

# Запись трансляции

- Описание
- Краткое руководство по тестированию
  - Запись трансляции
- Настройка
  - Серверная часть
    - Включение и отключение записи потоков
    - Поддержка контейнеров MP4, WebM, MKV
    - Запись потоков в контейнер MPEG-TS
      - Ограничения
    - Формирование имени файла записи потока
    - Ротация файлов записей
    - Вычисление времени начала, окончания и длительности записи
    - Скрипт обработки записанных файлов
    - Директория для записанных файлов
    - Настройка частоты дискретизации аудио при записи
    - Настройка размещения атома moov в метаданных записи
    - Настройка битрейта аудио при записи с использованием кодека FDK
    - Настройка количества каналов звука в записи
    - Настройка производительности записи под высокими нагрузками
    - Запись VP8 потоков в контейнер webm
    - Отдельный каталог для временных файлов
    - Контроль свободного места для записи на диск
    - Остановка записи при ошибках
  - Клиентская часть
- Запись потоков по требованию
  - REST-методы и статусы ответа
  - Параметры
- Получение имени записанного файла
- Загрузка и воспроизведение записанного файла
  - Загрузка и воспроизведение указанного фрагмента файла
    - Настройка каталога для размещения фрагментов
    - Ограничения
- Контроль записи потока на бэкенд сервере
- Запись нескольких потоков в один файл с последующим микшированием
  - Поддерживаемые кодеки
  - REST API для мультирекордера
    - REST-методы и статусы ответа
    - Параметры
  - Выбор контейнера для записи
  - Имя записываемого файла
  - Директория для файлов записей нескольких потоков
  - Инструмент для микширования записанных потоков
  - Получение информации о дорожках из записанного файла
  - Извлечение отдельных потоков из MKV контейнера
  - Скрипт для обработки записанных файлов
  - Многопоточное кодирование при микшировании записанных потоков
    - Количество процессорных потоков при многопоточном кодировании
  - Отображение имени записанного потока
    - Декодирование символов в имени записанного потока
  - Отправка данных о завершении записи нескольких потоков
  - Контроль свободного места при микшировании записи нескольких потоков
  - Контроль добавления потока в рекордер на бэкенд сервере
- Известные проблемы

## Описание

Медиапоток, захваченный WCS, может быть записан при публикации.

Поддерживаемые протоколы:

- WebRTC
- RTMP
- RTSP

Форматы записи:

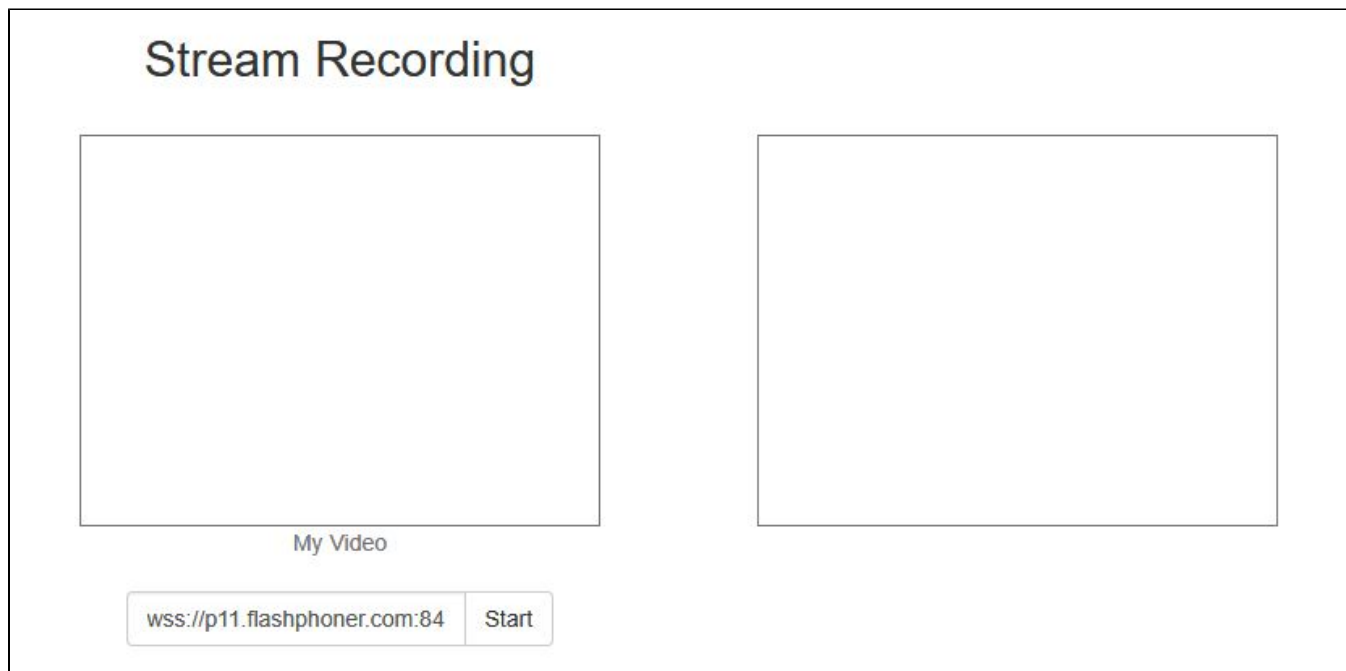
- MP4 для кодеков H.264 + AAC
- WebM для кодека VP8 + Vorbis
- TS для кодеков H.264 + ADTS
- MKV (начиная со сборки [5.2.1190](#))

## Краткое руководство по тестированию

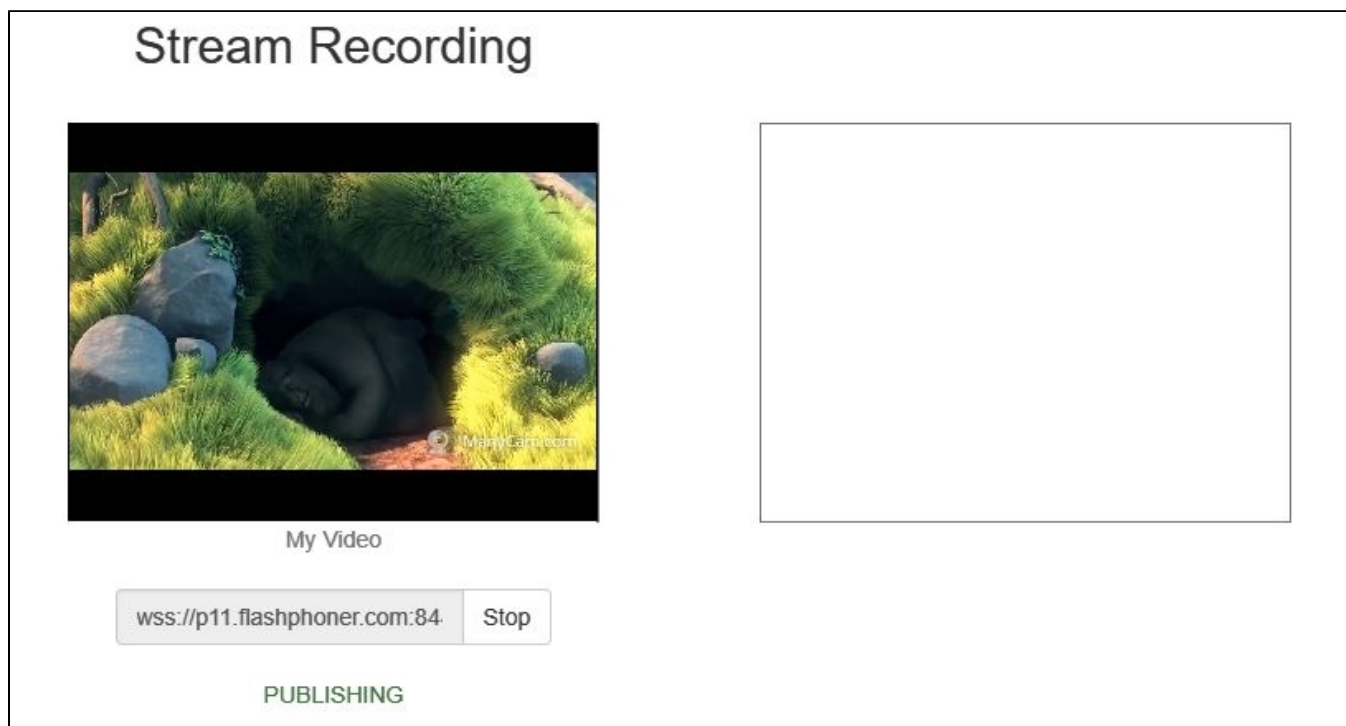
## Запись трансляции

1. Для теста используем демо-сервер [demo.flashphoner.com](https://demo.flashphoner.com) и веб-приложение Stream Recording

[https://demo.flashphoner.com/client2/examples/demo/streaming/stream\\_recording/recording.html](https://demo.flashphoner.com/client2/examples/demo/streaming/stream_recording/recording.html)

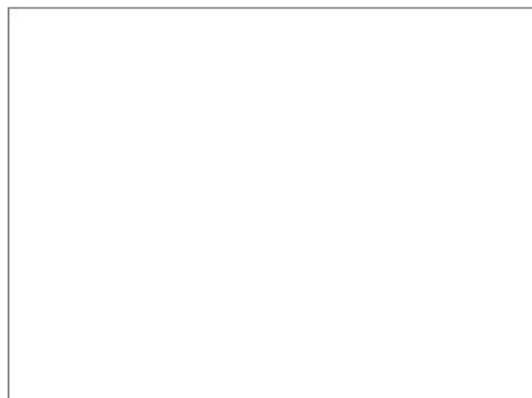


2. Нажмите кнопку "Start". Начнется захват и трансляция потока.



3. Нажмите кнопку "Stop". Трансляция остановится, отобразится ссылка на воспроизведение и скачивание записанного фрагмента

## Stream Recording



My Video

wss://p11.flashphoner.com:84

Start

UNPUBLISHED



Download recorded stream

## Настройка

### Серверная часть

#### Включение и отключение записи потоков

По умолчанию запись потоков включена на стороне WCS-сервера.

Для отключения в конфиг /usr/local/FlashphonerWebCallServer/conf/flashphoner.properties следует добавить

```
record_streams=false
```

Настройка

```
record_flash_published_streams=true
```

включает запись потоков, опубликованных при помощи Flash, RTMP-кодировщика или с другого RTMP-сервера.

Настройка

```
record_rtsp_streams=true
```

включает запись потоков, захваченных с RTSP IP-камер.

#### Поддержка контейнеров MP4, WebM, MKV

По умолчанию, H264 потоки записываются в контейнер MP4, VP8 потоки - в контейнер WebM. Начиная со сборки [5.2.1190](#), добавлена поддержка контейнера MKV, который универсален с точки зрения кодеков.

Для управления контейнерами используется настройка

```
record_formats=h264-mp4, vp8-webm
```

Запись всех кодеков в контейнер MKV включается следующим образом

```
record_formats=h264-mkv, vp8-mkv
```

В MKV может быть записан и один из кодеков (например, только VP8)

```
record_formats=h264-mp4, vp8-mkv
```

Обратите внимание, что потоки, которые не содержат видео дорожки, всегда записываются в MP4 контейнер (кодек AAC).

## Запись потоков в контейнер MPEG-TS

Начиная со сборки [5.2.610](#), H264 потоки могут быть записаны в контейнер MPEG-TS при помощи настройки

```
record_h264_to_ts=true
```

или, начиная со сборки [5.2.1190](#)

```
record_formats=h264-ts, vp8-webm
```

Настройка record\_h264\_to\_ts имеет приоритет над record\_formats , то есть при

```
record_formats=h264-mp4, vp8-mkv
record_h264_to_ts=true
```

H264 потоки будут записываться в контейнер MPEG-TS

### Ограничения

- 1. VLC до версии 3.0.8 может не играть записи в контейнере TS.
- 2. При проигрывании в VLC может не работать перемотка.

## Формирование имени файла записи потока

По умолчанию, имя файла формируется по шаблону, который задаетсянастройкойstream\_record\_policy\_template.

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

Доступны следующие элементы шаблона:

| Элемент           | Описание                                                      | Максимальный размер |
|-------------------|---------------------------------------------------------------|---------------------|
| {streamName}      | Имя потока                                                    |                     |
| {duration}        | Длительность файла, только для MP4-записей                    |                     |
| {startTime}       | Время начала записи потока или фрагмента                      | 20 символов         |
| {endTime}         | Время окончания записи потока или фрагмента                   | 20 символов         |
| {startTimeMillis} | Время сервера на момент начала записи потока или фрагмента    | 20 символов         |
| {endTimeMillis}   | Время сервера на момент окончания записи потока или фрагмента | 20 символов         |
| {sessionId}       | Идентификатор сессии в кодировке BASE64                       | 60 символов         |
| {mediaSessionId}  | Идентификатор медиасессии                                     | 36 символов         |
| {login}           | Логин                                                         | 32 символа          |
| {audioCodec}      | Аудиокодек                                                    | 4 символа           |
| {videoCodec}      | Видеокодек                                                    | 4 символа           |

Например, настройка

```
stream_record_policy_template={streamName}
```

означает, что имя файла будет соответствовать имени потока. Поток, опубликованный при помощи ffmpeg

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

будет записан в файл stream\_ffmpeg.mp4.

Расширение файла добавляется в зависимости от параметров потока и контейнера: mp4 для H264+AAC и webm для VP8+opus.

Если имя файла создается из имени потока, в нем могут быть символы, недопустимые к использованию в именах, например, прямой слэш '/'. В этом случае имя файла должно быть закодировано при помощи настройки

```
encode_record_name=true,HEX
```

При этом имя файла будет закодировано шестнадцатиричным числом. Настройка

```
encode_record_name=true,BASE64
```

кодирует имя файла при помощи BASE64.

Другой способ экранирования недопустимых символов - их удаление при помощи параметра exclude\_record\_name\_characters. По умолчанию

```
exclude_record_name_characters=
```

Например, для исключения двоеточия, запятой, точки и слэша необходимо указать

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

## Ротация файлов записей

Потоки могут записываться частями заданной длительности или объема при помощи параметра record\_rotation. Например, настройка

```
record_rotation=20
```

определяет длительность фрагмента в 20 секунд, а настройка

```
record_rotation=10M
```

задает объем фрагмента в 10 мегабайт.

Если шаблон имени файла содержит элемент {startTime}, в имя файла подставляется время начала записи фрагмента. Если шаблон содержит элемент {endTime}, в имя файла подставляется время окончания записи фрагмента. Например, при настройках

```
record_rotation=20  
stream_record_policy_template={streamName}-{startTime}-{endTime}
```

фрагменты записи потока test именуются следующим образом

```
test-1553577643961-1553577663988_1.mp4  
test-1553577663989-1553577683997_2.mp4  
test-1553577683997-1553577701626_3.mp4  
...
```

Фрагменты нумеруются последовательно, начиная с 1. Для новой медиасессии (даже если опубликован поток с таким же именем) нумерация начинается заново, т.е., если статическая часть шаблона не уникальна (например, только имя потока), файлы, записанные ранее, будут перезаписаны поверх.

При необходимости, нумерация может быть отключена настройкой

```
record_rotation_index_enabled=false
```

В этом случае индексы не добавляются и ротация осуществляется в полном соответствии с заданным шаблоном имени файла. При этом, если шаблон не обеспечивает уникальность, файлы, записанные ранее, будут перезаписаны поверх.

## Вычисление времени начала, окончания и длительности записи

Начиная со сборки [5.2.458](#), время начала, окончания и длительности записи вычисляется по меткам времени кадров потока. При этом, отсчет меток времени RTMP потока всегда начинается с 0, для WebRTC потока браузер фиксирует полную метку времени по данным своих часов.

```
stream_record_policy_template={streamName}-{startTime}-{endTime}-{duration}
```

В сборке [5.2.635](#) добавлена возможность указать время начала и окончания записи по часам сервера

```
stream_record_policy_template={streamName}-{startTimeMillis}-{endTimeMillis}
```

При этом метки времени в потоке в общем случае будут отличаться от показаний часов сервера.

Для более точного вычисления времени с учетом синхронизации аудио и видео в записи, необходимо включить буферизацию аудиоданных при записи. С этой целью, добавлена настройка

```
record_audio_buffer_max_size=100
```

По умолчанию, размер буфера установлен в 100 пакетов.

## Скрипт обработки записанных файлов

Настройка `on_record_hook_script` указывает на shell-скрипт, который вызывается по завершении записи потока.

По умолчанию скрипт располагается в каталоге `/usr/local/FlashphonerWebCallServer/bin`:

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

но может быть размещен в любом другом месте под другим именем, например:

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

Этот скрипт можно использовать для копирования или перемещения записи потока из директории `WCS_HOME/records` в другое место по завершении записи.

Пример:

```
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
```

Здесь

- `$1` - имя потока
- `$2` - абсолютное имя файла записи потока
- по завершении записи потока файл записи копируется в директорию `/var/www/html/stream_records/`

Необходимо учитывать длину абсолютного имени файла (с учетом имени каталога), которое будет получено при копировании. Если абсолютное имя целевого файла превышает 255 символов, команда копирования завершится с ошибкой, и скрипт не сработает в соответствии с ожиданиями.

Начиная со сборки [5.2.801](#), WCS запускается от пользователя 'flashphoner' для большей безопасности. В связи с этим, при перемещении записанных файлов в другой каталог, необходимо разрешить запись в этот каталог. Например, если файлы копируются в каталог /opt/media

```
STREAM_NAME=$1
FILE_NAME=$2

echo $STREAM_NAME:$FILE_NAME >> /usr/local/FlashphonerWebCallServer/logs/record.log
cp $FILE_NAME /opt/media
```

необходимо назначить права на запись в этот каталог

```
sudo chmod o+w /opt/media
```

## Директория для записанных файлов

По умолчанию записи потока сохраняются в каталог WCS\_HOME/records. Начиная со сборки [5.2.687](#), каталог для сохранения записей можно изменить при помощи параметра

```
record_dir=/usr/local/FlashphonerWebCallServer/records
```

Если в данной настройке указан не тот каталог, который используется по умолчанию, загрузка записей в примере Stream Recording не будет работать. В этом случае рекомендуется настроить собственный веб-сервер для загрузки файлов записей.

Начиная со сборки [5.2.801](#), WCS запускается от пользователя 'flashphoner' для большей безопасности. В связи с этим, при изменении данной настройки, необходимо разрешить запись в указанный каталог. как описано выше.

## Настройка частоты дискретизации аудио при записи

По умолчанию, запись звука ведется с частотой дискретизации 44.1 кГц. При необходимости, это значение можно изменить при помощи параметра

```
record_audio_codec_sample_rate=48000
```

В данном случае частота дискретизации будет установлена в 48 кГц.

## Настройка размещения атома moov в метаданных записи

Для того, чтобы файл записи можно было играть во время загрузки (progressive downloading), атом moov в метаданных записи должен предшествовать атому mdat. С этой целью в последних сборках добавлена настройка, установленная по умолчанию

```
mp4_container_moov_first=true
```

Для оптимизации процесса сохранения записи на диске и уменьшения количества дисковых операций, предусмотрено резервирование места под атом moov при создании файла. Эта возможность включается при помощи параметра

```
mp4_container_moov_first_reserve_space=true
```

Размер резервируемой области устанавливается в килобайтах настройкой

```
mp4_container_moov_reserved_space_size=2048
```

По умолчанию, резервируется 2048 килобайт. Таким образом, если резервирование места под атом moov включено, размер записанного файла будет не меньше указанного значения, это следует учитывать при настройке ротации записей по размеру.

## Настройка битрейта аудио при записи с использованием кодека FDK

В сборке [5.2.428](#) добавлена возможность указать [режим битрейта аудио](#) дорожки при записи с использованием кодека FDK. По умолчанию, установлен режим 5 (переменный битрейт в среднем 112 кбит/с). Это значение может быть изменено при помощи настройки

```
record_fdk_aac_bitrate_mode=5
```

Возможные режимы битрейта:

- 0 - постоянный битрейт
- 1-5 - переменный битрейт

Необходимо отметить, что воспроизведение записанных файлов с указанием определенного отрезка при помощи модуля `nginx_http_mp4_module` возможно только при использовании переменного битрейта.

## Настройка количества каналов звука в записи

В сборке [5.2.610](#) добавлена возможность указывать количество каналов звука в записи при помощи настройки

```
record_audio_codec_channels=2
```

По умолчанию, количество каналов установлено в 2 (стерео). Чтобы записать поток с моно звуком, необходимо указать

```
record_audio_codec_channels=1
```

## Настройка производительности записи под высокими нагрузками

При одновременной записи большого количества публикаций сохранение файлов на диск по окончании записи может занимать существенное время. Для того, чтобы ускорить сохранение, в сборке [5.2.639](#) добавлена возможность задать число процессорных потоков, которые будут заниматься записью, при помощи настройки

```
file_recorder_thread_pool_max_size=4
```

По умолчанию, обработкой записи занимаются 4 потока. При необходимости, их число может быть увеличено. Отметим, что не рекомендуется устанавливать количество потоков больше, чем имеется процессоров в системе.

## Запись VP8 потоков в контейнер webm

Начиная со сборки [5.2.905](#), для записи VP8 потоков в контейнер webm по умолчанию используется Java-реализация

```
webm_java_writer_enable=true
```

Для данной реализации доступны настройки длительности (в миллисекундах) и размера кластера (в байтах), по достижении которых данные потока записываются в файл на диске

```
webm_cluster_duration_limit=100000  
webm_cluster_size_limit=131072
```

При возникновении проблем с записью, существует возможность переключиться на реализацию на базе ffmpeg

```
webm_java_writer_enable=false
```

## Отдельный каталог для временных файлов

При записи потока, данные записываются во временный файл, а затем копируются в файл записи, именованный в соответствии с шаблоном. Начиная со сборки [5.2.963](#), для размещения временных файлов можно указать отдельный каталог при помощи настройки

```
record_tmp_dir=/tmp
```

Это позволяет, например, помещать временные файлы на RAM-диск, что существенно увеличивает скорость записи.

Процесс WCS должен иметь достаточно прав для записи в каталог для временных файлов, поэтому пользователь flashphoner должен быть назначен владельцем этого каталога. Например, если каталог настроен как

```
record_tmp_dir=/opt/wcs
```

то пользователю flashphoner должны быть назначены права на этот каталог

```
sudo chown -R flashphoner:flashphoner /opt/wcs
```

По умолчанию, временные файлы помещаются в каталог `/usr/local/FlashphonerWebCallServer/records`

## Контроль свободного места для записи на диск

В сборке [5.2.1209](#) добавлена проверка доступного свободного места на диске при записи. Если доступно меньше заданного значения, запись останавливается, или не стартует, если место недоступно к моменту старта записи. При этом в серверный лог выводится сообщение

```
Not enough available disk space
```

Ограничение задается настройкой (по умолчанию, 1 Гб)

```
file_recorder_min_space=1g
```

Допускается указывать значения в гигабайтах (суффикс `g`) и в мегабайтах (суффикс `m`), например

```
file_recorder_min_space=1024m
```

Если запись была остановлена, скрипт постобработки выполнится для файла записи.

## Остановка записи при ошибках

В сборке [5.2.1236](#) добавлены настройки, определяющие поведение при ошибках, возникающих при записи потока. По умолчанию, запись потока останавливается после 3 ошибок в течение 60 минут:

```
file_recorder_error_interval=60
file_recorder_max_errors_per_interval=3
```

Если к ошибке привел определенный кадр в медиапотоке, этот и последующие кадры не будут записаны, до успешного получения очередного ключевого кадра.

## Клиентская часть

При включении записи потоков на сервере, будет ли записан поток, или нет, зависит от значения параметра `record`, переданного функции `createStream` в скрипте публикующего клиента:

- `true` - поток, опубликованный с использованием этого клиента, будет записан;
- `false` - поток не будет записан.

Например, в скрипте веб-приложения Stream Recording [recording.html](#), [recording.js](#), содержится следующий код:

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

## Запись потоков по требованию

В некоторых случаях, необходимо записать поток, который уже транслируется сервером, например, выходной поток микшера. Это можно сделать при помощи REST API. Обратите внимание, что только потоки в статусе "PUBLISHING" могут быть записаны.

REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде:

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

Здесь:

- streaming.flashphoner.com- адрес WCS-сервера
- 8081 - стандартный REST / HTTP порт WCS-сервера
- 8444- стандартный HTTPS порт
- rest-api- обязательный префикс
- /recorder/startup- используемый REST-вызов

### REST-методы и статусы ответа

| REST-метод                                                | Пример тела REST-запроса                                                                                                                                                                        | Пример тела REST-ответа                                                                                                                                     | Статусы ответа                              | Описание                                        |
|-----------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|-------------------------------------------------|
| /stream<br>/startRecording,<br><br>/recorder<br>/startup  | <pre>{<br/>  "mediaSessionId": "5a072377-73c1-4caf-abd3",<br/>  "config": {<br/>    "fileTemplate":<br/>    "{streamName}-{startTime}-{endTime}",<br/>    "rotation": "20M"<br/>  }<br/>}</pre> |                                                                                                                                                             | 404 - Not found<br><br>500 - Internal error | Начать запись потока в указанной медиасессии    |
| /stream<br>/stopRecording,<br><br>/recorder<br>/terminate | <pre>{<br/>  "mediaSessionId": "5a072377-73c1-4caf-abd3"<br/>}</pre>                                                                                                                            |                                                                                                                                                             | 404 - Not found<br><br>500 - Internal error | Завершить запись потока в указанной медиасессии |
| /recorder<br>/find_all                                    |                                                                                                                                                                                                 | <pre>[<br/>  {<br/>    "fileName": "9c3e-test-1563776083752-{endTime}.mp4",<br/>    "mediaSessionId":<br/>    "5a072377-73c1-4caf-abd3"<br/>  }<br/>]</pre> | 404 - Not found<br><br>500 - Internal error | Найти записываемые сессии                       |

### Параметры

| Имя параметра  | Описание                                                             | Пример                             |
|----------------|----------------------------------------------------------------------|------------------------------------|
| mediaSessionId | Идентификатор сессии                                                 | 5a072377-73c1-4caf-abd3            |
| config         | Настройки записи, имеют приоритет над настройками сервера            |                                    |
| fileTemplate   | Шаблон имени файла записи                                            | {streamName}-{startTime}-{endTime} |
| rotation       | Включает/отключает ротацию файлов и способ нарезки (время или объем) | 20M                                |

Логика работы записи по требованию выглядит следующим образом:

- При вызове REST API/recorder/startup завершается текущая запись, если она идет в данный момент.
- Стартует новая запись, при этом применяются переданные через REST настройки.
- Если какая-либо из настроек не передана, применяется соответствующая настройка сервера.

Например, если необходимо передать точное имя файла для записи потока и отключить ротацию, должен быть передан запрос:

```
/recorder/startup
{
  "mediaSessionId": "1234567890abcdefg",
  "config": {
    "fileTemplate": "test",
    "rotation": "disabled"
  }
}
```

REST запрос /recorder/find\_all возвращает список сессий, записываемых в данный момент. В списке отражаются как записи по требованию, запущенные через REST API, так и записи, инициированные через WebSDK:

```
[
  {
    "fileName": "003f-1563776713987-{endTime}.mp4",
    "mediaSessionId": "5af9c820-ac49-11e9-9f06-693cb47c4042"
  },
  {
    "fileName": "stream-57882100-ac49-11e9-afdd-6752f5be57a9-jtdvnittjkrd8rsc3dnfbger2o.mp4",
    "mediaSessionId": "57882100-ac49-11e9-afdd-6752f5be57a9"
  }
]
```

## Получение имени записанного файла

Существуют следующие способы узнать имя записанного файла, например, для его скачивания:

1. На сервере имя файла получает [скрипт обработки записанных файлов](#) по окончании записи
2. Если имя файла необходимо знать в браузере, [шаблон](#) должен быть сформирован таким образом, чтобы в него входили параметры потока, которые могут быть получены при помощи REST API, например

```
stream_record_policy_template={streamName}-{mediaSessionId}
```

3. При использовании WebSDK для записи потока, имя записанного файла можно получить при помощи функции getRecordInfo()

```
...
}).on(STREAM_STATUS.UNPUBLISHED, function (stream) {
  setStatus(stream.status());
  showDownloadLink(stream.getRecordInfo());
  onStopped();
})
...
```

Отметим, что при большом размере записи событие STREAM\_STATUS.UNPUBLISHED может прийти через значительное время. В сборке [5.2.673](#) добавлена настройка, которая ограничивает интервал ожидания окончания записи (по умолчанию 15 секунд)

```
record_stop_timeout=15
```

## Загрузка и воспроизведение записанного файла

Записанный файл доступен во встроенном веб-сервере WCS по прямой ссылке вида

```
https://test.flashphoner.com:8444/client/records/stream.mp4
```

Здесь

- test.flashphoner.com - URL WCS сервера
- stream.mp4 - имя записанного файла

По умолчанию, WCS возвращает заголовок

```
Content-Disposition: inline;filename="stream.mp4"
```

в этом случае браузер попытается проиграть файл. Это поведение включается при помощи настройки

```
record_response_content_disposition_header_value=inline
```

Для того, чтобы браузер загружал записанный файл, не пытаясь его проиграть, необходимо установить настройку

```
record_response_content_disposition_header_value=attachment
```

## Загрузка и воспроизведение указанного фрагмента файла

В сборке [5.2.894](#) добавлена возможность загрузки и воспроизведения указанного фрагмента файла. Для этого необходимо запросить файл с указанием начала и конца фрагмента в секундах

```
https://test.flashphoner.com:8444/client/records/stream.mp4?start=11&end=60
```

Может быть указано только время начала воспроизведения или только время конца.

Запрошенные фрагменты записываются в тот же каталог, где расположены файлы записей, с добавлением в имени времен начала и конца, например

```
stream-s11-e60.mp4
```

После загрузки такие файлы не удаляются, если такой же фрагмент запрошен снова, сервер отправит уже существующий фрагмент.

Начиная со сборки [5.2.899](#), воспроизведение фрагментов поддерживается и для записей только с аудио.

### Настройка каталога для размещения фрагментов

По умолчанию, фрагменты записываются в каталог

```
/usr/local/FlashphonerWebCallServer/records
```

(туда же, куда по умолчанию помещены записи).

В сборке [5.2.957](#) добавлена возможность указать отдельный каталог для фрагментов при помощи настройки

```
mp4_cutter_dir=/tmp
```

### Ограничения

1. Загрузка и воспроизведение фрагментов поддерживается только для MP4 контейнера. При запросе webm файла запись всегда загружается полностью.
2. При указании времени начала фрагмента, воспроизведение может начаться чуть раньше, в зависимости от расположения ключевого фрейма в файле.

## Контроль записи потока на бэкенд сервере

В сборке [5.2.1416](#) добавлена возможность получения событий, сигнализирующих о том, что началась или завершилась запись потока. Для этого WCS отправляет на бэкенд сервер запрос /StreamEvent

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
OBJECT:
{
  "nodeId" : "d2hxbqNPE04vGeZ51NPhDuId6k3hUrBB@192.168.1.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.1.83:49977/192.168.1.39:8443-591009c4-e051-4722-b34d-71cf2ade3bed",
  "mediaSessionId" : "15de2290-4089-11ed-88fe-d78a87cf3386",
  "type" : "startedRecording",
  "payload" : {
    "fileName" : "stream-15de2290-4089-11ed-88fe-d78a87cf3386-8mv1of1o4fni58k0qdomu52kru.mp4"
  }
}
```

при старте записи и

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
OBJECT:
{
  "nodeId" : "d2hxbqNPE04vGeZ51NPhDuId6k3hUrBB@192.168.1.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.1.83:49977/192.168.1.39:8443-591009c4-e051-4722-b34d-71cf2ade3bed",
  "mediaSessionId" : "15de2290-4089-11ed-88fe-d78a87cf3386",
  "type" : "stoppedRecording",
  "payload" : {
    "fileName" : "stream-15de2290-4089-11ed-88fe-d78a87cf3386-8mv1of1o4fni58k0qdomu52kru.mp4"
  }
}
```

при остановке записи.

При обновлении WCS с предыдущих сборок в конфигурацию [бэкенд приложения](#) необходимо [добавить](#) метод StreamEvent

```
add app-rest-method defaultApp StreamEvent
add app-rest-method MyAppKey StreamEvent
```

## Запись нескольких потоков в один файл с последующим микшированием

В сборке [5.2.1012](#) добавлена возможность записи нескольких потоков в один файл. В дальнейшем потоки могут быть извлечены из этого файла и смикшированы специальным инструментом. Несколько потоков могут быть записаны в MP4 контейнер или, начиная со сборки [5.2.1440](#), в MKV контейнер. Эта возможность предназначена, например, для записи видеоконференций. В отличие от [MCU микшера](#), здесь микширование работает только при обработке уже записанного файла, и позволяет расходовать меньше ресурсов процессора непосредственно во время проведения конференции.

Запись нескольких потоков управляется по REST API.

### Поддерживаемые кодеки

Контейнер MP4:

- H264
- AAC

Контейнер MKV:

- H264
- VP8
- Opus
- AAC
- PCMA
- PCMU
- G722

### REST API для мультирекордера

REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде:

- HTTP:http://streaming.flashphoner.com:8081/rest-api/multipleRecorder/startup
- HTTPS:https://streaming.flashphoner.com:8444/rest-api/multipleRecorder/startup

Здесь:

- streaming.flashphoner.com- адрес WCS-сервера
- 8081 - стандартный REST / HTTP порт WCS-сервера
- 8444- стандартный HTTPS порт
- rest-api- обязательный префикс
- /multipleRecorder/startup - используемый REST-вызов

## REST-методы и статусы ответа

| REST-метод                  | Пример тела REST-запроса                                                                                           | Пример тела REST-ответа                                                                                                                                                                                                                                                                      | Статусы ответа                                                    | Описание                                           |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|----------------------------------------------------|
| /multipleRecorder/startup   | <pre>{   "uri": "multi-recorder://test-record" }</pre>                                                             |                                                                                                                                                                                                                                                                                              | 409 - Conflict<br><br>500 - Internal error                        | Запустить рекордер для записи нескольких потоков   |
| /multipleRecorder/add       | <pre>{   "uri": "multi-recorder://test-record",   "mediaSessionId": "866a9910-fbfe-11eb-aae4-6f99b0c80a3a" }</pre> |                                                                                                                                                                                                                                                                                              | 404 - Not found<br><br>409 - Conflict<br><br>500 - Internal error | Добавить в рекордер поток из указанной медиасессии |
| /multipleRecorder/find_all  |                                                                                                                    | <pre>[   {     "mediaSessionsId": [       "866a9910-fbfe-11eb-aae4-6f99b0c80a3a",       "9f1e2530-fbfe-11eb-9ec1-77172ac14d86",       "a970d0a0-fbfe-11eb-8fcc-912807bab442"     ],     "uri": "multi-recorder://test-record",     "fileName": "multi-recorder__test-record.mp4"   } ]</pre> | 404 - Not found<br><br>500 - Internal error                       | Найти все рекордеры                                |
| /multipleRecorder/remove    | <pre>{   "uri": "multi-recorder://test-record",   "mediaSessionId": "866a9910-fbfe-11eb-aae4-6f99b0c80a3a" }</pre> |                                                                                                                                                                                                                                                                                              | 404 - Not found<br><br>500 - Internal error                       | Удалить поток из рекордера                         |
| /multipleRecorder/terminate | <pre>{   "uri": "multi-recorder://test-record" }</pre>                                                             |                                                                                                                                                                                                                                                                                              | 404 - Not found<br><br>500 - Internal error                       | Остановить рекордер                                |

## Параметры

| Имя параметра  | Описание                            | Пример                               |
|----------------|-------------------------------------|--------------------------------------|
| uri            | URI рекордера                       | multi-recorder://test-record         |
| mediaSessionId | Идентификатор медиасессии потока    | 866a9910-fbfe-11eb-aae4-6f99b0c80a3a |
| filename       | Имя файла, куда производится запись | multi-recorder__test-record.mp4      |

## Выбор контейнера для записи

Данная возможность доступна, начиная со сборки [5.2.1440](#). По умолчанию, запись производится в контейнер MP4

```
multi_recorder_type=MP4
```

При необходимости (например, при публикации VP8+Opus потоков в конференции), можно выбрать контейнер MKV

```
multi_recorder_type=MKV
```

## Имя записываемого файла

Имя файла для записи нескольких потоков формируется по [шаблону](#). При этом:

1. Параметр{streamName} подставляется согласно URI рекордера, с заменой символов, не допустимых к использованию в именах файлов, на подчеркивания.

2. Параметры{startTime},{endTime} не могут быть определены, поскольку зависят от меток времени в потоке, а потоков в данном случае несколько. Поэтому рекомендуется для присвоения метки времени файлу использовать параметры{startTimeMillis},{endTimeMillis}, которые проставляются согласно часам сервера.

Например, с шаблоном

```
stream_record_policy_template={streamName}-{startTime}-{startTimeMillis}-{endTime}-{endTimeMillis}
```

имя файла для рекордера с URI

```
"uri": "multi-recorder://test-record"
```

будет следующим:

```
multi-recorder__test-record--1-1628821032180--1-1628821151750.mp4
```

Здесь{startTime},{endTime} заменены на -1.

## Директория для файлов записей нескольких потоков

По умолчанию файлы записей нескольких потоков сохраняются в каталог WCS\_HOME/records. Начиная со сборки [5.2.1088](#), каталог для сохранения записей можно изменить при помощи параметра

```
multi_record_dir=/usr/local/FlashphonerWebCallServer/records
```

Необходимо, чтобы указанный каталог был доступен для записи, Например, при настройке

```
multi_record_dir=/opt/media
```

права должны быть заданы следующим образом

```
sudo chmod o+w /opt/media
```

## Инструмент для микширования записанных потоков

Из файла с несколькими потоками внутри по умолчанию может быть воспроизведен только первый поток. Чтобы смотреть все потоки в файле, их необходимо смикшировать. Для этого предназначен инструмент OfflineMixerTool, запускаемый следующим образом:

```
cd /usr/local/FlashphonerWebCallServer/tools
./offline_mixer_tool.sh multi-recorder__test-record--1-1628821032180--1-1628821151750.mp4
```

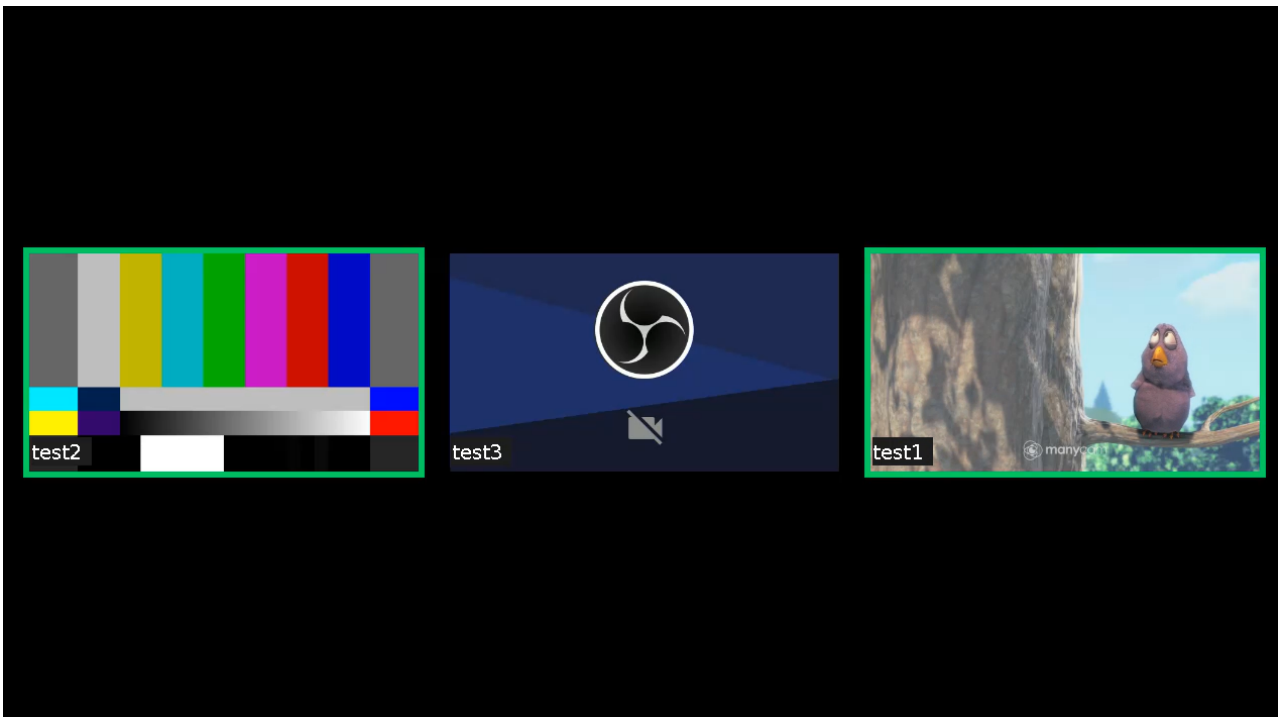
Настройки микширования задаются в файле `/usr/local/FlashphonerWebCallServer/conf/offline_mixer.json`. По умолчанию настройки следующие:

```
{
  "hasVideo": "true",
  "hasAudio": "true",
  "mixerDisplayStreamName": true
}
```

Микшированный файл помещается в тот же каталог, где лежит оригинальный файл, к его имени добавляется суффикс `_mixed`, например

```
multi-recorder__test-record--1-1628821032180--1-1628821151750_mixed.mp4
```

Пример кадра из микшированного файла



Начиная со сборки [5.2.1481](#), микшируются потоки как из MP4, так и из MKV контейнеров. Результат всегда записывается в MP4 контейнер.

## Получение информации о дорожках из записанного файла

В сборке [5.2.1049](#) с помощью инструмента для микширования записанных потоков можно получить информацию о дорожках в файле с несколькими потоками. Для этого необходимо запустить его следующим образом:

```
./offline_mixer_tool.sh --show-tracks-info ../records/multi-recorder__test-record.mp4
```

В этом случае инструмент выведет данные в формате JSON. Например, для файла с записью конференции с двумя участниками:

**Two participants track information example**

```
[
  {
    "durationInMS": "37282",
    "trackEdits": [
      {
        "endInMs": "14",
        "type": "pause",
        "startInMs": "0"
      },
      {
        "endInMs": "37282",
        "type": "media",
        "startInMs": "14"
      }
    ],
    "channels": "2",
    "trackType": "AUDIO",
    "trackId": "1",
    "timescale": "44100",
    "streamName": "room-09f012-user1-e6ff",
    "trackCodec": "mp4a",
    "sampleRate": "44100",
    "mediaSessionId": "e6ff54e0-1c2a-11ec-90e8-79a2a32f3d9d"
  },
  {
    "durationInMS": "37336",
    "trackEdits": [
      {
        "endInMs": "37336",
        "type": "media",
        "startInMs": "0"
      }
    ],
    "trackType": "VIDEO",
    "trackId": "0",
    "width": "320",
    "timescale": "90000",
    "streamName": "room-09f012-user1-e6ff",
    "trackCodec": "avc1",
    "mediaSessionId": "e6ff54e0-1c2a-11ec-90e8-79a2a32f3d9d",
    "height": "240"
  },
  {
    "durationInMS": "39274",
    "trackEdits": [
      {
        "endInMs": "100",
        "type": "pause",
        "startInMs": "0"
      },
      {
        "endInMs": "21534",
        "type": "pause",
        "startInMs": "100"
      },
      {
        "endInMs": "39274",
        "type": "media",
        "startInMs": "21534"
      }
    ],
    "channels": "2",
    "trackType": "AUDIO",
    "trackId": "3",
    "timescale": "44100",
    "streamName": "room-09f012-user2-f746",
    "trackCodec": "mp4a",
    "sampleRate": "44100",
    "mediaSessionId": "f74633a1-1c2a-11ec-bba5-af8cf43275a8"
  },
  {
```

```

"durationInMS": "39303",
"trackEdits": [
  {
    "endInMs": "21434",
    "type": "pause",
    "startInMs": "0"
  },
  {
    "endInMs": "39303",
    "type": "media",
    "startInMs": "21434"
  }
],
"trackType": "VIDEO",
"trackId": "2",
"width": "320",
"timescale": "90000",
"streamName": "room-09f012-user2-f746",
"trackCodec": "avc1",
"mediaSessionId": "f74633a1-1c2a-11ec-bba5-af8cf43275a8",
"height": "240"
}
]

```

Здесь:

- durationInMS - длительность дорожки в миллисекундах
- trackType - тип дорожки: AUDIO или VIDEO
- trackId - идентификатор дорожки
- streamName - имя потока, к которому принадлежит эта дорожка
- mediaSessionId - идентификатор медиасессии потока
- timescale - количество семплов дорожки в секунду
- trackCodec - кодек дорожки
- width, height - размеры картинки для видеодорожки, по первому ключевому кадру
- channels - количество каналов для аудиодорожки
- sampleRate - частота дискретизации для аудиодорожки, совпадает с timescale
- trackEdits - описание временной шкалы дорожки

Временная шкала дорожки описывается как набор отрезков, построенный в соответствии с атомом MP4 'edit lists', со следующими параметрами:

- startInMs - время начала отрезка в миллисекундах относительно начала файла
- endInMs - время окончания отрезка в миллисекундах относительно начала файла
- type - тип отрезка: медиаданные (media) или пауза (pause)

По этим данным из файла можно извлечь нужную дорожку при помощи ffmpeg или другого инструмента редактирования MP4 файлов.

Отметим, что, если один и тот же поток был добавлен в рекордер, затем удален из рекордера, и потом снова добавлен, он будет представлен в файле различными дорожками с последовательными идентификаторами trackId, например:

#### Re-added stream track information example example

```

[
  {
    "durationInMS": "78978",
    "trackEdits": [
      {
        "endInMs": "63050",
        "type": "pause",
        "startInMs": "0"
      },
      {
        "endInMs": "78978",
        "type": "media",
        "startInMs": "63050"
      }
    ],
    "channels": "2",
    "trackType": "AUDIO",
    "trackId": "3",
    "timescale": "44100",
    "streamName": "test",

```

```

    "trackCodec": "mp4a",
    "sampleRate": "44100",
    "mediaSessionId": "fbbf5b50-20ee-11ec-bf06-ef6ec6048b2c"
  },
  {
    "durationInMS": "39708",
    "trackEdits": [
      {
        "endInMs": "23150",
        "type": "media",
        "startInMs": "0"
      }
    ],
    "channels": "2",
    "trackType": "AUDIO",
    "trackId": "1",
    "timescale": "44100",
    "streamName": "test",
    "trackCodec": "mp4a",
    "sampleRate": "44100",
    "mediaSessionId": "c7bc1460-20ee-11ec-bf06-ef6ec6048b2c"
  },
  {
    "durationInMS": "39791",
    "trackEdits": [
      {
        "endInMs": "23233",
        "type": "media",
        "startInMs": "0"
      }
    ],
    "trackType": "VIDEO",
    "trackId": "0",
    "width": "640",
    "timescale": "90000",
    "streamName": "test",
    "trackCodec": "avc1",
    "mediaSessionId": "c7bc1460-20ee-11ec-bf06-ef6ec6048b2c",
    "height": "360"
  },
  {
    "durationInMS": "63050",
    "trackEdits": [
      {
        "endInMs": "39791",
        "type": "pause",
        "startInMs": "0"
      },
      {
        "endInMs": "50191",
        "type": "media",
        "startInMs": "39791"
      }
    ],
    "trackType": "VIDEO",
    "trackId": "2",
    "width": "640",
    "timescale": "90000",
    "streamName": "test",
    "trackCodec": "avc1",
    "mediaSessionId": "ed3ebda0-20ee-11ec-bf06-ef6ec6048b2c",
    "height": "360"
  }
]

```

## Извлечение отдельных потоков из MKV контейнера

В сборке [5.2.1440](#) с помощью инструмента для микширования записанных потоков можно извлечь отдельные потоки из MKV контейнера:

```
./offline_mixer_tool.sh --pull-streams ../records/multi-recorder___test-record.mkv
```

При этом будут созданы MKV файлы для каждого из потоков:

```
multi-recorder___test-record_fbbf5b50-20ee-11ec-bf06-ef6ec6048b2c.mkv
multi-recorder___test-record_c7bc1460-20ee-11ec-bf06-ef6ec6048b2c.mkv
multi-recorder___test-record_ed3ebda0-20ee-11ec-bf06-ef6ec6048b2c.mkv
```

Если поток был удален из мультирекордера и повторно добавлен, или был добавлен позднее, чем другие потоки, получившиеся паузы по умолчанию будут заполнены, чтобы выровнять извлеченные потоки по времени. При необходимости, заполнение можно отключить

```
multi_recorder_mkv_fill_gaps=false
```

## Скрипт для обработки записанных файлов

По окончании записи нескольких потоков в один файл, запускается скрипт обработки, заданный настройкой

```
on_multiple_record_hook_script=on_multiple_record_hook.sh
```

По умолчанию, скрипт запускает `offline_mixer_tool.sh`, передавая ему на вход имя записанного файла.

Начиная со сборки [5.2.1023](#), скрипт `on_multiple_record_hook.sh` по умолчанию записывает в лог `/usr/local/FlashphonerWebCallServer/logs/multi-record.log` только результат обработки, чтобы снизить нагрузку на диск во время работы инструмента микширования. При необходимости, можно включить подробное логирование для отладки, установив переменную в скрипте

```
LOGGER_ENABLED=true
```

## Многопоточное кодирование при микшировании записанных потоков

В сборке [5.2.1089](#) добавлена возможность включить многопоточное кодирование при микшировании записанных потоков. Для этого в файл `/usr/local/FlashphonerWebCallServer/conf/offline_mixer.json` необходимо добавить параметр

```
{
  ...,
  "multithreading": true
}
```

При использовании многопоточного кодирования записанные потоки микшируются в среднем в два раза быстрее по сравнению с однопоточным.

## Количество процессорных потоков при многопоточном кодировании

В сборке [5.2.1523](#) добавлена настройка количества процессорных потоков, используемых для многопоточного кодирования. По умолчанию, количество процессорных потоков равно половине доступных системе ядер CPU. Например, на сервере с 12 CPU по умолчанию будут использоваться 6 потоков

```
{
  ...,
  "threadCount": 6
}
```

Если микширование записи занимает долгое время, значение можно увеличить, но не рекомендуется указывать больше, чем количество ядер CPU на сервере, которое можно определить при помощи команды

```
lscpu | grep -E "^CPU\(s\)"
```

## Отображение имени записанного потока

По умолчанию, в микшированной записи нескольких потоков отображается имя каждого потока. При необходимости, это можно отключить настройкой в файле `/usr/local/FlashphonerWebCallServer/conf/offline_mixer.json`

```
{
  ...,
  "mixerDisplayStreamName": false
}
```

При записи потоков в конференции с использованием RoomApi, имя потока включает имя комнаты конференции и идентификатор потока участника, например `room-1882a6-bob-037c`. В сборке [5.2.1642](#) добавлена возможность исключить имя комнаты при помощи настроек

```
{
  ...,
  "mixerDisplayStreamName": true,
  "mixerTextDisplayRoom": false,
  "labelReplaceRegex": "\\w+-\\w+-([\\-]+)-\\w+",
  "labelReplaceWith": ""
}
```

Здесь:

- `labelReplaceRegex` - регулярное выражение для замены элементов в имени потока
- `labelReplaceWith` - строка, которая должна заменить элементы, найденные по регулярному выражению, пустая строка исключает найденные элементы

В этом случае для указанного выше примера будет отображаться только имя участника `bob`.

## Декодирование символов в имени записанного потока

В сборке [5.2.1751](#) добавлена возможность декодирования символов в имени потока, закодированных на стороне клиента при помощи `encodeURIComponent()`

```
{
  ...,
  "mixerDecodeStreamName": true
}
```

В этом случае в изображении будут отображаться декодированные символы, если такие символы есть в используемом шрифте, или близкие к ним по начертанию.

## Отправка данных о завершении записи нескольких потоков

В сборке [5.2.1123](#) добавлена возможность отправки POST запроса на указанный URL при завершении записи нескольких потоков в файл и микширования этой записи. Таким образом, можно получить информацию о том, в какой файл смикшированы записанные потоки из чат-комнаты.

URL для отправки запроса задается в файле `/usr/local/FlashphonerWebCallServer/conf/offline_mixer.json`:

```
{
  ...,
  "callbackUrl": "http://backend.url/multiRecorderCallback"
}
```

Данные для отправки передаются через скрипт `/usr/local/FlashphonerWebCallServer/bin/on_multiple_record_hook.sh` при вызове `offline_mixer_tool.sh`. Поэтому при установке сборки [5.2.1123](#) поверх предыдущей, или в том случае, если используется собственный скрипт `on_multiple_record_hook.sh`, необходимо модифицировать его следующим образом:

#### on\_multiple\_record\_hook.sh

```
# This script copies a recorded stream to client/records
FILE_NAME=$1
CREATION_MODIFICATION_TIME=$2
DURATION_IN_MS=$3
RECORDER_URI=$4
WCS_HOME=/usr/local/FlashphonerWebCallServer
LOG_FILE=$WCS_HOME/logs/multi-record.log
MIXER_TOOL=$WCS_HOME/tools/offline_mixer_tool.sh
# Set LOGGER_ENABLED to true to enable mixing debug logging
LOGGER_ENABLED=false

echo "[$(date '+%Y-%m-%d %H:%M:%S')] Start mixing multiple recording file $FILE_NAME" >> $LOG_FILE

if $LOGGER_ENABLED; then
    bash $MIXER_TOOL $FILE_NAME $CREATION_MODIFICATION_TIME $DURATION_IN_MS $RECORDER_URI >> $LOG_FILE 2>&1
else
    bash $MIXER_TOOL $FILE_NAME $CREATION_MODIFICATION_TIME $DURATION_IN_MS $RECORDER_URI > /dev/null 2>&1
fi

CODE=$?
if [ "$CODE" -ne "0" ]; then
    if [ "$CODE" -eq "64" ]; then
        echo "ERROR: File to mix not found" >> $LOG_FILE
    elif [ "$CODE" -eq "65" ]; then
        echo "ERROR: Offline mixer config not found" >> $LOG_FILE
    else
        echo "ERROR: Offline mixer tool error code: $CODE" >> $LOG_FILE
    fi
    exit $CODE
fi
echo "[$(date '+%Y-%m-%d %H:%M:%S')] Multiple recording file $FILE_NAME is mixed successfully" >> $LOG_FILE
exit 0
```

POST запрос содержит данные в формате JSON:

```
POST /multiRecorderCallback HTTP/1.1
Content-Type: application/json
Content-Length: 463
Host: localhost
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.3.5 (java 1.5)
Accept-Encoding: gzip,deflate

{
  "multiRecorderCreationModificationTime":3724973476,
  "multiRecorderDurationInMS":44061,
  "multiRecorderFilePath":"/usr/local/FlashphonerWebCallServer/multirecords/stream-32c7edd7-37bf-4bf2-a58d-955679c5287e-mockLogin.mp4",
  "recorderUri":"multi-recorder://room-bace1f",
  "mixerParams":
  [
    {
      "path":"/usr/local/FlashphonerWebCallServer/multirecords/stream-32c7edd7-37bf-4bf2-a58d-955679c5287e-mockLogin_mixed.mp4",
      "durationInMs":44000,
      "creationModificationTime":3724973524
    }
  ]
}
```

В запросе передаются параметры исходного файла записи нескольких потоков:

- multiRecorderCreationModificationTime - время создания файла записи
- multiRecorderDurationInMS - длительность файла записи в миллисекундах
- multiRecorderFilePath - путь к файлу записи

- recorderUri - идентификатор рекордера, при использовании RoomApi содержит имя комнаты

Параметры микшированного файла:

- path - путь к микшированному файлу
- durationInMs - длительность микшированного файла в миллисекундах
- creationModificationTime - время создания микшированного файла

## Контроль свободного места при микшировании записи нескольких потоков

В сборке [5.2.1317](#) добавлен [контроль свободного места](#) при микшировании записи нескольких потоков. Если места на диске остается меньше заданного, микширование не начнется или остановится. Значение задается настройкой в файле /usr/local/FlashphonerWebCallServer/conf/offline\_mixer.json

```
{
    ...,
    "minAvailableSpace": "1G"
}
```

По умолчанию, ограничение свободного места установлено в 1 Гб (так же, как для записей с одним потоком). Если значение достигнуто в момент, когда микширование уже работает, то микширование будет остановлено с сохранением того, что удалось записать. Полученный файл может быть нормально проигран после этого.

## Контроль добавления потока в рекордер на бэкэнд сервере

В сборке [5.2.1416](#) добавлена возможность получения событий, сигнализирующих о том, что поток добавлен или удален из рекордера для записи нескольких потоков. Для этого WCS отправляет на бэкэнд сервер запрос /StreamEvent

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
ОБЪЕКТ:
{
  "nodeId" : "d2hxbqNPE04vGeZ51NPhDuId6k3hUrBB@192.168.1.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.1.83:49977/192.168.1.39:8443-591009c4-e051-4722-b34d-71cf2ade3bed",
  "mediaSessionId" : "15de2290-4089-11ed-88fe-d78a87cf3386",
  "type" : "addedToMultiRecording",
  "payload" : {
    "fileName" : "stream-0389ff08-7e45-4f00-a579-9d253319cba4-mockLogin.mp4",
    "uri" : "multi-recorder://test-record"
  }
}
```

при добавлении потока и

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
ОБЪЕКТ:
{
  "nodeId" : "d2hxbqNPE04vGeZ51NPhDuId6k3hUrBB@192.168.1.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.1.83:49977/192.168.1.39:8443-591009c4-e051-4722-b34d-71cf2ade3bed",
  "mediaSessionId" : "15de2290-4089-11ed-88fe-d78a87cf3386",
  "type" : "removedFromMultiRecording",
  "payload" : {
    "fileName" : "stream-0389ff08-7e45-4f00-a579-9d253319cba4-mockLogin.mp4",
    "uri" : "multi-recorder://test-record"
  }
}
```

при его удалении из рекордера.

При обновлении WCS с предыдущих сборок в конфигурацию [бэкэнд приложения](#) необходимо [добавить](#) метод StreamEvent

```
add app-rest-method defaultApp StreamEvent
add app-rest-method MyAppKey StreamEvent
```

# Известные проблемы

1. Максимальная длина имени файла во всех актуальных файловых системах Linux ограничена 255 символами. При создании файла записи, имя будет сокращено до данного предела, включая расширение и номер части, если включена ротация.

2. При записи потоков, опубликованных в конференции, ротация будет автоматически отключена, в противном случае полученные файлы будет невозможно объединить.

3. Дата создания файла записывается в метаданные только в контейнере MP4.

4. Скрипт обработки записанных файлов требует повышения прав для копирования и других операций над файлами записей на виртуальных машинах Amazon

Симптомы: операции над записанными файлами не выполняются

Решение: использовать `sudo` для файловых операций и вызова внешних скриптов, если WCS установлен на виртуальной машине Amazon, например

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

5. На серверах небольшой мощности, запись двухканального звука приводит к росту нагрузки на процессор и к задержкам при одновременной записи нескольких потоков

Симптомы: при одновременной записи нескольких потоков, загрузка всех процессорных ядер достигает 100%, при завершении публикации потоков запись завершается с большой задержкой

Решение: отключить запись двухканального звука

```
record_audio_codec_channels=1
```

6. При публикации H264 потока из браузера Firefox на некоторых Android устройствах может портиться запись, при нормальном проигрывании этого же потока по WebRTC

Симптомы: при публикации потока из Android Firefox запись не проигрывается либо испорчена, файл имеет малый размер

Решение:

- a) использовать VP8 для публикации потока из Android Firefox
- b) использовать на этом устройстве Chrome или другой браузер для публикации

7. Некоторые Android устройства публикуют WebRTC H264 поток с профилем High, даже если этого профиля нет в SDP при установке WebRTC соединения

Симптомы: в данных файла записи опубликованного потока отображается профиль High

Решение: если возникают проблемы при проигрывании записей потоков, опубликованных с профилем High, перекодировать эти записи, например, при помощи `ffmpeg`, запуская постобработку скриптом `on_record_hook.sh`

8. Первая запись после запуска сервера может быть повреждена, если Java машина не успевает инициализировать необходимые модули

Симптомы: длительный фриз в начале первой записи после запуска сервера

Решение:

- a) обновить WCS до сборки [5.2.1105](#)
- b) если используется сборка [5.2.1105](#) и новее, убедиться, что в настройках включена предварительная инициализацию модулей WebRTC-стека при старте сервера

```
webrtc_pre_init=true
```

9. Записи в контейнере webm не проигрываются в iOS Safari

Симптомы: при щелчке по ссылке на файл записи начинается загрузка файла, а не воспроизведение

Решение: скачать запись на устройство и проиграть локальным плеером

10. При публикации RTMP потока только с аудио при настройках по умолчанию запись не создается

Симптомы: при публикации на WCS RTMP аудио потока такой поток не записывается

Решение: использовать [файл настройки SDP](#) `flash_handler_publish.sdp` без видео составляющей

```

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=audio 0 RTP/AVP 97 8 0 102 103 104 105 106 107 108 109 110
a=rtpmap:97 SPEEX/16000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:102 mpeg4-generic/48000/1
a=rtpmap:103 mpeg4-generic/44100/1
a=rtpmap:104 mpeg4-generic/32000/1
a=rtpmap:105 mpeg4-generic/24000/1
a=rtpmap:106 mpeg4-generic/22050/1
a=rtpmap:107 mpeg4-generic/16000/1
a=rtpmap:108 mpeg4-generic/12000/1
a=rtpmap:109 mpeg4-generic/11025/1
a=rtpmap:110 mpeg4-generic/8000/1
a=sendonly

```

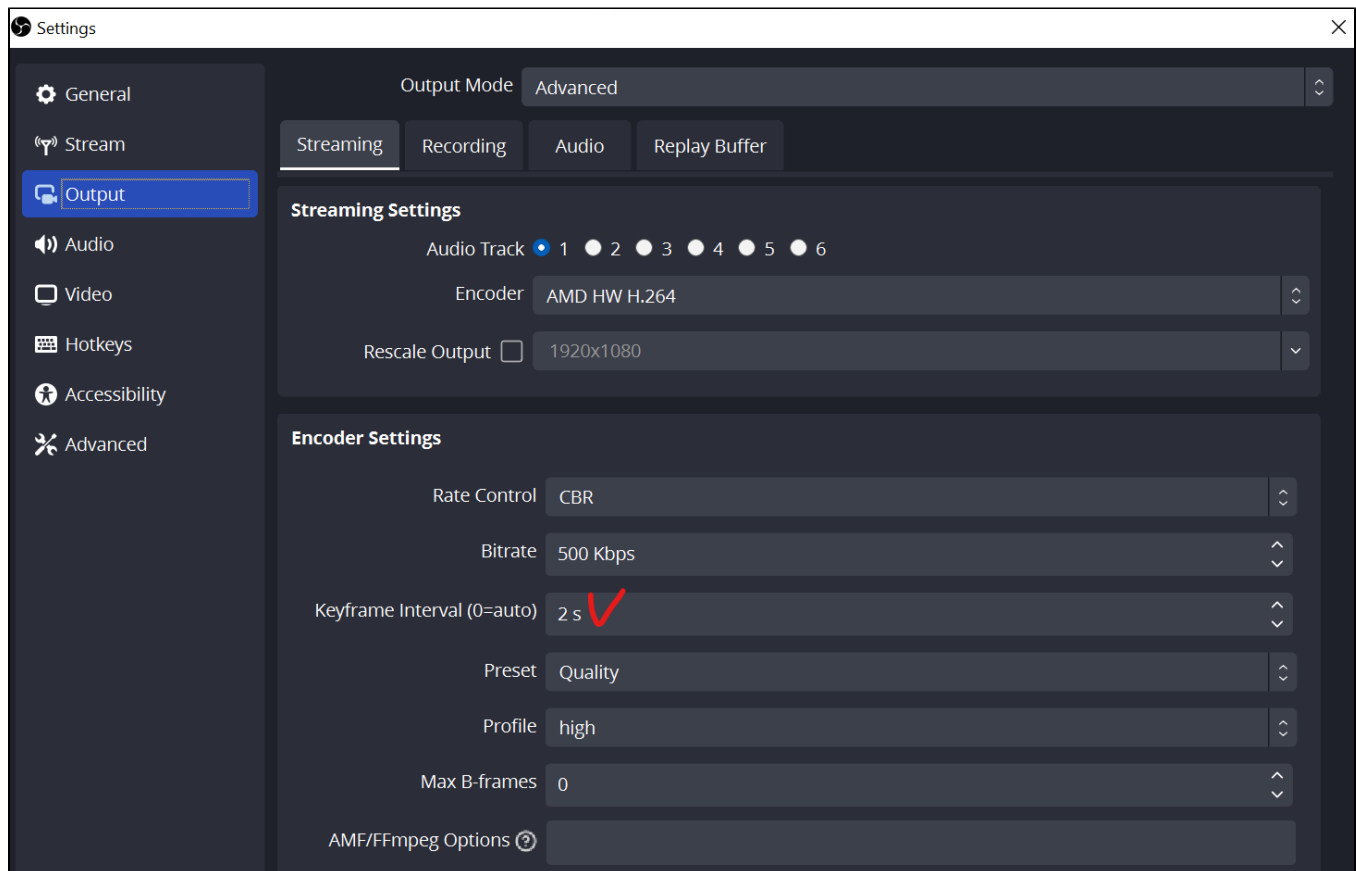
11. Запись потока начинается только после получения хотя бы одного ключевого кадра

Симптомы: при старте записи (например, по REST API) файл не создается, ошибок в серверном логе при этом нет, и поток играет в браузере (до переподключения зрителя)

Решение: обеспечить периодическую отсылку ключевых кадров для WebRTC публикаций при помощи настройки

```
periodic_fir_request=true
```

для RTMP публикаций соответствующими настройками кодировщика



12. При воспроизведении потока, извлеченного из записи мультирекордера в формате MKV, проигрыватель проскакивает паузы в потоке

Симптомы: при проигрывании извлеченного потока в VLC, пауза в потоке пропускается

Решение: использовать настройку

```
multi_recorder_mkv_fill_gaps=true
```

13. Если WebRTC поток записывается одновременно в отдельный файл MKV и в мультирекордер MKV, рекомендуется обеспечить регулярное поступление ключевых фреймов

Симптомы: при проигрывании отдельного файла MKV VLC перескакивает на время добавление этого потока в мультирекордер

Решение: обеспечить периодическую отсылку ключевых кадров для WebRTC публикаций при помощи настройки

```
periodic_fir_request=true
```

14. При извлечении из файла мультирекордера в контейнере MKV потока с аудио G722 могут наблюдаться кратковременные искажения аудио

Симптомы: при проигрывании извлеченного файла с аудио G722, если поток был удален и снова добавлен в мультирекордер, слышны кратковременные искажения звука при добавлении потока

Решение: использовать аудиокодек Opus

15. При извлечении из файла мультирекордера в контейнере MKV потока с аудио PCMA после удаления потока из мультирекордера аудио завершается раньше, чем видео

Симптомы: при проигрывании извлеченного файла с аудио PCMA, если поток был удален из мультирекордера, аудио завершается раньше, чем видео

Решение: использовать аудиокодек Opus