

User OnBoarding Experience

arcturus platform

source

1. [Santiago Palladino](#), ["Ethereum for developer"](#), Bs As, 2019
2. [Alex Van de Sander](#), ["Universal Ethereum Login"](#), DevCon4, Dic. 2018
3. [Philippe Castonguay](#), ["Discussion about smart account concept"](#), Tweet, Nov 2018

Purpose

Sign-up and sign-in process.

Think about bad onboardings.

Think about what kind of people we want to reach: normal people? technology enthusiast?

Next discussion is about **creating an account**

1. Create an account

Account -> Public key & Private key

From the user point of view, ¿Should an account be an abstract concept?

¿Should a normal user care about safekeeping account?

Let's talk about the following concepts (but not for so long):

1. STORAGE
2. BACK-UP

1. Create an account

OLD SCHOOL

1. personal wallet
2. local account

A NEW APPROACH

1. smart account



Philippe Castonguay
@PhABCD

How would you call contracts that support meta-transaction, universal logins, account recovery, 2FA for transfers, add any plugins, etc? I.e contracts like [@gnosisSafe](#), which can be used for individuals and organizations.

Identity Contract

24.2%

Smart Wallet

37.1%

Smart Account

33.9%

Other

4.8%

62 votes · Final results

10:12 AM · Nov 21, 2018 · [Twitter Web Client](#)

4 Retweets 12 Likes

1. create an account >> personal wallet

storage: it is delegated

back-up: traditional way -> mnemonic (12 o 24 words)

workflow:

- a. choose a wallet
- b. create an account
- c. writing down the mnemonics
- d. setting up a passphrase

references: MOC with Nifty, Compound with Metamask

problems: UX. No user friendly.

1. create an account >> local account

Create an account for the user within the app instead of asking the user to install an extension. The creation is client-side.

Storage: private key encrypted with user password. it could be stored on

1. browser's local storage
2. remote server
3. downloaded to the user's computer
4. synced across cloud storage

Back-up: mnemonics

1. create an account >> local account cont.

Limitations:

1. one user - multiple devices
2. two factor authentication for critical tx can not be implemented without trusting funds to a custodian

Two problems:

1. steps for safekeeping (mnemonics)
2. lost the password
 - a. the user could clear local data
 - b. the user could lose the device

1. create an account >> smart account

Move the account to the blockchain.

User identity (along with its funds) is represented by a **single contract** instead of one or several externally owned accounts.

storage: one account for each device

back-up: pick whatever solution you want; such as social recovery. Moving the user's identity to a smart contract allows us to implement any kind of account management features.

social recovery: choose n trusted friends to do a recovery

1. create an account >> smart account

New features: additional security & recovery

1. Register multiple devices
2. Different keys can have different access level
 - a. level 1: identify the user (managing the identity contract)
 - b. level 2: transfer funds
3. two factor authentication for critical operations
4. social recovery

Interesting Links

1. [Universal Ethereum login](#)

by Alex Van de Sande;

One of the devs who

came up with the idea

of “smart account”

Alex Van de Sande

Universal Ethereum Logins

“average” users...

- ▶ **don't care** about ether
- ▶ **don't care** about your token
- ▶ **don't care** about backing up private keys or seed phrases
- ▶ **don't care** about your browser plugin
- ▶ **don't care** about your gas price or transaction costs
- ▶ **don't care** about your KYC
- ▶ just want a normal username and purchase stuff with their credit cards **like normal people**
- ▶ **own multiple devices** and expect them to be in sync

Universal Ethereum Logins

(Now is Unilogin)

Alex Van de Sande

Universal Ethereum Logins

Bad solutions

- ▶ ~~New ether account for every app~~ (bad user experience)
- ▶ ~~Type your private key to login~~ (being deprecated!)
- ▶ Use our sleek proprietary ether login service (NOOOO)

Universal Ethereum Logins

(Now is Unilogin)

Alex Van de Sande

Universal Ethereum Logins

Did you miss something?

- ▶ **NO** mention of ether
- ▶ **NO** hex strings
- ▶ **NO** passwords
- ▶ **NO** QR Codes
- ▶ **Incremental** security as user needs it
- ▶ User just deployed a **multi-factor self-sovereign smart-contract-controlled identity** and didn't even know it!

Universal Ethereum Logins

“User doesn’t need to see it or back-up.

Keep it on the device or app as safely as possible.

Don’t keep any funds on it” Alex

Alex Van de Sande

Universal Ethereum Logins



Every app has its own private key

User doesn't need to see it or back it up, keep it on the device or app as safely as possible. **Don't keep any funds on it.**

Universal Ethereum Login

META-TRANSACTION

use the private key to sign a message;

A meta-transaction is a sign message telling a

smart contracts to do something

Alex Van de Sande
Universal Ethereum Logins



Messages are executed by **relayers**

Each app can have one or more relayers, that will pay the ether to execute the transaction. The relayer can either be **paid on-chain by the contract in tokens**, or off-chain via other incentives

Extra recovery keys

More creative recovery solutions, for example:

1. social recovery
2. after some inactive period of time

"You can be created on whatever you want to do because it is smart contract that is doing the recovery" Alex

Alex Van de Sande
Universal Ethereum Logins



Extra recovery keys

Backups are done via keys generated for the purpose, that might be kept cold and can only be used under specific circumstances. Contract can allow more creative recovery solutions, like deadman's switches or social recoveries.

Deterministic contract addresses

“Because it works on multiple chains, you can deploy it on a testnet, or a side chain, and only move it to the main chain after the user has passed some sort of threshold” Alex

Alex Van de Sande

Universal Ethereum Logins



✗ Mainnet (not deployed yet)

✓ Rinkeby (deployed)

✓ App side chain (deployed)

Deterministic contract addresses

Aka: counterfactual contracts: the address is known before it's deployed, so it can be claimed in multiple chains/shards and only be deployed after funds have been deposited

*thanks Adrian Guerrero for the name

Authorization terms per key

Different device, different security:

For example, some keys can add transaction,
while some keys can not add transaction.

Alex Van de Sande

Universal Ethereum Logins



✓ Can send N ether per day

✗ Cannot add other keys (1 of 4)

✓ Can make function calls

Authorization terms per key

each key can either be part of a simple multisig,
or call a more complex contract with will control
which types of calls it can be used for