# Hardware Wallets and Smart Contracts

EDCON
February 2017

Nicolas Bacca
@btchip

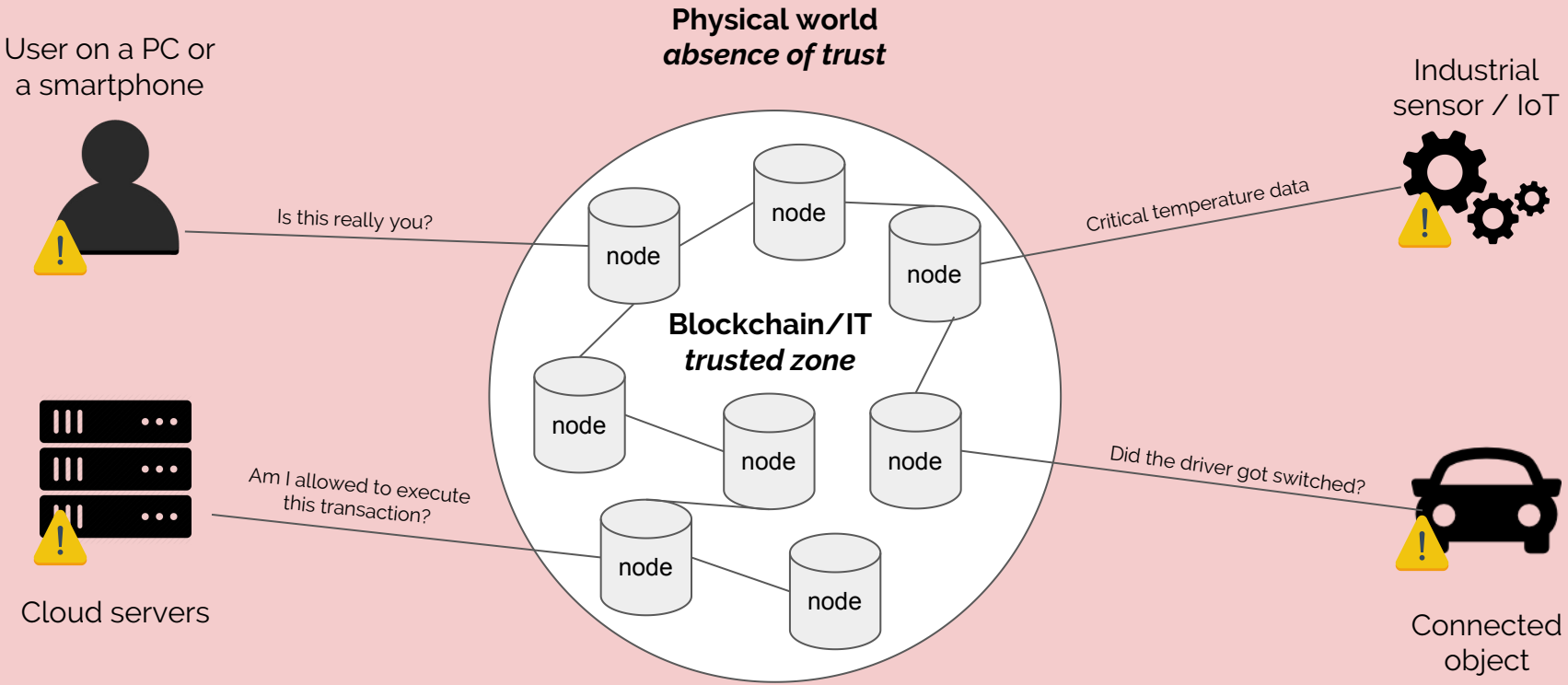
## LEDGER TECHNOLOGY

A trust layer between the **blockchain** and the **physical world**

For industrials, enterprises and consumers

Securing the first and last mile

# Without **trust**, data has no actionable **value**



# Hardware Wallets - high level overview

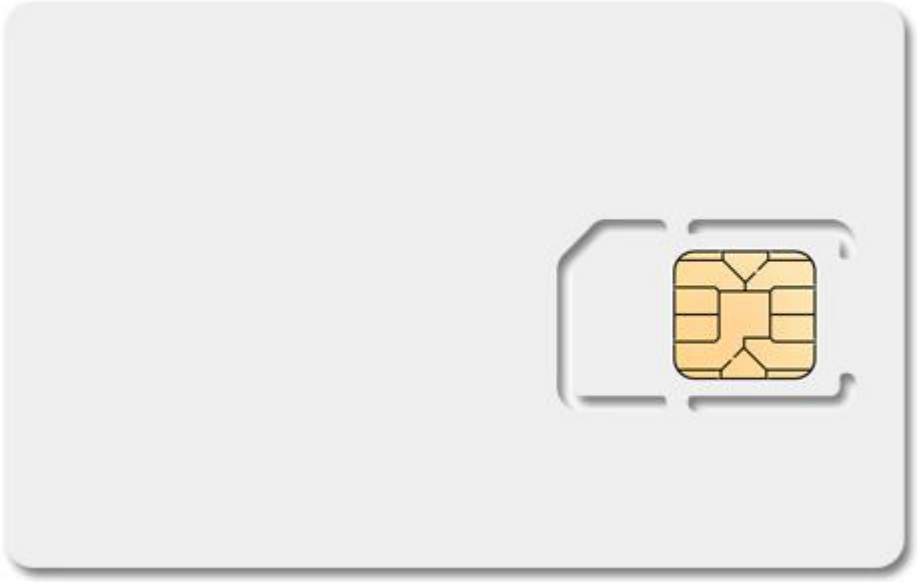
Public data

YES NO

Do you want to roll the dice for 3.1337 ETH ?

Operations on private data, with **user validation** and **proof of user presence**

# The Smartcard (198x)



Not convenient (reader mandatory and multiple middleware / drivers)

Not designed to operate in a hostile environment (turns into an oracle)

Not developer friendly (Java Card (2000) or bust)

# The USB Smartcard (2005)


More convenient (self reader, drivers preinstalled or no driver necessary)

Not designed to operate in a hostile environment (turns into an oracle with user blind approval)

Not developer friendly (Java Card or bust)

# The Hardware Wallet (2012)




**User Presence and validation**


Plug and play (self reader, no drivers necessary)

Designed to operate in a hostile environment

Developer friendly (Native code, Open Source)

# Why ? Cryptocurrencies come with built-in bug bounties

Ledger

#SFYL

Lost keys

Malwares, viruses - direct financial gain

Ransomware 2.0 - indirect financial gain

Because Why Not

# Ransomware 2.0 - Be Your Own Thief

Ledger

Derive private key on path 44’/60’/0’/0’/0 (BIP 32)

44’/60’/0’/0’/0/entropy

Do you want to send something to yourself ? It really belongs to you, no problem here, I checked it all

Ok, seems legit

Hey I got some entropy to sell you

# Safe Smart Contract interaction

Ledger

Bare minimum : verify that you’re performing the right action with the right contract

Extension : verify the action parameters

Not necessary / meaningful for all actions

Different UX might be necessary for individual contracts and actions

# Trustless / Networkless is complicated

Ledger

Associate an address to an ABI ?

Knowing the ABI is not enough - an example with ERC-20

0x89205A3A3b2A69De6Dbf7f01ED13B2108B2c43e7 

A9059CBB Keccak(“transfer(address,uint)”)

000000000000000000000000**70faa28A6B8d6829a4b1E649d26eC9a2a39ba413**
Destination address

0000000000000000000000000000000000000000000000000000000000000001
Amount with decimals **specified at contract instance creation**

# Mini Trusted ABI proposal

Ledger

Decorate a BSON version of the original ABI to mention :
- Methods for which the data can be ignored
- (Addresses that shall or may be internal)
- (Data components to convert to addresses that shall or may be internal)

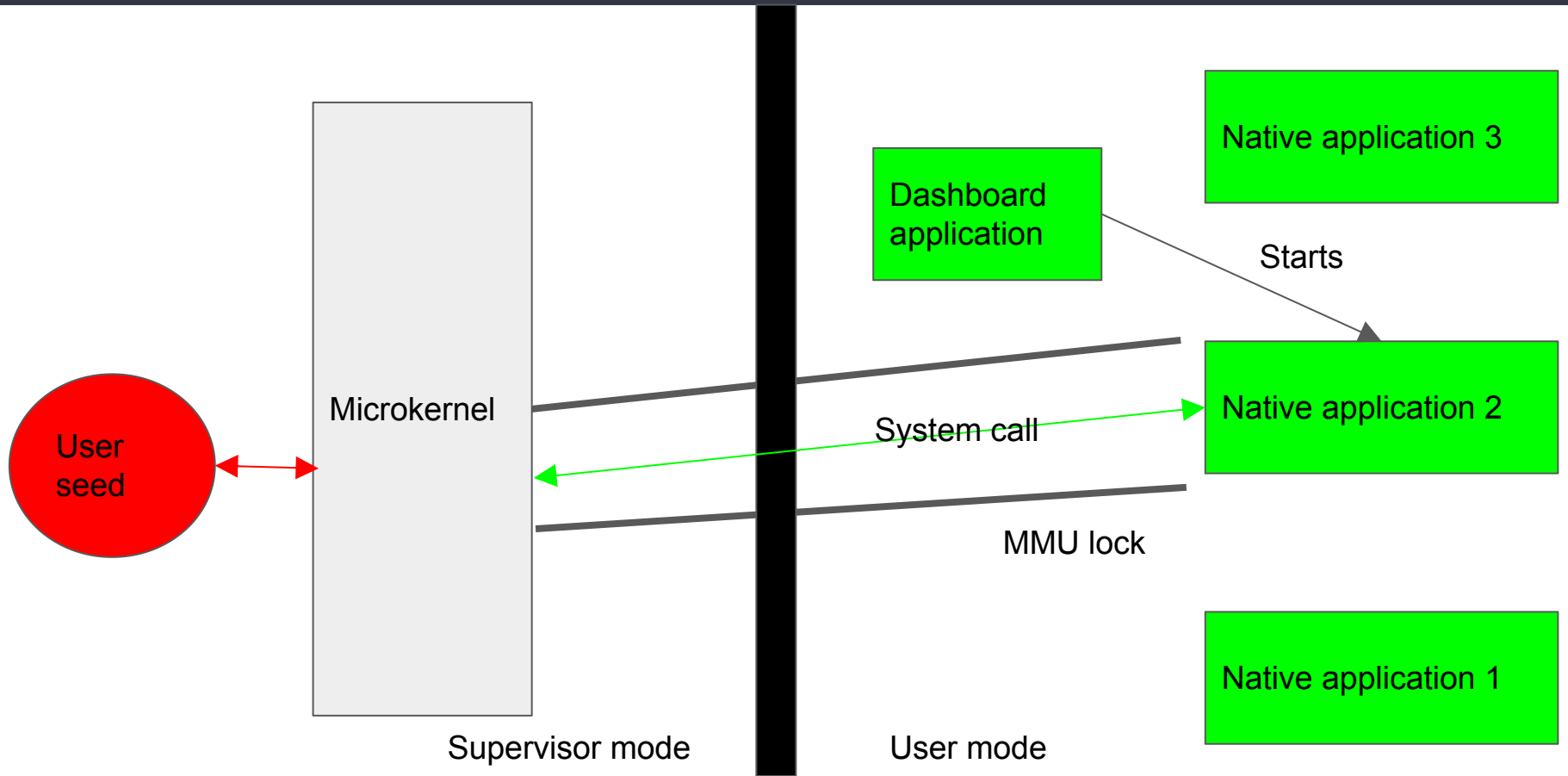
Add the nonce used when creating the contract

Sign with the creator address

Device can check contract address = Keccak(recovered creator address, nonce)

(Device requires the path along with addresses that shall or may be validated)

# Need more ? You can build your own app



# Want to dig into it ?

Ledger

Commit showing ERC-20 integration into our ETH application
https://github.com/LedgerHQ/blue-app-eth/commit/0c094f4fe6e2c2fedd6d05094072c8cdaab9f21c

Nano-S resources : compiler and SDK - https://github.com/ledgerhq/ledger-nano-s

Developer Slack : http://slack.ledger.co

Join us and discuss :)


Thank you
@btchip