

# С IP камеры по RTSP

- [Описание](#)
  - [RTSP-источники](#)
  - [Поддерживаемые кодеки](#)
  - [Поддерживаемые платформы и браузеры](#)
- [Схема работы](#)
- [Настройка](#)
- [Краткое руководство по тестированию](#)
  - [Захват видеопотока с IP-камеры и трансляция в браузер](#)
- [Управление захватом видеопотока с IP-камеры при помощи REST API](#)
  - [Тестирование](#)
  - [REST-вызовы](#)
    - [REST-методы и статусы ответа](#)
    - [Параметры](#)
- [Последовательность выполнения операций \(Call Flow\)](#)
- [Повторное использование подключения к камере](#)
- [Аутентификация при захвате потока](#)
- [Известные проблемы](#)

## 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

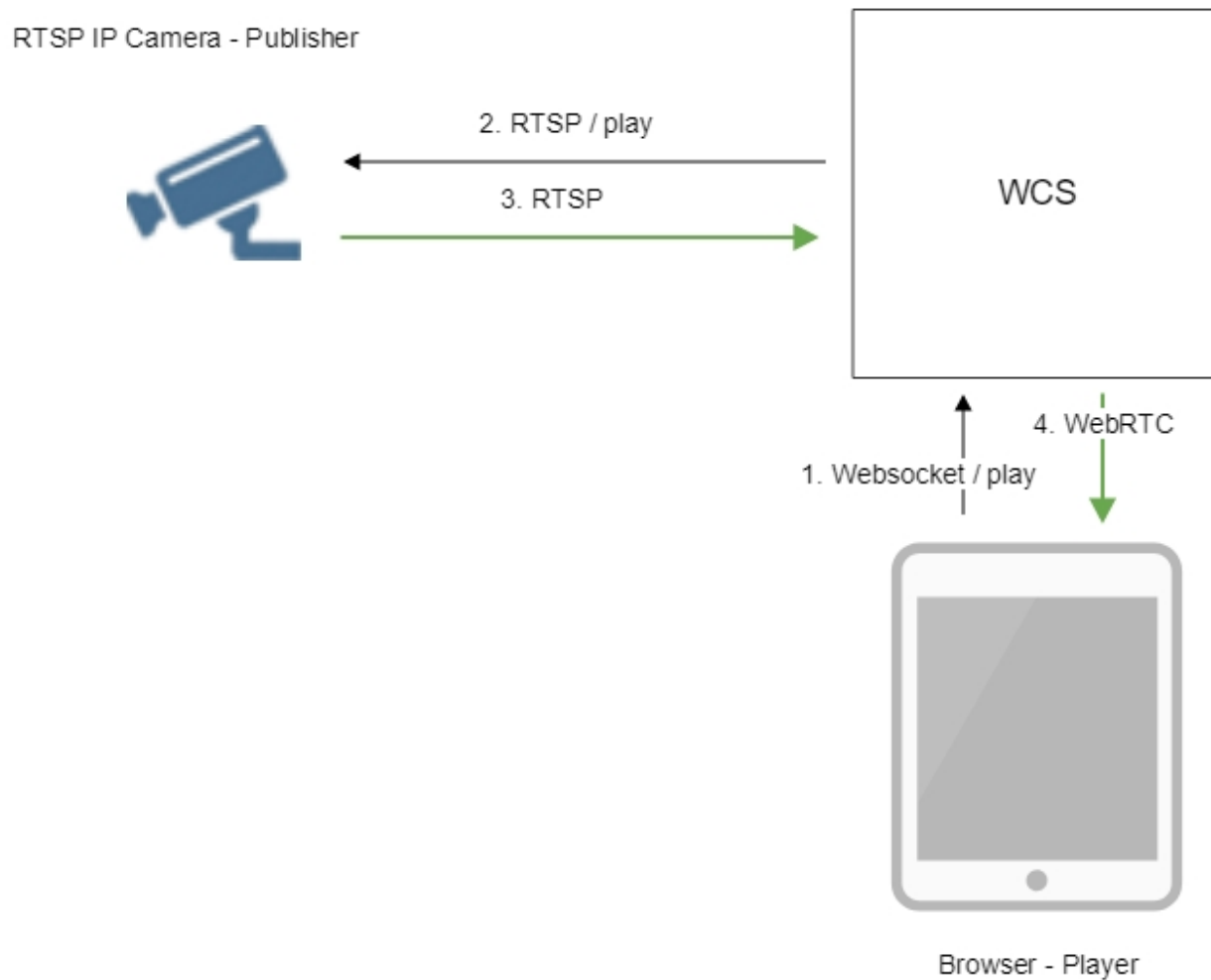
## Поддерживаемые кодеки

- H.264
- VP8
- 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

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 настроек `flashphoner.properties`, например

```
rtsp_client_address=172.16.0.3
```

## Краткое руководство по тестированию

### Захват видеопотока с IP-камеры и трансляция в браузер

1. Для теста используем:

- демо-сервер `demo.flashphoner.com`;
- веб-приложение [Player](#) для захвата и воспроизведения потока

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

WCS URL

wss://demo.flashphoner.com:844

Stream

rtsp://str81.creacast.com/grandlil

Volume

Full Screen

Start

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

Player

9:17

Lille 9.5° C

WCS URL

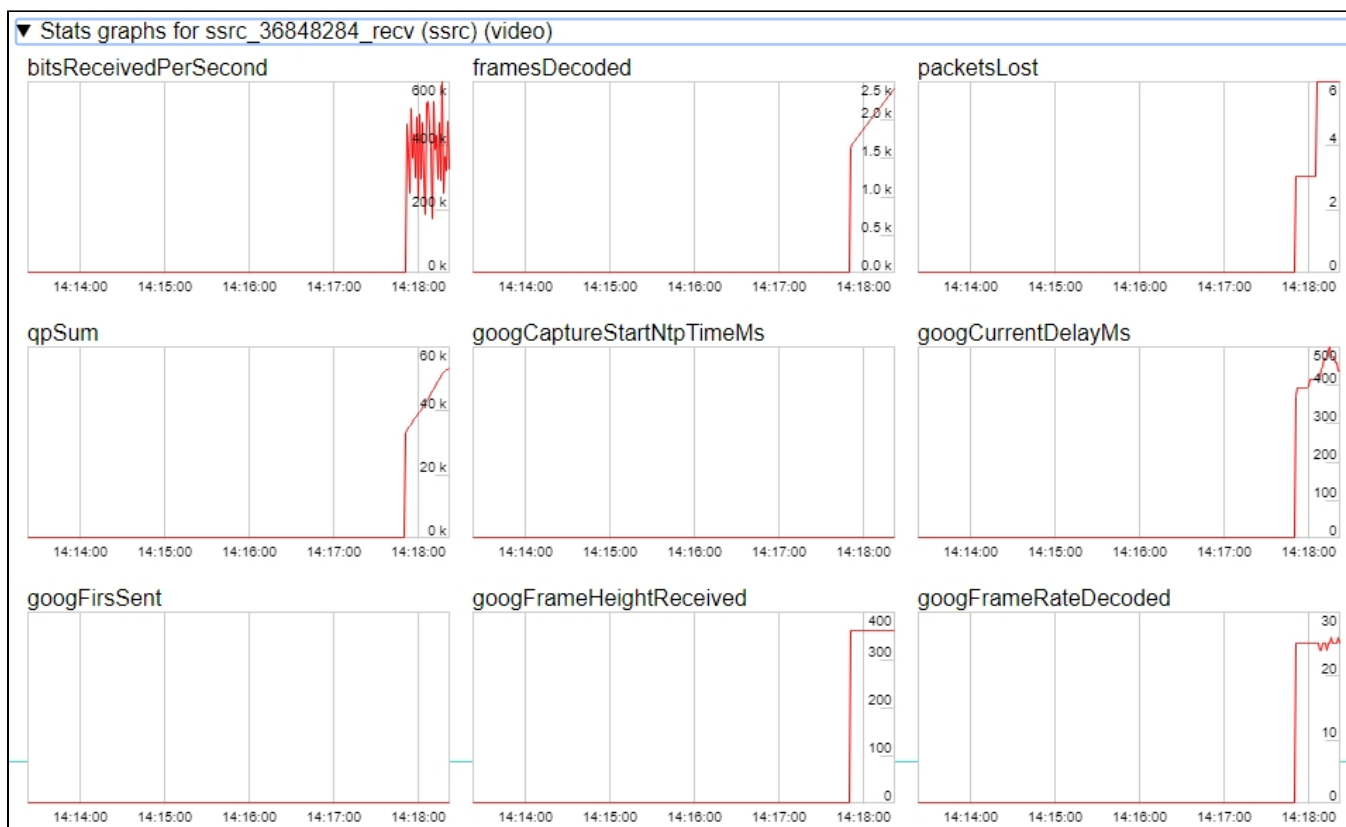
wss://demo.flashphoner.com:844

Stream

rtsp://str81.creacast.com/grandlil

Volume

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](http://demo.flashphoner.com);
- браузер Chrome и [REST-клиент](#) для отправки запросов на сервер;
- веб-приложение [Player](#) для воспроизведения захваченного потока в браузере.

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

Method POST Request URL <http://demo.flashphoner.com:9091/rest-api/rtsp/startup> SEND

Parameters ^

Headers Body Variables

Body content type application/json Editor view Raw input

FORMAT JSON MINIFY JSON

```
{
  "uri": "rtsp://str81.creacast.com/grandlilletv/low"
}
```

200 OK 399.30 ms DETAILS

3. Убедитесь, что поток захвачен сервером. Для этого отправьте запрос /rtsp/find\_all:

Method POST Request URL [http://demo.flashphoner.com:9091/rest-api/rtsp/find\\_all](http://demo.flashphoner.com:9091/rest-api/rtsp/find_all) SEND

Parameters ^

Headers Body Variables

Body content type application/json Editor view Raw input

FORMAT JSON MINIFY JSON

✖

200 OK223.00 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:

Player

11:11

Lille 10.5° C

GRAND LILLE TV

Robersart Color One, ce Samedi, à Wambrechies. Un hommage sera rendu à la petite Angélique...

HOMMAGE

WCS URL

wss://demo.flashphoner.com:844

Stream

rtsp://str81.creacast.com/grandlill

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

Method: POST Request URL: http://demo.flashphoner.com:9091/rest-api/rtsp/terminate

Parameters ^

Headers Body Variables

Body content type: application/json Editor view: Raw input

FORMAT JSON MINIFY JSON

```
{
  "uri": "rtsp://str81.creacast.com/grandlilletv/low"
}
```

200 OK 224.20 ms DETAILS v

6. Воспроизведение потока прервется с ошибкой:

WCS URL: wss://demo.flashphoner.com:844

Stream: rtsp://str81.creacast.com/grandlil

Volume: [Slider]

Full Screen: [Full Screen Icon]

**FAILED**  
RtspAgent shutdown

Start

## 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](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<br>/startup   | <pre>{   "uri": "rtsp://myserver.com /live/myStream" }</pre> |                                                                                     | 409 - Conflict<br>500 - Internal error          | Извлечь RTMP-поток по указанному URL |
| /rtsp<br>/find_all  |                                                              | <pre>{   "uri": "rtsp://myserver.com /live/myStream",   "status": "PLAYING" }</pre> | 200 – потоки найдены<br>404 – потоки не найдены | Найти все извлеченные RTMP-потоки    |
| /rtsp<br>/terminate | <pre>{   "uri": "rtsp://myserver.com /live/myStream" }</pre> |                                                                                     | 200 - поток завершен<br>404 - поток не найден   | Завершить извлеченный RTMP-поток     |

## Параметры

| Имя параметра | Описание              | Пример                                                                            |
|---------------|-----------------------|-----------------------------------------------------------------------------------|
| uri           | URL RTSP-потока       | <a href="rtsp://myserver.com/live/myStream">rtsp://myserver.com/live/myStream</a> |
| status        | Текущий статус потока | PLAYING                                                                           |

## 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 ESTABLISHED[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(){
    ...
}).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 PLAYING[code](#)

```
stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
    ...
}).on(STREAM_STATUS.PLAYING, function(stream) {
    $("#preloader").show();
    setStatus(stream.status());
    onStart(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 STOPPED [code](#)

```
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.

# Известные проблемы

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

Симптомы:

- а) поток не проигрывается, дает задержки видео или подергивания
- б) предупреждения [в клиентском 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 В-фреймов (понижить профиль кодирования, указать в командной строке и т.п.).

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

При этом [в клиентском логе](#) будут 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 файле [flashphoner.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 + AAC потока, опубликованного на сервере, а также в записи потока, звук не синхронизирован с видео или отсутствует

Решение:

- а) установить настройку в файле [flashphoner.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 в [файлах настроек 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 файле [flashphoner.properties](#) необходимо 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%401stream/channel1
```

7. Некоторые камеры не поддерживают поле спонсе в заголовке сообщения при установке RTSP-соединения.

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

Решение: в файле [flashphoner.properties](#) установить настройку

```
rtsp_auth_cnonce=
```

с пустым значением.