С IP камеры по RTSP

Описание

Видеопоток захватывается с RTSP-источника, отдающего аудио и видео в поддерживаемых кодеках. Далее видеопоток трансформируется на стороне сервера для воспроизведения в браузерах и мобильных устройствах.

RTSP-источники

- IP камеры
- Медиасерверы
- Системы наблюдения
- Конференц-серверы

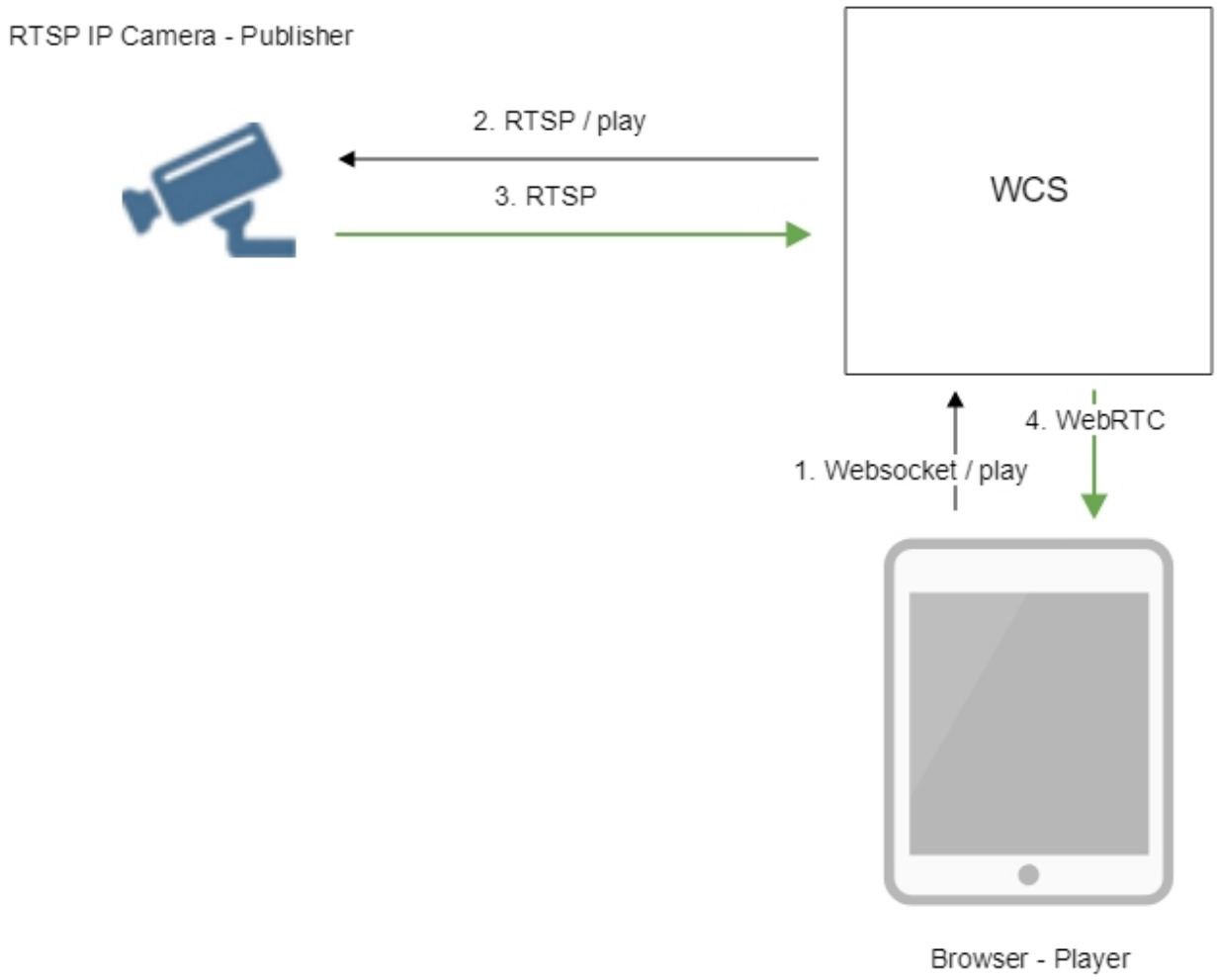
Поддерживаемые кодеки

- H.264
- VP8
- H265 (начиная со сборки 5.2.1579)
- AAC
- G.711
- Speex

Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari 11	Internet Explorer	Edge
Windows	+	+		+	+
Mac OS	+	+	+		
Android	+	+			
iOS	-	-	+		

Схема работы



1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду play.
2. Сервер соединяется с RTSP-источником и отправляет команду play.
3. RTSP-источник передает на сервер RTSP-поток.
4. Сервер трансформирует поток в WebRTC и отдает поток браузеру.

Настройка

Привязка RTSP клиента к определенному адресу

В некоторых случаях, например, если подключение к IP-камере для захвата потока по RTSP производится через VPN, RTSP-клиент должен быть привязан к определенному адресу. Адрес указывается при помощи опции rtsp_client_address в файле настроекflashphoner.properties, например

```
rtsp_client_address=172.16.0.3
```

Захват RTSP потока по UDP

По умолчанию, RTSP потоки захватываются по TCP. При необходимости, можно переключиться на захват потоков по UDP при помощи параметра

```
rtsp_interleaved_mode=false
```

Выбор аудио и видео каналов в RTSP потоке

По умолчанию, аудио и видео каналы в RTSP потоке выбираются динамически, в соответствии с SDP, полученным от камеры. При необходимости, выбор каналов может быть указан принудительно при помощи параметра, например

```
rtsp_interleaved_channels=2-3;0-1
```

Здесь

- 2-3 - каналы аудиопотока
- 0-1 - каналы видеопотока

Проигрывание потока в AnnexB формате

Некоторые IP камеры, например,Honeywell MAXPRO Video Streamer, публикуют поток H264 в AnnexB формате. Для проигрывания видео с таких камер в сборке5.2.636добавлен параметр, позволяющий включить обработку такого формата

```
h264_check_and_skip_annexb=true
```

Начиная со сборки5.2.946,данная опция удалена из настроек, и фреймы в AnnexB формате определяются и проигрываются автоматически.

Исключение аудио кодеков

В некоторых случаях, необходимо проиграть поток с камеры без аудио, либо исключить обработку аудио в некоторых кодеках, чтобы не транскодировать звук. Для этого предназначена настройка, перечисляющая кодеки, которые должны быть исключены при захвате потока с камер, например

```
rtsp_client_strip_audio_codecs=PCMA,PCMU
```

Данная настройка исключает кодеки PCMA (alaw) и PCMU (ulaw). Видео с камер, передающих звук в этих кодеках, будет проигрываться без аудио.

Кодеки исключаются на уровне SDP, по названиям.

Установка режима пакетизации H264

По умолчанию, согласно спецификации H264, если в SDP потока не указан явно режим пакетизации, он присваивается равным 0. Однако, некоторые RTSP камеры могут передавать поток, закодированный в режиме пакетизации 1, и при этом не указывать режим в SDP потока. Это приводит к включению транскодинга и снижению качества картинки в браузере Safari.

В сборке5.2.820добавлена возможность установить режим пакетизации по умолчанию для использования таких камер при помощи настройки

```
default_packetization_mode=1
```

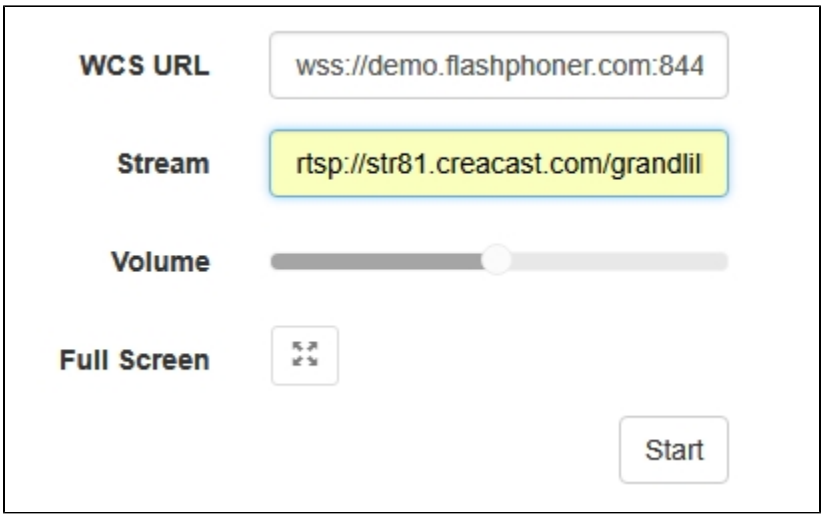
Краткое руководство по тестированию

Захват видеопотока с IP-камеры и трансляция в браузер

1. Для теста используем:

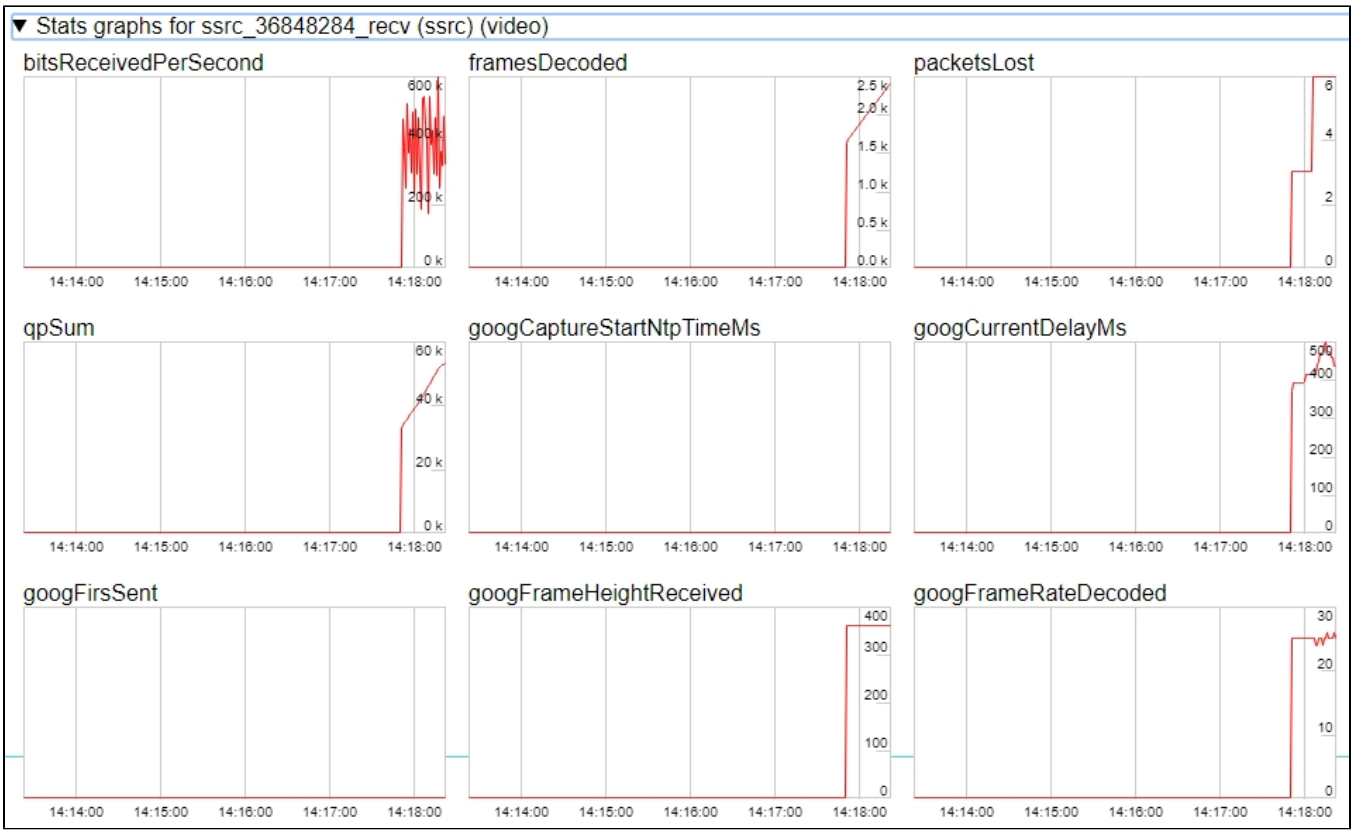
- демо-серверdemo.flashphoner.com;
- веб-приложениеPlayerдля захвата и воспроизведения потока

2. Откройте веб-приложение Player, укажите в поле "Stream" URL веб-камеры:



3. Нажмите кнопку "Start". Начнется трансляция захваченного потока.

4. Графики WebRTC internals:



Управление захватом видеопотока с IP-камеры при помощи REST API

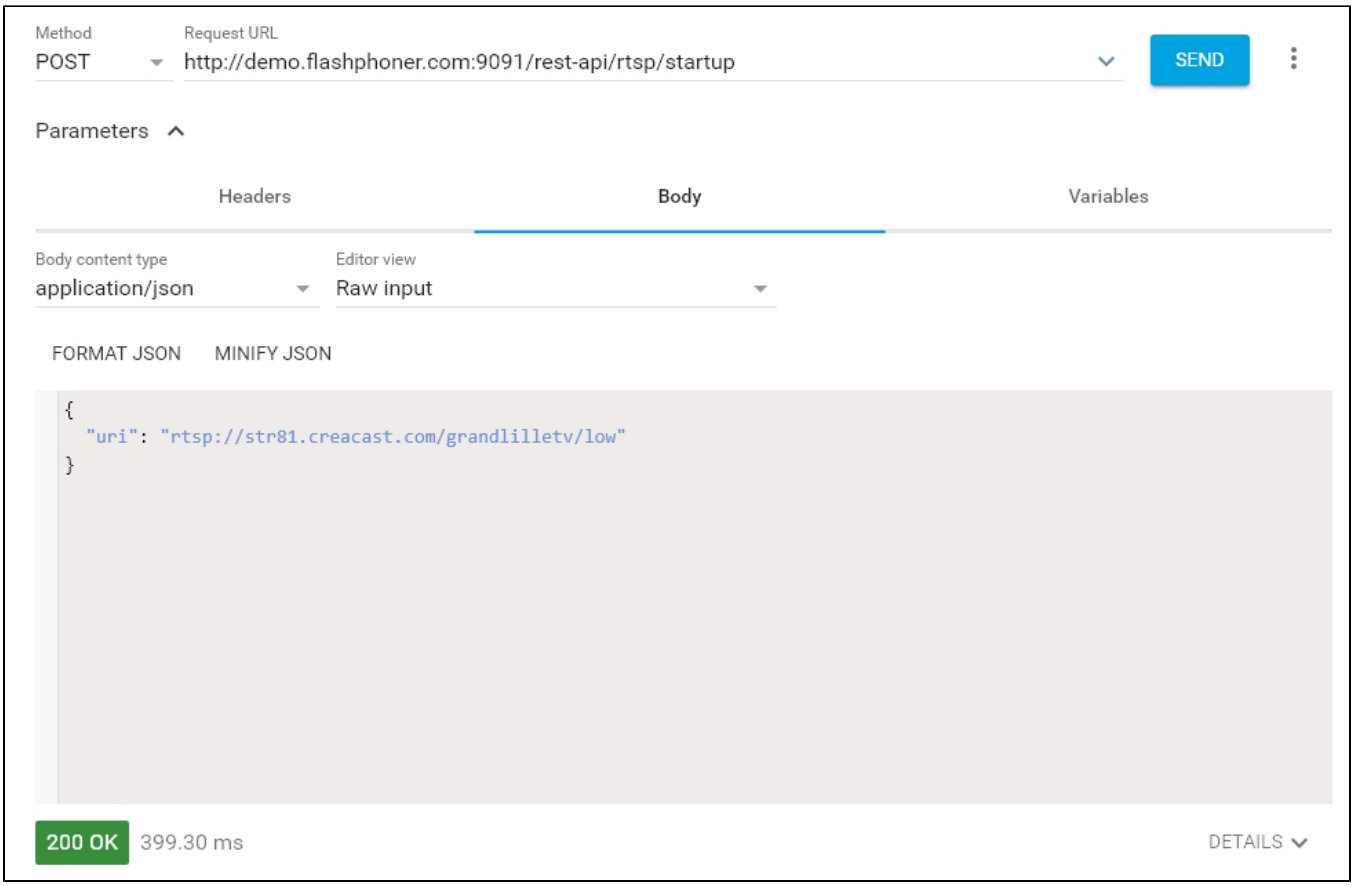
Как правило, для захвата потока с IP-камеры достаточно указать URL-камеры в качестве имени потока при его создании. Однако, при необходимости, возможно управлять захватом RTSP-потока при помощи REST API.

Тестирование

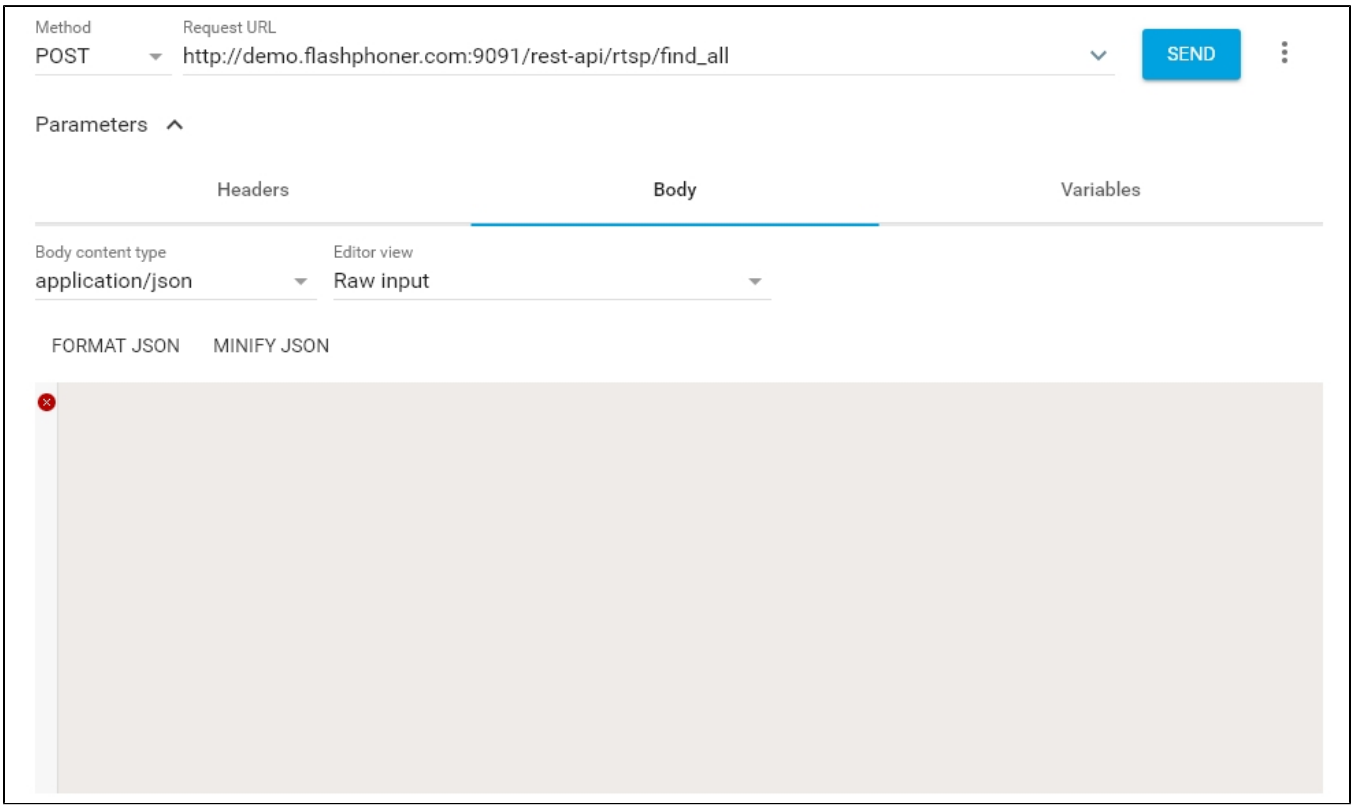
1. Для теста используем:

- демо-сервер demo.flashphoner.com;
- браузер Chrome иREST-клиентдля отправки запросов на сервер;
- веб-приложениеPlayerдля воспроизведения захваченного потока в браузере.

2. Откройте REST-клиент. Отправьте запрос /rtsp/startup, указав в параметрах URL веб-камеры:



3. Убедитесь, что поток захвачен сервером. Для этого отправьте запрос /rtsp/find_all:

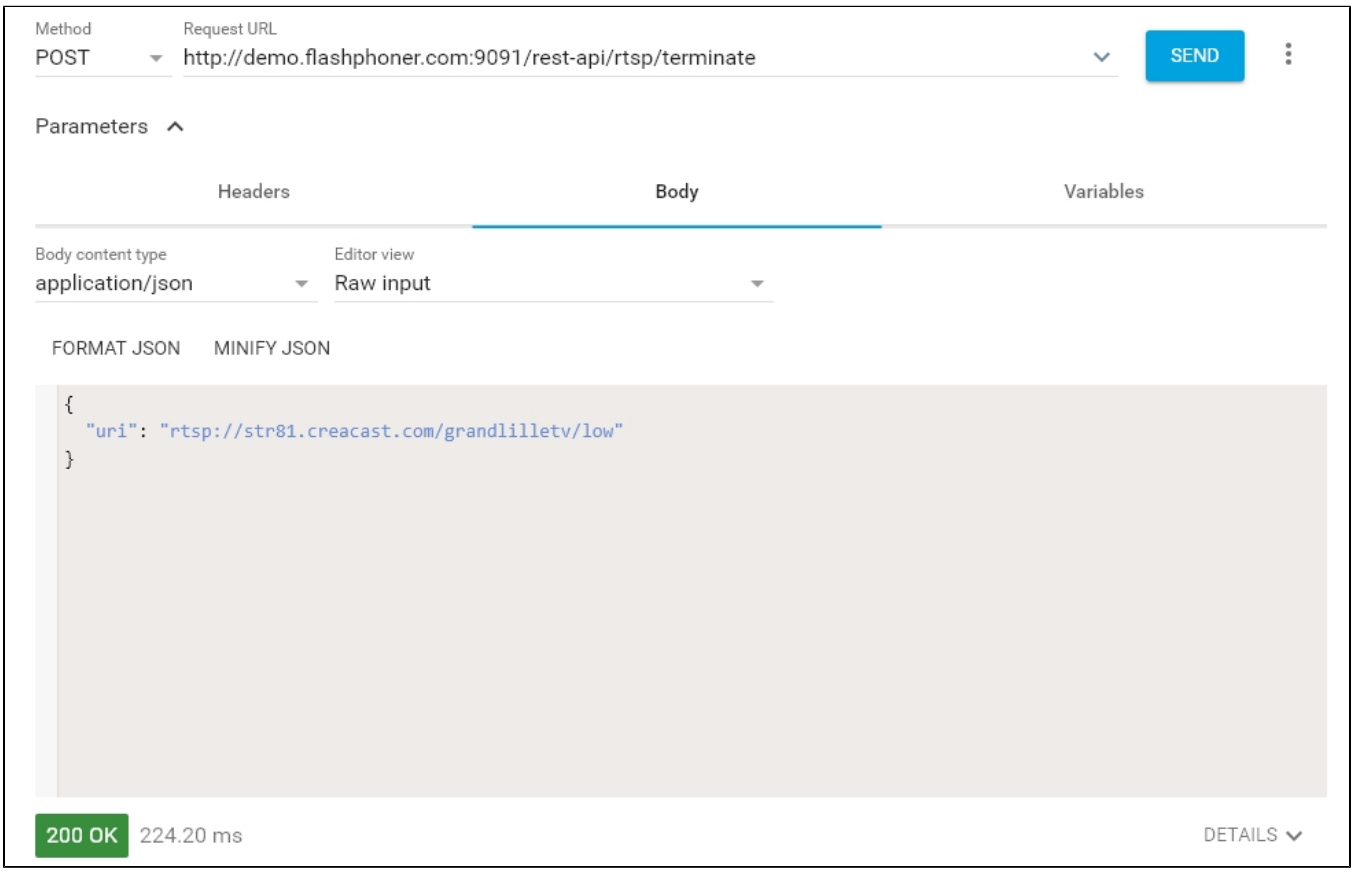


200 OK 223.00 ms DETAILS

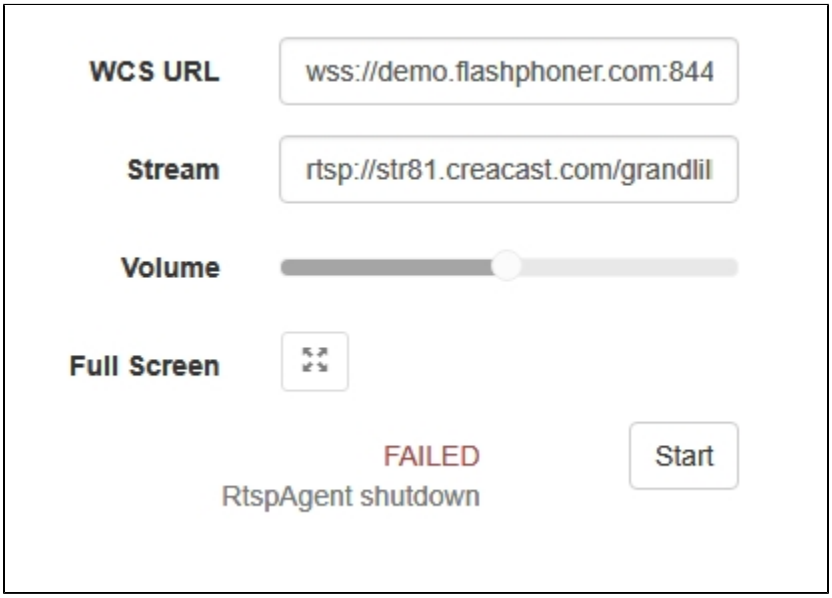

```
[Array[6]
  -0:  {
       "uri": "rtsp://65.151.173.44:1935/kosova/kamera4.stream",
       "status": "PLAYING"
    },
  -1:  {
       "uri": "rtsp://65.151.173.44:1935/kosova/kamera5.stream",
       "status": "PLAYING"
    },
  -2:  {
       "uri": "rtsp://65.151.173.44:1935/kosova/kamera2.stream",
       "status": "PLAYING"
    },
  -3:  {
       "uri": "rtsp://65.151.173.44:1935/kosova/kamera3.stream",
       "status": "PLAYING"
    },
  -4:  {
       "uri": "rtsp://65.151.173.44:1935/kosova/kamera1.stream",
       "status": "PLAYING"
    },
  -5:  {
       "uri": "rtsp://str81.creacast.com/grandlilletv/low",
       "status": "PLAYING"
    }
],
```


4. Откройте веб-приложение Player, укажите в поле "Stream" URL веб-камеры и нажмите Start. Начнется воспроизведение потока в браузере:

5. Отправьте запрос /rtsp/terminate, указав в параметрах URL веб-камеры:



6. Воспроизведение потока прервется с ошибкой:



REST-вызовы

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

- HTTP:http://test.flashphoner.com:8081/rest-api/rtsp/startup

- HTTPS:https://test.flashphoner.com:8444/rest-api/rtsp/startup

Здесь:

- test.flashphoner.com - адрес WCS-сервера
- 8081 - стандартный REST / HTTP порт WCS-сервера
- 8444 - стандартный HTTPS порт
- rest-api - обязательная часть URL
- /rtsp/startup - используемый REST-метод

REST-методы и статусы ответа

REST-метод	Пример тела REST-запроса	Пример тела REST-ответа	Статусы ответа	Описание
/rtsp /startup	`{ "uri":"rtsp://myserver.com /live/myStream", "localStreamName": "myRTSPstream" }`		409 - Conflict 500 - Internal error	Извлечь RTSP-поток по указанному URL
/rtsp /find_all		`{ "uri": "rtsp://myserver.com /live/myStream", "status": "PLAYING", "localStreamName": "myRTSPstream" }`	200 – потоки найдены 404 – потоки не найдены	Найти все извлеченные RTSP-потоки
/rtsp /terminate	`{ "uri":"rtsp://myserver.com /live/myStream" }`		200 - поток завершен 404 - поток не найден	Завершить извлеченный RTSP-поток

Параметры

Имя параметра	Описание	Пример
uri	URL RTSP-потока	rtsp://myserver.com/live/myStream
localStreamName	Имя, которое будет присвоено извлеченному потоку	myRTSPstream
status	Текущий статус потока	PLAYING

Повторный захват потока с тем же URI

При попытке повторного захвата потока с тем же URI запрос /rtsp/startup вернет 409 Conflict.Если поток с камеры уже опубликован на сервере, необходимо подключаться к этому потоку.

Последовательность выполнения операций (Call Flow)

Ниже описана последовательность вызовов при использовании примера Player

player.html

player.js

1. Установка соединения с сервером.

Flashphoner.createSession();code

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
        setStatus(session.status());
        //session connected, start playback
        playStream(session);
    }).on(SESSION_STATUS.DISCONNECTED, function(){
        setStatus(SESSION_STATUS.DISCONNECTED);
        onStopped();
    }).on(SESSION_STATUS.FAILED, function(){
        setStatus(SESSION_STATUS.FAILED);
        onStopped();
    });
```

2. Получение от сервера события, подтверждающего успешное соединение.

ConnectionStatusEvent ESTABLISHEDcode

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
        setStatus(session.status());
        //session connected, start playback
        playStream(session);
    }).on(SESSION_STATUS.DISCONNECTED, function(){
        ...
    }).on(SESSION_STATUS.FAILED, function(){
        ...
    });
```

3. Запрос на воспроизведение потока.

session.createStream(), stream.play();code

URL IP-камеры передается в метод createStream() как имя потока

```
function playStream(session) {
    var streamName = $('#streamName').val();
    var options = {
        name: streamName,
        display: remoteVideo,
        flashShowFullScreenButton: true
    };
    ...
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    });
    stream.play();
}
```

4. Запрос от WCS к RTSP-источнику на трансляцию потока.

5. Трансляция RTSP-потока

6. Получение от сервера события, подтверждающего успешный захват и проигрывание потока.

StreamStatusEvent, статус PLAYINGcode

```
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function(stream) {
        $("#preloader").show();
        setStatus(stream.status());
        onStarted(stream);
        ...
    });
    stream.play();
```

7. Отправка аудио-видео потока по WebRTC

8. Остановка воспроизведения потока.

stream.stop();code

```
function onStarted(stream) {
    $("#playBtn").text("Stop").off('click').click(function(){
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
    $("#fullScreenBtn").off('click').click(function(){
       stream.fullScreen();
    }).prop('disabled', false);
    $("#volumeControl").slider("enable");
    stream.setVolume(currentVolumeValue);
}
```

9. Получение от сервера события, подтверждающего остановку воспроизведения потока.

StreamStatusEvent, статус STOPPEDcode

```
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function(stream) {
        ...
    }).on(STREAM_STATUS.STOPPED, function() {
        setStatus(STREAM_STATUS.STOPPED);
        onStopped();
    }).on(STREAM_STATUS.FAILED, function(stream) {
        ...
    }).on(STREAM_STATUS.NOT_ENOUGH_BANDWIDTH, function(stream){
        ...
    });
    stream.play();
```

Повторное использование подключения к камере

Если к потоку, захваченному с RTSP IP-камеры, присоединяются другие подписчики, будет использовано ранее установленное подключение к камере, при условии, что все подписчики указали одинаковый адрес камеры. Например, запросы к одной и той же камере

```
rtsp://host:554/live.sdp
```

и

```
rtsp://host:554/live.sdp?p=1
```

отличаются, поэтому будет создано два RTSP-подключения, если запросить оба этих потока.

Аутентификация при захвате потока

WCS поддерживает аутентификацию по имени и паролю при захвате RTSP-потока. данные пользователя должны быть указаны в URL потока, например

```
rtsp://user:password@hostname/stream
```

Если в имени или пароле есть какие-либо спецсимволы, они должны быть экранированы. например

```
rtsp://user:p%40ssword@hostname/stream
```

Здесь

- user - имя пользователя
- p@ssword - пароль, символ '@' экранирован при указании URL.

Обработка перенаправления на другой IP-адрес

Некоторые IP-камеры возвращают 302 Moved Temporarily в ответ на запрос DESCRIBE или OPTIONS и перенаправляют клиента на другой адрес для установки RTSP-соединения. WCS поддерживает данную возможность, начиная со сборки 5.2.179.

При этом, если камера перенаправляет запросы на другой адрес, и если подключиться отдельно к этой камере и непосредственно к камере, куда производится перенаправление, с точки зрения WCS это будут два различных потока. Для каждого из этих потоков создается свой агент захвата, и подписчики присоединяются к тому или другому агенту в зависимости от того, какой адрес они указывают при подключении.

Публикация захваченного RTSP потока под заданным именем

В сборке5.2.479добавлена возможность опубликовать захваченный RTSP поток на сервере под заданным именем. Имя должно быть указано параметром `toStream` REST запроса /rtsp/startup, например

```
POST /rest-api/rtsp/startup HTTP/1.1
Content-Length: 75
Content-Type: application/json

{
    "toStream": "stream1",
    "uri": "rtsp://myserver.com/live/myStream"
}
```

По умолчанию, если параметр `toStream` не указан, имя формируется из URI потока.Если RTSP поток с таким URI уже был захвачен, или на сервере существует поток с указанным именем, в ответ на запрос сервер вернет 409 Conflict.

Если захваченному RSTP потоку назначено имя,этот поток может быть воспроизведен по имени в CDN (по умолчанию, для RTSP потоков эта функция недоступна, т.к. они захватываются локально).

Захват H265 RTSP потока

В сборке 5.2.1579 добавлена возможность захвата RTSP потока, публикуемого камерой в кодеке H265. Для этого H265 должен быть добавленв список поддерживаемых кодеков

```
codecs=opus,alaw,ulaw,g729,speex16,g722,mpeg4-generic,telephone-event,h264,vp8,flv,mpv,h265
```

и в списки исключений

```
codecs_exclude_sip=mpeg4-generic,flv,mpv,h265
codecs_exclude_sip_rtmp=opus,g729,g722,mpeg4-generic,vp8,mpv,h265
codecs_exclude_sfu=alaw,ulaw,g729,speex16,g722,mpeg4-generic,telephone-event,flv,mpv,h265
```

Захваченный поток может быть проигран как WebRTC, RTMP, MSE, HLS с транскодингом и как RTSP без транскодинга

Поток не должен содержать B-фреймы! Если в потоке есть B-фреймы, его можно проигрывать только по RTSP без транскодинга

Проблема первого подписчика

До сборки 5.2.1760 RTSP потоки могли долго начинать играть у первого подписчика. Это было вызвано тем, что процесс подписчика стартовал позде процесса публикации, и мог пропускать первые ключевые кадры. В сборке 5.2.1760 это поведение изменено: процесс публикации стартует после процесса подписчика. При необходимости, можно вернуться к старому варианту при помощи настройки

```
agent_use_subscriber_listener=false
```

Исправление временных меток в потоке

В некоторых RTSP потоках временные метки могут идти не в нужном порядке, например, у двух кадров подряд может быть одинаковая метка. При проигрывании такого потока по WebRTC поток может долго не отображаться, и периодически давать серый фон. Для исправления таких временных меток, в сборке 5.2.1794 добавлена настройка

```
jitter_buffer_attempt_to_correct_broken_timestamp=true
```

В этом случае в клиентском логе будут сообщения

```
Non-monotonous timestamp in input stream; previous: 453424, current: 453424; changing to 453425. This may
result in incorrect timestamps in the output
```

и проблемный поток будет играть нормально.

Известные проблемы

1. Поток, содержащий B-фреймы, не воспроизводится либо воспроизводится с артефактами (задержки, подергивания)

Симптомы:

- поток не проигрывается, дает задержки видео или подергивания
- предупреждения вклиентском логе:

```
09:32:31,238 WARN 4BitstreamNormalizer - RTMP-pool-10-thread-5 It is B-frame!
```

Решение:

- изменить настройки кодировщика таким образом, чтобы исключить использование B-фреймов (понизить профиль кодирования, указать в командной строке и т.п.).
- транскодировать поток, в этом случае в выходном потоке транскодера B-фреймов не будет

2. AAC фреймы типа 0 не поддерживаются декодером FFmpeg и будут игнорироваться при воспроизведении захваченного потока

При этом вклиентском логебудут выведены предупреждения:

```
10:13:06,815 WARN AAC - AudioProcessor-c6c22de8-a129-43b2-bf67-1f433a814ba9 Dropping AAC frame that starts with
0, 119056e500
```

Решение: использовать кодек Fraunhofer при помощи настройки в файлеflashphoner.properties

```
use_fdk_aac=true
```

3. При публикации и последующем воспроизведении и записи H264 + AAC потока возможна рассинхронизация видео и звука, либо полное отсутствие звука.

Симптомы: при воспроизведении H264 + AACпотока, опубликованного на сервере, а также в записи потока, звук не синхронизирован с видео или отсутствует

Решение:

а) установить настройку в файлеflashphoner.properties

```
disable_drop_aac_frame=true
```

Эта настройка, в том числе, отключает игнорирование AAC фреймов.

б) использовать кодек Fraunhofer при помощи настройки

```
use_fdk_aac=true
```

4. При преобразовании звуковой дорожки AAC к частоте дискретизации 11025 Гц звук искажен или отсутствует

Симптомы: при публикации H264 + AAC потока на WCS сервере и воспроизведении его как H264 + AAC с частотой дискретизации звука 11025 Гц звук искажен или отсутствует

Решение: не использовать частоту дискретизации звука 11025 Гц, либо избегать преобразования звука к данной частоте, например, не указывать данную частоту вфайлахнастроек SDP.

5. Соединение с IP-камерой разрывается при ошибке в любом из треков (аудио или видео)

Симптомы: соединение с IP-камерой разрывается, если один из треков вернул ошибку 4**.
Решение: данное поведение включено по умолчанию. Однако, если единичные ошибки не являются критичными и не требуют прекращения трансляции, в файлеflashphoner.propertiesнеобходимо указать

```
rtsp_fail_on_error_track=false
rtp_force_synchronization=true
```

6. Символы в имени потока, недопустимые в URI, должны быть экранированы

Симптомы: RTSP-поток не воспроизводится с признаком ошибки 'Bad URI'
Решение: любые символы, недопустимые при указании URI, должны быть экранированы в имени потока, например

```
rtsp://hostname/c@@lstream/channel1
```

должен быть записан как

```
rtsp://hostname/c%40%40lstream/channel1
```

7. Некоторые камеры не поддерживают полеcnonceв заголовке сообщения при установке RTSP-соединения.

Симптомы: RTSP-поток играется в VLC, но не играется в WCS.

Решение: в файлеflashphoner.propertiesустановить настройку

```
rtsp_auth_cnonce=
```

с пустым значением.

8. Поток с некоторых камер не играет из-за нехватки буфера для записи RBSP

Симптомы: RTSP поток не играет, в серверном логе исключение

```
13:10:16,988 ERROR  BitstreamNormalizer - pool-56-thread-1 Failed to normalize SPS 674d002a95a81e0089f950
java.lang.RuntimeException: Failed to write sps rbsp
```

Решение: увеличить настройку размера буфера RBSP (по умолчанию 1.5)

```
h264_sps_rbsp_scale=2
```

9. Поток с некоторых камер теряет синхронизацию между аудио и видео

Симптомы: RTSP поток фризит либо не проигрывается по HLS (отдельные сегменты не записываются), в статистике для потока ненормально большое значение синхронизации

```
streams_synchronization=camera1/-21800;camera2/2079600704
```

Решение: в сборках до 5.2.1775 увеличить буфер синхронизации для аудио и видео

```
audio_incoming_buffer_size=100
video_incoming_buffer_size=100
```

начиная со сборки 5.2.1775 увеличить интервал принудительной синхронизации для аудио и видео

```
video_force_sync_timeout=1000
audio_force_sync_timeout=1000
```

10. Поток с некоторых DVR не играет видео

Симптомы: RTSP поток играет с видео в VLC (возможно, с ошибками декодирования), в WCS браузер получает трафик, но не декодирует видео

Решение: обновить WCS до сборки 5.2.1988 и включить настройку

```
jitter_buffer_strictness=TOLERANT
```