SNARKs & QA-NIZKs

Behzad Abdolmaleki

University of Tartu, Estonia

27.08.2019 / AIT, Vienna
Interactive Proofs

Prover wants to convince Verifier that $x$ has some property (i.e. “I am the doctor!” Or “$x$ is in language $L$”)

pk, sk

$I$ am the doctor!

Prove it!

sk

Commit

Challenge

Responds

pk

OK, $x \in L$
Non-Interactive Proofs

- Prover wants to convince Verifier that “$x$ is in language $L$”.

- Non-Interactive proofs usually are constructed in two models:
  - Random Oracle (RO) Model [Micali 94]
    - Commit
    - Challenge
    - Responds
  - Common Reference String (CRS) Model
    - Trusted Third Party (TTP) generates CRS
Non-interactive Zero-knowledge (NIZK)

\[ x \in L, w \]

\[ \pi = P(x, w) \]

Correctness

Soundness

Zero-knowledge
Non-interactive Zero-knowledge

- $x \in L, w$
- $\pi = P(x, w)$
- $\pi \approx \pi^*$
- $\pi^* = \text{Sim}(x)$
- $V(x, \pi)$?
Non-interactive Zero-knowledge

- $P$ should be able to compute $\pi$ only when knowing valid $w$
- $Sim$ should be able to compute $\pi$ without knowing $w$
- $Sim$ must have some extra power

$x \in L, w$

$\pi = P(x, w)$

$\pi \approx \pi^*$

$\pi^* = Sim(x)$

$V(x, \pi)$?

$V(x, \pi^*)$?
NIZK in CRS model

\[ x \in L, \pi = P(\text{crs}, x, w) \]

\[ V(\text{crs}, x, \pi) \]

\[ V(\text{crs}, x, \pi) \]

\[ V(\text{crs}, x, \pi) \]
NIZK in CRS model

\[ x \in L, w \]

\[ \pi = P \left( crs, x, w \right) \]

\[ \pi \approx \pi^* \]

\[ \pi^* = \text{Sim} \left( \text{td}, x \right) \]

\[ V \left( crs, x, \pi \right) ? \]

\[ V \left( crs, x, \pi \right) ? \]
Special Type of NIZKs: *zk*-SNARKs

\[ \text{Witness } w \ (x, w) \in R_L \]

\[(CRS, td) \leftarrow \text{CRSGen}(1^n) \]

\[ \text{proof} \leftarrow \text{Prove}(CRS, \text{witness}) \]

\[ \{1, 0\} \leftarrow \text{Verify}(CRS, \text{word, proof}) \]

We are interested to make them succinct as much as possible.

- **Succinct**: Small size of proof and low verification complexity

[Groth 2010] [Lipmaa 2012] [Gennaro Gentry Parno Raykova 2013] [Groth 2016]

**Proof = 3 group elements**
Inventions in NIZK: zk-SNARKs

- for whole NP without costly reductions
  - For known arithmetic circuit $C$ & partially unknown $x, y = C(x)$
- Very strong (non-black-box) cryptographic assumptions
  - Knowledge assumption: if $P$ outputted $x$ then $P$ must know $y$
    - [Gentry Wichs 2011]: non-black-box assumptions needed
- The CRS depends on concrete circuit
  - need to generate new CRS for each circuit!
- Applications: verifiable computation, privacy-preserving cryptocurrencies, ...
What If CRS Generator is Malicious?
Subversion security: Trust issues

Interested in **zero knowledge**
crs should contain trusted \(\text{crs}_{zk}\)

Interested in **soundness**
crs should contain trusted \(\text{crs}_{snd}\)
Subversion security: Trust issues

**Subversion ZK:** zero knowledge even if CRS creator is malicious

$$\pi = P(\text{crs}, x, w)$$

Need to trust \(\text{crs}_{zk}\)

Interested in **zero knowledge**

crs should contain trusted \(\text{crs}_{zk}\)

**Subversion SND:** soundness even if CRS creator is malicious

\((\text{crs}_{zk}, \text{crs}_{snd}), \text{td}\)

Need to trust \(\text{crs}_{snd}\)

Interested in **soundness**

crs should contain trusted \(\text{crs}_{snd}\)

Research direction: **minimize** the needed trust

\(V(\text{crs}, x, \pi)?\)
If crs creator is malicious, there is no guarantee that:

- crs is correct
- td exists
- crs is from correct distribution
Impossibility Result

- [Bellare, Fuchsbauer, Scafuro 2016]

<table>
<thead>
<tr>
<th></th>
<th>WI</th>
<th>ZK</th>
<th>Sub-ZK</th>
</tr>
</thead>
<tbody>
<tr>
<td>SND</td>
<td>Possible</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Sub-SND</td>
<td>Possible</td>
<td>Impossible</td>
<td>Impossible</td>
</tr>
</tbody>
</table>

- [Abdolmaleki, Baghery, Lipmaa, Zajac 2017]
  - Efficient SND + Sub-ZK SNARK for NP
  - Asiacrypt 2017 “top three” paper
- [Fuchsbauer 2018]: related/improved constructions
[ABLZ17] recipe for Sub-ZK SNARK

- Design a **public** algorithm CV for checking crs is correct
  - If $CV(crs) = 1$: there exists some td

Proving Sub-ZK

- If $CV(crs) = 0$: no need to simulate
- If $CV(crs) = 1$:
  - Use extractor Ext to recover (unique) td from crs
  - Simulate by using extracted Ext
  - Ext exists by assumption
Sub-ZK SNARK

Witness \( w \) 
\[ (x, w) \in \mathbb{R}_L \]

\( \text{OK, } x \in L \)

\( (\text{CRS, } \text{td}) \leftarrow \text{CRSGen}(1^n) \)

\( \text{proof } \leftarrow \text{Prove(} \text{CRS, witness} \text{)} \)

\( \{1, 0\} \leftarrow \text{Verify(} \text{CRS, word, proof} \text{)} \)

If \( 1 \leftarrow \text{CV(crs)} \)

Then

\[ \text{proof } \leftarrow \text{Prove(} \text{CRS, witness} \text{)} \]
Summary of SNARK Branch

- Privacy-preserving technologies
  - ZK proofs
    - Interactive
    - Non-interactive
      - Random Oracle Model
      - CRS Model
        - zk-SNARKs
          - Updatable zk-SNARKs
          - Subversion zk-SNARKs

- Privacy-preserving Coins (Zcash,...), Smart Contracts (HAWK,...), E-voting....
More Faster NIZKs for Linear Language:

Quasi-Adaptive NIZK (QA-NIZK)
Quasi-Adaptive NIZK

- Language parameter $\rho$ is chosen before the CRS
- $\rho$ cannot adaptively depend on the CRS
- Usually, $\rho = \text{some public key}$

Applications in constructing efficient cryptographic primitives (commitment schemes, IBE, signature schemes, ...)

QA-NIZK

- “[Kiltz, Wee 2015]
  - most efficient known QA-NIZK for SUBSPACE language

Task of QA-NIZK for SUBSPACE:
- Fix language parameter $[\rho]_1 \in G^{n \times m}$
- Prove in zero knowledge that $[\vec{x}]_1 = [\rho]_1 \vec{w}$ for some $\vec{w} \in \mathbb{Z}_p^m$

$$L = \{[\vec{x}]_1 \in G_1^n \mid \exists \vec{w} \in \mathbb{Z}_p^m \text{ s.t. } [\vec{x}]_1 = [\rho]_1 \vec{w}\}$$
Our recipe: Sub-QA-NIZK

- Design a public algorithm $PKV$ for checking $crs$ is correct
  - If $PKV(\rho, crs) = 1$: there exists some $td$

- Proving Sub-ZK
  - If $PKV(\rho, crs) = 0$: no need to simulate
  - If $PKV(\rho, crs) = 1$:
    - Use extractor $Ext$ to recover $td$ from $crs$ Simulate by using extracted $Ext$
    - $Ext$ exists by KWKE assumption.

- Extraction of $td$ requires non-black-box “knowledge assumption”
Our recipe: Sub-QA-NIZK

- Design a public algorithm PKV for checking crs is correct
  - If $\text{PKV}(\rho, \text{crs}) = 1$: there exists $\text{td}$
- Proving Sub-ZK
  - If $\text{PKV}(\rho, \text{crs}) = 0$: no need to simulate
  - If $\text{PKV}(\rho, \text{crs}) = 1$: use extractor Ext to recover $\text{td}$ from crs
    - Simulate by using extracted Ext
  - Ext exists by KWKE assumption

- Extraction of $\text{td}$ requires non-black-box “knowledge assumption”

Weakness of the current Sub-QANIZK and open problems:

- It only works for some especial construction and for extraction it needs some invertible matrix (crs elements)
- It does not fit with Simulation Sound Extractable property
- The PKV is not general and for different security parameter one needs to define a new algorithm PKV.
Recent Inventions in NIZK

<table>
<thead>
<tr>
<th>SNARK</th>
<th>QA-NIZK</th>
</tr>
</thead>
<tbody>
<tr>
<td>succinct non-interactive argument of knowledge</td>
<td>quasi-adaptive non-interactive zero knowledge</td>
</tr>
</tbody>
</table>
Recent Inventions in NIZK

<table>
<thead>
<tr>
<th></th>
<th>SNARK</th>
<th>Succinct QA-NIZK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versatility</td>
<td>Whole NP</td>
<td>Limited class of languages</td>
</tr>
<tr>
<td>Efficiency</td>
<td>3 group elements</td>
<td>1 group element</td>
</tr>
<tr>
<td>Assumptions</td>
<td>Non-black-box (knowledge)</td>
<td>Very standard</td>
</tr>
</tbody>
</table>

[Groth 2010],
[Lipmaa 2012],
[Gennaro Gentry Parno Raykova 2013],
[Parno Howell Gentry Raykova 2013],
[Groth 2016],
[Groth Maller 2017],
[Abdolmaleki Baghery Lipmaa Zajac 2017],
...

[Sub-zk-SNARK]

[Sub-zk-QA-NIZK]

[Jutla Roy 2013],
[Libert Peters Joye Yung 2014],
[Jutla Roy 2014],
[Abdalla Benhamouda Pointcheval 2015],
[Kiltz-Wee 2015],
[González Hevia Ràfols 2015],
[Libert Peters Joye Yung 2014],
[Abdolmaleki Lipmaa Siim Zajac eprit2018-19],
The Tree of Succinct NIZK

NIZK in CRS Model

QA-NIZKs

zk-SNARKs

Sub-zk-SNARKs

Updatable QANIZKs

Sub-QANIZKs

？

Updatable zk-SNARKs

Sub-zk-SNARKs

COCO Framework

Simulation Sound Extractable (sub) zk-SNARKs
The Tree of Succinct NIZK

NIZK in CRS Model

- QA-NIZKs
  - Ad-Hoc based
    - Updatable QANIZKs
    - Sub-QANIZKs
  - Modular based
    - (Under working with Daniel)

- zk-SNARKs
  - Ad-Hoc based
    - Updatable zk-SNARKs
    - Sub-zk-SNARKs
  - COCO Framework
    - Simulation Sound Extractable (sub) zk-SNARKs
      - More Efficient SSE zk-SNARKs?
        - (Under working with Sebastian and Daniel)


Thank you