

Stream recording

- [Overview](#)
- [Quick manual on testing](#)
 - [Stream recording](#)
- [Configuration](#)
 - [Server side](#)
 - [Turning stream recording on and off](#)
 - [Forming the name of the stream record file](#)
 - [Record files rotation](#)
 - [Record files handling script](#)
 - [Adjusting record audio sample rate](#)
 - [Client side](#)
- [Stream recording on demand](#)
 - [REST methods and response statuses](#)
 - [Parameters](#)
- [Known issues](#)

Overview

A media stream captured by WCS can be recorded during publishing.

Supported protocols:

- WebRTC
- RTMP
- RTSP

Recording formats:

- MP4 for H.264 + AAC codecs
- WebM for VP8 + Vorbis codecs

Quick manual on testing

Stream recording

1. For this test we use the demo server at demo.flashphoner.com and the Stream Recording web application

https://demo.flashphoner.com/client2/examples/demo/streaming/stream_recording/recording.html

Stream Recording

My Video

wss://p11.flashphoner.com:84

Start

2. Click the "Start" button. Capturing and publishing of the stream starts.

Stream Recording



My Video

wss://p11.flashphoner.com:84

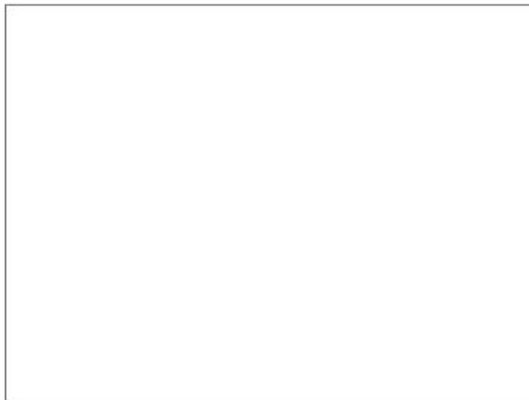
Stop

PUBLISHING



3. Click the "Stop" button. Broadcasting stops, and a link to play and download the recorded fragment appears.

Stream Recording



My Video

wss://p11.flashphoner.com:84

Start

UNPUBLISHED



Download recorded stream

Configuration

Server side

Turning stream recording on and off

By default, stream recording is turned on.

To turn recording off add the following line to the config /usr/local/FlashphonerWebCallServer/conf/flashphoner.properties:

```
record_streams=false
```

Parameter

```
record_flash_published_streams=true
```

turns on recording for the streams published with Flash, RTMP encoder or republished from another RTMP server.

Parameter

```
record_rtsp_streams=true
```

turns on recording for the streams captured from RTSP IP cameras.

Forming the name of the stream record file

Parameter stream_record_policy sets the way to form the name of the stream record file. For example,

```
stream_record_policy=streamName
```

means that the file name will match the stream name. So, the stream published with ffmpeg

```
ffmpeg -re -i BigBuckBunny.mp4 -preset ultrafast -acodec aac -vcodec h264 -strict -2 -f flv rtmp://test1.
flashphoner.com:1935/live/stream_ffmpeg
```

will be written to file stream_ffmpeg.mp4.

By default, the file name is formed by template

```
stream_record_policy=template
```

In its turn, the template is specified with stream_record_policy_template parameter. By default

```
stream_record_policy_template=stream-{mediaSessionId}-{login}
```

The following elements can be used in template:

Element	Description	Maximum size
{streamName}	Stream name	
{startTime}	Recording start time	20 characters
{sessionId}	Session ID in BASE64 encoding	60 characters
{mediaSessionId}	Media session ID	36 characters
{login}	Login	32 characters
{audioCodec}	Audiocodec	4 characters
{videoCodec}	Videocodec	4 characters

When the file name matches the stream name, it may contain characters that are not allowed in file names, slash '/' for example. In that case, the file name should be encoded using the parameter

```
encode_record_name=true,HEX
```

Then, the file name will be encoded with a hexadecimal number. The parameter

```
encode_record_name=true,BASE64
```

will encode the file name with BASE64 encoding.

Another way to escape invalid characters is to remove them using `exclude_record_name_characters` parameter. By default

```
exclude_record_name_characters=
```

For example, to remove colons, commas, periods and slashes set

```
exclude_record_name_characters=:.,/
```

Record files rotation

Stream records can be splitted to parts of a given duration using `record_rotation` parameter. For example, the setting

```
record_rotation=20
```

specifies a fragment duration as 10 seconds.

Record files handling script

The `on_record_hook_script` setting points to the shell script that is invoked when stream recording finishes.

The script is placed to the `/usr/local/FlashphonerWebCallServer/bin` folder by default:

```
on_record_hook_script=/usr/local/FlashphonerWebCallServer/bin/on_record_hook.sh
```

but it can be placed to any folder with any name, for example:

```
on_record_hook_script=/opt/on_record.sh
```

This script can be used to copy or move the stream record from the `WCS_HOME/records` directory to another location after recording completes.

Example:

```
STREAM_NAME=$1
SRC_FILE=$2
SRC_DIR="/usr/local/FlashphonerWebCallServer/records/"
REPLACE_STR="/var/www/html/stream_records/$STREAM_NAME-"
DST_FILE="${SRC_FILE/$SRC_DIR/$REPLACE_STR}"
cp $SRC_FILE $DST_FILE
```

Here

- \$1 - stream name
- \$2 - absolute path and file name of the stream record
- when stream recording ends, the record file is copied to `/var/www/html/stream_records/`

It is necessary to take into account the length of the absolute file name (including folder path) that will be formed when copying record file. If the absolute name of the target file exceeds 255 characters limit, copy command will fail with error, so the handling script will not work as expected.

Adjusting record audio sample rate

By default, audio track is recorded with sample rate 44.1 kHz. This value can be changed using the following parameter if necessary

```
record_audio_codec_sample_rate=48000
```

In this case, record audio sample rate will be set to 48 kHz.

Client side

If stream recording is enabled on the server, whether the stream is recorded or not is determined by the value of record parameter passed into the createStream function in the script of the publisher client:

- true - the stream published by this client is recorded;
- false - the stream is not recorded.

For instance, the script of the Stream Recording application recording.html, recording.js, contains the following code:

```
function publishStream(session) {  
    var streamName = $('#url').val().split('/')[3];  
    session.createStream({  
        name: streamName,  
        display: localVideo,  
        record: true,  
        receiveVideo: false,  
        receiveAudio: false  
        ...  
    }).publish();  
}
```

Stream recording on demand

Sometimes, it is necessary to record the stream that already exists on server, mixer output stream for example. This can be done with REST API. Note that only streams in "PUBLISHING" state can be recorded.

REST query must be HTTP/HTTPS POST query like this:

- HTTP: <http://streaming.flashphoner.com:8081/rest-api/stream/startRecording>
- HTTPS: <https://streaming.flashphoner.com:8444/rest-api/stream/startRecording>

Where:

- streaming.flashphoner.com is WCS server address
- 8081 is a standard WCS REST / HTTP port
- 8444 is a standard WCS REST / HTTPS port
- rest-api is mandatory prefix
- /stream/startRecording is REST method

REST methods and response statuses

REST method	Example of REST query	Example of REST response	Response statuses	Description
/stream /startRecording	<pre>{ "mediaSessionId": "5a072377-73c1-4caf-abd3" }</pre>		404 - Not found 500 - Internal error	Start stream recording in specified mediasession
/stream /stopRecording	<pre>{ "mediaSessionId": "5a072377-73c1-4caf-abd3" }</pre>		404 - Not found 500 - Internal error	Stop stream recording in specified mediasession

Parameters

Parameter name	Description	Example
mediaSessionId	Media session identificator	5a072377-73c1-4caf-abd3

Known issues

1. Maximum length of file name in all actual Linux file systems is limited to 255 characters. When record file is created, its name will be trimmed to this limit including extension and part number if rotation is enabled.
2. When stream published in chat room is recorded, file rotation will be automatically disabled, otherwise record files will not be merged.
3. In Amazon WCS instance, record files hook script requires sudo to execute any file operation.

Symptoms: record hook script does not perform any operation on record files

Solution: in Amazon WCS instance use sudo to make any file operation or call external script from record hook script, for example

```
sudo cp $SRC_FILE $DST_FILE
```