Lift-and-Shift: Obtaining Simulation Extractable Subversion and Updatable SNARKs Generically

Behzad Abdolmaleki[1] and **Sebastian Ramacher**[2] and Daniel Slamanig[2]

Young Researcher Crypto Seminar, 16.10.2020

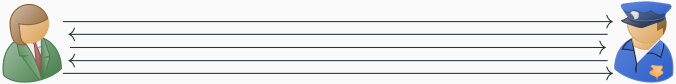
[1]University of Tartu, [2]**AIT Austrian Institute of Technology, Vienna**

Introduction

Zero-knowledge Proofs

NP-language L

- Prover wants to convince verifier that some $\mathtt{x} \in L$
- Without revealing information beyond the statement $\mathtt{x} \in L$
- Define relation R_L: $\mathtt{x} \in L \Leftrightarrow \exists \mathtt{w} : (\mathtt{x}, \mathtt{w}) \in R_L$



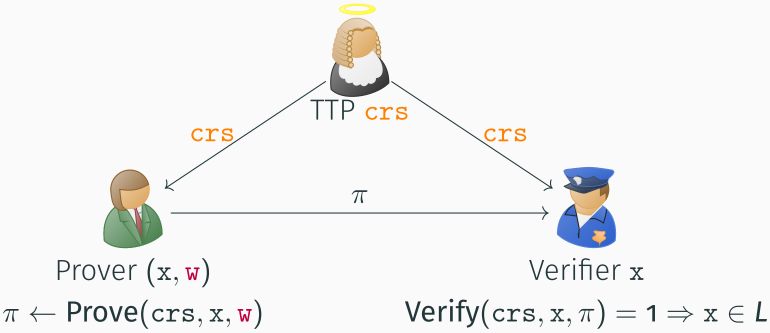
Prover $(\mathtt{x}, \mathtt{w})$

Verifier x

$\mathtt{x} \in L$ ✓

Making them Non-Interactive: CRS

Common reference string model



Important Properties

Prover cannot cheat

- Prover unable to produce valid proofs for $\mathbf{x} \notin L$
- Soundness
- Property desired by the verifier

Important Properties

Prover cannot cheat

- Prover unable to produce valid proofs for $\mathtt{x} \notin L$
- › Soundness
- Property desired by the verifier

Verifier does not learn any information on witness $\mathtt{w}$

- Real proofs cannot be distinguished from simulated proofs
- › Zero-knowledge
- Property desired by the prover

Important Properties

Proofs of Knowledge

- Special extractor can extract witness from proofs
- Knowledge Soundness

Important Properties

Proofs of Knowledge

- Special extractor can extract witness from proofs
- › Knowledge Soundness

Strong versions

- (Knowledge) Soundness also holds if adversary can query simulated proofs
- › Simulation (knowledge) soundness
- Also called simulation (sound) extractability (SE)

On Simulation Soundness

In a real world protocol:

- Adversary sees many different proofs
- Might be possible to turn proof π for word $\mathbf{x}$ into a proof $\pi' \neq \pi$
- Or worse: turn into a proof π' for a different word $\mathbf{x}' \neq \mathbf{x}$

On Simulation Soundness

In a real world protocol:

- Adversary sees many different proofs
- Might be possible to turn proof π for word $\mathbf{x}$ into a proof $\pi' \neq \pi$
- Or worse: turn into a proof π' for a different word $\mathbf{x}' \neq \mathbf{x}$

Hence

- Adversary may query proofs
- Must produce a proof not queried before

Similar issue for signatures: one-time EUF-CMA – EUF-CMA – strong EUF-CMA

NIZKs in the CRS Model

- Zero-knowledge contradicts extractor
- Knowledge soundness contradicts simulator

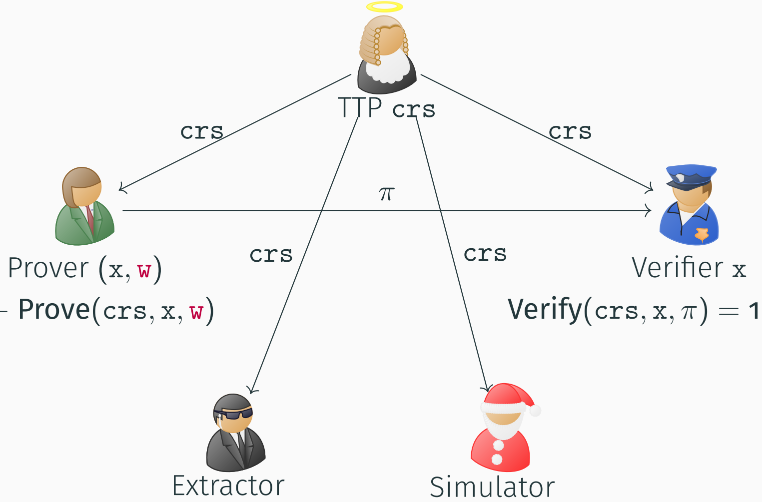
NIZKs in the CRS Model

- Zero-knowledge contradicts extractor
- Knowledge soundness contradicts simulator

They need to have more power

- Extractor gets extraction trapdoor
- Simulator gets simulation trapdoor

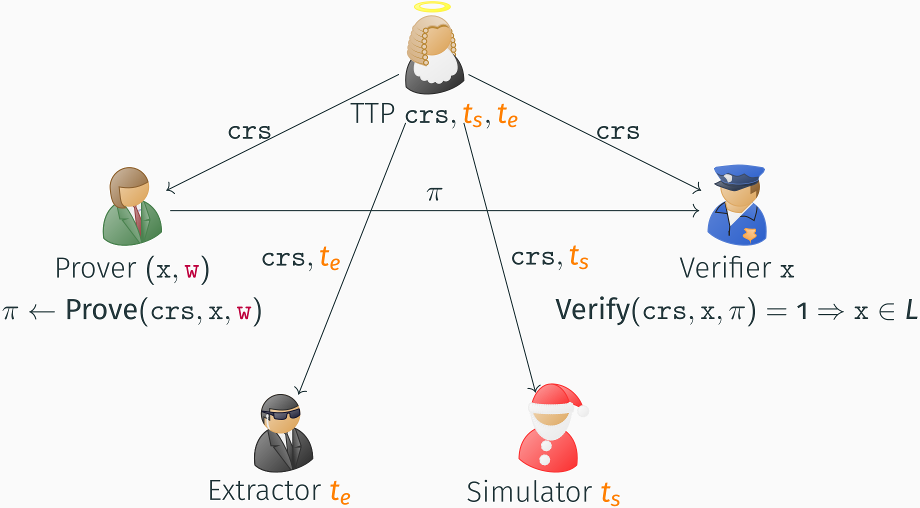
NIZKs in the CRS Model



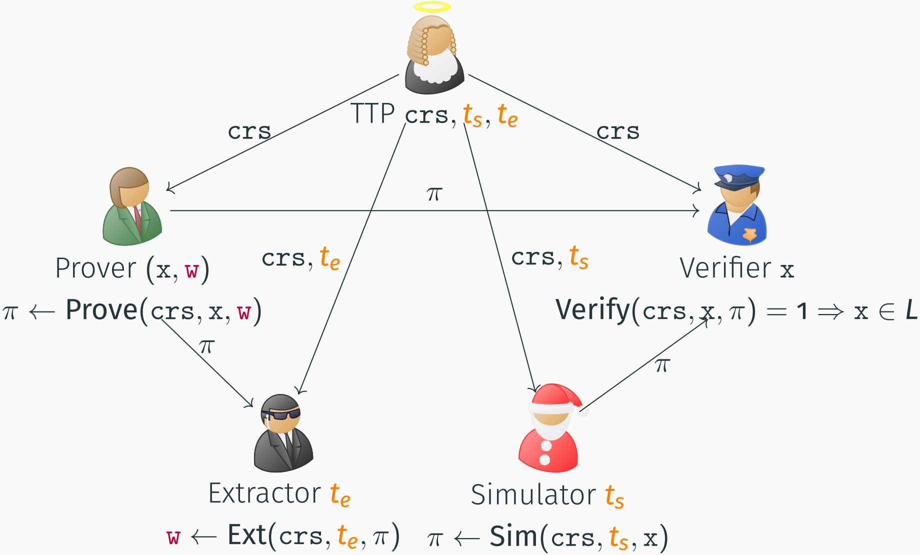
$\pi \leftarrow \mathsf{Prove}(\mathtt{crs}, \mathtt{x}, \mathtt{w})$

$\mathsf{Verify}(\mathtt{crs}, \mathtt{x}, \pi) = 1 \Rightarrow \mathtt{x} \in L$

NIZKs in the CRS Model



NIZKs in the CRS Model



Achieving Simulation Extractability

The C∅C∅ Framework [KZM+15]

Extend statement to

$$\mathsf{c} = \Omega.\mathsf{Enc}(\mathsf{pk}_\Omega, \mathtt{w}; r_0) \wedge ((\mathtt{x}, \mathtt{w}) \in R_L \vee (\mu = f_s(\mathsf{pk}_{\Sigma^1}) \ \wedge \ \rho = \mathsf{Commit}(s; r_1)))$$

and sign $(\mathtt{x}, \mathsf{c}, \mu, \pi_{L'})$ with sk_{Σ^1}

$\mathtt{crs}$ extended with ρ, pk_Ω; s, r_0 simulation trapdoor, sk_Ω extraction trapdoor

- Ω: public-key encryption
- Σ^1: strong one-time signature
- f: PRF
- Commit: Commitment

The C∅C∅ Framework [KZM+15]

Extend statement to

$$\mathsf{c} = \Omega.\mathsf{Enc}(\mathsf{pk}_\Omega, \mathtt{w}; r_0) \wedge ((\mathtt{x}, \mathtt{w}) \in R_L \vee (\mu = f_s(\mathsf{pk}_{\Sigma^1}) \ \wedge \ \rho = \mathsf{Commit}(s; r_1)))$$

and sign $(\mathtt{x}, \mathsf{c}, \mu, \pi_{L'})$ with sk_{Σ^1}

$\mathtt{crs}$ extended with ρ, pk_Ω; s, r_0 simulation trapdoor, sk_Ω extraction trapdoor

- Ω: public-key encryption
- Σ^1: strong one-time signature
- f: PRF
- **Commit**: Commitment using SHA256
  Proving pre-image of a random oracle

The OC∅C∅ Framework [ARS20]

Fixed-value key-binding PRF [CMR98; Fis99]

- For a PRF f with key s and special value β, hard to find s' with $f_s(\beta) = f_{s'}(\beta)$

The OC∅C∅ Framework [ARS20]

Fixed-value key-binding PRF [CMR98; Fis99]

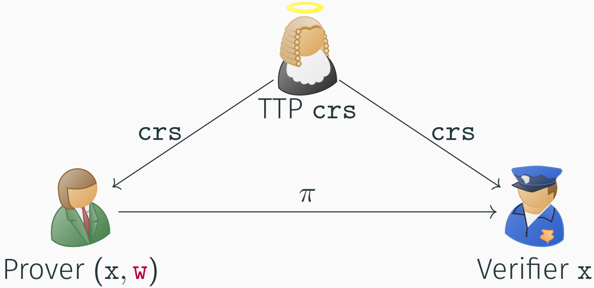
- For a PRF f with key s and special value β, hard to find s' with $f_s(\beta) = f_{s'}(\beta)$

Change statement to

$$(\mathtt{x}, \mathtt{w}) \in R_L \vee (\mu = f_s(\mathsf{pk}_{\Sigma 1}) \wedge \rho = f_s(\beta))$$

Allows instantiation with low complexity primitives

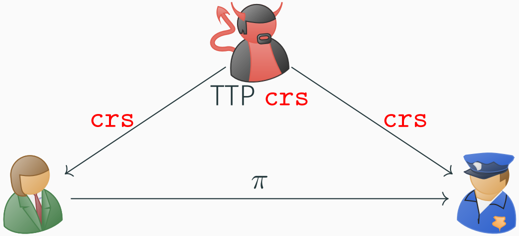
Subversion and Updatability



$\pi \leftarrow \mathsf{Prove}(\mathtt{crs}, \mathtt{x}, \mathtt{w})$

$\mathsf{Verify}(\mathtt{crs}, \mathtt{x}, \pi) = 1 \Rightarrow \mathtt{x} \in L$

CRS Generator



Prover $(\mathtt{x}, \mathtt{w})$

$\pi \leftarrow \textsf{Prove}(\mathtt{crs}, \mathtt{x}, \mathtt{w})$

Verifier $\mathtt{x}$

$\textsf{Verify}(\mathtt{crs}, \mathtt{x}, \pi) = 1 \Rightarrow \mathtt{x} \in L$

What if the CRS generator is malicious?

No guarantee that

- CRS is correct
- CRS from the correct distribution
- Trapdoors exist

What if the CRS generator is malicious?

No guarantee that

- CRS is correct
- CRS from the correct distribution
- Trapdoors exist

Perform CRS generation with MPC protocol

- Examples: zcash ceremony
- But in practice complicated, expensive and requires much effort beside techical realization

Subversion Resistance [BFS16]

- Subversion soundness: sound even if CRS subverted
- Subversion zero-knowledge: zero-knowledge even if CRS subverted
- Some combinations impossible

	WI	Zero-Knowledge	Subversion ZK
Soundness	✓	✓	✓
Subversion soundness	✓	✗	✗

Updatable NIZK [GKM+18]

- Assume adversary has complete (or partial) control over $\mathtt{crs}$ generation
- Add **Ucrs** algorithm: outputs a new CRS and proof of update
- Also add **Vcrs**: verifies CRS, updates and proofs

Idea: either $\mathtt{crs}$ was generated honestly or one update was done honestly

- Verifier updates CRS to ensure soundness
- Prover updates CRS to ensure zero-knowledge

Towards LAMASSU: Key-homomorphic Signatures / Updatable Signatures

Key-homomorphic signatures:

- Homomoprhism between private-key and public-key spaces: $\mu: S \to P$
  Natural in the DLOG setting: $x \mapsto g^x$
- Signatures can be adapted from pk to $\mathsf{pk}' = \mathsf{pk} \cdot \mu(\mathsf{sk}' - \mathsf{sk})$ if $\mathsf{sk}' - \mathsf{sk}$ known
- Examples: Schnorr, BLS, and many more

Updatable signatures:

- Upk: update pk and provide proof of update
- Vpk: verify update
- Idea: either original pk created honestly or update was done honestly
- Example: Schnorr in bilinear groups with BDH knowledge assumption

Towards LAMASSU: Simulation Soundness using Key-Homomorphic Signatures

Compiler [DS19]: "$\mathtt{x} \in L$ or I can sign with a public key in the CRS"

- Extend statement to

$$(\mathtt{x}, \mathtt{w}) \in R_L \vee \mathsf{pk}' = \mathsf{pk} \cdot \mu(\mathsf{sk}' - \mathsf{sk})$$

  - Generate key pairs $(\mathsf{sk}', \mathsf{pk}')$ for Σ and $(\mathsf{sk}^1, \mathsf{pk}^1)$ for Σ^1
  - Sign pk^1 with sk' and sign the proof with sk^1
- Σ: key-homormorphic EUF-CMA signature scheme
- Σ^1: one-time signature scheme
- Extend CRS with a public key of Σ: pk
- Put secret key sk of Σ in simulation trapdoor

Lamassu [ARS20]

Generic framework to obtain

- subversion or updatable
- and simulation extractable zk-SNARKs

Built from

- updatable signatures
- DS compiler for simulation soundess [DS19]

Conclusion

Conclusion

CØCØ, OCØCØ:

- CØCØ hard to instantiate correctly and efficiently
- Even if commitment with enough structure used, CØCØ does not seem to yield updatability
- sub-ZK SE SNARK if underlying SNARK already sub-ZK

LAMASSU:

- generic sub-ZK, updatable SE SNARK
- Open problems: key-homomorphic / updatable signatures from lattices, ...

Questions?

Full version: `https://eprint.iacr.org/2020/062.pdf`

References

[ARS20] B. Abdolmaleki, S. Ramacher, and D. Slamanig. Lift-and-shift: obtaining simulation extractable subversion and updatable snarks generically. Cryptology ePrint Archive, Report 2020/062, 2020. https://eprint.iacr.org/2020/062, to appear at ACM CCS 2020.

[BFS16] M. Bellare, G. Fuchsbauer, and A. Scafuro. Nizks with an untrusted CRS: security in the face of parameter subversion. In *ASIACRYPT (2)*, volume 10032 of *LNCS*, pages 777–804, 2016.

[CMR98] R. Canetti, D. Micciancio, and O. Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *STOC*, pages 131–140. ACM, 1998.

References ii

[DS19] D. Derler and D. Slamanig. Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. *Des. Codes Cryptogr.*, 87(6):1373–1413, 2019.

[Fis99] M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: improvements and applications. In *EUROCRYPT*, volume 1592 of *LNCS*, pages 432–445. Springer, 1999.

[GKM+18] J. Groth, M. Kohlweiss, M. Maller, S. Meiklejohn, and I. Miers. Updatable and universal common reference strings with applications to zk-snarks. In *CRYPTO (3)*, volume 10993 of *LNCS*, pages 698–728. Springer, 2018.

[KZM+15] A. Kosba, Z. Zhao, A. Miller, Y. Qian, H. Chan, C. Papamanthou, R. Pass, abhi shelat, and E. Shi. Coco: a framework for building composable zero-knowledge proofs. Cryptology ePrint Archive, Report 2015/1093, 2015. https://eprint.iacr.org/2015/1093.