Coconut: Threshold Issuance Selective Disclosure Credentials with Applications to Distributed Ledgers

Authors

Alberto Sonnino*

Mustafa Al-Bassam*

Shehar Bano*

George Danezis*

* University College London

LNETM 2018

The authors

Alberto Sonnino

Mustafa Al-Bassam

Bano Shehar

George Danezis

People love blockchains

- ✔ Fancy
- ✔ Involve money
- ✔ Look complicated
- ✔ About crypto magic

Big challenges in blockchains

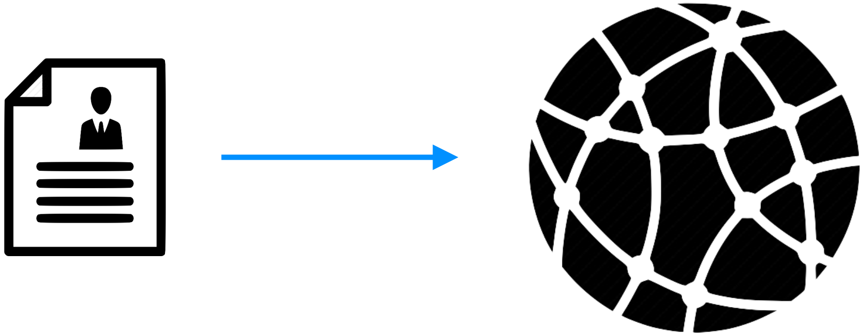
- ✗ **Poor privacy**
- ✗ Scalability
- ✗ Governance
- ✗ Regulations

Big challenges in blockchains

- Poor privacy
- Scalability
- Governance
- Regulations

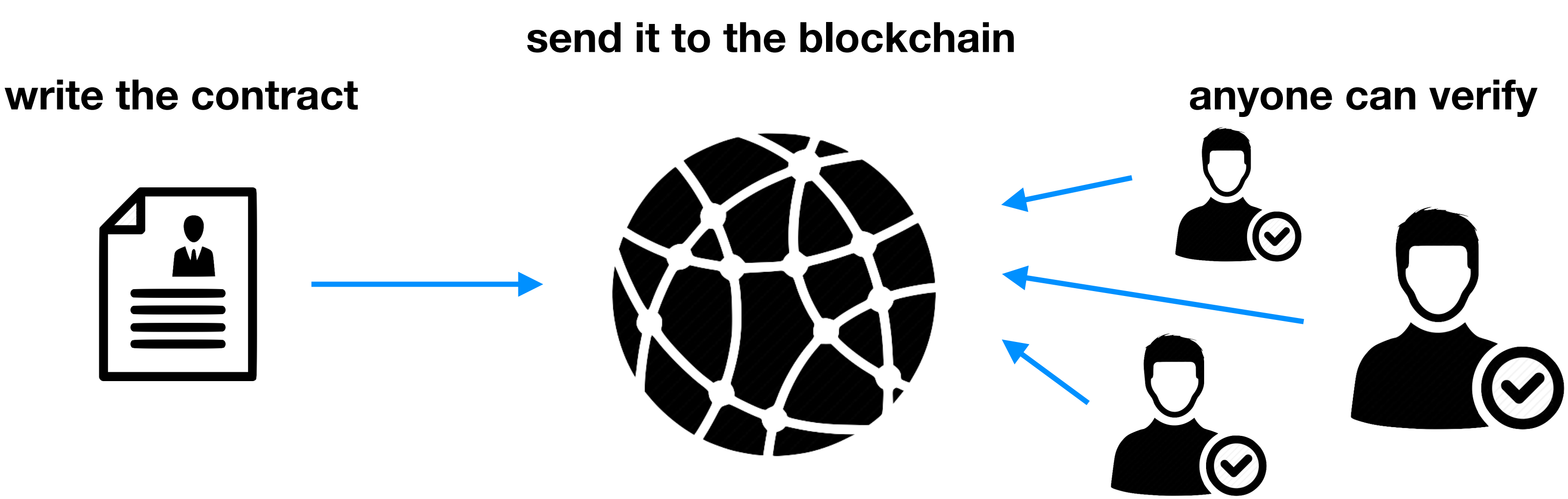
send it to the blockchain

write the contract



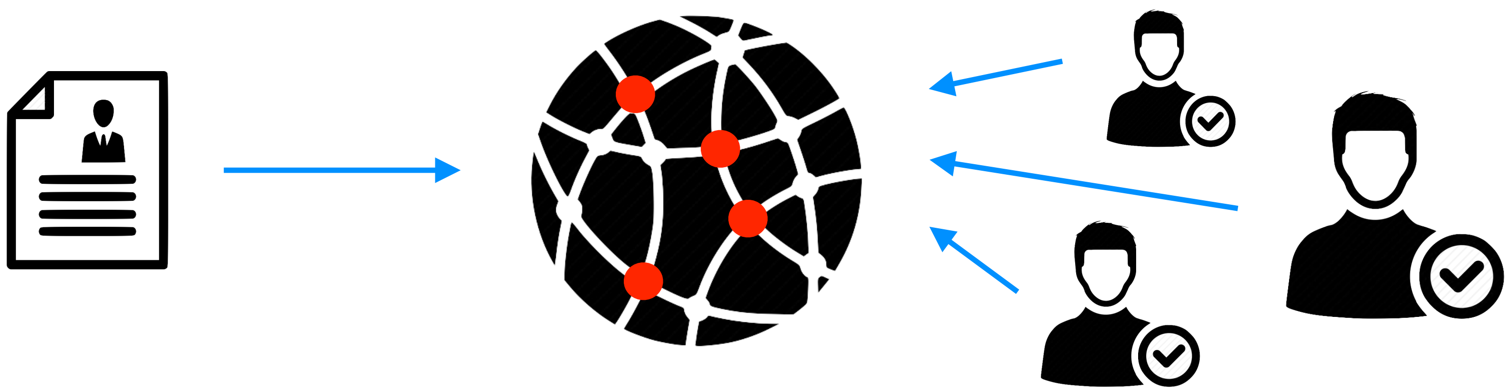
Big challenges in blockchains

- ✗ **Poor privacy**
- ✗ Scalability
- ✗ Governance
- ✗ Regulations



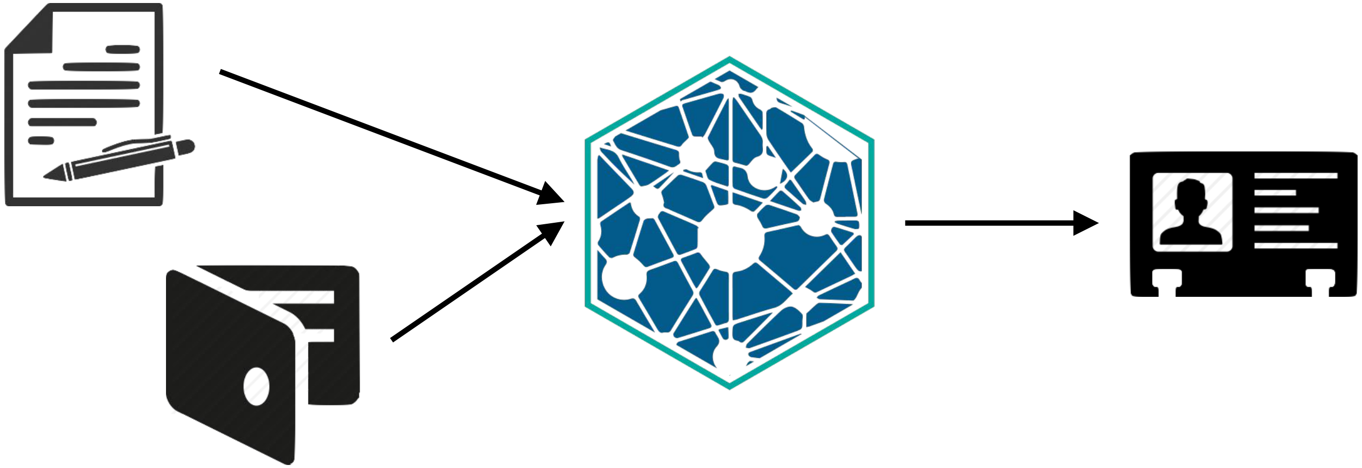
Big challenges in blockchains

- Poor privacy
- Governance
- Scalability
- Regulations



What are we trying to do?

- Issuing credentials through smart contracts



… while preserving privacy

What are we trying to do?

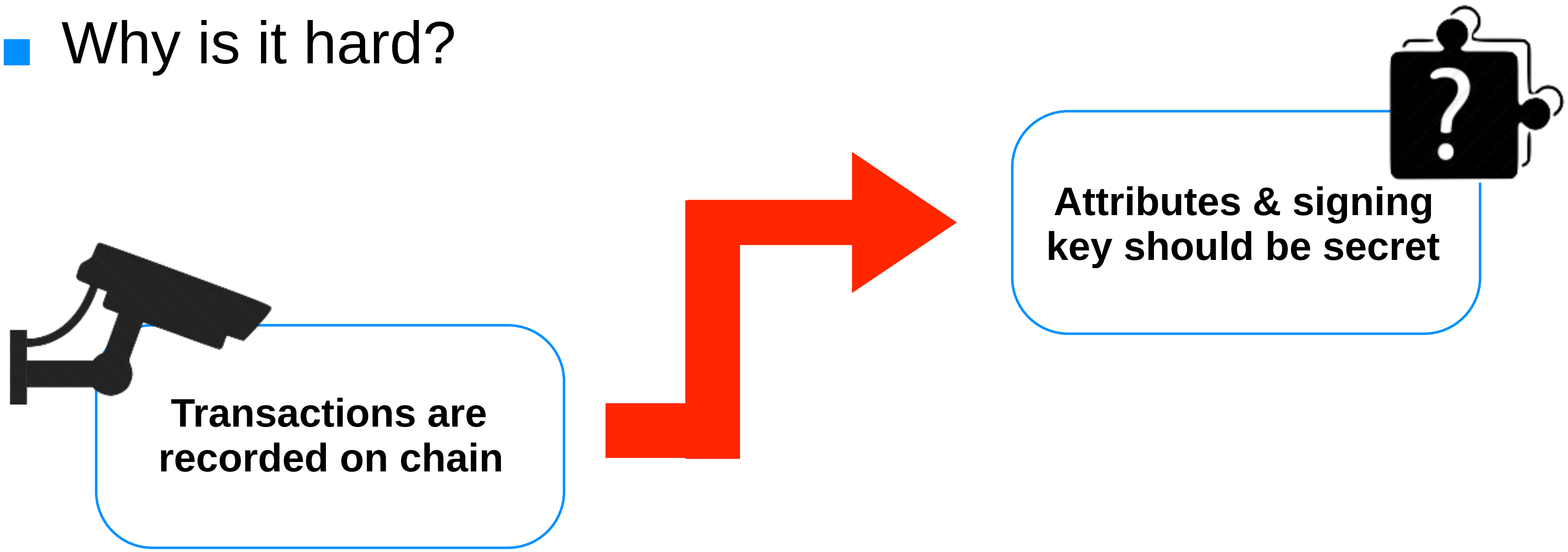
- Why is it hard?

Transactions are recorded on chain

In a decentralised setting

What are we trying to do?

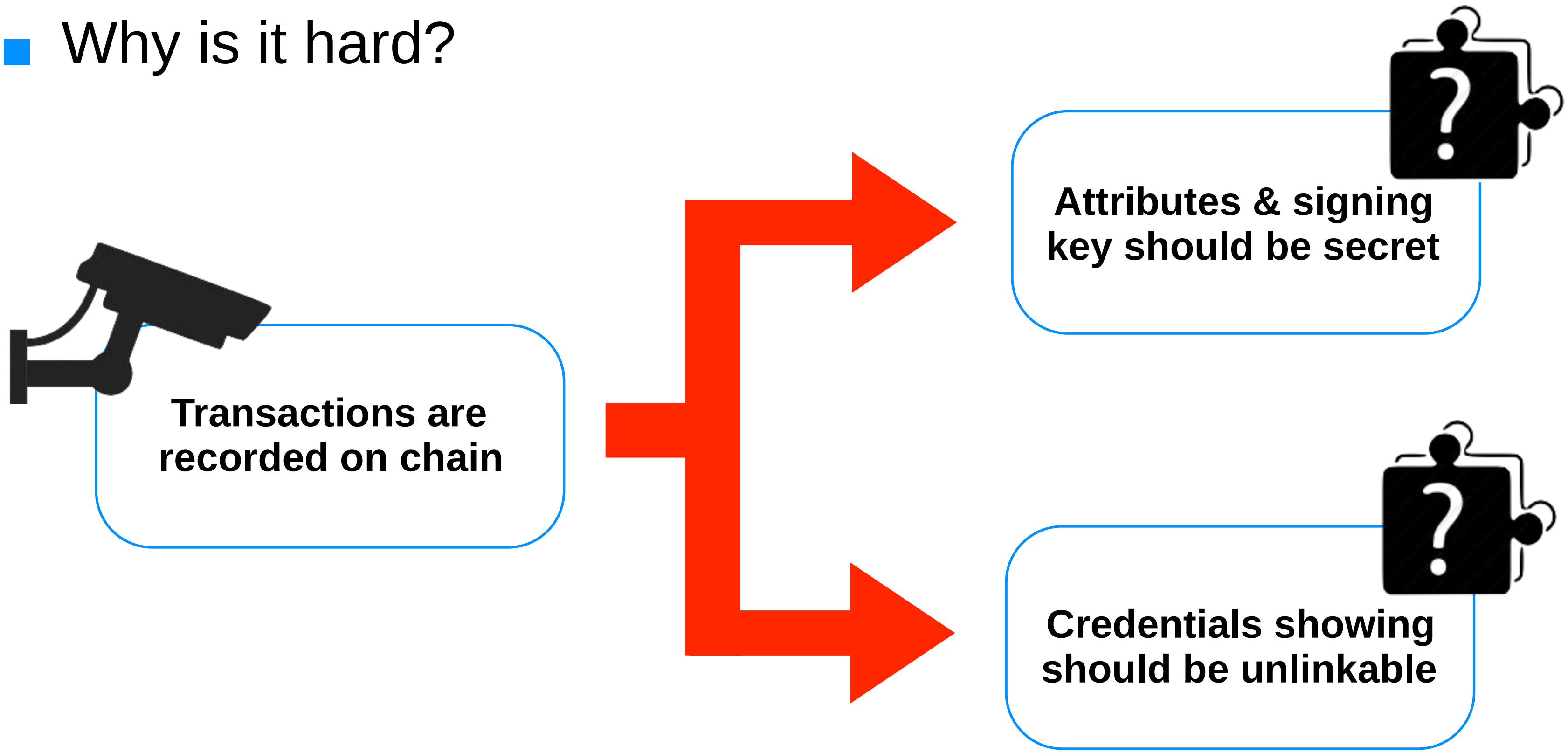
- Why is it hard?



In a decentralised setting

What are we trying to do?

- Why is it hard?



In a decentralised setting

So we built Coconut

Introduction

- What is coconut?

contribution I

Coconut credentials scheme

Introduction

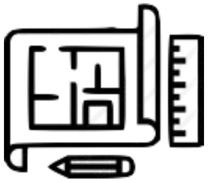
- What is coconut?

contribution I

Coconut credentials scheme

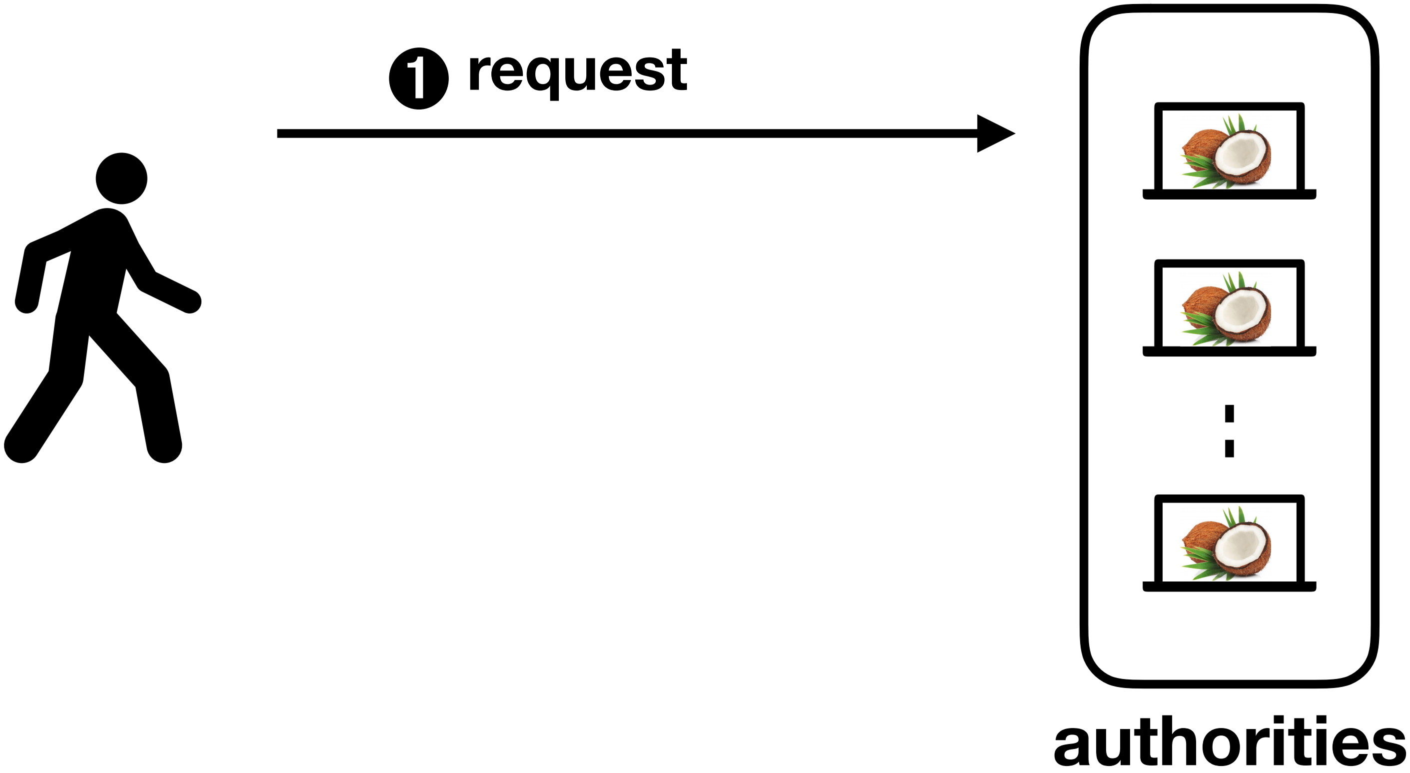
contribution II

Coconut smart contract library



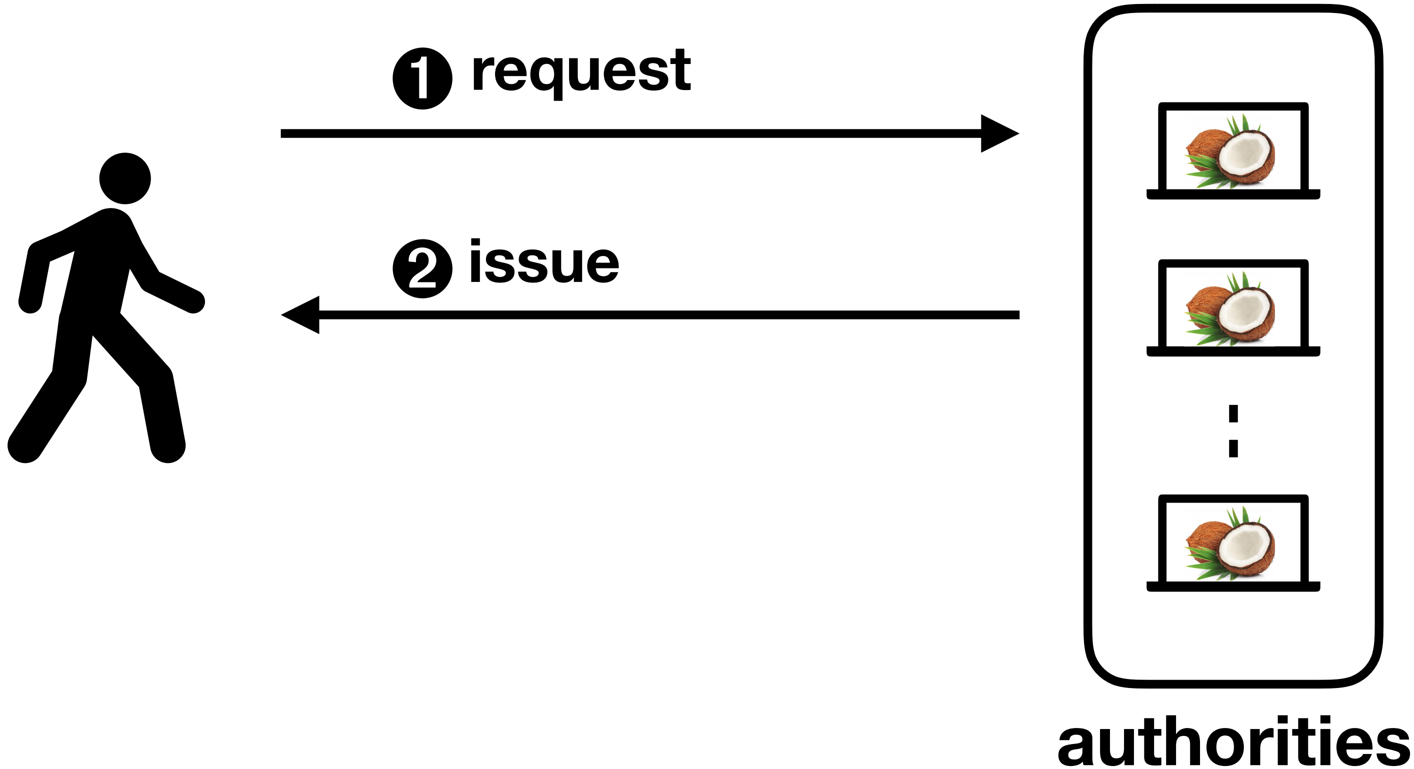
System Overview

- How Coconut works?



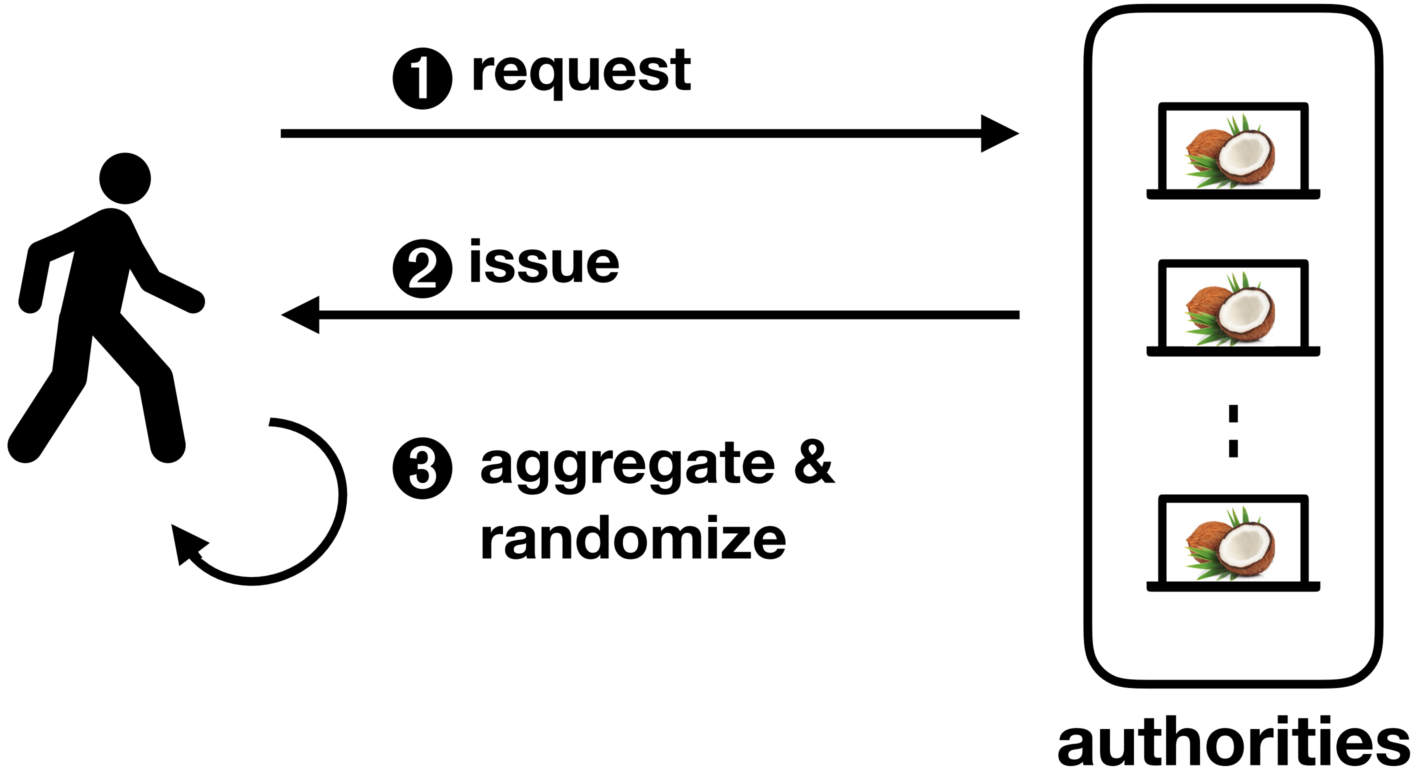
System Overview

- How Coconut works?



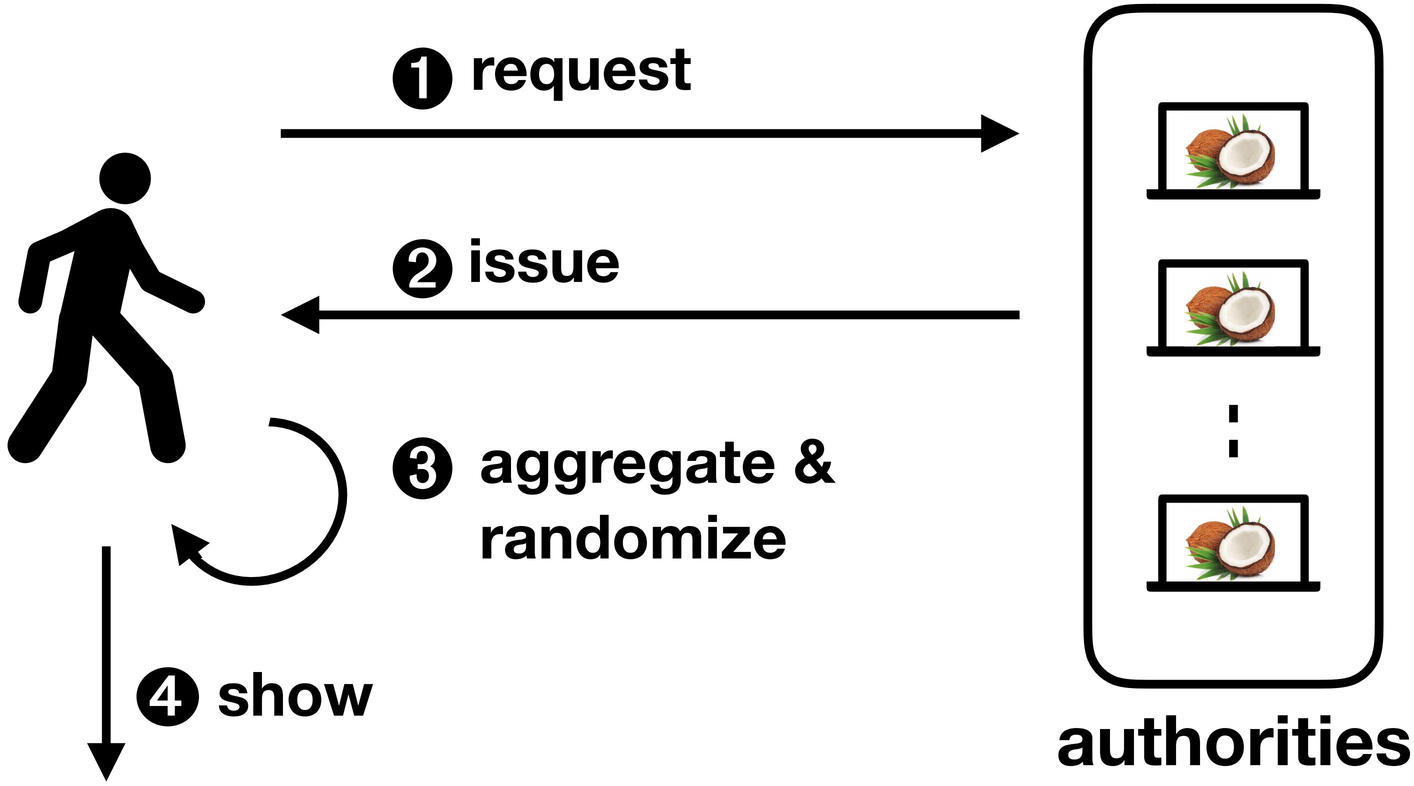
System Overview

- How Coconut works?



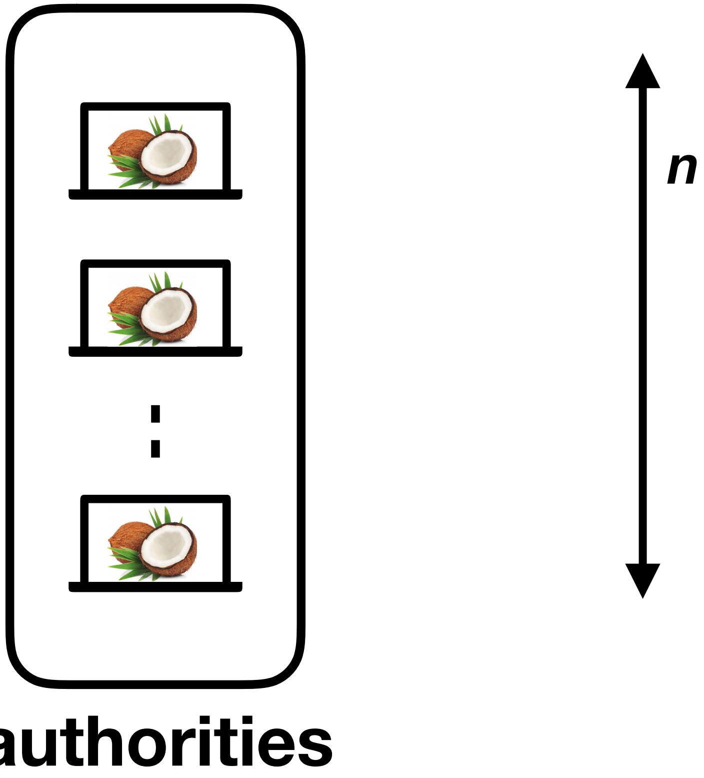
System Overview

- How Coconut works?



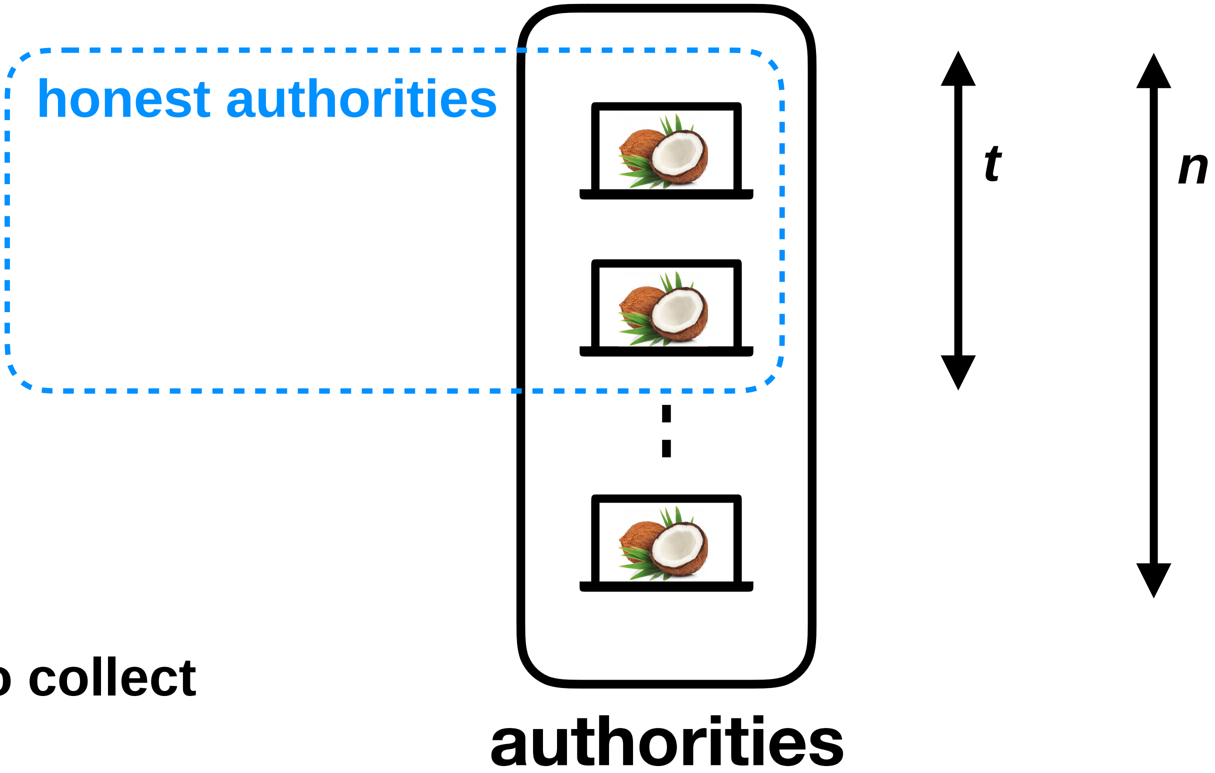
System Overview

- Threshold authorities



System Overview

- Threshold authorities



Users need to collect only *t* shares

Coconut Credentials Scheme

- Cryptographic primitives

initialisation	❖ Setup ❖ KeyGen ❖ AggregateKey
get signature	❖ ShowBlindSign ⟷ BlindSign
break linkability	❖ Randomize ❖ AggregateSign
verify	❖ PrepareBlindSign ⟷ BlindVerify

Coconut Credentials Scheme

- Where does Coconut come from?

Coconut

BLS Signature + **Waters Scheme**

Coconut Credentials Scheme

- Where does Coconut come from?

Coconut

BLS Signature + **Waters Scheme**

- What does it look like?

> take an an attribute: m
>
> compute: $h \leftarrow H(c_m)$
>
> signature: $\sigma \leftarrow (h, h^{x+my})$ & secret key: (x, y)

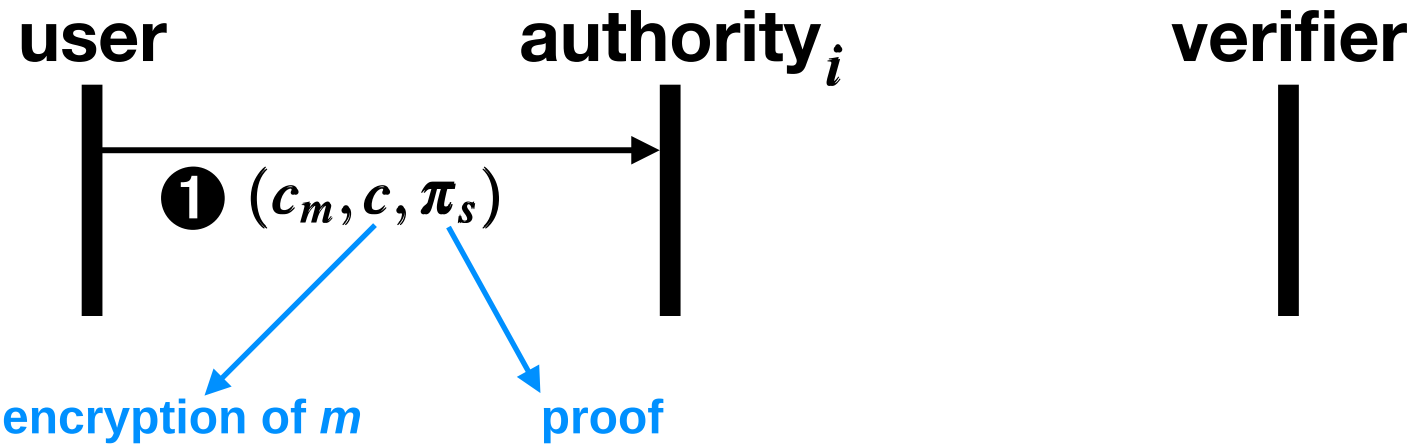
Coconut Credentials Scheme

- Issuing & showing protocols

user **authority** $_i$ **verifier**

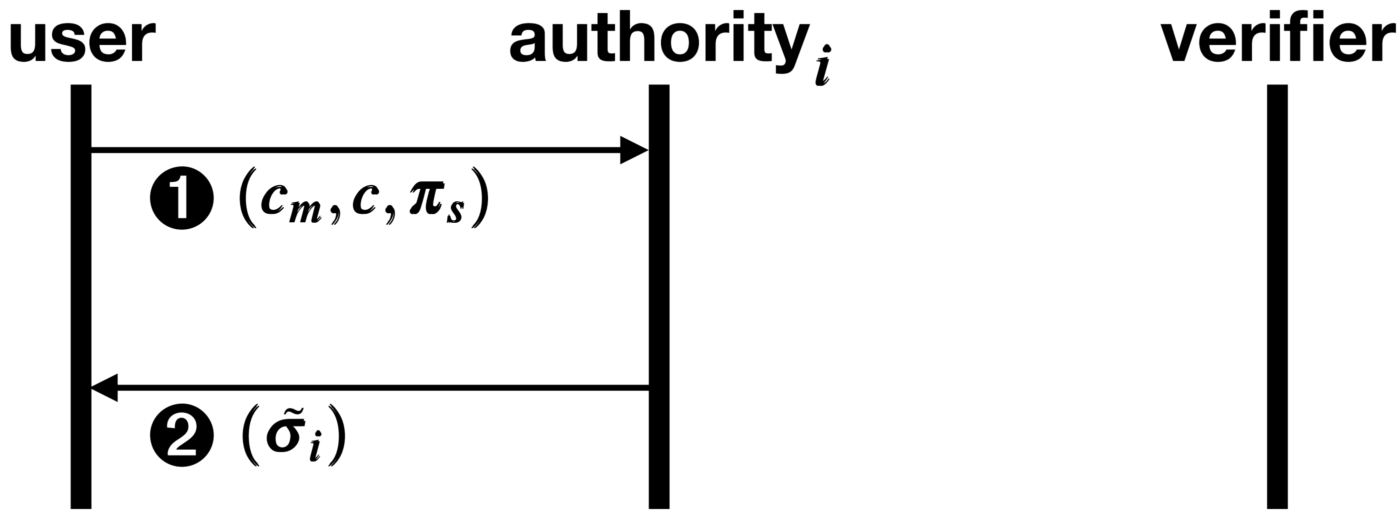
Coconut Credentials Scheme

- Issuing & showing protocols



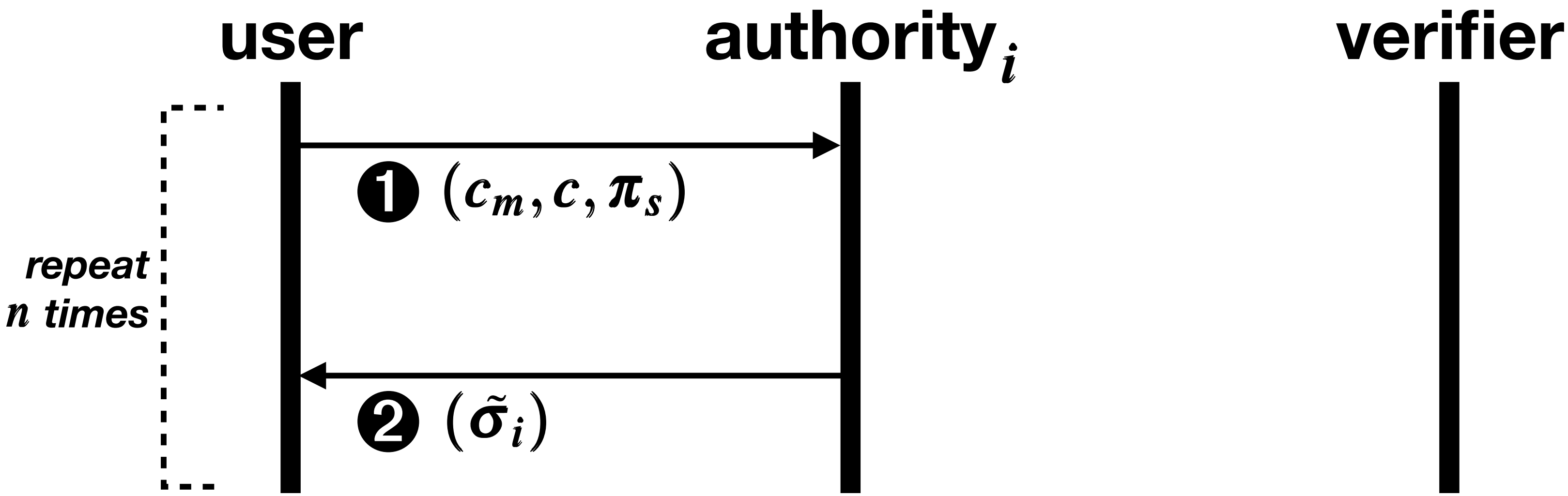
Coconut Credentials Scheme

- Issuing & showing protocols



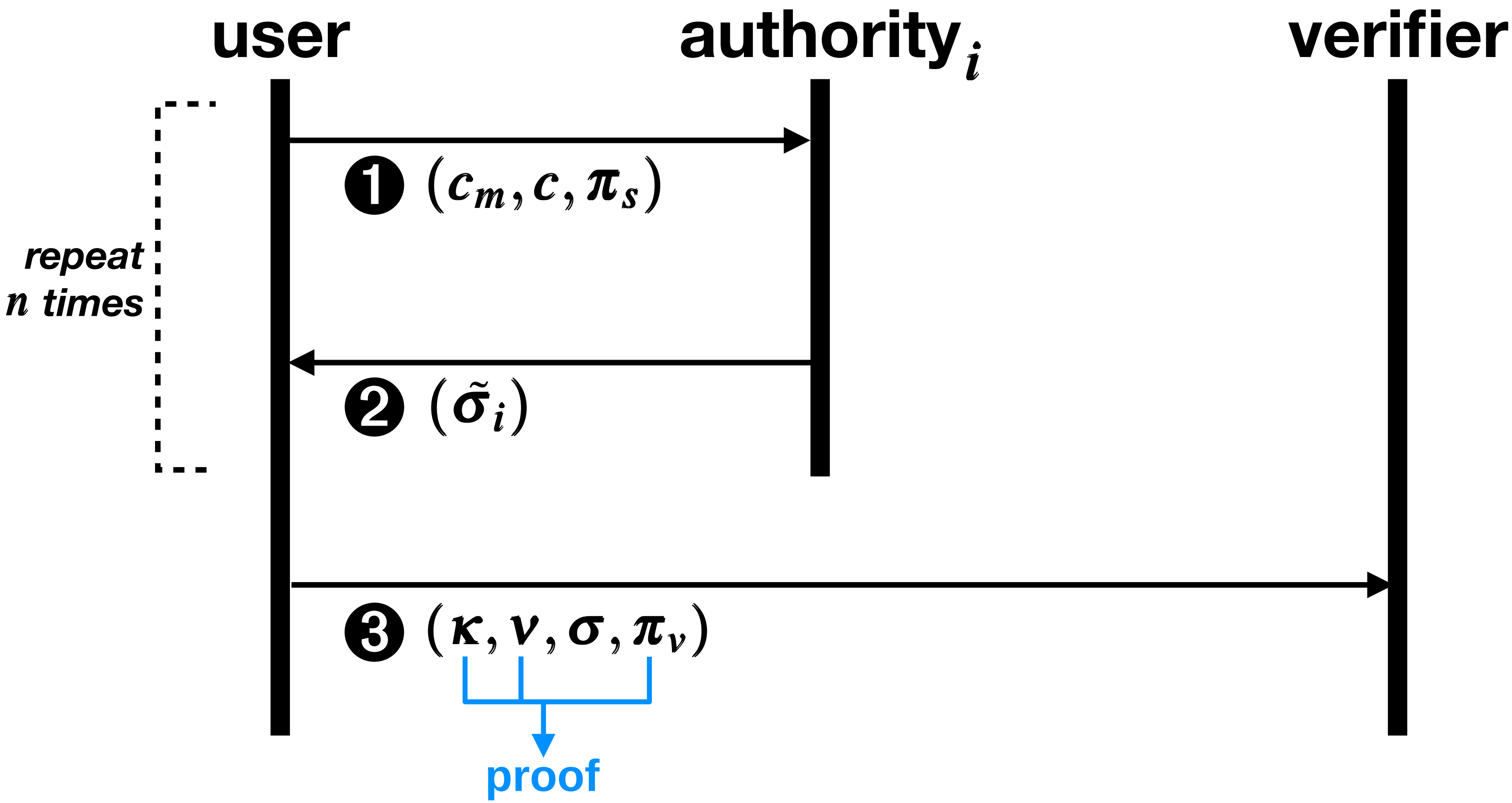
Coconut Credentials Scheme

- Issuing & showing protocols



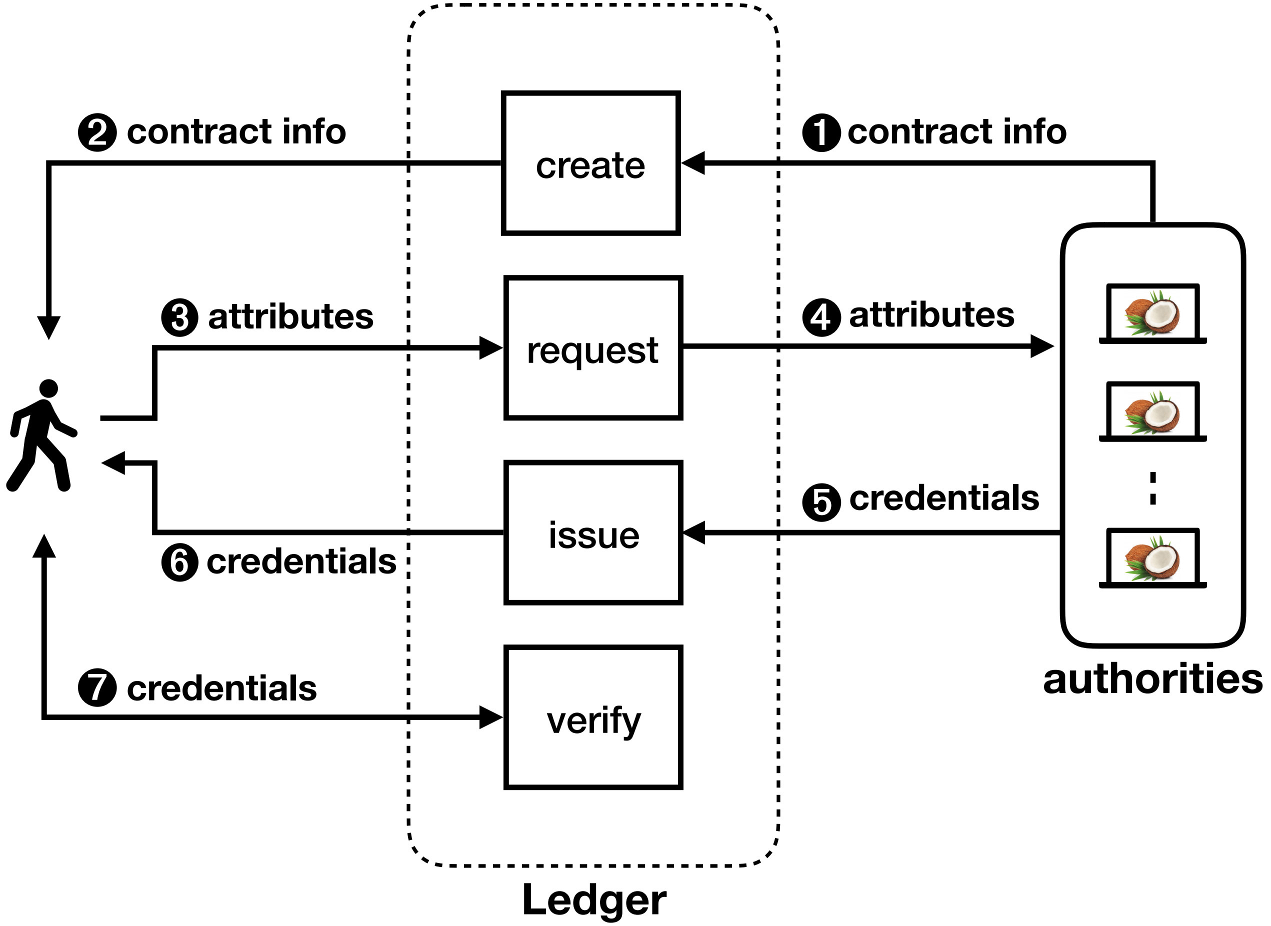
Coconut Credentials Scheme

- Issuing & showing protocols



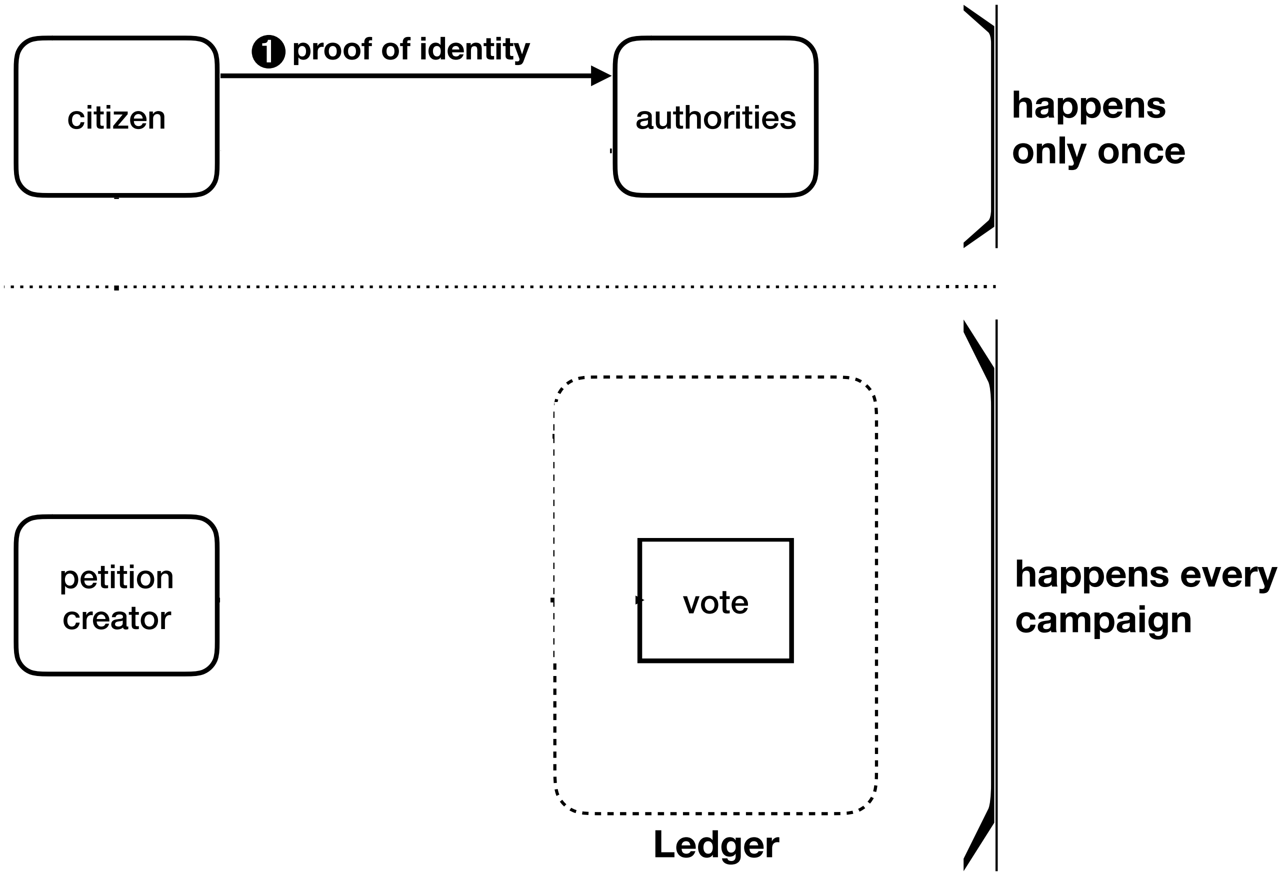
Smart Contract Library

- Chainspace Coconut library



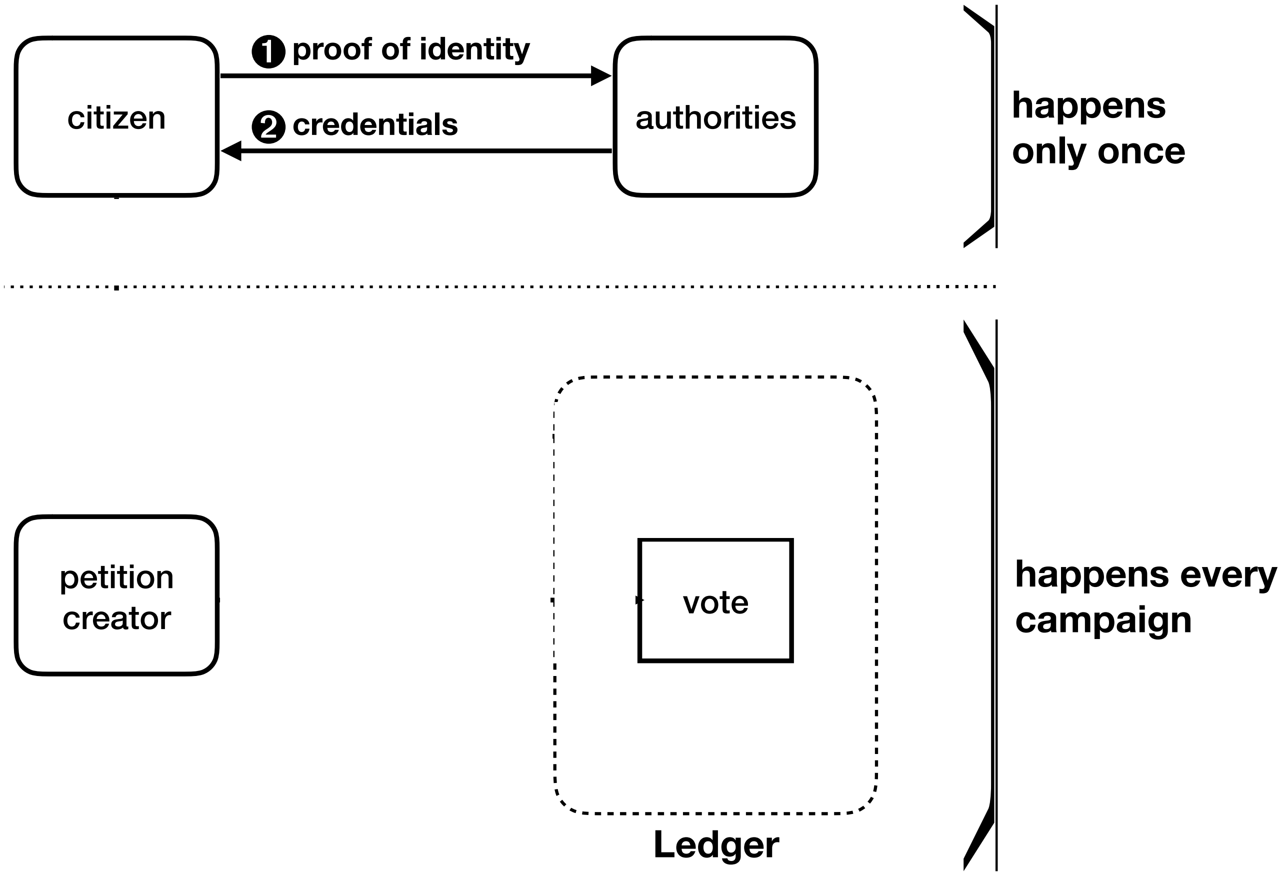
Applications

- Privacy-preserving petitions



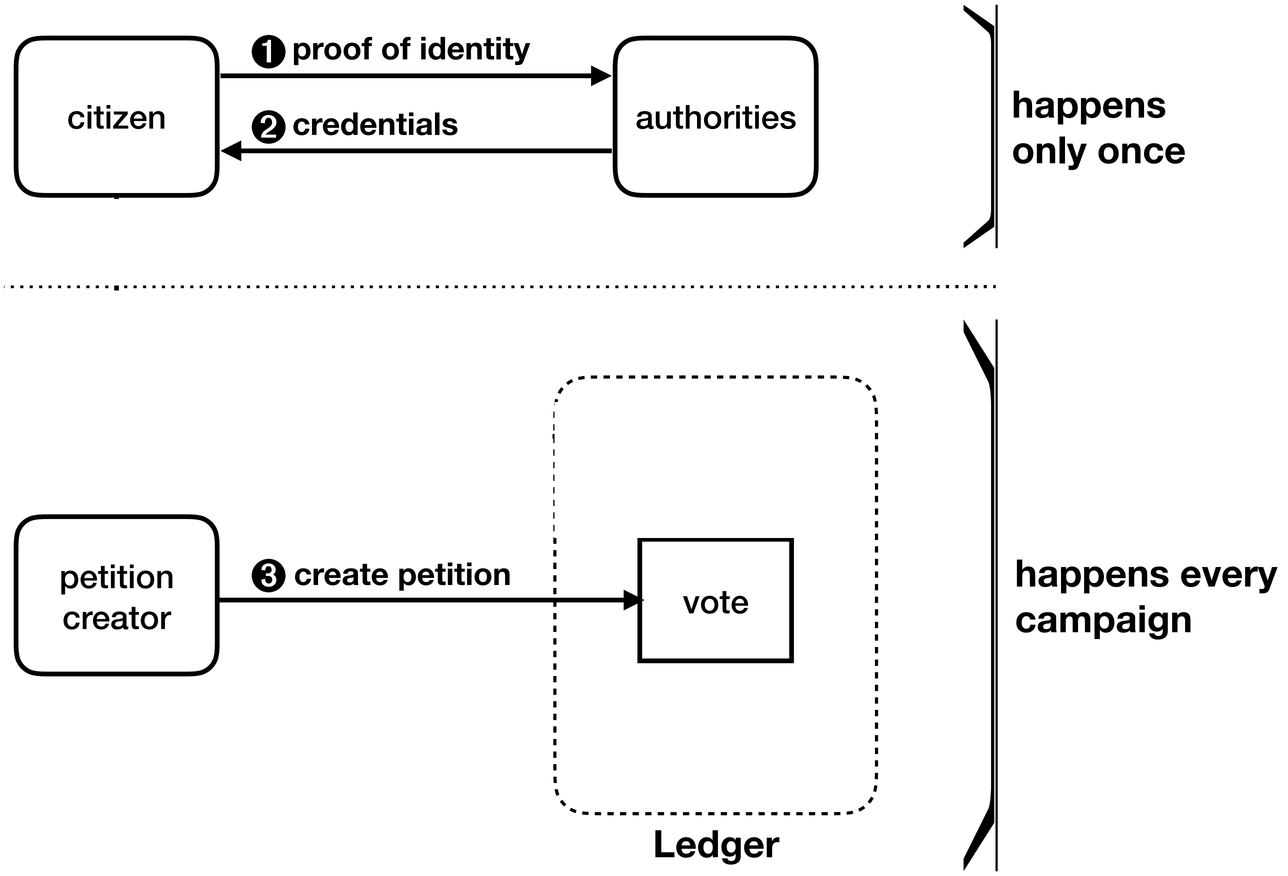
Applications

- Privacy-preserving petitions



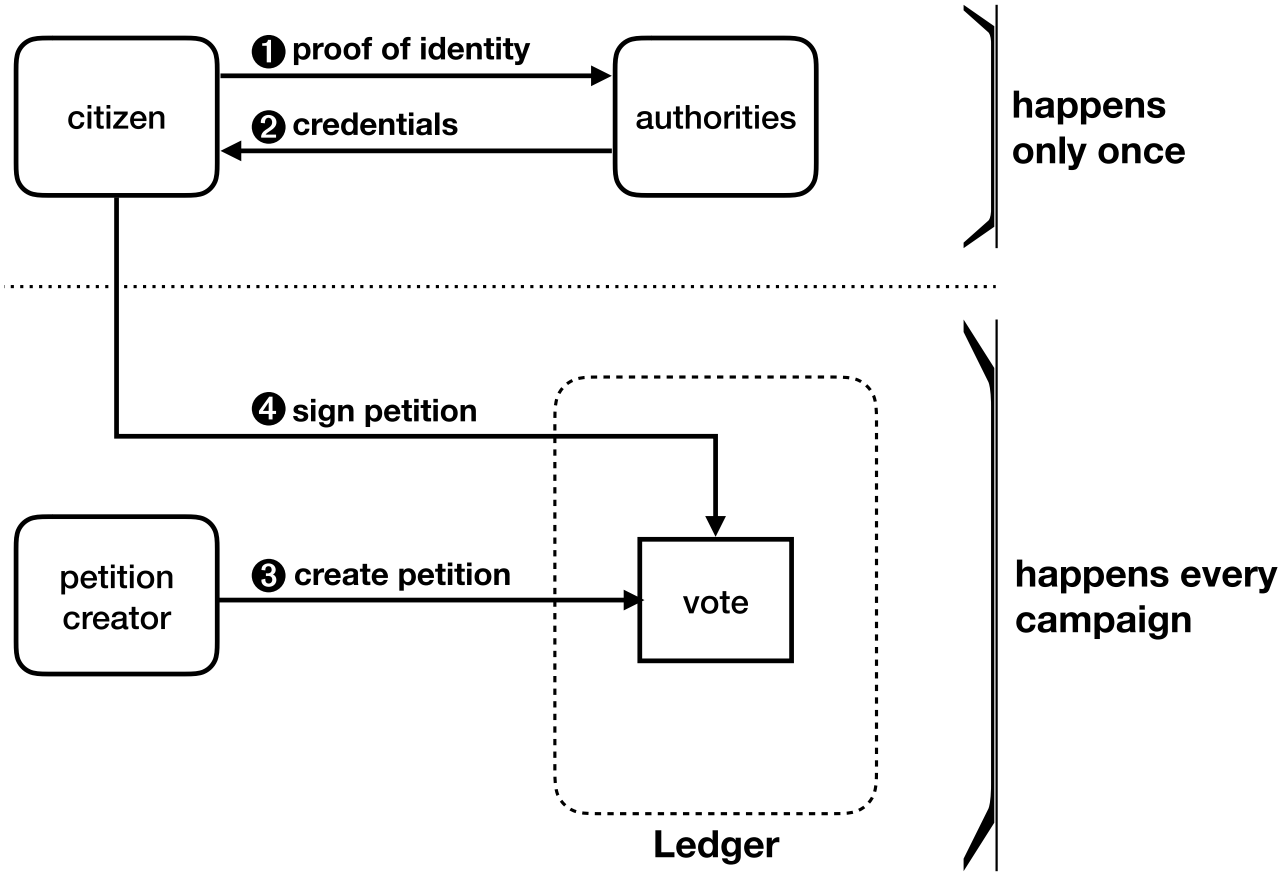
Applications

- Privacy-preserving petitions



Applications

- Privacy-preserving petitions



Performance

- What did we implement?

Performance

- What did we implement?

The Coconut cryptographic library

Python & Timing benchmark

Performance

- What did we implement?

The Coconut cryptographic library

Python & Timing benchmark

Smart contract library

&

UCL

Performance

- What did we implement?

The Coconut cryptographic library

Python & Timing benchmark

Smart contract library

&

Applications

Coin tumbler
E-Petition
(CRD proxy distribution)

Performance

- What did we implement?

The Coconut cryptographic library

Python & Timing benchmark

Smart contract library

&

Applications

Coin tumbler
E-Petition
(CRD proxy distribution)

Everything is released as open source software

`https://github.com/asonnino/coconut`

Performance

- What is the credentials size?

2 Group Elements

Performance

- What is the credentials size?

2 Group Elements

No matter how many attributes…

Performance

- What is the credentials size?

2 Group Elements

No matter how many attributes…

No matter how many authorities…

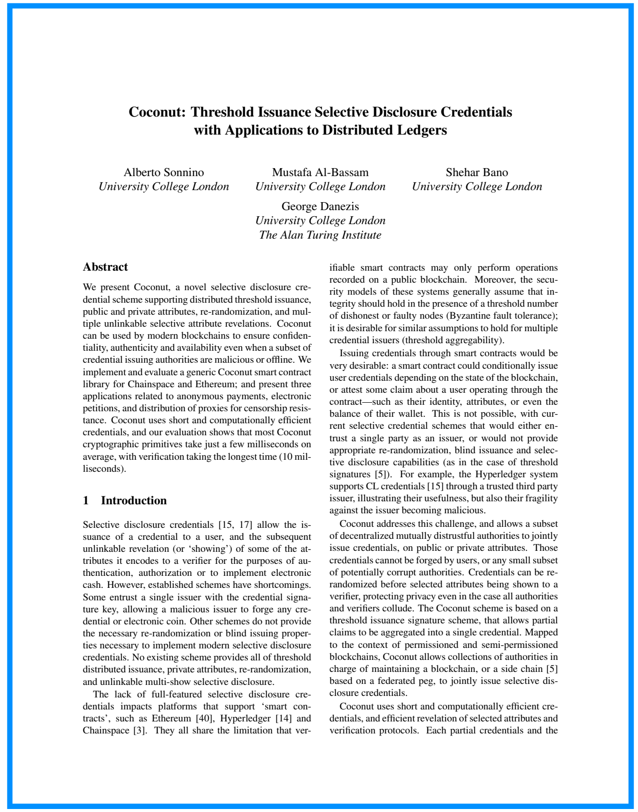
What else is in the paper?

Full cryptographic scheme

Smart contract library evaluation

Coin tumbler, CRD proxy applications

Applications evaluation and benchmarking



Coconut: Threshold Issuance Selective Disclosure Credentials with Applications to Distributed Ledgers

Alberto Sonnino
University College London

Mustafa Al-Bassam
University College London

Shehar Bano
University College London

George Danezis
University College London
The Alan Turing Institute

Abstract

We present Coconut, a novel selective disclosure credential scheme supporting distributed threshold issuance, public and private attributes, re-randomization, and multiple unlinkable selective attribute revelations. Coconut can be used by modern blockchains to ensure confidentiality, authenticity and availability even when a subset of credential issuing authorities are malicious or offline. We implement and evaluate a generic Coconut smart contract library for Chainspace and Ethereum; and present three applications related to anonymous payments, electronic petitions, and distribution of proxies for censorship resistance. Coconut uses short and computationally efficient credentials, and our evaluation shows that most Coconut cryptographic primitives take just a few milliseconds on average, with verification taking the longest time (10 milliseconds).

1 Introduction

Selective disclosure credentials [15, 17] allow the issuance of a credential to a user, and the subsequent unlinkable revelation (or 'showing') of some of the attributes it encodes to a verifier for the purposes of authentication, authorization or to implement electronic cash. However, established schemes have shortcomings. Some entrust a single issuer with the credential signature key, allowing a malicious issuer to forge any credential or electronic coin. Other schemes do not provide the necessary re-randomization or blind issuing properties necessary to implement modern selective disclosure credentials. No existing scheme provides all of threshold distributed issuance, private attributes, re-randomization, and unlinkable multi-show selective disclosure.

The lack of full-featured selective disclosure credentials impacts platforms that support 'smart contracts', such as Ethereum [40], Hyperledger [14] and Chainspace [3]. They all share the limitation that verifiable smart contracts may only perform operations recorded on a public blockchain. Moreover, the security models of these systems generally assume that integrity should hold in the presence of a threshold number of dishonest or faulty nodes (Byzantine fault tolerance); it is desirable for similar assumptions to hold for multiple credential issuers (threshold aggregability).

Issuing credentials through smart contracts would be very desirable: a smart contract could conditionally issue user credentials depending on the state of the blockchain, or attest some claim about a user operating through the contract—such as their identity, attributes, or even the balance of their wallet. This is not possible, with current selective credential schemes that would either entrust a single party as an issuer, or would not provide appropriate re-randomization, blind issuance and selective disclosure capabilities (as in the case of threshold signatures [5]). For example, the Hyperledger system supports CL credentials [15] through a trusted third party issuer, illustrating their usefulness, but also their fragility against the issuer becoming malicious.

Coconut addresses this challenge, and allows a subset of decentralized mutually distrustful authorities to jointly issue credentials, on public or private attributes. Those credentials cannot be forged by users, or any small subset of potentially corrupt authorities. Credentials can be re-randomized before selected attributes being shown to a verifier, protecting privacy even in the case all authorities and verifiers collude. The Coconut scheme is based on a threshold issuance signature scheme, that allows partial claims to be aggregated into a single credential. Mapped to the context of permissioned and semi-permissioned blockchains, Coconut allows collections of authorities in charge of maintaining a blockchain, or a side chain [5] based on a federated peg, to jointly issue selective disclosure credentials.

Coconut uses short and computationally efficient credentials, and efficient revelation of selected attributes and verification protocols. Each partial credentials and the

Conclusion

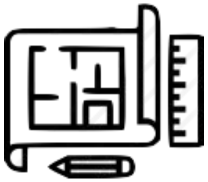
- What did we talked about ?

contribution I

Coconut signature scheme

contribution II

Coconut smart contract library



Thank you for your attention
Questions?

Alberto Sonnino

alberto.sonnino@ucl.ac.uk

https://sonnino.com

`https://github.com/asonnino/coconut`

This work is supported in part by EPSRC Grant EP/M013286/1, the EU H2020 DECODE project (grant agreement number 732546), and The Alan Turing Institute.