ZK programming and applications

Vu Vo

Agenda

- Understand basic keywords in zk programming
- Develop a small ZK circuit
- Discover a few ZK applications
- Help you have more questions about ZKP :D

Chapter I:
Intuitive

ZK buzzwords

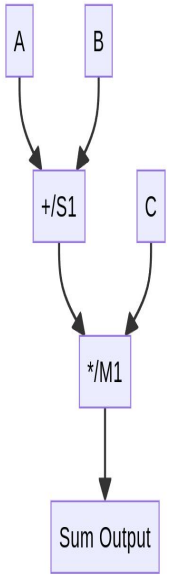
- Circuit
- Public/private input or output
- Statement
- Instance/Witness
- Prover/Verifier
- Prove/Verify
- …

Circuit

- Refer to the arithmetic circuit
- Directed acyclic graph
- Describes the computation

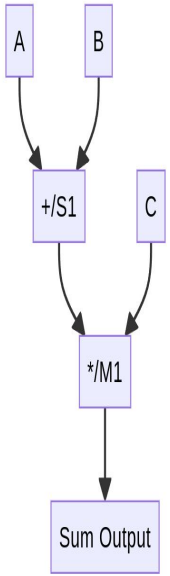
Circuit pt2

- Sum Output = (A + B) * C



Exercise 1.1

- Fill all value in circuit with
  - A = 10
  - B = 12
  - C = 3



Statement, witness, instance

- A claim prover want to prove this is true with verifier
- Witness the set of data verifier don’t read
- Instance is set of public value anybody can read

Public(Private) input(output)

	Input node	Output node	Intermediate node
Both Verifier and Prover know value	Public input	Public output	Public data
Only Prover know value	Private input	Private output	Private data

Intuitive

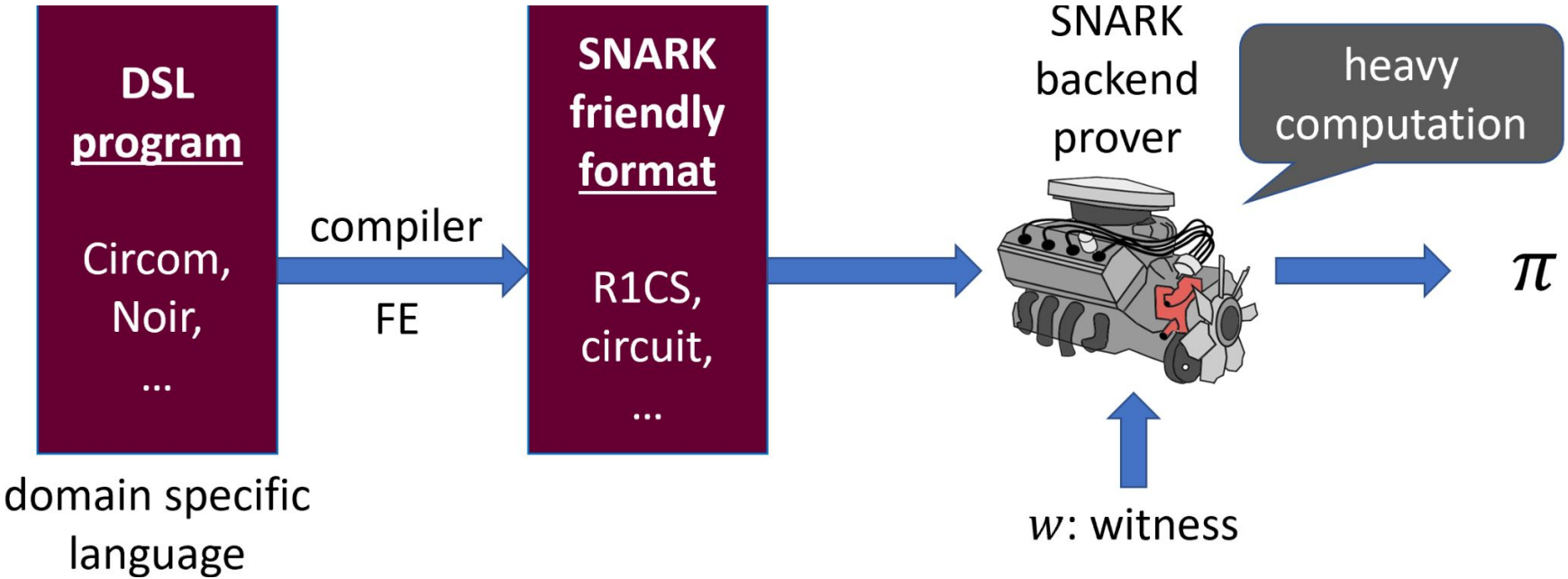
- Alice wants to prove to Bob she knows x s.t x^8 = 256
- x^8 = 256 is statement
- x = 2 is witness or private input.
- 256 is instance

Exercise 1.2

Draw circuit for statement : x^8 = 256, point out what is private input, private output on that circuit.

Prove/Verify

- Prove: Create the proof
- Verify: Verify the proof
- Depends on context:
  - Succinct: |proof| << |witness|
  - ZK: Verify can’t extract any knowledge about witness from proof


DSL program
Circom, Noir, …
domain specific language
compiler
FE
SNARK friendly format
R1CS, circuit, …
SNARK backend prover
heavy computation
π
w: witness

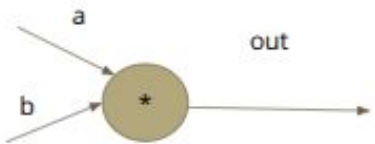
Chapter 2: Circom

Circom language

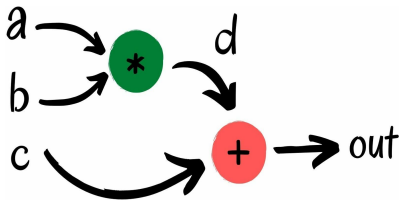
- Circom is DSL language use for building circuit in zkp (Thanks Iden3)
- Powerful toolchain and ecosystem
- Friendly with Ethereum ecosystem
- Project use circom: semaphore, Unirep, maci,... etc.
- https://zkrepl.dev

Signal and constraints

- Assigning signal with “<---” notation.
- Ex: out <--- 100
- Create constraint with “===” natiation.
- Ex: out === a * b
- Circuit view for constraint above:



Hello world



circuits > hello.circom

```
1  pragma circom 2.0.0;
2
3  template AC() {
4      //Declaration of signals
5      signal input a;
6      signal input b;
7      signal input c;
8      signal d;
9      signal output out;
10
11     // assigning signal d with value a * b;
12     d <-- a * b;
13     // create constraint d equal a * b;
14     d === a * b;
15
16     // sugar syntax
17     // out <-- c + d;
18     // out === c + d;
19     out <== c + d;
20 }
21
22 component main {public [a]} = AC();
23
```

Speed run circom syntax

- Variable & Signal
  - Signal and constraint a field F_p
- Function & template
- Control flow: if, for, while
- Main component
- Import lib use “include”
- https://docs.circom.io

Notes

- Circom only support constraint the expression have degree <= 2
- For examples:
  - X <== 10
  - X <== 10 * a
  - X <== a * b
  - X<== a * b * c (invalid)

Exercise 2.1

Write circuit prove this statement x^8 = 256.

Break

Chapter 3:
Circom Advances

circomlib

- Circom template collection
- https://github.com/iden3/circomlib
- Useful template:
  - Comparators: isEqual, IsZero
  - Poseidon hash
  - Sha256
  - Merkle

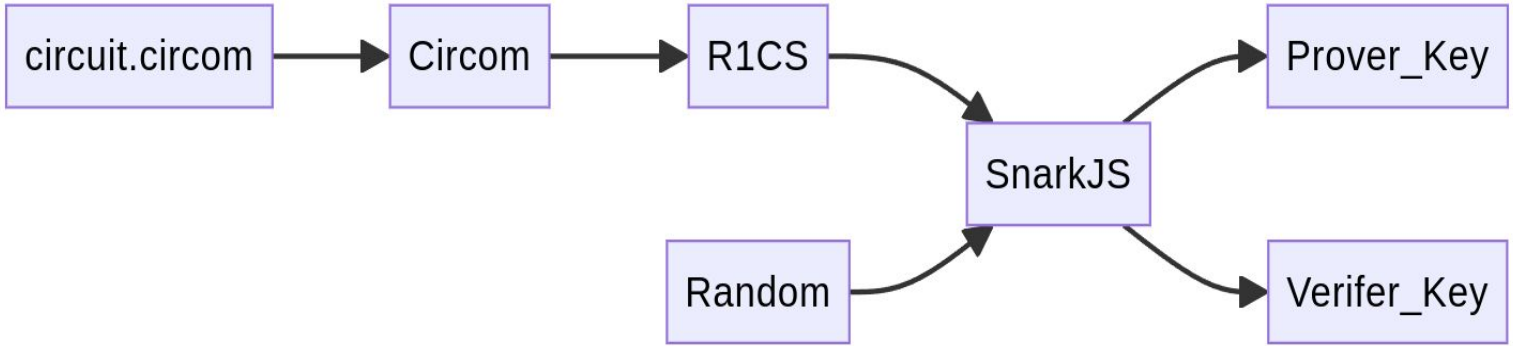
ZK Friendly

- Sha256 is not zk friendly
- Poseidon is zk friendly
- Zk friendly = same purpose + less constraints

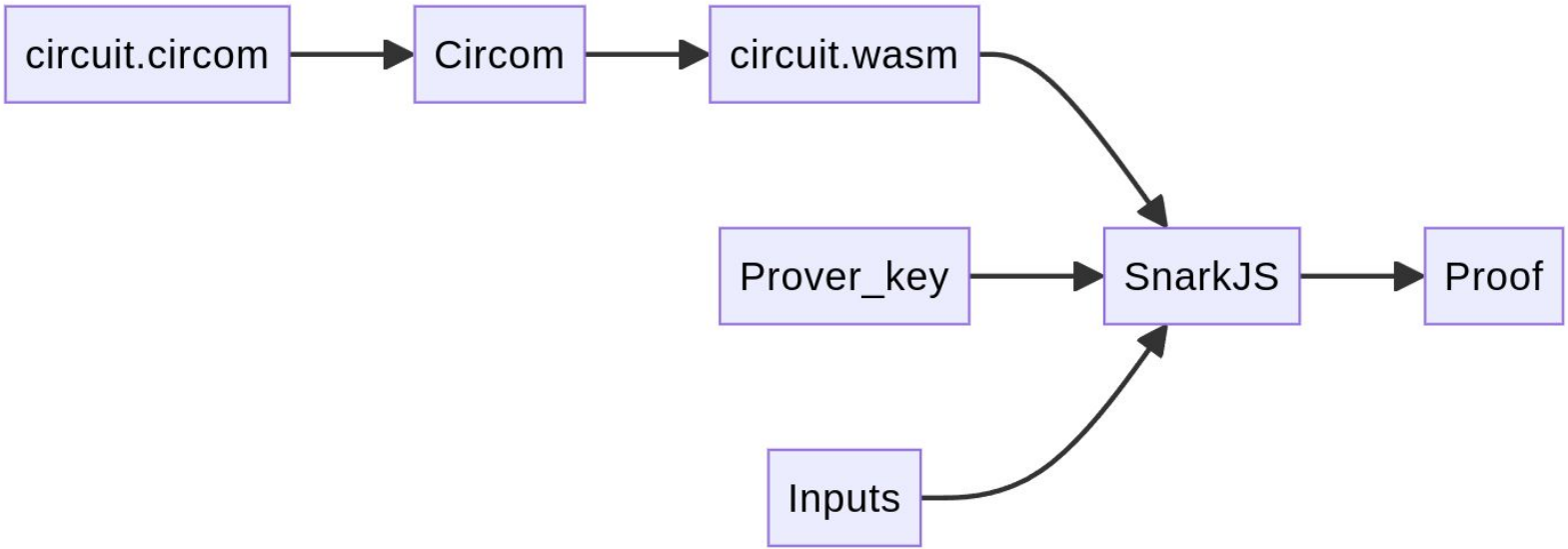
Circom and snarkjs

- snarkjs is backend for circom
- Support Groth16 and Plonk protocol

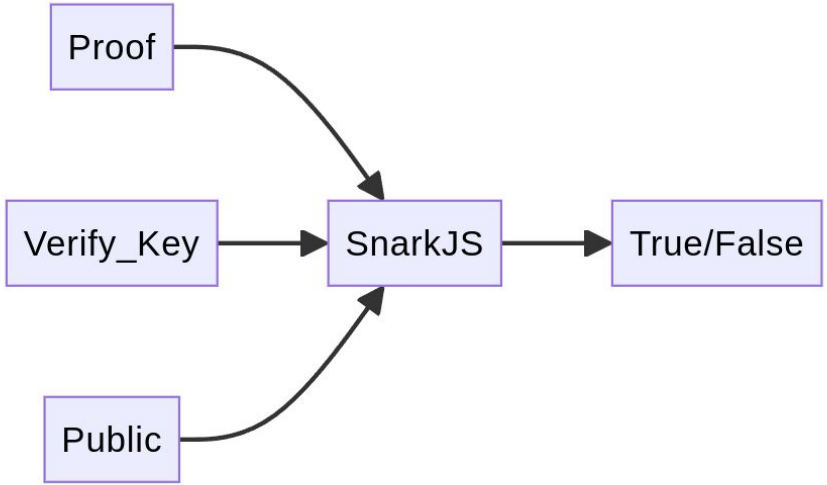
Circom + Snarkjs: Setup



SnarkJs + Circom: Prove



SnarkJS + Circom: Verify



Group programming: Sudoku

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sudoku rules:

- 9 x 9 grid
- Fill number from 1..9
- Not duplicate number on row, column and sub 3x3 grid
- We only check row/column for simple application

Group programming: Sudoku

Given sudoku table, verify solution without reveal solutions.

Chapter 4: ZK Applications

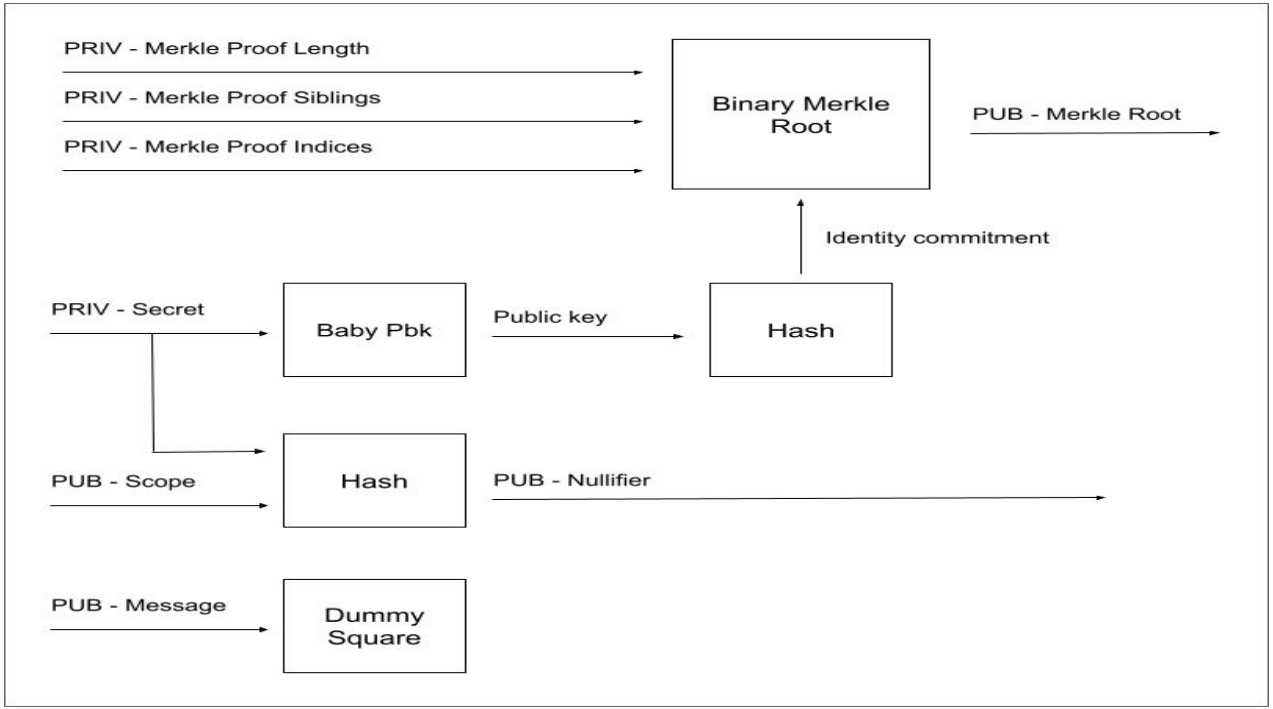
Computation + ZK
= ZK Application

Semaphore

What is Semaphore?

- Proving set membership in zero-knowledge.
- Example: S = [a, b, c, d]. Create proof of d in S without reveal d id.

How?



Source:https://docs.semaphore.pse.dev/technical-reference/circuits

Brainstorm

zkemail

What is zkemail?

- Prove things in email without review email's content.

How?

- DKIM (DomainKeys Identified Mail) allow receiver verify email from authorized source.
- Email server sign hash of email + metadata with domain private key
- Public key public on domain's DNS records.
- We “zk” DKIM signature verification.

Brainstorm

Break

zk rollup

What is zk rollup?

- Scaling L1 solution.
- Use ZK to off-chain heavy computation

How?

- Maintain the commitment of previous state of off-chain computation.
- Use zk to validate the next state create from previous state.

Build

- “Sign” transaction
- Merkle tree
- Proof Tx validate

Goal

- Zk rollup for simple payment transaction

More

- zkKYC, zkID
- Zk machine learning
- Cloud verify computation
- More and more…

zkVM

What is zkVM

- Run and create zk proof for general program.
- Example: Create zk proof for rust program

Specific circuit vs zkVM

- Less flexible
- DevEx: Hard to develop and maintain
- High Performance
- Devtool: Circom, Halo2,...

- More flexible
- DevEx: Easier to develop and maintain
- Low performance
- Devtool: Rics0, Succinct, Stacknet,...

Q&A