Zero-Knowledge Proofs: The Bridge Between Cryptographic Theory and Blockchain Scalability

Youssef El Housni

Morocco Blockchain & Cryptocurrency Days — January 15th, 2025

whoami

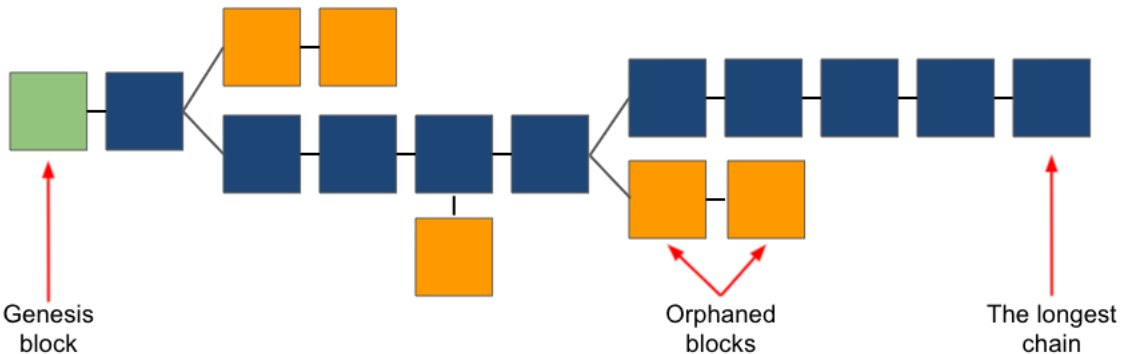
- PhD in cryptography — Ecole Polytechnique (Paris)
- Cryptographer — Consensys (New York)
- Co-developer of gnark and Linea

Overview

1. Blockchains
2. Zero-knowledge proofs
3. Applications
4. Research

Overview

Blockchains



Blockchains

- How is a tx included in a block?
- How is the longest chain agreed upon?

Blockchains

- How is a tx included in a block?
  - Signatures verification (Bitcoin: ECDSA/Schnorr, Ethereum: ECDSA/BLS)
- How is the longest chain is agreed upon?
  - Consensus (Bitcoin: proof-of-work, Ethereum: proof-of-stake)

Blockchains

A blockchain is a public peer-to-peer *decentralized*, *transparent, immutable, paying* ledger.

- *Transparent*: everything is visible to everyone
- *Immutable*: nothing can be removed once written
- *Paying*: everyone should pay a fee to use

Blockchains

A blockchain is a public peer-to-peer *decentralized*, *transparent, immutable, paying* ledger.

- *Transparent*: everything is visible to everyone
- *Immutable*: nothing can be removed once written
- *Paying*: everyone should pay a fee to use

$$\text{Transparent} \xrightarrow[\text{Problem}]{} \text{confidentiality} \xrightarrow[\text{Solution}]{} ?$$

$$\text{Immutable} \xrightarrow[\text{Problem}]{} \text{scalability} \xrightarrow[\text{Solution}]{} ?$$

$$\text{Paying} \xrightarrow[\text{Problem}]{} \text{cost} \xrightarrow[\text{Solution}]{} ?$$

Overview

Zero-knowledge proofs (ZKP)

Alice

I know the solution to this complex equation

Bob

No idea what the solution is but Alice claims to know it

Zero-knowledge proofs (ZKP)

Alice

I know the solution to this complex equation

Bob

No idea what the solution is but Alice claims to know it

- **Sound**: **Alice** has a wrong solution $\Longrightarrow$ **Bob** is not convinced.

Zero-knowledge proofs (ZKP)

Alice

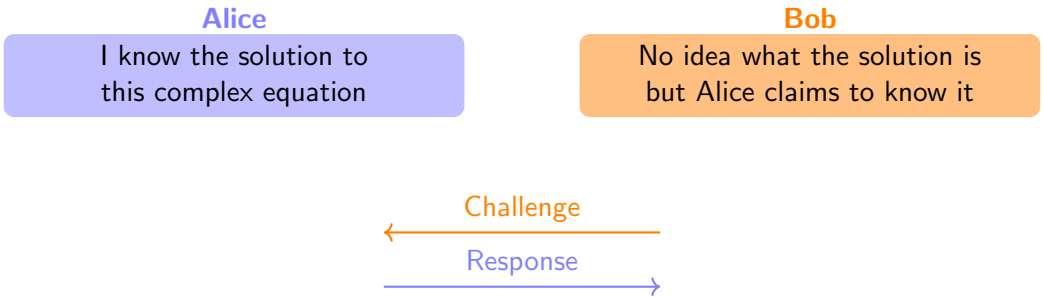
I know the solution to this complex equation

Bob

No idea what the solution is but Alice claims to know it

- **Sound**: **Alice** has a wrong solution $\implies$ **Bob** is not convinced.
- **Complete**: **Alice** has the solution $\implies$ **Bob** is convinced.

Zero-knowledge proofs (ZKP)



- **Sound**: **Alice** has a wrong solution $\implies$ **Bob** is not convinced.
- **Complete**: **Alice** has the solution $\implies$ **Bob** is convinced.
- **Zero-knowledge**: **Bob** does NOT learn the solution.

ZKP families

Expressivity

- *specific* statement vs. *general* statement

ZKP families

Expressivity

- *specific* statement vs. *general* statement

Deployability

- *interactive* vs. $non - interactive$ protocol
- *trapdoored* setup vs. *transparent* setup
- *Designated* verifier vs. *any* verifier

ZKP families

Expressivity

- *specific* statement vs. *general* statement

Deployability

- *interactive* vs. $non - interactive$ protocol
- *trapdoored* setup vs. *transparent* setup
- *Designated* verifier vs. *any* verifier

Complexity

- prover complexity (Alice)
- verifier complexity (Bob)
- communication complexity (size of the proof and the setup)

ZKP families

Expressivity

- *specific* statement vs. *general* statement

Deployability

- *interactive* vs. $non - interactive$ protocol
- *trapdoored* setup vs. *transparent* setup
- *Designated* verifier vs. *any* verifier

Complexity

- prover complexity (Alice)
- verifier complexity (Bob)
- communication complexity (size of the proof and the setup)

Security

- Cryptographic assumptions
- Cryptographic primitives

Blockchains and ZKP

A blockchain is a public peer-to-peer *decentralized*, *transparent, immutable, paying* ledger.

- *Transparent*: everything is visible to everyone
- *Immutable*: nothing can be removed once written
- *Paying*: everyone should pay a fee to use

Transparent $\xrightarrow[\text{Problem}]{}$ confidentiality $\xrightarrow[\text{Solution}]{}$ *ZKP*

setup, prover?, verifier?

Immutable $\xrightarrow[\text{Problem}]{}$ scalability $\xrightarrow[\text{Solution}]{}$ *ZKP*

Communication complexity

Paying $\xrightarrow[\text{Problem}]{}$ cost $\xrightarrow[\text{Solution}]{}$ *ZKP*

Verifier complexity, *prover*?

ZKP literature landmarks

- First ZKP work [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [Kil92]
- Succinct Non-Interactive ZKP [Mic94]

ZKP literature landmarks

- First ZKP work [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [Kil92]
- Succinct Non-Interactive ZKP [Mic94]
- Pairing-based succinct NIZK [Gro10]

ZKP literature landmarks

- First ZKP work [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [Kil92]
- Succinct Non-Interactive ZKP [Mic94]
- Pairing-based succinct NIZK [Gro10]
- "SNARK" terminology and characterization of existence [BCCT12]
- Pairing-based SNARK in quasi-linear prover time [GGPR13]
- Pairing-based SNARK with shortest proof and verifier time [Gro16]

ZKP literature landmarks

- First ZKP work [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [Kil92]
- Succinct Non-Interactive ZKP [Mic94]
- Pairing-based succinct NIZK [Gro10]
- "SNARK" terminology and characterization of existence [BCCT12]
- Pairing-based SNARK in quasi-linear prover time [GGPR13]
- Pairing-based SNARK with shortest proof and verifier time [Gro16]
- SNARK with universal and updatable setup [GKM$^+$18, MBKM19, GWC19, CHM$^+$20]

ZKP literature landmarks

- First ZKP work [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [Kil92]
- Succinct Non-Interactive ZKP [Mic94]
- Pairing-based succinct NIZK [Gro10]
- "SNARK" terminology and characterization of existence [BCCT12]
- Pairing-based SNARK in quasi-linear prover time [GGPR13]
- Pairing-based SNARK with shortest proof and verifier time [Gro16]
- SNARK with universal and updatable setup [GKM$^+$18, MBKM19, GWC19, CHM$^+$20]
- SNARK on small fields [plonky, circle-STARK, vortex, ...]

zk-SNARK: Zero-Knowledge Succinct Non-interactive ARgument of Knowledge

"I have a *computationally sound*, *complete*, *zero-knowledge*, **succinct**, **non-interactive** proof that a statement is true and that I know a related secret".

Succinct

A proof is very "short" and "easy" to verify.

Non-interactive

No interaction between the prover and verifier for proof generation and verification (except the proof message).

ARgument of Knowledge

Honest verifier is convinced that a computationally bounded prover knows a secret information.

Preprocessing zk-SNARK for NP language

F: public NP program, x, z: public inputs, w: private input

$$z := F(x, w)$$

Preprocessing zk-SNARK for NP language

F: public NP program, x, z: public inputs, w: private input

$$z := F(x, w)$$

A zk-SNARK consists of algorithms S, P, V s.t. for a security parameter λ:

$$\textit{Setup}: \quad (pk, vk) \leftarrow S(F, 1^{\lambda})$$

Preprocessing zk-SNARK for NP language

F: public NP program, x, z: public inputs, w: private input

$$z := F(x, w)$$

A zk-SNARK consists of algorithms S, P, V s.t. for a security parameter λ:

$$\begin{array}{lll} \textit{Setup}: & (pk, vk) & \leftarrow \quad S(F, 1^\lambda) \\ \textit{Prove}: & \pi & \leftarrow \quad P(x, z, w, pk) \end{array}$$

Preprocessing zk-SNARK for NP language

F: public NP program, x, z: public inputs, w: private input

$$z := F(x, w)$$

A zk-SNARK consists of algorithms S, P, V s.t. for a security parameter λ:

$$\begin{array}{lllll} \textit{Setup}: & (pk, vk) & \leftarrow & S(F, 1^{\lambda}) \\ \textit{Prove}: & \pi & \leftarrow & P(x, z, w, pk) \\ \textit{Verify}: & \texttt{false/true} & \leftarrow & V(x, z, \pi, vk) \end{array}$$

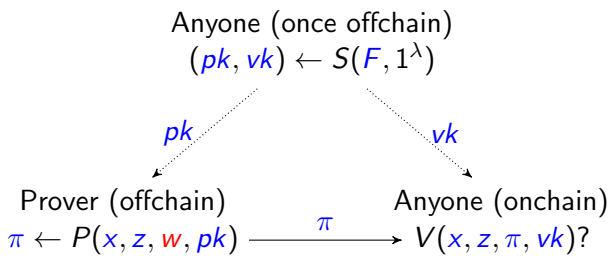
Preprocessing zk-SNARK for NP language

F: public NP program, x, z: public inputs, w: private input

$$z := F(x, w)$$

A zk-SNARK consists of algorithms S, P, V s.t. for a security parameter λ:

$$\begin{array}{lll} \textit{Setup}: & (pk, vk) & \leftarrow S(F, 1^\lambda) \\ \textit{Prove}: & \pi & \leftarrow P(x, z, w, pk) \\ \textit{Verify}: & \texttt{false/true} & \leftarrow V(x, z, \pi, vk) \end{array}$$



zk-SNARKs in a nutshell

Main ideas:

zk-SNARKs in a nutshell

Main ideas:

1. Reduce a "general statement" satisfiability to a polynomial equation satisfiability.

zk-SNARKs in a nutshell

Main ideas:

1. Reduce a "general statement" satisfiability to a polynomial equation satisfiability.
2. Use Schwartz-Zippel lemma to succinctly verify the polynomial equation with high probability.

zk-SNARKs in a nutshell

Main ideas:

1. Reduce a "general statement" satisfiability to a polynomial equation satisfiability.
2. Use Schwartz-Zippel lemma to succinctly verify the polynomial equation with high probability.
3. Use homomorphic hiding cryptography to blindly verify the polynomial equation.

zk-SNARKs in a nutshell

Main ideas:

1. Reduce a "general statement" satisfiability to a polynomial equation satisfiability.
2. Use Schwartz-Zippel lemma to succinctly verify the polynomial equation with high probability.
3. Use homomorphic hiding cryptography to blindly verify the polynomial equation.
4. Make the protocol non-interactive.

A pairing-based SNARK

Example: Groth16 [Gro16]

Given an instance $\Phi = (a_0, \ldots, a_\ell) \in \mathbb{F}_r^\ell$ of a public NP program F

A pairing-based SNARK

Example: Groth16 [Gro16]

Given an instance $\Phi = (a_0, \ldots, a_\ell) \in \mathbb{F}_r^\ell$ of a public NP program F

- Setup: $(pk, vk) \leftarrow S(F, \tau, 1^\lambda)$ where

$$vk = (vk_{\alpha,\beta}, \{vk_{\pi_i}\}_{i=0}^{\ell}, vk_\gamma, vk_\delta) \in \mathbb{G}_T \times \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2 \times \mathbb{G}_2$$

A pairing-based SNARK

Example: Groth16 [Gro16]

Given an instance $\Phi = (a_0, \ldots, a_\ell) \in \mathbb{F}_r^\ell$ of a public NP program F

- Setup: $(pk, vk) \leftarrow S(F, \tau, 1^\lambda)$ where

$$vk = (vk_{\alpha,\beta}, \{vk_{\pi_i}\}_{i=0}^{\ell}, vk_\gamma, vk_\delta) \in \mathbb{G}_T \times \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2 \times \mathbb{G}_2$$

- Prove: $\pi \leftarrow P(\Phi, w, pk)$ where

$$\pi = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \qquad (O_\lambda(1))$$

A pairing-based SNARK

Example: Groth16 [Gro16]

Given an instance $\Phi = (a_0, \ldots, a_\ell) \in \mathbb{F}_r^\ell$ of a public NP program F

- Setup: $(pk, vk) \leftarrow S(F, \tau, 1^\lambda)$ where

$$vk = (vk_{\alpha,\beta}, \{vk_{\pi_i}\}_{i=0}^{\ell}, vk_\gamma, vk_\delta) \in \mathbb{G}_T \times \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2 \times \mathbb{G}_2$$

- Prove: $\pi \leftarrow P(\Phi, w, pk)$ where

$$\pi = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \qquad (O_\lambda(1))$$

- Verify: $0/1 \leftarrow V(\Phi, \pi, vk)$ where V is

$$e(A, B) = vk_{\alpha,\beta} \cdot e(vk_x, vk_\gamma) \cdot e(C, vk_\delta) \qquad (O_\lambda(|\Phi|)) \qquad (1)$$

and $vk_x = \sum_{i=0}^{\ell} [a_i] vk_{\pi_i}$ depends only on the instance Φ and $vk_{\alpha,\beta} = e(vk_\alpha, vk_\beta)$ can be computed in the trusted setup for $(vk_\alpha, vk_\beta) \in \mathbb{G}_1 \times \mathbb{G}_2$.

Overview

Applications

- Privacy: Monero, zcash, Aleo... or Tornado cash...
- Scalability: Mina... or Linea, Aztec...




Monero and Zcash

- Layer-1 blockchains with privacy by design.
- Use ZK proofs to hide transaction data (sender/receiver/amount).

Tornado Cash

- A privacy application on Ethereum (not a base layer).
- Uses ZK proofs to break the link between deposits and withdrawals.

Aleo (ZEXE)

- Uses ZK proofs to hide program executions and state updates.
- ZEXE enables private computations with verifiable correctness.

Mina

- ZK proofs for scalability.
- Fixed-size blockchain via recursive proofs.

Linea and Aztec

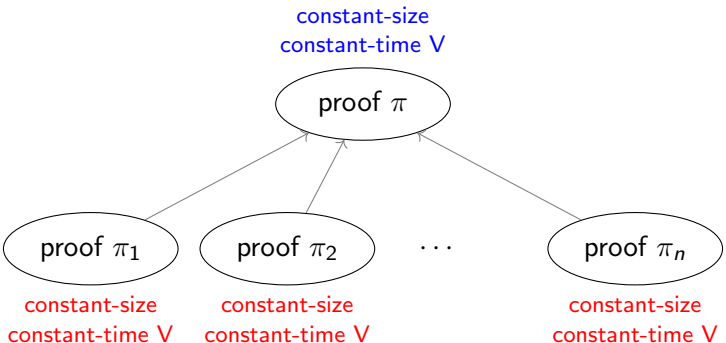
- Layer-2 systems using ZK proofs to scale Ethereum.
- Validity proofs compress many transactions into one.

Ethproofs

- Real-time Ethereum L1 proving.

Overview

Proof composition



Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)

Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**

Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**
- Proof composition for better confidentiality and scalability $\rightarrow$ 2-chains and 2-cycles **[CANS 2020, EuroCrypt 2022, DCC 2022, JoC 2024]**

Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**
- Proof composition for better confidentiality and scalability $\rightarrow$ 2-chains and 2-cycles **[CANS 2020, EuroCrypt 2022, DCC 2022, JoC 2024]**
- Pairings in R1CS for fast generation of the composed proof **[ACNS 2023]**

Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**
- Proof composition for better confidentiality and scalability $\rightarrow$ 2-chains and 2-cycles **[CANS 2020, EuroCrypt 2022, DCC 2022, JoC 2024]**
- Pairings in R1CS for fast generation of the composed proof **[ACNS 2023]**
- Multi-scalar multiplication for fast generation of proofs **[TCHES 2023, ZPRIZE winner]**

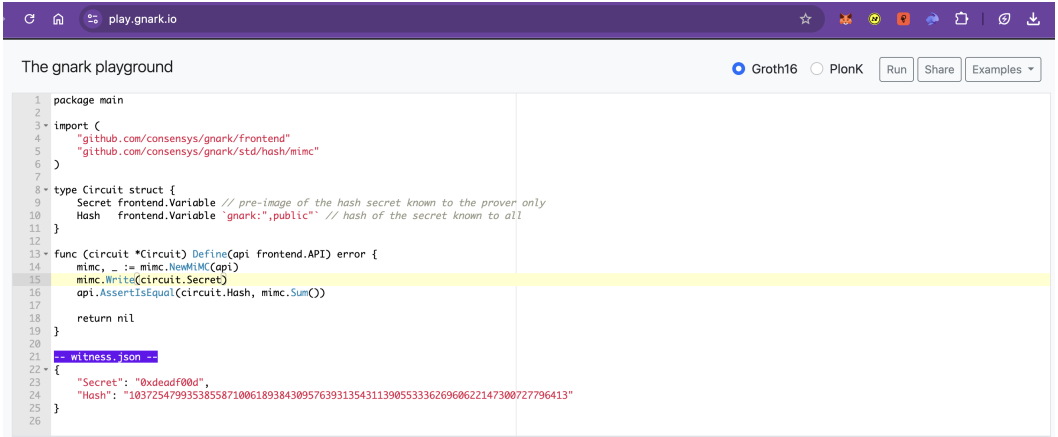
Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**
- Proof composition for better confidentiality and scalability $\rightarrow$ 2-chains and 2-cycles **[CANS 2020, EuroCrypt 2022, DCC 2022, JoC 2024]**
- Pairings in R1CS for fast generation of the composed proof **[ACNS 2023]**
- Multi-scalar multiplication for fast generation of proofs **[TCHES 2023, ZPRIZE winner]**
- Scalar multiplications in SN(T)ARKs **[LatinCrypt 2025]**

Some contributions

- Blockchain limitations: confidentiality and scalability
- pairing-based zk-SNARKs are a solution (constant-size proof and fast verification)
- What are SNARK-friendly curves? Fast arithmetic? **[DCC 2022, AfricaCrypt 2022]**
- Proof composition for better confidentiality and scalability $\rightarrow$ 2-chains and 2-cycles **[CANS 2020, EuroCrypt 2022, DCC 2022, JoC 2024]**
- Pairings in R1CS for fast generation of the composed proof **[ACNS 2023]**
- Multi-scalar multiplication for fast generation of proofs **[TCHES 2023, ZPRIZE winner]**
- Scalar multiplications in SN(T)ARKs **[LatinCrypt 2025]**
- Implementations: **gnark, linea, arkworks, sonobe, ...**

https://play.gnark.io



Thank you

- website: `https://yelhousni.github.io` or `https://yelhousni.eth.limo`
- email: `youssef.elhousni@consensys.net`
- telegram: `@ElMarroqui`
- x: `@YoussefElHoun3`
- github: `@yelhousni`

References I


Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer.
From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again.
In Shafi Goldwasser, editor, *ITCS 2012*, pages 326–349. ACM, January 2012.


Manuel Blum, Paul Feldman, and Silvio Micali.
Non-interactive zero-knowledge and its applications (extended abstract).
In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.


Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward.
Marlin: Preprocessing zkSNARKs with universal and updatable SRS.
In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Cham, May 2020.

References II

Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova.
Quadratic span programs and succinct NIZKs without PCPs.
In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Berlin, Heidelberg, May 2013.

Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers.
Updatable and universal common reference strings with applications to zk-SNARKs.
In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Cham, August 2018.

Shafi Goldwasser, Silvio Micali, and Charles Rackoff.
The knowledge complexity of interactive proof-systems (extended abstract).
In *17th ACM STOC*, pages 291–304. ACM Press, May 1985.

References III


Jens Groth.
Pairing-based non-interactive zero-knowledge proofs (invited talk).
In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *PAIRING 2010*, volume 6487 of *LNCS*, page 206. Springer, Berlin, Heidelberg, December 2010.


Jens Groth.
On the size of pairing-based non-interactive arguments.
In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Berlin, Heidelberg, May 2016.


Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru.
PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge.
Cryptology ePrint Archive, Report 2019/953, 2019.

References IV


Joe Kilian.
A note on efficient zero-knowledge proofs and arguments (extended abstract).
In *24th ACM STOC*, pages 723–732. ACM Press, May 1992.


Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn.
Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings.
In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019.


Silvio Micali.
CS proofs (extended abstracts).
In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994.