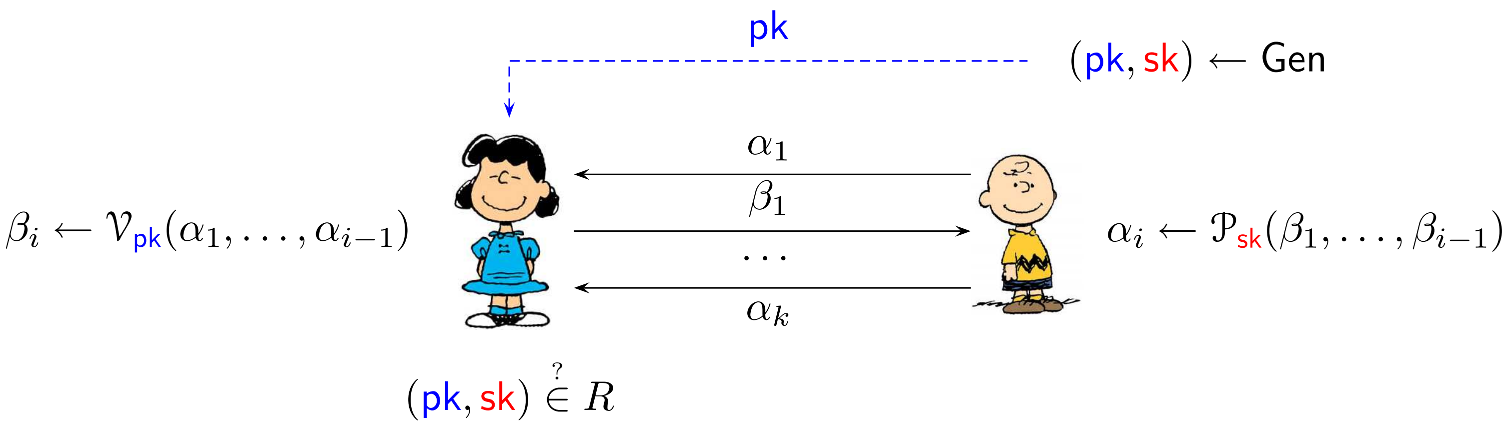
MTAT.07.003 Cryptology II

Zero-knowledge Proofs

Sven Laur
University of Tartu

Formal Syntax

Zero-knowledge proofs



In many settings, some system-wide or otherwise important parameters pk are generated by potentially malicious participants.

- ▷ Zero-knowledge proofs guarantee that the parameters pk are correctly generated without leaking any extra information.
- ▷ Often, public parameters pk are generated together with auxiliary secret information sk that is essential for the zero-knowledge proof.
- ▷ The secret auxiliary information sk is known as a *witness* of pk.

A few interesting statements

<u>An integer n is a RSA modulus</u>:

- ▷ A witness is a pair of primes (p, q) such that $n = p \cdot q$.
- ▷ The relation is defined as follows $(n, p, q) \in R \Leftrightarrow n = p \cdot q \wedge p, q \in \mathbb{P}$

<u>A prover has a secret key sk that corresponds to a public key pk</u>:

- ▷ A witness is a secret key sk such that $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Gen}$.
- ▷ More formally $(\mathsf{pk}, \mathsf{sk}) \in R \Leftrightarrow \forall m \in \mathcal{M} : \mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m)) = m$.

<u>A ciphertext c is an encryption of m wrt the public key pk</u>:

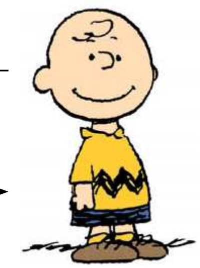
- ▷ A witness is a randomness $r \in \mathcal{R}$ such that $\mathsf{Enc}_{\mathsf{pk}}(m; r) = c$.
- ▷ The relation is defined as follows $(\mathsf{pk}, c, m, r) \in R \Leftrightarrow \mathsf{Enc}_{\mathsf{pk}}(m; r) = c$.

Two flavours of zero knowledge

An ideal implementation of a zero-knowledge proof

pk  $\xrightarrow{\text{pk}}$ / $\xleftarrow{y}$

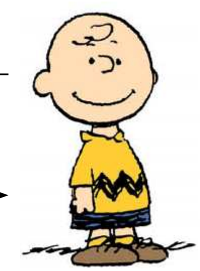
If $\exists$sk : (pk, sk) $\in R$
| $y \leftarrow 1$
Else
| $y \leftarrow 0$

$\xleftarrow{\text{pk}}$ / $\xrightarrow{\perp}$ pk 

An ideal implementation of a zero-knowledge proof of knowledge

pk  $\xrightarrow{\text{pk}}$ / $\xleftarrow{y}$

If (pk, sk) $\in R$
| $y \leftarrow 1$
Else
| $y \leftarrow 0$

$\xleftarrow{\text{pk, sk}}$ / $\xrightarrow{\perp}$ pk, sk 

Formal security requirements

Completeness. A zero-knowledge proof is *perfectly complete* if all runs between honest prover and honest verifier are accepting. A zero knowledge protocol is ε_1-*incomplete* if for all $(\mathsf{pk}, \mathsf{sk}) \in R$ the interaction between honest prover and honest verifier fails with probability at most ε_1.

Soundness. A zero-knowledge proof is ε_2-*unsound* if the probability that an honest verifier accepts an incorrect input pk with probability at most ε_2. An input pk is incorrect if $(\mathsf{pk}, \mathsf{sk}) \notin R$ for all possible witnesses sk.

Zero-knowledge property. A zero-knowledge proof is $(t_{\mathrm{re}}, t_{\mathrm{id}}, \varepsilon_3)$-*private* if for any t_{re}-time verifying strategy $\mathcal{V}_*$ there exists a t_{id}-time algorithm $\mathcal{V}_\circ$ that does not interact with the prover and the corresponding output distributions are statistically ε_3-close.

A Simple Example

Quadratic residuosity

$v = s^2$ $\qquad\qquad\qquad\qquad$ $s \in \mathbb{Z}_n^*$

$\beta \xleftarrow[u]{} \{0,1\}$

$$\xleftarrow{\alpha = r^2}$$

$$\xrightarrow{\beta}$$

$$\xleftarrow{\gamma = r s^\beta}$$

$r \xleftarrow[u]{} \mathbb{Z}_n^*$

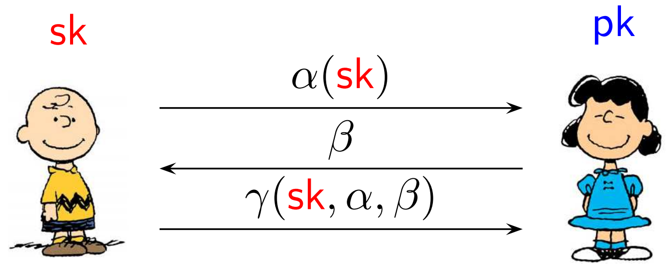
Halt if $\gamma \notin \mathbb{Z}_n^*$

$\gamma^2 = r^2 s^{2\beta} \stackrel{?}{=} \alpha v^\beta$

The modified Fiat-Shamir protocol is also secure against malicious verifiers.

- ▷ If we guess the challenge bit β then we can create α such that the transcript corresponds to the real world execution.
- ▷ Random guessing leads to the correct answer with probability $\frac{1}{2}$.
- ▷ By rewinding we can decrease the failure probability. The failure probability decreases exponentially w.r.t. maximal number of rewindings.

Simulation principle

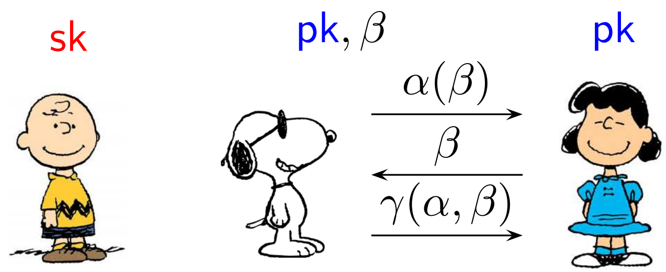


Even if Lucy is *honest*

- she might learn something about the secret sk.

since

- messages α and γ depend on the secret sk.



As Lucy is *malicious* the value of β is not known by her before the protocol and Snoopy must guess β to simulate the other messages.

Lucy should not be able to distinguish between these two experiments.

Simulation as rejection sampling

- As the Fiat-Shamir protocol is a sigma protocol, we can construct protocol transcripts $(\alpha_\circ, \beta_\circ, \gamma_\circ) \leftarrow \mathsf{Sim}_{\mathsf{Fiat\text{-}Shamir}}$ for honest verifier.
- Note that $\alpha_\circ$ has the same distribution than α in the real protocol run.
- Now consider a modified prover $\mathcal{P}_*$ that
  - generates $(\alpha_\circ, \beta_\circ, \gamma_\circ) \leftarrow \mathsf{Sim}$ and sends $\alpha_\circ$ to the verifier,
  - given a challenge β computes the correct reply γ,
  - outputs Sim-Success if $\beta_\circ = \beta$.

Important observations. Let $\mathcal{D}_\circ$ denote the distribution of the outputs of a verifier $\mathcal{V}_*$ which satisfy the condition $\mathcal{P}_*$ outputs Sim-Success. Then the distribution $\mathcal{D}_0$ coincides with the distribution of all outputs of $\mathcal{V}_*$.

- For each reply β, the condition $\beta = \beta_\circ$ holds with probability $\frac{1}{2}$.
- The distribution $\mathcal{D}_\circ$ is easily simulatable.

The complete simulator construction

$$
\begin{array}{l}
\mathcal{V}_\circ \\
\left\lfloor \begin{array}{l}
\text{For } i \in \{1, \ldots, k\} \text{ do} \\
\left\lfloor \begin{array}{l}
(\alpha_\circ, \beta_\circ, \gamma_\circ) \leftarrow \mathsf{Sim}_{\mathsf{Fiat\text{-}Shamir}} \\
\beta \leftarrow \mathcal{V}_*(\alpha_\circ) \\
\text{if } \beta = \beta_\circ \text{ then } \mathbf{return}\ \mathcal{V}_*(\gamma_\circ)
\end{array} \right. \\
\mathbf{return} \text{ failure}
\end{array} \right.
\end{array}
$$

By the construction the output distribution of $\mathcal{V}_\circ$ is

$$(1 - 2^{-k})\mathcal{D}_\circ + 2^{-k}\mathsf{failure} \equiv (1 - 2^{-k})\mathcal{D} + 2^{-k}\mathsf{failure}$$

and thus the statistical distance between outputs of $\mathcal{V}_*$ and $\mathcal{V}_\circ$ is 2^{-k}.

The corresponding security guarantees

Theorem. The modified Fiat-Shamir protocol is a zero-knowledge proof with the following properties:

- the protocol is perfectly complete;
- the protocol is $\frac{1}{2}$-unsound;
- for any k and t_{re} the protocol is $(t_{\mathrm{re}}, k \cdot t_{\mathrm{re}}, 2^{-k})$-private.

Further remarks

- Sequential composition of ℓ protocol instances decreases soundness error to $2^{-\ell}$. The compound protocol becomes $(t_{\mathrm{re}}, k \cdot \ell \cdot t_{\mathrm{re}}, \ell \cdot 2^{-k})$-private.
- The same proof is valid for all sigma protocols, where the challenge β is only one bit long. For longer challenges β, the success probability decreases with an exponential rate and simulation becomes inefficient.

Zero-Knowledge Proofs
and
Knowledge Extraction

Challenge-response paradigm

$v \in \mathsf{QNR}(n)$ $\qquad\qquad\qquad\qquad$ $n = p \cdot q$

$\beta \xleftarrow[u]{} \{0,1\}$

$r \xleftarrow[u]{} \mathbb{Z}_n^*$

$$\xrightarrow{\alpha = r^2 v^\beta}$$

$$\xleftarrow{\overline{\beta}}$$

$\overline{\beta} \leftarrow \mathsf{IsNQR}_{p,q}(\alpha)$

$\beta \stackrel{?}{=} \overline{\beta}$

For semi-honest provers it is trivial to simulate the interaction, since the verifier knows the expected answer $\beta = \overline{\beta}$. To provide security against malicious verifiers $\mathcal{V}_*$, we must assure that we can extract β from $\mathcal{V}_*$:

- ▷ Verifier must prove that she knows (r, β) such that $c = r^2 v^\beta$
- ▷ The corresponding proof of knowledge does not have be zero knowledge proof as long as it does not decrease soundness.

Classical construction

$v \in \mathsf{QNR}(n)$ $\qquad$ $n = p \cdot q$

$\beta \xleftarrow{u} \{0,1\}$

$r \xleftarrow{u} \mathbb{Z}_n^*$

$\alpha \leftarrow r^2 v^\beta$

$\xrightarrow{\alpha}$

$\mathrm{POK}_\beta[\alpha = r^2 v^\beta]$

$\xleftarrow{d}$

$\overline{\beta} \leftarrow \mathsf{IsNQR}_{p,q}(\alpha)$

Halt if $\mathrm{POK}_\beta[\cdot]$ fails

$\beta \stackrel{?}{=} \overline{\beta}$

We can use proofs of knowledge to assure that the verifier knows the end result β. The proof must perfectly hide the witness β.

- If $v \in$ QR then α is independent from β and malicious prover can infer information about β only through the proof of knowledge.
- Hence, we are actually interested in *witness hiding* property of the proof of knowledge, i.e., the proof transcripts should coincide for both β values.

Witness hiding provides soundness

We have to construct a sigma protocol for the following statement

$$\mathrm{POK}_\beta\left[\exists r : \alpha = r^2 v^\beta\right] \equiv \mathrm{POK}_r\left[r^2 = \alpha\right] \vee \mathrm{POK}_r\left[r^2 = \alpha v^{-1}\right]$$

Both sub-proofs separately can be implemented through the modified Fiat-Shamir protocol. To achieve witness hiding we just use OR-composition.

- ▷ For fixed challenge β, the sub-challenge pairs are uniformly chosen from a set $\mathcal{B} = \{(\beta_1, \beta_2) : \beta_1 + \beta_2 = \beta\}$.
- ▷ Hence, the interactions where $\mathcal{V}$ proves $\mathrm{POK}_r\left[r^2 = \alpha\right]$ and simulates $\mathrm{POK}_r\left[r^2 = \alpha v^{-1}\right]$ are indistinguishable form the interactions where $\mathcal{V}$ proves $\mathrm{POK}_r\left[r^2 = \alpha v^{-1}\right]$ and simulates $\mathrm{POK}_r\left[r^2 = \alpha\right]$.
- ▷ If $v = s^2$ then also $\alpha_0 = r^2$ and $\alpha_1 = r^2 v$ are indistinguishable.

Consequently, a malicious adversary succeeds with probability $\frac{1}{2}$ if $v = s^2$.

Simulator construction

$\mathcal{S}^{\mathcal{V}_*}$

- Choose randomness ω for $\mathcal{V}_*$ and store α.
- Use knowledge extractor to extract β.
- Run $\mathcal{V}_*$ once again.
- if $\mathrm{POK}_\beta\left[\exists r : \alpha = r^2 v^\beta\right]$ fails then
  - Send $\perp$ to $\mathcal{V}$ and output whatever $\mathcal{V}_*$ outputs.
- else
  - Send β to $\mathcal{V}$ and output whatever $\mathcal{V}_*$ outputs.

The simulation fails only if knowledge extraction fails and $\mathrm{POK}_\beta\left[\cdot\right]$ succeeds. With proper parameter choice, we can achieve failure ε in time $\Theta\left(\frac{t_{\mathrm{re}}}{\varepsilon-\kappa}\right)$.

Optimal choice of parameters

Let ε be the desired failure bound and let κ be the knowledge error of the sigma protocol. Now if we set the maximal number of repetitions

$$\ell = \frac{4\left\lceil \log_2(1/\varepsilon)\right\rceil}{\varepsilon - \kappa}$$

in the knowledge extraction algorithm so that the knowledge extraction procedure fails on the set of good coins

$$\Omega_{\mathrm{good}} = \{\omega \in \Omega : \Pr\left[\mathrm{POK}_\beta\left[\cdot\right] = 1|\omega\right] \geq \varepsilon\}$$

with probability less than ε. Consequently, we can estimate

$$\begin{aligned}\Pr\left[\mathsf{Fail}\right] \leq \Pr\left[\omega \notin \Omega_{\mathrm{good}}\right] \cdot \Pr\left[\mathrm{POK}_\beta\left[\cdot\right] = 1|\omega\right] \cdot \Pr\left[\mathsf{ExtrFailure}|\omega\right] \\ + \Pr\left[\omega \in \Omega_{\mathrm{good}}\right] \cdot \Pr\left[\mathrm{POK}_\beta\left[\cdot\right] = 1|\omega\right] \cdot \Pr\left[\mathsf{ExtrFailure}|\omega\right] \leq \varepsilon \enspace .\end{aligned}$$

Soundness through temporal order

$v \in \mathsf{QNR}(n)$		$n = p \cdot q$
$\beta \xleftarrow[u]{} \{0,1\}$ $r \xleftarrow[u]{} \mathbb{Z}_n^*$ $\alpha \leftarrow r^2 v^\beta$	 $\xrightarrow{\alpha}$ $\xleftarrow{c, \mathsf{pk}}$ $\xrightarrow{r, \beta}$ $\xleftarrow{d}$ 	$\mathsf{pk} \leftarrow \mathsf{Gen}$ $\overline{\beta} \leftarrow \mathsf{IsNQR}_{p,q}(\alpha)$ $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\overline{\beta})$ Halt if $\alpha \neq r^2 v^\beta$
$\beta \stackrel{?}{=} \mathsf{Open}_{\mathsf{pk}}(c, d)$		

Let $(\mathsf{Gen}, \mathsf{Com}, \mathsf{Open})$ is a perfectly binding commitment scheme such that the validity of public parameters can be verified (ElGamal encryption).

- Then the perfect binding property assures that the malicious prover $\mathcal{P}_*$ cannot change his reply. Soundness guarantees are preserved.
- A commitment scheme must be $(t_{\mathrm{re}} + t, \kappa)$-hiding for t_{re}-time verifier.
- By rewinding we can find out the correct answer in time $\Theta(\frac{1}{\varepsilon - \kappa})$, where ε is the success probability of malicious verifier $\mathcal{V}_*$.

Simulator construction

$\mathcal{S}^{\mathcal{V}_*}$

$$\left[\begin{array}{l}
\text{Choose randomness } \omega \text{ for } \mathcal{V}_* \text{ and store } \alpha. \\
\text{Use knowledge extractor to extract } \beta. \\
\text{Run } \mathcal{V}_* \text{ once again with } (c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\beta). \\
\text{if } \alpha \neq r^2 v^\beta \text{ then} \\
\quad \left[\text{Send } \perp \text{ to } \mathcal{V} \text{ and output whatever } \mathcal{V}_* \text{ outputs.}\right. \\
\text{else} \\
\quad \left[\text{Send } d \text{ to } \mathcal{V} \text{ and output whatever } \mathcal{V}_* \text{ outputs.}\right.
\end{array}\right.$$

Knowledge-extraction is straightforward. We just provide $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(0)$ and verify whether $\alpha = r^2 v^\beta$. The choice of parameters is analogous.

Further analysis

The output of the simulator is only computationally indistinguishable from the real protocol run, as the commitment is only computationally hiding. Let $\mathcal{A}$ be a t-time adversary that tries to distinguish outputs of $\mathcal{V}_*$ and $\mathcal{S}^{\mathcal{V}_*}$

- ▷ If $\alpha = r^2v^\beta$ and knowledge extraction succeeds, the simulation is perfect.
- ▷ If $\alpha \neq r^2v^\beta$ then from $(t_{\mathrm{re}} + t, \kappa)$-hiding, we get

$$\left|\Pr\left[\mathcal{A} = 1 | \mathcal{V}_*^{\mathcal{P}} \wedge \alpha \neq r^2v^\beta\right] - \Pr\left[\mathcal{A} = 1 | \mathcal{S}^{\mathcal{V}_*} \wedge \alpha \neq r^2v^\beta\right]\right| \leq \kappa \enspace .$$

- ▷ Similarly, $(t_{\mathrm{re}} + t, \kappa)$-hiding assures that

$$\left|\Pr\left[\alpha = r^2v^\beta | \mathcal{V}_*^{\mathcal{P}}\right] - \Pr\left[\alpha \neq r^2v^\beta | \mathcal{V}_* \wedge (c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(0)\right]\right| \leq \kappa \enspace .$$

Hence, the knowledge extractor makes on average $\frac{1}{\varepsilon - \kappa}$ probes.

Strengthening of Σ-protocols

Strengthening with commitments

$$\boxed{\mathrm{ZK}_{x}[(x, w) \in R]}$$

x

$\beta \xleftarrow[u]{} \mathcal{B}$

$(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\beta)$

$\mathsf{Ver}_{x}(\alpha, \beta, \gamma)$

$\xleftarrow{\mathsf{pk}}$

$\xrightarrow{c}$

$\xleftarrow{\alpha}$

$\xrightarrow{d}$

$\xleftarrow{\gamma}$

x, w

$\mathsf{pk} \leftarrow \mathsf{Gen}$

$\alpha \leftarrow \mathcal{P}_{w}$

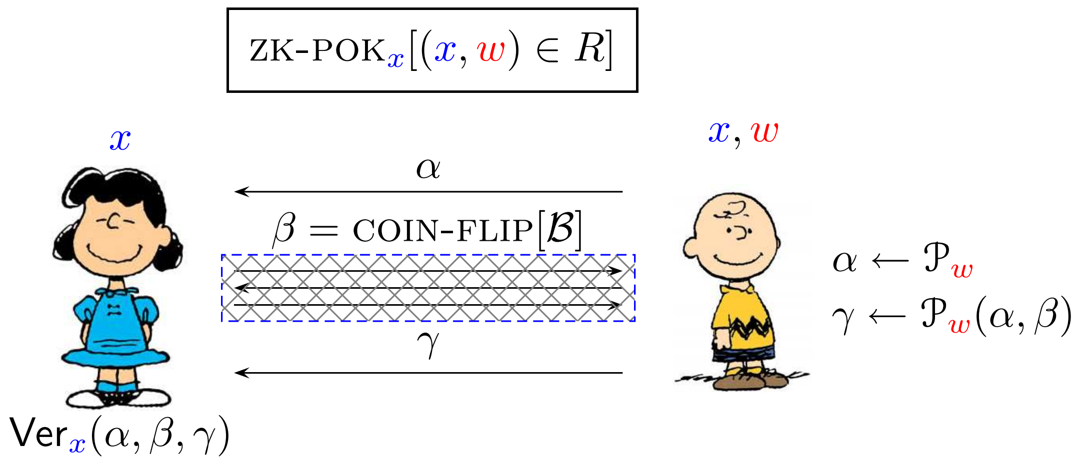
$\beta \leftarrow \mathsf{Open}_{\mathsf{pk}}(c, d)$

$\gamma \leftarrow \mathcal{P}_{w}(\alpha, \beta)$

If the commitment is statistically hiding then the soundness guarantees are preserved. Again, rewinding allows us to extract the value of β.

- If commitment scheme is $((\ell + 1) \cdot t_{\mathrm{re}}, \varepsilon_2)$-binding then commitment can be double opened with probability at most ε_2.
- Hence, we can choose $\ell = \Theta(\frac{1}{\varepsilon_1})$ so that simulation failure is $\varepsilon_1 + \varepsilon_2$.
- The protocol does not have knowledge extraction property any more.

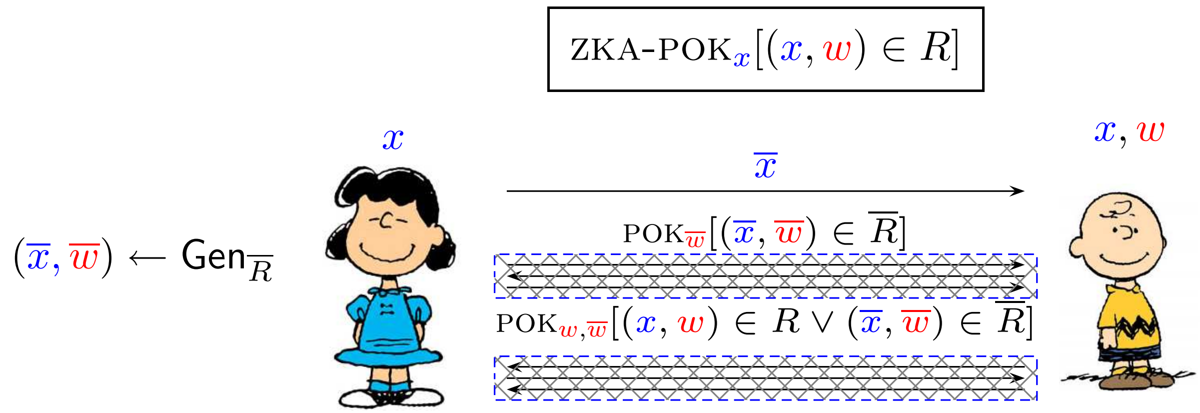
Strengthening with coin-flipping



We can substitute trusted sampling $\beta \xleftarrow[u]{} \mathcal{B}$ with a coin-flipping protocol.

- ▷ To achieve soundness, we need a coin-flipping protocol that is secure against unbounded provers.
- ▷ Statistical indistinguishability is achievable provided that the coin-flipping protocol is secure even if all internal variables become public afterwards.
- ▷ Rewinding takes now place inside the coin-flipping block.

Strengthening with disjunctive proofs



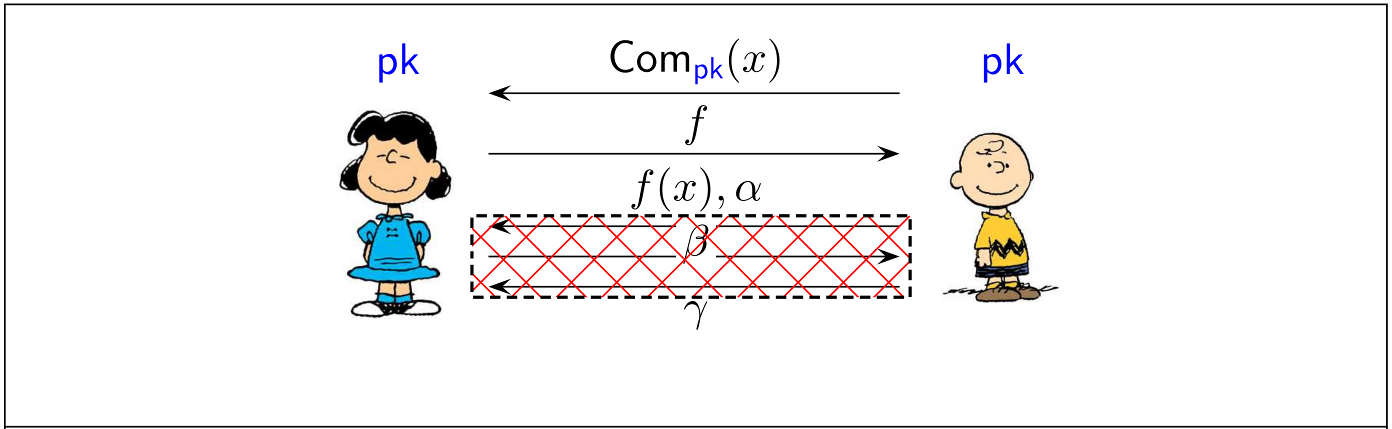
If the relation $\overline{R}$ generated by $\mathsf{Gen}_{\overline{R}}$ is hard, i.e., given $\overline{x}$ it is difficult to find matching $\overline{w}$, then the proof is computationally sound.

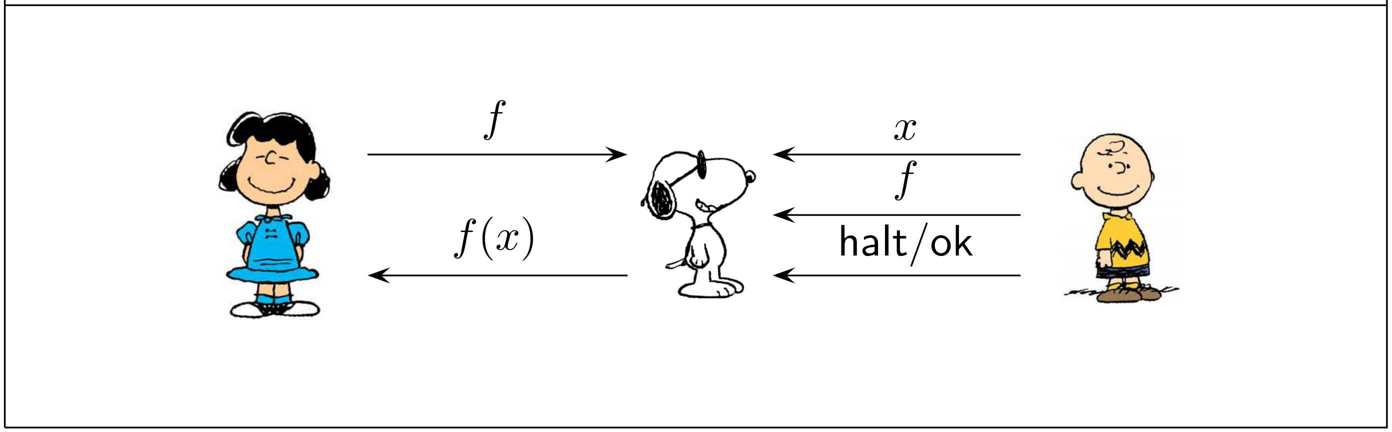
The hardness of $\overline{R}$ also guarantees that the second proof is witness hiding. Thus, we can extract first $\overline{w}$ and use it to by-pass the second proof.

Certified Computations

Malicious case

The concept

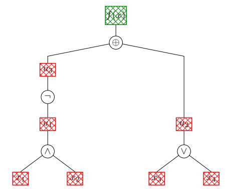




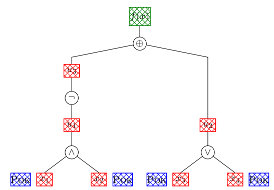
Lucy should learn $f(x)$ and nothing more even if Charlie is malicious.

A quick recap of the semihonest case

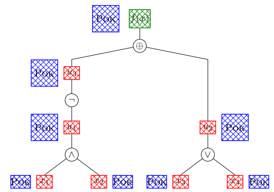
Construct the circuit and commit all values



Construct a sigma protocols for validity of inputs



Construct a sigma protocol for validity of intermediate values



Security against malicious verifiers

We can use several methods to strengthen the protocol.

- ▷ We can restrict challenge space $\mathcal{B}$ to $\{0, 1\}$ and then use sequential composition to achieve reasonable soundness level.
- ▷ We can use commitments to strengthen the sigma protocol.
- ▷ We can use coin-flipping protocol to generate the challenge β.
- ▷ We can use disjunctive proofs to strengthen the sigma protocol.

The resulting construction which is based on a coin-flipping protocol is often referred as $\textsc{Gmw}$-compiler, since it forces semihonest behaviour.