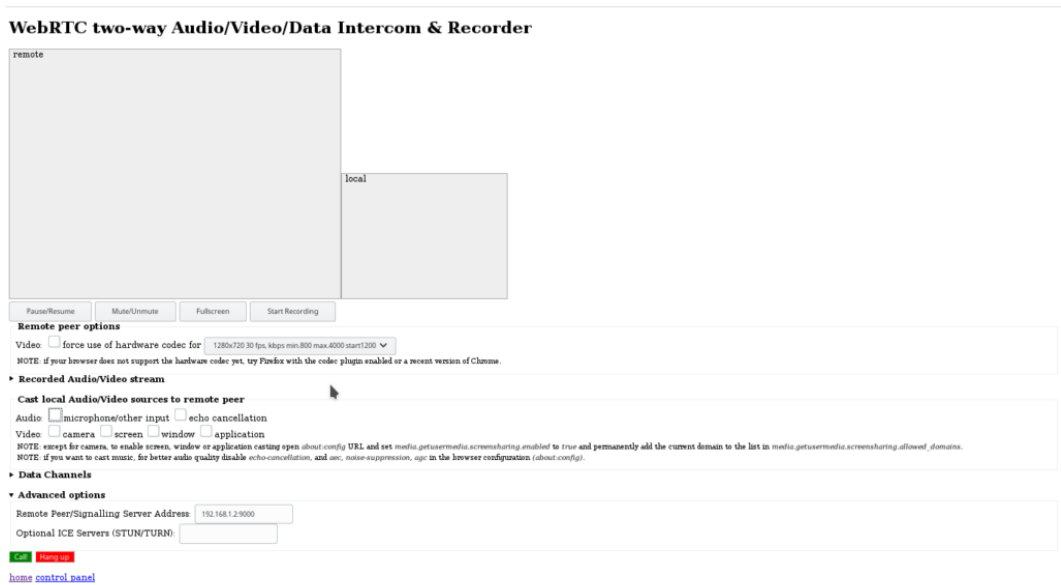
3. Open Firefox on your PC or laptop, type about:config in the address bar and press enter. In the Search field, type media.getusermedia.screensharing.<sup>[63]</sup>enabled and toggle its value to true with a double-click. Now search for media.getusermedia.screensharing.allowed\_domains and add your Raspberry Pi hostname or IP address or domain name to the list in the string (e.g. 192.168.1.2).

Now you are ready. For example, assuming you have a Full-HD display attached to the Raspberry Pi, open a terminal on the Raspberry Pi and run a new instance of uv4l with a command like this (in one line):

```
uv4l --driver dummy --auto-video_nr --enable-server --server-option '--use-
ssl=yes' --server-option '--ssl-private-key-
file=/etc/ssl/private/selfsign.key' --server-option '--ssl-certificate-
file=/etc/ssl/private/selfsign.crt' --server-option '--enable-webrtc-
video=no' --server-option '--enable-webrtc-audio=no' --server-option '--
webrtc-receive-video=yes' --server-option '--webrtc-receive-audio=yes' --
server-option '--webrtc-renderer-window=0 0 1920 1080' --server-option '--
webrtc-renderer-fullscreen=yes' --server-option '--webrtc-receive-
datachannels=no' --server-option '--port=9000'
```

At this point the UV4L Streaming Server should be running and listening to the HTTPS port 9000 for incoming connections. Open a tab in Firefox and enter the URL of the web page that allows you to mirror the desktop screen (or even a just window) and optionally cast the microphone, i.e. `https://raspberrypi:9000/stream/webrtc`, where “raspberrypi” has to be replaced with the hostname or IP address of your Raspberry Pi in the network (on the first access you may be asked to trust the self-signed certificate if you have generated them as shown above).

The web page should look like this:



To

start casting all have to do is to check the two “Audio: microphone/other input” and the “Video: screen” checkboxes under the “Cast local Audio/Video sources to remote peer” section and click on the green button “Call”. A browser pop-up will then appear asking you to select the screen and audio input device and to allow sharing the media sources:



## WebRTC two-way video recorder

remote

Pause/Resume

Mute/Unmute

Remote peer options

Video: ☐ force use of hardware acceleration

NOTE: if your browser does not support hardware acceleration, this option will be disabled.

Recorded Audio/Video stream

Cast local Audio/Video sources to remote peer

Audio: ☒ microphone/other input ☐ echo cancellation

Video: ☐ camera ☒ screen ☐ window ☐ application

NOTE: except for camera, to enable screen, window or application casting open about:config URL and set media.getusermedia.screensharing.enabled to true and permanently add the current domain to the list in media.getusermedia.screensharing.allowed\_domains.


NOTE: if you want to cast mouse, for better audio quality disable echo-cancellation, and aec, noise-suppression, agc in the browser configuration (about:config).


Data Channels

Advanced options

Remote Peer/Signalling Server Address: 192.168.1.2:9000

Optional ICE Servers (STUN/TURN):


 [Meeting call](#)

 [home control panel](#)

Will you allow 192.168.1.2 to use your microphone and see your screen?

Screen to share: Entire screen

All visible windows on your screen will be shared.



Only share screens with sites you trust. Sharing can allow deceptive sites to browse as you and steal your private data. [Learn More](#)

Microphone to share: default: Audio interno Stereo analogico

☐ Remember this decision

Don't Allow

Allow



