Extending Sign-In With Ethereum:
Session Keys, Capabilities, and Beyond

Wayne Chang, Spruce
Twitter: @wycdd
CASA 2022

Spruce

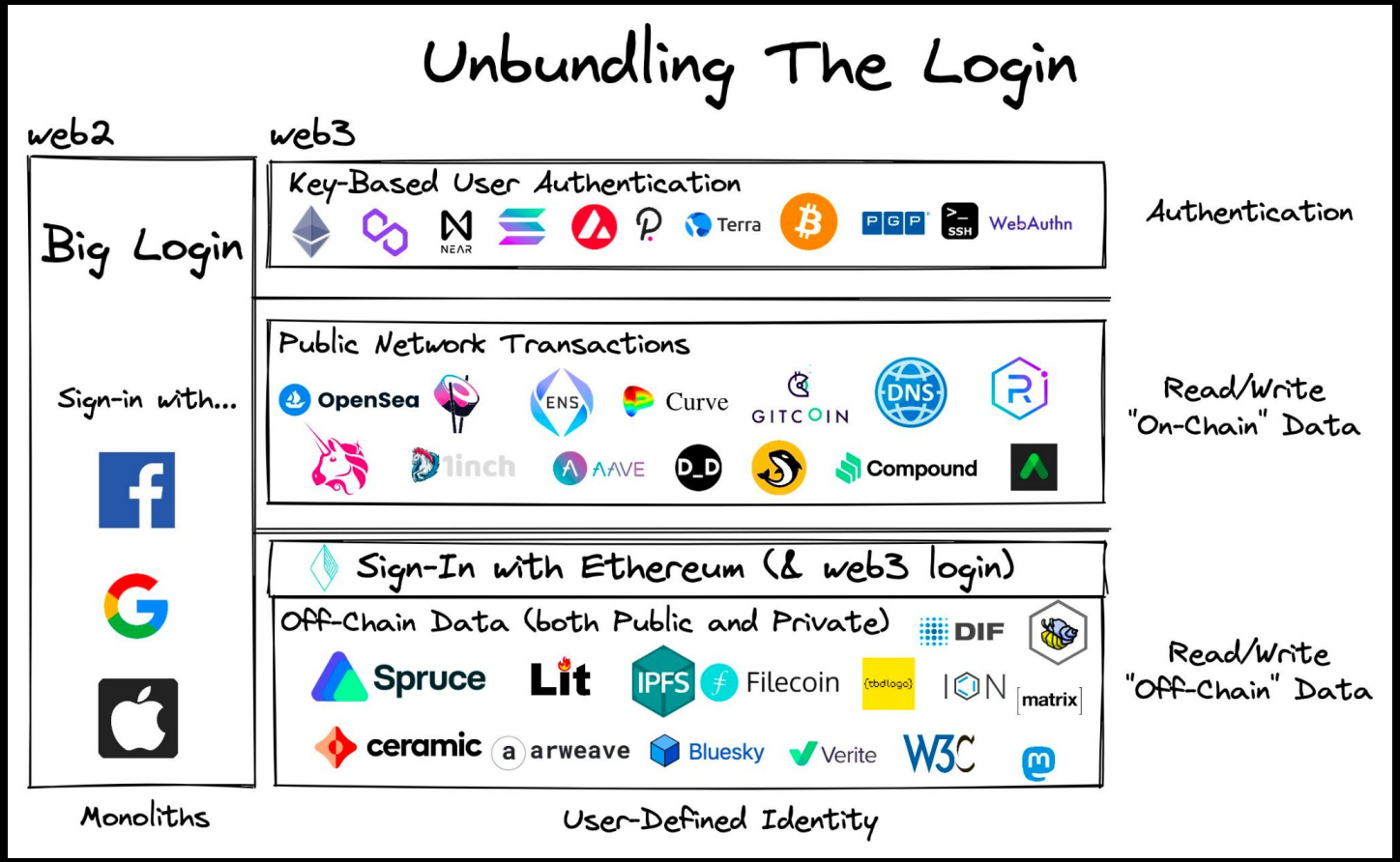
Web3 and Decentralized Identity

- Web3 has created the most successful form of decentralized identity **ever**.
- Today's wallets use keys to sign blockchain transactions, and not much else.
- With these keys, we can also move far beyond Web2 identity and simple SSO.
- **Sign-In with Ethereum** is leading the charge.

SIGN IN WITH
ETHEREUM

Spruce

Unbundling the Login



Sign-In with Ethereum Crash Course

EIP-4361 standardizes a message format for signing.

```
${domain} wants you to sign in with your Ethereum account:
${address}

${statement}

URI: ${uri}
Version: ${version}
Chain ID: ${chain-id}
Nonce: ${nonce}
Issued At: ${issued-at}
Expiration Time: ${expiration-time}
Not Before: ${not-before}
Request ID: ${request-id}
Resources:
- ${resources[0]}
- ${resources[1]}
...
- ${resources[n]}
```

Spruce

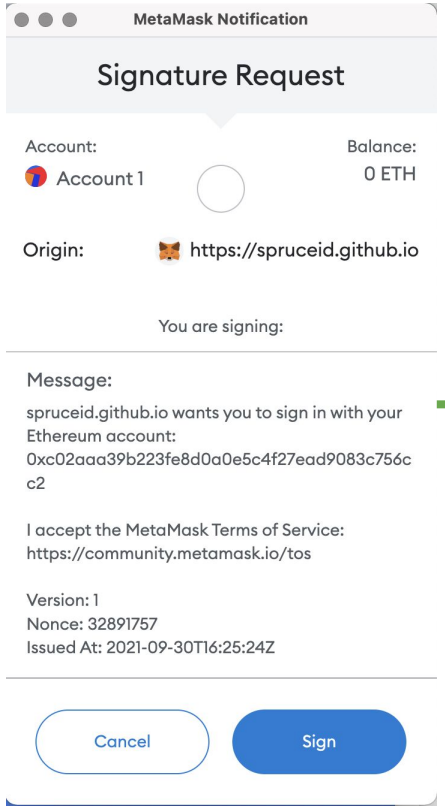
Sign-In with Ethereum Crash Course

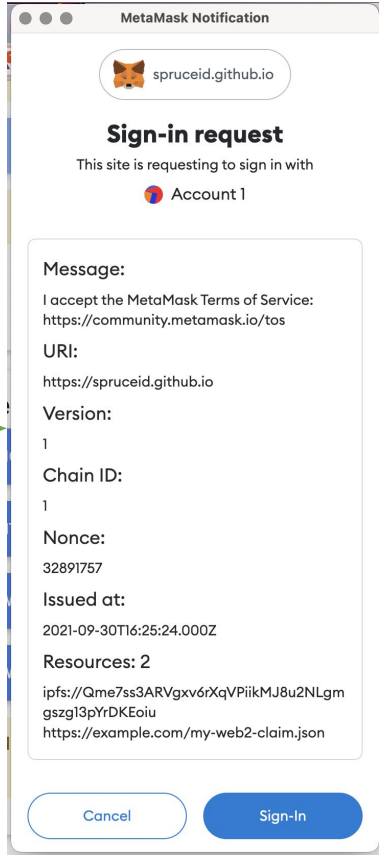
Requires EIP-191 personal sign + ABNF, with domain binding to add security.

```
sign-in-with-ethereum =
    domain %s" wants you to sign in with your Ethereum account:" LF
    address LF
    LF
    [ statement LF ]
    LF
    %s"URI: " uri LF
    %s"Version: " version LF
    %s"Chain ID: " chain-id LF
    %s"Nonce: " nonce LF
    %s"Issued At: " issued-at
    [ LF %s"Expiration Time: " expiration-time ]
    [ LF %s"Not Before: " not-before ]
    [ LF %s"Request ID: " request-id ]
    [ LF %s"Resources:"
    resources ]
```

Spruce

Now all Ethereum wallets can do passwordless login.





Spruce

Why do we sign things?

To do an action, such as authenticate, authorize, or execute with guarantees of:

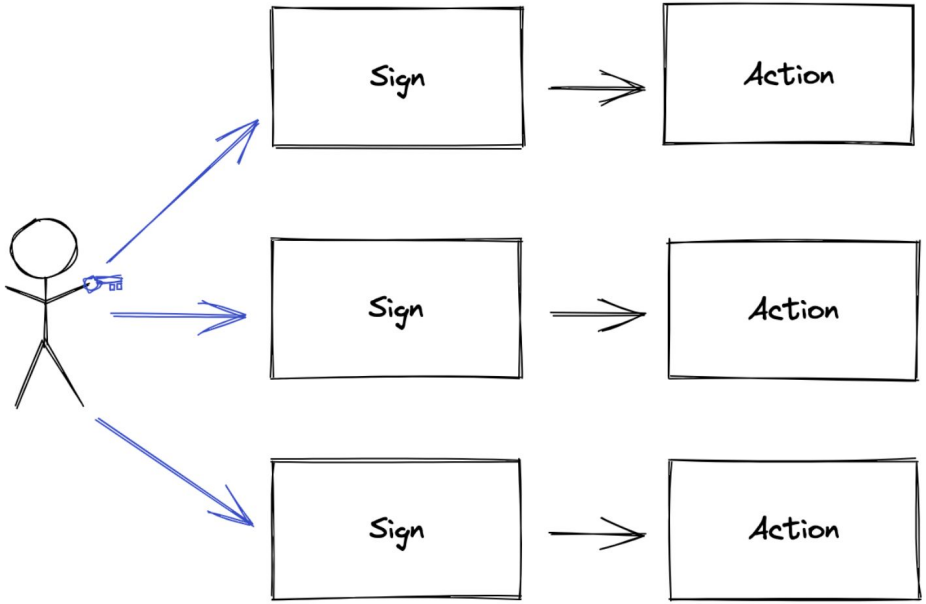
- Integrity: “only this exact message”
- Authenticity: “it was me who signed it”

More signing = more opportunity for user control.

Let’s be user control maxis.

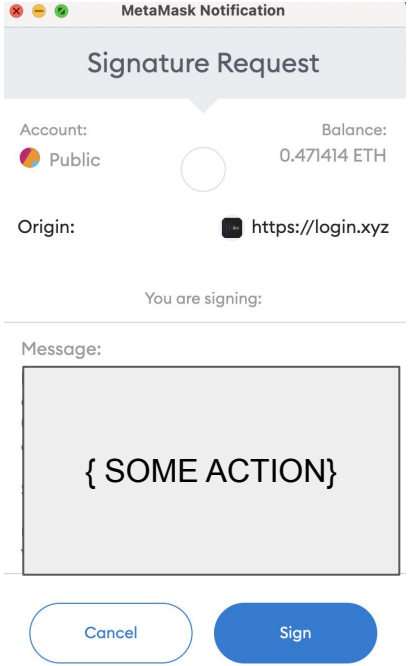
Spruce

The Problem with Signing Other Stuff Today



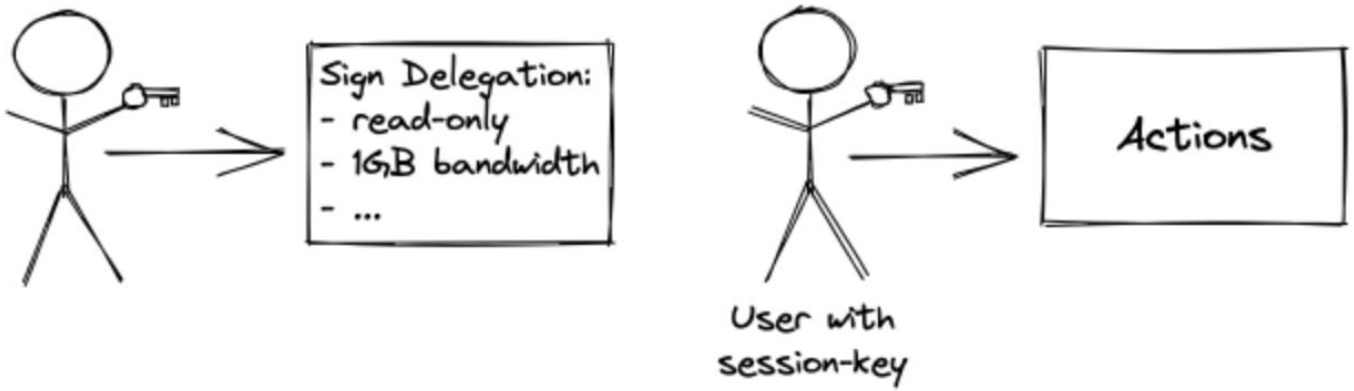
Each signing opens up a new prompt, interrupting the user and breaking flow

= 3 x



Spruce

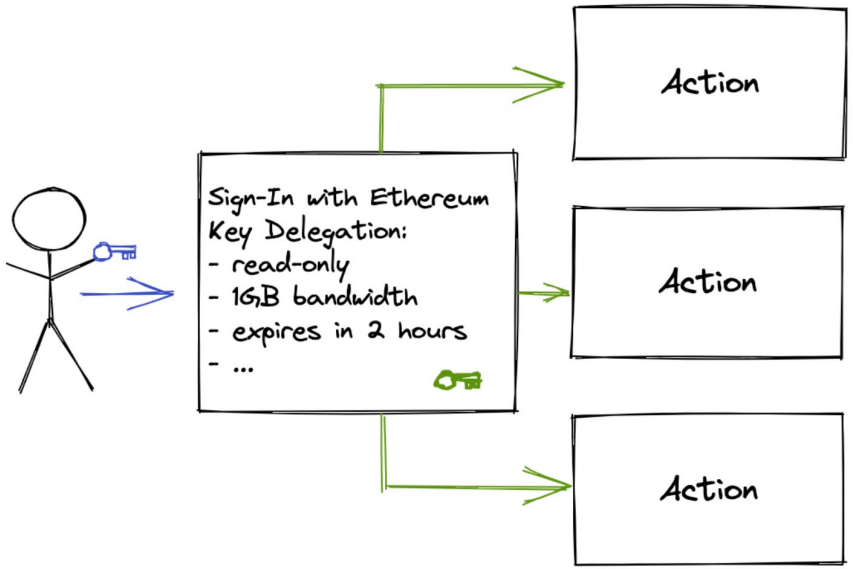
The Main Idea Behind Session Keys: Delegation



With delegation to a session key, there is only one signing to start the session with restrictions

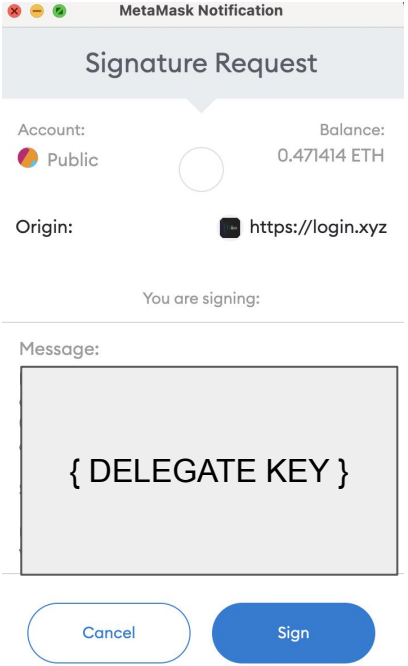
Spruce

How Session Keys Improve UX



With session keys, all actions are rooted in the user's keys

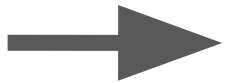
= 1 x



Spruce

Example Session Key Delegation using SIWE CACAOs

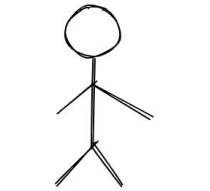
```
"p": {

    // address from EIP-4361 as did:pkh
    "iss": "did:pkh:eip155:...",
    // domain from EIP-4361
    "domain": "example.com",

    ...

    // uri from EIP-4361, in that case session DID
    "aud": "did:key:z6...",

    "resources": [
```



Then you can issue capabilities and more!

Spruce

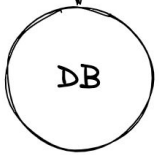
The Transition: web2 to web3



Login

Frontend

Backend Server



Delegate Key

Dapp

Spruce

Sometimes we need to prove validity

- How do we make sure that the key was used within the **validity period**?
- We can know it was used **before a certain time (expiration-time)** if we have a **merkle inclusion proof or similar** of the signature. What other ways are there?
- We can know it was used **after a certain time (not-before/issued-at)** by **including a recent blockhash in the payload**. What other ways are there?
- Revocation mechanisms can be further defined within claim structures, such as W3C VCs.

Spruce

Example Use: W3C Verifiable Credentials + Presentations

```
// ...
"proof": {
  "type": "CacaoProof2022",
  "created":"2019-12-11T03:50:55Z",
  "proofPurpose":"capabilityDelegation",
  "verificationMethod": "did:pkh:eip155:...#<vm>",
  "proofValue": "multibase(cacao)",
  "currentBlockhash": "blockhash(blockNumber)",
  "inclusionProof": "merkleInclusionProof()"
}
```

Spruce

Other Considerations for Key Delegation

- Security model evaluation across different environments and best practices per environment: DOM (localStorage vs sessionStorage?), Browser Extension, Browser Extension Extension, Native Apps, 3rd party server.
- Encryption Keys, e.g., secp256k1 delegating to ed25519.
- Other key types like BLS to support features like ZKPs.
- If a blockchain VM supports key delegation, then you can have an onramp to those chains.

Spruce

Relationship with HD Key Derivation

- Sure, it works, but for ephemeral keys it may be safer to rely on system entropy to ensure no key reuse.
- If you use HD Key Derivation and want ephemeral keys, you will need to remember which keys have already been used as state somewhere.

Spruce

Opportunities to collaborate

- What is the general shape of a “key delegation”? Is this something we can write abstract interfaces for, or is it too application-dependent?
- Explore useful features: preempted revocation? capability interoperability?
- Standardize proofs for use within validity period:
  - Proof-of-use-after-time (“**not-before**”/“**issued-at**”)
  - Proof-of-use-before-time (“**expiration-time**”)
- Standardize encrypt-to-pkh.

Spruce