

# Звонок между двумя браузерами через SIP сервер

## Описание

SIP звонок между браузерами через WCS является частным случаем [звонков между браузером и SIP-устройством](#), при этом веб-приложение в браузере исполняет роль программного телефона с обеих сторон звонка.

## Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari	Edge
Windows	✓	✓	✗	✓
Mac OS	✓	✓	✓	✓
Android	✓	✓	✗	✓
iOS	✓	✓	✓	✓

## Поддерживаемые протоколы

- WebRTC
- RTP
- SIP

## Поддерживаемые кодеки

- H.264
- VP8
- G.711
- Speex
- G.729
- Opus

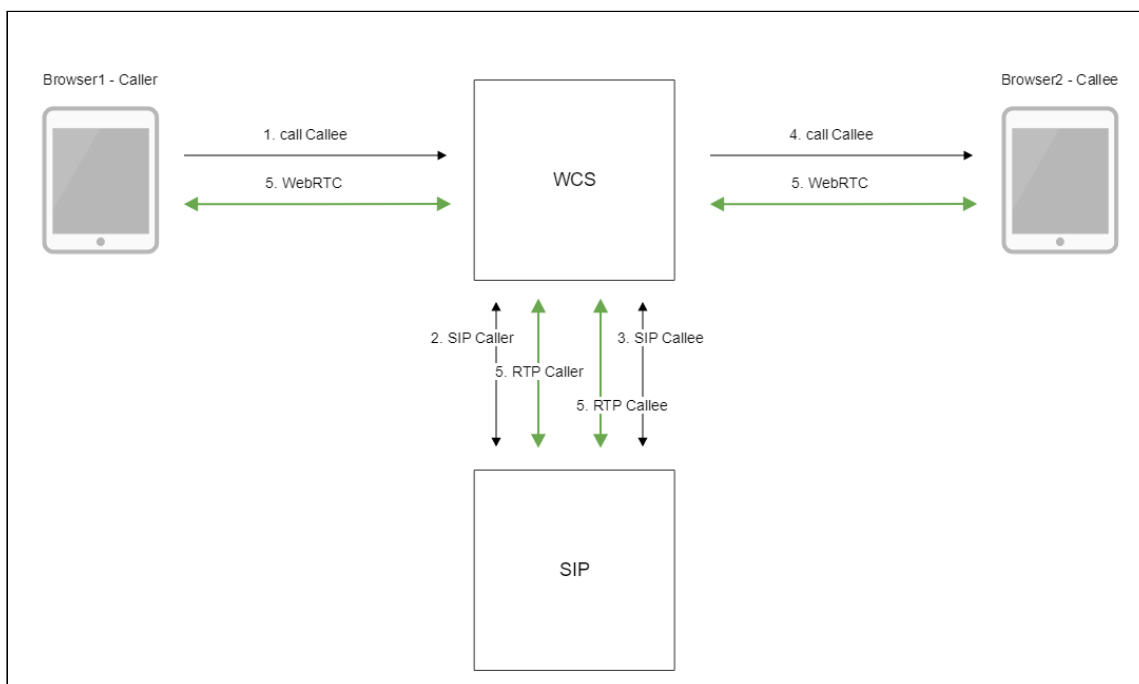
## Поддерживаемые SIP функции

- DTMF
- Удержание звонка
- Перевод звонка

SIP функции управляются при помощи WebSDK.

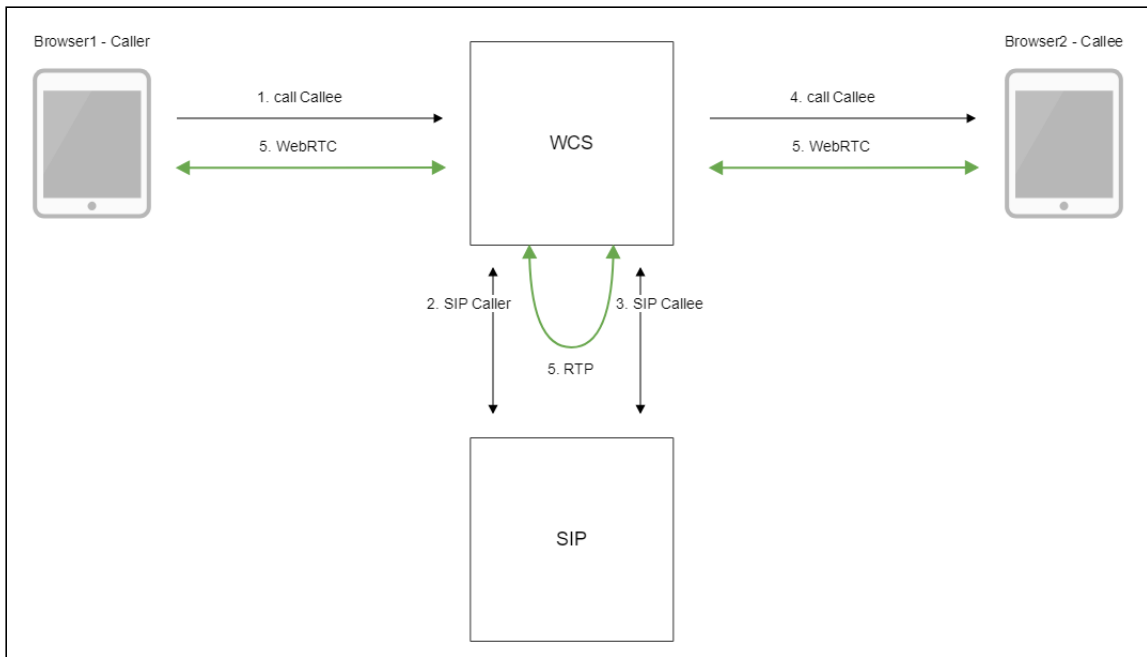
## Схема работы

### 1. SIP-сервер как прокси-сервер для передачи вызовов и RTP медиа



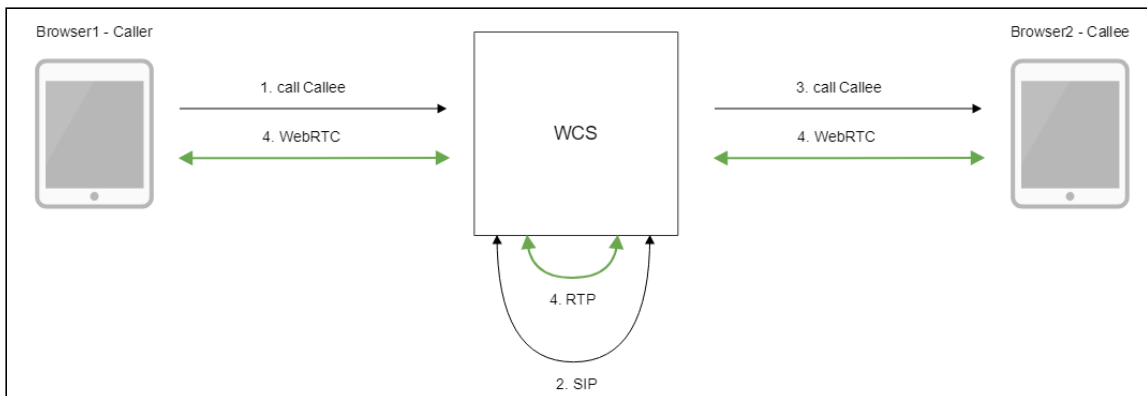
1. Браузер 1 начинает звонок с аккаунта **Caller** на аккаунт **Callee**
2. WCS соединяется с SIP-сервером
3. SIP-сервер передает WCS входящий вызов на аккаунт **Callee**
4. WCS передает браузеру 2 событие о поступлении звонка
5. Браузеры обмениваются аудио- и видеопотоками

### 2. SIP-сервер только как сервер для передачи вызовов



1. Браузер 1 начинает звонок с аккаунта **Caller** на аккаунт **Callee**
2. WCS соединяется с SIP-сервером
3. SIP-сервер передает WCS входящий вызов на аккаунт **Callee**
4. WCS передает браузеру 2 событие о поступлении звонка
5. Браузеры обмениваются аудио- и видеопотоками

### 3. Без внешнего SIP-сервера. SIP и RTP медиа обрабатываются на WCS.



1. Браузер 1 начинает звонок с аккаунта **Caller** на аккаунт **Callee**
2. WCS устанавливает SIP-соединение между аккаунтами
3. WCS передает браузеру 2 событие о поступлении звонка
4. Браузеры обмениваются аудио- и видеопотоками

## Краткое руководство по тестированию

1. Для тестирования используем:
2. два SIP-аккаунта;
3. веб-приложение [Phone](#) для совершения звонка
4. Откройте веб-приложение Phone. Введите данные SIP-аккаунта и нажмите кнопку **Connect** для установки соединения с сервером:

## Phone Min

### Connection

<b>WCS URL</b>	<input type="text" value="wss://p11.flashphoner.com:8443"/>
<b>SIP Login</b>	<input type="text" value="10006"/>
<b>SIP Auth Name</b>	<input type="text" value="10006"/>
<b>SIP Password</b>	<input type="password" value="....."/>
<b>SIP Domain</b>	<input type="text" value="yoursip.domain"/>
<b>SIP Outbound Proxy</b>	<input type="text" value="yoursip.domain"/>
<b>SIP Port</b>	<input type="text" value="5060"/>
<b>Register required</b>	<input checked="" type="checkbox"/>

5. Откройте веб-приложение Phone в другом окне браузера. Введите данные второго SIP-аккаунта и нажмите кнопку **Connect** :



# Phone Min

**Connection**

**WCS URL**

**SIP Login**

**SIP Auth Name**

**SIP Password**

**SIP Domain**

**SIP Outbound Proxy**

**SIP Port**

**Register required**

6. Введите идентификатор SIP-аккаунта, принимающего звонок, и нажмите кнопку **Call**:

**Mute**  off

7. Примите звонок, нажав кнопку **Answer**:

**Mute**  off

You have a new call from 10005

RING

Звонок начался:



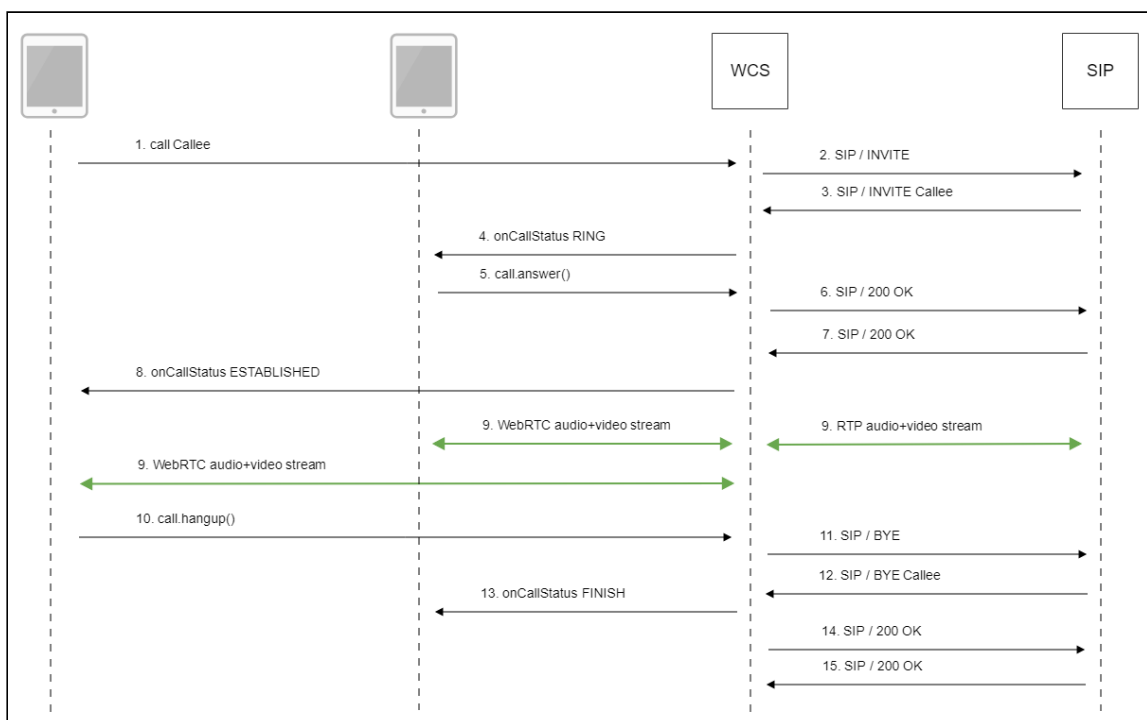
8. Для завершения звонка нажмите кнопку `Hangup`

## Последовательность выполнения операций

Ниже описана последовательность вызовов при использовании примера Phone для создания звонка. SIP-сервер используется как прокси-сервер для передачи команд и медиа

[phone.html](#)

[phone.js](#)



1. Создание звонка при помощи WebSDK

`Session.createCall()`, `Call.call()` code

```

var outCall = session.createCall({
  callee: $("#callee").val(),
  visibleName: $("#sipLogin").val(),
  localVideoDisplay: localDisplay,
  remoteVideoDisplay: remoteDisplay,
  constraints: constraints,
  receiveAudio: true,
  receiveVideo: false
  ...
});

outCall.call();

```

2. Отправка `SIP INVITE` на SIP сервер
3. SIP-сервер отправляет `SIP INVITE` на WCS, поскольку вызываемый абонент зарегистрирован с него же
4. Отправка второму браузеру события, оповещающего о входящем звонке `CALL_STATUS.RING` [code](#)

```

Flashphoner.createSession(connectionOptions).on(SESSION_STATUS.ESTABLISHED,
function(session, connection){
  ...
}).on(SESSION_STATUS.INCOMING_CALL, function(call){
  call.on(CALL_STATUS.RING, function(){
    setStatus("#callStatus", CALL_STATUS.RING);
    ...
  });
  ...
});

```

5. Второй браузер отвечает на звонок

`Call.answer()` [code](#)

```

function onIncomingCall(inCall) {
  currentCall = inCall;

  showIncoming(inCall.caller());

  $("#answerBtn").off('click').click(function(){
    $(this).prop('disabled', true);
    var constraints = {
      audio: true,
      video: false
    };
    inCall.answer({
      localVideoDisplay: localDisplay,
      remoteVideoDisplay: remoteDisplay,
      receiveVideo: false,
      constraints: constraints
    });
    showAnswered();
  }).prop('disabled', false);
}

```

```
...  
}
```

6. Передача подтверждения SIP-серверу
7. Получение подтверждения от SIP-сервера
8. Первый браузер получает от сервера событие, подтверждающего успешное соединение `CALL_STATUS.ESTABLISHED` code

```
var outCall = session.createCall({  
  ...  
}).on(CALL_STATUS.ESTABLISHED, function(){  
  setStatus("#callStatus", CALL_STATUS.ESTABLISHED);  
  $("#holdBtn").prop('disabled', false);  
  onAnswerOutgoing();  
  ...  
});
```

9. Стороны звонка обмениваются аудио- и видеопотоками
10. Завершение звонка  
`Call.hangup()` code

```
function onConnected(session) {  
  $("#connectBtn,  
  #connectTokenBtn").text("Disconnect").off('click').click(function(){  
    $(this).prop('disabled', true);  
    if (currentCall) {  
      showOutgoing();  
      disableOutgoing(true);  
      setStatus("#callStatus", "");  
      currentCall.hangup();  
    }  
    session.disconnect();  
  }).prop('disabled', false);  
}
```

11. Отправка `SIP BYE` на SIP-сервер
12. Получение `SIP BYE` от SIP-сервера
13. Отправка второму браузеру события, оповещающего о завершении звонка  
`CALL_STATUS.FINISH` code

```
Flashphoner.createSession(connectionOptions).on(SESSION_STATUS.ESTABLISHED,  
function(session, connection){  
  ...  
}).on(SESSION_STATUS.INCOMING_CALL, function(call){  
  call.on(CALL_STATUS.RING, function(){  
    ...  
  }).on(CALL_STATUS.FINISH, function(){  
    setStatus("#callStatus", CALL_STATUS.FINISH);  
    onHangupIncoming();  
    currentCall = null;  
  });  
});
```

```
});  
...  
});
```

14. Отправка подтверждения на SIP-сервер
15. Получение подтверждения от SIP-сервера

#### №# Звонки без использования внешнего SIP сервера

WCS может обрабатывать трафик SIP звонка без использования SIP сервера (см [схему выше](#)). Для этого необходимо установить следующие настройки в файле `flashphoner.properties`

```
enable_local_videochat=true  
sip_add_contact_id=false
```

1. Для тестирования используем:
2. веб-приложение Phone для совершения звонка
3. Откройте веб-приложение Phone. Введите:
4. имя пользователя
5. пароль
6. в поле SIP Domain укажите адрес WCS сервера (но не доменное имя!)
7. в поле SIP Outbound Proxy укажите адрес WCS сервера (но не доменное имя!)
8. в поле SIP Port укажите 0
9. снимите переключатель `Register required`  
Нажмите `Connect`

# Phone Min

## Connection

WCS URL

wss://test1.flashphoner.com:8443

SIP Login

test1

SIP Auth Name

test1

SIP Password

.....

SIP Domain

95.191

SIP Outbound  
Proxy

95.191

SIP Port

0

Register  
required

ESTABLISHED

Disconnect

Auth Token

/5.129 49184/95.191 8443-

Disconnect

Mute

off

Callee SIP username

Call

10. Откройте веб-приложение Phone в другом окне браузера. Введите:
11. имя второго пользователя
12. пароль
13. в поле SIP Domain укажите адрес WCS сервера (но не доменное имя!)

14. в поле SIP Outbound Proxy укажите адрес WCS сервера (но не доменное имя!)

15. в поле SIP Port укажите 0

16. снимите переключатель **Register required**

Нажмите **Connect**

## Phone Min

### Connection

**WCS URL**

**SIP Login**

**SIP Auth Name**

**SIP Password**

**SIP Domain**

**SIP Outbound Proxy**

**SIP Port**

**Register required**

**ESTABLISHED** Disconnect

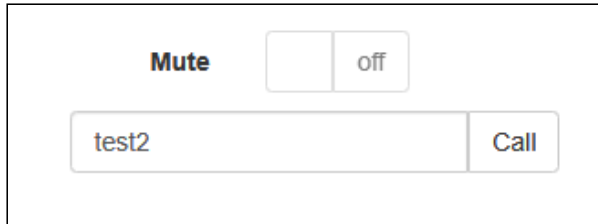
**Auth Token**

Disconnect

**Mute**  off

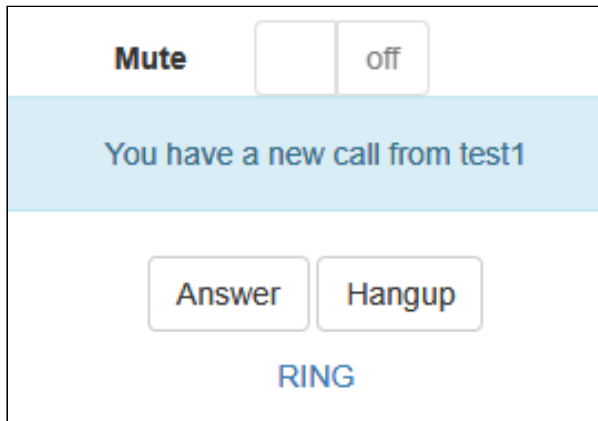
Call

17. Введите имя пользователя, принимающего звонок, и нажмите **Call**



The screenshot shows a user interface for a call. At the top, there is a "Mute" label followed by a square toggle switch and the text "off". Below this, there is a text input field containing "test2" and a "Call" button to its right.

18. Примите звонок, нажав кнопку **Answer**



The screenshot shows a user interface for an incoming call. At the top, there is a "Mute" label followed by a square toggle switch and the text "off". Below this, a light blue banner displays the text "You have a new call from test1". Underneath the banner, there are two buttons: "Answer" and "Hangup". At the bottom of the interface, the word "RING" is displayed in blue.



## 19. Звонок установлен

The image displays two side-by-side screenshots of the 'Phone Min' web interface, both showing a successful SIP connection. The left screenshot is for client 'test1' and the right for 'test2'.

**Left Screenshot (Client: test1):**

- WCS URL:** wss://test1.flashphoner.com:8443
- SIP Login:** test1
- SIP Auth Name:** test1
- SIP Password:** [Redacted]
- SIP Domain:** 95.191.[Redacted]
- SIP Outbound Proxy:** 95.191.[Redacted]
- SIP Port:** 0
- Register required:**
- Auth Token:** /5.129.[Redacted]49184/95.191.[Redacted]
- Status:** ESTABLISHED
- Buttons:** Disconnect

**Right Screenshot (Client: test2):**

- WCS URL:** wss://test1.flashphoner.com:8443
- SIP Login:** test2
- SIP Auth Name:** test2
- SIP Password:** [Redacted]
- SIP Domain:** 95.191.[Redacted]
- SIP Outbound Proxy:** 95.191.[Redacted]
- SIP Port:** 0
- Register required:**
- Auth Token:** /5.129.[Redacted]49188/95.191.[Redacted]84
- Status:** ESTABLISHED
- Buttons:** Disconnect

**Bottom Section (Common):**

- Mute:**  off
- Hold:** [Button]
- Hangup:** [Button]
- Status:** ESTABLISHED