Universally Composable Σ-protocols in the Global Random-Oracle Model

Anna Lysyanskaya
Leah Namisa Rosenbloom

Brown University

TCC 2022

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKP.K)

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-PROTOCOLS made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-PROTOCOLS made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application
  - Value v

Alice

Bob

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-PROTOCOLS made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application

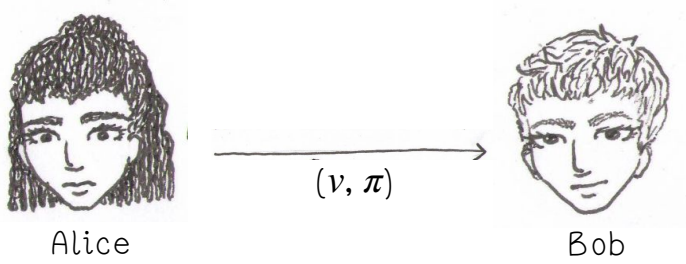
- Value v
- Proof π that v is correct

Alice

Bob

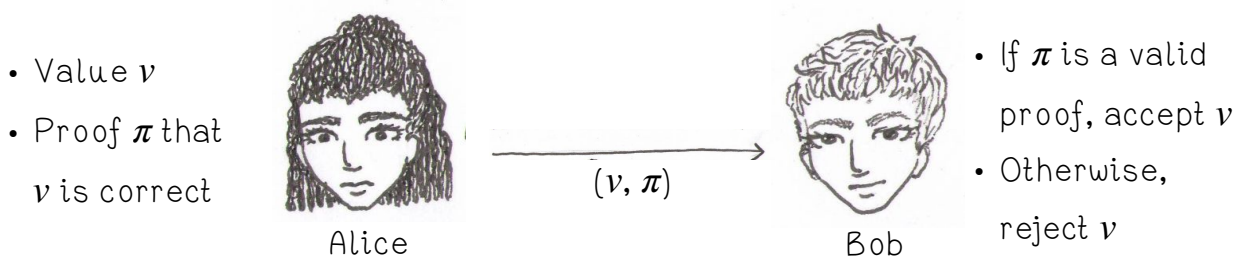
Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-PROTOCOLS made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application
  - Value v
  - Proof π that v is correct



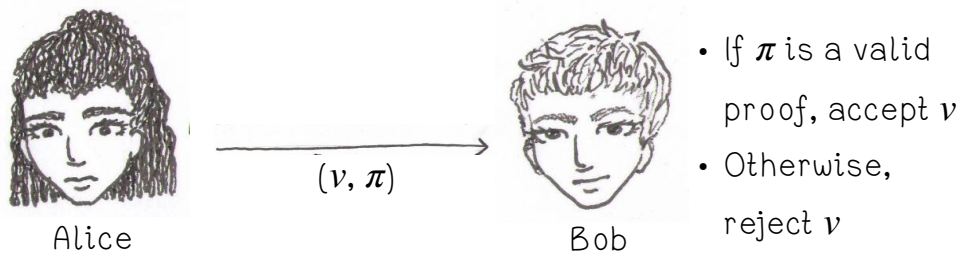
Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-PROTOCOLS made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application



Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-protocols made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application



signatures
- group [CS97]
- blind [HKL19]
- threshold [CKM21]
- aggregate [Ks22]
- multi- [DEF+19]

anonymous
- networks [CBM15]
- credentials [CLØ1]
- e-cash [CHLØ5]
- voting [AdidaØ8]

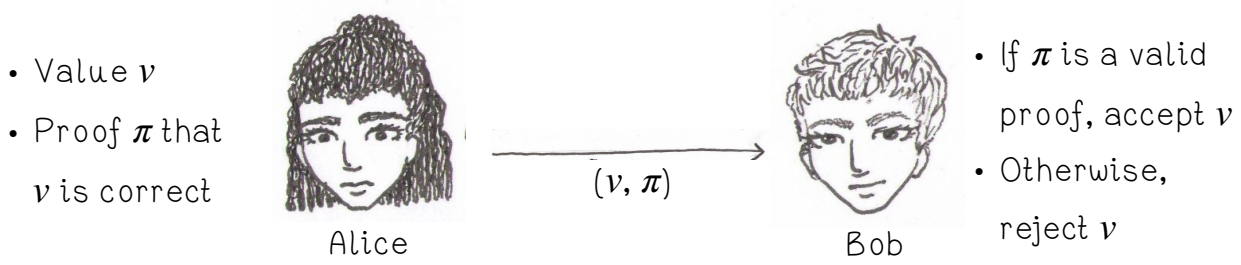
distributed ledgers [PR18]

cryptographic shuffles [WikströmØ9]

verifiable
- secret sharing [Pedersen92]
- encryption [CDØØ]

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-protocols made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application

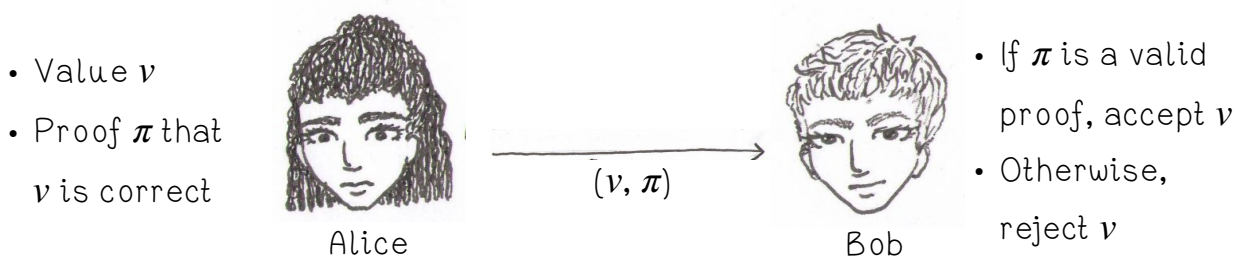


Pros

- Fiat-Shamir transform is efficient
- NIZKPoK are provably secure in the ROM (standalone security)

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-protocols made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application



Pros

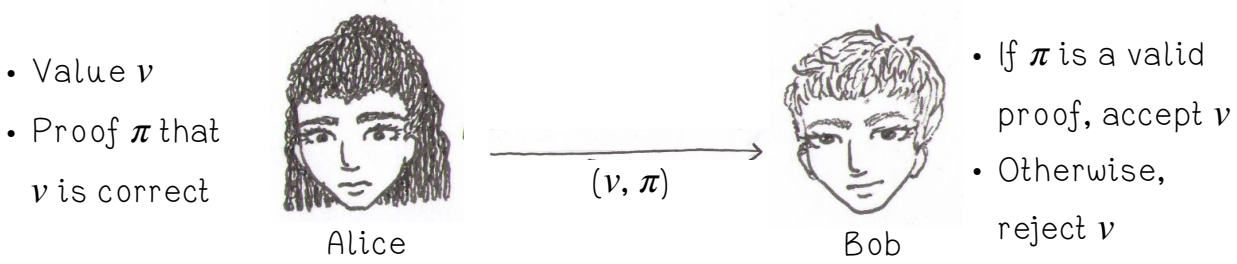
- Fiat-Shamir transform is efficient
- NIZKPoK are provably secure in the ROM (standalone security)

Cons

- Security is not guaranteed when NIZKPoK are composed
- Composable security is non-trivial

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- Σ-protocols made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application



Security of Blind Discrete Log Signatures against Interactive Attacks

Claus Peter Schnorr

A Generalized Birthday Problem (Extended Abstract)

David Wagner

On the (in)security of ROS

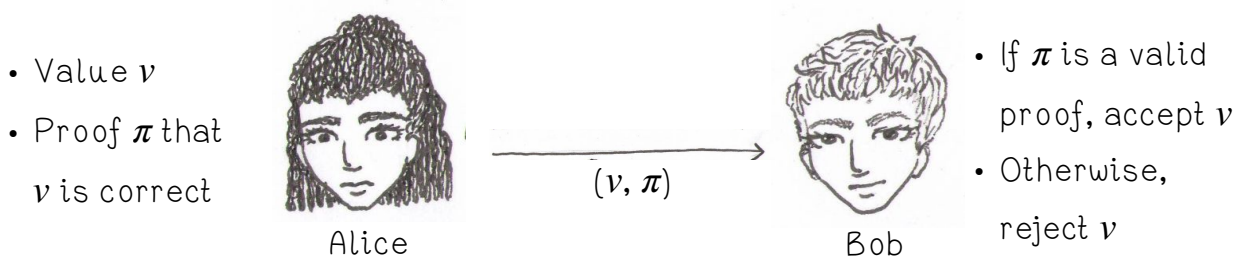
Fabrice Benhamouda[1], Tancrède Lepoint[2], Julian Loss[3], Michele Orrù[4], and Mariana Raykova[5]

On the Security of Two-Round Multi-Signatures

Manu Drijvers*†, Kasra Edalatnejad‡, Bryan Ford‡, Eike Kiltz§, Julian Loss§, Gregory Neven*, Igors Stepanovs¶

Typical Non-Interactive Zero-Knowledge Proofs of Knowledge (NIZKPoK)

- **Σ-PROTOCOLS** made efficient & non-interactive
  - using the Fiat-Shamir transform (Fiat & Shamir '86)
  - in the random-oracle model (ROM) (Bellare & Rogaway, '93)
- Common application



★ WANT: universally composable (UC) NIZKPoK with reasonable efficiency in the ROM

This Work

1) Defines NIZKPoK in a composition-friendly way

This Work

1) Defines NIZKPoK in a composition-friendly way

2) Identifies standard security properties that are both necessary and sufficient to obtain UC NIZKPoK in the global ROM

This Work

1) Defines NIZKPoK in a composition-friendly way

2) Identifies standard security properties that are both necessary and sufficient to obtain UC NIZKPoK in the global ROM

3) Realizes efficient UC NIZKPoK in the global ROM using the randomized Fischlin transform (Fischlin '05, Kondi & shelat '22)

This Work

1) Defines NIZKPoK in a composition-friendly way

2) Identifies standard security properties that are both necessary and sufficient to obtain UC NIZKPoK in the global ROM

3) Realizes efficient UC NIZKPoK in the global ROM using the randomized Fischlin transform (Fischlin '05, Kondi & shelat '22)

 Implications: plug-and-play NIZKPoK for any application (regardless of composition) in the global ROM.

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{\text{NIZK}}$

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{\text{NIZK}}$

Setup

Prove

Verify

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$

Setup

$\mathcal{F}_{NIZK}^{s}$

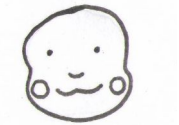
Prove

Verify

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$

Setup



HONEST PARTY (session s)

Setup session s

$\mathcal{F}_{NIZK}^{s}$

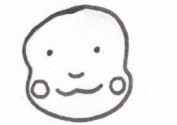
Prove

Verify

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$

Setup



HONEST PARTY (session s)

Setup session
s

$\mathcal{F}_{NIZK}^{S}$

IDEAL ADVERSARY

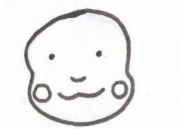
Prove

Verify

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$

Setup



HONEST PARTY (session s)

Setup session s →

$\mathcal{F}_{NIZK}^{S}$

Setup session s →

IDEAL ADVERSARY

Prove

Verify

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$

Algorithms:

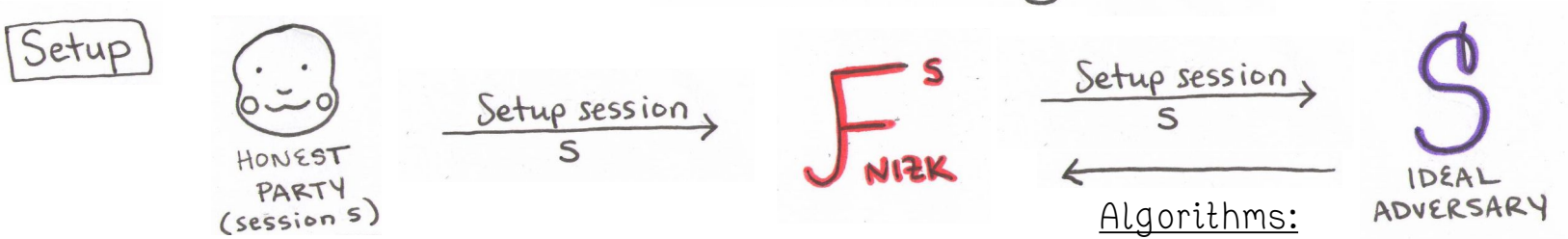
(Setup, Prove, Verify, Simulate, Extract)

Prove

Verify

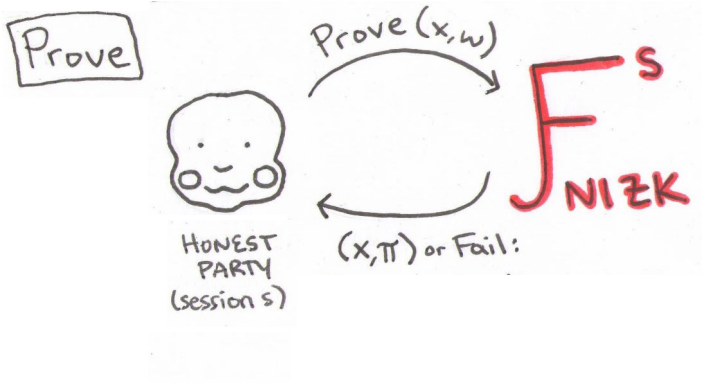
Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{NIZK}$



Algorithms:

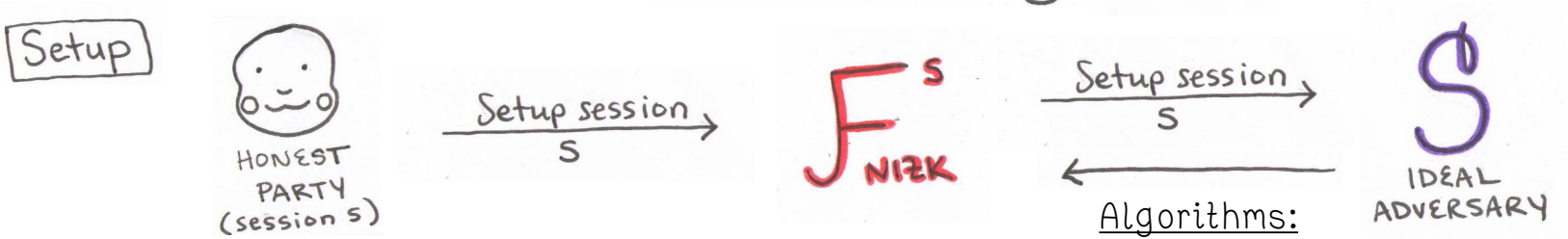
(Setup, Prove, Verify, Simulate, Extract)



Verify

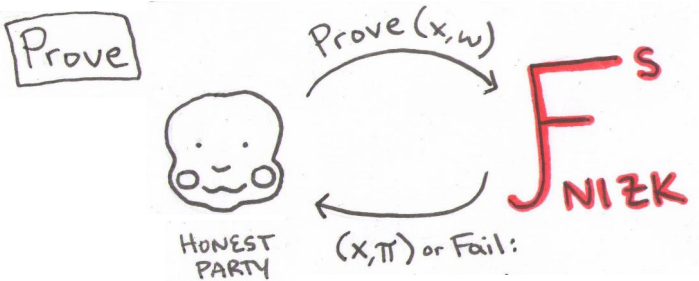
Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{\text{NIZK}}$



Algorithms:

(Setup, Prove, Verify, Simulate, Extract)

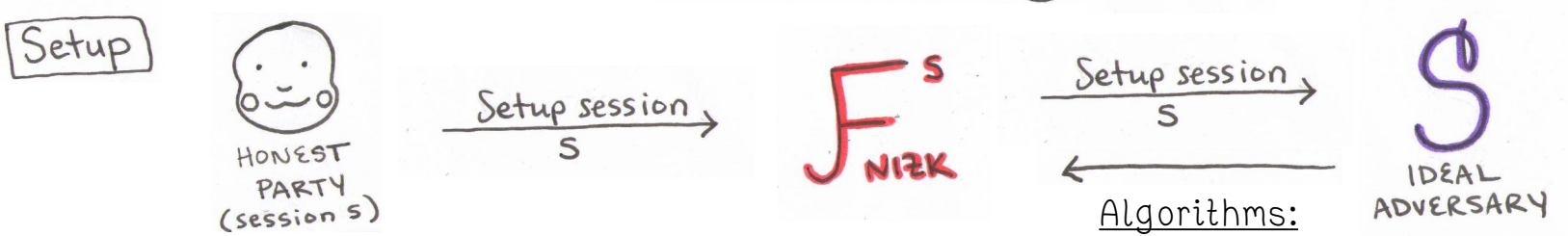


Verify

- throw out w
- compute π according to **Simulate**
- if Verify$(x, \pi) = \emptyset$, output Fail.
- otherwise, output (x, π).

Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{\text{NIZK}}$



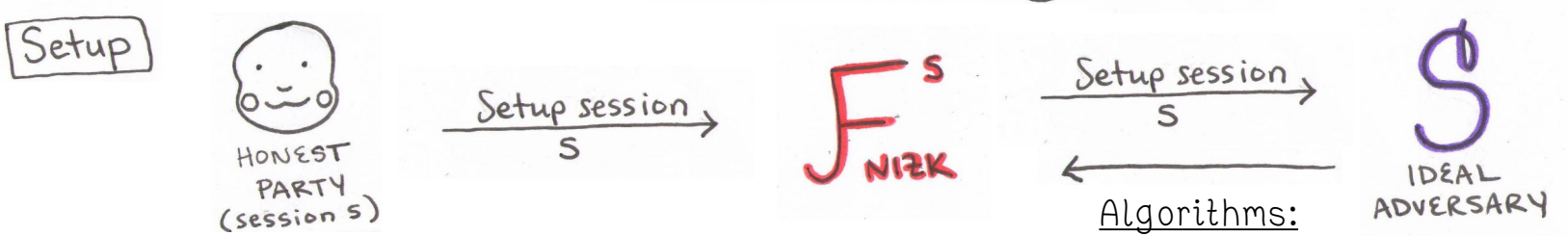
Algorithms:

(Setup, Prove, Verify, Simulate, Extract)

- throw out w
- compute π according to **Simulate**
- if Verify$(x, \pi) = \emptyset$, output Fail.
- otherwise, output (x, π).

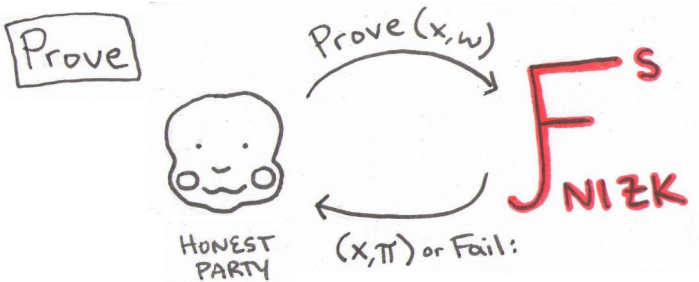
Non-Interactive Zero-Knowledge Proof of Knowledge

Ideal Functionality $\mathcal{F}_{\text{NIZK}}$

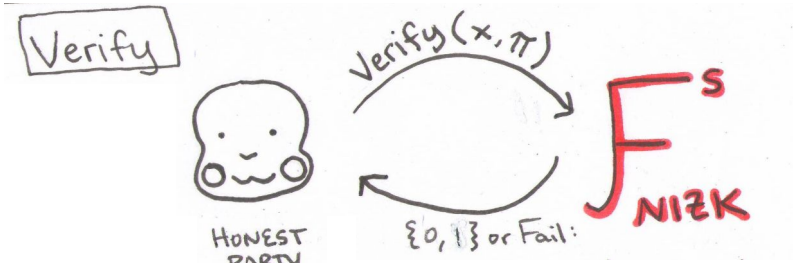


Algorithms:

(Setup, Prove, Verify, Simulate, Extract)



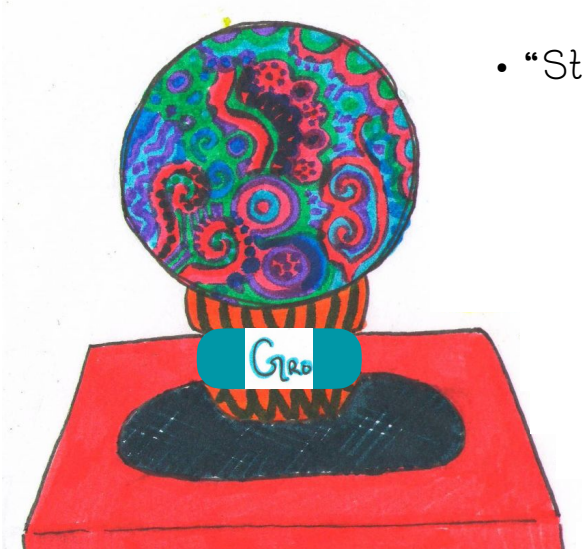
- throw out w
- compute π according to **Simulate**
- if Verify$(x, \pi) = \emptyset$, output Fail.
- otherwise, output (x, π).



- if π is valid and not simulated:
  - compute w according to **Extract**
  - if $R(x, w) = \emptyset$, output Fail.
- otherwise, output verification.

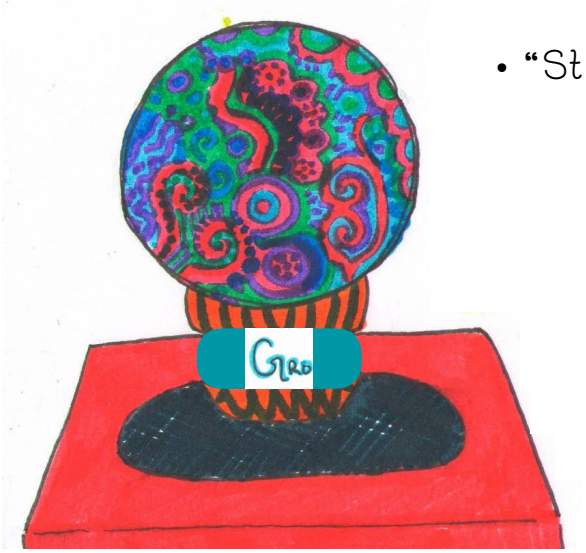
Global Random Oracles

Global Random Oracles



- "Strict" Global Random Oracle G_{RO}

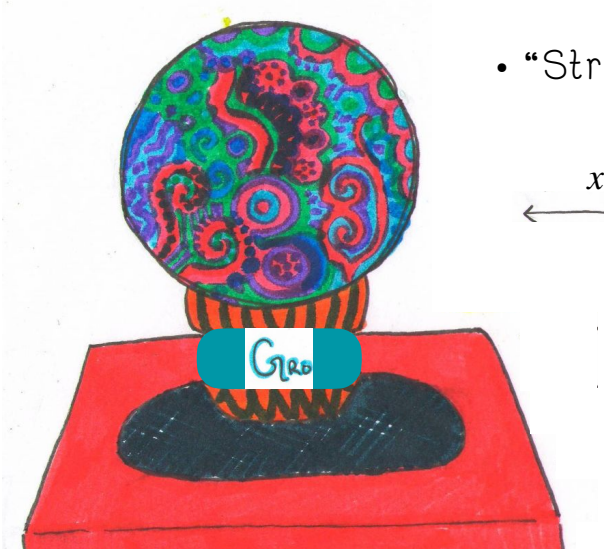
Global Random Oracles



- "Strict" Global Random Oracle G_{RO}

Alice

Global Random Oracles

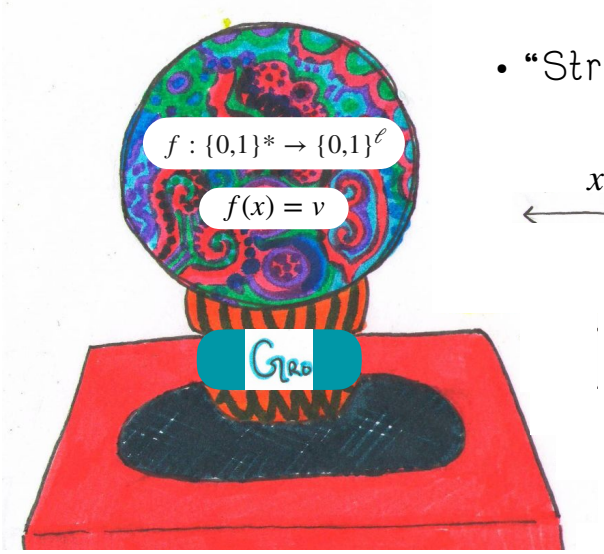


- "Strict" Global Random Oracle G_{RO}

$x \in \{0,1\}^*$ ←

Alice

Global Random Oracles

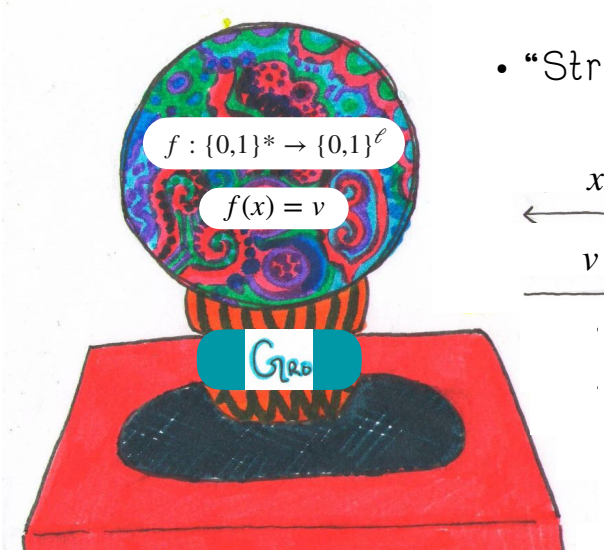


- "Strict" Global Random Oracle G_{RO}

$x \in \{0,1\}^*$ ←

Alice

Global Random Oracles



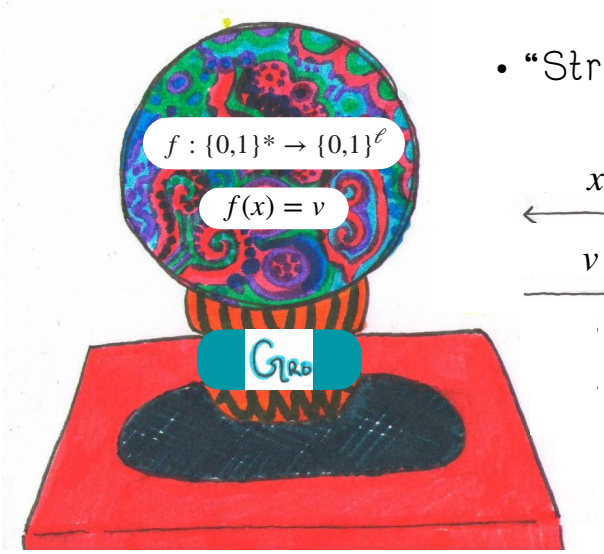
- "Strict" Global Random Oracle G_{RO}

$x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

Alice

Global Random Oracles



- "Strict" Global Random Oracle $\mathcal{G}_{\mathrm{RO}}$

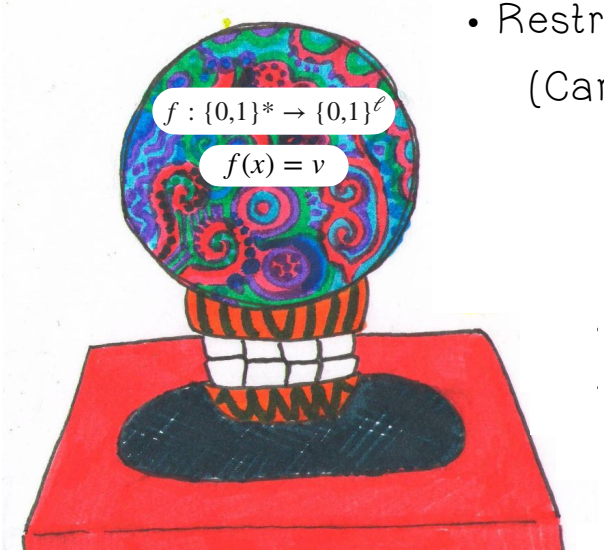
$x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

Alice

- Impossible to achieve NIZKPoK using a strictly global setup (Pass '03; Canetti, Jain, & Scafuro '14)

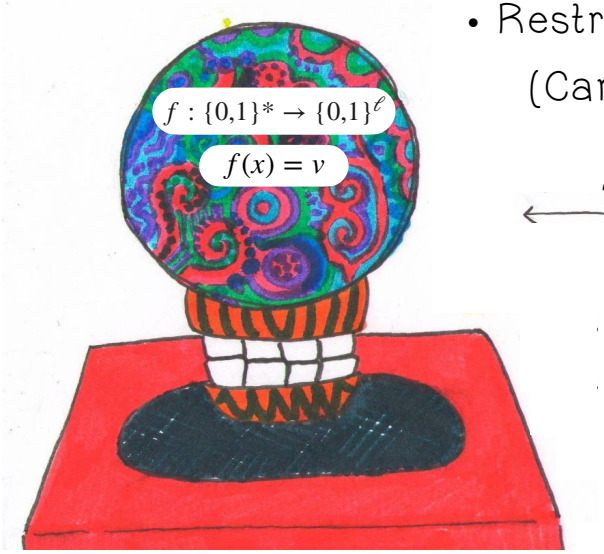
Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$
  (Canetti, Jain, & Scafuro '14)

Alice
sid s

Global Random Oracles

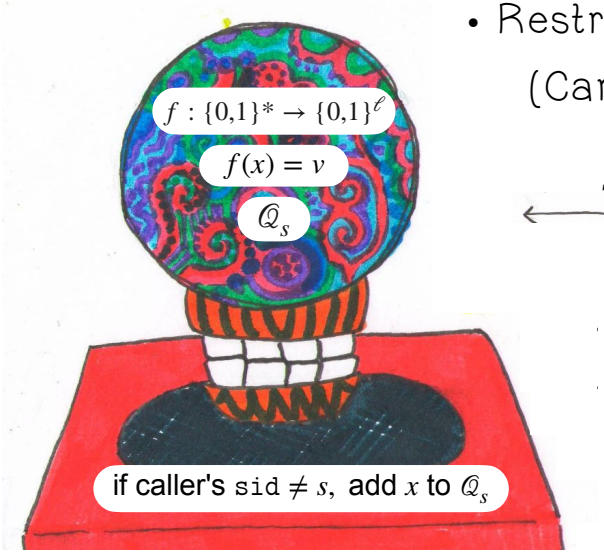


- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$
  (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ⟵

Alice
sid s

Global Random Oracles

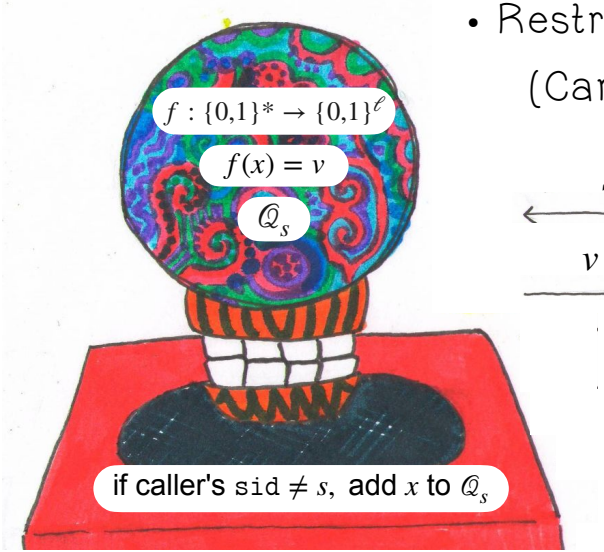


- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$
  (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

Alice
sid s

Global Random Oracles



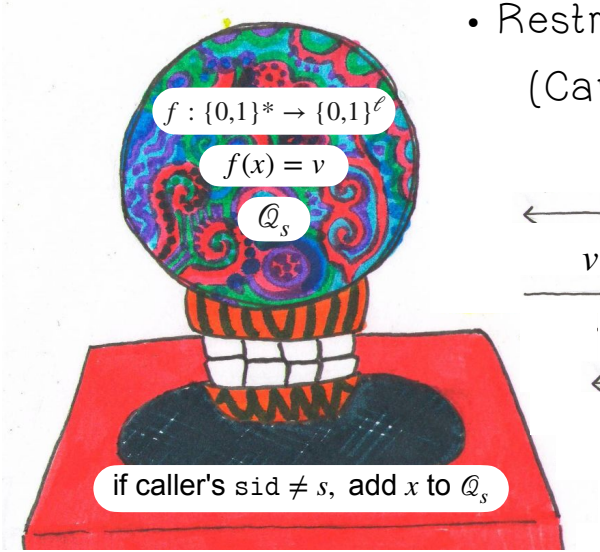
- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$
  (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$ (Canetti, Jain, & Scafuro '14)

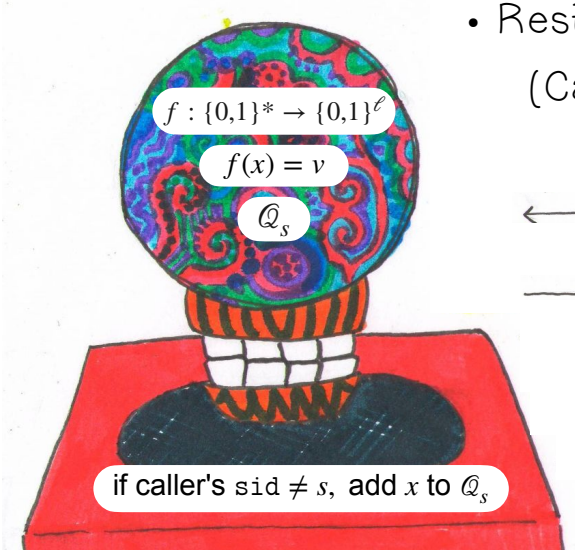
$s, x \in \{0,1\}^*$ ←

$v \in \{0,1\}^\ell$ →

illquery(s) ←

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{rORO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$

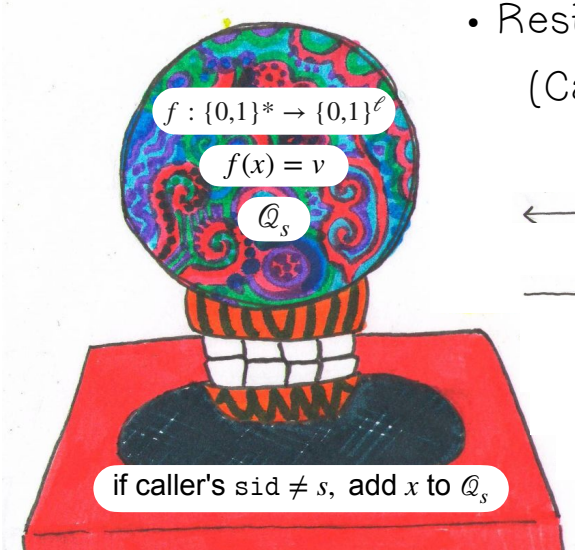
$v \in \{0,1\}^\ell$

illquery(s)

$\mathcal{Q}_s$

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RoRO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

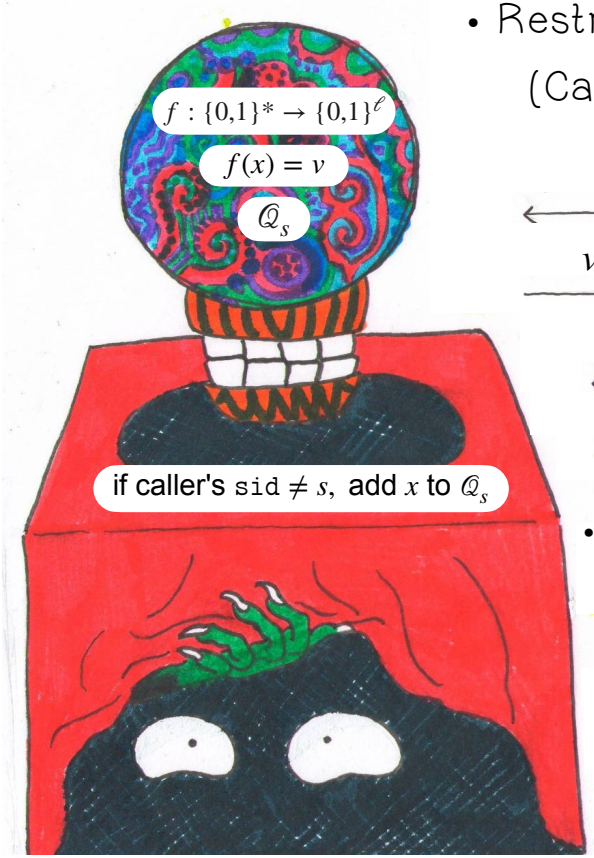
illquery(s) ←

→ $\mathcal{Q}_s$

Alice
sid s

- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

illquery(s) ←

→ $\mathcal{Q}_s$

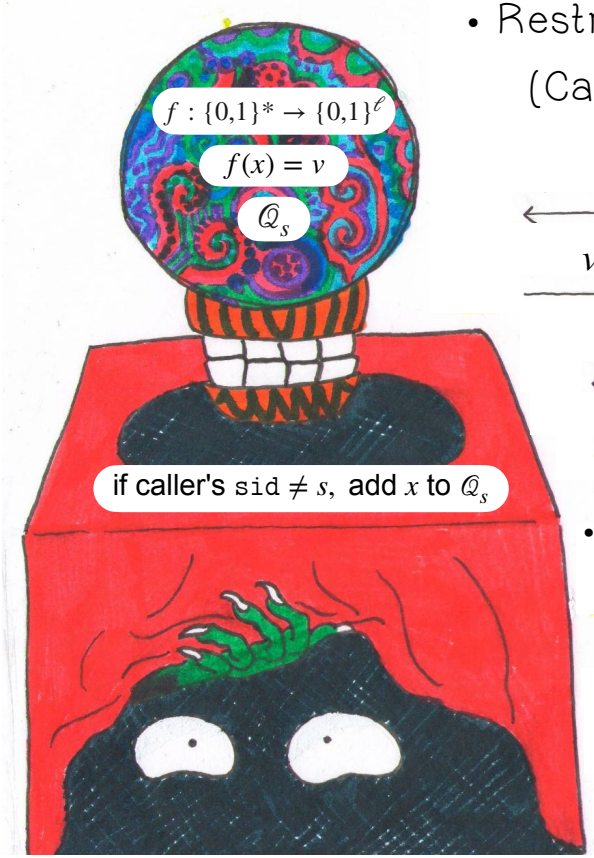
Alice
sid s

- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{RPORO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

Alice
sid s

illquery(s) ←

→ $\mathcal{Q}_s$

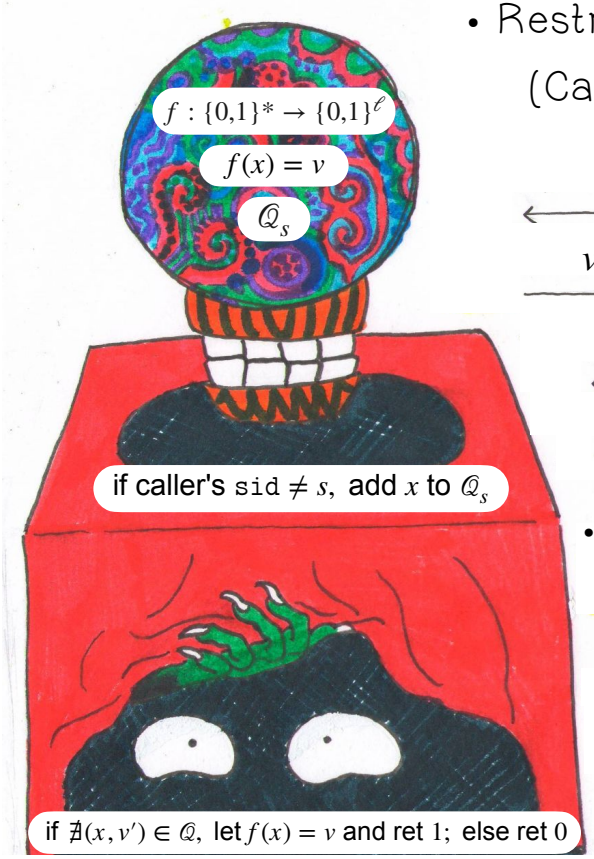
- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{rpoRO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)

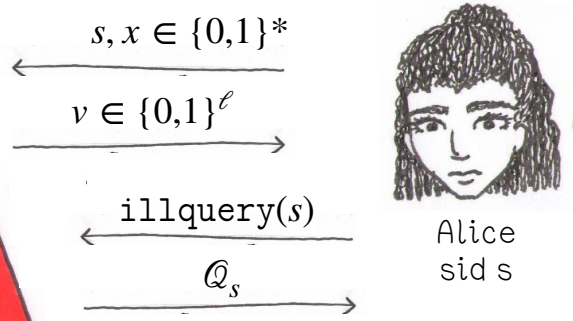
s, prog(x, v) ←

Alice
sid s

Global Random Oracles

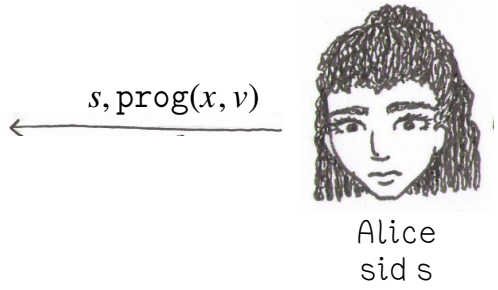


- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$ (Canetti, Jain, & Scafuro '14)

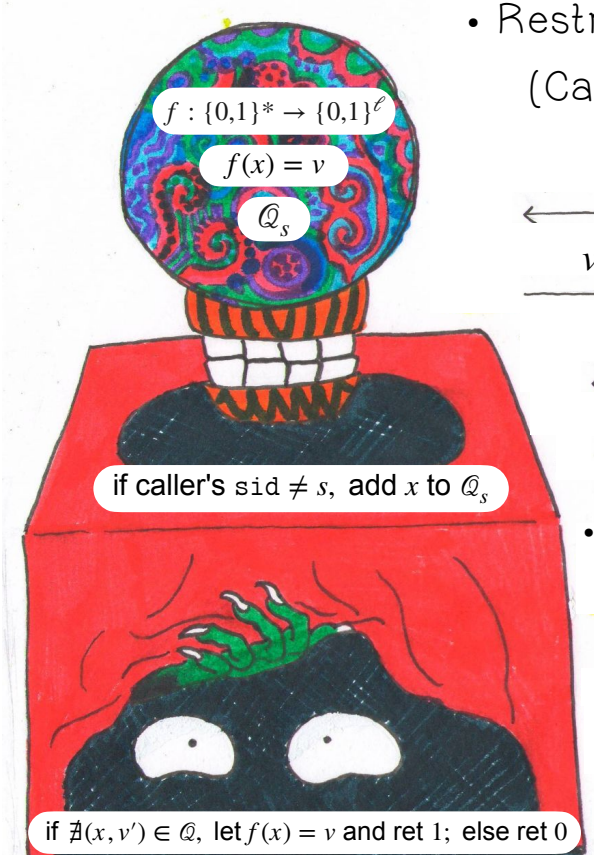


- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

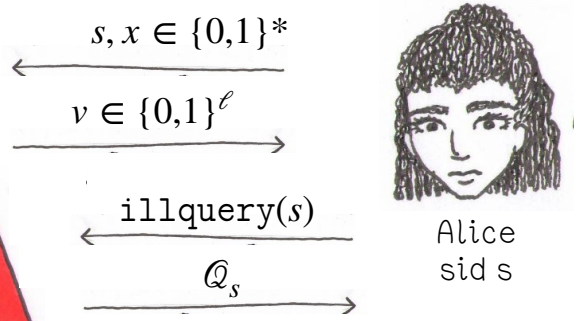
- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{RPORO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)



Global Random Oracles

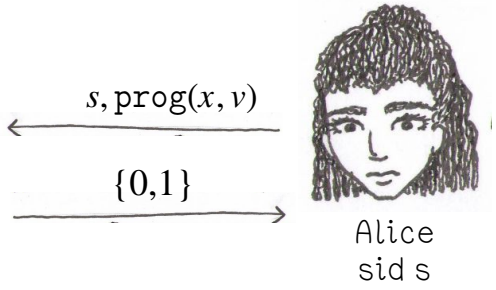


- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{ROM}}$ (Canetti, Jain, & Scafuro '14)

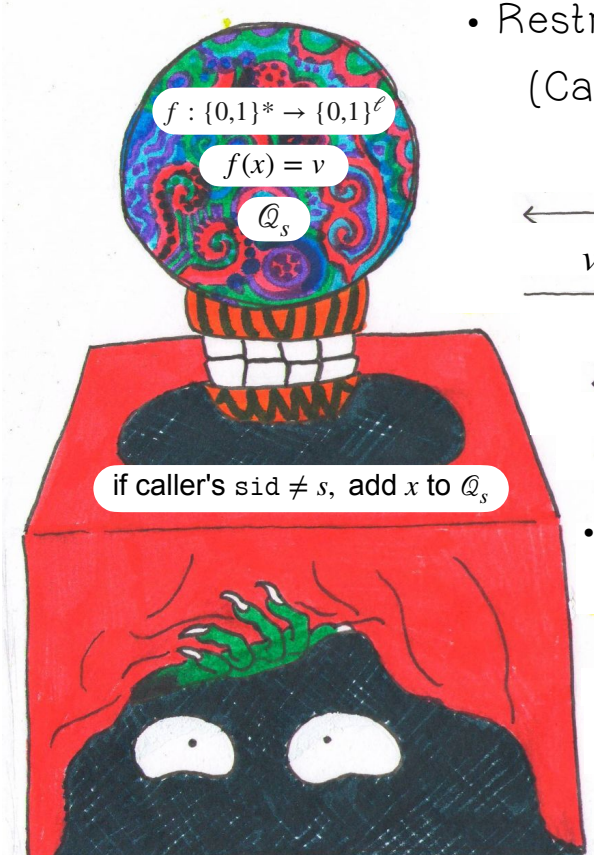


- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{RPOM}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)



Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$

$v \in \{0,1\}^\ell$

$\texttt{illquery}(s)$

$\mathcal{Q}_s$

Alice
sid s

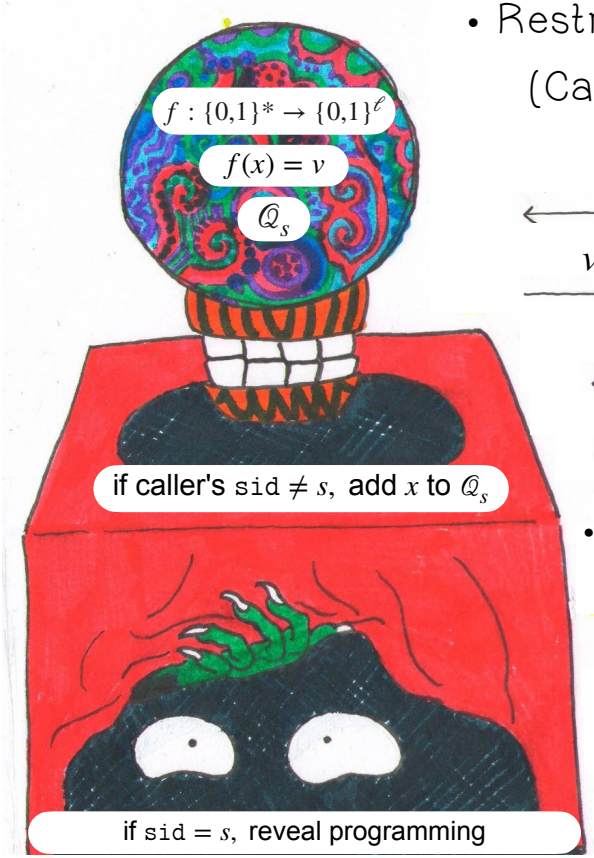
- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{rpoRO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)

$s, \texttt{isProg}(x)?$

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

illquery(s) ←

→ $\mathcal{Q}_s$

Alice
sid s

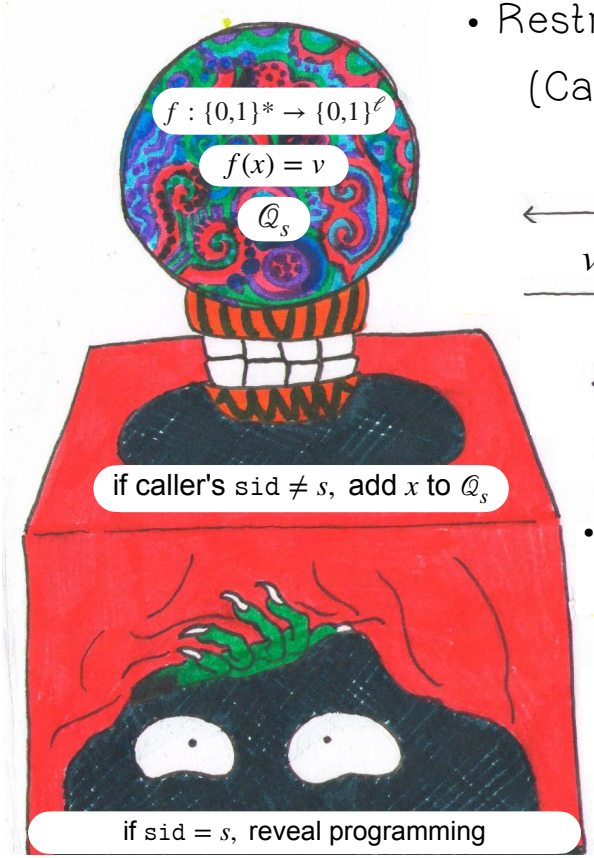
- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{RPORO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)

s, isProg(x)? ←

Alice
sid s

Global Random Oracles



- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{roRO}}$ (Canetti, Jain, & Scafuro '14)

$s, x \in \{0,1\}^*$ ←

→ $v \in \{0,1\}^\ell$

illquery(s) ←

→ $\mathcal{Q}_s$

Alice
sid s

- Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

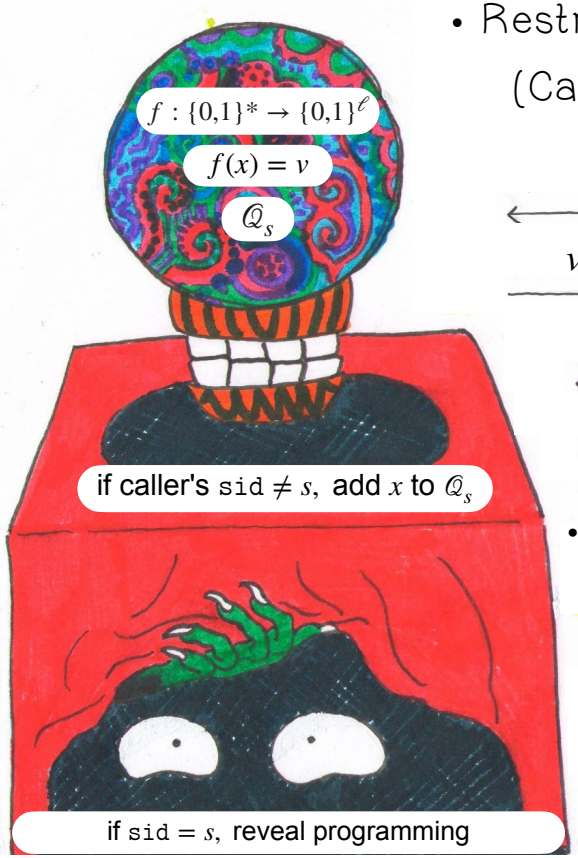
- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{rpoRO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)

s, isProg(x)? ←

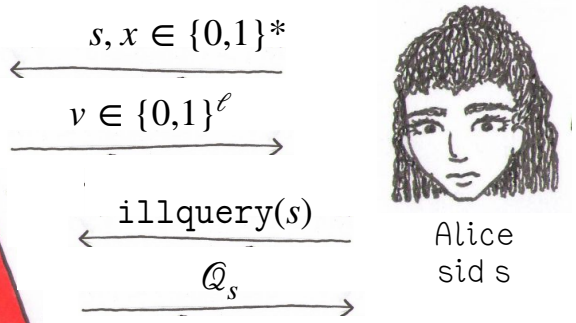
→ $\{0,1\}$

Alice
sid s

Global Random Oracles

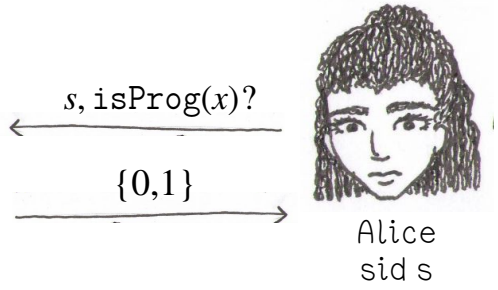


- Restricted Observable Global Random Oracle $\mathcal{G}_{\text{RORO}}$ (Canetti, Jain, & Scafuro '14)



  - Allows extractor to observe list of "illegitimate queries" $\mathcal{Q}_s$

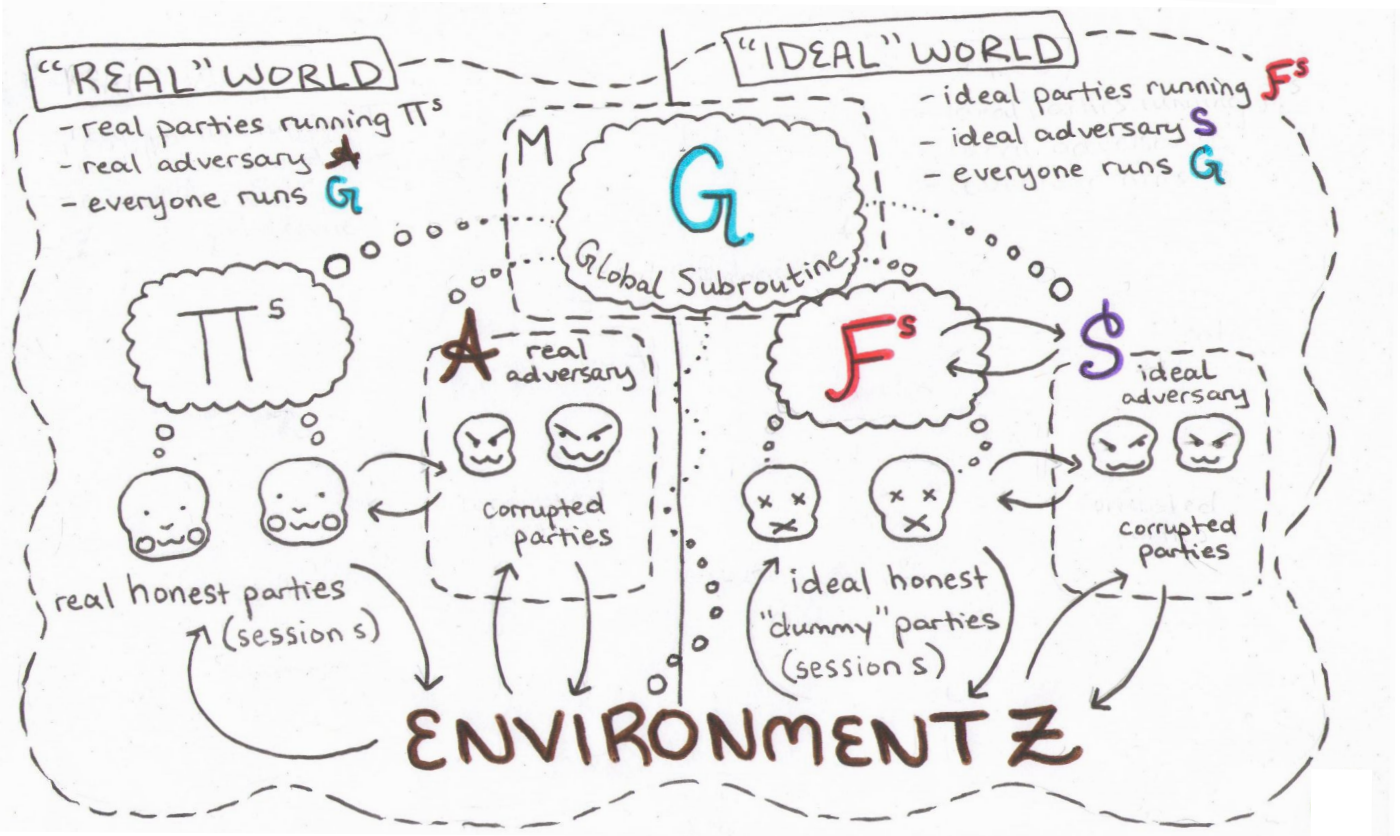
- Restricted Programmable Observable Global Random Oracle $\mathcal{G}_{\text{RPORO}}$ (Camenisch, Drijvers, Gagliardoni, Lehmann, & Neven '18)



  - Allows simulator to program
  - Reveals programming only to "legitimate" session participants

Universal Composability with Global Subroutines

(UCGS) Model (Badertscher, Canetti, Hesse, Tackman, & Zikas, '20)



Theorem 1.

Let π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{CRS}}$-hybrid model.

Then π must be...

Theorem 1.

Let π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{CRS}}$-hybrid model.

Then π must be... (1) overwhelmingly complete

Theorem 1.

Let Π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{CRS}}$-hybrid model.

Then Π must be... (1) overwhelmingly complete (2) non-interactive multiple special honest-verifier zero-knowledge (NIM-SHVZK)

Theorem 1.

Let Π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{CRS}}$-hybrid model.

Then Π must be...

1. overwhelmingly complete
2. non-interactive multiple special honest-verifier zero-knowledge (NIM-SHVZK)
3. non-interactive special simulation-sound (NI-SSS).

Theorem 1.

Let Π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{RO}}$-hybrid model.

Then Π must be... (1) overwhelmingly complete (2) non-interactive multiple special honest-verifier zero-knowledge (NIM-SHVZK) (3) non-interactive special simulation-sound (NI-SSS).

1) Overwhelming Completeness: given honestly-computed π for $x \in L_R$, Pr[V Accepts] = $1 - \mathbf{negl}(\lambda)$

Theorem 1.

Let Π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{CRS}}$-hybrid model.

Then Π must be...

1. overwhelmingly complete
2. non-interactive multiple special honest-verifier zero-knowledge (NIM-SHVZK)
3. non-interactive special simulation-sound (NI-SSS).

1) Overwhelming Completeness: given honestly-computed π for $x \in L_R$, Pr[V Accepts] = $1 - \mathbf{negl}(\lambda)$

2) Non-Interactive Multiple Special Honest-Verifier Zero-Knowledge (NIM-SHVZK):
poly-many proofs from **SimProve** are indistinguishable from poly-many proofs from **Prove**

Theorem 1.

Let Π be a protocol that UC-realizes $\mathcal{F}_{\text{NIZK}}$ in any $\mathcal{G}_{\text{RO}}$-hybrid model.

Then Π must be... (1) overwhelmingly complete (2) non-interactive multiple special honest-verifier zero-knowledge (NIM-SHVZK) (3) non-interactive special simulation-sound (NI-SSS).

1) Overwhelming Completeness: given honestly-computed π for $x \in L_R$, Pr[V Accepts] = $1 - \mathbf{negl}(\lambda)$

2) Non-Interactive Multiple Special Honest-Verifier Zero-Knowledge (NIM-SHVZK): poly-many proofs from **SimProve** are indistinguishable from poly-many proofs from **Prove**

3) Non-Interactive Special Simulation-Soundness (NI-SSS): given a valid and non-simulated proof, the **Extract** algorithm can extract a valid witness using only the adversary's RO queries, even when the adversary gets to see poly-many proofs from **SimProve**

Straight-line Compilers

A straight-line compiler SLC takes a Σ-protocol $\Sigma_{R,\tau}$ as input and produces a tuple of algorithms $\Pi^{\mathsf{SLC}}_{R,\tau}$ with the following properties:

Straight-line Compilers

A straight-line compiler SLC takes a Σ-protocol $\Sigma_{R,\tau}$ as input and produces a tuple of algorithms $\Pi^{\mathsf{SLC}}_{R,\tau}$ with the following properties:

$$\Pi^{\mathsf{SLC}}_{R,\tau} = (\mathsf{Setup}^{\mathrm{RO}}, \mathsf{Prove}^{\mathrm{RO}}, \mathsf{Verify}^{\mathrm{RO}}, \mathsf{SimSetup}, \mathsf{SimProve}, \mathsf{Extract})$$

Straight-line Compilers

A straight-line compiler SLC takes a Σ-protocol $\Sigma_{R,\tau}$ as input and produces a tuple of algorithms $\Pi^{\mathsf{SLC}}_{R,\tau}$ with the following properties:

$$\Pi^{\mathsf{SLC}}_{R,\tau} = (\mathbf{Setup}^{\mathsf{RO}}, \mathbf{Prove}^{\mathsf{RO}}, \mathbf{Verify}^{\mathsf{RO}}, \mathbf{SimSetup}, \mathbf{SimProve}, \mathbf{Extract})$$

1) Overwhelming Completeness: given honestly-computed π for $x \in L_R$, Pr[V Accepts] = $1 - \mathbf{negl}(\lambda)$

2) Non-Interactive Multiple Special Honest-Verifier Zero-Knowledge (NIM-SHVZK): poly-many proofs from **SimProve** are indistinguishable from poly-many proofs from **Prove**

3) Non-Interactive Special Simulation-Soundness (NI-SSS): given a valid and non-simulated proof, the **Extract** algorithm can extract a valid witness using only the adversary's RO queries, even when the adversary gets to see poly-many proofs from **SimProve**

Straight-line Compilers

A straight-line compiler SLC takes a Σ-protocol $\Sigma_{R,\tau}$ as input and produces a tuple of algorithms $\Pi^{\mathsf{SLC}}_{R,\tau}$ with the following properties:

$$\Pi^{\mathsf{SLC}}_{R,\tau} = (\mathbf{Setup}^{\mathsf{RO}}, \mathbf{Prove}^{\mathsf{RO}}, \mathbf{Verify}^{\mathsf{RO}}, \mathbf{SimSetup}, \mathbf{SimProve}, \mathbf{Extract})$$

1) Overwhelming Completeness: given honestly-computed π for $x \in L_R$, Pr[V Accepts] = $1 - \mathbf{negl}(\lambda)$

2) Non-Interactive Multiple Special Honest-Verifier Zero-Knowledge (NIM-SHVZK): poly-many proofs from **SimProve** are indistinguishable from poly-many proofs from **Prove**

3) Non-Interactive Special Simulation-Soundness (NI-SSS): given a valid and non-simulated proof, the **Extract** algorithm can extract a valid witness using only the adversary's RO queries, even when the adversary gets to see poly-many proofs from **SimProve**

Theorem: the randomized Fischlin transform (Fischlin '05, Kondi & shelat '22) is a straight-line compiler.

Theorem 2.

Let

- $\Sigma_{R,\tau}$ be any Σ-protocol,
- G_{rpoRO} be the restricted programmable observable global RO, &
- SLC be any straight-line compiler.

Theorem 2.

Let

- $\Sigma_{R,\tau}$ be any Σ-protocol,
- G_{rpoRO} be the restricted programmable observable global RO, &
- SLC be any straight-line compiler.

Then the tuple of algorithms $\Pi^{SLC}_{R,\tau}$ obtained by running $\Sigma_{R,\tau}$ through SLC UC-realizes F_{NIZK} in the G_{rpoRO}-hybrid model.

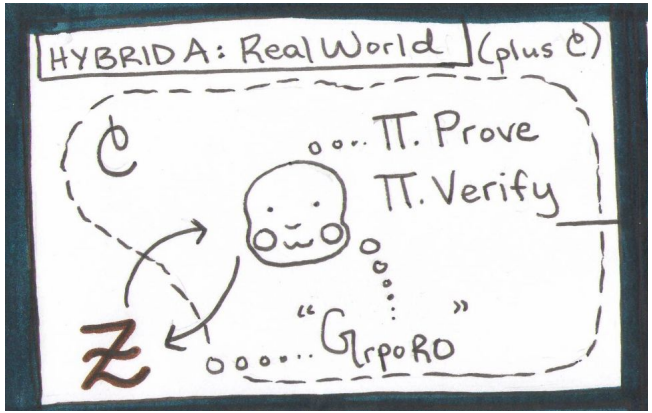
Theorem 2 (Proof).

Recall: want to prove

Theorem 2 (Proof).

Recall: want to prove

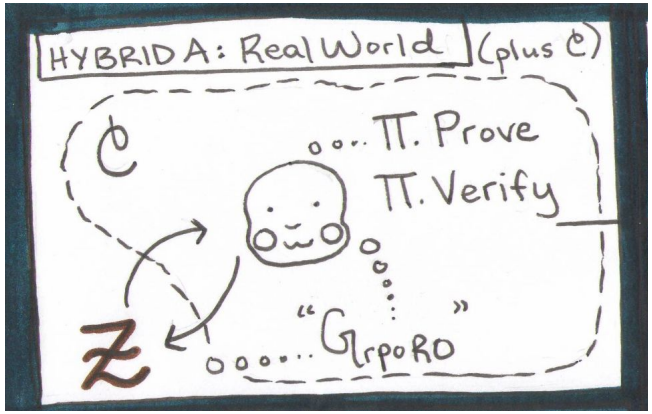
$$\text{IDEAL}^{\mathcal{G}_{\text{rpoRO}}}_{\mathcal{F}_{\text{mint}},S,\mathcal{Z}} \approx \text{REAL}^{\mathcal{G}_{\text{rpoRO}}}_{\pi^{\text{SLC}}_{R,\tau},\mathcal{A},\mathcal{Z}}$$

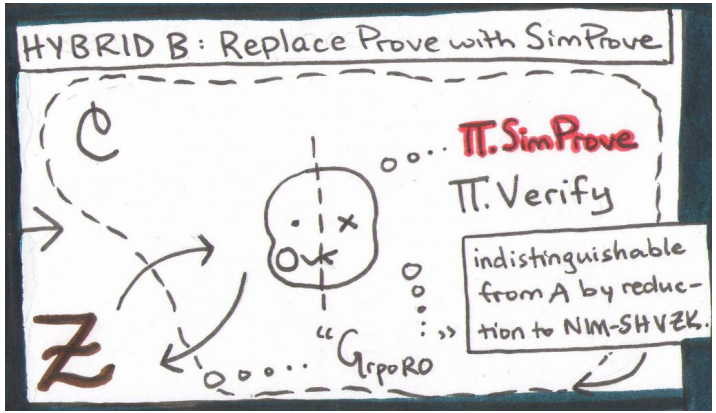


Theorem 2 (Proof).

Recall: want to prove

$$\text{IDEAL}^{\mathcal{G}_{\text{RPoRO}}}_{\mathcal{F}_{\text{NIZK}},S,\mathcal{Z}} \approx \text{REAL}^{\mathcal{G}_{\text{RPoRO}}}_{\pi^{\text{SLC}}_{R,\tau},\mathcal{A},\mathcal{Z}}$$

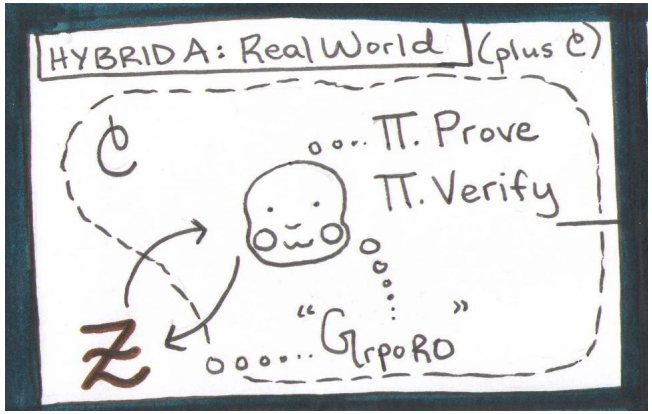


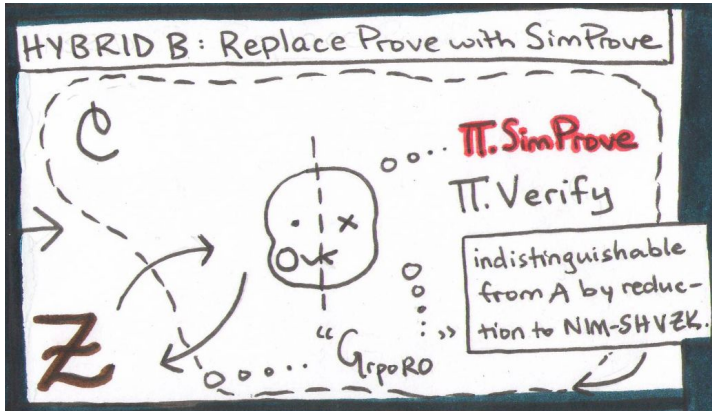


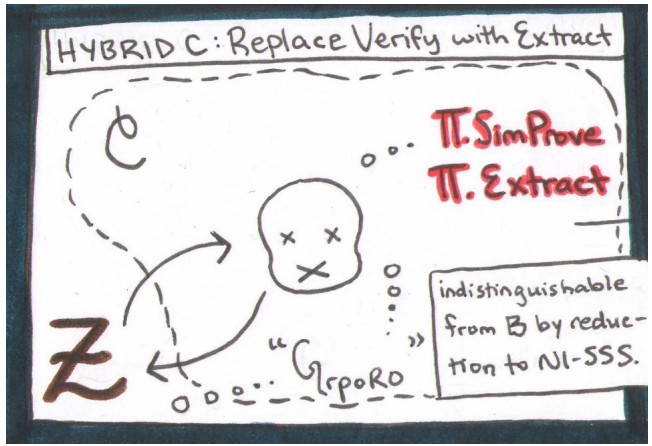
Theorem 2 (Proof).

Recall: want to prove

$$\text{IDEAL}^{\mathcal{G}_{\text{rpoRO}}}_{\mathcal{F}_{\text{NIZK}},S,\mathcal{Z}} \approx \text{REAL}^{\mathcal{G}_{\text{rpoRO}}}_{\Pi^{\text{SLC}}_{R,\tau},\mathcal{A},\mathcal{Z}}$$



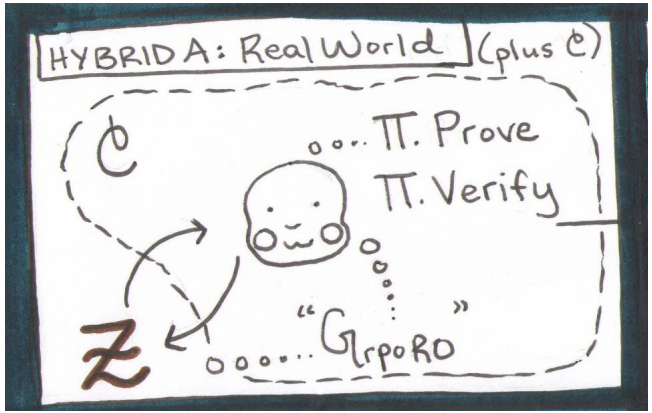


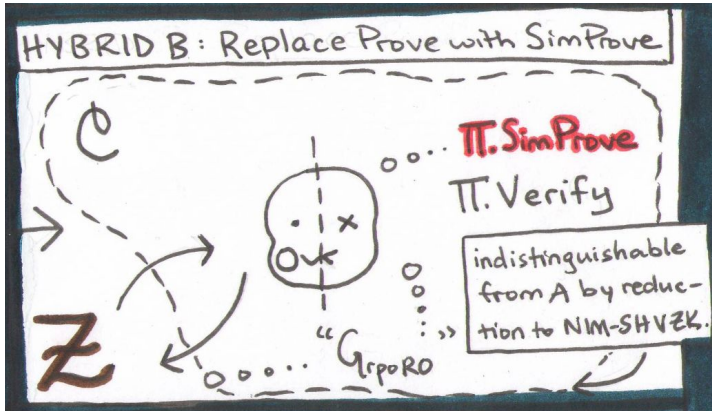


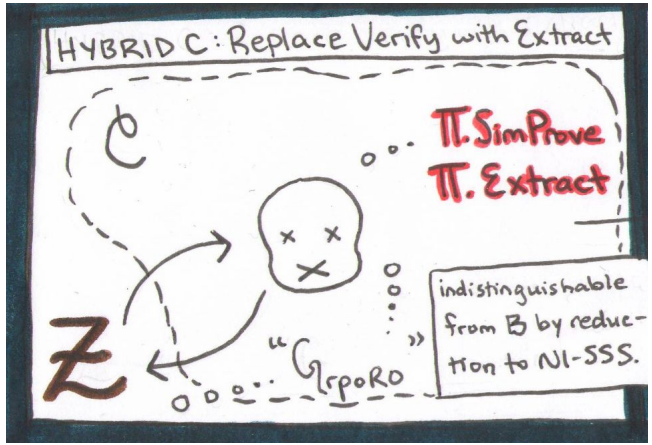
Theorem 2 (Proof).

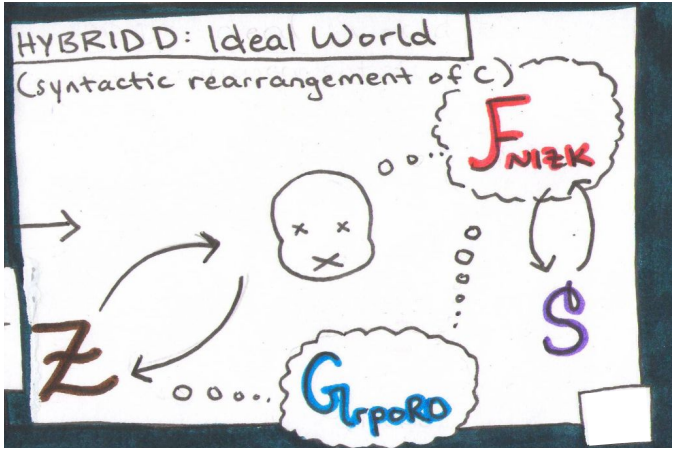
Recall: want to prove

$$\text{IDEAL}^{\mathcal{G}_{\text{rpoRO}}}_{\mathcal{F}_{\text{NIZK}},S,\mathcal{Z}} \approx \text{REAL}^{\mathcal{G}_{\text{rpoRO}}}_{\pi^{\text{SLC}}_{R,\tau},\mathcal{A},\mathcal{Z}}$$





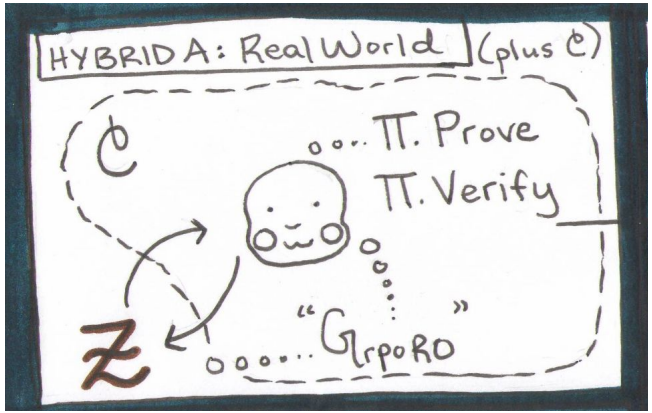


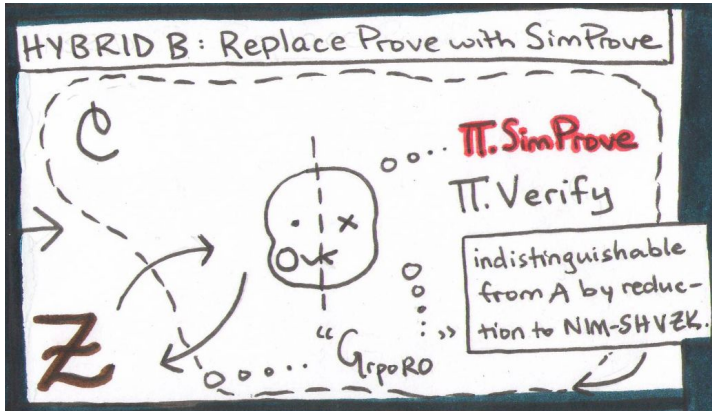


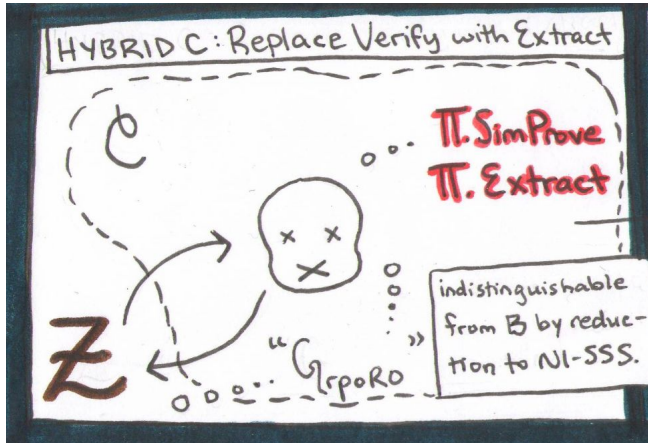
Theorem 2 (Proof).

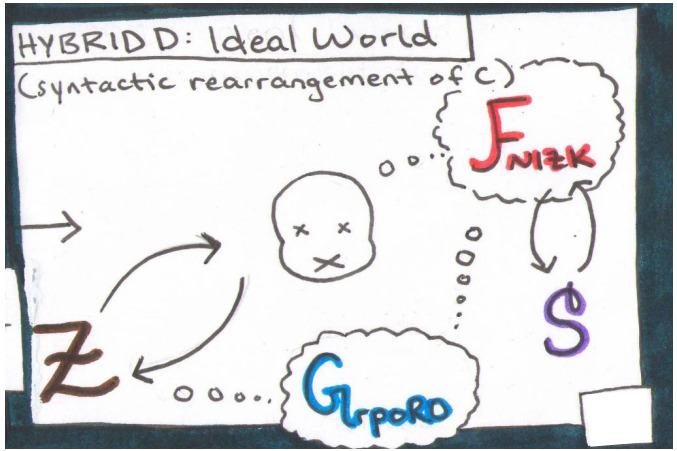
Recall: want to prove

$$\text{IDEAL}^{\mathcal{G}_{\text{rpoRO}}}_{\mathcal{F}_{\text{NIZK}},S,\mathcal{Z}} \approx \text{REAL}^{\mathcal{G}_{\text{rpoRO}}}_{\pi^{\text{SLC}}_{R,\tau},\mathcal{A},\mathcal{Z}}$$









requires localized programming interface

Theorem 3.

Let

- $\Sigma_{R_{VS}}$ be an OR-protocol over R or the CRS relation S,
- $G_{ro_{RO}}$ be the restricted observable global RO,
- SLC be any straight-line compiler, &
- F_{CRS} be the CRS ideal functionality.

Theorem 3.

Let

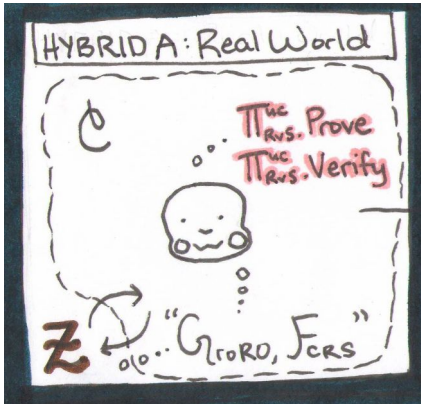
- Σ_{RvS} be an OR-protocol over R or the CRS relation S,
- $\mathcal{G}_{roRO}$ be the restricted observable global RO,
- SLC be any straight-line compiler, &
- $\mathcal{F}_{CRS}$ be the CRS ideal functionality.

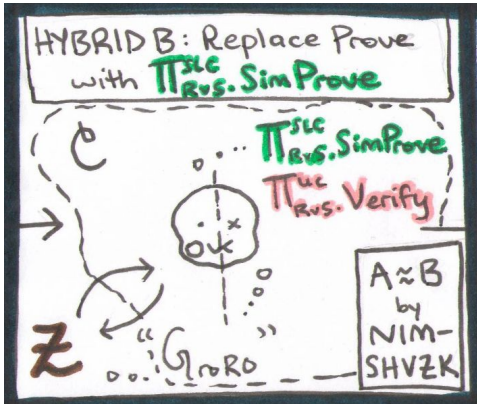
Then the tuple of algorithms Π^{UC}_{RvS} obtained by running Σ_{RvS} through SLC UC-realizes $\mathcal{F}_{NIZK}$ in the $\mathcal{G}_{roRO}$-$\mathcal{F}_{CRS}$-hybrid model.

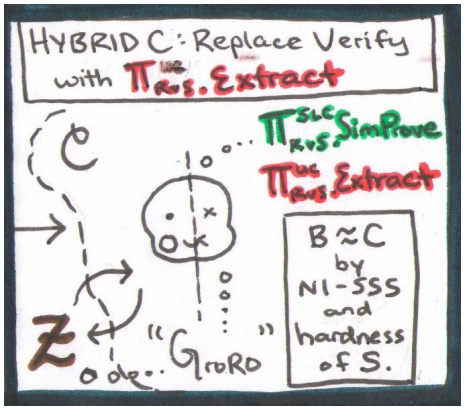
Theorem 3 (Proof).

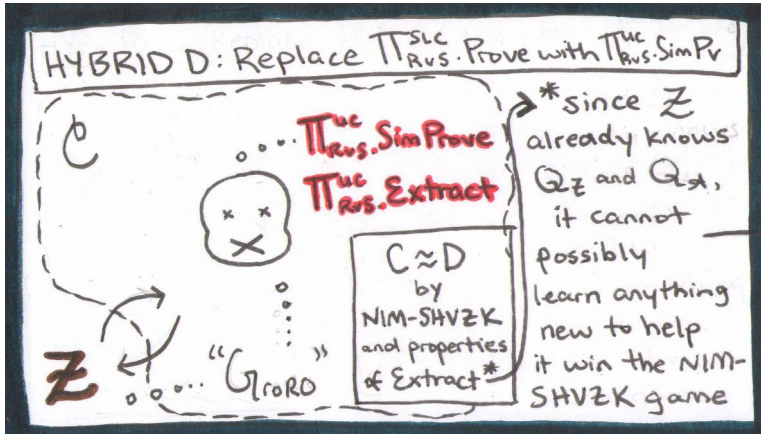
Recall: want to prove

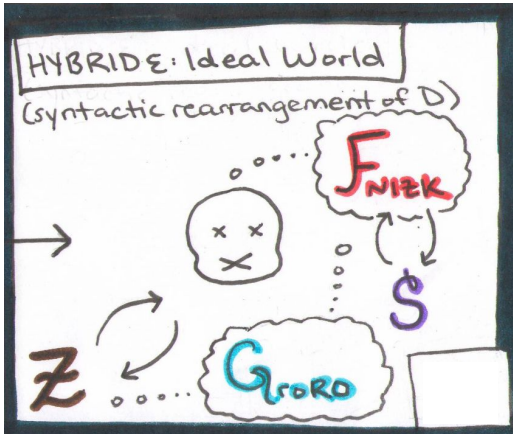
$$\text{IDEAL}^{G_{\text{ROM}}}_{F_{\text{NIZK}}, S, \mathcal{Z}} \approx \text{REAL}^{G_{\text{ROM}}, F_{\text{CRS}}}_{\Pi^{\text{uc}}_{\text{RWS}}, \mathcal{A}, \mathcal{Z}}$$











Thank you for listening!

References

1. Ben Adida. Helios: Web-based open-audit voting. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium*, pages 335–348, 2008.
2. Christian Badertscher, Ran Canetti, Julia Hesse, Björn Tackmann, and Vassilis Zikas. Universal composition with global subroutines: Capturing global setup within plain uc. In *Theory of Cryptography Conference*, pages 1–30. Springer, 2020.
3. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
4. Jan Camenisch and Ivan Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *Conference on the Theory and Application of Cryptology and Information Security*, pages 331–345. Springer, 2000.
5. Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. The wonderful world of global random oracles. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 280–312. Springer, 2018.
6. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact E-cash. In Ronald Cramer, editor, *Advances in Cryptology — Eurocrypt 2005*, volume 3494, pages 302–321. Springer, 2005.
7. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045, pages 93–118. Springer Verlag, 2001.
8. Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In *CRYPTO 1997*, pages 410–424. Springer, 1997.
9. Ran Canetti, Abhishek Jain, and Alessandra Scafuro. Practical uc security with a global random oracle. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 597–608, 2014.
10. Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. Riposte: An anonymous messaging system handling millions of users. In *2015 IEEE Symposium on Security and Privacy*, pages 321–338. IEEE, 2015.
11. Elizabeth Crites, Chelsea Komlo, and Mary Maller. How to prove schnorr assuming schnorr: Security of multi-and threshold signatures. *ePrint Archive*, 2021.
12. Ivan Damgård. On σ-protocols. University of Aarhus, Department of Computer Science, 2002.
13. Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven, and Igors Stepanovs. On the security of two-round multi-signatures. In *2019 IEEE Symposium on Security and Privacy*, pages 1084–1101. IEEE, 2019.
14. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the theory and application of cryptographic techniques*, pages 186–194. Springer, 1986.
15. Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 345–375. Springer, 2019.
16. Yashvanth Kondi and abhi shelat. Improved straight-line extraction in the random oracle model with applications to signature aggregation. *ePrint Archive*, 2022.
17. Somnath Panja and Bimal Kumar Roy. A secure end-to-end verifiable e-voting system using zero knowledge based blockchain. *ePrint*, 2018.
18. Torben Pryds Pedersen. In *CRYPTO '92*.
19. Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4(3):161–174, 1991.
20. Douglas Wikström. A commitment-consistent proof of a shuffle. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP*, pages 407–421. Springer, 2009.