How (Not) to Simulate PLONK

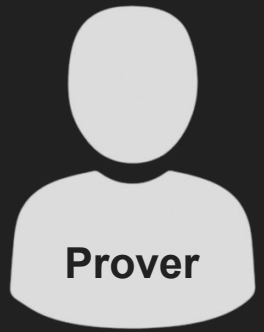
https://ia.cr/2024/848

Marek Sefranek
TU Wien

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Zero-Knowledge Proof

- Let *R* be an NP relation and *L* the corresponding language

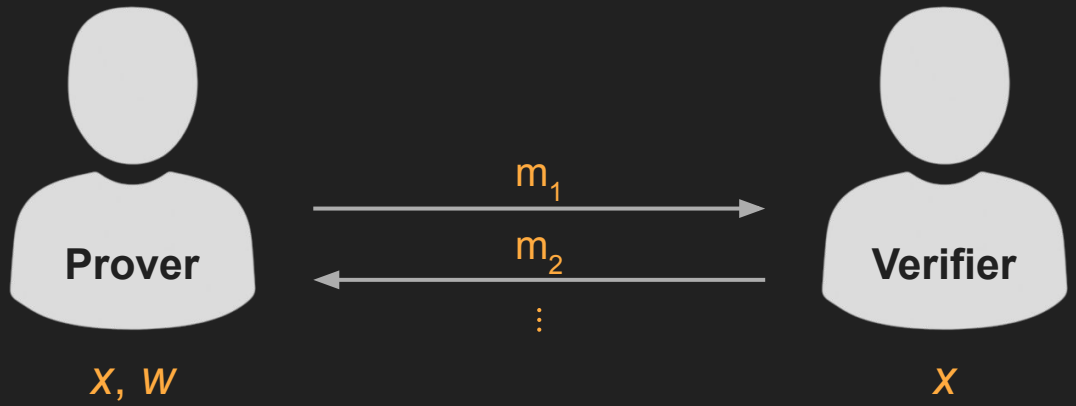


Zero-Knowledge Proof

- Let R be an NP relation and L the corresponding language
- Prove statement $x \in L$ without revealing witness w

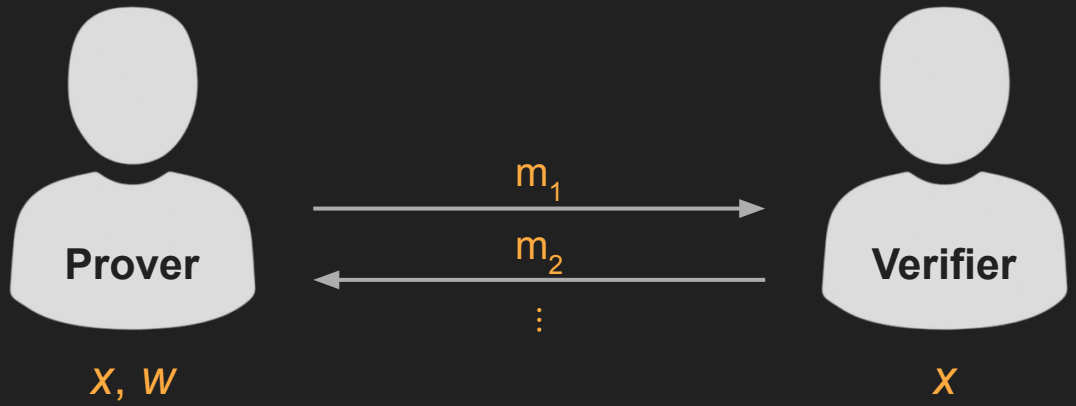
Zero-Knowledge Proof

- Let R be an NP relation and L the corresponding language
- Prove statement $x \in L$ without revealing witness w



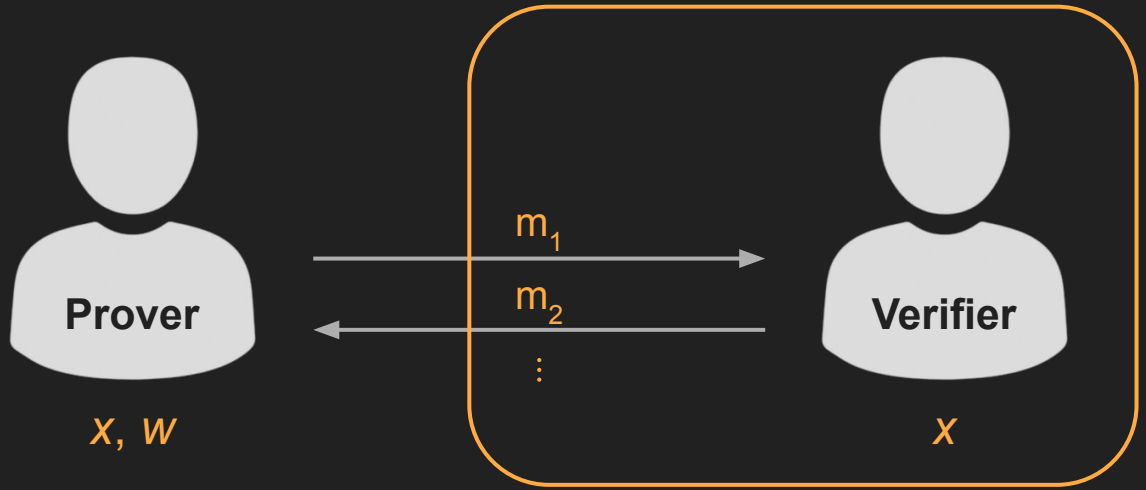
Zero-Knowledge Proof – Properties

- Completeness/Soundness: statement true ⟺ verifier accepts



Zero-Knowledge Proof – Properties

- Completeness/Soundness: statement true ⟺ verifier accepts
- Zero Knowledge: can efficiently simulate view of verifier only given *x*

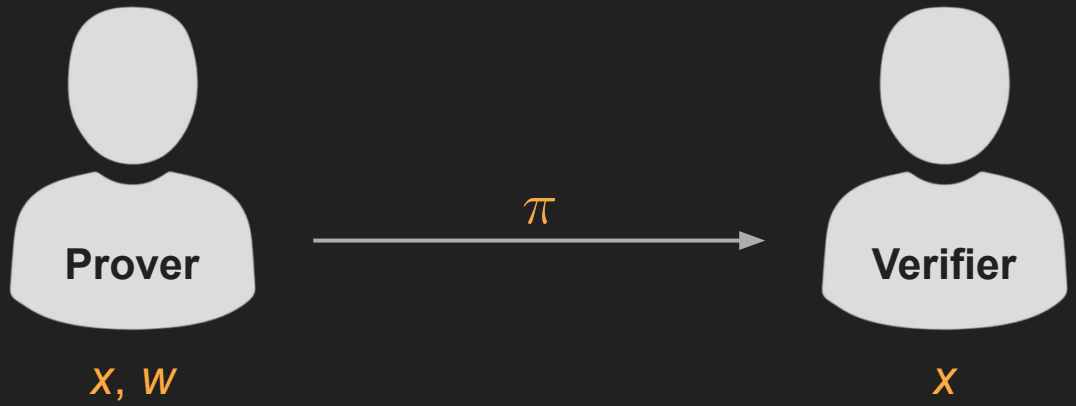


zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge

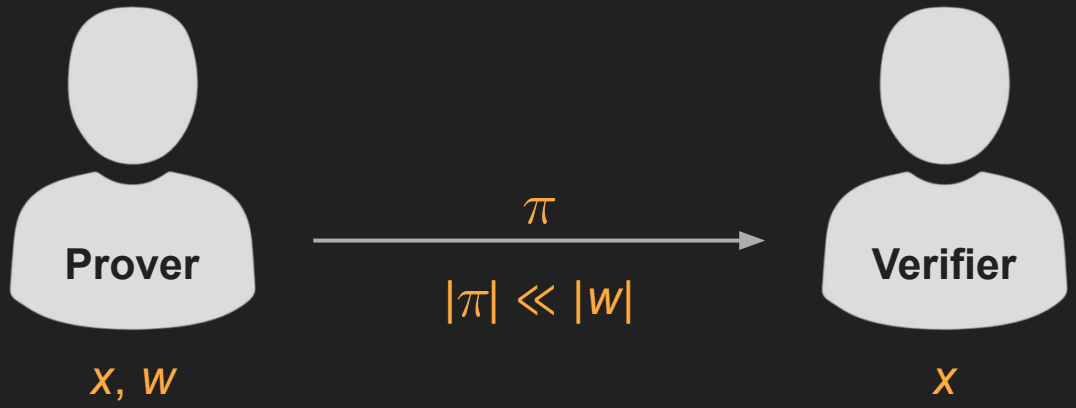
zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge



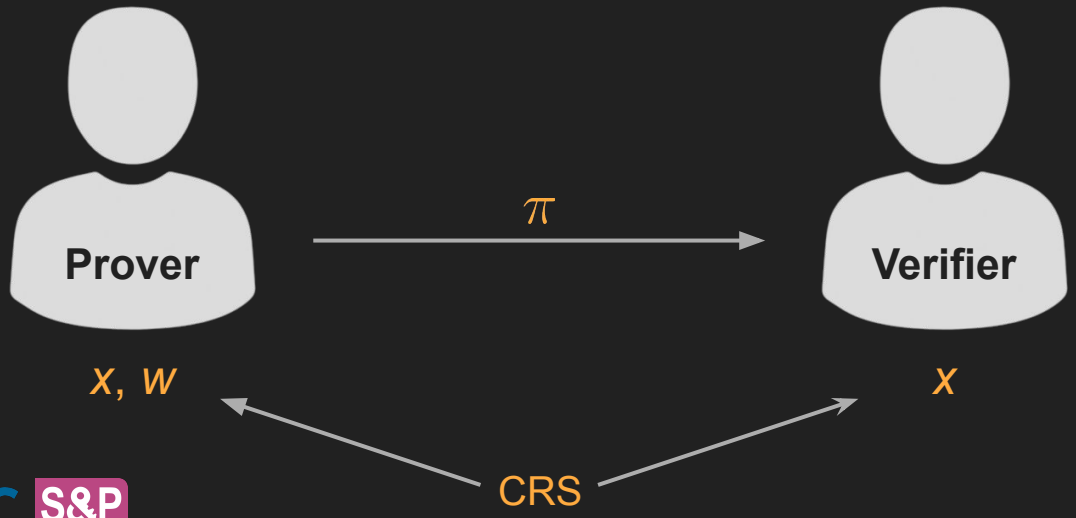
zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge



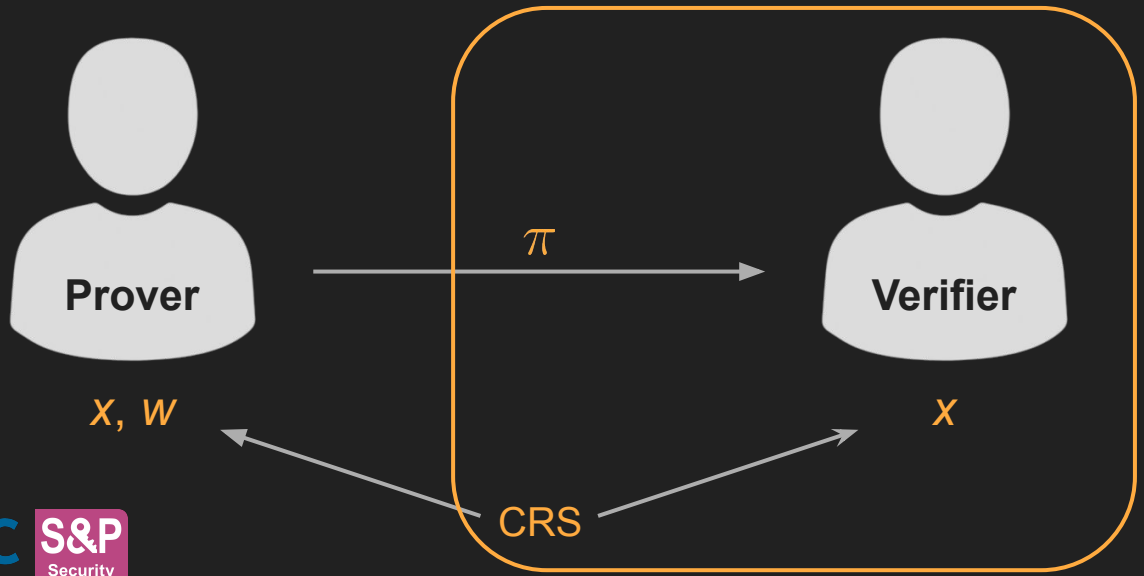
zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge
- Need trusted setup: common reference string (CRS)



zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge
- Need trusted setup: common reference string (CRS)
- Zero knowledge: simulator can set up CRS, knowing “trapdoor”



PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]

PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]
- A proof is ≈0.5 kB and can be verified in milliseconds

PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]
- A proof is ≈0.5 kB and can be verified in milliseconds
- Universal & updatable structured reference string (SRS)

PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]
- A proof is ≈0.5 kB and can be verified in milliseconds
- Universal & updatable structured reference string (SRS)
- Knowledge sound in AGM + ROM (or just ROM [LPS24])

PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]
- A proof is ≈0.5 kB and can be verified in milliseconds
- Universal & updatable structured reference string (SRS)
- Knowledge sound in AGM + ROM (or just ROM [LPS24])
- Supports custom gates and lookup gates

PLONK

- State-of-the-art zk-SNARK by Gabizon, Williamson & Ciobotaru [GWC19]
- A proof is ≈0.5 kB and can be verified in milliseconds
- Universal & updatable structured reference string (SRS)
- Knowledge sound in AGM + ROM (or just ROM [LPS24])
- Supports custom gates and lookup gates
- Deployed in a variety of real-world projects

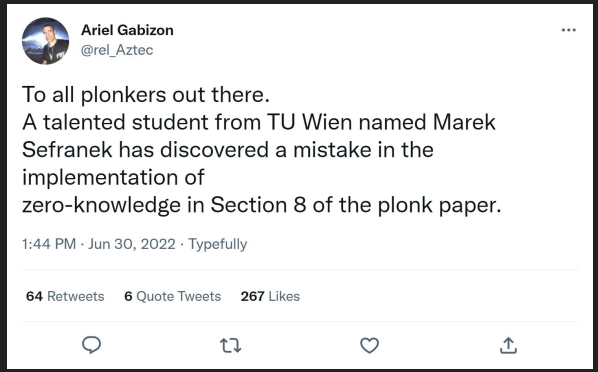
TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Main Contribution

- But no proof that PLONK is zero-knowledge!

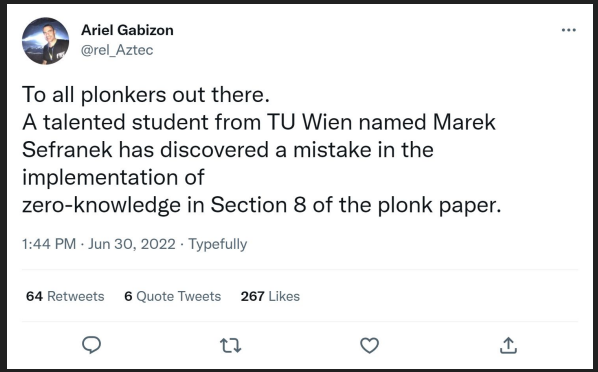
Main Contribution

- But no proof that PLONK is zero-knowledge!
- Found vulnerability in its ZK implementation & proposed fix



Main Contribution

- But no proof that PLONK is zero-knowledge!
- Found vulnerability in its ZK implementation & proposed fix



- Formal security proof that it now achieves statistical ZK

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Zero Knowledge – Definition

- Statistical ZK: there exists a PPT simulator $\mathcal{S}$ such that for all $n = \text{poly}(\lambda)$ and all adversaries $\mathcal{A}$

Zero Knowledge – Definition

- Statistical ZK: there exists a PPT simulator $\mathcal{S}$ such that for all $n = \mathsf{poly}(\lambda)$ and all adversaries $\mathcal{A}$

$$\left| \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{srs}, \tau) \leftarrow \mathcal{S}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathcal{S}(\tau, \mathsf{pp}, x); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] \right| \leq \mathsf{negl}(\lambda)$$

Zero Knowledge – Definition

- Statistical ZK: there exists a PPT simulator $\mathcal{S}$ such that for all $n = \mathsf{poly}(\lambda)$ and all adversaries $\mathcal{A}$

$$\left| \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{srs}, \tau) \leftarrow \mathcal{S}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathcal{S}(\tau, \mathsf{pp}, x); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] \right| \leq \mathsf{negl}(\lambda)$$

Zero Knowledge – Definition

- Perfect ZK: there exists a PPT simulator $\mathcal{S}$ such that for all $n = \mathsf{poly}(\lambda)$ and all adversaries $\mathcal{A}$

$$\left| \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{srs}, \tau) \leftarrow \mathcal{S}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathcal{S}(\tau, \mathsf{pp}, x); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] \right| = 0$$

Zero Knowledge – Definition

- Computational ZK: there exists a PPT simulator $\mathcal{S}$ such that for all $n = \mathsf{poly}(\lambda)$ and all PPT adversaries $\mathcal{A}$

$$\left| \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{srs}, \tau) \leftarrow \mathcal{S}(1^\lambda, n); \\ (i, x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathcal{S}(\tau, \mathsf{pp}, x); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w) \in \mathcal{R}_n \end{array}\right] \right| \leq \mathsf{negl}(\lambda)$$

KZG Polynomial Commitment [KZG10]

- Succinctly commit to a polynomial $f \in \mathbb{F}[X]$

KZG Polynomial Commitment [KZG10]

- Succinctly commit to a polynomial $f \in \mathbb{F}[X]$
- Later prove evaluations, i.e., for any point $x \in \mathbb{F}$ show that $f(x) = y$

KZG Polynomial Commitment [KZG10]

- Succinctly commit to a polynomial $f \in \mathbb{F}[X]$
- Later prove evaluations, i.e., for any point $x \in \mathbb{F}$ show that $f(x) = y$
- SRS: $\left(g_1, g_1^{\tau}, g_1^{\tau^2}, \ldots, g_1^{\tau^d}, g_2, g_2^{\tau}\right)$ for uniform “trapdoor” $\tau \in \mathbb{F}$

KZG Polynomial Commitment [KZG10]

- Succinctly commit to a polynomial $f \in \mathbb{F}[X]$
- Later prove evaluations, i.e., for any point $x \in \mathbb{F}$ show that $f(x) = y$
- SRS: $\left(g_1, g_1^{\tau}, g_1^{\tau^2}, \ldots, g_1^{\tau^d}, g_2, g_2^{\tau}\right)$ for uniform "trapdoor" $\tau \in \mathbb{F}$
- A commitment to a polynomial $f(X) = \sum_{i=0}^{d} f_i X^i \in \mathbb{F}[X]$ is

KZG Polynomial Commitment [KZG10]

- Succinctly commit to a polynomial $f \in \mathbb{F}[X]$
- Later prove evaluations, i.e., for any point $x \in \mathbb{F}$ show that $f(x) = y$
- SRS: $\left(g_1, g_1^{\tau}, g_1^{\tau^2}, \ldots, g_1^{\tau^d}, g_2, g_2^{\tau}\right)$ for uniform "trapdoor" $\tau \in \mathbb{F}$
- A commitment to a polynomial $f(X) = \sum_{i=0}^{d} f_i X^i \in \mathbb{F}[X]$ is

$$c := \prod_{i=0}^{d} \left(g_1^{\tau^i}\right)^{f_i} = g_1^{\sum_{i=0}^{d} f_i \tau^i} = g_1^{f(\tau)}$$

PLONK – Simplified Overview

- For $Z(X) \coloneqq (X - \omega^1)(X - \omega^2)\cdots(X - \omega^n)$, want to show $Z(X) \mid C(X)$

PLONK – Simplified Overview

- For $Z(X) \coloneqq (X - \omega^1)(X - \omega^2)\cdots(X - \omega^n)$, want to show $Z(X) \mid C(X)$
- Prover commits to $C(X)$ and quotient polynomial $T(X)$ [KZG10]

PLONK – Simplified Overview

- For $Z(X) \coloneqq (X - \omega^1)(X - \omega^2)\cdots(X - \omega^n)$, want to show $Z(X) \mid C(X)$
- Prover commits to $C(X)$ and quotient polynomial $T(X)$ [KZG10]
- Its degree is $3n$, where n is the number of gates

PLONK – Simplified Overview

- For $Z(X) \coloneqq (X - \omega^1)(X - \omega^2)\cdots(X - \omega^n)$, want to show $Z(X) \mid C(X)$
- Prover commits to $C(X)$ and quotient polynomial $T(X)$ [KZG10]
- Its degree is $3n$, where n is the number of gates
- Other polynomials have degree $n \Longrightarrow$ SRS has to be 3x as long

PLONK – Simplified Overview

- For $Z(X) \coloneqq (X - \omega^1)(X - \omega^2)\cdots(X - \omega^n)$, want to show $Z(X) \mid C(X)$
- Prover commits to $C(X)$ and quotient polynomial $T(X)$ [KZG10]
- Its degree is $3n$, where n is the number of gates
- Other polynomials have degree $n \Longrightarrow$ SRS has to be 3x as long
- To avoid this, PLONK splits T into 3 degree-n polynomials T_1, T_2, T_3 s.t.

$$T(X) = T_1(X) + X^n T_2(X) + X^{2n} T_3(X)$$

PLONK – Proof

$$\pi_{\text{PLONK}} := \left(g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}, g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \right.$$
$$\left. A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

PLONK – Proof

KZG commitments to witness polynomials

$$\pi_{\text{PLONK}} := \left(\boxed{g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}}, g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \right.$$
$$\left. A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

PLONK – Proof

KZG commitments to witness polynomials

KZG commitments to split quotient polynomial

$$\pi_{\text{PLONK}} := \left(\boxed{g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}}, \boxed{g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \right.$$
$$\left. A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PLONK – Proof

KZG commitments to witness polynomials

KZG commitment to quotient polynomial

$$\pi_{\text{PLONK}} := \left(\boxed{g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}}, \boxed{g_1^{T(\tau)}}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \right.$$
$$\left. A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

PLONK – Proof

KZG commitments to witness polynomials

KZG commitments to split quotient polynomial

Batched KZG opening proofs

$$\pi_{\text{PLONK}} := \left(\boxed{g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}}, \boxed{g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}}, \boxed{g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}}, \right.$$
$$\left. A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PLONK – Proof

KZG commitments to witness polynomials

KZG commitments to split quotient polynomial

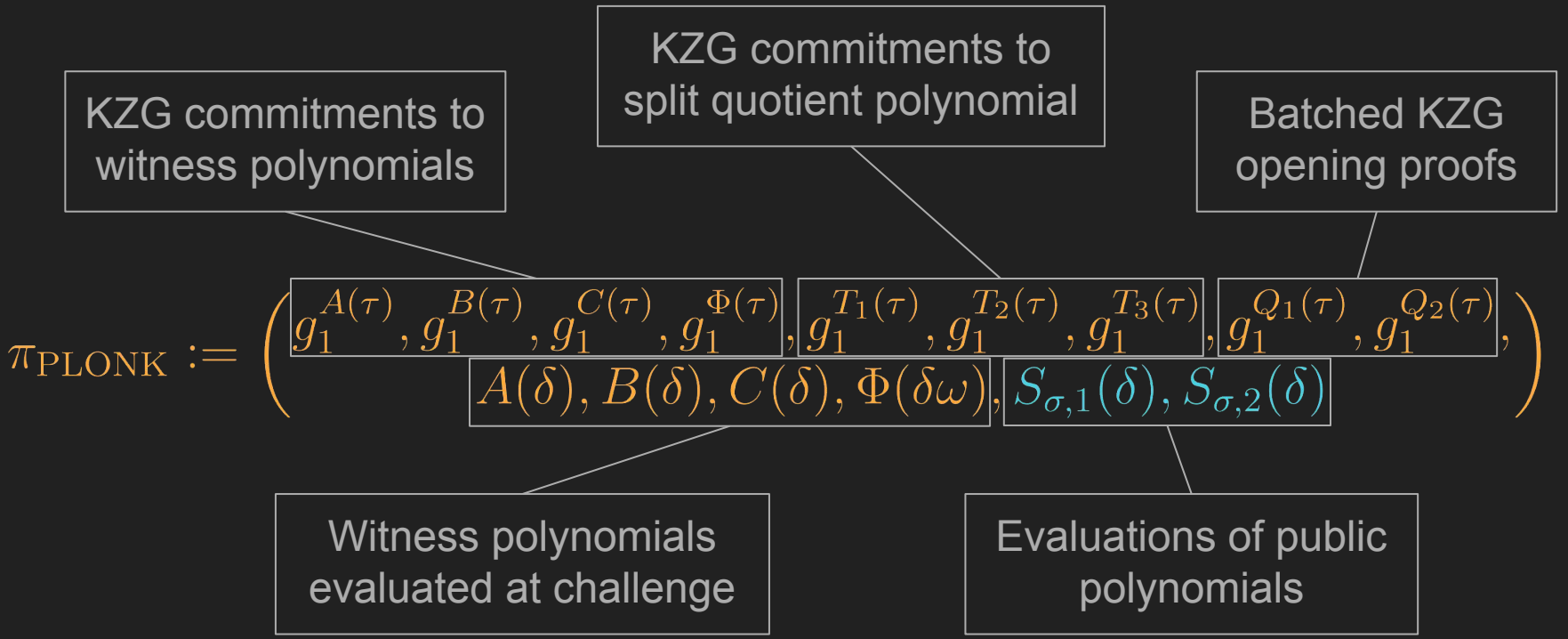
Batched KZG opening proofs

$$\pi_{\text{PLONK}} := \left(\boxed{g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}}, \boxed{g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}}, \boxed{g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}}, \right.$$
$$\left. \boxed{A(\delta), B(\delta), C(\delta), \Phi(\delta\omega)}, S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$$

Witness polynomials evaluated at challenge

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PLONK – Proof


KZG commitments to witness polynomials
KZG commitments to split quotient polynomial
Batched KZG opening proofs
$\pi_{\text{PLONK}} := \left(g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}, g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \right)$
Witness polynomials evaluated at challenge
Evaluations of public polynomials

PLONK's ZK Technique

- Let $S \subset \mathbb{F}$ and $Z_S(X) := \prod_{i \in S}(X - i)$ vanishing polynomial

PLONK's ZK Technique

- Let $S \subset \mathbb{F}$ and $Z_S(X) := \prod_{i \in S}(X - i)$ vanishing polynomial
- Take any polynomial $f \in \mathbb{F}[X]$

PLONK's ZK Technique

- Let $S \subset \mathbb{F}$ and $Z_S(X) := \prod_{i \in S}(X - i)$ vanishing polynomial
- Take any polynomial $f \in \mathbb{F}[X]$
- For a random degree-***d*** polynomial $\rho \in \mathbb{F}[X]$, define

$$\tilde{f}(X) := f(X) + Z_S(X)\rho(X)$$

PLONK's ZK Technique

- Let $S \subset \mathbb{F}$ and $Z_S(X) := \prod_{i \in S}(X - i)$ vanishing polynomial
- Take any polynomial $f \in \mathbb{F}[X]$
- For a random degree-***d*** polynomial $\rho \in \mathbb{F}[X]$, define

$$\tilde{f}(X) := f(X) + Z_S(X)\rho(X)$$

- Then evaluating $\tilde{f}$ at any ***d*** - 1 points $x_1, \ldots, x_{d-1} \in \mathbb{F} \setminus S$ results in uniform distr.

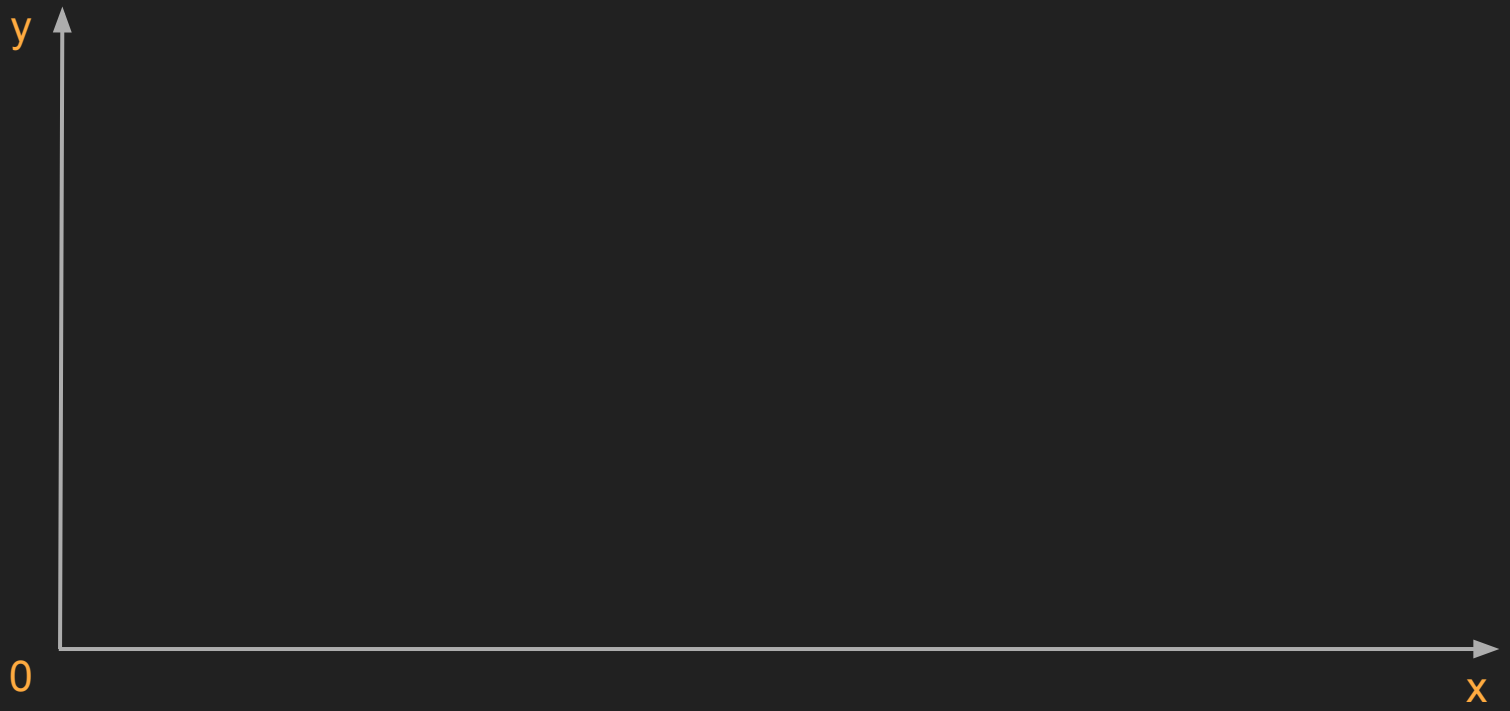
PLONK's ZK Technique

- Let $S \subset \mathbb{F}$ and $Z_S(X) := \prod_{i \in S}(X - i)$ vanishing polynomial
- Take any polynomial $f \in \mathbb{F}[X]$
- For a random degree-***d*** polynomial $\rho \in \mathbb{F}[X]$, define

$$\tilde{f}(X) := f(X) + Z_S(X)\rho(X)$$

- Then evaluating $\tilde{f}$ at any ***d*** - 1 points $x_1, \ldots, x_{d-1} \in \mathbb{F} \setminus S$ results in uniform distr.
- Note that $Z_S \mid \tilde{f}$ if and only if $Z_S \mid f$

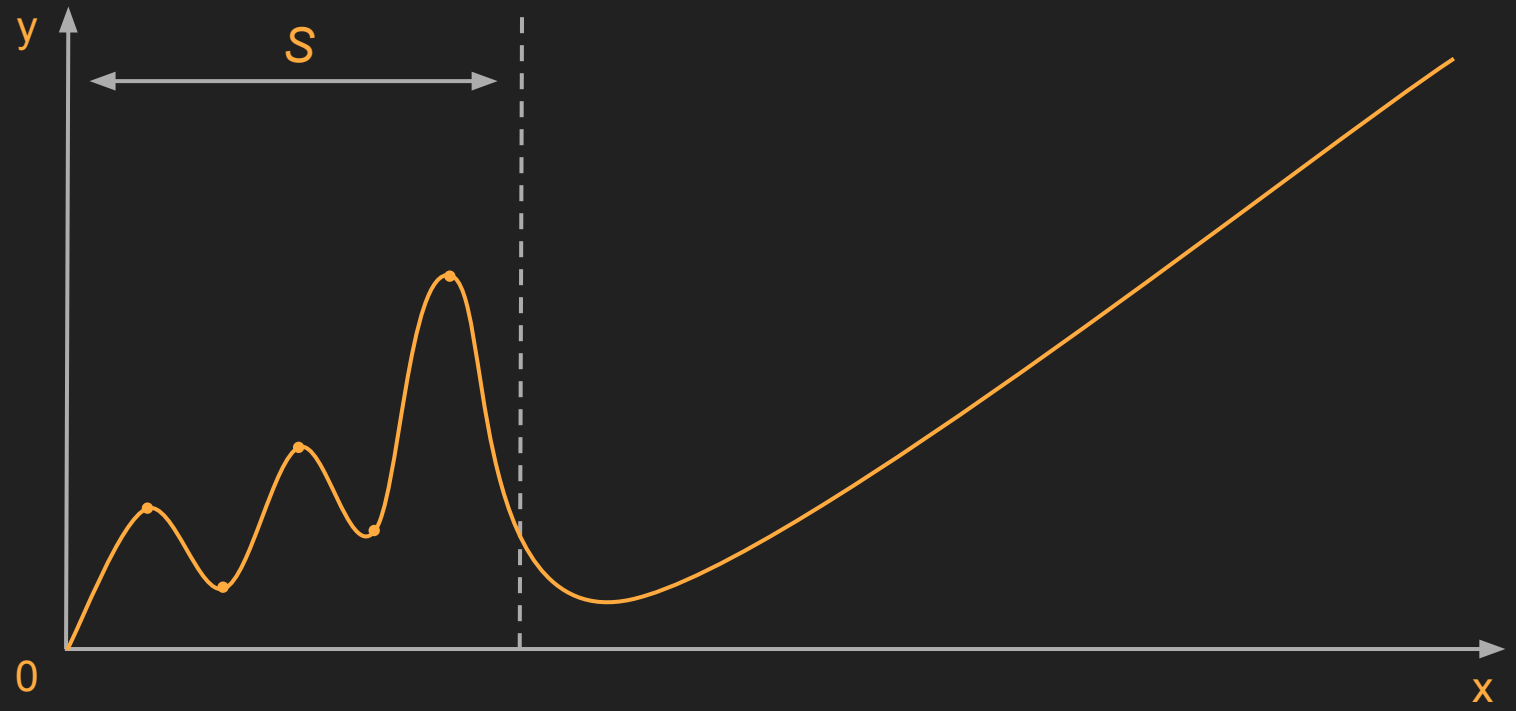
PLONK’s ZK Technique Visualized



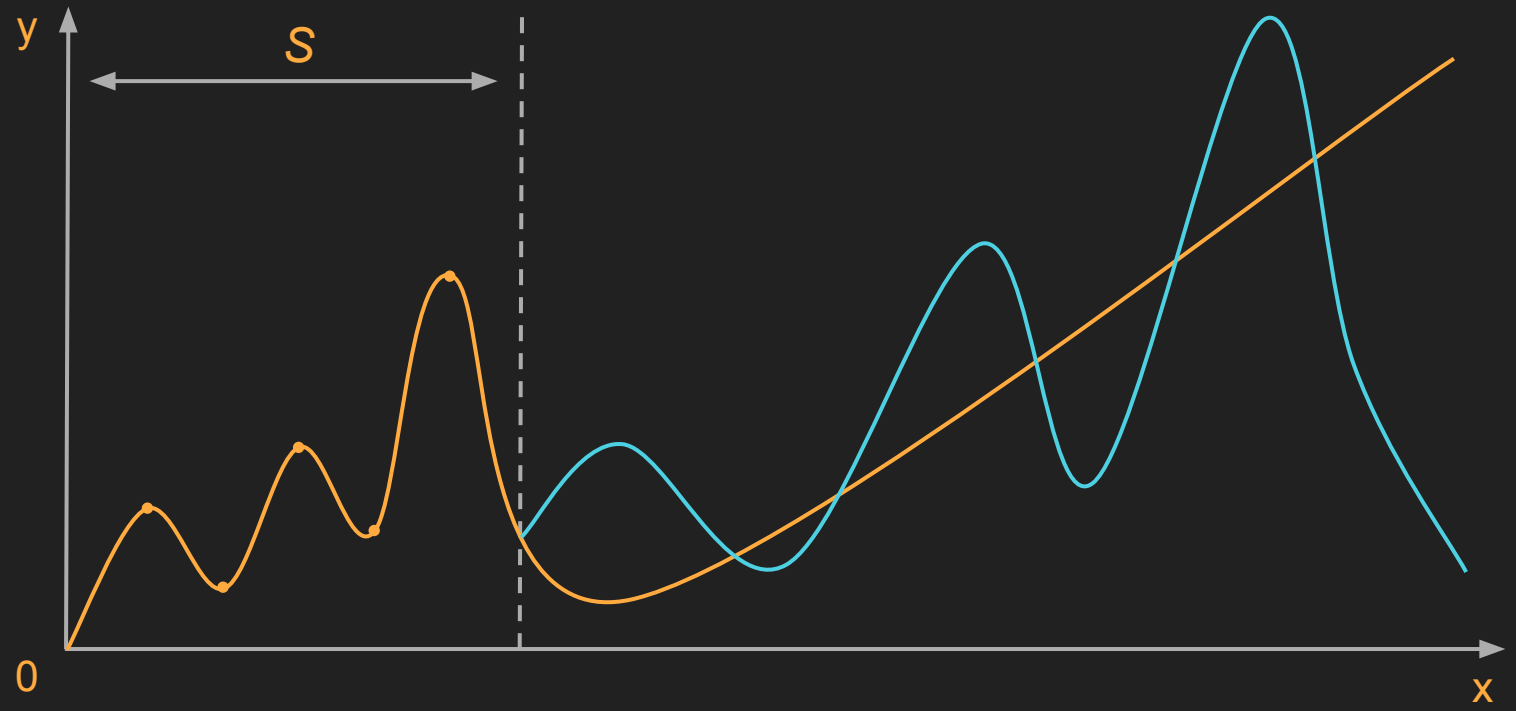
PLONK’s ZK Technique Visualized

PLONK's ZK Technique Visualized

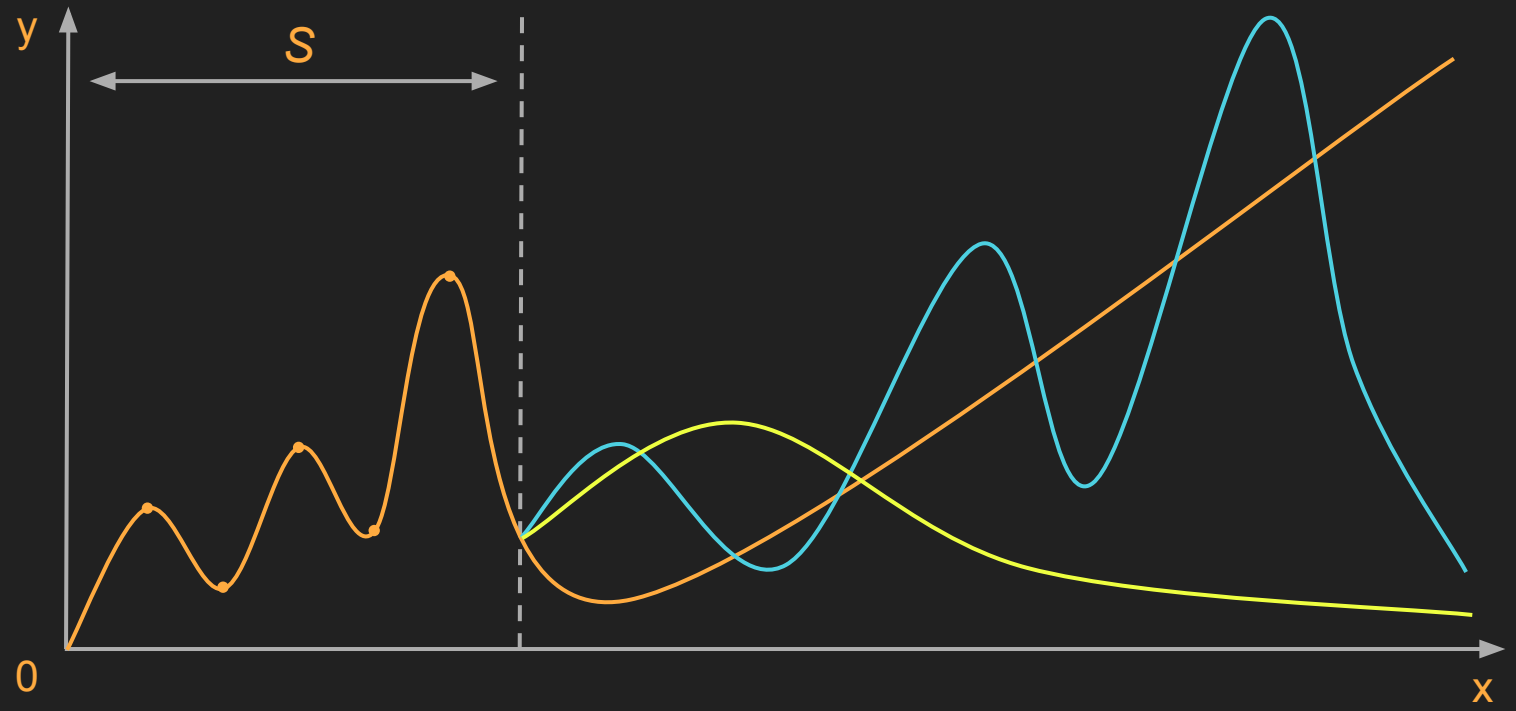
PLONK's ZK Technique Visualized



PLONK's ZK Technique Visualized



PLONK's ZK Technique Visualized



PLONK’s ZK Technique in Action

- Prover randomizes witness polynomials A, B, C, Φ

PLONK's ZK Technique in Action

- Prover randomizes witness polynomials A, B, C, Φ
- Pick random $\rho_1, \rho_2 \in \mathbb{F}$ and define $A(X) \coloneqq (\rho_1 X + \rho_2)\, Z(X) + \sum_{i \in [n]} w_i\, L_i(X)$

PLONK's ZK Technique in Action

- Prover randomizes witness polynomials A, B, C, Φ
- Pick random $\rho_1, \rho_2 \in \mathbb{F}$ and define $A(X) \coloneqq (\rho_1 X + \rho_2)\, Z(X) + \sum_{i \in [n]} w_i\, L_i(X)$
- Evaluations in proof are now independent of witness!

$$\pi_{\text{PLONK}} := \left(\begin{array}{c} g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}, g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \\ A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \end{array} \right)$$

PLONK's ZK Technique in Action

- Prover randomizes witness polynomials A, B, C, Φ
- Pick random $\rho_1, \rho_2 \in \mathbb{F}$ and define $A(X) \coloneqq (\rho_1 X + \rho_2)\, Z(X) + \sum_{i \in [n]} w_i\, L_i(X)$
- Evaluations in proof are now independent of witness!

$$\pi_{\text{PLONK}} := \left(\begin{array}{c} g_1^{A(\tau)}, g_1^{B(\tau)}, g_1^{C(\tau)}, g_1^{\Phi(\tau)}, g_1^{T_1(\tau)}, g_1^{T_2(\tau)}, g_1^{T_3(\tau)}, g_1^{Q_1(\tau)}, g_1^{Q_2(\tau)}, \\ A(\delta), B(\delta), C(\delta), \Phi(\delta\omega), S_{\sigma,1}(\delta), S_{\sigma,2}(\delta) \end{array} \right)$$

- Simulator: can just pick random values

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Zero Knowledge Vulnerability

- Without splitting $T(X)$:
  - Can be simulated as $T(\tau)$ can be computed given the KZG trapdoor τ
  - Proof independent of witness

Zero Knowledge Vulnerability

- Without splitting $T(X)$:
  - Can be simulated as $T(\tau)$ can be computed given the KZG trapdoor τ
  - Proof independent of witness
- With the optimization:
  - T_1, T_2, T_3 leak too much information about $T(X)$
  - Proof no longer independent of witness!

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + X^n \; T_2(X) + X^{2n} \; T_3(X)$$

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + r_1 X^n + X^n (T_2(X) - r_1) + X^{2n} T_3(X)$$

for randomly chosen $r_1 \in \mathbb{F}$

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + r_1 X^n + X^n (T_2(X) - r_1 + r_2 X^n) + X^{2n} (T_3(X) - r_2)$$

for randomly chosen $r_1, r_2 \in \mathbb{F}$

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + r_1 X^n + X^n (T_2(X) - r_1 + r_2 X^n) + X^{2n} (T_3(X) - r_2)$$

for randomly chosen $r_1, r_2 \in \mathbb{F}$

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + r_1 X^n + X^n (T_2(X) - r_1 + r_2 X^n) + X^{2n} (T_3(X) - r_2)$$

  for randomly chosen $r_1, r_2 \in \mathbb{F}$
- Can now be simulated as the value $T(\tau)$ can be:
  1. Choose uniform values for $T_2(\tau)$ and $T_3(\tau)$
  2. Set $T_1(\tau) := T(\tau) - \tau^n T_2(\tau) - \tau^{2n} T_3(\tau)$

Zero Knowledge Fix

- Randomize T_1, T_2, T_3 so they are uniform conditioned on satisfying

$$T(X) = T_1(X) + r_1 X^n + X^n (T_2(X) - r_1 + r_2 X^n) + X^{2n} (T_3(X) - r_2)$$

  for randomly chosen $r_1, r_2 \in \mathbb{F}$
- Can now be simulated as the value $T(\tau)$ can be:
  1. Choose uniform values for $T_2(\tau)$ and $T_3(\tau)$
  2. Set $T_1(\tau) := T(\tau) - \tau^n T_2(\tau) - \tau^{2n} T_3(\tau)$
- Preserves knowledge soundness as verifier remains the same!

Attack on Old PLONK

- “Old PLONK not stat. witness indistinguishable” $\Longrightarrow$ “not stat. ZK”

Attack on Old PLONK

- "Old PLONK not stat. witness indistinguishable" $\Longrightarrow$ "not stat. ZK"

$$\left| \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w_0, w_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w_0); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w_0) \in \mathcal{R}_n \\ \wedge\, (i, x, w_1) \in \mathcal{R}_n \end{array}\right] - \Pr\left[\begin{array}{c} \mathsf{srs} \leftarrow \mathsf{Setup}(1^\lambda, n); \\ (i, x, w_0, w_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{srs}); \\ (\mathsf{pp}, \mathsf{vp}) := \mathsf{Preproc}(\mathsf{srs}, i); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pp}, x, w_1); \\ b \leftarrow \mathcal{A}(\mathsf{st}, \pi) : \\ b = 1 \wedge (i, x, w_0) \in \mathcal{R}_n \\ \wedge\, (i, x, w_1) \in \mathcal{R}_n \end{array}\right] \right| \leq \mathsf{negl}(\lambda)$$

Attack on Old PLONK

- “Old PLONK not stat. witness indistinguishable” $\Longrightarrow$ “not stat. ZK”
- Idea: Solve system of linear equations to recover blinding scalars used by prover to mask witness polynomials

Attack on Old PLONK

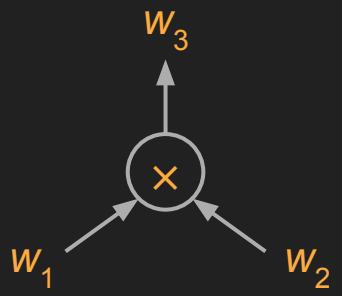
- "Old PLONK not stat. witness indistinguishable" $\Longrightarrow$ "not stat. ZK"
- Idea: Solve system of linear equations to recover blinding scalars used by prover to mask witness polynomials
- Compare against resulting values of $T_1(\tau)$, $T_2(\tau)$, $T_3(\tau)$
  1. If correct witness is used, check will always pass
  2. Otherwise, check will fail w.h.p.

Attack on Old PLONK

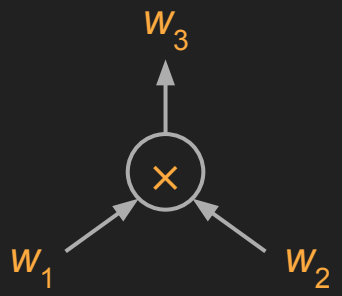
- "Old PLONK not stat. witness indistinguishable" $\Longrightarrow$ "not stat. ZK"
- Idea: Solve system of linear equations to recover blinding scalars used by prover to mask witness polynomials
- Compare against resulting values of $T_1(\tau)$, $T_2(\tau)$, $T_3(\tau)$
  1. If correct witness is used, check will always pass
  2. Otherwise, check will fail w.h.p.
- For example:
  - Prover picks random $\rho_1, \rho_2 \in \mathbb{F}$ and defines $A(X) := (\rho_1 X + \rho_2)\, Z(X) + \sum_{i\in[n]} w_i\, L_i(X)$
  - Proof reveals $A(\tau)$, $A(\delta)$ $\Longrightarrow$ system of 2 linear equations in 2 unknowns

Attack on Old PLONK – Example

- Statement: “Correctly compute one multiplication”



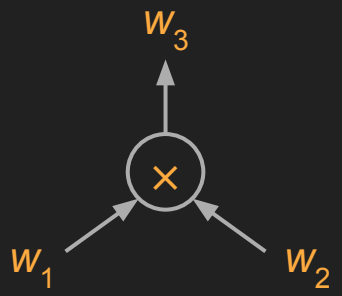
Attack on Old PLONK – Example



- Statement: “Correctly compute one multiplication”
- We get the following expression for T:

$$T(X) = (\rho_1\rho_3)X^3 + (\rho_1\rho_4 + \rho_2\rho_3 - \rho_1\rho_3 + \alpha^2\rho_7)X^2 + (w_1\rho_3 + w_2\rho_1 + \rho_2\rho_4 - \rho_1\rho_4 - \rho_2\rho_3 - \rho_5 + \alpha^2\rho_8)X + w_1\rho_4 + w_2\rho_2 - \rho_2\rho_4 - \rho_6 + \alpha^2\rho_9$$

Attack on Old PLONK – Example



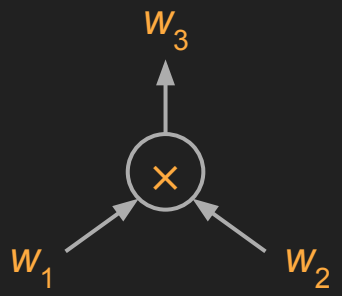
- Statement: “Correctly compute one multiplication”
- We get the following expression for T:

$$T(X) = (\rho_1\rho_3)X^3 + (\rho_1\rho_4 + \rho_2\rho_3 - \rho_1\rho_3 + \alpha^2\rho_7)X^2 + (w_1\rho_3 + w_2\rho_1 + \rho_2\rho_4 - \rho_1\rho_4 - \rho_2\rho_3 - \rho_5 + \alpha^2\rho_8)X + w_1\rho_4 + w_2\rho_2 - \rho_2\rho_4 - \rho_6 + \alpha^2\rho_9$$

- Which means:

$$T_1(X) = T(X) - (\rho_1\rho_3)X^3, \quad T_2(X) = \rho_1\rho_3, \quad T_3(X) = 0$$

Attack on Old PLONK – Example



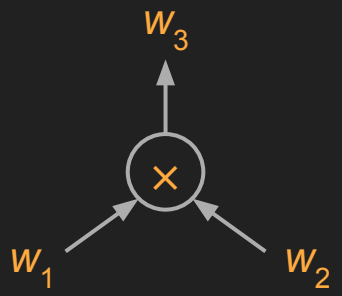
- Statement: “Correctly compute one multiplication”
- We get the following expression for T:

$$T(X) = (\rho_1\rho_3)X^3 + (\rho_1\rho_4 + \rho_2\rho_3 - \rho_1\rho_3 + \alpha^2\rho_7)X^2 + (w_1\rho_3 + w_2\rho_1 + \rho_2\rho_4 - \rho_1\rho_4 - \rho_2\rho_3 - \rho_5 + \alpha^2\rho_8)X + w_1\rho_4 + w_2\rho_2 - \rho_2\rho_4 - \rho_6 + \alpha^2\rho_9$$

- Which means:

$$T_1(X) = T(X) - (\rho_1\rho_3)X^3, \quad T_2(X) = \rho_1\rho_3, \quad T_3(X) = 0$$

Attack on Old PLONK – Example



- Statement: “Correctly compute one multiplication”
- We get the following expression for T:

$$T(X) = (\rho_1\rho_3)X^3 + (\rho_1\rho_4 + \rho_2\rho_3 - \rho_1\rho_3 + \alpha^2\rho_7)X^2 + (w_1\rho_3 + w_2\rho_1 + \rho_2\rho_4 - \rho_1\rho_4 - \rho_2\rho_3 - \rho_5 + \alpha^2\rho_8)X + w_1\rho_4 + w_2\rho_2 - \rho_2\rho_4 - \rho_6 + \alpha^2\rho_9$$

- Which means:

$$T_1(X) = T(X) - (\rho_1\rho_3)X^3, \quad T_2(X) = \rho_1\rho_3, \quad T_3(X) = 0$$

- If we use correct w_1, w_2, w_3, always get correct ρ_1, ρ_2 (compare against $T_2(\tau)$)

More in the Full Paper…

- Proof of statistical (computational) ZK in ROM (collision-resistant H)
- Unbounded attack on witness indistinguishability of old PLONK

https://ia.cr/2024/848

More in the Full Paper…

- Proof of statistical (computational) ZK in ROM (collision-resistant H)
- Unbounded attack on witness indistinguishability of old PLONK

https://ia.cr/2024/848

Thanks!

Questions?

References

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019. https://eprint.iacr.org/2019/953.

[KZG10] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-Size Commitments to Polynomials and Their Applications. In Advances in Cryptology – ASIACRYPT 2010, volume 6477 of LNCS, pages 177–194. Springer, 2010. https://doi.org/10.1007/978-3-642-17373-8_11.

[LPS24] Helger Lipmaa, Roberto Parisella, and Janno Siim. On Knowledge-Soundness of Plonk in ROM from Falsifiable Assumptions. Cryptology ePrint Archive, Paper 2024/994, 2024. https://eprint.iacr.org/2024/994.