Openpath Mobile Access SDK

Table of contents

Our SDKs are currently available for **Beta** testing and are subject to change.

Summary

The **Openpath Mobile Access SDK** lets your mobile app support Openpath readers and smart hubs, whether your app is for iOS, Android, or cross-platform using React Native.

The SDK provides support for the following functionality:

- Provision one or more users with valid Openpath mobile credentials.
- List entries (readers and smart hubs) that a provisioned user has access to.
- Unlock an entry.

The SDK establishes and manages all communication (over BLE, Wi-Fi, and cellular network) with Openpath readers and smart hubs.

Using the SDK

We provide different zip files, depending on what you are using:

- `vX.X.X-ios.zip` - SDK for iOS (can be used with or without Allegion support)
- `vX.X.X-android.zip` - Android SDK
- `vX.X.X-react-native.zip` - SDK wrapper for React Native

When unzipped, each folder contains a top-level `/docs` folder that has its respective documentation in it.

Android documentation includes a `javadoc` as well.

iOS documentation is provided in two formats: The `OpenpathMobile.doccarchive` package can be viewed in XCode. The `html` directory contains HTML documentation, which can be viewed by opening `index.html`.

React Native documentation is provided in HTML format. Open `docs/index.html` to get started.

Provisioning Workflow

This section describes how to provision your app with an Openpath user's mobile credential. Your mobile app uses this SDK and your cloud service calls Openpath REST APIs.

Setting up an API service user to call Openpath REST APIs

1. In your Openpath Control Center, create a user with Super Admin privileges. This will be a dedicated Service User. Use the Service User's credentials to call the Openpath REST API from your cloud service.

2. Create a way for end users to authenticate with your cloud service and request an Openpath token. This may be a REST endpoint, a web page, or anything else. Your cloud service in turn uses the Service User's credentials to request the token from Openpath's REST API, then passes it to the mobile app.

Please refer to our Openpath REST API documentation for details on the available Openpath REST APIs.

Generating a temporary mobile setup token

When a user logs in (or however you get a user onto your app), your cloud service calls the Openpath API to generate a temporary mobile setup token for that user's mobile credential.

`POST /orgs/{orgId}/users/{userId}/credentials/{credentialId}/generateSetupMobileToken`

You may need to make other Openpath APIs calls first (e.g., list users, list user credentials) to select the user you want to use, and the specific mobile credential to use for this app.

Passing the mobile setup token to the SDK

In your mobile app, once you have the token from your cloud service, call `provision` in the SDK. The app is now provisioned to use Openpath as this user.

NOTE: For security reasons, do not store the Service User's credentials in your mobile app. Your cloud server should be the one to authenticate and communicate with Openpath REST APIs.

Alternative workflow

If the user that you want to provision is already an admin and has a login to the Openpath Control Center, they can provision the app via their own email and password. In this scenario, the user is authenticating and generating the temporary mobile setup token with their own login auth credentials.

First authenticate through the login API:

`POST /auth/loginAll`

Then generate a temporary mobile setup token for their own mobile credential:

`POST /orgs/{orgId}/users/{userId}/credentials/{credentialId}/generateSetupMobileToken`

You may first need to make other Openpath APIs calls first (e.g., list user credentials) to select the specific mobile credential to use for this app.

Once the token is generated, it can be sent as a parameter to our `provision` SDK function.

Help

Please refer to this article for:

- how to access training materials for the Openpath Control Center (admin portal)
- how to submit a bug with the SDK
- how to get additional development/testing hardware