

# Injecting one stream into another

- [Overview](#)
  - [Supported codecs](#)
  - [Known limits](#)
- [Injection implementation in builds before 5.2.1618](#)
  - [Injection management using REST API](#)
    - [REST queries and responses](#)
    - [Parameters](#)
    - [Injecting a VOD stream from file](#)
  - [Configuration](#)
- [Injection implementation in build 5.2.1618 and newer](#)
  - [Configuration](#)
  - [REST API](#)
    - [REST queries and responses](#)
    - [Parameters](#)
    - [Injecting a VOD stream from file](#)
- [Quick testing](#)
- [Known issues](#)

## Overview

Since build [5.2.841](#) it is possible to inject one stream published on server into another. This feature can be used, for example, to add advertising material into a stream. The original stream content will be fully replaced by injected stream one until injected stream is stopped or injection is terminated.

## Supported codecs

Video:

- H264
- VP8

Audio:

- Opus
- AAC
- G711

## Known limits

1. Both streams to which injection is applied must be encoded with the same audio and video codecs.
2. Audio tracks in both streams should have the same samplerate and channels number.
3. Injection cannot be applied to SIP call streams. Use the special [audio](#) and [video](#) injection technologies for SIP call streams.
4. Only one stream can be injected into the stream simultaneously, but one stream can be injected into multiple streams.
5. Cyclic injection is not supported. It is not possible to inject stream1 into stream2 and then stream2 into stream1 without terminating the previous injection.

## Injection implementation in builds before [5.2.1618](#)

### Injection management using REST API

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

- HTTP: <http://test.flashphoner.com:8081/rest-api/stream/inject/startup>
- HTTPS: <https://test.flashphoner.com:8444/rest-api/stream/inject/startup>

Where:

- test.flashphoner.com - WCS server address
- 8081 - standard REST / HTTP port of WCS server
- 8444 - standard HTTPS port
- rest-api - mandatory URL part
- /stream/inject/startup - REST method used

## REST queries and responses

REST query	REST query body example	REST response example	Response states	Description
/stream/inject/startup	<pre>{   "localStreamName":   "stream1",   "remoteStreamName":   "stream2" }</pre>		200 - OK 400 - Bad request 404 - Not found 409 - Conflict 500 - Internal error	Inject stream2 into stream1
/stream/inject/find_all		<pre>[   {     "localStreamName":     "stream1",     "remoteStreamName":     "stream2"   } ]</pre>	200 - OK 404 - Not found	Find all injections on the server
/stream/inject/terminate	<pre>{   "localStreamName":   "stream1" }</pre>		200 - OK 400 - Bad request 404 - Not found 500 - Internal error	Stop injection into stream1

## Parameters

Name	Description	Example
localStreamName	Stream name to inject to	stream1
remoteStreamName	Stream name to be injected	stream2

## Injecting a VOD stream from file

Since build [5.2.1535](#) VOD stream directly from a file may be injected while sending the REST query /stream/inject/startup:

```
{
  "localStreamName": "host",
  "remoteStreamName": "vod-live://advertising.mp4"
}
```

In this case, injected file will play without a delay from the first key frame. The file can be injected to another stream, in this case the file also will be played from the beginning in that stream.

This feature is useful, for example, to inject advertising video into a stream being viewed.

## Configuration

Since build [5.2.1235](#) the parameter is added to set a time interval to wait for a keyframe in injected stream

```
inject_wait_keyframe_ms=1000
```

By default, the interval is 1000 milliseconds. If no keyframes arrived in injected stream during this time, server will generate a black picture (by default) or apicture from a file set by `custom_watermark_filename` parameter. This behaviour may be switched off by the following parameter

```
inject_wait_keyframe_ms=-1
```

In this case, the stream to be injected to will be played until keyframe arrives in the injected stream.

## Injection implementation in build 5.2.1618 and newer

### Configuration

Since build 5.2.1618 a new injector implementation is added allowing to choose what exactly to inject: audio, video or both. The feature may be enabled by the following parameter

```
use_new_injector=true
```

### REST API

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

- HTTP: <http://test.flashphoner.com:8081/rest-api/stream/inject2/startup>
- HTTPS: <https://test.flashphoner.com:8444/rest-api/stream/inject2/startup>

Where:

- test.flashphoner.com - WCS server address
- 8081 - standard REST / HTTP port of WCS server
- 8444 - standard HTTPS port
- rest-api - mandatory URL part
- /stream/inject2/startup - REST method used

### REST queries and responses

REST query	Request body	Response body	Response state	Description
/stream/inject2 /startup	<pre>{   "localStreamName":     "test",   "remoteStreamName":     "test2",   "video": true,   "audio": true,   "muteIfAbsent": true }</pre>		200 - OK  400 - Bad request  404 - Not found  409 - Conflict  500 - Internal error	Inject test2 stream into test stream

/stream/inject2 /find_all		<pre>[   {     "streamName": "test",     "videoInjectorInfo": {       "targetStreamName": "test2",       "rootStreamName": "test2",       "startTime": 1683344295099     },     "audioInjectorInfo": {       "targetStreamName": "test2",       "rootStreamName": "test2",       "startTime": 1683344295056     }   } ]</pre>	200 - OK 404 - Not found	Find all injections on the server
/stream/inject2 /terminate	<pre>{   "localStreamName": "test",   "video": true,   "audio": true }</pre>		200 - OK 400 - Bad request 404 - Not found 500 - Internal error	Stop injection into test stream

## Parameters

Parameter	Description	Example
localStreamName	Stream name to inject to	test
remoteStreamName	Stream name to be injected	test2
video	Replace video when injecting	true
audio	Replace audio when injecting	true
mutelfAbsent	Replace a track which is absent in a source stream to black picture or silence	true
videoInjectorInfo	Video information from injected stream	<pre>{   "targetStreamName": "test2",   "rootStreamName": "test2",   "startTime": 1683344295099 }</pre>
audioInjectorInfo	Audio information from injected stream	<pre>{   "targetStreamName": "test2",   "rootStreamName": "test2",   "startTime": 1683344295056 }</pre>

## Injecting a VOD stream from file

Since build [5.2.1719](#) VOD stream directly from a file may be injected while sending the REST query /stream/inject2/startup:

```
{
  "localStreamName": "host",
  "remoteStreamName": "vod-live://advertising.mp4",
  "video": true,
  "audio": true
}
```

In this case, injected file will play without a delay from the first key frame. The file can be injected to another stream, in this case the file also will be played from the beginning in that stream.

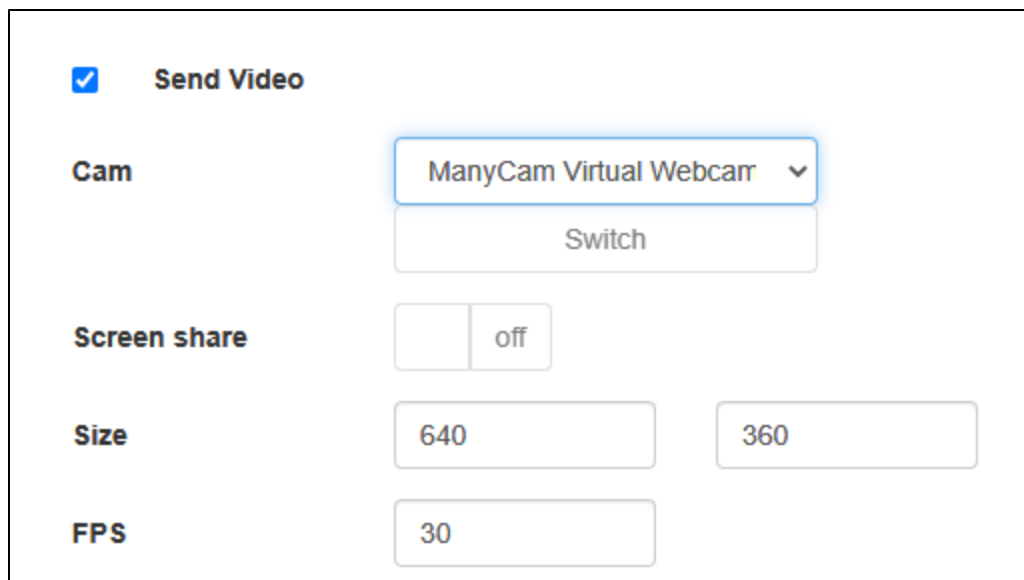
This feature is useful, for example, to inject advertising video into a stream being viewed.

## Quick testing

1. For test we use

- WCS server;
- Media Devices web application to publish streams;
- Two webcams, or two different PCs to publish streams;
- Playerweb application to play stream to be injected to;
- Chrome browser and [REST client](#) to send queries to the server

2. Open Media Devices application page, publish stream test in resolution 640x360



The screenshot shows a web interface for configuring video streaming. At the top, there is a checked checkbox labeled "Send Video". Below this, the "Cam" section features a dropdown menu currently set to "ManyCam Virtual Webcam" and a "Switch" button. The "Screen share" section has a toggle switch currently set to "off". The "Size" section contains two input fields, one with "640" and another with "360". The "FPS" section has a single input field with the value "30".

## Media Devices

### Video stats

Codec: H264  
Codec Rate: 90000  
Fir Count: 0  
Pli Count: 1  
Nack Count: 0  
Packets Sent: 361  
Bytes Sent: 270747  
Height: 360  
Width: 640  
Bitrate: 336728

### Audio stats

Codec: opus  
Codec Rate: 48000  
Packets Sent: 398  
Bytes Sent: 32444  
Bitrate: 32496

### Connection

Local



640x360

test

Stop

Player



dfda

Play

PUBLISHING

wss://test1.flashphoner.com:8443

Disconnect

Timeout

1000

msec

ESTABLISHED

### Video stats

### Audio stats

### Connection

2. Plat the stream test in Player example

# Player



WCS URL

wss://test1.flashphoner.com:8443

Stream

test

Volume



Full Screen



PLAYING

Stop

3. Publish adv stream in Media Devices example using another browser tab, another webcam or another PC

☒ Send Video

Cam

OBS Virtual Camera



Switch

Screen share

☐ off

Size

640

360

FPS

30

# Media Devices

**Video stats**  
Codec: H264  
Codec Rate: 90000  
Fir Count: 0  
Pli Count: 3  
Nack Count: 0  
Packets Sent: 781  
Bytes Sent: 417431  
Height: 360  
Width: 640  
Bitrate: 232864

**Audio stats**  
Codec: opus  
Codec Rate: 48000  
Packets Sent: 905  
Bytes Sent: 68422  
Bitrate: 31760

**Connection**

Local

1:20

640x360

advStop

Player

5f72Play

Video stats  
Audio stats  
Connection

PUBLISHING

wss://test1.flashphoner.com:8443Disconnect

Timeout1000msec

ESTABLISHED

4. Open REST client, send /stream/inject/startup query

Method  
POST

URL  
http://test1.flashphoner.com:8081/rest-api/stream/inject/startup

SEND

HEADERS

BODY

AUTHORIZATION

VARIABLES

1- {  
2- "localStreamName": "test",  
3- "remoteStreamName": "adv"  
4- }

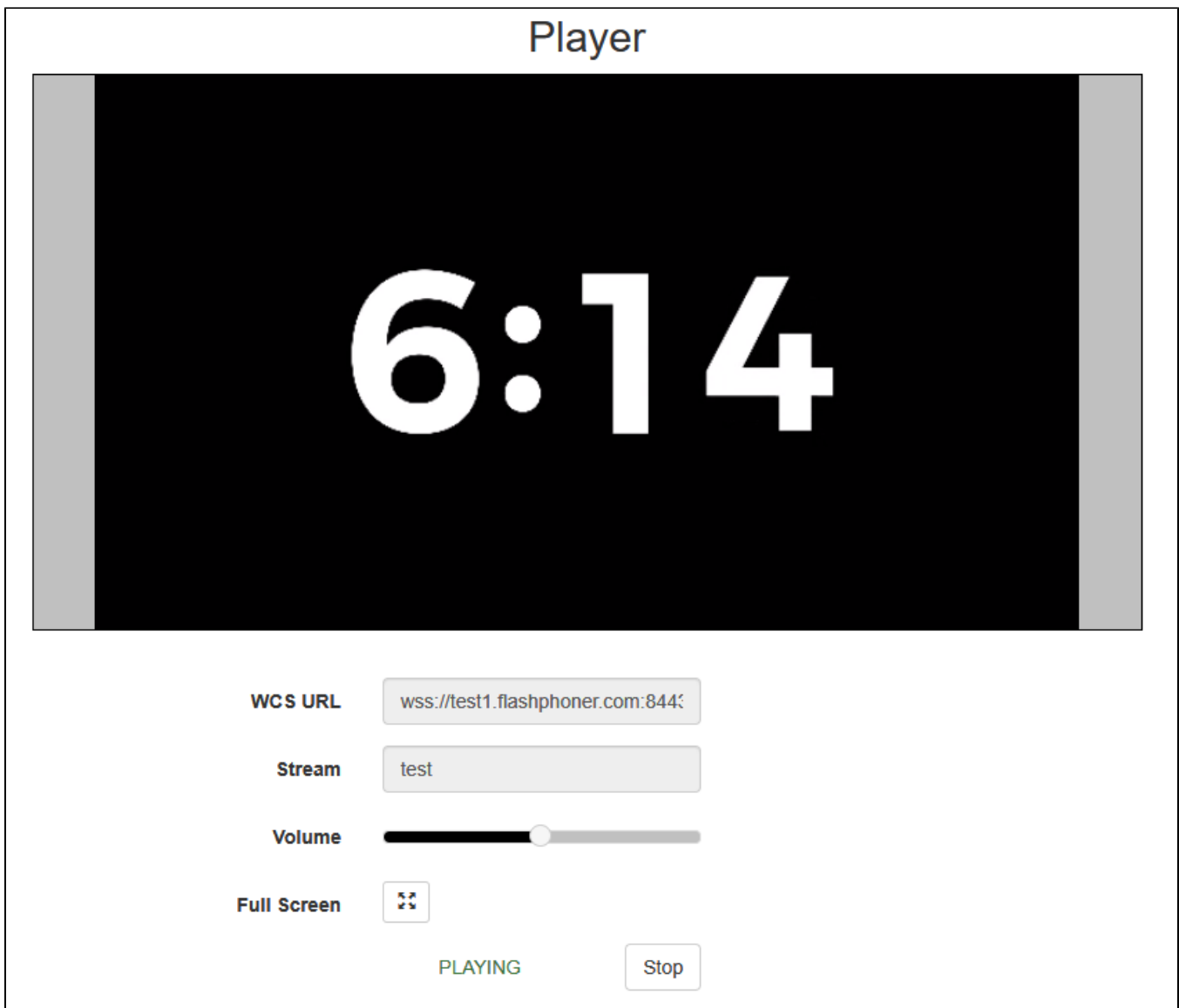
Response200 OK

91 B69 ms

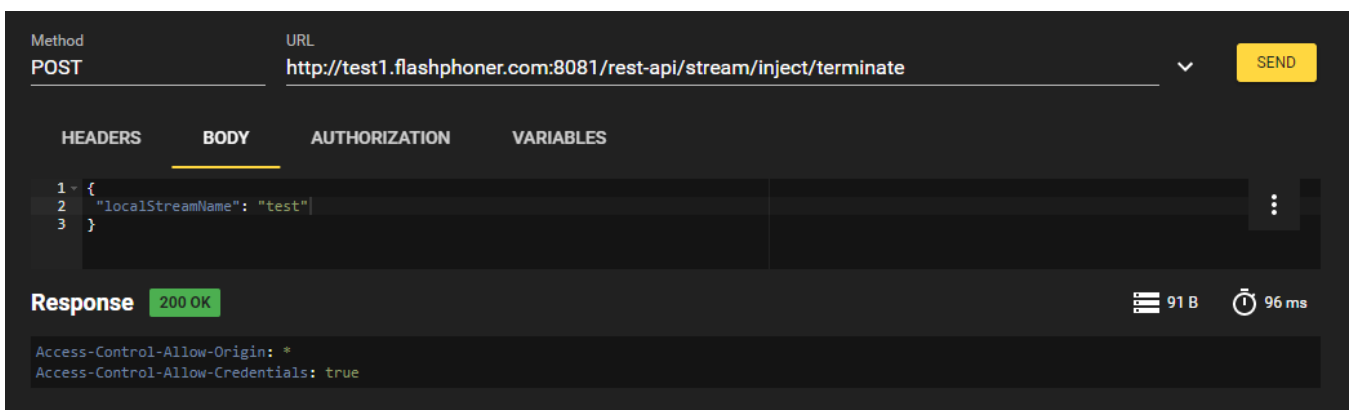
Access-Control-Allow-Origin: \*  
Access-Control-Allow-Credentials: true

5. adv stream content is playing in test stream





6. Send /stream/inject/terminate query



7. Original test stream content is playing again

## Player



**WCS URL**

wss://test1.flashphoner.com:8443

**Stream**

test

**Volume**



**Full Screen**



PLAYING

Stop

## Known issues

1. Video and audio may be out of sync after stopping injection of one RTMP stream into another

Symptoms: When one RTMP stream is injected into another, the original RTMP stream may play with a strong audio/video unsync after injected stream stops

Solution: enable RTMP incoming streams bufferization

```
rtmp_in_buffer_enabled=true
```