

RDK-C : Video Playback With WebRtc(Web Real-Time Communication)

- [Introduction](#)
- [Build and Flash Procedure](#)
- [server and client connectivity diagram](#)
- [openwebrtc Compilation Procedure For X86](#)
- [Validation Procedure of Webrtc](#)

Introduction

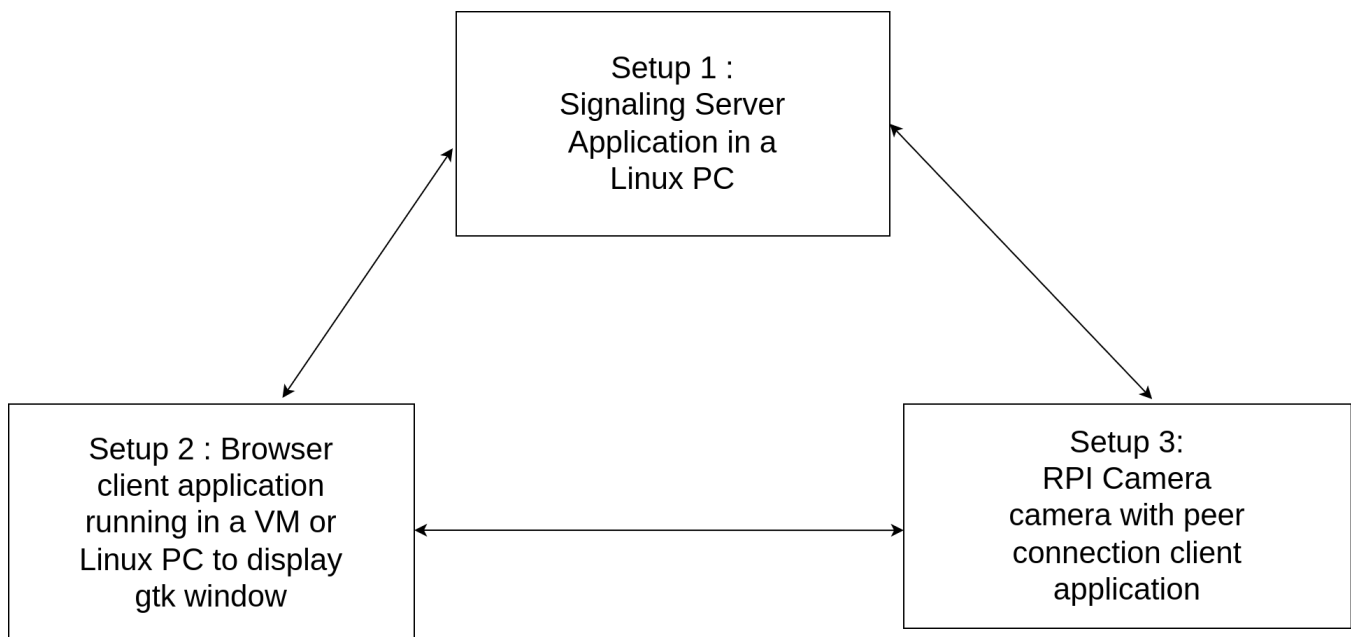
WebRTC (Web Real-Time Communication) is a technology that enables Web applications and sites to capture and optionally stream audio and/or video media, as well as to exchange arbitrary data between browsers without requiring an intermediary. The set of standards that comprise WebRTC makes it possible to share data and perform teleconferencing peer-to-peer, without requiring that the user install plug-ins or any other third-party software.

Build and Flash Procedure

Refer below link to build camera image

[RDK-C rdk-next Yocto 3.1 dunfell build for Raspberrypi](#)

server and client connectivity diagram



openwebrtc Compilation Procedure For X86

STEP 1:

Below command is cloning the openwebrtc source into current PC.

git clone <https://github.com/rdkcteam/native-webrtc.git>

Console output

```
user@BLTSLRM110:~$ git clone https://github.com/rdkcteam/native-webrtc.git
Cloning into 'native-webrtc'...
remote: Enumerating objects: 49, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 49 (delta 11), reused 41 (delta 11), pack-reused 8
Unpacking objects: 100% (49/49), 41.97 KiB | 511.00 KiB/s, done.
```

STEP 2:

Goto native-webrtc/PC_Streamer folder and give below export commands.

sudo chmod 777 webrtc_browser.sh

./webrtc_browser.sh

Console output

```
user@BLTSLRM110:~$ cd native-webrtc/PC_Streamer/
user@BLTSLRM110:~/native-webrtc/PC_Streamer$ sudo chmod 777 webrtc_browser.sh
[sudo] password for user:
user@BLTSLRM110:~/native-webrtc/PC_Streamer$ ./webrtc_browser.sh
```

STEP 3:

Completion of step2 the binaries are generated into **native-webrtc/PC_Streamer/webrtc-checkout/out/Default** folder inside.

Validation Procedure of Webrtc

Follwing steps are run into Desktop PC(X86) Side:

STEP 1 , STEP 2, STEP 3, STEP 5 , STEP 6, STEP 7.

Follwing step are run into RPI Board Side:

STEP 4

STEP 1:

Run the peerconnection_server binary located at (native-webrtc/PC_Streamer/webrtc-checkout).

./peerconnection_server

Console output

```
user@BLTSLRM110:~/Desktop/openwebrtc-m72/out/Default$ ./peerconnection_server
Server listening on port 8888
```

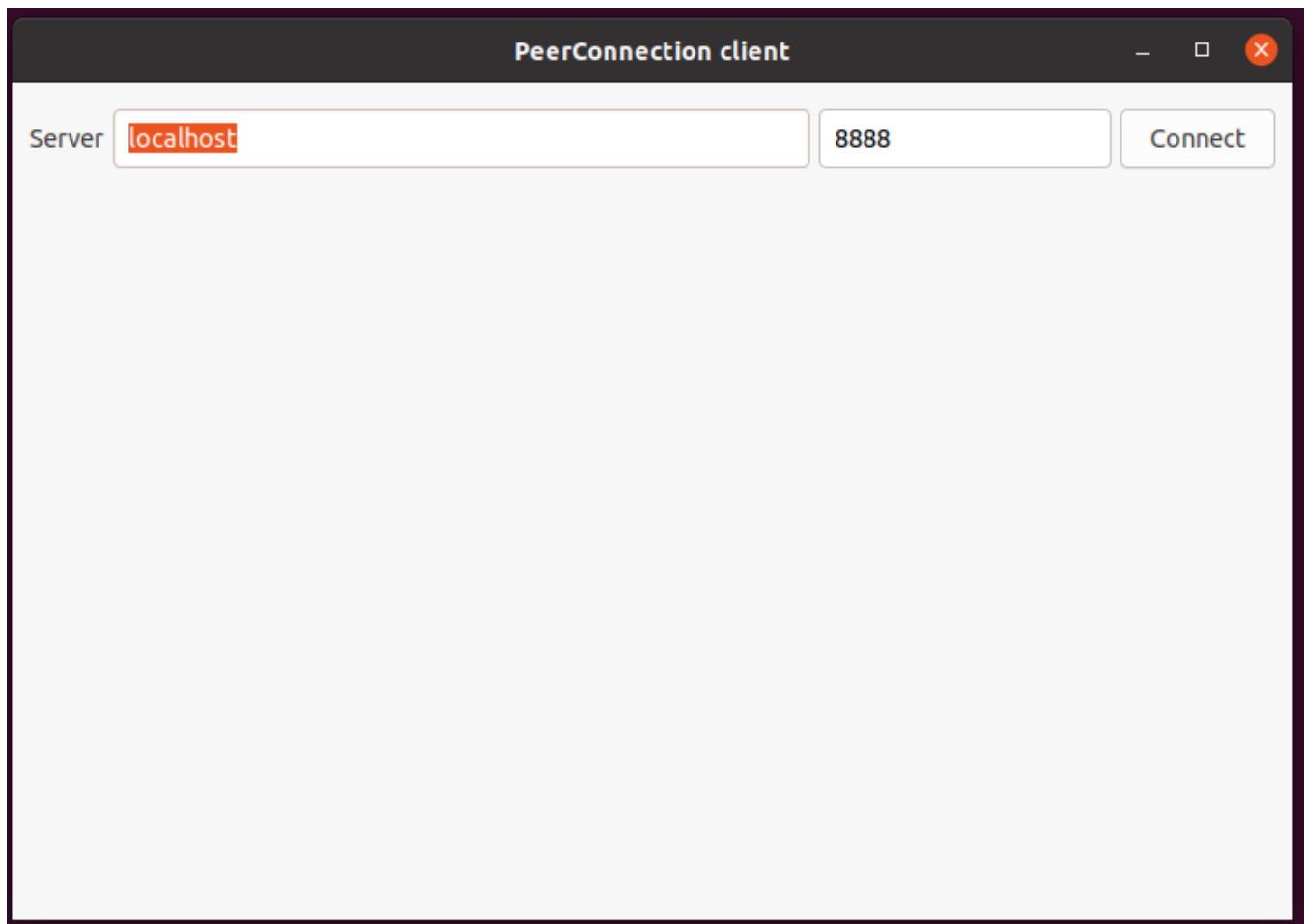
STEP 2:

Open new terminal Goto **openwebrtc-m72/out/Default** path, run the below two commands one by one.

export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:\$(pwd)

./peerconnection_client

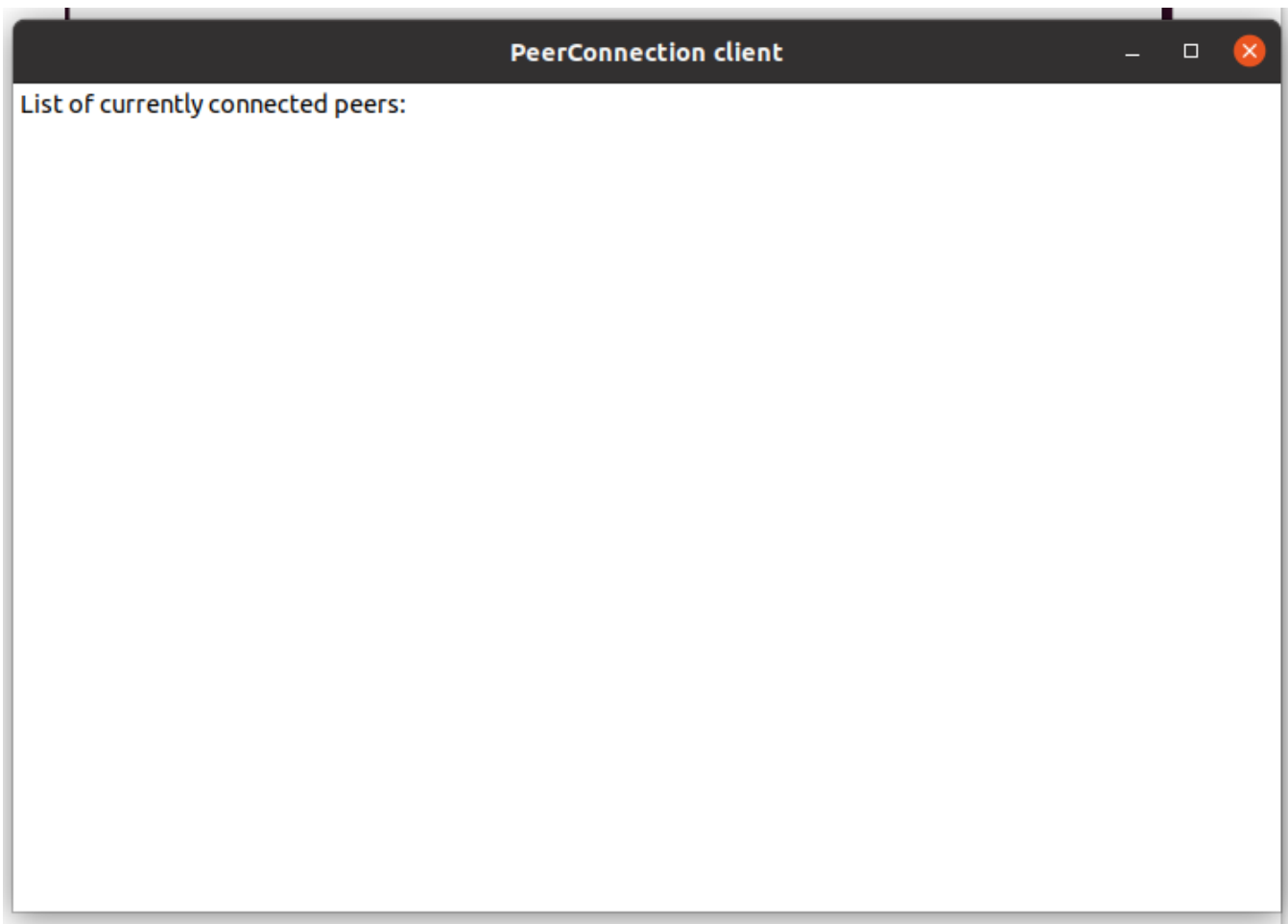
Refer the below peerconnection_client GTK window.



STEP 3:

Provide IP address of server pc and press connect button.

Refer the below peerconnection_client GTK window.



STEP 4:

Stop below service before running the peerconnection_client binary.

```
systemctl stop rms-launcher  
systemctl stop mst-launcher
```

Run the peerconnection_client binary into RPI board side and Enter server PC IP address, port number.

peerconnection_client

Console output

```
root@raspberrypi3-rdk-camera:~# peerconnection_client  
Inside constructor CustomSocketServer().....  
Initialized thread...  
"Registering PeerConnectionClient::RegisterObserver()"  
Enter server IP Address  
192.168.0.126  
Enter port number  
8888
```

STEP 5:

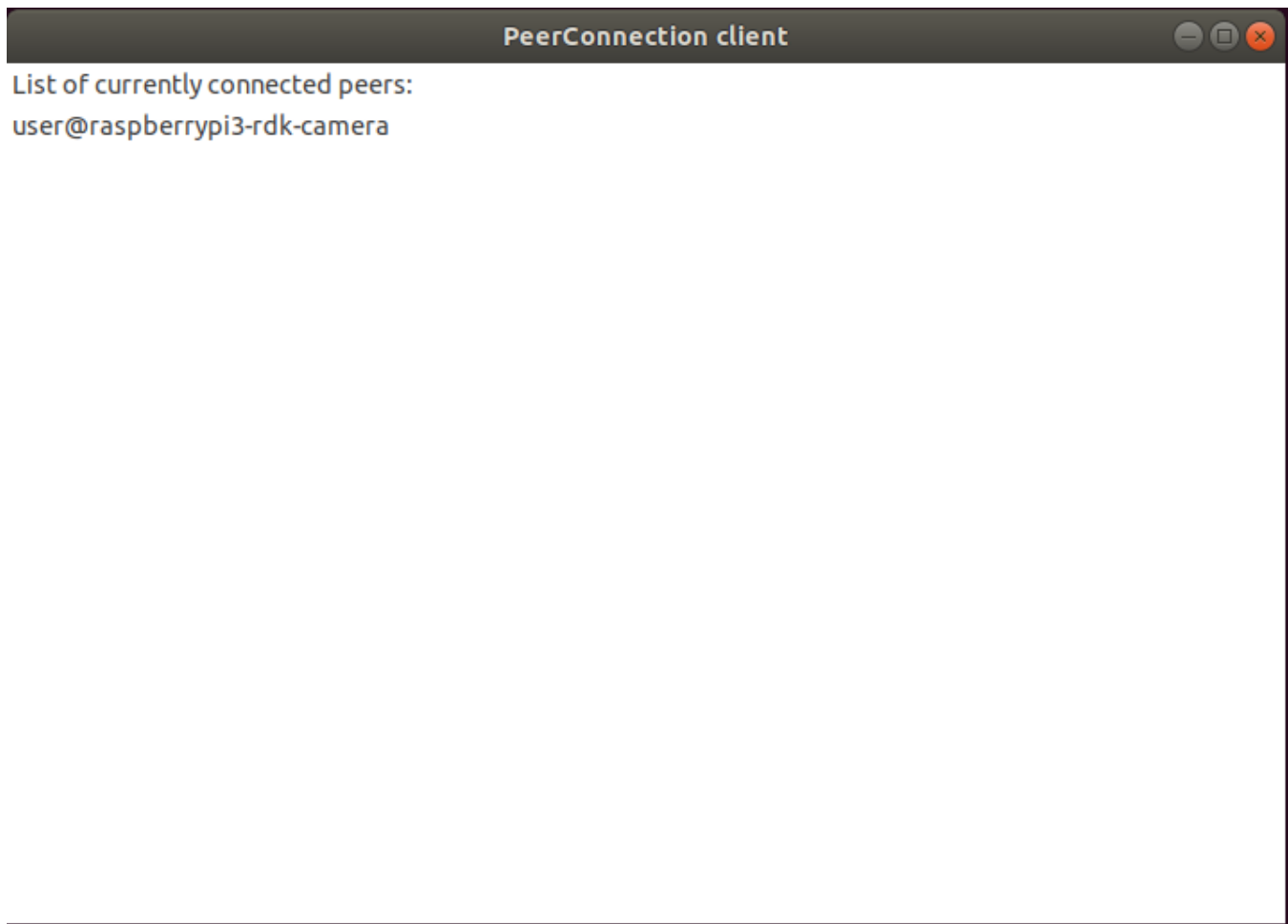
Need to Check whether the peerconnection_server side both clients are connected are not.

Console output

```
user@BLTSLRM110:~/Desktop/openwebrtc-m72/out/Default$ ./peerconnection_server
Server listening on port 8888
New connection...
New member added (total=1): zaid@zaid-ahmad
Disconnecting socket
Total connected: 1
New connection...
New connection...
New member added (total=2): user@raspberrypi3-rdk-camera
Disconnecting socket
Total connected: 2
New connection...
Disconnecting socket
Total connected: 2
New connection...
```

STEP 6:

PC side peerconnection_client GTK window shows the number of peers client devices are connected list.



STEP 7:

Double click [user@raspberrypi3-rdk-camera](#) from the listed peer connected devices.

