Circom

December 05, 2024

Distributed Lab

zkdl-camp.github.io

github.com/ZKDL-Camp

Plan

Introduction

Why do we need ZK?

Option

Solution to privacy

Example

1. *I know the private key that corresponds to this public key*
2. *I know a private key that corresponds to a public key from this list*

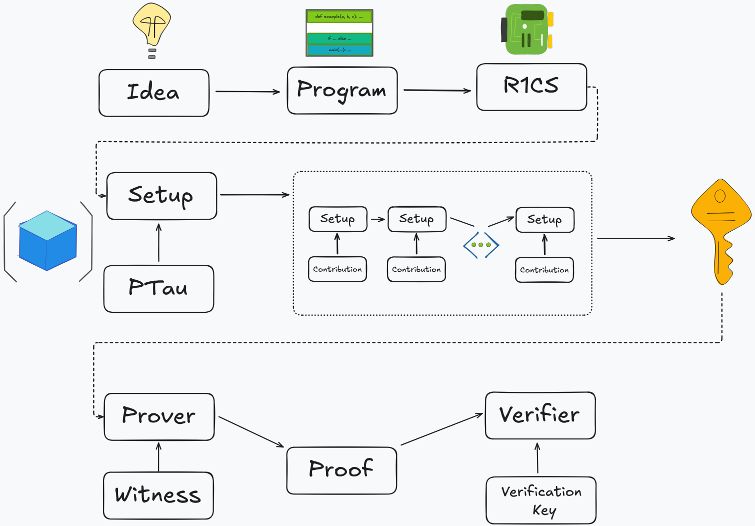
Option

Solution to scalability

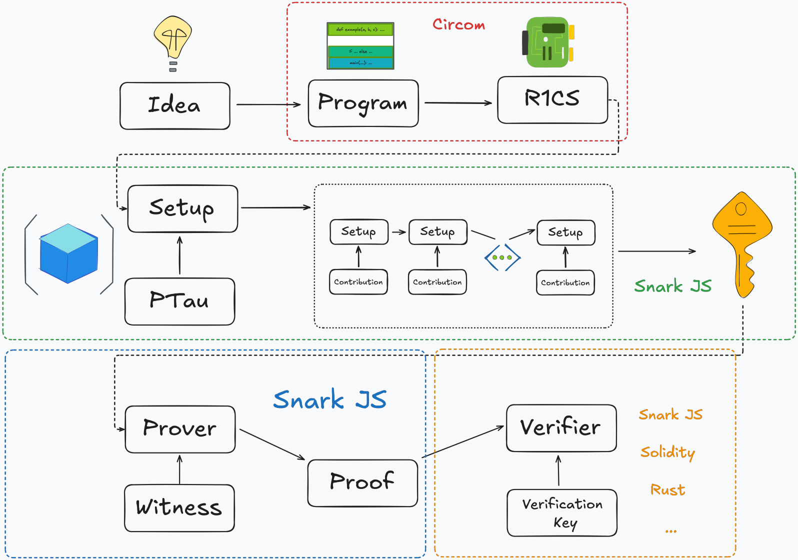
Example

This is the hash of a blockchain block that does not produce negative balances

Using ZKP


Idea
Program
R1CS
Setup
PTau
Setup
Setup
Setup
Contribution
Contribution
Contribution
Prover
Witness
Proof
Verifier
Verification Key

Toolchain


Circom
def example(a, b, c): ....
if ... else ...
main(...): ...
Idea
Program
R1CS
Setup
PTau
Setup
Setup
Setup
Contribution
Contribution
Contribution
Snark JS
Snark JS
Prover
Witness
Proof
Verifier
Verification Key
Snark JS
Solidity
Rust
...

Circom

Previously on ZKDL Camp

Probably you can recall the function

```
def r(x1: F, x2: F, x3: F) -> F:
    return x2 * x3 if x1 else x2 + x3
```

That can be expressed as:

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

We need a boolean restriction for x_1:

$$x_1 \times (1 - x_1) = 0$$

Thus, the next constraints can be build:

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

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

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

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

Previously on ZKDL Camp

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

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

Using the arithmetic in a large $\mathbb{F}_p$, consider the following values:

$$x_1 = 1, \quad x_2 = 3, \quad x_3 = 4$$

Verifying the constraints:

1. $x_1 \times x_1 = x_1 \quad (1 \times 1 = 1)$
2. $x_2 \times x_3 = \text{mult} \quad (3 \times 4 = 12)$
3. $x_1 \times \text{mult} = \text{selectMult} \quad (1 \times 12 = 12)$
4. $(1 - x_1) \times (x_2 + x_3) = r - \text{selectMult} \quad (0 \times 7 = 12 - 12)$

Previously on ZKDL Camp

By Groth16 Protocol the verifier should check the following condition:

$$e(\pi_L, \pi_R) = e(g_1^{\alpha}, g_2^{\beta}) e(\pi_{\text{io}}, g_2^{\gamma}) e(\pi_O, g_2^{\delta})$$

Recall

For BN254 (BN128), we have:

- Left inputs to e is of form $(x, y) \in \mathbb{G}_1$ — regular curve.
- Right inputs to e is of form $((x_1, y_1), (x_2, y_2)) \in \mathbb{G}_2$ — "complex" curve, consisting of two $\mathbb{F}_{p^2}$ coordinates.
- $e(g_1^{\alpha}, g_2^{\beta})$ is of form $(x_1, \dots, x_{12}) \in \mathbb{F}_{p^{12}}$

Thank you for your attention

♥

zkdl-camp.github.io

github.com/ZKDL-Camp