

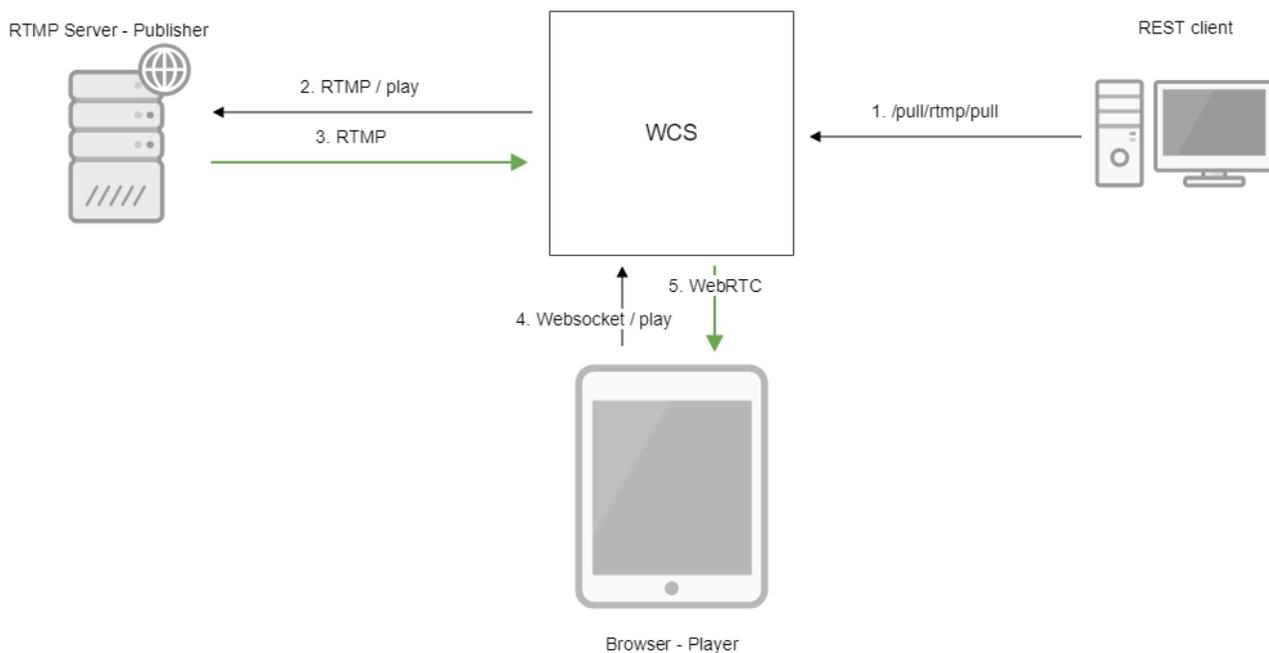
# From another server via RTMP

- [Overview](#)
  - [Operation flowchart](#)
- [REST queries](#)
  - [REST-methods and response statuses](#)
  - [Parameters](#)
- [Configuration](#)
- [Quick manual on testing](#)
  - [Capturing of an RTMP stream broadcast by another server using the REST-query /pull/rtmp/pull](#)
  - [Capturing of an RTMP stream broadcast by another server without using REST queries](#)
- [Known issues](#)

## Overview

WCS can capture an RTMP video stream published by another server, by request. The captured stream can be broadcast to [any of supported platforms](#) using [any of supported technologies](#). Managing of RTMP stream capturing is performed using REST API.

## Operation flowchart



1. The `/pull/rtmp/pull` REST query is sent to the WCS server
2. The WCS server requests the RTMP stream from the specified server
3. The RTMP stream is broadcast to the WCS server
4. The browser requests playing the captured stream via WebSocket
5. The browser receives the stream via WebRTC

## REST queries

A REST-query must be an HTTP/HTTPS POST request as follows:

- - HTTP:`http://test.flashphoner.com:8081/rest-api/pull/rtmp/pull`
- - HTTPS:`https://test.flashphoner.com:8444/rest-api/pull/rtmp/pull`

Where:

- -test.flashphoner.com- is the address of the WCS server
- - 8081- is the standard REST / HTTP port of the WCS server
- -8444- is the standard HTTPS port
- -rest-api- is the required part of the URL
- - /pull/rtmp/pull- is the REST method used

## REST-methods and response statuses

| REST-method         | Example of REST-query  | Example of REST response   | Response status                                 | Description                                 |
|---------------------|--|--|---|---|
| /pull/pull          | <pre>{   "uri": "rtmp://myserver. com/live/myStream",   "record": "true" }</pre> |  | 409 - Conflict<br>500 - Internal error          | Pull the WebRTC stream at the specified URL |
| /pull<br>/find_all  |  | <pre>{   "localMediaSessionId": "5a072377- 73c1-4caf-abd3",   "remoteMediaSessionId": null,   "localStreamName":   "rtmp://myserver.com/live /myStream",   "remoteStreamName": null,   "uri": "rtmp://myserver.com/live /myStream",   "status": "PROCESSED_REMOTE" }</pre> | 200 - streams are found<br>500 - Internal error | Find all pulled WebRTC streams              |
| /pull<br>/terminate | <pre>{   "uri": "rtmp://myserver. com/live/myStream" }</pre>                     |  | 200 - stream terminated<br>500 - Internal error | Terminate the pulled WebRTC stream          |

## Parameters

| Parameter name       | Description  | Example                         |
|----------------------|--|---------------------------------|
| uri                  | URL of the WebRTC stream   | wss://demo.flashphoner.com:8443 |
| localMediaSessionId  | Session identifier   | 5a072377-73c1-4caf-abd3         |
| remoteMediaSessionId | Session identifier on the remote server  | 12345678-abcd-dead-beaf         |
| localStreamName      | Local name assigned to the captured stream. By this name the stream can be requested from the WCS server | testStream                      |
| remoteStreamName     | Captured stream name on the remote server  | testStream                      |
| status               | Current stream status  | NEW                             |

## Configuration

In the /usr/local/FlashphonerWebCallServer/conf directory you can find the SDP description file for the RTMP agent rtmp\_agent.sdp:

```
o=- 1988962254 1988962254 IN IP4 0.0.0.0
c=IN IP4 0.0.0.0
t=0 0
a=sdplang:en
m=video 0 RTP/AVP 95
a=rtpmap:95 H264/90000
a=fmtp:95 profile-level-id=42e01f;packetization-mode=1
a=sendonly
m=audio 0 RTP/AVP 103 96 97 98 99 100 102 108 104
a=rtpmap:108 mpeg4-generic/48000/1
a=rtpmap:96 mpeg4-generic/8000/1
a=rtpmap:97 mpeg4-generic/11025/1
a=rtpmap:98 mpeg4-generic/12000/1
a=rtpmap:99 mpeg4-generic/16000/1
a=rtpmap:100 mpeg4-generic/22050/1
a=rtpmap:104 mpeg4-generic/24000/1
a=rtpmap:102 mpeg4-generic/32000/1
a=rtpmap:103 mpeg4-generic/44100/1
a=recvonly
```

To enable recording of both audio and video (instead of audio only) during captured stream recording specify the following attribute in this file

```
a=sendonly
```

for video.

## Quick manual on testing

### Capturing of an RTMP stream broadcast by another server using the REST-query /pull/rtmp/pull

1. For the test we use:

- the demo server at [demo.flashphoner.com](http://demo.flashphoner.com);
- the Chrome browser and the [REST-client](#) to send queries to the server;
- the [Two Way Streaming](#) web application to play the captured stream in a browser.

2. Open the REST client. Send the /pull/rtmp/pull query and specify the URL of the RTMP stream in parameters:

Method POST Request URL http://p11.flashphoner.com:9091/rest-api/pull/rtmp/pull SEND

Parameters ^

Headers Body Variables

Body content type application/json Editor view Raw input

FORMAT JSON MINIFY JSON

```
{
  "uri": "rtmp://str81.creacast.com/grandlilletv/low"
}
```

200 OK 61.20 ms DETAILS

3. Make sure the stream is captured by the server. To do this, send the /rtmp/pull/find\_all request:

Method: POST  
Request URL: http://p11.flashphoner.com:9091/rest-api/pull/rtmp/find\_all

Parameters: Headers, Body, Variables

Body content type: application/json  
Editor view: Raw input

FORMAT JSON MINIFY JSON

200 OK 61.20 ms

DETAILS

and copy the local name of the stream from the localStreamName response parameter:

```
[Array[1]  
  -0: {  
    "localMediaSessionId": "f087b936-9a61-47f7-b11c-7ff1dd1405f5",  
    "remoteMediaSessionId": null,  
    "localStreamName": "rtmp://str81.creacast.com/grandlilletv/low",  
    "remoteStreamName": null,  
    "uri": "rtmp://str81.creacast.com/grandlilletv/low",  
    "status": "PROCESSED_REMOTE"  
  }  
],
```

4. Open the page of the Two Way Streaming web application. Click "Connect" and specify the local stream name, then click "Play":

# Two-way Streaming

Local



29b8

Publish

Player



rtmp://

Stop

Available

PLAYING

wss://p11.flashphoner.com:8443

Disconnect

ESTABLISHED

## 5. WebRTC internals diagrams in a browser:

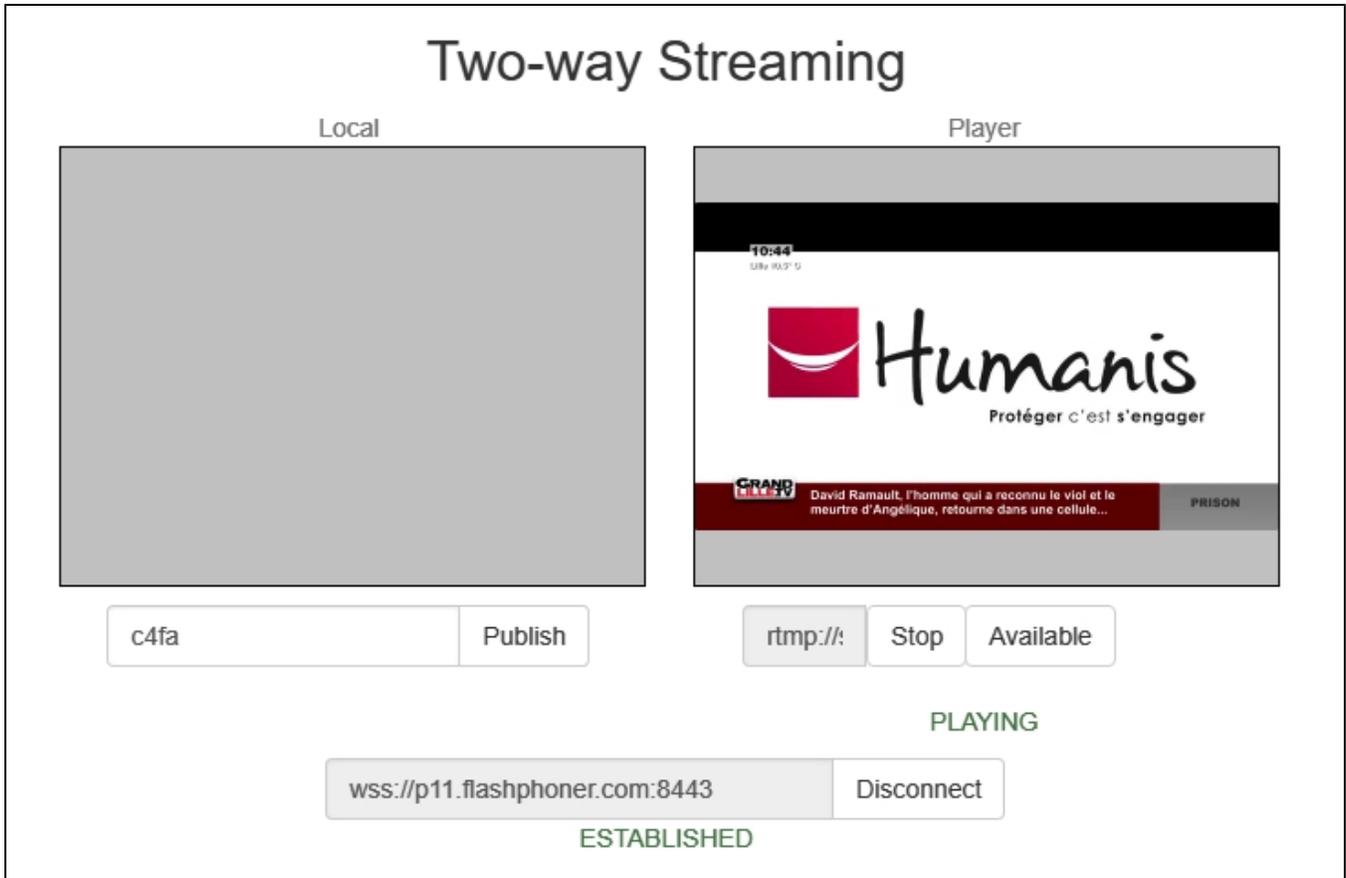


# Capturing of an RTMP stream broadcast by another server without using REST queries

1. For the test we use:

- the demo server at [demo.flashphoner.com](http://demo.flashphoner.com);
- the web application, [Two Way Streaming](#), to capture and play the captured stream in a browser.

2. Open the page of the Two Way Streaming web application. Click "Connect" and specify the name of the RTMP stream you want to capture, then click "Play":

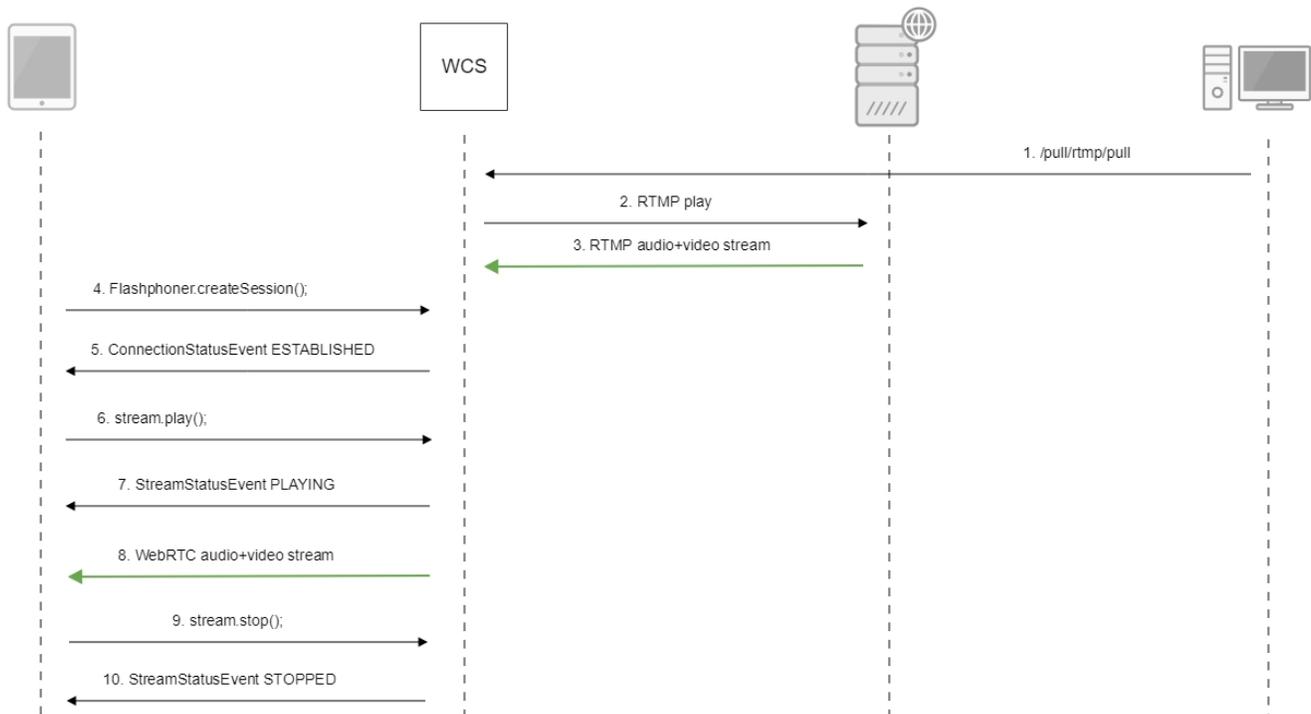


3. WebRTC internals diagrams in a browser:



Call flow

Below is the call flow when capturing an RTMP stream from another server



Known issues

1. A stream containing B-frames does not play or plays with artifacts (latencies, lags)

Symptoms:

- a stream sent by the RTMP encoder does not play or plays with latencies or lags
- warnings in the [client log](#):

```
09:32:31,238 WARN 4BitstreamNormalizer - RTMP-pool-10-thread-5 It is B-frame!
```

Solution: change the encoder settings so, that B-frames were not used (lower encoding profile, specify in the command line etc).

2. AAC frames of type 0 are not supported by decoder and will be ignored while stream pulled playback

In this case, warnings will be displayed in the [client log](#):

```
10:13:06,815 WARN AAC - AudioProcessor-c6c22de8-a129-43b2-bf67-1f433a814ba9 Dropping AAC frame that starts with 0, 119056e500
```

Solution: use Fraunhofer AAC codec with the following parameter in [flashphoner.properties](#) file

```
use_fdk_aac=true
```

3. When publishing and then playing and recording H264 + AAC stream video may be out of sync with sound, or no sound at all.

Symptoms: when playing H264 + AAC stream published on server, and when recordingsuch stream, sound is out of sync with video or absent

Solution:

a) set the following parameter in [flashphoner.properties](#) file

```
disable_drop_aac_frame=true
```

This parameter also turns off AAC frames dropping.

b) use Fraunhofer AAC codec

```
use_fdk_aac=true
```

4. Sound may be distorted or absent when resampled to 11025 Hz

Symptoms: when H264 + AAC stream published on WCS server is played with AAC sample rate 11025 Hz, sound is distorted or absent

Solution: do not use 11025 Hz sample rate, or escape AAC sound resampling to this rate, for example, do not set this sample rate in [SDP settings](#).

5. Some RTMP functions does not supported and will be ignored:

- FCSubscribe
- FCPublish
- FCUnpublish
- onStatus
- onUpstreamBase
- releaseStream

6. When recording the captured stream to the disk, only audio is recorded

Symptoms: when the "record": "true" parameter is set in the /pull/rtmp/pull REST query, the received file lacks video track, only audio is present.

Solution: in the [SDP settings](#) set the

```
a=sendonly
```

attribute for the video track.