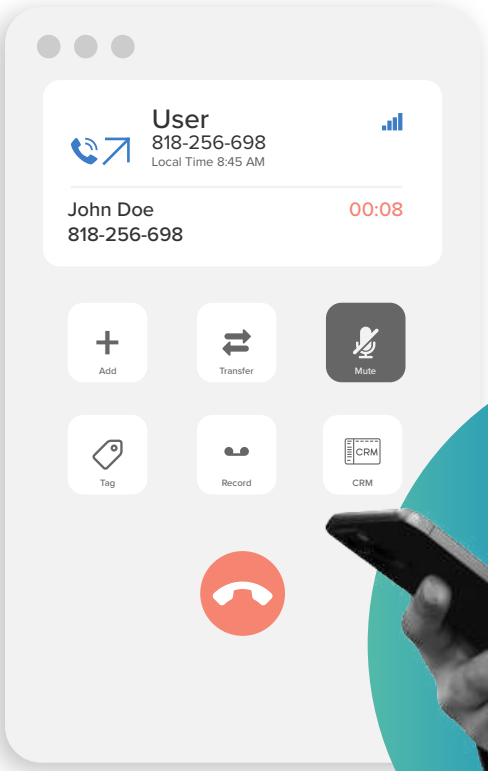
ecosmob
makes your vision a reality

Leveraging **React Native, WebRTC, and OpenSIPS** to develop cross-platform **softphone** applications

Whitepaper



The Abstract

Building a Softphone for a mobile platform is a tedious job. It requires access to native APIs, which need to use native languages for developing and maintaining applications for each platform. In the past few years, react native and WebRTC have evolved and now have capabilities to develop cross-platform softphone applications. There are some limitations of react native and WebRTC which can be resolved with OpenSIPS. Here, we will dive deeper into all the open-source libraries which work on top of React Native and WebRTC, along with OpenSIPS that can be leveraged to develop such applications.

Introduction

WebRTC and React Native have gained major traction in a couple of years, but they have some limitations that need to be addressed. This paves the way for OpenSIPS, which can be used to develop Softphones for mobile applications with all the more diverse features.

Problem Definition

A few techniques why React Native and WebRTC are taken as a first approach are: React Native code is reusable, cost-effective, there can be live reloading, high development speed, and most importantly, it is easy to learn.

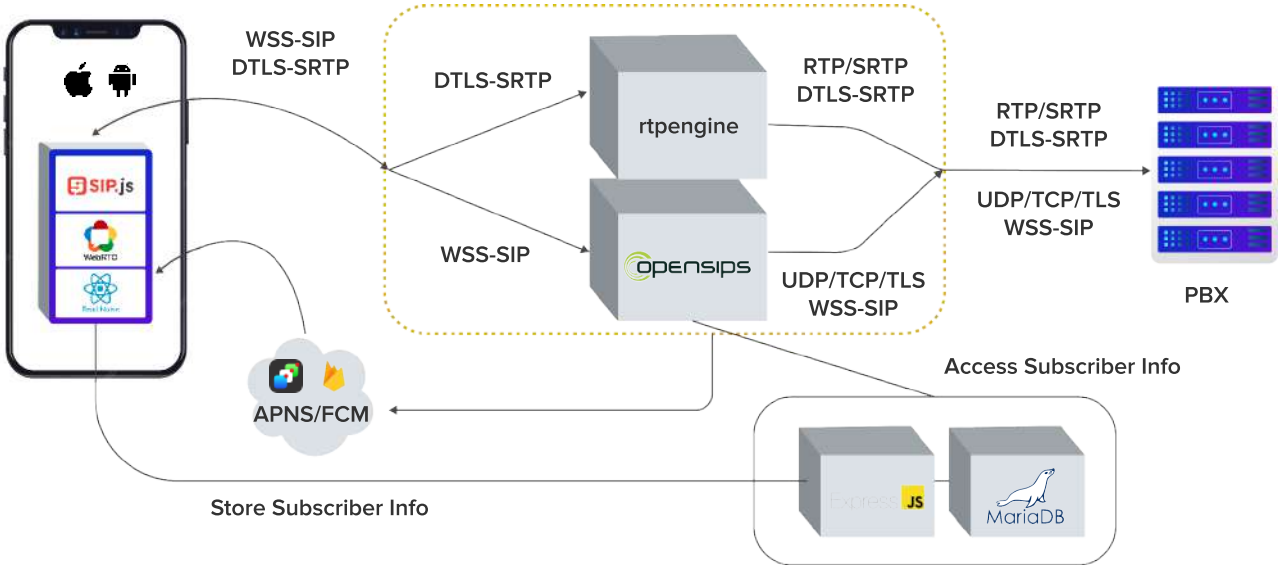
Similarly, WebRTC has a secure voice and video calls-making ecosystem; data can be shared, the technology tweaks itself per any network conditions, and it is platform & device-independent.

However, there are certain limitations of React Native and WebRTC while developing softphones for mobiles. This white paper targets to delve deeper into the technology and provides a solution for this subject with OpenSIPS technology.

High-Level Solution

OpenSIPS is a diverse and multipurpose signaling server. SIP is the most widely used protocol in VoIP, and OpenSIPS is accepted as a leader in VoIP platforms based on SIP. The whole VoIP ecosystem is now shifting towards IP and telephony is estimated to be revamped in the next decade. SIP has ushered this revolution, and it is one of the primary protocols of the next generation. The following diagram shows the best-fit approach to how OpenSIPS enables the development of softphones and makes calls.

Technology Stack and Architecture:



In more depth, OpenSIPS can translate wss to TCP/UDP, etc., before relaying it to PBX when it receives a register request from App. Android and iOS will not allow an application to run forever to save battery.

uac_registrant registers on behalf of the application and stays registered even if the application itself is not registered with opensips

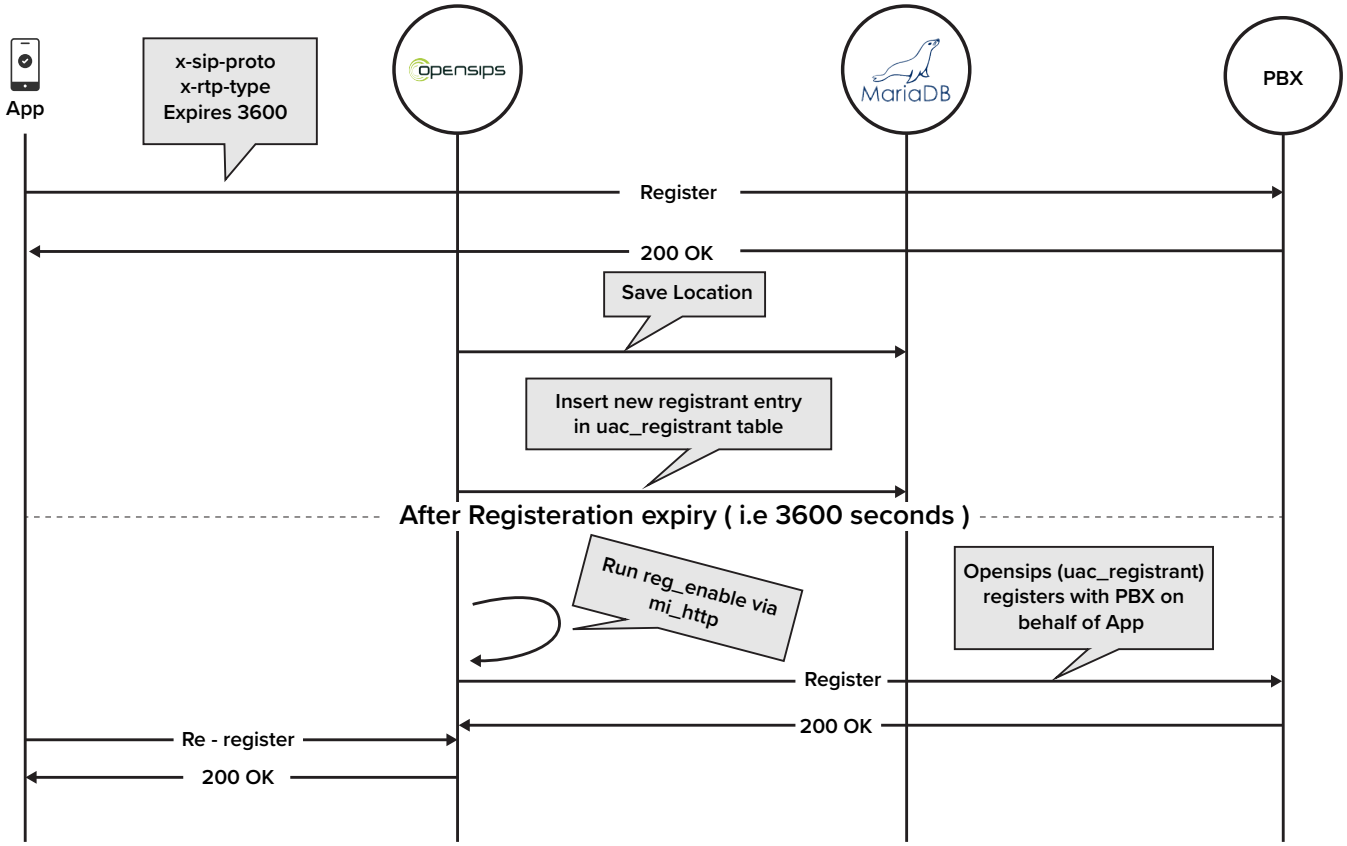
When OpenSIPS receives a call from PBX, it can send a push notification to wake up the application and deliver the call.

Solution Details

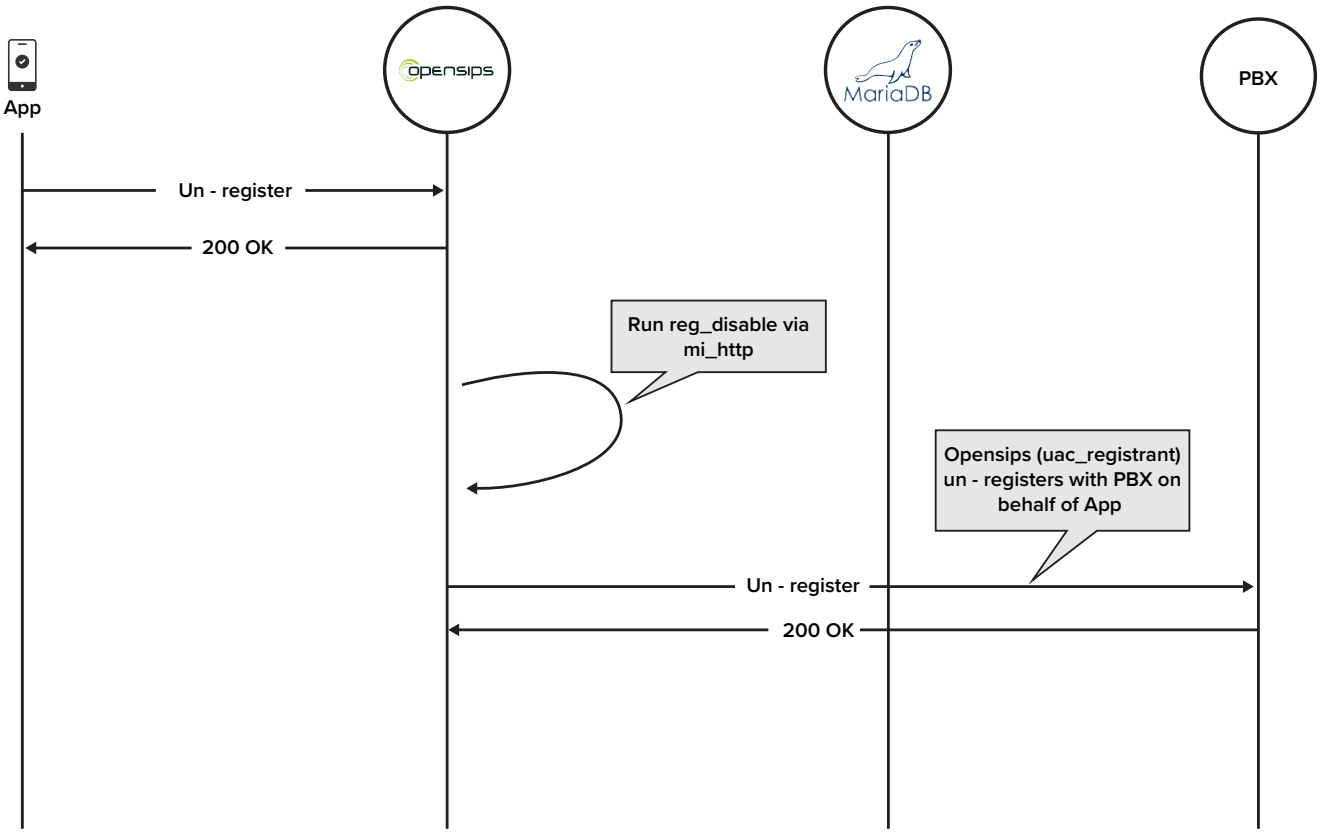
Registration Flow :

As a gateway OpenSIPS always stays registered with PBX. This way App does not have to stay registered with PBX all the time. When OpenSIPS receives registration request from App , the following happens :

- OpenSIPS will authenticate the registration request
- If authentication is successful , It will save location of App in its database
- OpenSIPS will send registration request to PBX on behalf of App with the help of uac_registrant module



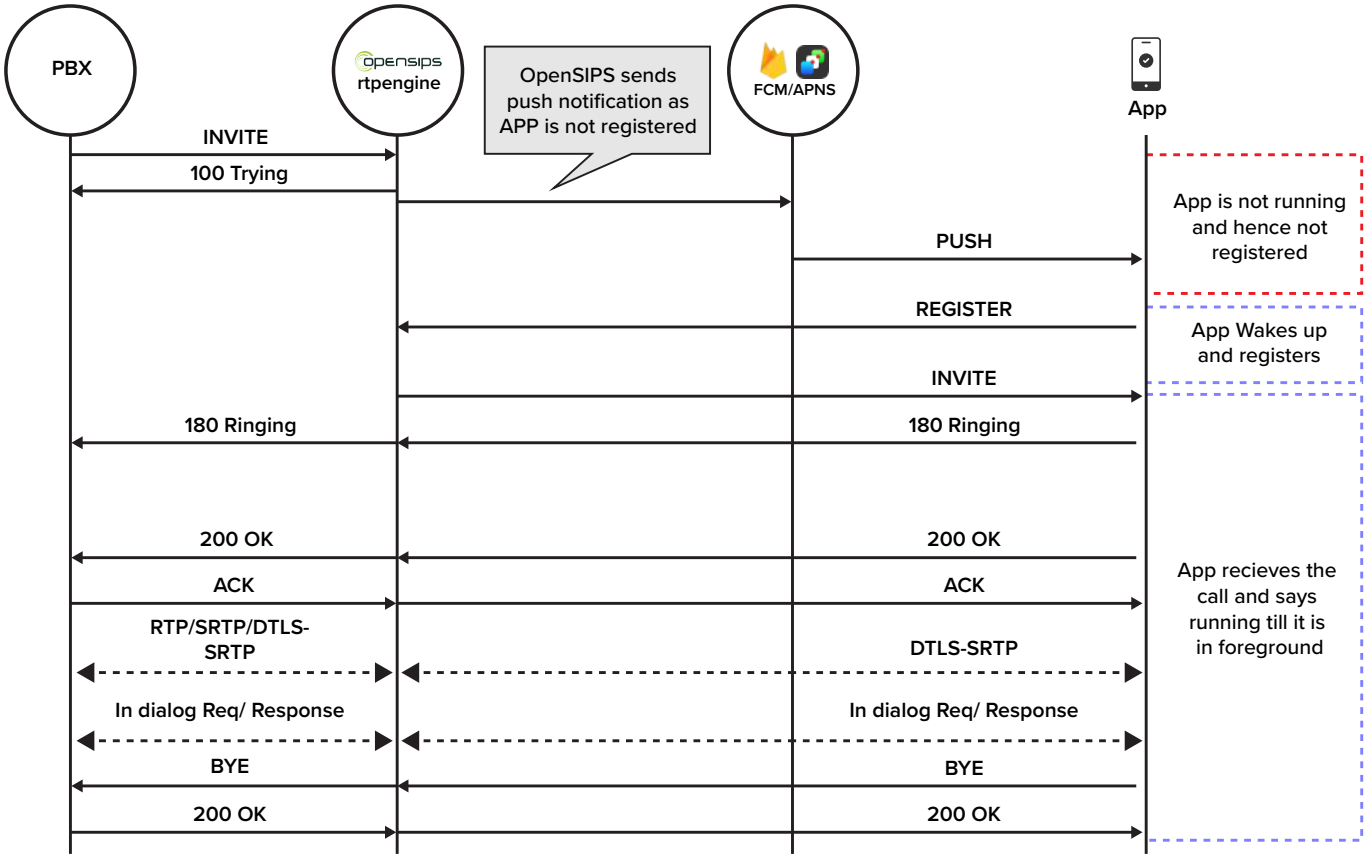
Un-Registration Flow :



Call Flow :

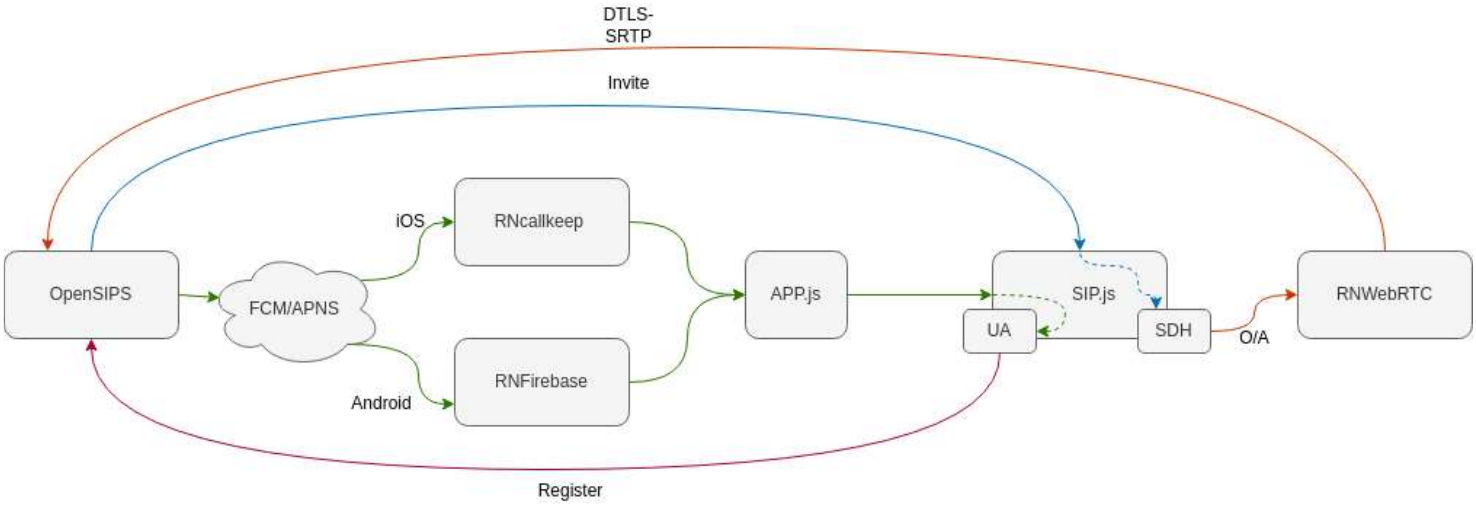
This diagram describes the call flow where the App is in killed mode and hence not registered but as OpenSIPS always stays registered with PBX , PBX does not have to care about it. When PBX wants to send call to App and if application is not registered then following happens :

- OpenSIPS will received Invite from PBX
- OpenSIPS will not find location of App in its database , hence it will know that App is killed
- OpenSIPS will send PUSH notification to wake up application
- App wakes up on receiving PUSH notification , it will send register request to OpenSIPS
- Register request sent by App notifies OpenSIPS that application is awake and reachable
- Finally OpenSIPS forwards Invite packet to application and hence call gets delivered to the App



Application Anatomy

In this diagram we can see various components involved on the App side.



Push notifications :

- React-native-callkeep - iOS CallKit framework and Android Connection Service for React Native
- https://github.com/react-native-webrtc/react-native-callkeep

Call Control :

- React-native-incall-manager - Handling media-routes/sensors/events during a audio/video chat on React Native
- https://github.com/react-native-webrtc/react-native-incall-manager

WebRTC

- React-native-webrtc - The WebRTC module for React Native
- https://github.com/react-native-webrtc/react-native-webrtc

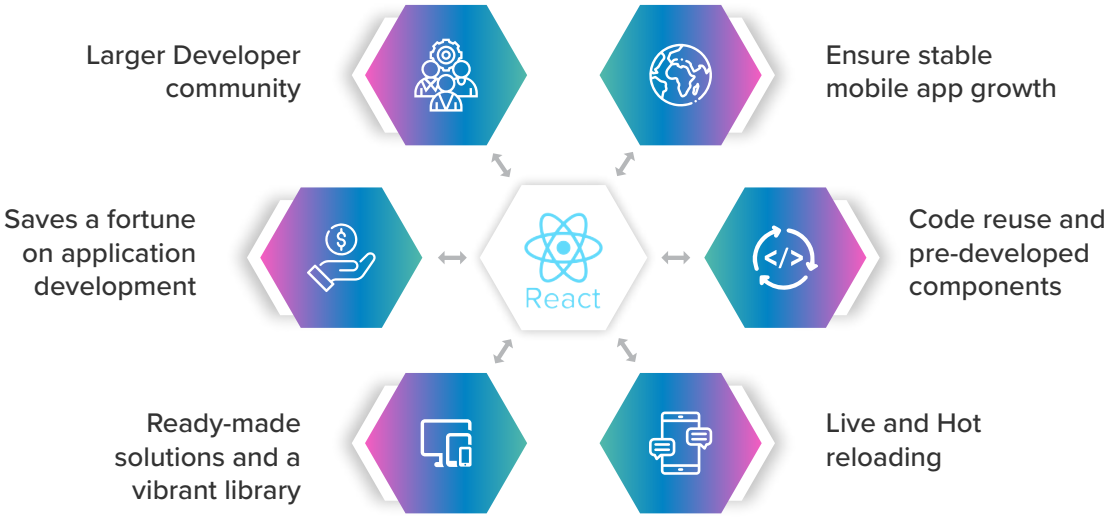
Signalling

- SIP.js - A simple, intuitive, and powerful JavaScript signaling library
- https://github.com/onsip/sip.js

Business Benefits

How react native is better than Native

The pivotal point to be considered while choosing between React Native and Native is platform dependence. In React Native, a single code is written for Android and iOS, while Native needs different codes for each platform. At the same time, when the cost is brought to light, React Native is 35% more cost-effective than Native. Apart from this, React Native has many open-source libraries of pre-designed modules that accelerate the development process.



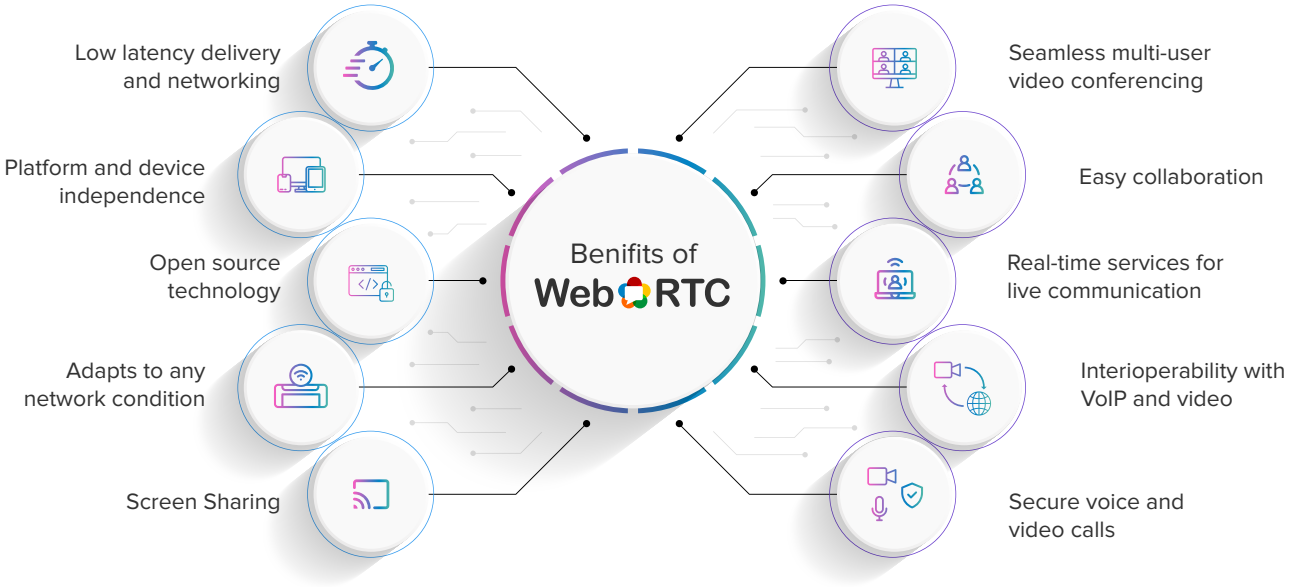
How push notifications can help save battery

With push notifications, the application is in use only when required. It implies that an app doesn't need to be up and running to see the notifications.
A smartphone user can see the notification even when their phone is locked or app is not running.

Why use WebRTC over DTLS-SRTP

Even though embedding audio and video in the browser via WebRTC without third-party plugins is exciting, it poses potential security breach threats. getUserMedia, RTCPeerConnection, and RTCDataChannel are the few APIs used by WebRTC to establish a connection between peers. TLS (Transport Layer Security) is the standard for web encryption, taken into account for the aim of such protocols as HTTPS. TLS is used for the reliable transport mechanism of (TCP) Transport Control Protocol, but VoIP apps (and games, etc.) typically utilize unreliable datagram transports such as UDP (User Datagram Protocol).

Since basic RTP does not hold any built-in security features, it is forbidden by WebRTC specifications. WebRTC thus utilizes SRTP for the encryption of media streams rather than DTLS. This is because SRTP is a simpler alternative to DTLS. However, the actual SRTP key interaction is initially performed terminally with DTLS-SRTP, enabling the detection of any MiTM (Man in The Middle) attacks.



What are the advantages of using open-source software instead of proprietary?

The open-source app is flexible and agile compared to proprietary software since the code is open to all. Speed of accessibility is fast in open-source, and the apps are cost-effective since no dedicated resources are required that develop targeted applications' code. In open-source, the apps can start small, relying on the community codes, and can be scaled as needed. It is challenging to adhere to security standards while following the development standards. Still, the regular revision of open-source codes by expert developers and peers makes it a robust practice and enhances the code most robustly.

Summary

In this whitepaper, we delved deeper into the open-source libraries that thrived on top of WebRTC and React Native in sync with OpenSIPS to develop softphones. Although this turns out to be a tiresome process, it becomes a bit affordable with the latest technology. Cross-platform softphone applications have made the calling experience seamless, and in this write-up, we have tried to explain what approach to follow to develop it with OpenSIPS.

If you wish to develop a cross-platform softphone for your business, you can contact us at sales@ecosmob.com. We can arrange a free consultation for you and look into your requirements.

REACH US @

INDIA
501-503, Binori B Square 1, Nr. Neptune House,
Ambli -Bopal Road, Ahmedabad-380058,
Gujarat, India.

USA
300 SE 2nd Street, Suite 600
Ft. Lauderdale, Florida, USA 33301

SOUTH AFRICA
340 witch hazel avenue centurion
South Africa.

CANADA
1285 West Broadway, Suite 600,
Vancouver, BC, V6H 3X8

PHONE
+1-303-997-3139
+91-7778842856
+1 (604) 900-8870
+27 0871353659

MAIL US AT
sales@ecosmob.com

VISIT US