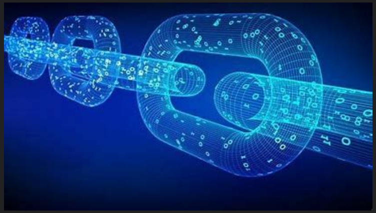
Hash functions in the context of PIOP-based SNARKs

ALPSY 2025

Marek Sefranek
TU Wien

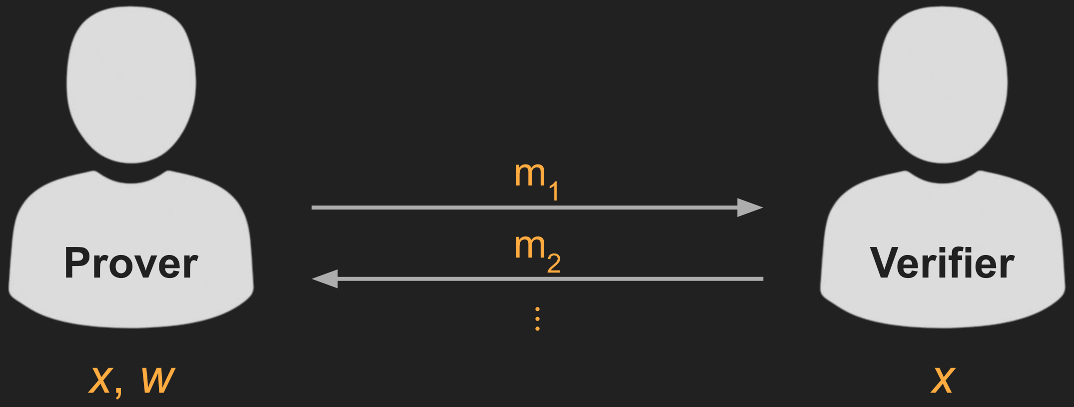
Motivation

- Zero-knowledge: prove something is true without revealing why
  - For example: prove age over certain limit ("digital ID") without revealing it
  - Comply with rules with minimal disclosure of information (GDPR)
- Applications:
  - Enforce parties follow a protocol (MPC)
  - Verifiable computation, anonymous credentials
  - Enable trust in decentralized systems such as blockchains
  - Fully anonymous cryptocurrencies, e.g. Zcash
  - …



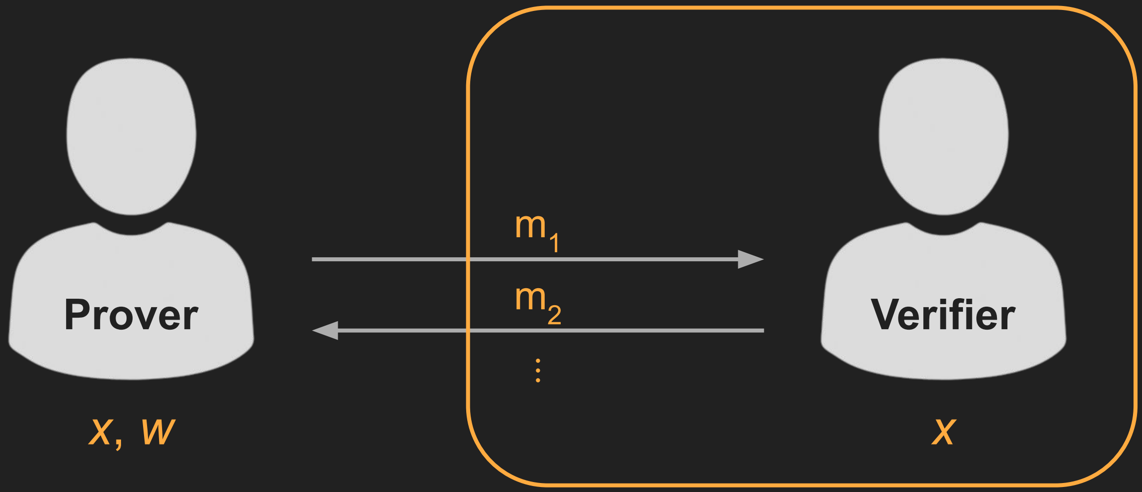
Zero-Knowledge Proof

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



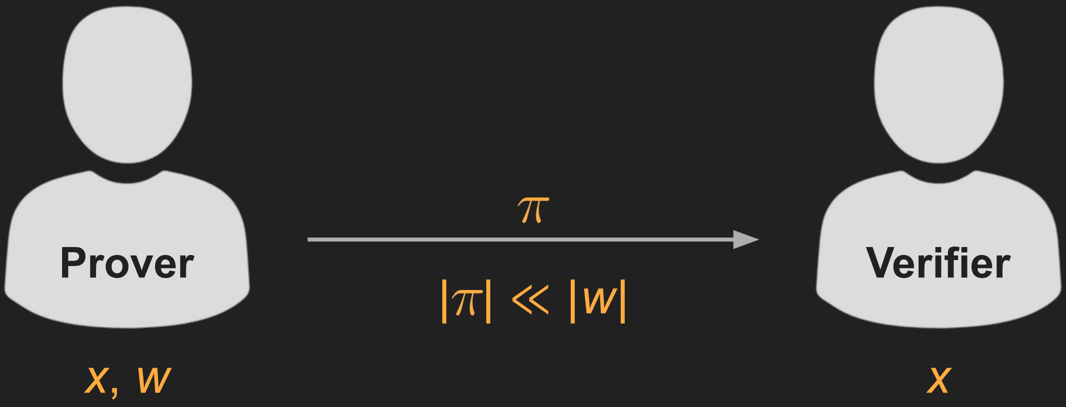
Zero-Knowledge Proof – Properties

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



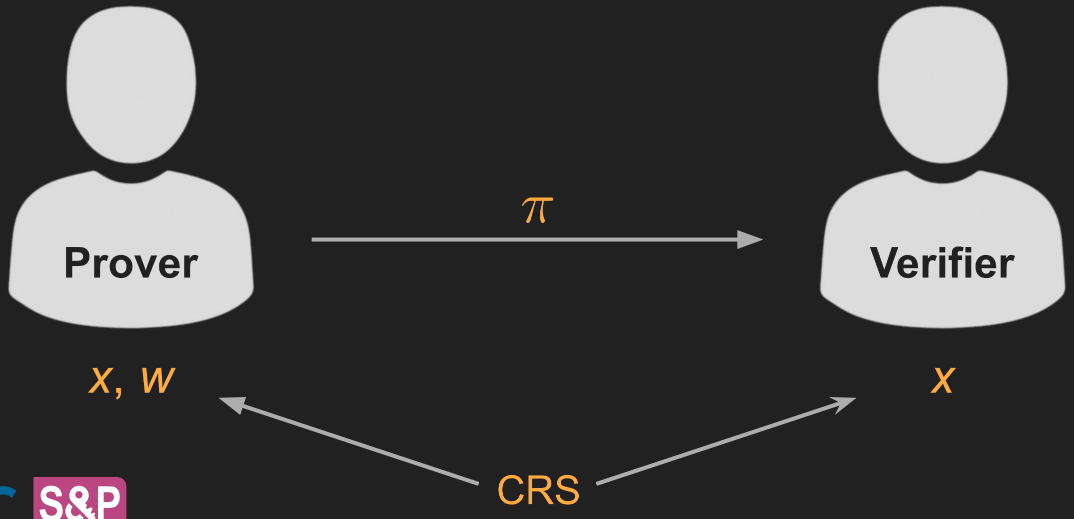
zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge



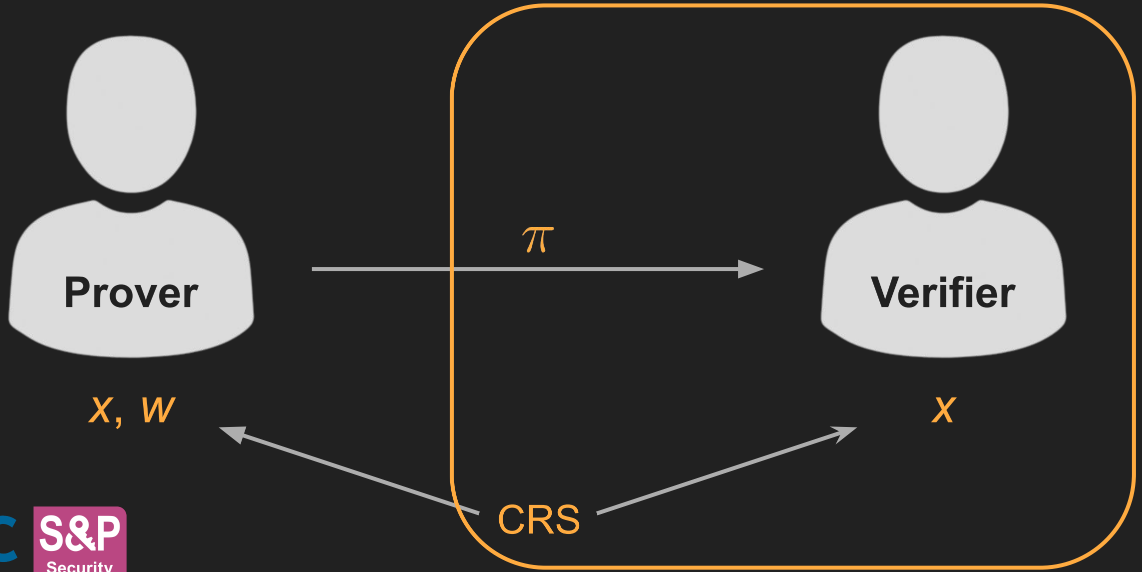
zk-SNARK

- Zero-Knowledge Succinct Non-interactive ARgument of Knowledge
- 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”

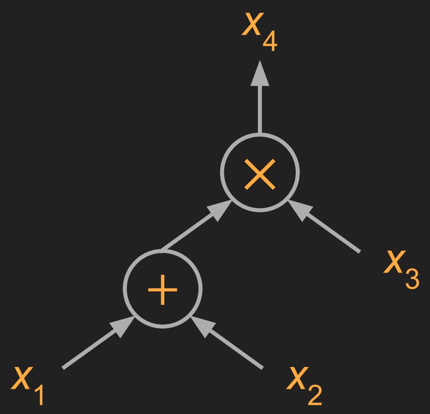


Hash Functions in SNARK Applications

- Already seen many examples throughout this workshop:
  - ZK proof of knowledge of hash preimage
  - Merkle trees, membership proofs
  - And many more applications in blockchains…
- NP relation R usually modelled via circuit satisfiability
- Arithmetic circuit over large prime field $\mathbb{F}_p$

$\Longrightarrow$ AO/algebraic hash functions preferable

$\Longrightarrow$ Low multiplicative depth desirable

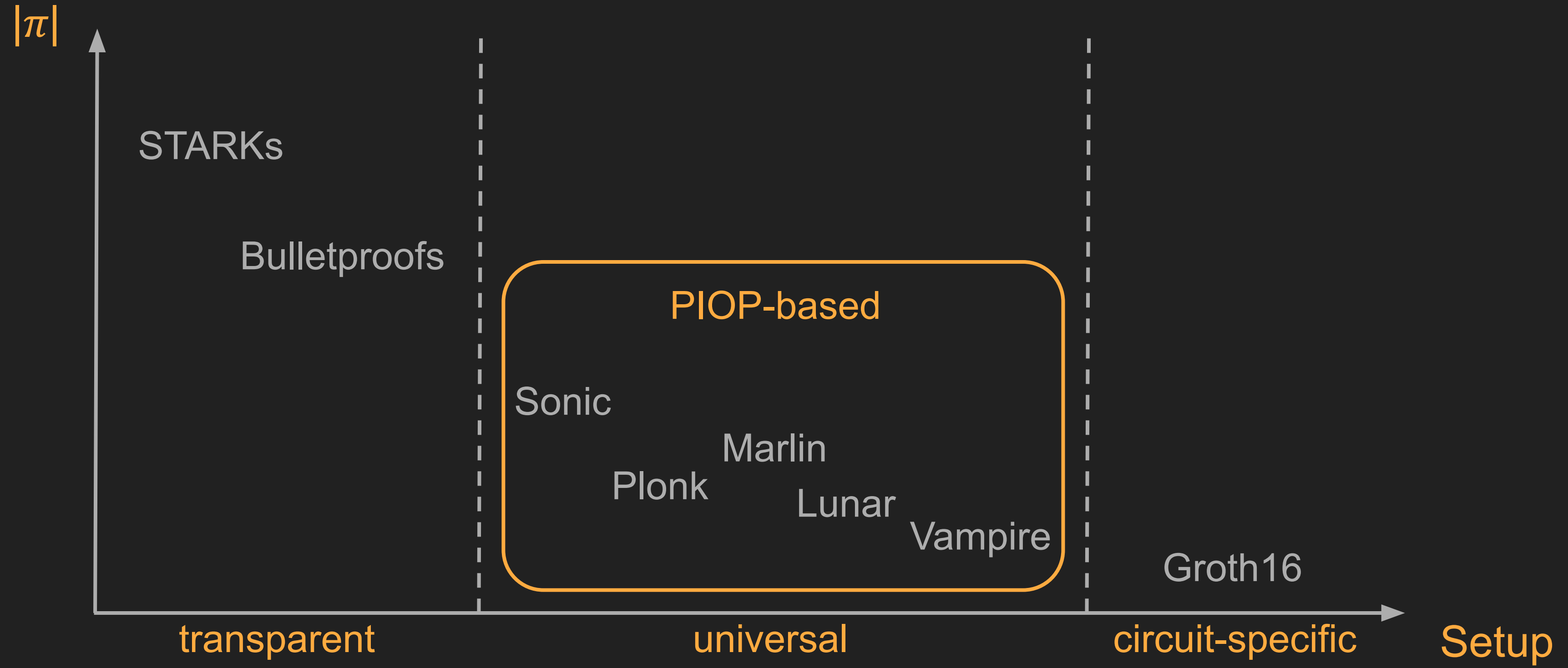


Recursive SNARKs

- Instead of computing k proofs $\pi_1, \ldots, \pi_k$ for the statements $x_1, \ldots, x_k$:
  - Compute the proof π_1 for x_1
  - Compute the proof π_2 for x_2 and the validity of π_1
  - …
  - Compute the proof π_k for x_k and the validity of π_{k-1}
- Validity of π_k implies that all the statements $x_1, \ldots, x_k$ hold
- Recursive proofs need verifier in-circuit (which calls H)
- Applications: incrementally verifiable computation, constant-size blockchains

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

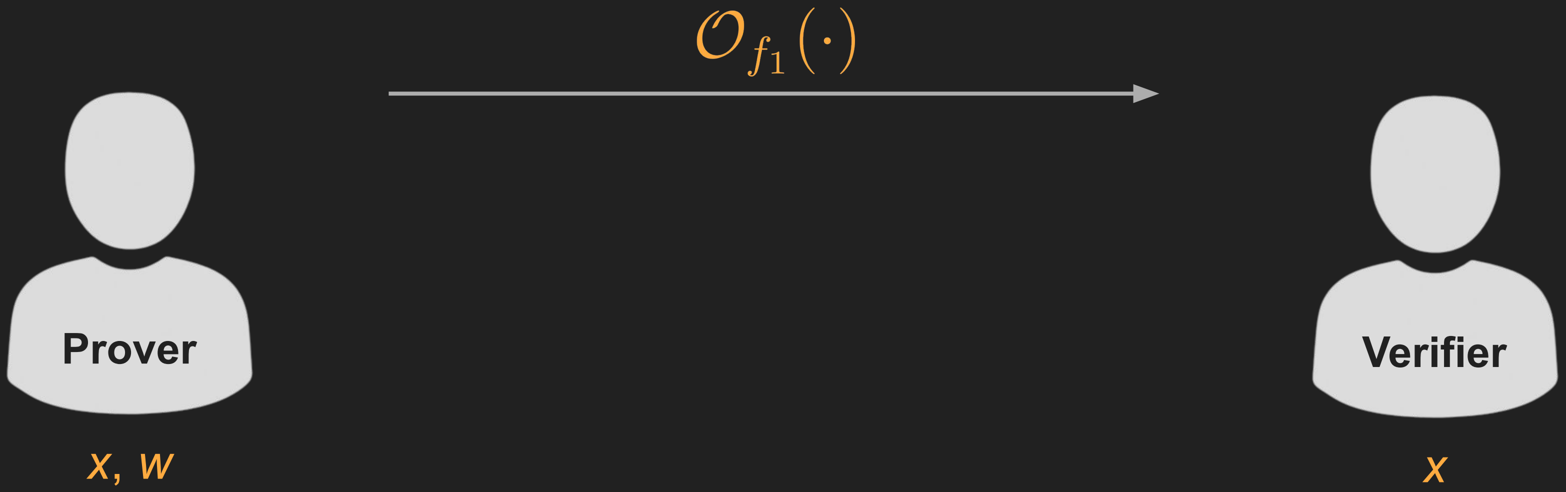
The SNARK Landscape



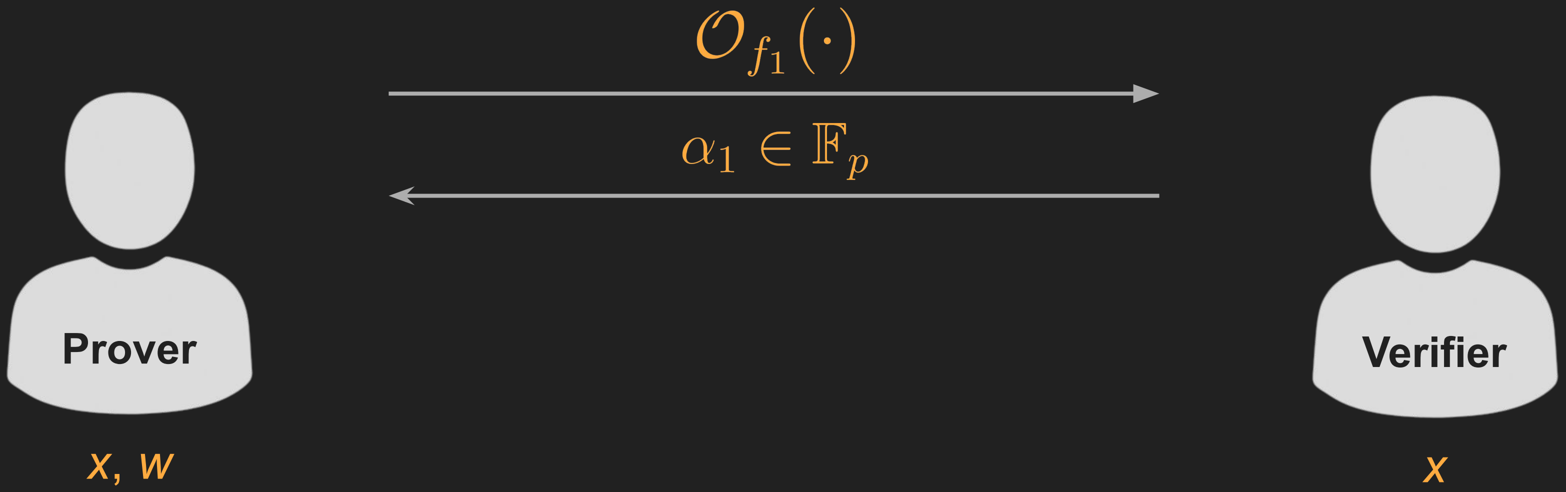
PIOPs – Polynomial Interactive Oracle Proofs

CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PIOPs – Polynomial Interactive Oracle Proofs

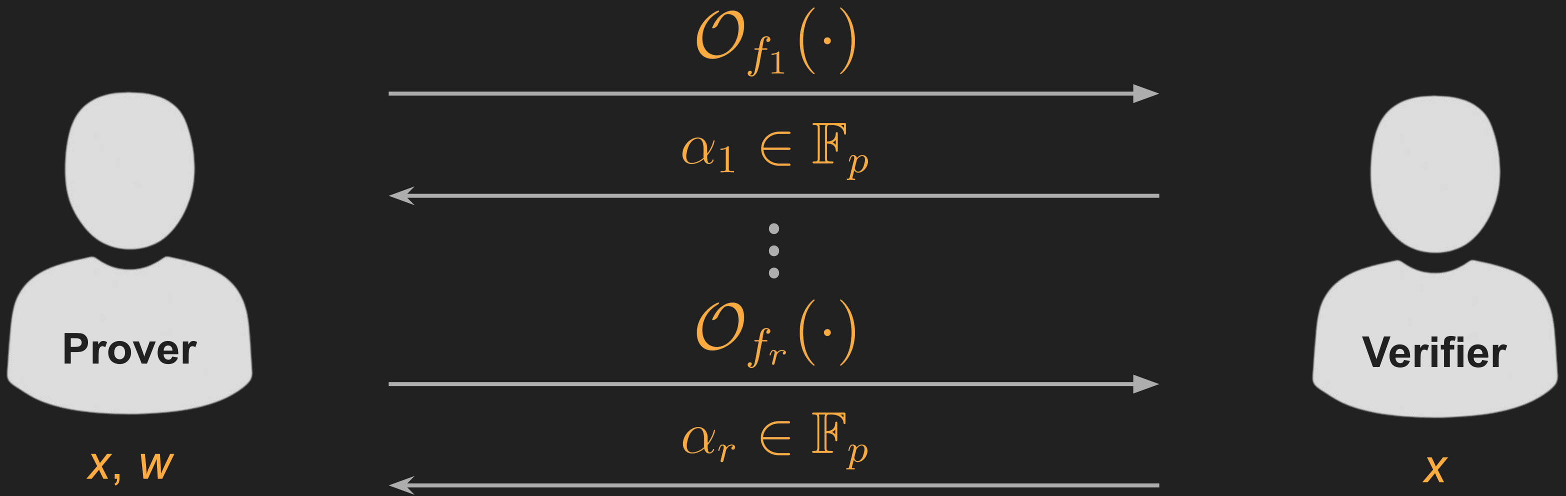


PIOPs – Polynomial Interactive Oracle Proofs

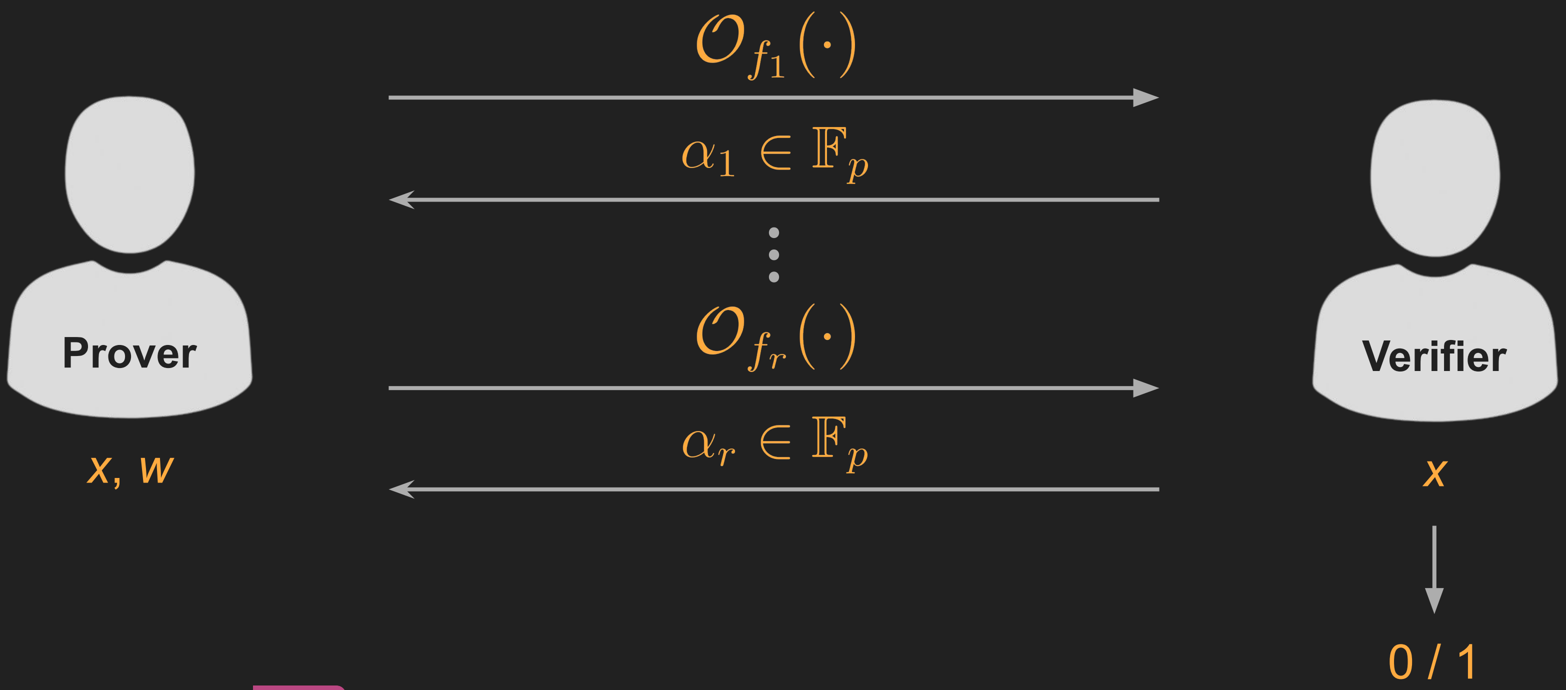


TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PIOPs – Polynomial Interactive Oracle Proofs

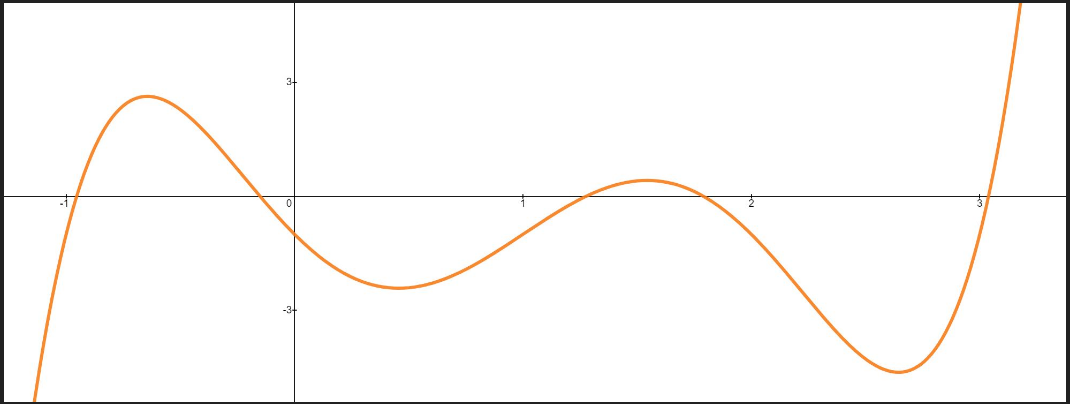


PIOPs – Polynomial Interactive Oracle Proofs



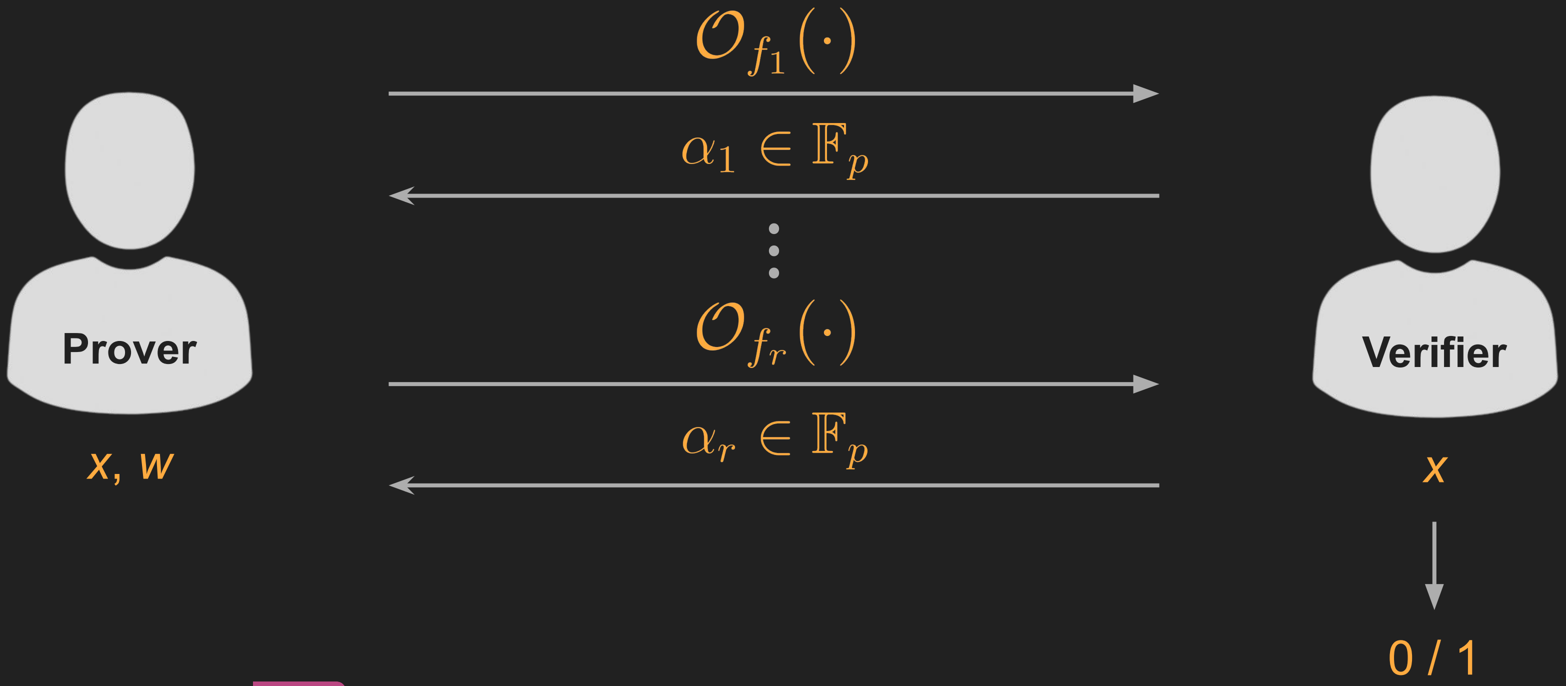
Why Polynomials?

- Schwartz–Zippel lemma: "a non-zero polynomial is non-zero almost everywhere"



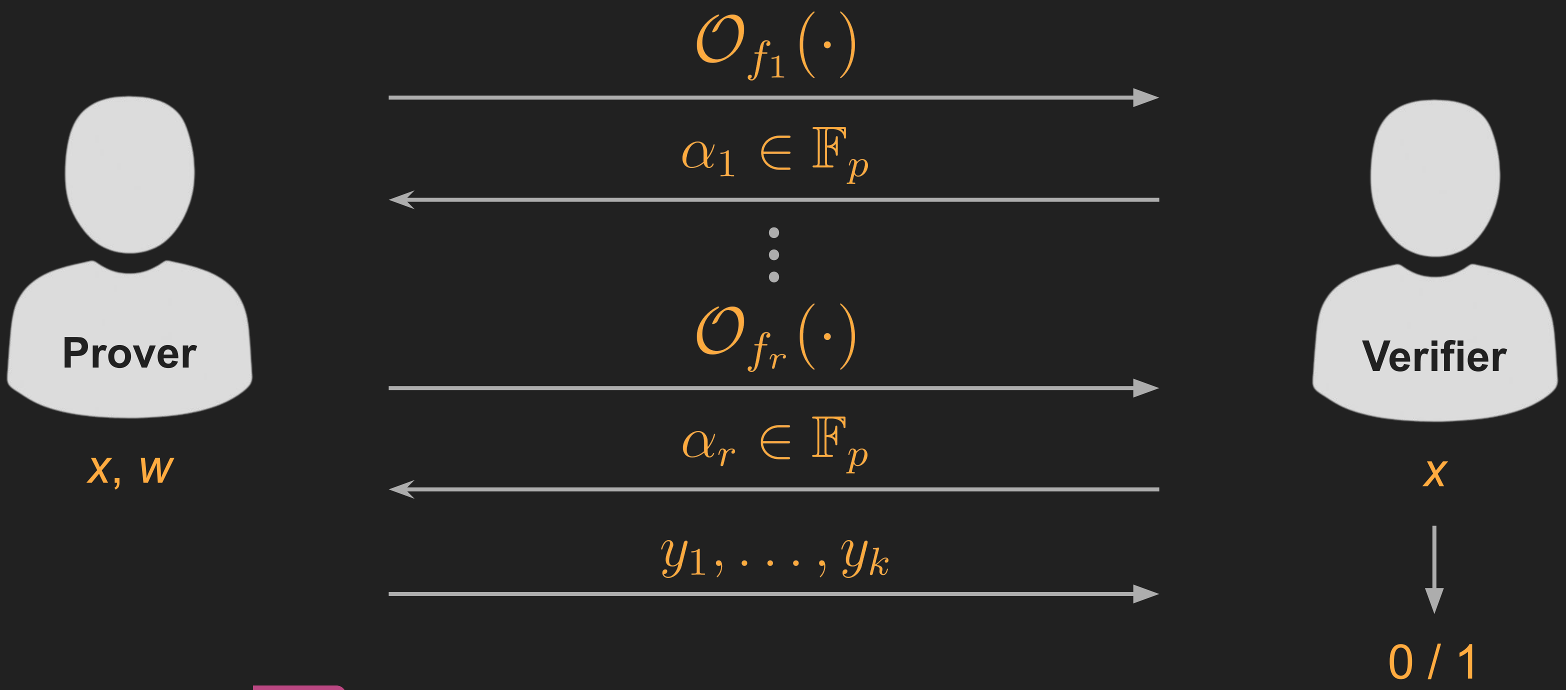
- In particular: for a finite field $\mathbb{F}_p$, the probability is at most $\deg f / p$

PIOPs – Compilation to a SNARK



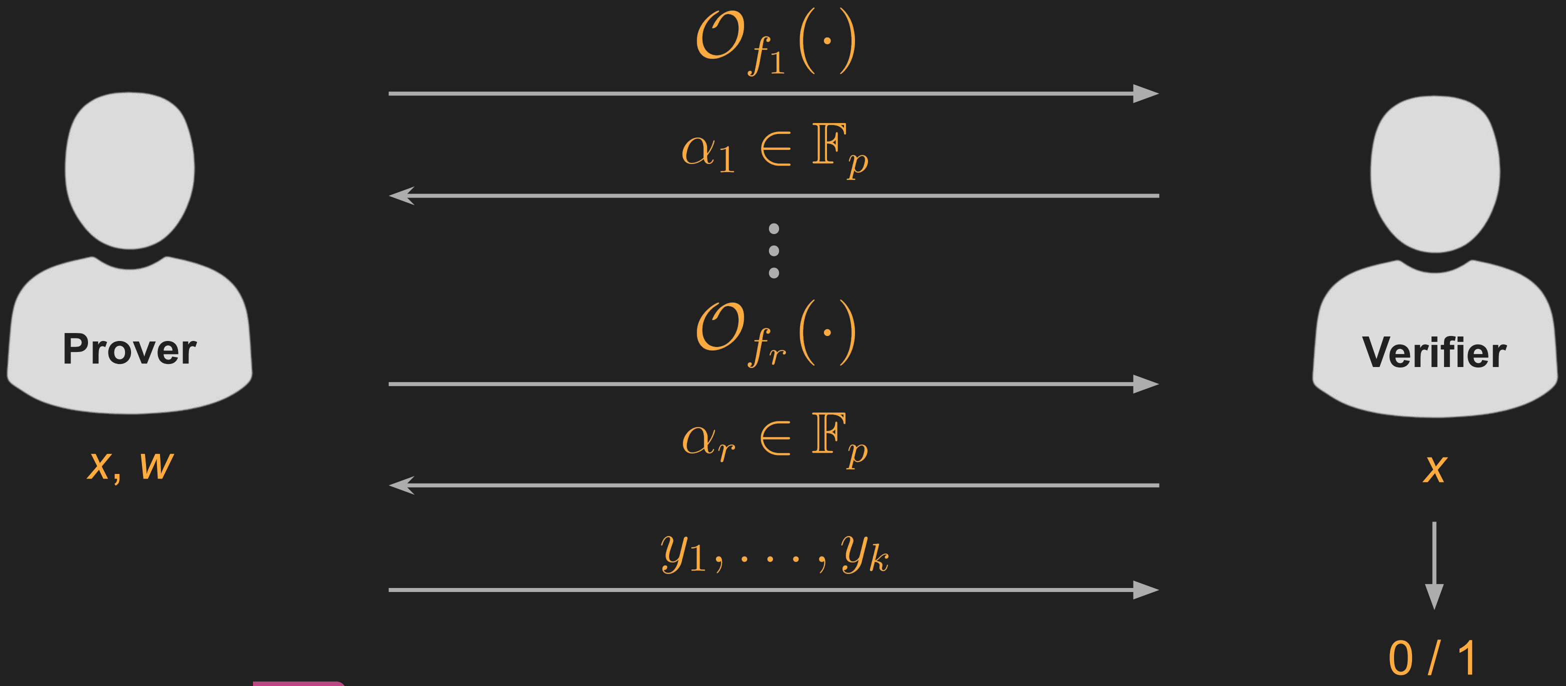
TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PIOPs – Compilation to a SNARK



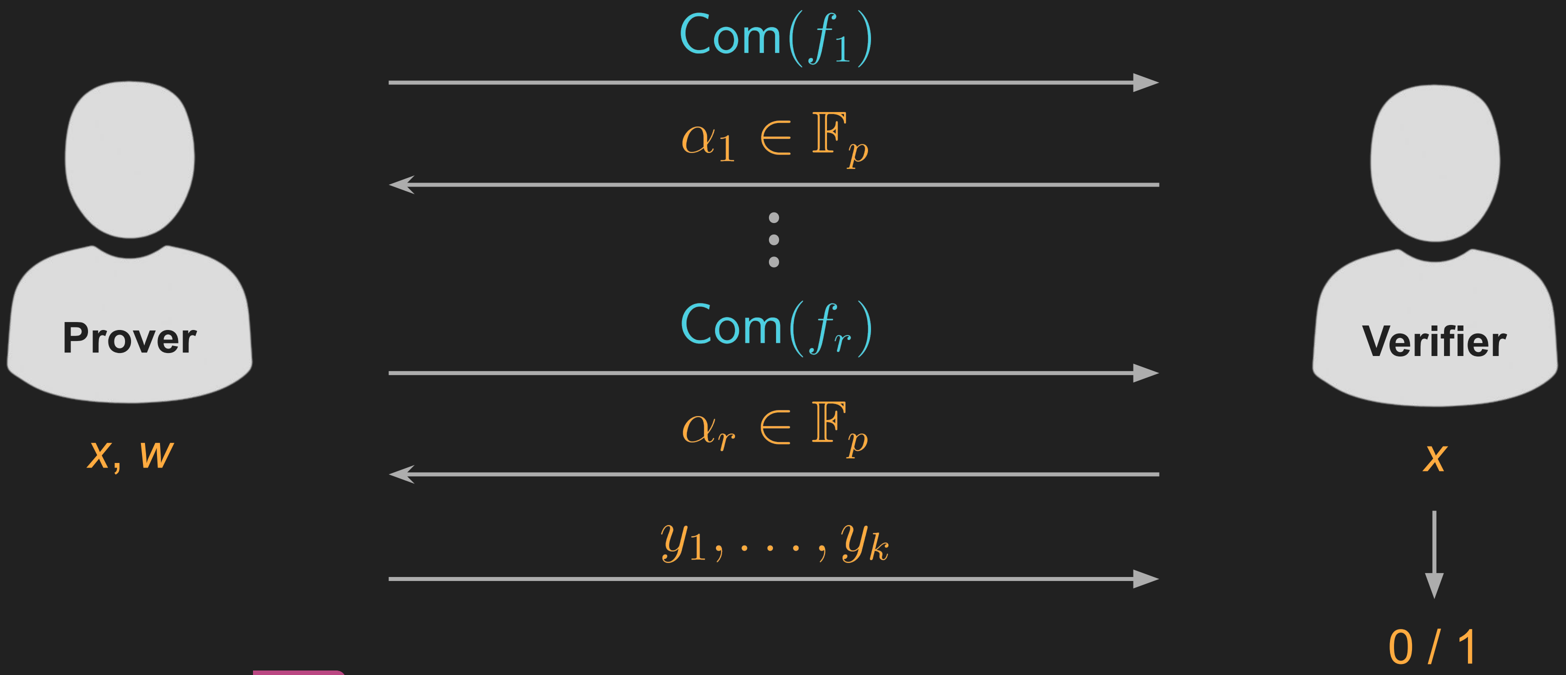
PIOPs – Compilation to a SNARK

- Step 1: Polynomial Commitment Scheme



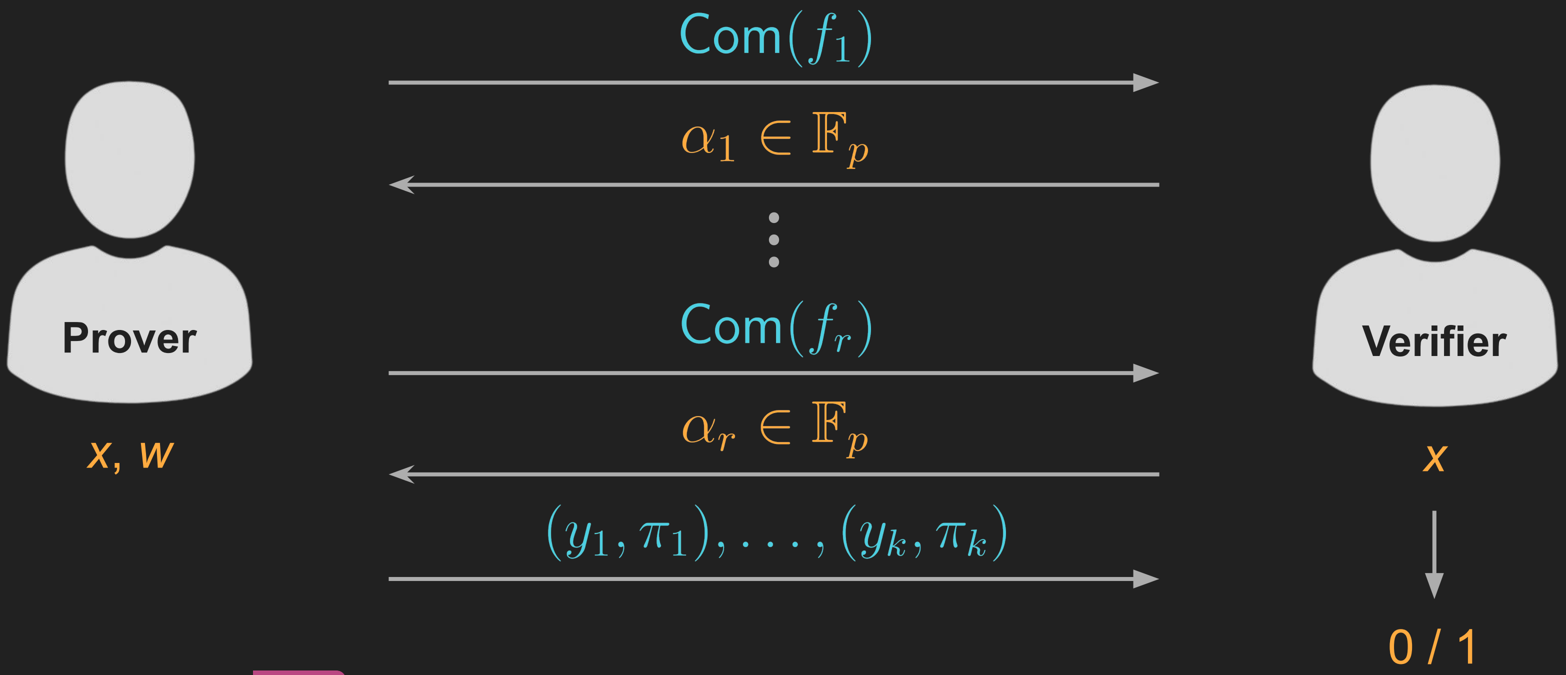
PIOPs – Compilation to a SNARK

- Step 1: Polynomial Commitment Scheme



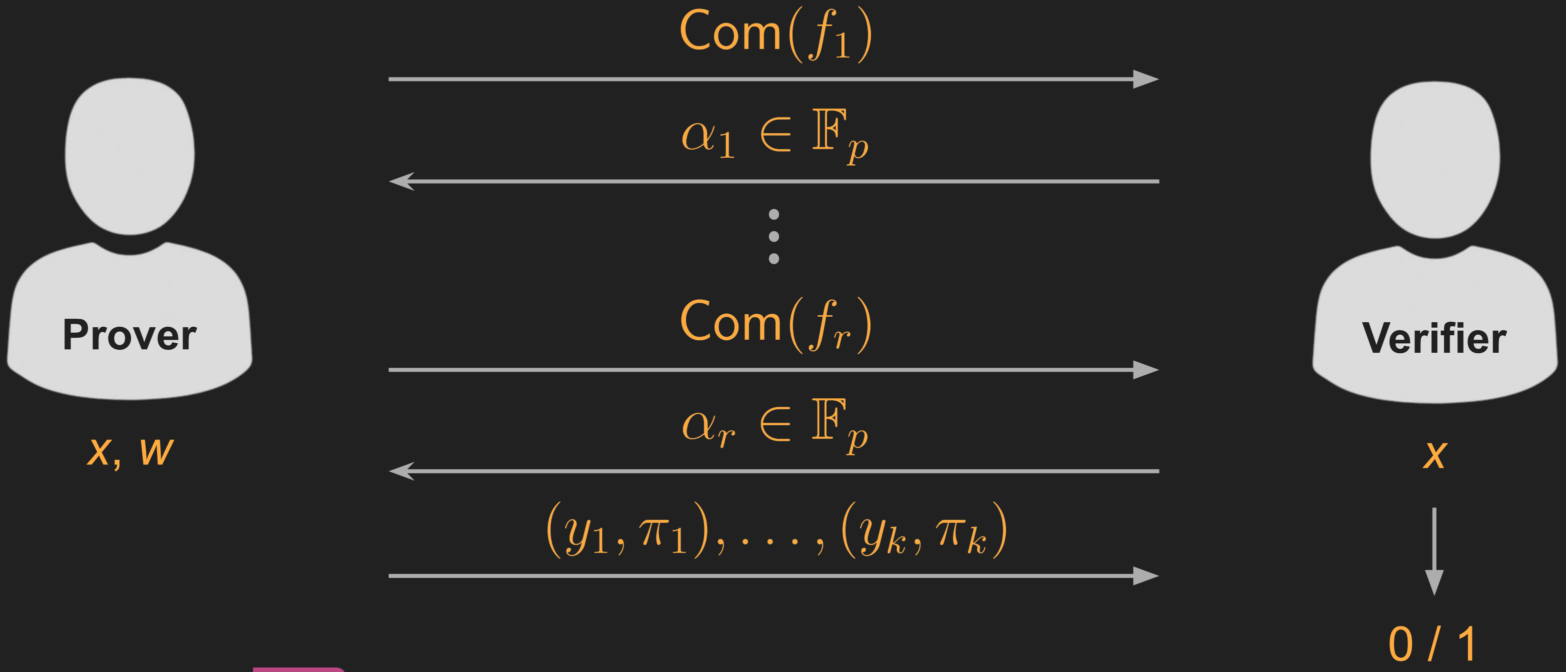
PIOPs – Compilation to a SNARK

- Step 1: Polynomial Commitment Scheme



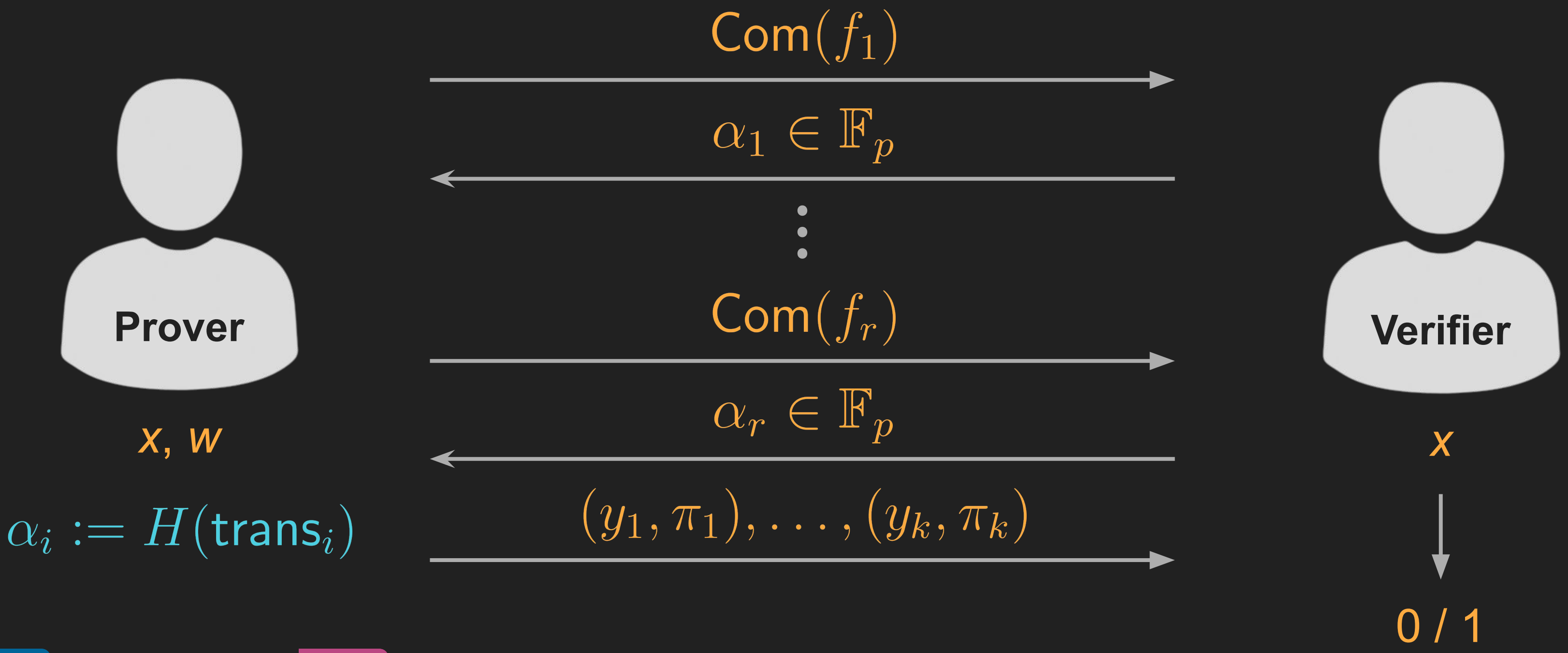
PIOPs – Compilation to a SNARK

- Step 2: Fiat-Shamir Transformation



PIOPs – Compilation to a SNARK

- Step 2: Fiat-Shamir Transformation



PIOPs – Compilation to a SNARK

- Step 2: Fiat-Shamir Transformation

Prover

x, w

$\alpha_i := H(\text{trans}_i)$

$\text{Com}(f_1)$

$\alpha_1 := H(\text{Com}(f_1))$

$\vdots$

$\text{Com}(f_r)$

$\alpha_r := H(\text{Com}(f_1), \ldots, \text{Com}(f_r))$

$(y_1, \pi_1), \ldots, (y_k, \pi_k)$

Verifier

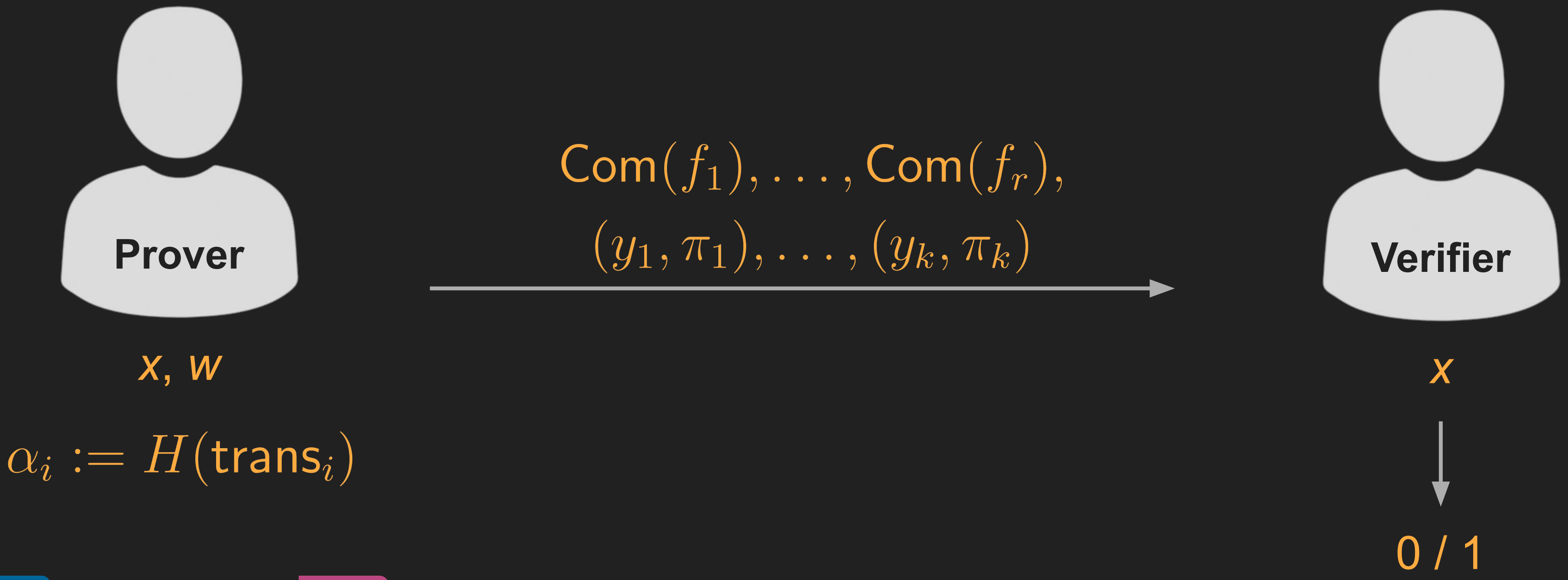
x

0 / 1

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

PIOPs – Compilation to a SNARK

- Step 2: Fiat-Shamir Transformation



Properties Needed from H

- Note that P and V both need to evaluate H
- Inputs $\text{trans}_1, \ldots, \text{trans}_r$ to H prefixes of each other ⟹ stateful H more efficient
- Usually modelled as a random oracle in security proofs, i.e., H needs to be a strong cryptographic hash function
- Another approach:
  - Correlation-intractable hash functions
  - For some fixed (sparse) relation R, it should be hard to find x s.t. $(x, \mathrm{H}(x)) \in R$
  - For example: $R = \{ (f, z) \mid f$ low-degree non-zero polynomial, $f(z) = 0 \}$

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Open Questions

- Prove security of Fiat-Shamir transformation without ROM
- Find necessary properties of H for this (CR, PR not enough!)
- Construct correlation-intractable hash functions:
    - Theoretical construction (feasibility result)
    - Practical construction?

Thanks!

Questions?

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

Appendix

Uniformity Property

- Let Samp be an efficient sampling algorithm with $|\mathsf{Samp}(1^\lambda)| = \lambda^{\omega(1)}$
- We want $\Pr[H(x) = y] = \mathsf{negl}(\lambda)$ for all y and $x \leftarrow \mathsf{Samp}(1^\lambda)$
- Can be used to prove ZK of PLONK (without ROM)
- Implied by collision resistance, but is a weaker information-theoretic property
- Do all (cryptographic) hash functions have this property?

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 $f(x) = y$
- CRS: $\left(g_1, g_1^{\tau}, g_1^{\tau^2}, \ldots, g_1^{\tau^d}, g_2, g_2^{\tau}\right)$ 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

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

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy

References

[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.

TU WIEN CYSEC CYBERSECURITYCENTER S&P Security & Privacy