zk-DEAP

A Cryptography Protocol

Jude Ramanan
Professors Apthorpe and Haldeman
COSC 482

Motivating Problem

5 friends are deciding if they want to go to the beach. They will all vote. They need a ⅗ majority to go to the beach.

Commitments:
Individual votes

Aggregate:
Sum of the Votes

Red Problem

Jack does not want to go to the beach, but he is afraid his friends will judge him.

Red Solution
Homomorphic Encryption

Instead of voting yes or no, vote an encrypted value like:

9cc4c48b243092ccba9112869e94580c08a8ba0c19daf6197796032d61cdada7f9fa
5efe44bd3361fe8ec99e8d07e7a1099ace241d738cec98fbef507e28e963

No one knows if this means yes or no!

“Homomorphic” means you can sum it with other encrypted votes, and you will wind up with an encrypted aggregate

Green Problem

No one trusts each other enough to decide who will tally the votes (decrypt the aggregate)

(Also for homomorphic encryption you need to set up cryptographic keys, and we don’t trust one person to do that either)

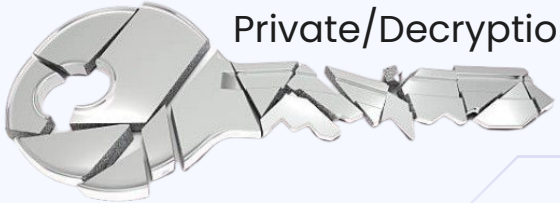
Green Solution
Distributed Cryptography

Traditional asymmetric encryption (RSA) only works for two people

1: **Distributed Key Generation**: They work together to create a shared public key for encryption. Each person then gets a “share” or piece of the private key.

2: **Threshold Cryptography**: No one has the entire private key, so no one can decrypt the aggregate alone, they must work together!

Public/Encryption Key



Private/Decryption Key

Blue Problem

Sarah really wants to go to the beach.

Sarah realizes that because the ballot looks like this:

9cc4c48b243092ccba9112869e94580c08a8ba0c19d

She can submit the value 3 instead of 1 (yes), and no one could tell.

This guarantees they go to the beach!

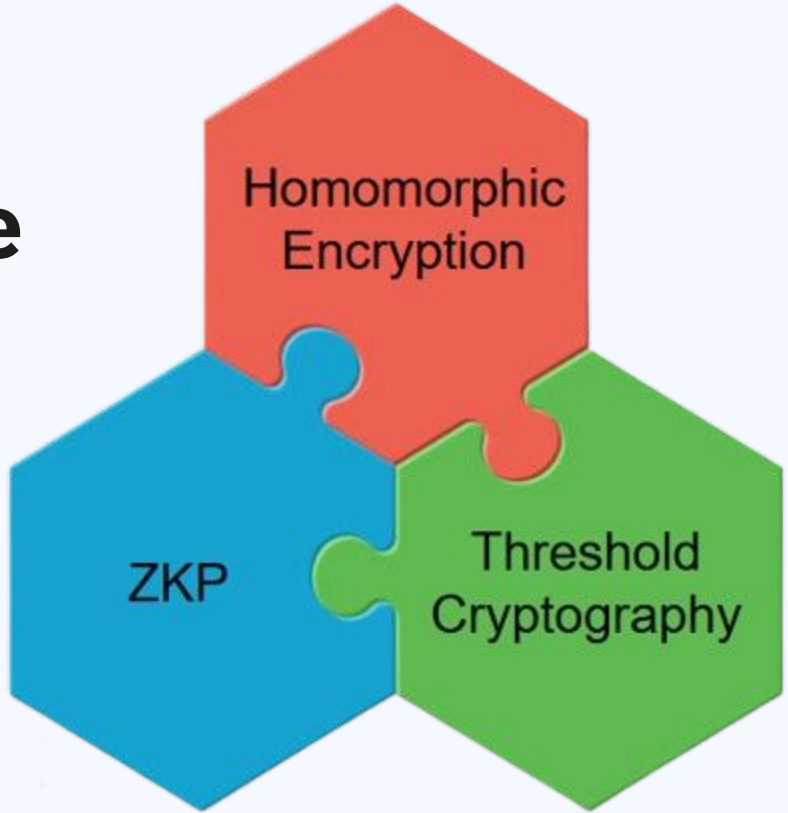
Blue Solution
Zero Knowledge Proofs (ZKPs)

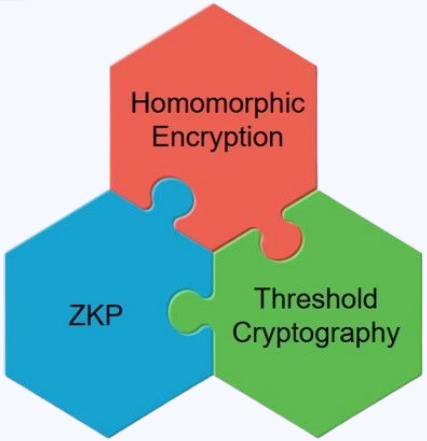
In addition to the encrypted vote, everyone also submits a zero knowledge proof, which verifies:

The value of the encrypted vote is a 0 or 1 (it is a valid vote)

Key insight: The “zero-knowledge” means all we know is that the condition is held. We still do not know what the vote is.

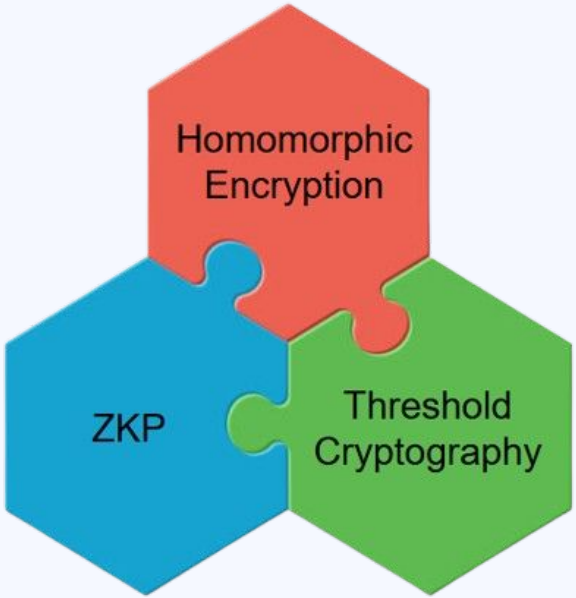
The Puzzle



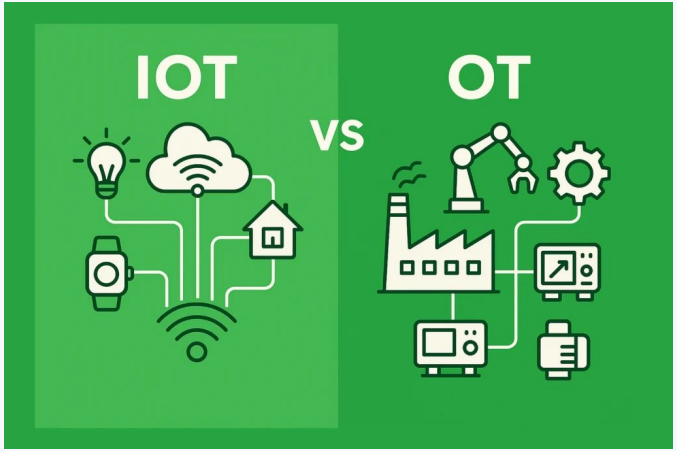


Authors	Short Name	Year	HE	ZKP	TC	Context	Puzzl	Pieces (from Figure 1)
Loukil et al. [3]	PrivDA	2021	✓	X	X	IoT/Blockchain		1/3
Yang et al. [7]	n/a	2023	✓	✓	X	Smart-Grid		2/3
T. Wang et al. [9]	PPVDA	2024	✓	X	✓	IoV		2/3
de Oliveira et al. [5]	DC-nets FL	2024	X	X	✓	FL		1/3
X. Wang et al. [8]	TrustDLF	2024	X	✓	X	FL		1/3
Zhou et al. [10]	GVSA	2025	X	~	✓	FL		1.5/3
Our Paper	zk-DEAP	2026	✓	✓	✓	Generic		3/3 ★

zk-DEAP: Zero-Knowledge Distributed Encrypted Aggregation Protocol

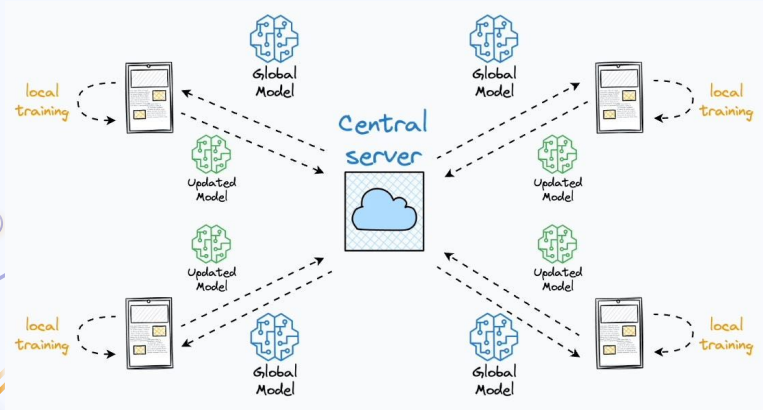


IoT/OT



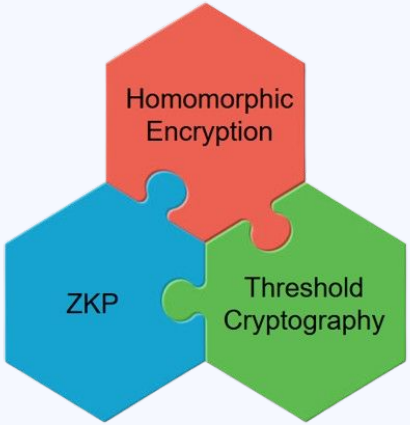
Voting Security

Federated Learning



Blockchains

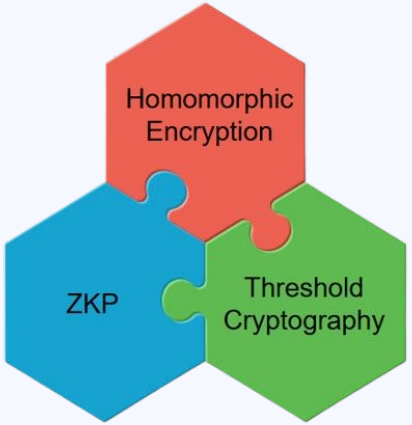
Methods 1: Cryptographic Theory



+

1) Fit the pieces together
2) Glue the puzzle to stop substituting/changing pieces

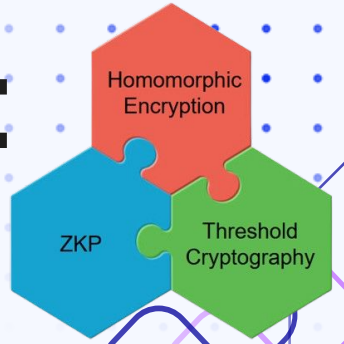
Lifted ElGamal



1) Bulletproofs
2) zk-SNARK (Halo2)
3) zk-STARK

Flexible Round-Optimized Schnorr Threshold (FROST)

Protocol Flowchart



Layer 1: FROST DKG

Layer 2: ZKP and ElGamal

Layer 3: FROST Lagrange

Layer 4: BSGS

DKG Setup

Devices Generate and Broadcast Proofs for their States

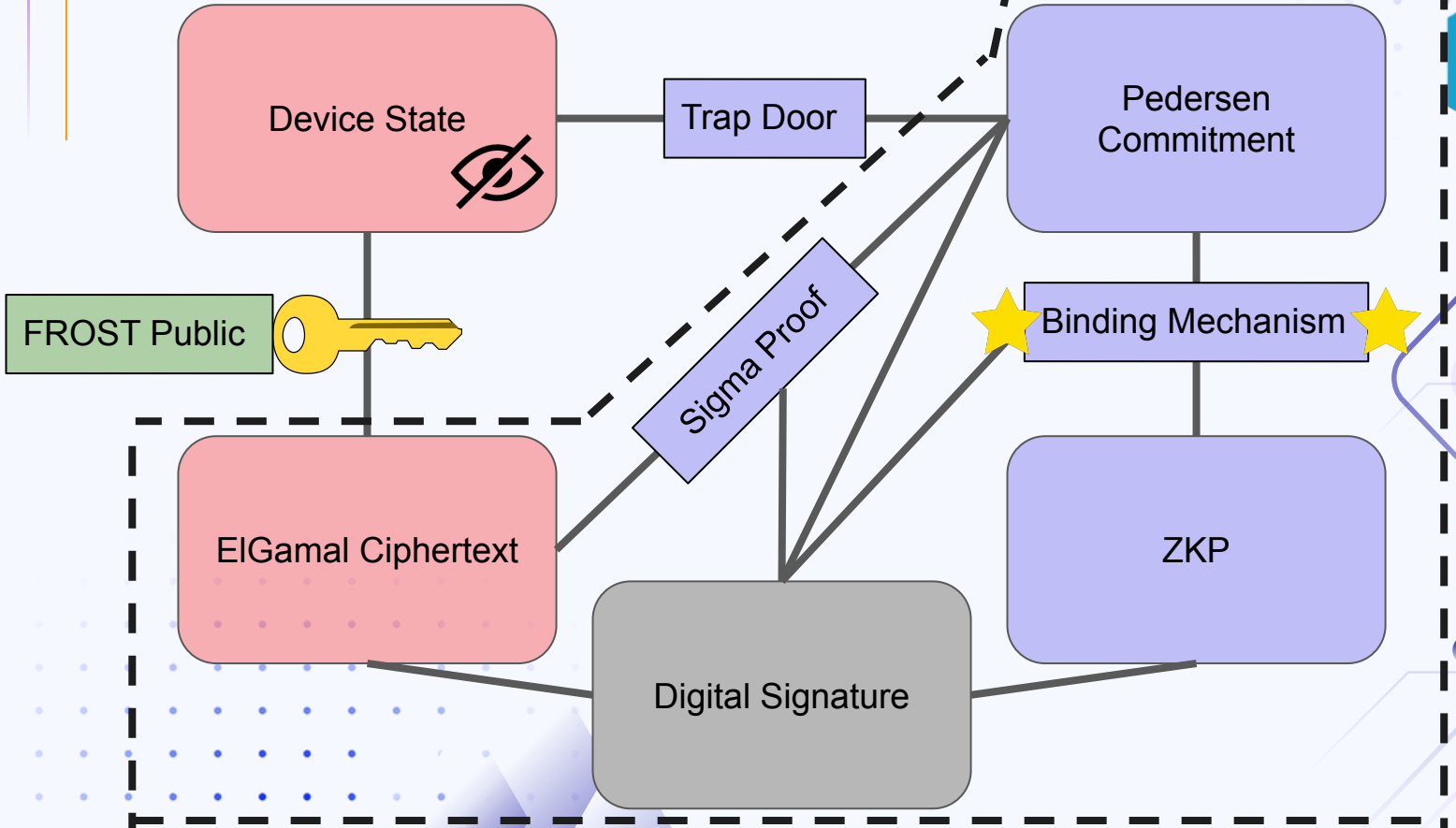
Devices Recieve, Verify, and Strip the Ciphertext to Aggregate

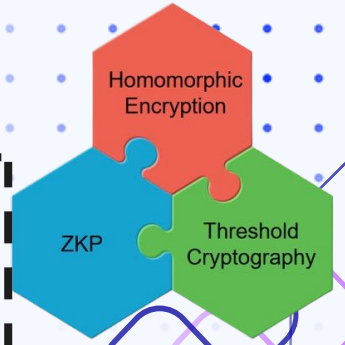
Devices Generate and Broadcast Partial Decryptions

Devices Receive and Combine Partial Decryptions

Devices Extract Aggregate from Plaintext

Proof Structure





Binding Mechanism

Device State

Pedersen Commitment

$$\varepsilon_i = \phi(C(s_i, r_i))$$

$$\text{Pseudo-Hash: } \mathcal{H}_{pseudo} = s_i^5 + \eta_i^5 + \varepsilon_i^5 + s_i \cdot \eta_i + s_i \cdot \varepsilon_i \pmod{|\mathbb{F}|}$$

$$\mathcal{T}[3,i] = \mathcal{T}[0,i]^5 + \mathcal{T}[1,i]^5 + \mathcal{T}[2,i]^5 + \mathcal{T}[0,i] \cdot \mathcal{T}[1,i] + \mathcal{T}[0,i] \cdot \mathcal{T}[2,i]$$

$$\mathcal{T}[3,63] = \zeta_i, \quad \mathcal{T}[1,63] = \eta_i, \quad \mathcal{T}[2,63] = \varepsilon_i$$

$$g^{r_{resp}} = R \cdot (c_1^{(i)})^c$$

$$g^{s_{resp}} \cdot Y^{r_{resp}} = S \cdot (c_2^{(i)})^c$$

$$g^{s_{resp}} \cdot Y^{r_{resp}} = P \cdot C(s_i, r_i)^c$$

Sigma Proof

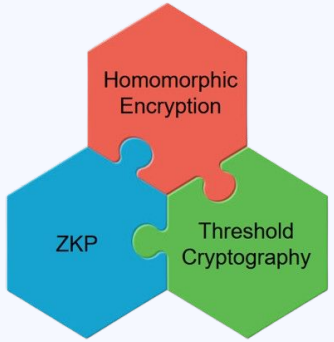
ZKP

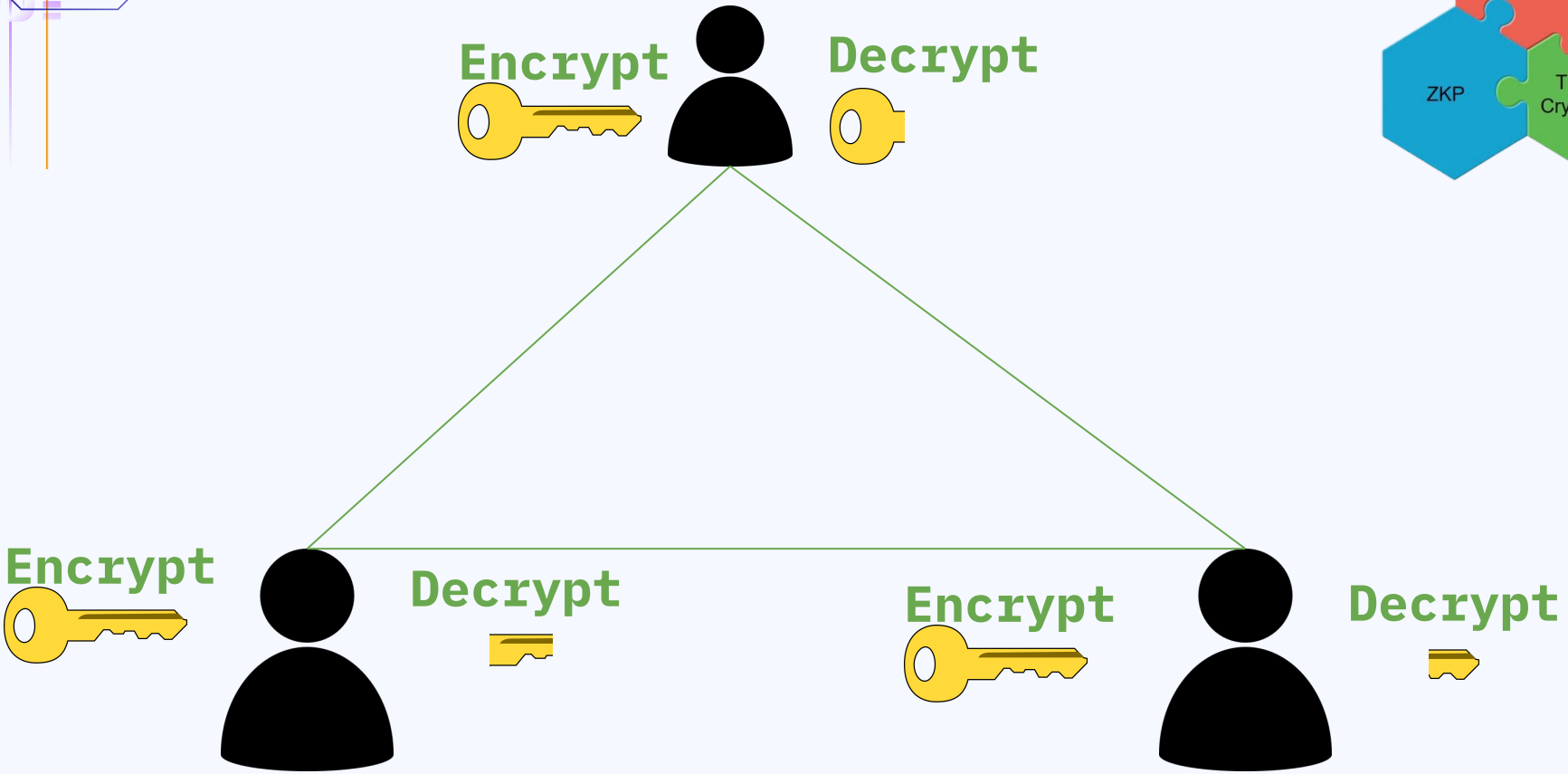
Pedersen Commitment

ElGamal Ciphertext

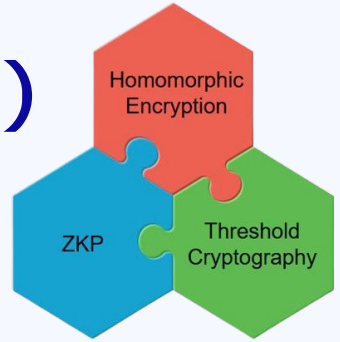
$$\text{Circuit Proof} \xrightarrow{\zeta_i = \mathcal{H}_{pseudo}(s_i, \eta_i, \varepsilon_i)} C(s_i, r_i) \xrightarrow{Schnorr} (c_1^{(i)}, c_2^{(i)})$$

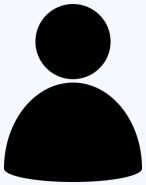
1. Frost DKG (done once)





2a. Proof Broadcast (simplified)

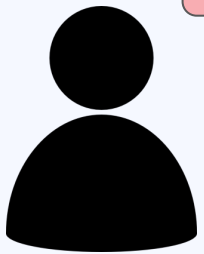


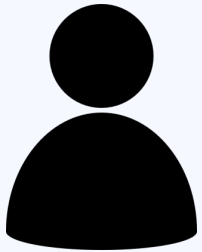


Vote

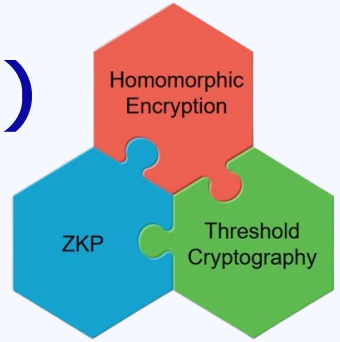
Vote

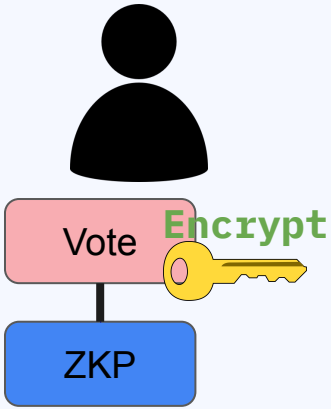
Vote

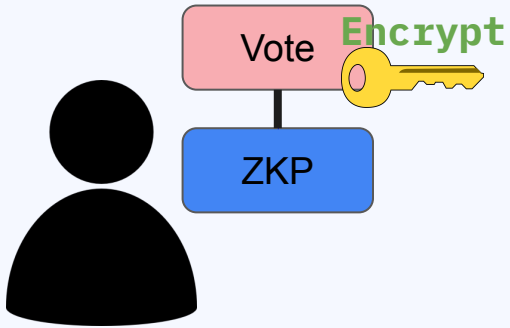


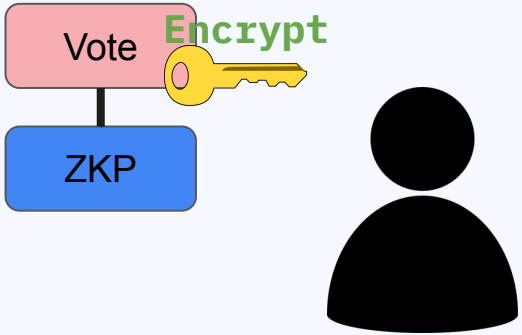


2a. Proof Broadcast (simplified)

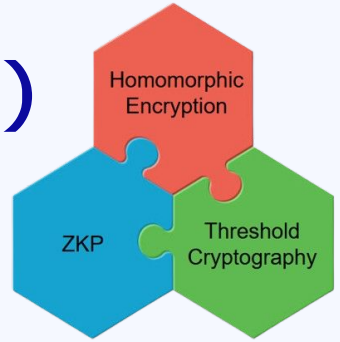


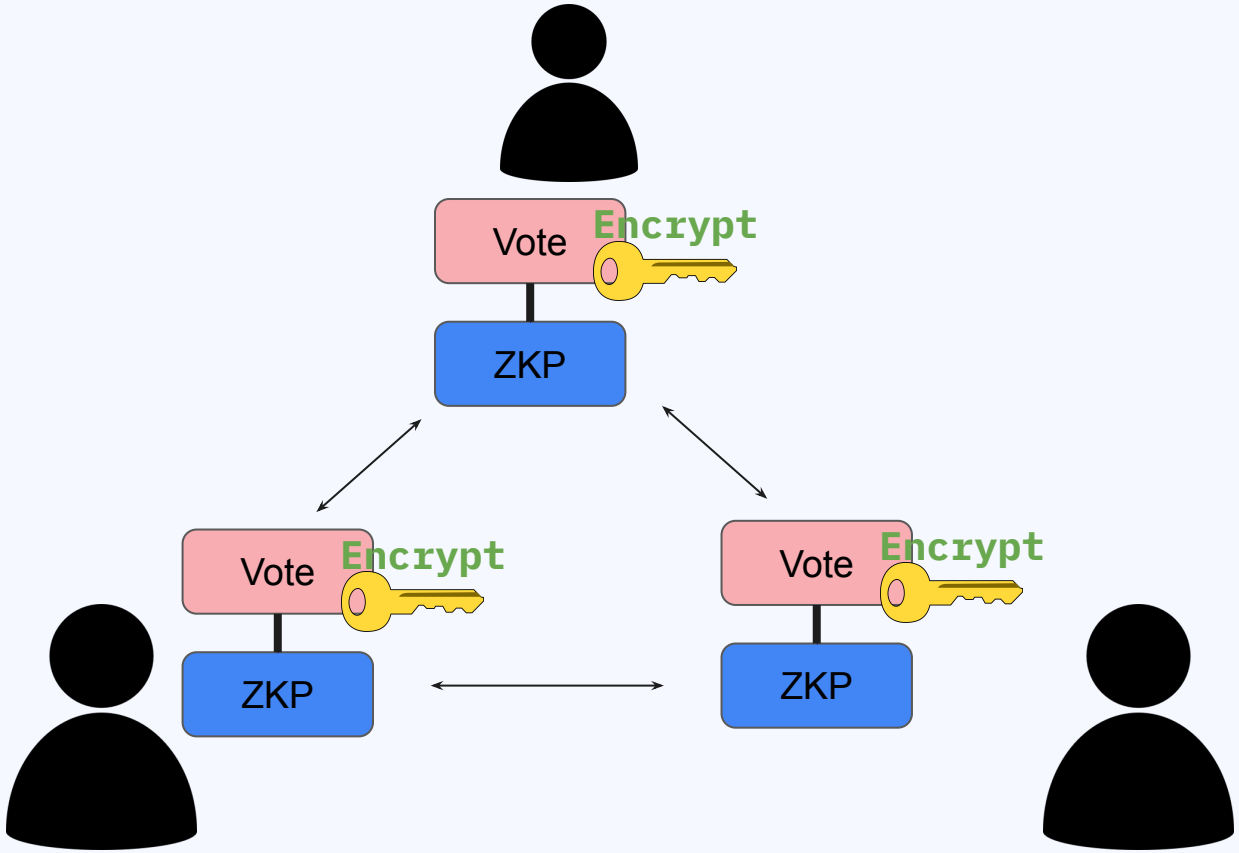




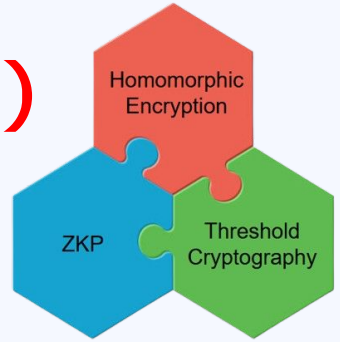


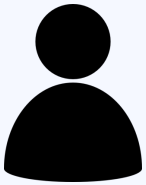
2a. Proof Broadcast (simplified)

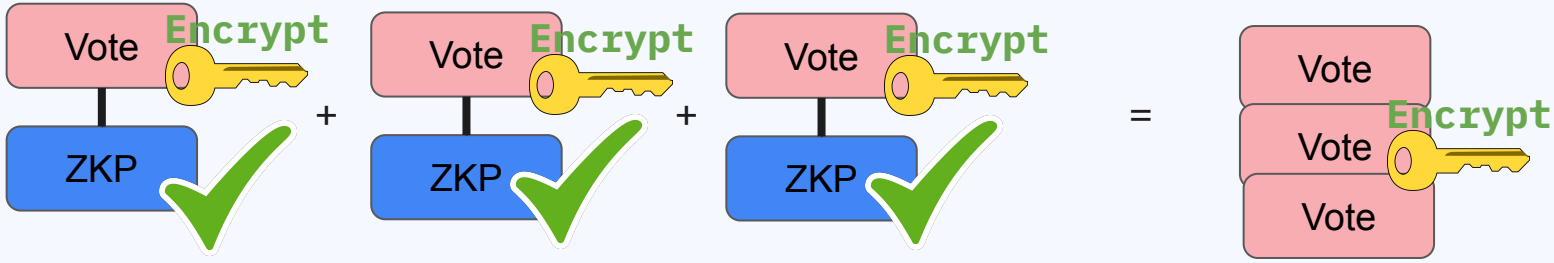


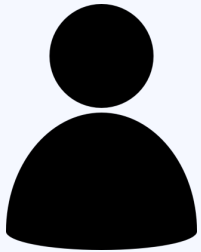


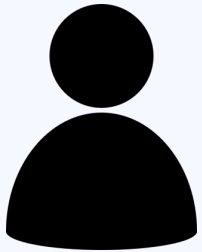
2b. Proof Reception (simplified)



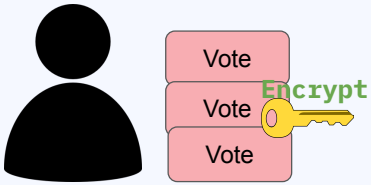


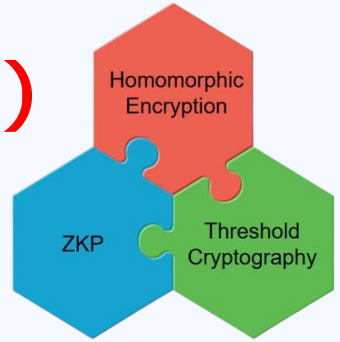


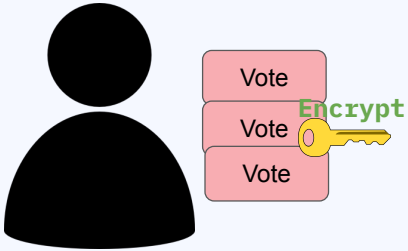


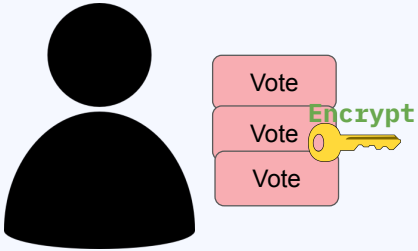


2b. Proof Reception (simplified)

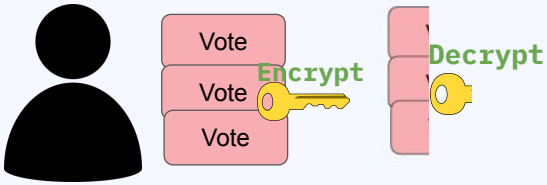


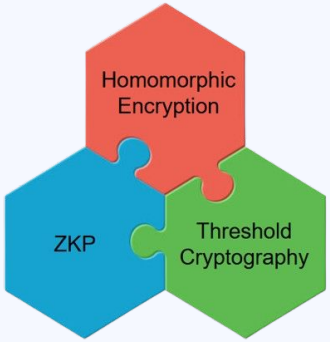


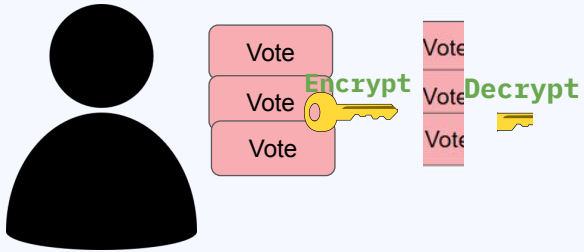


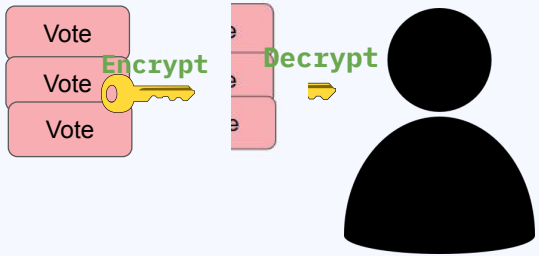


3a. Frost Lagrange

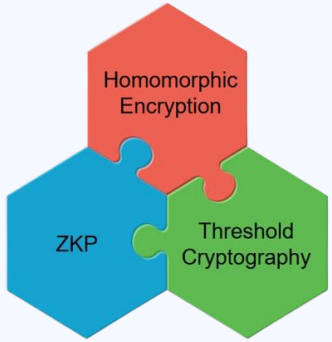


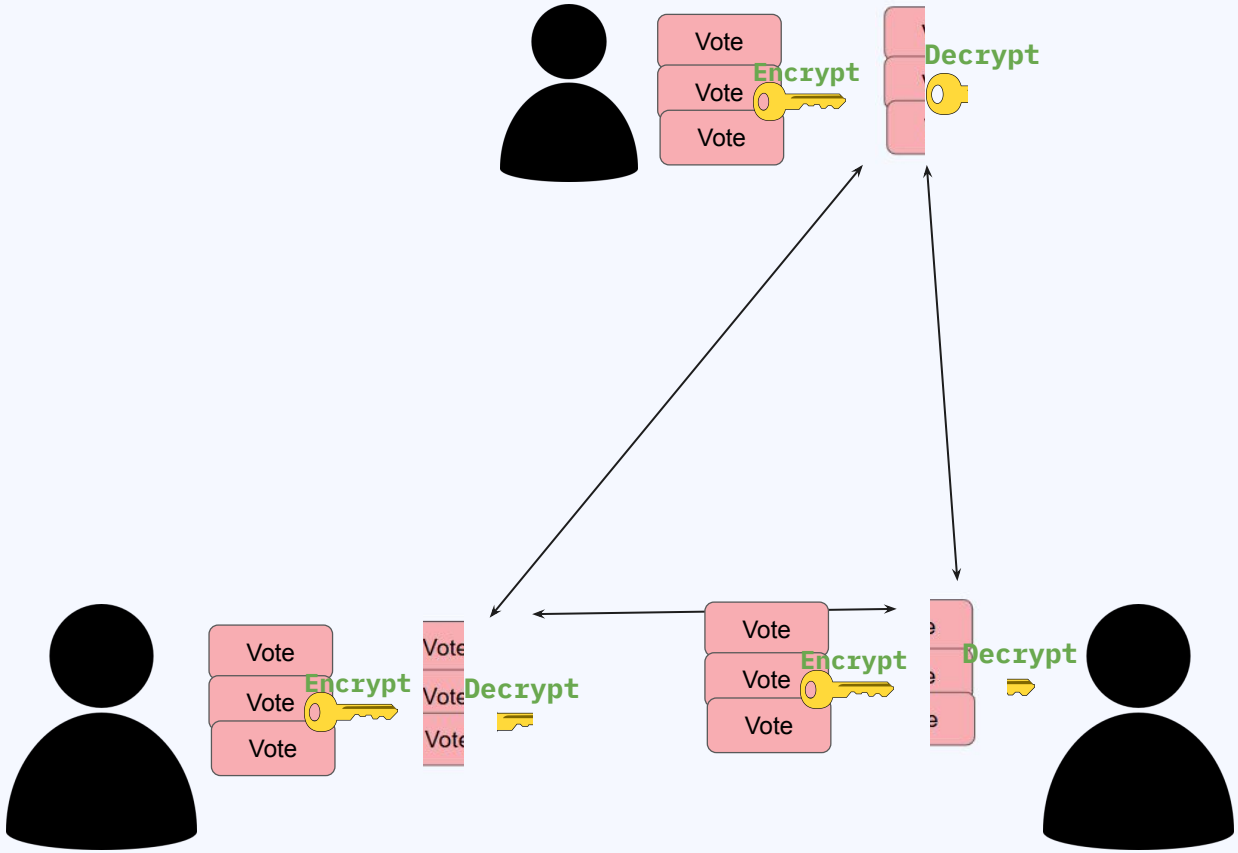




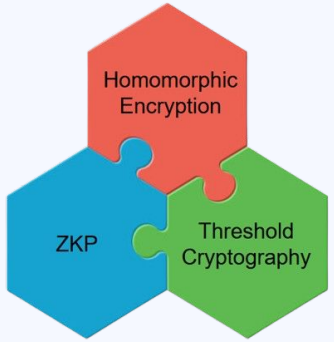


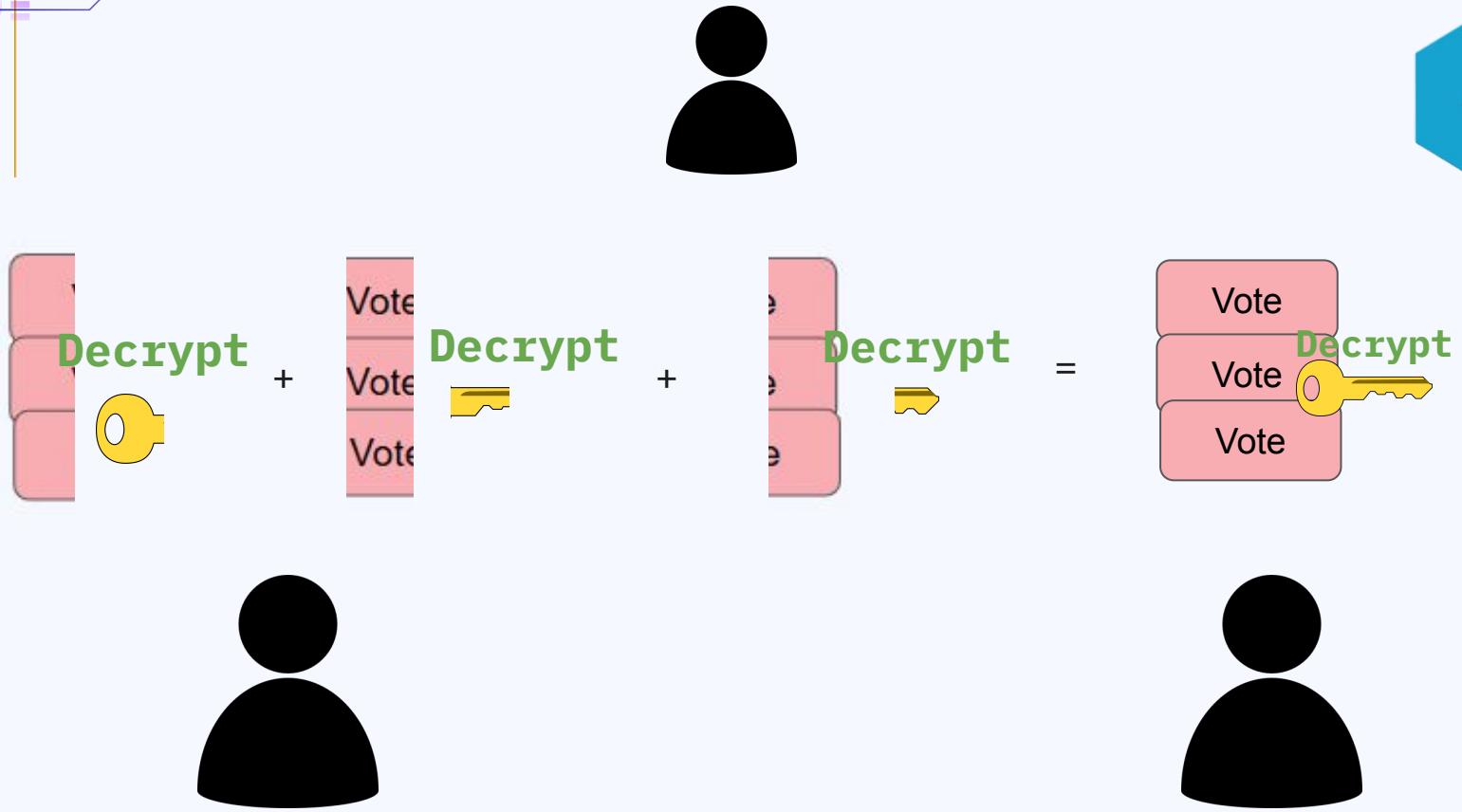
3a. Frost Lagrange



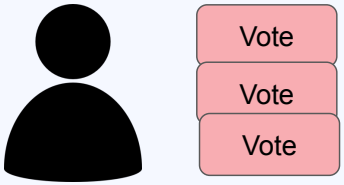


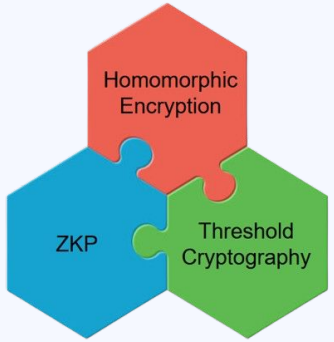
3b. Frost Lagrange

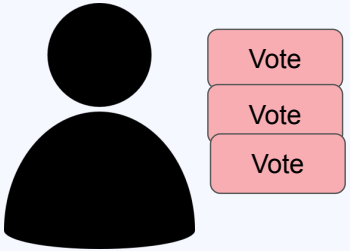


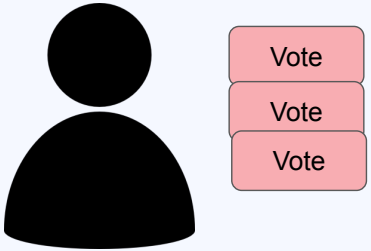


3b. Frost Lagrange

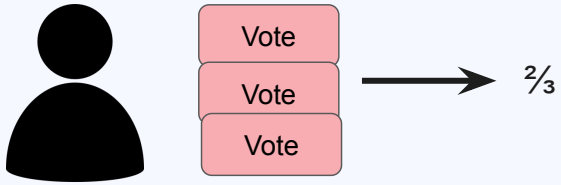


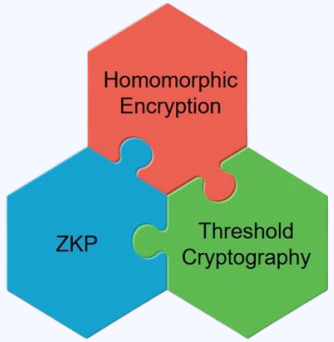


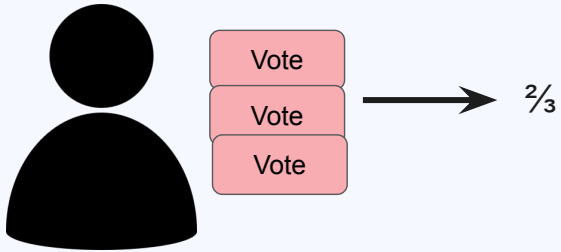


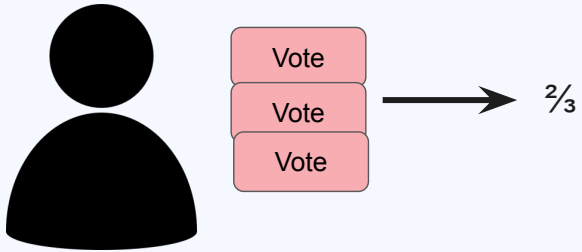


4. BSGS Extraction









Methods 2: Engineering

1) Create the Rust library implementing the theory and weaving existing libraries

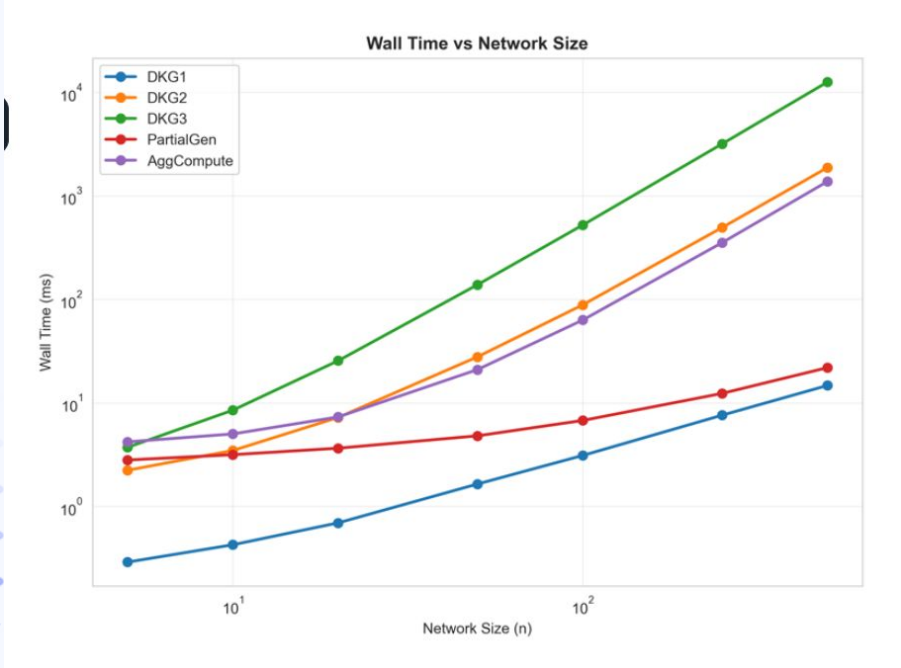
2) Add comprehensive production and defense-in-depth features: Side-Channel Protection, Rate-Limiting, DoS, Signatures, Byzantine Recognition, Error Handling...

Methods 3: Efficiency Analysis

1) Simulate different environments on Azure (5-500 participants, varying thresholds, ZKP variants, etc.)

Efficiency Results

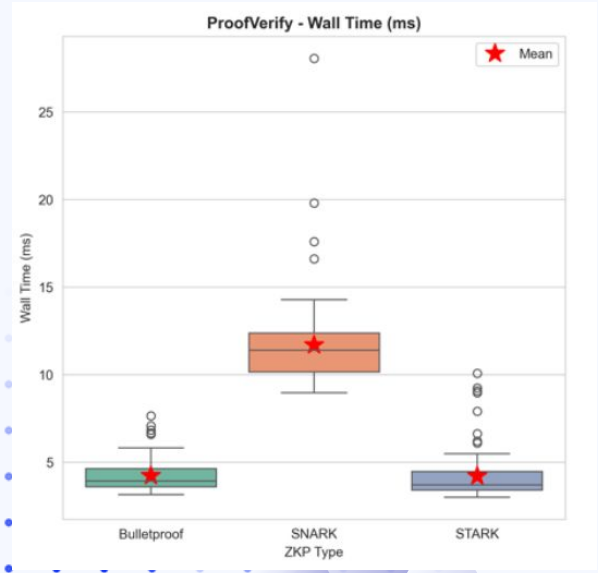
Although efficient in small and medium environments (0-500), much larger networks introduce bottlenecks.

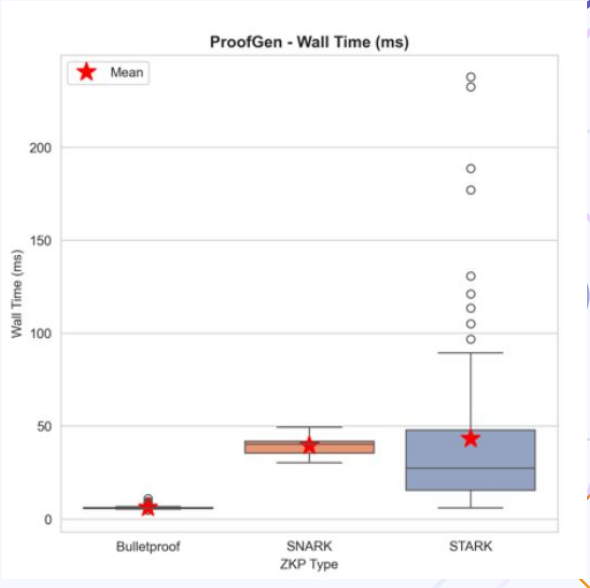


Efficiency Results

In terms of options for ZKPs

- Bulletproofs are the most efficient, but least flexible
- zk-STARKs are the most secure, but least efficient
- zk-SNARKs represent a solid compromise





zk-DEAP: Zero-Knowledge Distributed Encrypted Aggregation Protocol

Conclusion/Contributions:

- First to combine all 3 in the real-world context of aggregation
- Engineer and analyze a general purpose, production Rust library

Note: Planned to be submitted in a month…

Thank You!

zk-DEAP

Jude Ramanan
Professors Apthorpe and Haldeman
COSC 482

(And thank you Professor Perkins)

CREDITS: This presentation template was created by **Slidesgo**, and includes icons, infographics & images by **Freepik**