Making Σ-Protocols Non-Interactive without Random Oracles

Pyrros Chaidos, Jens Groth
University College London

Engineering and Physical Sciences Research Council

Overview

- Zero knowledge proofs are an important tool, often made non-interactive using Fiat-Shamir transformation.
- Damgård-Fazio-Nicolosi (DFN) transformation: alternative to Fiat-Shamir for a class of Σ-protocols. Requires complexity leveraging assumption.

- We revisit the transformation, using culpable soundness to model the adversary.
- We give a protocol proving that ciphertexts contain 0/1, and a voting application.

Outline

Definitions

Culpable Soundness for DFN

Applications

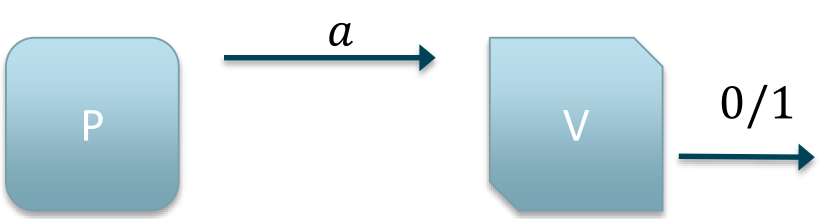
Σ-Protocols

- 3-move protocols for some NP relation R.
- Prover demonstrates a statement $x \in L_R$: $(x, w) \in R$, for some witness w.

- Completeness: V outputs 1 for $x \in L_R$.
- ***Relaxed*** Special Soundness: If $x \notin L_R$, at most one value of *e* can lead to Verifier outputting 1.
- Special Honest Verifier Zero Knowledge: transcripts between P and honest V can be efficiently simulated. Special: simulator targets a challenge e.

Σ-Protocols

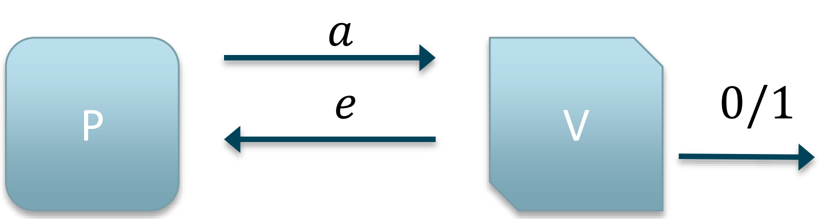
- 3-move protocols for some NP relation R.
- Prover demonstrates a statement $x \in L_R$: $(x, w) \in R$, for some witness w.



- Completeness: V outputs 1 for $x \in L_R$.
- ***Relaxed*** Special Soundness: If $x \notin L_R$, at most one value of *e* can lead to Verifier outputting 1.
- Special Honest Verifier Zero Knowledge: transcripts between P and honest V can be efficiently simulated. Special: simulator targets a challenge e.

Σ-Protocols

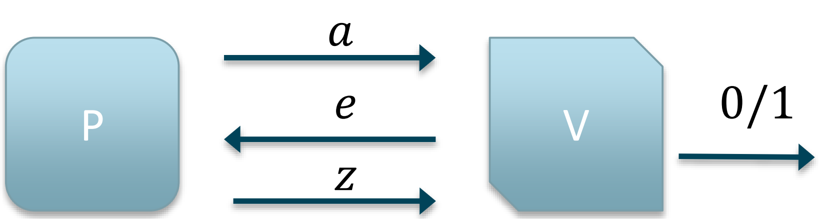
- 3-move protocols for some NP relation R.
- Prover demonstrates a statement $x \in L_R: (x, w) \in R$, for some witness w.



- Completeness: V outputs 1 for $x \in L_R$.
- ***Relaxed*** Special Soundness: If $x \notin L_R$, at most one value of *e* can lead to Verifier outputting 1.
- Special Honest Verifier Zero Knowledge: transcripts between P and honest V can be efficiently simulated. Special: simulator targets a challenge e.

Σ-Protocols

- 3-move protocols for some NP relation R.
- Prover demonstrates a statement $x \in L_R$: $(x, w) \in R$, for some witness w.



- Completeness: V outputs 1 for $x \in L_R$.
- ***Relaxed*** Special Soundness: If $x \notin L_R$, at most one value of *e* can lead to Verifier outputting 1.
- Special Honest Verifier Zero Knowledge: transcripts between P and honest V can be efficiently simulated. Special: simulator targets a challenge e.

Homomorphic Encryption

- Additively Homomorphic:
  - $E_{pk}(m_1; r_1) \cdot E_{pk}(m_2; r_2) = E_{pk}(m_1 + m_2; r_1 + r_2)$
- Strongly Additively Homomorphic:
  - Decryption Homomorphic and efficiently verifiable ciphertext space: any c either fails verification or decrypts and respects homomorphic property.
  - Extended Randomness: randomness can be any $r \in \mathbb{Z}$.
  - Prime order message space.
  - Verifiable Keys (efficient to check if (pk, vk) are a keypair).
- IND-CPA Security

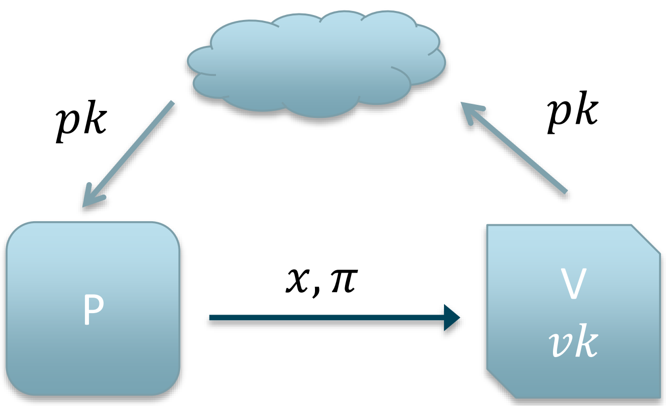
Culpable Soundness

- Standard soundness: hard for adversary to prove **any** false statements.
- Culpable soundness: hard for adversary to prove **some** false statements, and be **aware** of the falsehood.
- Guilt relation R_g consists of (x, w_g) such that $x \notin L_R$.
- Culpable Soundness for a guilt relation R_g:
  no efficient adversary can produce x, π, w_g
  s.t. $(x, w_g) \in R_g$ and $Ver(vk, x, \pi)$ accepts.

Soundness with Unique Identifiable Challenge

- Relaxed Special Soundness: for fixed a, adversary can only prove false statement x for **one** value of e.
- Unique Identifiable Challenge: for **some** false statements, adversary must also be **aware** of the e value in successful proofs.
- Unique Identifiable Challenge for a guilt relation R_g: Given w_g and x, a: $(x, w_g) \in R_g$ and $Ver(x, a, e, z) = 1$ for some e, z we can extract the unique "good" e.

Designated Verifier NIZK



- Verifier has (pk, vk) keypair.
  - Public key pk used to generate proofs. The choice of pk designates who can verify the proof.
  - Verification key vk used to verify.

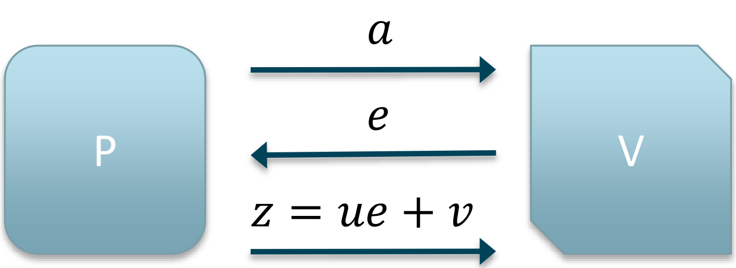
Outline

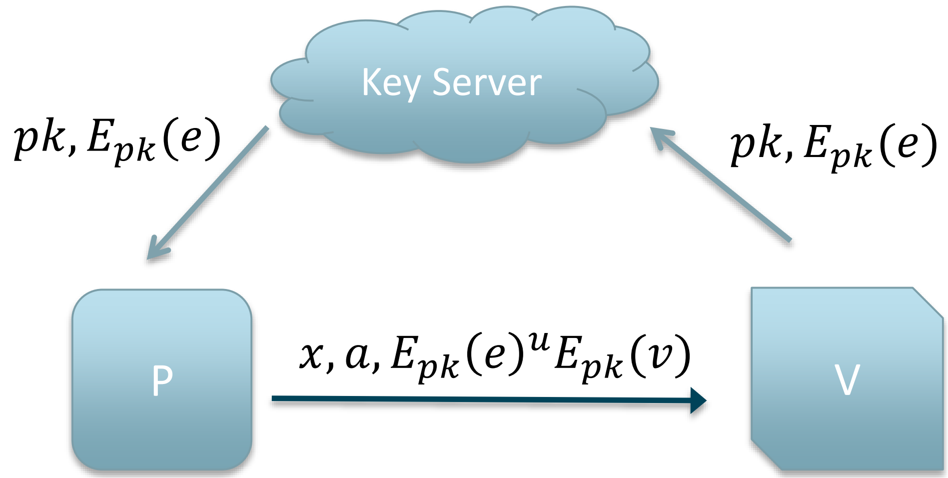
Definitions

Culpable Soundness for DFN

Applications

The DFN Transformation





- For ZK, simulator obtains vk in registration step, decrypts $E_{pk}(e)$, calls the original SHVZK simulator and encrypts answer.

Using UIC-soundness in DFN

- Soundness with Unique Identifiable Challenge (UIC) provides us with a challenge extractor using w_g as a "hint".
- No need for complexity leveraging: UIC extractor runs in polynomial time.

Theorem 2: Applying DNF transformation to a UIC-sound Σ-protocol with linear answer over the integers, produces a DV NIZK with culpable soundness for the same guilt relation.

Culpable soundness follows from IND-CPA and UIC

- From an accepting proof of a false statement and a guilt witness we can extract the unique challenge e in c.
- We can easily adapt a cheating prover to an IND-CPA adversary:
- Obtain challenge ciphertext from IND-CPA game, use as encrypted challenge. If adversary succeeds in forging, we succeed in decrypting challenge.

Outline

Definitions

Culpable Soundness for DFN

Applications

UIC-sound Σ-protocol for ciphertext containing 0 or 1

- Argument that a ciphertext c contains 0 or 1, for a Strongly Additively Homomorphic encryption scheme (e.g Okamoto-Uchiyama).

$$R = \left\{ ((ek, c), (m, r)) : c = \mathcal{E}_{ek}(m; r) \text{ and } m \in \{0,1\} \text{ and } r \in \{0,1\}^{\ell_r(n)} \right\}$$

$$R_g = \left\{ ((ek, c), dk) : c \in \mathcal{C}_{ek} \text{ and } \mathcal{D}_{dk}(c) \notin \{0,1\} \text{ and } \mathrm{VerifyKey}(ek, dk) = 1 \right\}$$

- Applications:
  - Encrypted wires satisfying a circuit:
    $c = (a \text{ NAND } b) \Leftrightarrow a + b + 2c \in \{0,1\}$
  - Vote Encoding
  - More complex variants possible ($c \approx 0$, $c_1 \approx c_2$, etc.)

Proving UIC Soundness

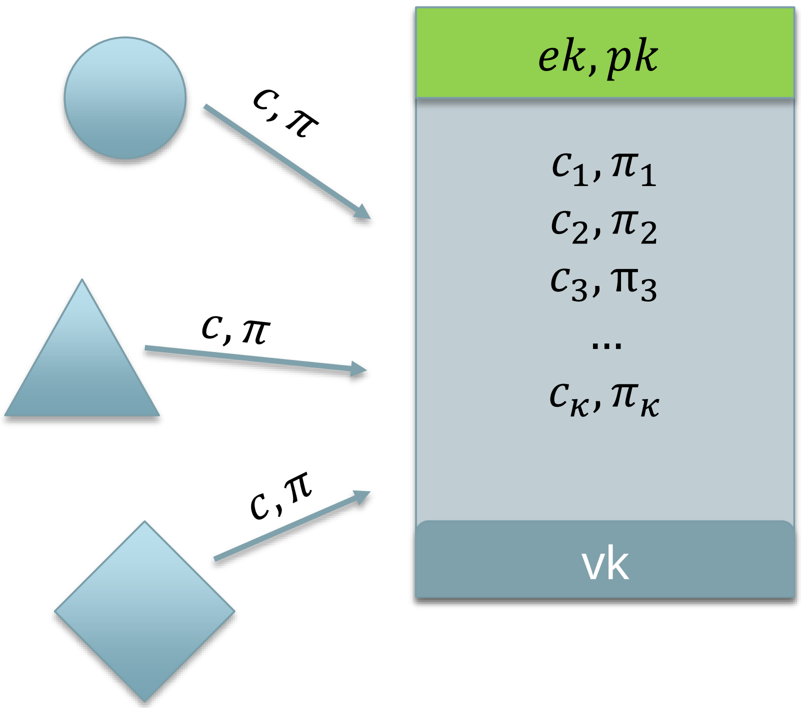
Prover$((ek, c), m)$		Verifier(ek, c)
$m_a \leftarrow [2^n, 2^{n+1}]$ $a \leftarrow \mathcal{E}_{ek}(m_a)$ $b \leftarrow \mathcal{E}_{ek}(-mm_a)$	$\xrightarrow{a, b}$	
	$\xleftarrow{e}$	Accept if: $a, b, c \in \mathcal{C}_{ek}$
$f := em + m_a$	$\xrightarrow{f}$	$c^e a = \mathcal{E}_{ek}(f)$ $c^{f-e} b = \mathcal{E}_{ek}(0)$

- We use the guilt witness (dk) to decrypt a, b, c, obtaining values m_a, m_b, m.
- Combining the verification equations, we have: $e(m-1)m + m_a m + m_b = 0 \bmod p$.
- Since $m \notin \{0,1\}$ this determines e uniquely mod p.

Using Culpable Soundness

- Need broad enough L_g, otherwise, we may allow a large class of invalid statements to be accepted.
  - We will achieve this by requiring the decryption is not $0/1$, and relying on strongly additively homomorphic property.
- Need w_g to be available somehow.
  - Depending on the setting, it is possible that the environment has the decryption key. If an adversary succeeds in forging a proof, we can “plant” the key on him to satisfy Culpable Soundness.

Voting Application



Voting Application

ek, pk

~~c_1, π_1~~

c_2, π_2

~~c_3, π_3~~

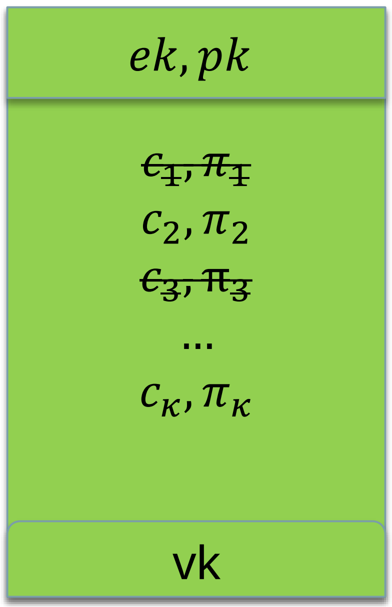
…

c_κ, π_κ

vk

$$r = D_{dk}\left(\prod_{Ver(c,\pi,vk)=1} c\right)$$

Voting Application



$$r = D_{dk}\Big(\prod_{Ver(c,\pi,vk)=1} c\Big)$$

- We prove correctness and ballot privacy. Adversary can use standard functionality and also submit arbitrary ballots.
- Correctness:
  - Adversary cannot force result to be out of bounds.
  - Follows from CS: ballots that do not contain $0/1$ contradict soundness
- Ballot Privacy
  - Adversary cannot distinguish between normal run, and run with all honest $0/1$ ballots swapped to honest 0 ballots but tallied normally.

Voting Privacy

- We use a series of hybrid arguments to argue that the adversary can distinguish between games that differ in a single ciphertext.
- We want to reduce the difference to IND-CPA, but we must provide the (correct) tally before the adversary can guess.
- Workaround: suspend adversary, guess tally r, resume. Feasible to try all values because of referendum.
- Also need to know which guess was true (best).
  Before playing out all cases we can test using known ciphertexts to determine optimal r value.

Conclusion

- The DFN transformation can produce Designated Verifier NIZKs from a wide range of Σ-protocols, without Random Oracles.
- We show how to avoid complexity leveraging using culpable soundness and restricting to UIC-sound protocols.
- We demonstrate that this restricted class of Σ-protocols is useful for settings where culpable soundness is achievable e.g. voting applications.

Thanks!

Questions?