CCA Secure Pseudorandom Codes in the Standard Model

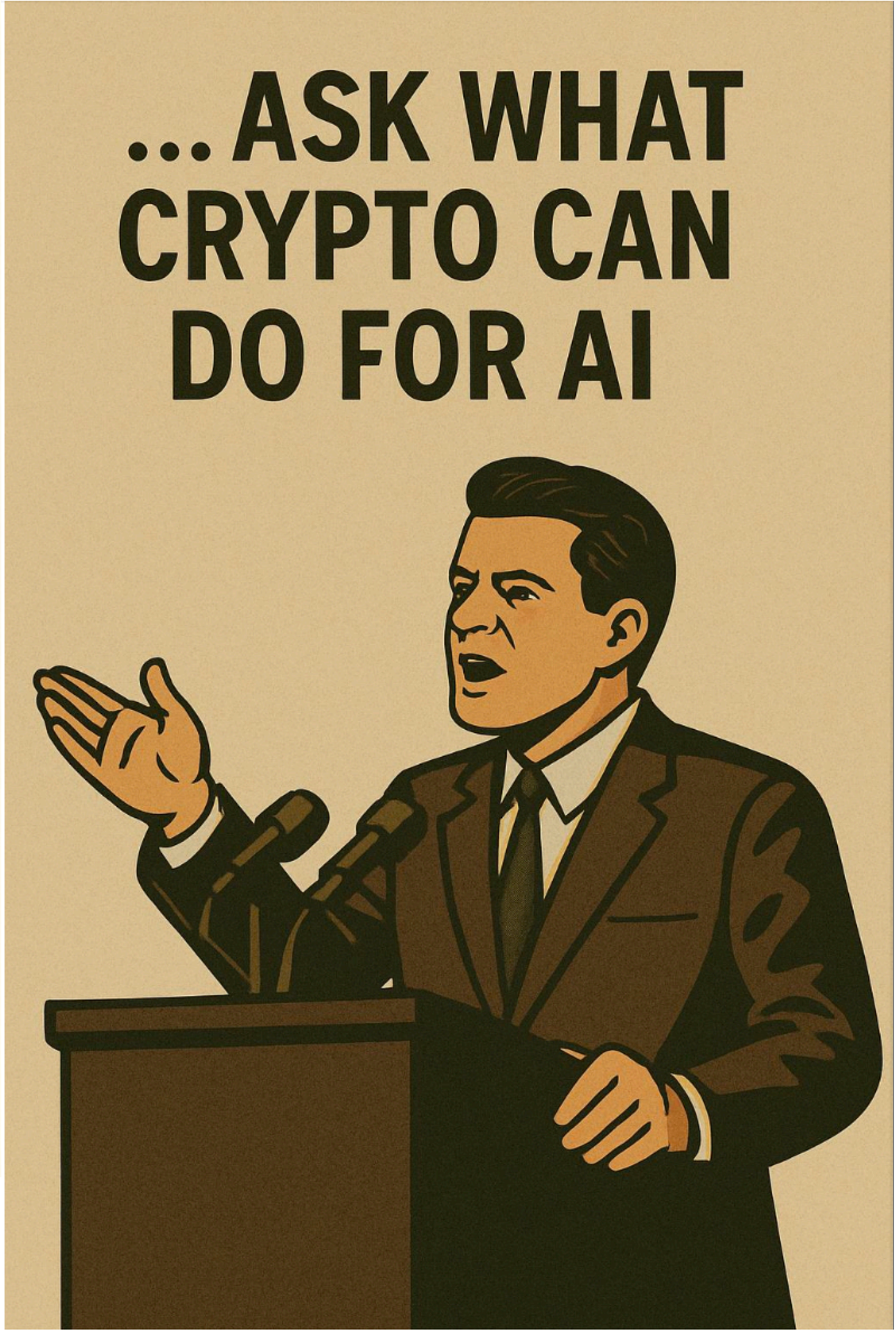
NICO DÖTTLING
CISPA

ANTOINE JOUX
CISPA

VENKATA KOPPULA
IIT DELHI

MAHESH RAJASREE
CISPA

HENDRIK WALDNER
CISPA



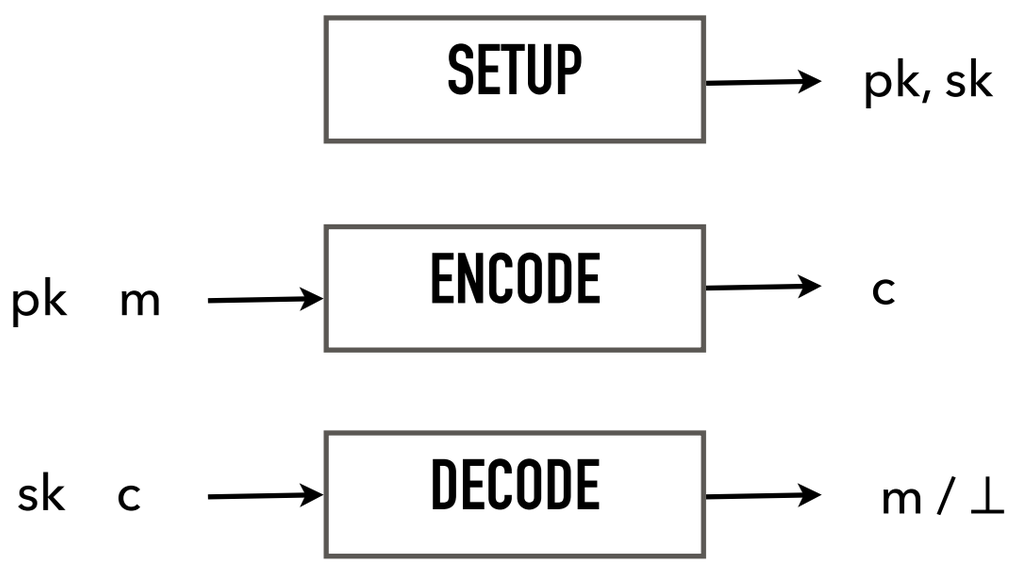
(Source: Vinod Vaikuntanathan's TCC 2025 talk)

INTRODUCTION:

CCA SECURE PSEUDORANDOM CODES

PSEUDORANDOM CODES (PRC) [CG 24]

Encryption with pseudorandom ciphertexts
where decryption can tolerate errors in ciphertext

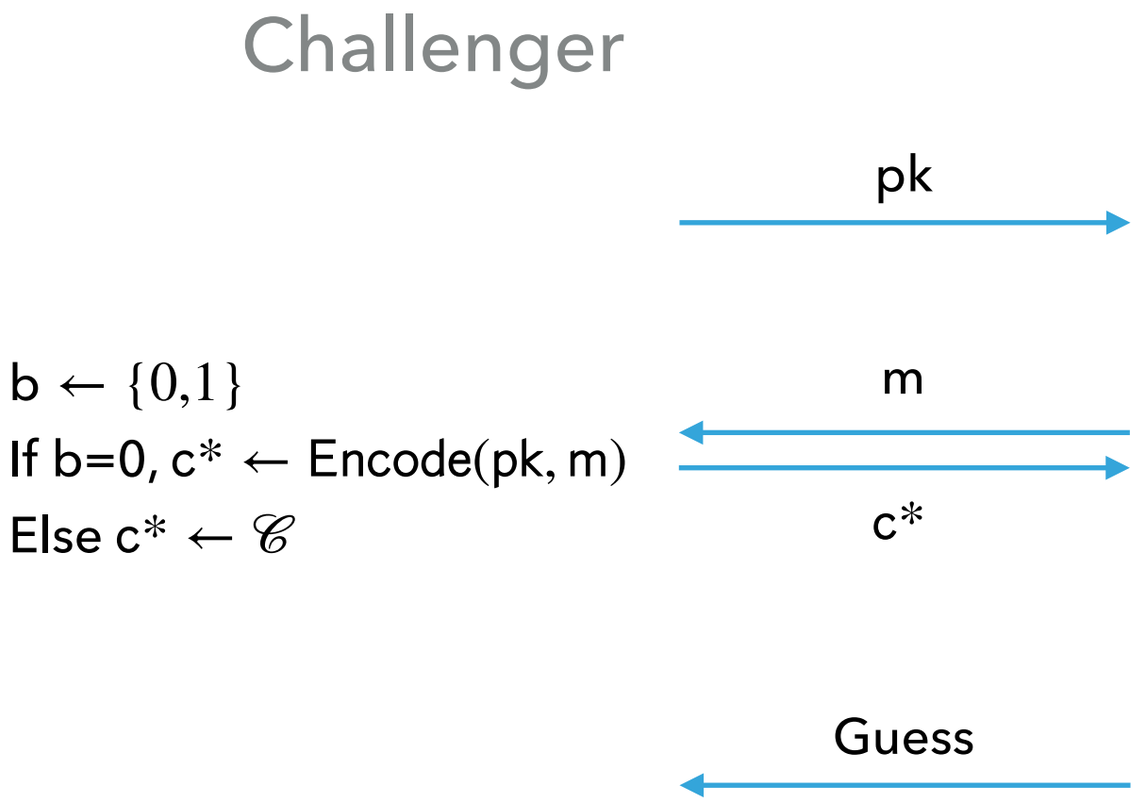


α-robust: Decode can tolerate α fraction of errors in codeword

Oblivious robustness | Adaptive robustness | Strong adaptive robustness

SECURITY AGAINST CHOSEN PLAINTEXT ATTACKS (CPA)

[CG 24]

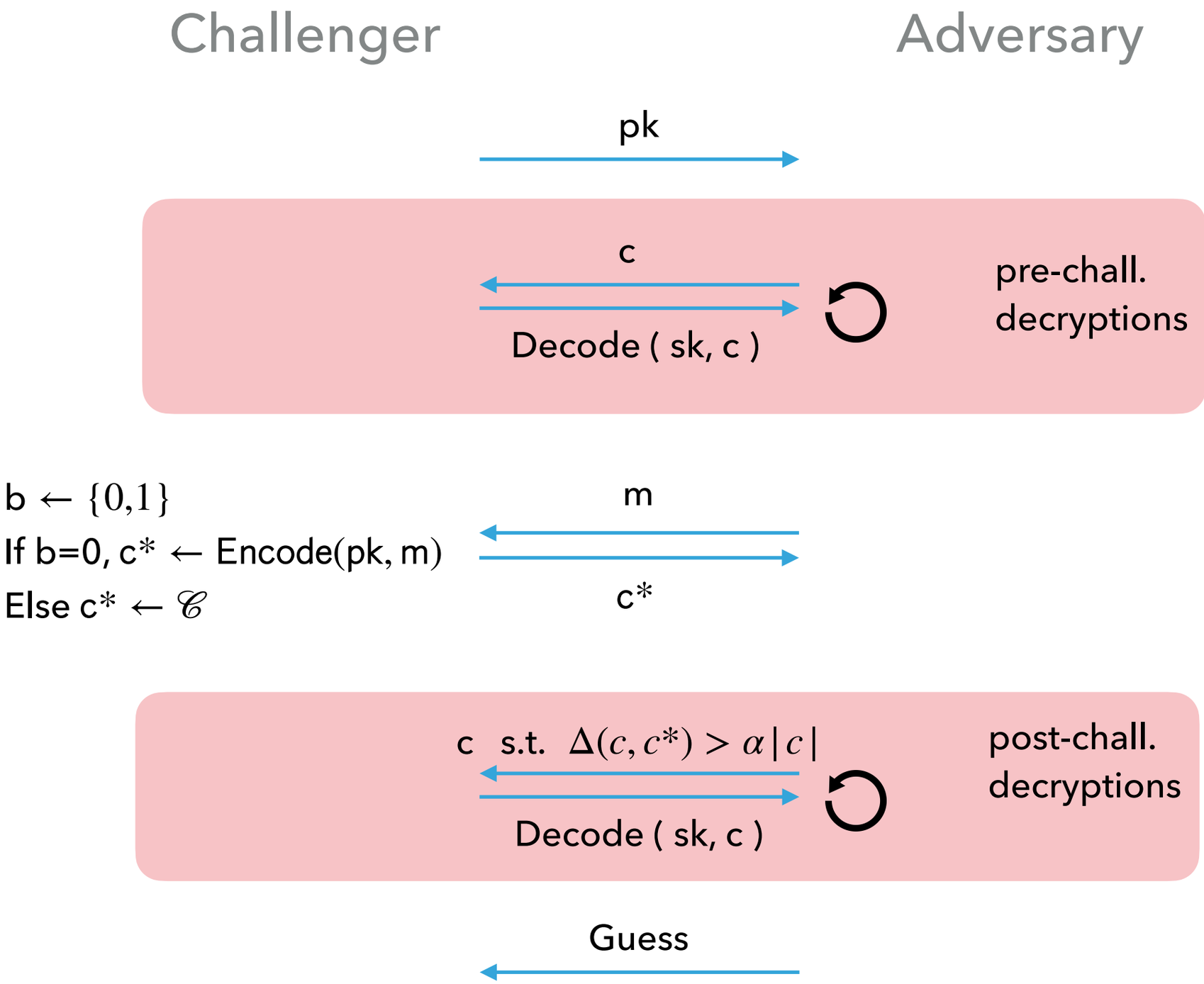


Constructions of CPA-PRC

[Christ-Gunn 24], [Golowich-Moitra 25], [Ghentiyala-Guruswami 25]
[Christ-Golowich-Gunn-Moitra-Wichs 26], [Dyseryn, Francati, Venturi 26],
[Döttling, Joux, K, Rajasree, Waldner 26], [Chen, Mao, Yi 26]

SECURITY AGAINST CHOSEN CODEWORD ATTACKS (CCA)

[AACDG 25]



CCA SECURE PRC [AACDG 25]

Prior work [AACDG25] :

CPA-PRC → CCA-PRC in the random oracle model

[Fujisaki-Okamoto 99] inspired transformation

Our work :

CPA-PRC + Hinting Pseudorandom Generators

→ CCA-PRC in the standard model

[K-Waters 19] inspired transformation

INGREDIENTS:

CPA-PRC

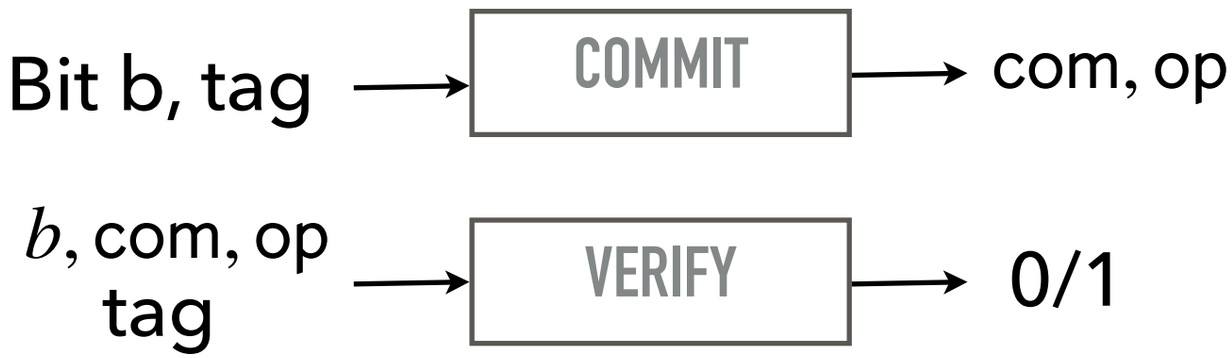
CPA-PKE

ONE-TIME SIGNATURES

TAGGED COMMITMENTS

HINTING PRG

TAGGED COMMITMENTS



Equivocal-Mode : $\text{com}^*, (\text{op}_0^*, \text{op}_1^*)$ for tag^*

SECURITY

Hard to generate $i, \text{com}, \text{op}_0, \text{op}_1$ such that

$\text{Verify}(\ 0, \text{com}, \text{op}_0, \text{tag}\ ) = \text{Verify}(\ 1, \text{com}, \text{op}_1, \text{tag}\ ) = 1$ for tag ≠ tag*

… can be constructed from any PRG [Naor 90]

INGREDIENTS:

CPA-PRC

CPA-PKE

ONE-TIME SIGNATURES

TAGGED COMMITMENTS

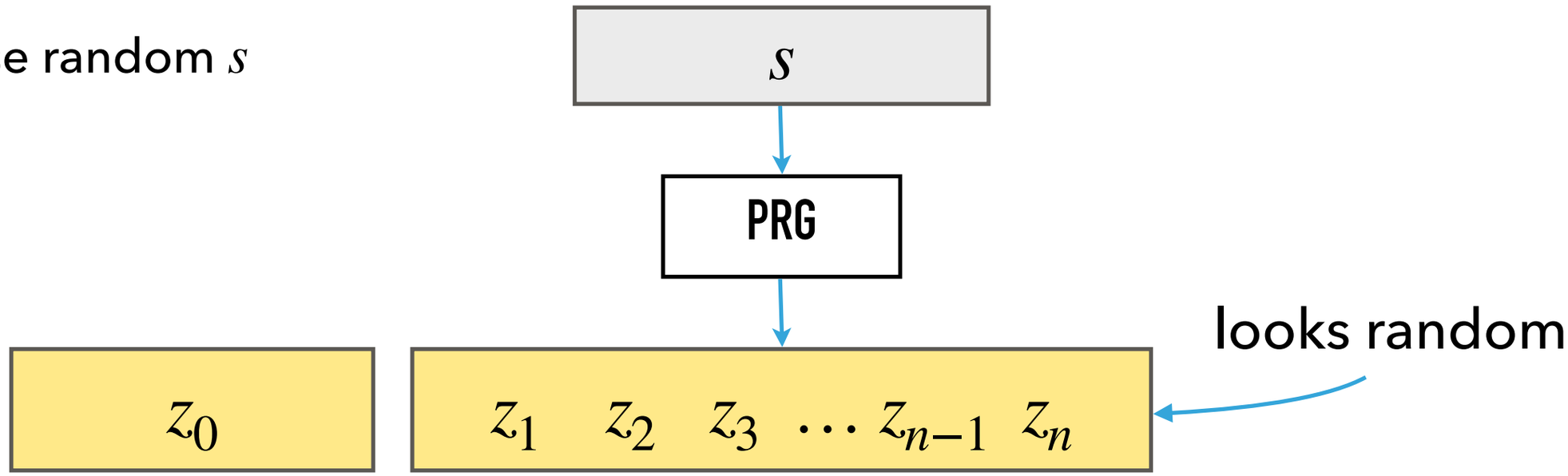
HINTING PRG

PSEUDORANDOM GENERATORS

$$\textbf{PRG}: \{0,1\}^n \rightarrow \{0,1\}^{n^2+n} = \{\{0,1\}^n\}^{(n+1)}$$

$(n+1)$ strings

Choose random s

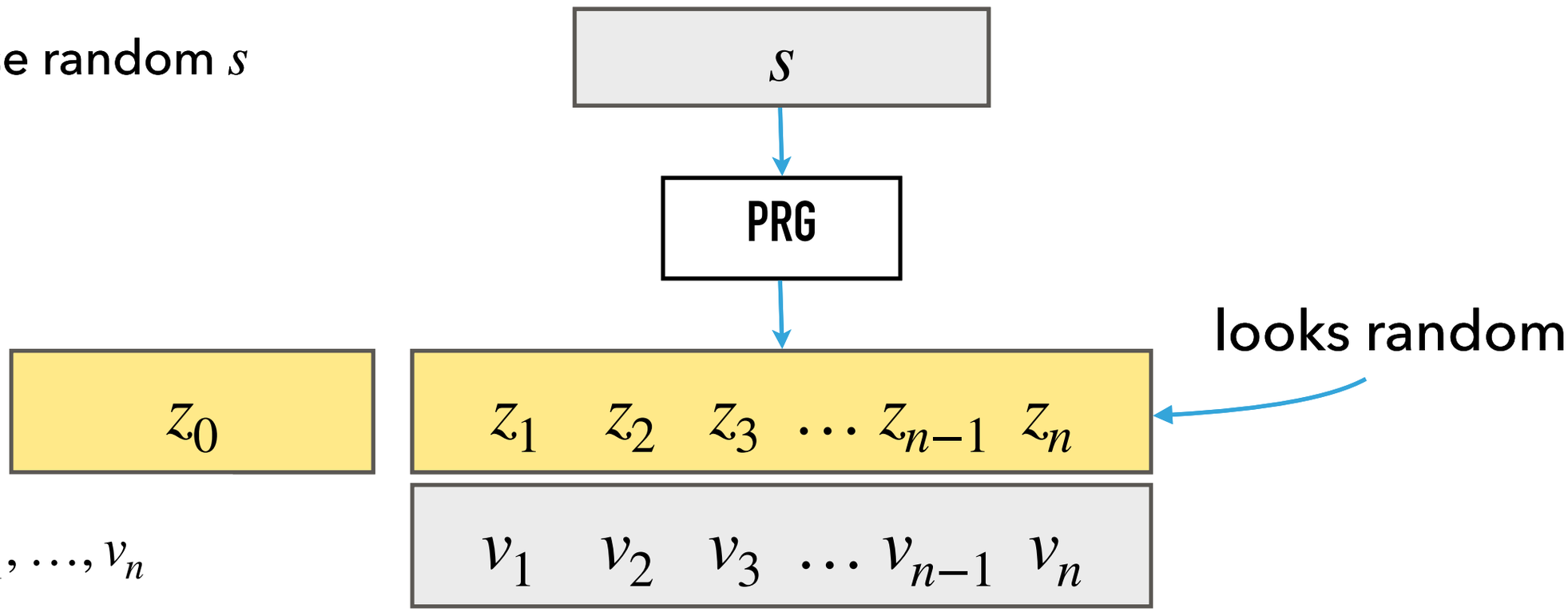


PSEUDORANDOM GENERATORS

PRG : $\{0,1\}^n \to \{0,1\}^{n^2+n} = \{\{0,1\}^n\}^{(n+1)}$

$(n+1)$ strings

Choose random s



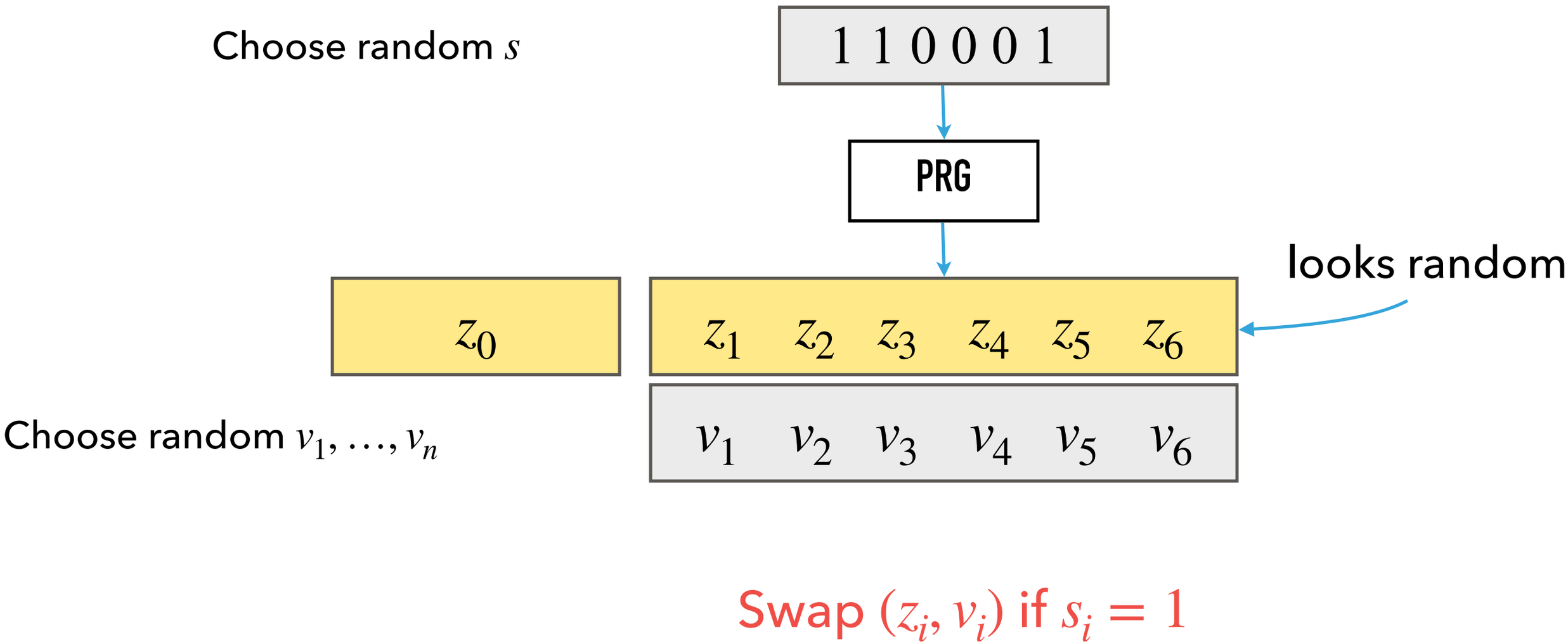
Choose random $v_1, \ldots, v_n$

Swap (z_i, v_i) if $s_i = 1$

PSEUDORANDOM GENERATORS

$$\textbf{PRG}: \{0,1\}^n \to \{0,1\}^{n^2+n} = \{\{0,1\}^n\}^{(n+1)}$$

$(n+1)$ strings



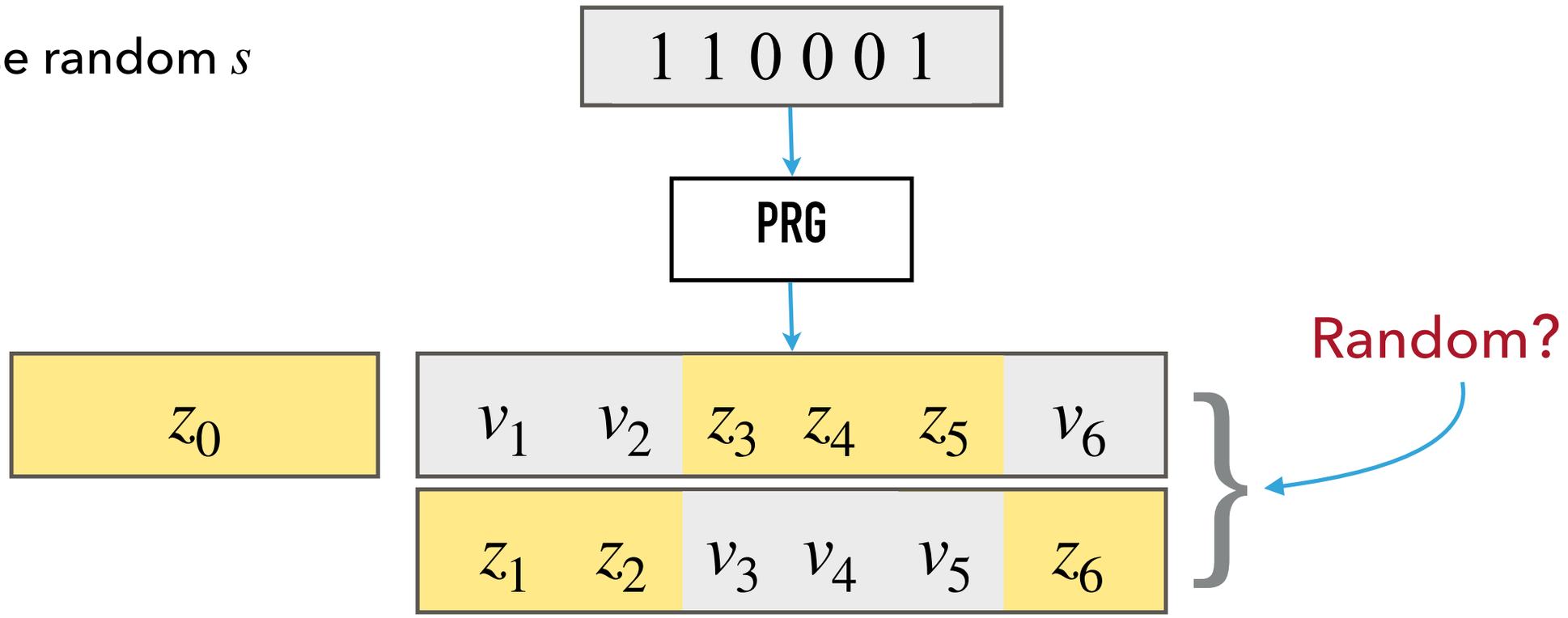
Swap (z_i, v_i) if $s_i = 1$

HINTING PSEUDORANDOM GENERATORS

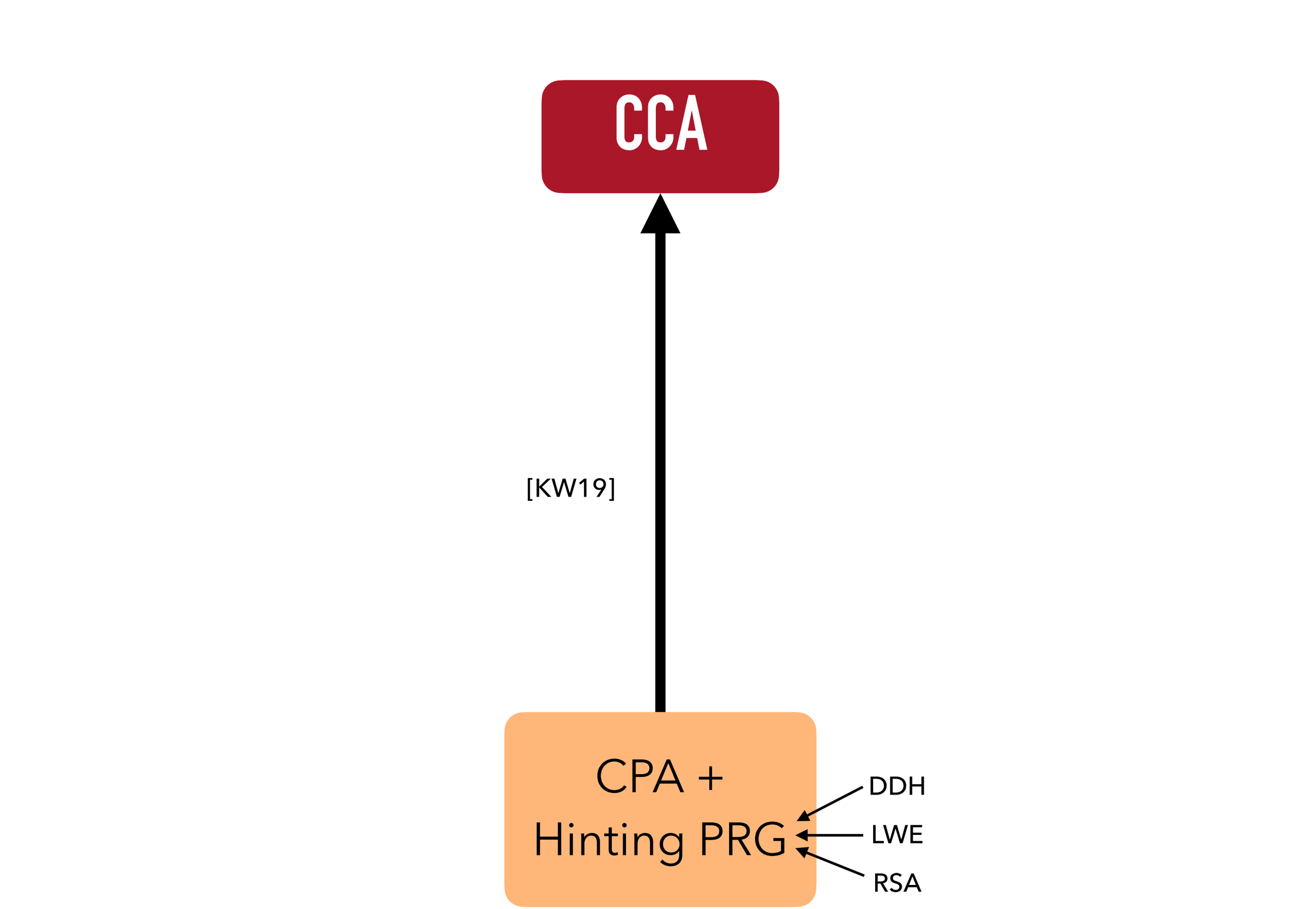
$$\textbf{PRG}: \{0,1\}^n \to \{0,1\}^{n^2+n} = \{\{0,1\}^n\}^{(n+1)}$$

$(n+1)$ strings

Choose random s



Swap (z_i, v_i) if $s_i = 1$


CCA
[KW19]
CPA +
Hinting PRG
DDH
LWE
RSA

THE KW-19 TEMPLATE

CPA-PKE + HPRG $\longrightarrow$ CCA-PKE

CPA Scheme

Decryption possible using encryption randomness

Hinting PRG : H

Supporting role: tagged commitment scheme, one-time signature scheme

SETUP

PK:

$\mathsf{pk}_{1,0}$ $\mathsf{pk}_{2,0}$ … $\mathsf{pk}_{n,0}$

$\mathsf{pk}_{1,1}$ $\mathsf{pk}_{2,1}$ … $\mathsf{pk}_{n,1}$

SK:

$\mathsf{sk}_{1,0}$ $\mathsf{sk}_{2,0}$ … $\mathsf{sk}_{n,0}$

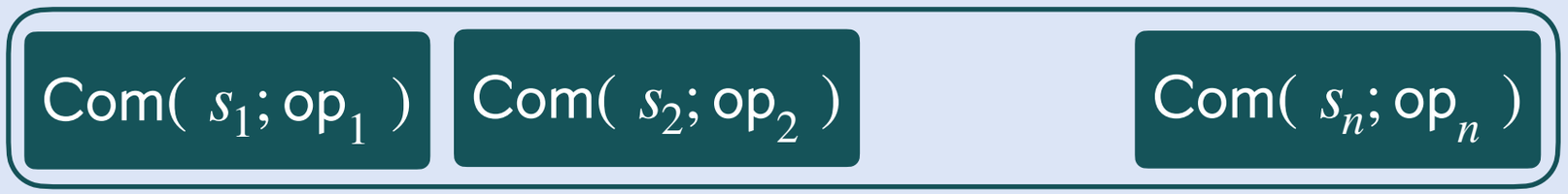
Could have used any $\{\mathsf{sk}_{i,b_i}\}_i$

ENCRYPT (PK , m)

$$s \leftarrow \{0,1\}^n \,.\, \mathsf{H}(s) = (z_0, z_1, \ldots, z_n).$$

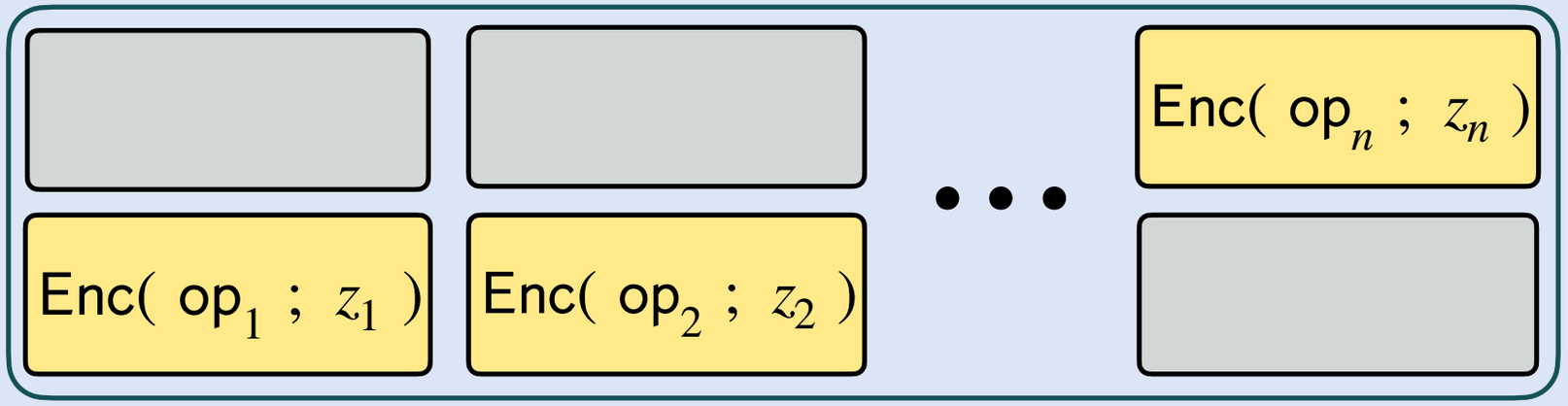
(sigk, vk) : signing/verification keys

Commitment to s with tag vk



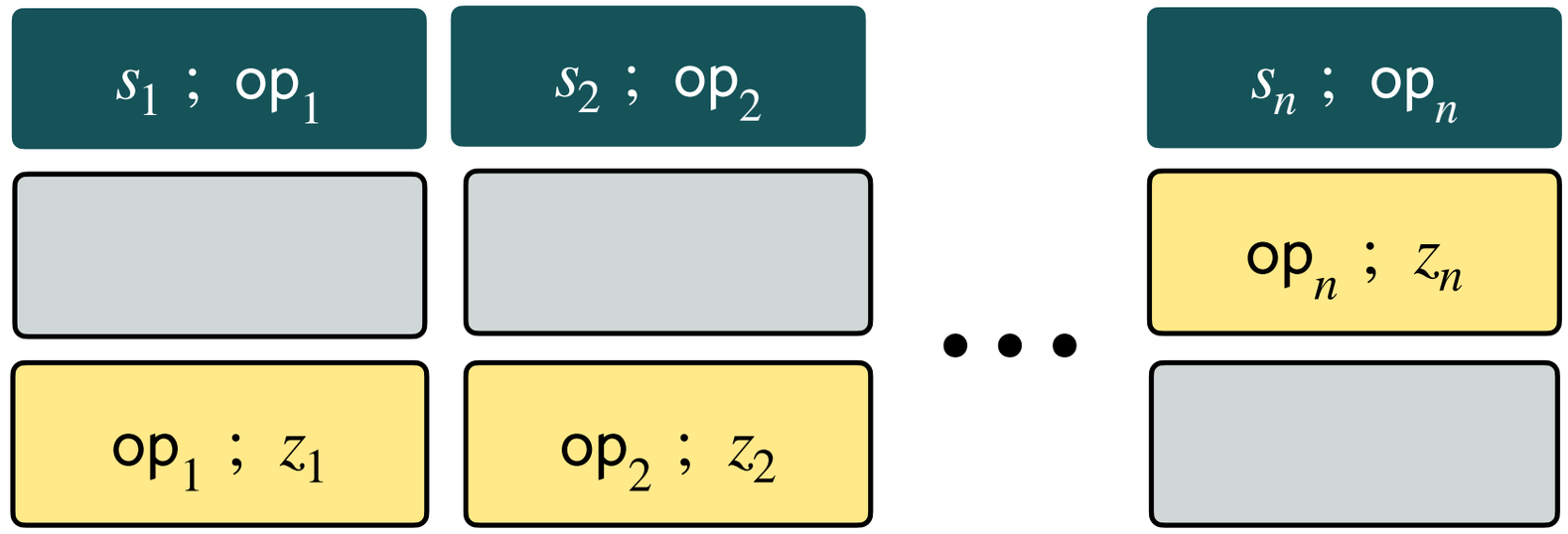
$\mathsf{ct}_0 = z_0 \oplus m$

The $2n$ ciphertexts are used to 'signal' s



$\sigma \leftarrow$ Sign all commitment and ciphertext components using sigk

$\mathsf{ct}_0 = z_0 \oplus m$



σ

vk

DECRYPT (SK, CT)

Recover s using $\{\mathsf{ct}_{i,0}\}_i$ and $\{\mathsf{sk}_{i,0}\}_i$.

$\mathsf{H}(s) = (z_0\ z_1\ \ \ldots\ \ z_n)$. Use z_i to decrypt ct_{i,s_i}.

Perform consistency checks.

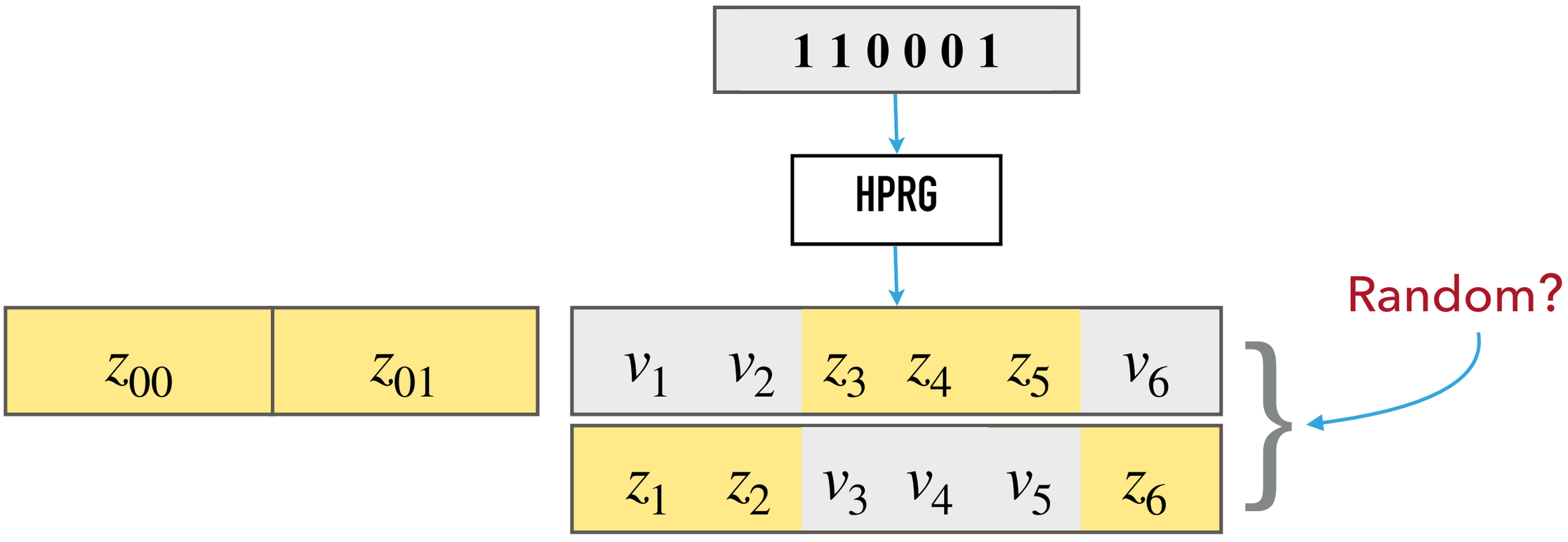
Verify σ using sig . vk

Output $\mathsf{ct}_0 \oplus z_0$

Can switch the decryption oracle to use $\{\mathsf{sk}_{i,b_i}\}_i$

Indistinguishable for any adversary

OUR CCA-PRC CONSTRUCTION



MAKING KW-19 CIPHERTEXTS PSEUDORANDOM

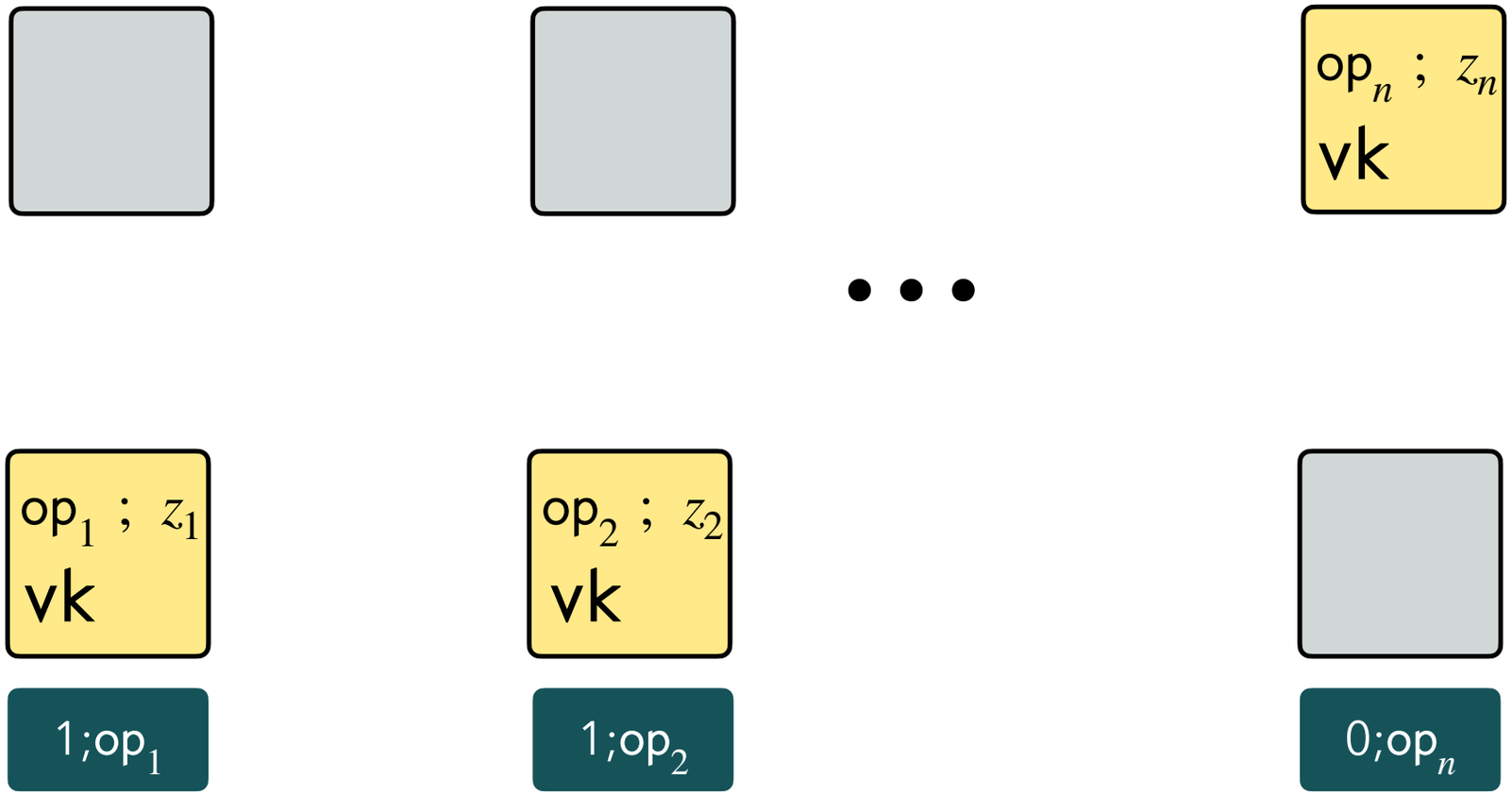
$z_{00} \oplus m$

$z_{01} \oplus \sigma$

vk

- ➡ Put vk inside PKE ciphertext, mask σ
- ➡ PKE ciphertexts, and commitments should be pseudorandom

CCA-PKE with pseudorandom ciphertexts
[Kitagawa, Matsuda, Tanaka 19]



ADDING ROBUSTNESS

$z_{00} \oplus \mathrm{ECC}(m)$

$z_{01} \oplus \mathrm{ECC}(\sigma)$

PRC.Encode

$1;\mathrm{op}_1$

$\mathrm{prc}.\mathrm{pk}_{1,0}$

PRC.Encode

$1;\mathrm{op}_2$

$\mathrm{prc}.\mathrm{pk}_{2,0}$

PRC.Encode

$0;\mathrm{op}_n$

$\mathrm{prc}.\mathrm{pk}_{n,0}$

op_n ; z_n

vk

Cannot switch the decryption oracle :(

• • •

PRC.Encode

$\mathrm{prc}.\mathrm{pk}_{1,1}$

op_1 ; z_1

vk

$1;\mathrm{op}_1$

PRC.Encode

$\mathrm{prc}.\mathrm{pk}_{2,1}$

op_2 ; z_2

vk

$1;\mathrm{op}_2$

PRC.Encode

$\mathrm{prc}.\mathrm{pk}_{n,1}$

$0;\mathrm{op}_n$

MAKING IT CCA SECURE

$z_{00} \oplus \mathsf{ECC}(\ m,\ \{\mathsf{ct}_{i,b}, \mathsf{com}_i\}\ ,$ randomness used for PRC encoding $)$

$z_{01} \oplus \mathsf{ECC}(\sigma)$

PRC.Encode
$1;\mathsf{op}_1$
$\mathsf{prc}.\mathsf{pk}_{1,0}$

PRC.Encode
$1;\mathsf{op}_2$
$\mathsf{prc}.\mathsf{pk}_{2,0}$

PRC.Encode
$0;\mathsf{op}_n$
$\mathsf{prc}.\mathsf{pk}_{n,0}$
$\mathsf{op}_n\ ;\ z_n$
vk

• • •

PRC.Encode
$\mathsf{prc}.\mathsf{pk}_{1,1}$
$\mathsf{op}_1\ ;\ z_1$
vk
$1;\mathsf{op}_1$

PRC.Encode
$\mathsf{prc}.\mathsf{pk}_{2,1}$
$\mathsf{op}_2\ ;\ z_2$
vk
$1;\mathsf{op}_2$

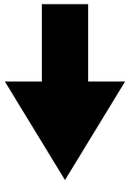
PRC.Encode
$\mathsf{prc}.\mathsf{pk}_{n,1}$
$0;\mathsf{op}_n$

Can switch the decryption oracle to use $\{\mathsf{sk}_{i,b_i}\}_i$

Indistinguishable for any adversary

CONCLUSIONS, CONCURRENT / FOLLOW-UP WORKS

CPA-PRC scheme with robustness α



Hinting PRGs

CCA-PRC scheme with robustness $\Omega(\alpha)$

Concurrent work [Chen, Mao, Yi 2026]

CCA1-PRC, using hinting PRG framework, robustness $\approx 0.0005\alpha$

Follow-up work [Döttling, Joux, K, Rajasree, Waldner 2026]

CPA-PRC + CCA-PKE $\rightarrow$ CCA-PRC : simpler construction, robustness $\approx \alpha/2$

Detectable-CCA framework of Bishop, Hohenberger, Waters 2012

OPEN QUESTIONS

α-robust CPA-PRC + <insert favorite non-PRC primitive here> $\rightarrow$ α-robust CCA-PRC?

Direct constructions of CCA-PRC ?

THANK YOU!