In a player via RTMP

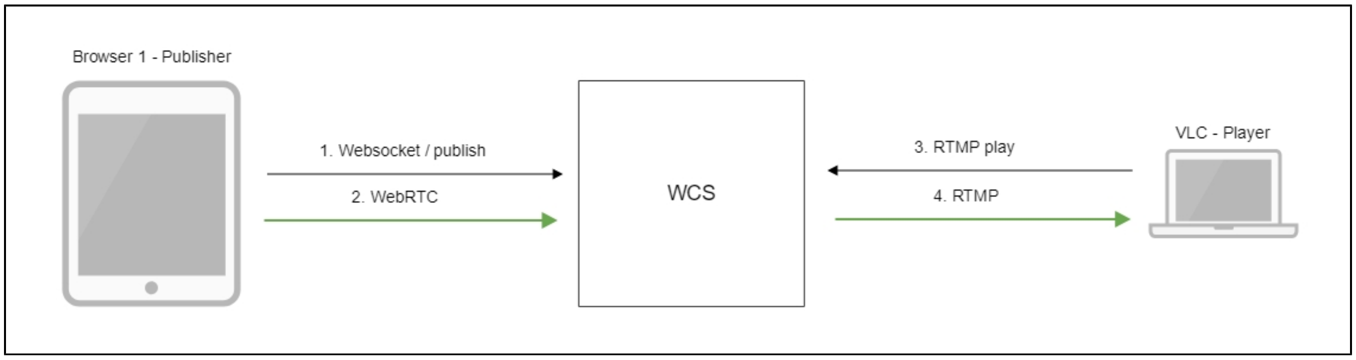
Overview

A stream published on the WCS server can be played via RTMP in a third-party player. In this case, WCS itself serves as anRTMP-source.

RTMP-codecs

- Video: H.264
- Audio: AAC, G.711, Speex

Operation flowchart



1. The browser establishes a connection to the server via Websocket
2. The browser captures the camera and the microphone and sends the WebRTC stream to the server
3. VLC Player establishes a connection to the server via RTMP
4. VLC Player receives the stream from the server and plays it

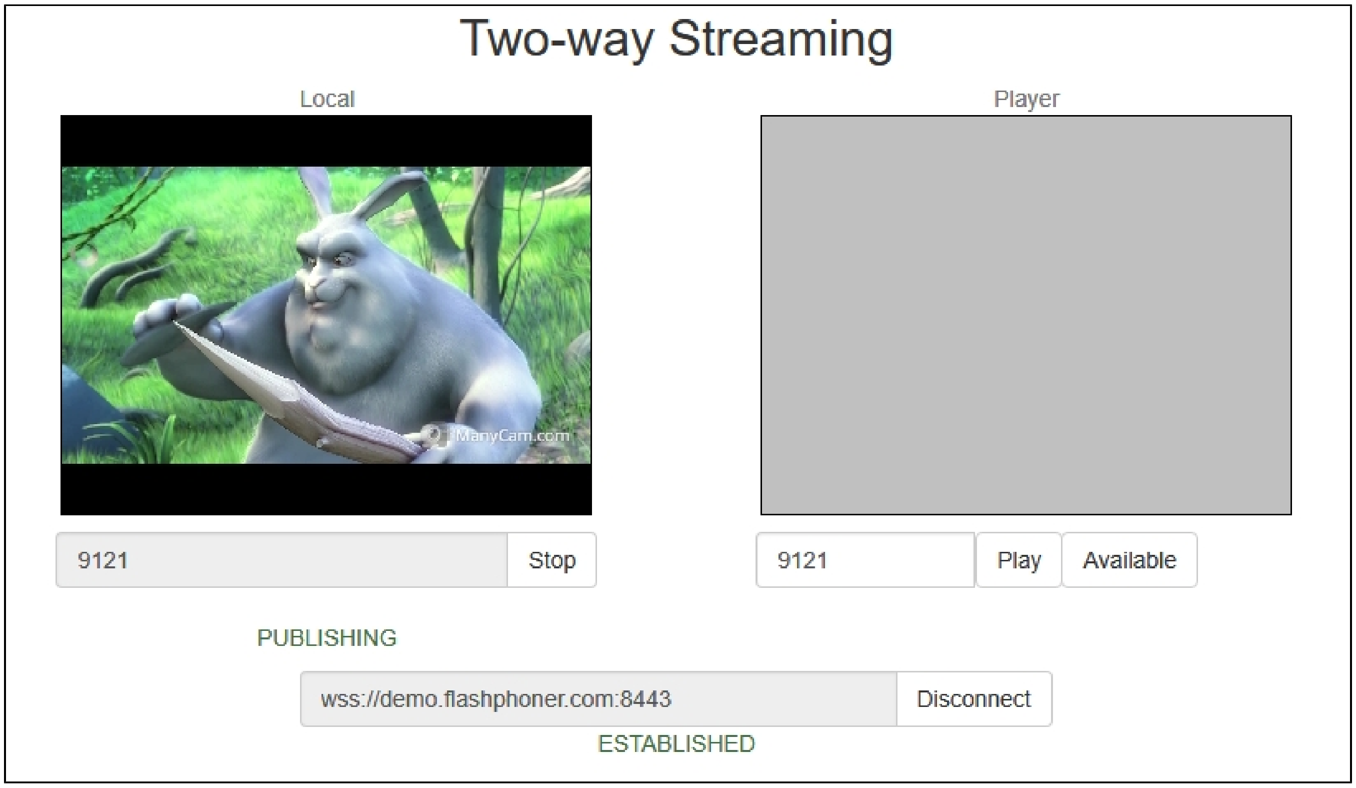
Quick manual on testing

Publishing a video stream on the server and playing it via RTMP in a software player

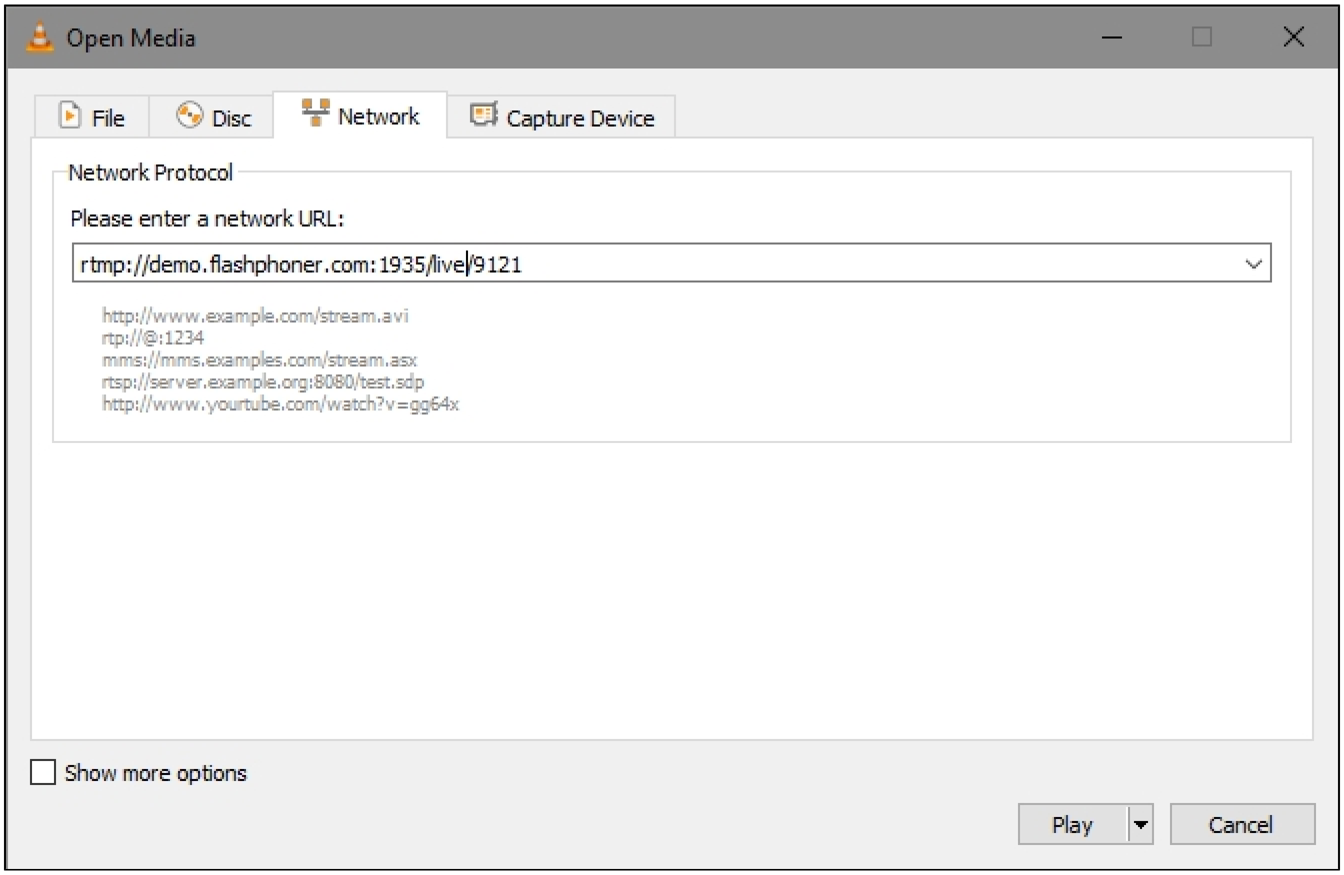
1. For the test we use:

- the demo server at demo.flashphoner.com;
- theTwo Way Streamingweb application to publish the stream;
- VLC Player to play the stream.

2. Open the Two Way Streaming application. Click Connect, then Publish. Copy the identifier of the stream:

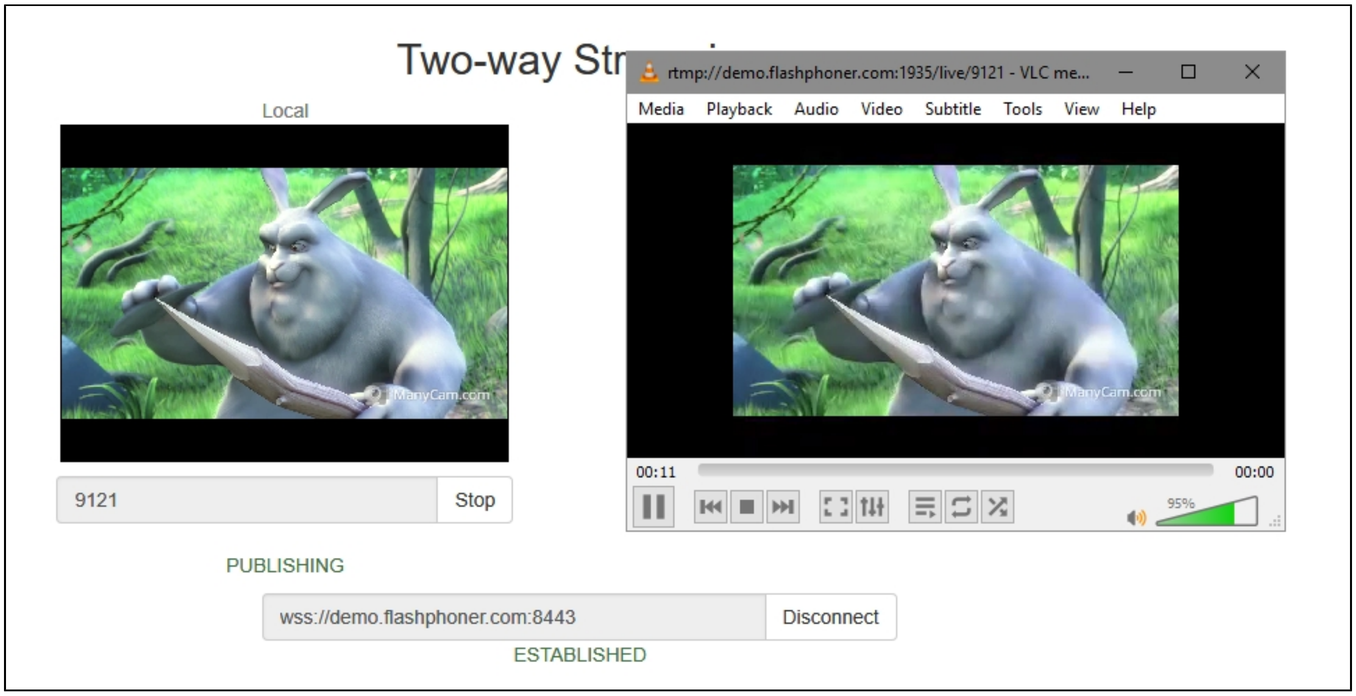


3. Run VLC, select the "Media - Open network stream" menu. Enter the URL of the WCS server and enter the identifier of the stream, in this exampe: rtmp://demo.flashphoner.com:1935/live/9121:


Open Media
File
Disc
Network
Capture Device
Network Protocol
Please enter a network URL:
http://www.example.com/stream.avi
rtp://@:1234
mms://mms.examples.com/stream.asx
rtsp://server.example.org:8080/test.sdp
http://www.yourtube.com/watch?v=gg64x
Show more options
Play
Cancel

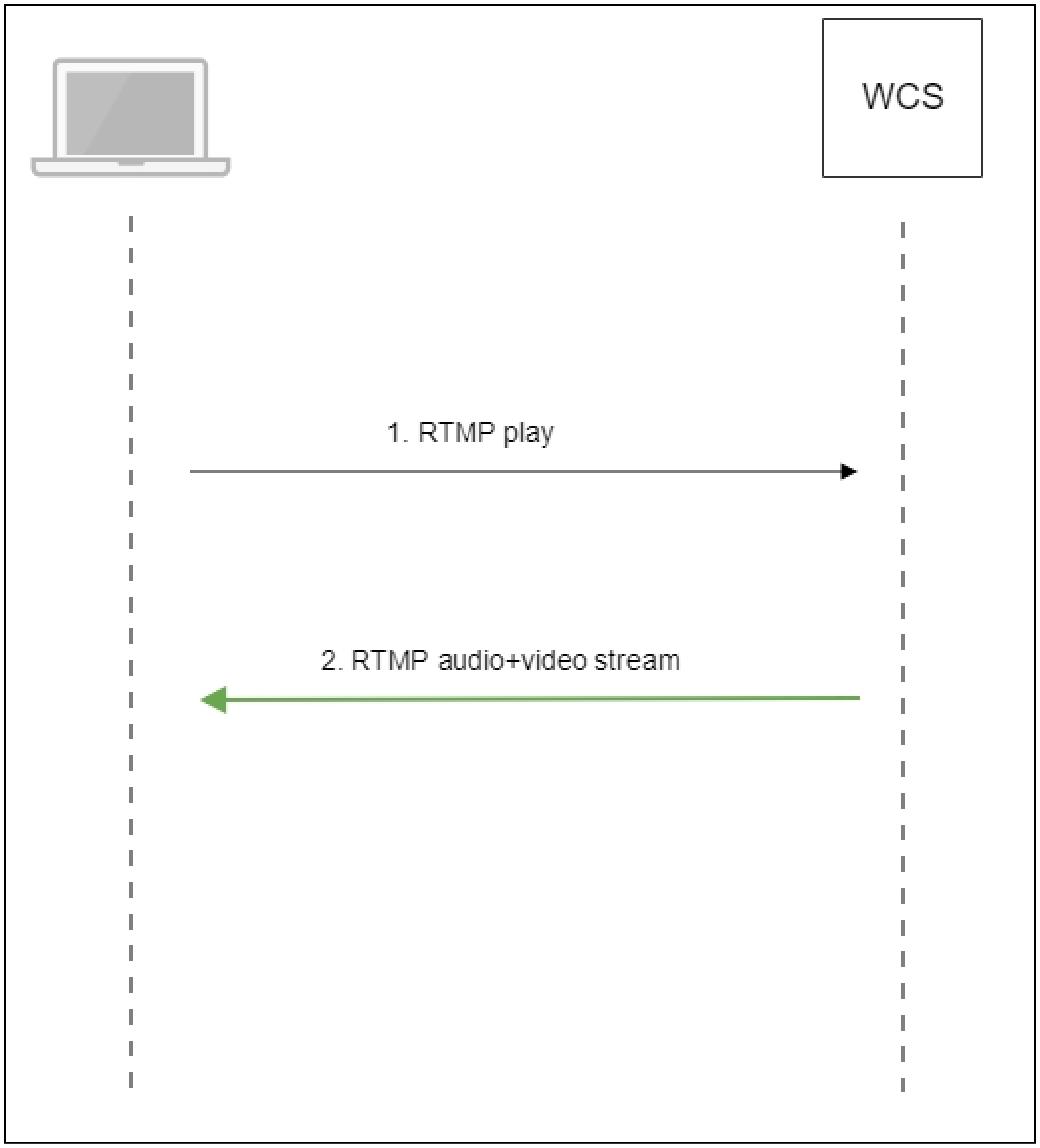
rtmp://demo.flashphoner.com:1935/live/9121

4. Click the "Play" button. The player starts playing the stream:


Two-way Str
rtmp://demo.flashphoner.com:1935/live/9121 - VLC me...
Media Playback Audio Video Subtitle Tools View Help
Local
00:11
00:00
95%
9121
Stop
PUBLISHING
wss://demo.flashphoner.com:8443
Disconnect
ESTABLISHED

Call flow

Below is the call flow when playing a stream via RTMP in a software player.



1. The software player establishes a connection to the WCS server via RTMP.
2. The software player receives the media stream from WCS.

Parsing stream URL parameters

When RTMP stream is published or played on WCS, RTMP connection and stream parameters may be set in stream URL like this:

```
rtmp://host:1935/live?connectParam1=val1&connectParam2=val2/streamName?streamParam1=val1&streamParam2=val2
```

Where

- host is WCS server hostname;
- connectParam1, connectParam2 are RTMP connection parameters;
- streamName is stream name on server;
- streamParam1, streamParam2 are stteam parameters.

WCS server passes the parameters to backend server in REST hook in `custom`field, for example:

Connection parameters

```
URL:http://localhost:8081/apps/EchoApp/connect
OBJECT:
{
"nodeId" : "Qb3rAjf3lzoy6PEl1WZkUhRG1DsTykgj@192.168.1.1",
"appKey" : "flashStreamingApp",
"sessionId" : "/127.0.0.1:5643/192.168.1.1:1935",
"useWsTunnel" : false,
"useWsTunnelPacketization2" : false,
"useBase64BinaryEncoding" : false,
"keepAlive" : false,
"custom" : {
"connectParam1" : "val1",
"connectParam2" : "val2"
},
"login" : "rQq83sodiCPY0pJXCxGO"
}
```

Publishing parameters

```
URL:http://localhost:8081/apps/EchoApp/publishStream
OBJECT:
{
"nodeId" : "Qb3rAjf3lzoy6PEl1WZkUhRG1DsTykgj@192.168.1.1",
"appKey" : "flashStreamingApp",
"sessionId" : "/127.0.0.1:5643/192.168.1.1:1935",
"mediaSessionId" : "627990f9-8fe5-4e92-bb2a-863cc4eb43de",
"name" : "stream1",
"published" : true,
"hasVideo" : false,
"hasAudio" : true,
"status" : "NEW",
"record" : true,
"width" : 0,
"height" : 0,
"bitrate" : 0,
"minBitrate" : 0,
"maxBitrate" : 0,
"quality" : 0,
"mediaProvider" : "Flash",
"custom" : {
"streamParam1" : "val1",
"streamParam2" : "val2"
}
}
```

Playback parameters

```
URL:http://localhost:8081/apps/EchoApp/playStream
OBJECT:
{
"nodeId" : "Qb3rAjf3lzoy6PEl1WZkUhRG1DsTykgj@192.168.1.1",
"appKey" : "flashStreamingApp",
"sessionId" : "/127.0.0.1:5643/192.168.1.1:1935",
"mediaSessionId" : "stream1/127.0.0.1:5643/192.168.1.1:1935",
"name" : "stream1",
"published" : false,
"hasVideo" : true,
"hasAudio" : true,
"status" : "NEW",
"record" : false,
"width" : 0,
"height" : 0,
"bitrate" : 0,
"minBitrate" : 0,
"maxBitrate" : 0,
"quality" : 0,
"mediaProvider" : "Flash",
"custom" : {
"streamParam1" : "val1",
"streamParam2" : "val2"
}
}
```

This feature can be used for example to authenticate client on backend server while publishing or playing RTMP-stream on WCS server.

Connection parameters passing as stream parameters

In some cases it is necessary to pass RTMP connection parameters as stream parameters, authentication parameter for example

```
rtmp://test.flashphoner.com:1935/live/test?auth=key
```

This feature is enabled by the following setting

```
rtmp_use_stream_params_as_connection=true
```

In this case, the RTMP URL example above will be interpreted as

```
rtmp://test.flashphoner.com:1935/live?auth=key/test
```

Track order management in RTMP stream

Most players on various platforms suppose video track to be first in RTMP stream. To guarantee this order and to send videodata before audiodata, set the following parameter in flashphoner.propertiesfile:

```
rtmp_send_video_first=true
```

Note that if this setting is active, a stream containing audio track only can not be played as RTMP because audiodata will not be sent to client.

RTMP playback sound suppression

Sound may be disabled while stream published on server playback as RTMP. To do this, the following RTMP URL parameter should be passed:

```
rtmp://yourserver:1935/live?suppress_sound=true/streamName
```

In this case audio track will be replaced by silence.

Disabling RTMP playback

By default, RTMP playback is enabled. Since build 5.2.1081 this feature may be disabled if needed

```
rtmp_server_enabled=false
rtmfp_server_enabled=false
```

Known issues

1. When playing FullHD, 2K, 4K streams with big frame size, data packets to send may not fit to socket buffer, this leads to artifacts in some players

Symptoms: artifacts occur while playing RTMP stream via good channel

Solution: enable RTMP packets buffering with the parameter

```
rtmp.server_buffer_enabled=true
```