Cogito Architecture

Stefan van den Oord Mark Spanbroek

October 3, 2017

Abstract

Cogito is a self-sovereign identity management app. It manages your digital identity, consisting of cryptographic key pairs for signing/verifying and encrypting/decrypting data, as well as attestations: claims about your identity. This document describes the architecture of systems that use *Cogito*. Specifically, it explains how identity information from OpenID Connect systems can be integrated in a *Cogito* identity. This way users can sign using a self-sovereign identity, which is provably backed by an identity from a third-party OpenID Connect server. Typical use cases are systems where users need to sign blockchain transactions using a corporate identity from an OpenID Connect server.

Use Case: Signing Data with OpenID Connect-backed Identity

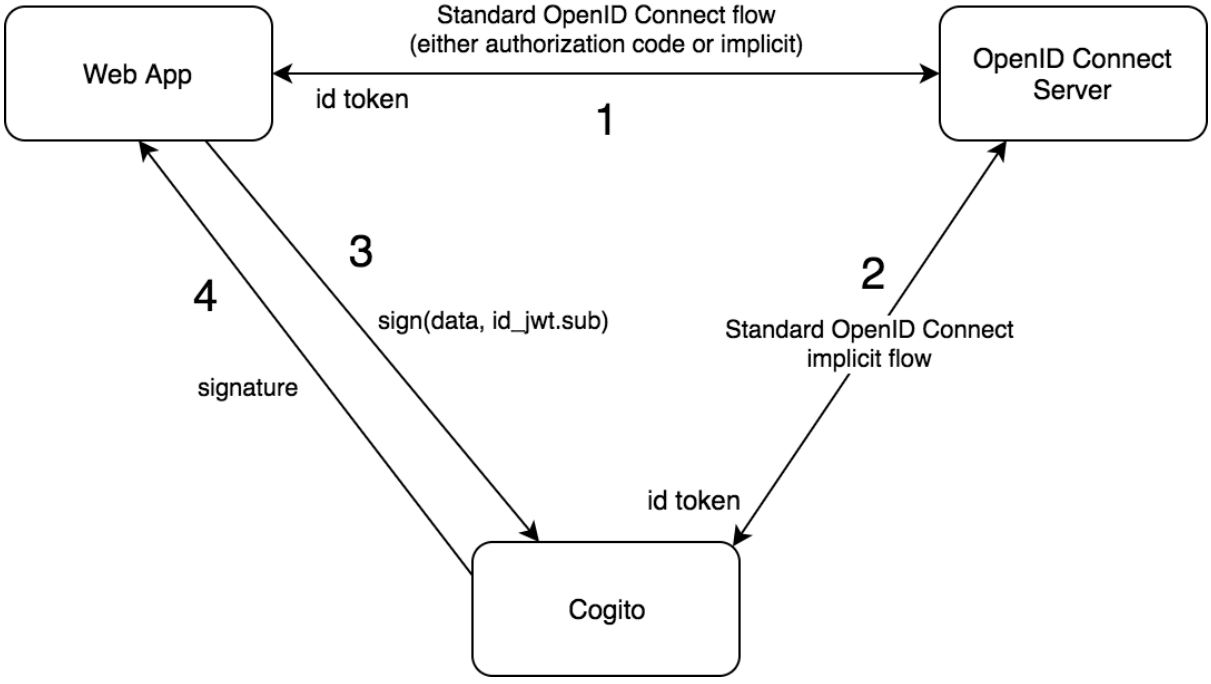

Figure 1: Signing data with Cogito

Attestations

Attestation from OpenID Connect Server

One type of attestation that users can have for their identity, is an attestation from an OpenID Connect Server. The most obvious use case for this is that a user gets an attestation from their company. In other words: the company signs a claim that indeed the user is an employee of the company.

This attestation can be added to the user's identity using the standard OpenID Connect *implicit flow*. The diagram "OpenID Connect Attestations" illustrates this.

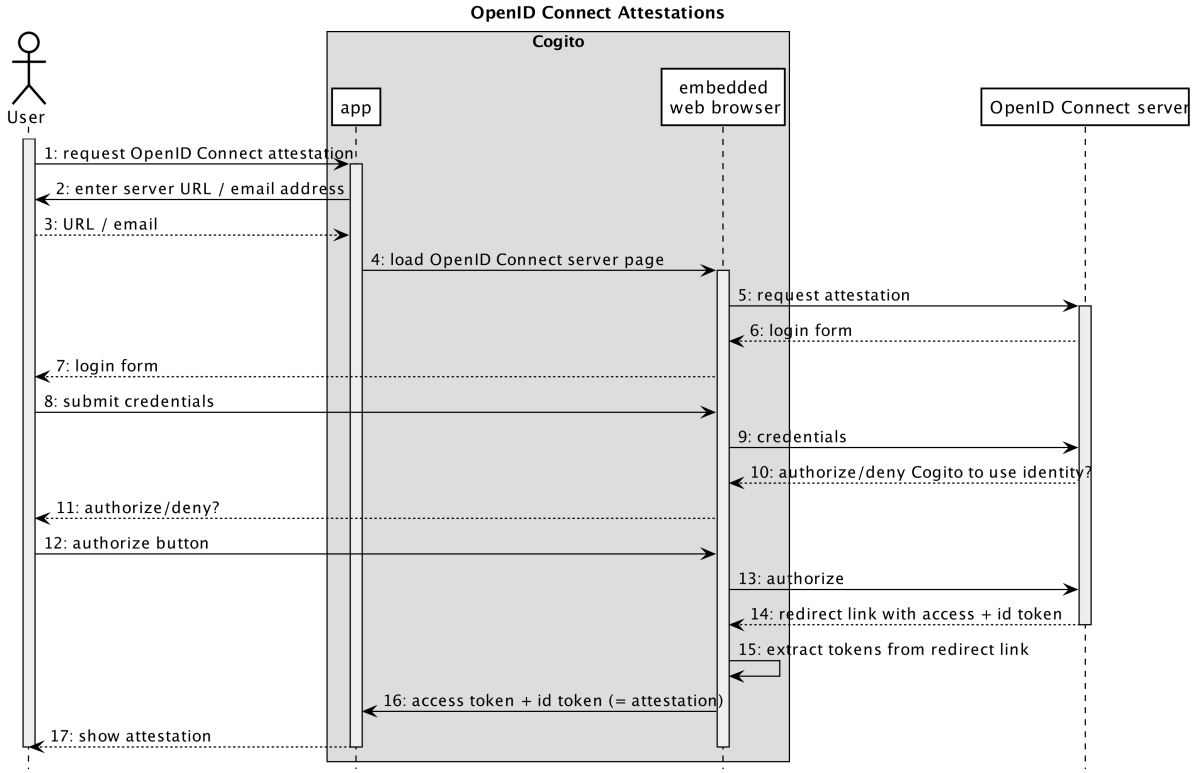

Figure 2: OpenID Connect Attestations

These are the steps displayed in the sequence diagram:

1. Inside the Cogito app, the user indicates that she wants to receive an attestation from an OpenID Connect server.
2. Cogito asks the user to identify which OpenID Connect server is to be used.
3. The user provides server URL or email address; in the case of an email address, it is converted to an OIDC Connect server URL by the app.
4. Cogito tells the embedded web browser to load the OpenID Connect server URL.
5. Here the standard OpenID Connect *implicit flow* starts. The browser sends the OpenID Connect authentication request.
6. The OpenID Connect server responds with an HTML login form.
7. The form is displayed to the user.
8. The user submits the form with their credentials.
9. The credentials are sent to the OpenID Connect server.

10. The OpenID Connect server verifies the credentials, and if OK responds with an HTML page asking the user to allow Cogito to user their OpenID Connect identity.
11. The page is displayed to the user.
12. The user reviews the information and clicks the 'authorize' button.
13. The browser forwards to the OpenID Connect server.
14. The OpenID Connect server redirects the browser to the URL specified in step (5), passing the access token and id token in the URL fragment.
15. The browser extracts the tokens from the URL fragment...
16. ... and sends them to the app.

TODO: Other Attestation Types