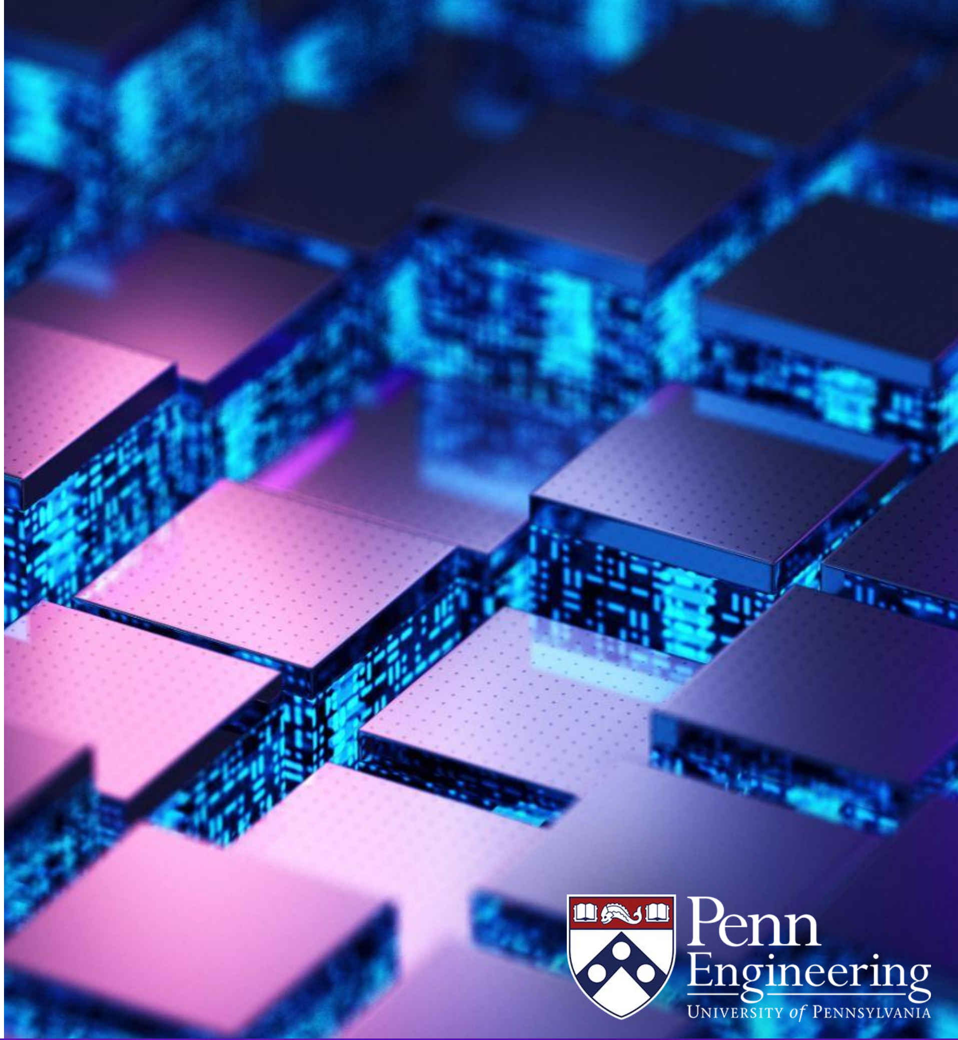
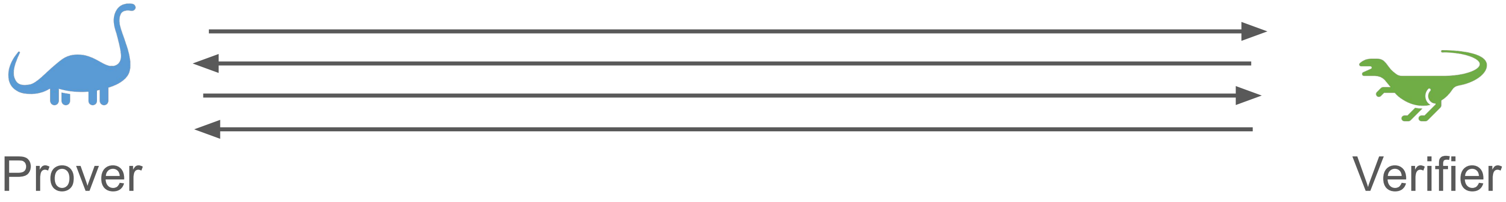
EAS 5830: BLOCKCHAINS

Zero-Knowledge Proofs

Professor Brett Hemenway Falk




Prover
Verifier

Provable “statements”

- $x \in L$ for some NP language L
- $L = \{ x \mid \exists\ w$ such that $(x,w) \in R \}$
  - w is the ‘witness’
  - R is a polynomial-time computable ‘relation’

Example: Hash Preimage

- y is a valid hash
    - $L = \{ y \mid \exists\ x \text{ such that } \mathrm{Hash}(x) = y \}$
    - witness = x

Example: Equality of Plaintexts

- C_1 and C_2 encrypt the same plaintext (under key pk)
    - $L = \{ (C_1,C_2) \mid \exists\ r_1, r_2, m \text{ such that } \mathrm{Enc}_{pk}(m;r_1) = C_1 \text{ and } \mathrm{Enc}_{pk}(m;r_2) = C_2 \}$
    - witness = (r_1,r_2, m)

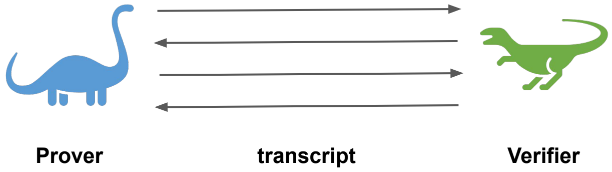
Example: Correctness of a shuffle

- Correctness of a shuffle
  - ciphertexts $C_1,\ldots,C_n$ and $C_1',\ldots,C_n'$ encrypt the same values (in permuted order)
  - witness is plaintexts, encryption randomness and permutation

Goals

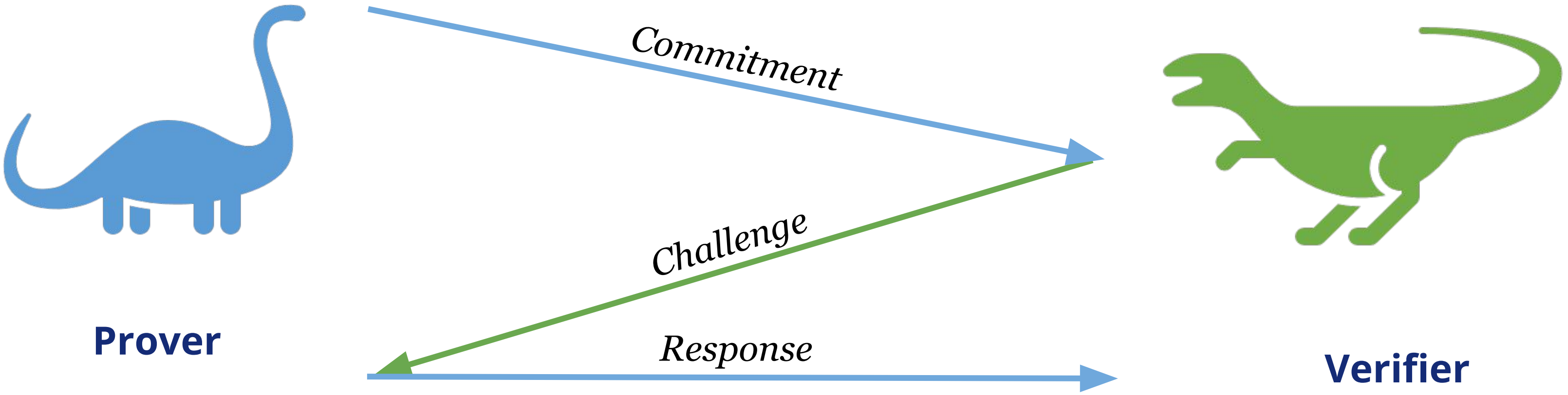
- **Completeness**
  - Prover can convince a verifier of any true statement
- **Soundness**
  - Prover cannot convince a verifier of a false statement
- **Zero-Knowledge**
  - Protocol reveals nothing more than the truth of the statement

Defining Zero Knowledge (interactive)

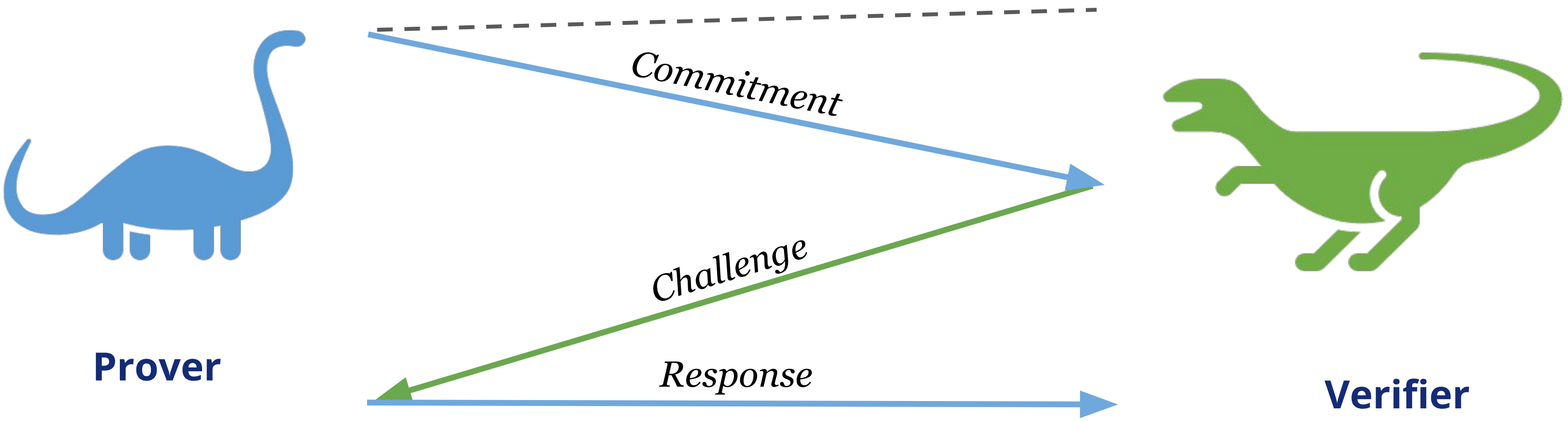


- There exists a simulator, S
- S can generate protocol transcripts
  - S does not know a witness
  - S gets to play both sides (prover and verifier)
- Distribution of transcripts produced by S is *indistinguishable* from the distribution of transcripts produced by an honest prover and verifier
- Since S can simulate transcripts without knowing a witness, the transcript cannot reveal any information about the witness
- Guarantees non-transferability:
  - A verifier cannot convince another party by showing only the transcript

Σ-protocols



Σ-protocols



Looks like a Σ?

Σ-protocols

3 Round Protocol:

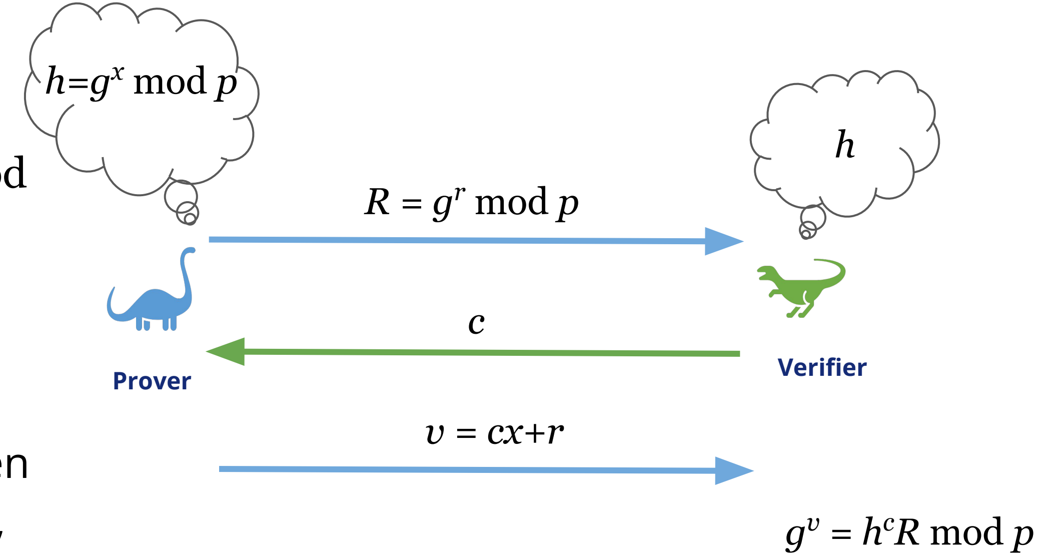
1. (P→V) commitment
2. (V→P) challenge
3. (P→V) response

Properties:

1. **Completeness:** If prover inputs x and $(x,w) \in R$, then verifier accepts
2. **Extractability ("special soundness"):** Given two transcripts (a,c,r), (a,c',r') there is an extractor that can recover w
3. **Honest-verifier Zero Knowledge:** There is a simulator S, such that $S(x,c)$ outputs transcripts (a,c,r). If c is chosen uniformly, then the simulator's distribution is indistinguishable from the real distribution

Schnorr's proof is a Σ-protocol

- **Completeness:** $g^v = g^{cx+r} = (g^x)^c g^r = h^c R \bmod p$
- **Extractability:** given (g^r, c_1, v_1) and (g^r, c_2, v_2) then $x = (v_2 - v_1)/(c_2 - c_1)$
- **Honest-Verifier ZK:** given (h,c) choose v at random, and set $R = g^v h^{-c} \bmod p$



Making a Σ-protocol non-interactive

- The Fiat-Shamir heuristic
  - Replace challenge with Random Oracle applied to commitment
  - In practice, use a hash function
- Non-interactive protocol
  - Prover generates a “commitment” a
  - Prover calculates $c = \mathrm{hash}(a)$
  - Prover calculates response r
  - Prover sends proof = (a,c,r)
- Better to include statement in hash as well
  - $c = \mathrm{hash}(x,a,aux)$

Non-interactive proofs

- All NIZKs require at least one of
  - Common Random String (CRS)
  - Common Reference String (CRS)
  - Random Oracle (RO)
- Zero-knowledge simulator
  - Gets to generate CRS or program RO

Proving general statements

Given a function, f, prove: "I know an input, x, such that $f(x) = o$"

Arguments vs Proofs

- **Arguments**
  - Computational soundness
  - Unbounded prover can prove false statements
- **Proofs**
  - Information-theoretic soundness

ZK-SNARKs

Zero

Knowledge

Succinct

Non-interactive

ARgument of

Knowledge

Implemented in libSNARK

- Requires Common Reference String (trusted setup)
- Can prove arbitrary statements
  - Universal CRS
  - Different CRS for each statement
- Very small proof size