Basics of Circom programming

A learning group for ZK and SNARK application development

Daniel Szego

In: https://www.linkedin.com/in/daniel-szego/

Logistics: ZK Learning Group

Every month, third thursday in 2025, from 18 (CET)

One hour, presentation + short discussion

Different topics on zero knowledge proof,

- mostly from programmer and application developers perspective
- with some theory

Coordination:

- Discord channel: LF Decentralized Trust
  https://discord.com/channels/905194001349627914/1329201532628898036
- Meetup.com: https://www.meetup.com/lfdt-hungary/events/305634614/
- Repo with all the contents:https://github.com/LF-Decentralized-Trust-labs/

https://github.com/Daniel-Szego/zk-leraning-group

Quizzes and small programming challenges, LFDT merchs at the end

Logistics: Hunting for the SNARK

February - Introduction, Theory : Definitions and building blocks

March - Theory : Polynomial commitments

April - Theory : Interactive oracle proofs

May - Programming : Circom

June - Programming : Circom

July - Programming : Noir

August - Programming : Noir

September : Applications : Off-chain transaction

October : Applications : Proving solvency

November : Applications : Rollup

December : Wrap up, Applications

Subject to change based on community discussion ….

Agenda

- *zkSNARK*
- *zk programming*
- *Core algorithmic considerations*
- *DSL language and tool selection*
- *Circom*
- *Demo, code*
- *Challenge*
- *Links, Resources, Literature*
- *Q&A*

(zk)SNARK - Succinct Non-interactive ARgument of Knowledge

Computation: arithmetic circuit : *C(x, w) -> F*

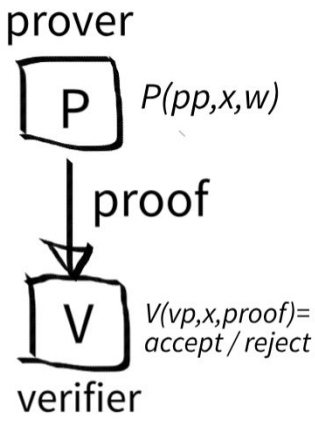
- *x* public input
- *w* private input, witness
- high level computation
- arithmetic circuit
- polynomials

Prover algorithm: *P (pp, x, w) -> proof*

Verifier algorithm: *V (vp, x, proof) -> accept / reject*

Properties:

- *Succinct:*
- *Complete:*
- *Knowledge sound:*
- *Zero knowledge*



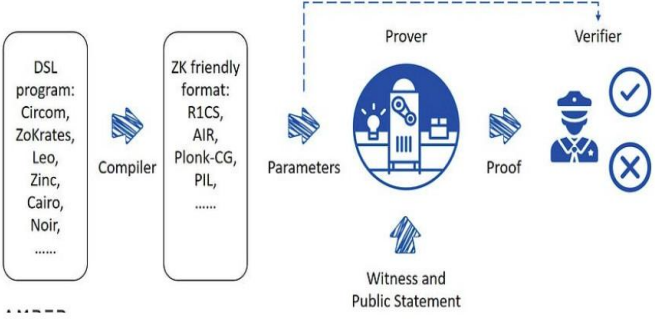
ZK programming - engineering flow

Domain specific languages for SNARK or *zk*SNARK programming

Abstracting away some of the mathematical and theoretical complexity

ZK and SNARK programming without cryptographic knowledge ? Not yet :)

Complex development frameworks, compile, test, prover, verifier module integrations.



Core algorithm consideration

Under very active development

Proof size

Verification time

Setup :
- **per circuit**
- **universal**
- **transparent**

Post quantum readiness

	size of proof π	verifier time	setup	post-quantum?
Groth'16	≈ 200 Bytes $O_\lambda(1)$	≈ 1.5 ms $O_\lambda(1)$	trusted per circuit	no
Plonk / Marlin	≈ 400 Bytes $O_\lambda(1)$	≈ 3 ms $O_\lambda(1)$	universal trusted setup	no
Bulletproofs	≈ 1.5 KB $O_\lambda(\log \lvert C \rvert)$	≈ 3 sec $O_\lambda(\lvert C \rvert)$	transparent	no
STARK	≈ 100 KB $O_\lambda(\log^2 \lvert C \rvert)$	≈ 10 ms $O_\lambda(\log^2 \lvert C \rvert)$	transparent	yes

DSL language and tool selection

Core algorithm consideration

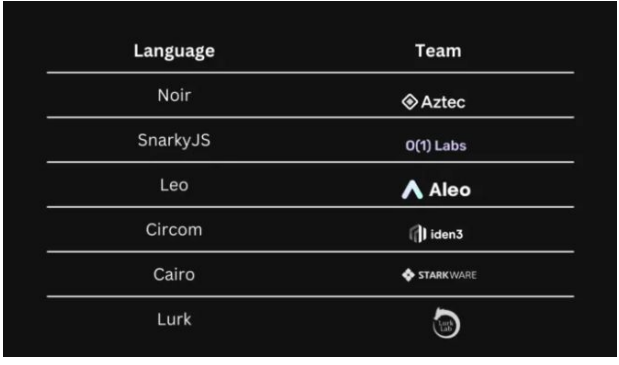
Imperative / description / circuit languages

Different base programming language

Different programming language and framework integration modules

Technological life cycle: all are early stage, but:

- **Successful productive usage**
- **Stable releases**



Circom

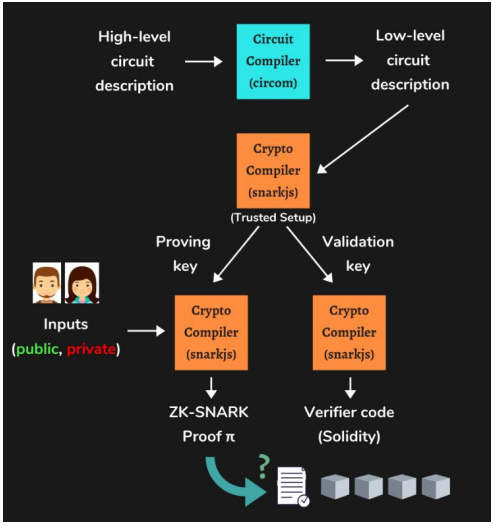
DLS / circuit programming language and development environment for arithmetic circuits and constraints

Used e.g. in tornado cash

Supports:
- **Groth16**
- **Plonk**

Well established (exist 3 years :)

Supported integration:
- **javascript (snarkjs)**
- **cpp**
- **solidity verifier**



https://docs.circom.io/

Demo

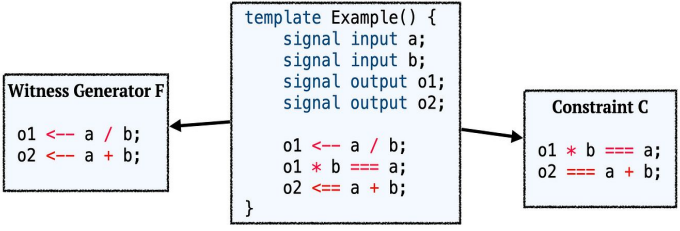
Creating module / template for arithmetic circuits:
- **public inputs**
- **private inputs**
- **output**

Creating rank1 constraints and calculations

Setup, power of tau

Creating proof for certain inputs

Verifying proof
- **javascript**
- **solidity**



code: */circom_basics*

Challenge

Developer challenge:

Create a proof that an output is the 10th series of a Fibonacci like series of two hidden (private) starting elements of the series

Links, Resources, Literature

Circom tutorial:
https://www.rareskills.io/post/circom-tutorial

Circom Workshop 1:
https://learn.0xparc.org/materials/circom/learning-group-1/circom-1/

Circom repo:
https://github.com/iden3/circom

Install circom:
https://docs.circom.io/getting-started/installation/#installing-circom

Circom 2 documentation:
https://docs.circom.io/

Happy Hunting for the SNARK :)

Q & A

Daniel Szego

In: https://www.linkedin.com/in/daniel-szego/