Social Authentication

C. Lewis (@ctjlewis), for Moonbounce

November 22, 2021

Abstract

Notes on social authentication via Auth0 or Firebase.

Contents

Problem **1**

Research **2**
- Demo flow . . . 2
- Implementation . . . 2
  - At the application layer . . . 2
  - At the authentication layer . . . 2
- Progress . . . 3
  - Firebase . . . 3
  - Auth0 . . . 4

Problem

Given a pseudonymous user and various social OAuth integrations, what is the best way to architect a social oracle infrastructure?

Research

Demo flow

For the MVP, we want only to allow a user to:

1. **Authenticate** via one or more OAuth channels
2. **Update** their address for one or more networks in the social oracle service.
3. **Prove** their association with an address by sharing a link to the service which demonstrates various social channels' connection with a given address.

Implementation

Originally, it was suggested that this system be implemented with Firebase, i.e. by allowing a user to authenticate with Firebase Auth and update their addresses in a Realtime Database or Firebase Storage DB.

At the application layer

Using Firebase to associate user identities with addresses effectively involves building the social oracle itself as a Firebase application, which comes with several downsides:

- Locked into the Firebase ecosystem
- Verbose solution that relies on multiple pieces of legacy internet infrastructure
- Exponential security risk due to several moving parts

Though it might be appropriate to use Firebase for the core application client, it seems that it might make more sense to associate this data more closely with user identity directly.

At the authentication layer

While both approaches were researched and starter repos for both have been created, it seems more appropriate to store user addresses (and other core on-chain identity) metrics **at the auth layer directly**.

- Nearly infinite extensibility
- Highly secure (as secure as we can get)
- Supersedes all application logic: On-chain identity is merged into the auth protocol
  - Potentially possible to extend Auth0's core authentication with on-chain 2FA[1]

[1] In practice, this looks like evaluating, at the authentication layer, e.g. what ETH addresses are linked to a given user's identity token, and require a signature from one of those addresses to authenticate.

The difference in practice is that between asking the service to query a Firebase DB:

And users' Auth0 session tokens containing this data itself, like so:

```
{
  "user": {
    "nickname": "ctj.lewis",
    /** ... */
    "picture": "https://lh3.googleusercontent.com/a-/AOh14Ghu2aqZ2STwrynZ5UZeRfO_RV38kyA1mWy
    "locale": "en",
    "updated_at": "2021-11-20T22:32:02.737Z",
    "email": "ctj.lewis@icloud.com",
    "email_verified": true,
    "sub": "[REDACTED]"
  },
  "idToken": "[READACTED]",
  "accessTokenScope": "openid profile email",
  "accessTokenExpiresAt": 1637708315,
  "token_type": "Bearer",
  /*
   * Notice the on-chain identity is merged into the Auth0 identity token
   * directly.
   */
  "app_metadata": {
    "addresses": {
      "ETH": "0xABC1231a41d10709daB9BD53080E624B8C2bDD5f"
    }
  }
}
```

In the latter case, we are unrestricted in how we build around the authentication layer, as Auth0 can be integrated into virtually anything, and does not depend on Firebase services to run.

Progress

Firebase

https://github.com/ctjlewis/moonbounce-firebase

To finish the oracle demo with Firebase, we must complete the following:

- Update NextJS-Firebase template to be compatible with Datastore projects (was originally built for Firebase Storage, now deprecated)

  https://cloud.google.com/firestore/docs/firestore-or-datastore

- Implement demo flow

  - Add endpoint for users to update addresses and other on-chain identity information by writing to Datastore
  - Add endpoint to users to view addresses and other on-chain identity information by reading from Datastore
  - Add simple frontend

Auth0

https://github.com/ctjlewis/moonbounce

- Implement working logic to call the `update:user_app_metadata` Auth0 API action, either with an Axios request or using the `ManagementClient`
- Implement demo flow
  - Add endpoint for users to update addresses and other on-chain identity information by updating their `app_metadata` JWT claim
    * Create Auth0 Rule to add `app_metadata` to ID token on `postLogin`

      https://auth0.com/docs/users/metadata/manage-metadata-rules
  - Add (cached) endpoint for users to view addresses and other on-chain identity information by reading from another user's `app_metadata` identity claim