Verifiable Weighted Secret Sharing

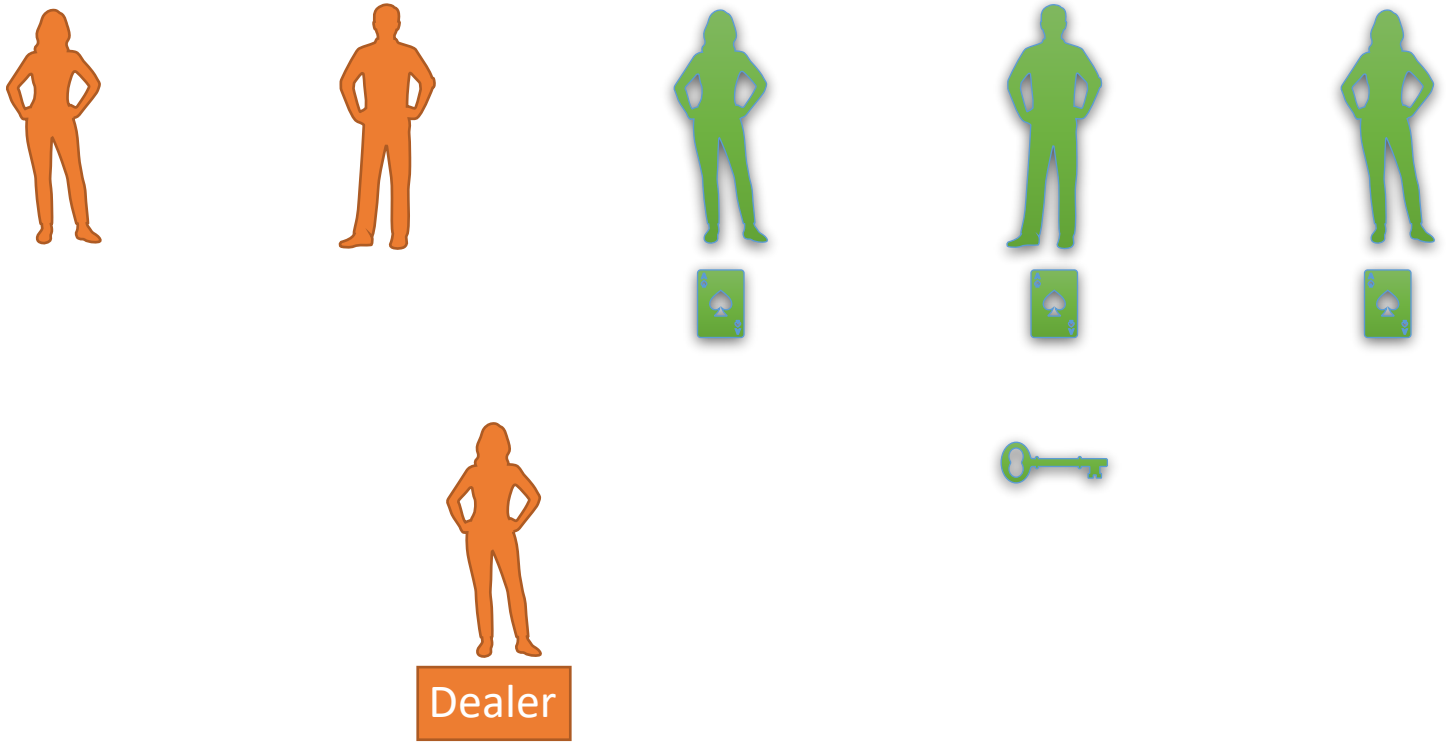
Kareem Shehata

Crypto Valley Conference, 6 June 2025

Joint work with Han Fangqi, National University of Singapore and Sri AravindaKrishnan Thyagarajan, University of Sydney

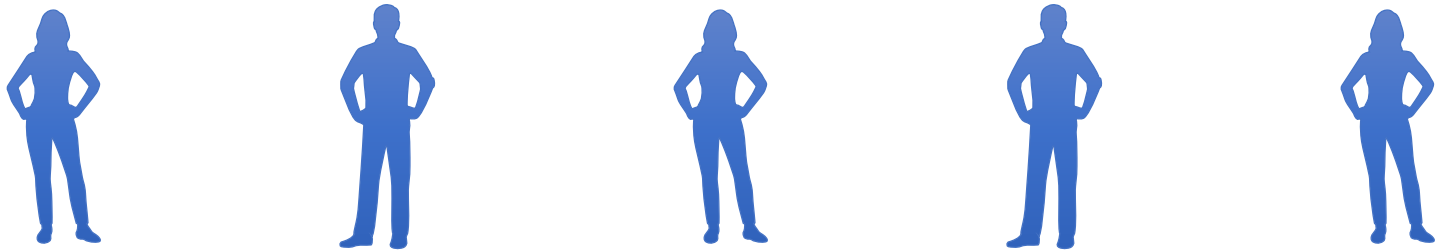
2 Secret Sharing

3 Secret Sharing

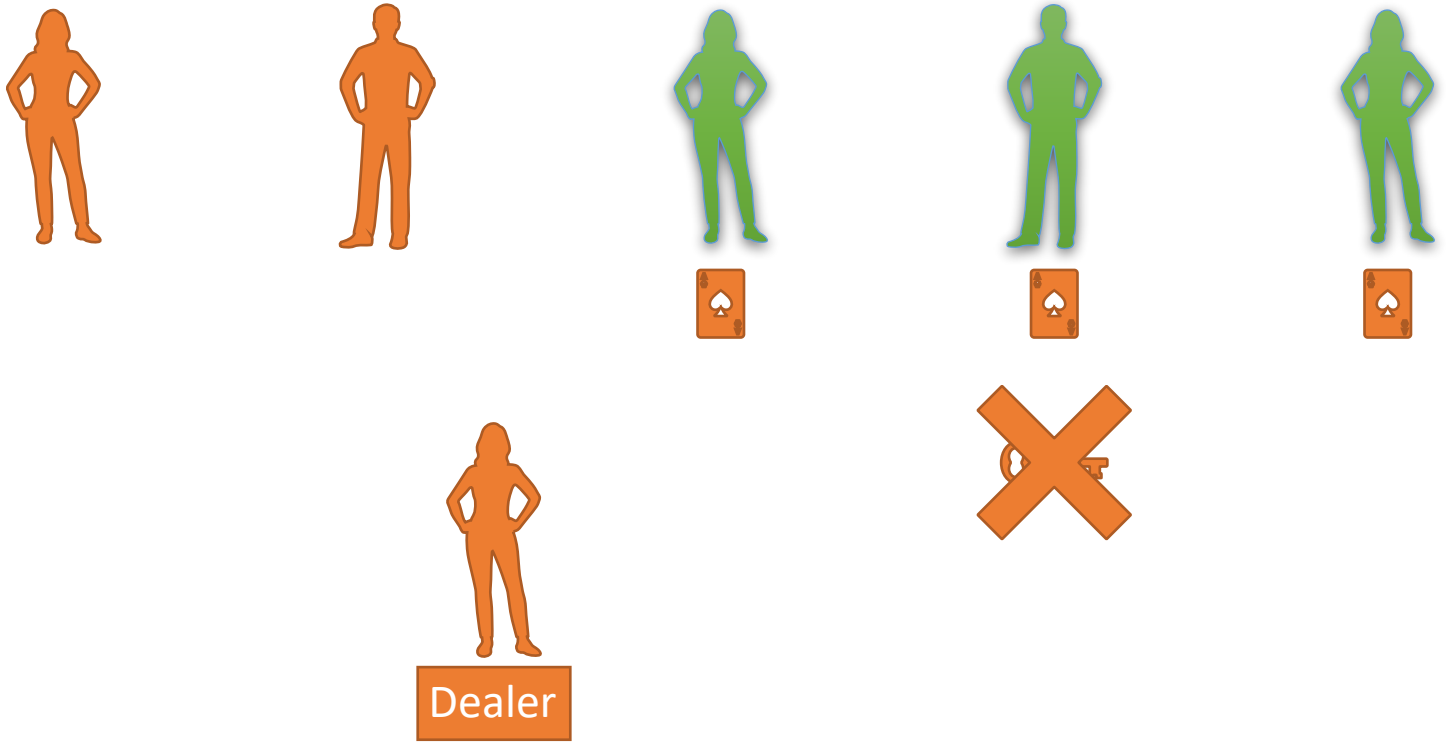


4 Secret Sharing Security

5 What about the Dealer?



6 What about the Dealer?

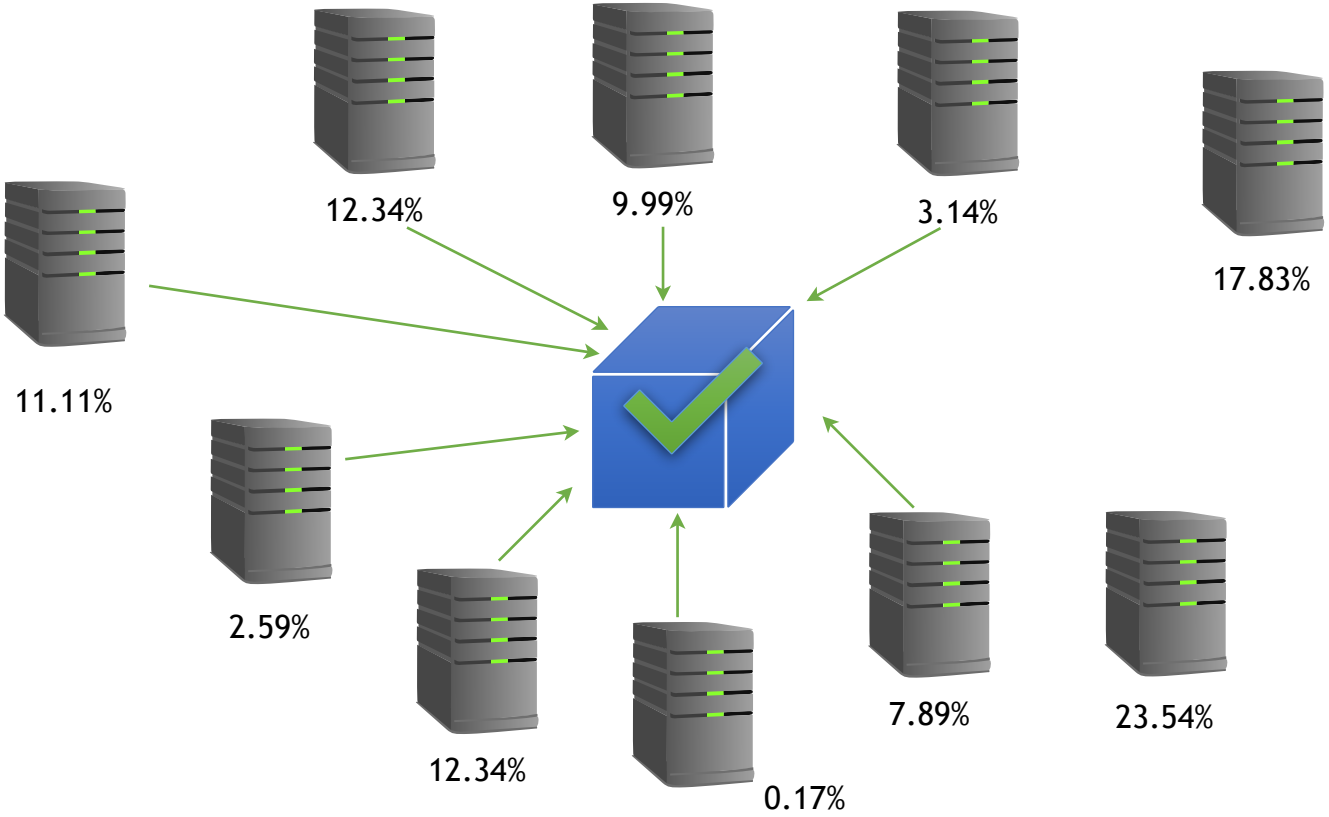


Verifiable Secret Sharing

Why care about Secret Sharing?

- Fundamental concept that underpins many other protocols
- Distributed Key Generation, Threshold Signatures, Consensus, many others…

Proof of Stake Blockchain



Implicit Assumption: Equal Weights

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

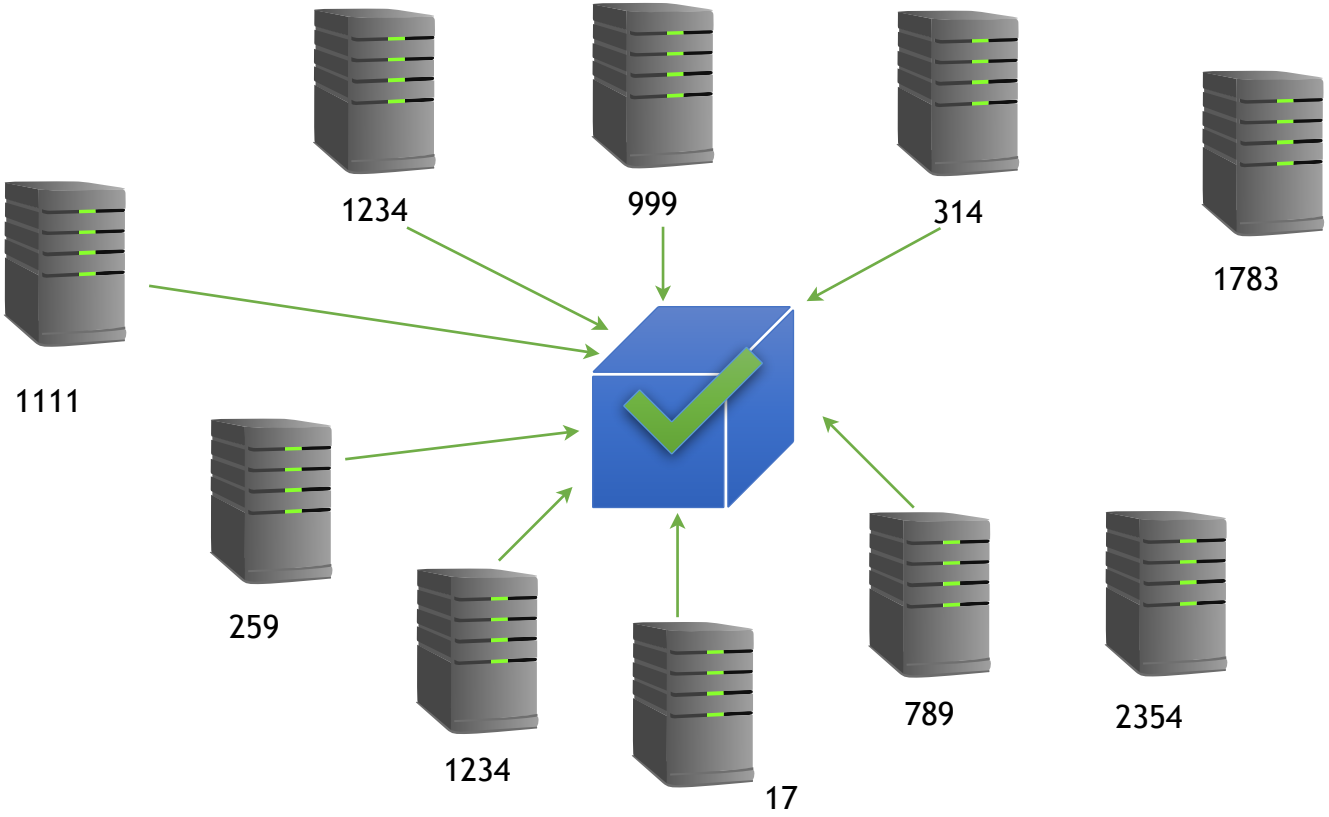
11 Ethereum Stake

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”



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

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.

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

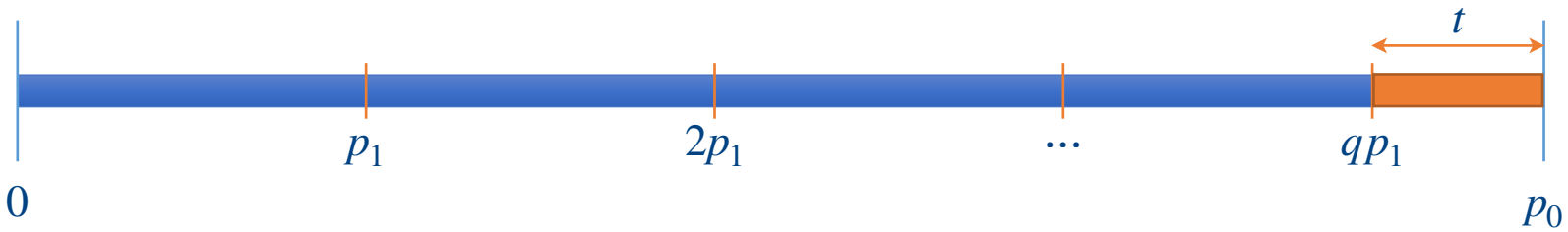
If we use the same cyclic group for commitments as the desired crypto system, then:

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 $\mod p_0$

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" $b = a \mod p_1, a, b \in \mathbb{Z}_p$

Prover has a, b, sends verifier $A = \text{Com}(a; r_a), B = \text{Com}(b; r_b)$

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

1. Prover sends $V = \text{Com}(k; r_k)$
2. Prover sends proof that $b + kp_1 = a \mod p_0$
3. Use disjunctive proof strategy on following statements:
   - A. $(0 \leq k < q) \wedge (0 \leq b < p_1)$ OR
   - B. $(0 \leq k \leq q) \wedge (0 \leq b < t)$

Both A and B above are just range proofs, can use Bulletproofs or others

With these in place, have a proof-of-mod, since $b + kp_1 < p_0$

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. Use the expanded proof-of-mod to prove correct dealing for each share

CRT-VSS using a single DL group

Participants:

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

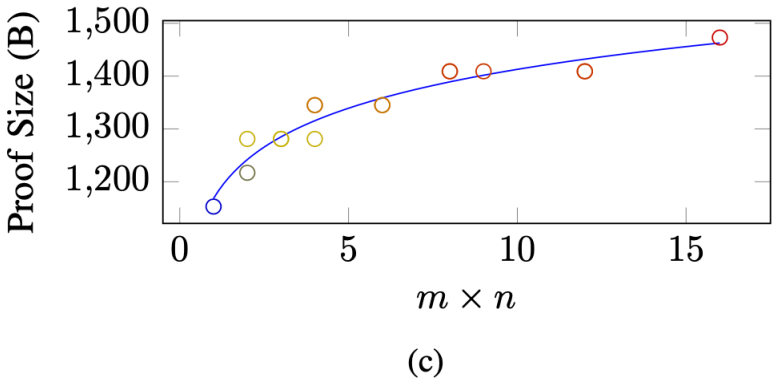
Performance Improvement of WR-VSS

- 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	$28,528$

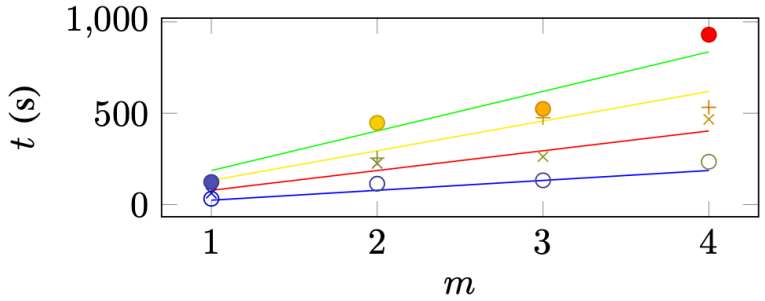
Proof Size and Running Time

Proof Size

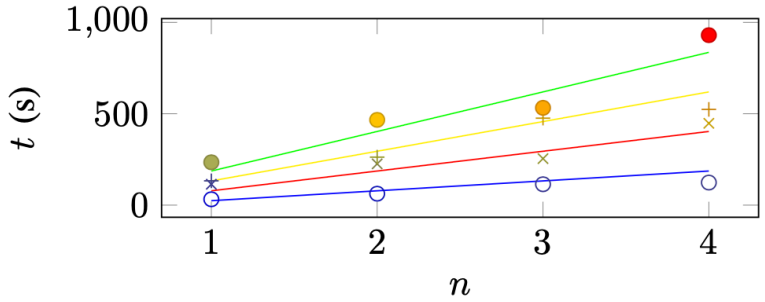


(c)

Running Time

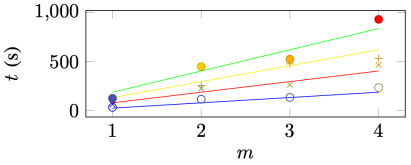


(a)

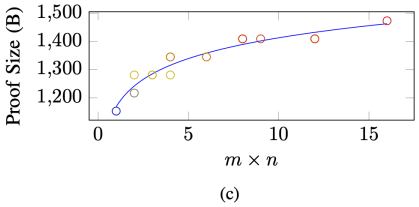


Summary

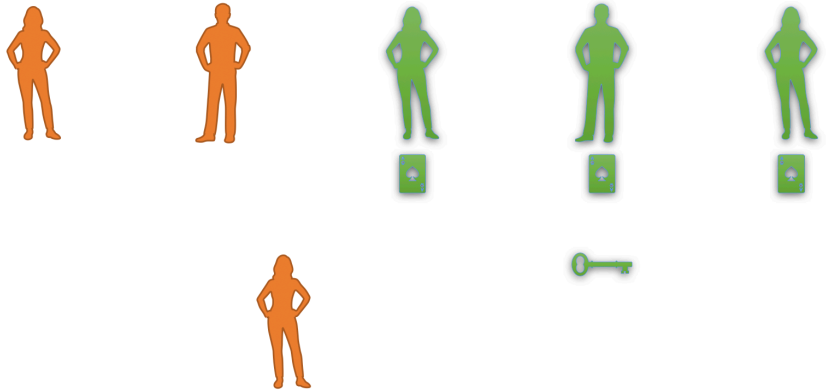
- Shown how to construct the first verifiable and weighted secret sharing scheme that uses only a single discrete-log group.
- WR-VSS produces much smaller proofs than using even the simplest non-weighted VSS.
- But current R1CS proof systems have high overhead in proving time, not yet practical for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



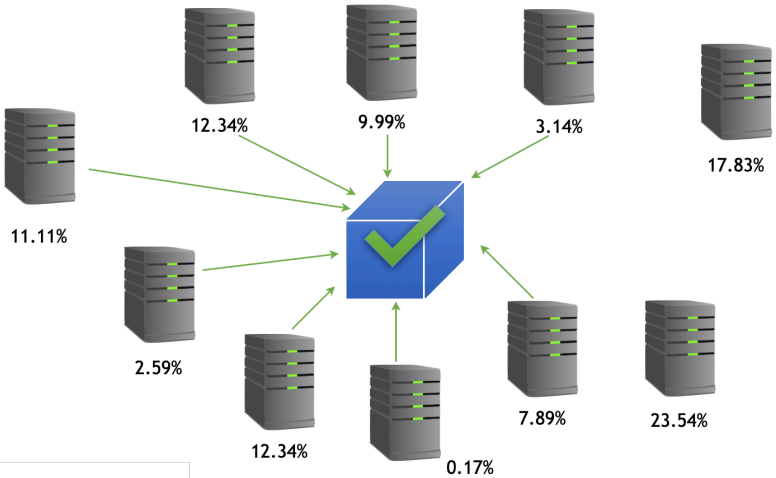
Running Time



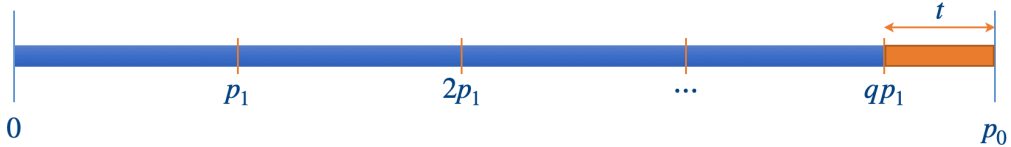
Proof Size



Questions?



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



Details

Proof Size and Running Time

- Current design optimizes proof size, $O\left(\log mn\right)$
- Can aggregate multiple proofs
- Running time is $O\left(mn\right)$, but with large constants
- With current implementation of Bulletproofs / R1CS, takes ~ hours to produce proof.

Shamir's Secret Sharing

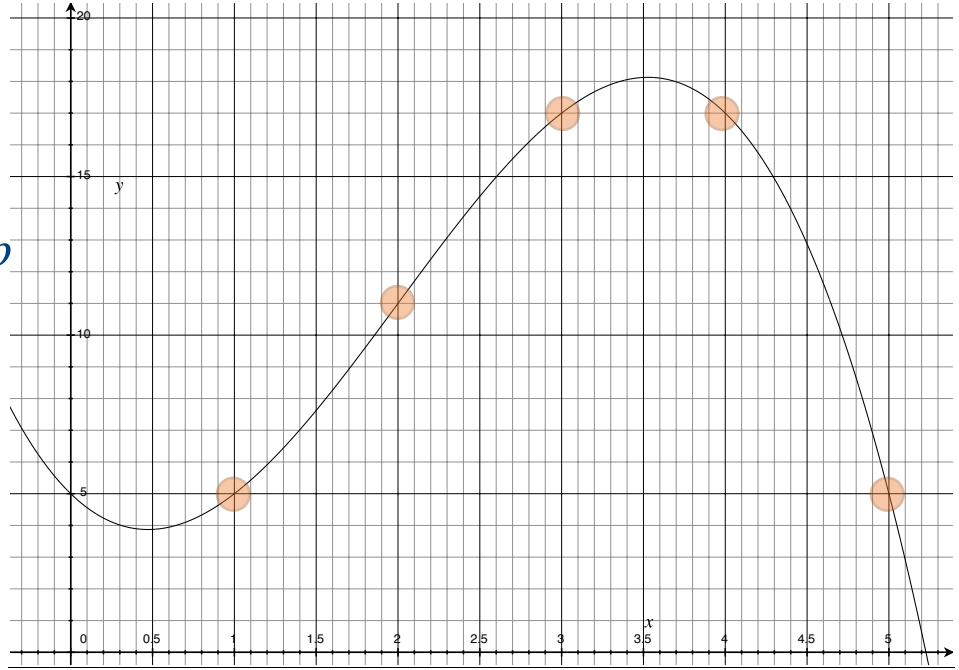
Secret s, Threshold t

1. Chooses $a_1, \ldots, a_t \xleftarrow{\$} \mathbb{Z}_p$ to define polynomial:

$$f(x) = s + a_1 x + a_2 x^2 + \ldots + a_t x^t \mod p$$

2. Evaluates $s_i = f(i)$ for $i \in [n]$
3. Sends s_i to party i

Recovery: Lagrange Interpolation



$$f(x) = 5 - 5x + 6x^2 - x^3$$

Shamir's Secret Sharing

Setup: choose some large prime p, work in $\mathbb{Z}_p$

Dealer: given a secret $s \in \mathbb{Z}_p$, threshold t

1. Chooses $a_1, \ldots, a_t \xleftarrow{\$} \mathbb{Z}_p$ to define polynomial:

$$f(x) = s + a_1 x + a_2 x^2 + \ldots + a_t x^t \mod p$$

2. Evaluates $s_i = f(i)$ for $i \in [n]$
3. Sends s_i to party i

Shamir's Secret Sharing - Reconstruction

Fact: poly of degree t uniquely determined by $t+1$ points

Lagrange Interpolation:

$$\ell_i = \prod_{j \neq i} \frac{x_j}{x_j - x_i}, \; f(0) = \sum_{i=0}^{t+1} \ell_i f_i = s$$

Verifiable Secret Sharing

- Homomorphic commitments: allow dealer to commit to a value and allow comparisons / operations on it.
- Dealer provides commitments to the secret and the coefficients.
- Participants check that all shares were calculated correctly, if not accuse the dealer.
- In practice slightly more complicated to avoid biasing problems.

Problems with SSS

- Pros:
  - Easy and flexible
  - Linear (makes it easy to use)
- Cons:
  - Every share has equal weight

CRT-Based Secret Sharing

CRT Based Secret Sharing

Given:

- secret $s_0 \in \mathbb{Z}_{p_0}$ that we want to share with n parties
- let $p_1, \ldots, p_n$ all be co-prime with each other and p_0

Sharing:

- choose $u \xleftarrow{\$} [L]$ and set $s = s_0 + up_0$
- $s_i = s \mod p_i$

CRT Based SS - Reconstruction

For some authorised set of parties A, have a system of equations:

$$\{s_i = s \mod p_i\}_{i \in A}$$

By CRT, there is a unique solution mod $P_A = \prod_{i \in A} p_i$

$$s' = \sum_{i \in A} \lambda_i s_i \mod P_A$$

If $P_A > s$ then $s = s'$

Weight-Ramp Scheme, Setup

- Each party i has weight $w_i \in \mathbb{N}$
- Reconstruction threshold T and privacy threshold t
  - Set of parties A is *authorised* if $\sum_{i \in A} w_i \geq T$
  - Set of parties $\bar{A}$ is *unauthorised* if $\sum_{i \in \bar{A}} w_i \leq t$
  - NB: Any other set is neither authorised nor unauthorised

CRT Based SS - Parameters

Let $\bar{\mathscr{A}}$ all unauthorised sets

Define: $P_{max} = \max_{\bar{A} \in \bar{\mathscr{A}}} P_{\bar{A}}$

and $\mathscr{A}$ be all authorised sets

Define: $P_{min} = \min_{A \in \mathscr{A}} P_A$

Security error:

$$P_{max}/L$$

For reconstruction:

$$s \leq (L+1)p_0 < P_{min}$$

$$P_{max} << L << P_{min}/p_0$$

CRT Solution / Partial Proof

Let $q_i = \prod_{j \neq i} p_i$ & $q_i' = (q_i)^{-1} \mod p_i$ & $\lambda_i = q_i q_i'$

$$a = \sum_{i=1}^{n} \lambda_i a_i \mod (p_1 \cdots p_n)$$

Why?

Notice that: $\lambda_i \mod p_j = 1$ if $i = j$ else 0

A Little Lemma

For arbitrary integers M, L, p, s s.t. $M < L$, and p, M co-prime

Let $U_L \xleftarrow{\$} [L], U_M \xleftarrow{\$} [M]$

Then:

$$SD((s_0 + U_L p_0) \mod M, U_M) < \frac{M}{L}$$

Where SD is the statistical distance, i.e. Total Variation Distance

Threshold Scheme

Easy to construct a scheme with threshold t:

- Pick $p_0, p_1, \ldots, p_n$ s.t. $||p_0|| = \lambda, ||p_i|| = 2\lambda \forall i \in [n]$
- Set: $L = 2^{2\lambda t - \lambda} - 1$

$$\left(P_{max} \approx 2^{2\lambda t - 2\lambda}\right) << \left(L = 2^{2\lambda t - \lambda} - 1\right) < \left(P_{min}/p_0 \approx 2^{2\lambda t - \lambda}\right)$$

Weight-Ramp Scheme

Pick p_i s.t. $||p_i|| = w_i$, $||p_0|| = \lambda$

$$P_A = \prod_{i \in A} p_i < 2^{\sum_{i \in A} w_i} \implies P_{max} < 2^t,\ P_{min} \geq 2^{T - O(1)}$$

Set $L = 2^{\lambda + t}$

Correct if: $(L+1)p_0 = 2^{2\lambda + t} < 2^{T - O(1)} \implies T - t > 2\lambda + O(1)$

Security: $\frac{P_{MAX}}{L} < \frac{2^t}{2^{\lambda + t}} = 2^{-\lambda}$

Can amplify by setting $||p_i|| = c w_i$ to get $c(T-t) > 2\lambda + O(1)$

Weight-Ramp Scheme Results

- Shares are now exactly w_i bits long
- Everything scales by weight exactly as you would expect
- If shares are unequal, should be much more efficient than SSS!

Weight-Ramp Scheme Problems

- Not linear!
- While it’s easy to reconstruct the secret, not clear how to build threshold crypto
- Some existing work on threshold ElGamal decryption and threshold signatures, but not as efficient as standard constructions

CRT-Based VSS

- Existing work on building CRT-Based VSS by Kaya et al
  - Uses RSA assumption and large groups of unknown order
  - Requires trusted setup
  - Inefficient because of the need for very large groups for security
- Not ideal for blockchains for reasons above.
- It would be great if we could do something in the elliptic-curve / discrete log setting using the tools that blockchains already use

But first… a few tools

Commitments

In general:

$$\text{Com}(x; r) \rightarrow c$$

Pedersen Commitments:

$$\text{Com}(x; r) \rightarrow g^x h^r$$

Binding: no adversary can find x_0, x_1, r_0, r_1 s.t. $x_0 \neq x_1$

$$\text{Com}(x_0, r_0) = \text{Com}(x_1, r_1)$$

Hiding: Adversary chooses x_0, x_1 given $\text{Com}(x_b; r)$, cannot guess b with non-negl prob.

Theorem: Pedersen Commitments are perfectly hiding and computationally binding under discrete log assumption.

Homomorphic Commitments

In general:

$$\mathsf{Com}(x_1; r_1) + \mathsf{Com}(x_2; r_2) = \mathsf{Com}(x_1 + x_2; r_1 + r_2)$$

Pedersen Commitments:

$$\left(g^{x_1} h^{r_1}\right) \cdot \left(g^{x_2} h^{r_2}\right) = \left(g^{x_1 + x_2} h^{r_1 + r_2}\right)$$

Range Proof (e.g. Bulletproofs)

Given a commitment to a value:

$$V = \mathsf{Com}(v; \gamma)$$

Want to prove to a verifier that:

$$v \in [0, 2^n - 1]$$

Bulletproofs: Zero-Knowledge Proof of Range with perfect completeness, perfect HVZK, and computational soundness with proof size $O(\log n)$

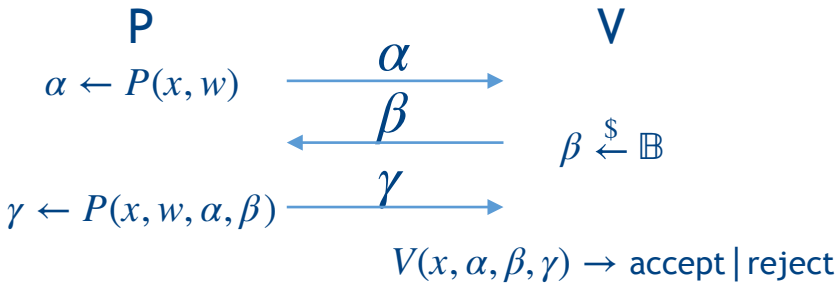
... provided $2^n << ||\mathbb{G}||$

ZKP For Disjunctions

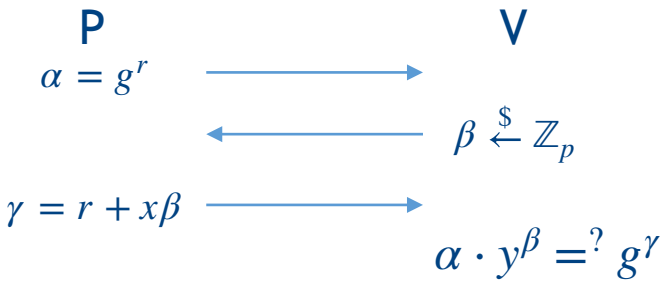
How can we prove the statement A OR B, without the verifier learning whether it's A or B that's true?

Assume we have public coin ZKP for both A and B each with 3 rounds:

General Form



Example: Schnorr



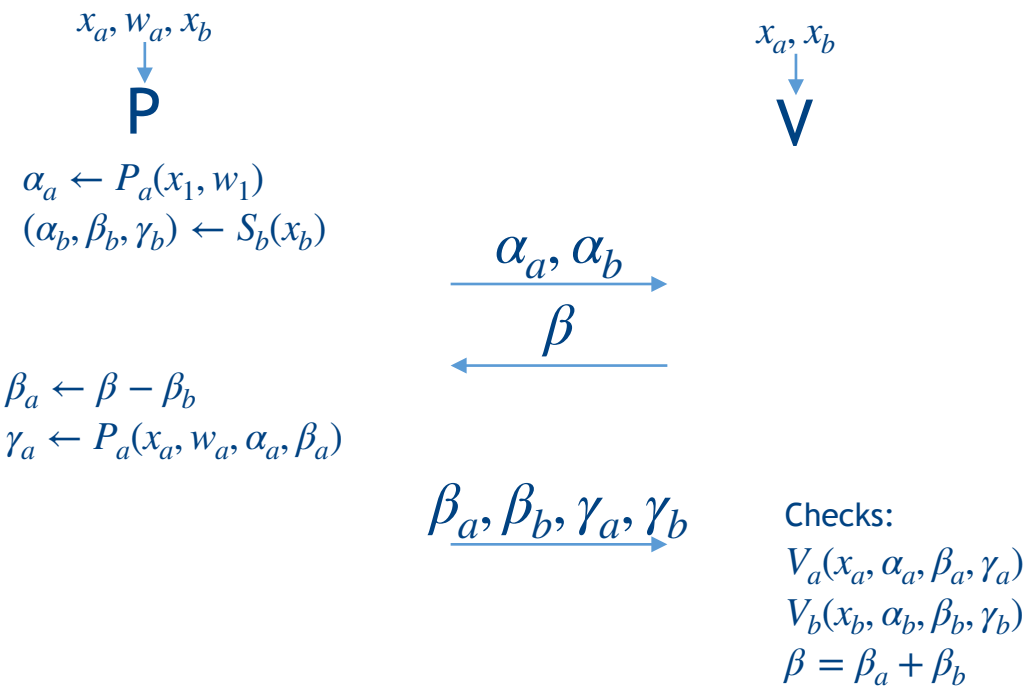
ZKP for Disjunctions

Key Insight: to have ZKP, must have simulators for A & B.

To prove $A \vee B$ in Zero-Knowledge (WLOG prove A):

1. Run actual prover for A, and simulator for B
2. Get only a single challenge from verifier, split up such that two challenges used by subproofs sum to the total challenge.
3. Now have two proofs that both pass verifier!

ZKP for Disjunctions



Verifiable CRT-Based Secret Sharing

CRT-VSS Goal

Dealer:

1. Given $s_0 \in \mathbb{Z}_{p_0}, u \in [L]$
2. Sends each party i value $s_i = s_0 + up_0 \mod p_i$

How can the dealer *prove* the deal was correct without revealing s_0, s_i?

Naïve answer:

1. Give commitments to $s_0, s_1, \ldots, s_n$
2. Provide commitments to $k_1, \ldots, k_n$
3. Prove that $s_0 + up_0 = s_i + k_i p_i$

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.

Simplified Problem: Proof-of-Mod in p_0

Start with an easier problem:

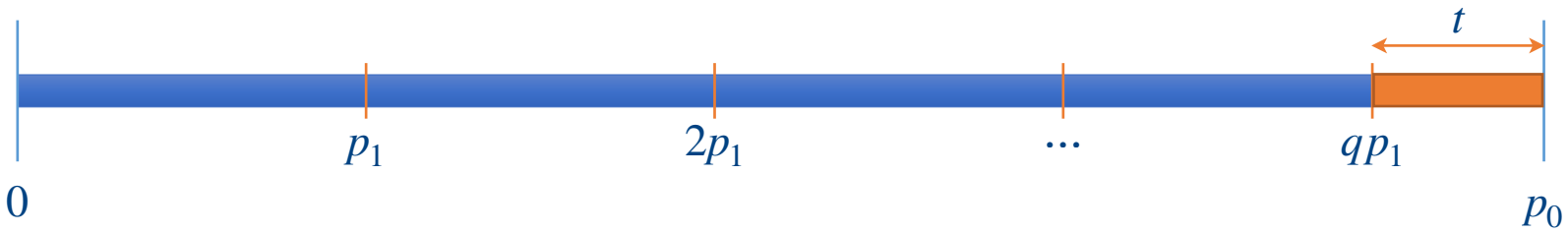
1. Assume the group we're using is *very* big, i.e. $p_1 << p_0$
2. Given two commitments, can we prove that one is the mod of the other?

Example: $A = \text{com}(a), B = \text{com}(b)$

Can we prove that $b = a \mod p_1$?

Wraparound $\mod p_0$

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" $b = a \mod p_1, a, b \in \mathbb{Z}_p$

Prover has a, b, sends verifier $A = \text{Com}(a; r_a), B = \text{Com}(b; r_b)$

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

1. Prover sends $V = \text{Com}(k; r_k)$
2. Prover sends proof that $b + kp_1 = a \mod p_0$ (see next slide)
3. Use disjunctive proof strategy on following statements:
   - A. $(0 \leq k < q) \wedge (0 \leq b < p_1)$ OR
   - B. $(0 \leq k \leq q) \wedge (0 \leq b < t)$

Both A and B above are just range proofs, can use Bulletproofs or others

With these in place, have a proof-of-mod, since $b + kp_1 < p_0$

Proof of $a = b + kp_1 \mod p_0$

Prover has a, b, k, r_a, r_b, r_k

Verifier has $A = \mathsf{Com}(a; r_a), B = \mathsf{Com}(b; r_b), V = \mathsf{Com}(k; r_k)$

All prover has to do is provide $r' = r_a - r_b - r_k p_1 \mod p_0$

Verifier checks (assuming Pedersen commitments):

$$A =^{?} \left(B \cdot V^{p_1}\right) \cdot h^{r'}$$

$$\left(B \cdot V^{p_1}\right) \cdot h^{r'} = \left(g^b h^{r_b} \cdot g^{kp_1} h^{r_k p_1}\right) h^{r_a - r_b - r_k p_1} = g^{b + kp_1} h^{r_a}$$

Expansion to larger values of s

Can we do better? What if we assume:

$$p_i << p_0 \quad \text{...but...} \quad p_0 < P_{max} << p_0^2 < P_{min}$$

In this case:

$$s = s_0 + ap_0, \text{ where } a \xleftarrow{\$} \mathbb{Z}_{p_0}$$

As before:

$$s_i = s \mod p_i$$

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:

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

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

Proof of mod for values $< p_0^2$

Let:

$$t = p_0 \mod p_1$$
$$s_0' = s_0 \mod p_1$$
$$a' = a \mod p_1$$

$$s_1 = s_0 + ap_0 \mod p_1$$
$$\downarrow$$
$$s_1 = s_0' + a't \mod p_1$$

Prover:

1. Gives commitments for s_0, s_1, a, s_0', a'
2. Proof of mod for s_0', a'
3. Proof of mod for s_1, or can show:
    1. $s_1 + k'' = s_0' + a't$, AND
    2. $k'' < p_1(p_1 + 1) < p_0$

Proof of mod for values $< p_0^2$

Let:

$$p_0 = qp_1 + r$$
$$s_0 = s_0' + k_0' p_1$$
$$u = u' + k_1' p_1$$
$$s_0 + up_0 = s_1 + kp_1$$

$$s_1 = s_0 + up_0 \mod p_1$$

$$s_1 = (s_0' + k_0' p_1) + (u' + k_1' p_1)(qp_1 + r) - kp_1$$
$$= (s_0' + u'r) + (k_0' + u'q + k_1' r + k_1' qp_1 - k)p_1$$

Let: $s_1 + k''p_1 = s_0' + u'r$

Proof of mod for values $< p_0^2$

Prover:

1. Provides commitments for $s_0, u, s_0', u', k_0', k_1', k''$
2. Proof of mod for s_0', u'
3. Range proof that $k'' < p_1$
4. Proof that $s_1 + k''p_i = s_0' + u'r$

Expansion to <u>even larger</u> values of s

Can we expand even more?? What if we assume:

$$p_i << p_0 \quad \text{...but...} \quad p_0 < P_{max} << p_0^m < P_{min}$$

In this case:

$$s = s_0 + a_1 p_0 + a_2 p_0^2 + \ldots + a_m p_0^m, \text{ where } a_1, \ldots, a_m \xleftarrow{\$} \mathbb{Z}_{p_0}$$

As before:

$$s_i = s \mod p_i$$

Proof of mod for values $< p_0^m$

Let:

$$t_j = p_0^j \mod p_1$$
$$s_0' = s_0 \mod p_1$$
$$a_j' = a_j \mod p_1$$

$$s_1 = s_0 + \sum_{j=1}^{m} a_j p_0^j \mod p_1$$

$$s_1 = s_0' + \sum_{j=1}^{m} a_j' t_j \mod p_1$$

Prover:

1. Gives commitments for $s_0', a_1', \ldots, a_m'$
2. Proof of mod for $s_0', a_1', \ldots, a_m'$
3. Proof of mod for s_1, or can show:
   1. $s_1 + k'' = s_0' + \sum_{j=1}^{m} a_j' t_j$
   2. $k'' < p_1(mp_1 + 1) < p_0$

Can split into chunks if needed

Proof of mod for values $< p_0^m$

Let:

$$p_0^j = q_j p_1 + r_j$$
$$s_0 = s_0' + k_0' p_1$$
$$a_j = a_j' + k_j' p_1$$
$$s_0 + u p_0 = s_1 + k p_1$$

$$s_1 = s_0 + \sum_{j=1}^{m} a_j p_0^j \mod p_1$$

$$s_1 = (s_0' + k_0' p_1) + \sum_{j=1}^{m} (a_j' + k_j' p_1)(q_j p_1 + r_j) - k p_1$$

$$= \left(s_0' + \sum_{j=1}^{m} a_j' r_j \right) + \left(k_0' - k + \sum_{j=1}^{m} a_j' q_j + k_j' r_j + k_j' q_j p_1 \right) p_1$$

Let: $s_1 + k'' p_1 = s_0' + \sum_{j=1}^{m} a_j' r_j$

Proof of mod for values $< p_0^m$

Prover:

1. Provides commitments for
   $s_0, s_0', k_0', a_1, \ldots, a_m, a_1', \ldots, a_m', k_1', \ldots, k_m', k''$
2. Proof of mod for $s_0', a_1', \ldots, a_m'$
3. Range proof that $k'' < mp_1 < p_0$
    1. If $mp_1 > p_0$, then break into chunks
4. Proof that $s_1 + k''p_1 = s_0' + \sum_{j=1}^{m} a_j' r_j$

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. Use the expanded proof-of-mod to prove correct dealing for each share

Limits of this approach

Security error: $= P_{max}/L = P_{max}/p_0^m$

$$\forall i, p_0 > p_i(mp_i + 1) \implies m < \min_{i \in [n]}(p_0/p_i^2 - 1/p_i)$$

(Or break things into chunks)

Next Steps

- Please help me check my work! This is all very early results
- Can we combine the proofs to make it more efficient, e.g. using polynomial evaluations or some other aggregation method?
- Can we extend this to arbitrary arithmetic circuits over the integers?
- How do we build a DKG with the correct distribution of keys?
  - Previous work I'm pretty sure is flawed

