Satellite

Contest:
A contest to demonstrate how to link your accounts and content.

Title:
An IndieWeb-inspired method to decentralize identity & authentication via open web & formats

Name:
Barack Sokullu

Pseudonym:
Turkix

Would you like your real name published if you win?
Yes

Associated Url(s)
https://indieweb.org/IndieAuth
https://dev.uniresolver.io/
https://identity.foundation/ion/

Explanation & Instructions
Platforms should agree on a standard to define the relationship between the person (“P”) and their public online identities (“OIs”). We will refer to this set of standards and protocols as “Satellite” while some of them are full replicas or highly inspired by other open standards.

Implementing Satellite would be as simple as adding the following hidden identity cards in HTML, ideally within the head section, rather than the body as IndieWeb proposes:

So instead of a block element like:

```
<a rel="me" href="https://twitter.com/EmreSokullu" />
```

add the following meta data:

```
<meta property="did:version" content="latest" />
<meta property="did:me" content="ipfs:EiClkZMDxPKqC9c-umQfTkR8vvZ9JPhl_xLDI9Nfk38w5w" />
<meta property="did:verification" content="10" />
```

Enclosed in the *<head>* of the P's personal homepage, as well as their Twitter, LinkedIn, Facebook profiles, which would depend on the platforms' level of support.

The canonical address value of the did.me attribute would resolve into a DID response as defined by identity.foundation. While those specs are open for further changes and improvements, they are already functional. A sample JSON document would look like:


```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    {
      "@base": "did:web:did.actor:bob"
    }
  ],
  "id": "did:ipfs:EiClkZMDxPKqC9c-umQfTkR8vvZ9JPhl_xLDI9Nfk38w5w",
  "service": [
    {
      "id": "#linkedin",
      "type": "linkedin",
      "serviceEndpoint": "linkedin.com/in/EmreSokullu"
    },
    {
      "id": "#github",
      "type": "github",
      "serviceEndpoint": "github.com/esokullu"
    }
  ],
  "publicKey": [
    {
      "id": "#z6MkkQBvgvqb6zGvS4cydworpUaRDzpszSFixq49ahbDeUTG",
      "type": "Ed25519VerificationKey2018",
      "controller": "",
      "publicKeyBase58": "6wvt6gb9mSnTKZnGxNr1yP2RQRZ2aZ1NGp9DkRdCjFft"
    }
  ]
}
```


either stored in a decentralized storage system like IPFS or anchored on a public blockchain such as Bitcoin or Ethereum mainnet. To keep the latency low and transaction fees reasonable while working with the public blockchains, one should leverage sidechains similarly to Lightning Networks and ION.

With the public key provided in the DID document, we can verify the authenticity of any content created and signed by this particular P and their private keys.

On the authentication-side, to authenticate a P, the website would

(1) Ask for P's online identity URL ("PURL")

If the PURL is Satellite-compatible, then:

(2) Generate a session ID ("SID")
(3) Generate a random sentence ("RS") for the P to sign
(4) Present these two as well as a submit link to the P with a QR code

The P would launch their decentralized identity app (on a mobile phone with camera access) to scan the QR code, which then would:

(1) Sign the RS presented with its private key (which would produce the signed random sentence, "SRS")
(2) Send the SRS and the SID to the submit link in POST request over HTTPS

The website would verify the SRS with the public key provided by the DID document and authenticate the user accordingly.

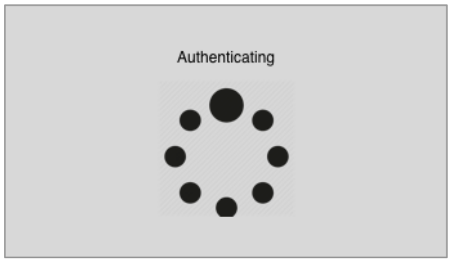
To summarize, the process would be as follows:

Step 1: The website asking for user information

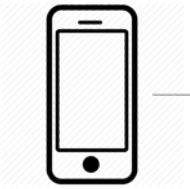
Step 2: The website generated with the information provided (https://twitter/sokullu)


```
<satellite version="0.1">
    <session>7F7552D8-8B88-4BB1-AD1C-61EE649965CB</session>
    <identity>https://twitter.com/sokullu</identity>
    <randomText>Pellentesque habitant morbi tristique senectus</randomText>
    <submit>https://mywebsite.com/submit.php</submit>
</satellite>
```

Step 3: The user would use their identity app to scan the QR code

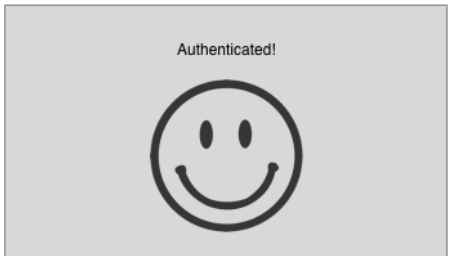


Step 4: The app would sign the random text with the user's private key, and send it to the submit URL with relevant information



```
<satellite version="0.1">
    <session>7F7552D8-8B88-4BB1-AD1C-61EE649965CB</session>
    <identity>https://twitter.com/sokullu</identity>
    <signedRandomText>2ahUKEwjRgsbc9uzyAhWMjKQKHeydCTYQ2</signedRandomText>
</satellite>
```

Step 5: happy ending! :)



One may use the *publicKey* provided in the DID document to verify any text that is claimed to be P's. To prove that the content does actually belong to the P, the platform would have to enclose it with its signature;

```
<span data-did-signature="34a1988c0a5878ddcea0cdb42d651e1e"> Lorem ipsum dolor sit amet, consectetur adipiscing elit. </span>
```

This would mean the content does actually belong to the P shown in the head meta.

If the content appears somewhere else (not in the P's online profile) one would have to include their online identity URL as an attribute in the span tag:

```
<span data-did-url="https://github.com/esokullu"
data-did-signature="34a1988c0a5878ddcea0cdb42d651e1e"> Lorem ipsum dolor sit amet, consectetur adipiscing elit. </span>
```

Please note the *did:verification* property in the meta tags represents the verification status of the P. It would be a score from zero to ten, representing verification confidence. An unverified user would be zero by default, and any score below five would represent unverified. While ten means full verification, any number between five and ten would represent partial verification. A federated service may provide identity verification for a certain fee to be paid in cryptocurrency, and propagate it to the other parts of the network.

Final Notes:

If I happen to win, please donate the amount to Electronic Frontier Foundation

Appendix:

The code used in this submission:

--

<meta property="did:version" content="latest" />
<meta property="did:me" content="ipfs:EiClkZMDxPKqC9c-umQfTkR8vvZ9JPhl_xLDI9Nfk38w5w" />
<meta property="did:verification" content="10" />

--

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
```

```
    {
      "@base": "did:web:did.actor:bob"
    }
  ],
  "id": "did:ipfs:EiClkZMDxPKqC9c-umQfTkR8vvZ9JPhl_xLDI9Nfk38w5w",
  "service": [
    {
      "id": "#linkedin",
      "type": "linkedin",
      "serviceEndpoint": "linkedin.com/in/EmreSokullu"
    },
    {
      "id": "#github",
      "type": "github",
      "serviceEndpoint": "github.com/esokullu"
    }
  ],
  "publicKey": [
    {
      "id": "#z6MkkQBvgvqb6zGvS4cydworpUaRDzpszSFixq49ahbDeUTG",
      "type": "Ed25519VerificationKey2018",
      "controller": "",
      "publicKeyBase58": "6wvt6gb9mSnTKZnGxNr1yP2RQRZ2aZ1NGp9DkRdCjFft"
    }
  ]
}
```


--


```
<span data-did-signature="34a1988c0a5878ddcea0cdb42d651e1e"> Lorem ipsum dolor sit amet, consectetur adipiscing elit. </span>


<span data-did-url="https://github.com/esokullu" data-did-signature="34a1988c0a5878ddcea0cdb42d651e1e"> Lorem ipsum dolor sit amet, consectetur adipiscing elit. </span>
```