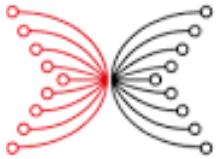
12th Conference on Security and Cryptography for Networks (SCN2020 - Virtual)

Account Management in Proof of Stake Ledgers



Dimitris Karakostas

THE UNIVERSITY of EDINBURGH

Aggelos Kiayias

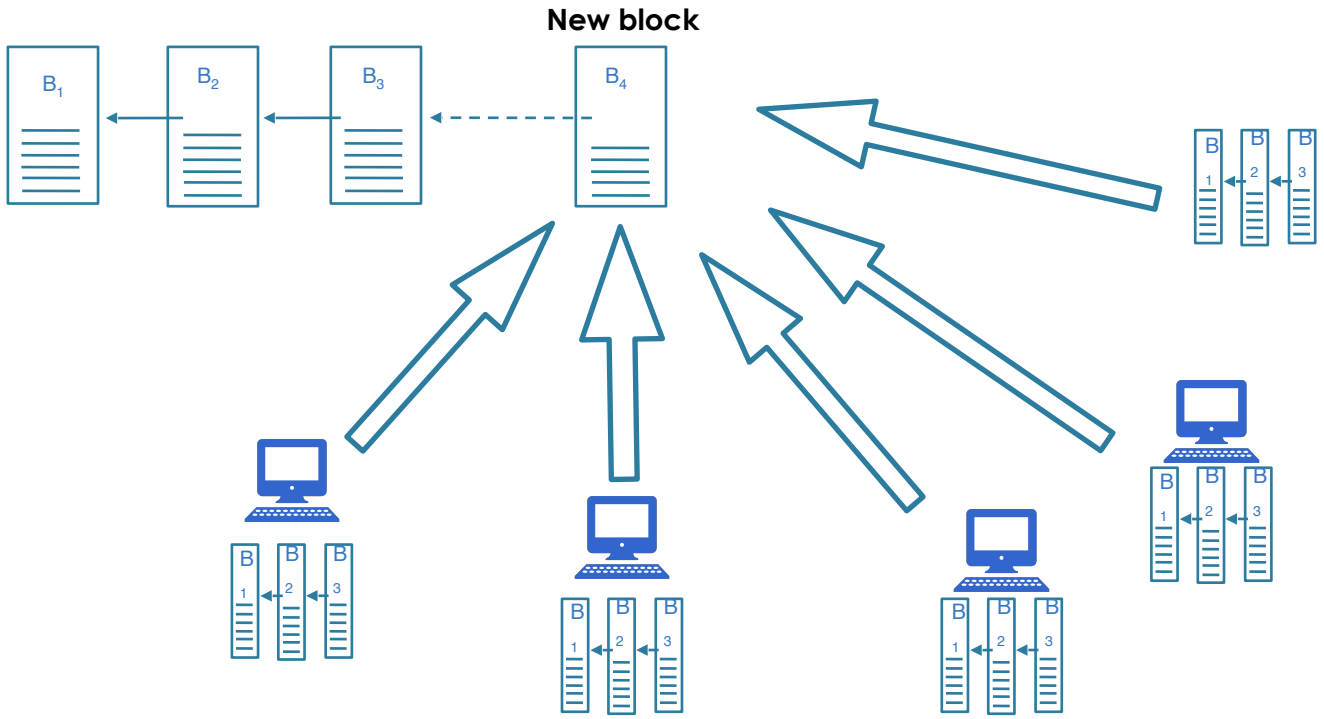
Mario Larangeira

Overview

- Review:
  - Proof-of-Work (PoW) vs Proof-of-Stake (PoS)
  - Motivation/Goal

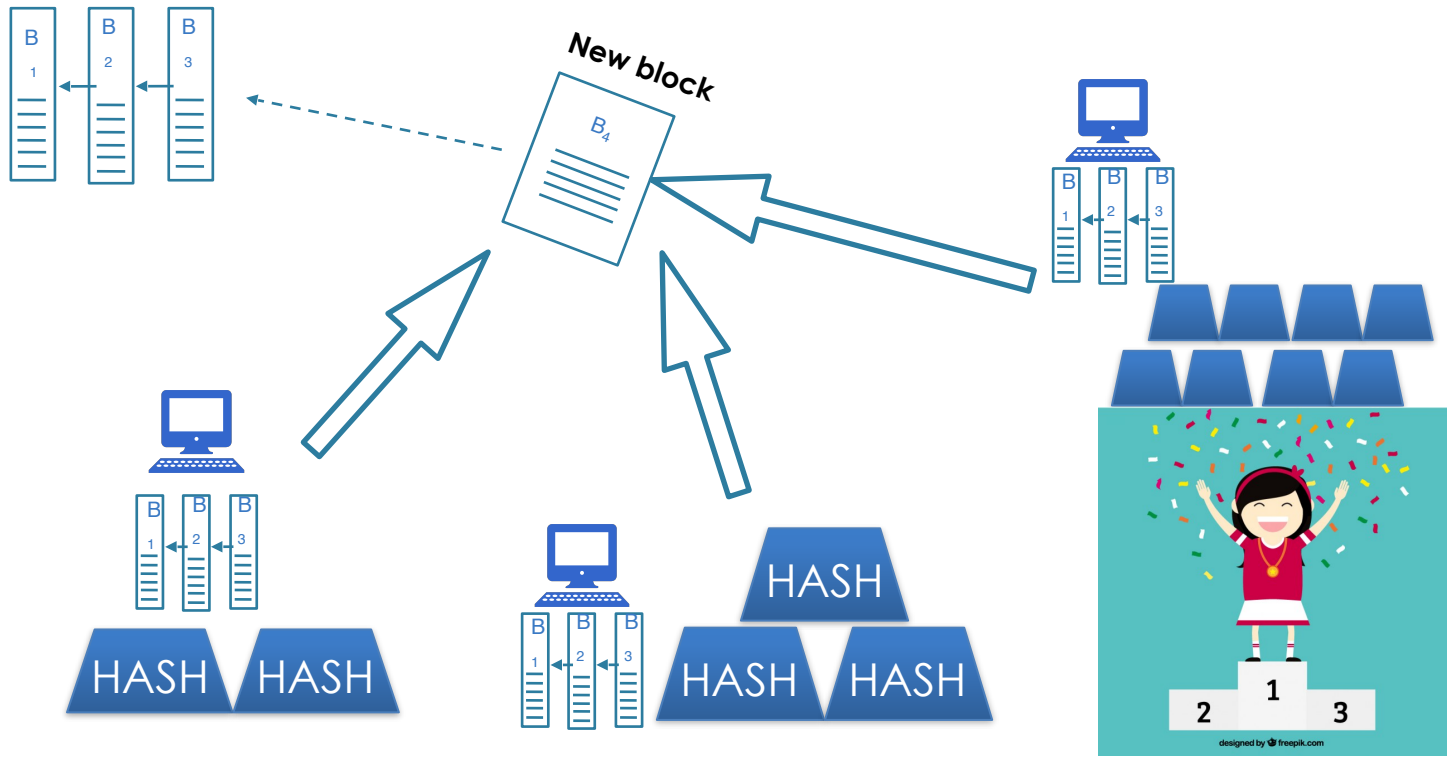
- Our Contributions:
  - Desiderata and Formal Definition of a PoS wallet
  - Malleability attacks
  - Wallet modes

PoW = Miners' "Race"



Miners continuously solve puzzles

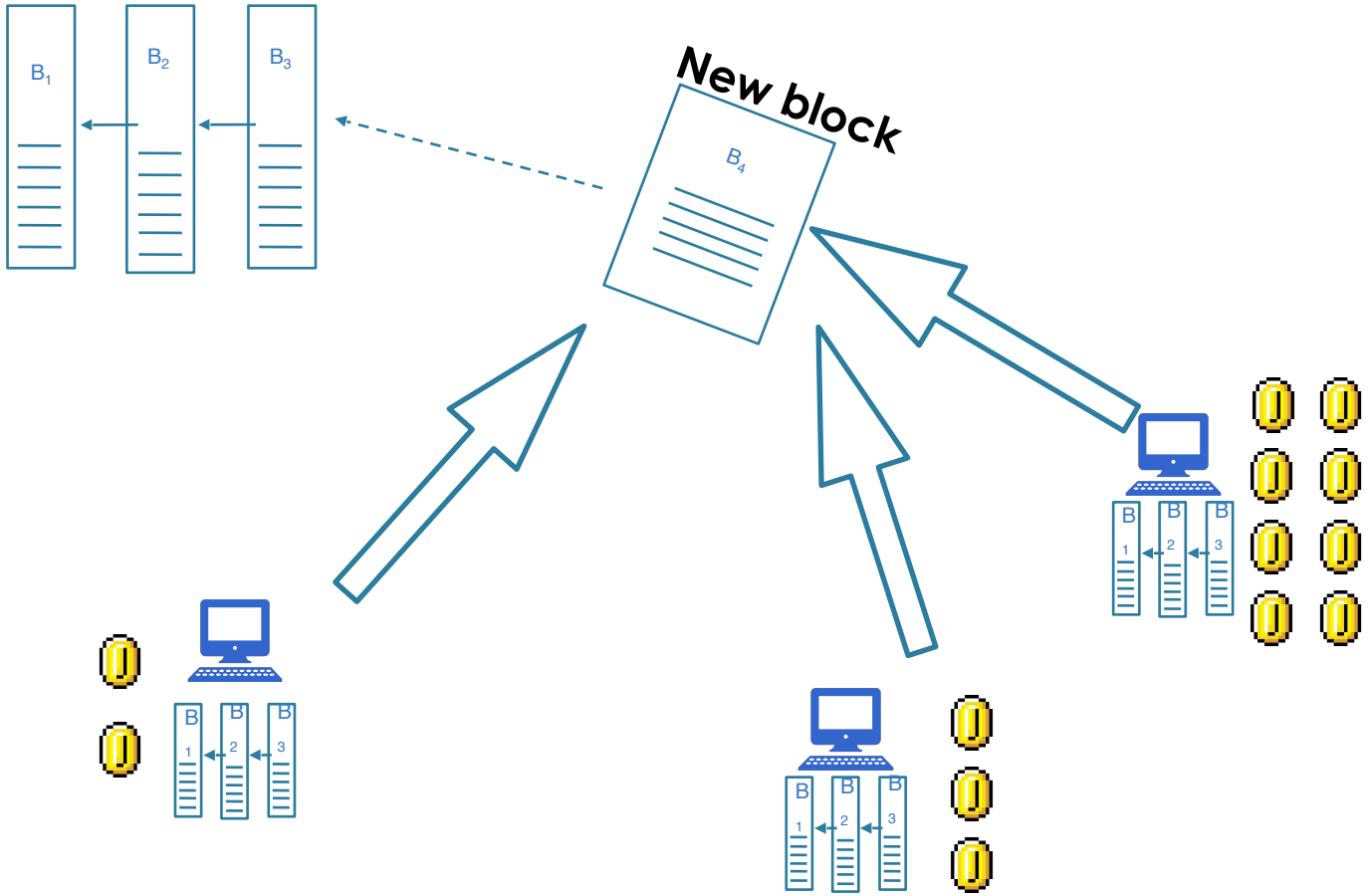
Hashing Power



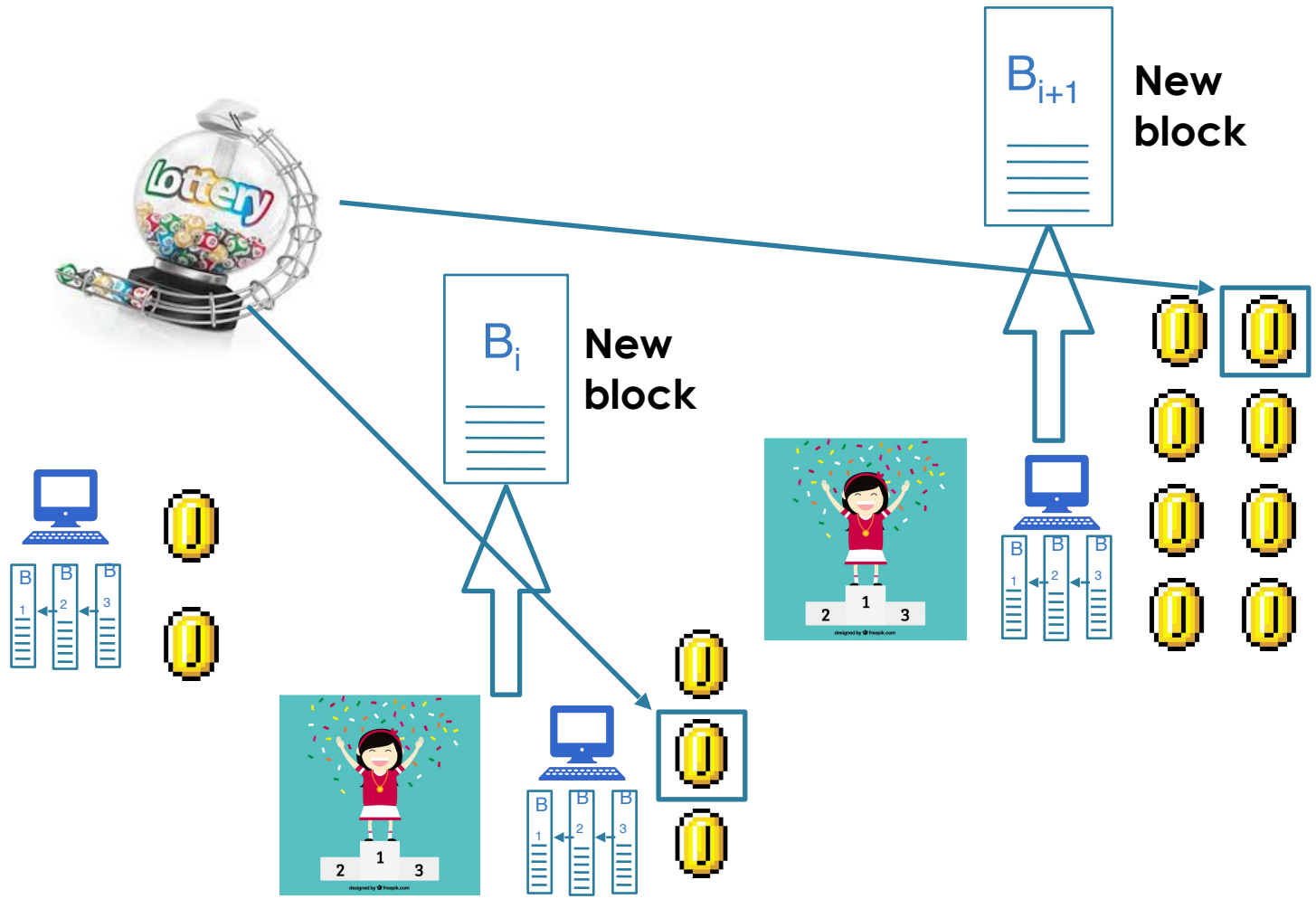
Review

Proportional to the hash power

Stake “Race”

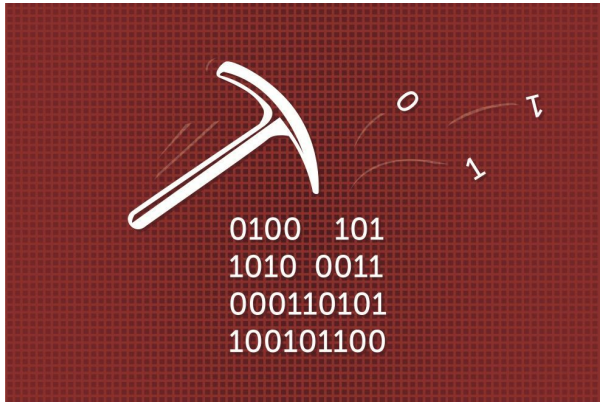


PoS: Stake Lottery



PoW vs. PoS

- **PoW**
  - Only miners participate
  - Miners always online



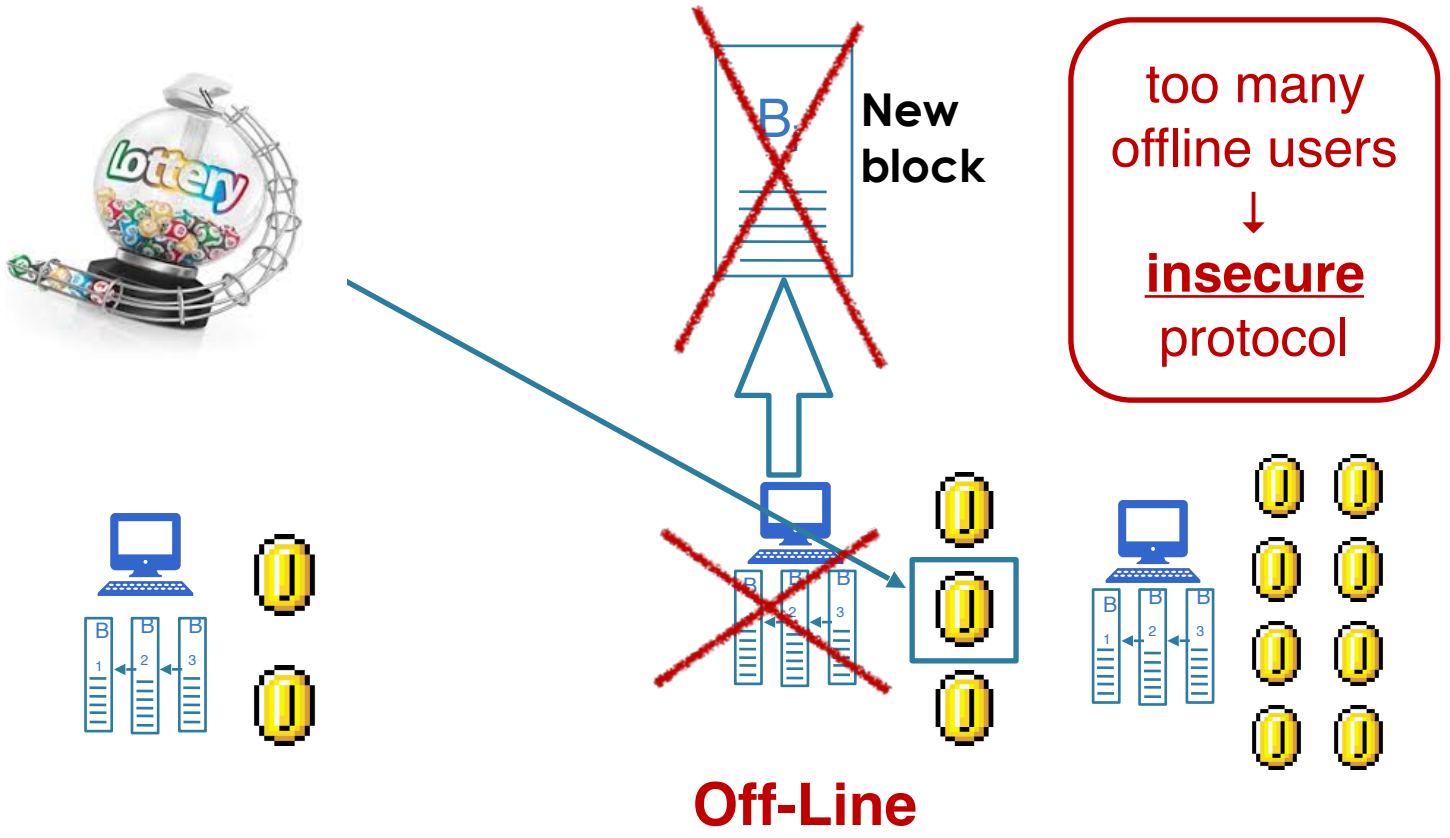
Hashes, hashes, hashes…

- **PoS**
  - All players (potentially) participate
  - All players online for security
  - No hashing “race”
  - Exchanges may keep assets and their stake

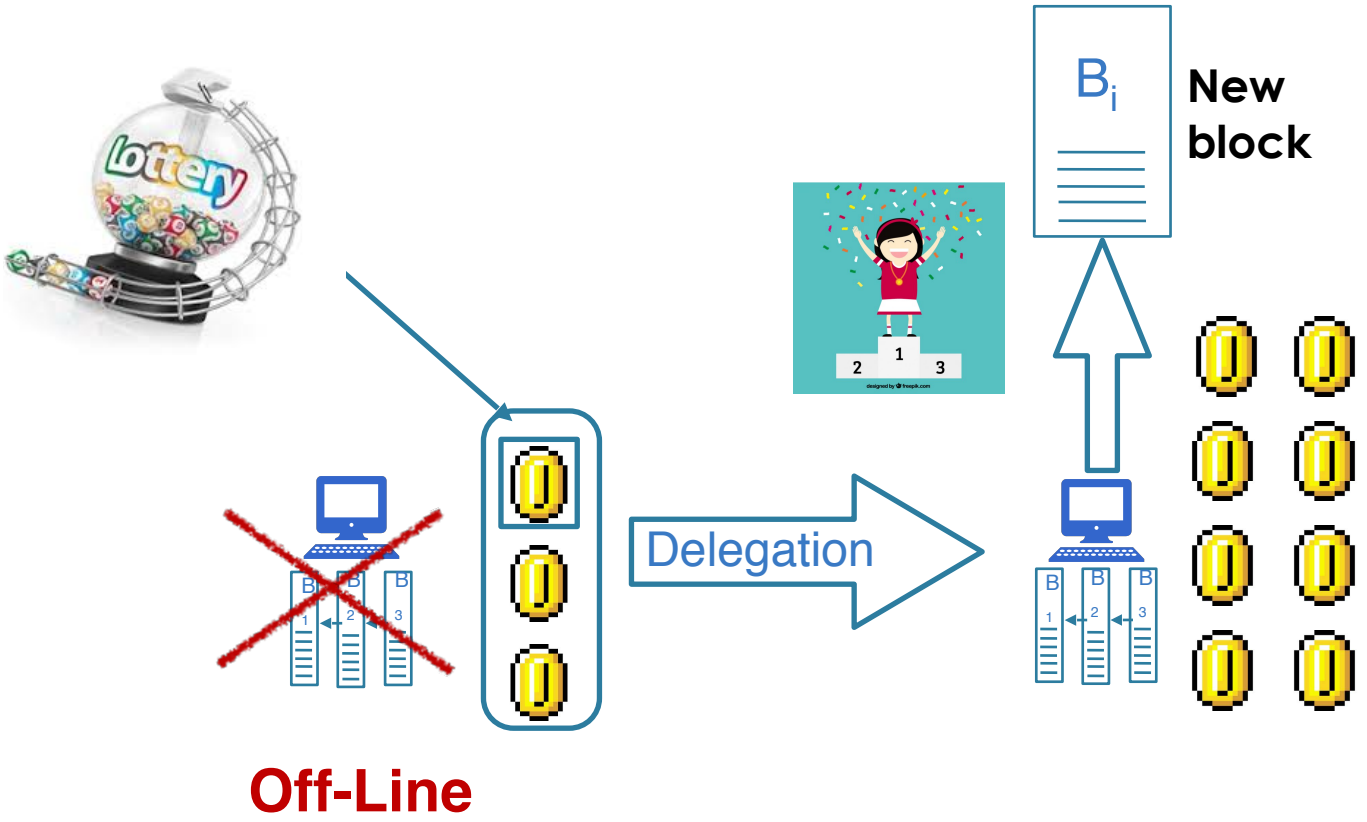
Issues

- Every player potentially joins the protocol
- All players need to be online
  - Unrealistic
- Exchanges: keep funds but do not own them
- Technical challenges:
  - embed information on addresses
  - formally models of accounts

Offline Users



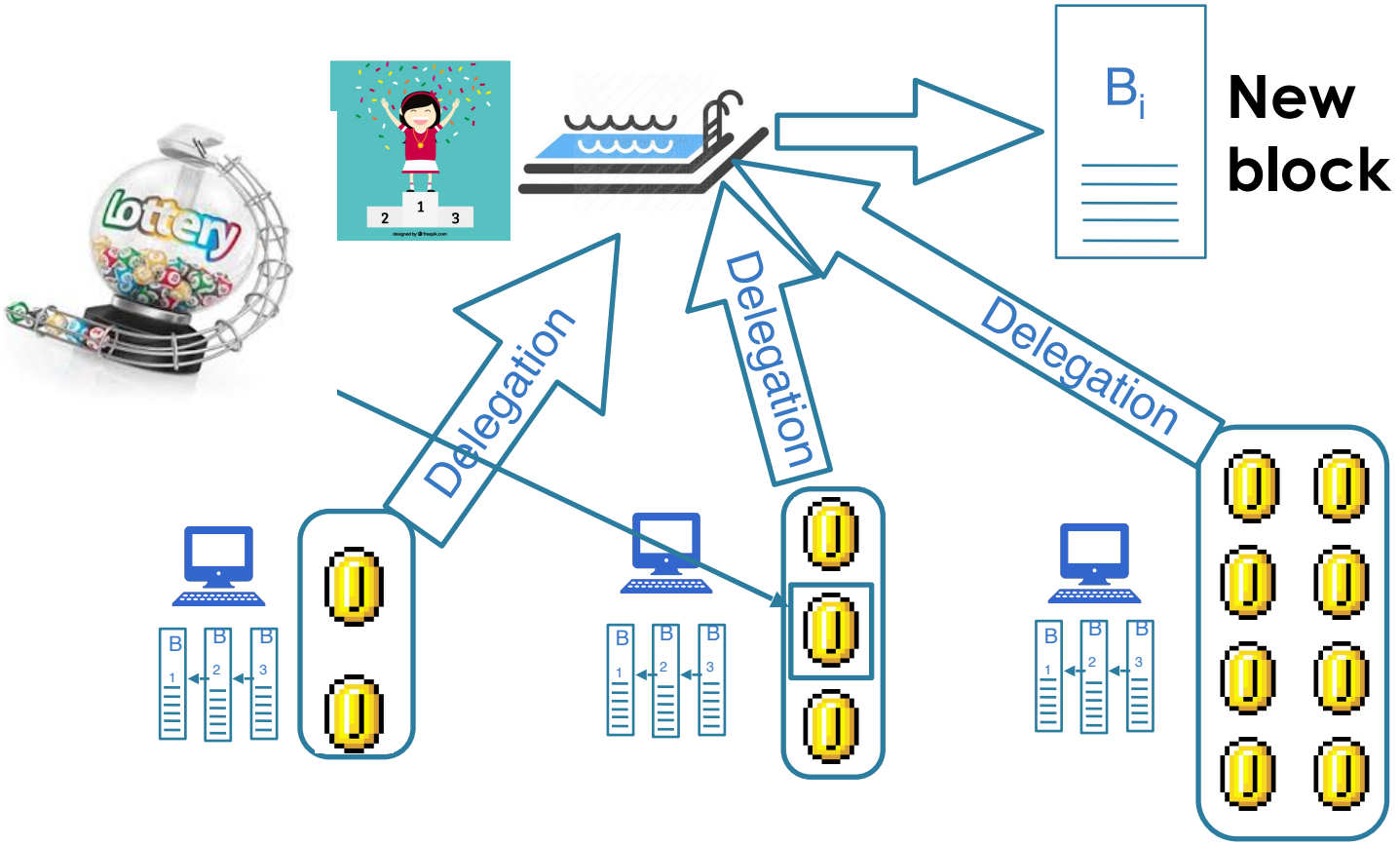
Intuition: Delegation



Delegation

Solution for off-line users: delegation of stake

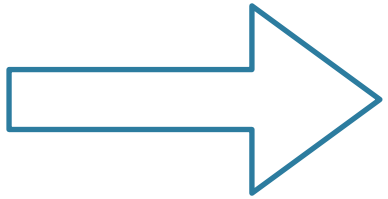
Stake Pools


lottery
2
1
3
B_i
New block
Delegation
Delegation
Delegation
B
1
2
3

Goal

Issues

- Every player (potentially) joins the protocol
- All players need to be online
  - unrealistic
- Exchanges keep funds they do not own
- Technical challenges:
  - embed information on addresses
  - formally model accounts



No framework addresses these and other issues for PoS systems

Our goal:
A PoS wallet Framework

Our Contributions

Our Contributions

- Desiderata of a PoS system
- Formal treatment: accounts/addresses/attributes
- Ideal functionality/Security Definition
- Generic protocol:
  - Two concrete constructions for the protocol
  - Wallet operation modes

Desiderata Overview

- Address and Attributes
  - uniqueness/non-malleable
  - identification (from ledger)
  - multiple types of address (exchanges, stake pools, etc)

- Basic Operations
  - account master (hierarchical keys)
  - staking and payment separation (two keys)

- Delegation
  - cost effective
  - publicly verifiable

Core-Wallet Functionality

- Ideal Functionality/Security Definition $\mathcal{F}_{core-wallet}$
  - based on Canetti's Signature Functionality
  - offers basic actions for PoS
  - receives a malleability predicative parameter

- Malleable predicative $\mathcal{M}$
  - configures $\mathcal{F}_{core-wallet}$
  - different predicate, different level of malleability

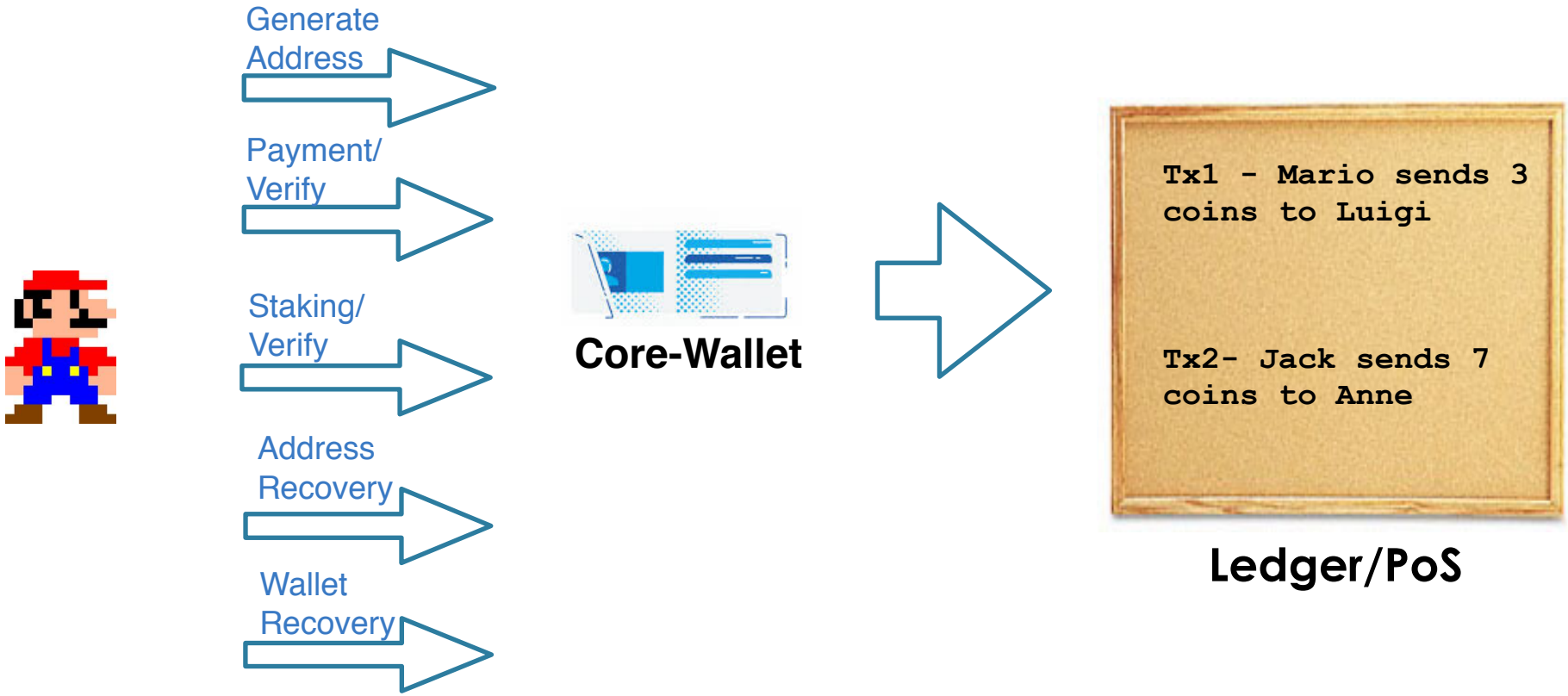
Malleability Scenarios

- **Network adversary**: relies only on the information on the ledger
- **Targeting adversary**: in addition to the ledger, access extra attributes of the target user
- **Self-verification**: a wallet can identify the forgery on its on set of addresses
- **Cross-verification**: a wallet can recognize forgery on other wallets' address

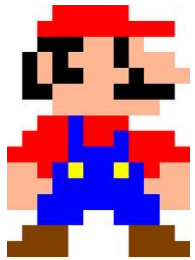
Malleability Levels

Malleability Level	Network protection	Targeting protection	Self-verification	Cross-verification	Address length (bytes)
1. Full	✗	✗	✗	✗	64
2. Ex post	✓	✗	✗	✗	96
3. Sink	✓	✓	✓	✗	128
4. Non-malleable	✓	✓	✓	✓	129

Protocol Overview



Example: Address Generation



Generate Address

Core-Wallet

Wallet addresses are the “account”

1JoLnyfSK2rqT88rRZyyABMjNud6uCNJJP

1L99PHhxKgGKfUPaG7cgR6RwE5TDYx8nV3

1EjUJY9hzTVjDdRBpQExQJW2XUq15KVggk

1PRTAHF7uCwqjyfr58N14Q1WRyH9oBUnuu

1DtA52Tt79pH7sVAWwpa8MucAVcJSdTRJy

Internally

secret and verification keys for payment

secret and verification keys for staking

Index for hierarchical address generation

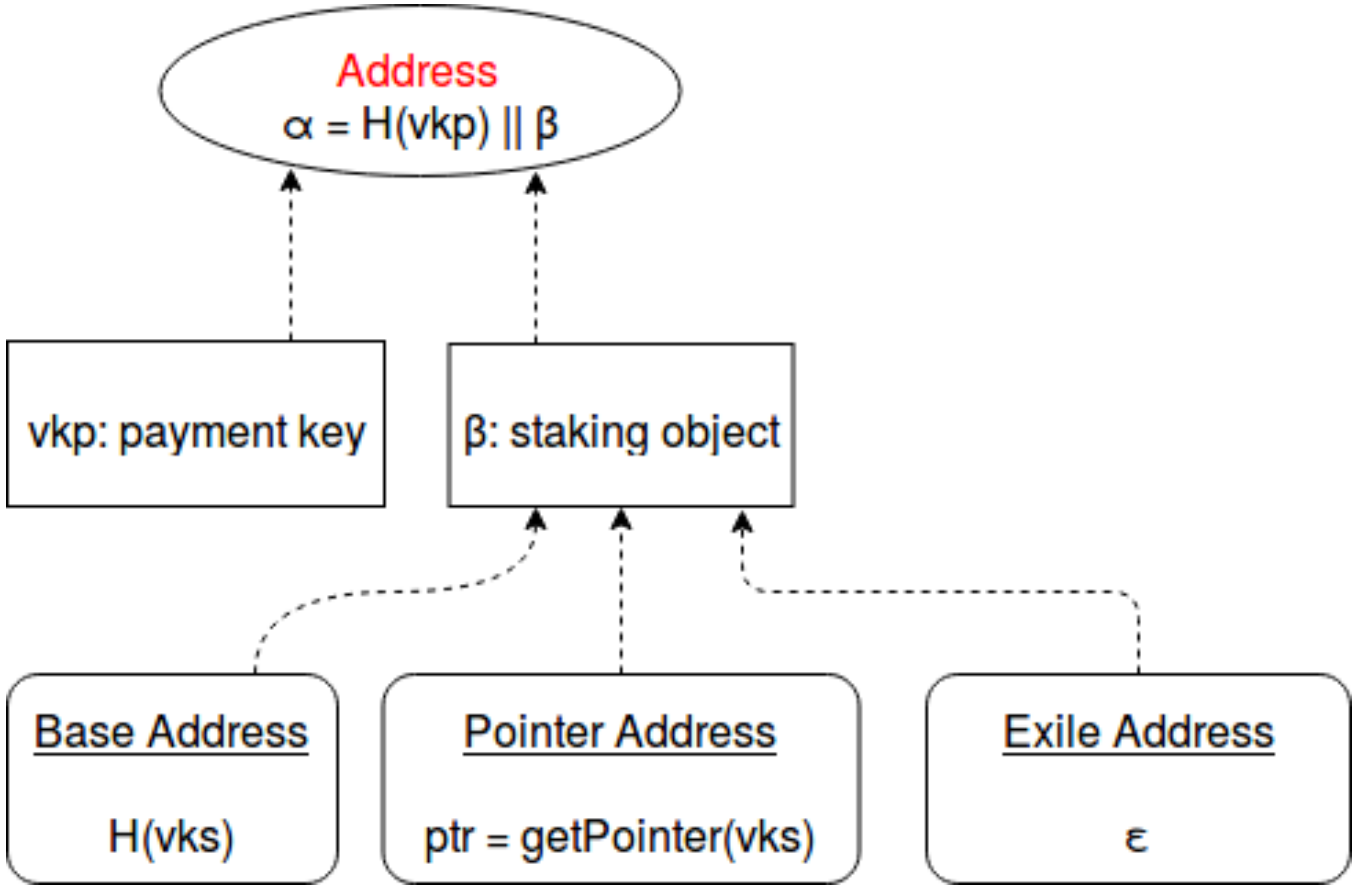
For example: Addressing

Management of addresses = account

Address Types

- Every address linked to two keys: payment and staking
- Hierarchical Address Generation:
  - Child address constructed from master seed (and index)
- Every address directly contains payment key
- *Base* address:
  - Directly contains staking key
  - Useful to create new staking key and bootstrap wallet
- *Pointer* address:
  - Points to an existing staking key (delegation or stake pool certificate)
- *Exile* address:
  - Has no staking key (useful for exchanges)

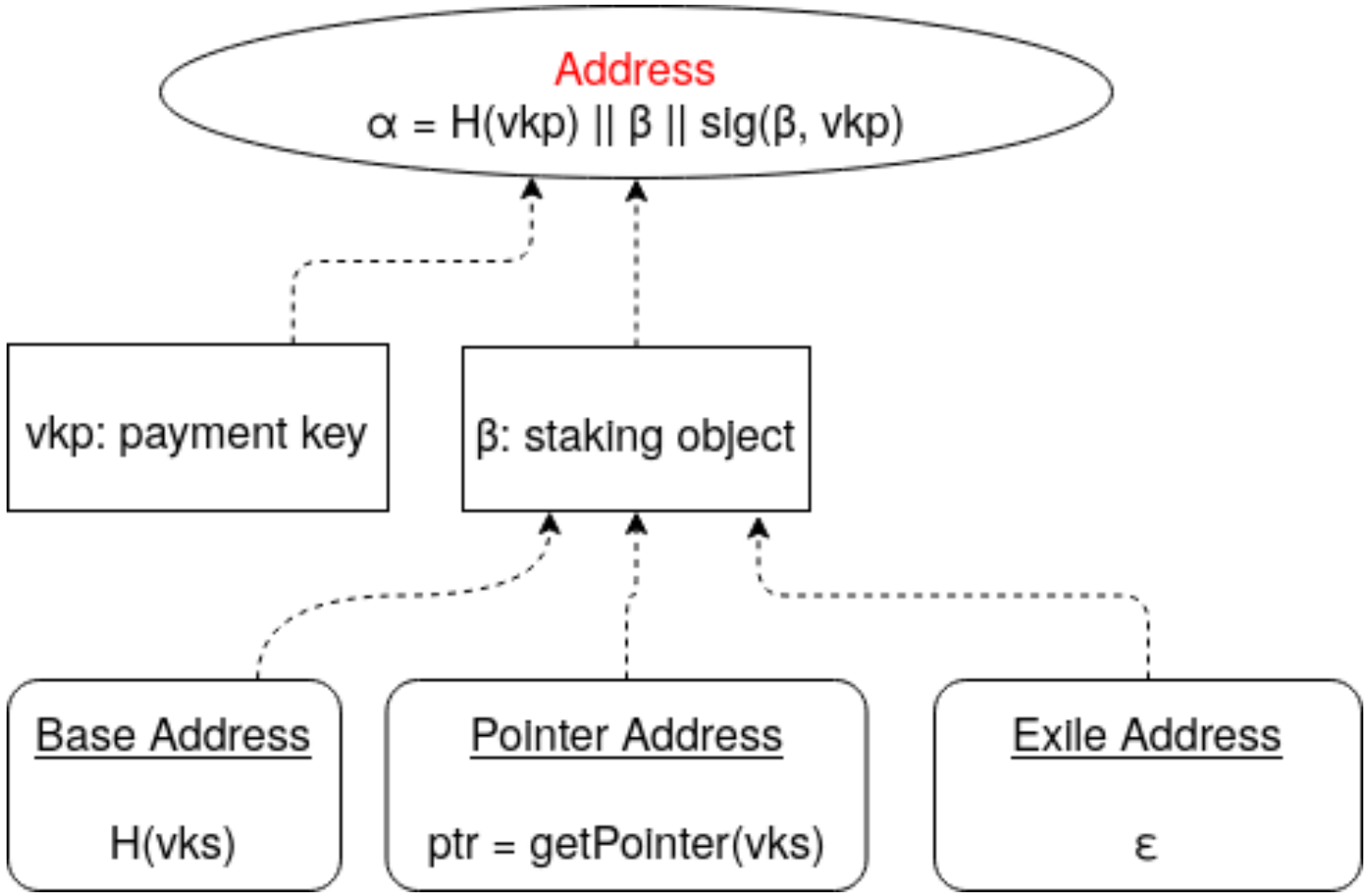
Malleable Address



Addressing

Malleable construction: address string hash, no signature

Non Malleable Address



Address Recovery

- Hierarchical address generation
  - Index chosen (deterministically from fixed sets)
  - Used to construct child payment key from master seed
- Tag:
  - Every address has one
  - Linked to payment key
- Recovery:
  - Wallet recreates payment key and tag for all indexes in set
  - Compare ledger's addresses with recreated tags

Certificate-based Operations

- Stake pool registration certificate:
  - Pool's public key
  - Metadata

- Delegation certificate:
  - Source key
  - Target key (delegate)
  - Metadata:
    - Timestamp when delegation activates
    - Replay protection
  - Signature from source key

Stake-pooled Security

- Sybil attacks:
  - Adversary creates multiple pools
  - Solution: pool leader pledge their stake to the pool
- Replay attacks:
  - An adversary may replay a delegation certificate to change a user's choice
  - Solutions: address whitelist, certificate deadline, UTxO linking

Theorem: The execution of a stake pooled protocol is secure if $\rho \geq \tau$ (τ: the protocol's security parameter, ρ: the percentage of stake controlled by honest stake pools).

Final Remarks

- A guideline for PoS wallets
  - Desiderata
  - Formal model of address generation/recovery and staking
- Security definition of a PoS wallet
  - (Theoretical) Ideal Functionality
  - Concrete, highly-parameterizable protocol
- Address Malleability
  - Hazard against addresses with multiple keys/metadata
- Security analysis in the presence of Stake Pools
- Wallet modes for enhanced unlinkability and safety

Thank You!