

WebRTC Manual

Introduction of WebRTC

WebRTC (Web Real-Time Communication) is an API definition drafted by the World Wide Web Consortium(W3C) and supported by companies such as Google, Mozilla and Opera to allow browsers and mobile applications Real-Time Communications (RTC) capabilities via simple APIs. This results in browser to browser applications for voice communications without the need of either internal or external plugins. Web Extensions is possibly a better description of what WebRTC actually is. One potential benefit of adopting this innovative WebRTC technology is that enterprises can serve their customers with direct voice communication via a website.

The following section details how to install WebRTC and how to utilize it to make calls. There are two methods for reference:

Method 1:

Embed WebRTC UI on Coovox System

Method 2:

Deploy WebRTC UI on a Web server

Method 1 Embed WebRTC UI on Coovox System

Objective:

Embedding WebRTC UI on your Coovox system makes it convenient for users to complete Web phone functionality by utilizing an existing IP PBX. The user will be able to make a call without installing any plugins or softphone.

There are two different deployment scenarios for Method 1.

- 1) WebRTC UI client and IP PBX are in the same LAN
- 2) Coovox IP PBX is behind Router NAT

Take Google Chrome browser as an example to clarify the above deployment scenario's.

1) WebRTC UI client and IP PBX are both in the same LAN

If WebRTC UI client and IP PBX are in the same LAN then there is no need for NAT traversal.

Please configure as below:

Step 1:

Download ulmage-md5.u20v2.webrtc_patch (firmware of U20V2),
ulmage-md5.u50v2.webrtc_patch (firmware of U50V2) and upgrade webrtc-ui on CooVox V2.

Note: CooVox V2 system will automatically reset after upgrading.

Resetting time will be around 1 min.

Step 2:

Once upgrade has successfully completed, open the following link
<https://192.168.1.65:9999/webrtc/> in Google Chrome browser. See below Figure1&2:

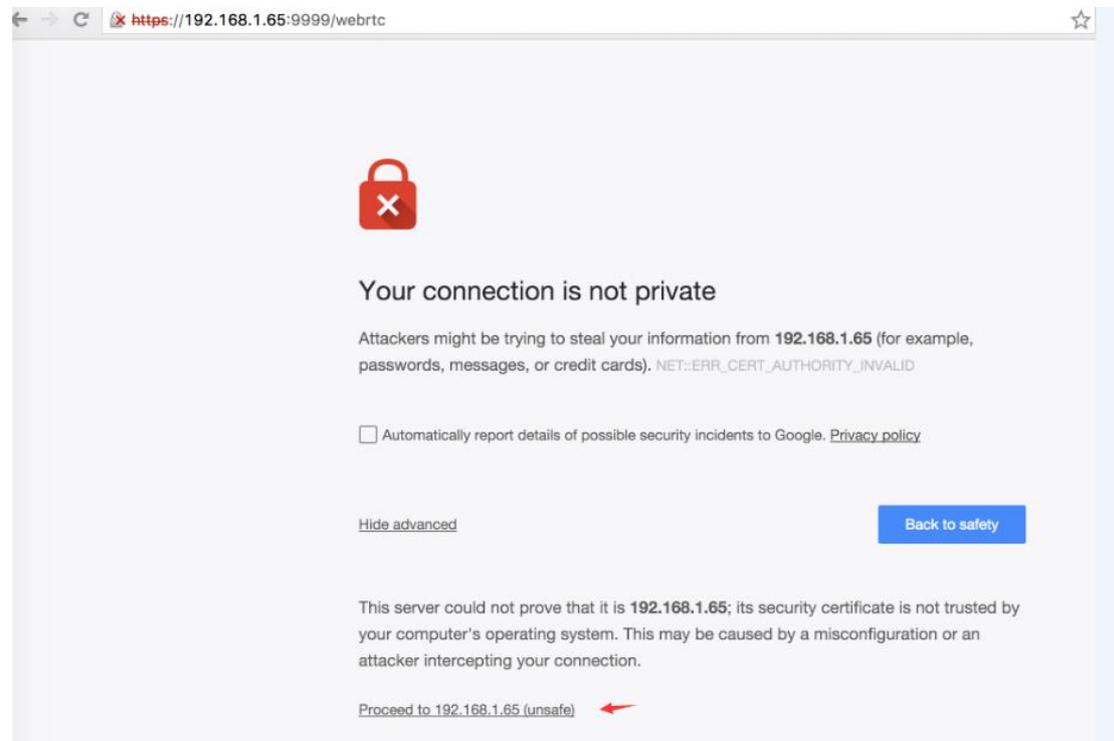


Figure 1. Safety Warning

Chrome will warn you that the page is unsafe. To bypass this warning, you must click “Advanced” and then click “Proceed to 192.168.1.65(unsafe)” to continue. You will then be redirected to the web extension register page as below Figure 2.

The image shows a web-based login form for a service called "Webphone". The form has a light brown background and a white header with the "Webphone" logo in yellow and white. Below the header, there are four input fields, each with a label and a placeholder example. The first field is labeled "Name" with the example "i.e. Homer Simpson" and contains the number "682". The second field is labeled "SIP URI" with the example "i.e. sip:homer@your-domain.com" and contains "682@192.168.1.65". The third field is labeled "SIP password" with the example "i.e. sip:homer@your-domain.com" and contains six dots. The fourth field is labeled "WS URI" with the example "i.e. wss://your-domain.com:8089/ws" and contains "wss://192.168.1.65:8089/ws". Each input field has a red question mark icon to its right. At the bottom right of the form, there is a link labeled "advanced settings".

Figure 2. Login Extension

Note:

1. You are able to login and use the Webphone provided that you have created a Web Extension on your Coovox IP PBX. Your Webphone user name and password must match that set on your Web Extension (Web Extensions are not the SIP/IAX2 extensions on Coovox).
2. There are two possible protocols options to choose from when setting up your connection. First option is WS (WebSocket) Protocol which is an independent TCP-based protocol providing full-duplex communication channels over a single TCP connection. The WebSocket protocol was standardized by the IETF as RFC 6455 in 2011, and the WebSocket API in Web IDL is being standardized by the W3C.
3. Option 2 is WSS (WebSockets over SSL/TLS), like HTTPS, WSS is encrypted and we strongly recommend the secure wss:// protocol over the insecure ws:// transport. A variety of attacks against WebSockets are almost impossible if the transport is secured.

Create Web Extension

See below Figure 3: Select “WSS” for Transport when adding a Web Extension, then save and activate the settings.

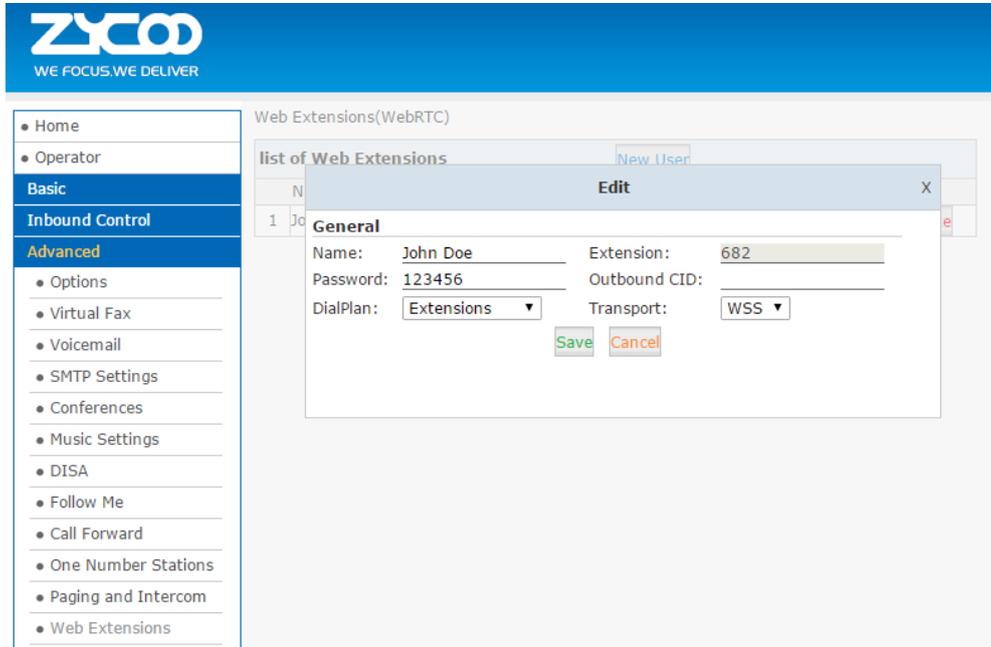


Figure 3. Create Web Extension

Step 3:

Input Web Extension (Name), registration address (SIP URI), registration port of WebRTC, use WSS(wss://192.168.1.65:8089/ws) by default in WS URI (192.168.1.65 is the IP of IP PBX). See below Figure 4.

The image shows a registration form for 'Webphone'. The form has a light brown background with the 'Webphone' logo at the top. Below the logo are four input fields, each with a red question mark icon to its right. The first field is 'Name' with the example 'i.e. Homer Simpson' and the value 'John Doe'. The second field is 'SIP URI' with the example 'i.e. sip:homer@your-domain.com' and the value '682@192.168.1.65'. The third field is 'SIP password' with the example 'i.e. sip:homer@your-domain.com' and the value '.....'. The fourth field is 'WS URI' with the example 'i.e. wss://your-domain.com:8089/ws' and the value 'wss://192.168.1.65:8089/ws'. At the bottom right of the form is a link for 'advanced settings'.

Figure 4. Registration

Step 4:

Once registration has completed a user can dial and make calls. See below Figure 5.

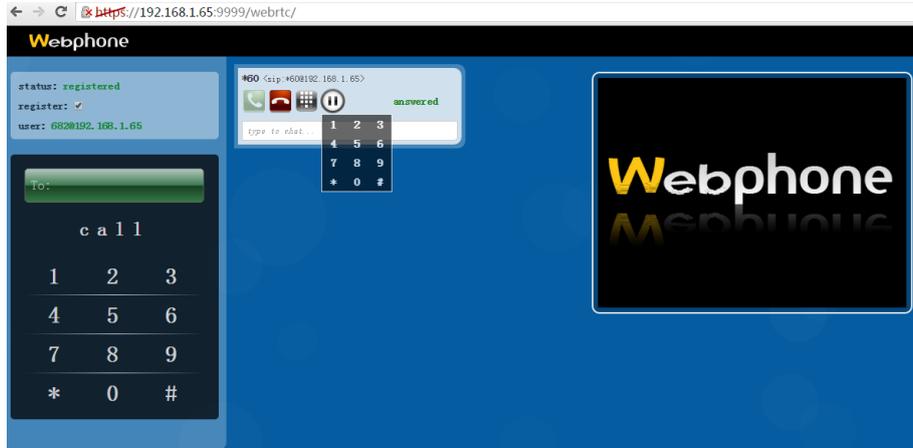


Figure 5. Make Calls from Web Extension

Notice: If you want to use transport protocol WS (not recommended), configure as below Figure

6.

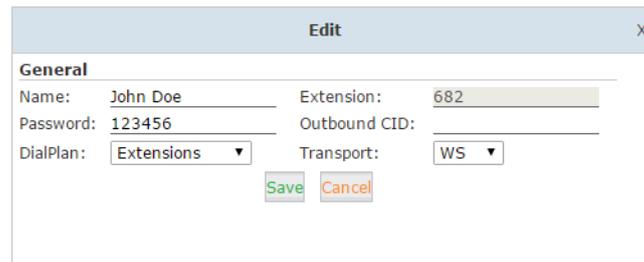


Figure 6. WS Transport Protocol

When you register a web extension using WS as transport then you must set the "WS URI" to "ws://192.168.1.65:8088/ws" as below Figure 7.

Figure 7. WS Registration

WS does not require encryption, however, Google Chrome has its security mechanism configured so that you have to manually set Google Chrome to load unsafe scripts to proceed. This is achieved by first clicking the shield icon on the right side of the address bar, and then click “Load unsafe scripts” as below figure 8.

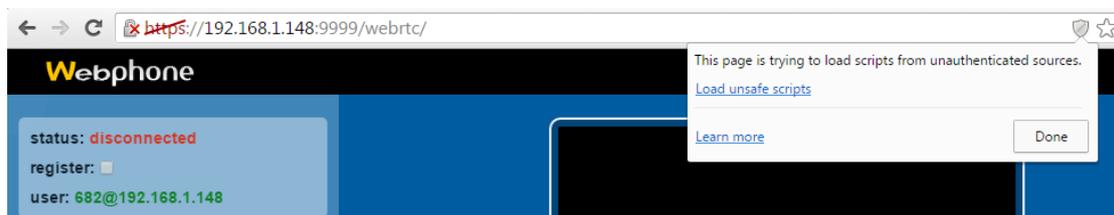


Figure 8. Load Unsafe Scripts

The page will now reload and you have to complete the registration process again after which you’ll be able to register.

2) Coovox PBX is behind Router NAT

Under this deployment scenario, WebRTC requires the support of ICE server. In the WebRTC UI firmware, ICE client has been configured through the coordination of Google (stun server) and Zycoo (turn server). If you have your own ICE server, then configure through “Advanced Settings” on the login page. See below Figure 9:

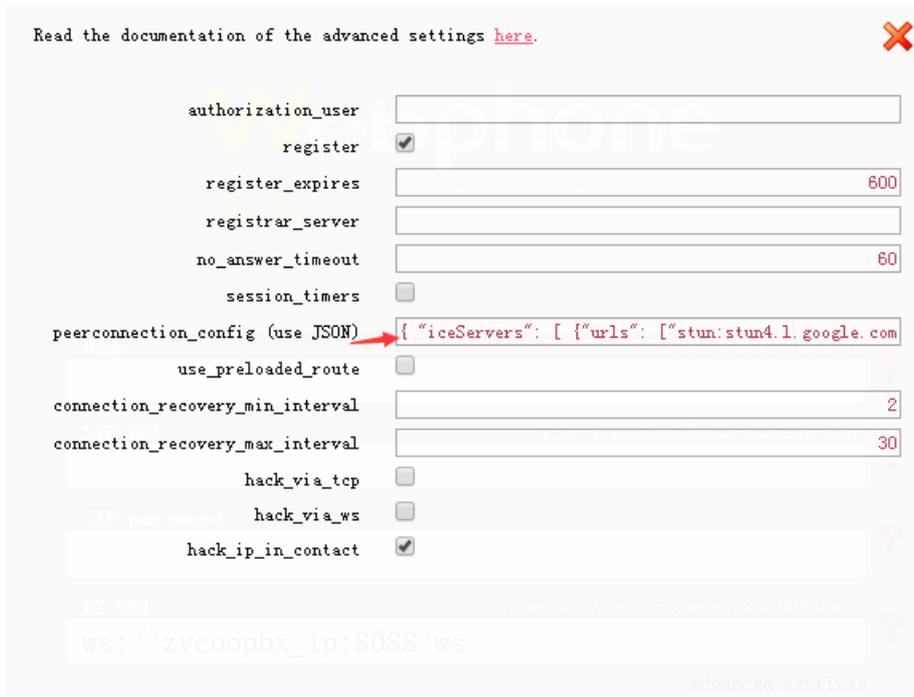


Figure 9. Modify ICE server settings

Please use below settings as an example (only one line of code is used for Zycoo test). You need to change this based on your iceServers parameters while retaining this format. Close after completing this field.

```
{ "iceServers": [ {"urls":
["stun:stun1.l.google.com:19302"]}, {"url": "turn:117.176.159.163", "username": "toto", "credential": "password"} ], "gatheringTimeout": 2000 }
```

As you are behind a router, you are required to configure the port on NAT router mapping to WebRTC registration port 8089 and Web visiting port 9999(default) of Coovox IP PBX.

Then complete the registration with public address, see below Figure 10:



Figure 10. Register with Public IP Address

Method 2 Deploy WebRTC UI on Web Server

Requirement:

Deploying in this scenario is only recommended for highly experienced technicians.

Objective:

Deploying WebRTC UI on your own Website is a convenient way for customers to ask for direct support /service from the website.

Step1:

Download original reference code ClickCall.tar.gz, note that you are required to modify some code.

Please see below assets directory structure in Figure 11.

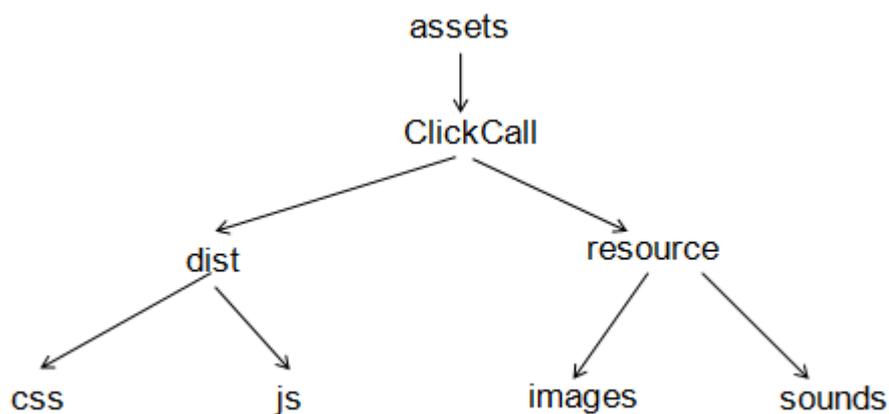


Figure 11. Reference codes directory structure

Modify the file under css directory as required to create Web format; you can use UltraEdit to modify the file "style.css" under css directory and "clickCall.js"; "gui.js"; "init.js" three files under js directory then complete registration information and links.

1. Modify file clickCall.js: set www.zycoo.com as your Web IP or domain.

```
urls.rtcninja =  
'http://www.zycoo.com/assets/ClickCall/dist/js/rtcninja.js';  
urls.jquery =  
'http://www.zycoo.com/assets/ClickCall/dist/js/jquery.js';  
urls.i18n =  
"http://www.zycoo.com/assets/ClickCall/dist/js/jquery.i18n.js";  
urls.messagestore =  
"http://www.zycoo.com/assets/ClickCall/dist/js/jquery.i18n.messages  
tore.js";  
urls.jssip =
```

```
'http://www.zycoo.com/assets/ClickCall/dist/js/jssip.js';
urls.init =
'http://www.zycoo.com/assets/ClickCall/dist/js/init.js';
urls.gui = 'http://www.zycoo.com/assets/ClickCall/dist/js/gui.js';
urls.css =
'http://www.zycoo.com/assets/ClickCall/dist/css/style.css';
```

2. Modify registration information in the file init.js (at the end of the file).

```
var user = local_config.user || {
    username: '682',
    password: '123456',
    host: '117.176.159.157',
    wsPort: '8088',
    wsProtocol: 'ws'
};

createUA(user);

phone_call_button.click(function (event) {
    console.log(event);

    varivr_num = local_config.ivr_num || '910';

    GUI.phoneCallButtonPressed(ivr_num);
});
```

Note: Modify username as Web Extension; password as Web Extension's password and also modify host as the IP address of Coovox IP PBX. Keep all other settings unchanged.

If Coovox IP PBX is behind router NAT, the port should be mapped to the internal IP PBX's 8088 port. (Above settings are just for reference)

3. Modify the link information in gui.js file: Search www.zycoo.com in the file and change it to your Web server IP or domain.

Modify the click icon as required: search icon-register.png in the file and set it as your required icon under image directory. E.g.: icon-register-en.png, icon-offline-en.png

4. Under dist css directory, modify the www.zycoo.com in the style.css file as your Web IP or domain.(Same way as previous)

Note: After modification, place the assets directory under the DocumentRoot of your Web server

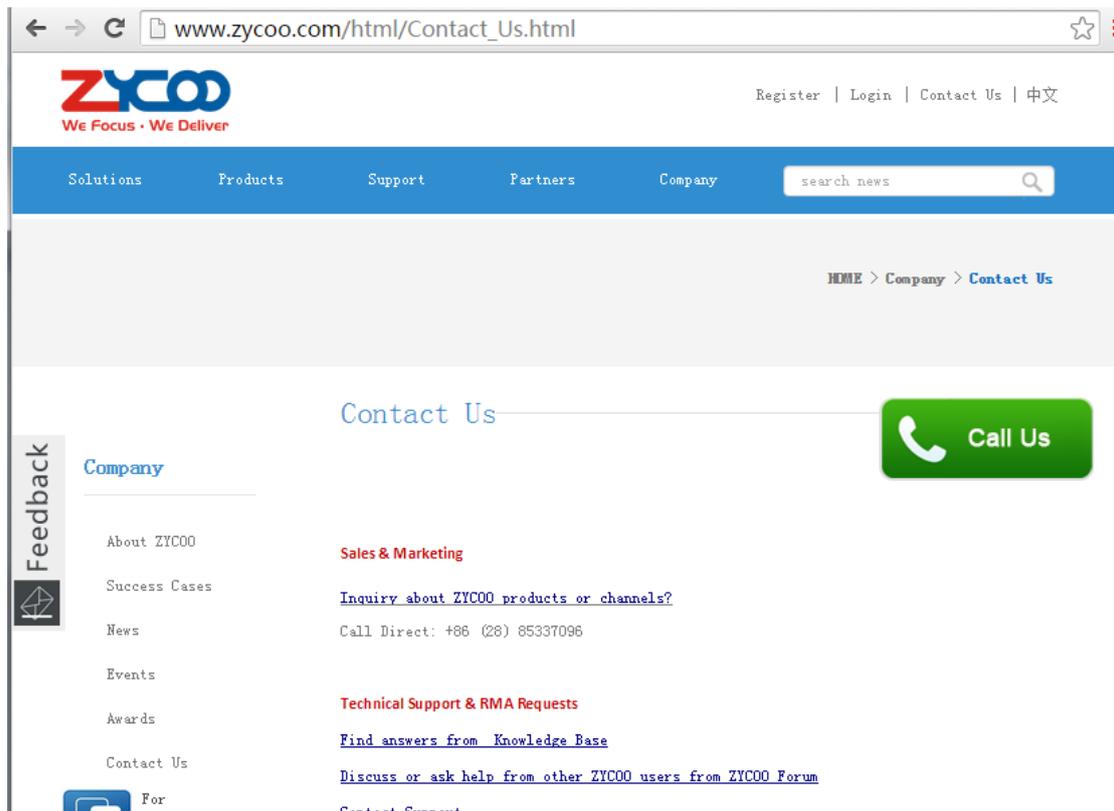
Step2:

Embed modified codes to the appropriate place on your Web page <body></body> . Please see below as reference.

```
<script>
document.write('<scriptsrc="http://www.zycoo.com/assets/ClickCall/dist/js/clickCall.js">/script>')
</script>
```

Step 3:

Visit your Website. E.g.: Zycoo Website and you can see the icon Call Us.



The screenshot shows a web browser window displaying the Zycoo website's 'Contact Us' page. The browser's address bar shows the URL 'www.zycoo.com/html/Contact_Us.html'. The website's header includes the Zycoo logo with the tagline 'We Focus · We Deliver' and navigation links for 'Register', 'Login', 'Contact Us', and '中文'. A blue navigation bar contains links for 'Solutions', 'Products', 'Support', 'Partners', and 'Company', along with a search bar. Below the navigation bar, a breadcrumb trail reads 'HOME > Company > Contact Us'. The main content area is titled 'Contact Us' and features a large green 'Call Us' button with a telephone icon. On the left side, there is a 'Feedback' sidebar with a list of links: 'Company', 'About ZYCOO', 'Success Cases', 'News', 'Events', 'Awards', 'Contact Us', and 'For'. The main content area also includes sections for 'Sales & Marketing' with a link 'Inquiry about ZYCOO products or channels?' and 'Call Direct: +86 (26) 85337096', and 'Technical Support & RMA Requests' with links 'Find answers from Knowledge Base' and 'Discuss or ask help from other ZYCOO users from ZYCOO Forum'. A 'Contact Support' link is also visible at the bottom.

Appendix: Browsers Support WebRTC

Zycoo Test Result:

Browser	Version	Test Condition(Register, dial, pickup, hangup, hold, second dial)---http	Test Result
Google Browser (Andriod Mobile)	38.0	OK	Success
	40.0	OK	Success
	45.0	OK	Success
	46.0	OK	Success
	47.0	Register succeeded, but inbound/outbound call failed	Fail
Google Browser (PC)	38.0	OK	Success
	39.0	OK	Success
	40.0	OK	Success
	41.0	OK	Success
	42.0	Outbound call-single pass and no voice on Web Extension side; No auto-hangup, inbound call is normal for both sides.	Fail
	43.0	OK	Success
	44.0	OK	Success
	45.0	OK	Success
Opera Browser (PC)	27	OK	Success
	28	OK	Success
	29	No voice for outbound call whatever internal or external calls. Single pass; inbound call is normal.	Fail
	30	OK	Success
	31	OK	Success
	32	OK	Success
	33	OK	Success
	Firefox Browser	All version	Test result is not good enough. Many bugs existed in Firefox while supporting WebRTC. Firefox is not recommended.
Opera Browser (AndriodMobile)	All version	Doesn't Support WebRTC	Fail
IE Browser	All version	Doesn't Support WebRTC	Fail