RMS feature validation with RTSP streaming

Introduction

The RDKC Media Server is much more than a multi-format, multi-protocol server that delivers your media rich content across multiple screens and platforms. The RDK camera software runs on RPi-3 device. With this RTSP streaming we able to play live streaming content in VLC player. This page dedicated to bringing up and validation of RMS functionality with RTSP streaming in RPI-3.

Yocto Build Steps

Refer below link to build camera image

RDK-C Build Instruction for RPI-3 with Dunfell branch

Image Flash Procedure

Image Flash step

```
$ bzcat "Image Name" | sudo dd of="Device Name" bs=4M iflag=fullblock oflag=direct conv=fsync

Example:
bzcat rdk-generic-camera-image_default_20201127105606.rootfs.wic.bz2 | sudo dd of=/dev/sdc bs=4M
iflag=fullblock oflag=direct conv=fsync
```

Validation Procedure of RMS with RTSP streaming

STEP 1:

Add require SSID and PSK in /etc/wpa_supplicant.conf file in below format

network={

ssid="username"

psk="password"

}

Console output

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
ssid="XXXX"
psk="YYYYYYYYY"
}
```

STEP 2:

Reboot the Target

After Reboot don't do step 1 and 2.

Note : Step 1 & 2 is only applicable for fresh target boot-up with new image.

STEP 3:

Modify needed resolution in rms configuration file

Supported Resolution:

SD:

Width - 640 , Height - 480

Width - 720 , Height - 576

HD:

Width - 1280 , Height - 720

FULL HD :

Width - 1920 , Height - 1080

Modify resolution in below configuration file

cd /usr/local/rms/bin

vi rms.conf

Console output

```
RRSIP=XXX.XXX.XXX.XXX
RRSPORT=81
ROOMID=rpi0
RRSSSL=0
WIDTH=1280
HEIGHT=72
```

After resolution modification need to reboot the target.

Note: This step is not necessary,it is depends on your resolution validation.

STEP 4:

WiFi connection is must needed for RMS validation.

Check WiFi connection by using below command.

ifconfig

Console output

```
root@raspberrypi3-rdk-camera:~# ifconfig
eth0      Link encap:Ethernet  HWaddr B8:27:EB:87:67:91
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:89842 errors:0 dropped:0 overruns:0 frame:0
          TX packets:89842 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25639748 (24.4 MiB)  TX bytes:25639748 (24.4 MiB)

wlan0     Link encap:Ethernet  HWaddr B8:27:EB:D2:32:C4
          inet addr:192.168.43.146  Bcast:192.168.43.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:44 errors:0 dropped:0 overruns:0 frame:0
          TX packets:103 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4484 (4.3 KiB)  TX bytes:10216 (9.9 KiB)
```

STEP 5:

check loaded module by using below command

lsmod

Console output

```
root@raspberrypi3-rdk-camera:~# lsmod
Module                  Size  Used by
bcm2835_v4l2           40563  0
v4l2_common             4809  1 bcm2835_v4l2
videobuf2_vmalloc       6264  1 bcm2835_v4l2
videobuf2_memops        1528  1 videobuf2_vmalloc
videobuf2_v4l2         12640  1 bcm2835_v4l2
videobuf2_core         27389  2 bcm2835_v4l2,videobuf2_v4l2
videodev              154457  4 v4l2_common,videobuf2_core,bcm2835_v4l2,videobuf2_v4l2
media                  23307  1 videodev
brcmfmac              258239  0
brcmutil                7590  1 brcmfmac
snd_bcm2835            21405  0
cfg80211              492836  1 brcmfmac
snd_pcm                79872  1 snd_bcm2835
rfkill                 19936  3 cfg80211
snd_timer              20294  1 snd_pcm
snd                    52949  3 snd_timer,snd_bcm2835,snd_pcm
lirc_rpi                6840  0
lirc_dev                7533  1 lirc_rpi
uio_pdrv_genirq         3469  0
uio                     8703  1 uio_pdrv_genirq
fixed                   2876  0
sch_fq_codel            9662  2
ipv6                  384101  18
```

STEP 6:

check camera device there or not by using below command

ls /dev/video0

Console output

```
root@raspberrypi0-rdk-camera:~# ls /dev/video0
/dev/video0
```

STEP 7:

Check mediastreamer and rms binaries are running or not before RTSP streaming validation.

ps -Af | grep media

Console output

```
root@raspberrypi3-rdk-camera:~# ps -Af | grep media
root       197     1 13 10:57 ?        00:04:40 mediastreamer
root      1007     1  3 10:57 ?        00:01:05 ./rdkcmediaserver ../config/config.lua
root     10959  1943  0 11:32 ttyS0    00:00:00 grep media
```

STEP 8:

Enter into the Telnet console with telnet command

telnet localhost 1222

Console output

```
root@raspberrypi3-rdk-camera:~# telnet localhost 1222
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
```

After entering the telnet console need to stop webrtc streaming, so we should check webrtc status with "listconfig" command.

Console output

```
listconfig
Command entered successfully!
Run-time configuration

    dash: []
    hds: []
    hls: []
    metalistener: []
    mss: []
    process: []
    pull:
      --
        configId: 1
        localStreamName: stream2
        status:
          current:
            description: Streaming
            uniqueStreamId: 1
        uri: sercom://0
    push: []
    record: []
    webrtc:
      --
        configId: 2
        roomid: rpi0
        rrsip: 18.224.54.11
        rrsport: 81
```

If the webrtc detail is available in listconfig, we need to stop webrtc with "stopwebrtc" command

Console output

```
stopwebrtc
Command entered successfully!
Stopped WebRTC Negotiation Service
```

After did the stopwebrtc, check the listconfig

Console output

```
listconfig
Command entered successfully!
Run-time configuration

    dash: []
    hds: []
    hls: []
    metalistener: []
    mss: []
    process: []
    pull:
      --
        configId: 1
        localStreamName: stream2
        status:
          current:
            description: Streaming
            uniqueStreamId: 1
        uri: sercom://0
    push: []
    record: []
    webrtc: []
```

STEP 9:

At that same telnet console,need to give the below command for RTSP streaming

pushStream uri=rtsp://camera_ip:5544 localStreamName=stream2

Example:

pushStream uri=rtsp://192.168.43.146:5544 localStreamName=stream2

Console output

```
pushStream uri=rtsp://192.168.43.146:5544 localStreamName=stream2
Command entered successfully!
Local stream stream2 enqueued for pushing to rtsp://192.168.43.146:5544 as stream2

    configId: 4
    forceTcp: false
    keepAlive: true
    localStreamName: stream2
    targetStreamName: stream2
    targetStreamType: live
    targetUri:
      fullUri: rtsp://192.168.43.146:5544
      port: 5544
      scheme: rtsp
```

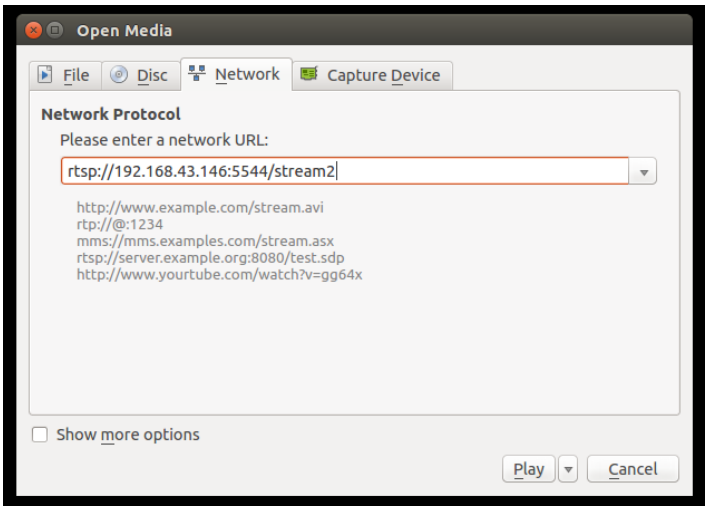
STEP 10:

On VLC player, for RTSP streaming

need to enter media Open Network Stream option and then give rtsp URL to play streaming content in VLC

rtsp://camera_ip:5544/stream2

Example :rtsp://192.168.43.146:5544/stream2



We can able to see the live streaming content on VLC Player.

(OR)

On PC, Able to play the camera live streaming content with the below gstreamer pipeline,before trying this pipeline we should install gstreamer specific packages.

gst-launch-1.0 rtspsrc location=rtsp://camera_ip:5544/stream2 ! rtph264depay name=demux ! h264parse ! avdec_h264 ! videoconvert ! autovideosink

Example:

gst-launch-1.0 rtspsrc location=rtsp://192.168.43.146:5544/stream2 ! rtph264depay name=demux ! h264parse ! avdec_h264 ! videoconvert ! autovideosink