Pairing-Based SNARKs. Pinocchio And Groth16

October 10, 2024

Distributed Lab

zkdl-camp.github.io
github.com/ZKDL-Camp

Plan

Recap

Recap. R1CS

Each **constraint** in the Rank-1 Constraint System must be in the form:

$$\langle \boldsymbol{a}, \boldsymbol{w} \rangle \times \langle \boldsymbol{b}, \boldsymbol{w} \rangle = \langle \boldsymbol{c}, \boldsymbol{w} \rangle$$

Recap. R1CS

Each **constraint** in the Rank-1 Constraint System must be in the form:

$$\langle \boldsymbol{a}, \boldsymbol{w} \rangle \times \langle \boldsymbol{b}, \boldsymbol{w} \rangle = \langle \boldsymbol{c}, \boldsymbol{w} \rangle$$

Where $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ is a dot product.

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle := \boldsymbol{u}^\top \boldsymbol{v} = \sum_{i=1}^{n} u_i v_i$$

Recap. R1CS

Each **constraint** in the Rank-1 Constraint System must be in the form:

$$\langle \boldsymbol{a}, \boldsymbol{w} \rangle \times \langle \boldsymbol{b}, \boldsymbol{w} \rangle = \langle \boldsymbol{c}, \boldsymbol{w} \rangle$$

Where $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ is a dot product.

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle := \boldsymbol{u}^\top \boldsymbol{v} = \sum_{i=1}^{n} u_i v_i$$

Thus

$$\left(\sum_{i=1}^{n} a_i w_i \right) \times \left(\sum_{j=1}^{n} b_j w_j \right) = \sum_{k=1}^{n} c_k w_k$$

That is, actually, a quadratic equation with multiple variables.

Recap. R1CS

Consider the simplest program:

```
def example(a: F, b: F, c: F) -> F:
    if a:
        return b * c
    else:
        return b + c
```

Recap. R1CS

$$r = x_1 \times (x_2 \times x_3) + (1 - x_1) \times (x_2 + x_3)$$

Recap. R1CS

$$r = x_1 \times (x_2 \times x_3) + (1 - x_1) \times (x_2 + x_3)$$

Thus, the next constraints can be build:

$$x_1 \times x_1 = x_1 \quad \text{(binary check)} \tag{1}$$

$$x_2 \times x_3 = \text{mult} \tag{2}$$

$$x_1 \times \text{mult} = \text{selectMult} \tag{3}$$

$$(1 - x_1) \times (x_2 + x_3) = r - \text{selectMult} \tag{4}$$

Recap. R1CS

$$r = x_1 \times (x_2 \times x_3) + (1 - x_1) \times (x_2 + x_3)$$

Thus, the next constraints can be build:

$$x_1 \times x_1 = x_1 \quad \text{(binary check)} \tag{1}$$
$$x_2 \times x_3 = \text{mult} \tag{2}$$
$$x_1 \times \text{mult} = \text{selectMult} \tag{3}$$
$$(1 - x_1) \times (x_2 + x_3) = r - \text{selectMult} \tag{4}$$

The witness vector: $\boldsymbol{w} = (1, r, x_1, x_2, x_3, \text{mult}, \text{selectMult})$.

Recap. R1CS

$$r = x_1 \times (x_2 \times x_3) + (1 - x_1) \times (x_2 + x_3)$$

Thus, the next constraints can be build:

$$x_1 \times x_1 = x_1 \quad \text{(binary check)} \tag{1}$$
$$x_2 \times x_3 = \text{mult} \tag{2}$$
$$x_1 \times \text{mult} = \text{selectMult} \tag{3}$$
$$(1 - x_1) \times (x_2 + x_3) = r - \text{selectMult} \tag{4}$$

The witness vector: $\boldsymbol{w} = (1, r, x_1, x_2, x_3, \text{mult}, \text{selectMult})$.

The coefficients vectors:

$$\begin{aligned}
\boldsymbol{a}_1 &= (0,0,1,0,0,0,0), & \boldsymbol{b}_1 &= (0,0,1,0,0,0,0), & \boldsymbol{c}_1 &= (0,0,1,0,0,0,0) \\
\boldsymbol{a}_2 &= (0,0,0,1,0,0,0), & \boldsymbol{b}_2 &= (0,0,0,0,1,0,0), & \boldsymbol{c}_2 &= (0,0,0,0,0,1,0) \\
\boldsymbol{a}_3 &= (0,0,1,0,0,0,0), & \boldsymbol{b}_3 &= (0,0,0,0,0,1,0), & \boldsymbol{c}_3 &= (0,0,0,0,0,0,1) \\
\boldsymbol{a}_4 &= (1,0,-1,0,0,0,0), & \boldsymbol{b}_4 &= (0,0,0,1,1,0,0), & \boldsymbol{c}_4 &= (0,1,0,0,0,0,-1)
\end{aligned}$$

Recap. QAP

R1CS provides us with the following constraint vectors:

$$\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_m, \quad \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_m, \quad \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_m,$$

Recap. QAP

R1CS provides us with the following constraint vectors:

$$\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_m, \quad \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_m, \quad \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_m,$$

Of course, they form corresponding matrices:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \text{ same goes for } B \text{ and } C$$

Recap. QAP

R1CS provides us with the following constraint vectors:

$$\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_m, \quad \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_m, \quad \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_m,$$

Of course, they form corresponding matrices:

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}, \text{ same goes for } B \text{ and } C$$

An example of a single “if“ statement:

$$\begin{aligned} \boldsymbol{a}_1 &= (0, 0, 1, 0, 0, 0, 0) \\ \boldsymbol{a}_2 &= (0, 0, 0, 1, 0, 0, 0) \\ \boldsymbol{a}_3 &= (0, 0, 1, 0, 0, 0, 0) \\ \boldsymbol{a}_4 &= (1, 0, -1, 0, 0, 0, 0) \end{aligned}$$

$$\begin{array}{cc} & \begin{array}{ccccccc} & & 3 & & & & \end{array} \\ \begin{array}{c} \\ \\ \\ 4 \end{array} & \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & \boxed{-1} & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Recap. QAP

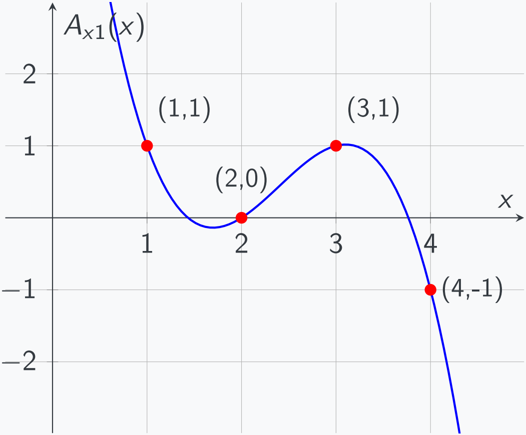


Illustration: The Lagrange inteprolation polynomial for points $\{(1,1),(2,0),(3,1),(4,-1)\}$ visualized over $\mathbb{R}$.

Recap. QAP

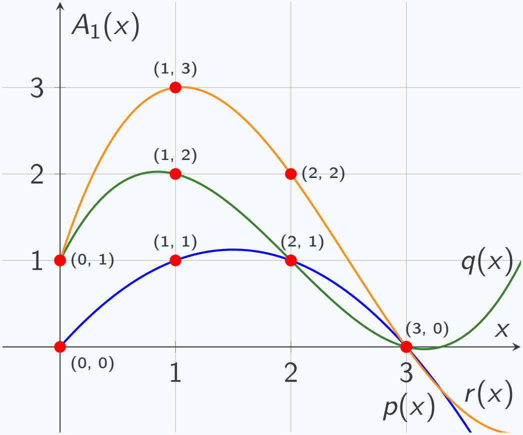


Figure: Addition of two polynomials

Now, using coefficients encoded with polynomials, we can build a constraint number $X \in \{1, \ldots\ m\}$ in the next way:

$$
\begin{aligned}
&(w_1 A_1(X) + w_2 A_2(X) + \cdots + w_n A_n(X)) \times \\
\times &(w_1 B_1(X) + w_2 B_2(X) + \cdots + w_n B_n(X)) = \\
= &(w_1 C_1(X) + w_2 C_2(X) + \cdots + w_n C_n(X))
\end{aligned}
$$

Now, using coefficients encoded with polynomials, we can build a constraint number $X \in \{1, \dots\ m\}$ in the next way:

$$
\begin{aligned}
&(w_1 A_1(X) + w_2 A_2(X) + \cdots + w_n A_n(X)) \times \\
\times &(w_1 B_1(X) + w_2 B_2(X) + \cdots + w_n B_n(X)) = \\
= &(w_1 C_1(X) + w_2 C_2(X) + \cdots + w_n C_n(X))
\end{aligned}
$$

Or written more concisely:

$$
\left(\sum_{i=1}^{n} w_i A_i(X) \right) \times \left(\sum_{i=1}^{n} w_i B_i(X) \right) = \left(\sum_{i=1}^{n} w_i C_i(X) \right)
$$

$$
A(X) \times B(X) = C(X)
$$

Recap. QAP

Now, we can define a polynomial $M(X)$, that has zeros at all elements from the set $\Omega = \{1, \ldots, m\}$

$$M(X) = A(X) \times B(X) - C(X)$$

Recap. QAP

Now, we can define a polynomial $M(X)$, that has zeros at all elements from the set $\Omega = \{1, \ldots, m\}$

$$M(X) = A(X) \times B(X) - C(X)$$

It means, that $M(X)$ can be divided by **vanishing polynomial** $Z_\Omega(X)$ without a remainder!

$$Z_\Omega(X) = \prod_{i=1}^{m}(X - i), \qquad H(X) = \frac{M(X)}{Z_\Omega(X)} \text{ is a polynomial}$$

Encrypted Verification

Current Point

We've managed to encode into **a single polynomial** an entire computation (a program), of any size, independent of how much data it consumes.

Current Point

We've managed to encode into **a single polynomial** an entire computation (a program), of any size, independent of how much data it consumes.

Now, we need to figure our the protocol, how a prover can succinctly proof the knowledge of a correct witness for some circuit to a verifier, additionally, make it zero-knowledge and non-interactive.

Current Point

We've managed to encode into **a single polynomial** an entire computation (a program), of any size, independent of how much data it consumes.

Now, we need to figure our the protocol, how a prover can succinctly proof the knowledge of a correct witness for some circuit to a verifier, additionally, make it zero-knowledge and non-interactive.

Where the knowledge of the correct witness is a knowledge of the quotient polynomial $H(X)$.

$$M(X) = H(X) \times Z_{\Omega}(X)$$

Notation Preliminaries

In this section we'll use a group of points on elliptic curve denoted as $\mathbb{G}$ of prime order q with a generator g.

Notation Preliminaries

In this section we'll use a group of points on elliptic curve denoted as $\mathbb{G}$ of prime order q with a generator g.

The symmetric pairing function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $(\mathbb{G}_T, \times)$ is a target group.

Naive Proof

Suppose, we are given a circuit $\mathcal{C}$ with a maximum degree d of polynomials used underneath.

Thus, all parties additionally know the target polynomial $Z(x)$ and QAP polynomials $\{L_i(x)\}_{i\in[n]}, \{R_i(x)\}_{i\in[n]}, \{O_i(x)\}_{i\in[n]}$, where n is number of witness elements.

Naive Proof

Suppose, we are given a circuit $\mathcal{C}$ with a maximum degree d of polynomials used underneath.

Thus, all parties additionally know the target polynomial $Z(x)$ and QAP polynomials $\{L_i(x)\}_{i\in[n]}, \{R_i(x)\}_{i\in[n]}, \{O_i(x)\}_{i\in[n]}$, where n is number of witness elements.

Prover

✓ Provides witness $\boldsymbol{w}$ to a Verifier.

Naive Proof

Suppose, we are given a circuit $\mathcal{C}$ with a maximum degree d of polynomials used underneath.

Thus, all parties additionally know the target polynomial $Z(x)$ and QAP polynomials $\{L_i(x)\}_{i\in[n]}, \{R_i(x)\}_{i\in[n]}, \{O_i(x)\}_{i\in[n]}$, where n is number of witness elements.

Prover

✓ Provides witness $\boldsymbol{w}$ to a Verifier.

Verifier

✓ Checks $(\sum_{i=1}^{n} w_i A_i(X)) \times (\sum_{i=1}^{n} w_i B_i(X)) = (\sum_{i=1}^{n} w_i C_i(X))$

Naive Proof

- ✗ Succint
- ✓ Non-Interactive
- ✗ Zero-Knowledge

Naive Proof

- ✗ Succint
- ✓ Non-Interactive
- ✗ Zero-Knowledge

The verifier could actually just run a program that represents a circuit $\mathcal{C}$ on witness data $\boldsymbol{w}$.

Naive Proof

- ✗ Succint
- ✓ Non-Interactive
- ✗ Zero-Knowledge

The verifier could actually just run a program that represents a circuit $\mathcal{C}$ on witness data $\boldsymbol{w}$.

We, definitely, need to encrypt the witness data $\boldsymbol{w}$ somehow...

Let's define the *encryption* operation as follows:

$$\mathsf{Enc} : \mathbb{F} \to \mathbb{G}, \quad \mathsf{Enc}(x) := g^x$$

Let's define the *encryption* operation as follows:

$$\mathsf{Enc} : \mathbb{F} \to \mathbb{G}, \quad \mathsf{Enc}(x) := g^x$$

Essentially, $\mathsf{Enc}(p(\tau))$ is the **KZG Commitment**.

Example

Consider the polynomial: $p(x) = x^2 - 5x + 2$, the encryption of $p(\tau)$:

$$\mathsf{Enc}(p(\tau)) = g^{p(\tau)} = g^{\left(\tau^2 - 5\tau + 2\right)} = \left(g^{\tau^2}\right)^1 \cdot \left(g^{\tau^1}\right)^{-5} \cdot \left(g^{\tau^0}\right)^2$$

Let's define the *encryption* operation as follows:

$$\mathrm{Enc} : \mathbb{F} \to \mathbb{G}, \quad \mathrm{Enc}(x) := g^x$$

Essentially, $\mathrm{Enc}(p(\tau))$ is the **KZG Commitment**.

Example

Consider the polynomial: $p(x) = x^2 - 5x + 2$, the encryption of $p(\tau)$:

$$\mathrm{Enc}(p(\tau)) = g^{p(\tau)} = g^{\left(\tau^2 - 5\tau + 2\right)} = \left(g^{\tau^2}\right)^1 \cdot \left(g^{\tau^1}\right)^{-5} \cdot \left(g^{\tau^0}\right)^2$$

Question

KZG Commitment requires encrypted powers of τ: $\{g^{\tau^i}\}_{i \in [d]}$. But where the prover can take them?

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

✓ Calculates the public parameters $\{g^{\tau^i}\}_{i \in [d]}$.

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

✓ Calculates the public parameters $\{g^{\tau^i}\}_{i\in[d]}$.

✓ **Deletes** τ (toxic waste).

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

✓ Calculates the public parameters $\{g^{\tau^i}\}_{i \in [d]}$.

✓ **Deletes** τ (toxic waste).

✓ **Outputs** prover parameters $\mathsf{pp} \leftarrow \{g^{\tau^i}\}_{i \in [d]}$ and verifier parameters $\mathsf{vp} \leftarrow \mathsf{com}(Z)$.

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

✓ Calculates the public parameters $\{g^{\tau^i}\}_{i\in[d]}$.

✓ **Deletes** τ (toxic waste).

✓ **Outputs** prover parameters $\mathsf{pp} \leftarrow \{g^{\tau^i}\}_{i\in[d]}$ and verifier parameters $\mathsf{vp} \leftarrow \mathsf{com}(Z)$.

Trusted Setup

Trusted Party Setup

✓ Picks a random value $\tau \xleftarrow{R} \mathbb{F}$.

✓ Calculates the public parameters $\{g^{\tau^i}\}_{i\in[d]}$.

✓ **Deletes** τ (toxic waste).

✓ **Outputs** prover parameters $\mathsf{pp} \leftarrow \{g^{\tau^i}\}_{i\in[d]}$ and verifier parameters $\mathsf{vp} \leftarrow \mathsf{com}(Z)$.

This way, we can find the KZG commitment for each polynomial. For example:

$$\mathsf{com}(L) \triangleq g^{L(\tau)} = g^{\sum_{i=0}^{d} L_i \tau^i} = \prod_{i=0}^{d} (g^{\tau^i})^{L_i},$$

Now, we can calculate:

$$g^{L(\tau)}, g^{R(\tau)}, g^{O(\tau)}, g^{H(\tau)}, g^{Z(\tau)}$$

But how can we verify $H(x)Z(x) = L(x)R(x) - O(x)$ in the ecnrypted space?

Well, first notice that the check is equivalent to:

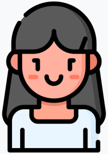
$$L(\tau)R(\tau) = Z(\tau)H(\tau) + O(\tau).$$

So, we can check this equality as follows:

$$e(\text{com}(L), \text{com}(R)) = e(\text{com}(Z), \text{com}(H)) \cdot e(\text{com}(O), g),$$

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

Prover $\mathcal{P}$

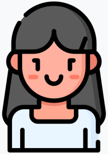


Verifier $\mathcal{V}$

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, **delete**(τ).

✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, **delete**(τ).

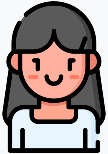
✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:

$\pi_L \leftarrow \mathrm{com}(L)$, $\pi_R \leftarrow \mathrm{com}(R)$,

$\pi_O \leftarrow \mathrm{com}(O)$, $\pi_H \leftarrow \mathrm{com}(H)$,

Prover $\mathcal{P}$

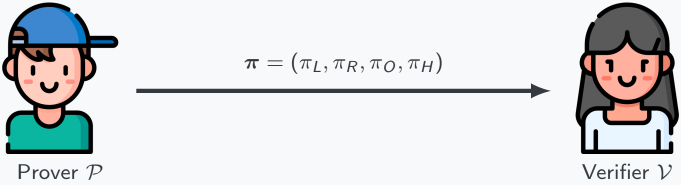


Verifier $\mathcal{V}$

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, $\textbf{delete}(\tau)$.

✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:
$\pi_L \leftarrow \mathrm{com}(L)$, $\pi_R \leftarrow \mathrm{com}(R)$,
$\pi_O \leftarrow \mathrm{com}(O)$, $\pi_H \leftarrow \mathrm{com}(H)$,



Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

✓ $H(x) = \frac{L(x) \times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:

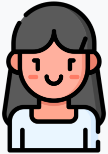
$\pi_L \leftarrow \text{com}(L)$, $\pi_R \leftarrow \text{com}(R)$,

$\pi_O \leftarrow \text{com}(O)$, $\pi_H \leftarrow \text{com}(H)$,

✓ $e(\pi_L, \pi_R) ==$
$e(\text{com}(Z), \pi_H) \cdot e(\pi_O, g)$.

Prover $\mathcal{P}$

$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H)$



Verifier $\mathcal{V}$

- ✓ Succint
- ✓ Non-Interactive
- ✓ Zero-Knowledge

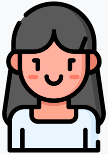
- ✓ Succint
- ✓ Non-Interactive
- ✓ Zero-Knowledge

- ✗ Does it work?

Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, **delete**(τ).

Prover $\mathcal{P}$

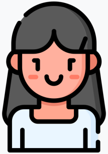


Verifier $\mathcal{V}$

Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Problem

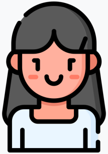
Prover isn't forced to use the values from the trusted setup.

Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

✓ $H'(x) \xleftarrow{R} \mathbb{F}[x]$, $M'(x) = Z(x) \times H'(x)$.

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Problem

Prover isn't forced to use the values from the trusted setup.

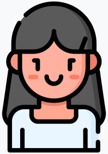
Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

✓ $H'(x) \xleftarrow{R} \mathbb{F}[x]$, $M'(x) = Z(x) \times H'(x)$.

✓ Finds $L'(x), R'(x), O'(x)$ such that:
$L'(x) \times R'(x) - O'(x) = M'(x)(x)$

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Problem

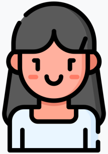
Prover isn't forced to use the values from the trusted setup.

Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, **delete**(τ).

- ✓ $H'(x) \xleftarrow{R} \mathbb{F}[x]$, $M'(x) = Z(x) \times H'(x)$.
- ✓ Finds $L'(x), R'(x), O'(x)$ such that:
  $L'(x) \times R'(x) - O'(x) = M'(x)(x)$
- ✓ KZG commitments:
  $\pi_{L'(x)} \leftarrow \mathsf{com}(L'(x))$, $\pi_{R'(x)} \leftarrow \mathsf{com}(R'(x))$,
  $\pi_{O'(x)} \leftarrow \mathsf{com}(O'(x))$, $\pi_{H'(x)} \leftarrow \mathsf{com}(H'(x))$,

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Problem

Prover isn't forced to use the values from the trusted setup.

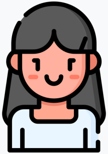
Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i\in[d]}$, $g^{Z(\tau)}$, **delete**(τ).

- ✓ $H'(x) \xleftarrow{R} \mathbb{F}[x]$, $\quad M'(x) = Z(x) \times H'(x)$.
- ✓ Finds $L'(x), R'(x), O'(x)$ such that:
  $L'(x) \times R'(x) - O'(x) = M'(x)(x)$
- ✓ KZG commitments:
  $\pi_{L'(x)} \leftarrow \text{com}(L'(x)), \quad \pi_{R'(x)} \leftarrow \text{com}(R'(x)),$
  $\pi_{O'(x)} \leftarrow \text{com}(O'(x)), \quad \pi_{H'(x)} \leftarrow \text{com}(H'(x)),$

Prover $\mathcal{P}$

$$\xrightarrow{\boldsymbol{\pi} = (\pi_{L'(x)}, \pi_{R'(x)}, \pi_{O'(x)}, \pi_{H'(x)})}$$



Verifier $\mathcal{V}$

Problem

Prover isn't forced to use the values from the trusted setup.

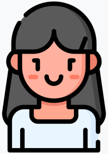
Why it doesn't work??

Trusted Setup: $\tau \xleftarrow{R} \mathbb{F}$, $\{g^{\tau^i}\}_{i \in [d]}$, $g^{Z(\tau)}$, **delete**(τ).

- ✓ $H'(x) \xleftarrow{R} \mathbb{F}[x]$, $\quad M'(x) = Z(x) \times H'(x)$.
- ✓ Finds $L'(x), R'(x), O'(x)$ such that:
  $L'(x) \times R'(x) - O'(x) = M'(x)(x)$
- ✓ KZG commitments:
  $\pi_{L'(x)} \leftarrow \mathrm{com}(L'(x))$, $\quad \pi_{R'(x)} \leftarrow \mathrm{com}(R'(x))$,
  $\pi_{O'(x)} \leftarrow \mathrm{com}(O'(x))$, $\quad \pi_{H'(x)} \leftarrow \mathrm{com}(H'(x))$,

✓ $e(\pi_{L'(x)}, \pi_{R'(x)}) ==$
$e(\mathrm{com}(Z), \pi_H) \cdot e(\pi_{O'(x)}, g)$.

$\boldsymbol{\pi} = (\pi_{L'(x)}, \pi_{R'(x)}, \pi_{O'(x)}, \pi_{H'(x)})$



Prover $\mathcal{P}$ Verifier $\mathcal{V}$

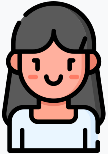
Problem

Prover isn't forced to use the values from the trusted setup.

Proof Of Exponent

Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

Prover $\mathcal{P}$

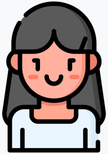


Verifier $\mathcal{V}$

Proof Of Exponent

Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

✓ $H(x) = \frac{L(x) \times R(x) - O(x)}{Z(x)}$.



Proof Of Exponent

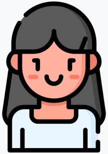
Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

✓ $H(x) = \frac{L(x) \times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Proof Of Exponent

Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

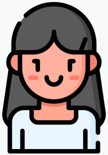
✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:

$$
\begin{aligned}
\pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\
\pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\
\pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\
\pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}.
\end{aligned}
$$

Prover $\mathcal{P}$

$$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H) \longrightarrow$$



Verifier $\mathcal{V}$

Proof Of Exponent

Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

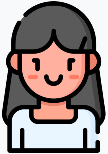
✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

✓ $e(\pi_L, \pi_R) ==$
$e(\mathbf{com}(Z), \pi_H) \cdot e(\pi_O, g)$.

$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H)$

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Proof Of Exponent

Trusted Setup: $\tau, \alpha \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}\}$, $\{g^{Z(\tau)}, g^{\alpha}\}$, **delete**$(\tau, \alpha)$.

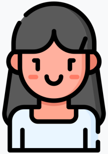
- ✓ $H(x) = \frac{L(x) \times R(x) - O(x)}{Z(x)}$.
- ✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

- ✓ $e(\pi_L, \pi_R) == e(\mathsf{com}(Z), \pi_H) \cdot e(\pi_O, g)$.
- ✓ Proof of Exponent:

$$\begin{aligned} e(\pi_L, g^{\alpha}) &= e(\pi'_L, g), \\ e(\pi_R, g^{\alpha}) &= e(\pi'_R, g), \\ e(\pi_O, g^{\alpha}) &= e(\pi'_O, g), \\ e(\pi_H, g^{\alpha}) &= e(\pi'_H, g). \end{aligned}$$

$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H)$



Prover $\mathcal{P}$ Verifier $\mathcal{V}$

Including PoE

- ✓ Succint
- ✓ Non-Interactive
- ✓ Zero-Knowledge

Including PoE

- ✓ Succint
- ✓ Non-Interactive
- ✓ Zero-Knowledge
- ✗ Sound

Including PoE

- ✓ Succint
- ✓ Non-Interactive
- ✓ Zero-Knowledge
- ✗ Sound

Problem

There is no guarantee that the same witness w was used to calculate all the commitments $\pi_L, \pi_R, \pi_O, \pi_H$.

Make It Sound

Additional Optimization

Recal that:

$$L(x) = \sum_{i=0}^{n} w_i L_i(x), \quad R(x) = \sum_{i=0}^{n} w_i R_i(x), \quad O(x) = \sum_{i=0}^{n} w_i O_i(x).$$

Additional Optimization

Recal that:

$$L(x) = \sum_{i=0}^{n} w_i L_i(x), \quad R(x) = \sum_{i=0}^{n} w_i R_i(x), \quad O(x) = \sum_{i=0}^{n} w_i O_i(x).$$

Here public data is:

$$\{L_i(x)\}_{i\in[n]}, \{R_i(x)\}_{i\in[n]}, \{O_i(x)\}_{i\in[n]}$$

Additional Optimization

Recal that:

$$L(x) = \sum_{i=0}^{n} w_i L_i(x), \quad R(x) = \sum_{i=0}^{n} w_i R_i(x), \quad O(x) = \sum_{i=0}^{n} w_i O_i(x).$$

Here public data is:

$$\{L_i(x)\}_{i \in [n]}, \{R_i(x)\}_{i \in [n]}, \{O_i(x)\}_{i \in [n]}$$

Moreover, it's defined only by the circuit and trusted setup, thus, it can calculated before proof generation as a part of the trusted setup.

Additional Optimization

Updated Trusted Setup:

$$\{g^{\tau^i}\}_{i\in[d]}, \quad \{g^{\alpha\tau^i}\}_{i\in[d]},$$
$$\{g^{L_i(\tau)}\}_{i\in[n]}, \quad \{g^{\alpha L_i(\tau)}\}_{i\in[n]},$$
$$\{g^{R_i(\tau)}\}_{i\in[n]}, \quad \{g^{\alpha R_i(\tau)}\}_{i\in[n]},$$
$$\{g^{O_i(\tau)}\}_{i\in[n]}, \quad \{g^{\alpha O_i(\tau)}\}_{i\in[n]}$$

Additional Optimization

Updated Trusted Setup:

$$\begin{aligned} &\{g^{\tau^i}\}_{i\in[d]}, && \{g^{\alpha\tau^i}\}_{i\in[d]}, \\ &\{g^{L_i(\tau)}\}_{i\in[n]}, && \{g^{\alpha L_i(\tau)}\}_{i\in[n]}, \\ &\{g^{R_i(\tau)}\}_{i\in[n]}, && \{g^{\alpha R_i(\tau)}\}_{i\in[n]}, \\ &\{g^{O_i(\tau)}\}_{i\in[n]}, && \{g^{\alpha O_i(\tau)}\}_{i\in[n]} \end{aligned}$$

Consider the polynomial $L(x) = \sum_{i=0}^{n} w_i L_i(x)$.

Additional Optimization

Updated Trusted Setup:

$$
\begin{array}{ll}
\{g^{\tau^i}\}_{i\in[d]}, & \{g^{\alpha\tau^i}\}_{i\in[d]}, \\
\{g^{L_i(\tau)}\}_{i\in[n]}, & \{g^{\alpha L_i(\tau)}\}_{i\in[n]}, \\
\{g^{R_i(\tau)}\}_{i\in[n]}, & \{g^{\alpha R_i(\tau)}\}_{i\in[n]}, \\
\{g^{O_i(\tau)}\}_{i\in[n]}, & \{g^{\alpha O_i(\tau)}\}_{i\in[n]}
\end{array}
$$

Consider the polynomial $L(x) = \sum_{i=0}^{n} w_i L_i(x)$.

$\mathcal{P}$ can compute the KZG commitment π_L and its PoE π'_L as follows:

$$
\pi_L \triangleq g^{L(\tau)} = g^{\sum_{i=0}^{n} w_i L_i(\tau)} = \prod_{i=0}^{n} (g^{L_i(\tau)})^{w_i},
$$

$$
\pi'_L \triangleq g^{\alpha L(\tau)} = g^{\alpha \sum_{i=0}^{n} w_i L_i(\tau)} = \prod_{i=0}^{n} (g^{\alpha L_i(\tau)})^{w_i}.
$$

Witness Consistency Check

Witness Consistency Check

To prove that the same w is used in all commitments, we need some "checksum" term that will somehow combine all polynomials $L(x)$, $R(x)$, and $O(x)$ with the witness w.

Witness Consistency Check

To prove that the same w is used in all commitments, we need some "checksum" term that will somehow combine all polynomials $L(x)$, $R(x)$, and $O(x)$ with the witness w.

We can do so by proving the next simple statement:

$$g^{L(\tau)+R(\tau)+O(\tau)} =$$

Witness Consistency Check

To prove that the same w is used in all commitments, we need some "checksum" term that will somehow combine all polynomials $L(x)$, $R(x)$, and $O(x)$ with the witness w.

We can do so by proving the next simple statement:

$$g^{L(\tau)+R(\tau)+O(\tau)} = \prod_{i=1}^{n} \left(g^{L_i(\tau)+R_i(\tau)+O_i(\tau)}\right)^{w_i}$$

Witness Consistency Check

To prove that the same w is used in all commitments, we need some "checksum" term that will somehow combine all polynomials $L(x)$, $R(x)$, and $O(x)$ with the witness w.

We can do so by proving the next simple statement:

$$g^{L(\tau)+R(\tau)+O(\tau)} = \prod_{i=1}^{n} \left(g^{L_i(\tau)+R_i(\tau)+O_i(\tau)}\right)^{w_i}$$

And we already know how to do that - POE!

Witness Consistency Check

Let's introduce one more coefficient...

$$\beta \xleftarrow{R} \mathbb{F}$$

Witness Consistency Check

Let's introduce one more coefficient...

$$\beta \xleftarrow{R} \mathbb{F}$$

Extended trusted setup contains additional values:

$$g^{\beta}, \quad \{g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))}\}_{i\in[n]}$$

Witness Consistency Check

Let's introduce one more coefficient...

$$\beta \xleftarrow{R} \mathbb{F}$$

Extended trusted setup contains additional values:

$$g^{\beta}, \quad \{g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))}\}_{i\in[n]}$$

Prover needs to calculate π_β:

$$\pi_\beta \leftarrow g^{\beta(L(\tau)+R(\tau)+O(\tau))} =$$

Witness Consistency Check

Let's introduce one more coefficient...

$$\beta \xleftarrow{R} \mathbb{F}$$

Extended trusted setup contains additional values:

$$g^{\beta}, \quad \{g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))}\}_{i\in[n]}$$

Prover needs to calculate π_β:

$$\pi_\beta \leftarrow g^{\beta(L(\tau)+R(\tau)+O(\tau))} = \prod_{i=1}^{n} g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))} g^{w_i}$$

Witness Consistency Check

Let's introduce one more coefficient...

$$\beta \xleftarrow{R} \mathbb{F}$$

Extended trusted setup contains additional values:

$$g^{\beta}, \quad \{g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))}\}_{i\in[n]}$$

Prover needs to calculate π_β:

$$\pi_\beta \leftarrow g^{\beta(L(\tau)+R(\tau)+O(\tau))} = \prod_{i=1}^{n} g^{\beta(L_i(\tau)+R_i(\tau)+O_i(\tau))} g^{w_i}$$

And easy check for verifier:

$$e(\pi_L \pi_R \pi_O, g^{\beta}) = e(\pi_\beta, g).$$

One more time... that doesn't work

If the witness is consistent, the following condition must hold:

$$(w_{L,i}L_i(\tau)+w_{R,i}R_i(\tau)+w_{O,i}O_i(\tau))\beta = w_i\beta(L_i(\tau)+R_i(\tau)+O_i(\tau)) \quad \forall i \in [n]$$

One more time... that doesn't work

If the witness is consistent, the following condition must hold:

$$(w_{L,i}L_i(\tau)+w_{R,i}R_i(\tau)+w_{O,i}O_i(\tau))\beta = w_i\beta(L_i(\tau)+R_i(\tau)+O_i(\tau)) \quad \forall i \in [n]$$

But, what if $L_i \equiv R_i$. Let's call them q, thus:

$$(w_{L,i} + w_{R,i})q + w_{O,i}O_i(\tau) = w_{\beta,i}(2q + O_i(\tau)) \quad \forall i \in [n]$$

One more time... that doesn't work

If the witness is consistent, the following condition must hold:

$$(w_{L,i}L_i(\tau)+w_{R,i}R_i(\tau)+w_{O,i}O_i(\tau))\beta = w_i\beta(L_i(\tau)+R_i(\tau)+O_i(\tau)) \quad \forall i \in [n]$$

But, what if $L_i \equiv R_i$. Let's call them q, thus:

$$(w_{L,i} + w_{R,i})q + w_{O,i}O_i(\tau) = w_{\beta,i}(2q + O_i(\tau)) \quad \forall i \in [n]$$

The adversary can choose $w_{L,i}$, $w_{R,i}$ and $w_{O,i}$ such that:

$$w_i := w_{O,i} \quad \text{and} \quad w_{L,i} = 2w_{O,i} - w_{R,i}$$

One more time... that doesn't work

If the witness is consistent, the following condition must hold:

$$(w_{L,i}L_i(\tau)+w_{R,i}R_i(\tau)+w_{O,i}O_i(\tau))\beta = w_i\beta(L_i(\tau)+R_i(\tau)+O_i(\tau)) \quad \forall i \in [n]$$

But, what if $L_i \equiv R_i$. Let's call them q, thus:

$$(w_{L,i} + w_{R,i})q + w_{O,i}O_i(\tau) = w_{\beta,i}(2q + O_i(\tau)) \quad \forall i \in [n]$$

The adversary can choose $w_{L,i}$, $w_{R,i}$ and $w_{O,i}$ such that:

$$w_i := w_{O,i} \quad \text{and} \quad w_{L,i} = 2w_{O,i} - w_{R,i}$$

Example

$$w = w_O = 5, \quad w_L = 7, \quad w_R = 3$$

$$(7+3)q + 5O(\tau) = 5(2q + O(\tau))$$
$$10q + 5O(\tau) = 10q + 5O(\tau)$$

More coefficients!

The main problem is that if $R_i \equiv L_i$ then $\beta R_i \equiv \beta L_i$.

More coefficients!

The main problem is that if $R_i \equiv L_i$ then $\beta R_i \equiv \beta L_i$.

Let's fix that by introducing a separate β coefficients for L, R and O.

$$(\beta_L, \beta_R, \beta_O) \xleftarrow{R} \mathbb{F}^3$$

More coefficients!

The main problem is that if $R_i \equiv L_i$ then $\beta R_i \equiv \beta L_i$.

Let's fix that by introducing a separate β coefficients for L, R and O.

$$(\beta_L, \beta_R, \beta_O) \xleftarrow{R} \mathbb{F}^3$$

Therefore, $\beta_R R_i \neq \beta_L L_i$ even if $R_i \equiv L_i$, so the previous hack doesn't work.

More coefficients!

The main problem is that if $R_i \equiv L_i$ then $\beta R_i \equiv \beta L_i$.

Let's fix that by introducing a separate β coefficients for L, R and O.

$$(\beta_L, \beta_R, \beta_O) \xleftarrow{R} \mathbb{F}^3$$

Therefore, $\beta_R R_i \neq \beta_L L_i$ even if $R_i \equiv L_i$, so the previous hack doesn't work.

So, finally, the trusted setup is updated with:

$$g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, \{g^{(\beta_L L_i(\tau) + \beta_R R_i(\tau) + \beta_O O_i(\tau))}\}_{i \in [n]}$$

More coefficients!

The main problem is that if $R_i \equiv L_i$ then $\beta R_i \equiv \beta L_i$.

Let's fix that by introducing a separate β coefficients for L, R and O.

$$(\beta_L, \beta_R, \beta_O) \xleftarrow{R} \mathbb{F}^3$$

Therefore, $\beta_R R_i \neq \beta_L L_i$ even if $R_i \equiv L_i$, so the previous hack doesn't work.

So, finally, the trusted setup is updated with:

$$g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}$$

Verification:

$$e(\pi_L, g^{\beta_L}) \cdot e(\pi_R, g^{\beta_R}) \cdot e(\pi_O, g^{\beta_O}) = e(\pi_\beta, g)$$

Even more coefficients!

As the adversary has an access to the public g^{β_L}, g^{β_R}, g^{β_O} he still can cheat verifier by calculating modified π_β.

Example

Consider a constraint $w_1 \times w_1 = w_2$. Let's try to assign 2 and 5 for w_1 in a single constraint. As $2 \times 5 = 10$, the w_2 should contains value 10.

$$\mathrm{w} = (w_1, w_2) = (2, 10)$$

Even more coefficients!

As the adversary has an access to the public g^{β_L}, g^{β_R}, g^{β_O} he still can cheat verifier by calculating modified π_β.

Example

Consider a constraint $w_1 \times w_1 = w_2$. Let's try to assign 2 and 5 for w_1 in a single constraint. As $2 \times 5 = 10$, the w_2 should contains value 10.

$$\mathrm{w} = (w_1, w_2) = (2, 10)$$

The next QAP can be built:

$$L(x) = 2L_1(x) + 10L_2(x)$$
$$R(x) = 2R_1(x) + 3 + 10R_2(x)$$
$$O(x) = 2O_1(x) + 10O_2(x)$$

Even more coefficients!

As the adversary has an access to the public g^{β_L}, g^{β_R}, g^{β_O} he still can cheat verifier by calculating modified π_β.

Example

Consider a constraint $w_1 \times w_1 = w_2$. Let's try to assign 2 and 5 for w_1 in a single constraint. As $2 \times 5 = 10$, the w_2 should contains value 10.

$$\mathrm{w} = (w_1, w_2) = (2, 10)$$

The next QAP can be built:

$$\begin{aligned} L(x) &= 2L_1(x) + 10L_2(x) \\ R(x) &= 2R_1(x) + 3 + 10R_2(x) \\ O(x) &= 2O_1(x) + 10O_2(x) \end{aligned}$$

Compute π_β as:

$$(g^{(\beta_L \mathbf{L_1}(\tau) + \beta_R \mathbf{R_1}(\tau) + \beta_O \mathbf{O_1}(\tau))})^2 \cdot (g^{\beta_R})^3 \cdot (g^{(\beta_L \mathbf{L_2}(\tau) + \beta_R \mathbf{R_2}(\tau) + \beta_O \mathbf{O_2}(\tau))})^{10}$$

Even more coefficients!

To prevent this, let's introduce... one more coefficient!

$$\gamma \xleftarrow{R} \mathbb{F}$$

Even more coefficients!

To prevent this, let's introduce... one more coefficient!

$$\gamma \xleftarrow{R} \mathbb{F}$$

So, finally... the trusted setup is updated with:

$$g^{\gamma}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}$$

Even more coefficients!

To prevent this, let's introduce... one more coefficient!

$$\gamma \xleftarrow{R} \mathbb{F}$$

So, finally... the trusted setup is updated with:

$$g^{\gamma}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, \{g^{(\beta_L L_i(\tau) + \beta_R R_i(\tau) + \beta_O O_i(\tau))}\}_{i \in [n]}$$

Proving process isn't changed, unlike verification:

$$e(\pi_L, g^{\beta_L \gamma}) \cdot e(\pi_R, g^{\beta_R \gamma}) \cdot e(\pi_O, g^{\beta_O \gamma}) = e(\pi_\beta, g^{\gamma})$$

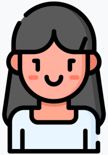
That makes it unfeasible to cheat.

Sound SNARK Protocol

Trusted Setup:

$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\}$,
$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}$, **delete**$(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma)$.

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

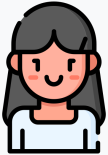
Sound SNARK Protocol

Trusted Setup:

$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\}$, $\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}$, **delete**$(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma)$.

✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Sound SNARK Protocol

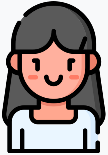
Trusted Setup:

$$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}, \quad \{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \quad \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\},$$
$$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}, \quad \textbf{delete}(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma).$$

- ✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.
- ✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi_L' &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi_R' &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi_O' &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi_H' &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Sound SNARK Protocol

Trusted Setup:

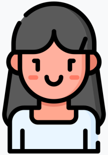
$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\}$,
$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}$, **delete**$(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma)$.

- ✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.
- ✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

- ✓ $\pi_\beta \leftarrow g^{\beta_L L(\tau)+\beta_R R(\tau)+\beta_O O(\tau)}$

Prover $\mathcal{P}$



Verifier $\mathcal{V}$

Sound SNARK Protocol

Trusted Setup:

$$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}, \quad \{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \quad \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\},$$
$$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}, \quad \textbf{delete}(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma).$$

- ✓ $H(x) = \frac{L(x)\times R(x)-O(x)}{Z(x)}$.
- ✓ KZG commitments:

$$\pi_L \leftarrow g^{L(\tau)}, \qquad \pi'_L \leftarrow g^{\alpha L(\tau)},$$
$$\pi_R \leftarrow g^{R(\tau)}, \qquad \pi'_R \leftarrow g^{\alpha R(\tau)},$$
$$\pi_O \leftarrow g^{O(\tau)}, \qquad \pi'_O \leftarrow g^{\alpha O(\tau)},$$
$$\pi_H \leftarrow g^{H(\tau)}, \qquad \pi'_H \leftarrow g^{\alpha H(\tau)}.$$

- ✓ $\pi_\beta \leftarrow g^{\beta_L L(\tau)+\beta_R R(\tau)+\beta_O O(\tau)}$

$$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H, \pi_\beta)$$

Prover $\mathcal{P}$ Verifier $\mathcal{V}$

Sound SNARK Protocol

Trusted Setup:

$$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}, \quad \{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \quad \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\},$$
$$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}, \quad \textbf{delete}(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma).$$

✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.

✓ KZG commitments:

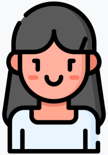
$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

✓ $\pi_\beta \leftarrow g^{\beta_L L(\tau)+\beta_R R(\tau)+\beta_O O(\tau)}$

✓ $e(\pi_L, \pi_R) ==$
$e(\mathsf{com}(Z), \pi_H)\cdot e(\pi_O, g)$.

Prover $\mathcal{P}$

$$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H, \pi_\beta)$$



Verifier $\mathcal{V}$

Sound SNARK Protocol

Trusted Setup:

$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}$, $\{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}$, $\{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\}$,
$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}$, **delete**$(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma)$.

- ✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.
- ✓ KZG commitments:

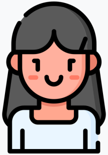
$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

- ✓ $\pi_\beta \leftarrow g^{\beta_L L(\tau)+\beta_R R(\tau)+\beta_O O(\tau)}$

- ✓ $e(\pi_L, \pi_R) ==$ $e(\mathsf{com}(Z), \pi_H) \cdot e(\pi_O, g)$.
- ✓ Proof of Exponent:

$$\begin{aligned} e(\pi_L, g^{\alpha}) &= e(\pi'_L, g), \\ e(\pi_R, g^{\alpha}) &= e(\pi'_R, g), \\ e(\pi_O, g^{\alpha}) &= e(\pi'_O, g), \\ e(\pi_H, g^{\alpha}) &= e(\pi'_H, g). \end{aligned}$$

$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H, \pi_\beta)$



Prover $\mathcal{P}$ Verifier $\mathcal{V}$

Sound SNARK Protocol

Trusted Setup:

$$\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma \xleftarrow{R} \mathbb{F}, \quad \{\{g^{\tau^i}, g^{\alpha\tau^i}\}_{i\in[d]}, \quad \{g^{(\beta_L L_i(\tau)+\beta_R R_i(\tau)+\beta_O O_i(\tau))}\}_{i\in[n]}\},$$
$$\{g^{Z(\tau)}, g^{\alpha}, g^{\beta_L}, g^{\beta_R}, g^{\beta_O}, g^{\beta_L\gamma}, g^{\beta_R\gamma}, g^{\beta_O\gamma}, g^{\gamma}\}, \quad \textbf{delete}(\tau, \alpha, \beta_L, \beta_R, \beta_O, \gamma).$$

- ✓ $H(x) = \frac{L(x)\times R(x) - O(x)}{Z(x)}$.
- ✓ KZG commitments:

$$\begin{aligned} \pi_L &\leftarrow g^{L(\tau)}, & \pi'_L &\leftarrow g^{\alpha L(\tau)}, \\ \pi_R &\leftarrow g^{R(\tau)}, & \pi'_R &\leftarrow g^{\alpha R(\tau)}, \\ \pi_O &\leftarrow g^{O(\tau)}, & \pi'_O &\leftarrow g^{\alpha O(\tau)}, \\ \pi_H &\leftarrow g^{H(\tau)}, & \pi'_H &\leftarrow g^{\alpha H(\tau)}. \end{aligned}$$

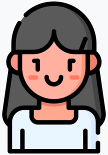
- ✓ $\pi_\beta \leftarrow g^{\beta_L L(\tau)+\beta_R R(\tau)+\beta_O O(\tau)}$

- ✓ $e(\pi_L, \pi_R) ==$ $e(\mathsf{com}(Z), \pi_H)\cdot e(\pi_O, g)$.
- ✓ Proof of Exponent:

$$\begin{aligned} e(\pi_L, g^{\alpha}) &= e(\pi'_L, g), \\ e(\pi_R, g^{\alpha}) &= e(\pi'_R, g), \\ e(\pi_O, g^{\alpha}) &= e(\pi'_O, g), \\ e(\pi_H, g^{\alpha}) &= e(\pi'_H, g). \end{aligned}$$

- ✓ $e(\pi_L, g^{\gamma\beta_L})\cdot e(\pi_R, g^{\gamma\beta_R})\cdot$ $\cdot\, e(\pi_O, g^{\gamma\beta_O}) = e(\pi_\beta, g^{\gamma})$

$$\boldsymbol{\pi} = (\pi_L, \pi_R, \pi_O, \pi_H, \pi'_L, \pi'_R, \pi'_O, \pi'_H, \pi_\beta)$$



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

Thank you for your attention

♥

zkdl-camp.github.io

github.com/ZKDL-Camp