MTAT.07.003 Cryptology II

Sigma Protocols

Sven Laur
University of Tartu

Formal Syntax

Sigma protocols

$\alpha \leftarrow \mathcal{R}$

sk

$$\xrightarrow{\ \alpha\ }$$

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

$$\xrightarrow{\ \gamma = \gamma_{\mathsf{sk}}(\alpha, \beta)\ }$$

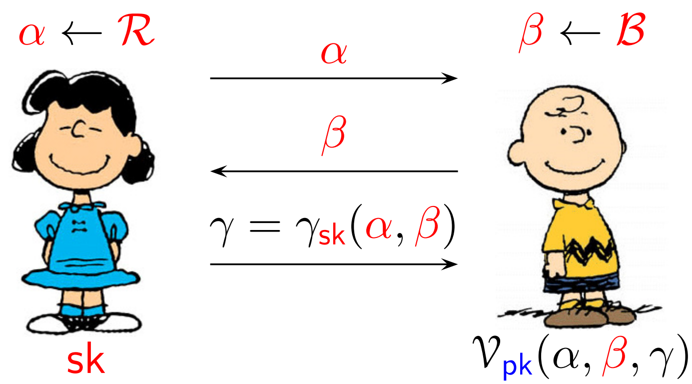
$\beta \leftarrow \mathcal{B}$

$\mathcal{V}_{\mathsf{pk}}(\alpha, \beta, \gamma)$

A *sigma protocol* for an efficiently computable relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is a three move protocol that satisfies the following properties.

- ▷ **Σ-structure.** A prover first sends a commitment, next a verifier sends *varying* challenge, and then the prover must give a consistent response.
- ▷ **Functionality.** The protocol run between an honest prover $\mathcal{P}(\mathsf{sk})$ and verifier $\mathcal{V}(\mathsf{pk})$ is always accepting if $(\mathsf{sk}, \mathsf{pk}) \in R$.

Security properties of sigma protocols



- ▷ **Perfect simulatability.** There exists an efficient *non-rewinding* simulator $\mathcal{S}$ such that the output distribution of a semi-honest verifier $\mathcal{V}_*$ in the real world and the output distribution of $\mathcal{S}^{\mathcal{V}_*}$ in the ideal world coincide.
- ▷ **Special soundness.** There exists an efficient extraction algorithm Extr that, given two accepting protocol runs $(\alpha, \beta_0, \gamma_0)$ and $(\alpha, \beta_1, \gamma_1)$ with $\beta_0 \neq \beta_1$ that correspond to pk, outputs sk_* such that $(\mathsf{sk}_*, \mathsf{pk}) \in R$

Soundness of Sigma Protocols

Soundness in the standalone model

Main Theorem. Denote $\kappa = |\mathcal{B}|^{-1}$. Now if a t-time prover $\mathcal{P}_*$ succeeds in the sigma protocol with probability at least $\varepsilon > \kappa$, there exists a knowledge-extraction algorithm $\mathcal{K}^{\mathcal{P}_*}$ that always recovers a secret sk_* corresponding to pk and the expected running-time of $\mathcal{K}^{\mathcal{P}_*}$ is

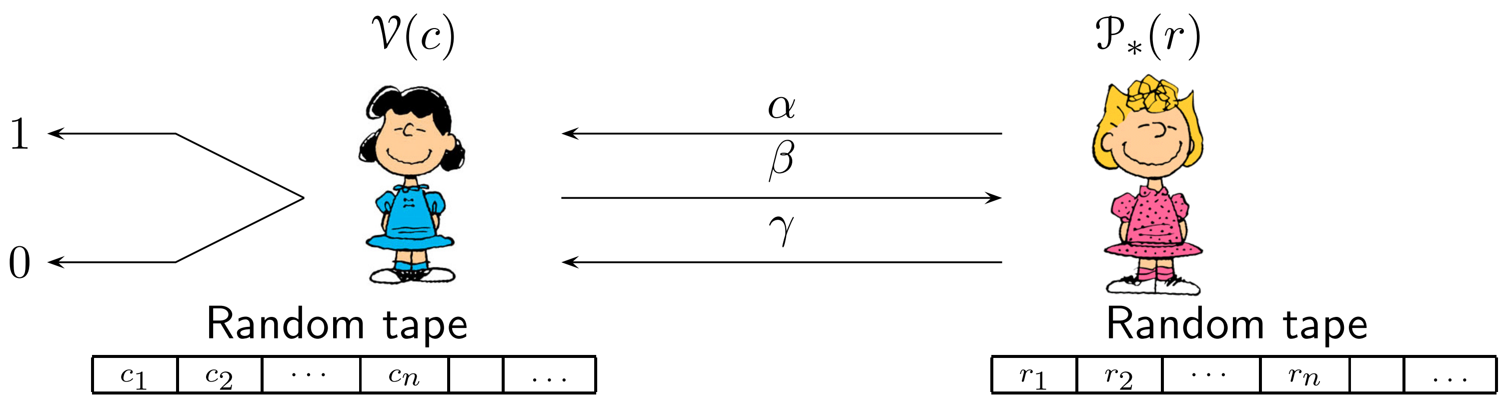
$$c_1 \cdot \frac{2t}{\varepsilon - \kappa} + c_2$$

for some small constants $c_1, c_2 \in \mathbb{R}$.

Remark.

- ▷ The coefficient c_1 depends on the total complexity of the protocol.
- ▷ The coefficient c_2 depends on the complexity of the Extr algorithm.

The corresponding matrix game



Let $A(r, c)$ be the output of the honest verifier $\mathcal{V}(c)$ that interacts with a potentially malicious prover $\mathcal{P}_*(r)$.

- ▷ Then all matrix elements in the same row $A(r, \cdot)$ lead to same α value.
- ▷ To extract the secret key sk, we must find two ones in the same row.
- ▷ We can compute the entries of the matrix on the fly.

Classical algorithm

Task: Find two ones in a same row.

<u>Rewind</u>:

1. Probe random entries $A(r, c)$ until $A(r, c) = 1$.
2. Store the matrix location (r, c).
3. Probe random entries $A(r, \overline{c})$ in the same row until $A(r, \overline{c}) = 1$.
4. Output the location triple $(r, c, \overline{c})$.

<u>Rewind-Exp</u>:

1. Repeat the procedure Rewind until $c \neq \overline{c}$.
2. Use the extraction algorithm Extr to extract sk.

Average-case running time

Theorem. If a $m \times n$ zero-one matrix A contains ε-fraction of nonzero entries, then the Rewind and Rewind-Exp algorithm make on average

$$\mathbf{E}[\text{probes}|\text{Rewind}] = \frac{2}{\varepsilon}$$

$$\mathbf{E}[\text{probes}|\text{Rewind-Exp}] = \frac{2}{\varepsilon - \kappa}$$

probes where $\kappa = \frac{1}{n}$ is a *knowledge error*.

Average case complexity I

Assume that the matrix contains ε-fraction of nonzero elements, i.e., $\mathcal{P}_*$ convinces $\mathcal{V}$ with probability ε. Then on average we make

$$\mathbf{E}\left[\mathsf{probes}_1\right] = \varepsilon + 2(1-\varepsilon)\varepsilon + 3(1-\varepsilon)^2\varepsilon + \cdots = \tfrac{1}{\varepsilon}$$

matrix probes to find the first non-zero entry. Analogously, we make

$$\mathbf{E}\left[\mathsf{probes}_2|r\right] = \tfrac{1}{\varepsilon_r}$$

probes to find the second non-zero entry. Also, note that

$$\mathbf{E}[\mathsf{probes}_2] = \sum_r \Pr\left[r\right] \cdot \mathbf{E}[\mathsf{probes}_2|r] = \sum_r \frac{\varepsilon_r}{\sum_{r'} \varepsilon_{r'}} \cdot \frac{1}{\varepsilon_r} = \frac{1}{\varepsilon}\ ,$$

where ε_r is the fraction of non-zero entries in the r^{th} row.

Average case complexity II

As a result we obtain that the Rewind algorithm does on average

$$\mathbf{E}[\text{probes}] = \tfrac{2}{\varepsilon}$$

probes. Since the Rewind algorithm fails with probability

$$\Pr\left[\text{failure}\right] = \sum_{r} \Pr\left[c = \overline{c} | \text{halting}\right] \leq \frac{\kappa}{\varepsilon}$$

we make on average

$$\mathbf{E}[\text{probes}^{*}] = \frac{1}{\Pr\left[\text{success}\right]} \cdot \mathbf{E}[\text{probes}] \leq \frac{\varepsilon}{\varepsilon - \kappa} \cdot \frac{2}{\varepsilon} = \frac{2}{\varepsilon - \kappa}$$

probes.

Soundness of sequential compositions

Main Theorem. Consider a setting where a prover $\mathcal{P}_*$ and honest verifier $\mathcal{V}$ sequentially execute the same sigma protocol ℓ times. Denote $\kappa = |\mathcal{B}|^{-1}$. Also let $\mathcal{P}_*$ be successful if $\mathcal{P}_*$ succeeds at least in one protocol instance. Now if a t-time prover $\mathcal{P}_*$ succeeds with probability at least $\varepsilon > \ell\kappa$, there exists a knowledge-extraction algorithm $\mathcal{K}^{\mathcal{P}_*}$ that always recovers a secret sk_* corresponding to pk and the expected running-time of $\mathcal{K}^{\mathcal{P}_*}$ is

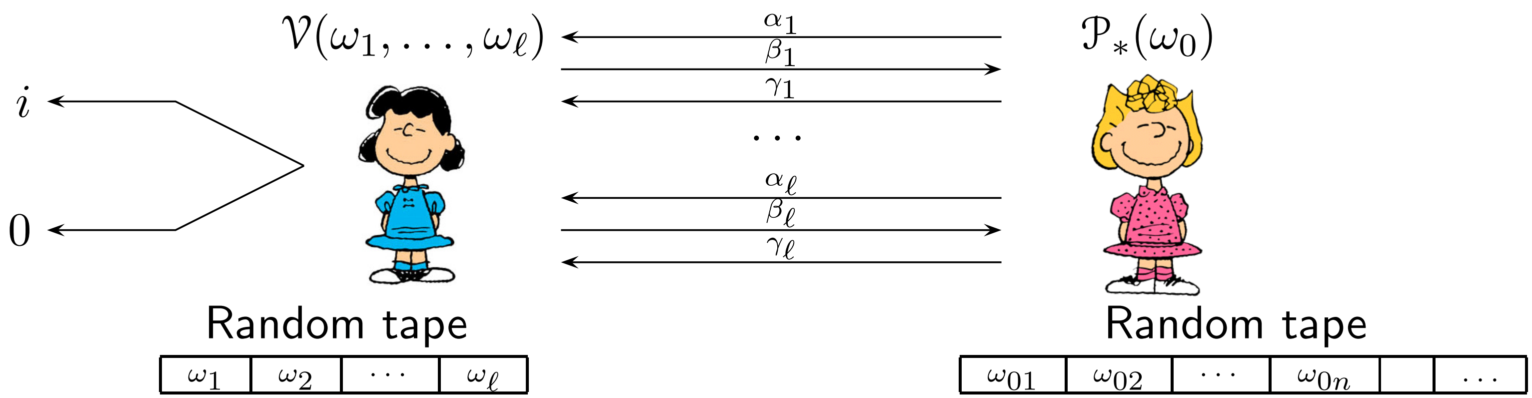
$$c_1 \cdot \frac{\ell + 1}{\varepsilon - \ell\kappa} + c_2$$

for some small constants $c_1, c_2 \in \mathbb{R}$.

Remark.

- ▷ The coefficient c_1 depends on the total complexity of the protocol.
- ▷ The coefficient c_2 depends on the complexity of the Extr algorithm.

The corresponding matrix game



Let $A(\omega_0, \omega_1, \ldots, \omega_\ell)$ denote the index of the first successful protocol between honest verifier $\mathcal{V}(\omega_1, \ldots, \omega_\ell)$ and a prover $\mathcal{P}_*(\omega_0)$.

- ▷ Then a randomness prefix $\omega_0, \ldots, \omega_{i-1}$ leads to the same α_i value.
- ▷ To extract the secret key sk, we must find two i-s with the same prefix.
- ▷ We can compute the entries of the array on the fly.

Classical algorithm

Rewind:

1. Probe random entries $A(\boldsymbol{\omega})$ until $A(\boldsymbol{\omega}) \neq 0$.
2. Store the matrix location $\boldsymbol{\omega}$ and the rewinding point $i \leftarrow A(\boldsymbol{\omega})$.
3. Probe random entries $A(\overline{\boldsymbol{\omega}})$ with the prefix $\omega_0, \ldots, \omega_{i-1}$ until $A(\overline{\boldsymbol{\omega}}) = i$.
4. Output the location tuple $(\boldsymbol{\omega}, \overline{\boldsymbol{\omega}})$.

Rewind-Exp:

1. Repeat the procedure Rewind until $\omega_i \neq \overline{\omega}_i$.
2. Use the extraction algorithm Extr to extract sk.

Average-case running time

Theorem. If a array $A(\boldsymbol{\omega})$ with entries in $\{0, \ldots, \ell\}$ contains ε-fraction of nonzero entries, then Rewind and Rewind-Exp make on average

$$\mathbf{E}[\text{probes}|\text{Rewind}] = \frac{\ell + 1}{\varepsilon}$$

$$\mathbf{E}[\text{probes}|\text{Rewind-Exp}] = \frac{\ell + 1}{\varepsilon - \kappa}$$

probes where the *knowledge error*

$$\kappa = \sum_{i=1}^{\ell} \Pr\left[\omega_i = \overline{\omega}_i\right] \quad .$$

Average case complexity I

Assume that $\mathcal{A}$ succeeds with probability ε. Then the results proved for the zero-one matrix with fixed dimensions imply

$$\mathbf{E}[\mathsf{probes}_1] = \tfrac{1}{\varepsilon} \qquad \text{and} \qquad \mathbf{E}[\mathsf{probes}_2 | A(\boldsymbol{\omega}) = i] = \tfrac{1}{\varepsilon_i}$$

where ε_i is the fraction of entries labelled with i. Thus

$$\mathbf{E}[\mathsf{probes}_2] = \sum_{i=1}^{\ell} \Pr\left[A(\boldsymbol{\omega}) = i\right] \cdot \mathbf{E}[\mathsf{probes}_2 | A(\boldsymbol{\omega}) = i]$$

$$\mathbf{E}[\mathsf{probes}_2] = \sum_{i=1}^{\ell} \frac{\varepsilon_i}{\varepsilon} \cdot \frac{1}{\varepsilon_i} = \frac{\ell}{\varepsilon} \enspace .$$

Average case complexity II

Consequently, the Rewind algorithm does on average

$$\mathbf{E}[\mathsf{probes}] = \frac{\ell + 1}{\varepsilon}$$

probes. Since the Rewind algorithm fails with probability

$$\Pr\left[\mathsf{failure}\right] = \sum_{i=1}^{\ell} \Pr\left[A(\boldsymbol{\omega}) = i\right] \Pr\left[\omega_i = \overline{\omega}_i | \mathsf{halting}\right] \leq \frac{\kappa_1 + \cdots + \kappa_\ell}{\varepsilon}$$

we make on average

$$\mathbf{E}[\mathsf{probes}^*] = \frac{1}{\Pr\left[\mathsf{success}\right]} \cdot \mathbf{E}[\mathsf{probes}] \leq \frac{\varepsilon}{\varepsilon - \kappa} \cdot \frac{\ell + 1}{\varepsilon} = \frac{\ell + 1}{\varepsilon - \kappa} \ .$$

Various Parallel Compositions

Conjunctive proofs

pk $\qquad\qquad$ $\mathsf{sk}_1, \mathsf{sk}_2$

$\alpha_1 \leftarrow \mathcal{P}_1(\mathsf{sk}_1)$

$\alpha_2 \leftarrow \mathcal{P}_2(\mathsf{sk}_2)$

$\xleftarrow{\alpha_1, \alpha_2}$

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

$\xrightarrow{\beta}$

$\gamma_1 \leftarrow \mathcal{P}_1(\mathsf{sk}_1, \beta)$

$\gamma_2 \leftarrow \mathcal{P}_2(\mathsf{sk}_2, \beta)$

$\xleftarrow{\gamma_1, \gamma_2}$

Halt if $\mathcal{V}_1(\mathsf{pk}, \alpha_1, \beta, \gamma_1) = 0$

Halt if $\mathcal{V}_2(\mathsf{pk}, \alpha_2, \beta, \gamma_2) = 0$

If we run two sigma protocols for different relations R_1 and R_2 in parallel, we get a sigma protocol for new relation $R_1 \wedge R_2$

$$(\mathsf{sk}_1, \mathsf{sk}_2, \mathsf{pk}) \in R_1 \wedge R_2 \quad \Leftrightarrow \quad (\mathsf{sk}_1, \mathsf{pk}) \in R_1 \wedge (\mathsf{sk}_2, \mathsf{pk}) \in R_2 \ .$$

provided that both sigma protocols have the same challenge space $\mathcal{B}$ and it a *perfect simulation of transcripts* $(\alpha_i, \beta, \gamma_i)$ is *efficient* for any β.

The corresponding proof

Perfect simulatability. Let $\mathcal{S}_1$ be a canonical simulator for $\mathcal{V}_1$. Now if $\mathcal{S}_1$ outputs a properly distributed triple $(\alpha_1, \beta, \gamma_1)$, then we can augment this with properly distributed $(\alpha_2, \beta, \gamma_2)$ and thus we get a properly distributed protocol transcript $(\alpha_1, \alpha_2, \beta, \gamma_1, \gamma_2)$.

Special soundness. Given two accepting transcripts

$$(\alpha_1, \alpha_2, \beta^0, \gamma_1^0, \gamma_2^0), (\alpha_1, \alpha_2, \beta^1, \gamma_1^1, \gamma_2^1), \quad \text{with} \quad \beta^0 \neq \beta^1 \ ,$$

we can decompose them into original colliding transcripts

$$\begin{aligned} &(\alpha_1, \beta^0, \gamma_1^0), (\alpha_1, \beta^1, \gamma_1^1), \qquad \beta^0 \neq \beta^1 \ , \\ &(\alpha_2, \beta^0, \gamma_2^0), (\alpha_2, \beta^1, \gamma_2^1), \qquad \beta^0 \neq \beta^1 \ . \end{aligned}$$

Disjunctive proofs

pk

$\mathsf{sk}_1 \vee \mathsf{sk}_2$

$\alpha_i \leftarrow \mathcal{P}_i(\mathsf{sk}_i)$

$(\alpha_j, \beta_j, \gamma_j) \leftarrow \mathcal{S}_j(\mathsf{pk})$

$\xleftarrow{\alpha_1, \alpha_2}$

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

$\xrightarrow{\beta}$

$\beta_i \leftarrow \beta - \beta_j$

$\gamma_i \leftarrow \mathcal{P}_i(\mathsf{sk}_i, \beta_i)$

$\xleftarrow{\beta_1, \beta_2, \gamma_1, \gamma_2}$

Halt if $\beta_1 + \beta_2 \neq \beta$

Halt if $\mathcal{V}_1(\mathsf{pk}, \alpha_1, \beta_1, \gamma_1) = 0$

Halt if $\mathcal{V}_2(\mathsf{pk}, \alpha_2, \beta_2, \gamma_2) = 0$

Assume that we have two sigma protocols for relations R_1 and R_2 such that the challenge is chosen uniformly from a commutative group $(\mathcal{B}; +)$.

Then a prover can use a simulator $\mathcal{S}_j$ to create the transcript for missing secret sk_j and then create response using the known secret sk_i.

Disjunctive proofs

Verifier ($\color{blue}{\mathsf{pk}}$)	Messages	Prover ($\color{red}{\mathsf{sk}_1} \vee \color{red}{\mathsf{sk}_2}$)
		$\alpha_i \leftarrow \mathcal{P}_i(\color{red}{\mathsf{sk}_i})$
	$\xleftarrow{\alpha_1, \alpha_2}$	$(\alpha_j, \beta_j, \gamma_j) \leftarrow \mathcal{S}_j(\color{blue}{\mathsf{pk}})$
$\beta \xleftarrow[u]{} \mathcal{B}$	$\xrightarrow{\beta}$	$\beta_i \leftarrow \beta - \beta_j$
	$\xleftarrow{\beta_1, \beta_2, \gamma_1, \gamma_2}$	$\gamma_i \leftarrow \mathcal{P}_i(\color{red}{\mathsf{sk}_i}, \beta_i)$

Halt if $\beta_1 + \beta_2 \neq \beta$
Halt if $\mathcal{V}_1(\color{blue}{\mathsf{pk}}, \alpha_1, \beta_1, \gamma_1) = 0$
Halt if $\mathcal{V}_2(\color{blue}{\mathsf{pk}}, \alpha_2, \beta_2, \gamma_2) = 0$

As a result, we get a sigma protocol for new relation $R_1 \vee R_2$

$$(\mathsf{sk}_1, \mathsf{sk}_2, \mathsf{pk}) \in R_1 \vee R_2 \quad \Leftrightarrow \quad (\mathsf{sk}_1, \mathsf{pk}) \in R_1 \vee (\mathsf{sk}_2, \mathsf{pk}) \in R_2 \enspace .$$

The corresponding proof

Perfect simulatability. Note that β_1 and β_2 are independent and have a uniform distribution over $\mathcal{B}$. Consequently, we can run the canonical simulators $\mathcal{S}_1$ and $\mathcal{S}_2$ be for $\mathcal{V}_1$ and $\mathcal{V}_2$ in parallel to create the properly distributed transcript $(\alpha_1, \alpha_2, \beta_1 + \beta_2, \beta_1, \beta_2, \gamma_1, \gamma_2)$.

Special soundness. Given two transcripts

$$(\alpha_1, \alpha_2, \beta_1^0 + \beta_2^0, \beta_1^0, \beta_2^0, \gamma_1^0, \gamma_2^0), (\alpha_1, \alpha_2, \beta_1^1 + \beta_2^1, \beta_1^1, \beta_2^1, \gamma_1^1, \gamma_2^1)$$

such that $\beta_1^0 + \beta_2^0 \neq \beta_1^1 + \beta_2^1$, we can extract a colliding sub-transcript

$$\begin{cases} (\alpha_1, \beta_1^0, \gamma_1^0), (\alpha_1, \beta_1^1, \gamma_1^1), & \text{if } \beta_1^0 \neq \beta_1^1 \enspace , \\ (\alpha_2, \beta_2^0, \gamma_2^0), (\alpha_2, \beta_2^1, \gamma_2^1), & \text{if } \beta_2^0 \neq \beta_2^1 \enspace . \end{cases}$$

Parallel composition

pk  $\mathsf{sk}_1, \mathsf{sk}_2$ 

		$\alpha_1 \leftarrow \mathcal{P}_1(\mathsf{sk}_1)$
	$\xleftarrow{\quad \alpha_1, \alpha_2 \quad}$	$\alpha_2 \leftarrow \mathcal{P}_2(\mathsf{sk}_2)$
$\beta_1 \xleftarrow[u]{} \mathcal{B}_1$ $\beta_2 \xleftarrow[u]{} \mathcal{B}_2$	$\xrightarrow{\quad \beta_1, \beta_2 \quad}$	
	$\xleftarrow{\quad \gamma_1, \gamma_2 \quad}$	$\gamma_1 \leftarrow \mathcal{P}_1(\mathsf{sk}_1, \beta_1)$ $\gamma_2 \leftarrow \mathcal{P}_2(\mathsf{sk}_2, \beta_2)$

Halt if $\mathcal{V}_1(\mathsf{pk}, \alpha_1, \beta_1, \gamma_1) = 0$
Halt if $\mathcal{V}_2(\mathsf{pk}, \alpha_2, \beta_2, \gamma_2) = 0$

If we run two sigma protocols for different relations R_1 and R_2 in parallel, we get a sigma protocol* for new relation $R_1 \wedge R_2$

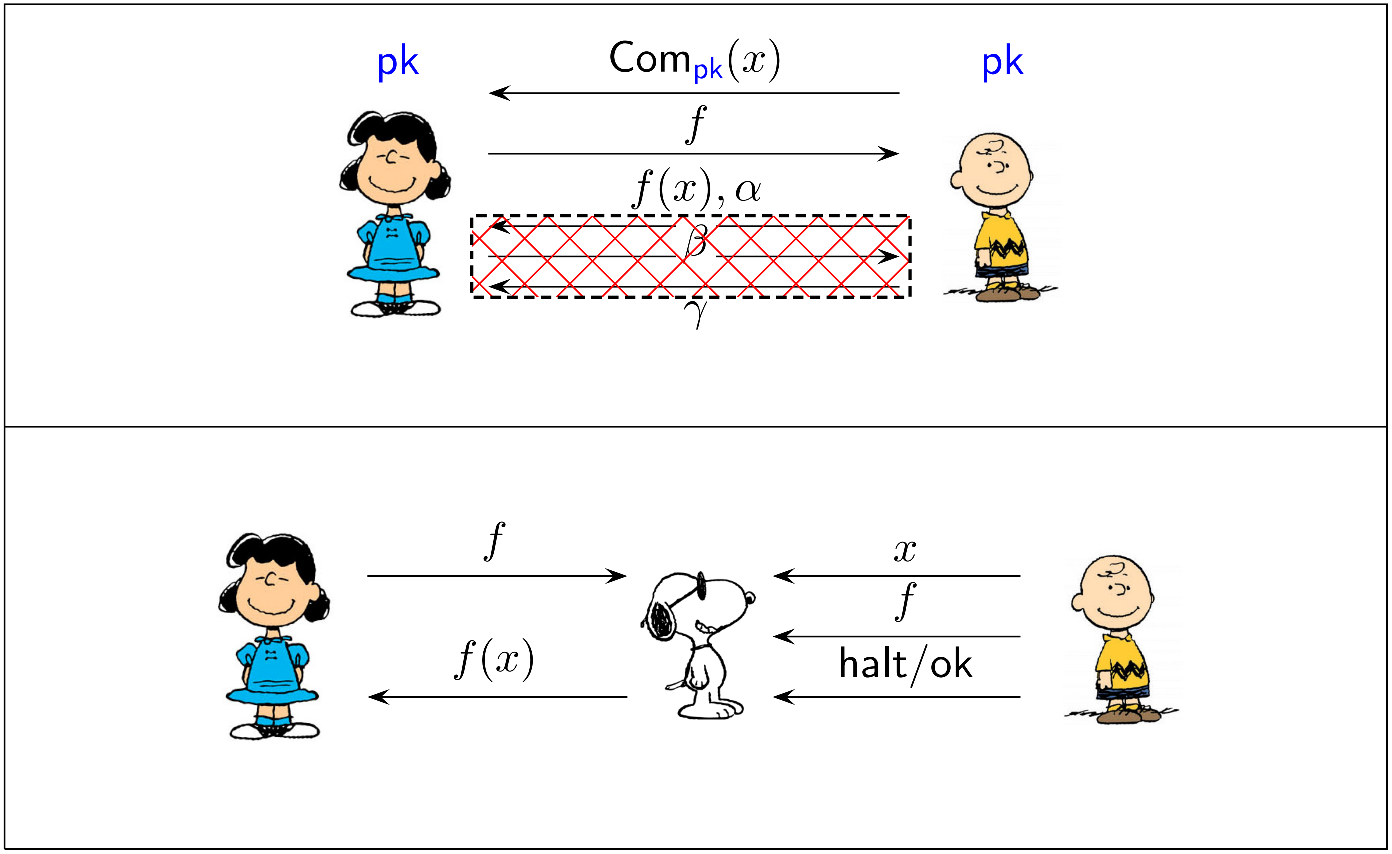
$$(\mathsf{sk}_1, \mathsf{sk}_2, \mathsf{pk}) \in R_1 \wedge R_2 \quad \Leftrightarrow \quad (\mathsf{sk}_1, \mathsf{pk}) \in R_1 \wedge (\mathsf{sk}_2, \mathsf{pk}) \in R_2 \ .$$

* Modulo small details—not all colliding transcripts reveal both secrets.

Certified Computations

Semihonest case

The concept



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

Basic tools

There are many ways how to build protocols for certified computations. Here, we consider one of the simplest protocols that is based DL group.

▷ We use Pedersen commitments with a public parameter $y \xleftarrow[u]{} \mathbb{G}$

$$(y^x g^r, (x, r)) \leftarrow \mathsf{Com}(x; r)$$

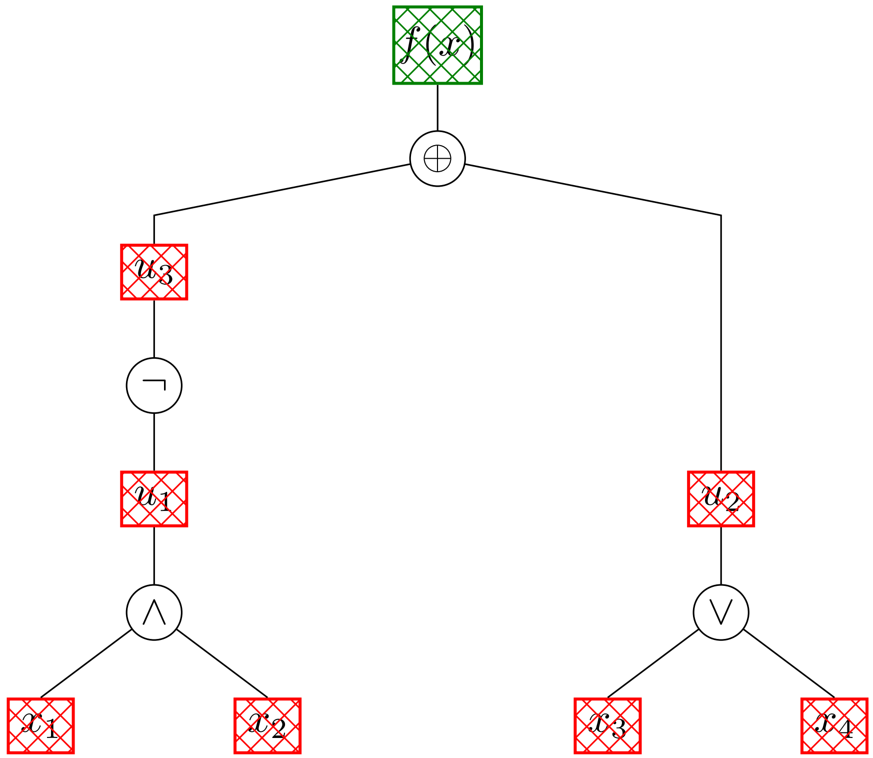
▷ We use proofs of knowledge for various relations about discrete logarithms

$$\mathrm{POK}_{z,g}\left[\exists x : g^x = z\right]$$
$$\mathrm{POK}_{g_1,g_2,z}\left[\exists x_1, x_2 : {g_1}^{x_1}{g_2}^{x_2} = z\right]$$

to prove more complex properties about Pedersen commitments.

Boolean circuit of commitments

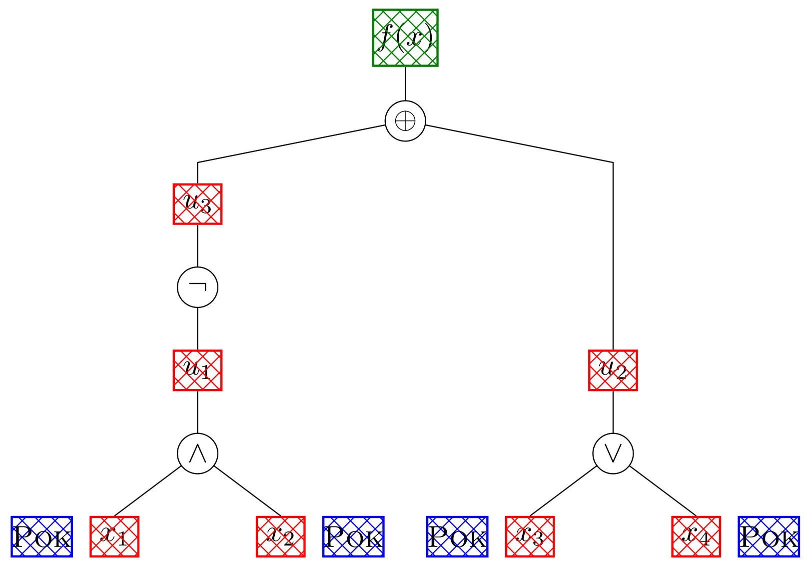
Charlie prepares a Boolean circuit for f and commits all intermediate values.



Augmentation by proofs of knowledge I

Charlie proves that all commitments $\mathsf{Com}(x_i)$ are commitments of bits

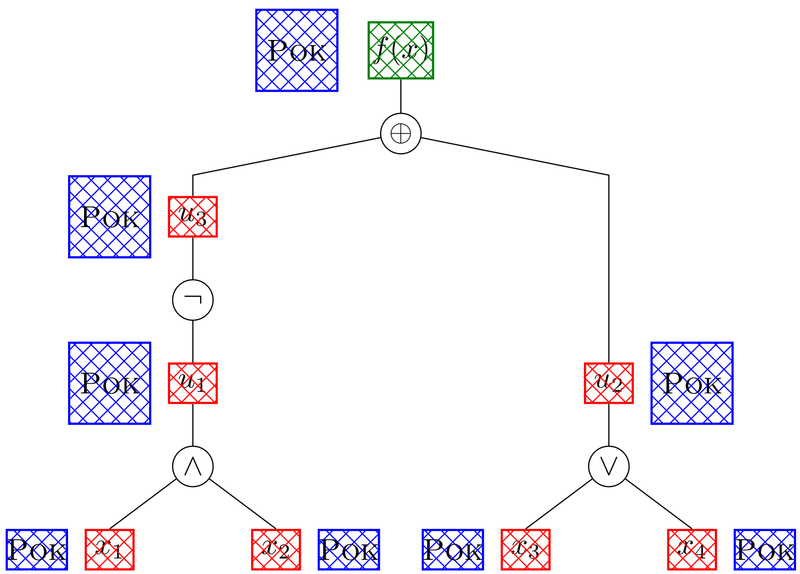
$$\boxed{\mathrm{POK}_{g,y,c}\left[\exists r : g^r = c \vee g^r = cy^{-1}\right]}$$



Augmentation by proofs of knowledge II

Charlie proves that all intermediate commitments are correct, e.g.

$$\boxed{\mathrm{POK}^{\neg}_{g,y,c_1,c_2}\left[\exists r_1 r_2 : g^{r_1} = c \wedge g^{r_2} = c_2 y^{-1} \vee \ldots\right]}$$



Final protocol

Since we can use disjunctive composition to combine all sigma proofs, we get the following protocol for certified computations.

- ▷ Charlie commits his input bit by bit using Pedersen commitment.
- ▷ Lucy sends the description of a function f.
- ▷ Charlie creates Boolean circuit and commits all values.
- ▷ Both parties agree one the corresponding validity proof.
- ▷ Charlie decommits $f(x)$ and sends the first proof message α.
- ▷ Lucy sends the challenge message $\beta \leftarrow \mathcal{B}$.
- ▷ Charlie sends back the corresponding response γ.
- ▷ Lucy accepts $f(x)$ only if the sigma protocol succeeds.