onelogin

OneLogin NAPPs Server

www.onelogin.com | twitter.com/onelogin

OneLogin, Inc. | 150 Spear Street, Suite 1400, San Francisco, CA 94105

855.426.7227

Table of Contents

1. INTRODUCTION

Welcome to OneLogin's NAPPs server-side documentation. NAPPS is a framework for providing single sign-on capabilities to applications within their native launch environments. While our focus is primarily mobile operating systems like iOS and Android, NAPPS is not limited in scope to just those platforms. NAPPS is based upon an OAuth 2.0 framework.

There are a few key elements to the functionality of the NAPPs system:

Authorization Server - This is the IdP that authenticates the user and provides primary and secondary tokens. The authorization server also validates secondary tokens that are presented to it by the third party Resource Server.

Third Party Applications - Apps that reside on the device that require some form of login capability, provided by the Token Agent.

Token Agent - This is the software that resides on the device that communicates with the Authorization Server and the Third Party Applications that reside on the device.

Resource Server - This is the third party server at which a third party app sends an access request for a Secondary Token, which will eventually be directed to the Authorization Server.

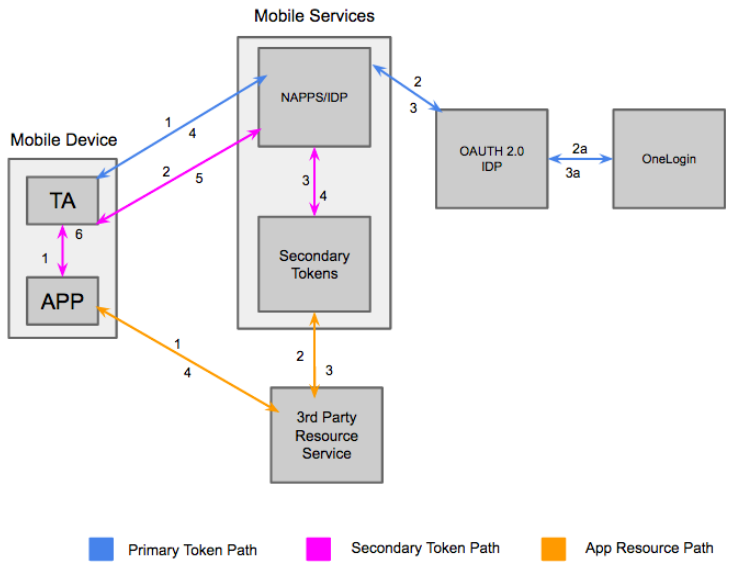
Resource Server Key - Generated by the resource server address within OneLogin, this secret key allows for verification of two things: that the resource server in question is still valid, and that the secondary token, provided by the Token Agent, is still valid. It is also used to register and de-register both items in question.

Primary Token - This is a token that is provided to the Token Agent after the user has been authenticated for the first time and upon refresh by the Authorization server. The token or parts of the token are used when making requests for the AppInfo list, Secondary Token requests, and refresh of the primary token request. The primary token is composed of the following:

- **access_token**: This is used to make requests to the AppInfo Endpoint.
- **token_type**:This must be a "bearer" token and support the OAuth 2.0 bearer token usage specified in [RFC6750]
- **id_token**: Primary ID token
- **expires in**: Expiration time of the access token in seconds
- **primary refresh token**:

Secondary Token: This token is provided to the third party applications by the Resource Server via the Token Agent.

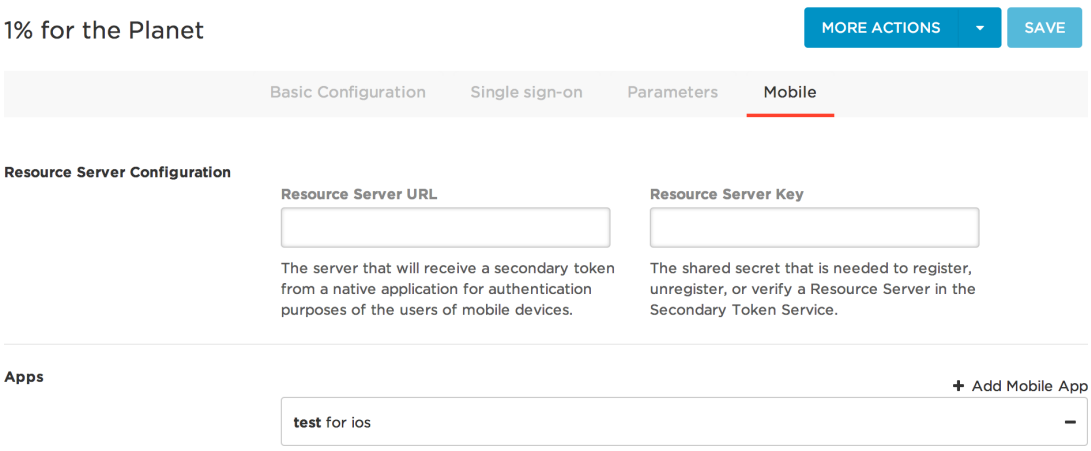
A data-flow diagram for the NAPP's mobile token and authentication systems is provided below:



2. UI ELEMENTS

2.1 ADDING A 3RD PARTY RESOURCE SERVICE

The interface for NAPPs is very similar to the typical OneLogin application connector, and can be setup via accessing the desktop portal. Under the new **Mobile** tab, you'll have access to fields that will allow you to configure your NAPPs connector, specifically by means of registering your **3rd Party Resource Server**. Shown below is the example mobile connector:

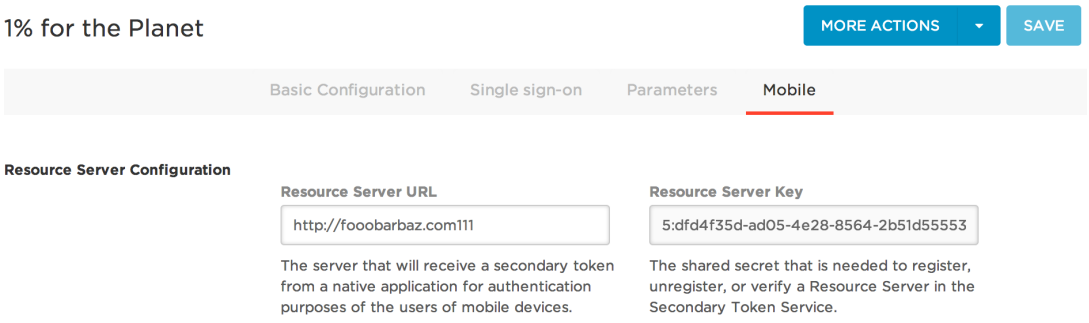


Within the **Resource Server URL** field, you will provide a custom address that will target a custom application resource server. This server will be responsible for creating and verifying secondary tokens before presenting them to the Authorization Server.

This can be considered the resource server 'registration' process. Once the resource server is registered within the UI, a Resource Server Key will immediately be generated that is associated with the Resource Server URL.

2.2 RESOURCE SERVER KEY RETRIEVAL

The resource server key will be generated immediately upon registration of the server. Shown below, we have an example Resource Server registered within the connector, as well as the resultant Server Key to the right:

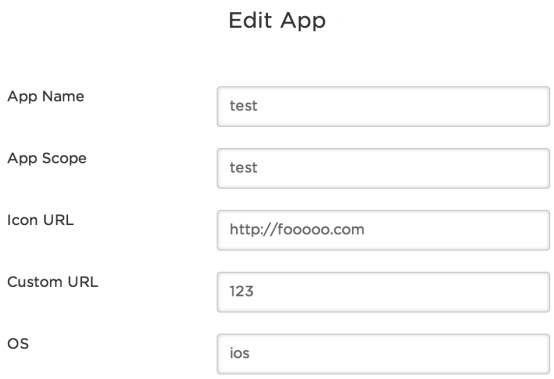


The **Resource Server Key** will be used to, aside from proving that the associated resource server is properly registered, validates if the resource server and secondary token are still both valid. This will be done through the public facing API which is covered in Section 3.

The next step will cover the management of specific configuration details that will allow access by the NAPPs enabled application connector to the various mobile devices.

2.3 MANAGING APPLICATIONS

At the bottom of the mobile connector page, selecting **+ Add Mobile App** will bring up the **Edit App** pane (shown below) where you'll be able to build out the mobile aspects of the connector in question.



Think of this pane as an application sub-connector that specifically target variou operating systems for application usage. Below are the required fields within the mobile **Edit App** connector pane:

App Name - This will be the application's name displayed at the bottom of the mobile connector tab.

App Scope - The scope is either the bundle ID or the package ID for the application in question, depending on whether its for Android of iOS respectively.

Icon URL - This is the target URL for the icon image that you want displayed for the mobile app.

Custom URL - This is the URL schema that defines the endpoint to which the token and authentication request is sent.

OS - Select between iOS or Android for the operating system you'd like this connector to support.

You may have up to three Mobile sub-connectors present (one for each mobile operating system), and with these properly configured, NAPPs can function successfully on those specifically configured mobile devices.

3. NAPPS API

The API endpoint is considered public, meaning NAPPS enabled Third-Party Resource Servers should be able to interact with the API.

This route is protected by the registered_rs (registered resource server) express middleware, which will ensure the requesting resource server has registered itself with the secondary token service via the Backbone UI. To perform an external request, ensure your resource server is registered, and that the secret-key header is set in the proper format, dsl_resource_server_id:secret_key.

- /api/v1/tokens/verify - verifies that the current token is still valid