MPC With Weights

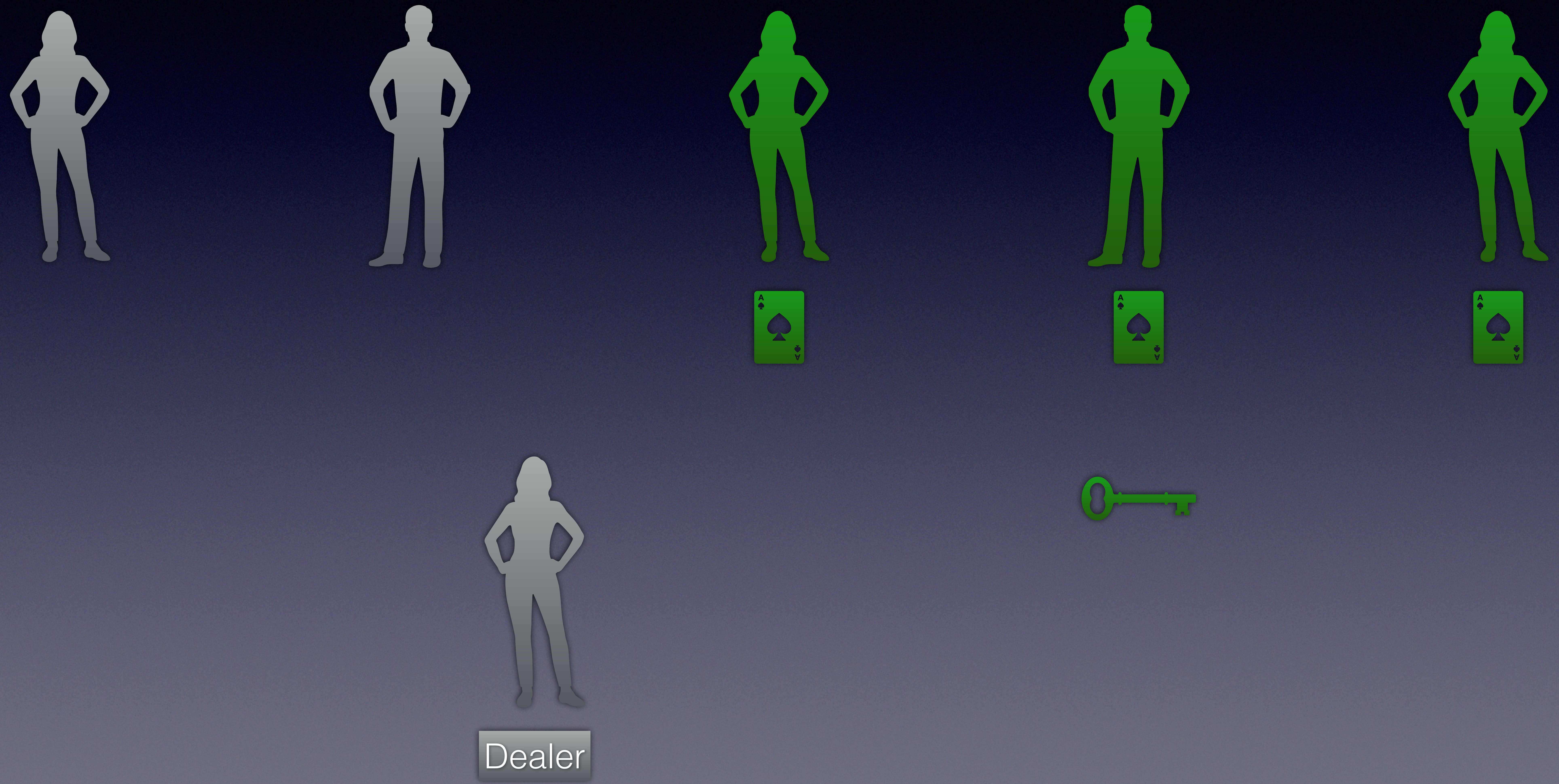
Kareem Shehata
DeCompute Conference
30 September 2025

Slides, Paper, & Code

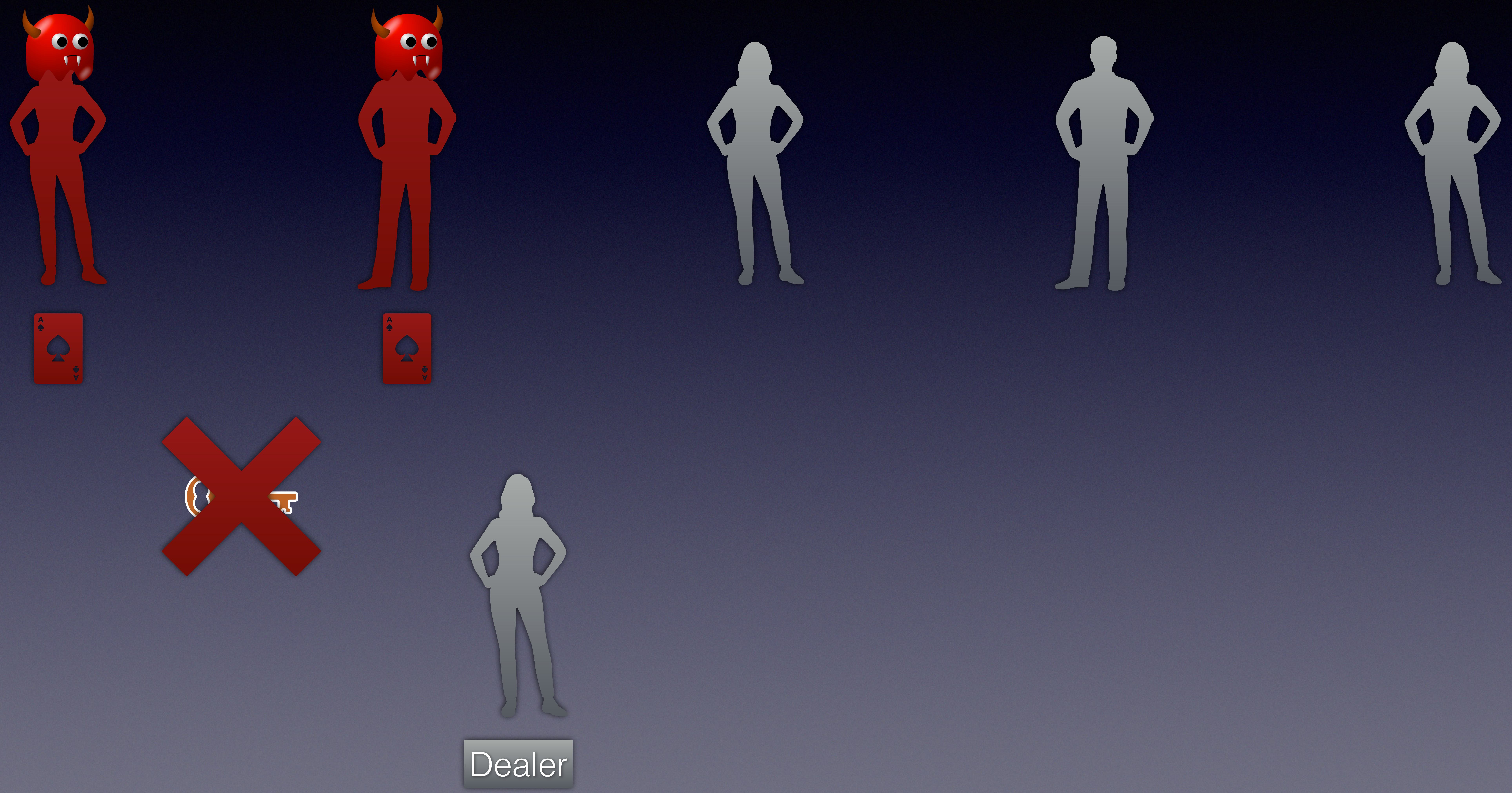
https://tinyurl.com/decompute-weighted-mpc

Secret Sharing

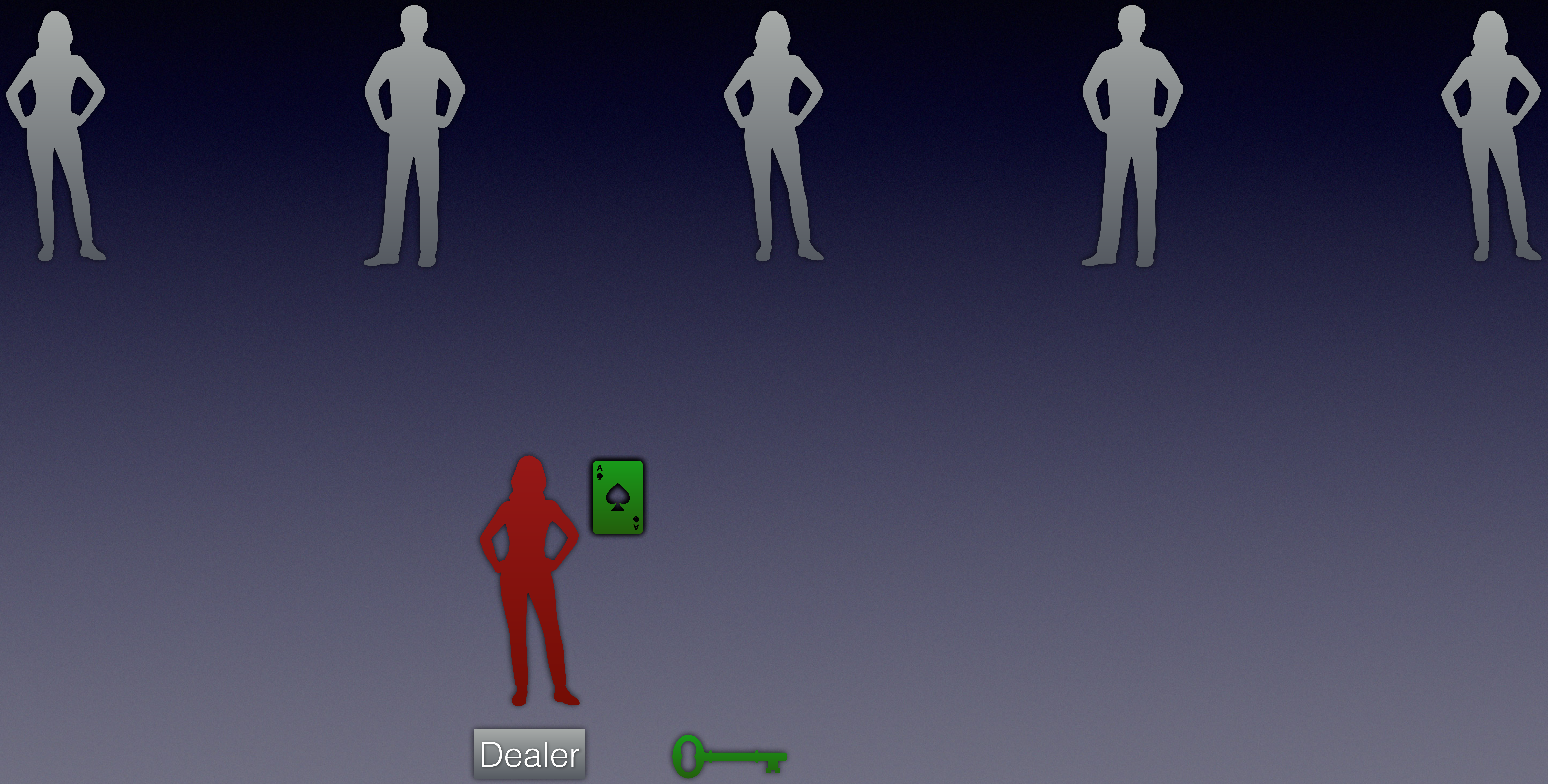
Secret Sharing



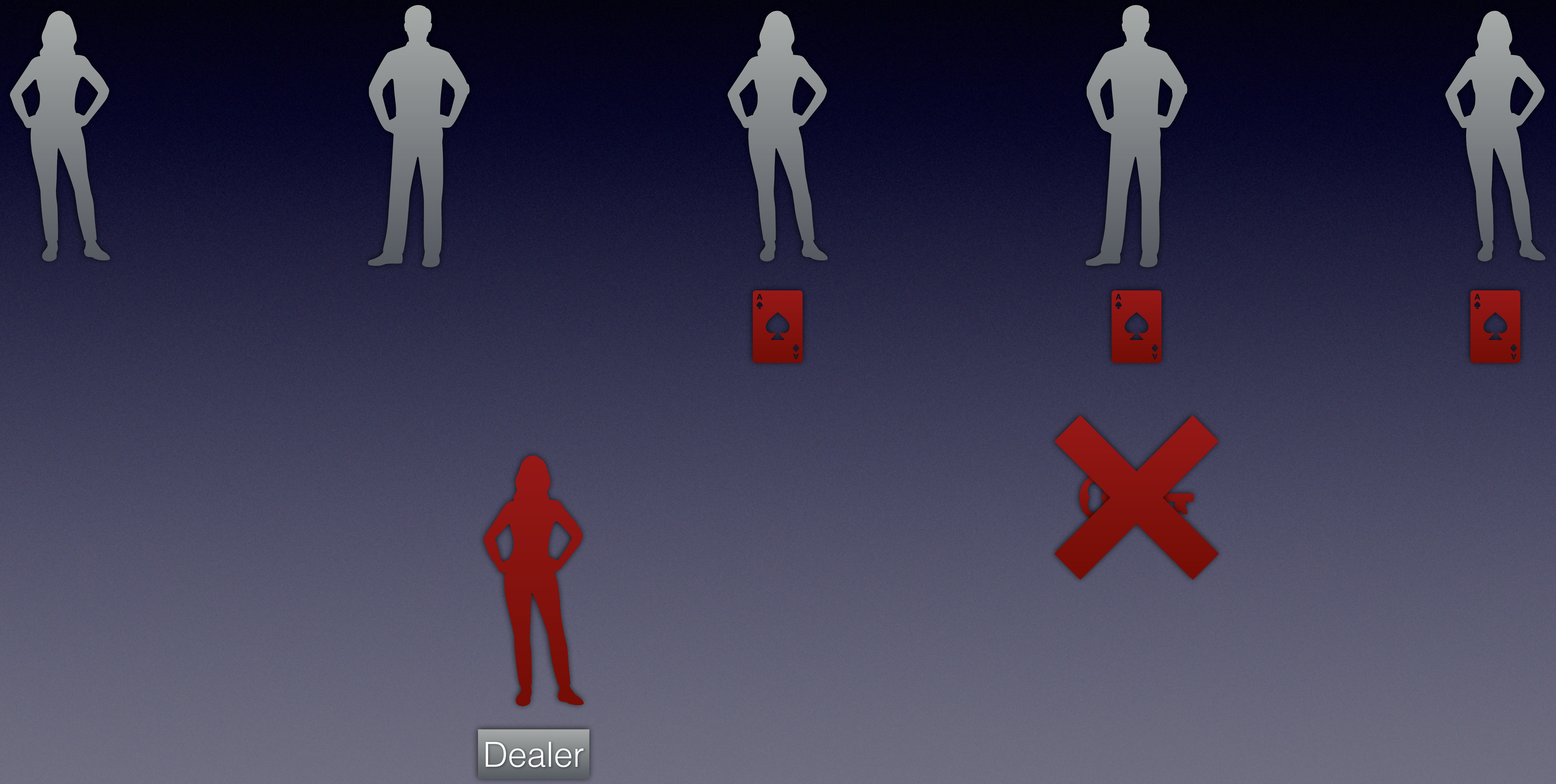
Secret Sharing Security



What about the Dealer?



What about the Dealer?

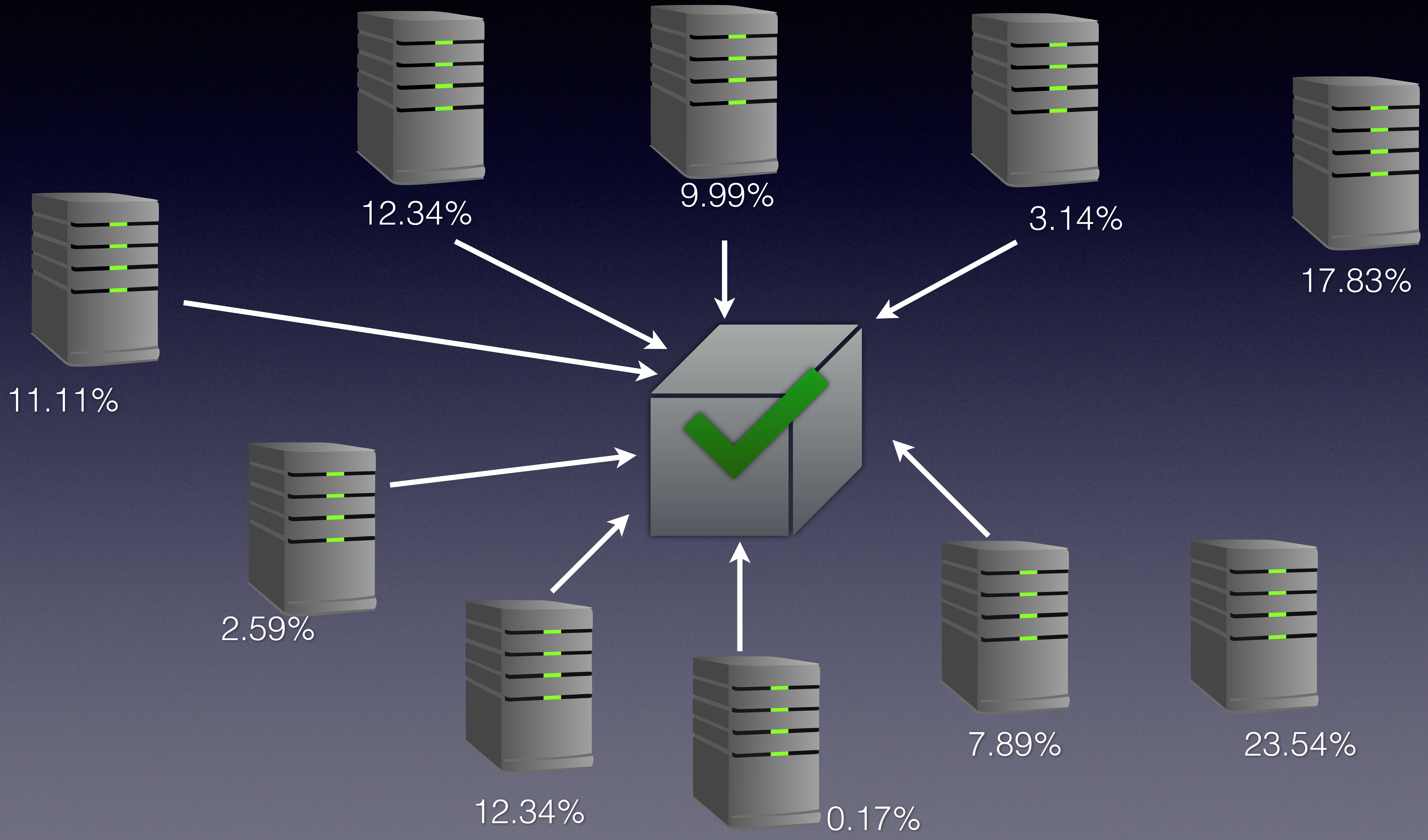


Verifiable Secret Sharing

Why care about Secret Sharing?

- Fundamental concept that underpins many other protocols
- Distributed Key Generation, Threshold Signatures, Consensus, many others…
- Simple example that allows us to reason about more complex protocols and problems

Proof of Stake Blockchain



Implicit Assumption:

Equal Weights

What happens if all parties don’t have the same level of importance or “weight”?

Ethereum Stake

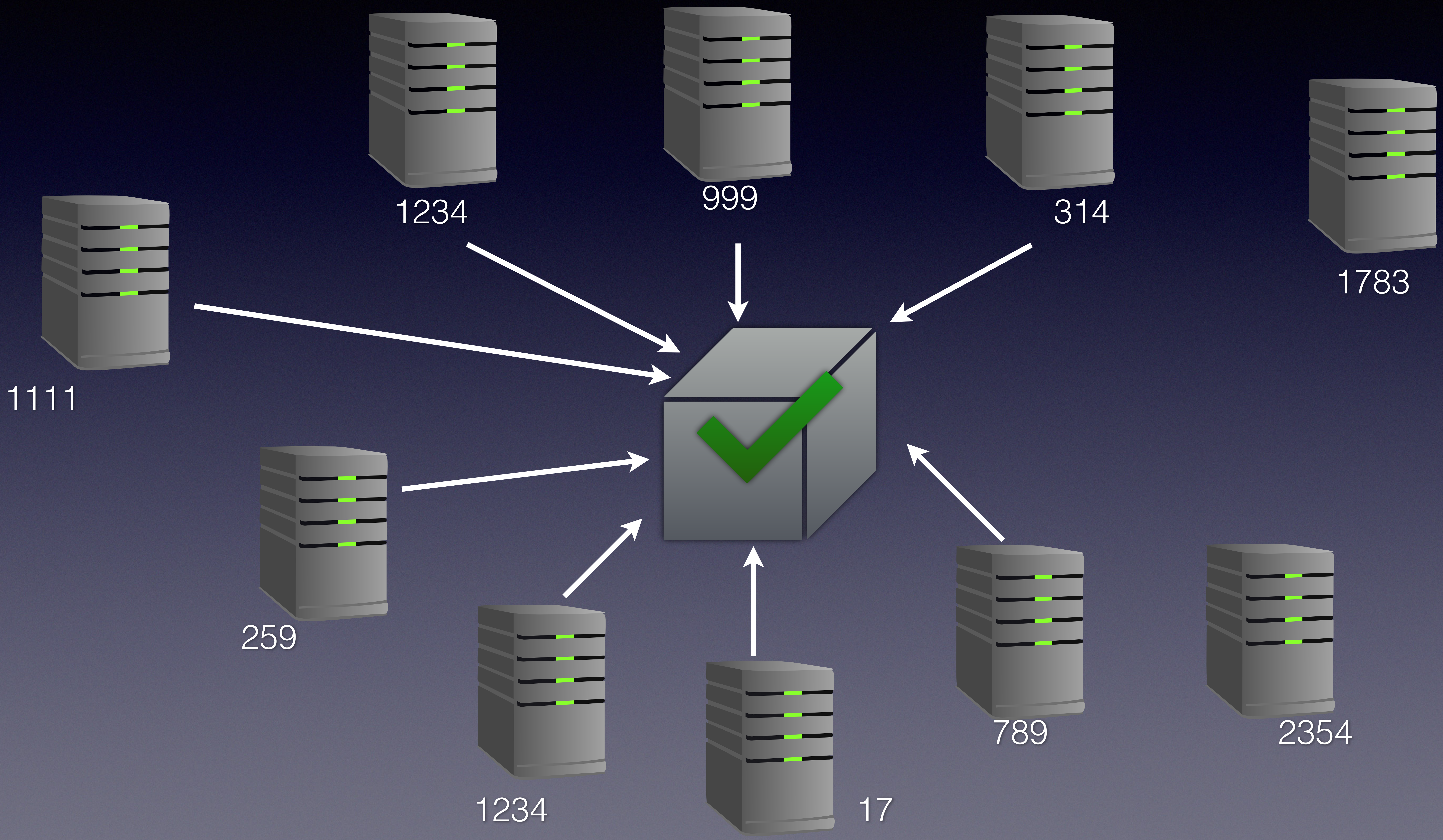
11
0.04
11
0.16
15
0.42
9
0.53
10
1.1
6
1.3
7
3.4
7
7.3
3
6.1
5
16
0.01
0.02
0.04
0.08
0.16
0.32
0.64
1.28
2.56
5.12
Stake (%)
0
5
10
15
Count
Stake (%)

Fig. 2: Distribution of Ethereum Stakes for pools other than Lido and Coinbase. Note that the x-axis is logarithmic.

Virtualisation

- Naïve solution: give parties with more weight more shares.
- Convert all weights to integers, give each party a number of shares equal to their weight.
- Very inefficient: have to do communication and computation that grows with at least $\mathcal{O}(w)$!

“Virtualized Shares”



(Verifiable) Shamir’s Secret Sharing

- Shamir’s Secret Sharing:
  - Secret s, sample $a_1, \ldots, a_t$ randomly
  - Let $f(x) = s + a_1x + a_2x^2 + \ldots + a_tx_t$
  - Shares: $s_i = f(i)$, reconstruct s with $t + 1$ points

- Verify using homomorphic commitments, e.g. commit$(x) = g^x$
  - Publish $y = g^s, y_i = g^{a_i}$
  - Check $\boxed{y \cdot y_1^j \cdots y_t^j = g^{s+a_1\cdot j+\ldots+a_t\cdot j} = g^{s_j}}$

Requires Linearity!

Linear vs CRT Secret Sharing

- Linear (SSS):
  - Equal Weights
  - Easy and flexible
  - Verifiable constructions
  - Single group

- CRT (non-linear):
  - Weighted constructions
  - Non-linearity makes it more difficult to work with
  - No verifiable constructions with a single group*

*K. Kaya and A. A. Selçuk, "Secret sharing extensions based on the Chinese remainder theorem," provides a construction using unknown order groups and requires the strong RSA assumption and a trusted setup.

Chinese Remainder Theorem

- Let $p_1, \ldots, p_n$ be arbitrary integers, all co-prime

- Chinese Remainder Theorem:
- Given $a_1, \ldots, a_n, a_i \in [p_i]$,
- The system of equations $\{a_i = a \mod p_i\}$
- Has a unique solution $a \in [0, p_1 \cdots p_n]$

CRT-Based Secret Sharing

- Uses Chinese Remainder Theorem instead of polynomials
- Divisor p_i determines "weight"
- Non-linear, only known verifiable version requires strong RSA assumption and unknown order groups, not good for blockchain.

S. Garg, A. Jain, P. Mukherjee, R. Sinha, M. Wang, and Y. Zhang, "Cryptography with weights: Mpc, encryption and signatures," in Advances in Cryptology – CRYPTO 2023,

CRT Deal Proof

To prove a correct deal starting from a secret s to a share s_i with "weight" value p_i, we just need to prove that:

$$s_i = s + kp_i$$

For some $k < p_i$,

Why not R1CS / Bulletproofs?

- We can easily prove $s_i = s + kp_i$ using R1CS proofs
- … but only if all the values live in one group.
- For the security of any practical system, we'll want the base secret to be in the group, and the rest of the values much *much* larger than the group.

Problems with Cyclic Groups

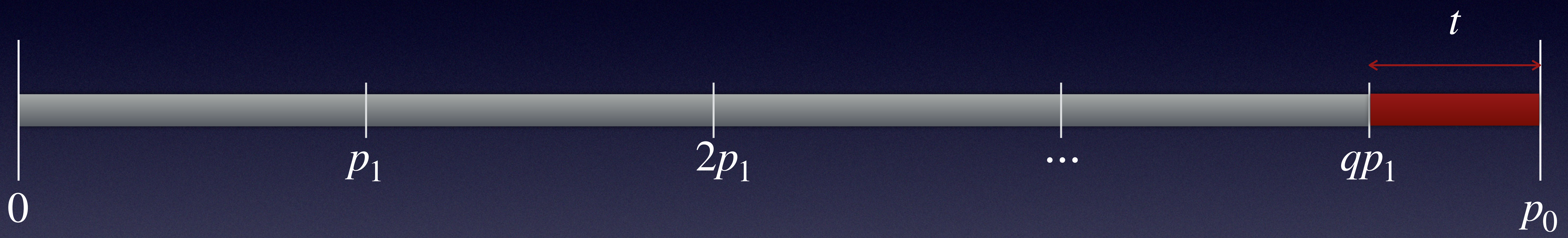
If we use the same cyclic group for commitments as the desired crypto system, then:

1. $s = s_0 + up_0 = s_0 \mod p_0$
2. Can *always* find k' such that $s = s_i + k'p_i \mod p_0$ for any s, s_i!

Either we need to use another, much larger group (previous solutions), change our setup, or be a lot more clever.

Wraparound $\mathbf{mod} p_0$

Let $p_0 = qp_1 + t, 0 \leq t < p_1$



If $a = b + kp_1$, and $a < p_0$ then, either:

- $k < q$ and b can be any value in p_1, OR
- $k = q$ and $b < t$

“Proof of Mod” $b = a \mod p_1, a, b \in \mathbb{Z}_p$

Let $p_0 = qp_1 + t$, where $0 \le t < p_1$, and

1. Prover commits to a, b, k such that $a = b + kp_1$
2. Construct R1CS proof that is only satisfied if:
   A. $b + kp_1 = a \mod p_0$ AND
   B. $(0 \le k < q) \wedge (0 \le b < p_1)$ OR
   C. $(0 \le k \le q) \wedge (0 \le b < t)$

Proof of mod for values $< p_0^2$

Intuitive idea: use the "proof of mod" several times in a row to progressively bring things in range to show:

$$s_1 = s_0 + ap_0 \mod p_1$$

↓

$$s_1 = (s_0 \mod p_1) + (a \mod p_1) \cdot (p_0 \mod p_1) \mod p_1$$

CRT-VSS using a single DL group

If $p_i < < p_0$ and $p_0 < P_{max} < < p_0^m < P_{min}$

Then the dealer can:

1. Distribute shares as in CRT-SS

2. Provide commitments to all shares

3. Expanded proof-of-mod to prove correct dealing for all shares

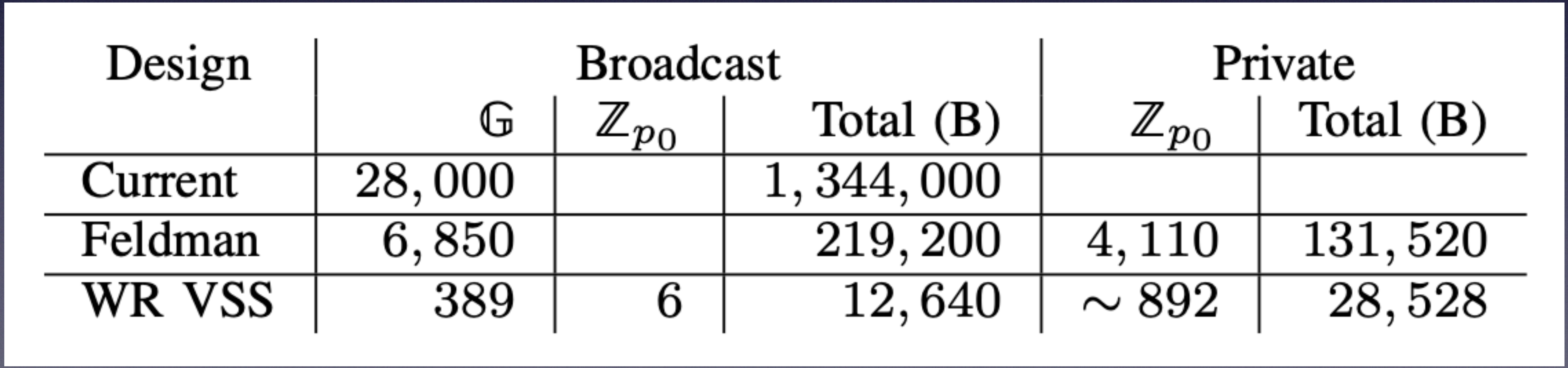
CRT-VSS using a single DL group

Participants:

1. Check that shares match commitments
2. Verify the proof-of-mod for all shares

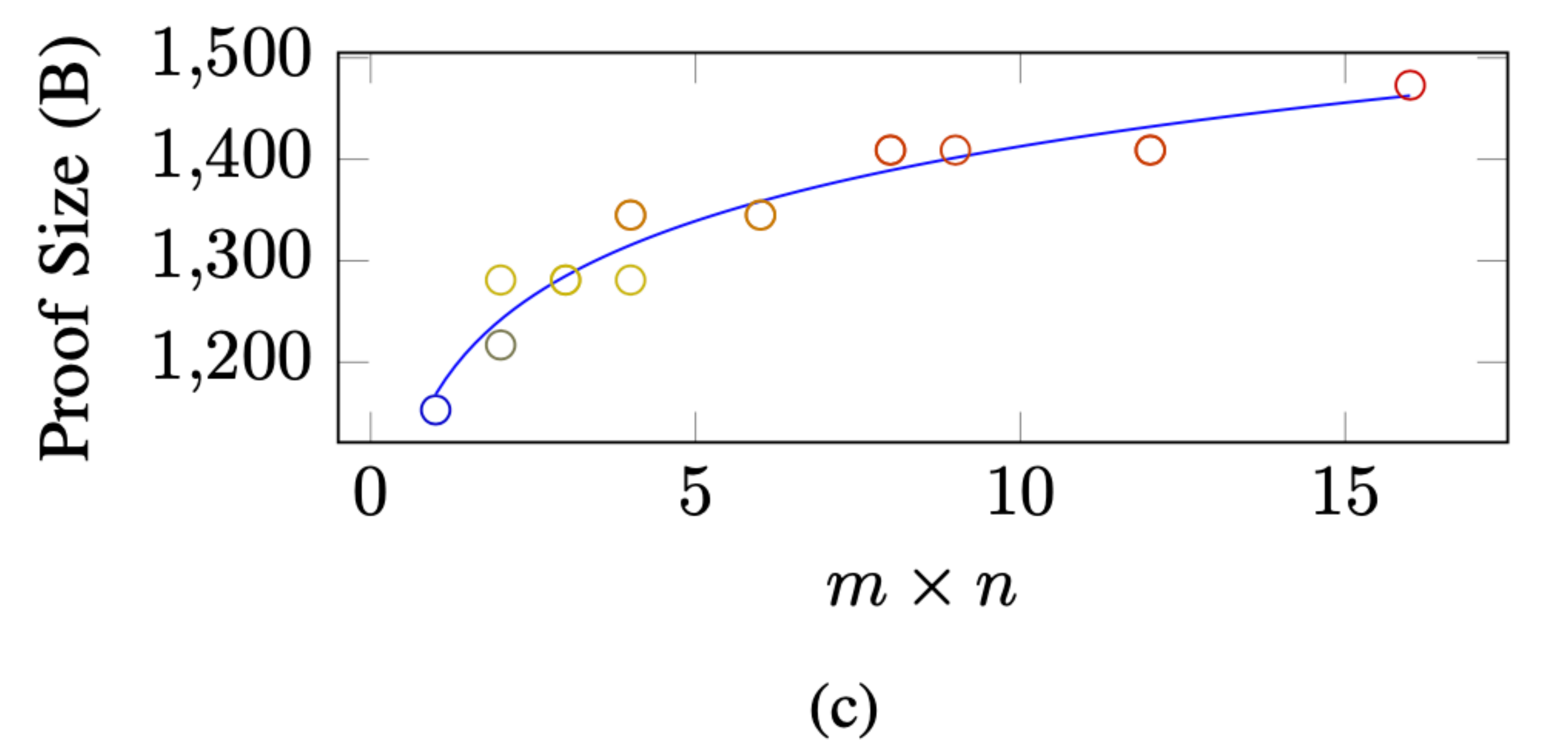
Performance Improvement of WR-VSS

- 100x improvement in broadcast bw on current implementation
- 20x improvement in broadcast bw vs virtualized VSS
- 5x improvement in private bw vs virtualised VSS



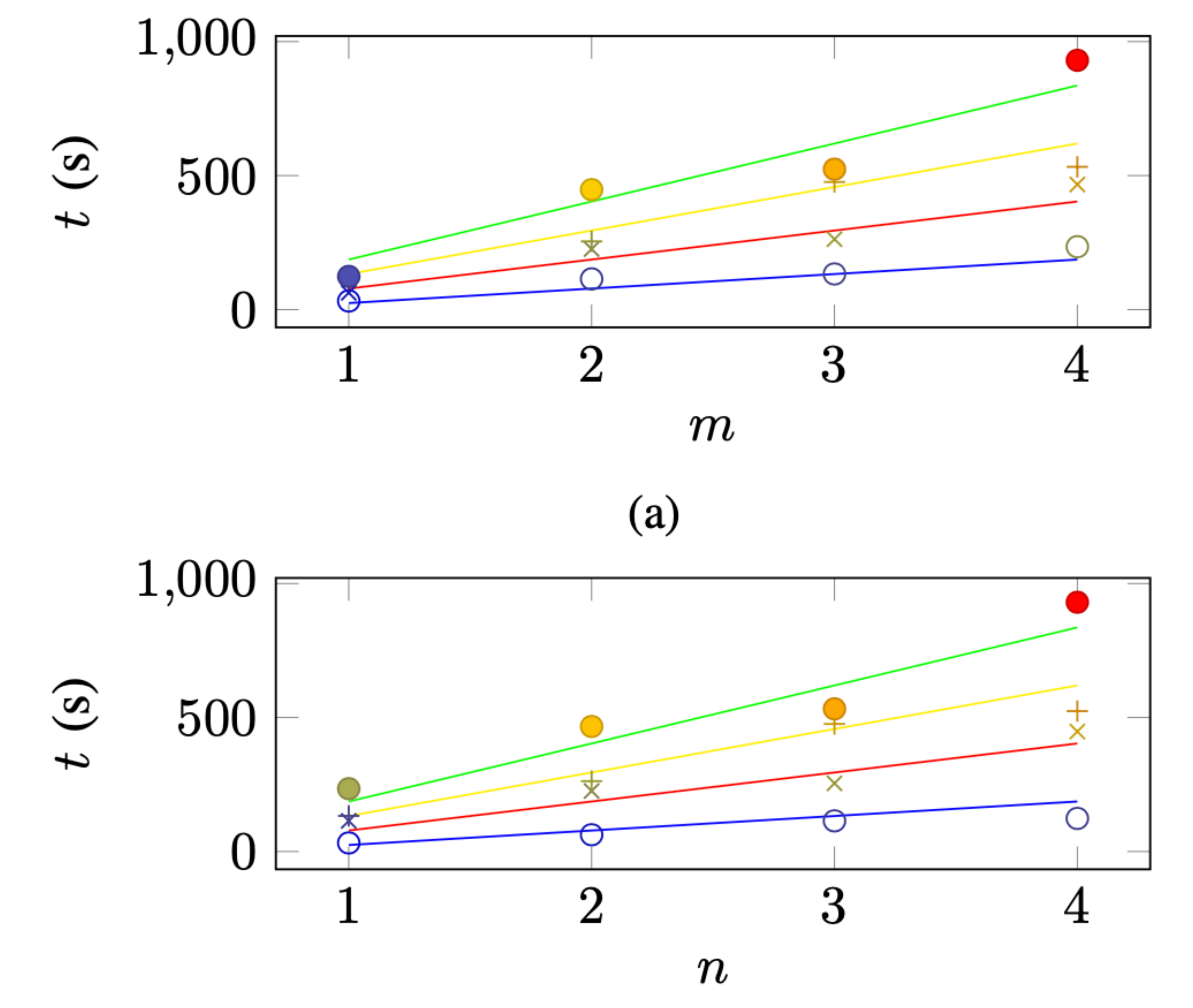
Design	Broadcast			Private	
	$\mathbb{G}$	$\mathbb{Z}_{p_0}$	Total (B)	$\mathbb{Z}_{p_0}$	Total (B)
Current	28,000		1,344,000		
Feldman	6,850		219,200	4,110	131,520
WR VSS	389	6	12,640	~ 892	28,528

Proof Size and Running Time



(c)

Proof Size



(a)

Running Time

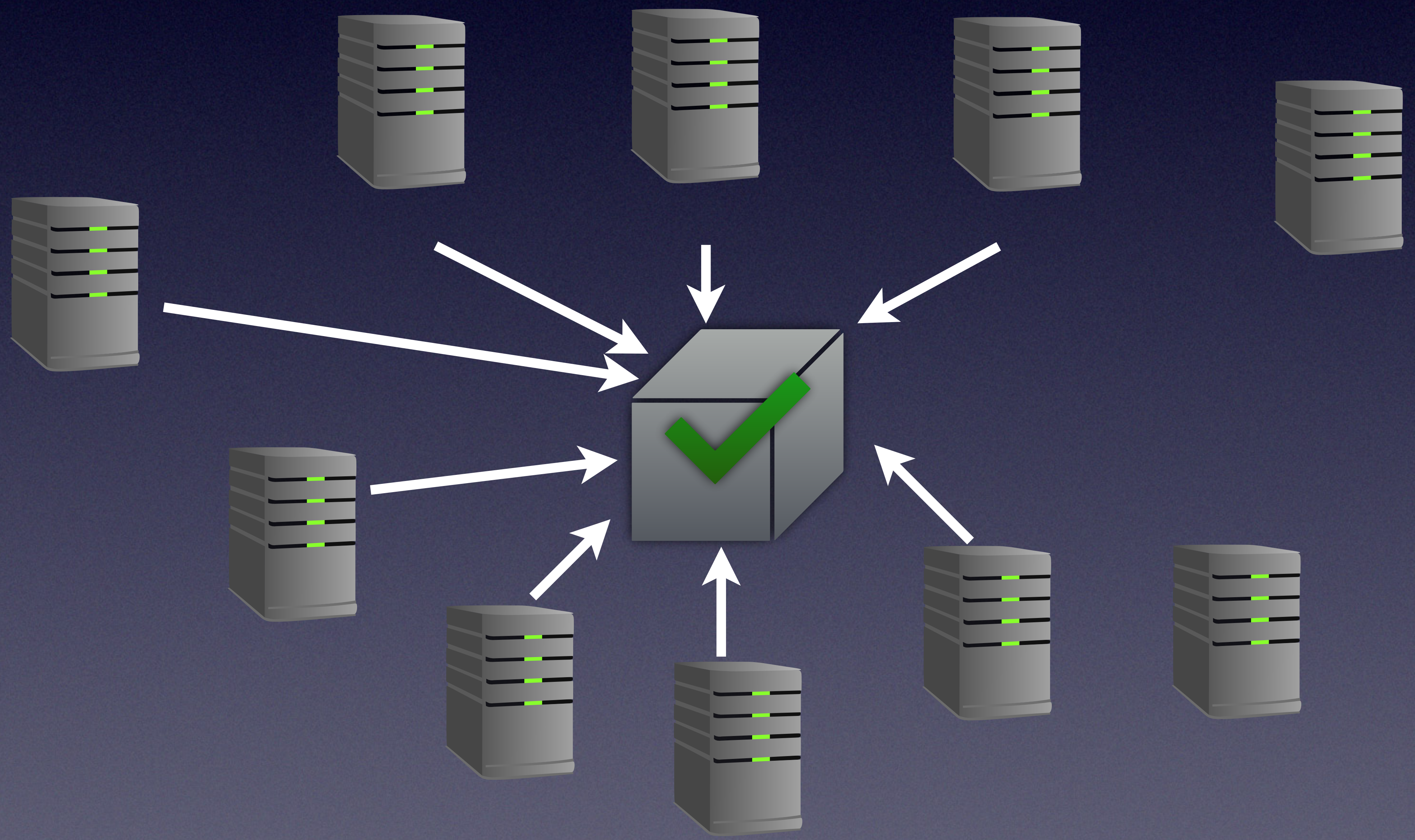
WR-VSS Summary

- Shown how to construct the first verifiable and weighted secret sharing scheme that uses only a single discrete-log group.
- WR-VSS produces much smaller proofs than using even the simplest non-weighted VSS.
- But current Bulletproofs R1CS has high overhead in proving time, not yet practical for use.

What does this mean for Blockchains?

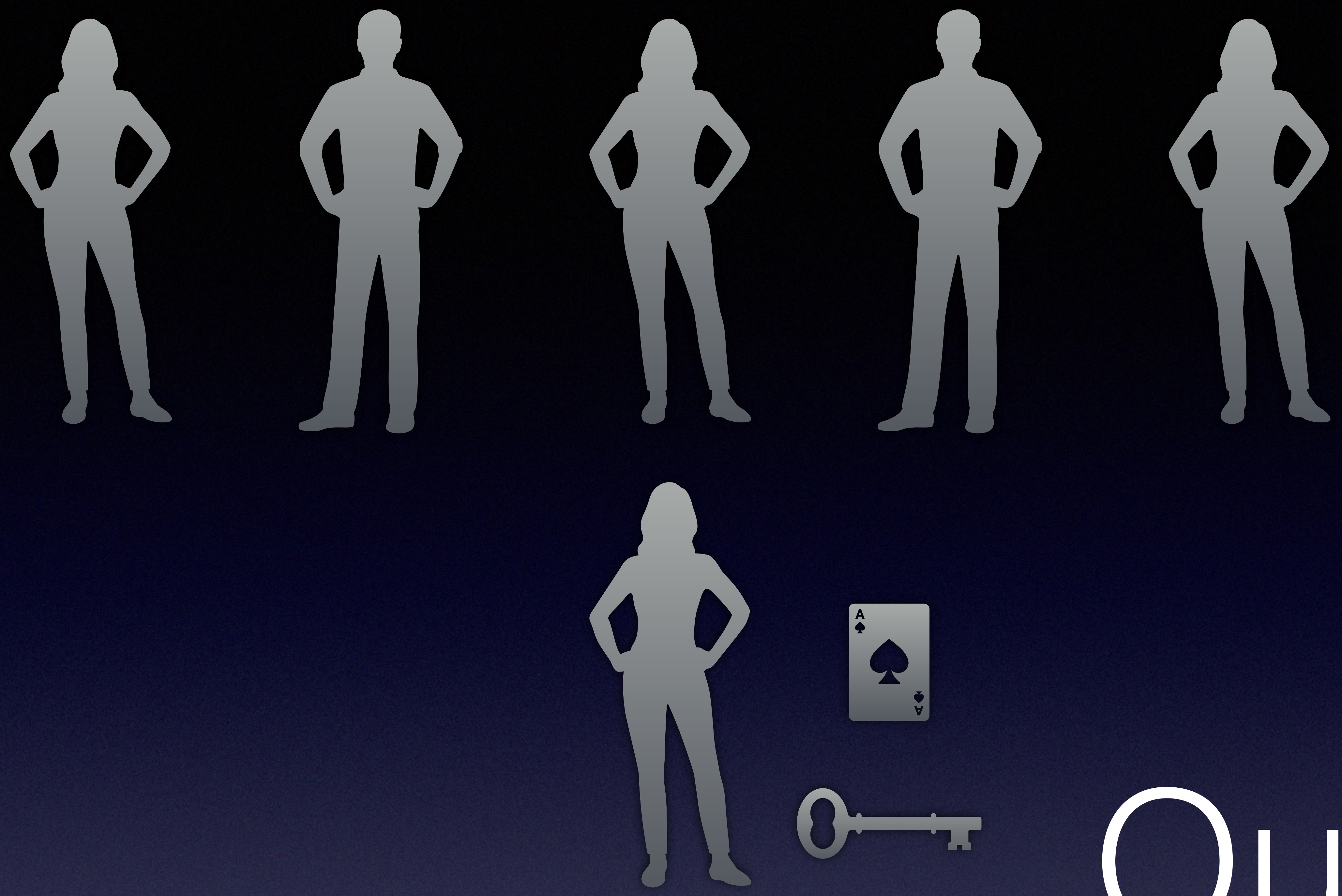
SS → VSS → DKG → MPC/TSS

- We know how to do each of these now
- Currently the constants aren't practical
- And linear schemes are hyper-optimised
- … but soon, we'll be able to bring the constants to something more competitive.

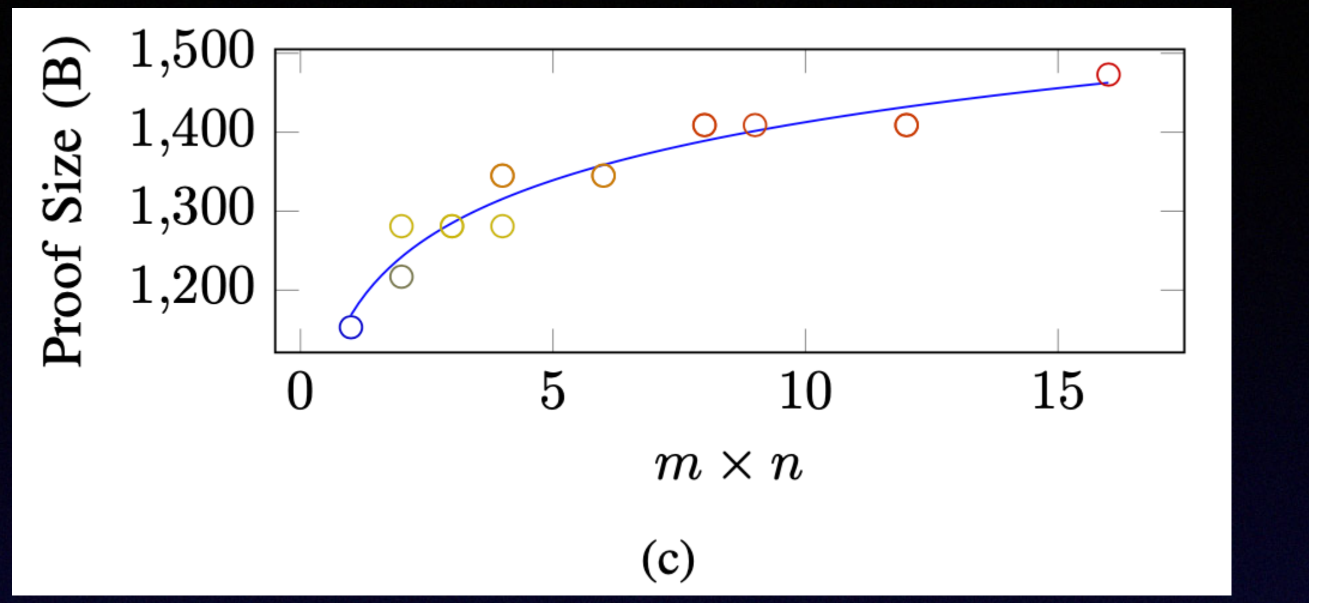


Summary

- MPC / TSS / Distributed Crypto with Weights is possible.
- Can yield very large benefits in highly unequal distributions, such as bandwidth for Ethereum staking.
- Currently the constants are too large for practical use, but soon we could see weighted MPC outperform virtualised unweighted schemes.

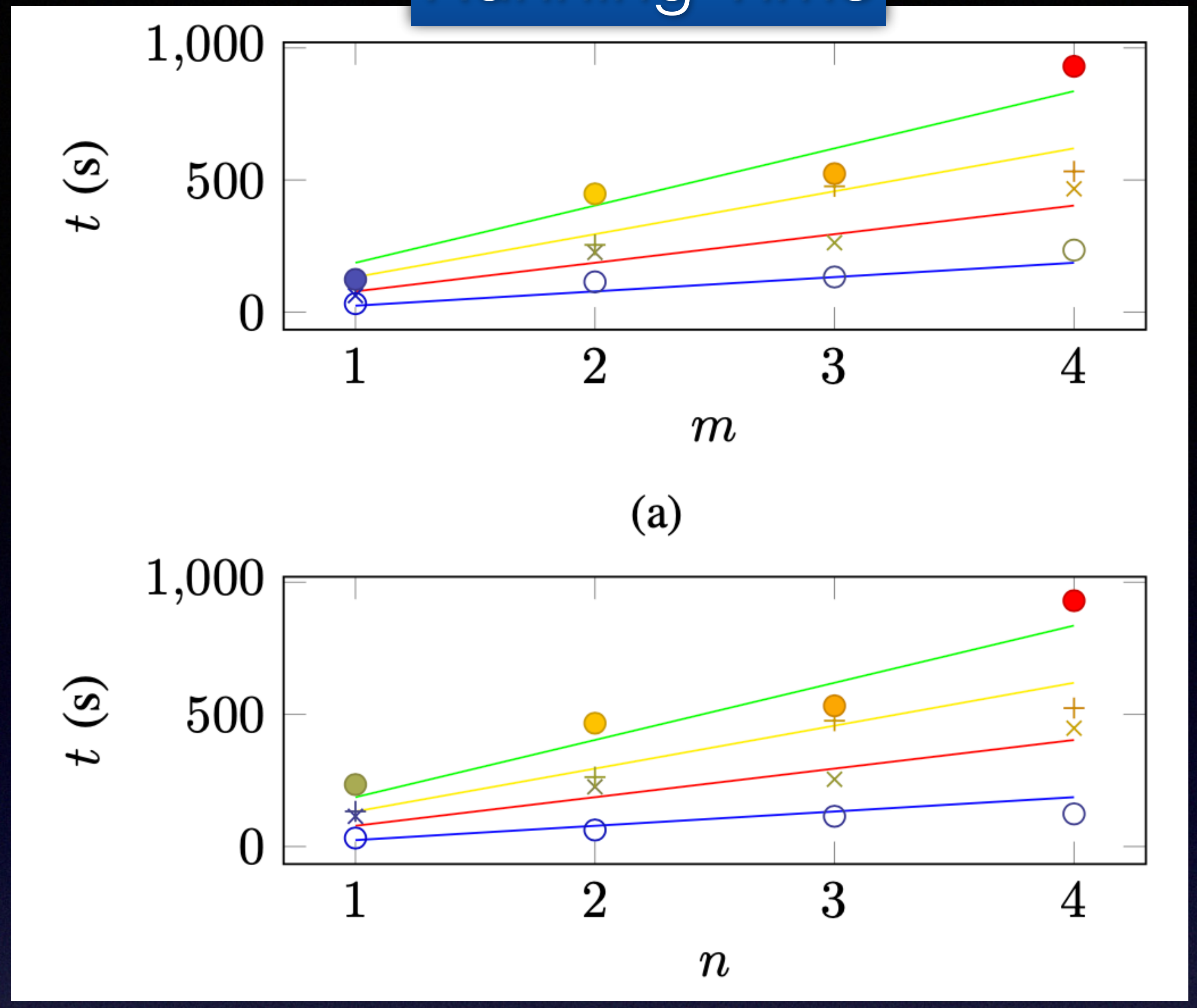


Proof Size

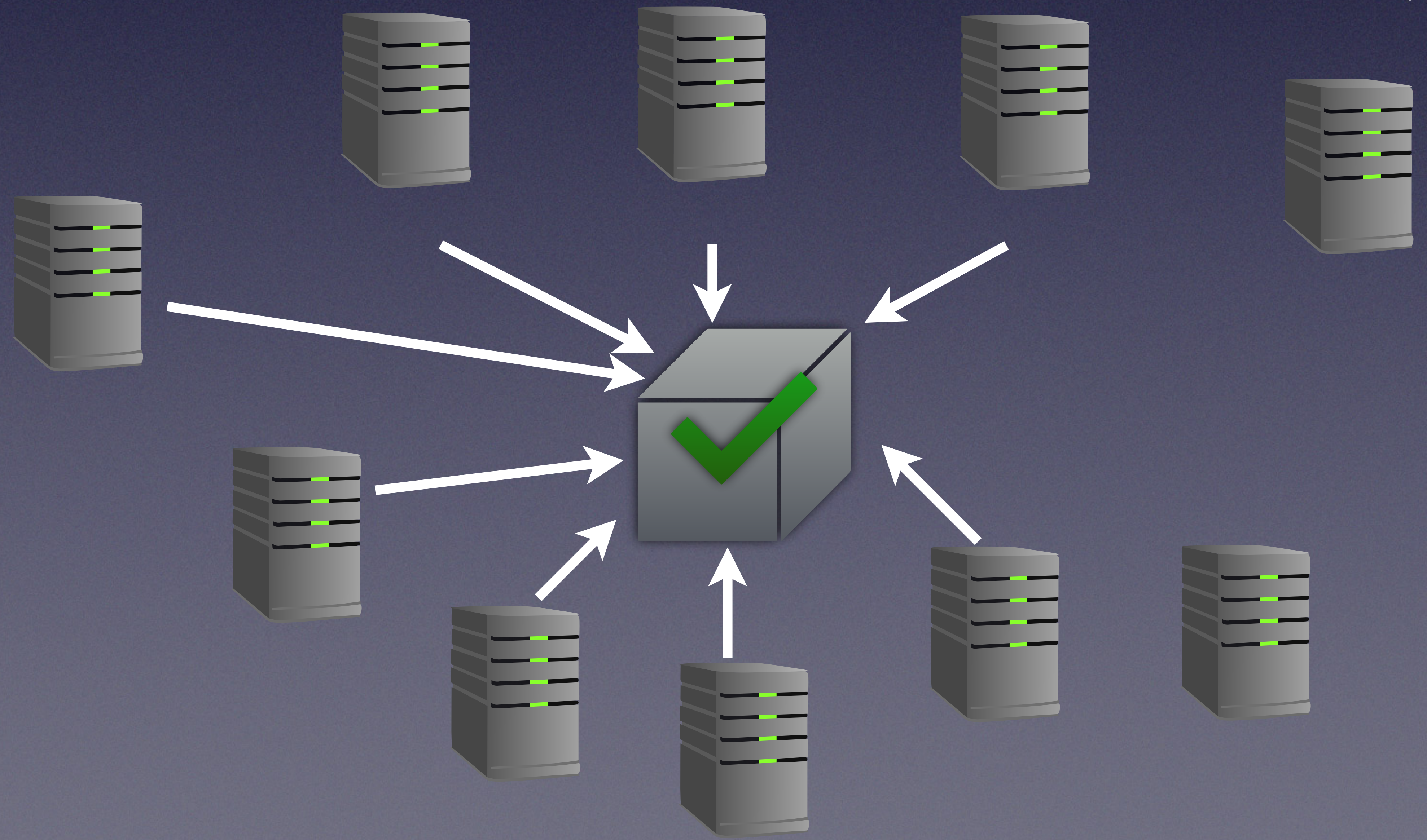


(c)

Running Time



Questions?



Slides, Paper, & Code

https://tinyurl.com/decompute-weighted-mpc