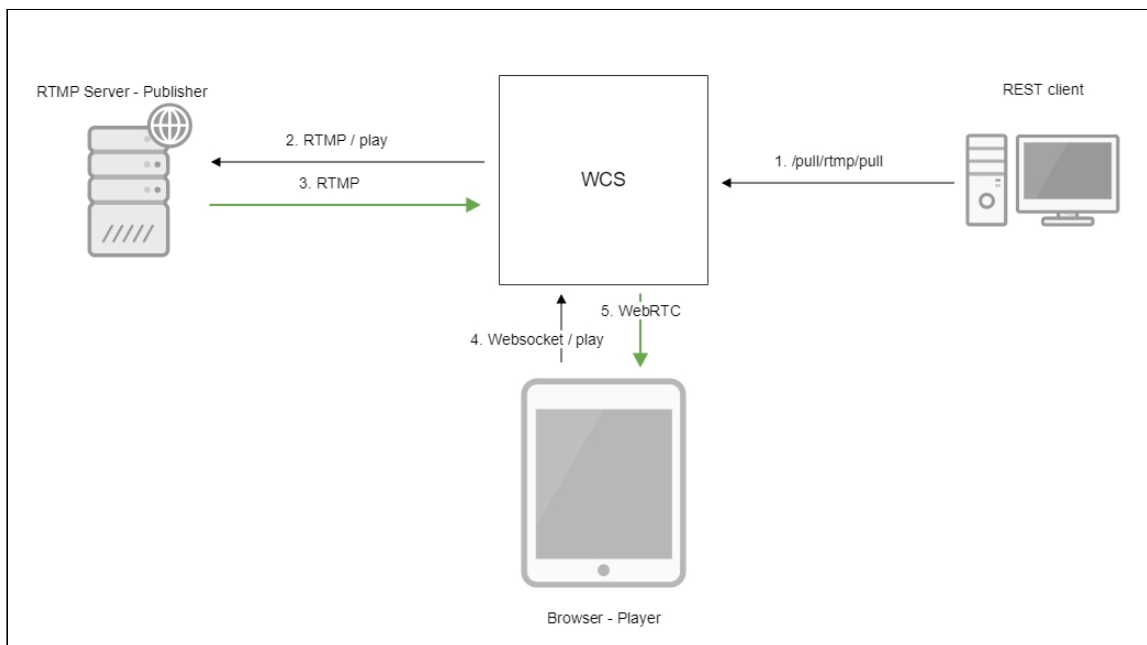


# From other server via RTMP

## Overview

WCS can capture on demand an RTMP video stream published by another server. The captured stream can be played on [any of supported platforms](#) using any of supported technologies. RTMP stream capturing is managed 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 API

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 queries and responses

REST method	Request body	Response body	Response status	Description
<code>/pull/rtmp/pull</code>	<pre>{    "uri": "rtmp://myserver.com/live/myStream",    "localStreamName": "stream1",    "record":     :     "false",    "hasAudio":     true,    "hasVideo": true }</pre>		409 Conflict 500 Internal error	Pull the RTMP stream at the specified URL

REST method	Request body	Response body	Response status	Description
<code>/pull/rtmp/findAll</code>		<pre>[   {     "localMediaSessionId":     "5a072377-73c1-4caf-abd3"      "localStreamName":     "stream1"      "uri":     "rtmp://myserver.com/live/myStream",      "status":     "PROCESSED_REMOTE",      "hasAudio":     true,      "hasVideo":     true,      "record": false   } ]</pre>	200 OK 404 Not found 500 Internal error	Find all pulled RTMP streams

REST method	Request body	Response body	Response status	Description
<code>/pull/rtmp/terminate</code>	<pre>{   "uri": "rtmp://myserver.com/live/myStream" }</pre>		200 OK 404 Not found 500 Internal error	Terminate the pulled RTMP stream

## Parameters

Parameter	Description	Example
uri	URL of the RTMP stream	<code>rtmp://myserver.com/live/myStream</code>
localMediaSessionId	Session identifier	<code>5a072377-73c1-4caf-abd3</code>
localStreamName	Local name assigned to the captured stream. By this name the stream can be requested from the WCS server	<code>stream1</code>
status	Current stream status	<code>PROCESSED_REMOTE</code>
record	Is the pulled stream recording	<code>false</code>
hasAudio	Stream has audio track	<code>true</code>
hasVideo	Stream has video track	<code>true</code>

## Pulled stream publishing with a given name

Since build [5.2.724](#) it is possible to set the name to publish stream on server using `localStreamName` query parameter. If the parameter is not set, the stream name will be set to `uri`, as done in previous builds.

## RTMP stream repeatedly capturing with the same URI

`/pull/rtmp/pull` query returns `409 Conflict` when trying to repeatedly capture RTMP stream with the same URI. If the stream is already published on the server, it is necessary to subscribe to it.

## Configuration

### Codecs parameters definition

The SDP description file for the RTMP agent `rtmp_agent.sdp` placed to `/usr/local/FlashphonerWebCallServer/conf` folder may be used to change audio codec parameters:

```
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.

### Publishing a stream without audio or video

If audio or video description is removed from `rtmp_agent.sdp`, an RTMP stream captured will be published on WCS without audio or video respectively. For example, use the following SDP to publish video only:

```
v=0
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
```

## Frame type detection

Sometimes, RTMP stream source may set a media frame type incorrectly at RTMP protocol level. In this case frame type should be detected according to frame content. To enable this, use the following parameter added in build [5.2.1446](#)

```
rtmp_detect_h264_frame_type=true
```

## 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:
2. the demo server at `demo.flashphoner.com`;
3. the Chrome browser and the [REST-client](#) to send queries to the server;
4. the [Two Way Streaming](#) web application to play the captured stream in a browser.
5. 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

6. 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](http://p11.flashphoner.com:9091/rest-api/pull/rtmp/find_all) SEND

Parameters ^

Headers Body Variables

Body content type application/json Editor view Raw input

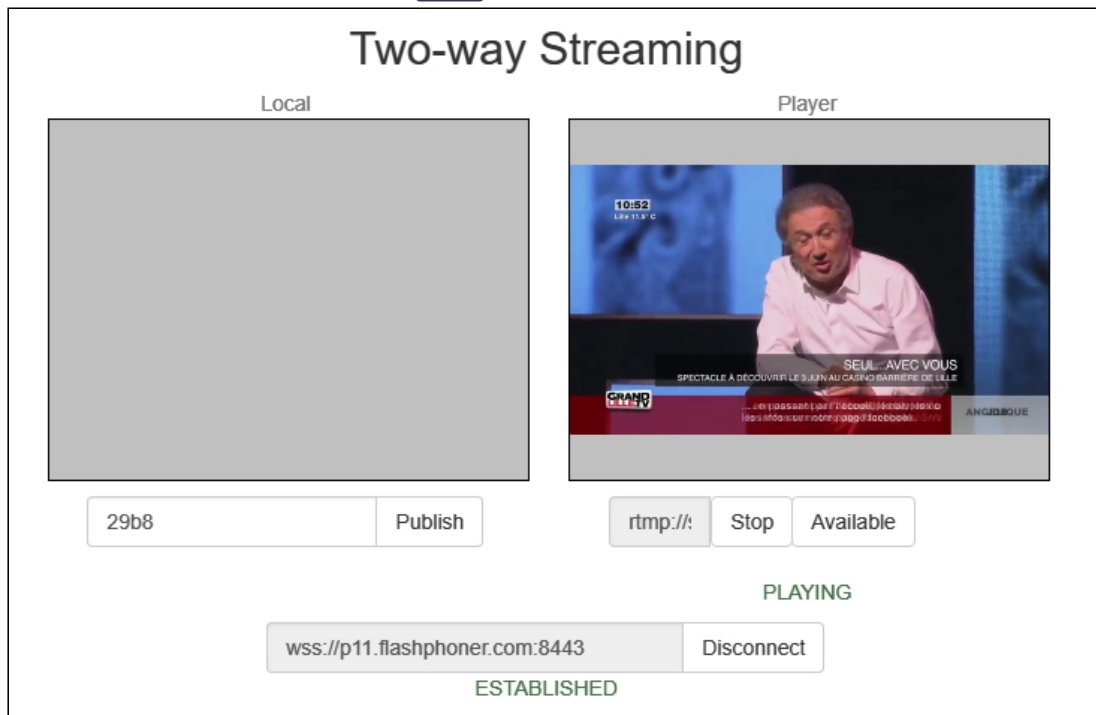
FORMAT JSON MINIFY JSON

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"
  }
],
```

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



8. WebRTC internals diagrams in a browser:

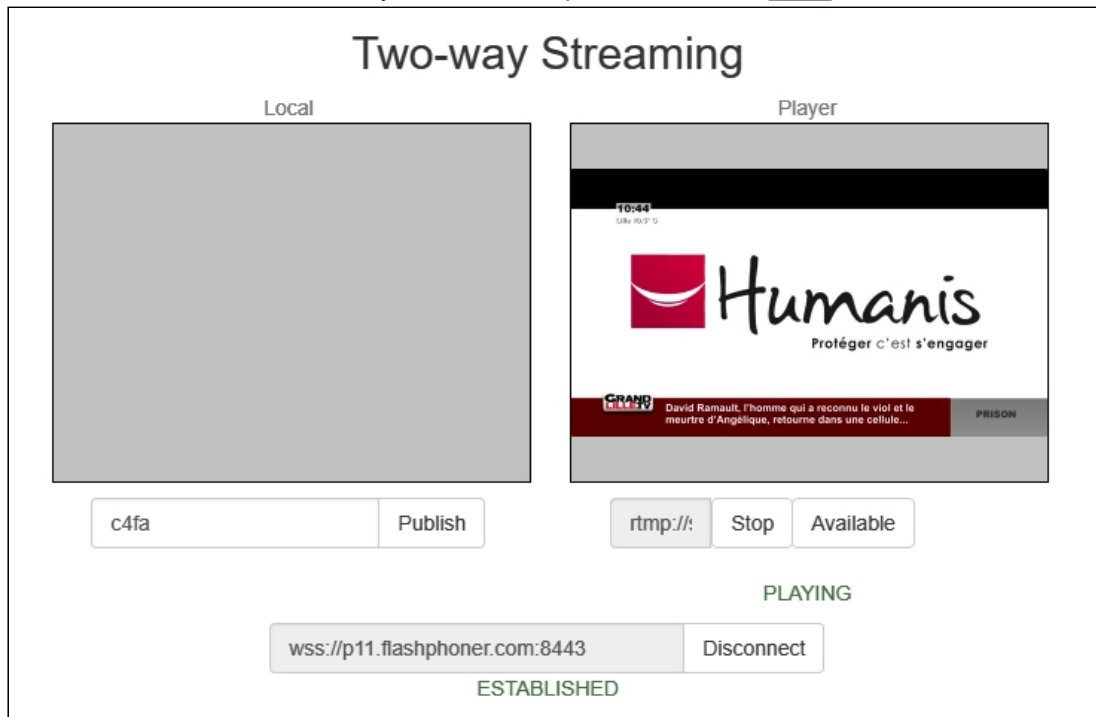


## Capturing of an RTMP stream broadcast by another server without using REST API

1. For the test we use:
2. the demo server at [demo.flashphoner.com](https://demo.flashphoner.com):



- the web application, [Two Way Streaming](#), to capture and play the captured stream in a browser.
- 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**:

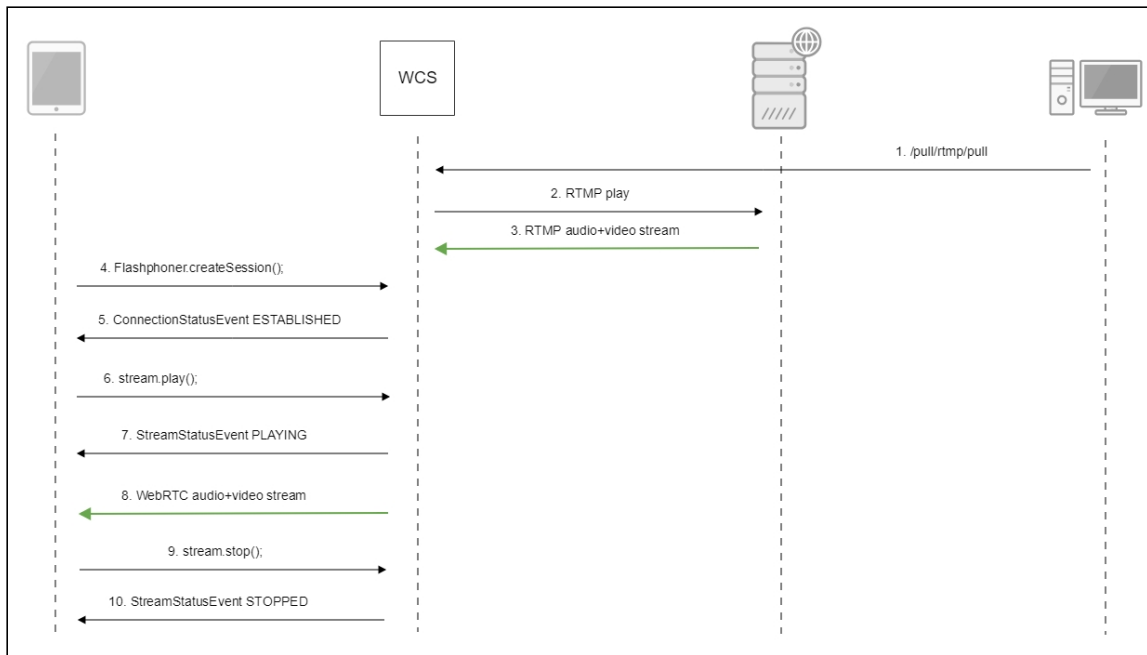


- WebRTC internals diagrams in a browser:



## Call flow

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



## Authentication on a source server

WCS supports Adobe authentication on RTMP server while capturing a stream from it using RTMP URL parameters:

```
rtmp://username:password@server:1935/live/streamKey
```

Note that if an RTMP stream is requested from another WCS server, this kind of authentication is not supported.

Since build [5.2.1069](#) it is possible to pass authentication parameters after stream name

```
rtmp://server:1935/live/streamKey?user=username&password=password
```

In this case the parameters will be passed to RTMP server in `connect` message.

The parameters can be set after application name too

```
rtmp://server:1935/live?user=username&password=password/streamKey
```

In this case the parameters will also be passed to RTMP server in `connect` message. If the stream is requested from another WCS server via RTMP, authentication parameters will be available in REST hook `/connect`.

## 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)
- [transcode](#) the stream, in this case there will be no B-frames in transcoded stream

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



### Symptoms

There are warnings 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

Enable Fraunhofer AAC codec with the following parameter

```
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 recording such 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 following attribute

```
a=sendonly
```

for the video track.

## 7. Freezes are possible when RTMP stream is captured from Adobe Media Server with Aggregate messages enabled



### Symptoms

Freezes may occur, publishing may fail by timeout when capturing RTMP stream from Adobe Media Server



### Solution

Disable Aggregate messages in AMS configuration by switching off the following parameters:

- all occurrences of `EnableAggMsgs` in `Server.xml`
- `Client/AggregateMessages` and `Queue/AggregateMessages` in `conf/_defaultRoot/_defaultVHost/Application.xml`
- `AggregateMessages` in `Vhost.xml`