You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients

SERIAL ’19

Kristof Jannes, Bert Lagaisse, Wouter Joosen

9 December 2019


OIL & VINEGAR
INGREDIENTS: activated

eLoyalty: loyalty points across merchants

Integrated loyalty programs between several small stores

Redeem points at any participating store

! Double-spending problem

Customer spends same loyalty point twice

! Decentralized: no central authority

Merchants do not fully trust each other

Your local merchant can’t run a blockchain node

Expensive hardware requirements

Requires large amount of processing power and storage space

Complex backend setup

Requiring large infrastructure to setup permissioned network

Hard to setup consortia between companies

High legal and monetary burden

INVOICE
INVOICE
TERMS
Due on receipt
F.O.B. POINT
SHIPPED VIA
REQUISITIONER
P.O. NUMBER
ESPERSON
DESCRIPTION
UNIT PRICE
TOTAL
UANTITY
1 Non-Reflective Mirror 119.95 $119.95
1 Silent Alarm Clock 139.95 $139.95
1 Teflon Bath Mat 7.95 $7.95
5 Water-Resistant Sponge 60.05 $300.25
5 Water-Soluble Dishcloth 60.05 $300.25
3 Black Light Bulb 40.09 $120.27
SUBTOTAL $988
SALES TAX
SHIPPING
TOTAL DU
PAID
invoice total is 1,206.34 $.
payable to Success Business
concerning this invoice, contact info@succcess_business.com
for your business!
Thank you
INVOICE
Business Air
555 South Tremont Way
Calgary, AB T5J 8K0 CA
Date
Description
21/04 Transient Room Revenue
21/04 Room Tax- 6%
21/04 Occupancy Tax- 4
22/04 Transient R
22/04 Room
22/04


LOAN

eLoan: loans with unpaid invoices as collateral

Financial institutions offer loans to companies

Using unpaid invoices as collateral, verified by the network

! Double-spending problem

Company uses invoice twice

! Financial institutions boycott each other

They all want to give the loan themselves

Your bank is not going to rely on proof-of-work

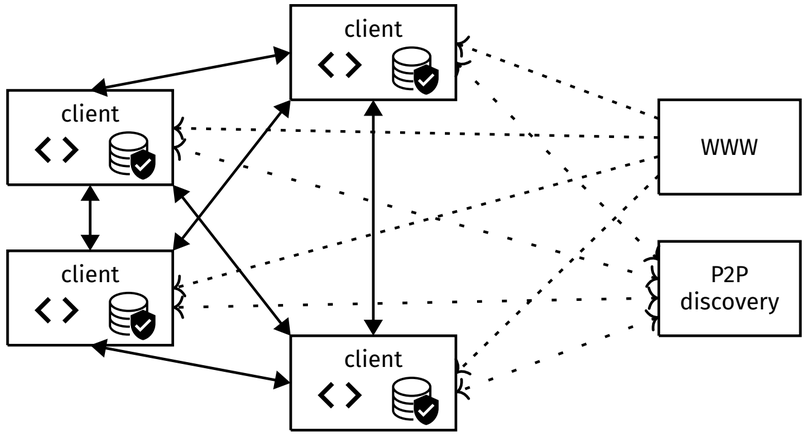
No consensus finality

Forks can happen with decreasing probability over time

Blockchain keeps track of all data

Can impose problems with privacy laws such as GDPR

You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients



Lightweight middleware for consensus

Architecture and consensus protocol

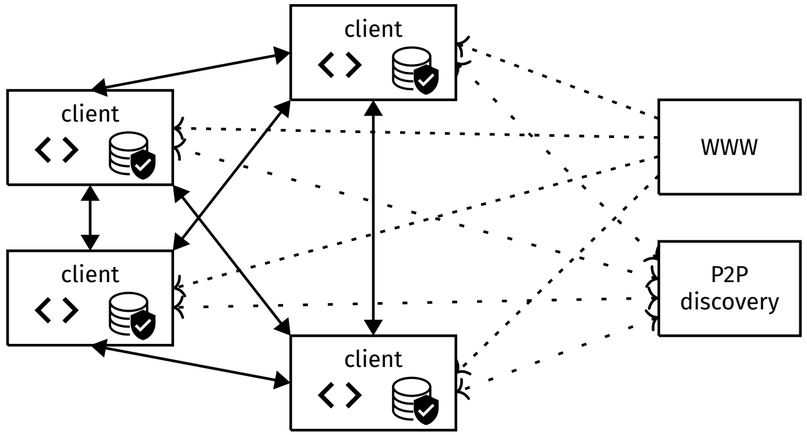
Evaluation

Performance and infrastructure

Zooming out

State-of-the-art and future work

You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients



Lightweight middleware for consensus

Architecture and consensus protocol

Evaluation

Performance and infrastructure

Zooming out

State-of-the-art and future work

Use peer-to-peer state-synchronization with voting for Byzantine fault-tolerance

State-based approach using signed data-structures

No ledger with history as in operation-based approaches

Peer-to-peer state-synchronization

No central component that manages the system

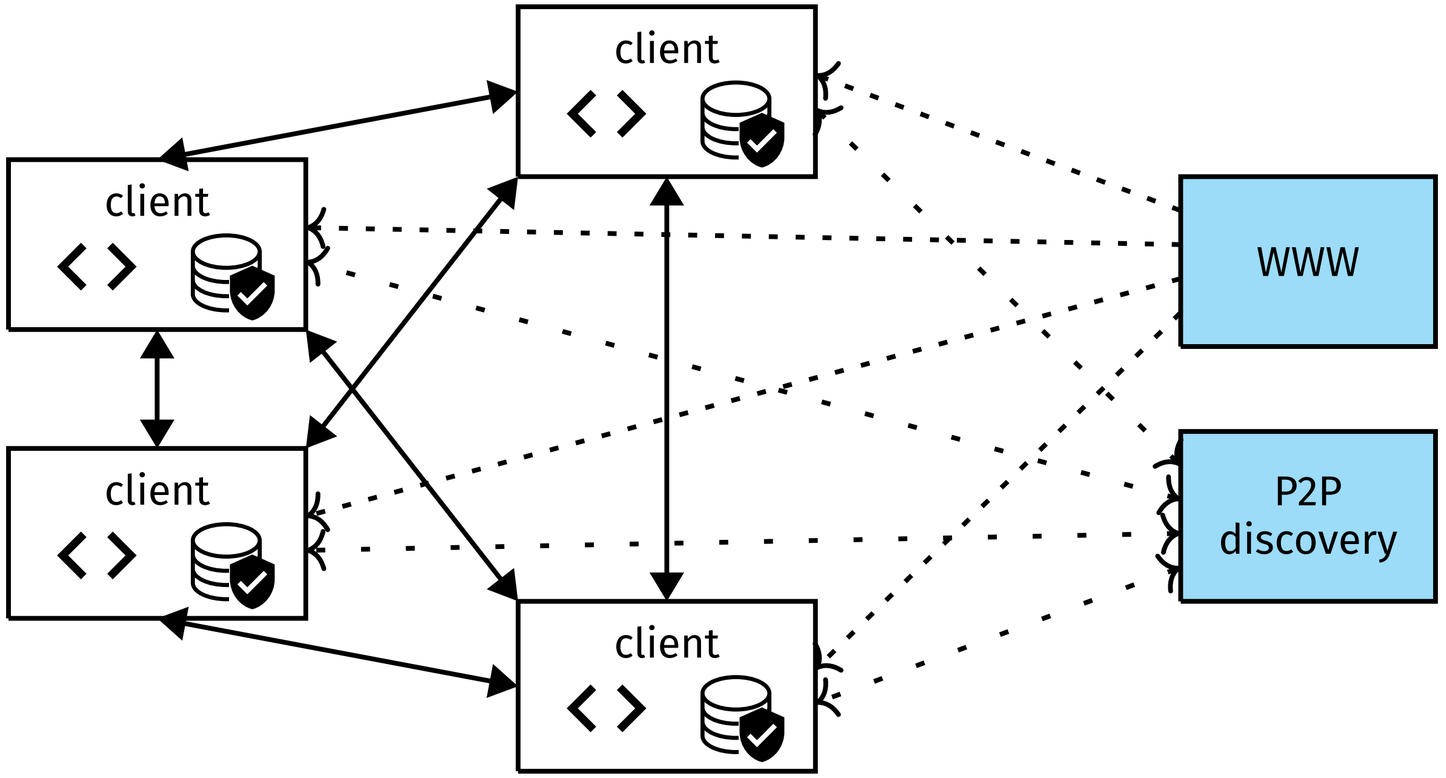
Voting for Byzantine fault-tolerance

Tolerate both crash failures as well as malicious nodes

Lightweight setup

Client-side peer-to-peer network running in the browser

The lightweight architecture requires only two small backend components



There is no distributed ledger with coins, only tickets which can be used only once

- Coins can be used forever
  - Ownership changes each time it is used
- Tickets can be used only once
  - Ticket is redeemed when it is used
- → Only track the unspent tickets
  - No need for a ledger

Protocol has two steps

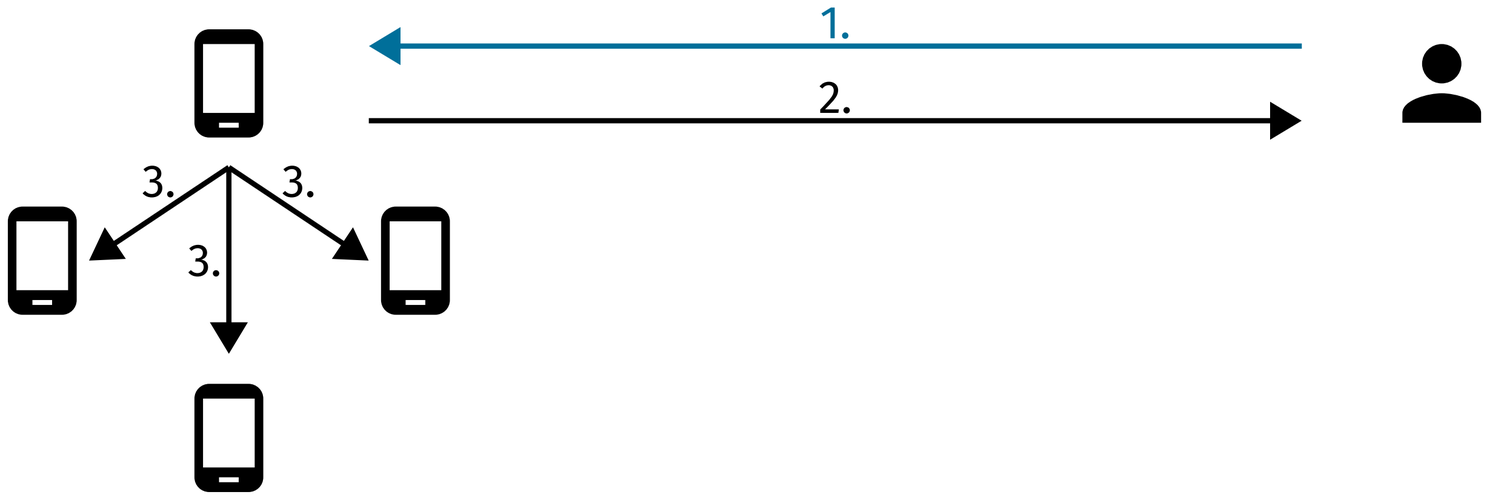
1. Issue new token

2. Redeem token for asset (e.g. loan or loyalty reward)

Protocol to issue tokens

1. Customer asks a new token from a node

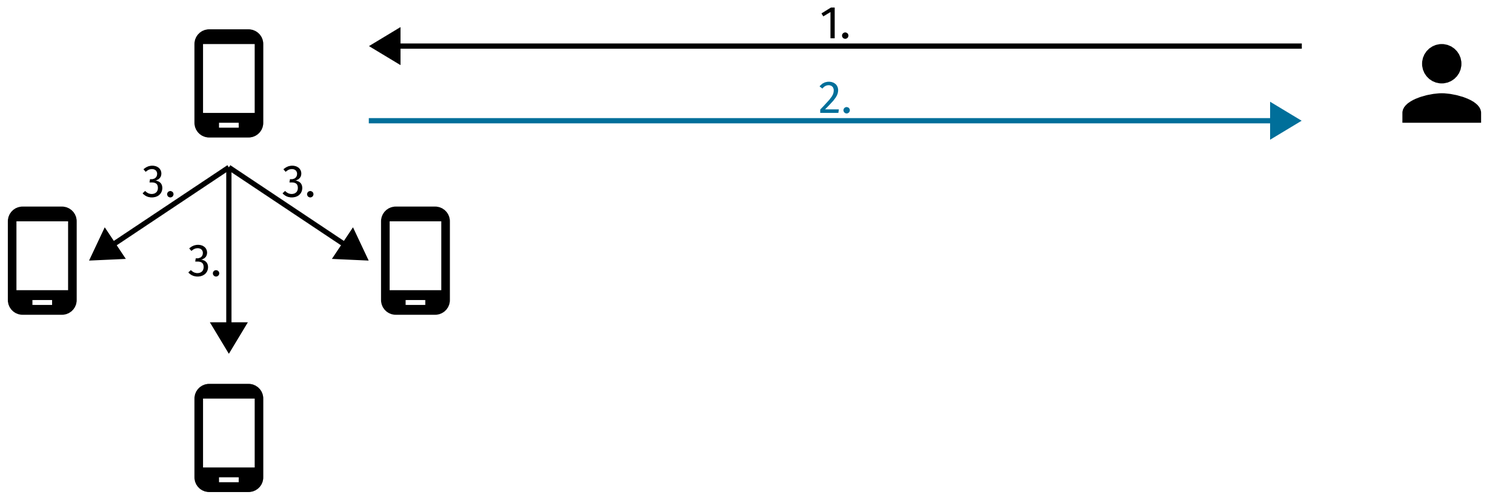
   Providing its public key and application-specific info



Protocol to issue tokens

1. Customer asks a new token from a node
2. Node verifies the info and creates a new token

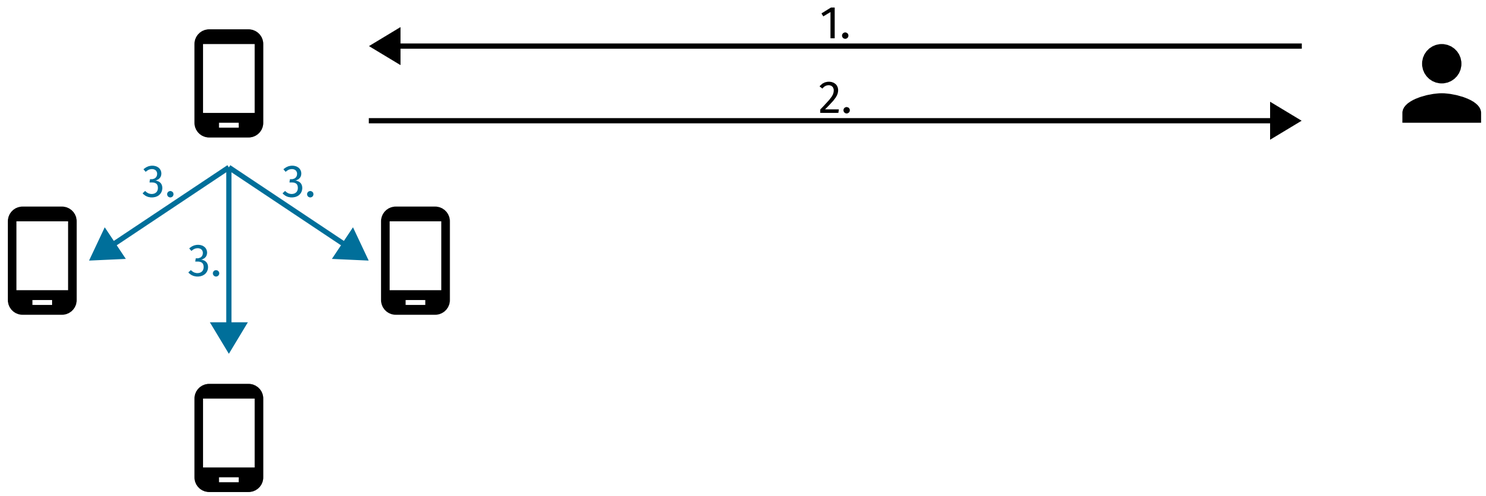
   Stores the token in the local database and sends it back to the customer



Protocol to issue tokens

1. Customer asks a new token from a node
2. Node verifies the info and creates a new token
3. Node synchronizes token asynchronously with other nodes

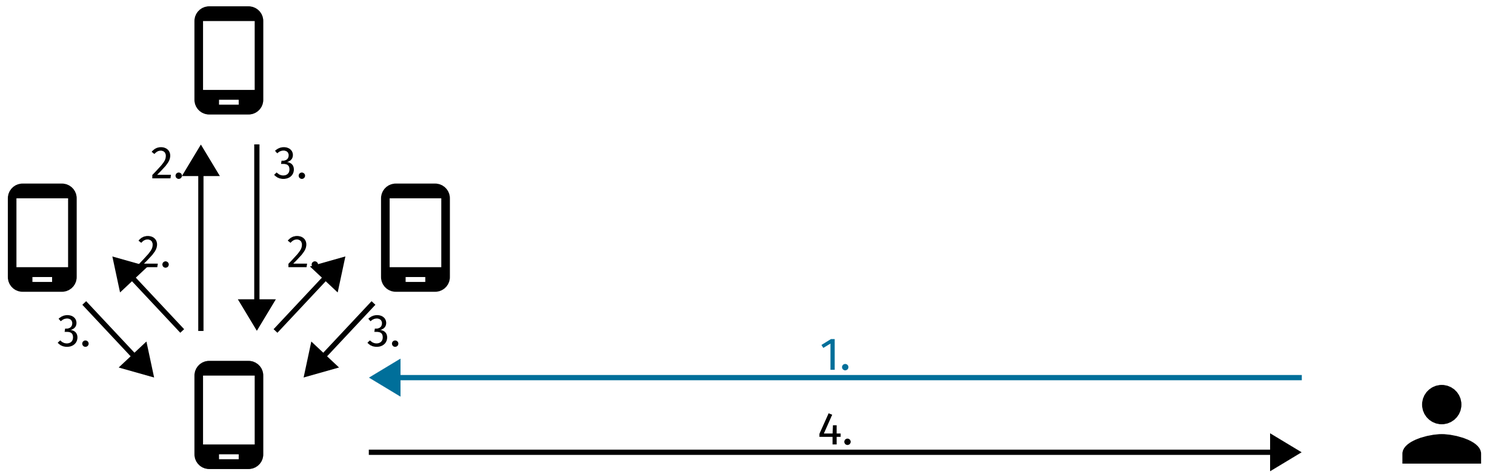
   Using a signature to guarantee authenticity and a timestamp against replay attacks



Protocol to redeem tokens

1. Customer sends ID of token to node and asks to redeem it

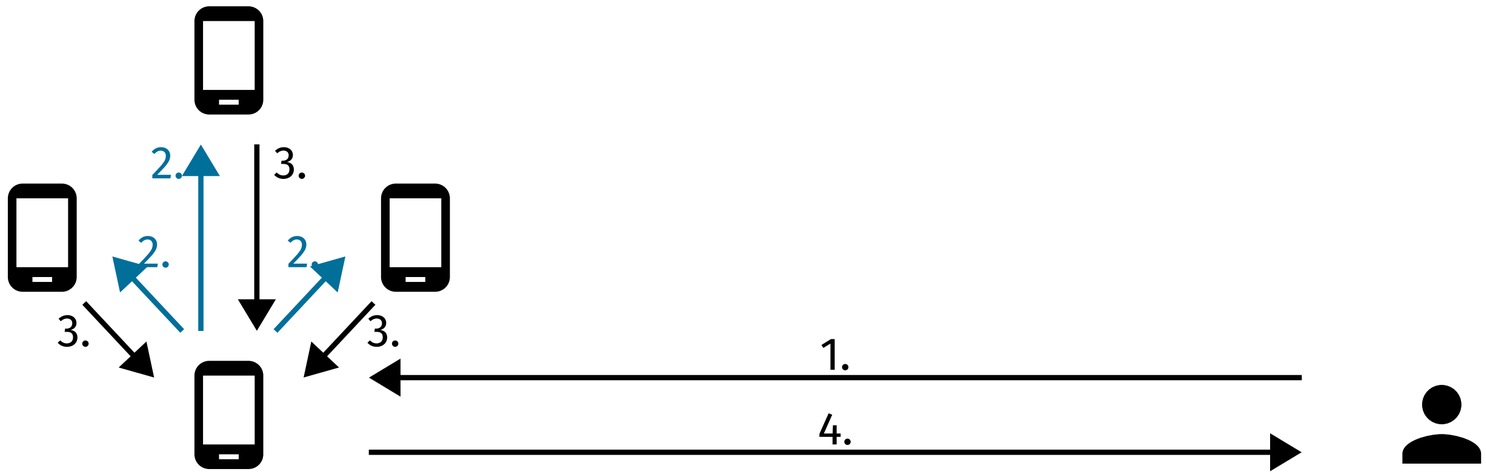
   Using its private key to prove ownership



Protocol to redeem tokens

1. Customer sends ID of token to node and asks to redeem it
2. Node verifies token and starts a vote to redeem it

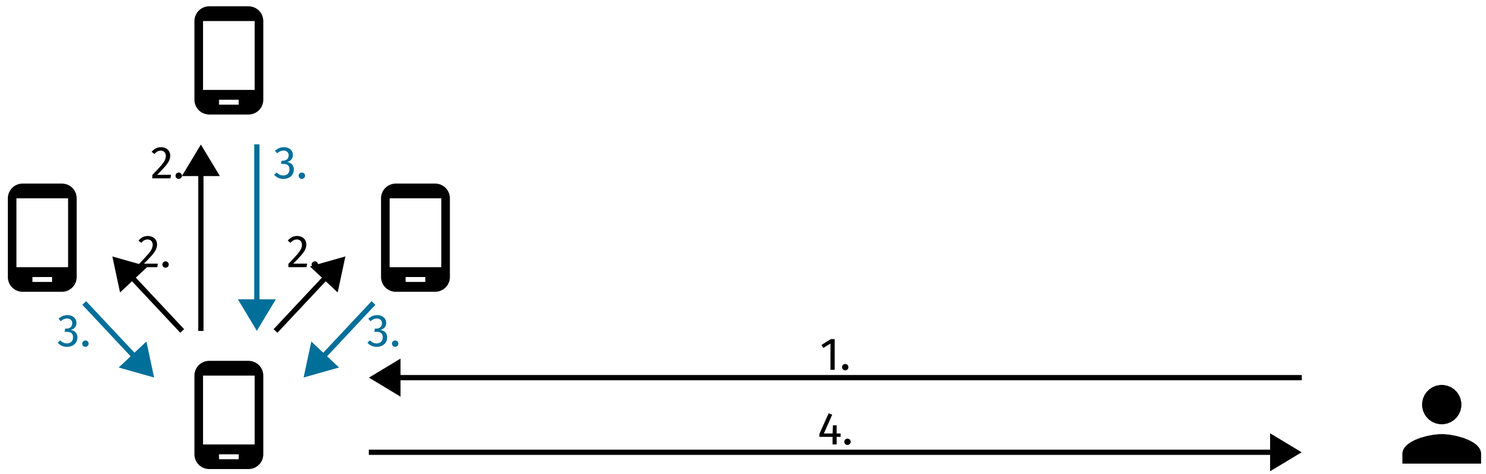
   A node can only start a vote when customer has sent his signature



Protocol to redeem tokens

1. Customer sends ID of token to node and asks to redeem it
2. Node verifies token and starts a vote to redeem it
3. Other nodes receive votes and vote themselves

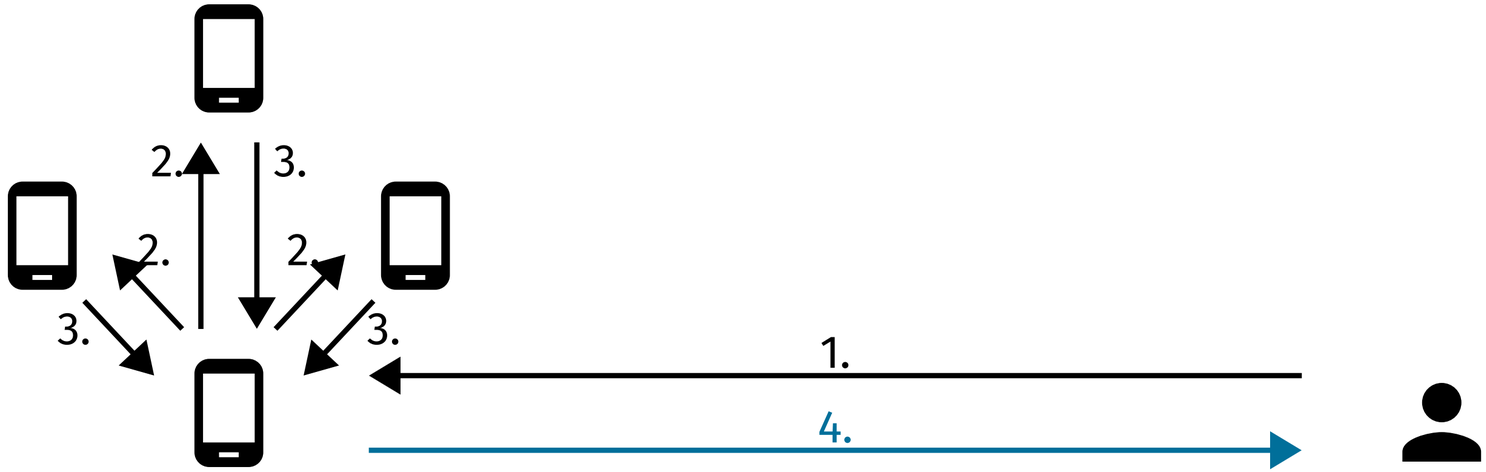
   Every node can vote exactly once for each token



Protocol to redeem tokens

1. Customer sends ID of token to node and asks to redeem it
2. Node verifies token and starts a vote to redeem it
3. Other nodes receive votes and vote themselves
4. Original node waits until it receives ⅔ × n + 1 of the votes

   Gives the real-life product to the customer afterwards



The protocol guarantees liveness and safety for honest customers

Safety: nothing bad happens

No double spending of tokens

Liveness: eventually something good happens

Tokens can actually be redeemed

Malicious customers can lock their tokens and lose them

But safety is still guaranteed

Adversary model

- Tolerates up to ⅓ × (n - 1) byzantine nodes
  - Tolerating both crash failures and malicious nodes
- Attacker cannot break the used cryptographic primitives
  - ECDSA with P-256 and SHA-256
- P2P discovery service must be honest
  - Otherwise liveness will be broken due to network partitions

Designed for a closed group of members with infrequent changes

Members can be added or removed using the same protocol

But with manual voting and verification

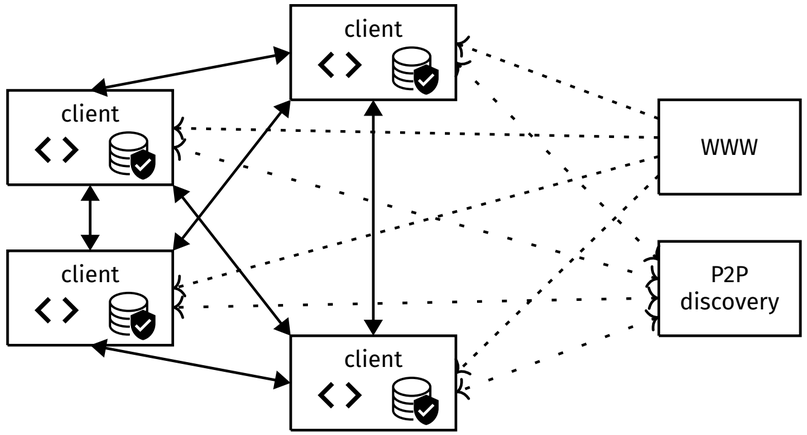
New members only participate in votes for new tokens

Membership is fixed on token creation

If too many members leave, older tokens become invalid

The ⅔ majority for those tokens cannot be reached anymore

You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients



Lightweight middleware for consensus

Architecture and consensus protocol

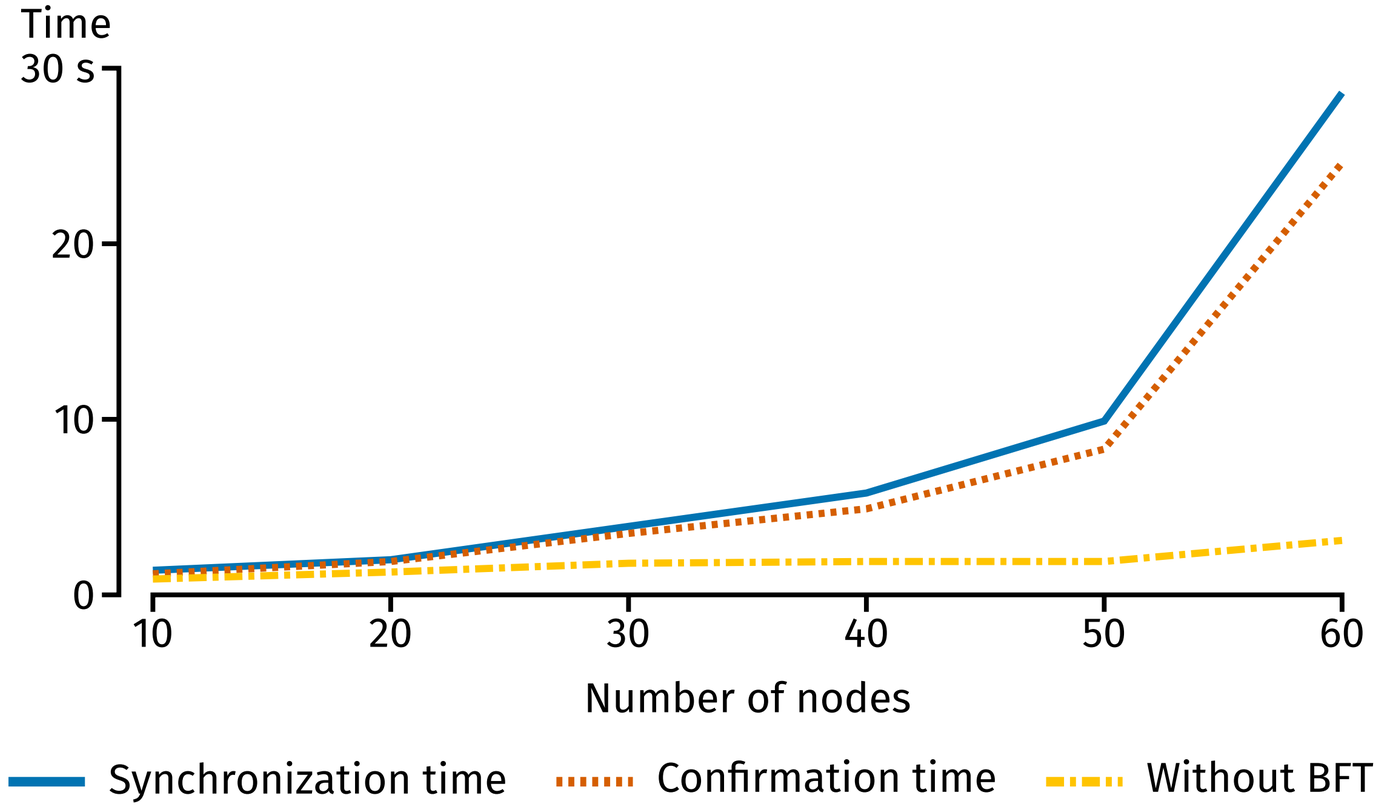
Evaluation

Performance and infrastructure

Zooming out

State-of-the-art and future work

The middleware scales up to 50 nodes



Each node can have 1000s of end-users

eLoyalty: nodes are the merchants

Scale: local farmer’s market or shopping street

eLoan: nodes are the financial institutions

Scale: small country with 50 different banks

The lightweight middleware requires only two small backend components

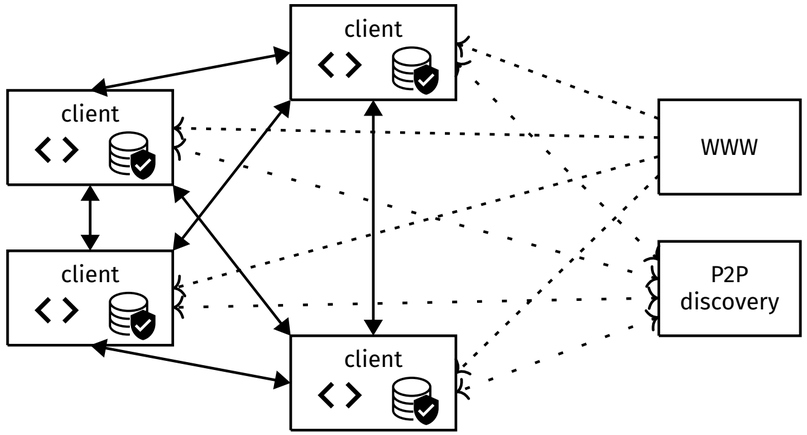
This middleware

Web server, signaling server

Hyperledger Fabric using Hyperledger composer

Web server, peers, orderers, REST-, CouchDB- and CA-servers, membership service

You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients



Lightweight middleware for consensus

Architecture and consensus protocol

Evaluation

Performance and infrastructure

Zooming out

State-of-the-art and future work

State-of-the-art and related work

Peer-to-peer data synchronization systems

Legion, Yjs

PoW blockchains

Bitcoin, Ethereum

Blockchains using BFT consensus

Hyperledger Fabric, HoneyBadger, Algorand, Avalanche

Off-chain protocols

Bitcoin lightning network

The middleware solves the double-spending problem in a specific range of applications

Keep track of transient resources

Not for perpetual resources such as coins

Un-frequent membership changes

Members can leave and join at a slow rate

Up to 50 nodes that are online most of the time

Network can be unstable, some nodes may be offline

Challenges and future work

One ECDSA signature per token per node for consensus

Not scalable to 100-1000 nodes

Look into non-interactive signature aggregation schemes

E.g. BLS

Current P2P network is constructed randomly

Look into structured networks such as fat-tree overlay

You Don’t Need a Ledger:

Only keep the current state using signed data-structures

Lightweight Decentralized Consensus

Simple voting for asynchronous BFT between the clients

Between Mobile Web Clients

Running in a web browser on a mobile device

You Don’t Need a Ledger: Lightweight Decentralized Consensus Between Mobile Web Clients

SERIAL ’19

Kristof Jannes, Bert Lagaisse, Wouter Joosen
kristof.jannes@cs.kuleuven.be
9 December 2019

KU LEUVEN DistriNet