# SoK: Privacy-Preserving Computing in the Blockchain Era

Ghada Almashaqbeh (University of Connecticut), **Ravital Solomon*** (Sunscreen)

*ravital@sunscreen.tech

IEEE EuroS&P 2022

# Why care about privacy?

Privacy matters for more than just payments in blockchain!

### > Decentralized exchanges

- Rising popularity due to low exchange fees
- Lack of privacy makes users susceptible to front-running attacks


### > Decentralized autonomous organizations (DAOs)

- More web3 companies set up as DAOs
- We would expect voting to be private

# Privacy is hard!

By privacy, we mean (at minimum) confidentiality = hiding inputs and outputs to programs

Privacy is harder to achieve for general computation than it is for basic payments…

- Advanced cryptographic primitives needed
- Creative techniques to address issues regarding efficiency, concurrency, security
- Any program of the user’s choice (potentially complex operations)
- Application-dependent conditions to be checked

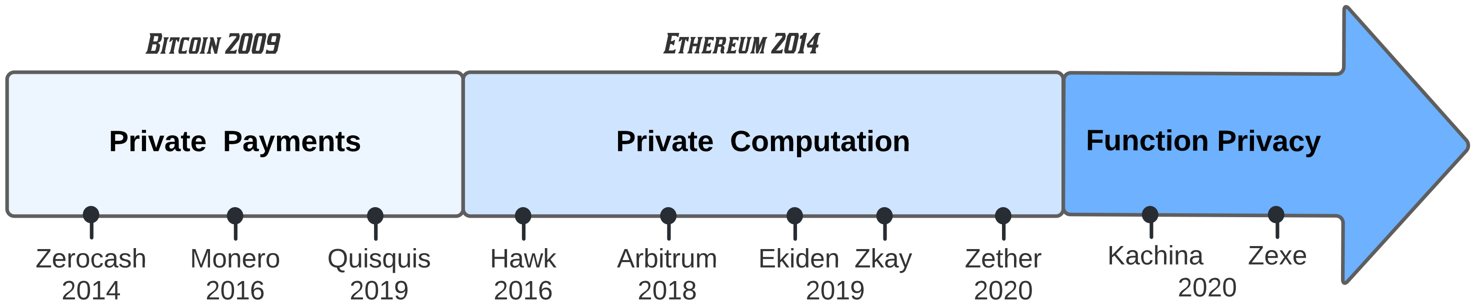
# {Criteria}

System provides *at least* confidentiality

No use-case specific systems

Peer-reviewed works that either have operational projects or introduce new design paradigms

**= 10 systems total**



# Goals of our work

Identify the major design paradigms used to enable confidential computation + explore their key features and limitations

Provide recommendations for system designers (based on their privacy goals, system requirements, envisioned user and miner)

Guide directions for future work

# Roadmap

Building blocks

Zero-knowledge proofs

Design paradigms for private computation

Takeaways

# How to view cryptocurrencies

**Bitcoin-like**: offers limited scripting ability

**Smart contract-enabled**: end users can deploy arbitrary programs

# Cryptographic building blocks for privacy

### Commitments

<Setup, Commit, Open>

- Used to record private data on blockchain
- Guarantees an owner cannot change the original data

### Homomorphic encryption

<KeyGen, Encrypt, Decrypt>

- Allows for performing computation on encrypted data
- Partially vs. fully homomorphic

### Zero knowledge proofs

<Setup, Prove, Verify>

- Allows for proving conditions on hidden inputs have been satisfied
- Focus on succinctness

# Cryptographic building blocks for privacy

### Commitments

<Setup, Commit, Open>

- Used to record private data on blockchain
- Guarantees an owner cannot change the original data

### Homomorphic encryption

<KeyGen, Encrypt, Decrypt>

- Allows for performing computation on encrypted data
- Partially vs. fully homomorphic

### Zero knowledge proofs

<Setup, Prove, Verify>

- Allows for proving conditions on hidden inputs have been satisfied
- Focus on succinctness

# Why ZKPs are important

**Problem**

Parties often need to prove their hidden inputs have satisfied appropriate conditions for the application

**Solution**

ZKPs are a cryptographic solution to this problem!

- Almost all surveyed works use ZKPs
- Certain features have important consequences for the system at large
- Many privacy-preserving systems designed to be <u>modular</u>

# What features of ZKPs matter?

### Flexibility

- Universality
- Can the same reference string be used to prove any NP statement?

### Security

- Trusted setup process vs. transparent proof system

### Efficiency

- Biggest concern in deployment!
- ZKPs can be one of the largest contributors to transaction size and time

# What features of ZKPs matter?

## Flexibility

- Universality
- Can the same reference string be used to prove any NP statement?

## Security

- Trusted setup process vs. transparent proof system

## Efficiency

- Biggest concern in deployment!
- ZKPs can be one of the largest contributors to transaction size and time

# What does efficiency mean?

Time

- Proof generation time → users need to generate proofs
- Setup time → especially important for non-universal proof systems
- Verification time → miners must verify all proofs in the system

Space

- Ideally small constant-sized proofs → miners need to store these
- "Smallest" proofs often require trusted setups + non-universal proof systems

# Now for the private computing schemes…

The goal of these systems is to provide input/output privacy for arbitrary computation

# Major design paradigms

*Where* is the private computation performed?

*What* enables the computation?

**On chain**

Homomorphic encryption (HE)-based approach

Miner performs the computation

**Off chain**

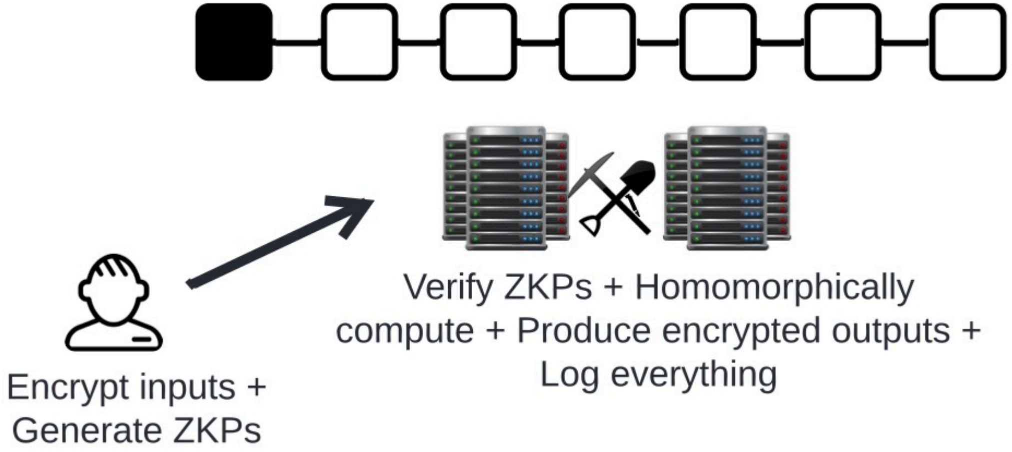
Zero knowledge proof (ZKP)-based approach

User performs the computation

Delegation-based approach

Third party performs the computation

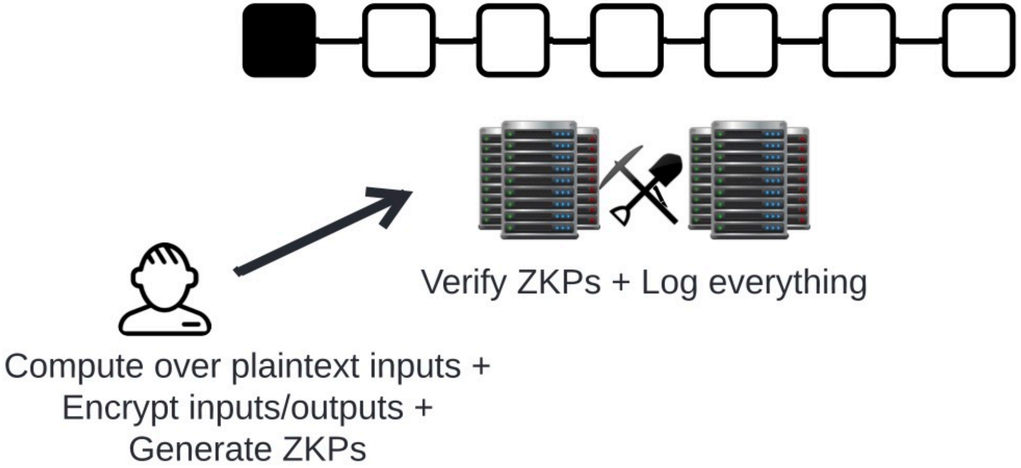
# Homomorphic encryption-based approach




Zether (ElGamal)
smartFHE* (BFV FHE scheme)

| Benefits | Drawbacks |
|---|---|
| “Low” computational overhead for the user | To support arbitrary comp, need FHE |
| | Often results in larger transaction sizes |
| | Often results in longer verification times for miners |

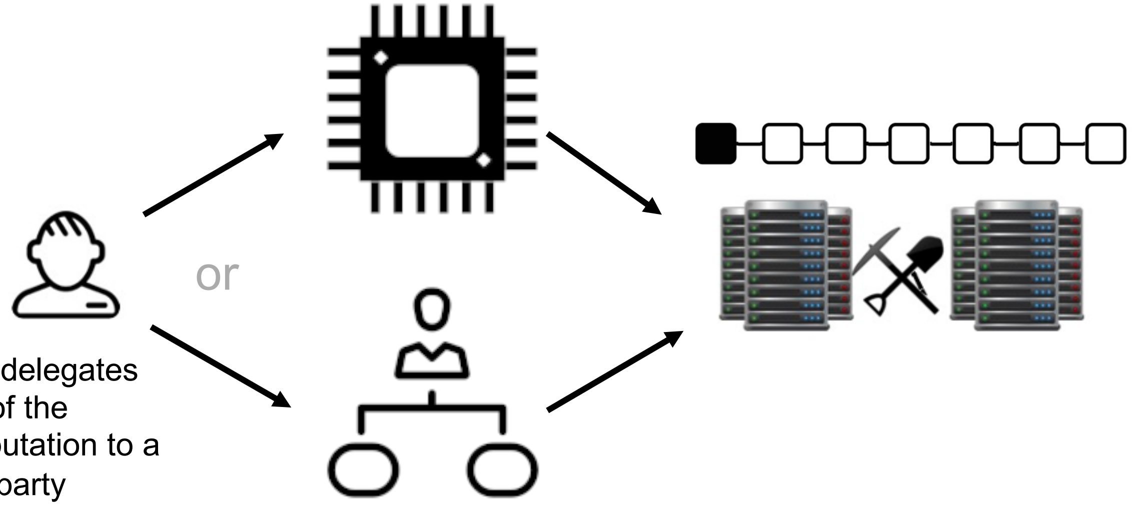
# Zero knowledge proof-based approach




| Benefits | Drawbacks |
| --- | --- |
| “Low” overhead for the miner | Computationally intensive/expensive for user |
| | For non-universal ZKP, need to repeat ZKP setup for each new app |

Zexe (GM17)
Zkay (GM17)
Kachina (N/A)
Hawk* (PHGR13 + Kosba)

# Delegation-based approach




User delegates part of the computation to a third party

| Benefits | Drawbacks |
| --- | --- |
| Fairly low overhead for both user + miner | User potentially compromises on privacy |
| | Must trust third party, be it hardware or managers |

Ekiden (TEEs)
Arbitrum (managers)
Hawk* (manager)

# Which approach to use when?

- Who is your envisioned user?

  → **HE-based approach for lightweight users**

- Is high system throughput critical?

  → **ZKP-based approach (but proof system has tradeoffs as well)**

- Are you willing to compromise on user privacy in exchange for supporting lightweight users + high system throughput?

  → **Delegation-based approach (but may still have latency issues)**

# Where are privacy solutions headed from here?

## Standalone systems

- Building on Ethereum is challenging and expensive for users
- Systems will likely move towards being standalone systems

## Rise of the HE-based approach

- Greater focus on supporting lightweight users in industry
- We expect HE-based approach to gain more traction

## Winner takes all?

- Each approach excels in different situations
- We expect all three approaches to develop and exist in parallel

# Questions?

ePrint: https://eprint.iacr.org/2021/727