Verifiable delegated computation: zk-SNARKs and Applications

Anca Nitulescu
CRYPTO team

Outline

Motivation

Cloud Computing

Blockchain Privacy

SNARKs

Definition

Properties

Quantum resistance

Construction

Tool Chain

Encodings

Assumptions

Security

Directions

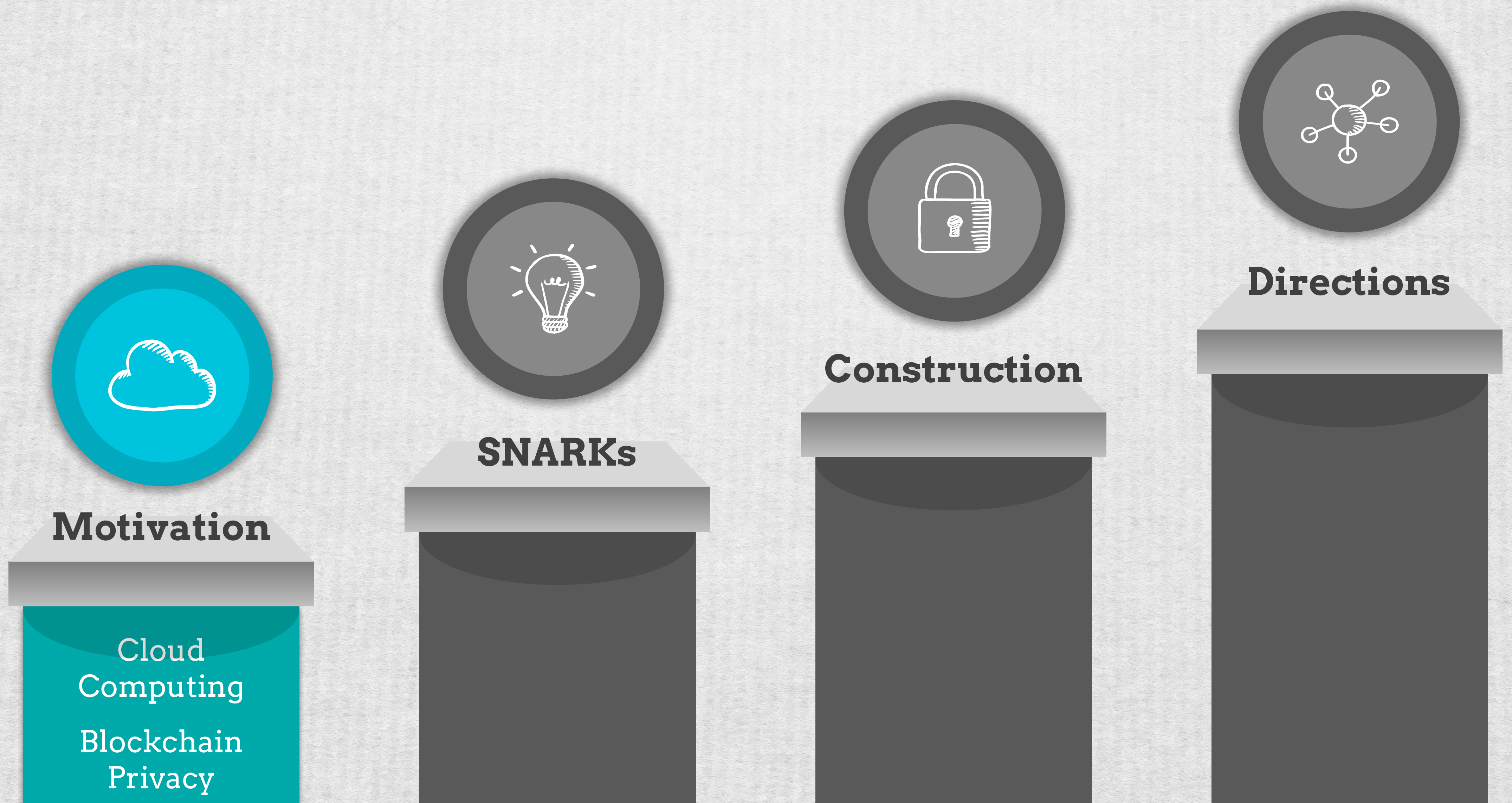
Post-quantum SNARK

Difficulties Comparison

Open Problems

Conclusions

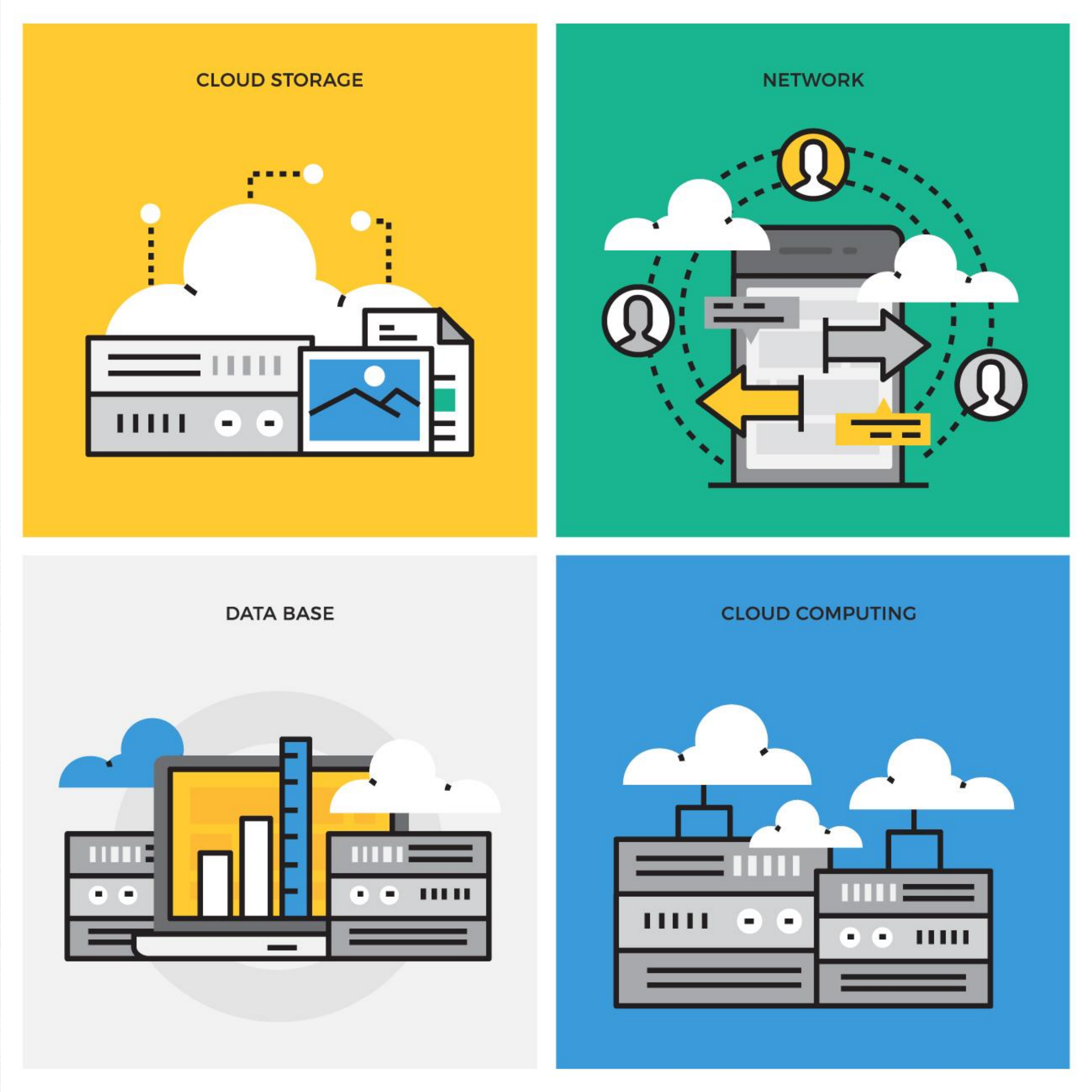
Motivation


Motivation
Cloud Computing
Blockchain Privacy
SNARKs
Construction
Directions

Cloud: Available for Everything

Store documents, photos, videos, etc

Share them with colleagues, friends, family



Ask queries on the data

Process the data

Outsourced Processing

The Cloud Provider:

- **knows the content**
- **performs the computations**

Claims to

- safely store the data
- securely process the data
- answer correctly to our queries
- protect privacy

Risks

For economical reasons, by accident, or attacks

- data can get deleted
- results of computation can be modified
- one can use your private data to analyze and sell/negotiate the information

Cryptography in the Cloud

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

Compute f(x)

Client

Worker

Delegated Computation

“I have the result $\mathbf{y}$=f(x)”

Client

Worker

Unreliable worker

$y^* \neq f(x)$

Client

Corrupted Worker

Ask for a proof

Verify π

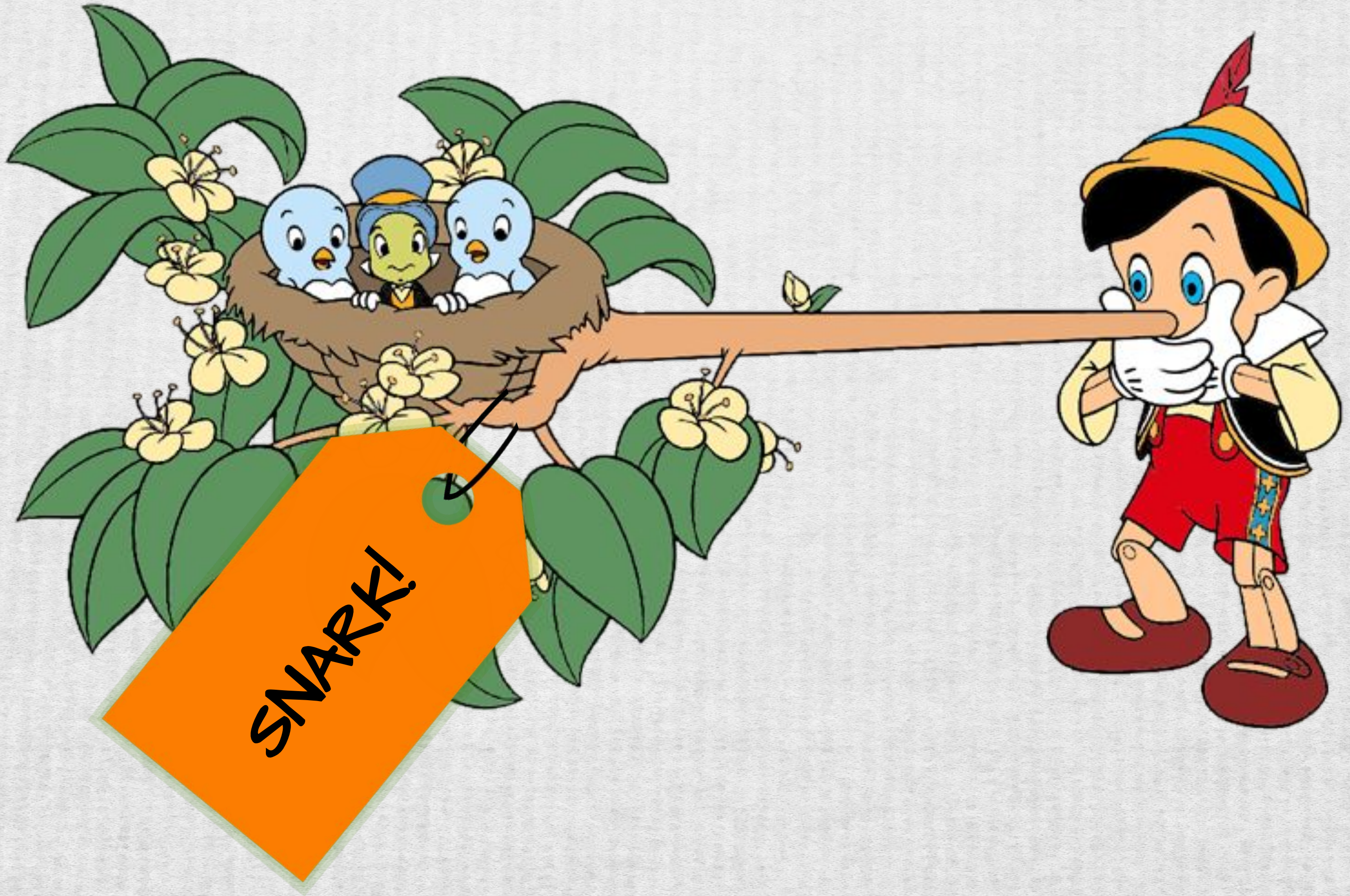
$y^* \neq f(x)$

Proof π

Client
Verifier

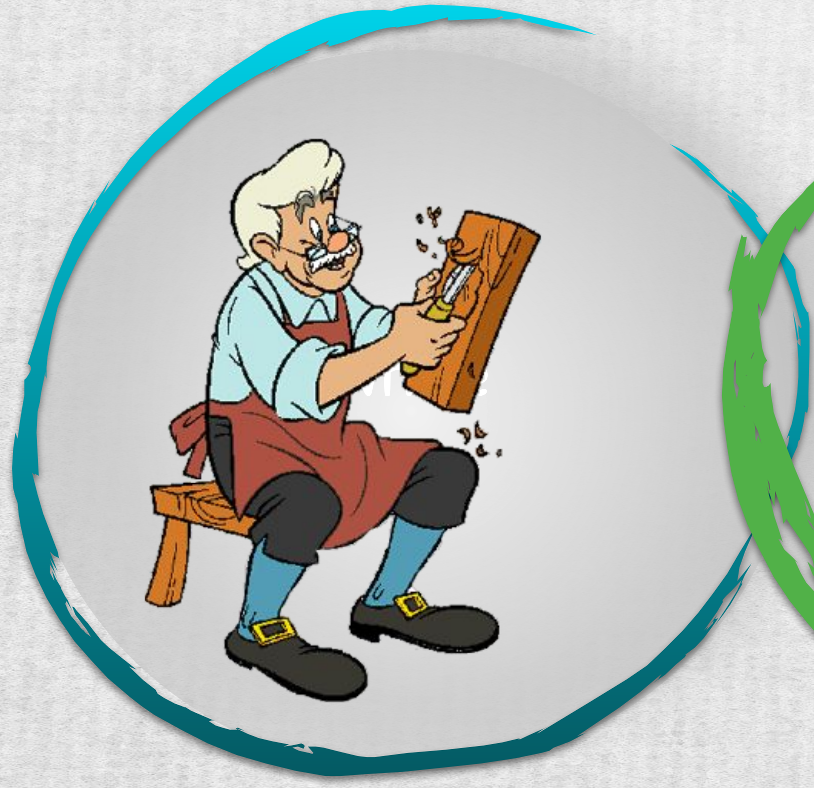
Worker
Prover

Integrity for Computation

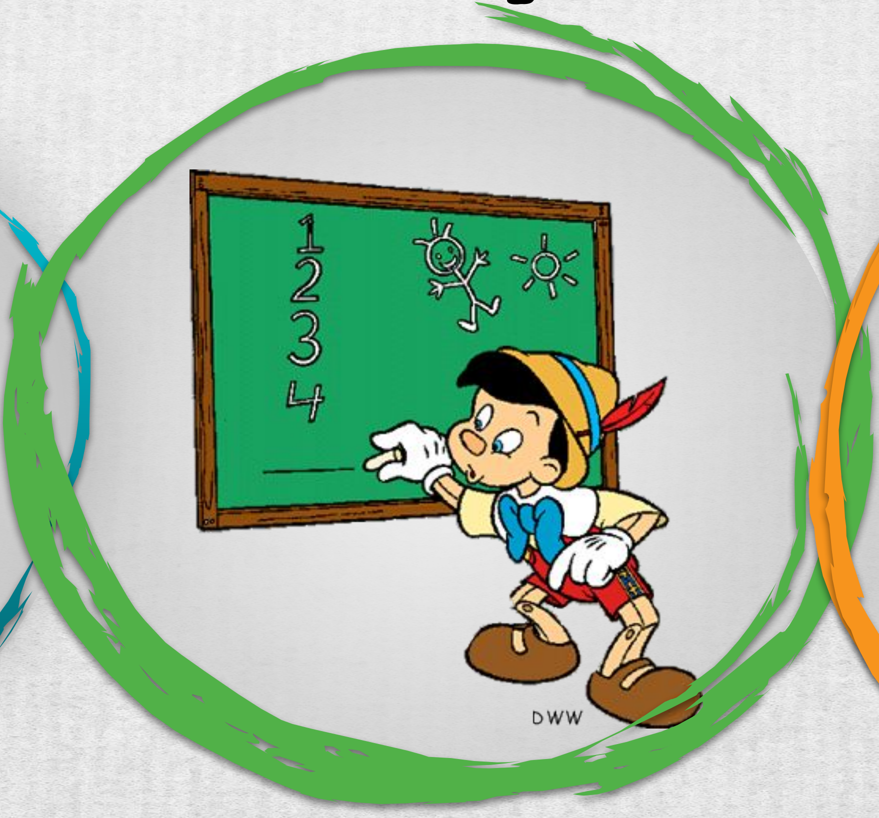


SNARK: Properties of a Proof

Fast Verification

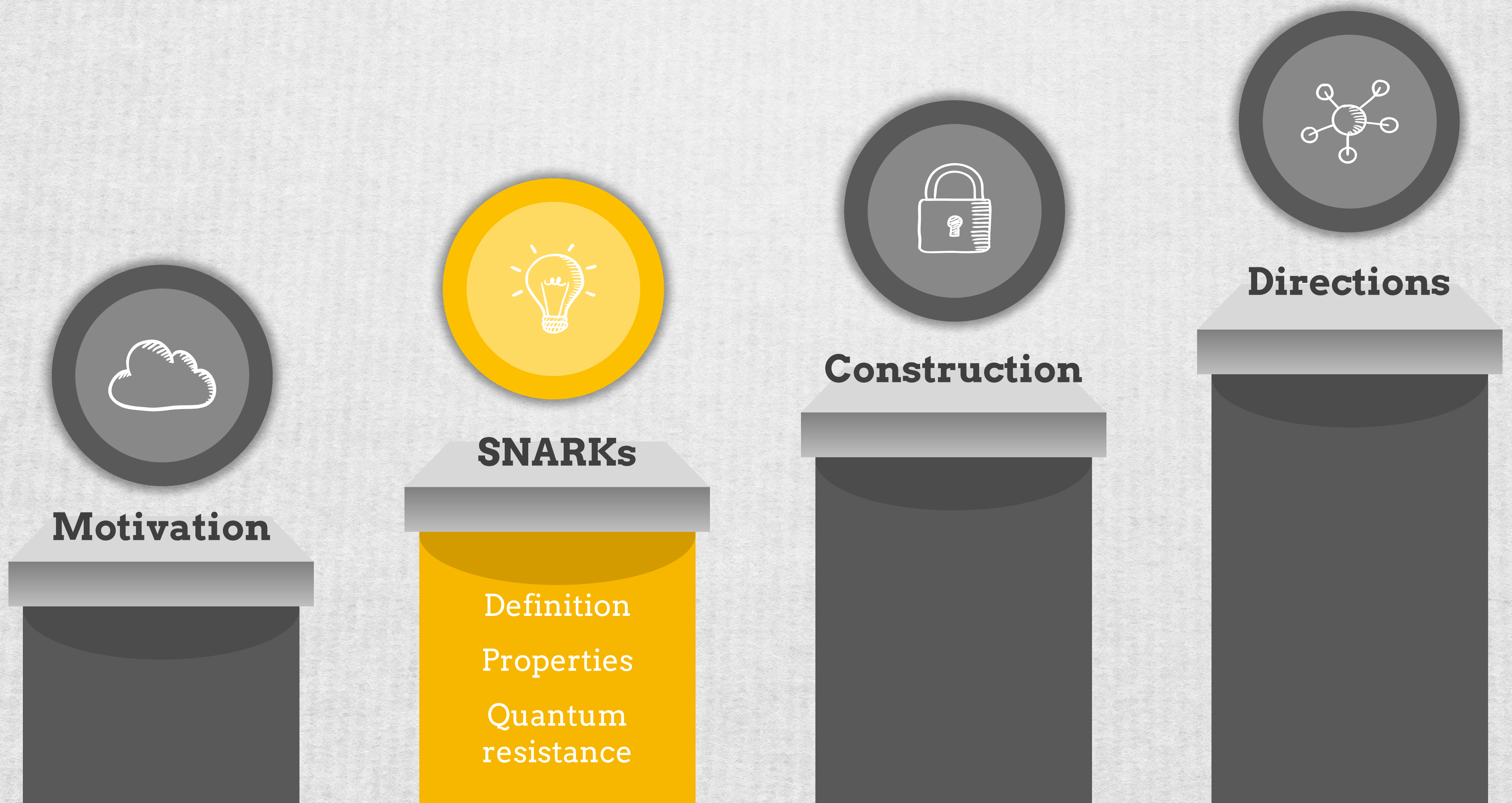


Proof of Knowledge

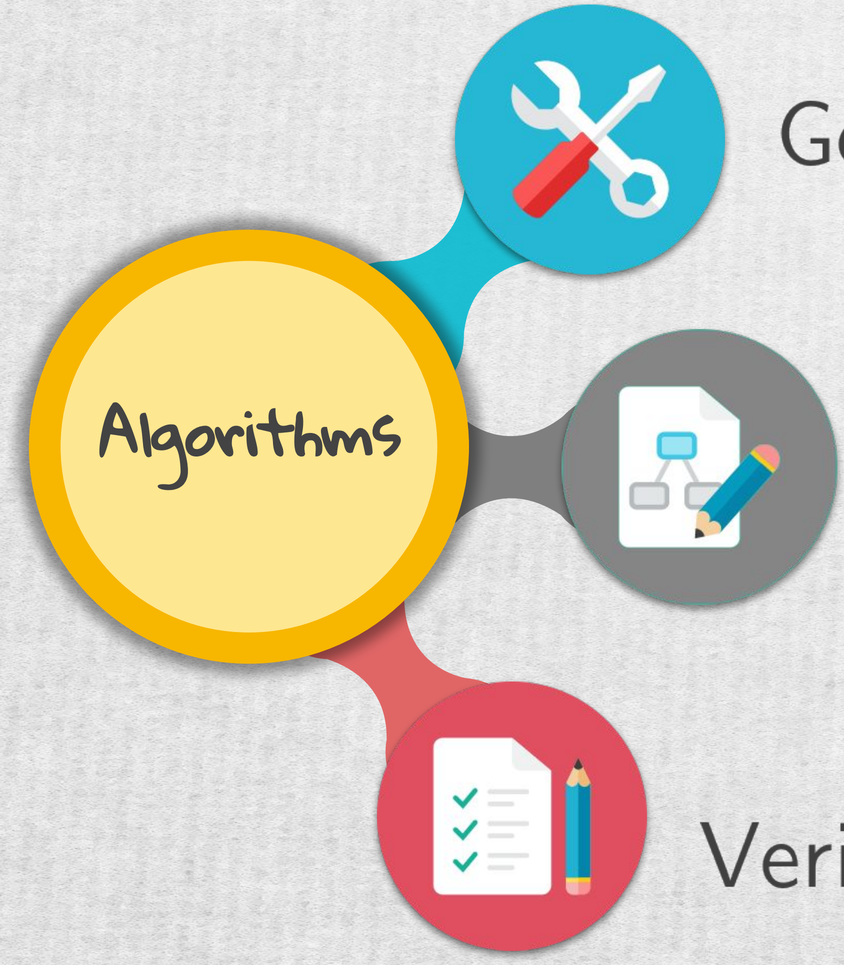


Succinct

SNARKs


Motivation
SNARKs
Definition
Properties
Quantum resistance
Construction
Directions

Proof Systems



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

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

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

Proofs in Crypto: since 1980s

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

Proofs

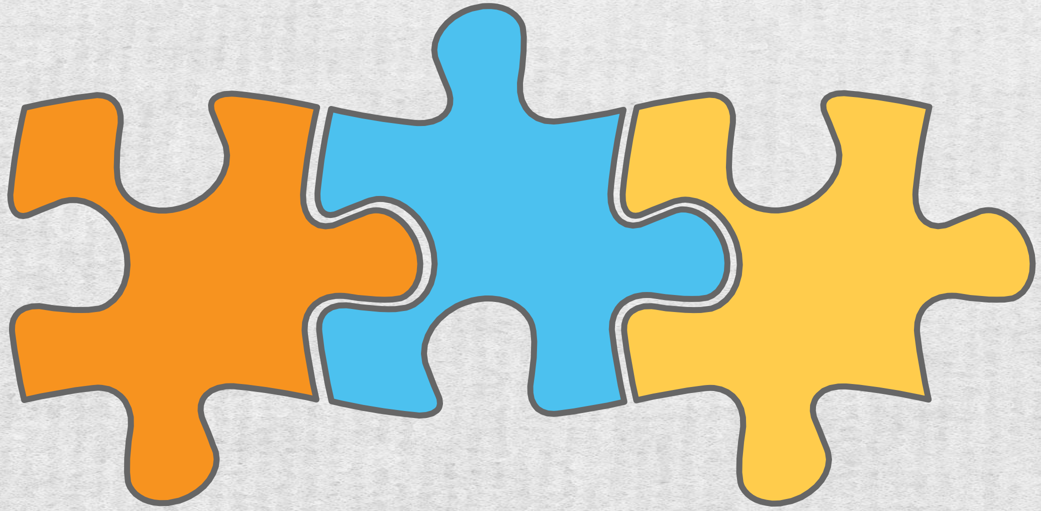
Zero-Knowledge
does not leak anything
about the witness

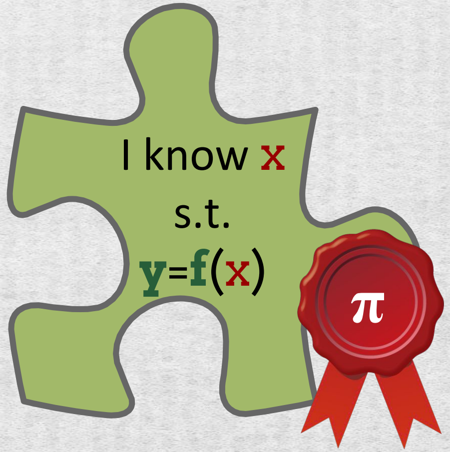
Efficiency
verification faster
than computing **f(x)**

Proofs in Private Blockchain

Key Properties for usage in Distributed Protocols

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





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

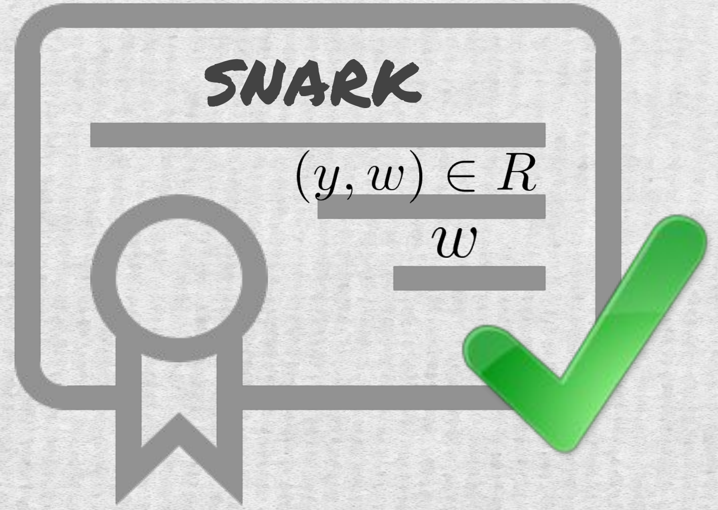
Argument of Knowledge

$\mathcal{A}$

crs, aux

Adversary

π



Argument of Knowledge

$\mathcal{A}$

crs, aux

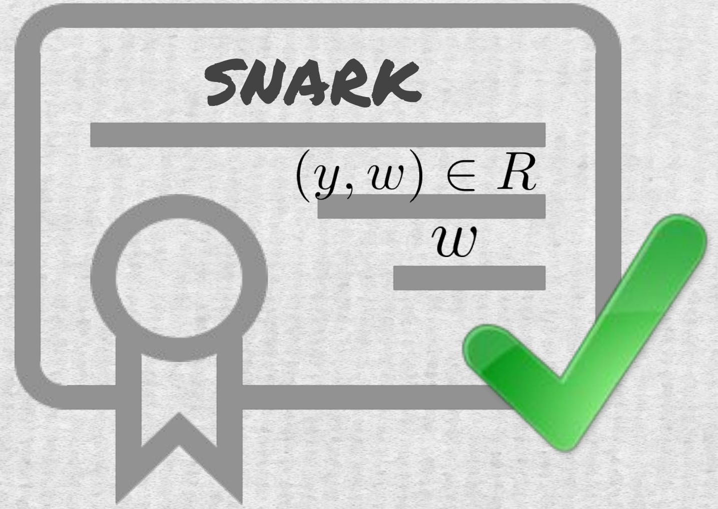
Adversary

$\mathcal{E}$

crs, aux

extractor

π



Efficient Constructions: 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

Existent SNARKs:

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

Quantum Attacks

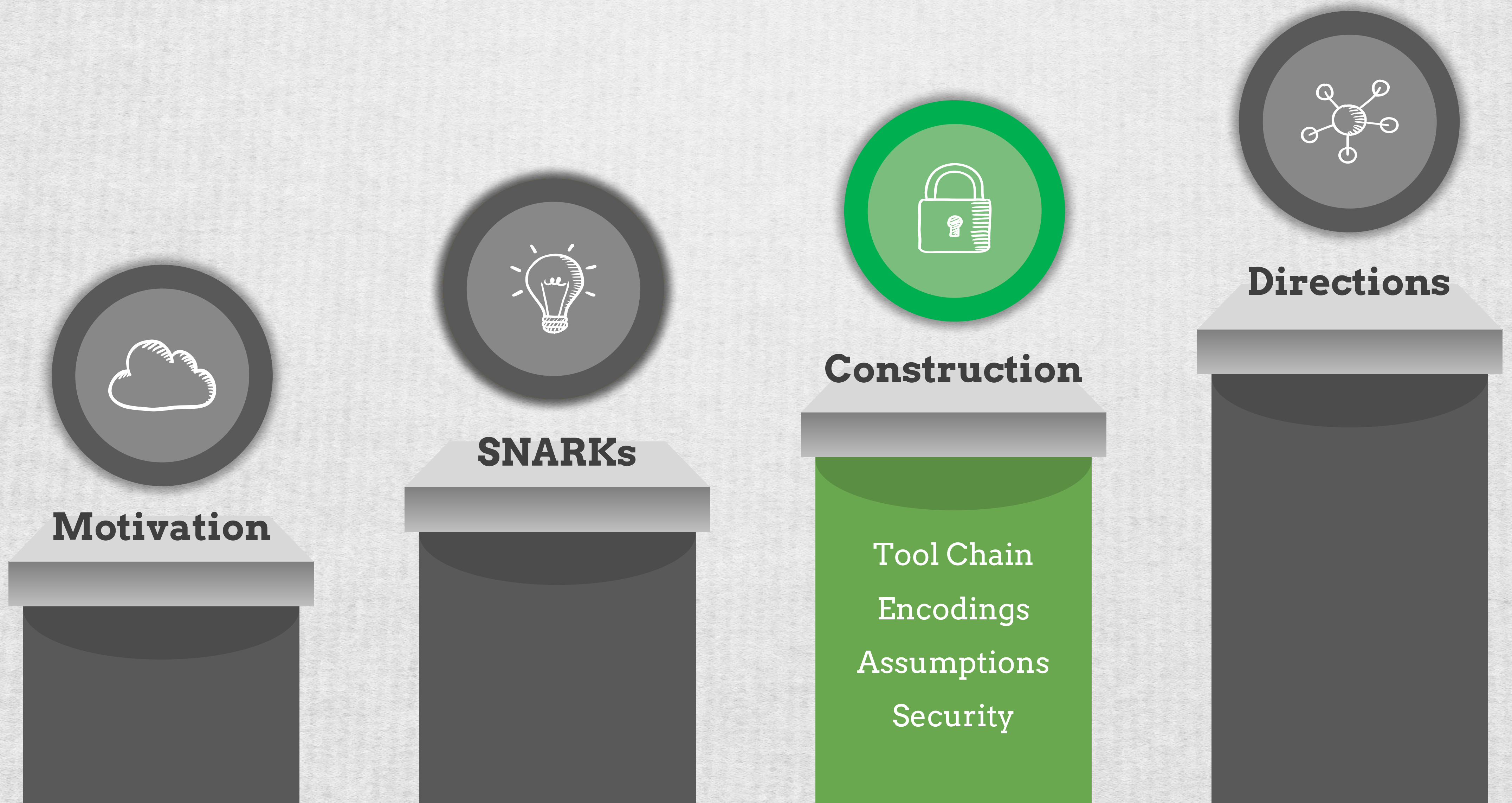
Existent SNARKs:

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

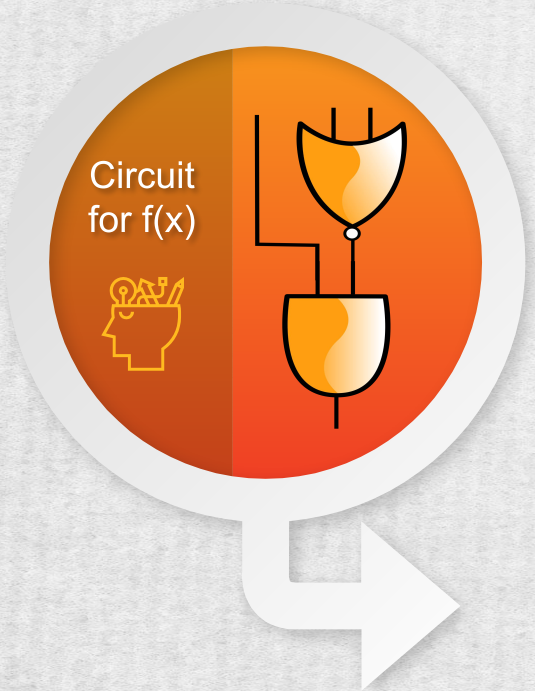
Post-Quantum SNARKs:

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

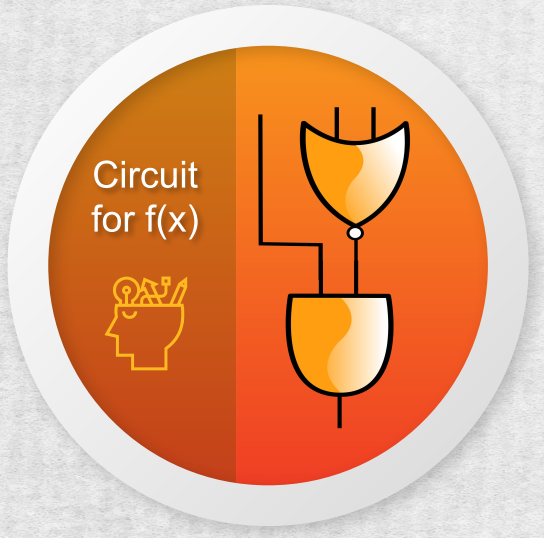
Construction



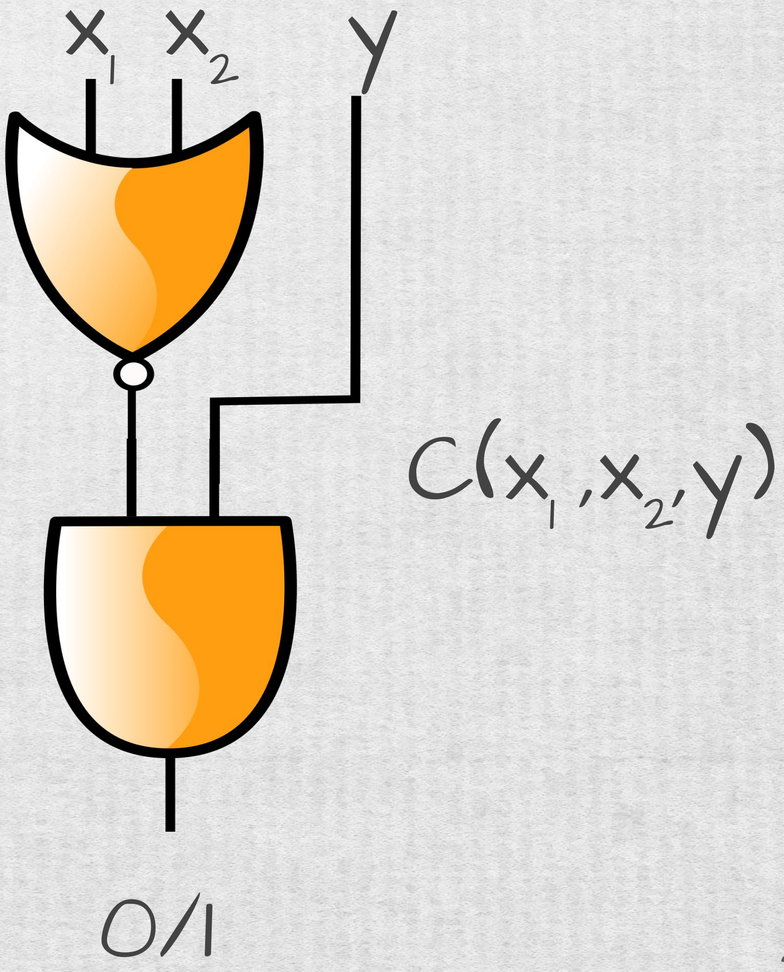
SNARK: overview of Toolchain



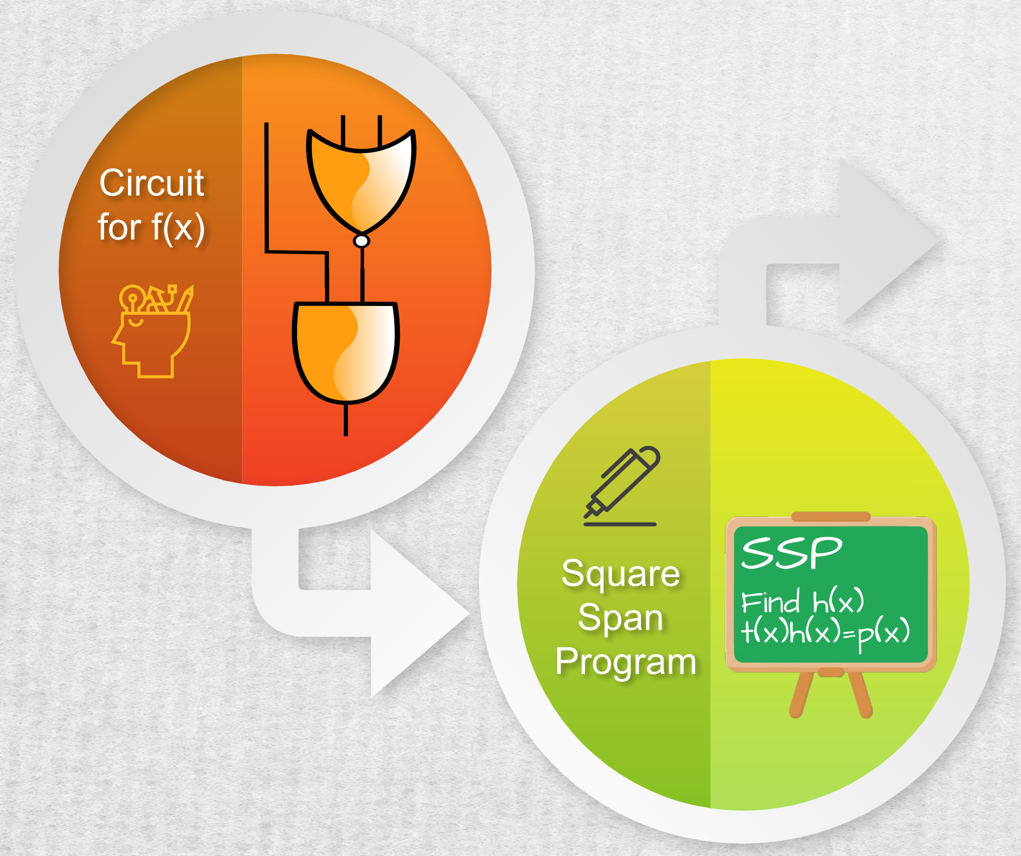
Circuit Satisfiability Problem



$f(x_1, x_2) = y$

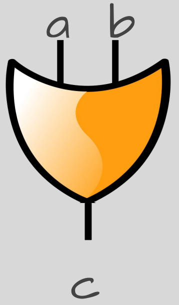


SNARK: overview of Toolchain

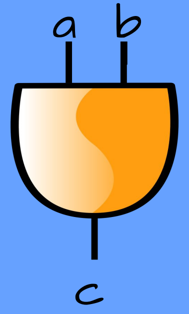


Step 1. Linearization of logic gates

OR gate



AND gate



XOR gate

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
$-a-b+2c \in \{0,1\}$	$a+b-2c \in \{0,1\}$	$a+b+c \in \{0,2\}$	$3-3c \in \{0,1\}$	$2a, 2b \in \{0,2\}$

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

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

$$(V\,a + \delta) \circ (V\,a + \delta - 2) = 0$$

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

$$\left(Va + \delta - 1\right) \circ \left(Va + \delta - 1\right) = 1$$

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

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

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

Step 4. Polynomial Problem SSP

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

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

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

Step 4. Polynomial Problem SSP

Square Span Program

SSP

Find h(x)

$t(x)h(x)=p(x)$

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

SSP

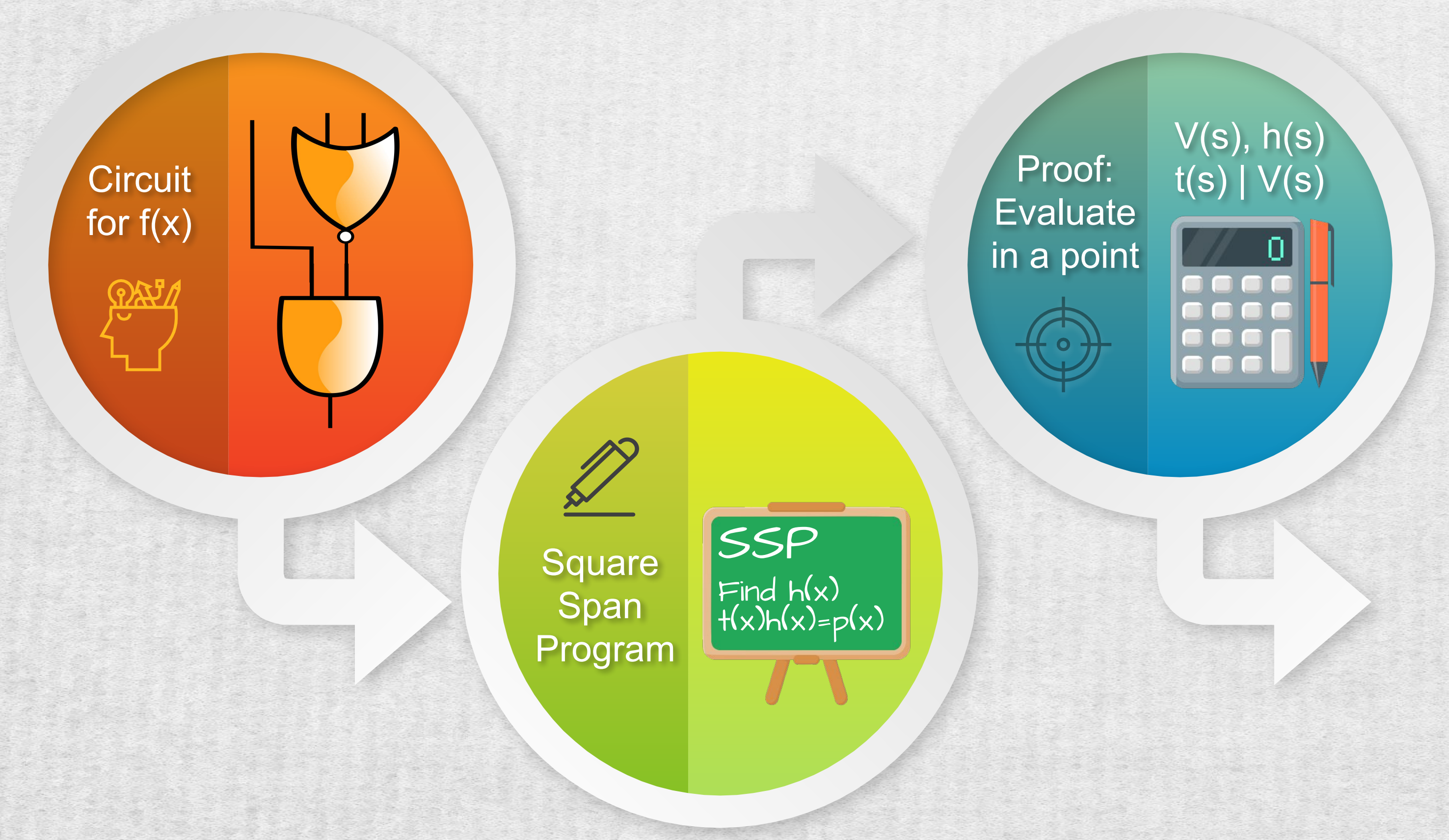
For $\{v_i(x)\}_{i=1,m},\ t(x) \in \mathbb{F}[x]$

find $V(x), h(x)$ such that

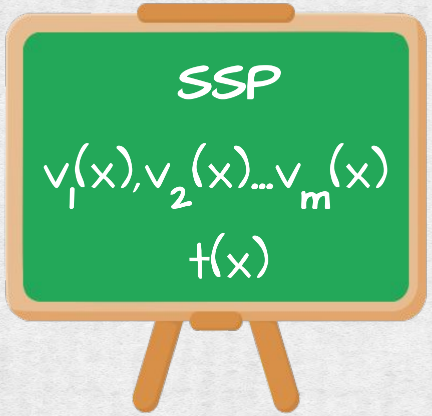
$$V(x) = v_0(x) + \sum_{i=1}^{m} a_i v_i(x)$$

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

SNARK: overview of Toolchain



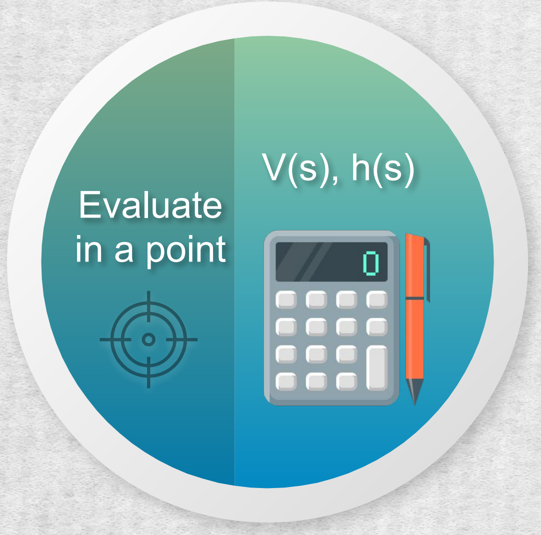
Proving on top of SSP: Idea



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

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

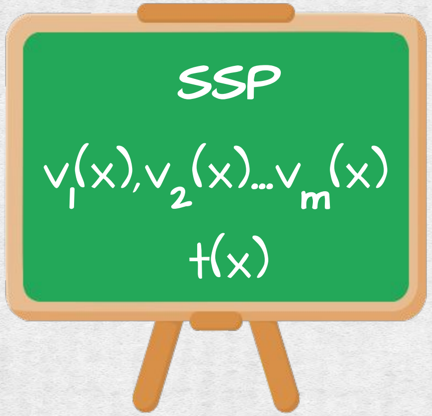
$$t(s)h(s) = V(s)^2 - 1$$



Prover: Evaluate the solution in a random unknown point **s**

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

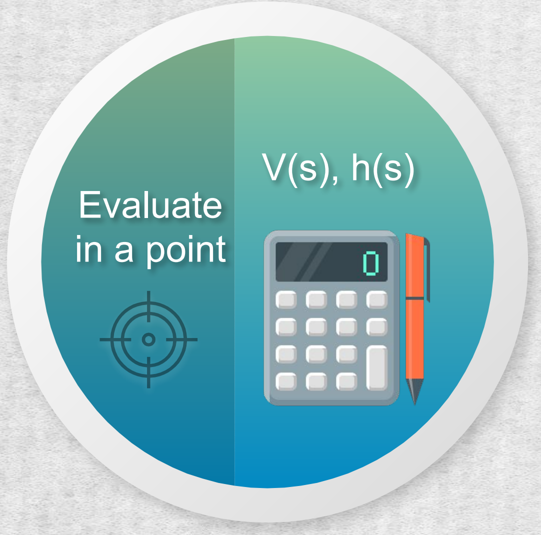
Proving on top of SSP: Idea

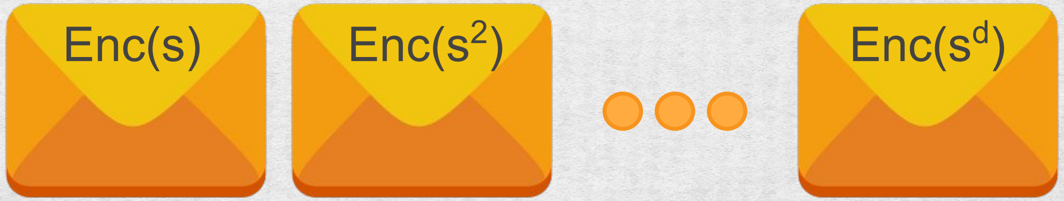


$$V(s), h(s) = ?$$

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

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





Proving on top of SSP: Idea

SSP

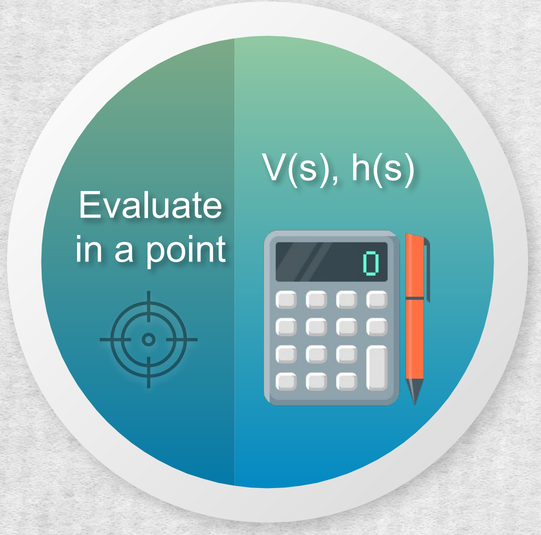
$v_1(x), v_2(x) \ldots v_m(x)$

$t(x)$

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

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

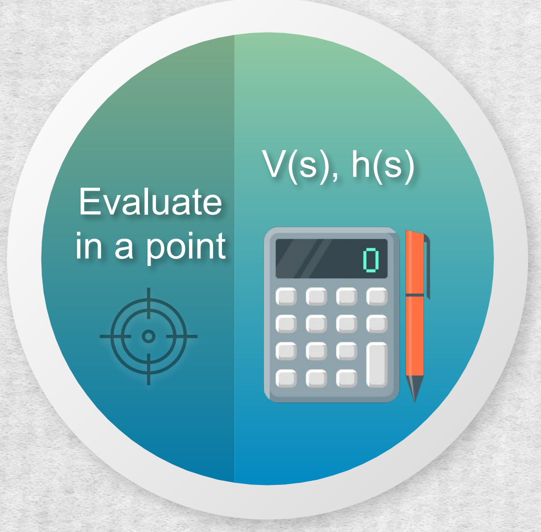
$$t(s)h(s) = V(s)^2 - 1$$

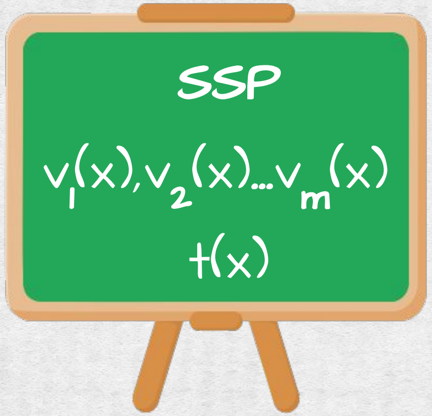


$\mathrm{Enc}(s)$ $\mathrm{Enc}(s^2)$ … $\mathrm{Enc}(s^d)$ → $\mathrm{Enc}(h(s))$

$\mathrm{Enc}(h(s)) = \mathrm{Enc}(\sum h_i s^i)$

Proving on top of SSP: Idea





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

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

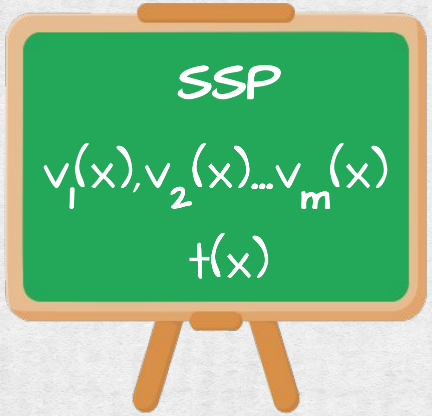
$$t(s)h(s) = V(s)^2 - 1$$

Encoding:

- ***linearly*** homomorphic

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

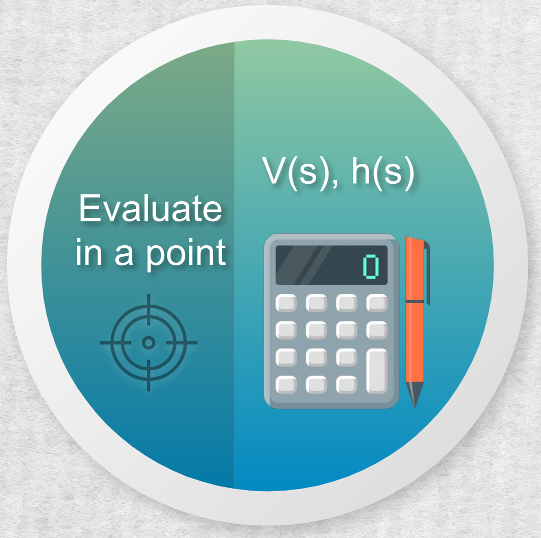
Proving on top of SSP: Idea



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

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

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



Enc(s) Enc(s^2) … Enc(s^d)

Proof  = Enc(V(s)), Enc(h(s))

Not an argument of Knowledge!

SSP

$v_1(x), v_2(x) \dots v_m(x)$

$t(x)$

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

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

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

Evaluate in a point

V(s), h(s)

Enc(s) Enc(s^2) ... Enc(s^d)

$\left\{ \begin{array}{c} \text{Enc}(\beta v_i(s)) \\ \text{Enc}(\beta) \end{array} \right\}_{i=0,m}$

Proof π = Enc(V(s)), Enc(h(s)), Enc(βV(s))

Assumption PKE: Power Knowledge of Exponent

d-PKE

Enc(P) Enc(αP)

Enc(s) Enc(s^2) ... Enc(s^d)

Enc(αs) Enc(αs^2) ... Enc(αs^d)

Assumption PKE: Power Knowledge of Exponent

d-PKE

Enc(P)　Enc(αP)

Enc(s)　Enc(s^2)　…　Enc(s^d)

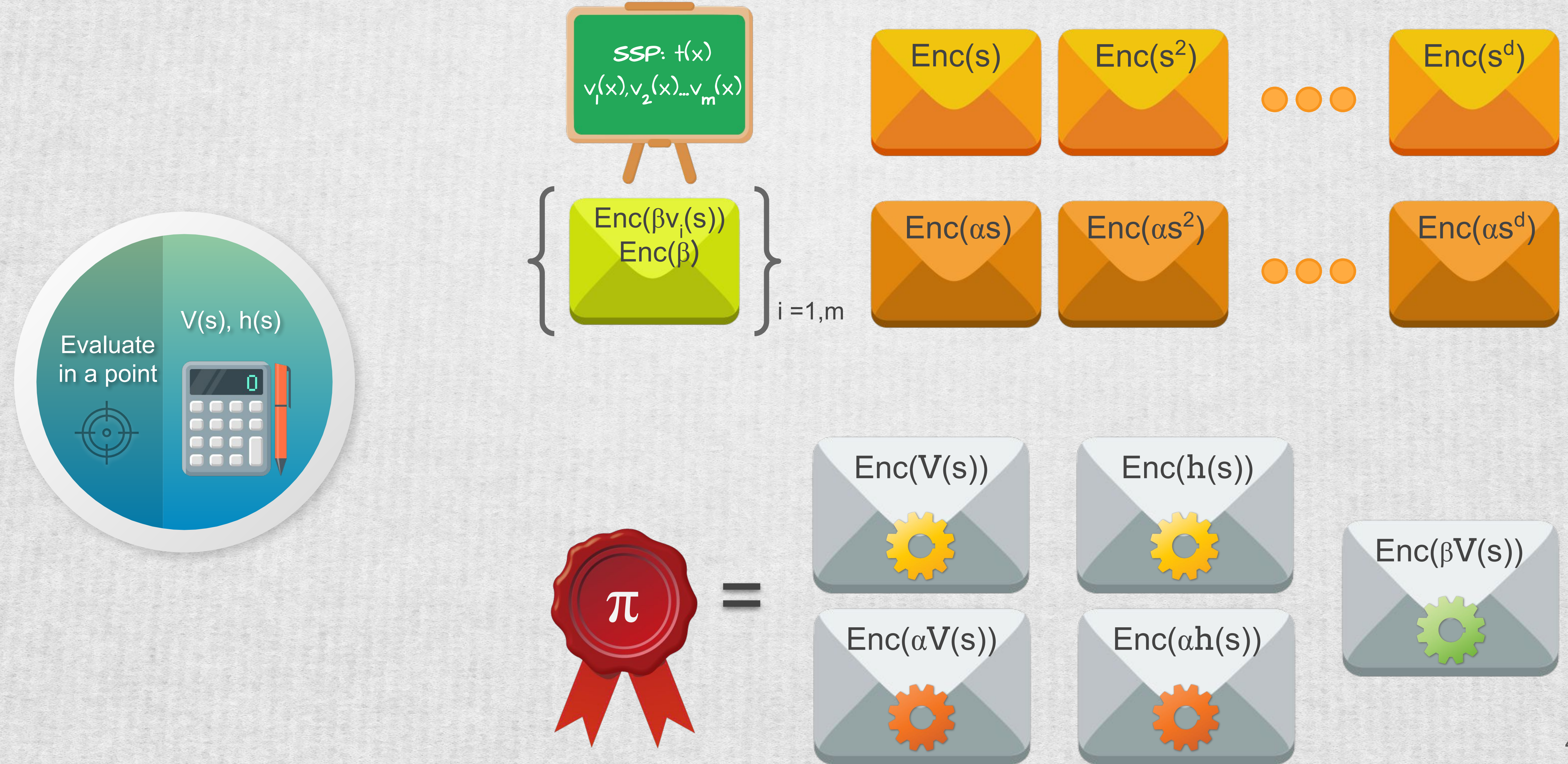
Enc(αs)　Enc(αs^2)　…　Enc(αs^d)

$\mathcal{E}$

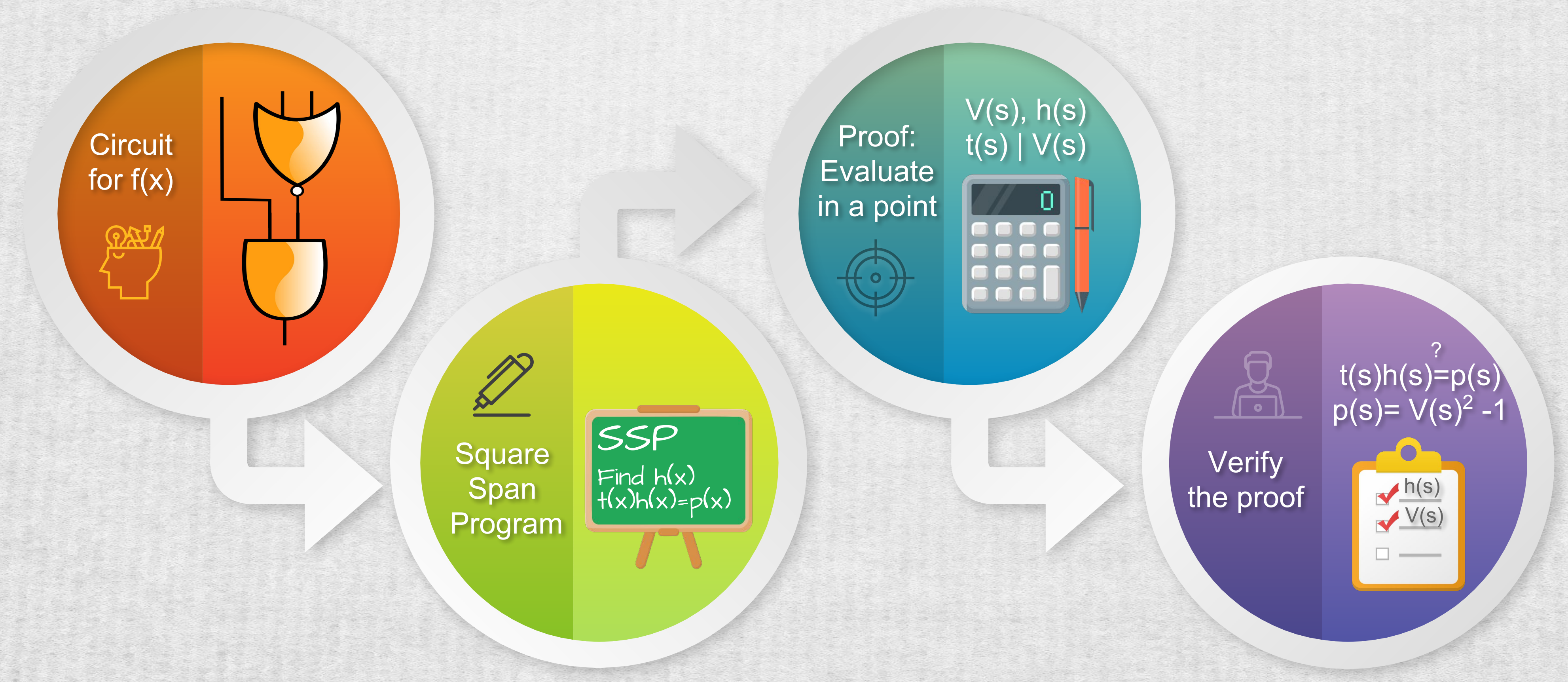
p_1　p_2　…　p_d

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

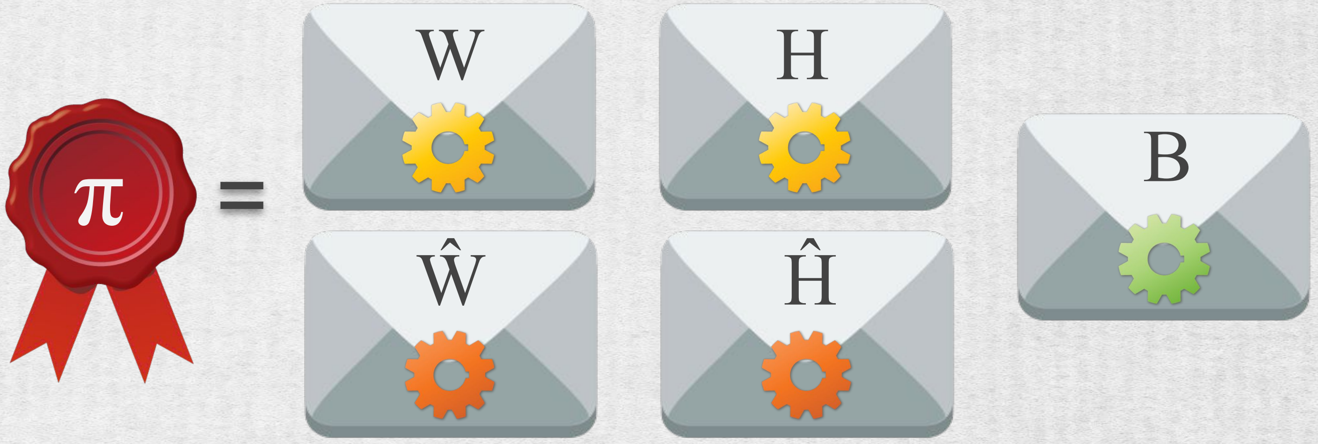
Setup and Proof

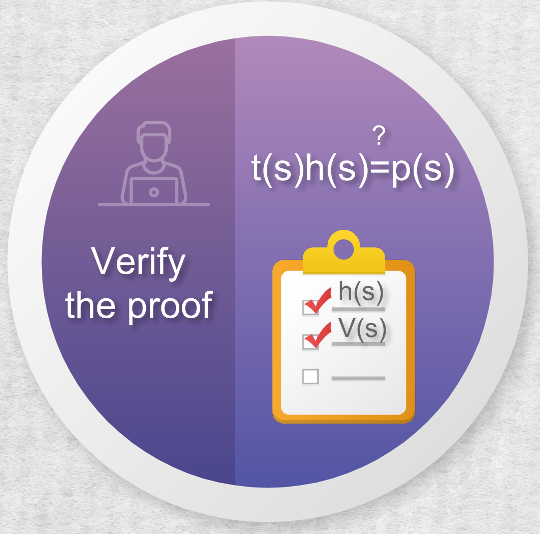


SNARK: overview of Toolchain

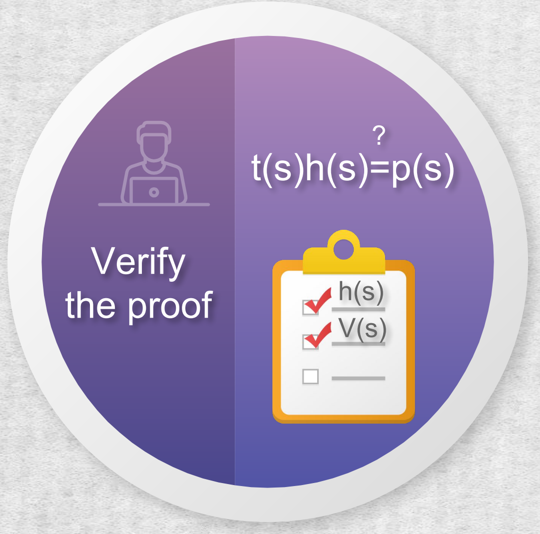


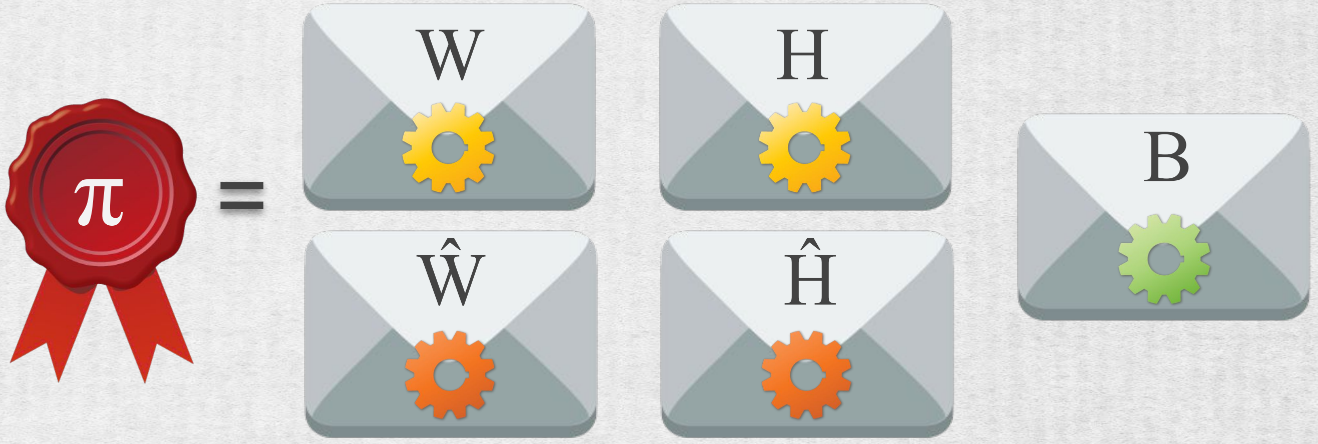
Proving on top of SSP: verifier





Proving on top of SSP: verifier

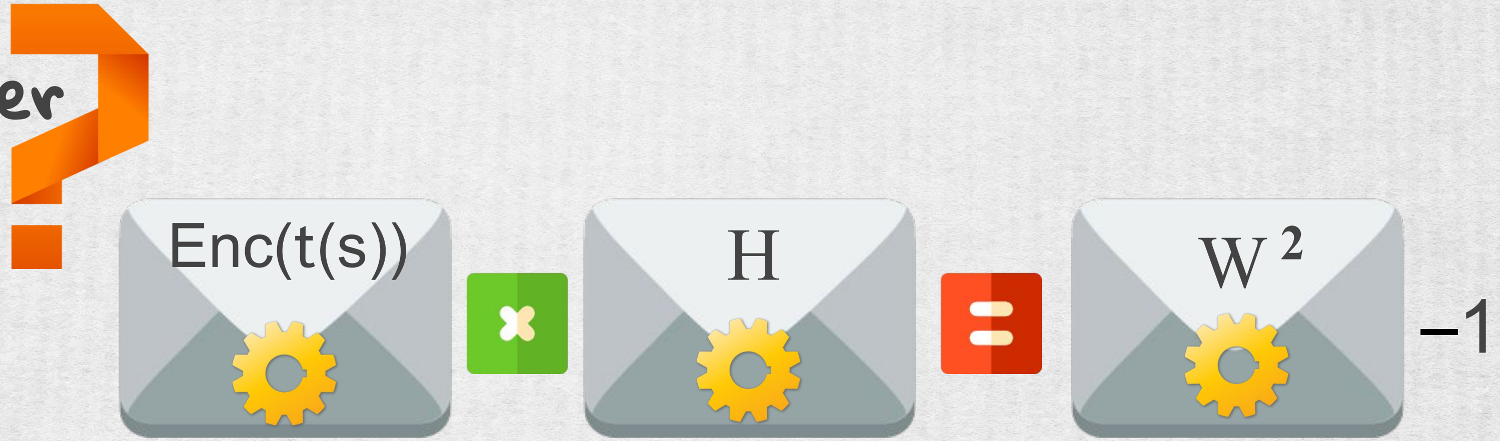




Verifier

Proving on top of SSP: verifier

Verifier

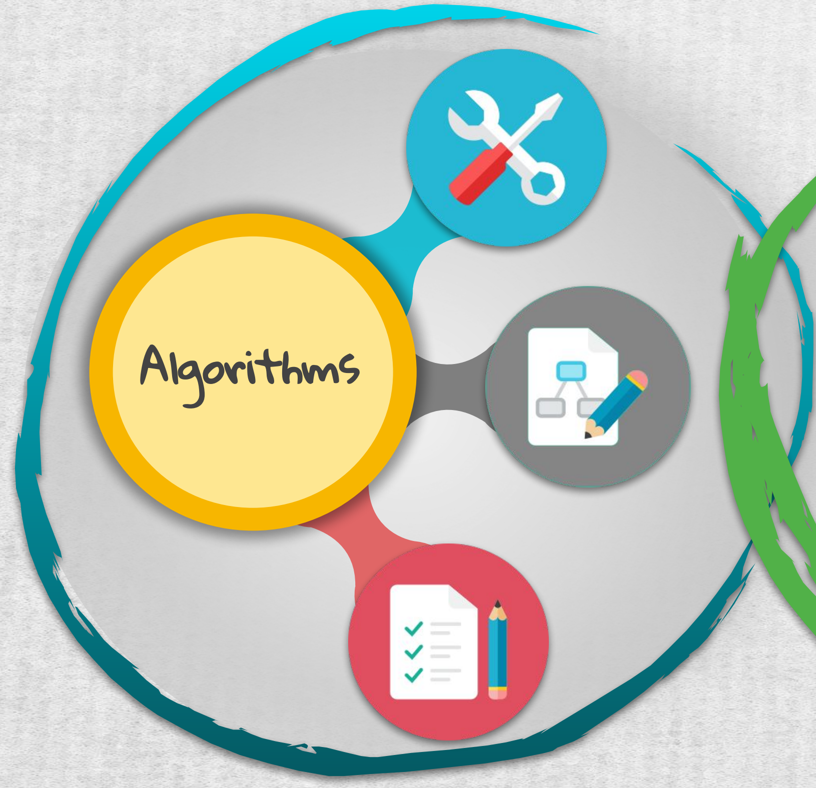


Encoding:

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

Formal Construction and Security

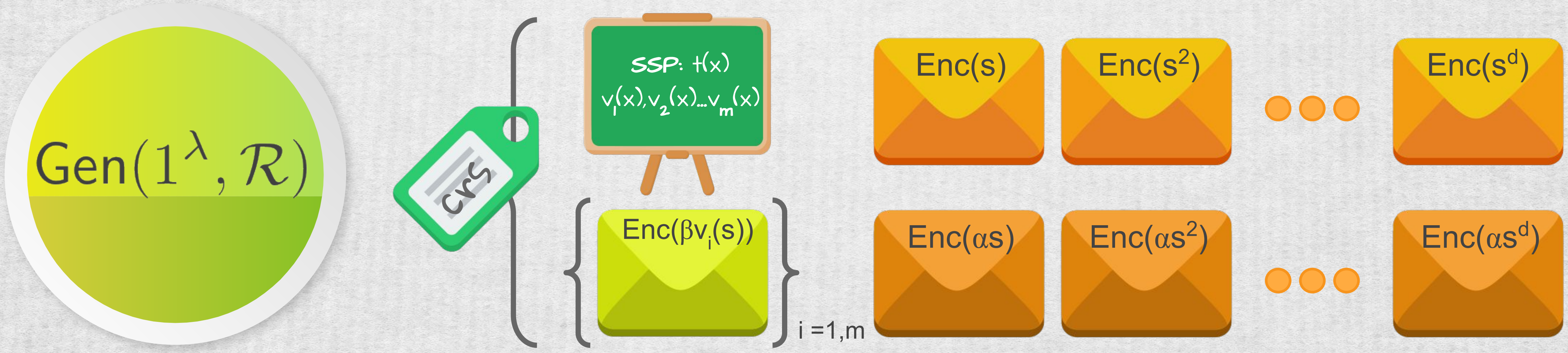
Protocol



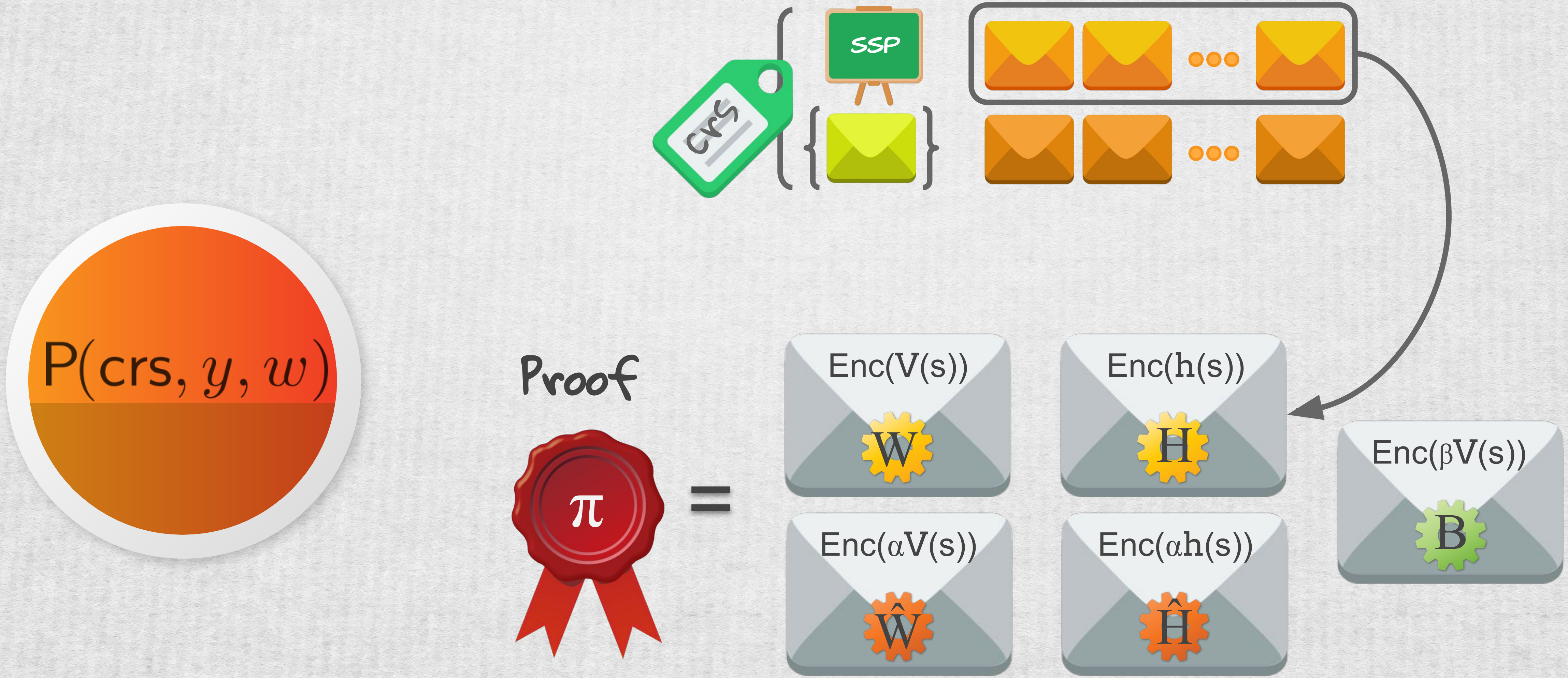
Assumptions

Security

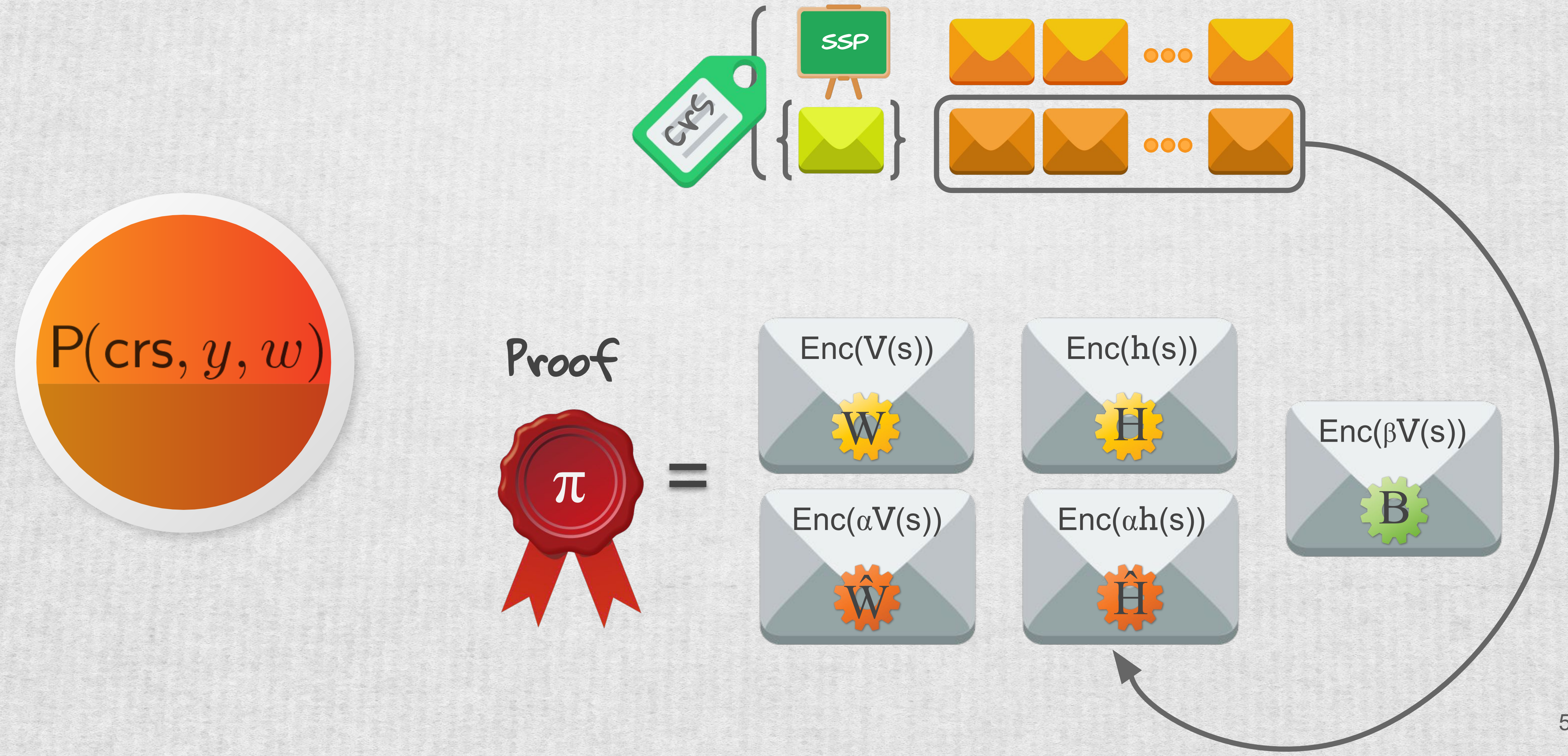
Setup Algorithm



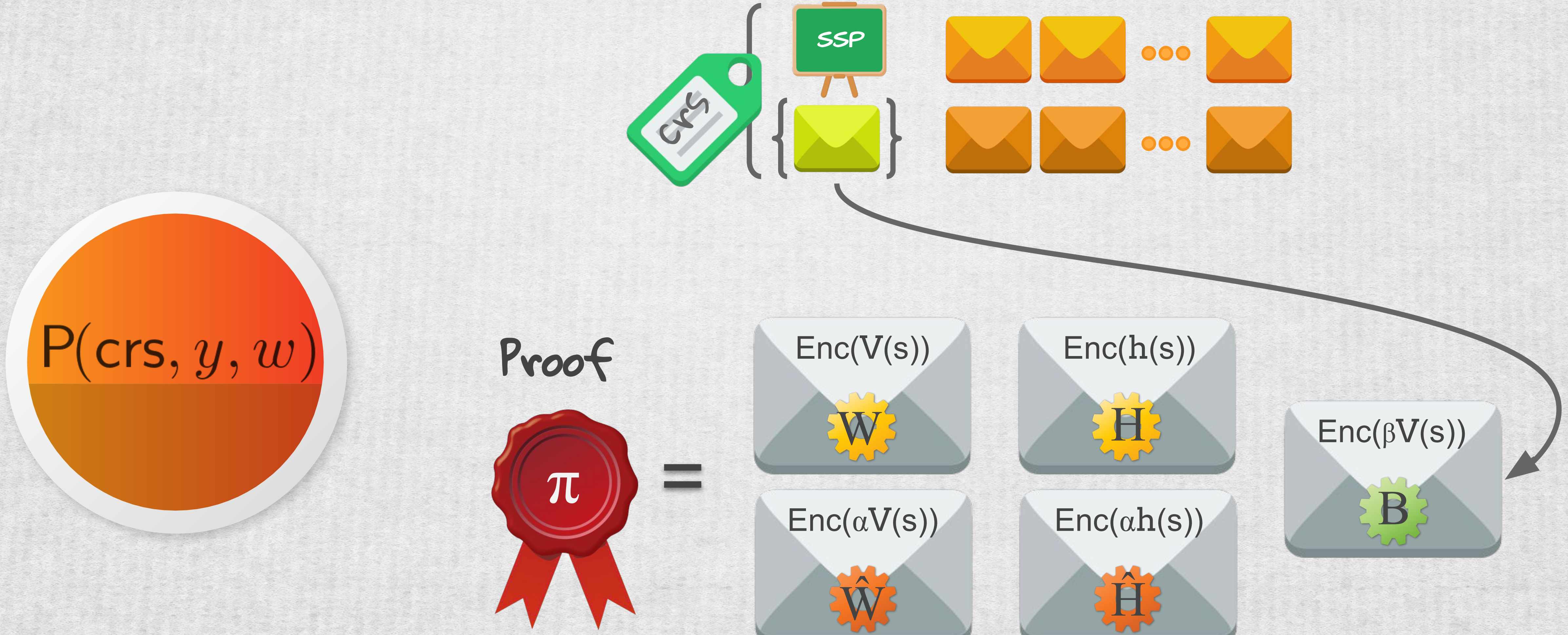
Prover Algorithm



Prover Algorithm



Prover Algorithm



Verifier Algorithm

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

$\hat{W} = \alpha\, W$

$\hat{H} = \alpha\, H$

$B = \beta\, W$

$\mathrm{Enc}(t(s)) \times H = W \times W - 1$

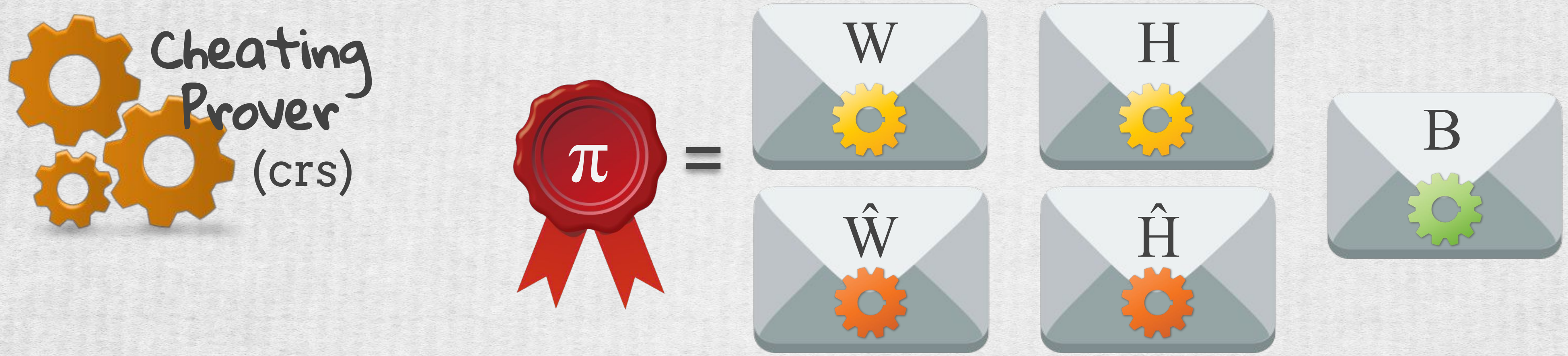
Security Reduction

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

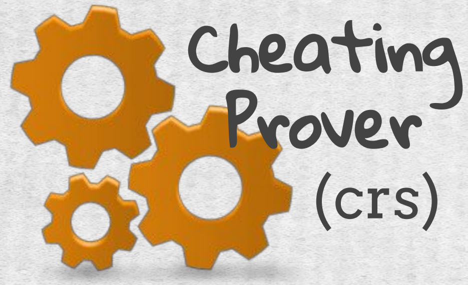
$\mathsf{P}(\mathsf{crs}, y, w)$

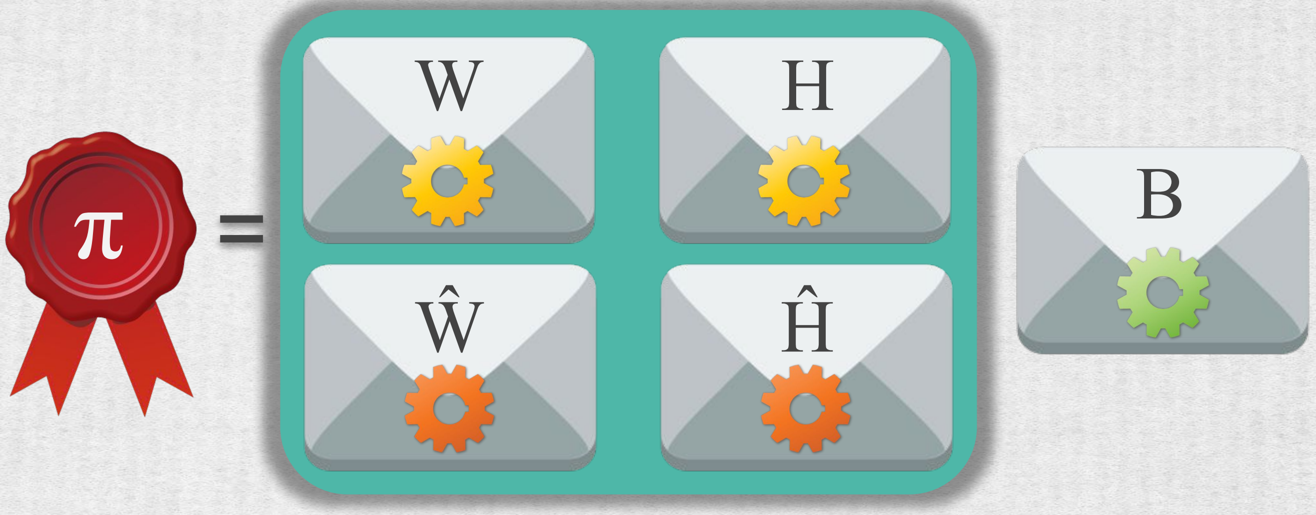
$\mathsf{V}(\mathsf{vk}, y, \pi)$

Cheating Strategy

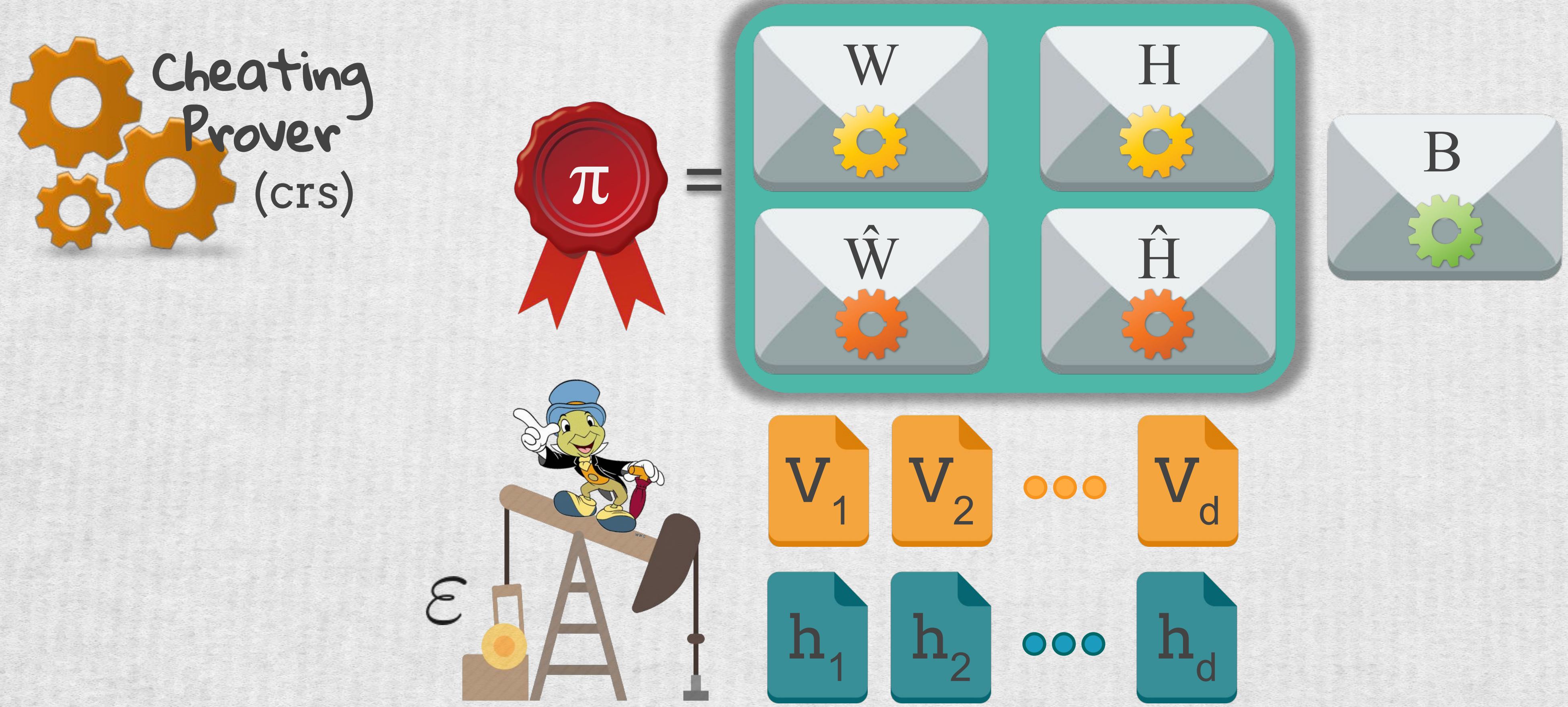


Cheating Strategy

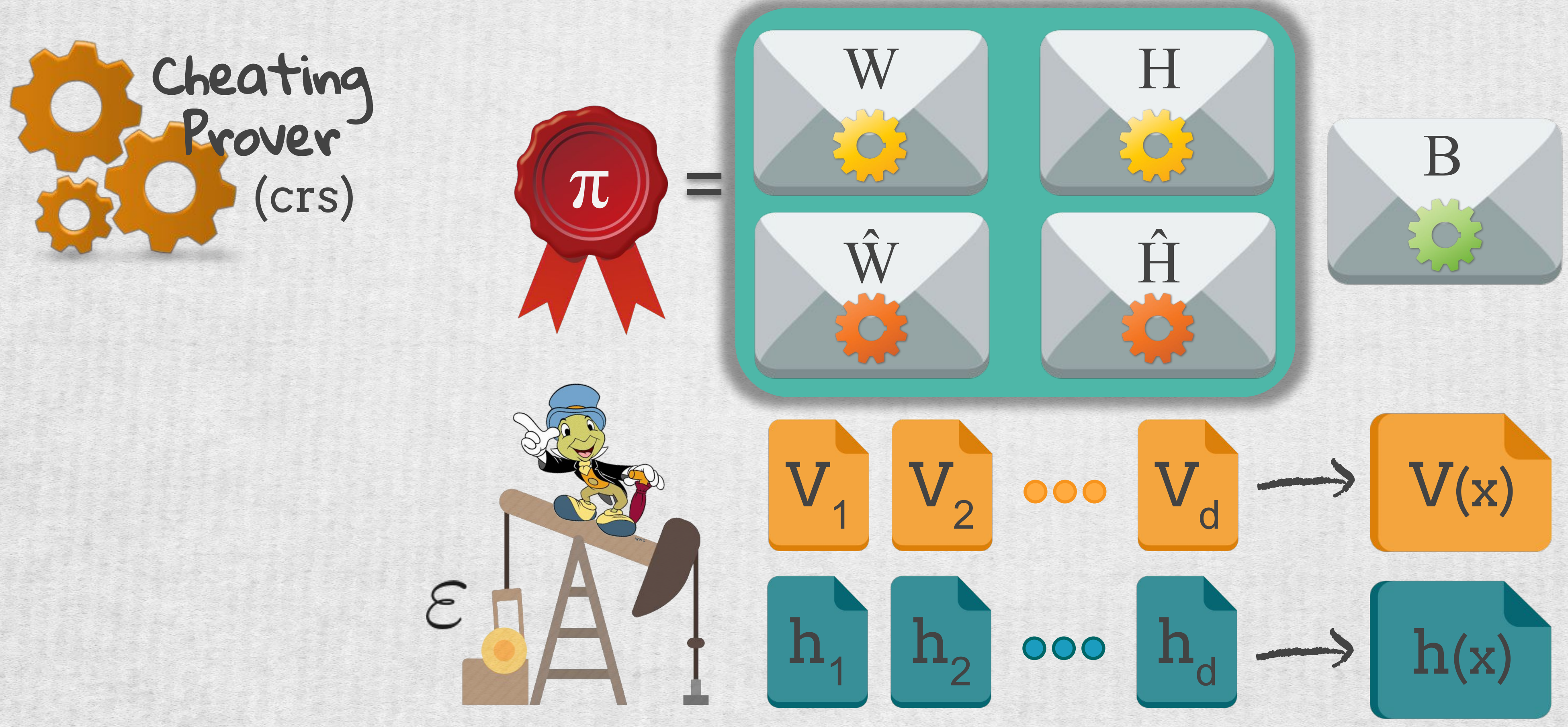




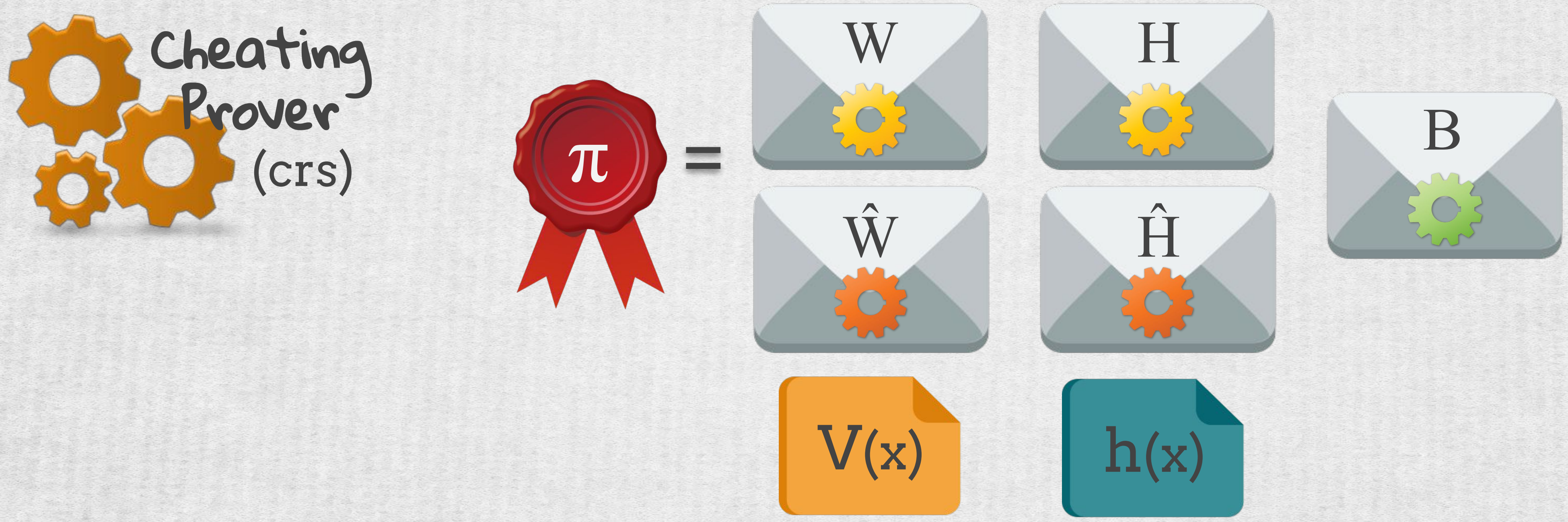
Cheating Strategy



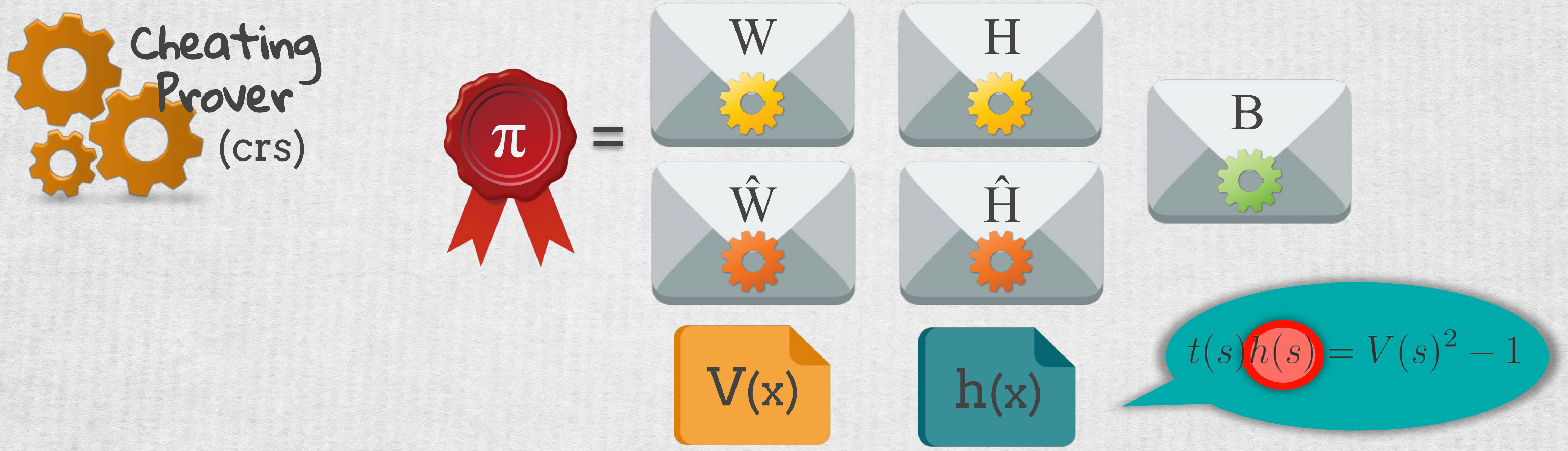
Cheating Strategy



Cheating Strategy



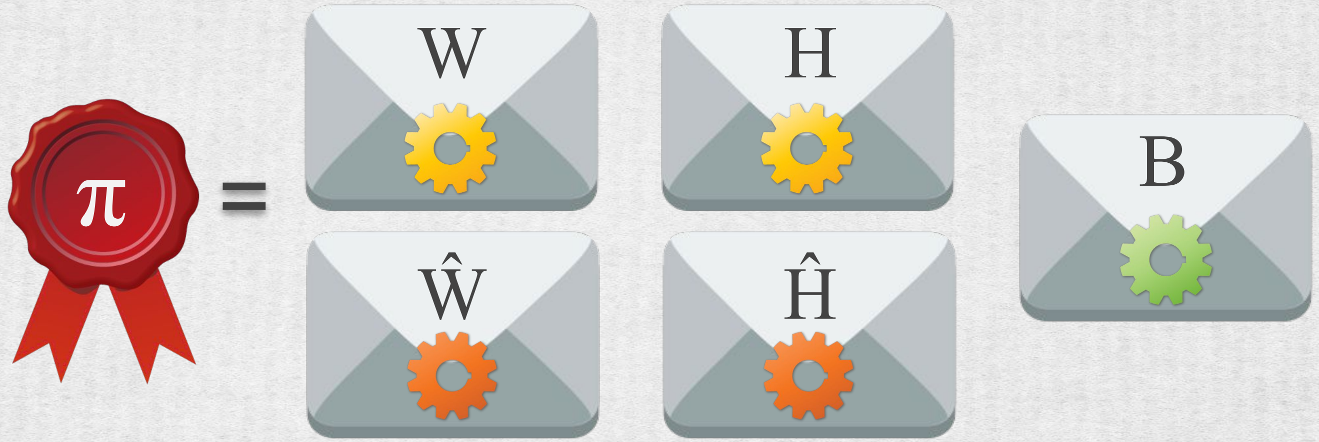
Division does not hold



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

Invalid linear combination

Cheating Prover (crs)



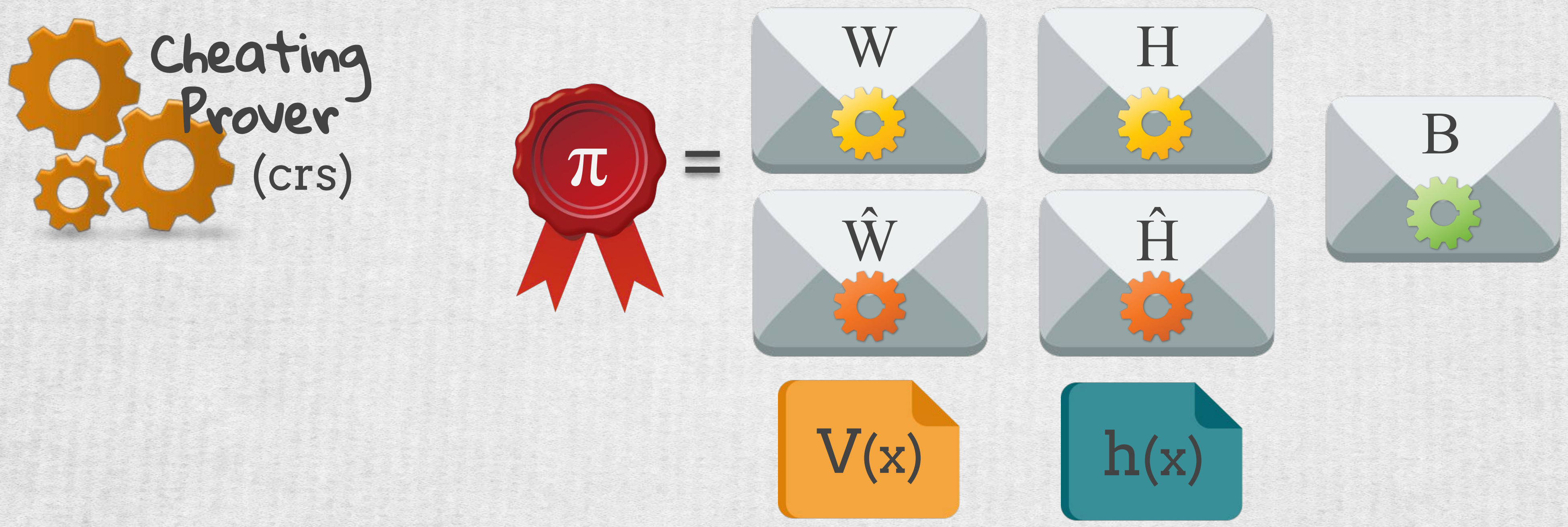
$$V(s) = v_0(s) + \sum_{i=1}^{m} a_i v_i(s)$$

V(x)

h(x)

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

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 $B = \mathsf{Enc}(\beta V(s))$

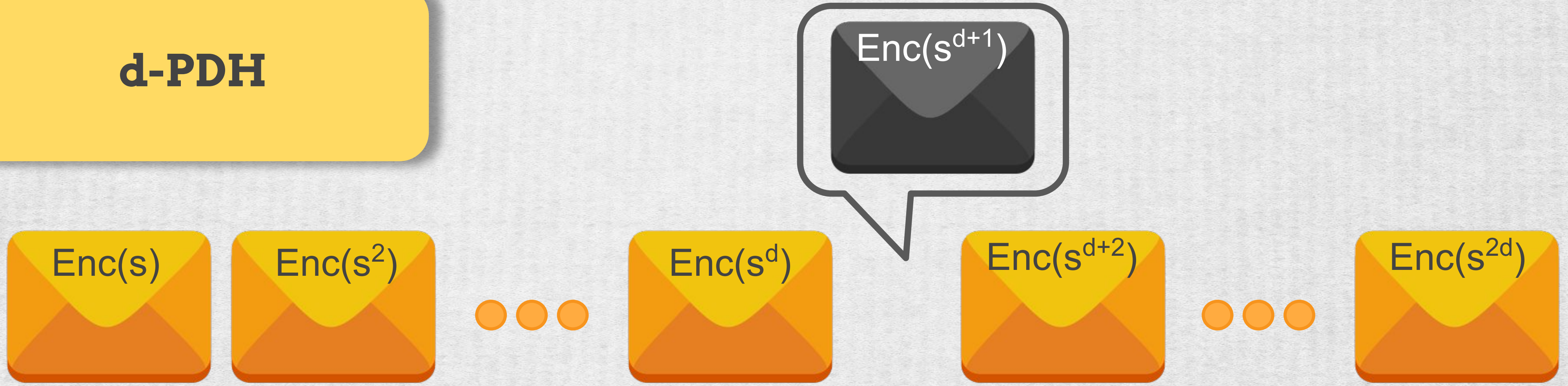
Assumption PDH: Power Diffie-Hellman

d-PDH

Enc(s) Enc(s^2) ••• Enc(s^d) Enc(s^{d+2}) ••• Enc(s^{2d})

Assumption PDH: Power Diffie-Hellman

d-PDH



Security Reduction: Cheating Prover to d-PDH

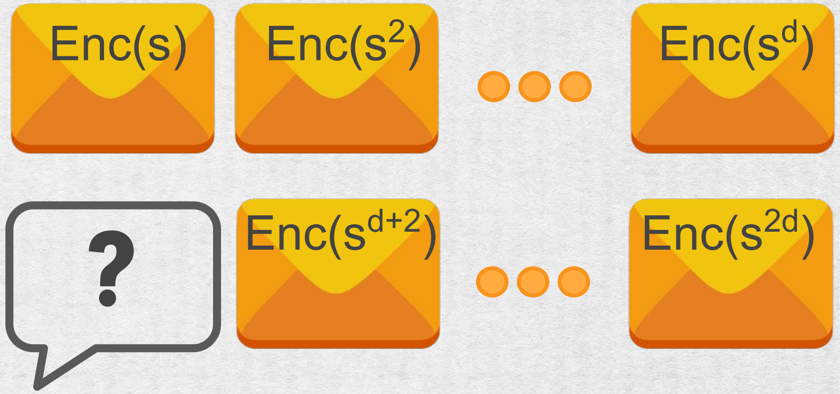
d-PDH

Enc(s) Enc(s^2) ... Enc(s^d)

? Enc(s^{d+2}) ... Enc(s^{2d})

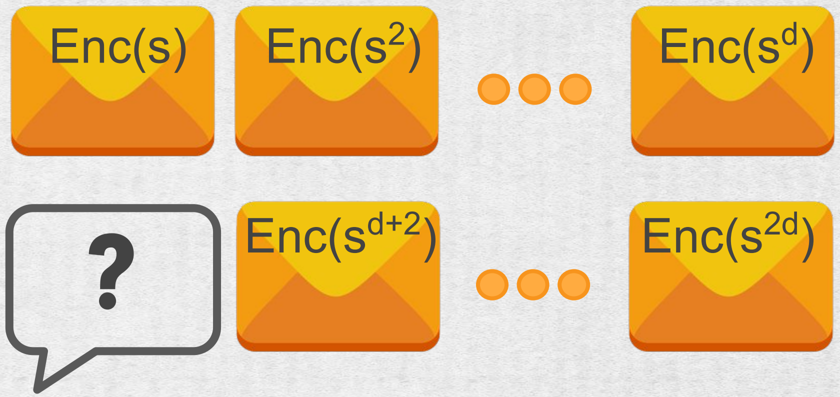
Security Reduction: Cheating Prover to d-PDH

d-PDH



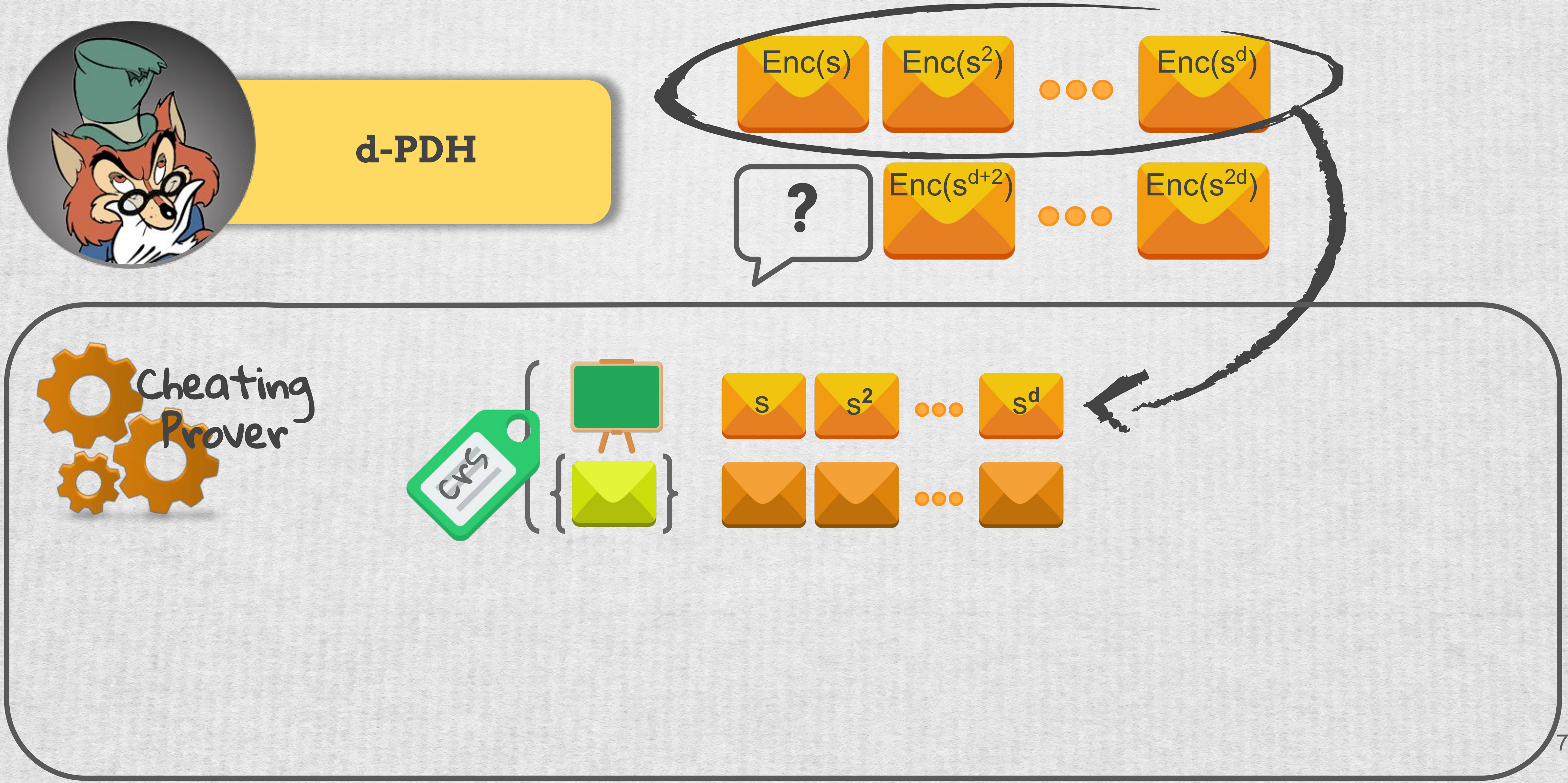
Security Reduction: Cheating Prover to d-PDH

d-PDH

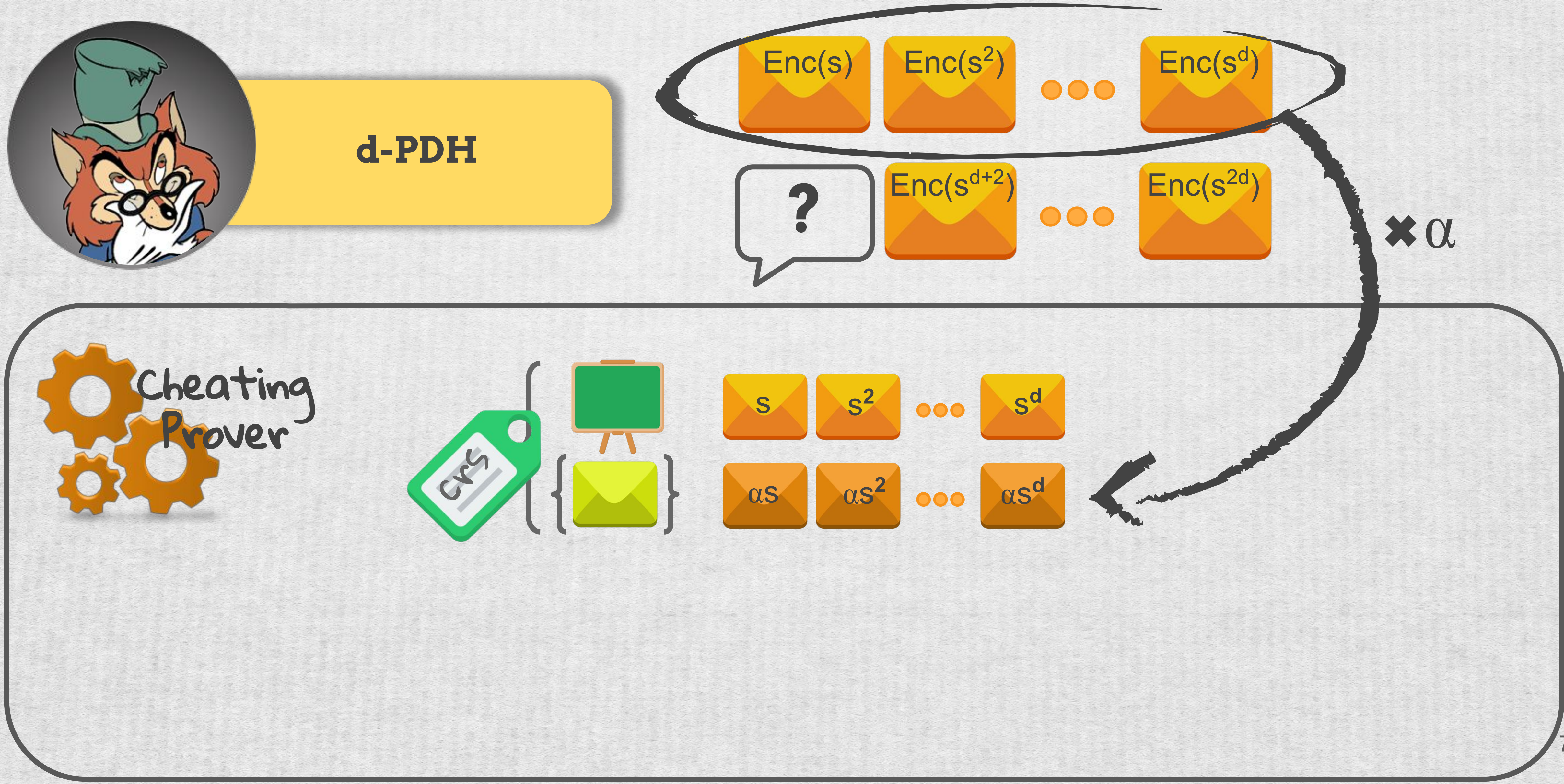


Cheating Prover

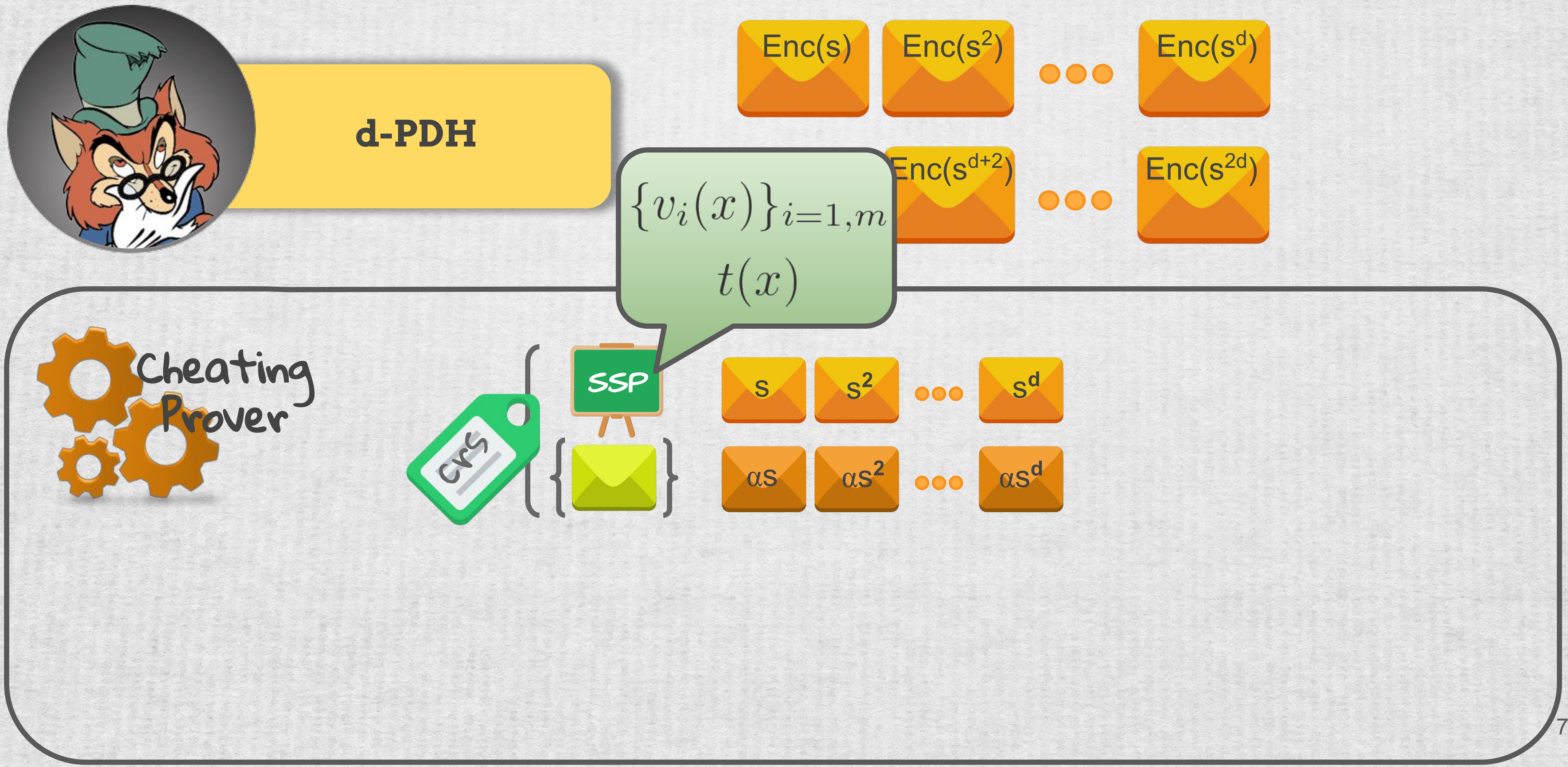
Security Reduction: Cheating Prover to d-PDH



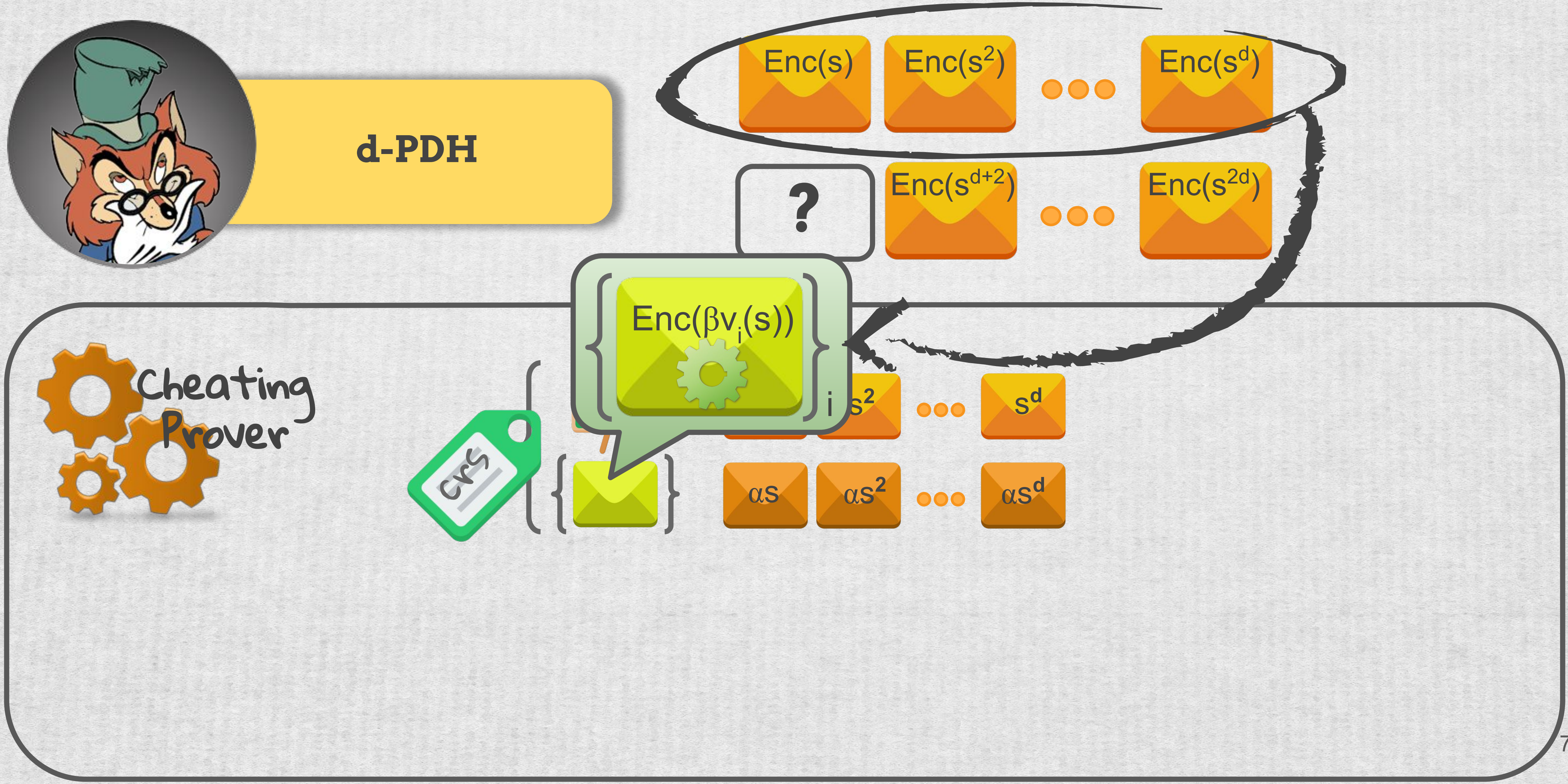
Security Reduction: Cheating Prover to d-PDH



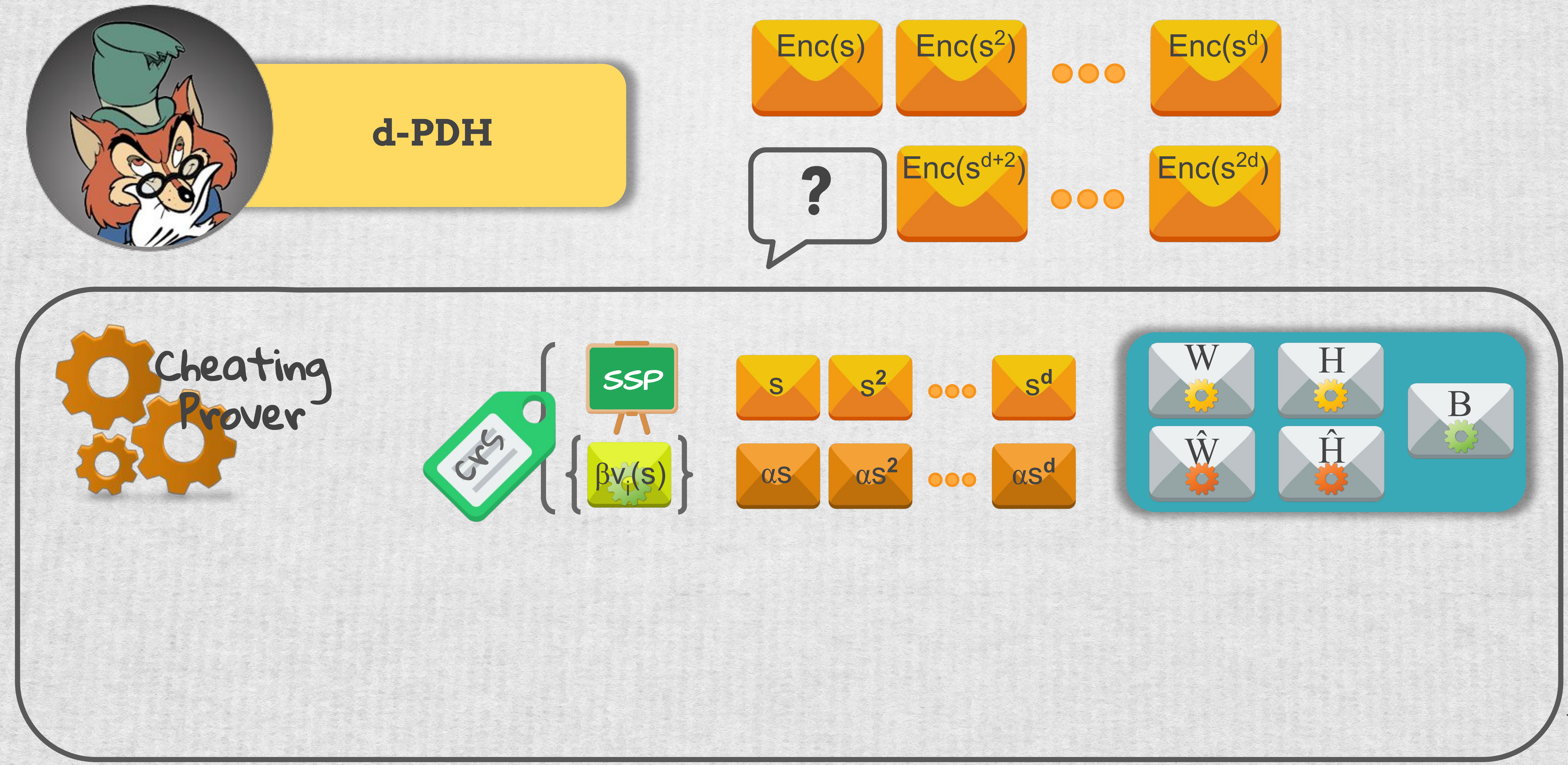
Security Reduction: Cheating Prover to d-PDH



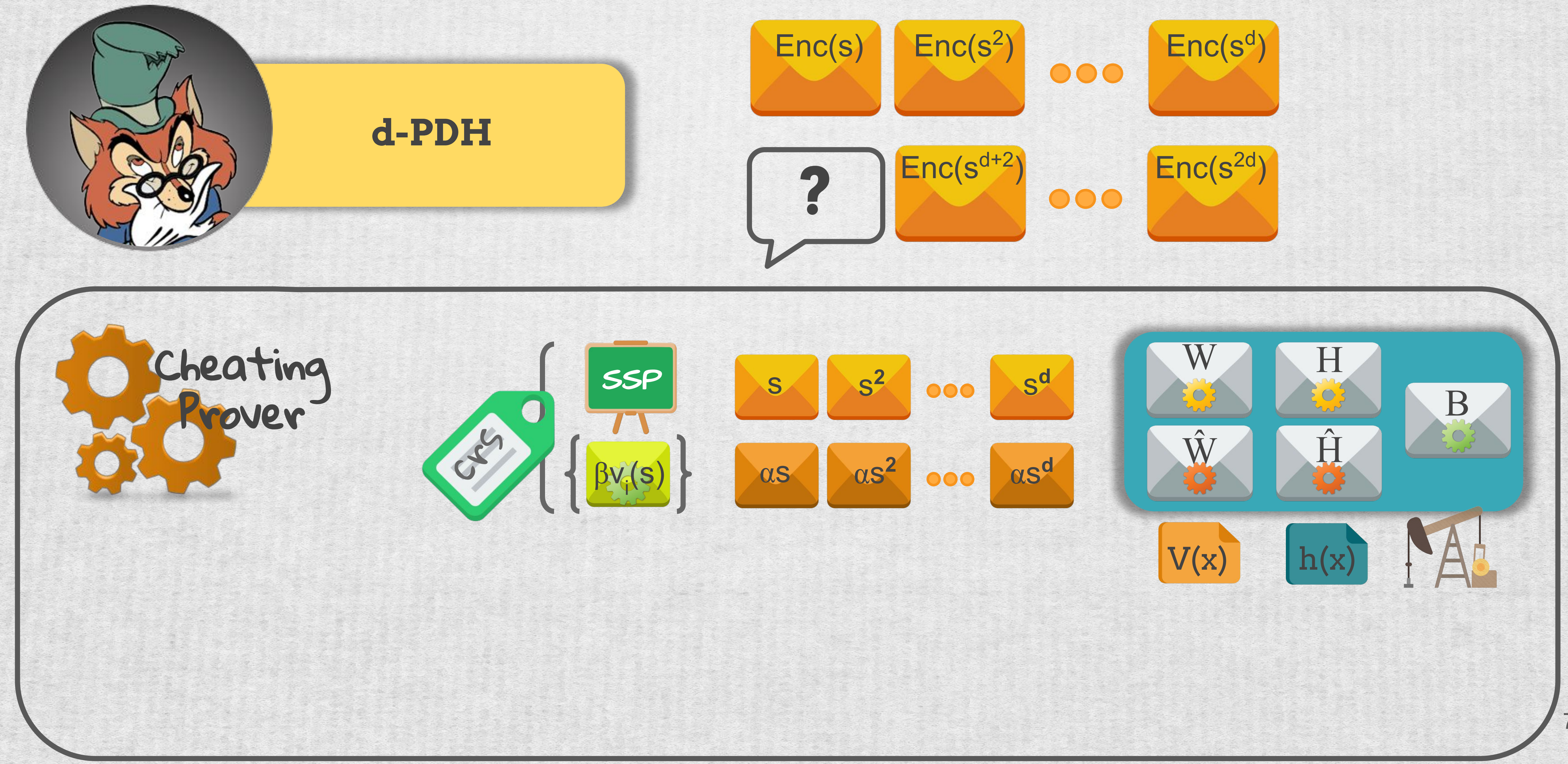
Security Reduction: Cheating Prover to d-PDH



Security Reduction: Cheating Prover to d-PDH



Security Reduction: Cheating Prover to d-PDH

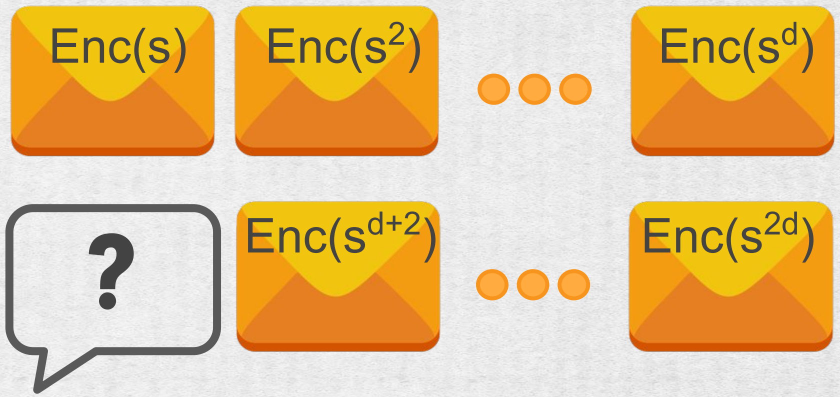


Security Reduction: Cheating Prover to d-PDH

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

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

Security Reduction: Cheating Prover to d-PDH

d-PDH

Enc(s^{d+1})

Enc(s) Enc(s^2) ... Enc(s^d) Enc(s^{d+2}) ... Enc(s^{2d})

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

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

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

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:

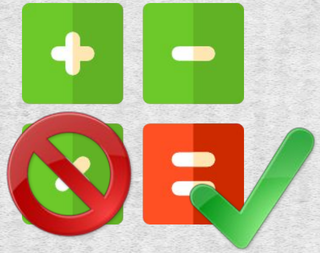
- linear operation using **crs**
- quadratic root detection using **crs**
- image verification using **crs**

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:

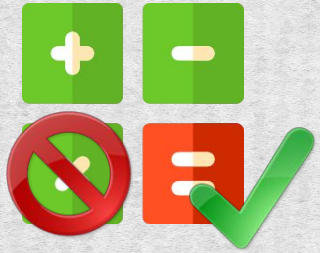
- linear operation using **crs**
- quadratic root detection using **crs**
- image verification using **crs**

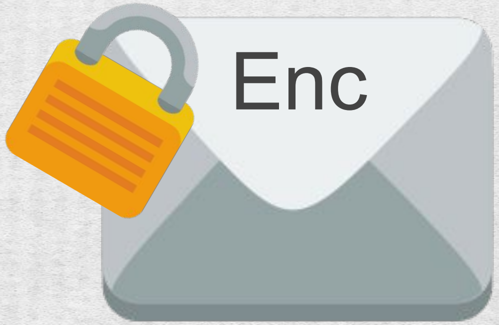
Designated Verifiable Encoding:

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

Prover

Verifier





Prover

Verifier

Security: Publicly Verifiable Encoding

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

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

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

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

Prover

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

Security: Publicly Verifiable Encoding

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

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

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

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

Prover

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

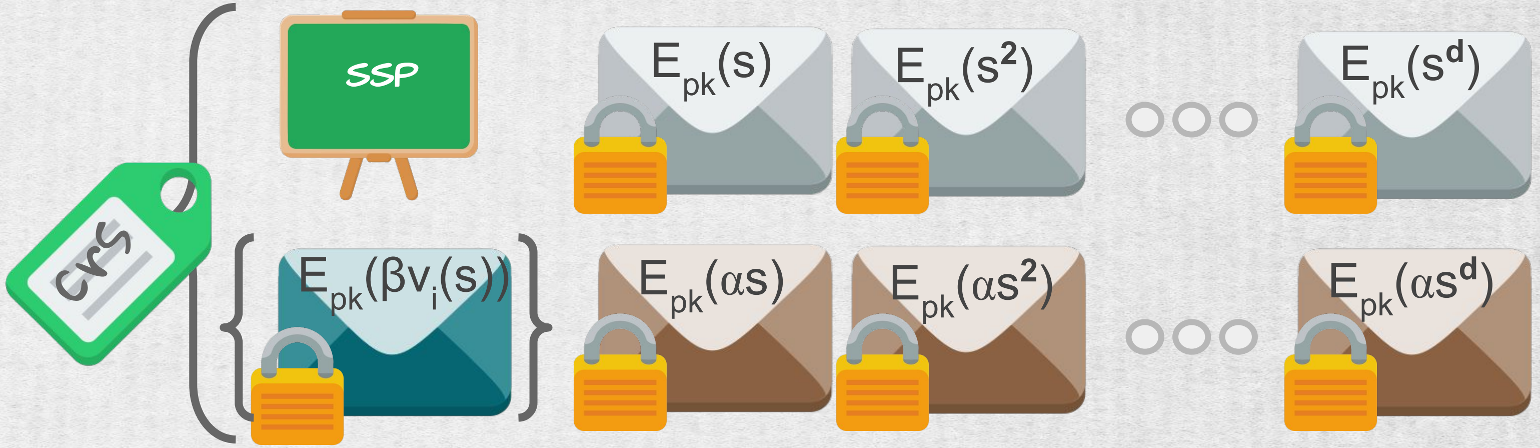
Verifier

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

Security: Designated Verifiable Encoding

Encryption: $E_{pk}(m) = c$

Decryption: $D_{sk}(c) = m$



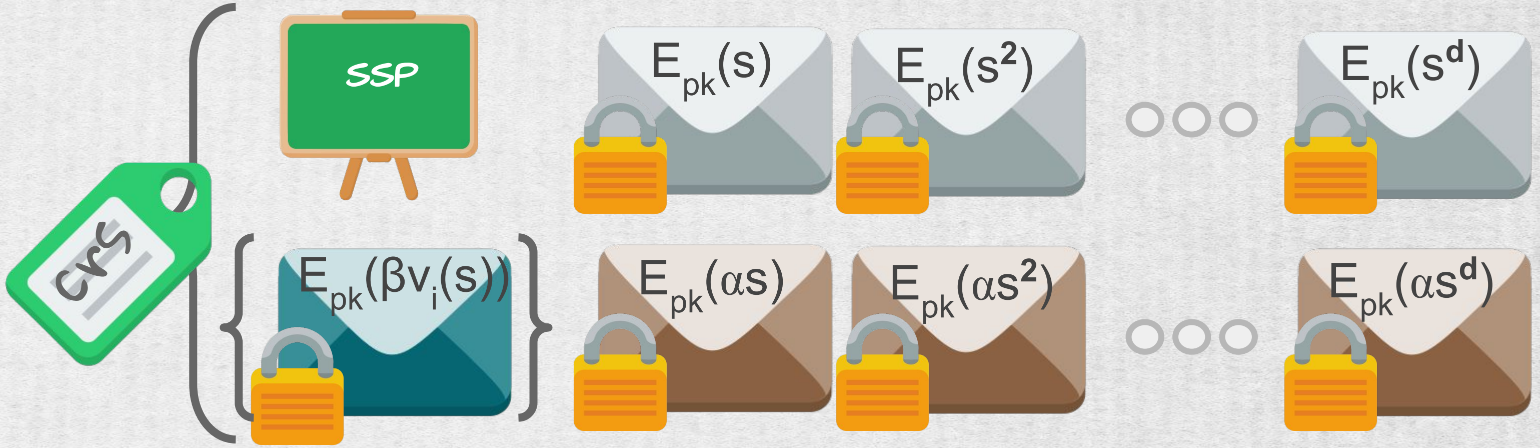
Prover

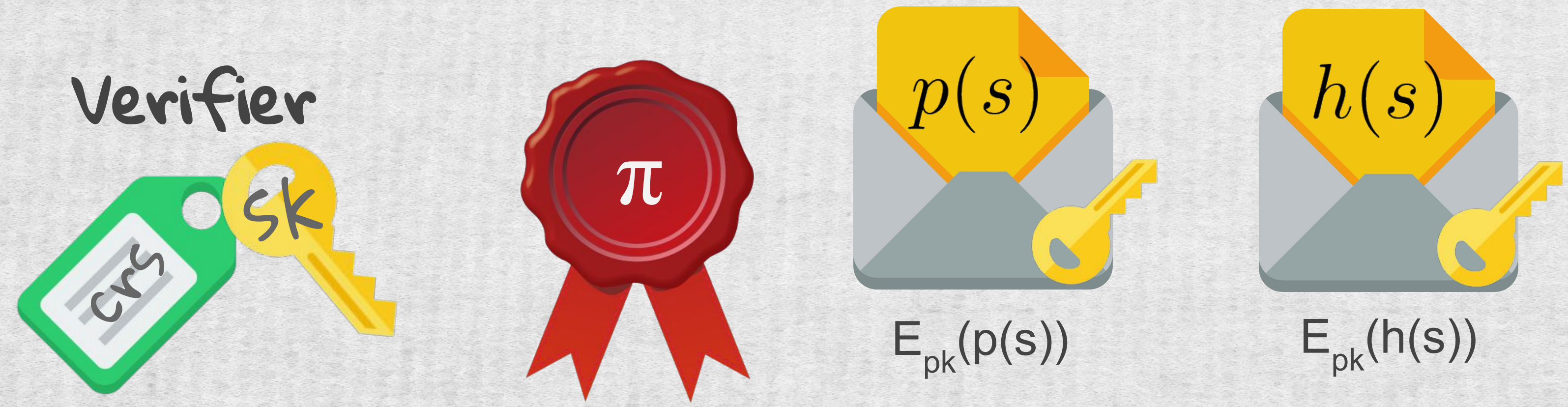
$$Enc(p(s)) = E_{pk}(p(s)) = E_{pk}\left(\sum p_i s^i\right) = \sum p_i E_{pk}(s^i)$$

Security: Designated Verifiable Encoding

Encryption: $E_{pk}(m) = c$

Decryption: $D_{sk}(c) = m$





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

Directions

Motivation

SNARKs

Construction

Directions

Post-quantum SNARK

Difficulties
Comparison

Open Problems

Conclusions

Lattice-based Encodings: Regev Encryption Scheme

error

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

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

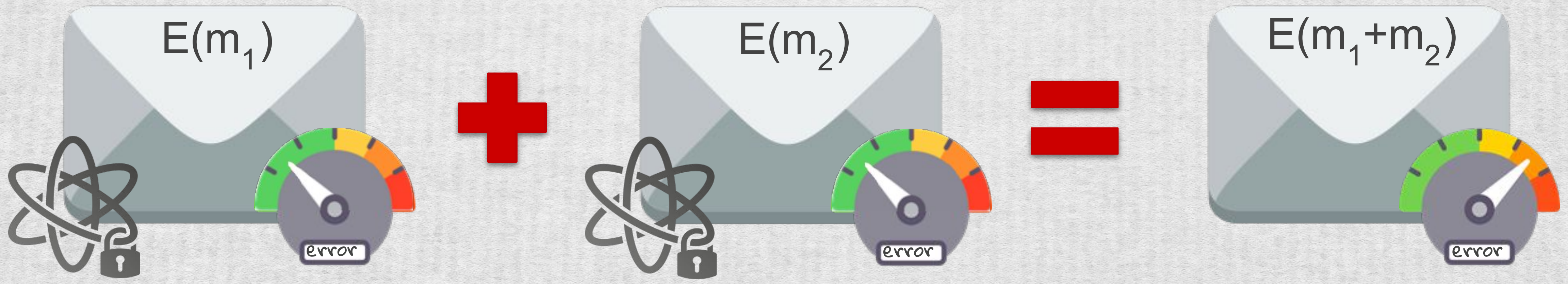
Lattice-based Encodings: Regev Encryption Scheme

error

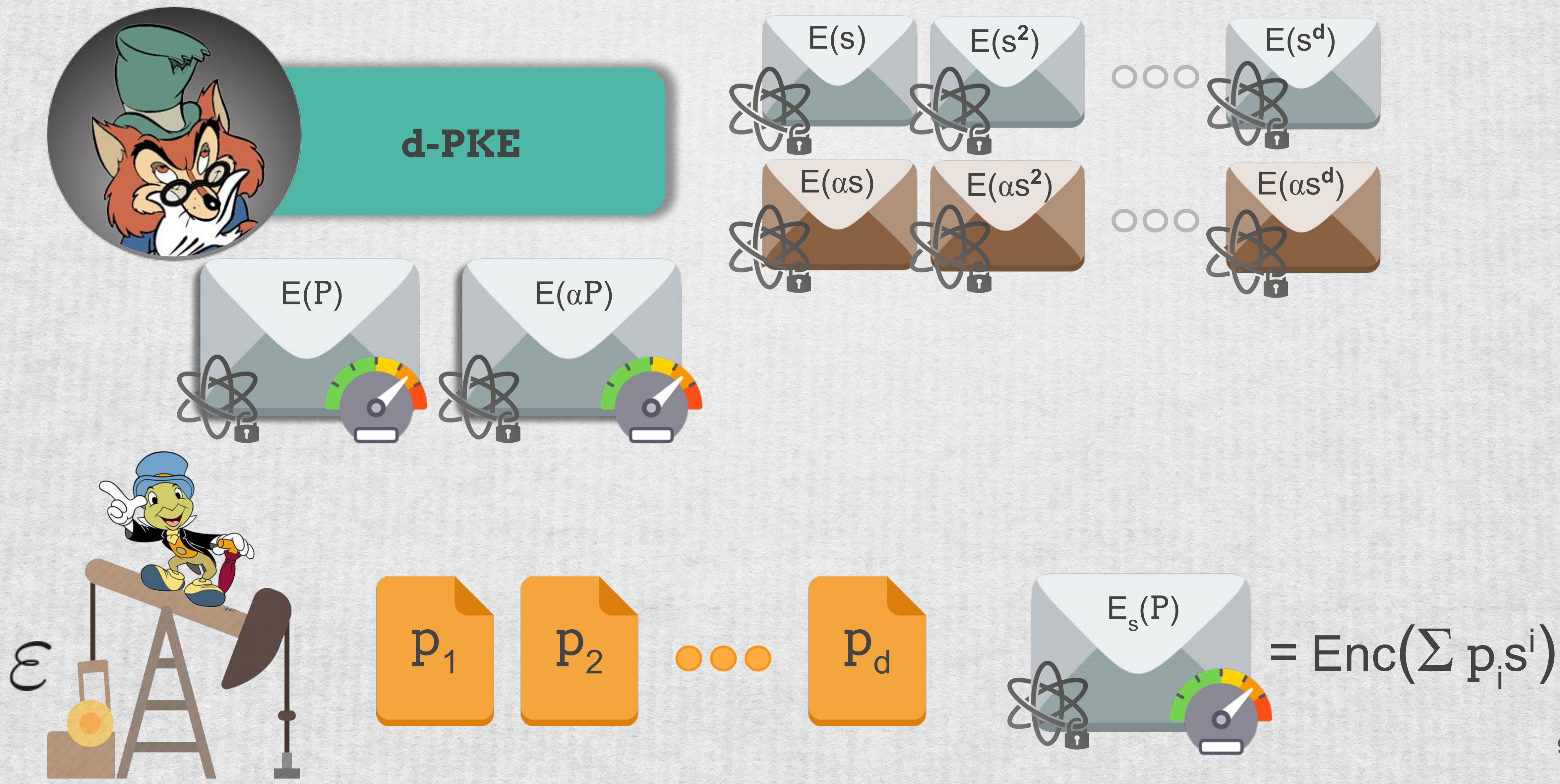
Encryption: $E_{\vec{s}}(m) = (-\vec{a}, \vec{a}\vec{s} + pe + m), \;\; e \in \chi$

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

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

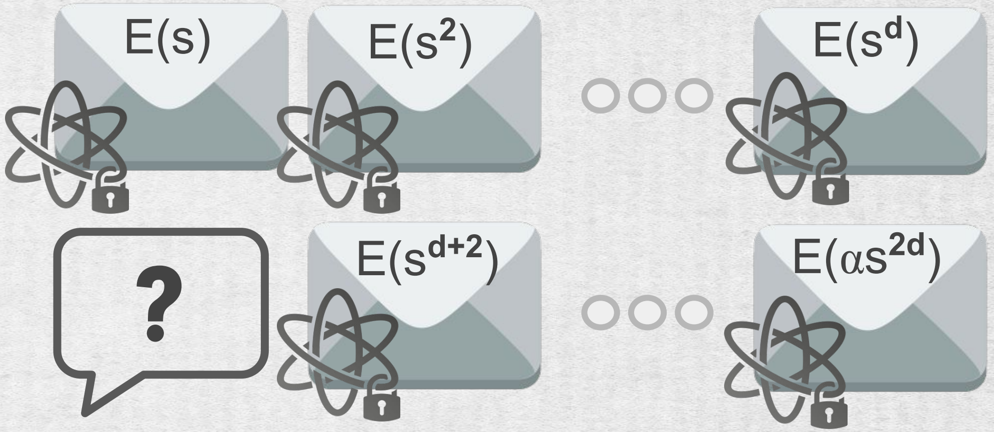


New Lattice Assumptions

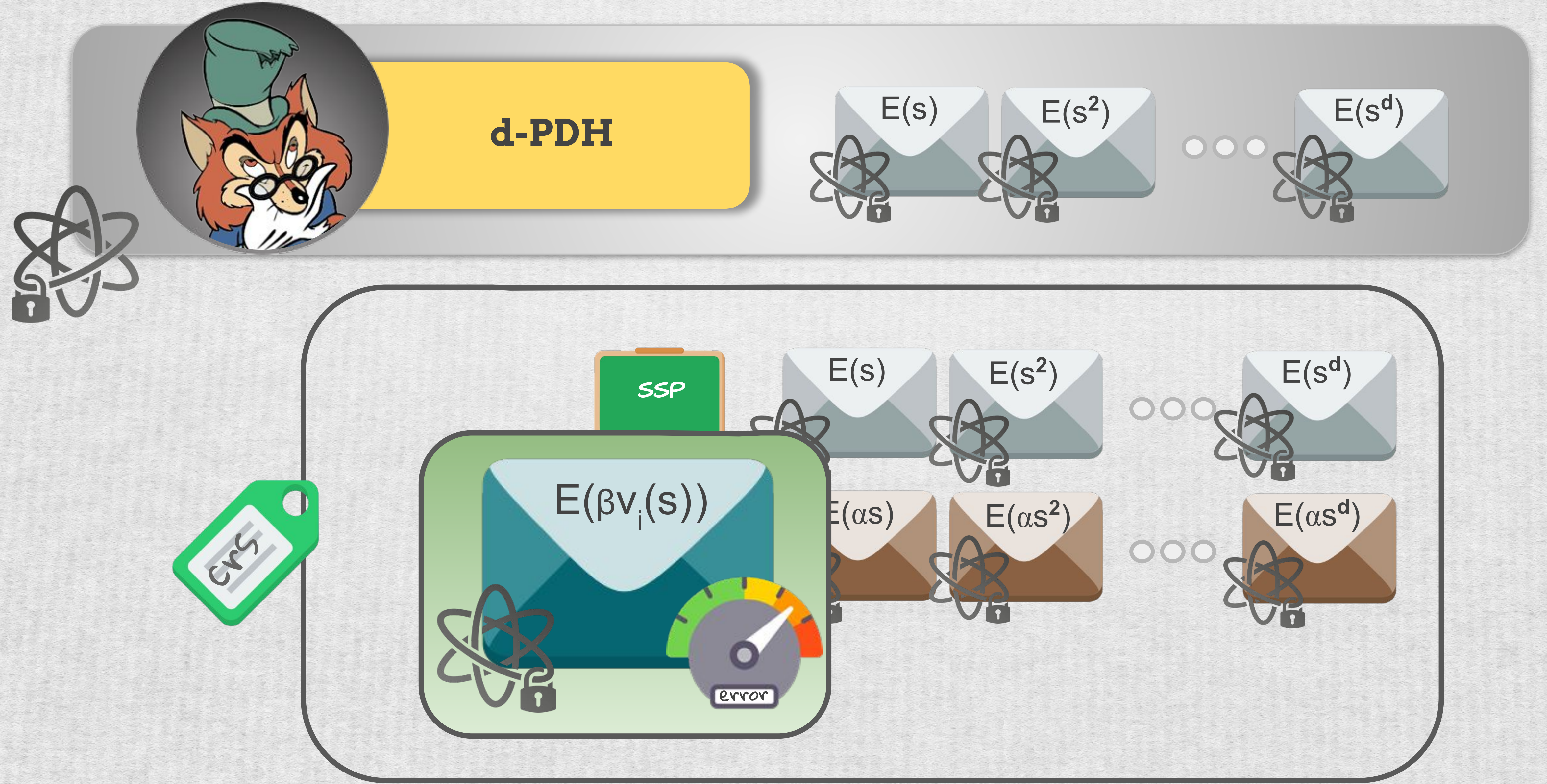


Assumption d-PDH

d-PDH



Technical Aspects



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

More trustful Cloud

THANK YOU

www.di.ens.fr/~nitulesc