OMI Identity #380

robertlong started this conversation in **R&D**

robertlong on May 18, 2021 Maintainer

OMI Identity

Contributors

- Robert Long, Independent Contributor

Summary

I'd like to kick off discussion around an interoperable identity specification. By identity I'm referring to an identifier that resolves to a document full of profile data. I'd like to scope the data contained in this document to two fields for now: display name, and avatar model. The spec will also cover verifying a user's identity.

The spec should have three core properties. It should be:

1. Addressable: Given a public identifier you should be able to resolve the document
2. Verifiable: You should be able to determine if the user you are interacting with, is the user associated with the id they are giving you
3. Well formed: This document needs to be specified in a standard format so that an application can resolve your profile data

Example Use Cases

In my work on Mozilla Hubs we purposely made our identities lightweight, both for privacy reasons and for the ability to integrate Hubs into other systems. As soon as people started hosting their own Hubs Cloud instances we noticed the friction associated with even this lightweight identity system. You need to set your name and avatar every time you move to a new server. This generally requires uploading your avatar model or going through and finding an avatar again. People are generally very attached to their avatars, it represents their digital self. When people find something that they identity with, they want to use it everywhere.

People also want to be able to verify that a person are who they claim to be. In Hubs, unless your room is connected to a Discord room, you can set whatever name you wish. Including duplicates of a given name. You can spoof someone's identity very easily. In Hubs, most of the meetings we have are private and we can trust the users in the room. However, as we create more public spaces, being able to verify a user's identity becomes more important.

Implementation

Decentralized Identifiers (DID)

Decentralized Identifiers look like a perfect spec to use for address resolution and identity verification. It is really only missing the profile data model.

DID identifiers look like this:

```
did:example:123456
```

Where `did` is the scheme, `example` is the method, and `123456` is the id.

DID identifiers resolve to documents which contain data for verification of the subject via public key encryption, authentication as the subject of the identifier, or a number of other verification relationships expressed in the spec.

However, the documents are not intended for storing store personal data. Documents can contain an array of services which can then have access control for this personal data. However, linking to services with public personal data still seems to be discouraged.

I'm still a bit confused on how best personal data should be resolved from a DID and could use some help here. In the section of the spec on service privacy the authors recommend a single service url without any additional user identifiers. In our context, perhaps this is a home server domain and the DID identifier can then be used to get the public user profile data. **@dmitrizagidulin** I could use some of your help here.

You may also want to lock down your profile data so that only your close connections can see your avatar/display name. I'm not so sure how to express this within the DID ecosystem.

ActivityPub

Another solution could be to use ActivityPub. This is the protocol behind Mastodon and PixelFed. My public ID is: `@robertlong@mastodon.online` and from there you can find my public profile. The ActivityPub spec is already built to be extensible and add things like avatar info. **@wmurphyrd** already has an implementation of such a thing with immers.space.

Sensitive user profile information could be hidden except for mutual followers, which is a big plus. There's also a social graph which is appealing for the friends list spec I'd love to propose next.

I do have a few outstanding questions about ActivityPub though:

1. In a public space, how can one user verify the identity of another? If I'm connected with another user over P2P and another user broadcasts their id as `@example@mastodon.online` how can I check that they

are in-fact that user? **@cwebber** and **@dmitrizagidulin** I could use your help here. I see there's already Keybase support on Mastodon. What other methods of verification are already out there or could be used?

2. How much work is it going to be for apps to implement fetching user data from ActivityPub? **@wmurphyrd**, you've got very tight integration with ActivityPub in Immers.space, but what about for those apps which want a more lightweight integration? What options are out there?

Data Model

With any solution the profile data I think we should focus on in its simplest form is this:

```
{
  "displayName": "Robert Long",
  "avatar": {
    "mimeType": "model/gltf-binary+vrm",
    "url": "https://robertlong.me/files/avatar.vrm"
  }
}
```

Application specific avatars or an avatar collection are things we should discuss as well. We also should discuss supporting multiple identities.

I'd like to keep discussing this topic in this issue. If you have specific proposals for identity, go ahead and create another issue where we can discuss the implementation further. I'm hoping we can get a couple subject matter experts to weigh in and guide us towards the correct solution. If you are a possible implementer of an application that might want to use this identity specification, please let us know what your specific requirements are below.

Looking forward to talking more!

↑ 1 🎉 2 👀 2

15 comments Oldest Newest Top

wmurphyrd on May 19, 2021

Some notes on how these items are commonly implemented in the fediverse.

Addressable: We use webfinger (Mastodon explainer). This is the same web standard e-mail uses. Webfinger means you can translate an identifier to a URL by using a standarized API path. E.g. given `username@domain` you know that `https://domain/.well-known/webfinger?resource=acct:username@domain` will return a JSON document that tells you exactly where to fetch the user's full profile. I like this approach because it **does not** require standardization of every application's API routes for fetching or interacting with users, just the one "well known" route for dereferencing a human readable user identifier

Verifiable: Public key encryption, must commonly used with the http-signatures standard. (Mastodon explainer). The user profile (Actor object) retrieved from the URL you get via webfinger includes a public key, and all ActivityPub messages sent by that user are signed with their private key. With just pure ActivityPub + signatures you could verify that a specific user is truly in the room with you (a signed Arrive activity), but we'd probably want to add another signed message here on the realtime layer to also verify that a specific entity in the room is the right person

Well formed: That's ActivityPub itself - the Actor object. The standard gives you everything you need to exchange messages and is easily extensible

> How much work is it going to be for apps to implement fetching user data from ActivityPub? **@wmurphyrd**, you've got very tight integration with ActivityPub in Immers.space, but what about for those apps which want a more lightweight integration? What options are out there?

That's the purpose of the Immers server - it is a microservice that runs alongside your XR experience. It handles all ActivityPub compliance and federation, account registration, and OAuth2 authorization server so that users can login to your experience with accounts created on other experiences (with recently released access grant control for users), but it does not dictate anything about your XR experience. You choose what and how to integrate into your experience without having to worry about specs, authorization, and server-to-server communications.

↑ 1 👍 1 0 replies

wmurphyrd on May 19, 2021

@robertlong here's a theoretical example of using ActivityPub and DID together: https://github.com/WebOfTrustInfo/rwot5-boston/blob/master/draft-documents/activitypub-decentralized-distributed/activitypub-decentralized-distributed.md#distributed-identity

↑ 1 ❤️ 1 0 replies

lyuma on May 19, 2021 Collaborator

We need to make sure that whatever implementation of OMI identity we base our specification on be compatible with both proprietary use cases and open source licenses, be it "Immers" or one of the Mastodon based implementations.

About other implementations...

- From chatting with **@fire**, the possibility of using Matrix was brought up. There is a possible simplified implementation of Matrix identity in Jel by **@gfodor** , though from what I can see, it uses a **axa**, a fork of the Synapse matrix.org server. As this is also based on Synpase it would fail the "two-implementations rule". Perhaps Greg has some insight on a second implementation.

- OpenID Connect were brought up. I think the issue with them is they do not provide public/private keys for users. Also, it seems crippled: fire notes that in practice OpenIDC's design forces implementers to pick a small, finite set of identity providers (for example, Google, Facebook, Apple, Twitter, GitHub) and therefore, decentralized identity providers would not be compatible with almost all implementations in practice. For example, Fire notes that you can't go to Steam and login with your Mozilla identity provider because it's too niche.

There's also a question of complexity: the simpler the implementation, the easier it will be to port to other server-side languages.

In particular, if I have an existing account database or identity provider written in some language, I would need to be able easily integrate some implementation of the OMI Identity service into my existing backend.

↑ 1 0 replies

wmurphyrd on May 19, 2021

@lyuma could you expand on your concerns here:

> We need to make sure that whatever implementation of OMI identity we base our specification on be compatible with both proprietary use cases and open source licenses, be it "Immers" or one of the Mastodon based implementations.

↑ 1 0 replies

lyuma on May 19, 2021 Collaborator

Specifically, GPL and AGPL code cannot be embedded inside of a proprietary application without having implications regarding the release of linked source code.

For the purpose of OMI, we need to be inclusive of proprietary applications, so any code used as part of a standard would ideally be MIT or BSD type licenses.

↑ 1 0 replies

wmurphyrd on May 19, 2021

@lyuma In my opinion, OMI should not be endorsing any specific implementation, just choosing a standard. Then licensing isn't an issue, especially if it's from an open standards organization like W3C.

For ActivityPub in particular, there are open source implementations in several languages already. Here's some off the top of my head

General purpose libraries:

- activitypub-express - NodeJS, MIT (created by me)
- GoFed/activity - Go, BSD

Applications that include ActivityPub implementation:

- Mastodon - Ruby, AGPL
- Pleroma - Elixir, AGPL

1 0 replies

robertlong on May 21, 2021 Maintainer Author

Thanks for the responses!

> E.g. given username@domain you know that https://domain/.well-known/webfinger?resource=acct:username@domain will return a JSON document that tells you exactly where to fetch the user's full profile. I like this approach because it does not require standardization of every application's API routes for fetching or interacting with users, just the one "well known" route for dereferencing a human readable user identifier

Yeah this is great! I've looked at WebFinger before, but admittedly haven't given it enough thought recently. It's a pretty simple yet powerful solution for addressability. It's not quite as flexible as DID, seeing as it requires a domain and the identifiers follow a more specific format. I think that's better in some ways because it's more human-readable. However, for those of us looking to look up someone's profile from something like a Ethereum address, I imagine you could use Ethereum Name Service, or Namecoin, or Handshake, then do a reverse lookup to get the record for WebFinger.

Then adding DID fields to the Actor object or a public key would allow you to do proper verification. Agreed that some sort of realtime protocol would be necessary to properly verify you are communicating with a specific user. I don't think we need to define that at this point, but we really should ensure that public keys are in the profile data so that apps can use it.

> For the purpose of OMI, we need to be inclusive of proprietary applications, so any code used as part of a standard would ideally be MIT or BSD type licenses.

Agreed on this. We need non-copyleft options for applications to use. However, I also agree with **@wmurphyrd** that we shouldn't endorse a specific implementation. The challenge then is just making sure that creating your own implementation is simple enough.

I had a look at activitypub-express. It looks really good! I like the modular approach and the integration into express looks super simple. Still, the implementation itself is somewhat large. Requiring other developers to implement all of the ActivityPub spec if they just want to create a homeserver for OMI seems unnecessary. Perhaps we can agree on a required subset of the ActivityPub spec to handle identity?

On the W3C SocialCG call this morning, **@wmurphyrd**, **@dmitrizagidulin**, and I talked a bit about possibly creating an extremely limited subset of the ActivityPub spec to manage identity. I asked if an ActivityPub implementation that only included the `Actor` object and an inbox/outbox would be in the spirit of the fediverse community. I'll post shortly in the SocialCG forum about this and get their feedback on it. However, I think it should be accepted as valid. ActivityPub has been accepting of other projects that only support a subset of the spec. This is an extreme example though.

If we were to go down this route, developers looking to implement an OMI homeserver would only need to support activities for modifying an actor and implement the correct data model for getting the required `Actor` fields. I think that would be really simple to implement. It also gives us a base to go ahead and start implementing other things like a friends list or inventory collection. They'd be modular and could be expressed via various JSON-LD vocabularies.

↑ 1 0 replies

robertlong on May 21, 2021 Maintainer Author

> From chatting with **@fire**, the possibility of using Matrix was brought up. There is a possible simplified implementation of Matrix identity in Jel by **@gfodor** , though from what I can see, it uses a axa, a fork of the Synapse matrix.org server. As this is also based on Synpase it would fail the "two-implementations rule". Perhaps Greg has some insight on a second implementation.

I'd also like to invite more discussion around Matrix. It also has addressable identity and verification. It is extensible so you could add the necessary profile information. But like ActivityPub, it comes with a ton of baggage. It's a chat protocol, so if your application/homeserver doesn't need chat, then why bring that with the minimal spec? It kinda goes against the modular approach.

On the other hand, it could provide a standard method for realtime communication. I feel like Matrix makes sense to integrate into the OMI ecosystem in another way. Realtime notifications for trades, friend requests, etc. could be sent via Matrix. It could also be used to set up P2P networking for certain applications. So I don't want to push Matrix out of our larger discussion around OMI specs. I just think if we're talking identity, requiring all of Matrix seems heavy handed.

↑ 1 0 replies

robertlong on May 21, 2021 Maintainer Author

I've posted to the SocialHub here: https://socialhub.activitypub.rocks/t/minimum-viable-activitypub-implementation/1790 feel free to chime in there.

↑ 1 0 replies

wmurphyrd on May 22, 2021

> Agreed on this. We need non-copyleft options for applications to use. However, I also agree with **@wmurphyrd** that we shouldn't endorse a specific implementation. The challenge then is just making sure that creating your own implementation is simple enough.

For the record, we consider use of the Immers server to be compatible with proprietary software. Because it packaged as a microservice, the AGPL requirements do not extend beyond the container boundary. Here's an entry in our FAQ on the subject. I believe this is a well-established interpretation with documented support from FSF (if there's doubt/interest on this subject, I could write an article). For the client side, we'll be putting out an MIT licensed library very soon.

> I had a look at activitypub-express. It looks really good! I like the modular approach and the integration into express looks super simple. Still, the implementation itself is somewhat large. Requiring other developers to implement all of the ActivityPub spec if they just want to create a homeserver for OMI seems unnecessary. Perhaps we can agree on a required subset of the ActivityPub spec to handle identity?

@robertlong I'd like to have a longer discussion on this topic sometime. When I look at ActivityPub (and the implementation in ApEx), I see a lightweight spec that covers the bare bones of establishing and federating identity, so I'd like to learn more about your concerns here.

↑ 1 0 replies

sebilasse on May 24, 2021

> **@robertlong** here's a theoretical example of using ActivityPub and DID together: https://github.com/WebOfTrustInfo/rwot5-boston/blob/master/draft-documents/activitypub-decentralized-distributed/activitypub-decentralized-distributed.md#distributed-identity

yep, and there is this nice DID method:
https://trustbloc.github.io/did-method-orb/

↑ 1 0 replies

robertlong on May 24, 2021 Maintainer Author

> **@robertlong** I'd like to have a longer discussion on this topic sometime. When I look at ActivityPub (and the implementation in ApEx), I see a lightweight spec that covers the bare bones of establishing and federating identity, so I'd like to learn more about your concerns here.

I took some more time over the weekend to dig into WebFinger and ActivityPub further. I think you're right. It can be a very simple spec around WebFinger and the ActivityStreams Actor model. I'd like to talk more about the server to server protocol as well. Are there any parts that we would need to implement for the most barebones implementation?

↑ 1 0 replies

robertlong on Sep 7, 2021 Maintainer Author

Moving to backlog due to inactivity. We'll pick this back up at a later date. Discussion ongoing in the #omi-identity channel.

↑ 1 0 replies

mrmetaverse last week Maintainer edited

Hey! We are reviewing old tickets today during backlog refinement. Any interest in resuming this discussion or should I convert this to a discussion?

We are now realizing that things like this could open as discussions in the discussions section too since we are still using discussions.

↑ 1 0 replies

robertlong last week Maintainer Author edited

Sounds good to me. Curious to see what people have in mind on identity nowadays. We're using Matrix ids for identity at Third Room, you can bridge all sorts of other platforms with Matrix and connect third party identities. It's a good solution for us. DID looks interesting as well but I've not seen it get any traction. Mastodon just passed 10m users and is based on ActivityPub. They're doing great! So perhaps that should get more people's attention in this space.

↑ 1 0 replies

Category

R&D

Labels

Make the metaverse... Empower the peopl... Call for Participation

5 participants

Converted from issue

This discussion was converted from issue #36 on April 12, 2023 22:22.