# В браузере по WebRTC

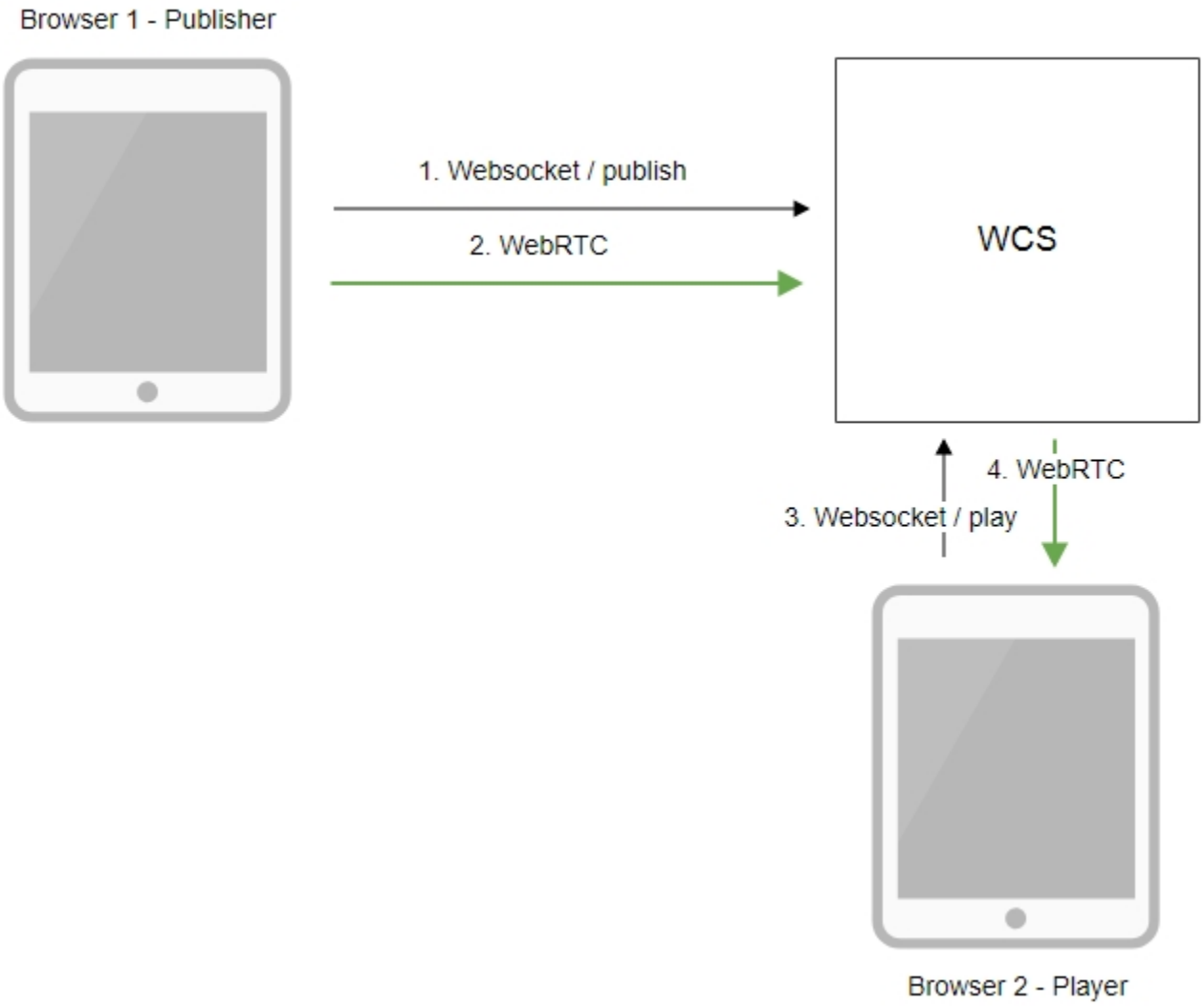

- Описание
  - Поддерживаемые платформы и браузеры
  - Схема работы
- Краткое руководство по тестированию
  - Трансляция видеопотока на сервер и воспроизведение его по WebRTC в браузере
- Последовательность выполнения операций (Call flow)
- Воспроизведение двух и более потоков на одной странице
- Автозапуск воспроизведения
  - Особенности автозапуска воспроизведения в браузерах
    - Chrome
    - iOS Safari
- Тонкая настройка воспроизведения звука в iOS Safari
- Известные проблемы


## Описание

### Поддерживаемые платформы и браузеры

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

### Схема работы




1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду publish.
2. Браузер захватывает микрофон и камеру и отправляет WebRTC поток на сервер.
3. Второй браузер устанавливает соединение также по Websocket и отправляет команду play.
4. Второй браузер получает WebRTC поток и воспроизводит этот поток на странице.

# Краткое руководство по тестированию

## Трансляция видеопотока на сервер и воспроизведение его по WebRTC в браузере

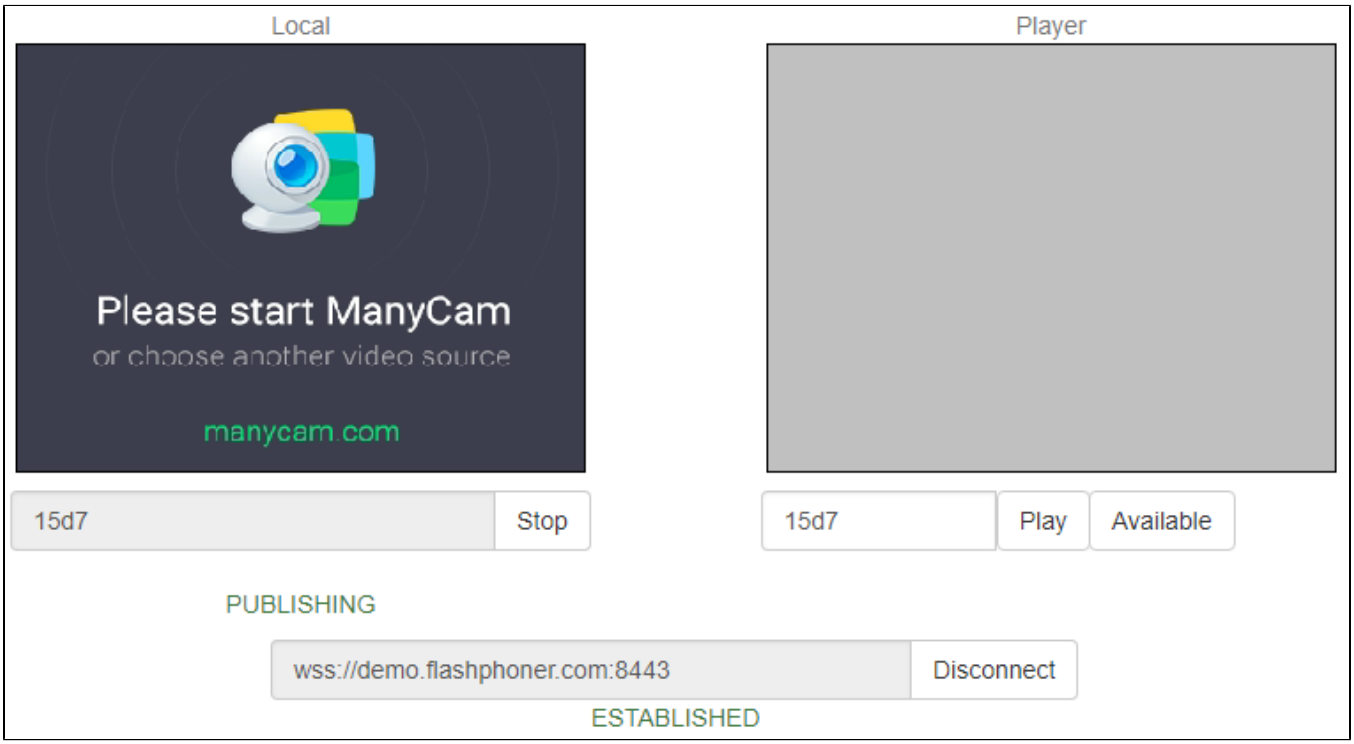
1. Для теста используем демо-сервер demo.flashphoner.com и веб-приложение Two Way Streaming

https://demo.flashphoner.com/client2/examples/demo/streaming/two_way_streaming/two_way_streaming.html

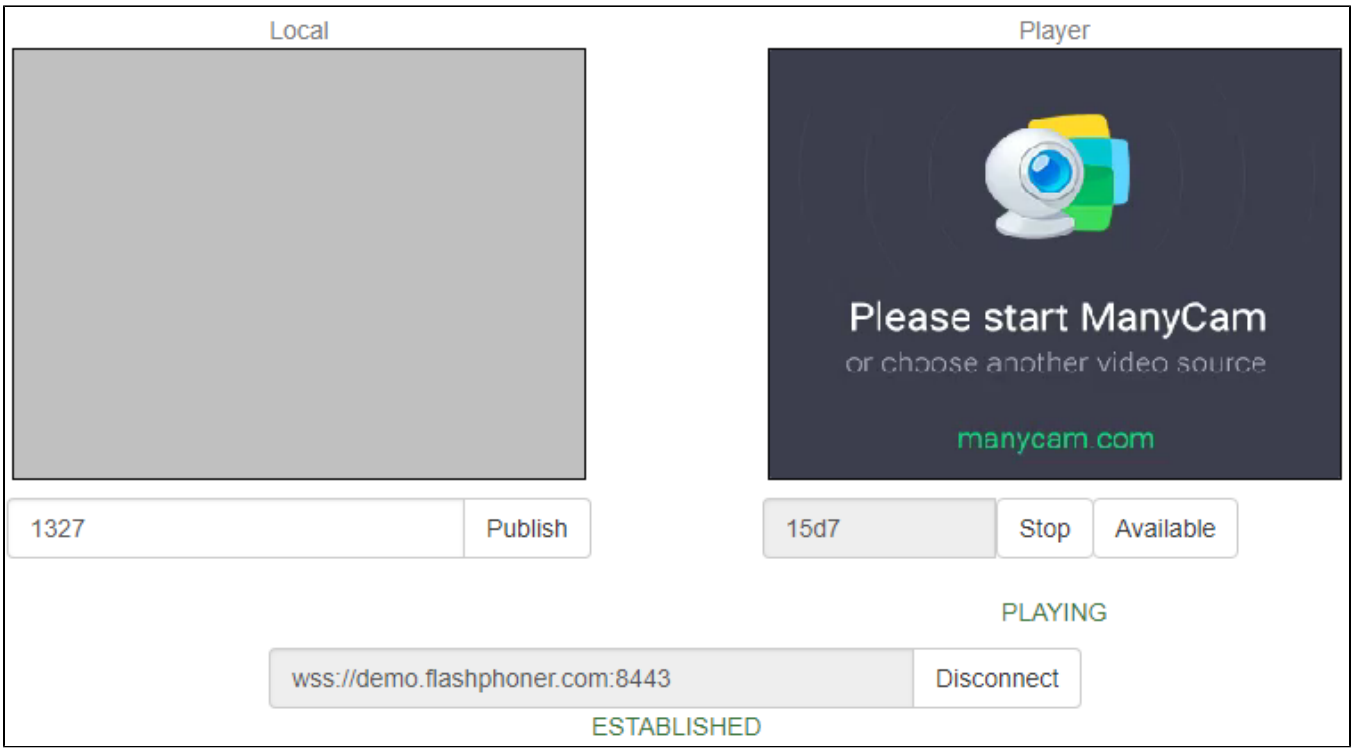
2. Установите соединение с сервером по кнопке Connect



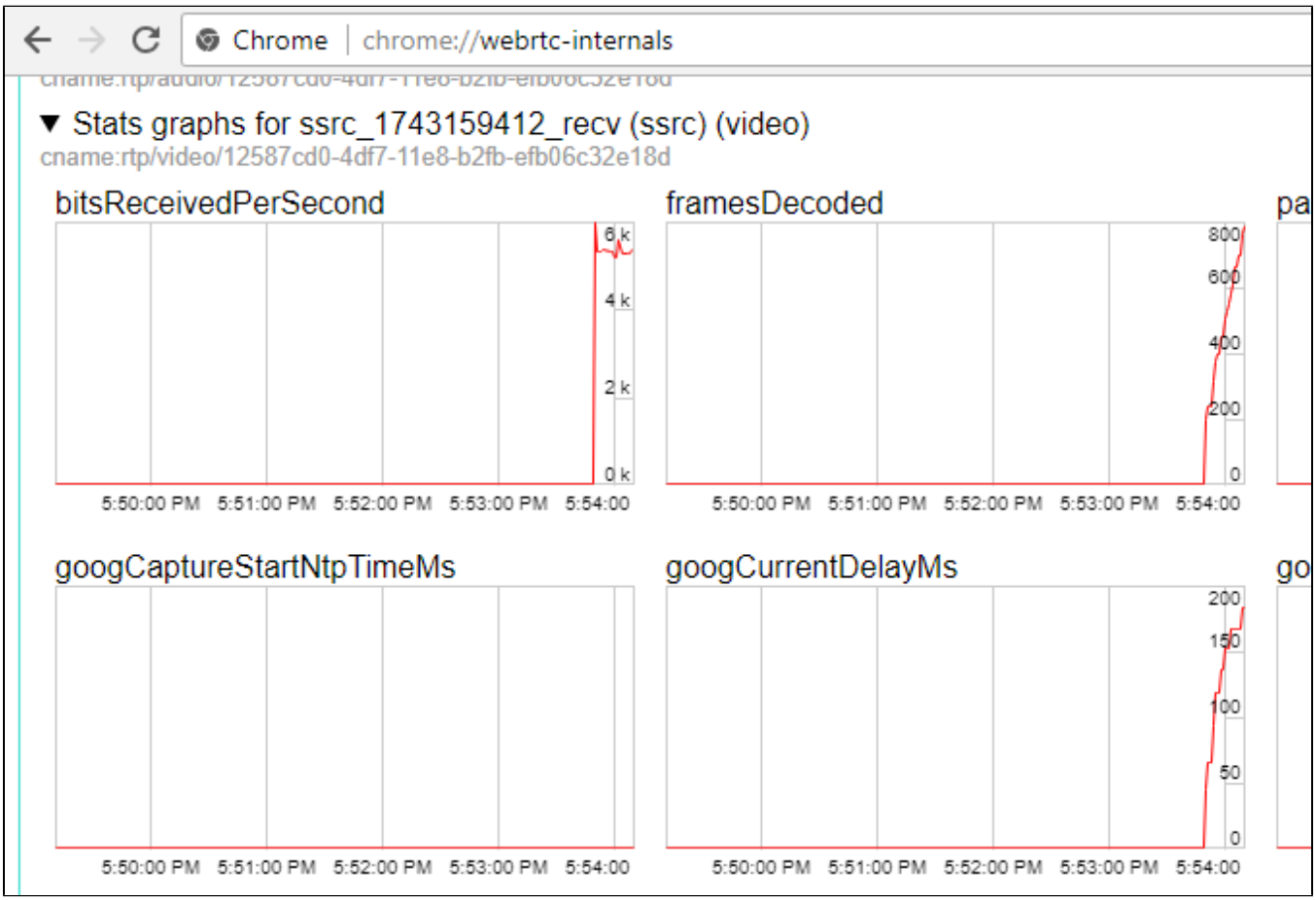

3. Нажмите Publish. Браузер захватывает камеру и отправляет поток на сервер.




4. Откройте Two Way Streaming в отдельном окне, нажмите Connect и укажите идентификатор потока, затем нажмите Play.




5. Графики воспроизведения chrome://webrtc-internals




## Последовательность выполнения операций (Call flow)

Ниже описана последовательность вызовов при использовании примера Two Way Streaming для воспроизведения потока

two_way_streaming.html

two_way_streaming.js

1. Установка соединения с сервером.

Flashphoner.createSession();code

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        setStatus("#connectStatus", session.status());
        onConnected(session);
    }).on(SESSION_STATUS.DISCONNECTED, function () {
        setStatus("#connectStatus", SESSION_STATUS.DISCONNECTED);
        onDisconnected();
    }).on(SESSION_STATUS.FAILED, function () {
        setStatus("#connectStatus", SESSION_STATUS.FAILED);
        onDisconnected();
    });
```

2. Получение от сервера события, подтверждающего успешное соединение.

ConnectionStatusEvent ESTABLISHEDcode

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        setStatus("#connectStatus", session.status());
        onConnected(session);
    }).on(SESSION_STATUS.DISCONNECTED, function () {
        ...
    }).on(SESSION_STATUS.FAILED, function () {
        ...
    });
```

3. Воспроизведение потока.

stream.play();code

```
    session.createStream({
        name: streamName,
        display: remoteVideo
        ...
    }).play();
```

4. Получение от сервера события, подтверждающего успешное воспроизведение потока.

StreamStatusEvent, статус PLAYINGcode

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        setStatus("#playStatus", stream.status());
        onPlaying(stream);
    }).on(STREAM_STATUS.STOPPED, function () {
        ...
    }).on(STREAM_STATUS.FAILED, function (stream) {
        ...
    }).play();
```

5. Прием аудио-видео потока по WebRTC

6. Остановка воспроизведения потока.

stream.stop();code

```
function onPlaying(stream) {
    $("#playBtn").text("Stop").off('click').click(function () {
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
    $("#playInfo").text("");
}
```

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

StreamStatusEvent, статус STOPPEDcode

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        ...
    }).on(STREAM_STATUS.STOPPED, function () {
        setStatus("#playStatus", STREAM_STATUS.STOPPED);
        onStopped();
    }).on(STREAM_STATUS.FAILED, function (stream) {
        ...
    }).play();
```

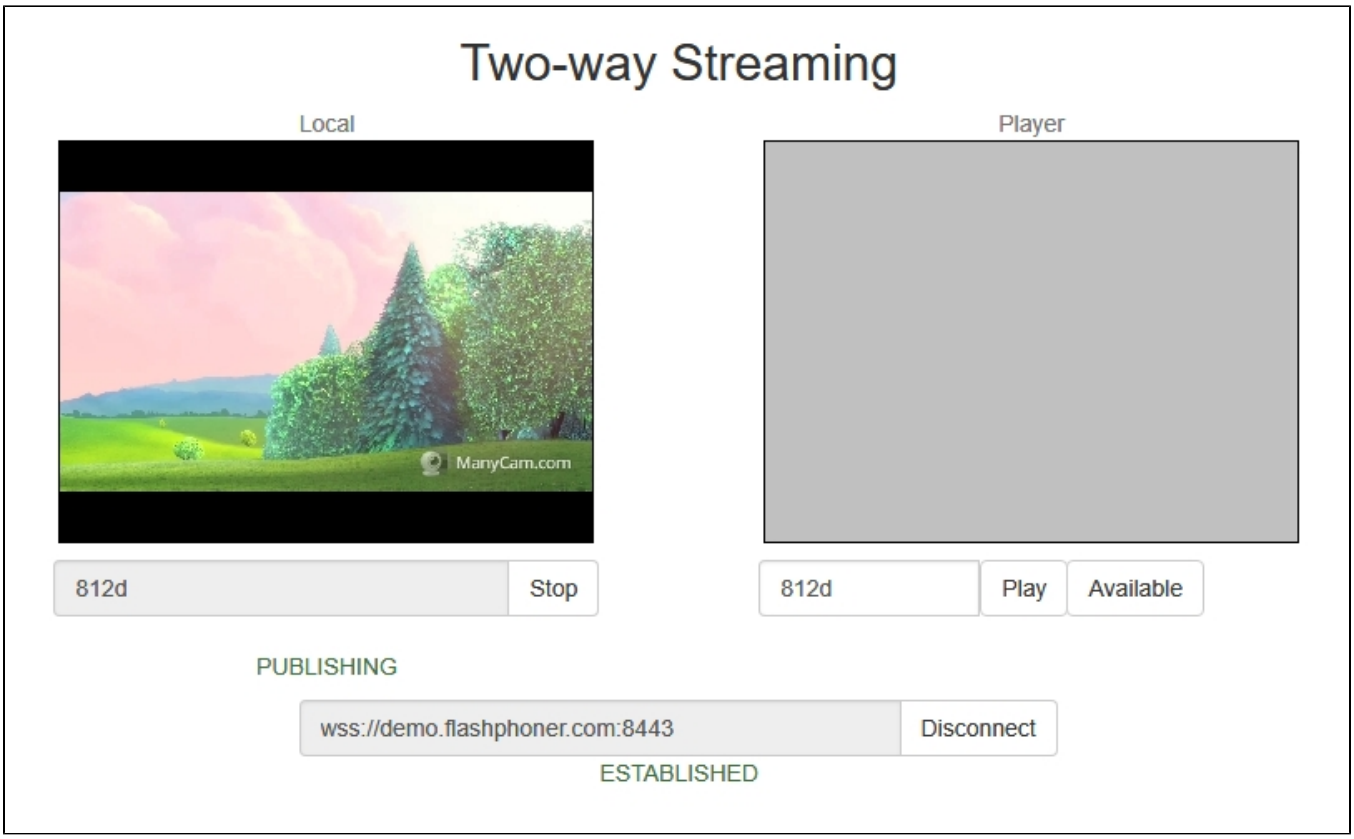
# Воспроизведение двух и более потоков на одной странице

WCS предоставляет возможность воспроизведения двух и более потоков на одной странице. С точки зрениясхемы работыипоследовательности выполнения операцийвоспроизведение любого числа потоков не отличается от воспроизведения одного.

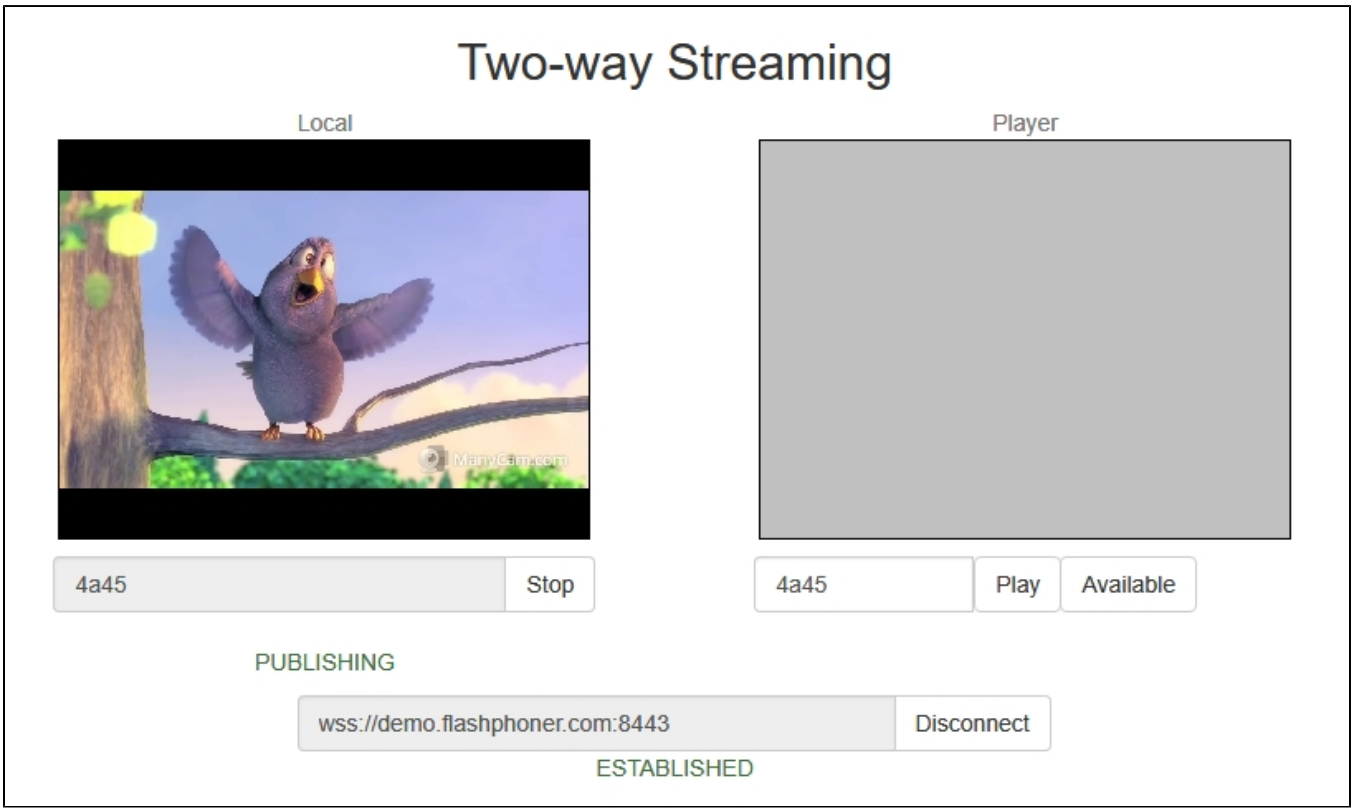
1. Для теста используем:

- демо-сервер demo.flashphoner.com;
- веб-приложениеTwo Way Streamingдля публикации потоков
- веб-приложение2 Playersдля воспроизведения потоков

2. Откройте веб-приложение Two Way Streaming, нажмите Connect, затем Publish. Скопируйте идентификатор первого потока из окна Play:



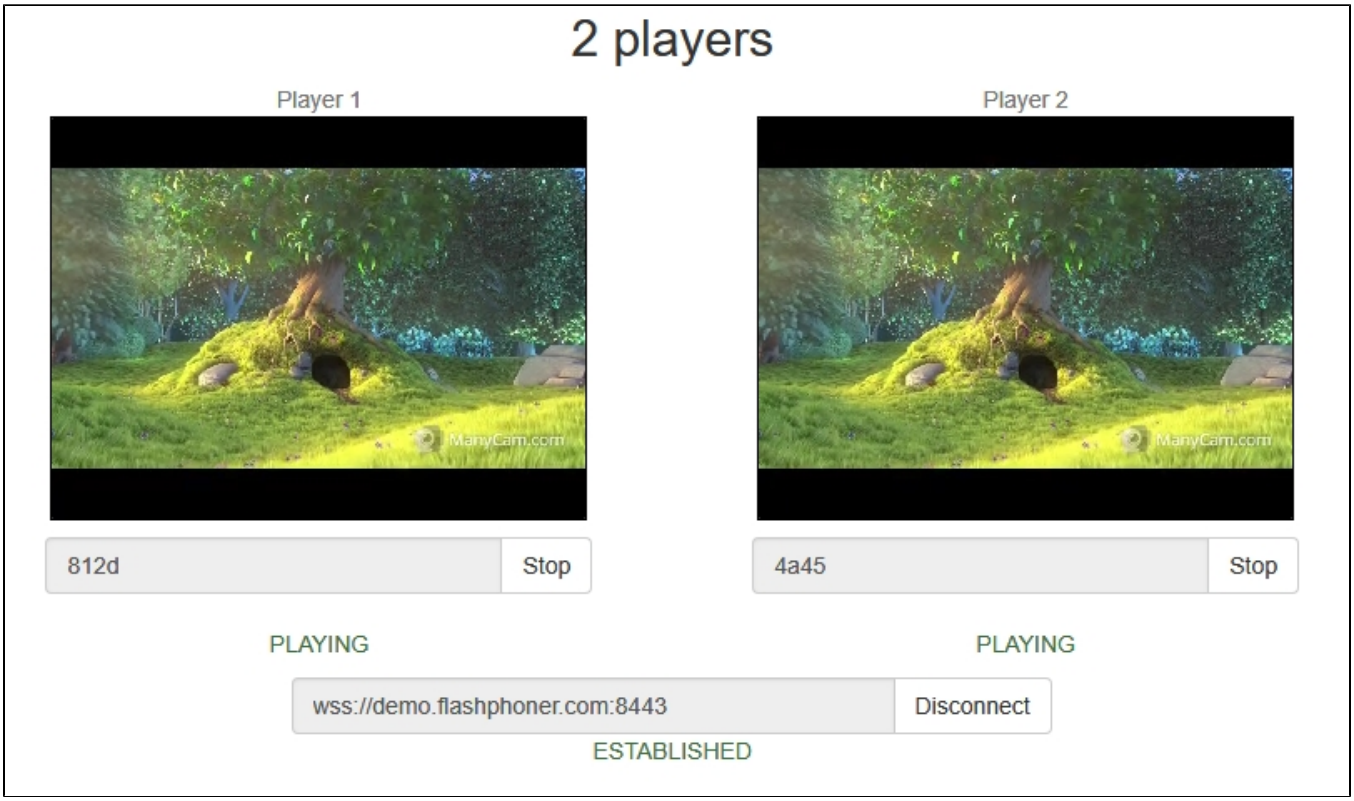

3. В другой вкладке откройте веб-приложение Two Way Streaming, нажмите Connect, затем Publish. Скопируйте идентификатор второго потока из окна Play:



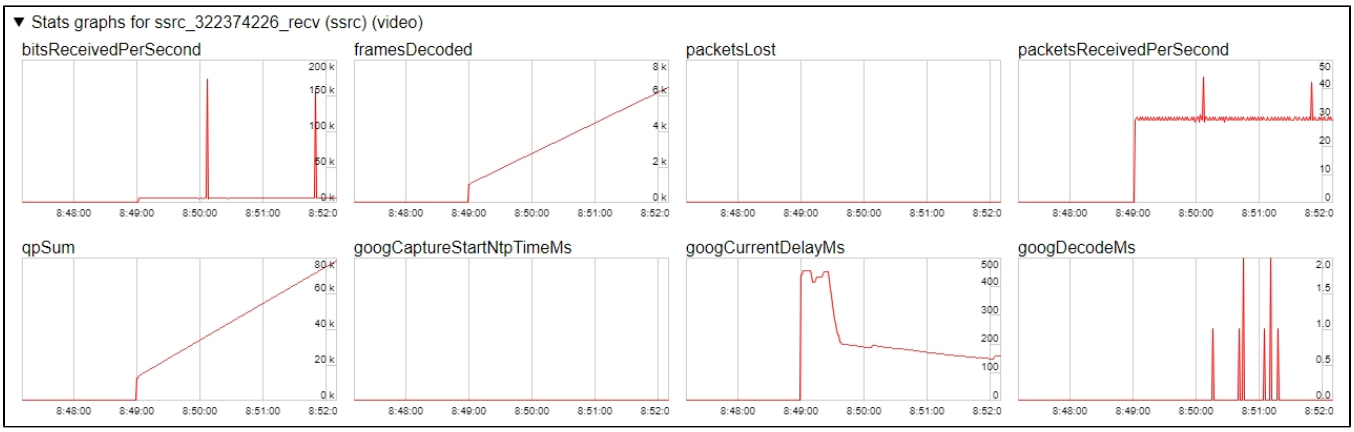

4. Откройте веб-приложение 2 Players, укажите идентификаторы первого (слева) и второго (справа) потоков:



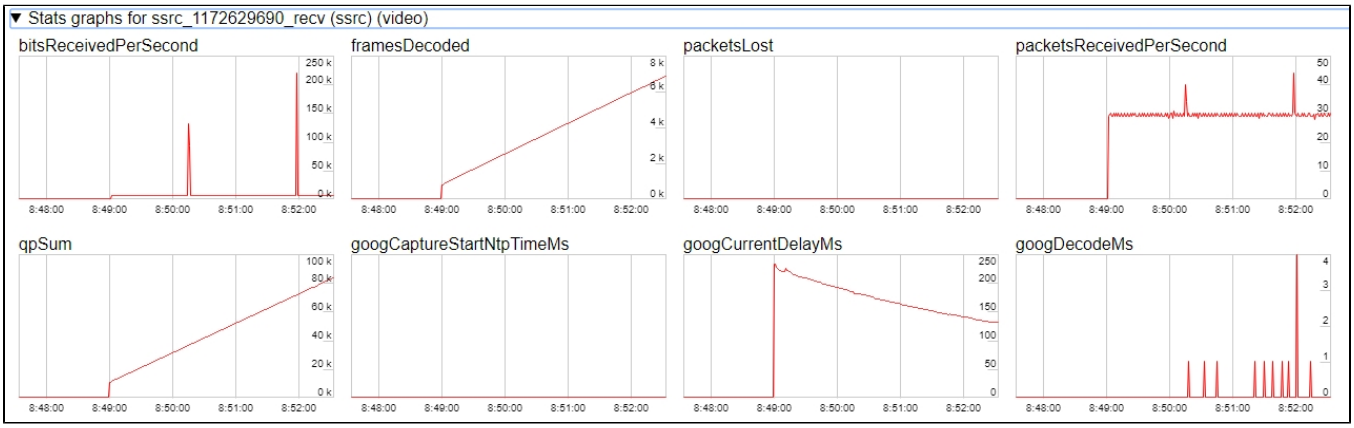

5. Нажмите Play под правым и левым окнами плеера:




6. Графики chrome://webrtc-internals для первого потока:




7. Графики chrome://webrtc-internals для второго потока:




# Автозапуск воспроизведения

Примеры Player и Embed Player поддерживают автозапуск воспроизведения при помощи параметра

```
autoplay=true
```

например

```
https://hostname:8888/embed_player?urlServer=wss://hostname:
8443&streamName=stream1&autoplay=true&mediaProviders=WebRTC
```

Здесь

- hostname - имя WCS-сервера
- stream1 - имя потока на сервере

## Особенности автозапуска воспроизведения в браузерах

### Chrome

В последних версиях браузера Chrome (71 и выше) была изменена политика автозапуска воспроизведения контента на веб-страницах. Теперь для запуска воспроизведения видео необходимо, чтобы пользователь совершил какое-либо действие, например, нажатие на кнопку.

Это изменение влияет на создание аудиоконтекста, который необходим, чтобы работала регулировка громкости. Согласно новой политике, создание аудиоконтекста также требует действия от пользователя.

В связи с этим, в Chrome 71, а также в других браузерах на базе Chromium, поддерживающих изменение политики автозапуска, видео при автозапуске может проигрываться без звука. Для того, чтобы включить звук, пользователь должен переместить регулятор громкости в окне Embed Player.

Существует способ обойти данное ограничение, не требующий действий от пользователя. Для этого необходимо на веб-страницу плеера добавить код

```
<iframe src="silence.mp3" allow="autoplay" id="audio" style="display:none"></iframe>
```

и разместить в одном каталоге со страницей файл silence.mp3, содержащий 0.25 секунды тишины.

В этом случае при загрузке страницы будет воспроизведена тишина, после чего можно будет создавать аудиоконтекст и играть видео со звуком.

### iOS Safari

Автозапуск воспроизведения работает, начиная с iOS 12.2. При этом политика автозапуска, как и в Chrome, требует, чтобы пользователь переместил регулятор громкости для воспроизведения звука.

В версиях iOS 12.2-12.3 звук может не начать воспроизводиться и при движении регулятора громкости. В таких случаях необходимо повторно запустить воспроизведение видео, не обновляя страницу.

При включенном Low Power Mode автозапуск воспроизведения в iOS Safari не работает.

## Тонкая настройка воспроизведения звука в iOS Safari

В случае воспроизведения и последующей публикации видео на одной странице (например, видеочат) в iOS Safari уровень звука для проигрываемого потока может меняться. Избежать этого можно двумя способами:

1. Запрашивать доступ к медиаустройствам при создании сессии перед проигрыванием

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        ...
        if (Browser.isSafariWebRTC() && Browser.isiOS() && Flashphoner.getMediaProviders()[0] === "WebRTC") {
            Flashphoner.playFirstVideo(localVideo, true, PRELOADER_URL).then(function () {
                Flashphoner.getMediaAccess(null, localVideo).then(function (disp) {
                });
            });
        }
        ...
    });
```

2. Через 1-1.5 секунды после получения статуса потока PLAYING, отключить и снова включить звук и/или видео

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function (stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        setStatus("#playStatus", stream.status());
        onPlaying(stream);
        if (Browser.isSafariWebRTC() && Browser.isiOS() && Flashphoner.getMediaProviders()[0] === "WebRTC") {
            setTimeout(function () {
                muteVideo();
                unmuteVideo();
            }, 1500);
        }
        ...
    }).play();
```

## Известные проблемы

1. Возможный баг в браузере Safari на iOS приводит к фризам при воспроизведении WebRTC

Симптомы: останавливается воспроизведение видео, звуковая дорожка при этом может продолжать играть, для восстановления требуется перезагрузка страницы либо перезапуск браузера.

Решение:
а) включить транскодер на сервере, указав в файлеflashphoner.properties

```
disable_streaming_proxy=true
```

б) при воспроизведении потока с iOS Safari явно указать ширину и высоту, например:

```
session.createStream({constraints:{audio:true,video:{width:320,height:240}}}).play();
```

2. При публикации потока по RTMP и воспроизведении в браузере по WebRTC вместо аудиокодека Opus используется PCMU

Симптомы: в chrome://webrtc-internals отображается кодек PCMU

Решение: отключить алгоритм избегания транскодинга (Avoid Transcoding Alhorithm) при помощи опции в файлеflashphoner.properties

```
disable_rtc_avoid_transcoding_alg=true
```

3.При публикации потока при помощи Flash Streaming, воспроизведении этого потока в iOS Safari по WebRTC и одновременной публикации потока по WebRTC из Safari перестает воспроизводиться звук.

Симптомы:

а) Публикация потока stream1 из приложения Flash Streaming в браузере Chrome под Windows
б) Воспроизведение потока stream1 на iOS Safari в приложении Two Way Streaming. Звук и видео воспроизводятся нормально.
в) Публикация потока из iOS Safari в приложении Two Way Streaming. Воспроизведение звука пропадает.
г) Остановка публикации в iOS Safari. Воспроизведение звука восстанавливается.

Решение: отключить алгоритм избегания транскодинга (Avoid Transcoding Alhorithm) на сервере при помощи опции в файлеflashphoner.properties

```
disable_rtc_avoid_transcoding_alg=true
```

4. При публикации потока по RTMP и отключении Keep Alive для всех протоколов воспроизведение в браузере по WebRTC останавливается по истечению WebSocket-таймаута

Симптомы: при публикации потока по RTMP воспроизведение в браузере по WebRTC останавливается без явного указания ошибки

Решение: если Keep Alive отключен для всех протоколов при помощи настройкив файлеflashphoner.properties

```
keep_alive.algorithm=NONE
```

необходимо также отключить таймаут на чтение WebSocket настройкой

```
ws_read_socket_timeout=false
```

5. Кодек G722 не работает в браузере Edge

Симптомы: поток со звуком G722 не воспроизводится в браузере Edge или воспроизводится без звука, с фризами

Решение: использовать другой кодек или другой браузер. В случае, если использование другого браузера невозможно, исключить кодек G722 при помощи настройки

```
codecs_exclude_streaming=g722,telephone-event
```

6. Некоторые браузеры, основанные на Chromium, например Opera, Yandex, в зависимости от версии и ОС не поддерживают кодек H264

Симптомы: не работает публикация, не работает воспроизведение частично (только звук) или полностью при трансляции WebRTC потока H264

Решение: разрешить поддержку vp8 на стороне сервера

```
codecs=opus,...,h264,vp8,...
```

исключить H264 для трансляции или воспроизведения на стороне клиента

```
publishStream = session.createStream({
    ...
    stripCodecs: "h264,H264"
}).on(STREAM_STATUS.PUBLISHING, function (publishStream) {
    ...
});
publishStream.publish();
```

Необходимо отметить, что при трансляции H264 потока и воспроизведении его как VP8 на сервере включаетсятранскодинг.

7. Если в настройках браузера Chrome 71 и выше для сайта разрешен Flash, при воспроизведении по WebRTC в консоль браузера может выводиться ошибка "Cross-origin content must have visible size large than 400 x 300 pixels, or it will be blocked"

Симптомы: при воспроизведении по WebRTC в консоль браузера Chrome выводится сообщение "Cross-origin content must have visible size large than 400 x 300 pixels, or it will be blocked", при этом воспроизведение работает

Решение: использовать WebSDK без поддержки Flash

```
flashphoner-no-flash.js
```