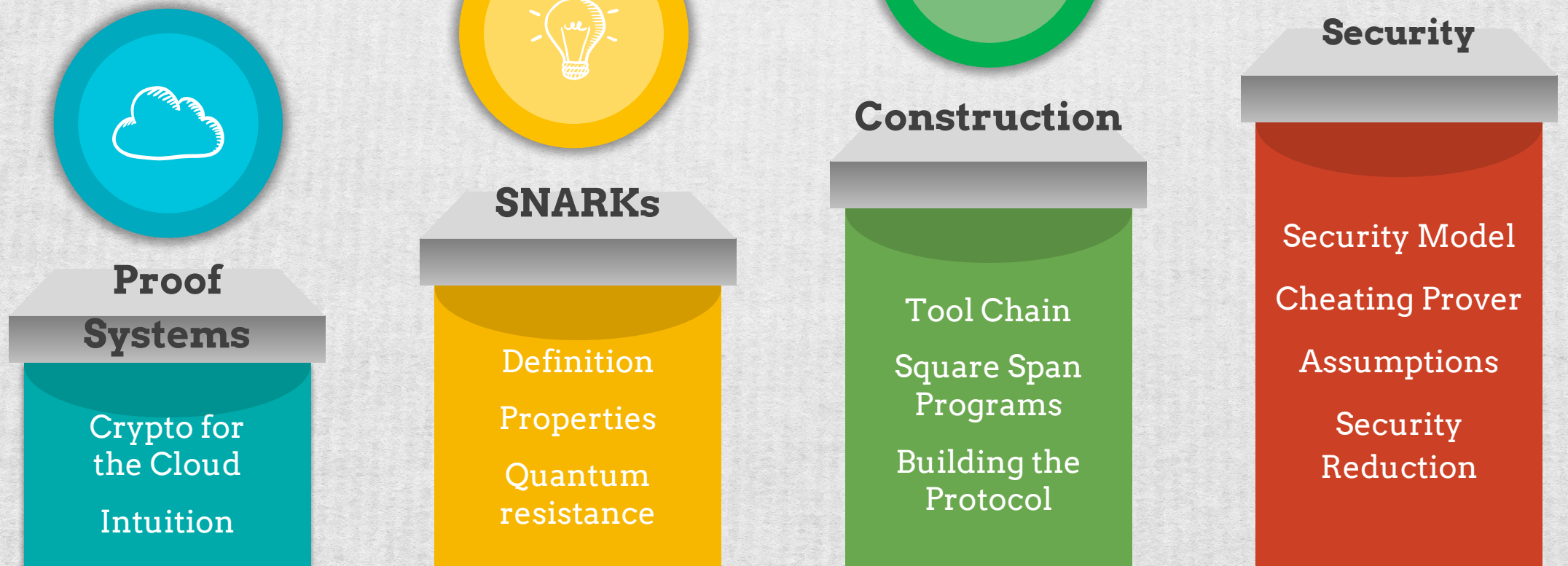
Lattice-Based zk-SNARKs from Square Span Programs

joint work with Rosario Gennaro, Michele Minelli

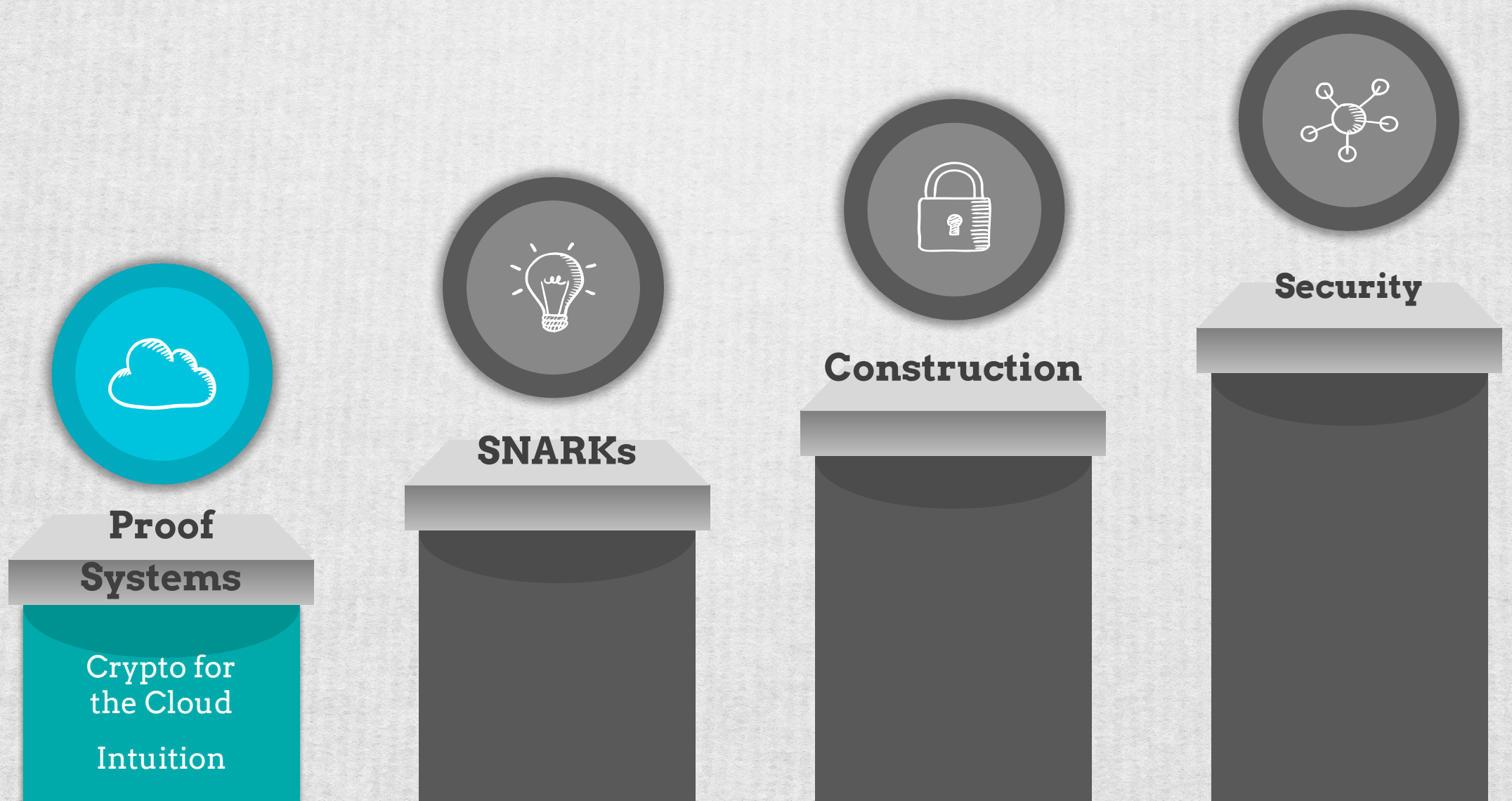
Anca Nitulescu, Michele Orrù

CRYPTO team
ENS Paris

Outline


Proof Systems
Crypto for the Cloud
Intuition
SNARKs
Definition
Properties
Quantum resistance
Construction
Tool Chain
Square Span Programs
Building the Protocol
Security
Security Model
Cheating Prover
Assumptions
Security Reduction

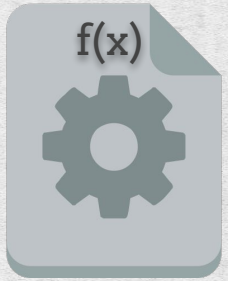
Motivation


Proof Systems
Crypto for the Cloud
Intuition
SNARKs
Construction
Security

Cryptography in the Cloud

Privacy

Authenticity



Integrity

Cryptography

Much of the cryptography used today offers security properties for **data** and **communication**.

Aspects in information security:

- **data confidentiality**
- **authentication**
- **data integrity**

what about **computations?**

Delegated Computation

Client

Worker

Delegated Computation

Client

Task

compute f(x)

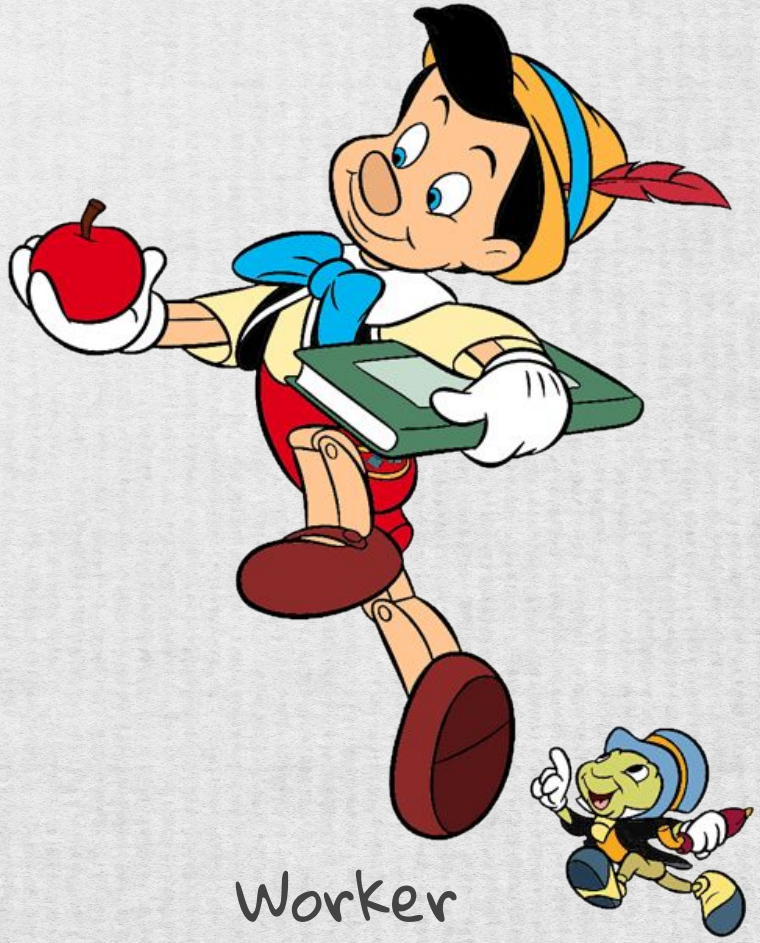
Worker

Delegated Computation

Client

"I have the

result **y**=f(x)"



Worker

Unreliable worker

Client

$y^* \neq f(x)$

Corrupted Worker

Ask for a proof

Verify π

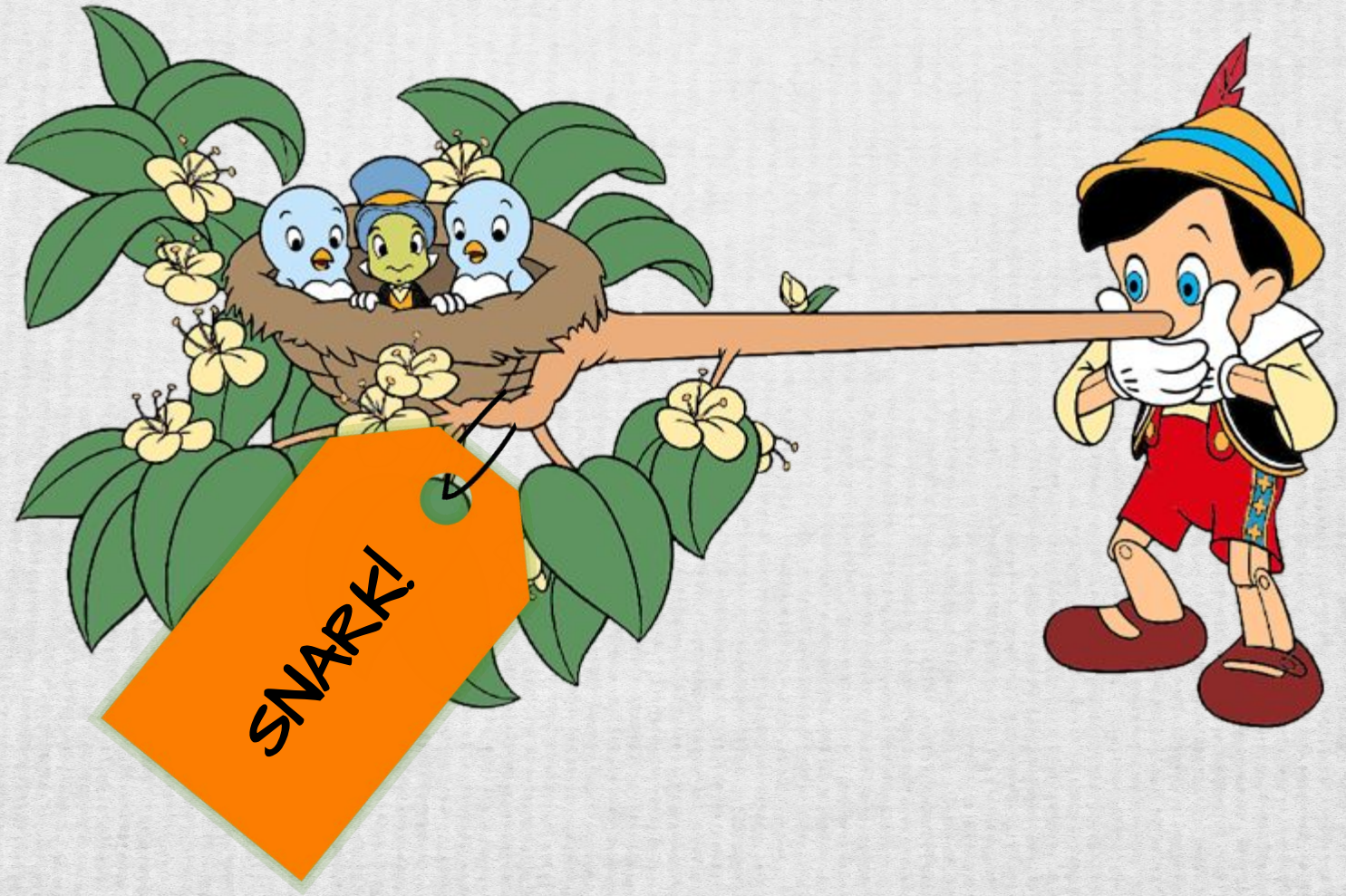
$y^* \neq f(x)$

Proof π

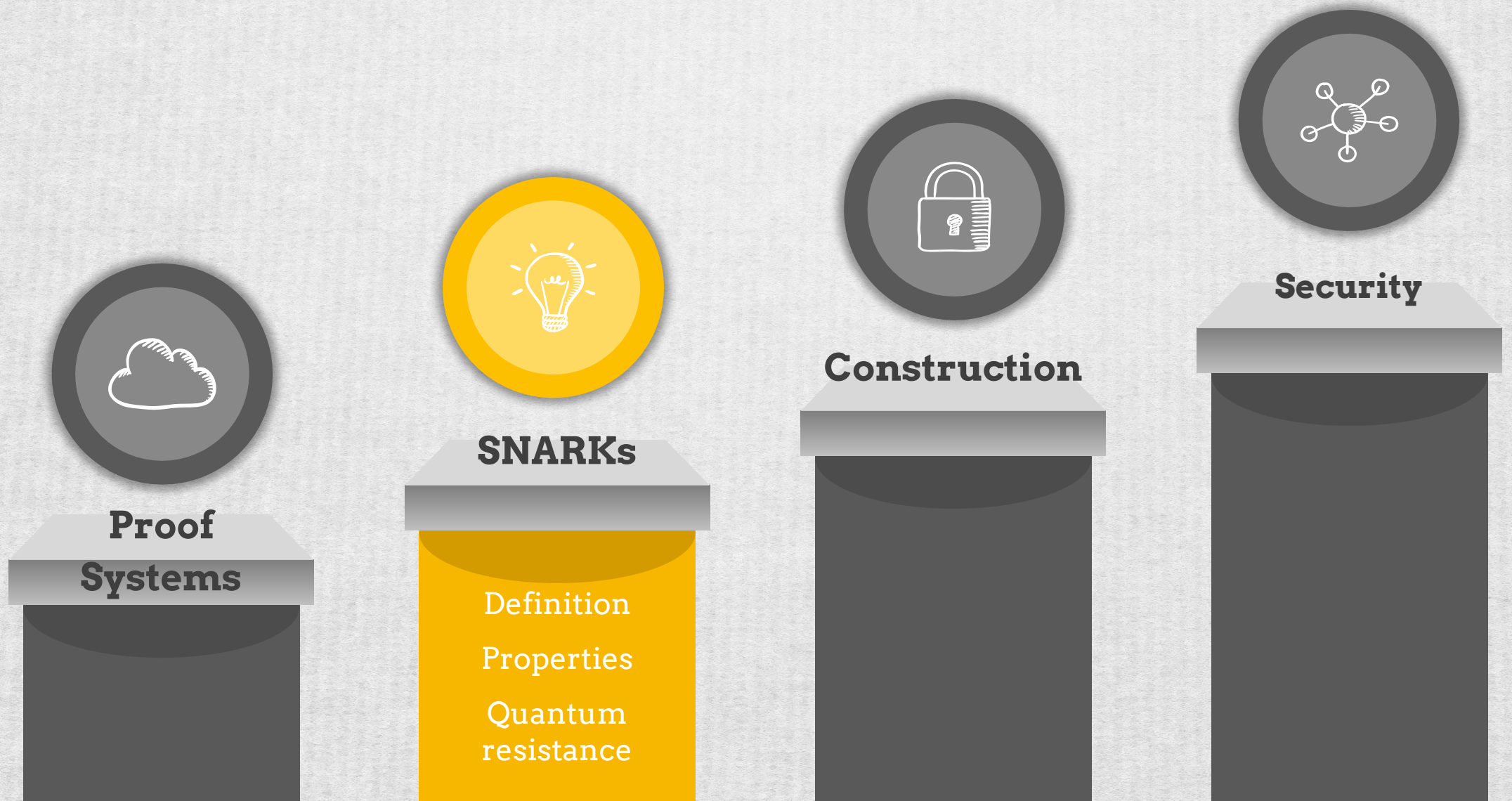
Client
Verifier

Worker
Prover

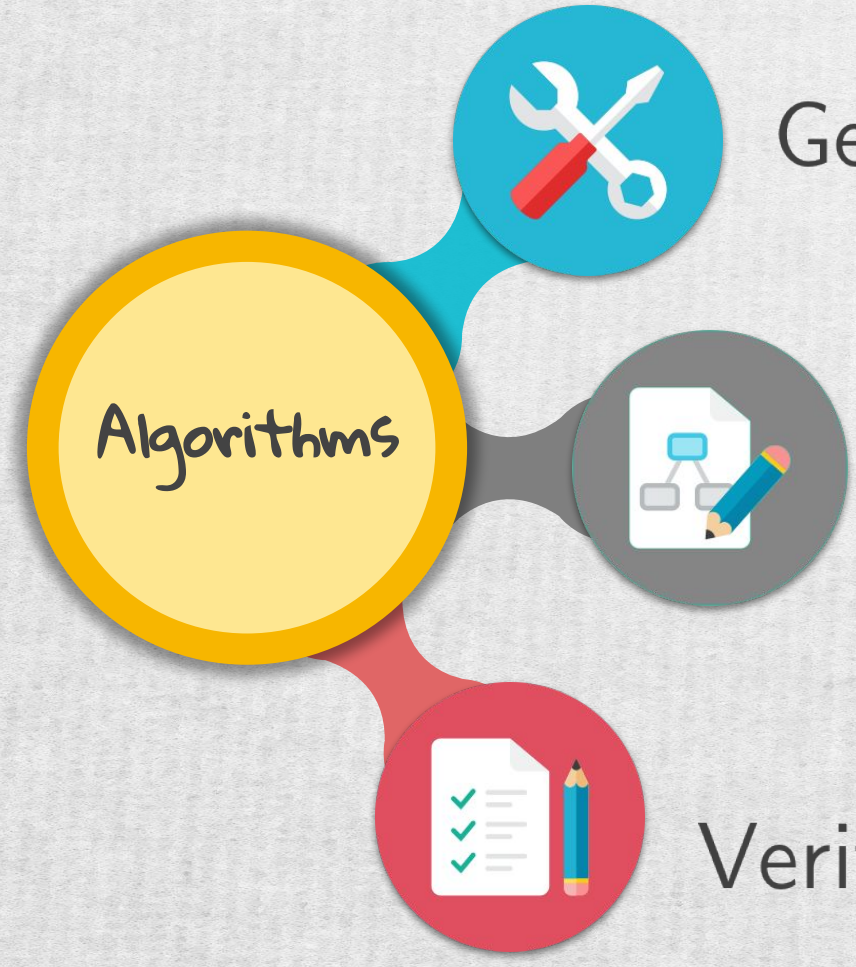
Integrity for Computation



SNARKs


Proof Systems
SNARKs
Definition
Properties
Quantum resistance
Construction
Security

Proof Systems: Since 1980s



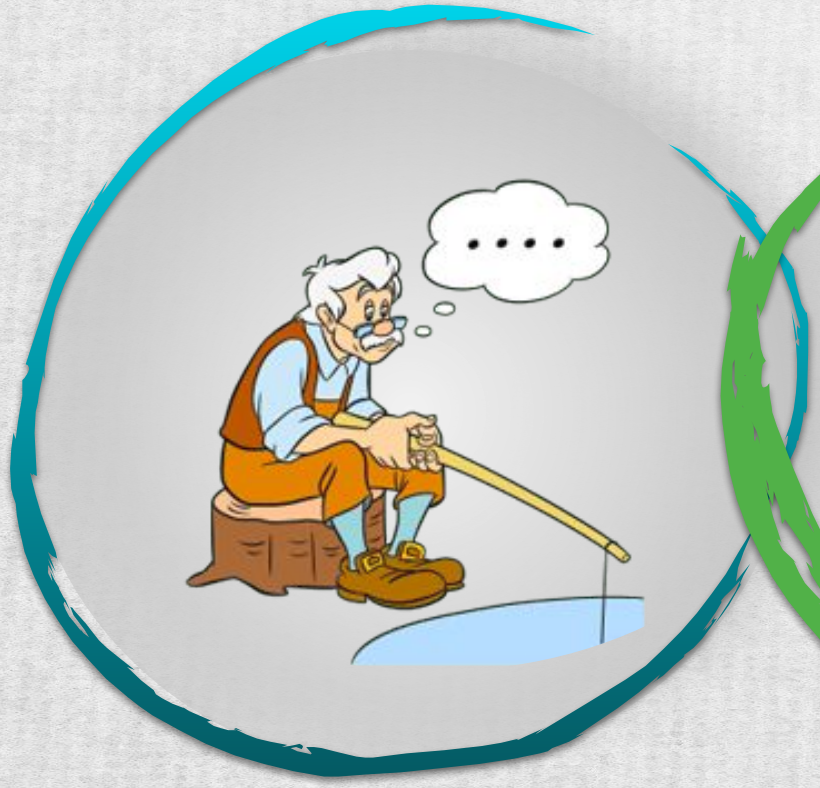
$$\mathsf{Gen}(1^\lambda, \mathcal{R}) \to (\mathsf{crs}, \mathsf{vk})$$

$$\mathsf{Prove}(\mathsf{crs}, y, w) \to \pi : (y, w) \in \mathcal{R}$$

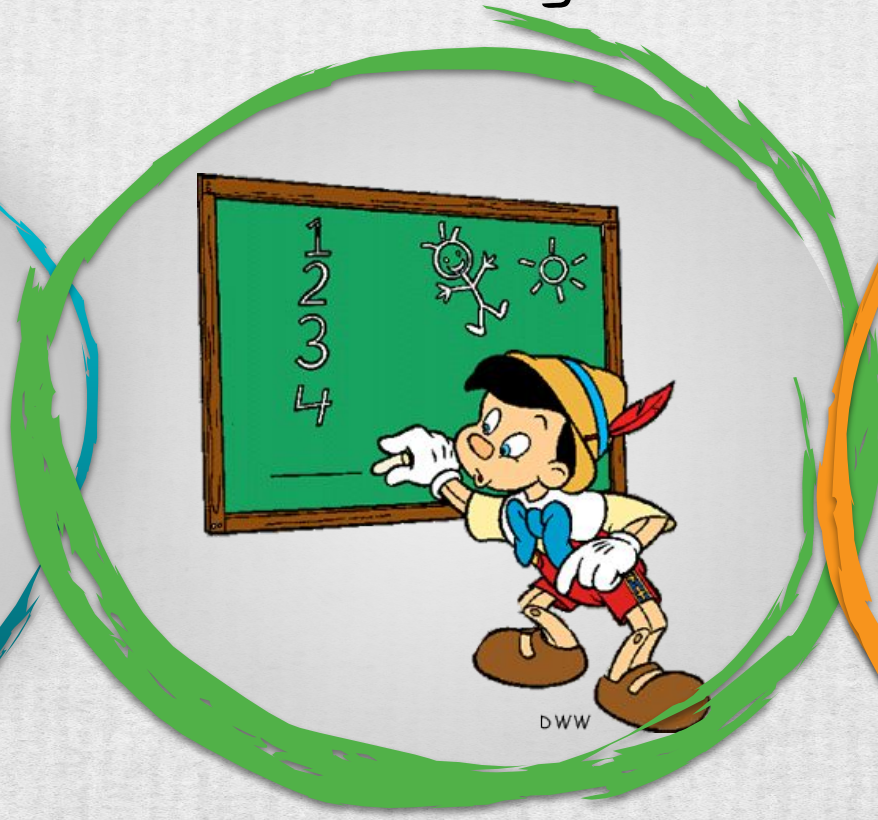
$$\mathsf{Verify}(\mathsf{vk}, y, \pi) \to 0/1$$

Properties for the Proof

Zero-Knowledge



Proof of Knowledge



Succinct

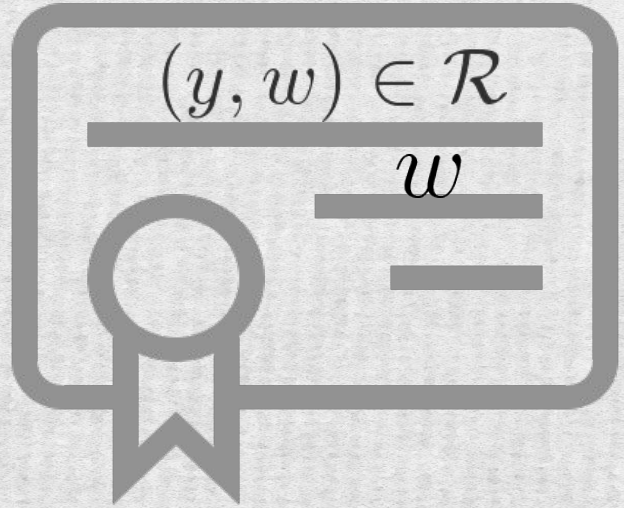
Properties for the Proof

Zero-Knowledge

Zero-Knowledge

Prover

Simulator

π 

$\approx$

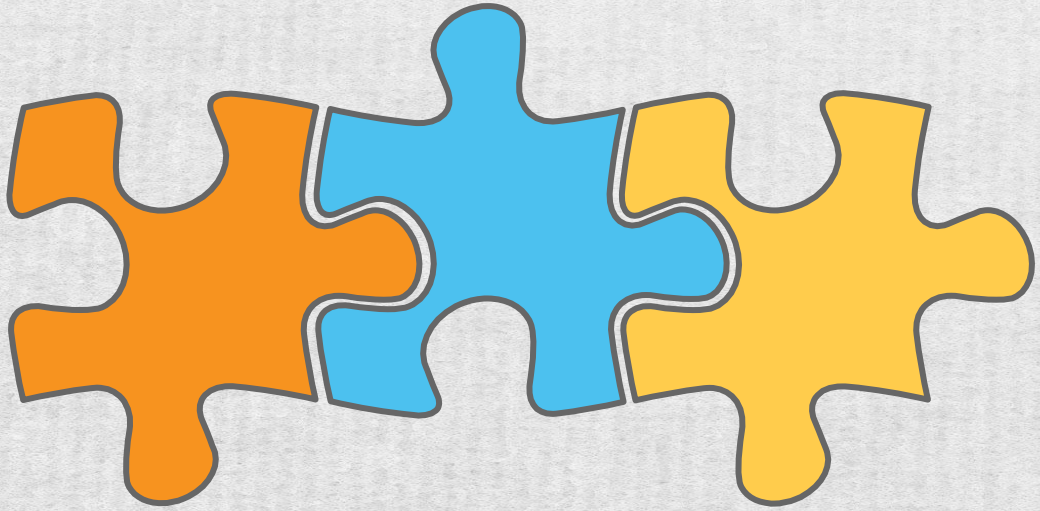
π' 

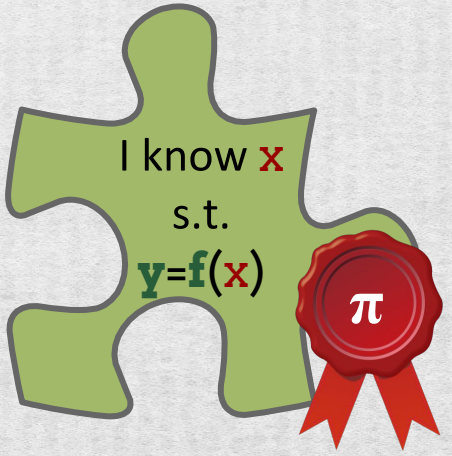
Proofs in Confidential Transactions

Key Properties for usage in Distributed Protocols

Private Blockchain

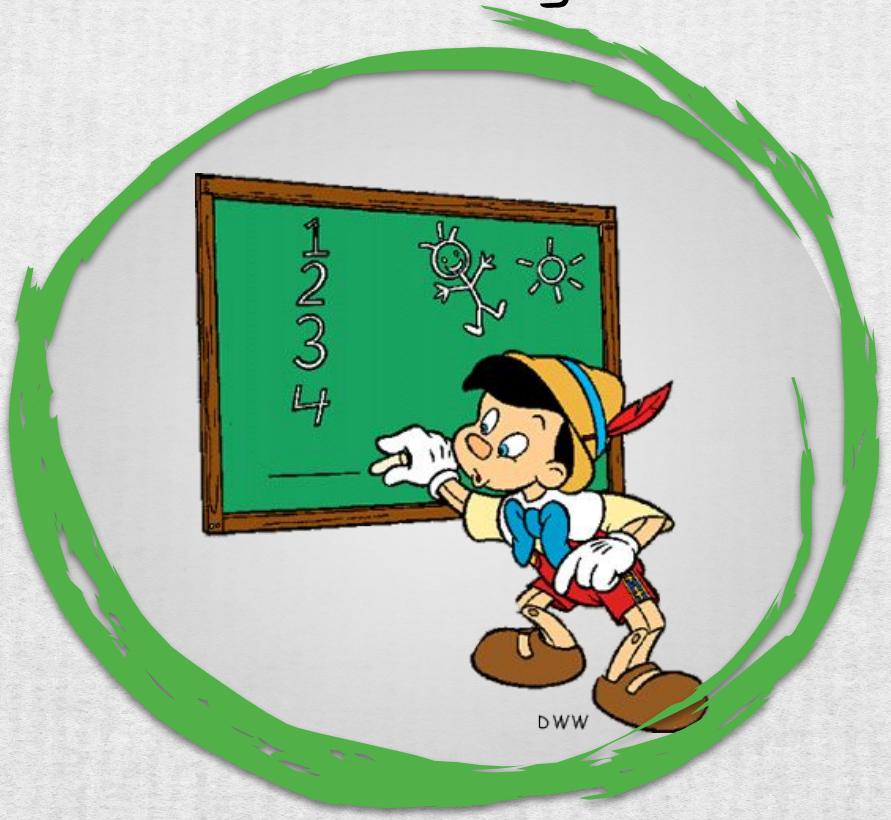
- **zero knowledge**
- **proof of knowledge**
- non-interactivity
- publicly verifiable
- succinctness





Properties for the Proof

Proof of Knowledge



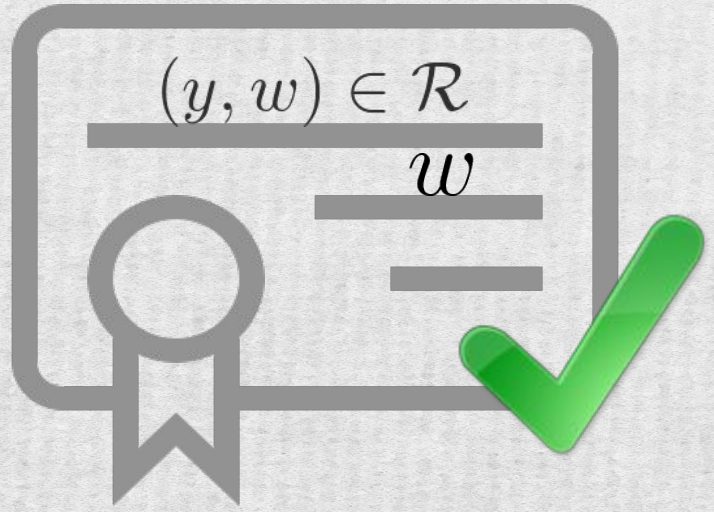
Argument of Knowledge

$\mathcal{A}$

crs

Adversary

π



Argument of Knowledge

$\mathcal{A}$

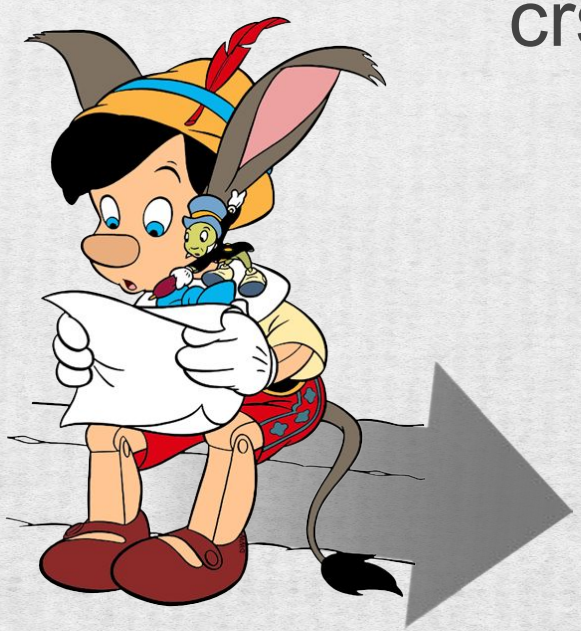
crs

Adversary

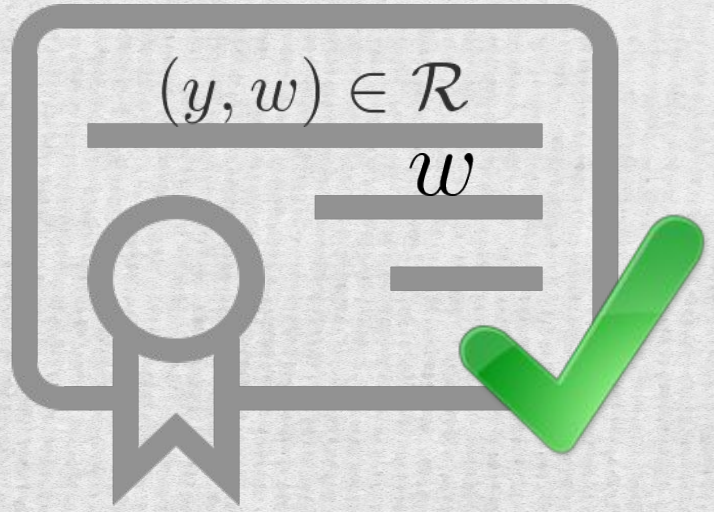
$\mathcal{E}$

crs

extractor



π



SNARK: Succinct Non-Interactive ARgument of Knowledge

Correctness
any correct evaluation **f(x)** has a valid proof

Succinctness
proof size independent of NP witness size

Zero-Knowledge
does not leak anything about the witness

Efficiency
verification faster than computing **f(x)**

Argument of Knowledge

Non-Interactivity
no exchange between prover and verifier

Implementations: Pinocchio, Geppetto

Pinocchio:
Nearly Practical
Verifiable
Computation

Bryan Parno,
Jon Howell,
Craig Gentry,
Mariana Raykova

Geppetto: Versatile
Verifiable
Computation

Craig Costello,
Cédric Fournet,
Jon Howell,
Markulf Kohlweiss,
Benjamin Kreuter, Michael
Naehrig,
Bryan Parno,
Samee Zahur

Quantum Attacks

Existing SNARKs:

- zero-knowledge
- publicly-verifiable (only **crs**)
- based on DLog in EC groups
- not quantum resistant

Quantum Attacks

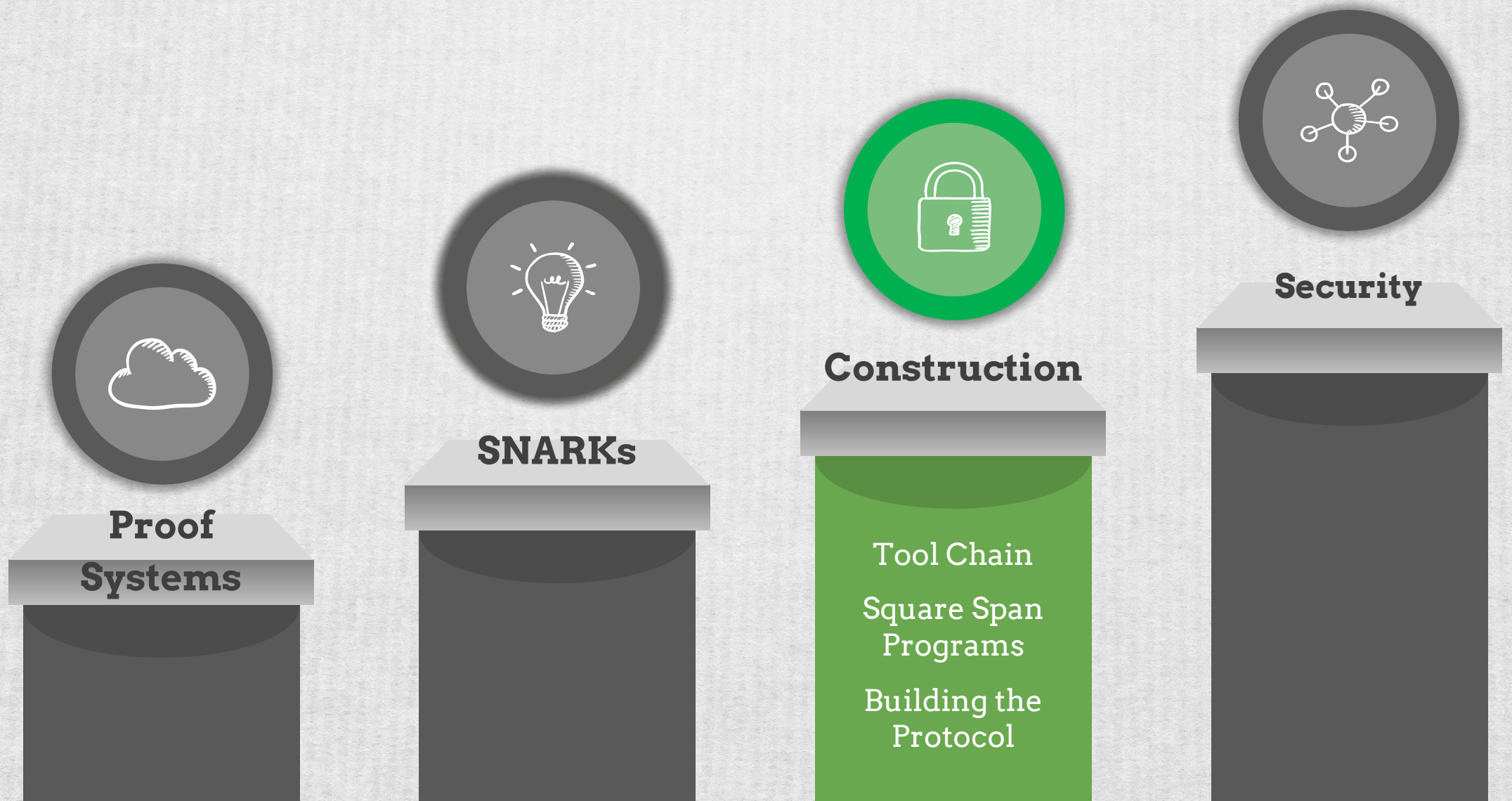
Existing SNARKs:

- zero-knowledge
- publicly-verifiable (only **crs**)
- based on DLog in EC groups
- not quantum resistant

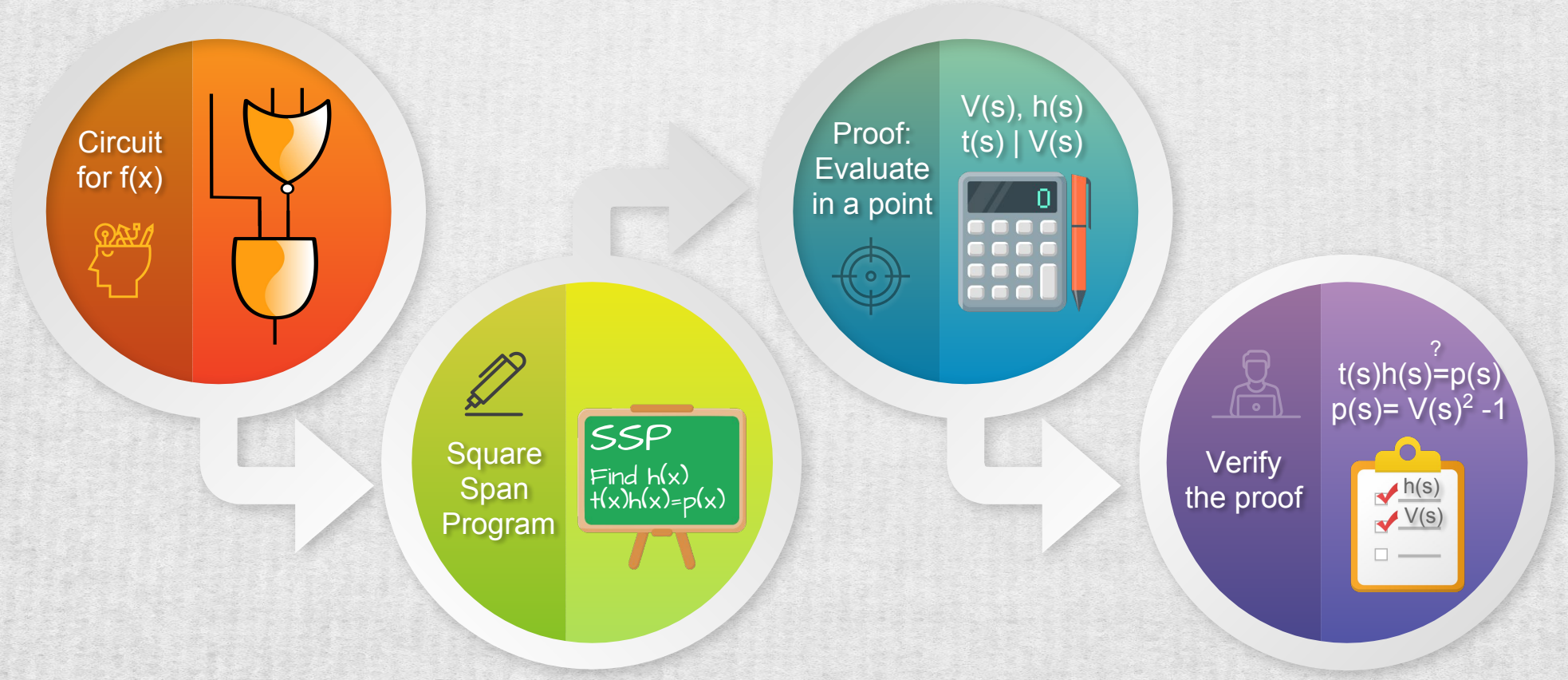
Post-Quantum zk-SNARKs:

- based on lattice assumptions
- designated-verifiable (**vk**)
- zero-knowledge

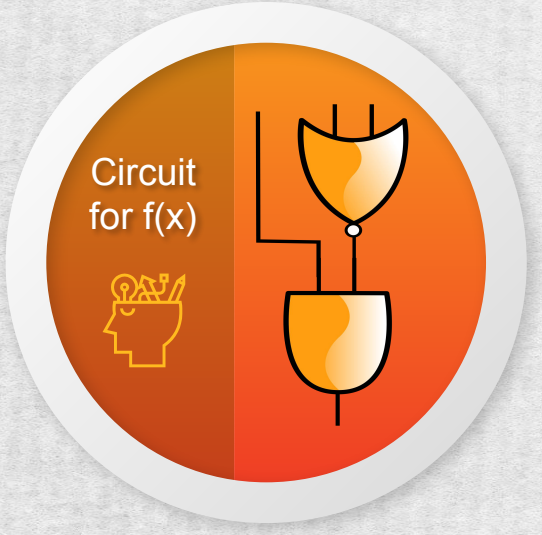
Construction



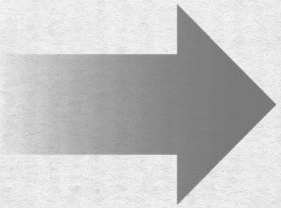
SNARK: overview of Toolchain

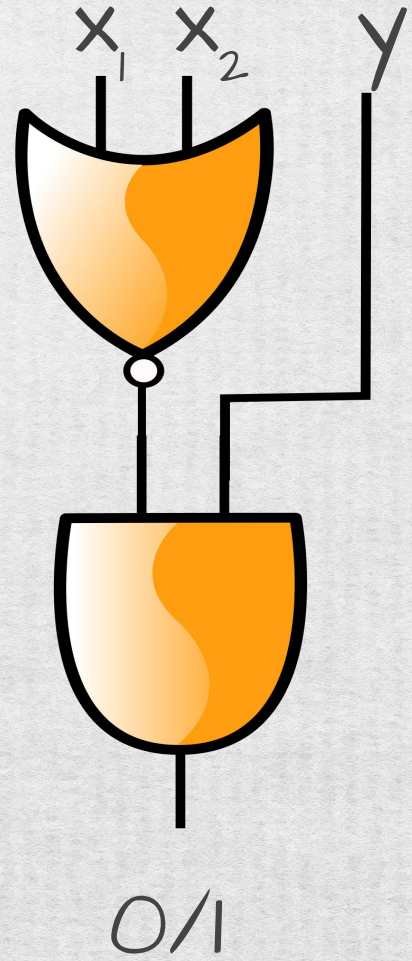


Circuit Satisfiability Problem



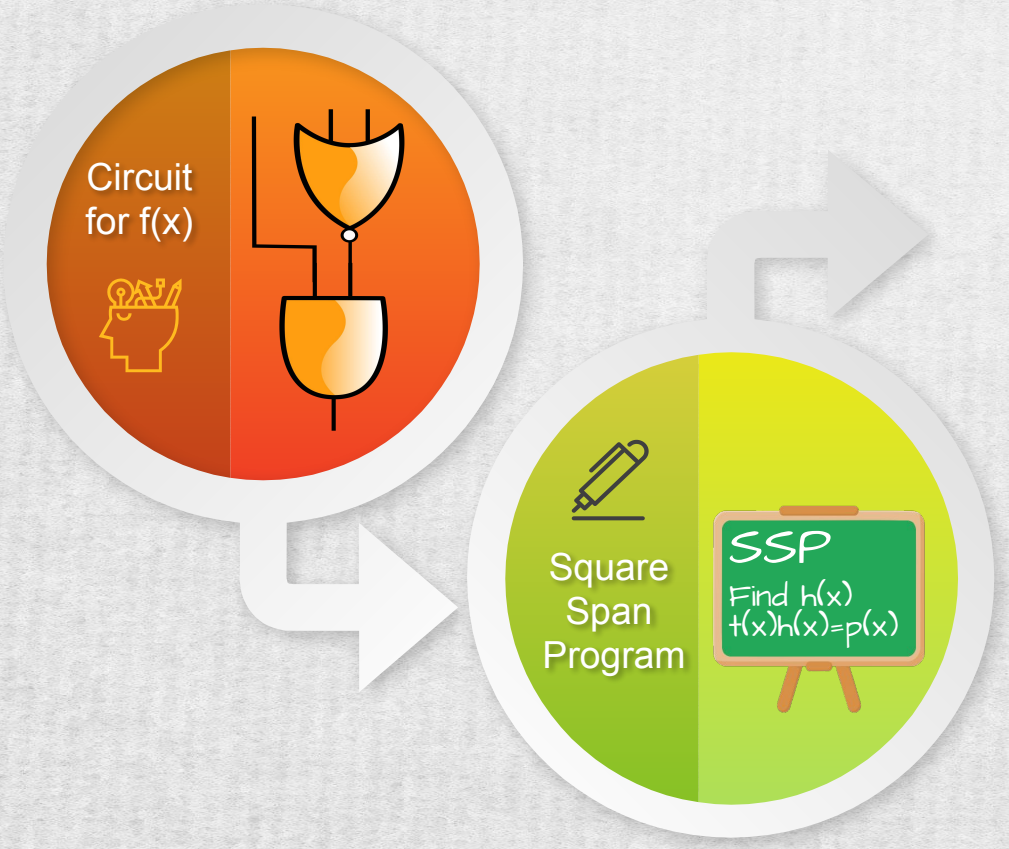
$f(x_1, x_2) = y$



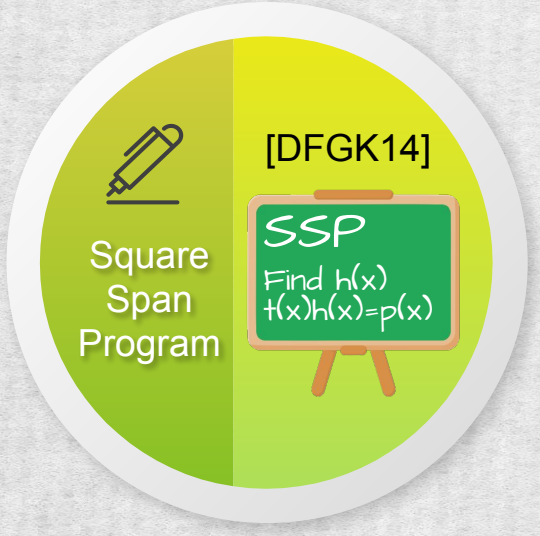


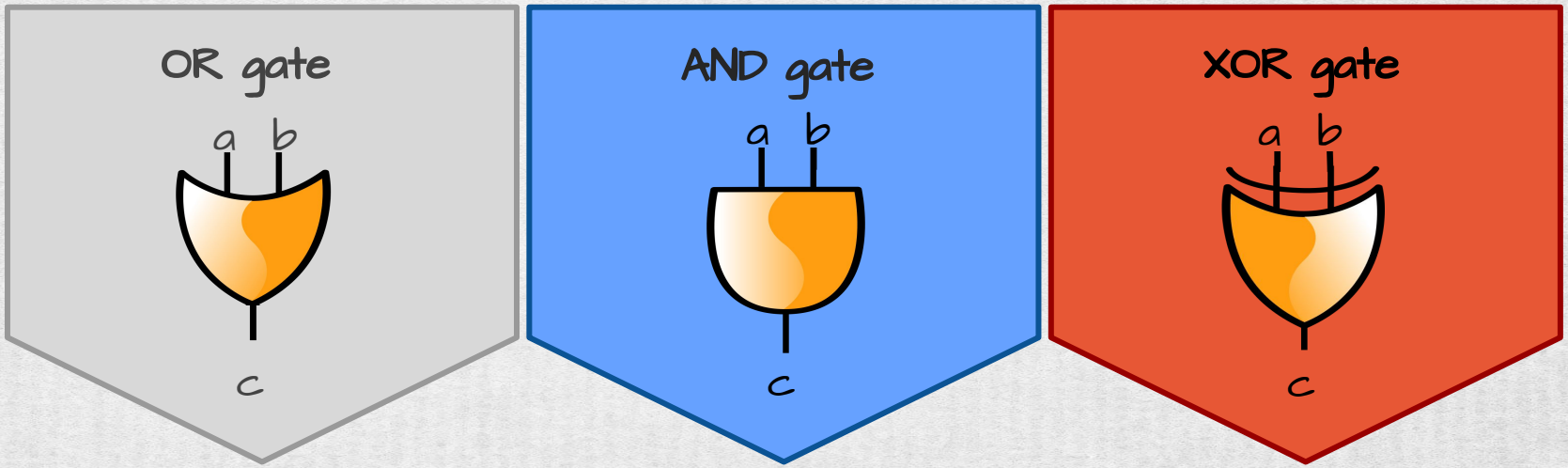
$C(x_1, x_2, y)$

SNARK: overview of Toolchain



Step 1. Linearization of logic gates

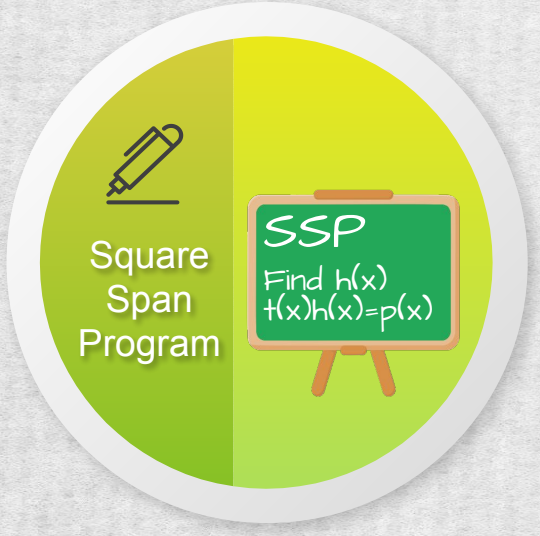




a b c	a b c	a b c
0 0 0	0 0 0	0 0 0
0 1 1	0 1 0	0 1 1
1 0 1	1 0 0	1 0 1
1 1 1	1 1 1	1 1 0
$-a-b+2c \in \{0,1\}$	$a+b-2c \in \{0,1\}$	$a+b+c \in \{0,2\}$

Step 2. Matrix equation for circuit

OR gate	AND gate	XOR gate	Output gate = 1	Entries = bits
$-a-b+2c \in \{0,1\}$	$a+b-2c \in \{0,1\}$	$a+b+c \in \{0,2\}$	$3-3c \in \{0,1\}$	$2a, 2b \in \{0,2\}$



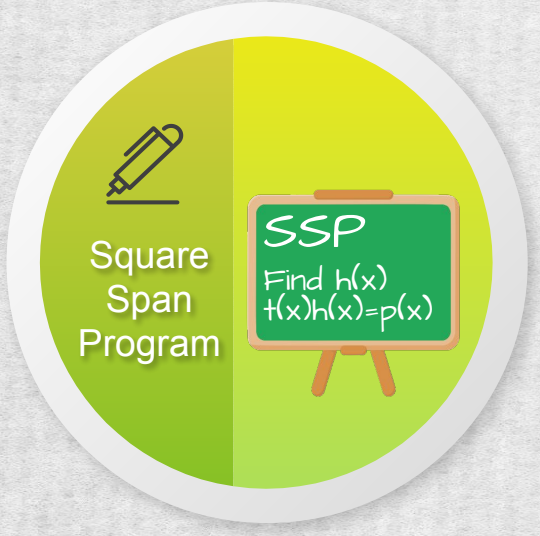
$$\alpha a + \beta b + \gamma c + \delta \in \{0,2\}$$

$$V \quad A + \delta \quad \in \{0,2\}^d$$

$$\left(V \quad A + \delta \right) \circ \left(V \quad A + \delta - 2 \right) = 0$$

Step 2. Matrix equation for circuit

OR gate	AND gate	XOR gate	Output gate = 1	Entries = bits
$-a-b+2c \in \{0,1\}$	$a+b-2c \in \{0,1\}$	$a+b+c \in \{0,2\}$	$3-3c \in \{0,1\}$	$2a, 2b \in \{0,2\}$

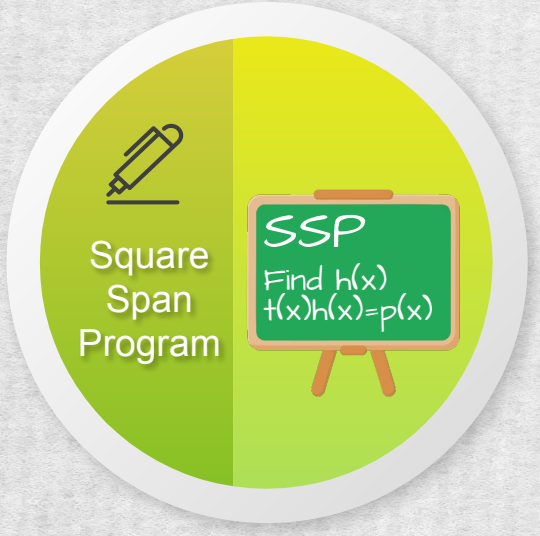


$$\alpha a + \beta b + \gamma c + \delta \in \{0,2\}$$

$$\left(V\,A + \delta\right) \circ \left(V\,A + \delta - 2\right) = 0$$

$$\left(V\,A + \delta - 1\right) \circ \left(V\,A + \delta - 1\right) = 1$$

Step 3. Polynomial Interpolation



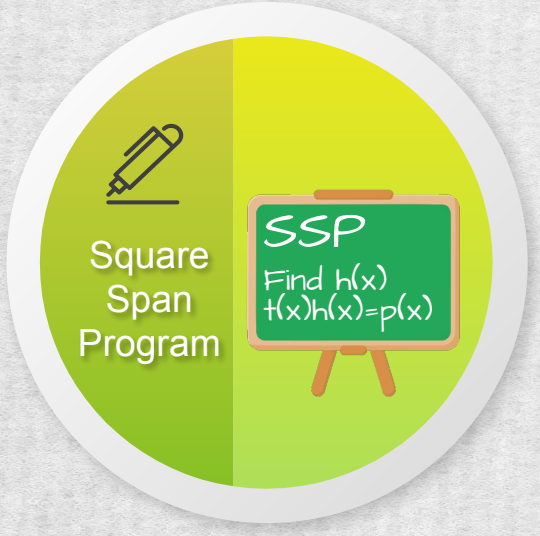
$$\left(V \cdot A + \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}\right) \circ \left(V \cdot A + \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}\right) = \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}$$

$$\forall \{r_j\} \in \mathbb{F}^d$$

$$\left(v_0(r_j) + \sum_{i=1}^{m} a_i v_i(r_j)\right)^2 - 1 = 0$$

$$v_0(r_j) = \delta_j - 1 \qquad v_i(r_j) = V_{ji}$$

Step 4. Polynomial Problem SSP

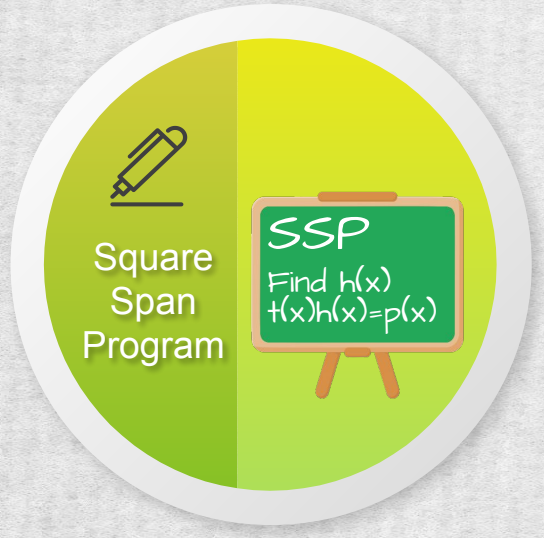


$$\left(v_0(r_j) + \sum_{i=1}^{m} a_i v_i(r_j)\right)^2 - 1 = 0$$

$$\forall \{r_j\} \in \mathbb{F}^d$$

$$\prod_{j=1}^{d}(x - r_j) \Bigg| \left(v_0(x) + \sum_{i=1}^{m} a_i v_i(x)\right)^2 - 1$$

Step 4. Polynomial Problem SSP



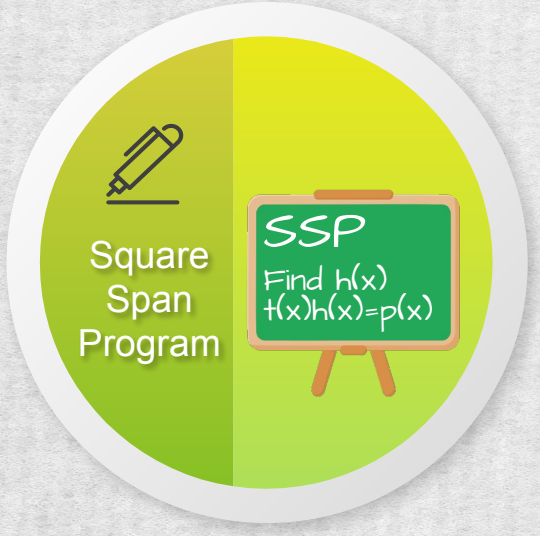
For $\{v_i(x)\}_{i=1,m},\ t(x) \in \mathbb{F}[x]$

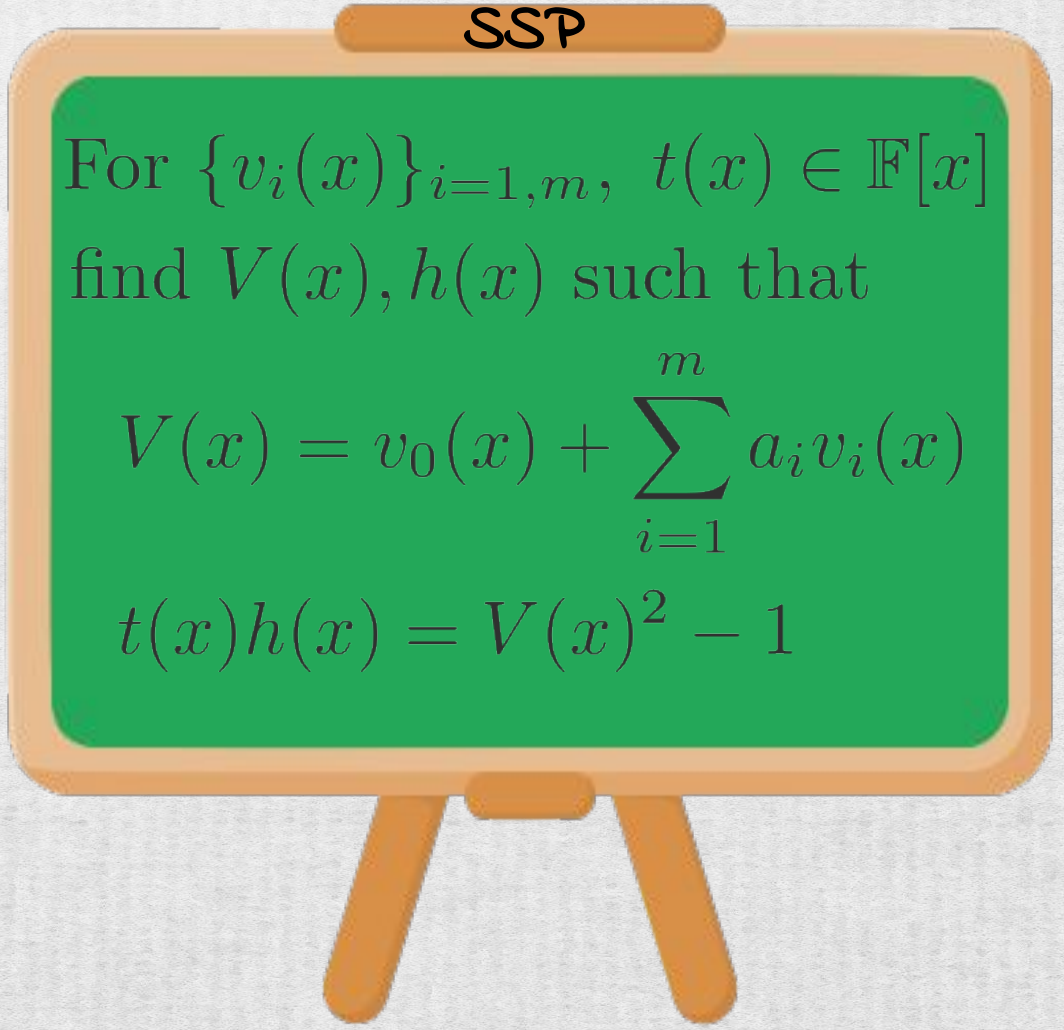
$$t(x) \mid V(x)^2 - 1$$

$$V(x) = v_0(x) + \sum_{i=1}^{m} a_i v_i(x)$$

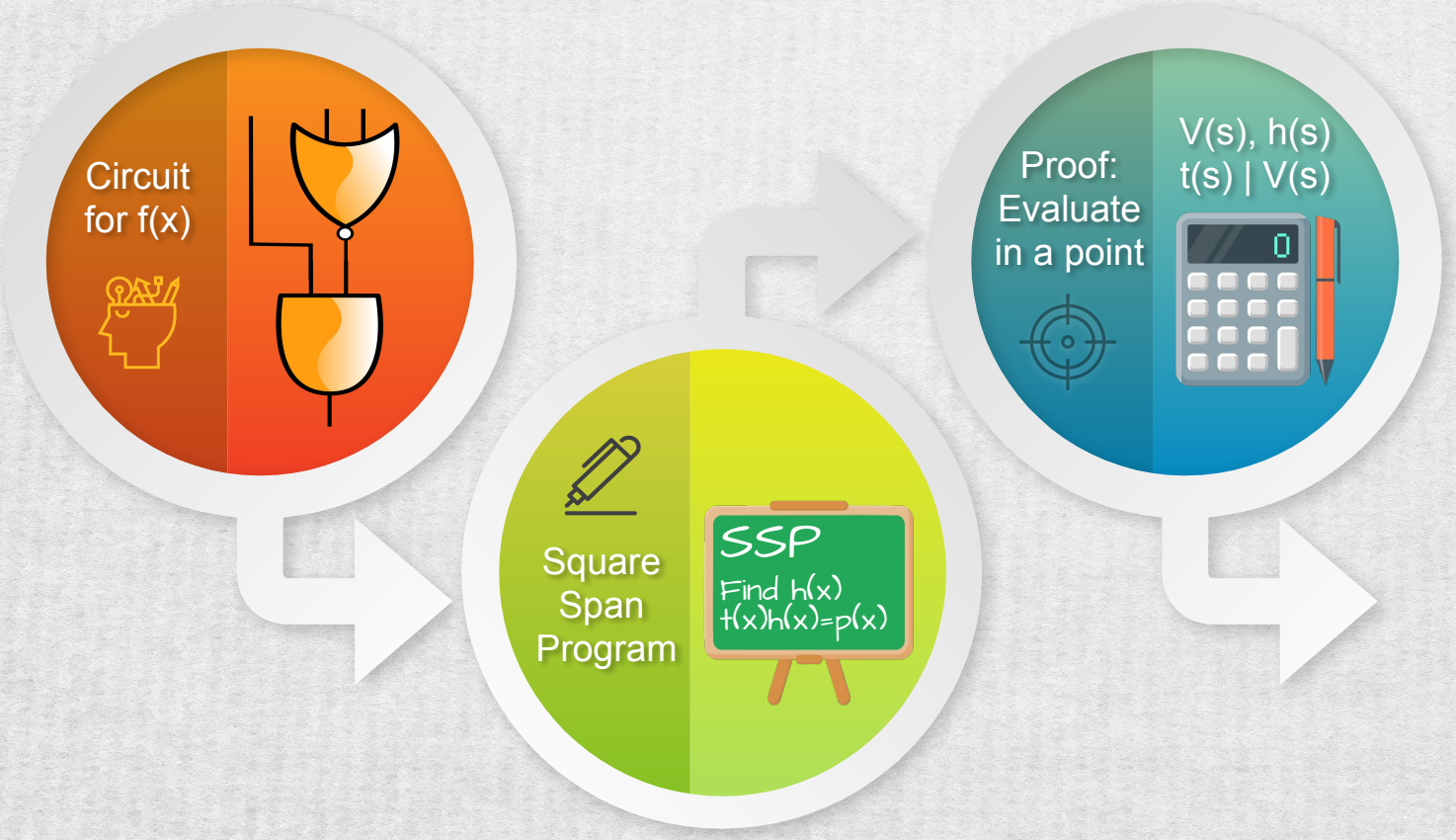
$$t(x) = \prod_{j=1}^{d} (x - r_j)$$

Step 4. Polynomial Problem SSP

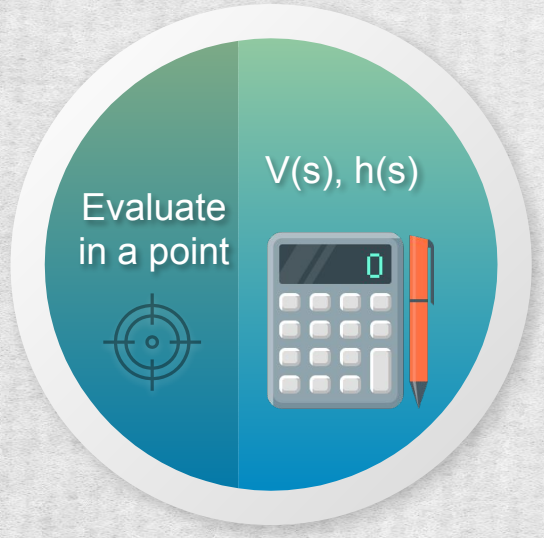


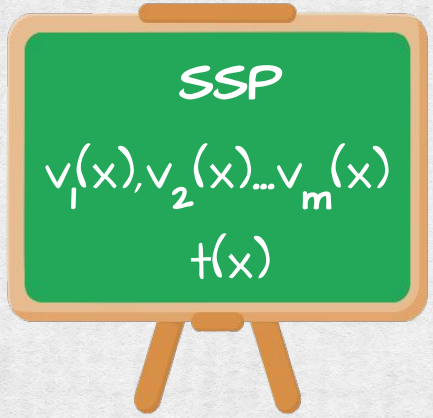


SNARK: overview of Toolchain



Proving on top of SSP: Idea





$$V(x), h(x) = ?$$

$$V(x) = v_0(x) + \sum_{i=1}^{m} a_i v_i(x)$$

$$t(x)h(x) = V(x)^2 - 1$$

Prover: Has to find 2 polynomials that satisfy the conditions of the SSP problem.

degree of each polynomial = **size** of the Circuit

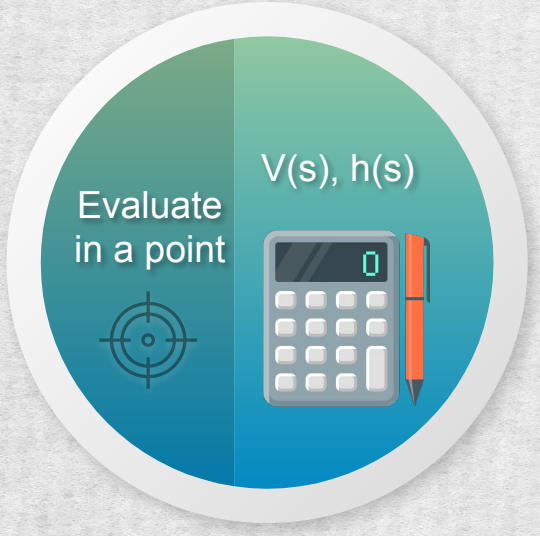
Send all the coefficients to the Verifier…

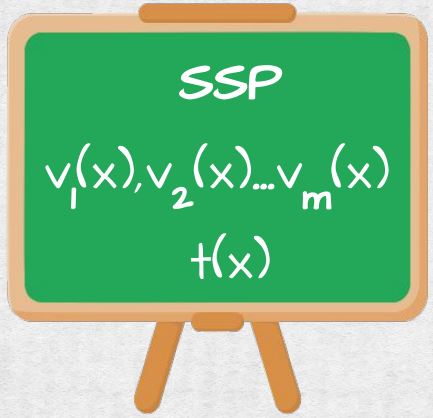
Size of the proof = TOO Large!!!

Succinct Proof

Succinct

Proving on top of SSP: Idea





$$V(s), h(s) = ?$$

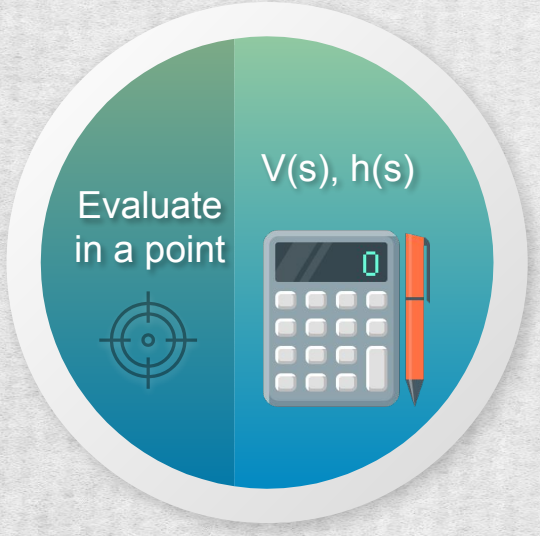
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

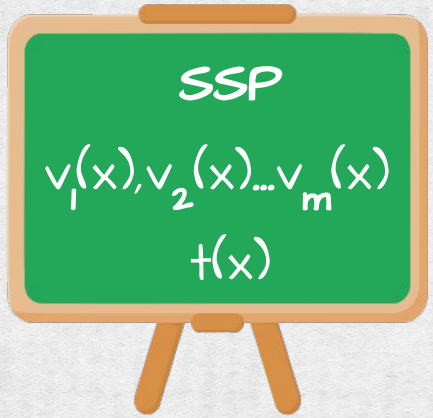
$$t(s)h(s) = V(s)^2 - 1$$

Prover: Evaluate the solution (the two polynomials) in a random unknown point **s**

Preprocessing: Publish all necessary powers of **s** (hidden from the **Prover**)

Proving on top of SSP: Idea

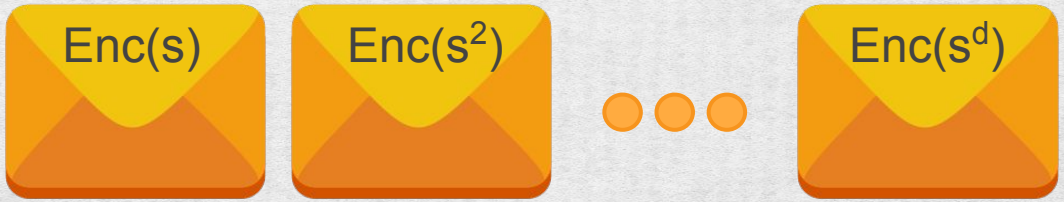




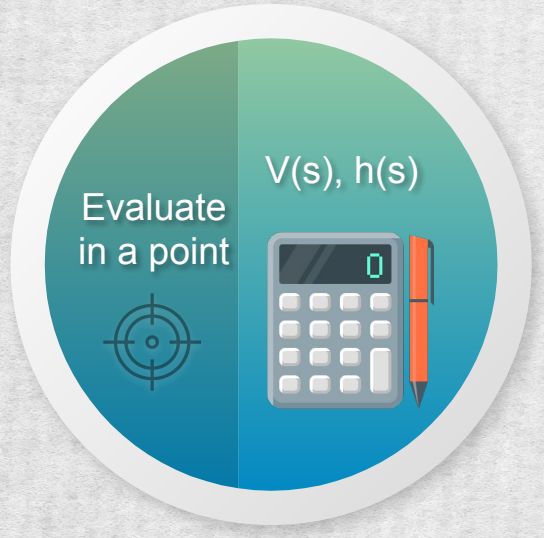
$$V(s), h(s) = ?$$

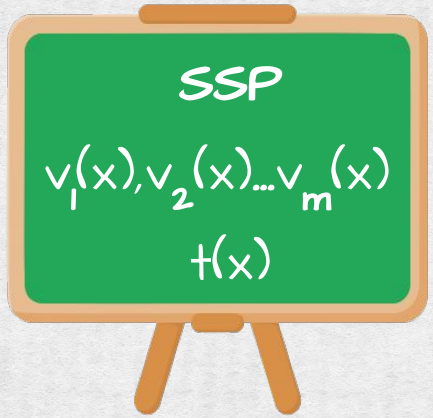
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

$$t(s)h(s) = V(s)^2 - 1$$



Proving on top of SSP: Idea

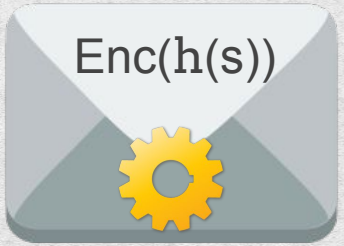




$$V(s), h(s) = ?$$

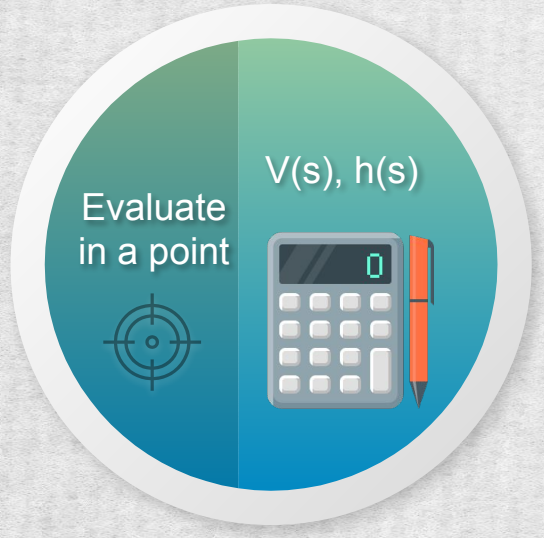
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

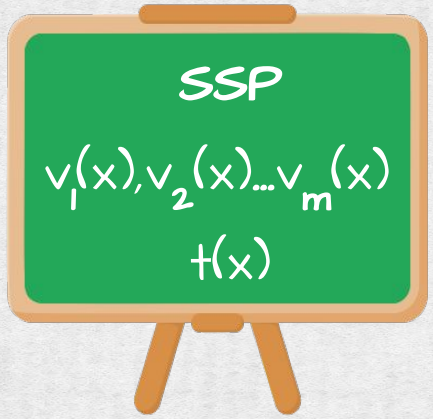
$$t(s)h(s) = V(s)^2 - 1$$



$= \mathrm{Enc}(\Sigma\, h_i s^i)$

Proving on top of SSP: Idea





$$V(s), h(s) = ?$$

$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

$$t(s)h(s) = V(s)^2 - 1$$

Encoding:

- ***linearly*** homomorphic

$$\mathrm{Enc}(\Sigma\, h_j s^j) = \Sigma_j\, h_j\, \mathrm{Enc}(s^j)$$

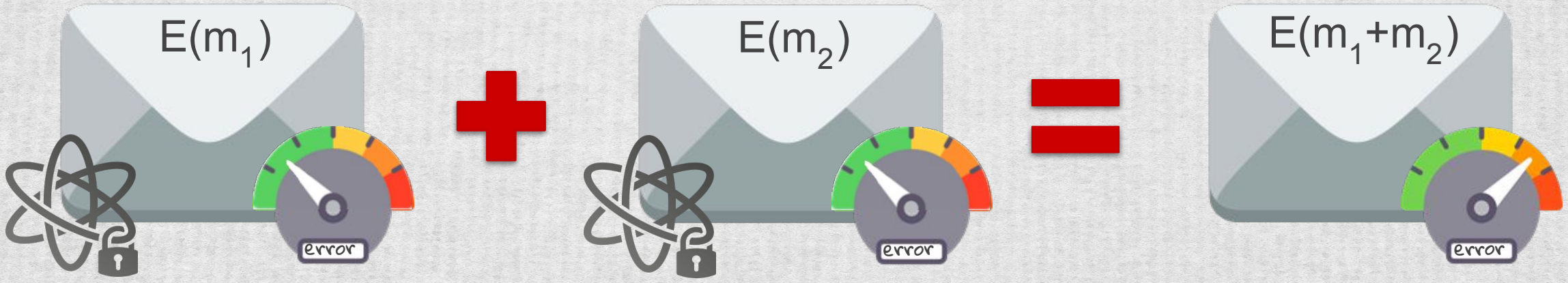
Lattice-based Encodings: Regev Encryption Scheme

error

Encryption: $E_{\vec{s}}(m) = (-\vec{a}, \vec{a}\vec{s} + pe + m),\ e \in \chi$

Decryption: $D_{\vec{s}}((\vec{c_0}, c_1)) = \vec{c_0} \cdot \vec{s} + c_1 \pmod{p}$

$$E_{\vec{s}}(m_1) + E_{\vec{s}}(m_2) = (-\vec{a}_1 - \vec{a}_2, (\vec{a}_1 + \vec{a}_1)\vec{s} + p(e_1 + e_2) + m_1 + m_2)$$



Discrete Log Encoding

$$Enc(s) = g^s$$

$$\langle g \rangle = \mathbb{G}, \langle \tilde{g} \rangle = \tilde{\mathbb{G}}$$

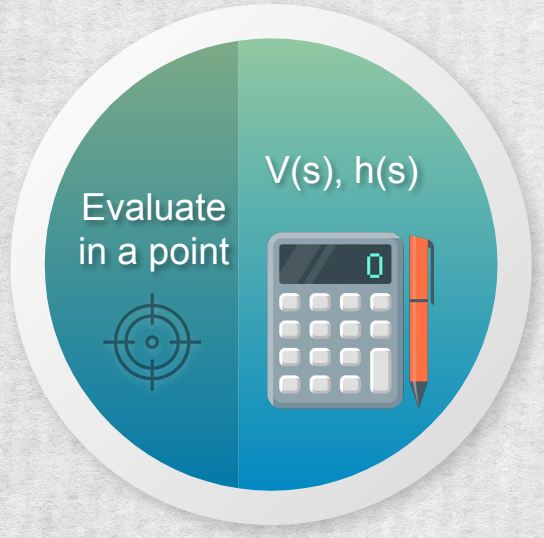
$$e : \mathbb{G} \times \mathbb{G} \to \tilde{\mathbb{G}}$$

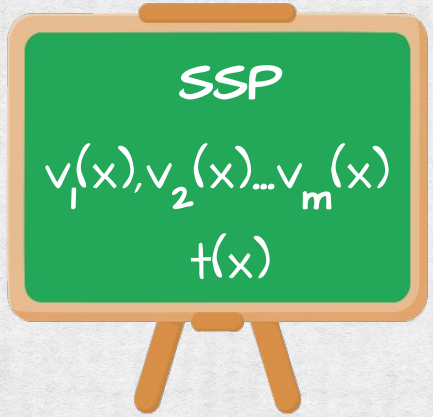
$$e(g^a, g^b) = \tilde{g}^{ab}$$

Linearly homomorphic

$$Enc(p(s)) = g^{p(s)} = g^{\sum_i p_i s^i} = \prod (g^{s^i})^{p_i}$$

Proving on top of SSP: Idea

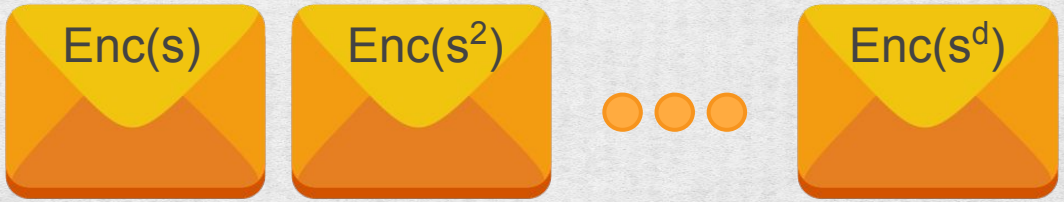


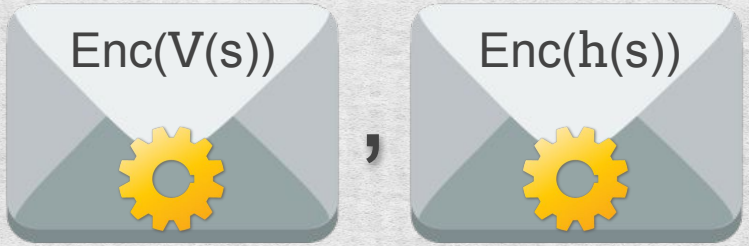


$$V(s), h(s) = ?$$

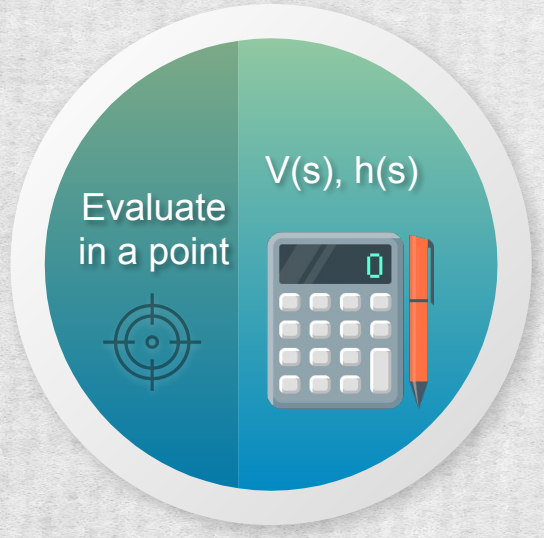
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

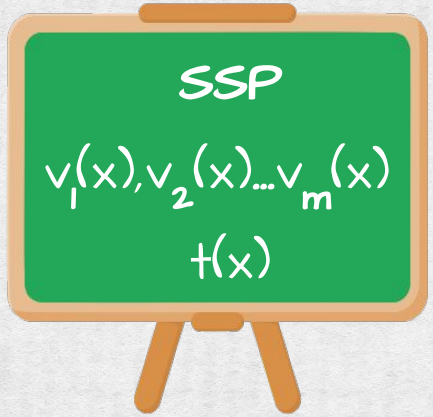
$$t(s)h(s) = V(s)^2 - 1$$



Proof  = 

Span Property

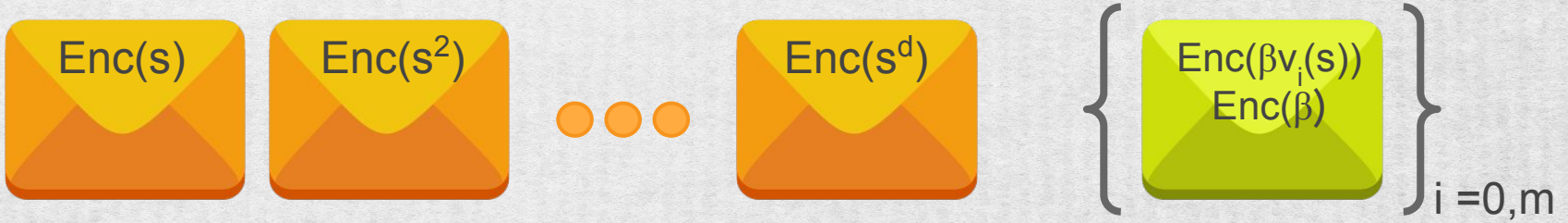


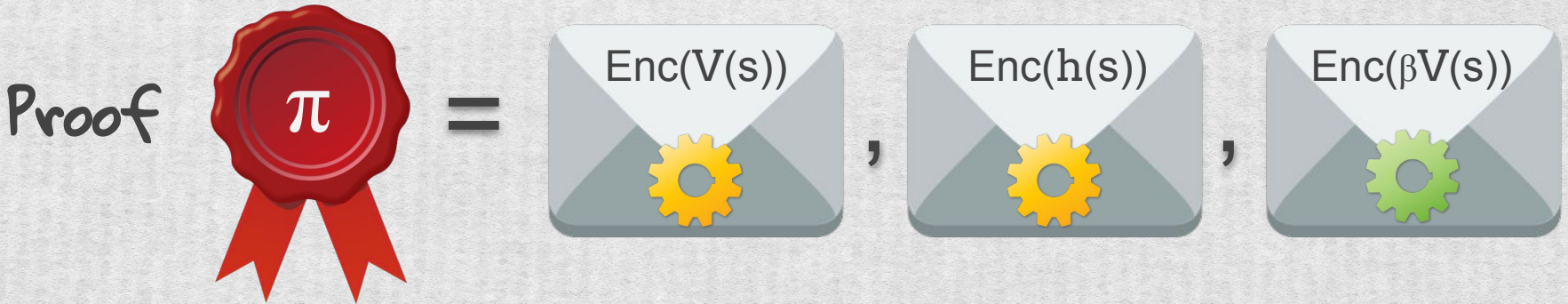


$$V(s), h(s) = ?$$

$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

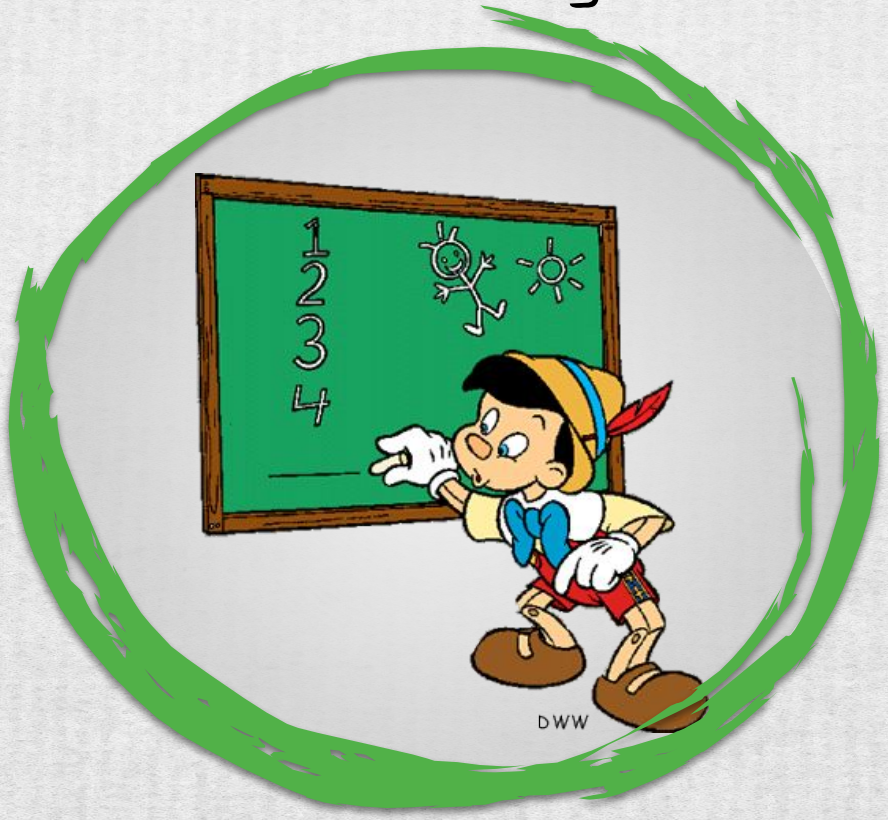
$$t(s)h(s) = V(s)^2 - 1$$





Not an Argument of Knowledge! Extractability?

Argument of Knowledge



Argument of Knowledge

$\mathcal{A}$

crs

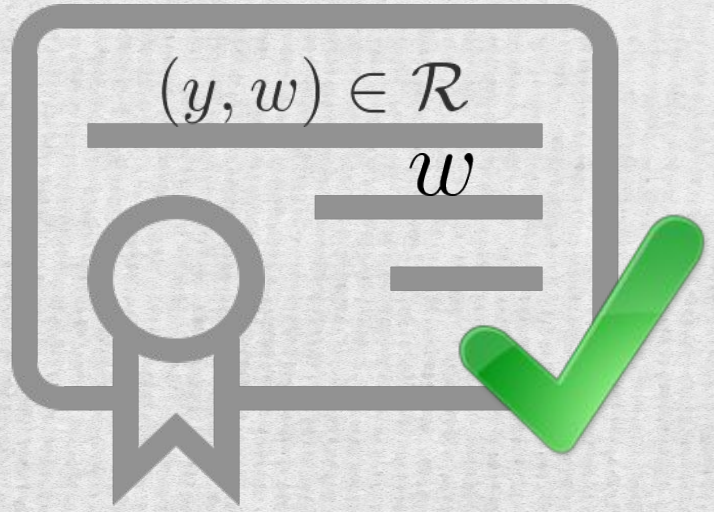
Adversary

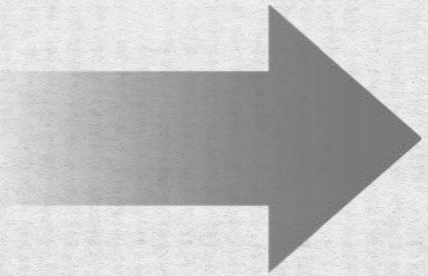
$\mathcal{E}$

crs

extractor

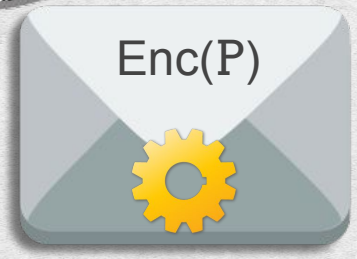
π

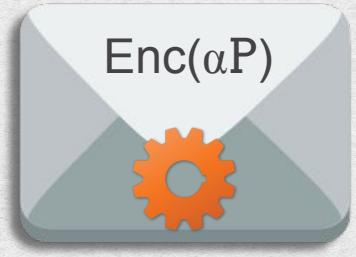


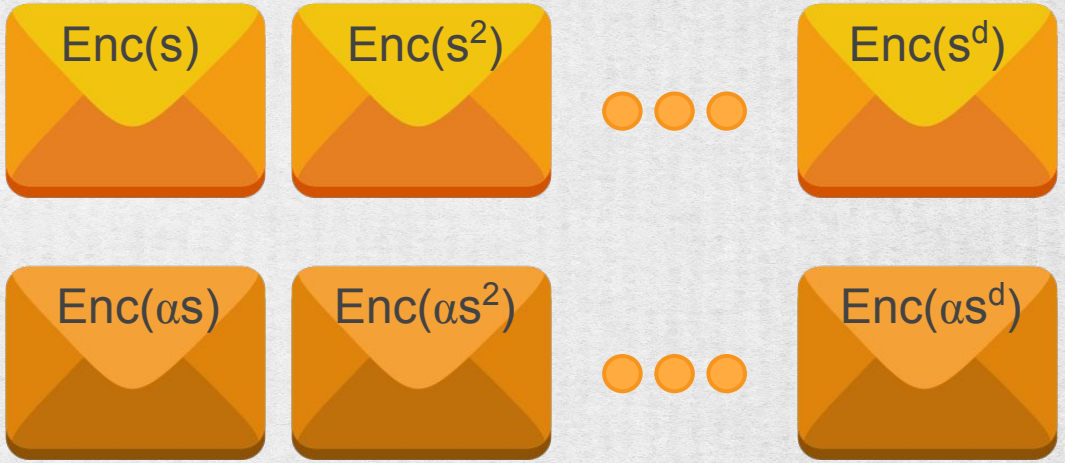


Assumption PKE: Power Knowledge of Exponent

d-PKE

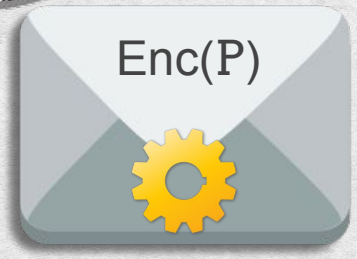


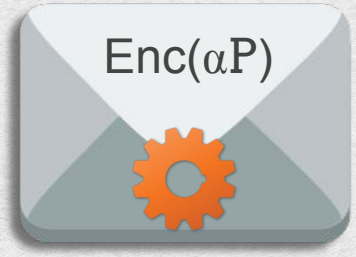


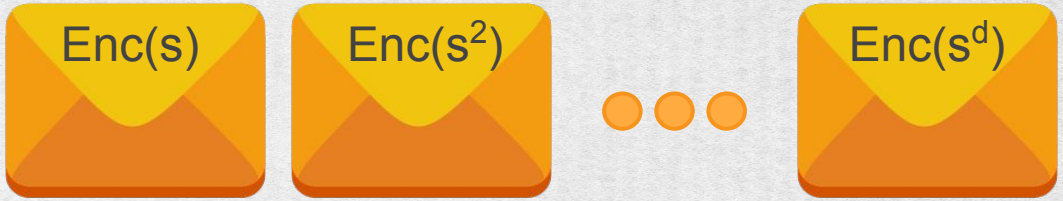


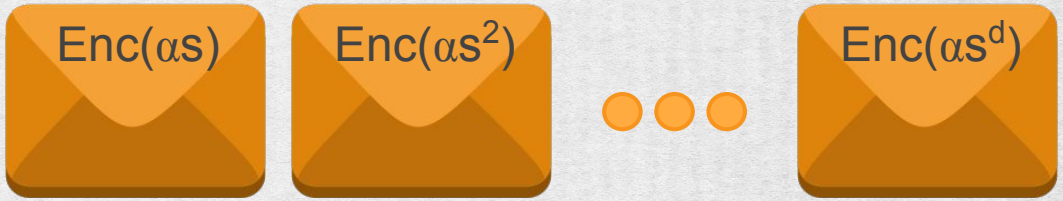
Assumption PKE: Power Knowledge of Exponent

d-PKE

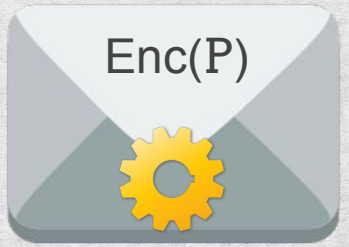








$\mathcal{E}$



$= \mathrm{Enc}(\Sigma\, p_i s^i)$

Assumption on Discrete Log Encoding

d-PKE

$E(g^P)$ $E(g^{\alpha P})$

g^s g^{s^2} ••• g^{s^d}

$g^{s\alpha}$ $g^{\alpha s^2}$ ••• $g^{\alpha s^d}$

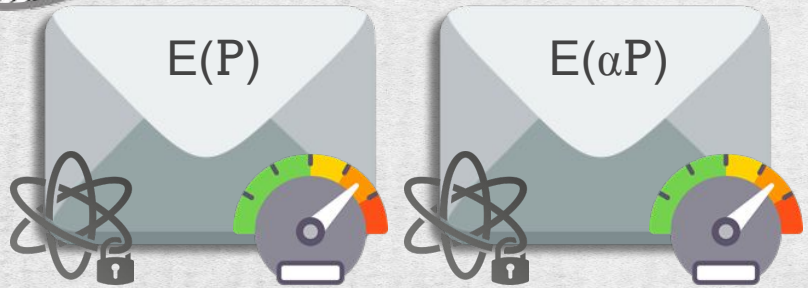
$\mathcal{E}$

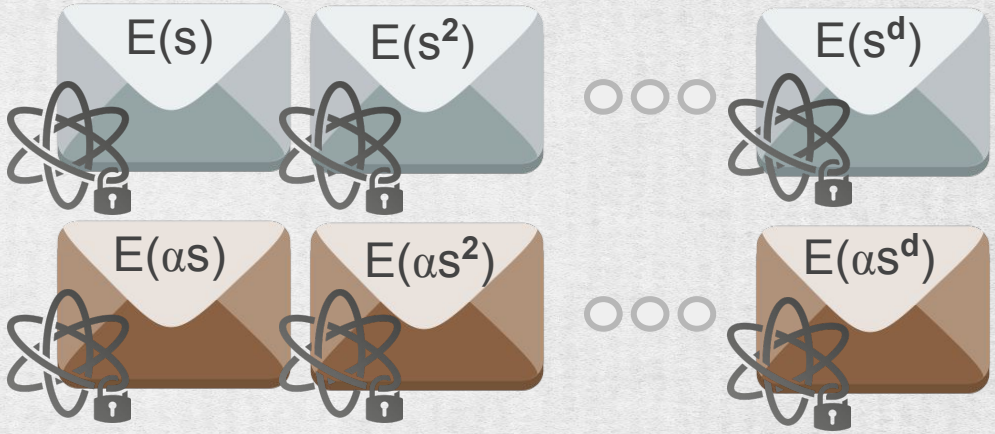
g^{p_1} g^{p_2} ••• g^{p_d}

$g^P = g^{\sum p_i s^i}$

Assumptions on Lattice Encodings

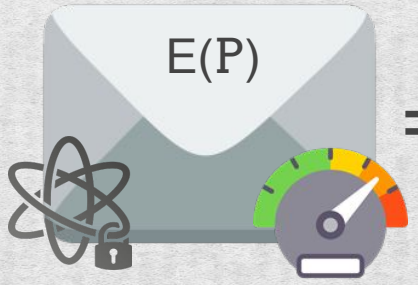
d-PKE





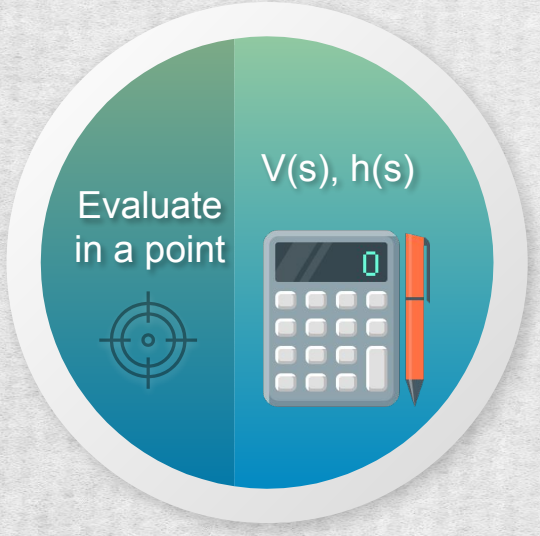
ε

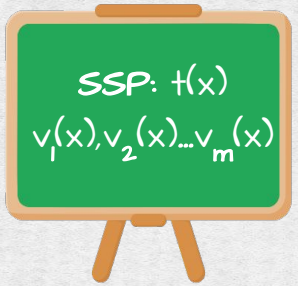
p_1 p_2 ••• p_d



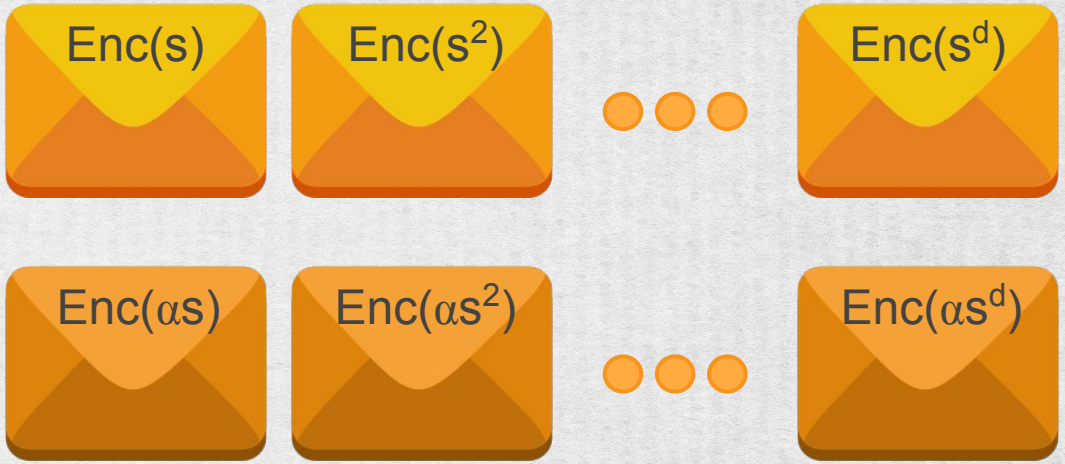
$= E(\Sigma\, p_i s^i)$

Setup and Proof

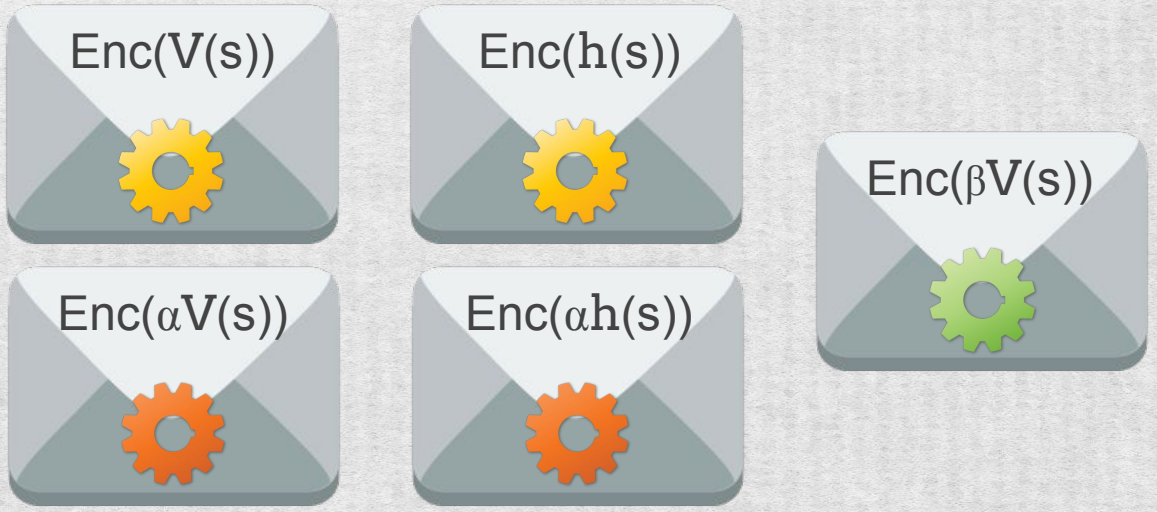




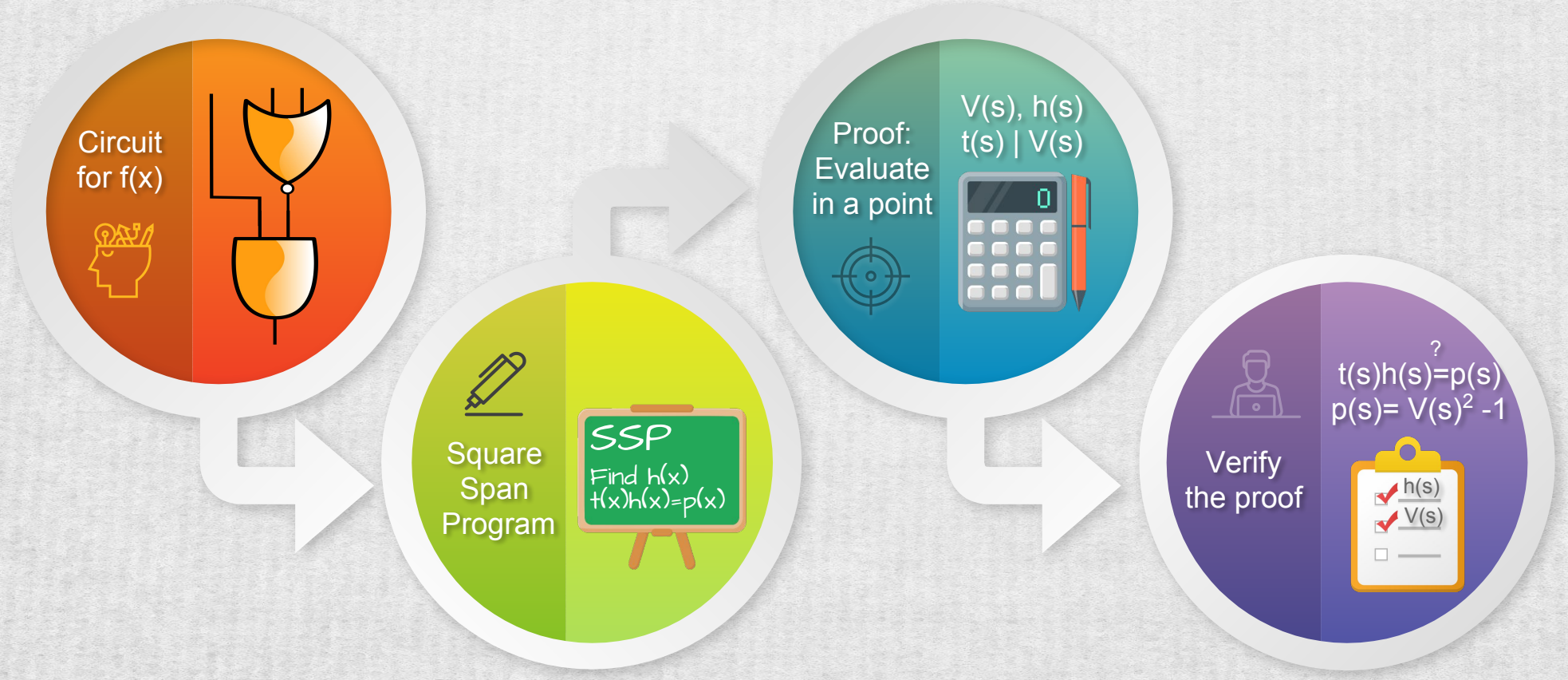
$\{ \mathrm{Enc}(\beta v_i(s)),\ \mathrm{Enc}(\beta) \}_{i=1,m}$



=



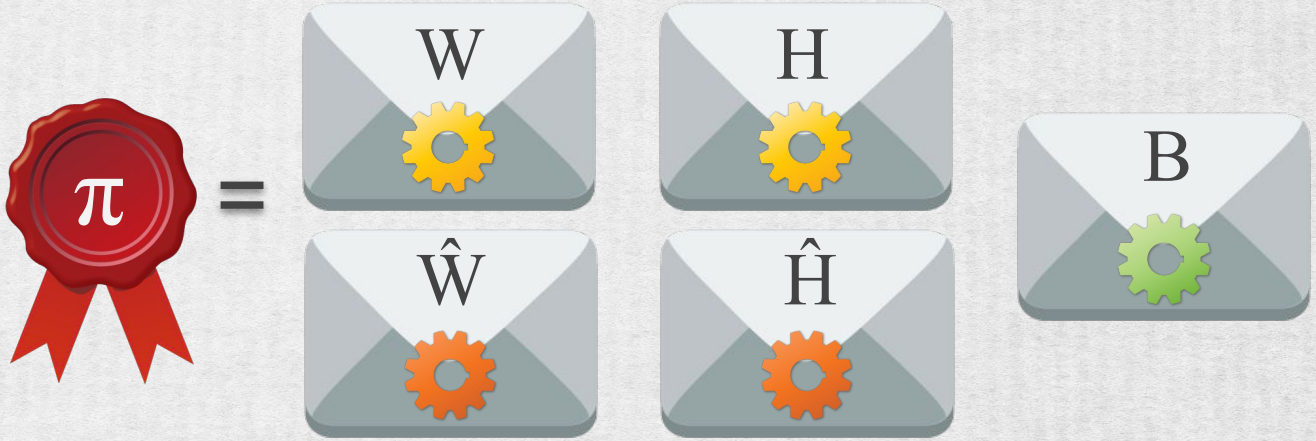
SNARK: overview of Toolchain



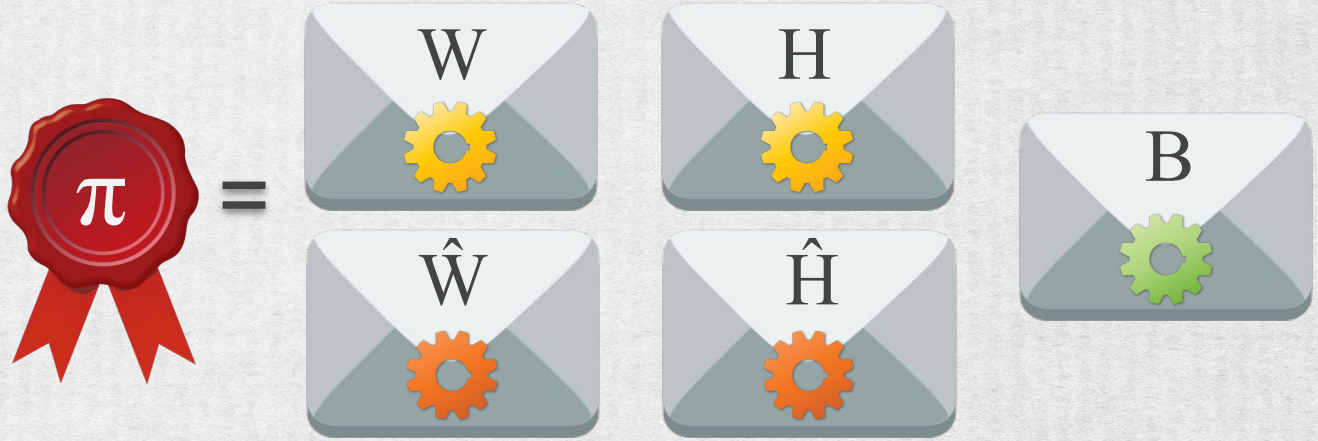
SNARK: verification

Fast
Verification

Proving on top of SSP: Verifier

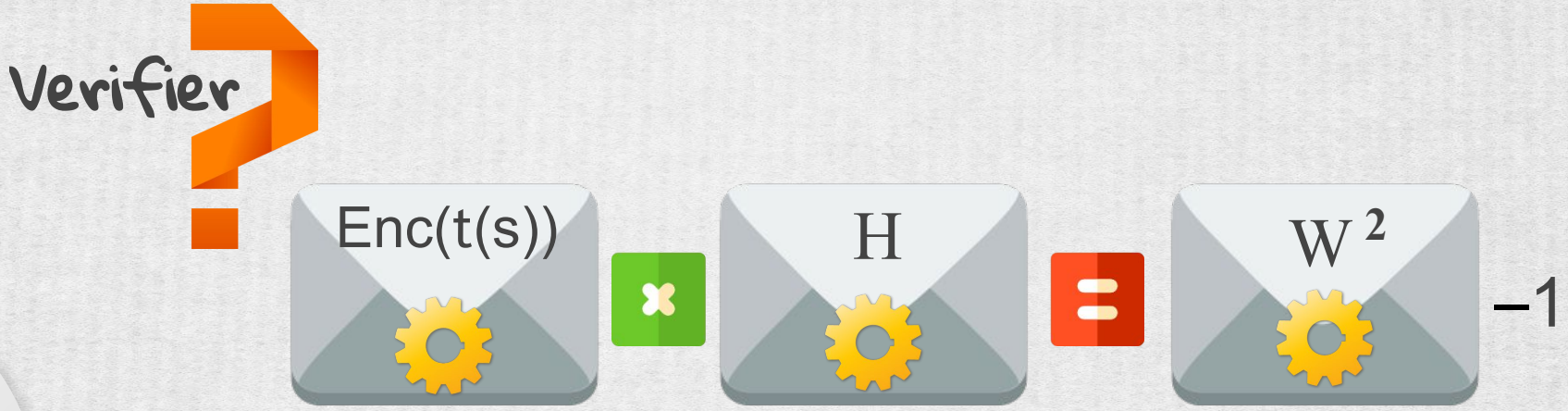


Proving on top of SSP: Verifier



Verifier ?

Proving on top of SSP: Verifier



Encoding:

- *linearly* homomorphic
- quadratic root detection
- image verification

Discrete Logarithm Encoding

$$Enc(s) = g^s$$

$$\langle g \rangle = \mathbb{G}, \langle \tilde{g} \rangle = \tilde{\mathbb{G}}$$
$$e : \mathbb{G} \times \mathbb{G} \to \tilde{\mathbb{G}}$$
$$e(g^a, g^b) = \tilde{g}^{ab}$$

Quadratic root detection

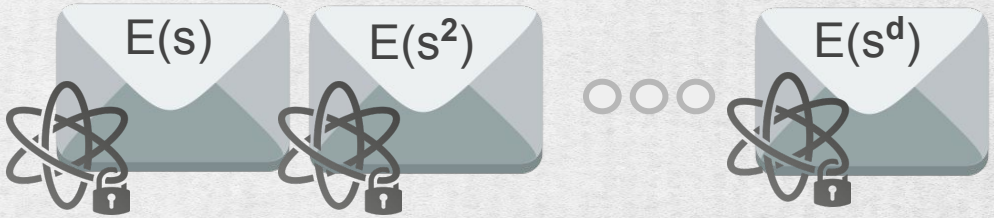
$$t(s)h(s) \stackrel{?}{=} p(s)$$

$$e(g^{t(s)}, g^{h(s)}) \stackrel{?}{=} e(g^{p(s)}, g)$$

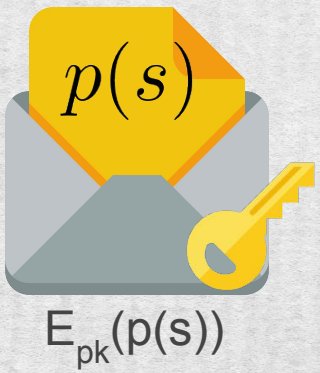
Encryption Encoding

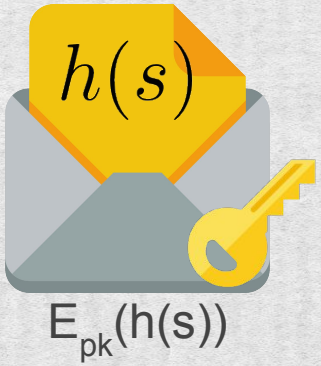
Encryption: $E(m) = c$

Decryption: $D(c) = m$



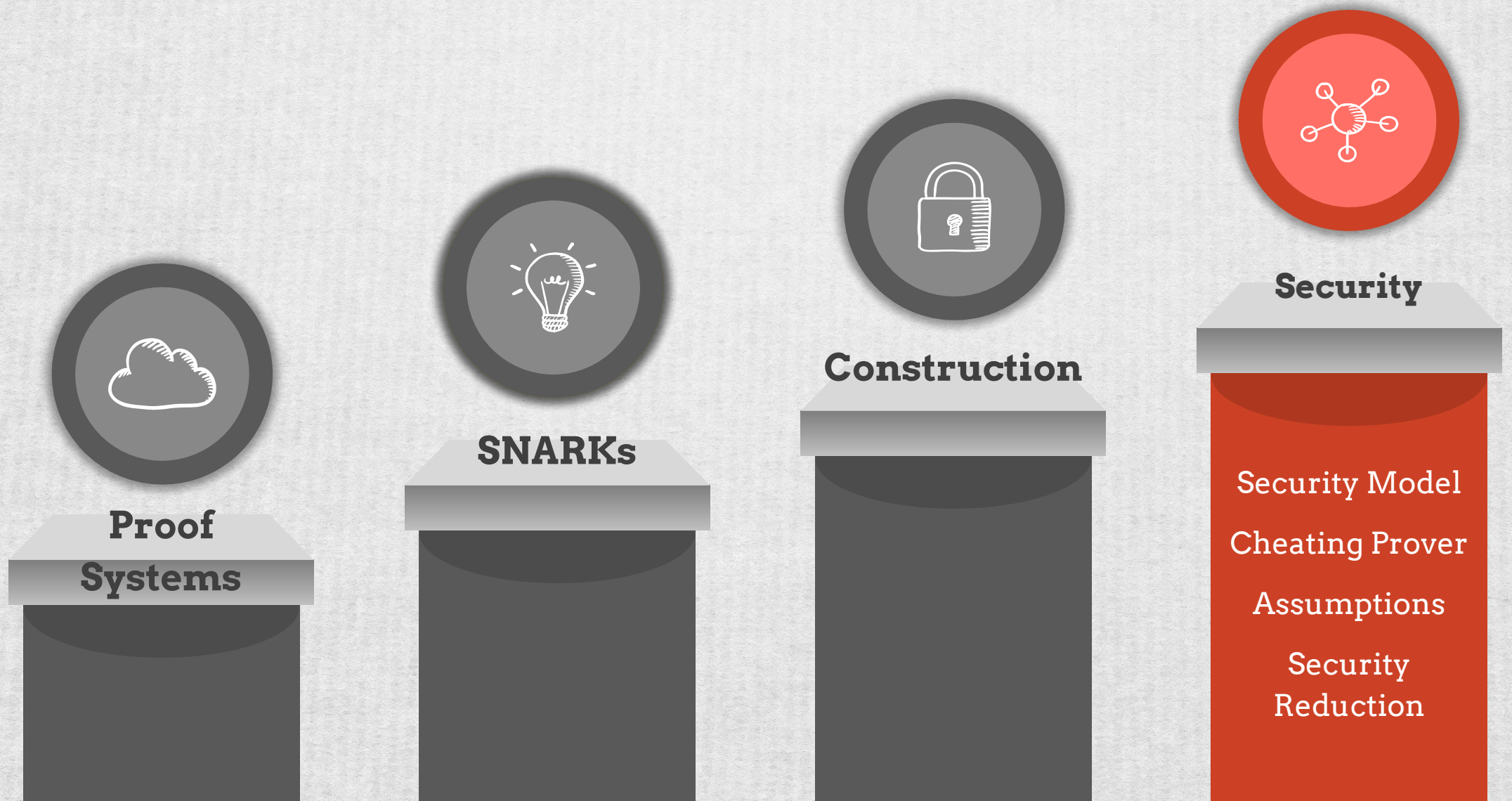
Quadratic root detection





$$t(s)h(s) \stackrel{?}{=} p(s)$$

Motivation

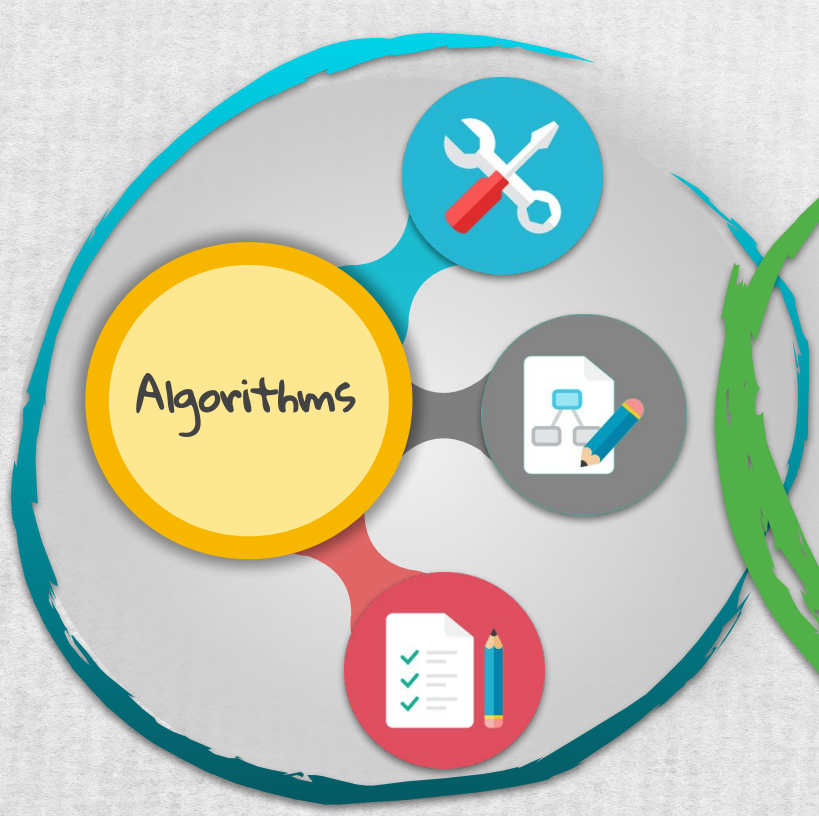

Proof Systems
SNARKs
Construction
Security
Security Model
Cheating Prover
Assumptions
Security Reduction

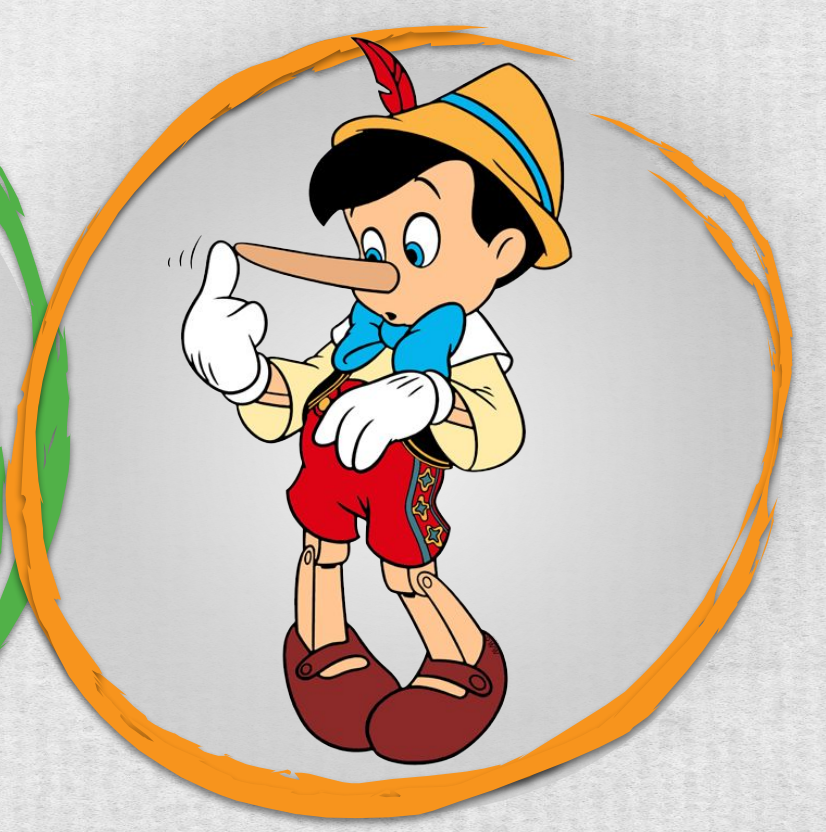
Formal Construction and Security

Protocol

Assumptions

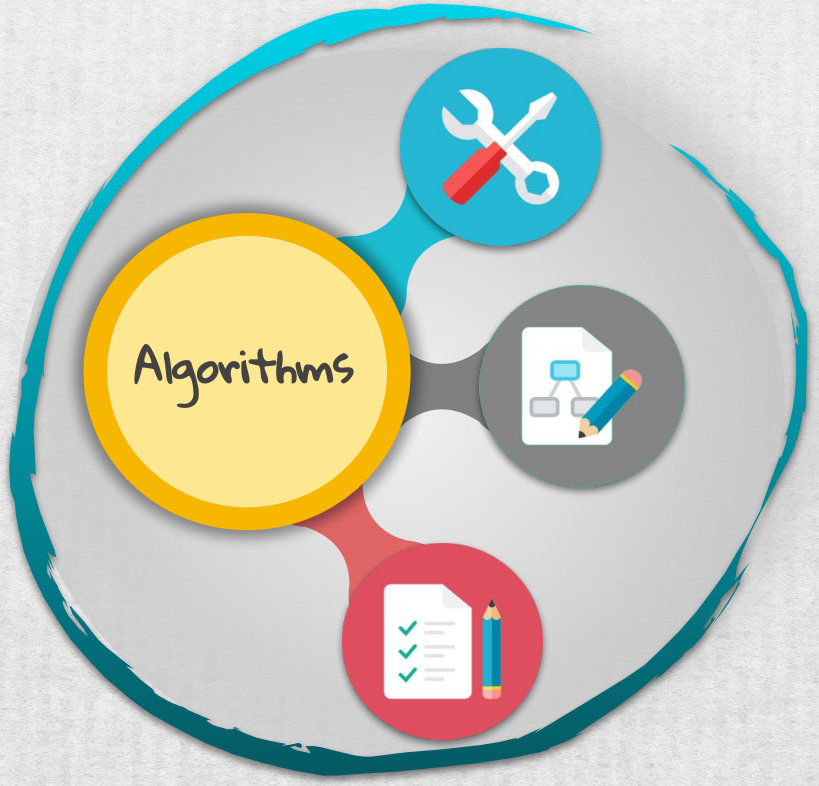
Security



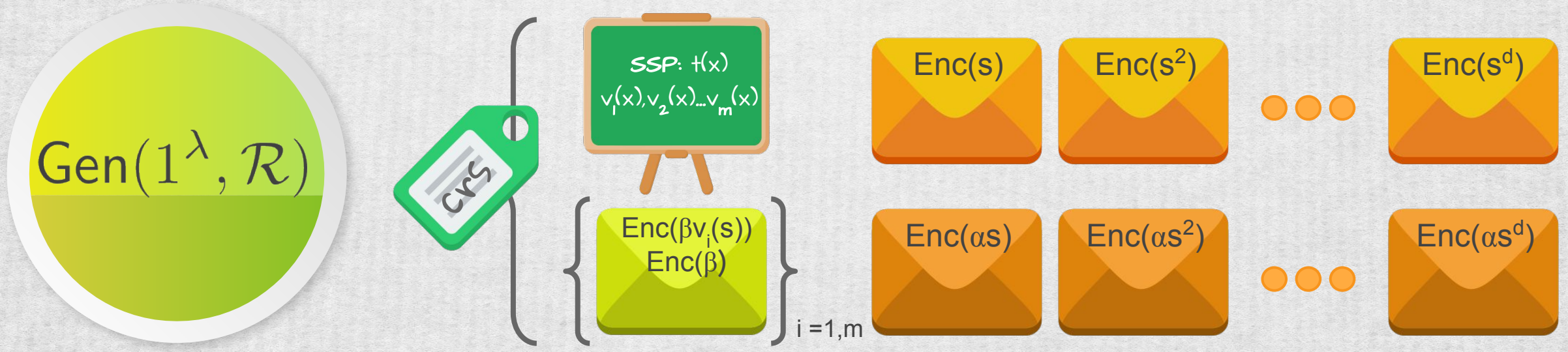


Protocol Description

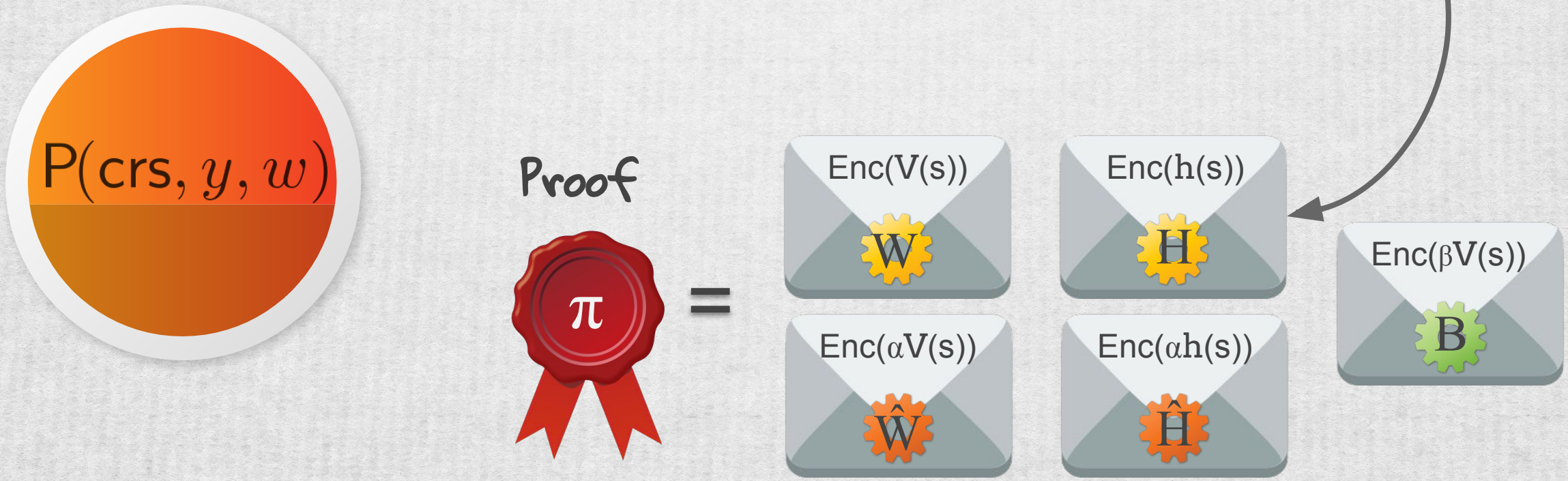
Protocol



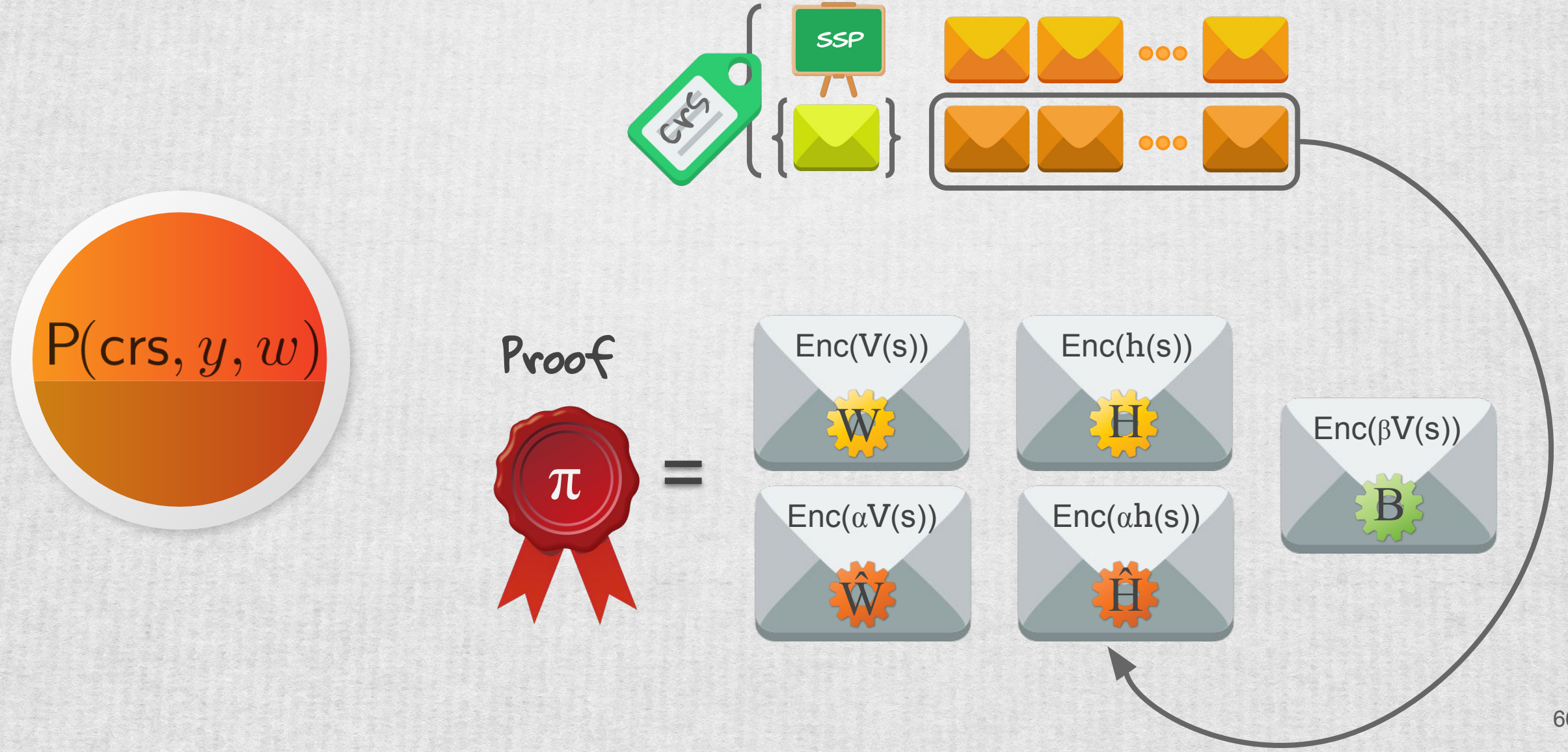
Setup Algorithm



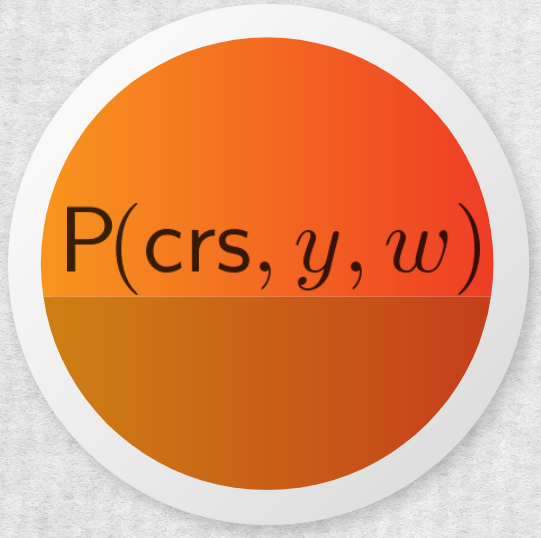
Prover Algorithm

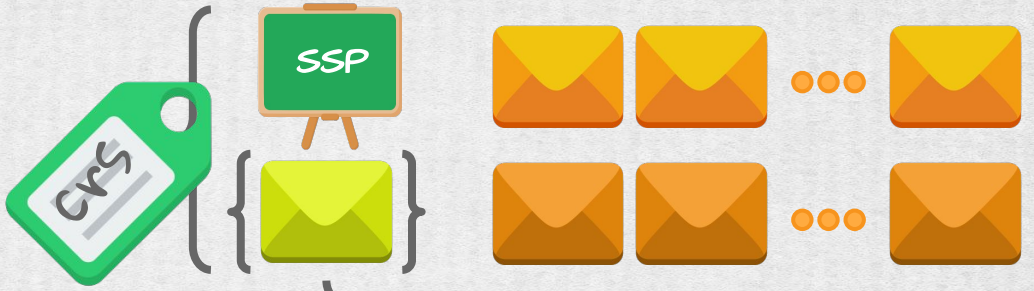


Prover Algorithm

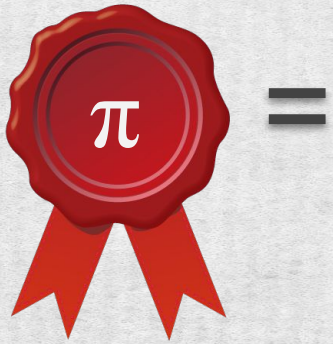


Prover Algorithm

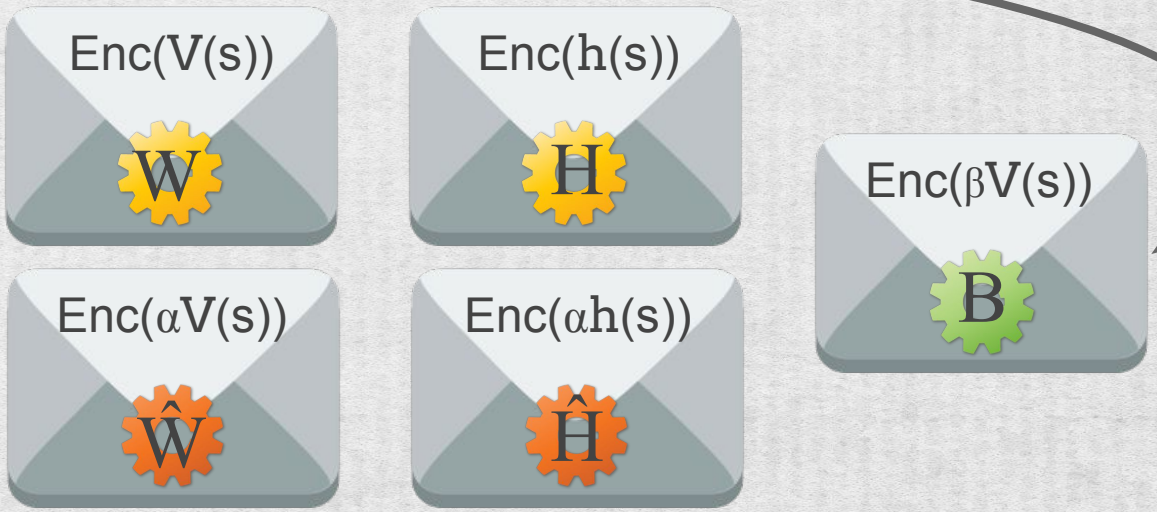




Proof

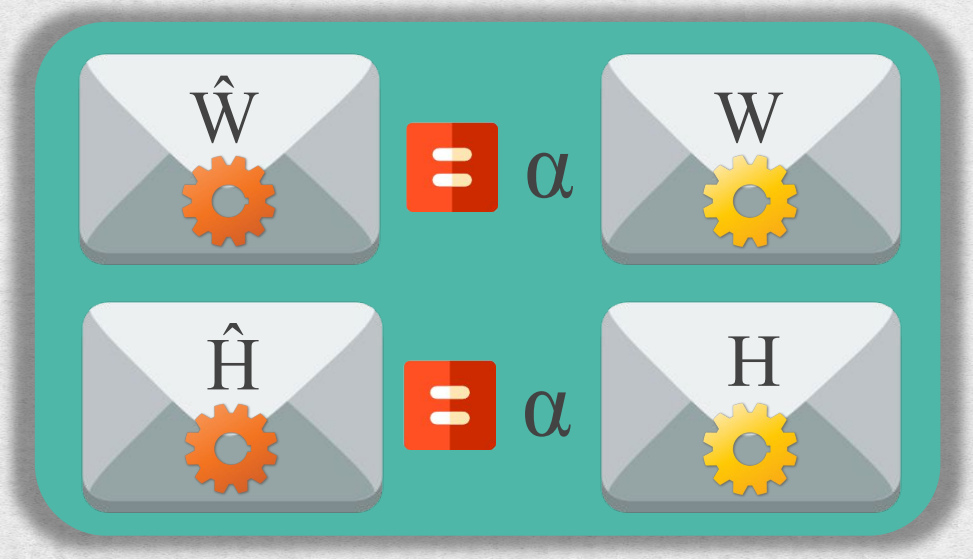


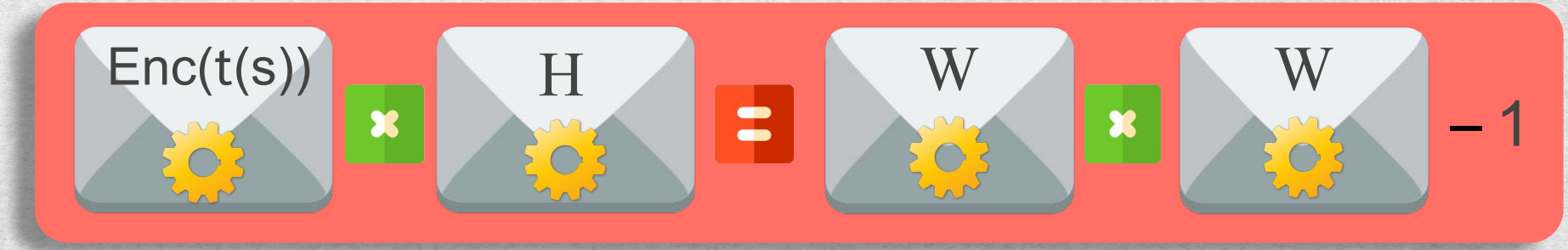
=



Verifier Algorithm

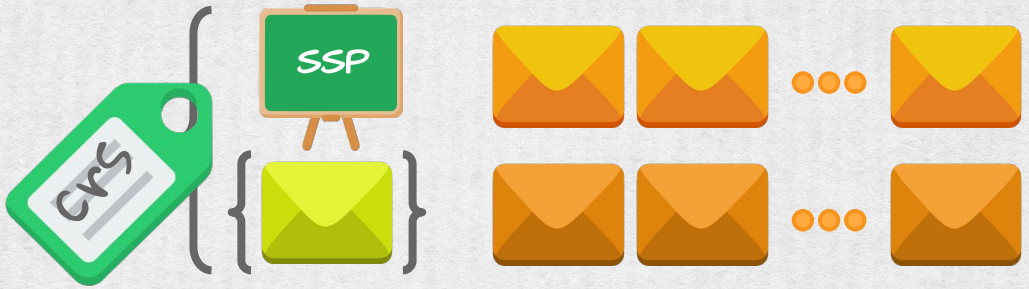
$V(\mathsf{vk}, y, \pi)$





Review of the Protocol (Algorithms)

$\mathsf{Gen}(1^\lambda, \mathcal{R})$



$\mathsf{P}(\mathsf{crs}, y, w)$

$\pi = \hat{W}\ \hat{H}\ W\ H\ B$

$\mathsf{V}(\mathsf{vk}, y, \pi)$

$\hat{W} = \alpha W$

$\hat{H} = \alpha H$

$B = \beta W$

$t(s) \cdot H = W \cdot W - 1$

Cheating Strategy

Security

Security Reduction in Cryptography

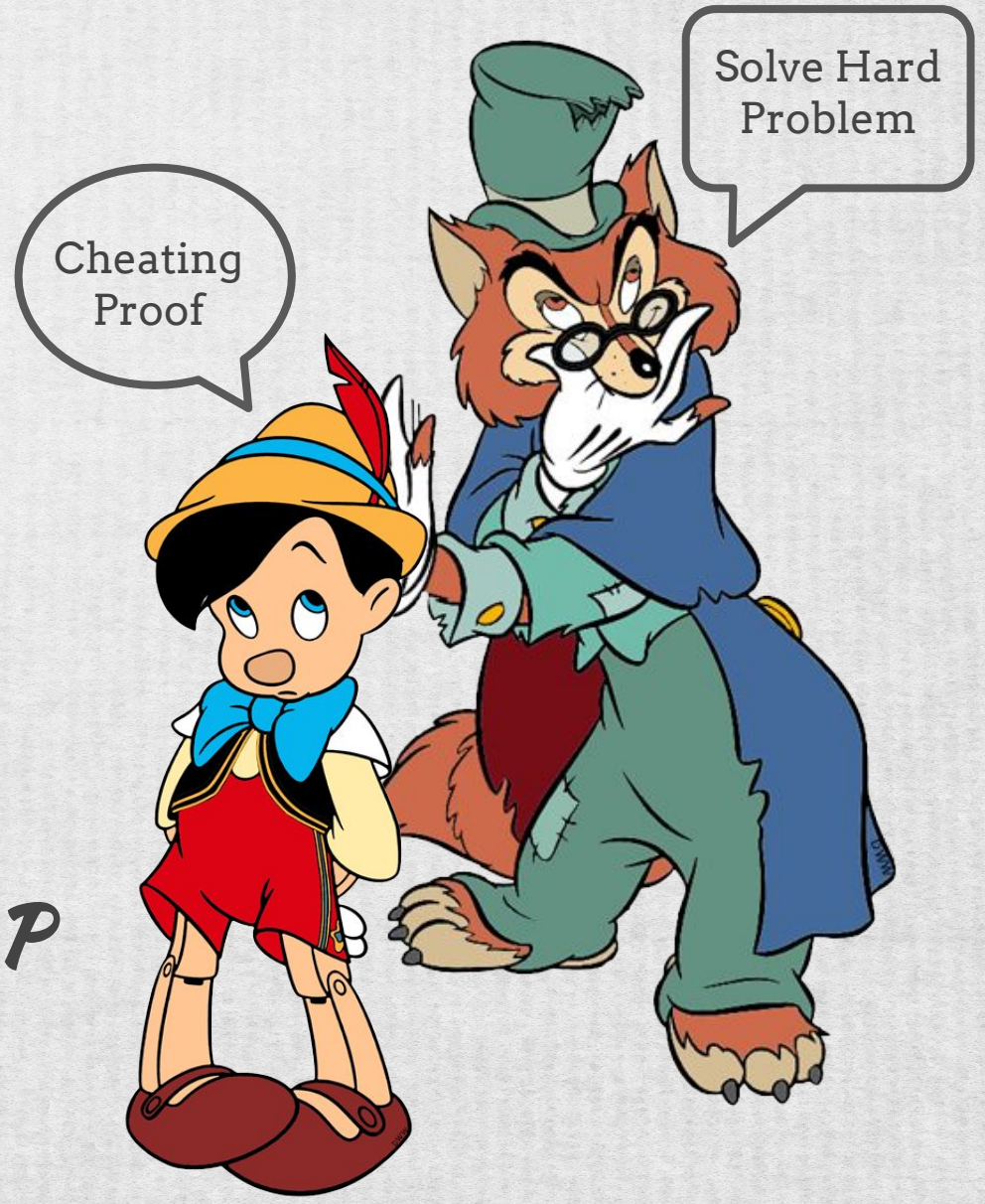
- **Reduction:**

  Assuming $\mathcal{P}$ is hard $\rightarrow$ Protocol security

- **Contraposition:**

  Breaking the protocol $\rightarrow$ Solution for $\mathcal{P}$

Successful Adversary $\rightarrow$ Solution for $\mathcal{P}$



Security Reduction

Assumption (Hard Problem)

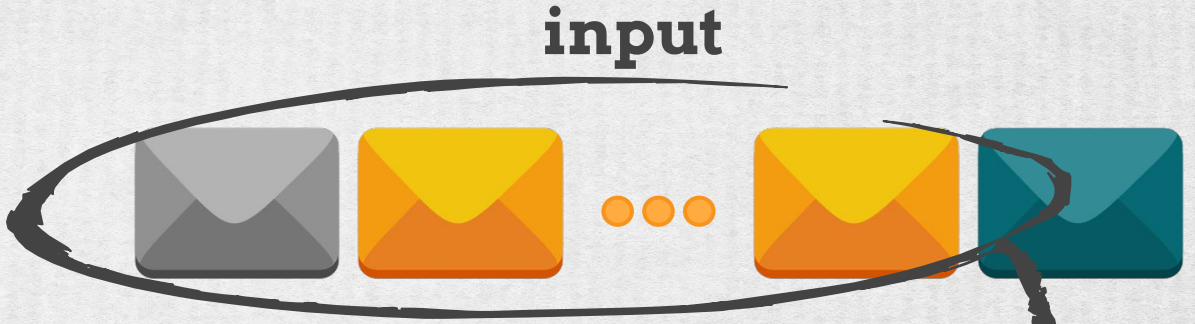
input

?

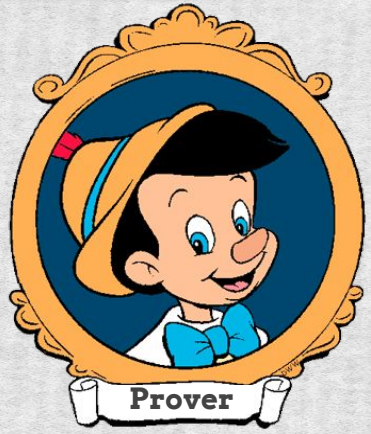
Cheating Prover

Security Reduction

Assumption (Hard Problem)



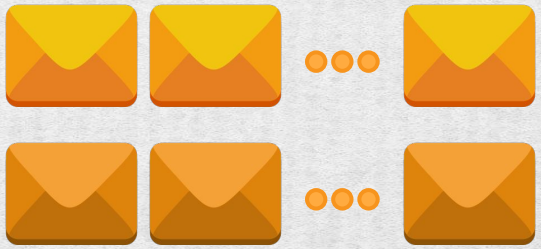
Cheating Prover

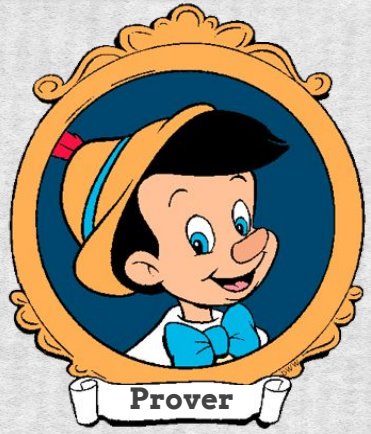


Security Reduction

Assumption (Hard Problem)

input





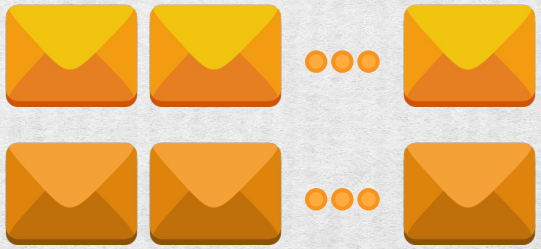
=

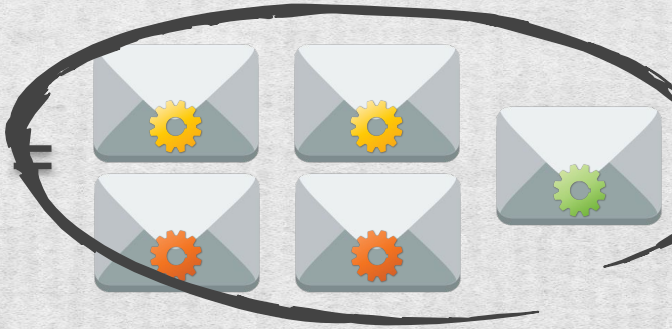
Security Reduction

Assumption (Hard Problem)

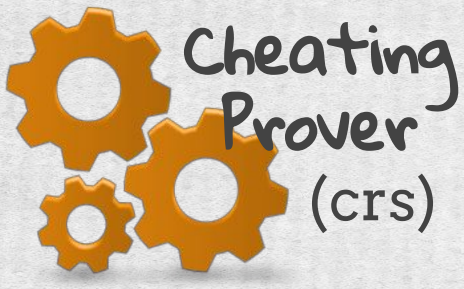
input

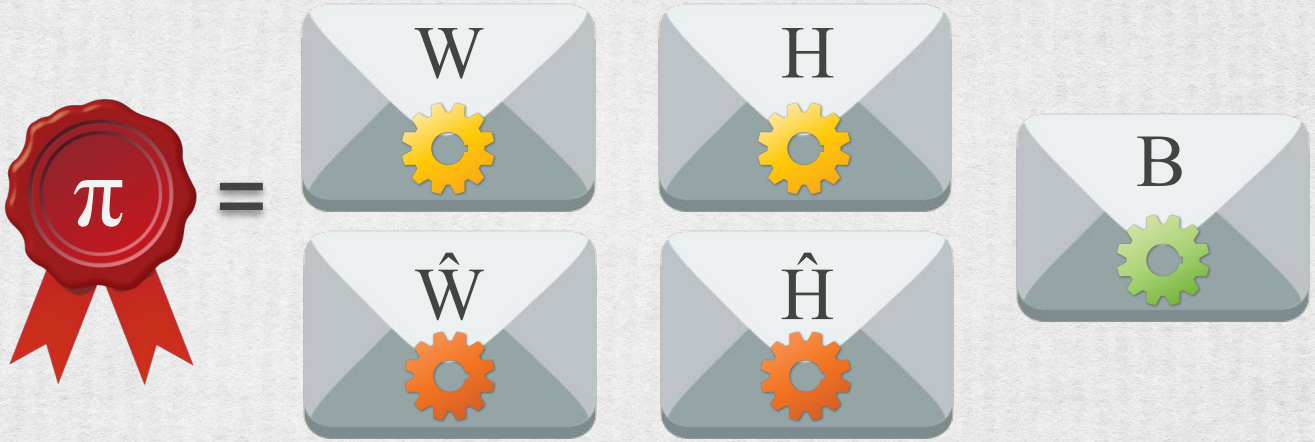
output



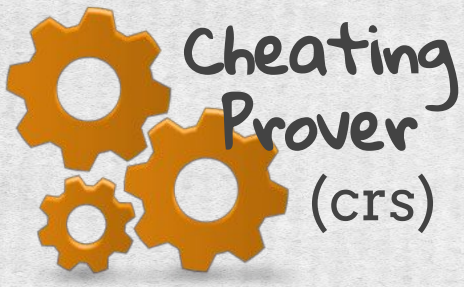


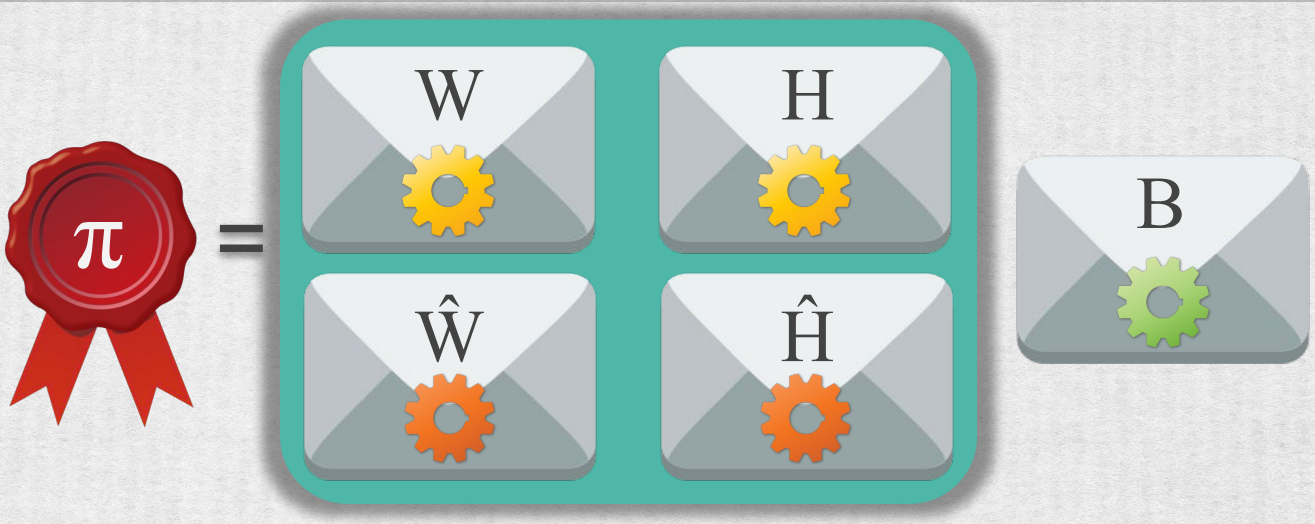
Cheating Strategy



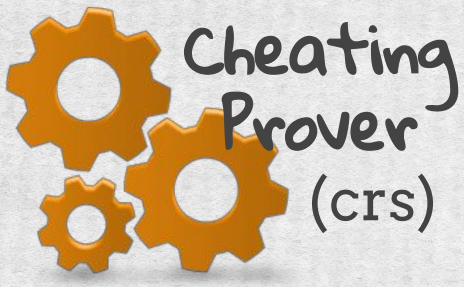


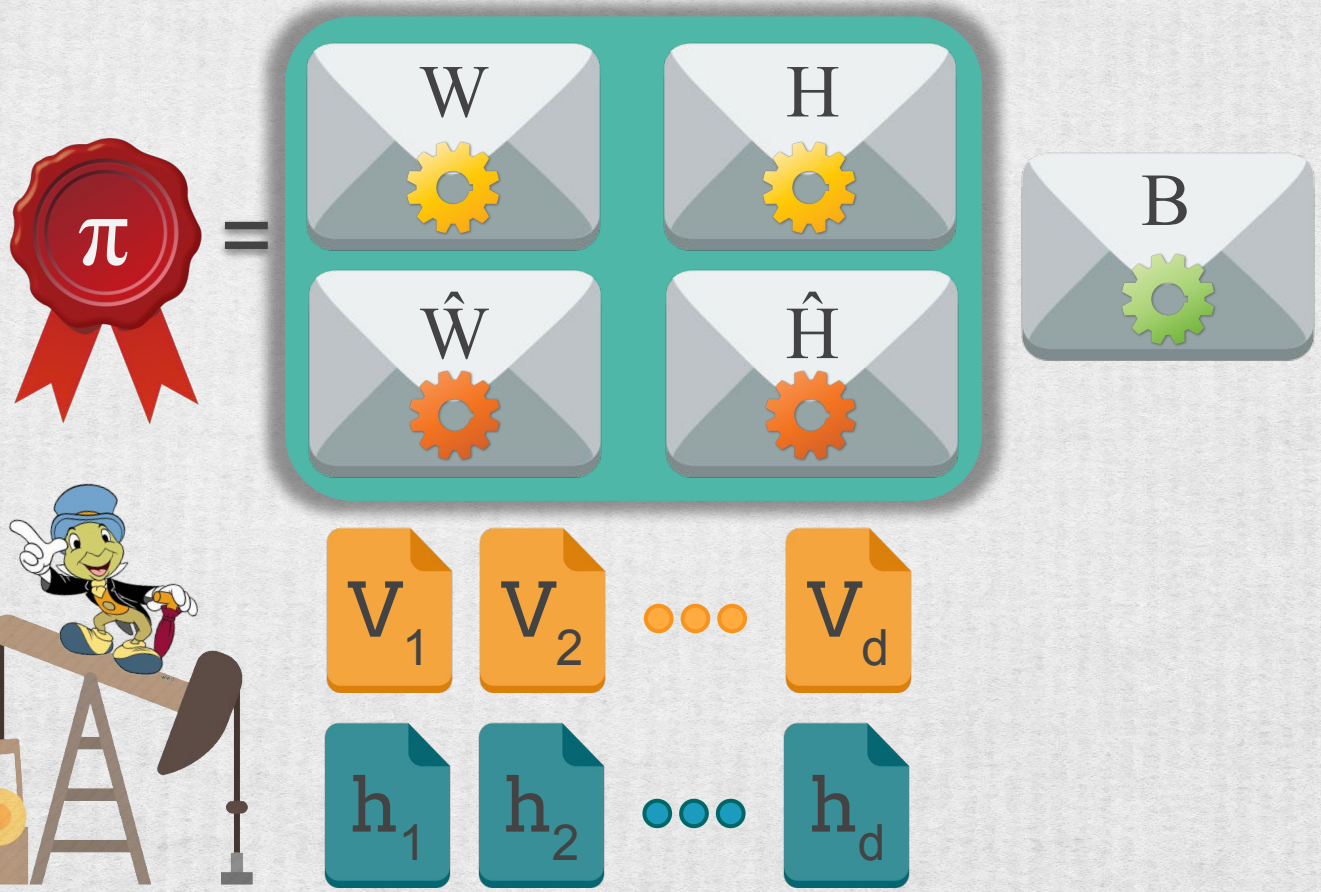
Cheating Strategy



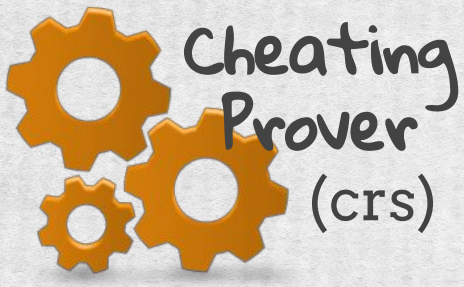


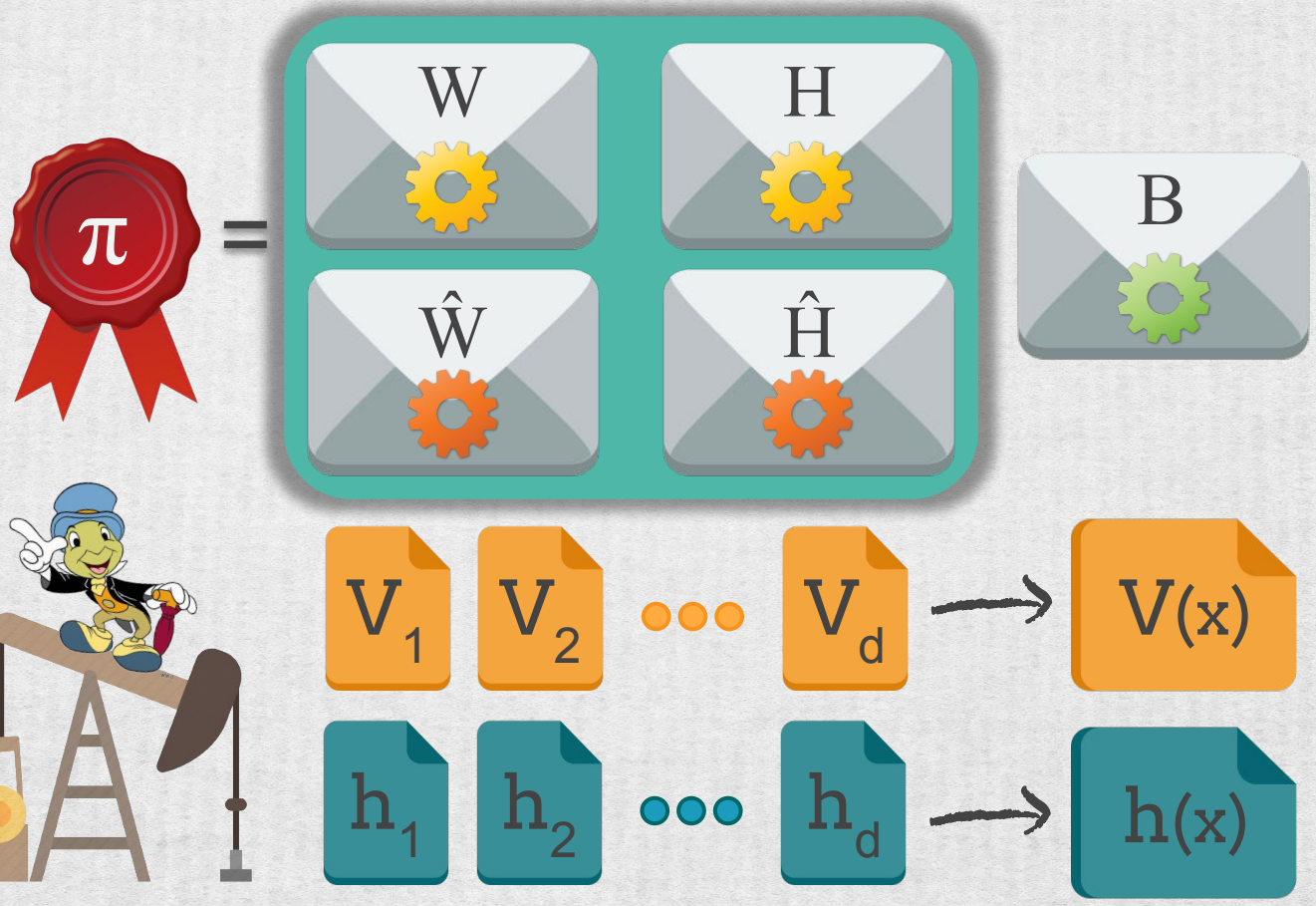
Cheating Strategy



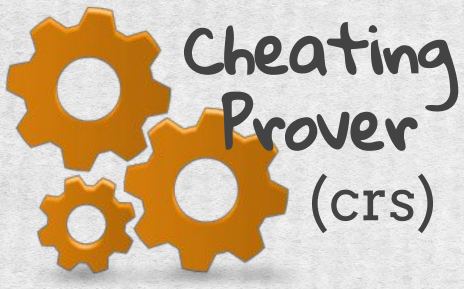


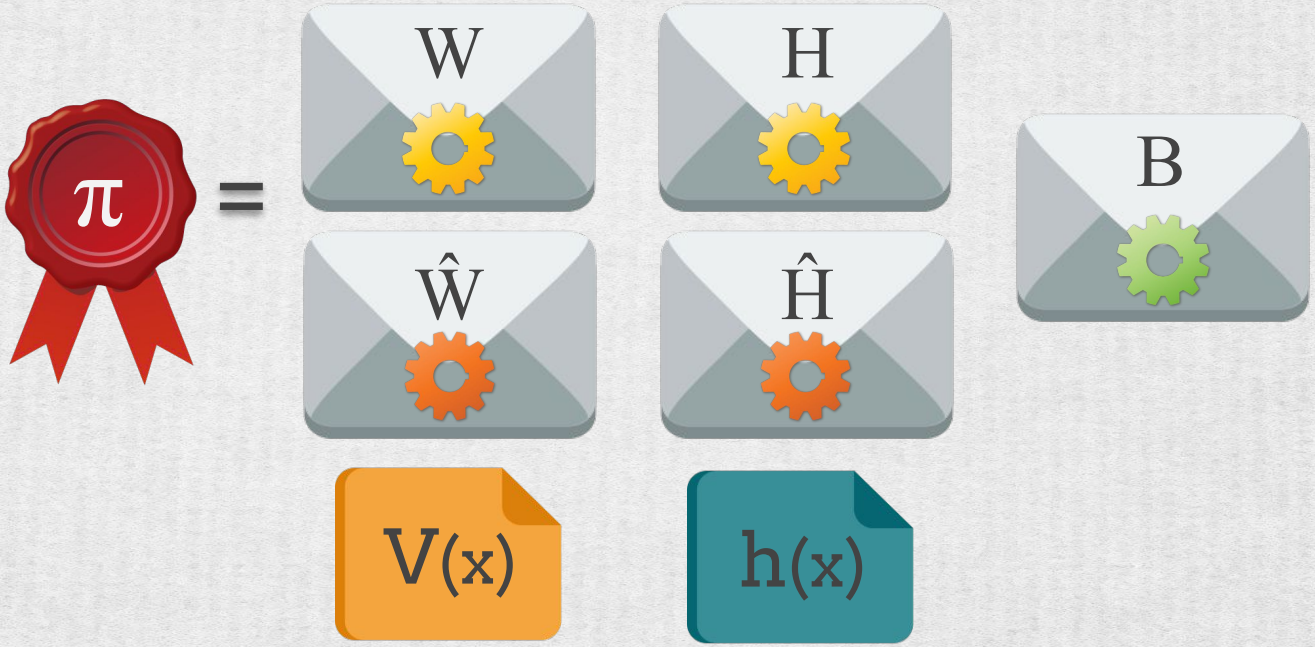
Cheating Strategy



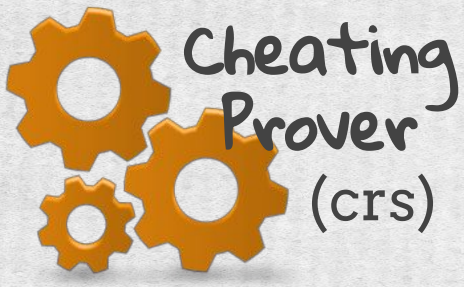


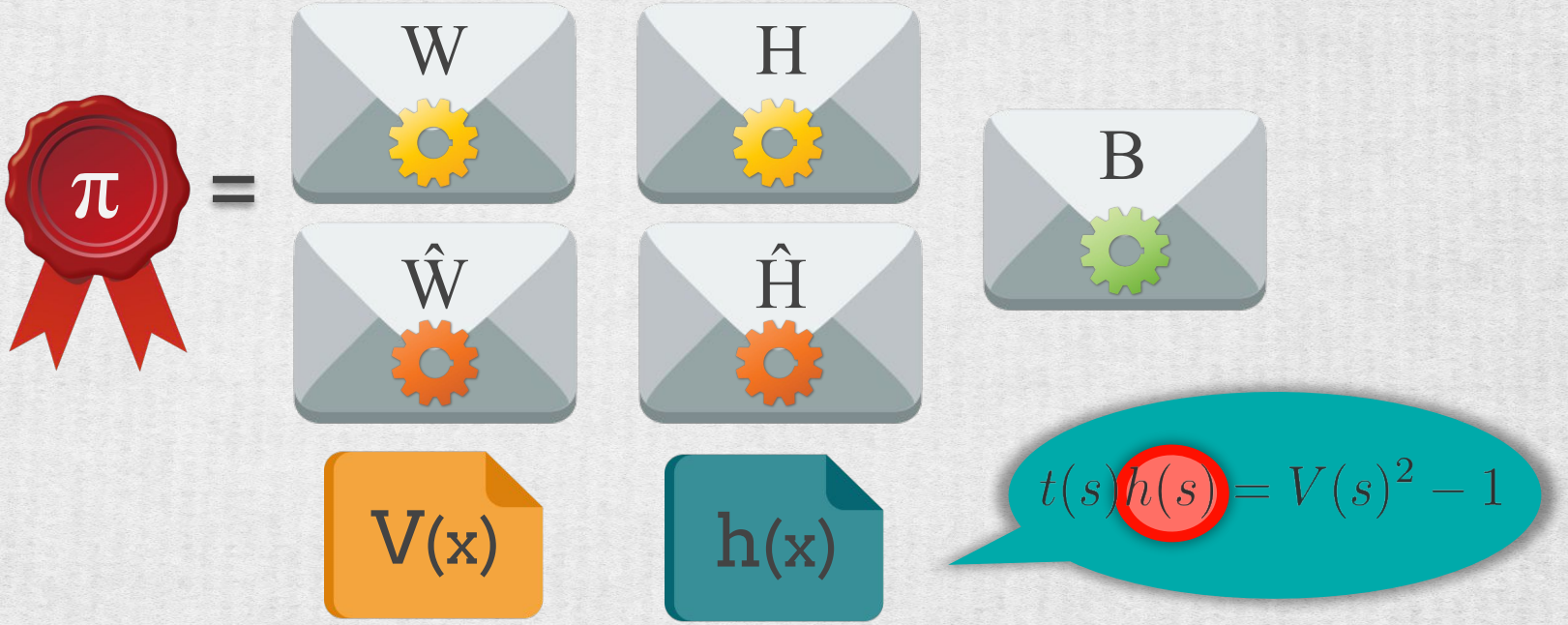
Cheating Strategy





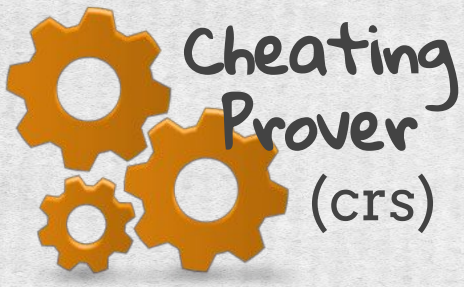
Division does not hold





- $t(x)h(x) \neq V^2(x) - 1$, but

Invalid linear combination



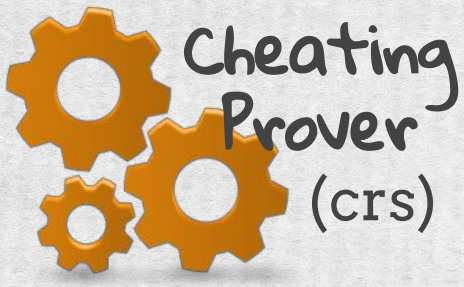
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

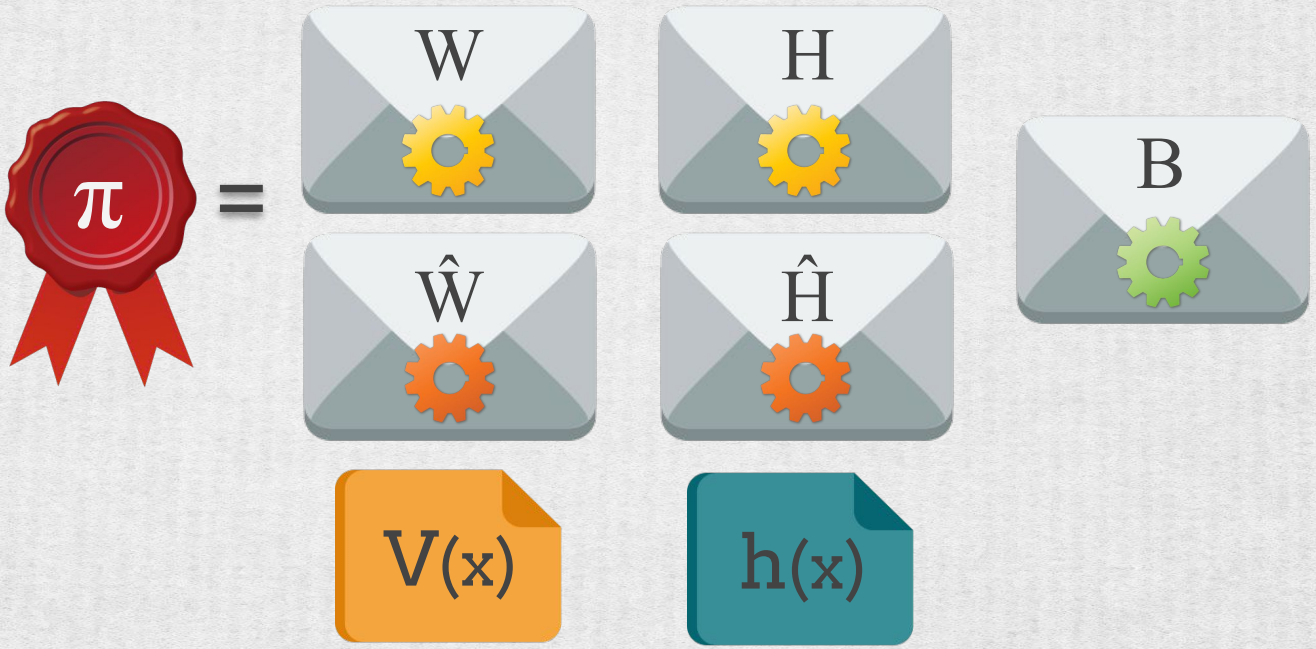
V(x)

h(x)

- $\mathrm{t(x)h(x)} \neq V^2(x) - 1$, but $\mathsf{Enc}(t(s))H = W^2 - 1$
- $\mathrm{V(x)} \notin \mathsf{Span}(v_1, \ldots v_m)$, but

Prover unable to compute higher degree polynomials





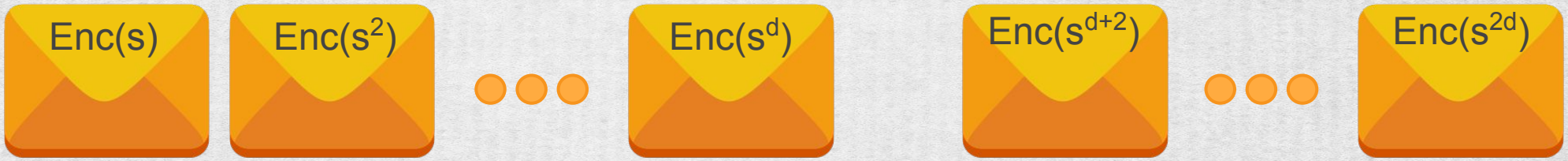
- $\mathrm{t(x)h(x)} \neq V^2(x) - 1$, but $\mathsf{Enc}(t(s))H = W^2 - 1$
- $\mathrm{V(x)} \notin \mathsf{Span}(v_1, \ldots v_m)$, but $B = \mathsf{Enc}(\beta V(s))$

Hard Problems

Assumptions

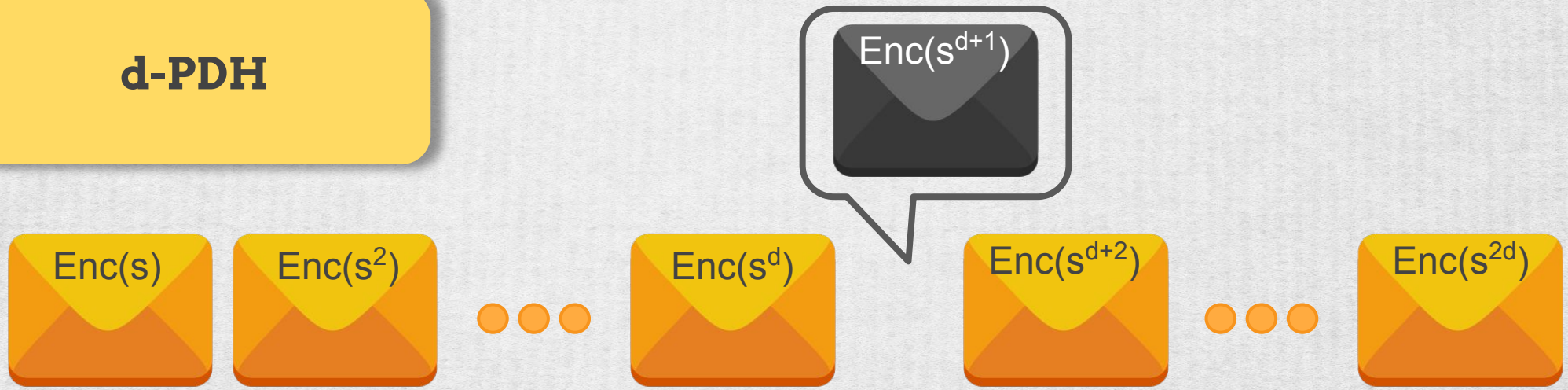
Assumption PDH: Power Diffie-Hellman

d-PDH



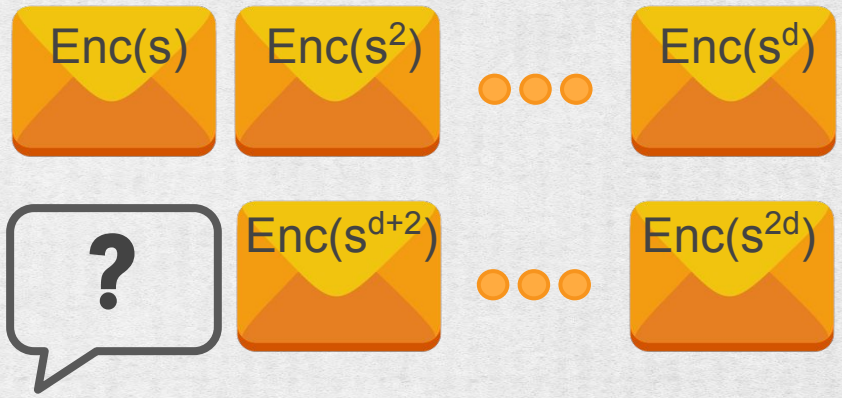
Assumption PDH: Power Diffie-Hellman

d-PDH



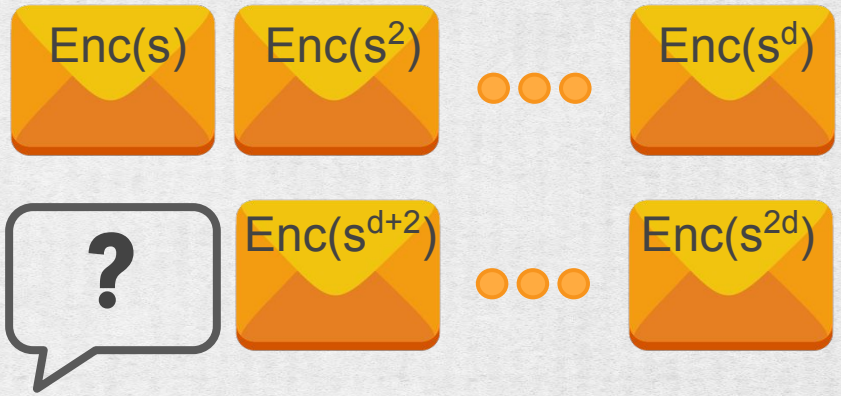
Security Reduction: Cheating Prover to d-PDH

d-PDH



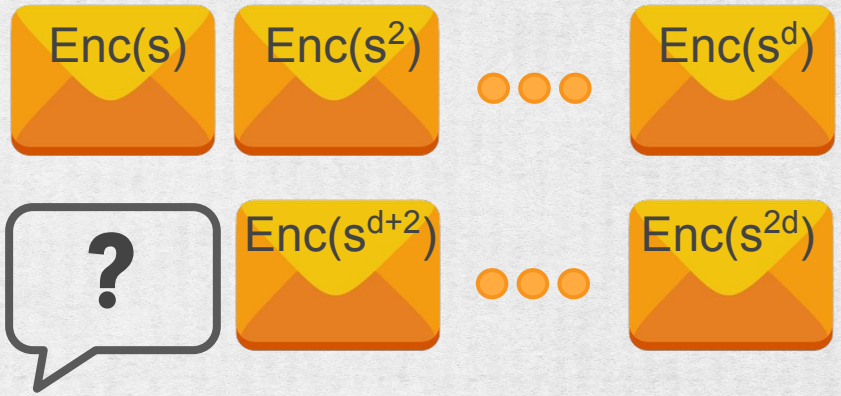
Security Reduction: Cheating Prover to d-PDH

d-PDH



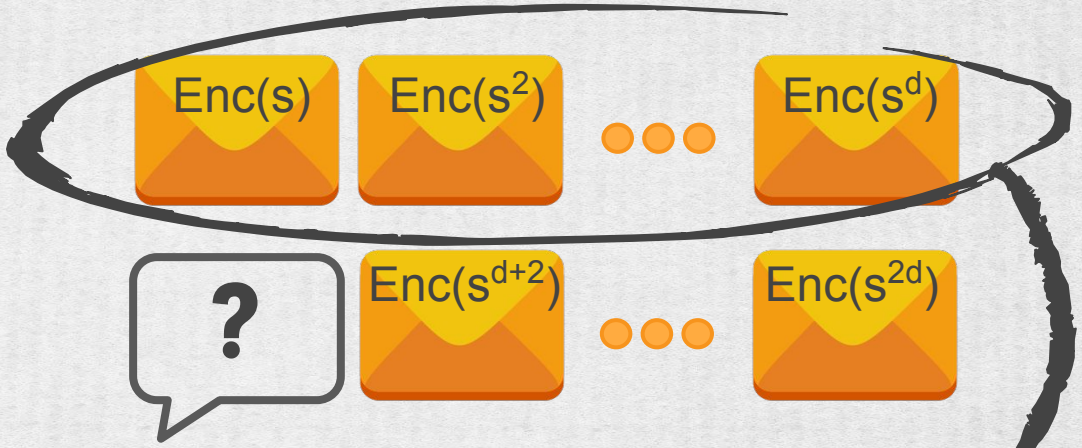
Security Reduction: Cheating Prover to d-PDH

d-PDH



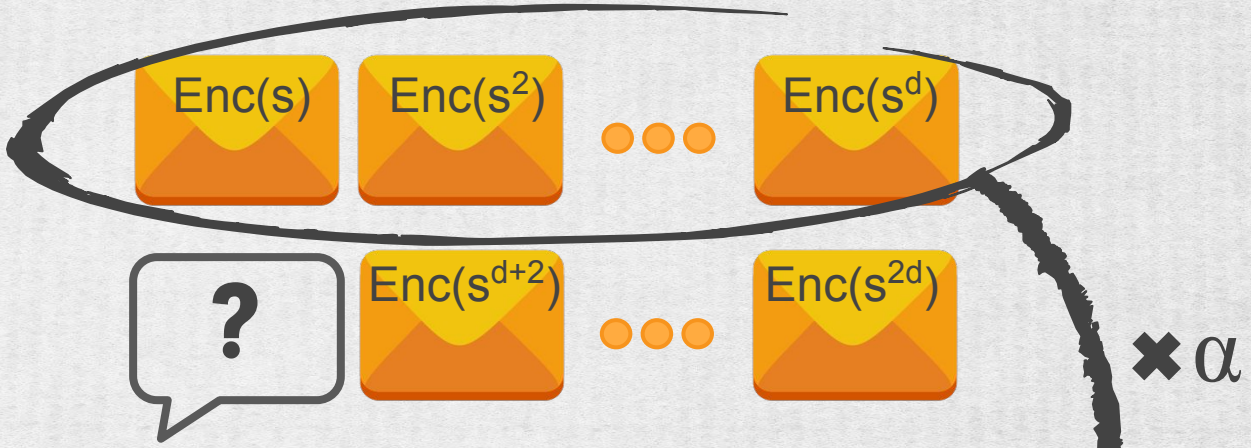
Security Reduction: Cheating Prover to d-PDH

d-PDH

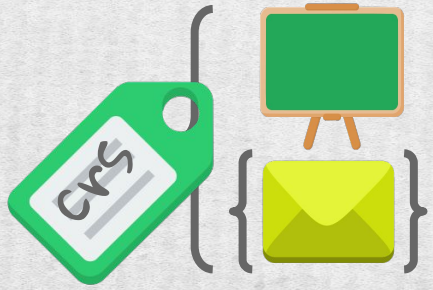


Security Reduction: Cheating Prover to d-PDH

d-PDH



Cheating Prover



Security Reduction: Cheating Prover to d-PDH

d-PDH

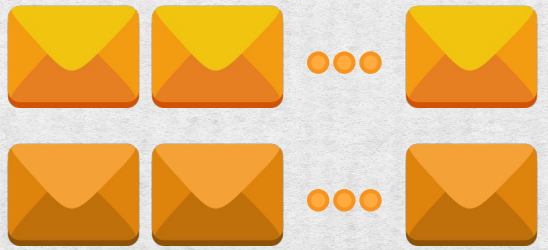
Enc(s) Enc(s^2) ••• Enc(s^d)

Enc(s^{d+2}) ••• Enc(s^{2d})

$\{v_i(x)\}_{i=1,m}$

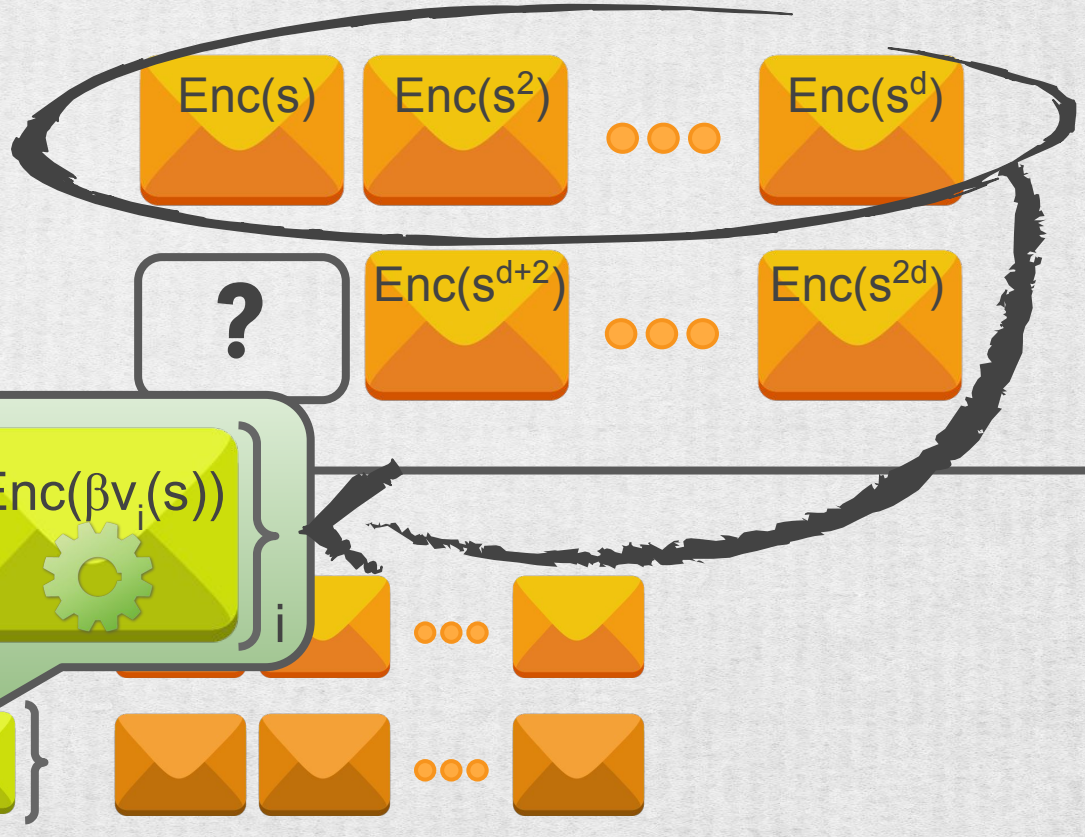
$t(x)$

Cheating Prover



Security Reduction: Cheating Prover to d-PDH

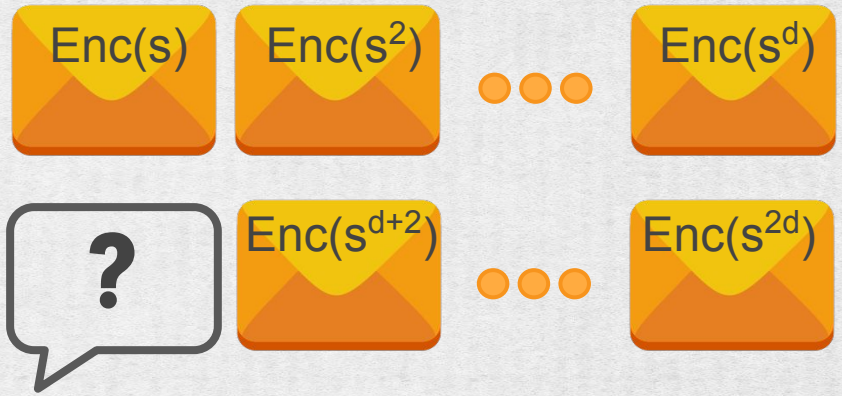
d-PDH



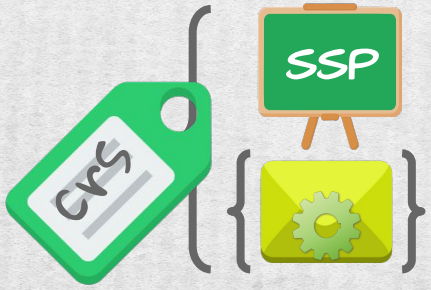
Cheating Prover

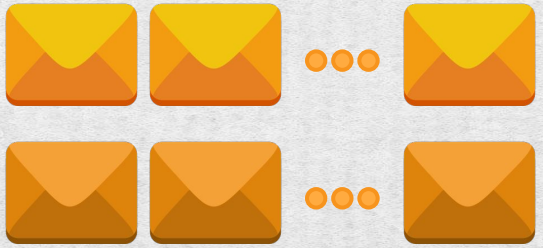
Security Reduction: Cheating Prover to d-PDH

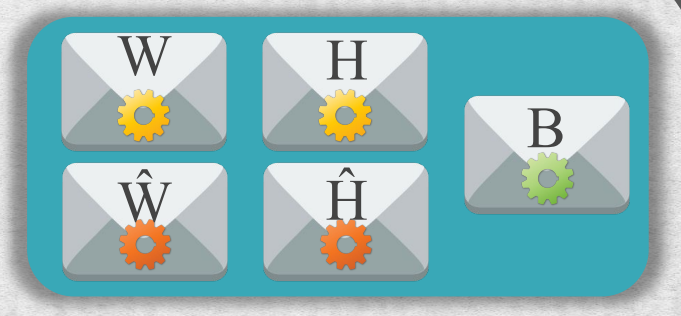
d-PDH



Cheating Prover

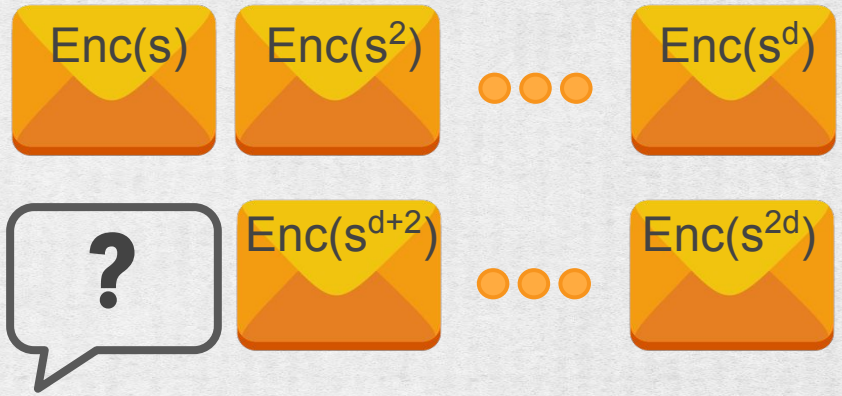




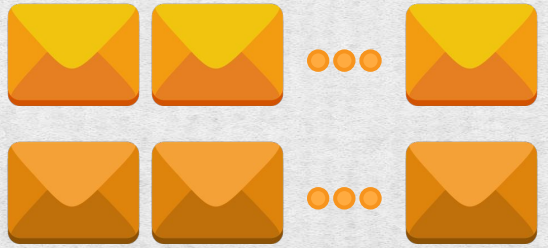


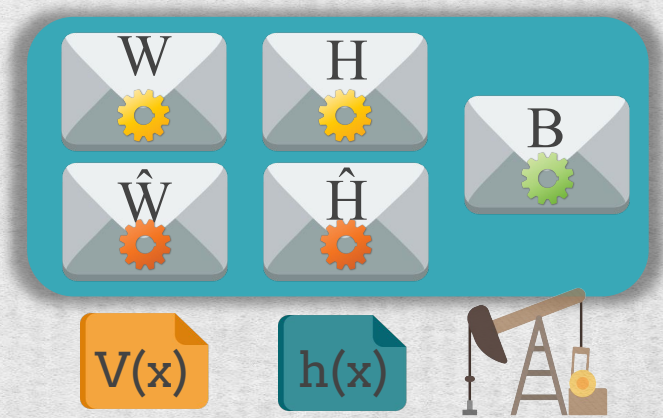
Security Reduction: Cheating Prover to d-PDH

d-PDH



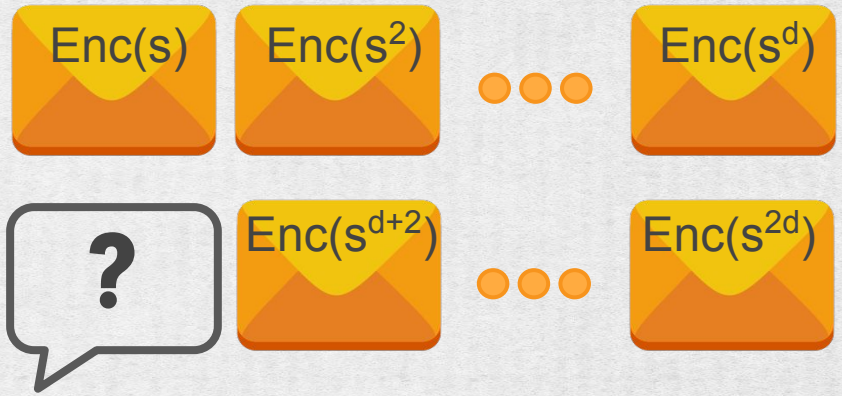
Cheating Prover



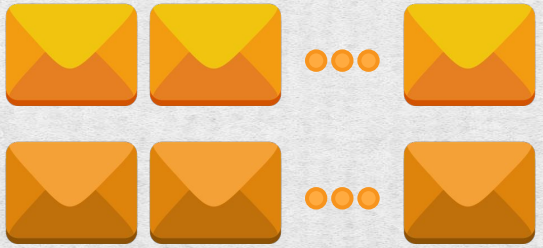


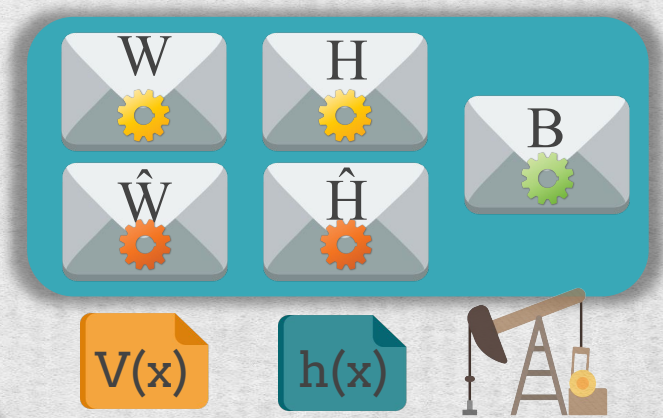
Security Reduction: Cheating Prover to d-PDH

d-PDH



Cheating Prover

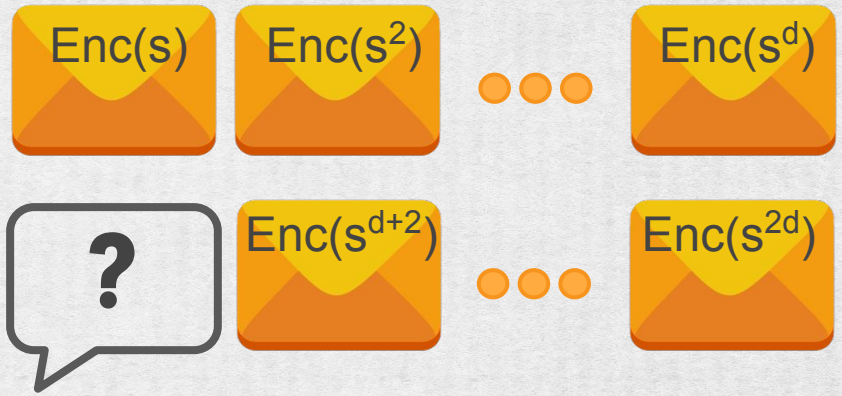


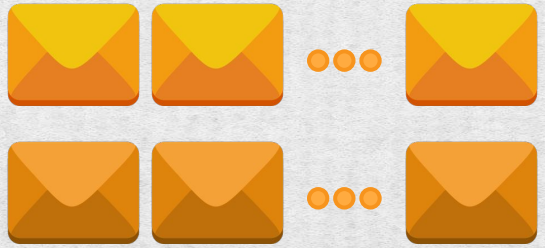


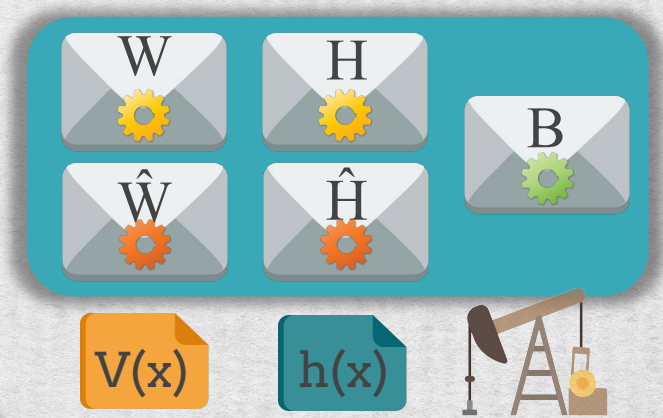
- $t(x)h(x) \neq V^2(x) - 1$, but $\mathrm{Enc}(t(s))H = W^2 - 1$

Security Reduction: Cheating Prover to d-PDH

d-PDH





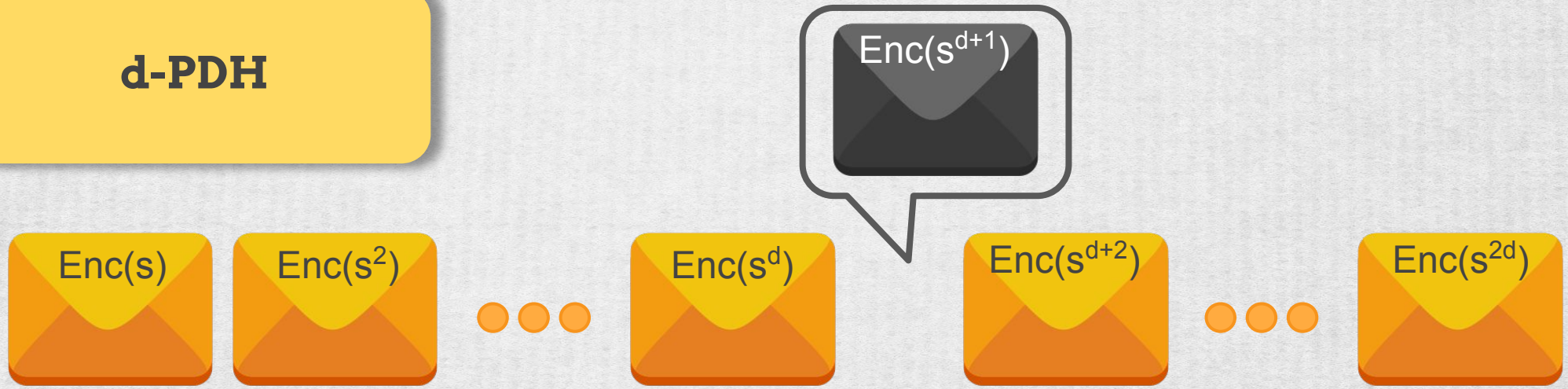


- $t(x)h(x) \neq V^2(x) - 1$, but $\mathrm{Enc}(t(s))H = W^2 - 1$

$p(x) = t(x)h(x) - V^2(x) + 1 \neq 0$, but $p(s) = 0$

Security Reduction: Cheating Prover to d-PDH

d-PDH



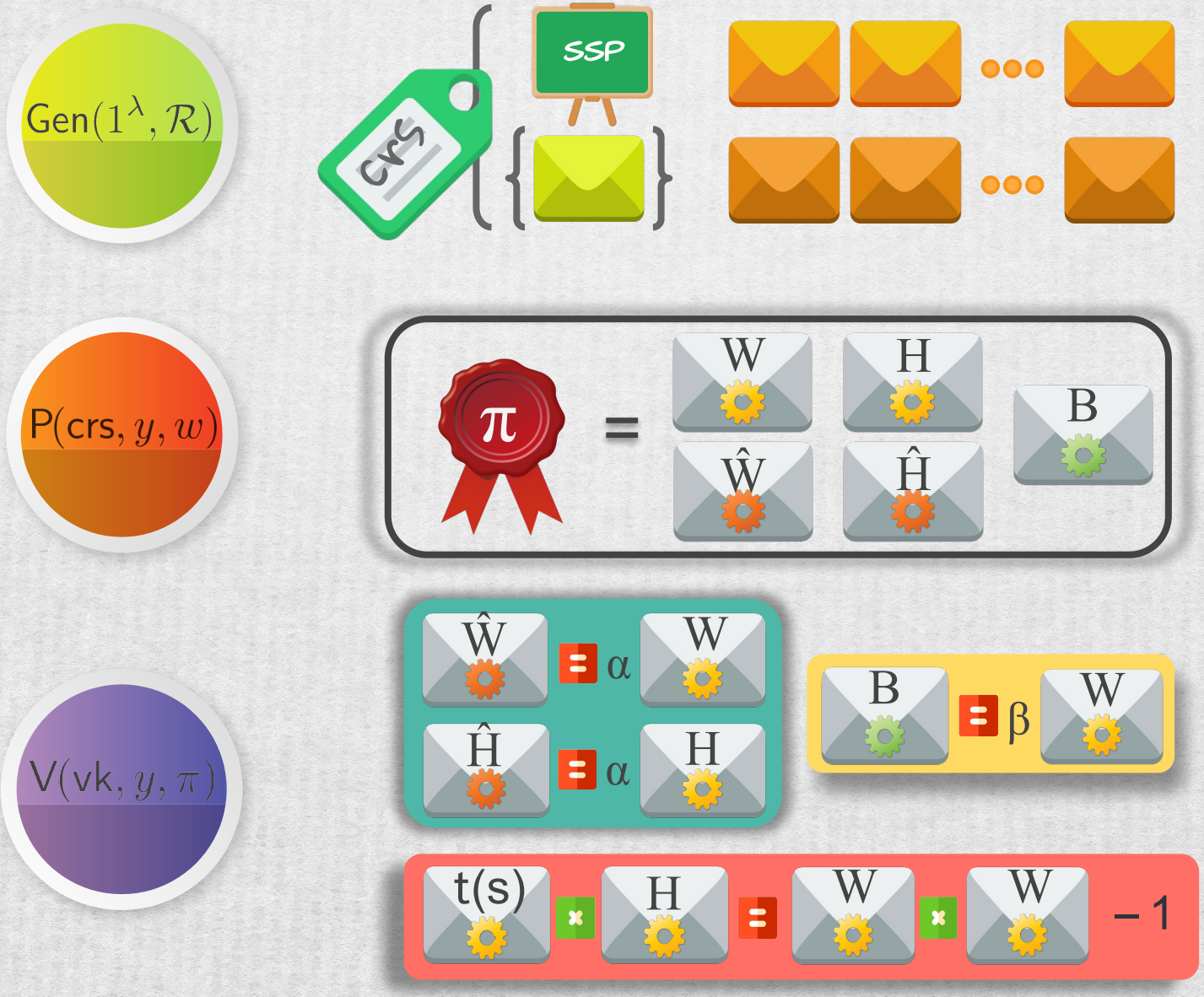
- $\mathrm{t(x)h(x)} \neq V^2(x) - 1$, but $\mathsf{Enc}(t(s))H = W^2 - 1$

$$p(x) = t(x)h(x) - V^2(x) + 1 \neq 0, \text{ but } p(s) = 0$$

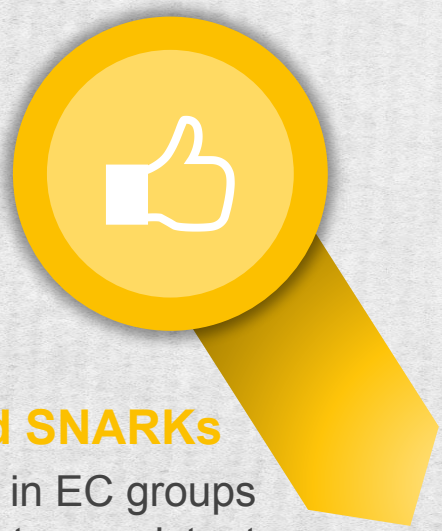
$$p_{d+1}\,\mathsf{Enc}(s^{d+1}) = Enc(p(s)) - \sum_{i=1,\ldots d}^{d+2,\ldots 2d} p_i Enc(s^i)$$

Implementation: Mangiafuoco

Review of the Protocol (Algorithms)



SNARKs: Further Directions



Standard SNARKs

based on DLog in EC groups
not quantum resistant
publicly-verifiable
zero-knowledge

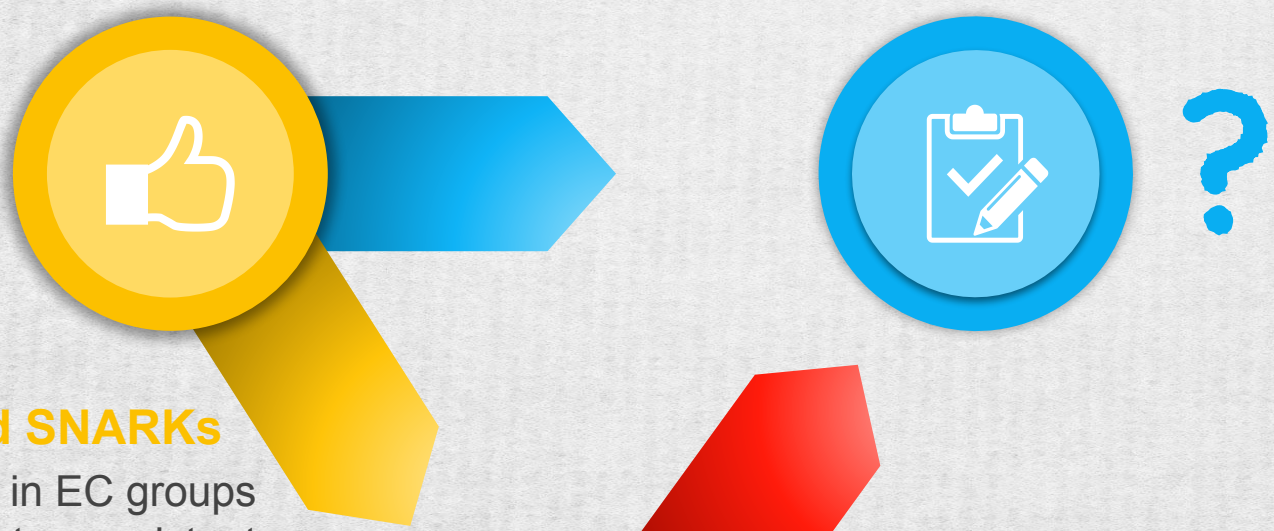
Post-Quantum SNARKs

based on lattice assumptions
designated-verifiable
zero-knowledge

SNARKs: Further Directions

Publicly Verifiable

post-quantum SNARKs



Standard SNARKs

based on DLog in EC groups
not quantum resistant
publicly-verifiable
zero-knowledge

Post-Quantum SNARKs

based on lattice assumptions
designated-verifiable
zero-knowledge

THANK YOU

Encodings: Publicly vs. Designated Verifiable

Publicly Verifiable Encoding:

- linear operation using **crs**
- quadratic root detection using **crs**
- image verification using **crs**

Designated Verifiable Encoding:

- linear operation using **crs**
- quadratic root detection needs **sk**
- image verification using **crs**

Security: Types of encodings

Publicly Verifiable Encoding:

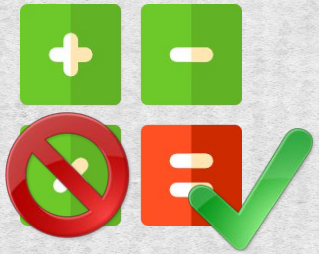
- linear operation using **crs**
- quadratic root detection using **crs**
- image verification using **crs**

Designated Verifiable Encoding:

- linear operation using **crs**
- quadratic root detection needs **sk**
- image verification using **crs**

Prover

Verifier



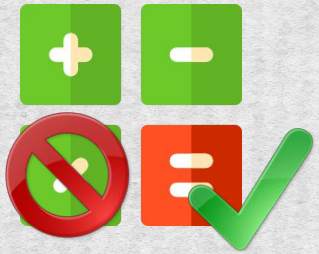
Security: Types of encodings

Publicly Verifiable Encoding:

- linear operation using **crs**
- quadratic root detection using **crs**
- image verification using **crs**

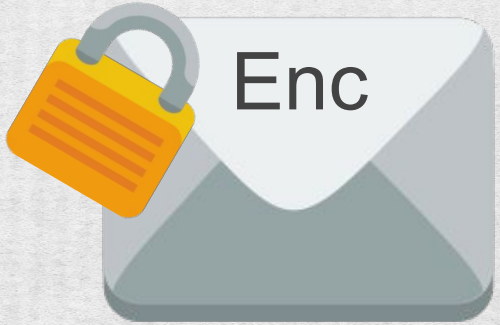
Prover

Verifier



Designated Verifiable Encoding:

- linear operation using **crs**
- quadratic root detection needs **sk**
- image verification using **crs**

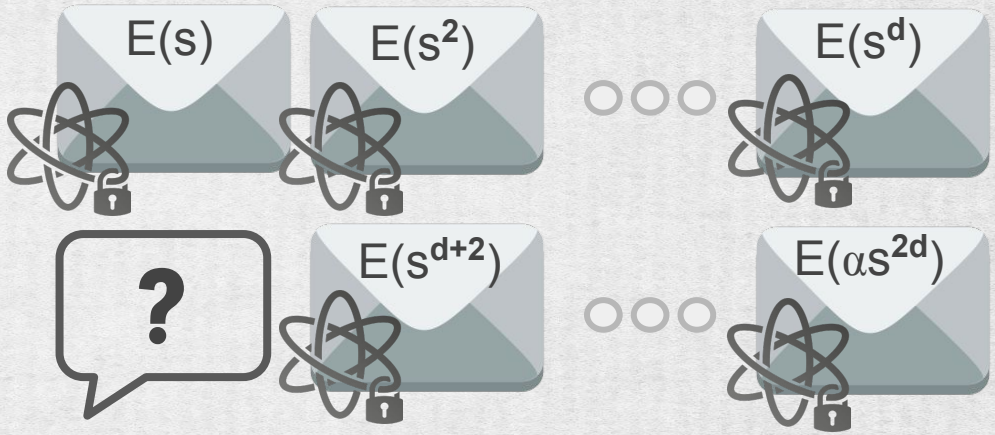


Prover

Verifier

Assumption d-PDH

d-PDH



Technical Aspects