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

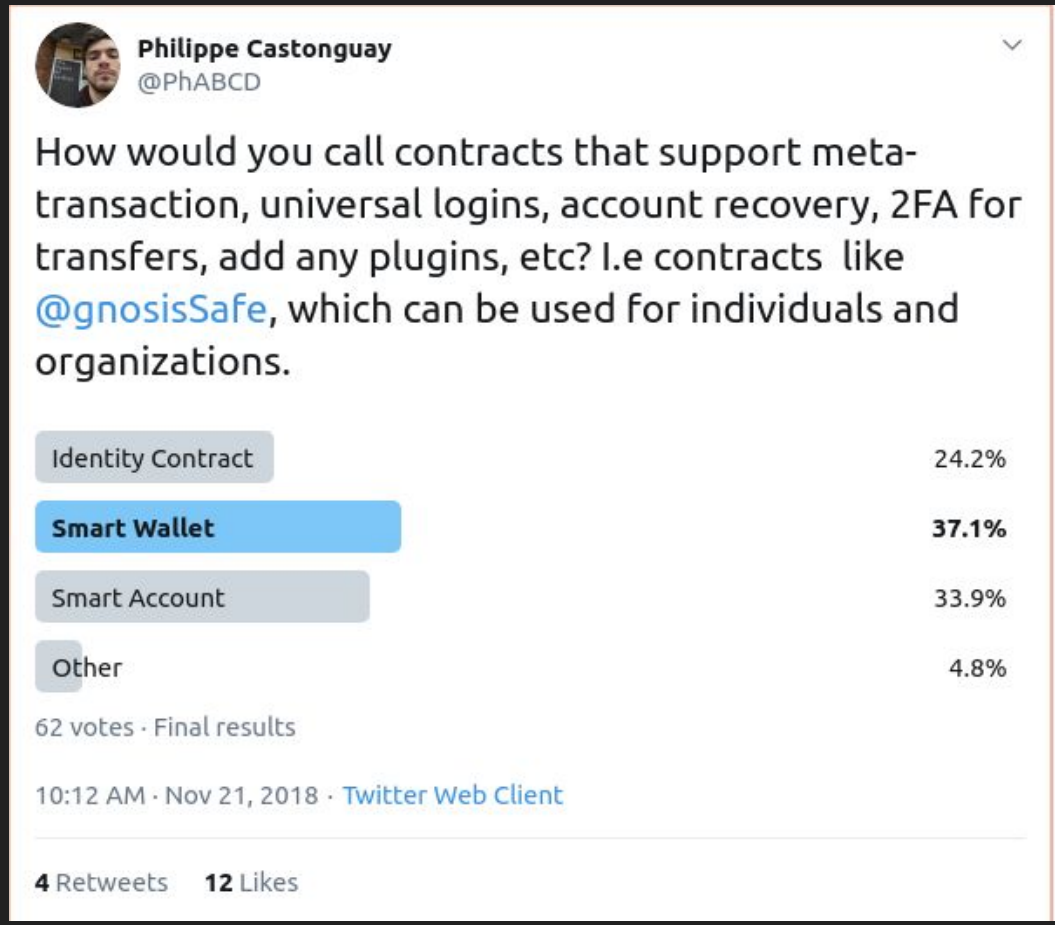
1. Create an account

OLD SCHOOL

1. personal wallet
2. local account

A NEW APPROACH

1. smart account



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;

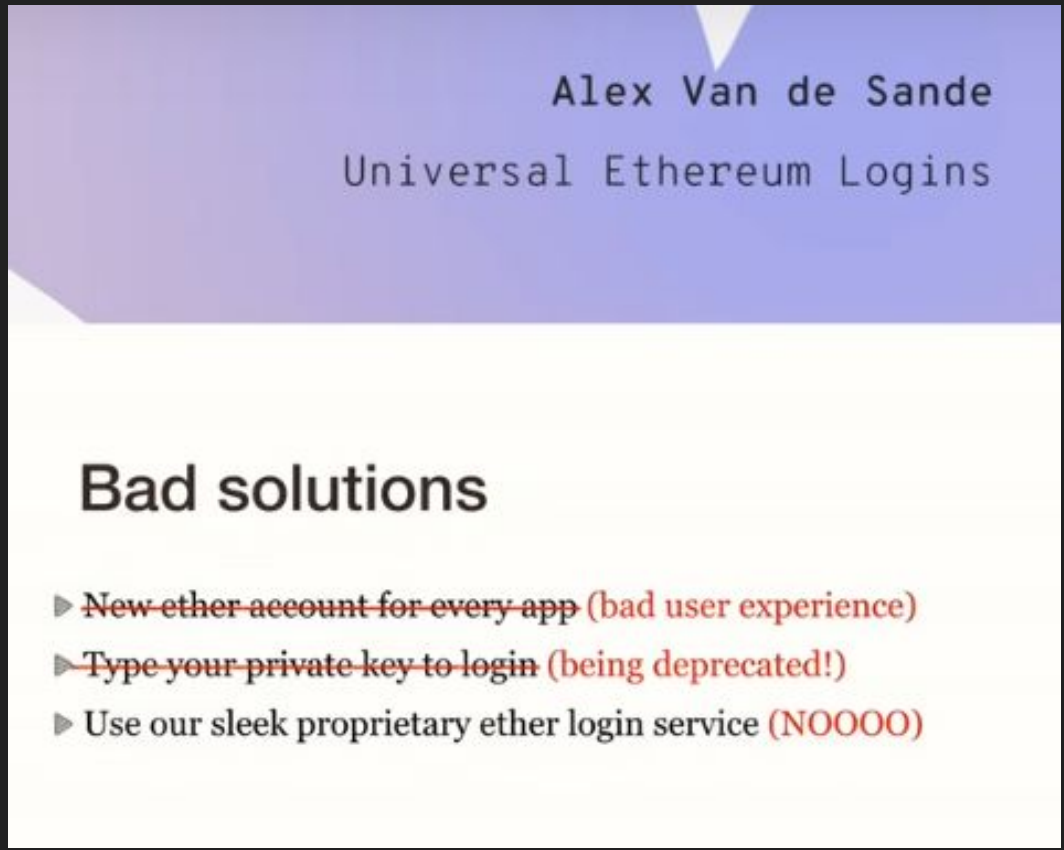
   One of the devs who

   came up with the idea

   of “smart account”

Universal Ethereum Logins

(Now is Unilogin)



Universal Ethereum Logins

(Now is Unilogin)

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

Universal Ethereum Login

META-TRANSACTION

use the private key to sign a message;

A meta-transaction is a sign message telling a

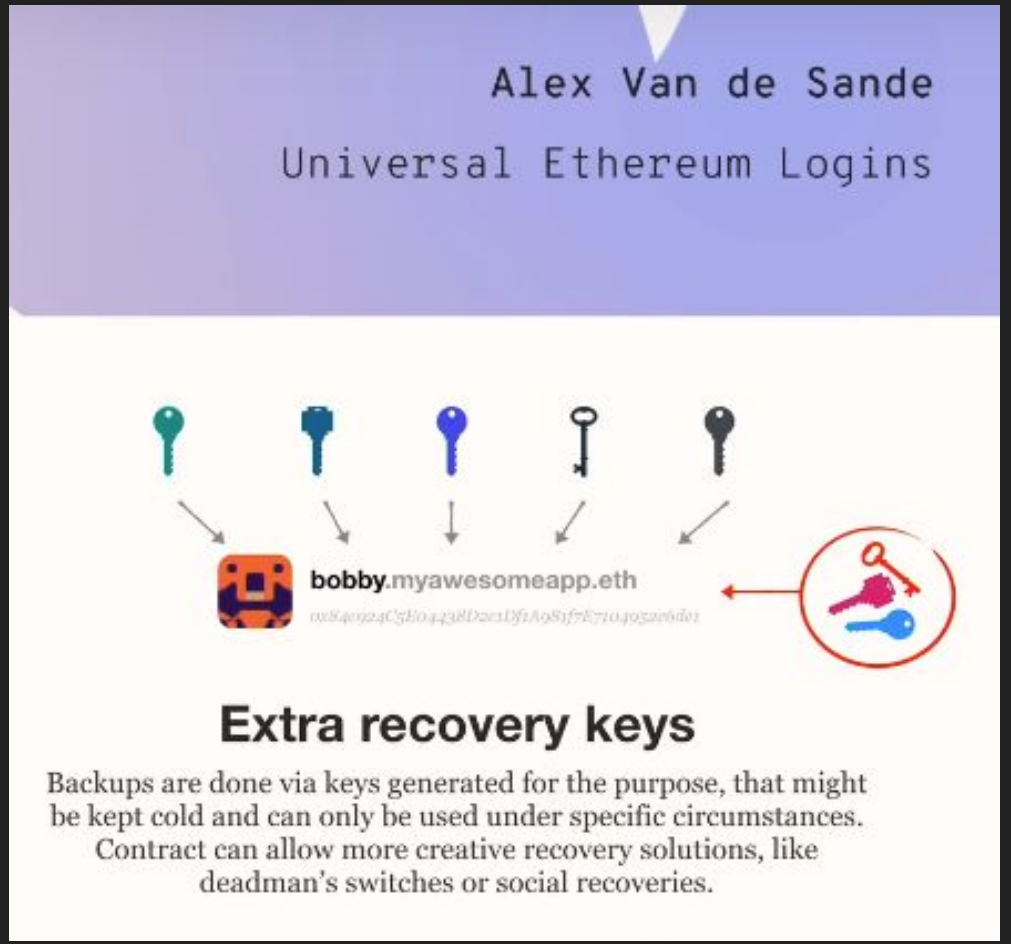
smart contracts to do something

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



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

Authotization tems per key

Different device, different security:

For example, some keys can add transaction,

while some keys can not add transaction.