RSAConference2019

San Francisco | March 4 – 8 | Moscone Center

BETTER.

SESSION ID: CRYPT W12

Post-Quantum Cryptography

Mélissa Rossi

PhD Student
Thales & ENS Paris

Saba Eskandarian

PhD Student
Stanford University

Assessment of the Key-Reuse Resilience of NewHope

Aurélie Bauer - Henri Gilbert - Guénaël Renault - Mélissa Rossi

Outline

- NewHope
- Key caching and attack model
- An attack on the CPA version
- The attack can be extended to the CCA version with a stronger model

RSAConference2019

NewHope

NewHope is a Key Encapsulation Mechanism (KEM)

NewHope is a Key Encapsulation Mechanism (KEM)

1. **Key Generation**

   A pair (pk,sk) is drawn

NewHope is a Key Encapsulation Mechanism (KEM)

1. Key Generation

A pair (pk,sk) is drawn

pk →

2. Encapsulation

A random key is drawn and encapsulated with Alice's public key

NewHope is a Key Encapsulation Mechanism (KEM)

1. **Key Generation**

A pair (pk,sk) is drawn

pk →

2. **Encapsulation**

A random key is drawn and encapsulated with Alice's public key

2. **Decapsulation**

← Encapsulated key

Alice recovers Bob's random key using her secret key

NewHope is lattice based KEM

The elements are polynomials in $\dfrac{\mathbb{Z}_q[x]}{x^N+1}$

$$q = 12289 \qquad N = 1024$$

NewHope is lattice based KEM

The elements are polynomials in $\frac{\mathbb{Z}_q[x]}{x^N+1}$

$$q = 12289 \qquad N = 1024$$

Based on the hardness of **RLWE problem**

$\mathbf{A} \leftarrow$ uniformly random

$\mathbf{S}, \mathbf{E} \leftarrow$ small coefficients (binomial distribution)

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

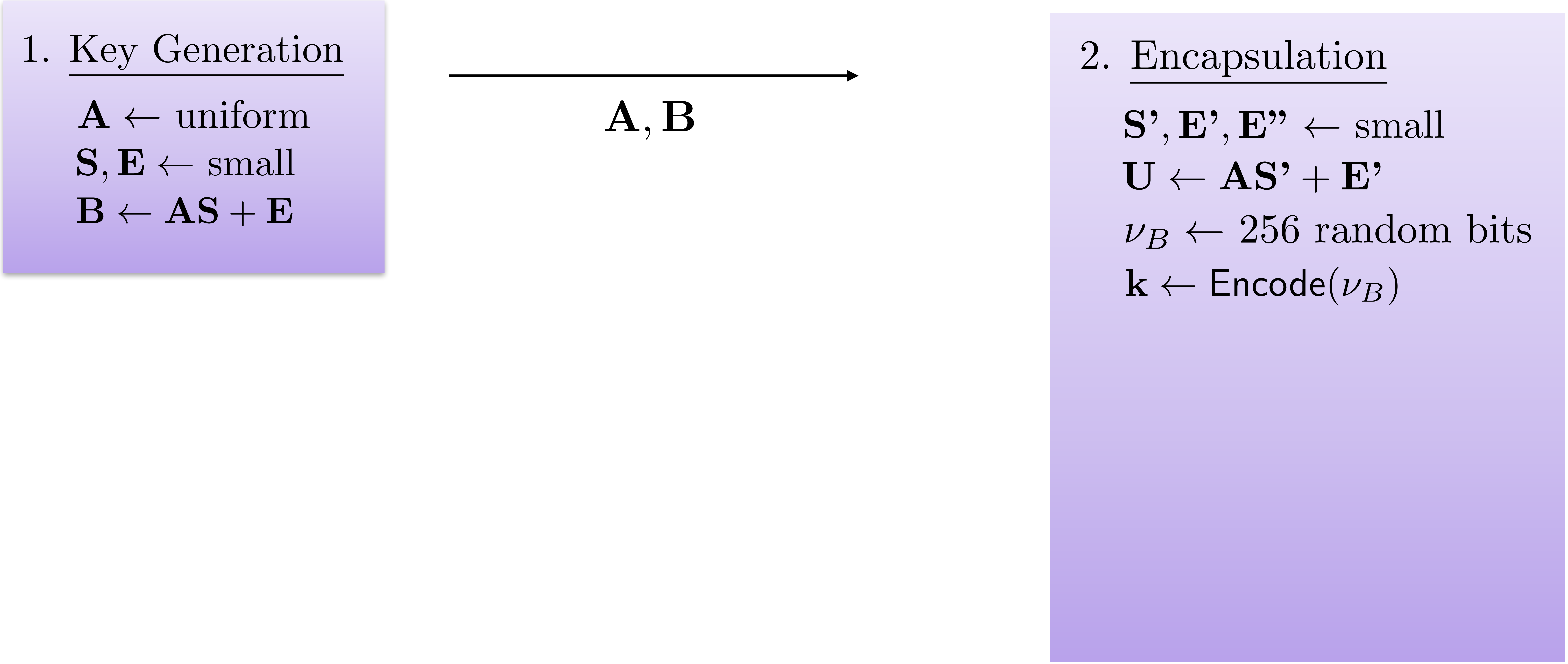
Hard to distinguish from uniform

NewHope CPA

1. Key Generation

$$\mathbf{A} \leftarrow \text{uniform}$$
$$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$$
$$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$$

NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$$\xrightarrow[\mathbf{A}, \mathbf{B}]{}$$

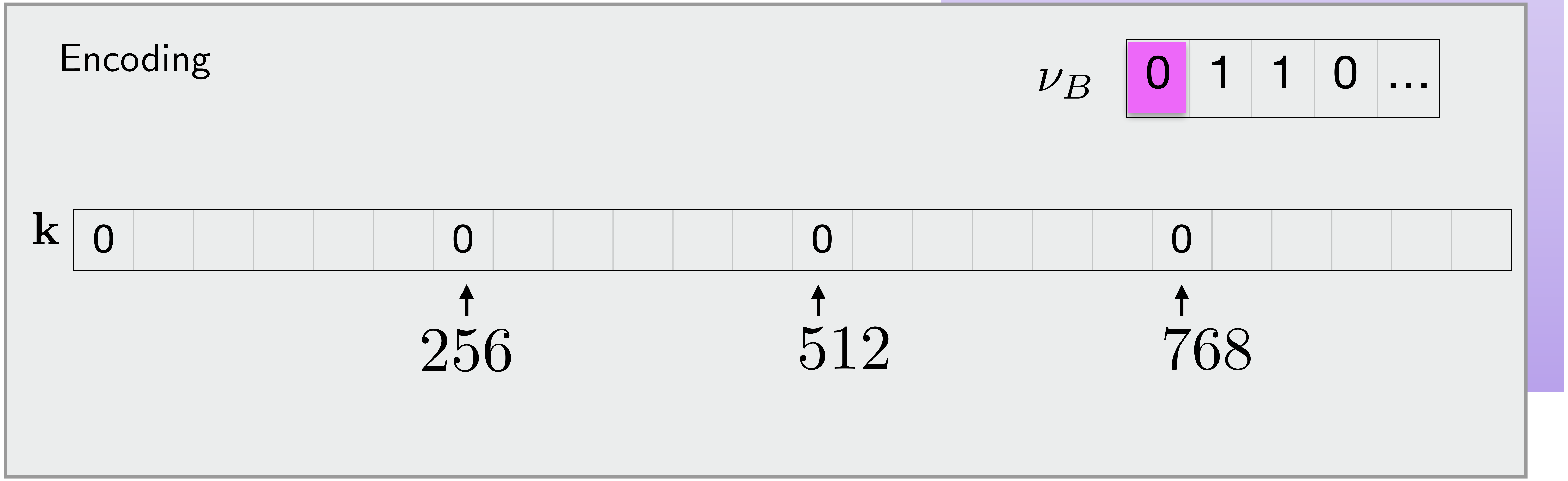
2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256$ random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$$\xrightarrow{\quad\quad\quad\quad\quad\quad}$$

$\mathbf{A}, \mathbf{B}$

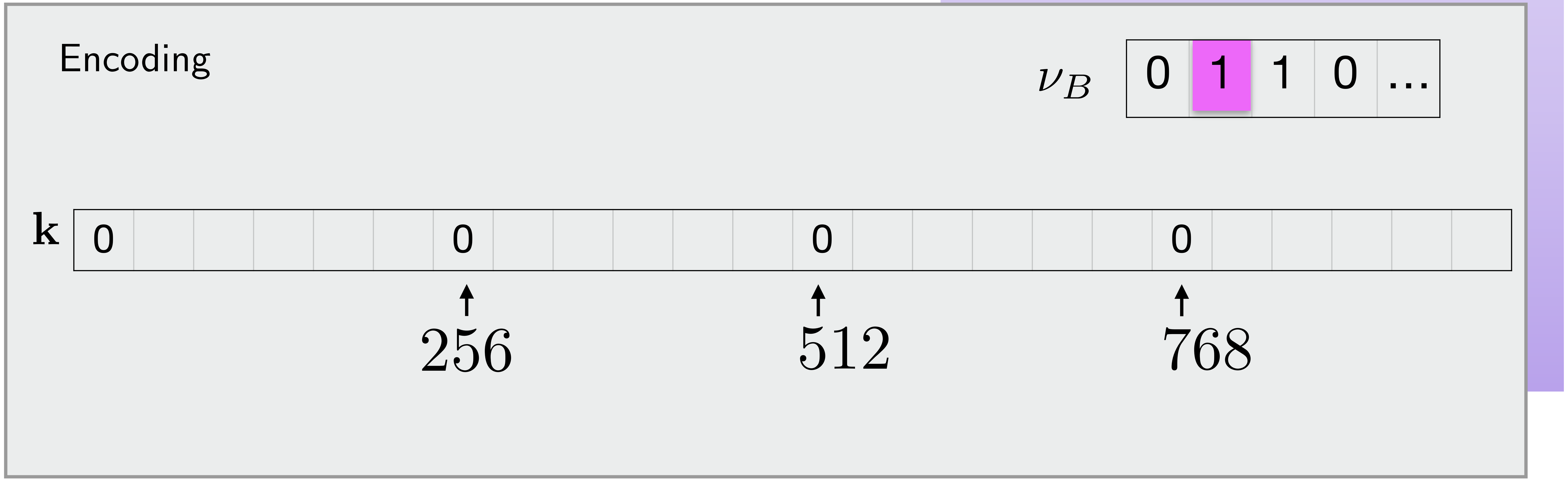
2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256$ random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$$\xrightarrow{\qquad\qquad\qquad}$$

$\mathbf{A}, \mathbf{B}$

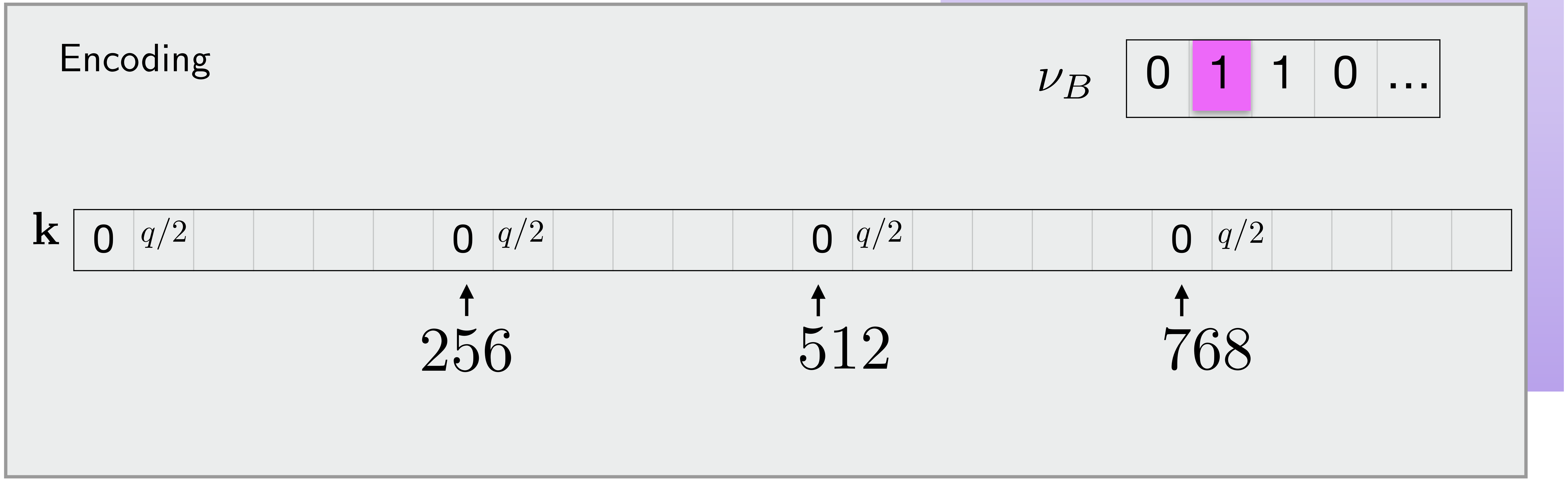
2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow$ 256 random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$



Encoding

ν_B	0	1	1	0	...

$\mathbf{k}$

0	$q/2$					0	$q/2$					0	$q/2$					0	$q/2$			

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$$\xrightarrow[\mathbf{A}, \mathbf{B}]{}$$

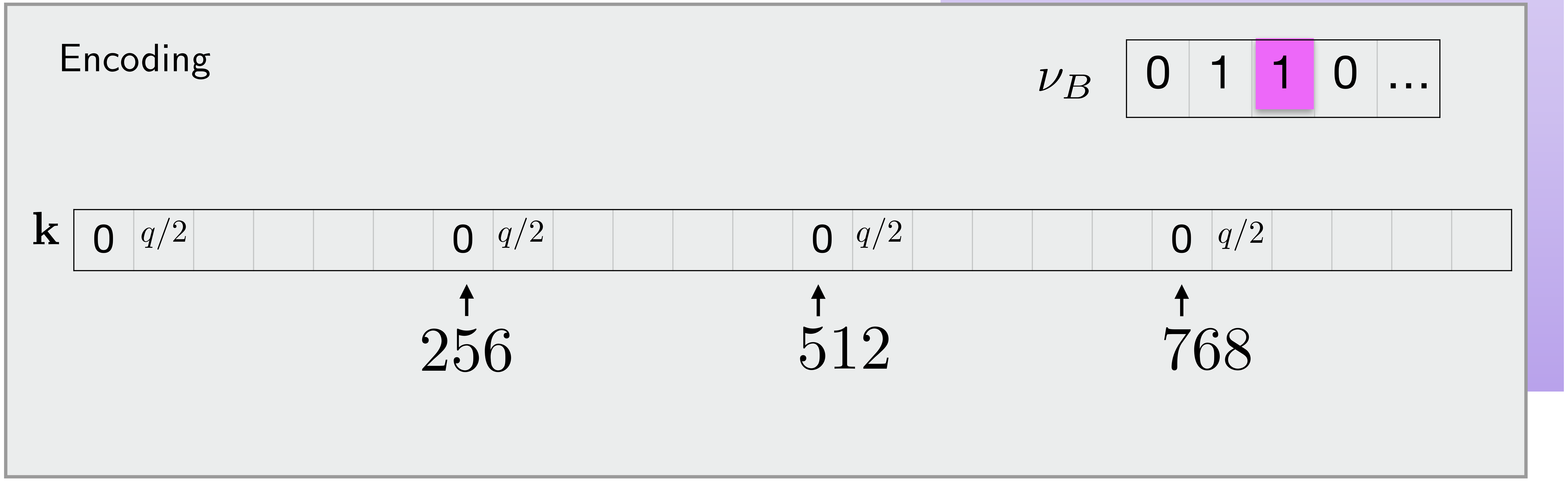
2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$



Encoding

ν_B: 0 1 1 0 ...

$\mathbf{k}$: 0 $q/2$ … 0 $q/2$ … 0 $q/2$ … 0 $q/2$ …

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{}$

$\mathbf{A}, \mathbf{B}$

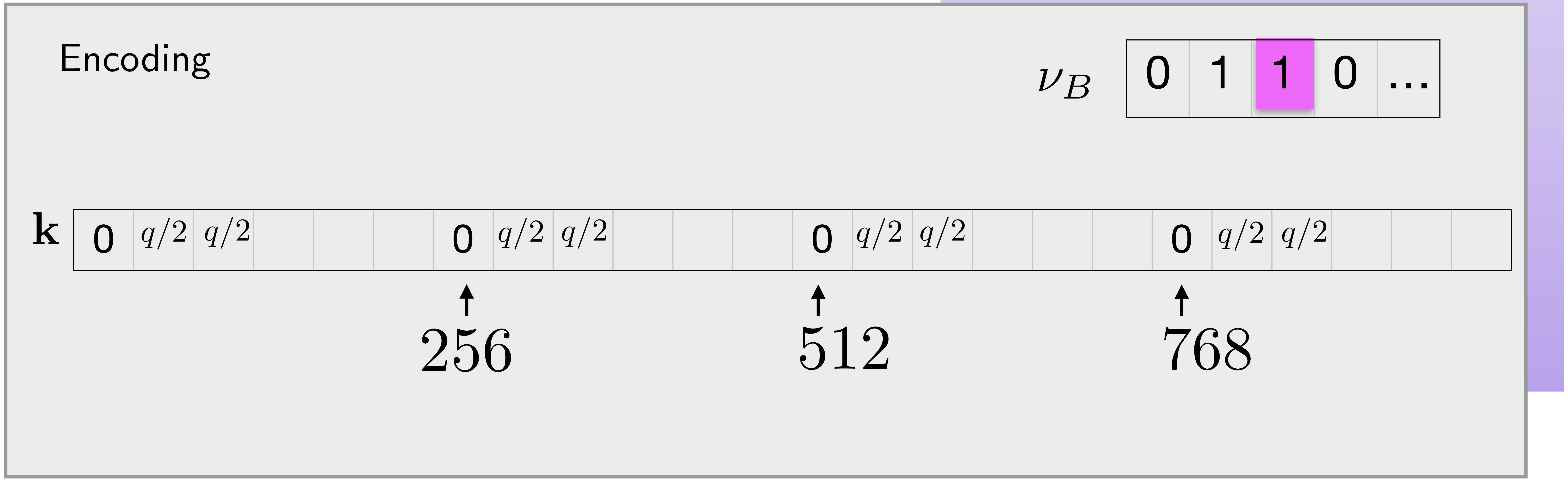
2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

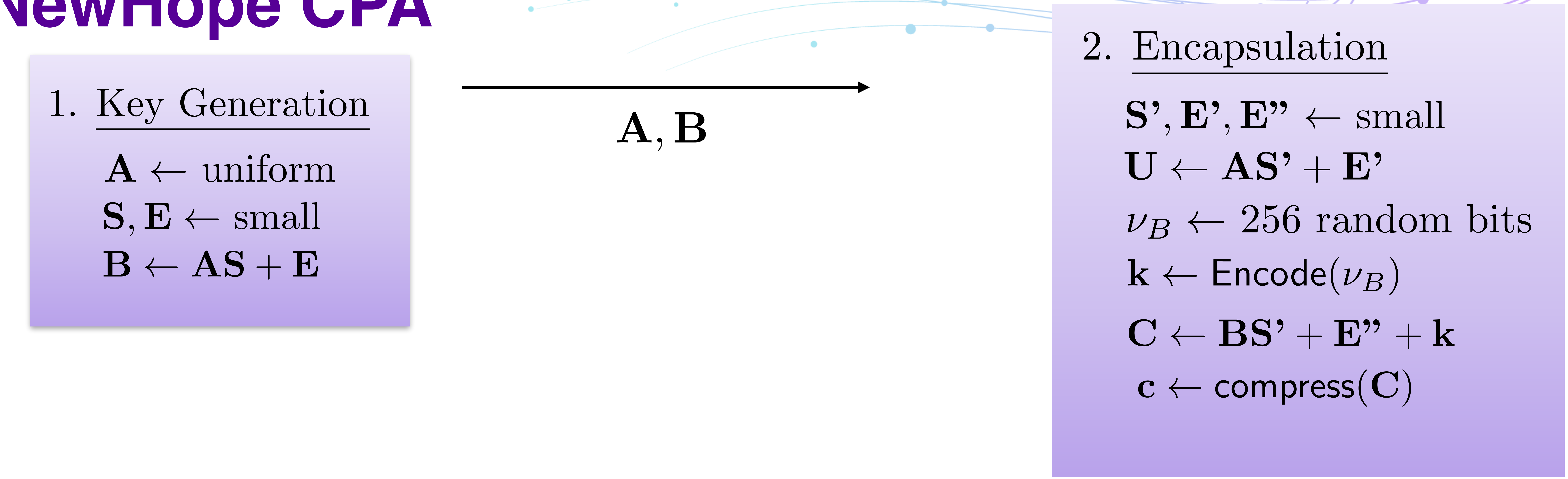


Encoding

ν_B 0 1 1 0 ...

$\mathbf{k}$ 0 $q/2$ $q/2$ 0 $q/2$ $q/2$ 0 $q/2$ $q/2$ 0 $q/2$ $q/2$

NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

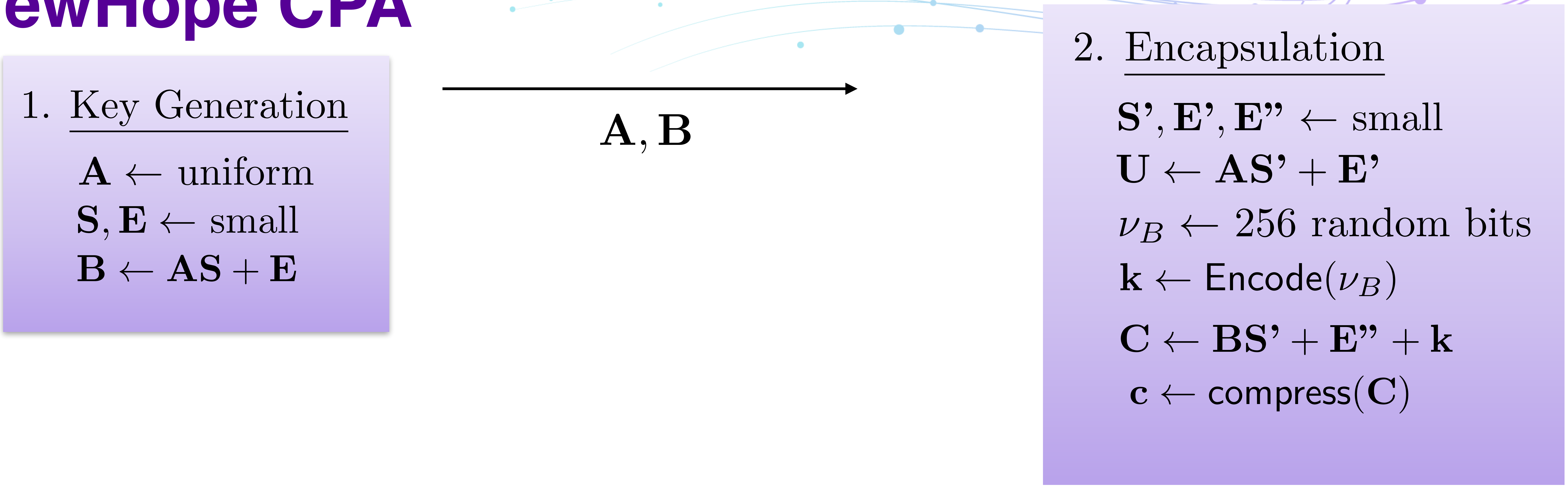
$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow[\mathbf{A}, \mathbf{B}]{}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

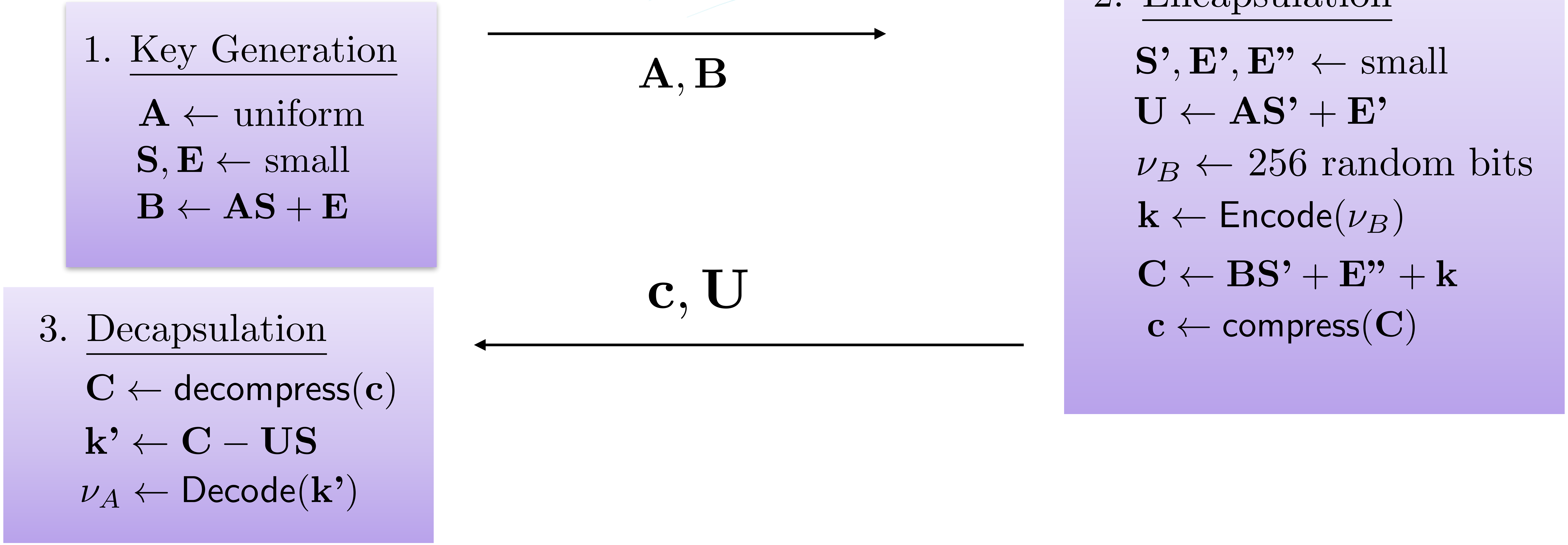
$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$$\mathbf{C\text{-}US} = \overset{\text{small}}{\boxed{\mathbf{ES'{+}E''{-}E'S}}} + \mathbf{k}.$$

NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

$$\mathbf{C\text{-}US} = \overbrace{\mathbf{ES'}+\mathbf{E''}-\mathbf{E'S}}^{\text{small}}+\mathbf{k}.$$

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\quad\quad\quad\quad}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{\quad\quad\quad\quad}$

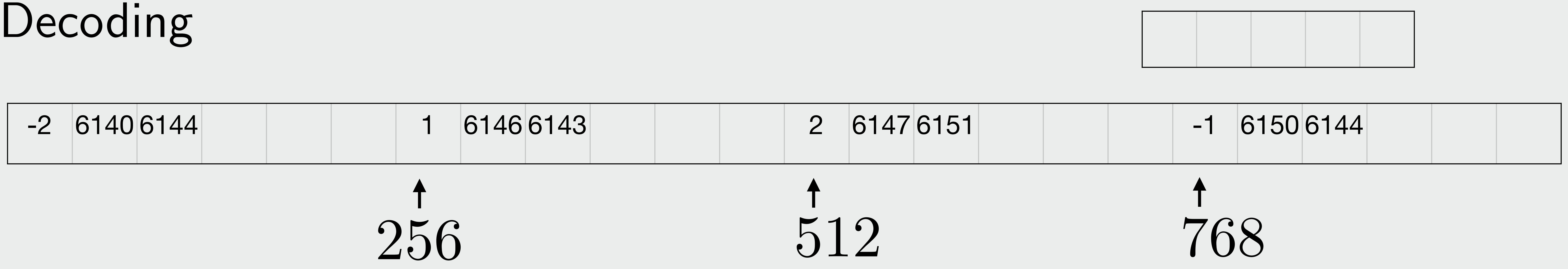
3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding



-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\quad\quad\quad\quad}$ $\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow$ 256 random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$ $\xleftarrow{\quad\quad\quad\quad}$

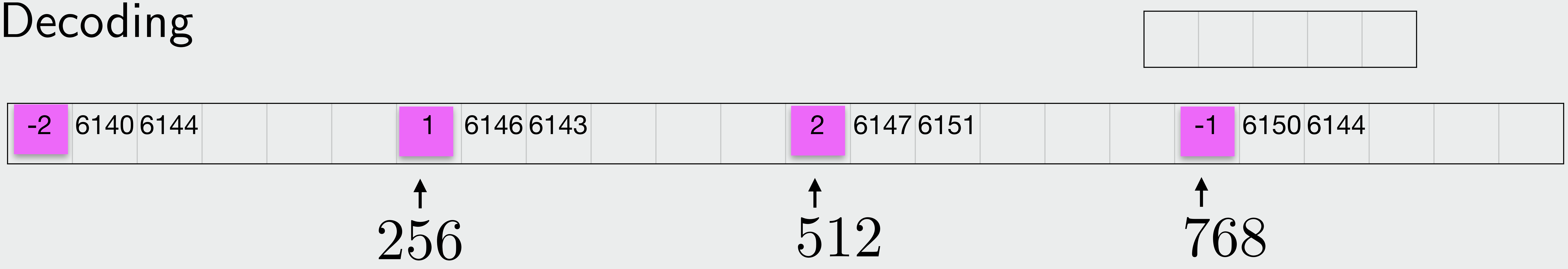
3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\quad\quad\quad}$ $\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$ $\xleftarrow{\quad\quad\quad}$

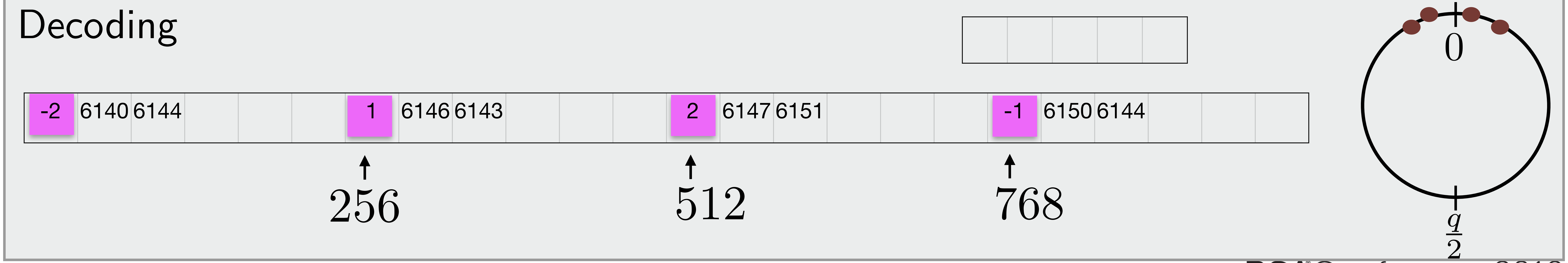
3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

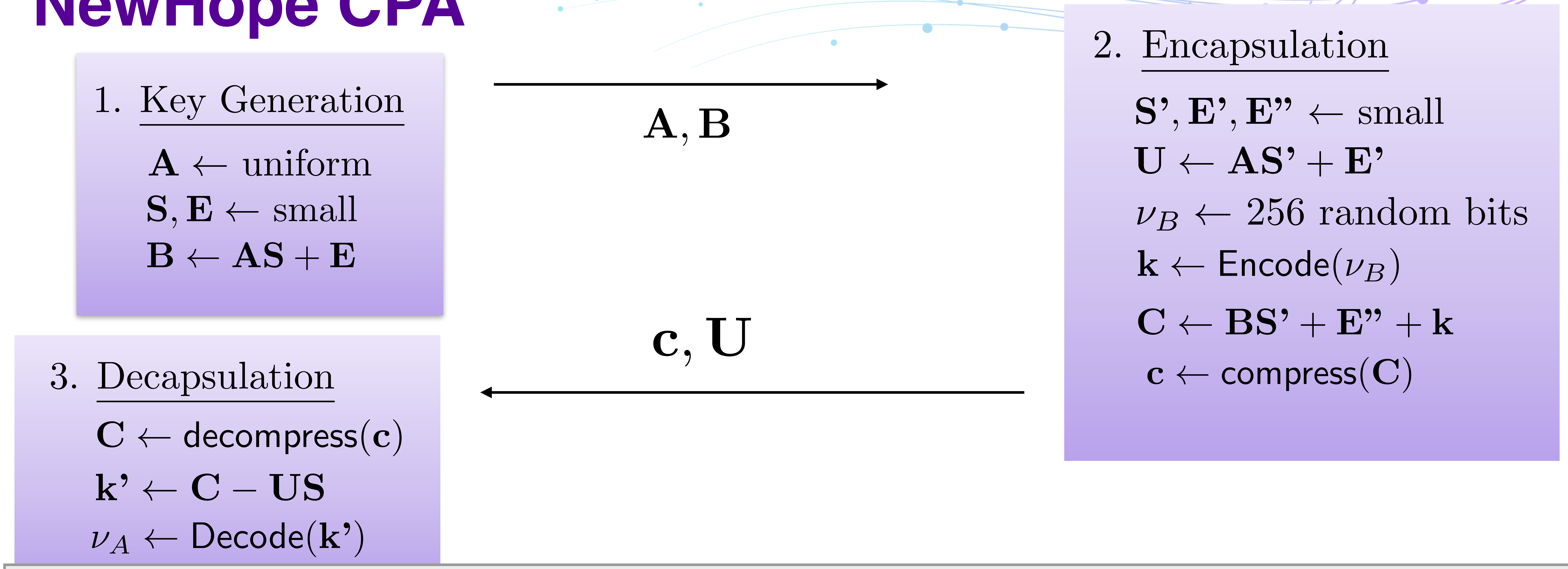
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding



NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow$ 256 random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

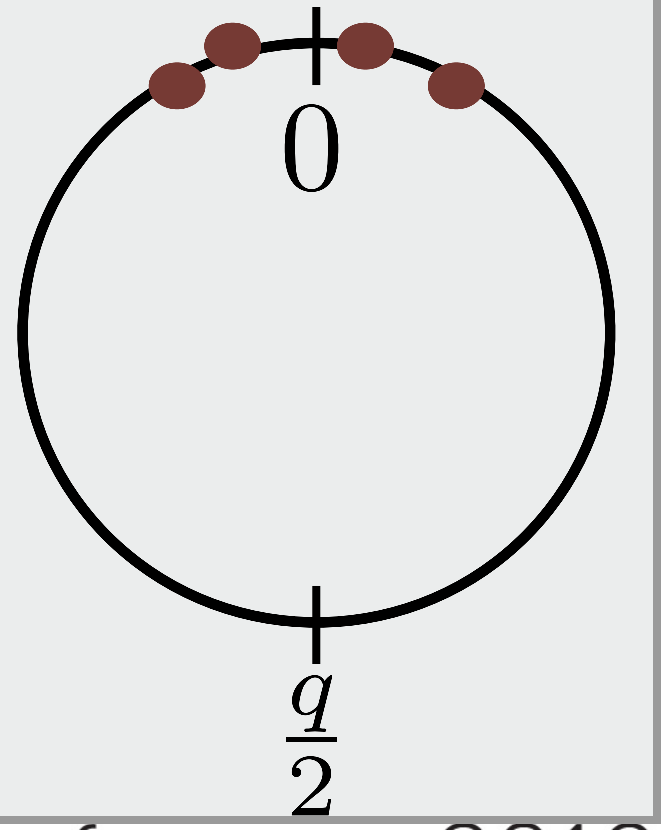
$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding

0				

-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

↑ 256 ↑ 512 ↑ 768



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\hspace{4em}}$ $\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$ $\xleftarrow{\hspace{4em}}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

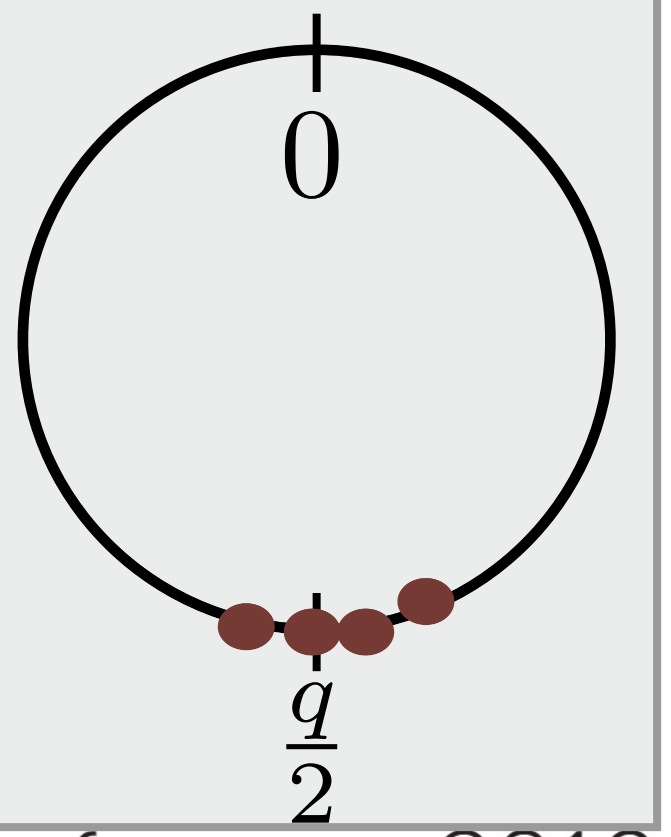
$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding

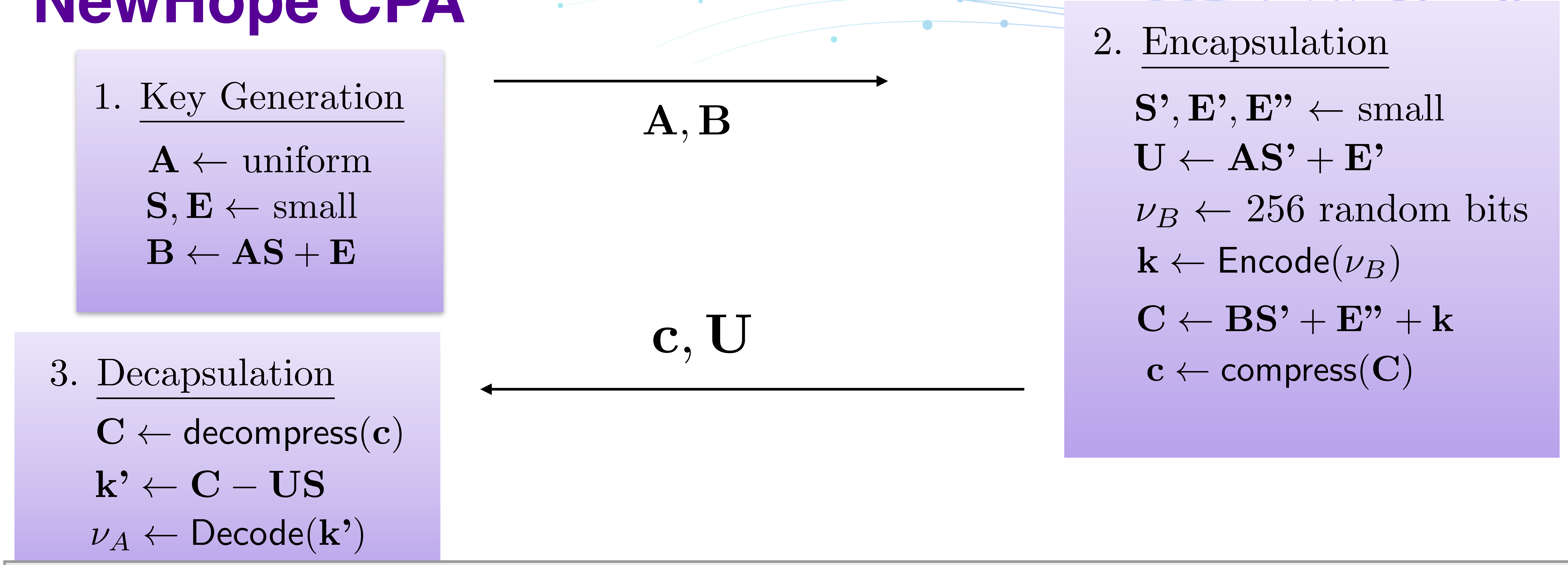
0				

-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

↑ 256 ↑ 512 ↑ 768



NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow$ small

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow$ 256 random bits

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

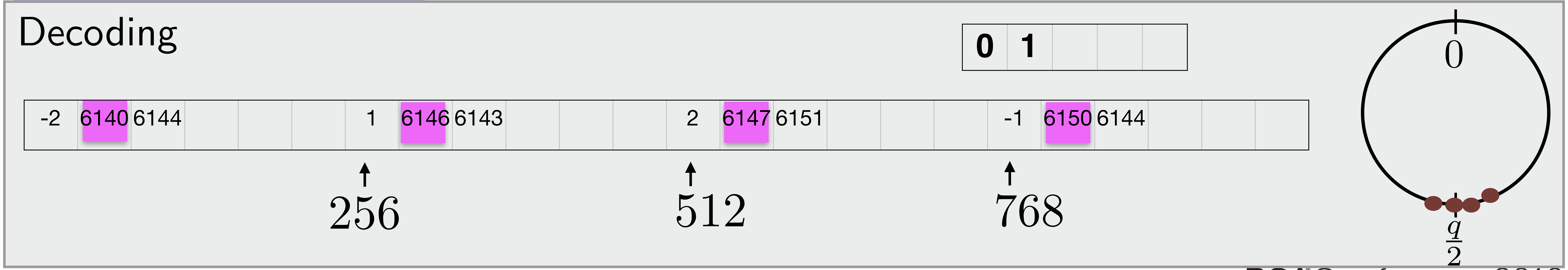
3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

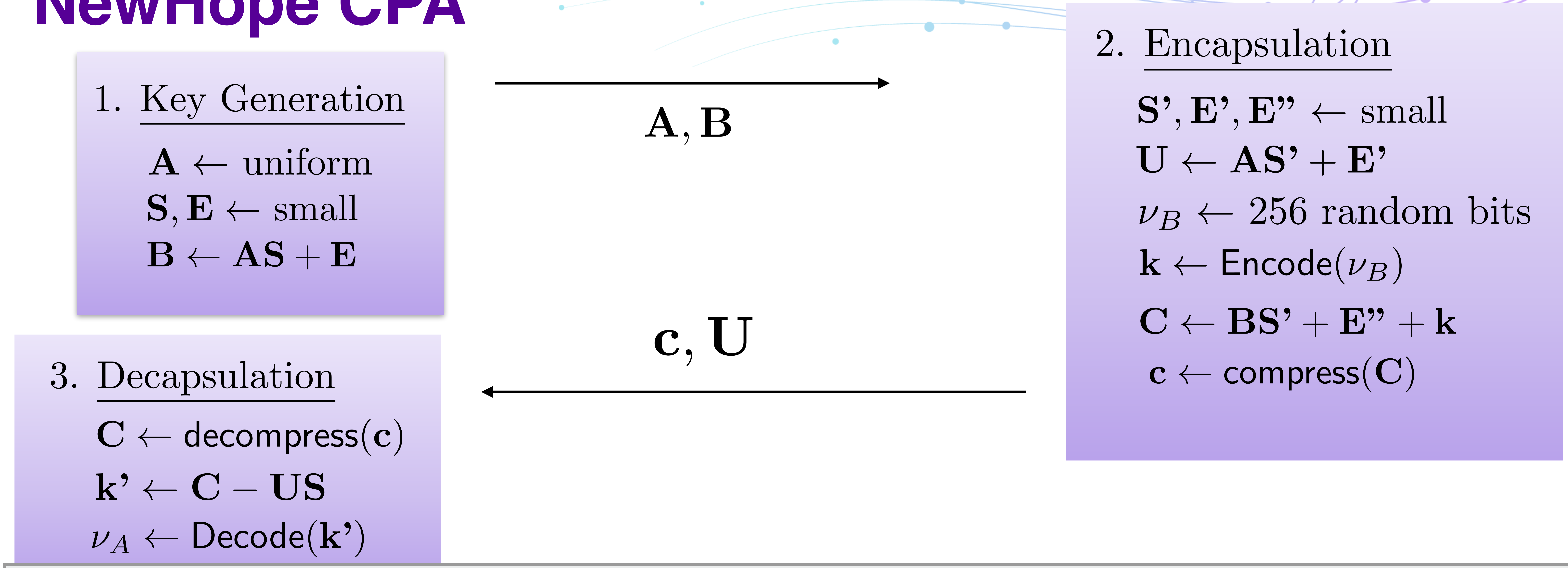
Decoding



0	**1**			

-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

NewHope CPA



1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$
$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$
$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$
$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$
$\nu_B \leftarrow 256 \text{ random bits}$
$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$
$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$
$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

3. Decapsulation

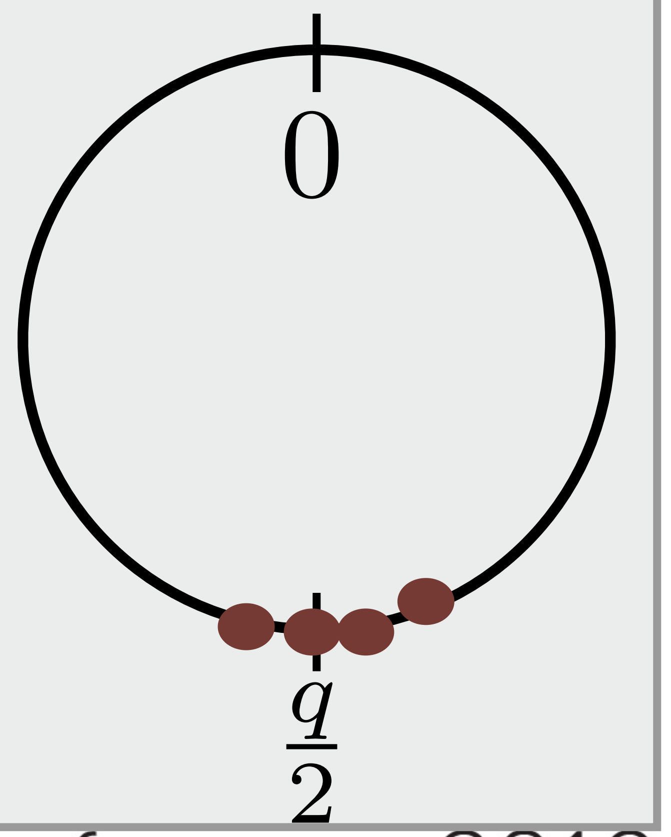
$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding

0	**1**			

-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

↑ 256 ↑ 512 ↑ 768



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\mathbf{A}, \mathbf{B}$ →

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

← $\mathbf{c}, \mathbf{U}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

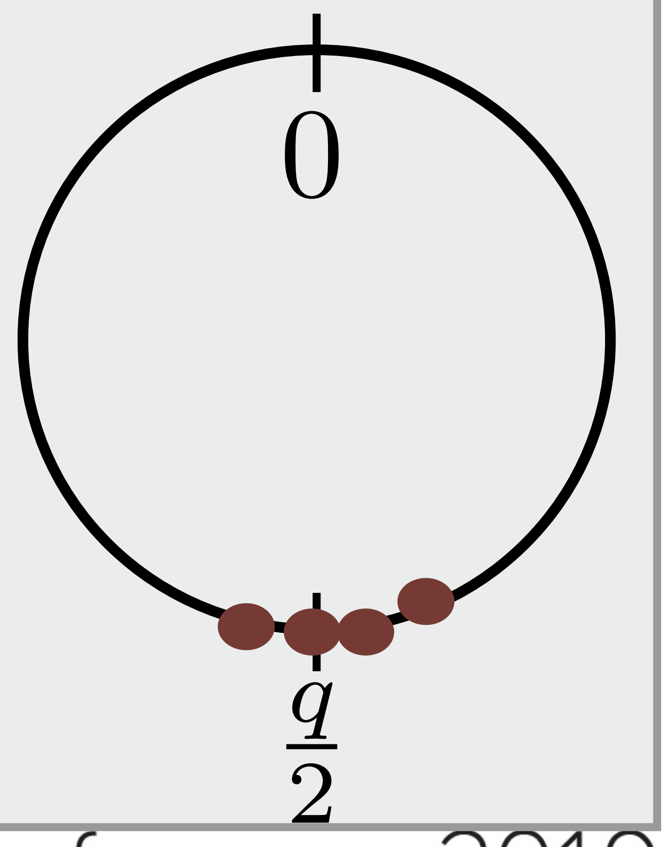
$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Decoding

0	**1**	**1**		

-2	6140	6144				1	6146	6143				2	6147	6151				-1	6150	6144			

↑ 256 ↑ 512 ↑ 768



NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\qquad\qquad}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{\qquad\qquad}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

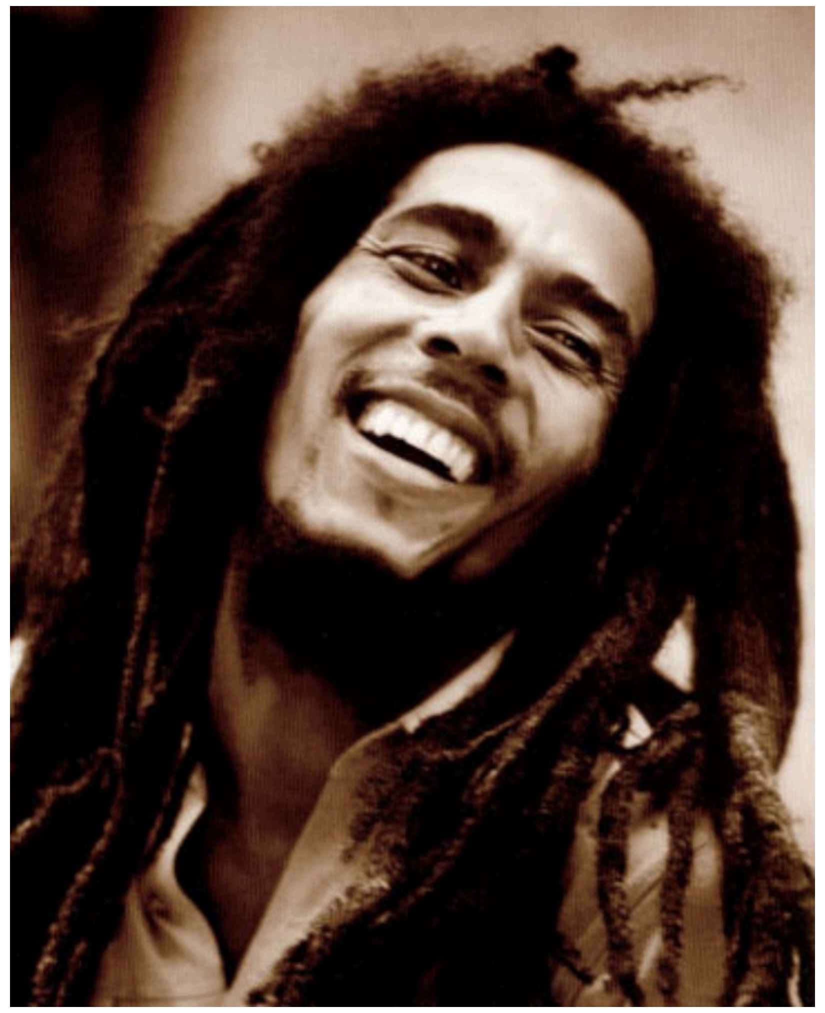
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Symmetric key encryption

$m = \mathsf{Senc}_{\nu_B}(''HelloAlice'')$

$\xleftarrow{\qquad\qquad}$



$\mathsf{Sdec}_{\nu_A}(m)$

Key Caching and attack model

NewHope CPA

1. <u>Key Generation</u>

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{}$

$\mathbf{A}, \mathbf{B}$

2. <u>Encapsulation</u>

$\mathbf{S'}, \mathbf{E'}, \mathbf{E''} \leftarrow \text{small}$

$\mathbf{U} \leftarrow \mathbf{AS'} + \mathbf{E'}$

$\nu_B \leftarrow 256 \text{ random bits}$

$\mathbf{k} \leftarrow \mathsf{Encode}(\nu_B)$

$\mathbf{C} \leftarrow \mathbf{BS'} + \mathbf{E''} + \mathbf{k}$

$\mathbf{c} \leftarrow \mathsf{compress}(\mathbf{C})$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{}$

3. <u>Decapsulation</u>

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

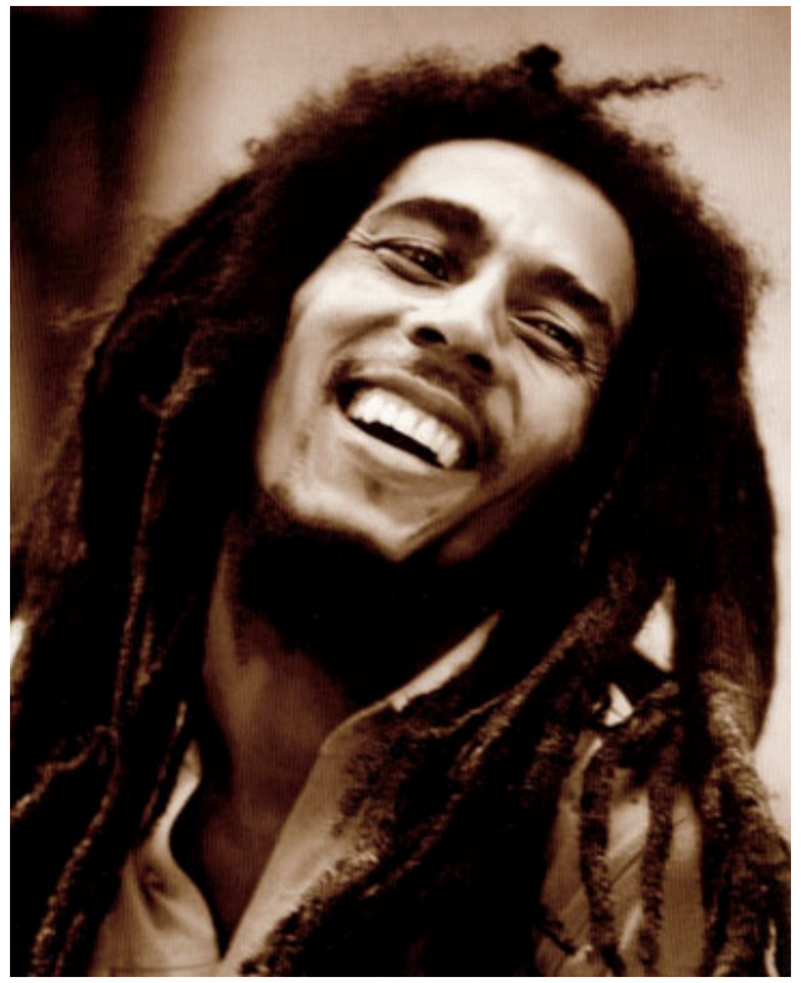
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Symmetric key encryption

$m = \mathsf{Senc}_{\nu_B}(\text{"}HelloAlice\text{"})$

$\xleftarrow{}$



$\mathsf{Sdec}_{\nu_A}(m)$

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\qquad\qquad}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\nu_B, \mathbf{c}, \mathbf{U} \leftarrow$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{\qquad\qquad}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

Symmetric key encryption

$m = \mathsf{Senc}_{\nu_B}(''HelloAlice'')$

$\xleftarrow{\qquad\qquad}$

$\mathsf{Sdec}_{\nu_A}(m)$

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow$ uniform

$\mathbf{S}, \mathbf{E} \leftarrow$ small

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{\hspace{4em}}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\nu_B, \mathbf{c}, \mathbf{U} \leftarrow$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{\hspace{4em}}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

$m = \mathsf{Senc}_{\nu_B}(\text{"}\mathit{HelloAlice}\text{"})$

$\xleftarrow{\hspace{4em}}$

$\mathsf{Sdec}_{\nu_A}(m)$

NewHope CPA

1. Key Generation

$\mathbf{A} \leftarrow \text{uniform}$

$\mathbf{S}, \mathbf{E} \leftarrow \text{small}$

$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$

$\xrightarrow{}$

$\mathbf{A}, \mathbf{B}$

2. Encapsulation

$\nu_B, \mathbf{c}, \mathbf{U} \leftarrow$

$\mathbf{c}, \mathbf{U}$

$\xleftarrow{}$

3. Decapsulation

$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$

$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$

$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$

$m = \mathsf{Senc}_{\nu_B}(\text{"}HelloAlice\text{"})$

$\xleftarrow{}$

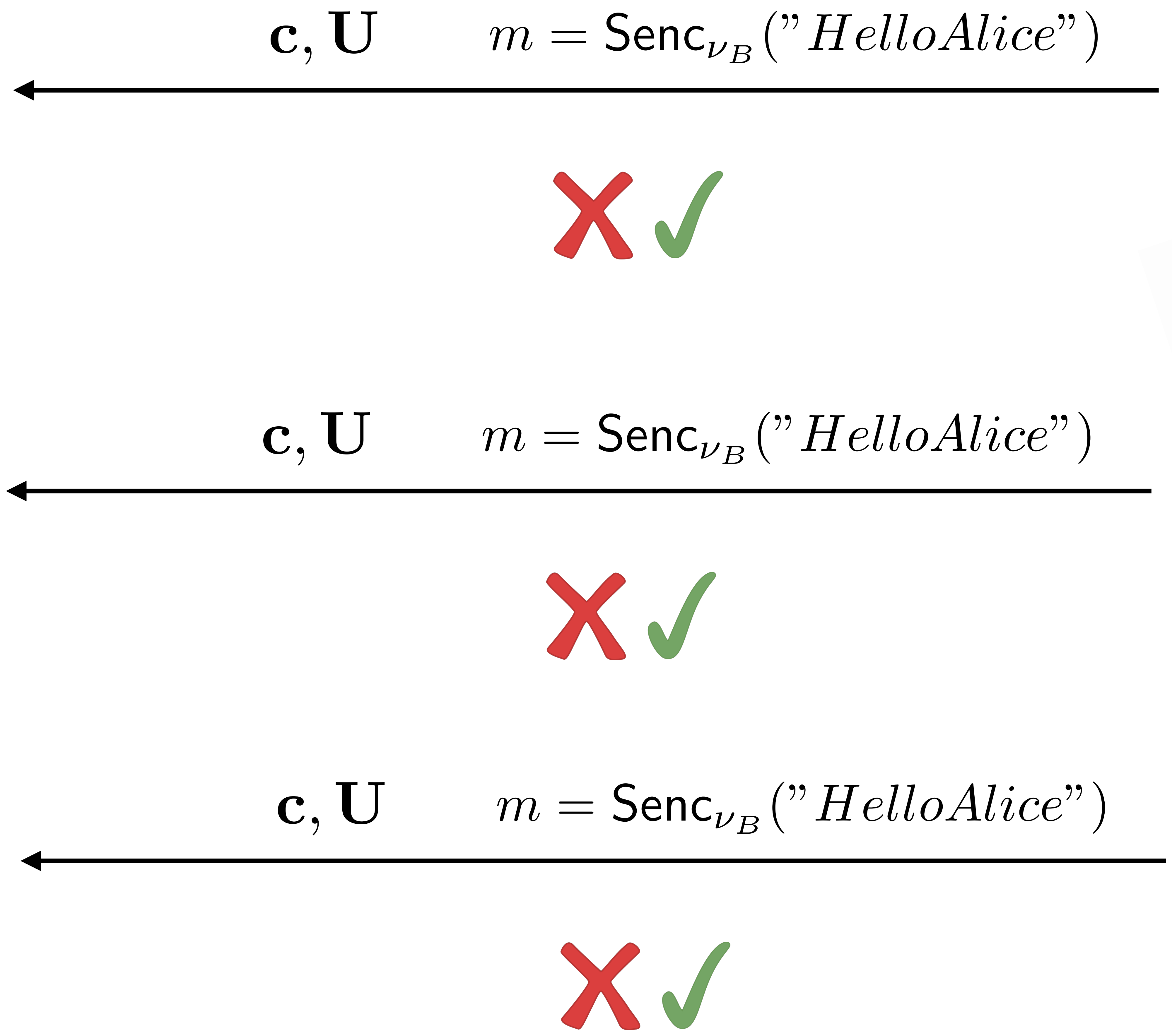
$\mathsf{Sdec}_{\nu_A}(m)$

✗ ✓

Possible key mismatch

Key caching and attack model

Key caching : the secret key is **fixed** for several queries



$\mathbf{c}, \mathbf{U}$ $m = \mathsf{Senc}_{\nu_B}(\text{"}HelloAlice\text{"})$

✗ ✓

$\mathbf{c}, \mathbf{U}$ $m = \mathsf{Senc}_{\nu_B}(\text{"}HelloAlice\text{"})$

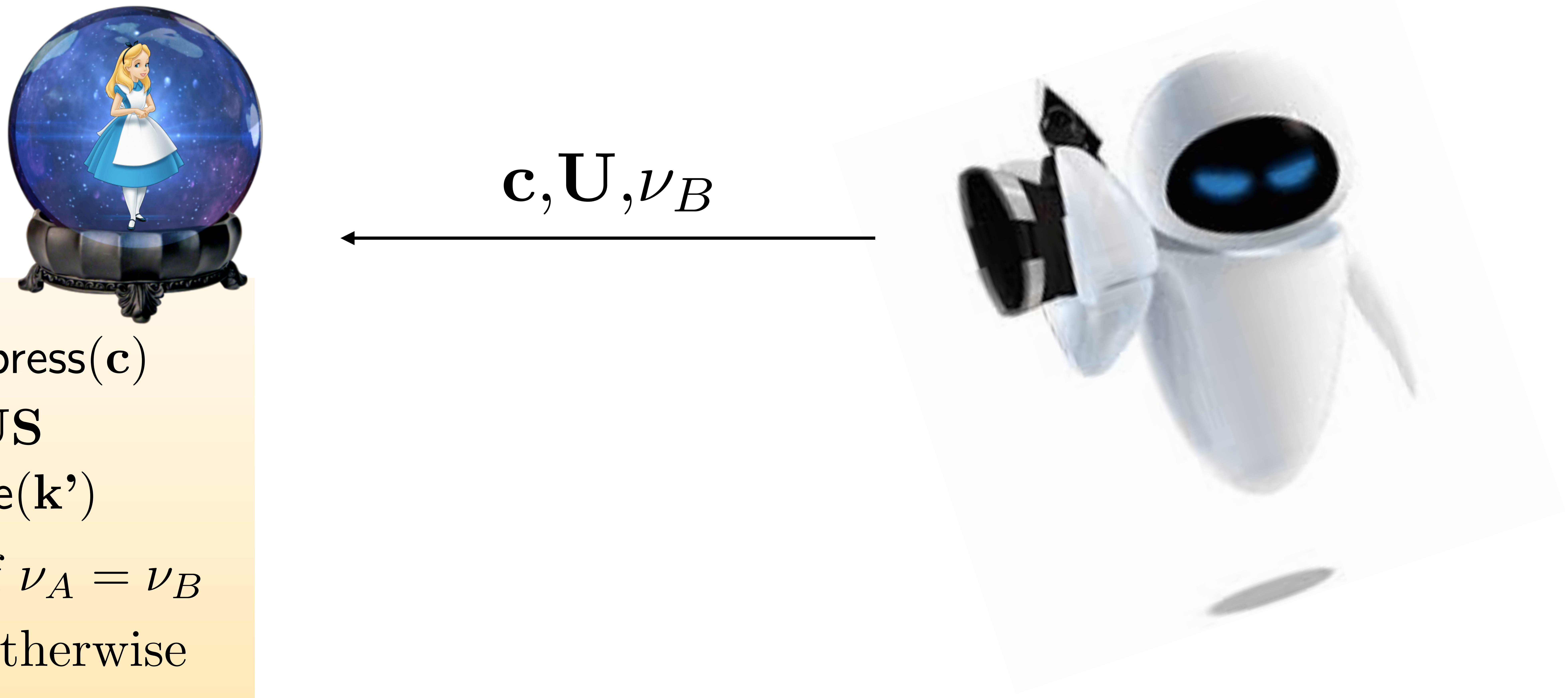
✗ ✓

$\mathbf{c}, \mathbf{U}$ $m = \mathsf{Senc}_{\nu_B}(\text{"}HelloAlice\text{"})$

✗ ✓

Key mismatch oracle

Key caching hypothesis → The attack has access to the following oracle



$\mathbf{c},\mathbf{U},\nu_B$

```
$\mathbf{C} \leftarrow \text{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \text{Decode}(\mathbf{k'})$
$\textbf{Return } 1 \text{ if } \nu_A = \nu_B$
$\textbf{Return } 0 \text{ otherwise}$
```

Key mismatch oracle

2016 : Concrete attack introduced by Fluhrer on the LWE scheme

2017: Improvements by Ding et al.

2017 : Similar approach on HILA5 by Bernstein et al.

$$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$$

$$\nu_A \leftarrow \mathsf{HelpRec}(\mathbf{k'})$$

Key mismatch oracle

```
$\mathbf{C} \leftarrow \text{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \text{Decode}(\mathbf{k'})$
$\textbf{Return } 1 \text{ if } \nu_A = \nu_B$
$\textbf{Return } 0 \text{ otherwise}$
```

2016 : Concrete attack introduced by Fluhrer on the LWE scheme

2017: Improvements by Ding et al.

2017 : Similar approach on HILA5 by Bernstein et al.

New difficulties for NewHope

Key mismatch oracle

2016 : Concrete attack introduced by Fluhrer on the LWE scheme

2017: Improvements by Ding et al.

2017 : Similar approach on HILA5 by Bernstein et al.

```
$\mathbf{C} \leftarrow \text{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \text{Decode}(\mathbf{k'})$
$\mathbf{Return}\ 1 \text{ if } \nu_A = \nu_B$
$\mathbf{Return}\ 0 \text{ otherwise}$
```

New difficulties for NewHope

The decompress function keeps only the 3 most significant bits

Key mismatch oracle

2016 : Concrete attack introduced by Fluhrer on the LWE scheme

2017: Improvements by Ding et al.

2017 : Similar approach on HILA5 by Bernstein et al.

```
$\mathbf{C} \leftarrow \text{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \text{Decode}(\mathbf{k'})$
Return 1 if $\nu_A = \nu_B$
Return 0 otherwise
```

New difficulties for NewHope

The decompress function keeps only the 3 most significant bits

The decode function takes the coefficients four by four

How realistic is the key caching hypothesis ?

Key caching is a misuse case :
Designers strongly recommends to **draw fresh keys** at each exchange

How realistic is the key caching hypothesis ?

Key caching is a misuse case :
Designers strongly recommends to **draw fresh keys** at each exchange

Is it possible for a practical implementation ?

How realistic is the key caching hypothesis ?

Key caching is a misuse case :
Designers strongly recommends to **draw fresh keys** at each exchange

Is it possible for a practical implementation ?

It depends on the **constraints**

Expensive randomness

Many queries per second

Our purpose

This vulnerability is intrinsic to the scheme

Assessing the power of the attacker is necessary

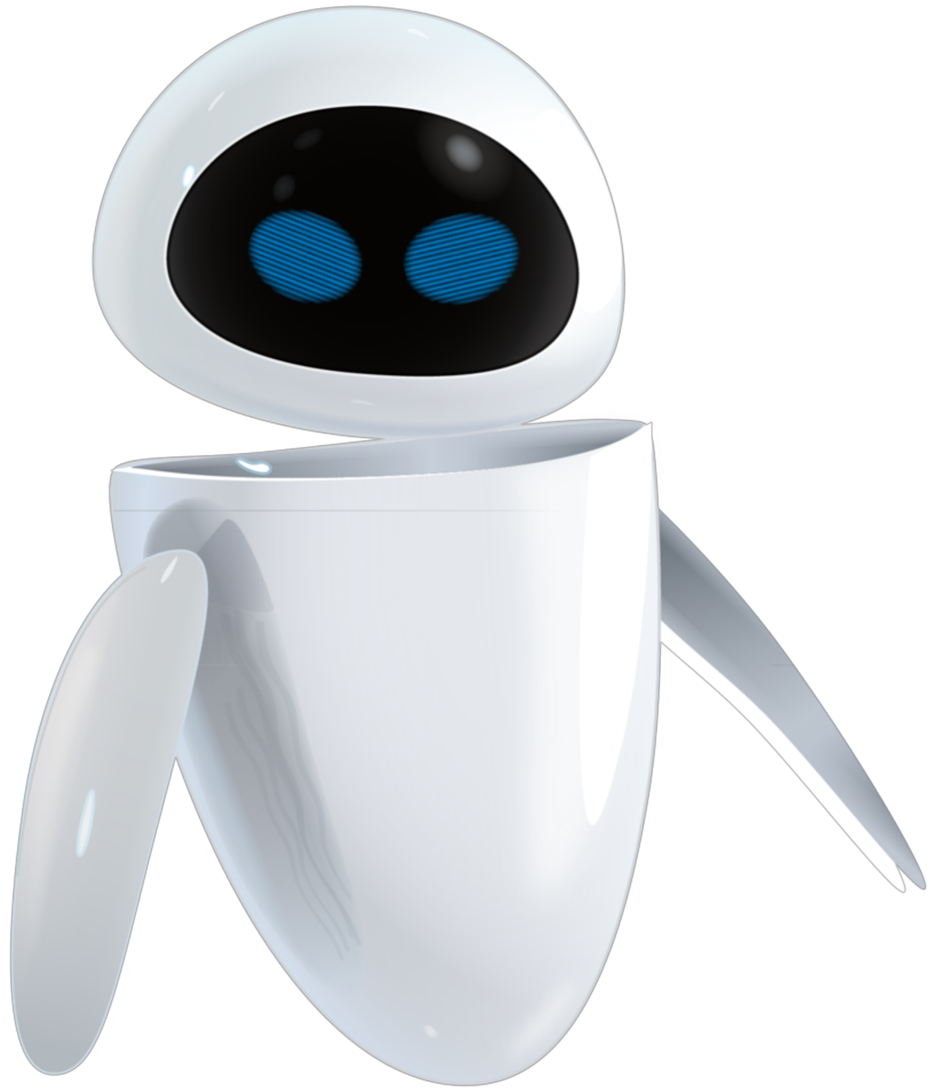
Our purpose

This vulnerability is intrinsic to the scheme

Assessing the power of the attacker is necessary

How many queries are necessary to recover the secret key S ?

```
$\mathbf{C} \leftarrow \text{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \text{Decode}(\mathbf{k'})$
$\textbf{Return } 1 \text{ if } \nu_A = \nu_B$
$\textbf{Return } 0 \text{ otherwise}$
```



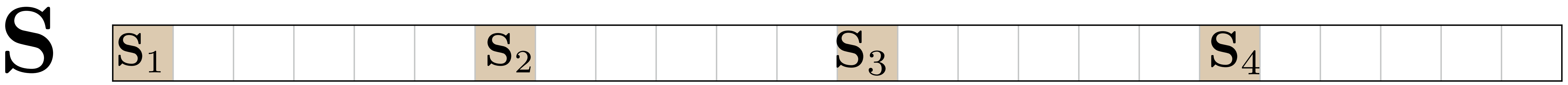
An attack on the CPA version

The idea

```
$\mathbf{C} \leftarrow \mathrm{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \mathrm{Decode}(\mathbf{k'})$
$\mathbf{Return}\ 1\ \mathrm{if}\ \nu_A = \nu_B$
$\mathbf{Return}\ 0\ \mathrm{otherwise}$
```

$\longleftarrow$ $\mathbf{c}, \mathbf{U}, \nu_B$

Find **specific queries** that induce an information in the output



The idea

```
$\mathbf{C} \leftarrow \mathsf{decompress}(\mathbf{c})$
$\mathbf{k'} \leftarrow \mathbf{C} - \mathbf{US}$
$\nu_A \leftarrow \mathsf{Decode}(\mathbf{k'})$
$\textbf{Return } 1 \text{ if } \nu_A = \nu_B$
$\textbf{Return } 0 \text{ otherwise}$
```

$\longleftarrow \quad \mathbf{c},\mathbf{U},\nu_B$

Find **specific queries** that induce an information in the output

$\mathbf{S}$

$\mathbf{s}_1$						$\mathbf{s}_2$						$\mathbf{s}_3$						$\mathbf{s}_4$					

$\mathbf{C}$

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$\mathbf{U} \gg$

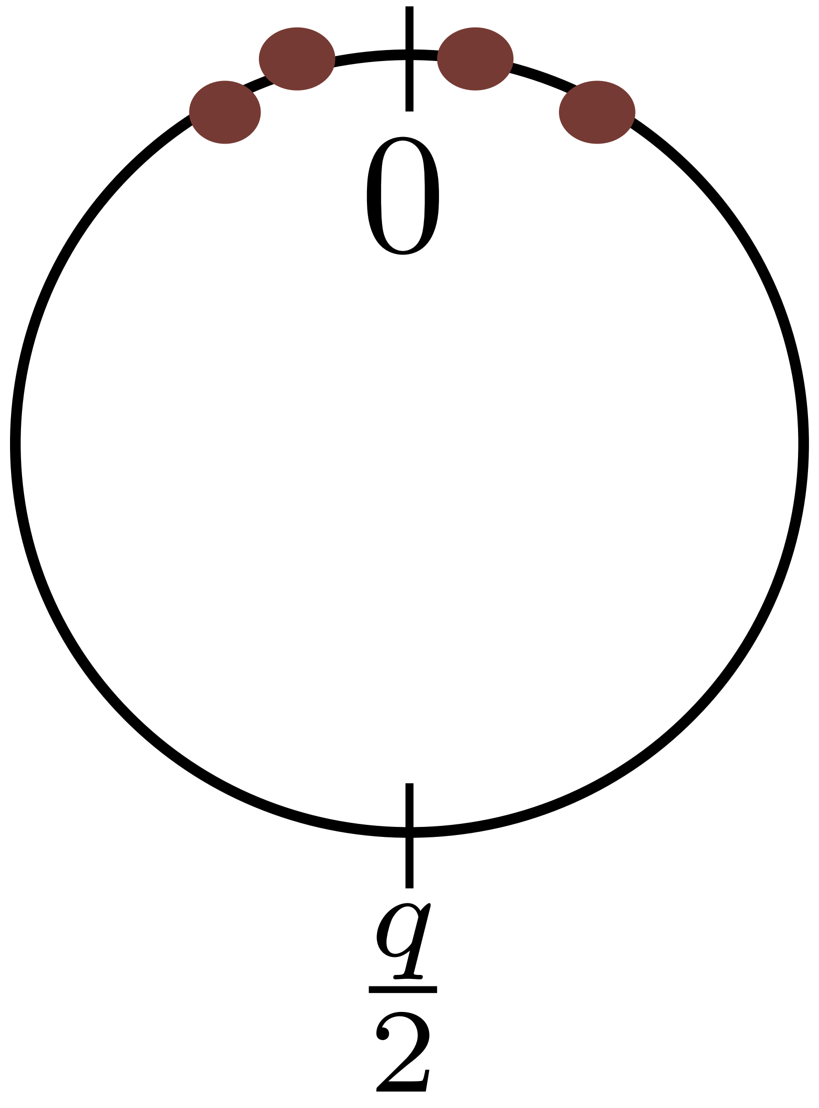
ν_B

0	0	0	0	...

The idea

$\mathbf{S}$	$\mathbf{S}_1$						$\mathbf{S}_2$						$\mathbf{S}_3$						$\mathbf{S}_4$					
$\mathbf{C}$	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$\mathbf{U} \gg$



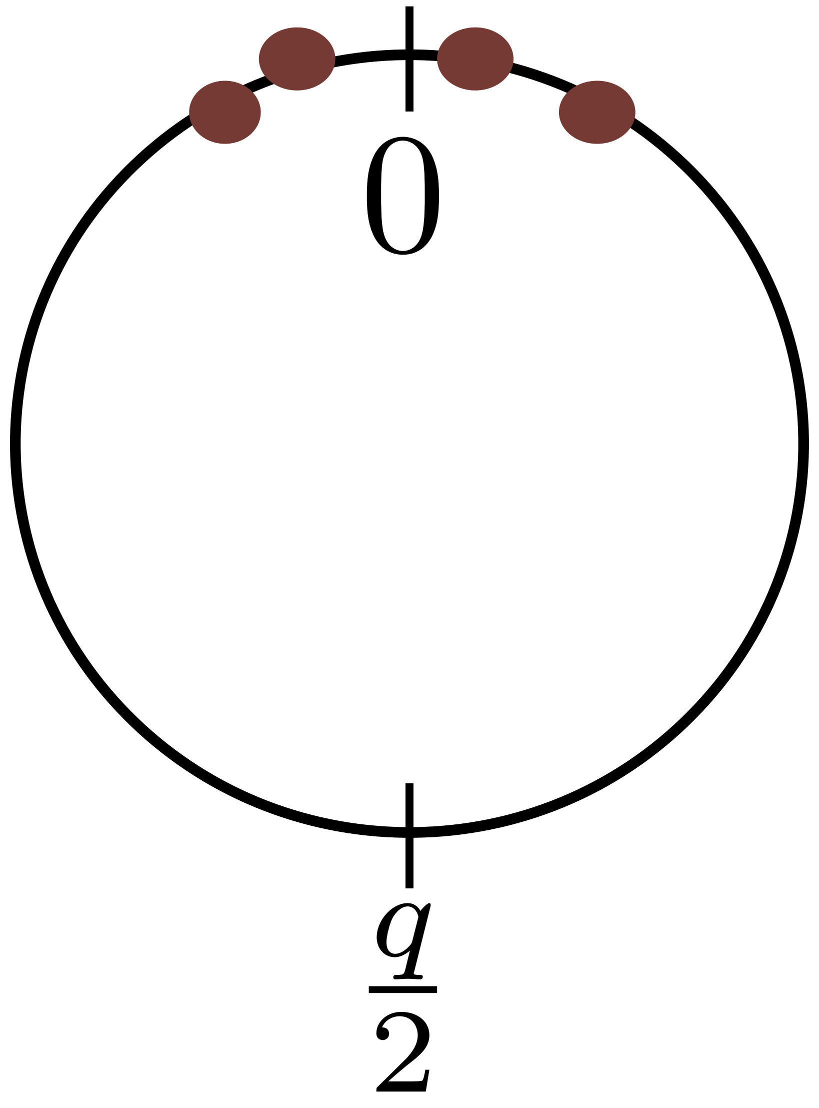
Decode$(\mathbf{C} - \mathbf{US})$

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right)$$

The idea

$\mathbf{S}$	$\mathbf{S}_1$						$\mathbf{S}_2$						$\mathbf{S}_3$						$\mathbf{S}_4$					
$\mathbf{C}$	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$\mathbf{U} \gg$



Decode$(\mathbf{C} - \mathbf{US})$

$$Sign\left(\sum_{j=1}^{j=4}\left|\,\ell_j - \mathbf{S}_j\,\right| - 8\right)$$

Can be chosen

Unknown

Sketch of the attack

$\mathbf{S}$	$\mathbf{S}_1$						$\mathbf{S}_2$						$\mathbf{S}_3$						$\mathbf{S}_4$					
$\mathbf{C}$	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$\mathbf{U} \gg$

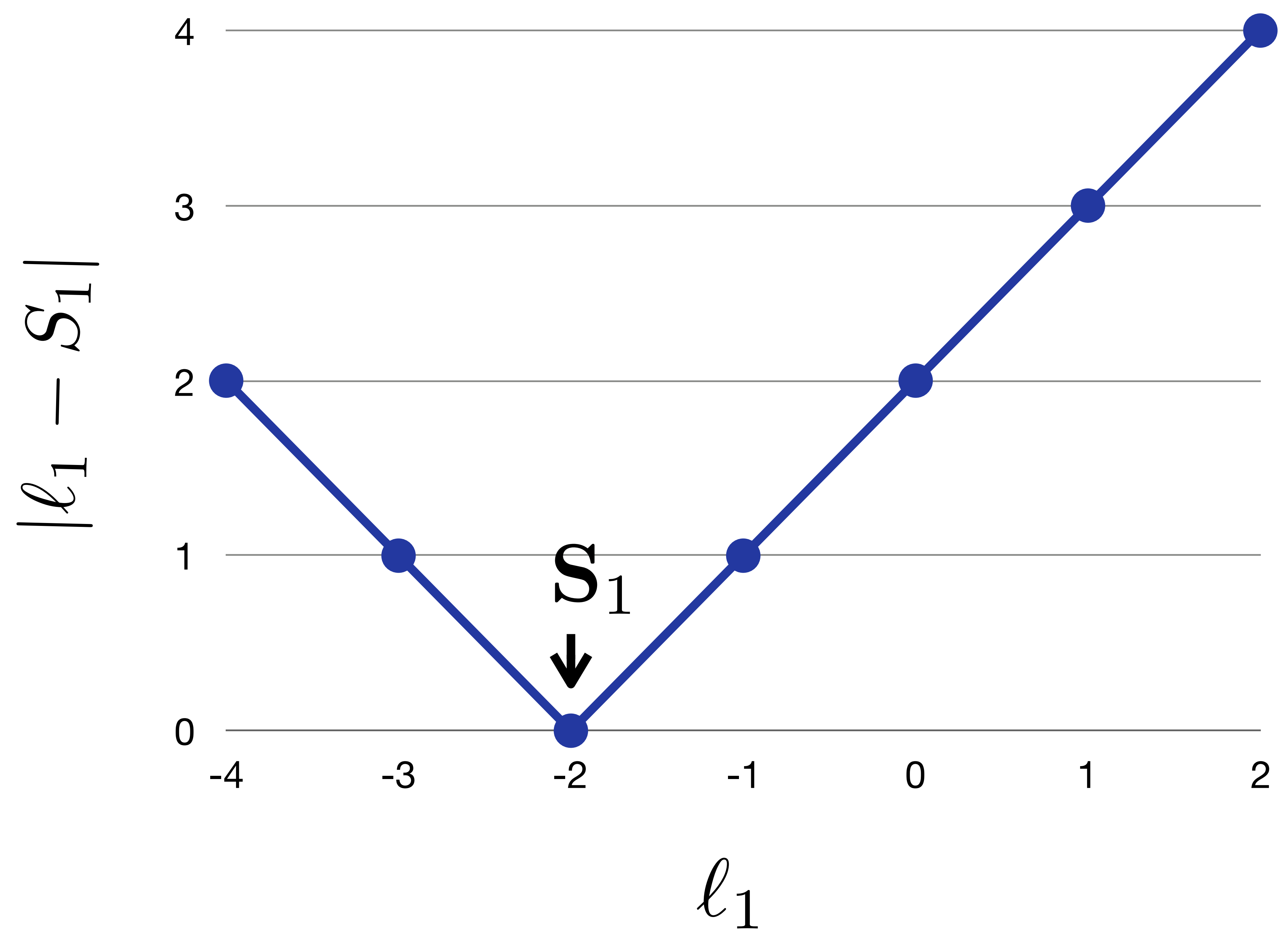
$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(|\ell_1 - \mathbf{S}_1| + \sum_{j=2}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right)$$

Sketch of the attack

$\mathbf{S}$	$\mathbf{s}_1$						$\mathbf{s}_2$						$\mathbf{s}_3$						$\mathbf{s}_4$					
$\mathbf{C}$	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$\mathbf{U} \gg$

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{s}_j\right| - 8\right) = Sign\left(|\ell_1 - \mathbf{s}_1| + \sum_{j=2}^{j=4}\left|\ell_j - \mathbf{s}_j\right| - 8\right)$$

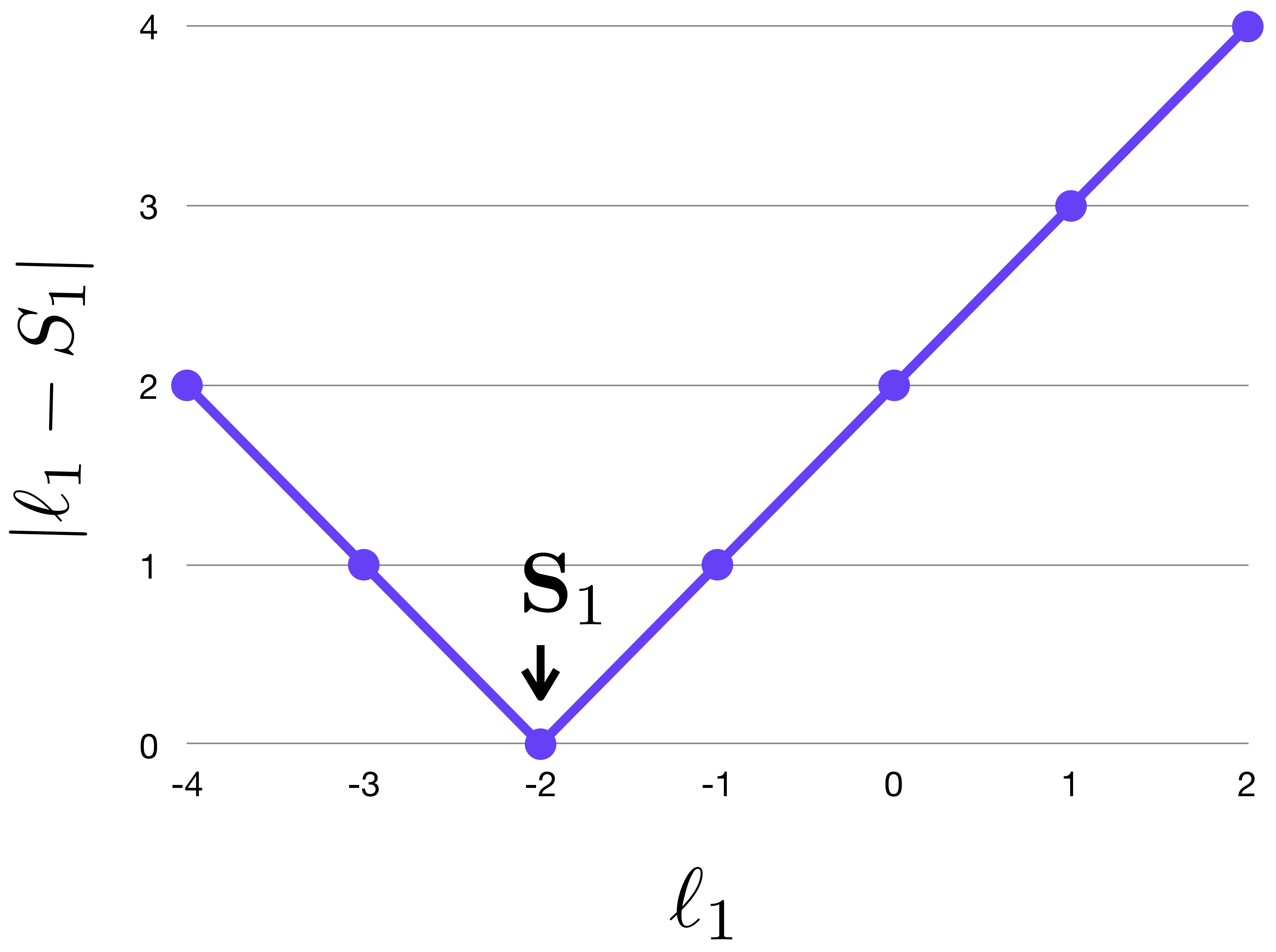


Sketch of the attack

$\mathbf{C}$

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(|\ell_1-\mathbf{S}_1|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow \mathbf{S}_1$ is found (symmetry)

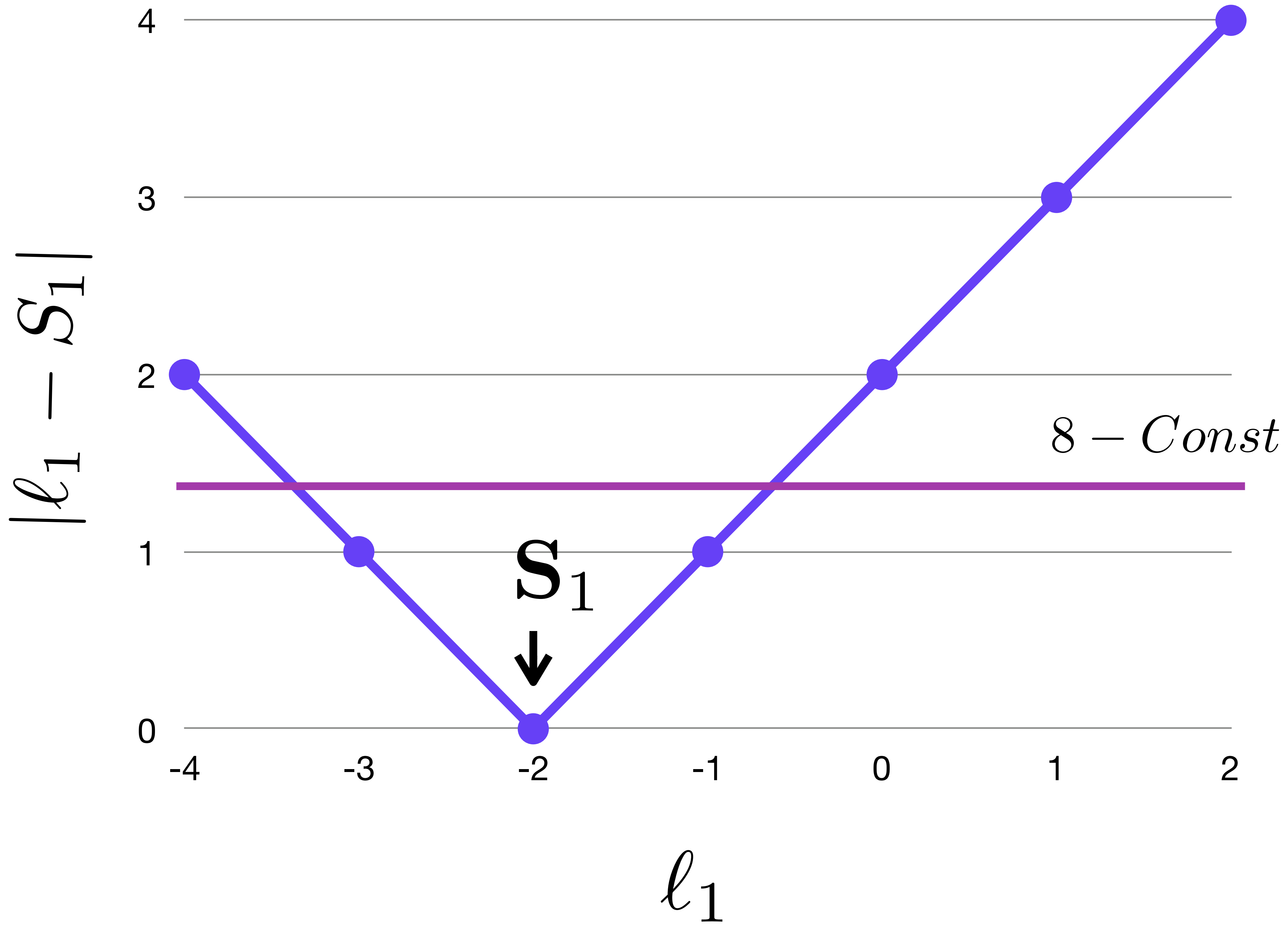
Else go back to step 1

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(\left|\ell_1-\mathbf{S}_1\right|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

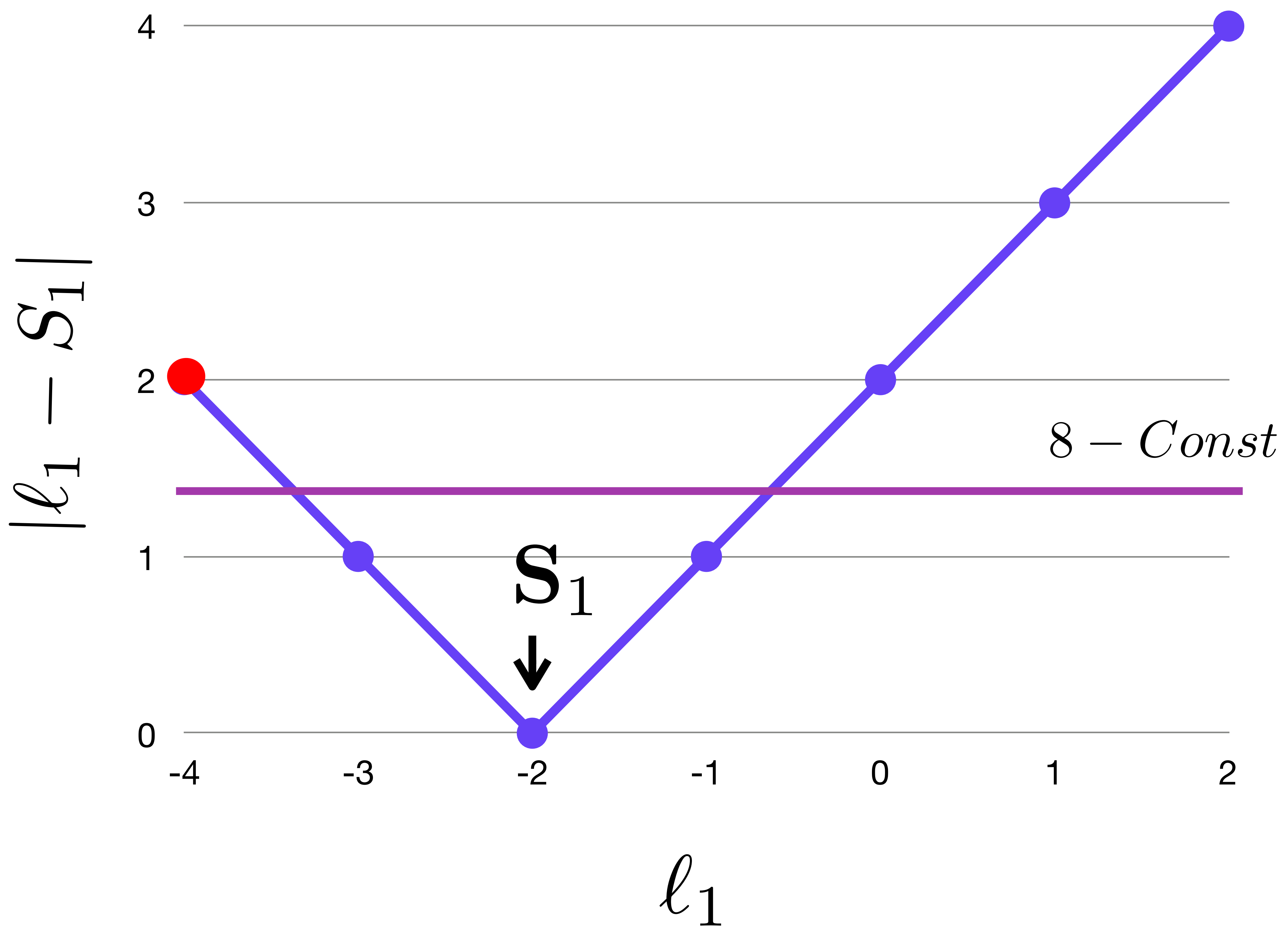
$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

Sketch of the attack

C	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(\left|\ell_1-\mathbf{S}_1\right|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

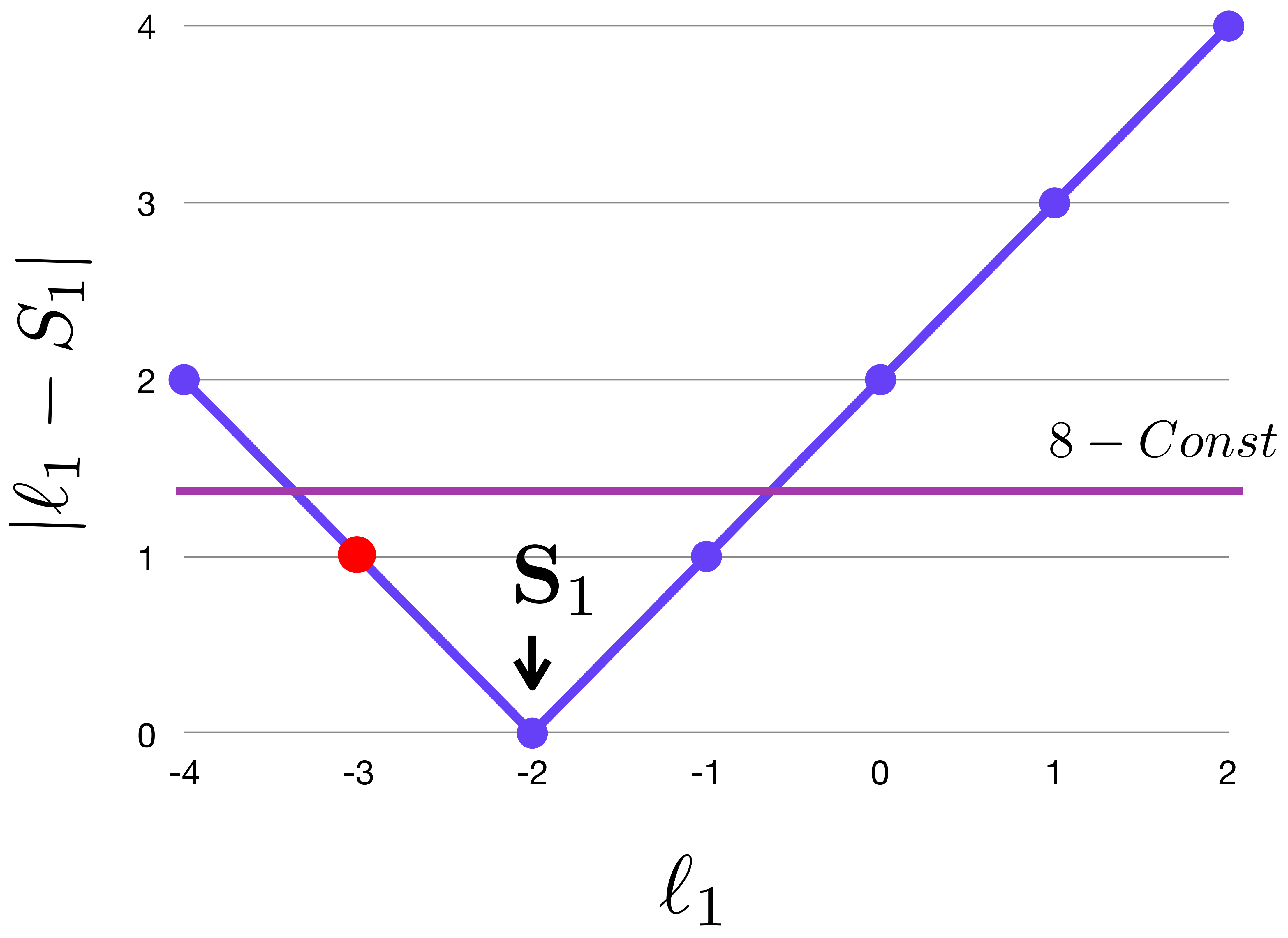
Else go back to step 1

+	-	-	-	+	+	+	+

Sketch of the attack

C	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(\left|\ell_1 - \mathbf{S}_1\right| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

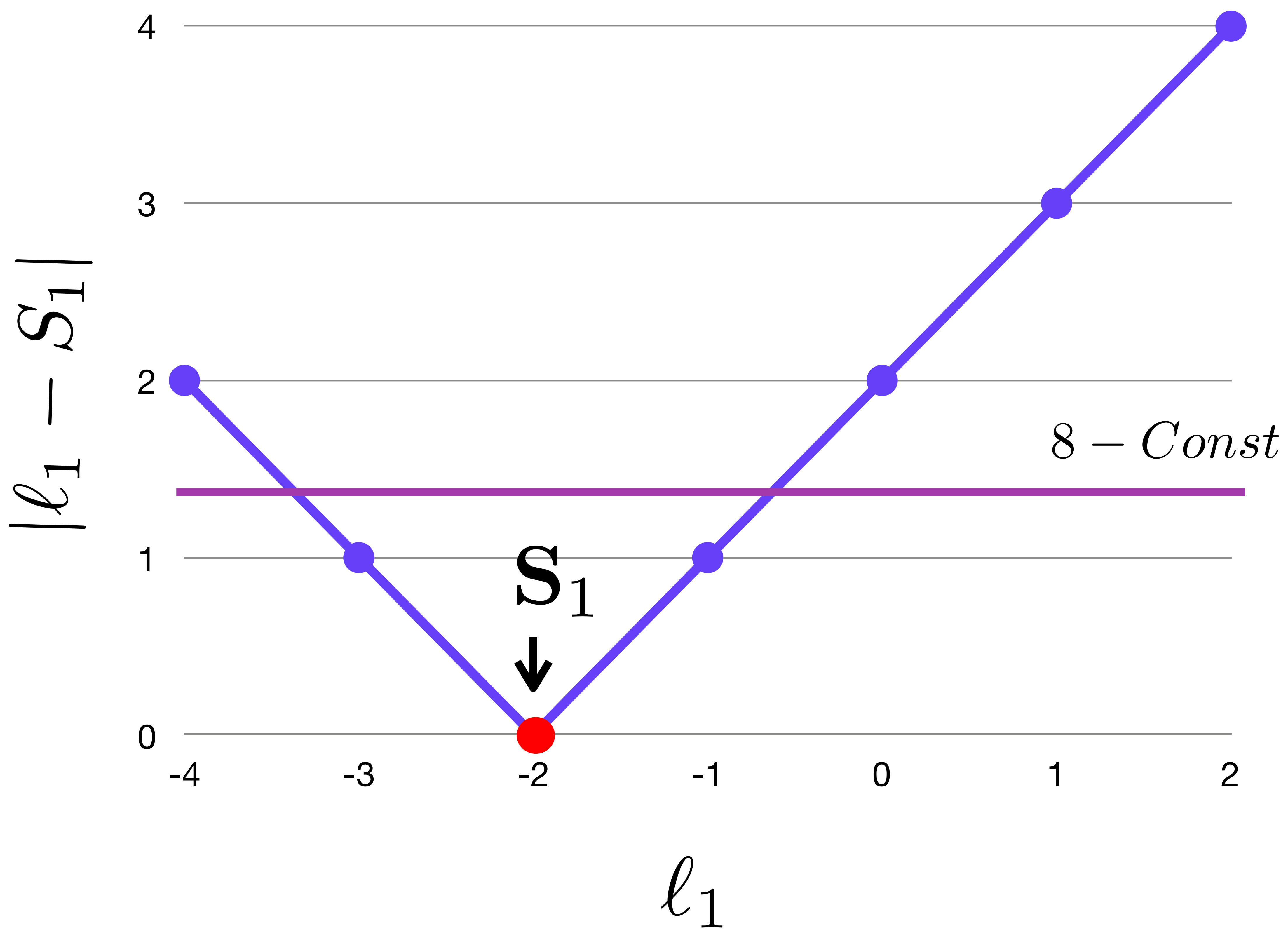
+	-	-	-	+	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(|\ell_1 - \mathbf{S}_1| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

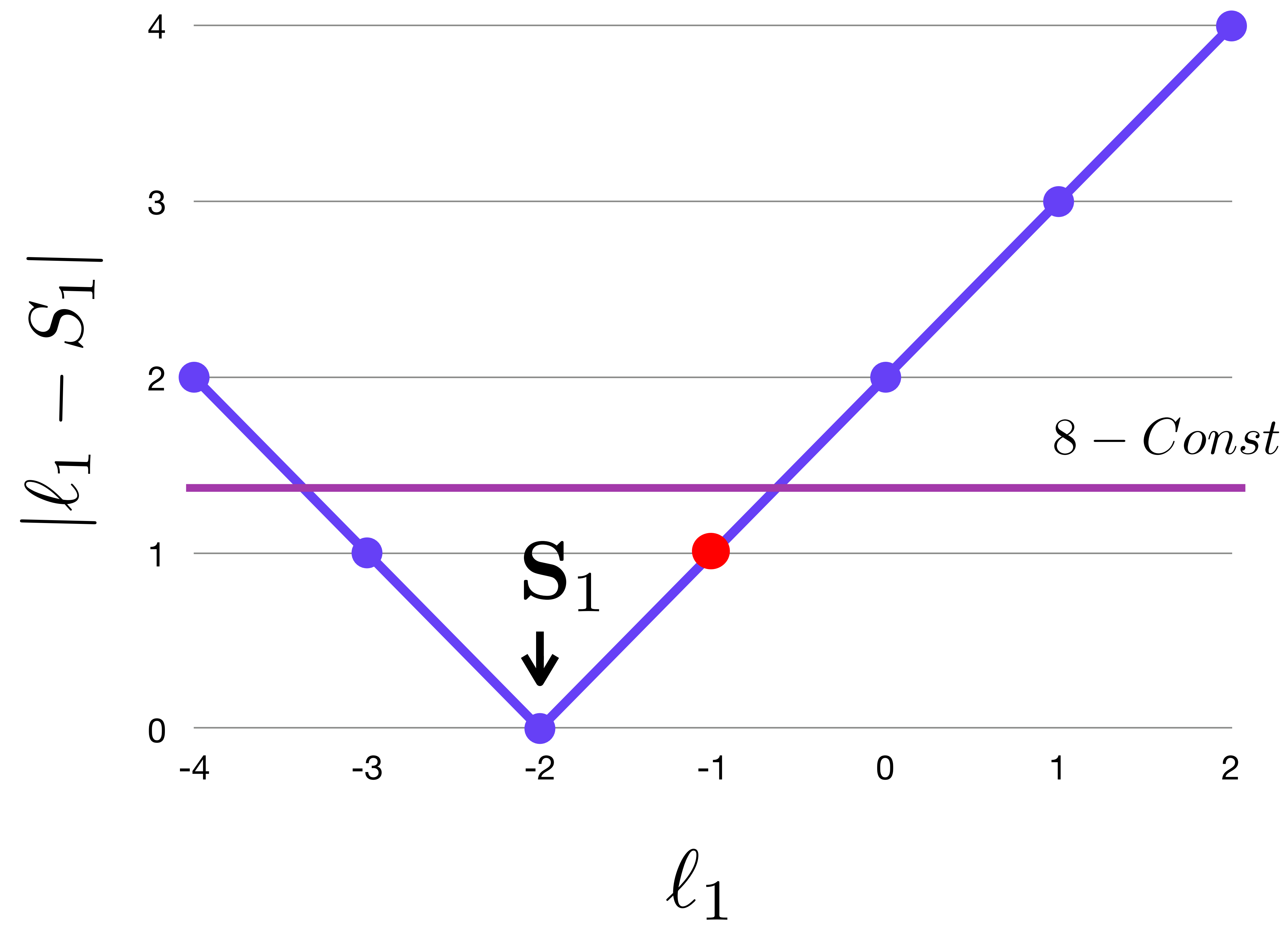
+	-	-	-	+	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(\left|\ell_1-\mathbf{S}_1\right|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of *Sign*

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

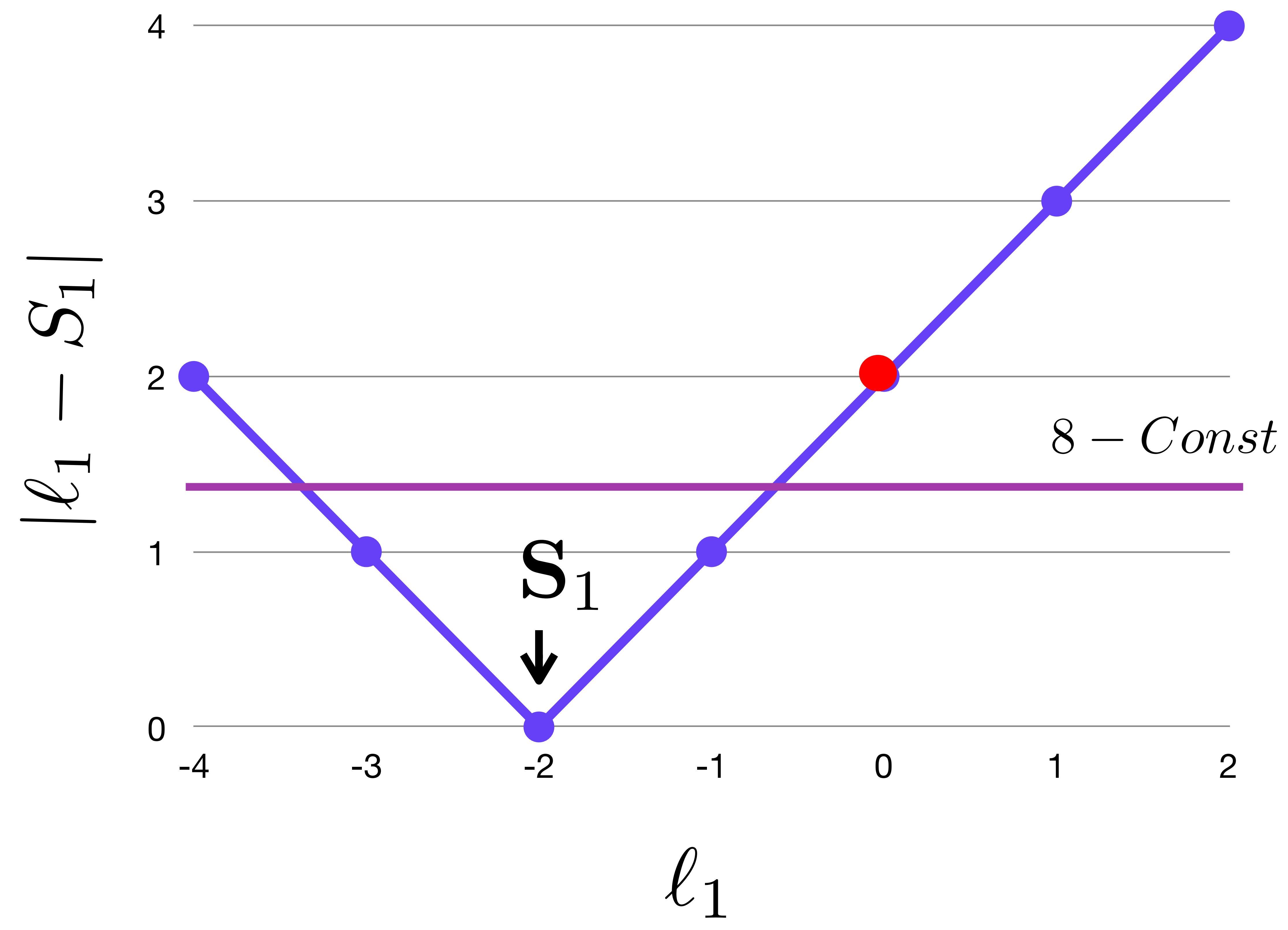
+	-	-	-	+	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(\left|\ell_1 - \mathbf{S}_1\right| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

→ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

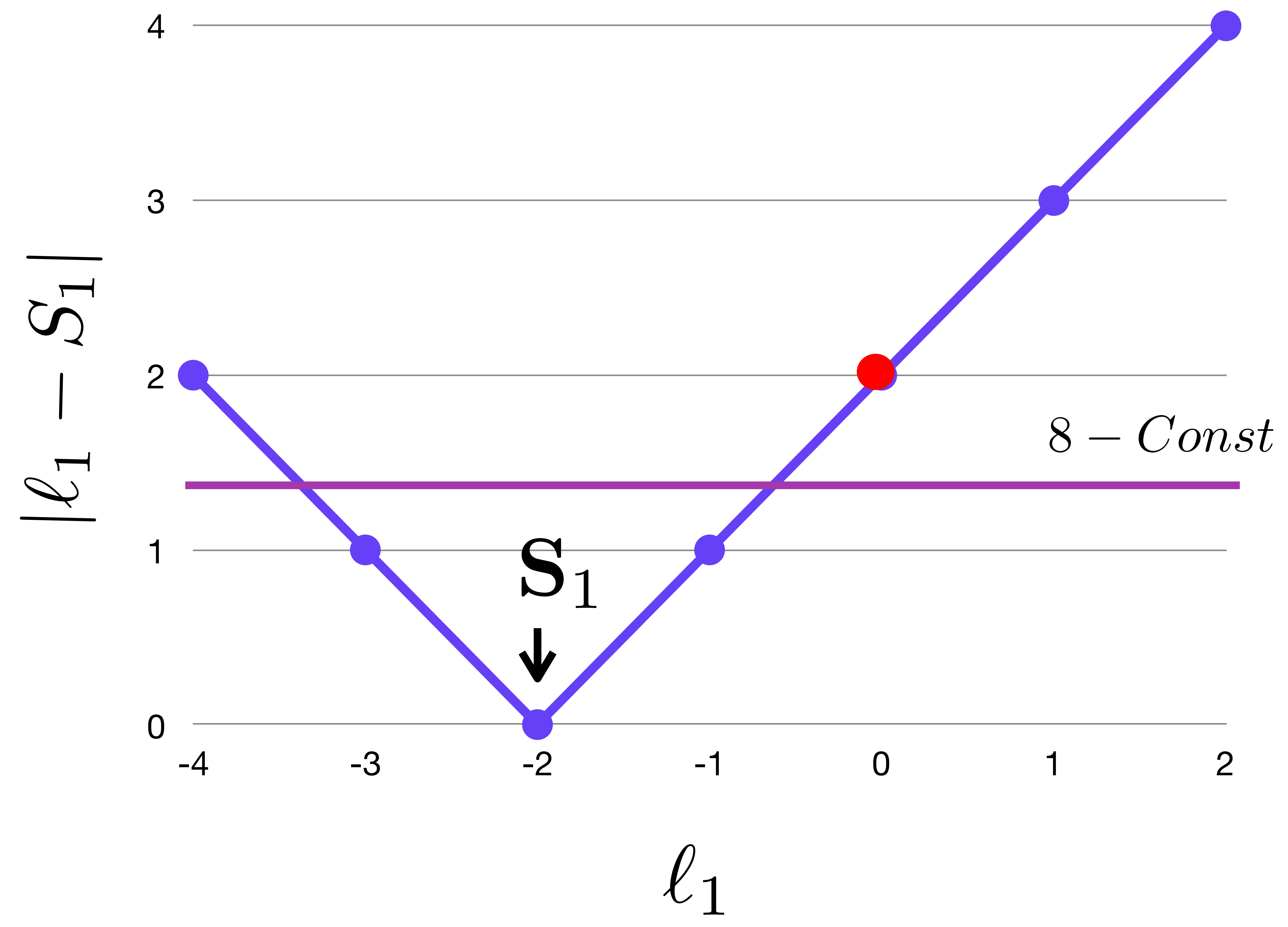
+	-	-	-	+	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(\left|\ell_1-\mathbf{S}_1\right|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

→ If 2 changes of *Sign*

$\rightarrow \mathbf{S}_1$ is found (symmetry)

Else go back to step 1

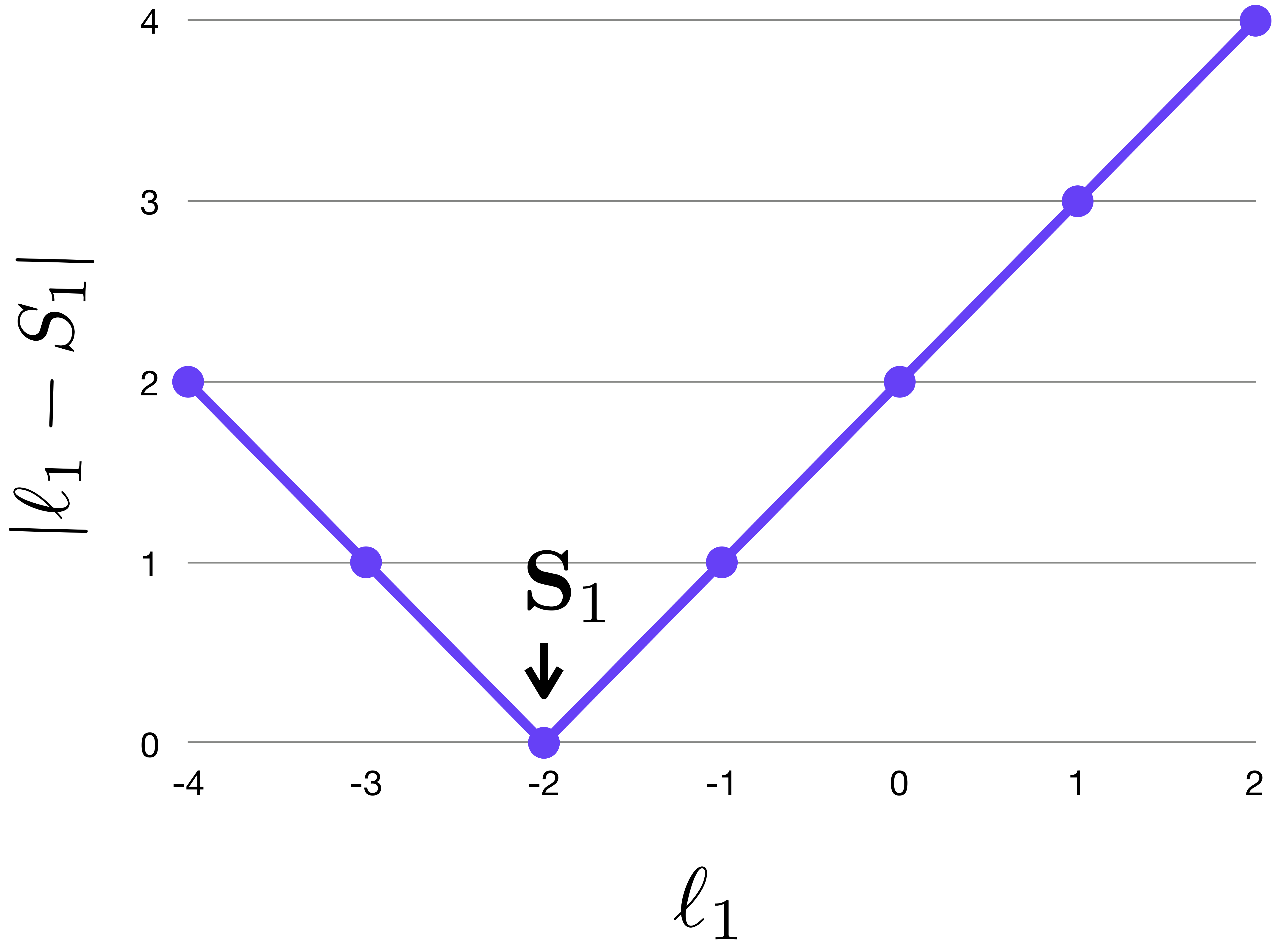
+	-	-	-	+	+	+	+

Sketch of the attack

$\mathbf{C}$

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(|\ell_1 - \mathbf{S}_1| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow \mathbf{S}_1$ is found (symmetry)

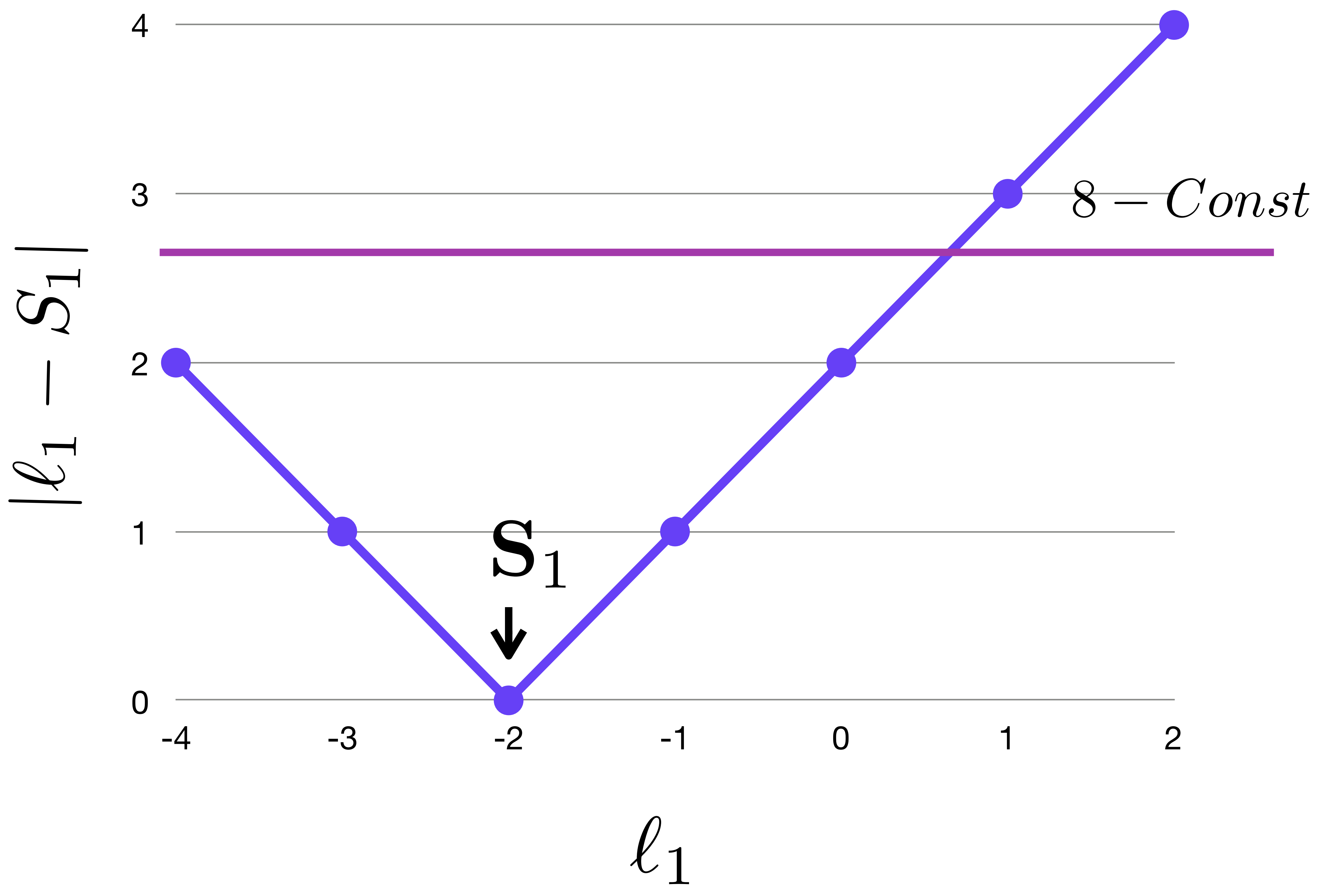
Else go back to step 1

Sketch of the attack

$\mathbf{C}$

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(|\ell_1-\mathbf{S}_1|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

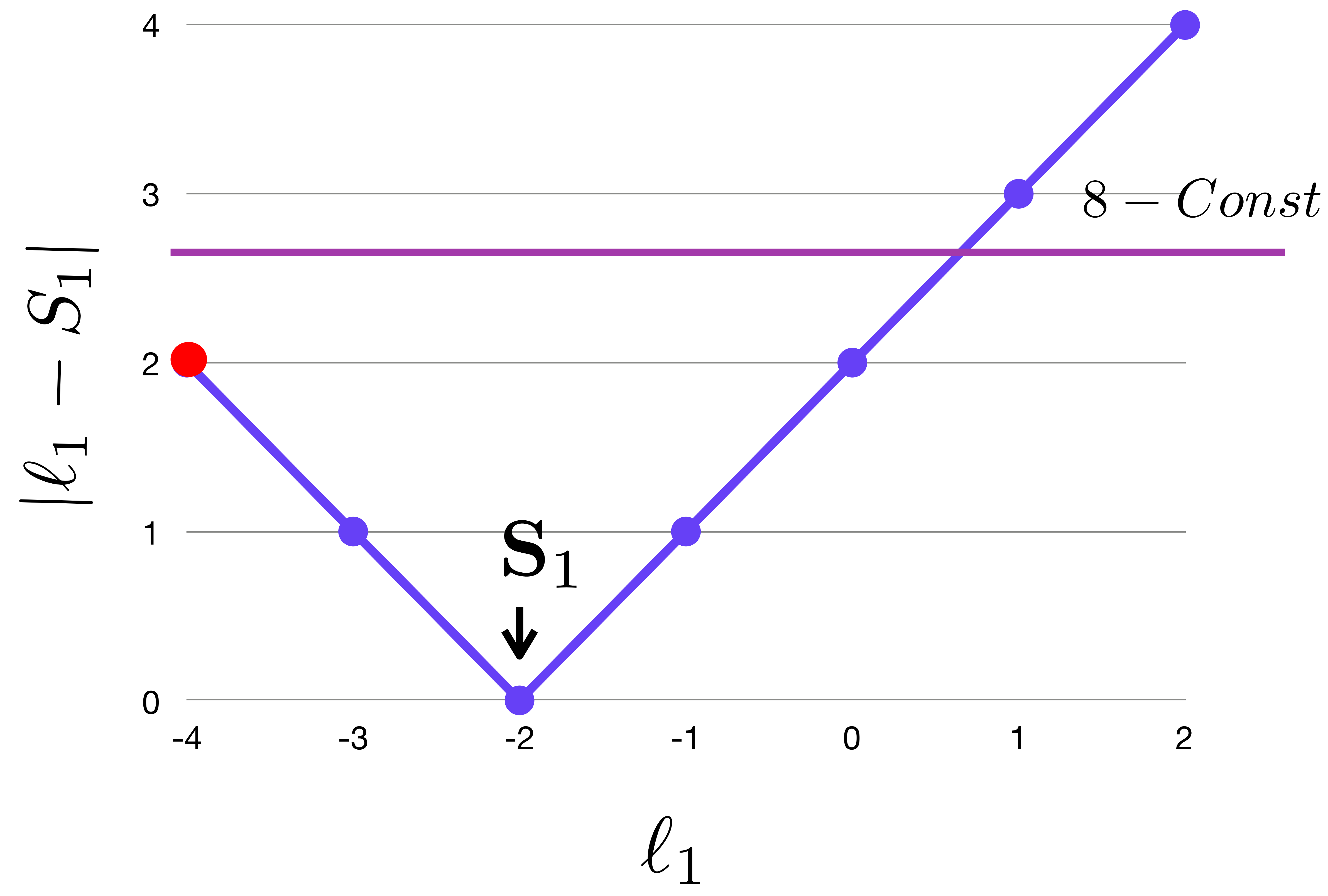
$\rightarrow \mathbf{S}_1$ is found (symmetry)

Else go back to step 1

Sketch of the attack

C	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(|\ell_1-\mathbf{S}_1|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow \mathbf{S}_1$ is found (symmetry)

Else go back to step 1

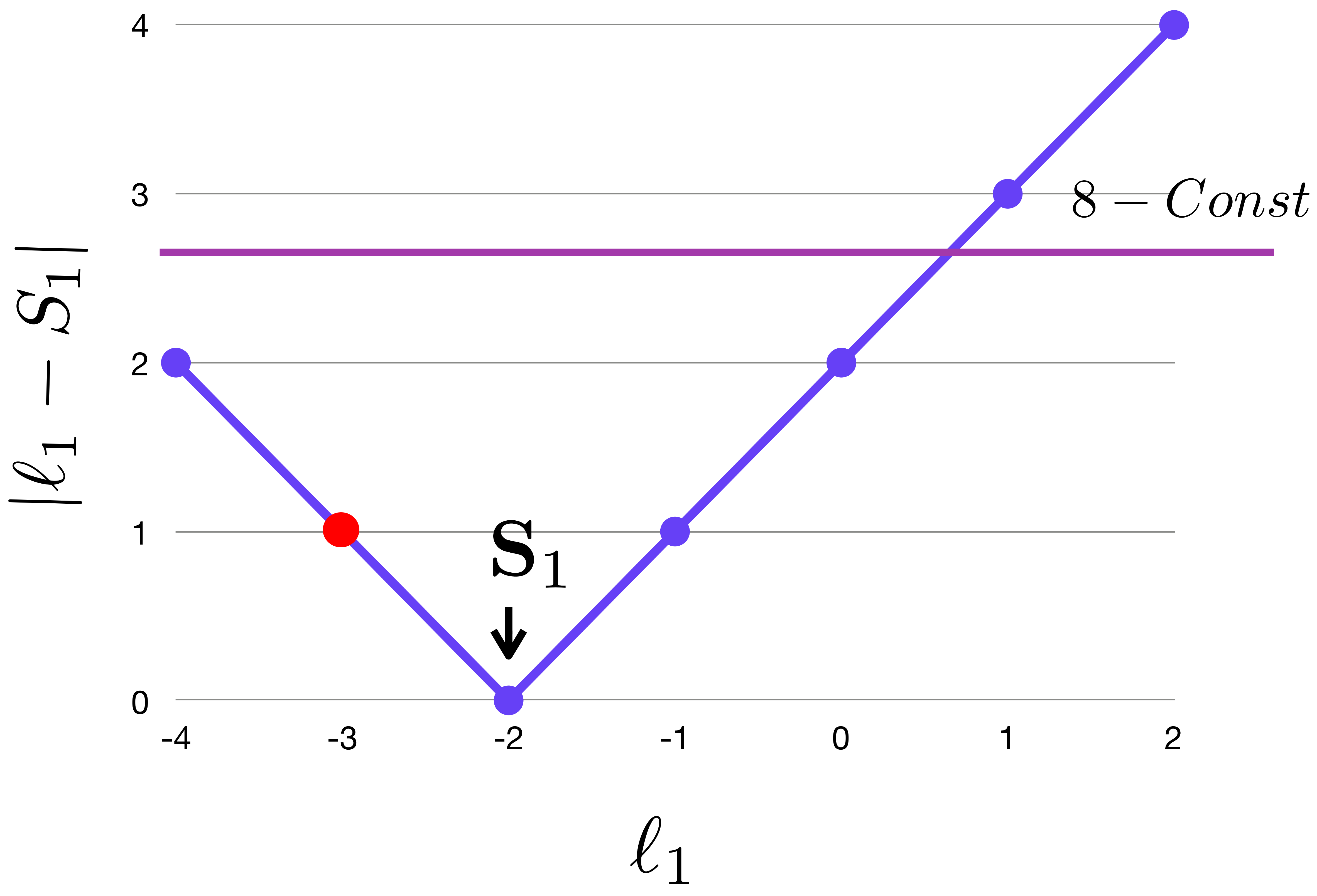
-	-	-	-	-	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(\left|\ell_1-\mathbf{S}_1\right|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

→ $\mathbf{S}_1$ is found (symmetry)

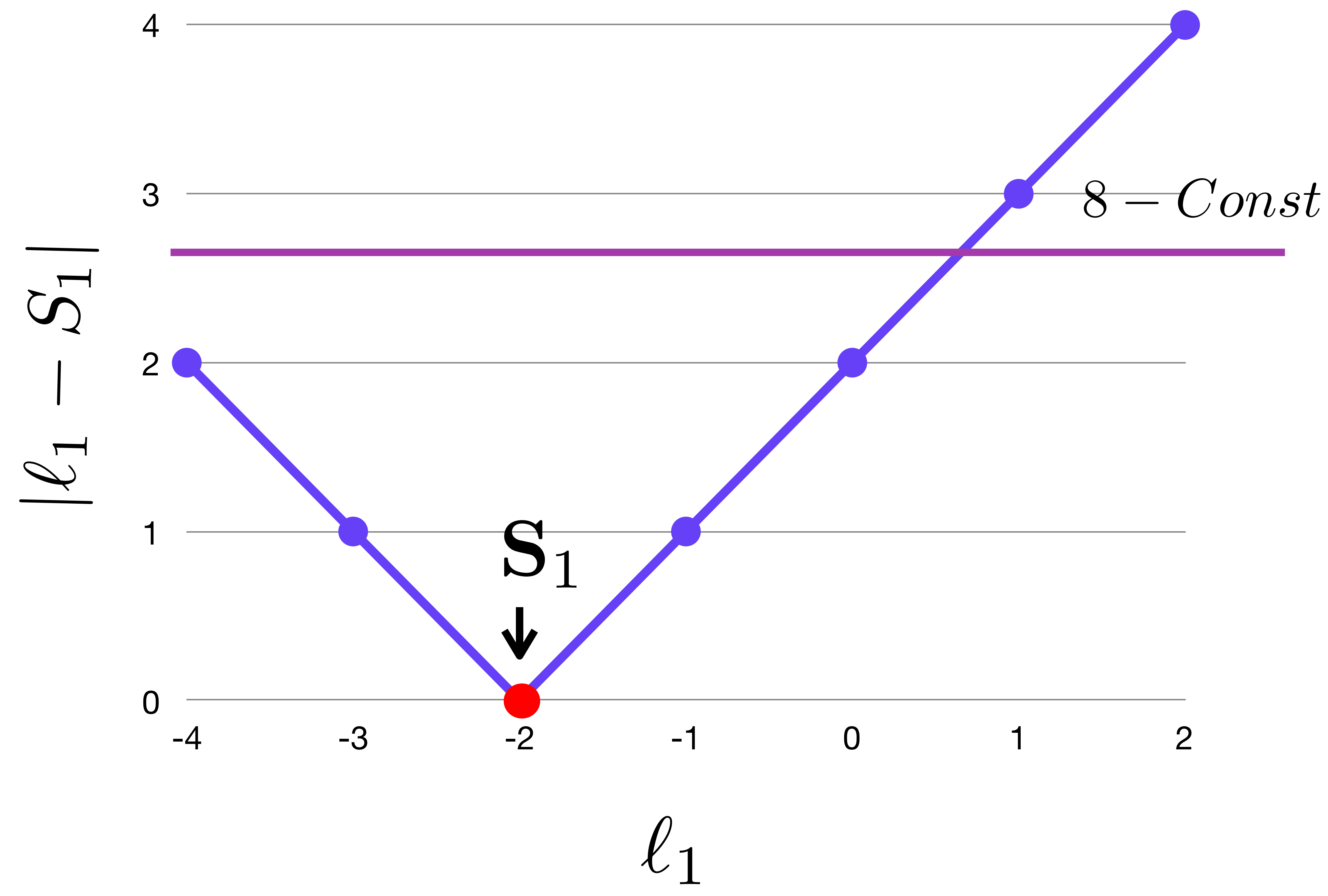
Else go back to step 1

-	-	-	-	-	+	+	+

Sketch of the attack

C	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j-\mathbf{S}_j\right|-8\right)=Sign\left(|\ell_1-\mathbf{S}_1|+Const-8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

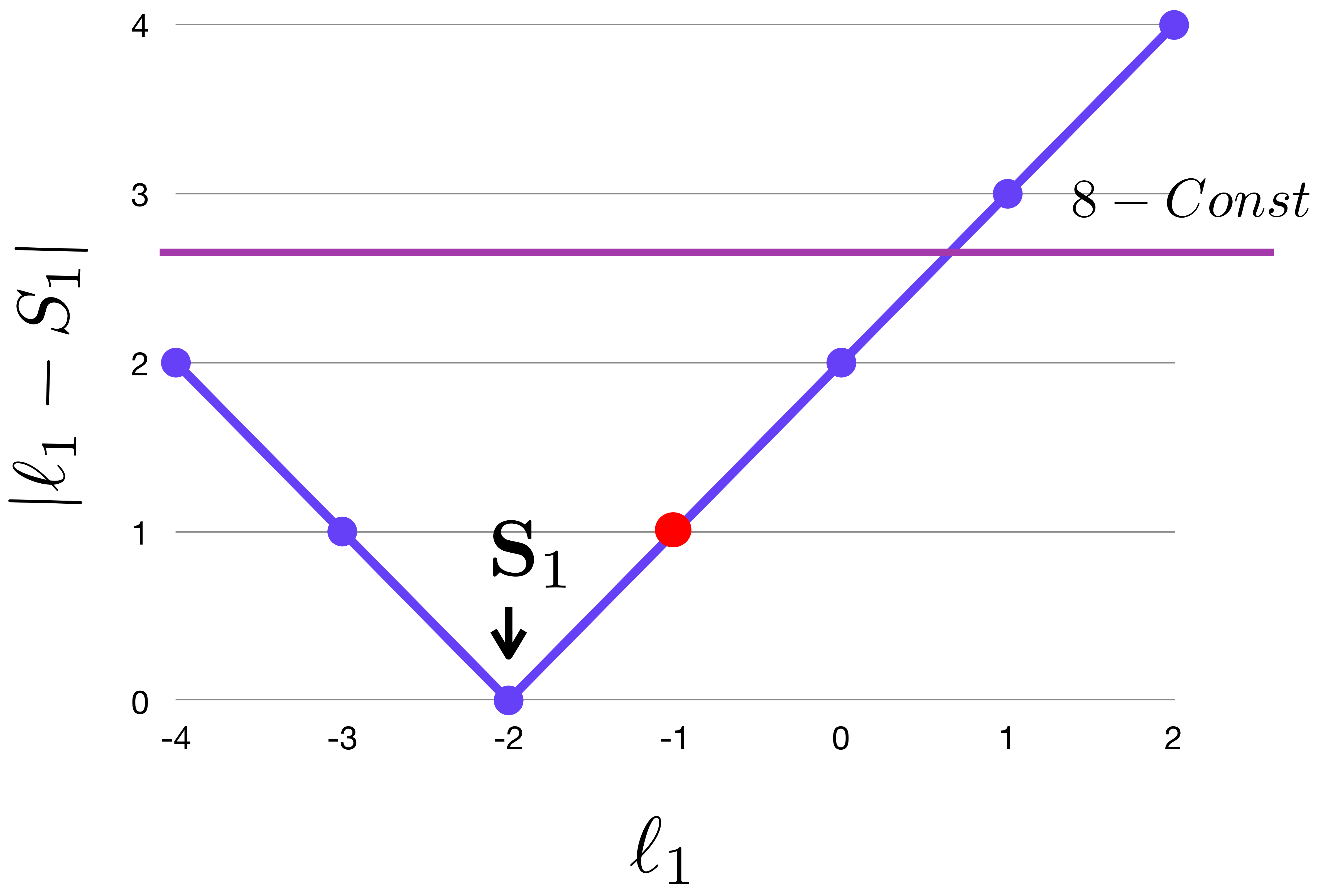
-	-	-	-	-	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(\left|\ell_1 - \mathbf{S}_1\right| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

→ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

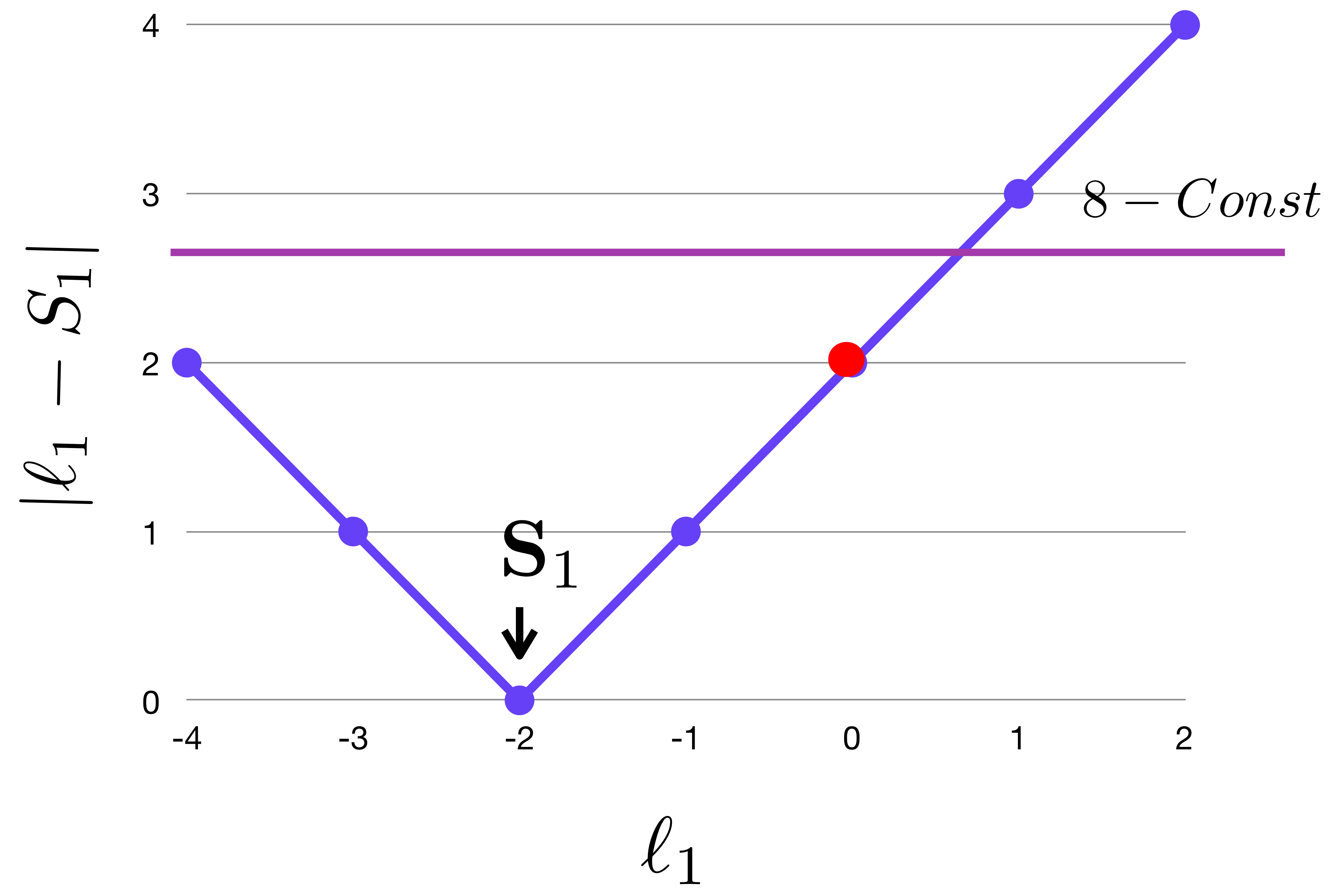
-	-	-	-	-	+	+	+

Sketch of the attack

$\mathbf{C}$

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(\left|\ell_1 - \mathbf{S}_1\right| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

→ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

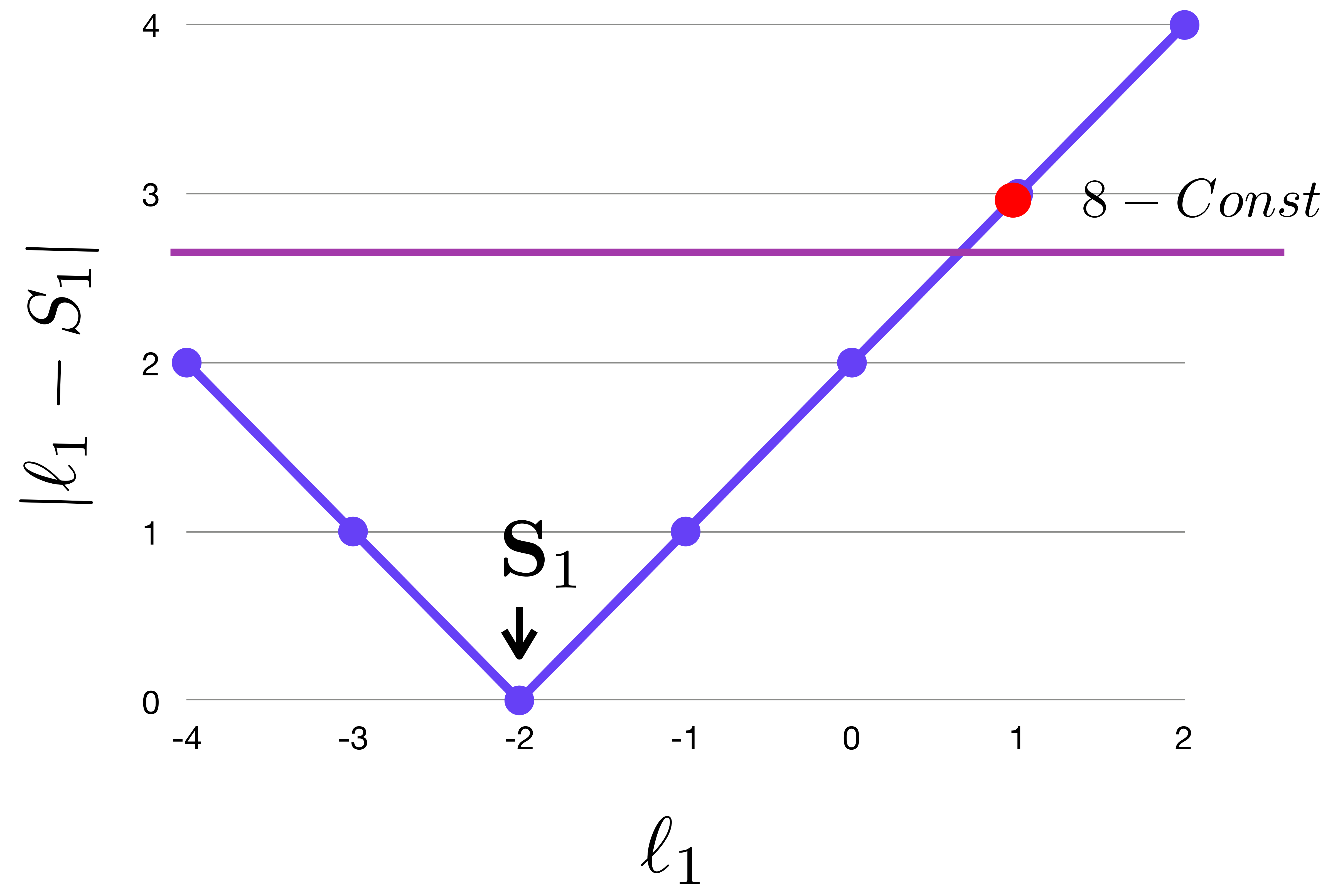
-	-	-	-	-	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\left|\ell_j - \mathbf{S}_j\right| - 8\right) = Sign\left(|\ell_1 - \mathbf{S}_1| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

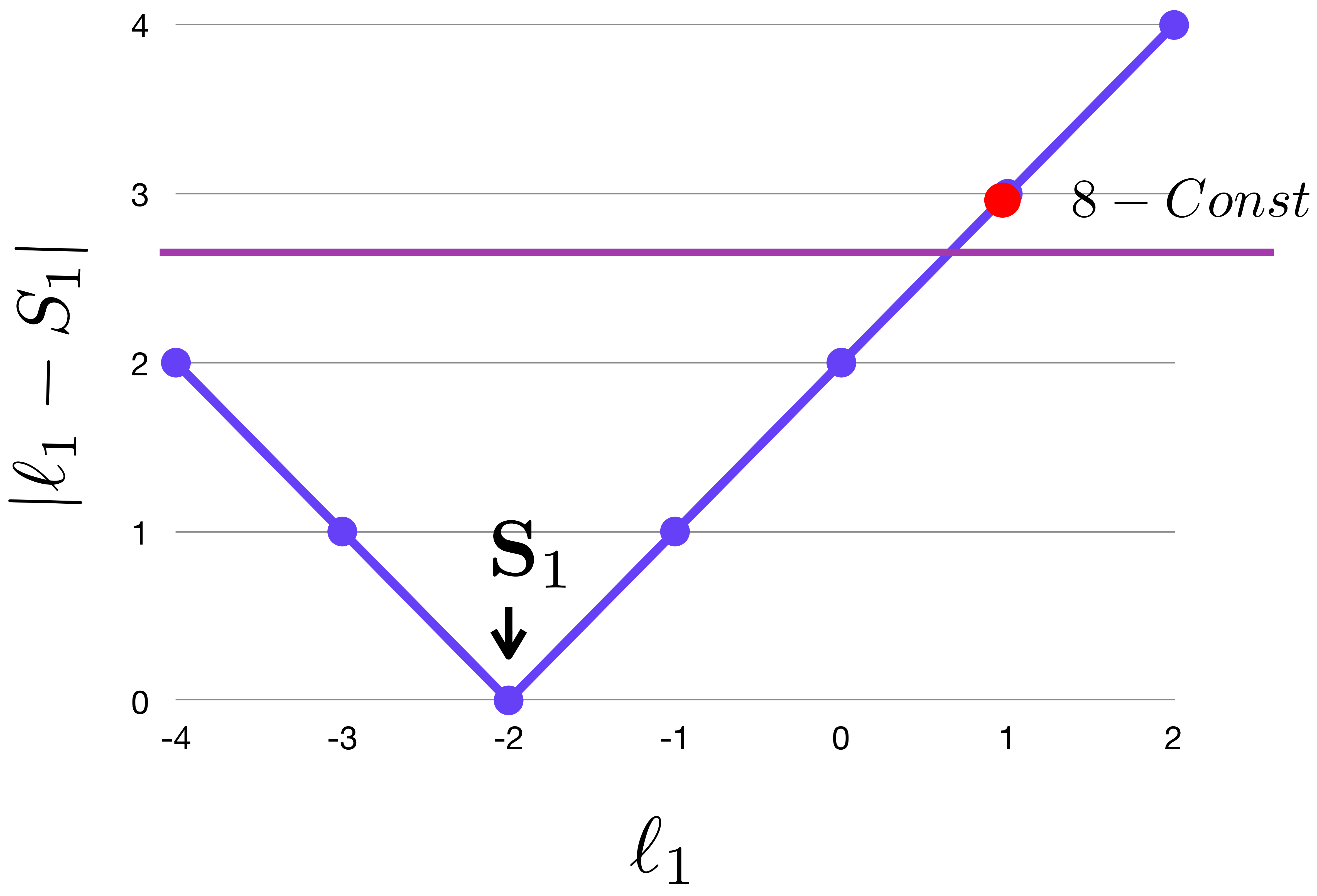
-	-	-	-	-	+	+	+

Sketch of the attack

C

ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4	0	0	0	0	0

$$Sign\left(\sum_{j=1}^{j=4}\Big|\ell_j - \mathbf{S}_j\Big| - 8\right) = Sign\left(|\ell_1 - \mathbf{S}_1| + Const - 8\right)$$



1. Draw ℓ_2, ℓ_3 and ℓ_4 at random
2. Query for successive ℓ_1

If 2 changes of $Sign$

$\rightarrow$ $\mathbf{S}_1$ is found (symmetry)

Else go back to step 1

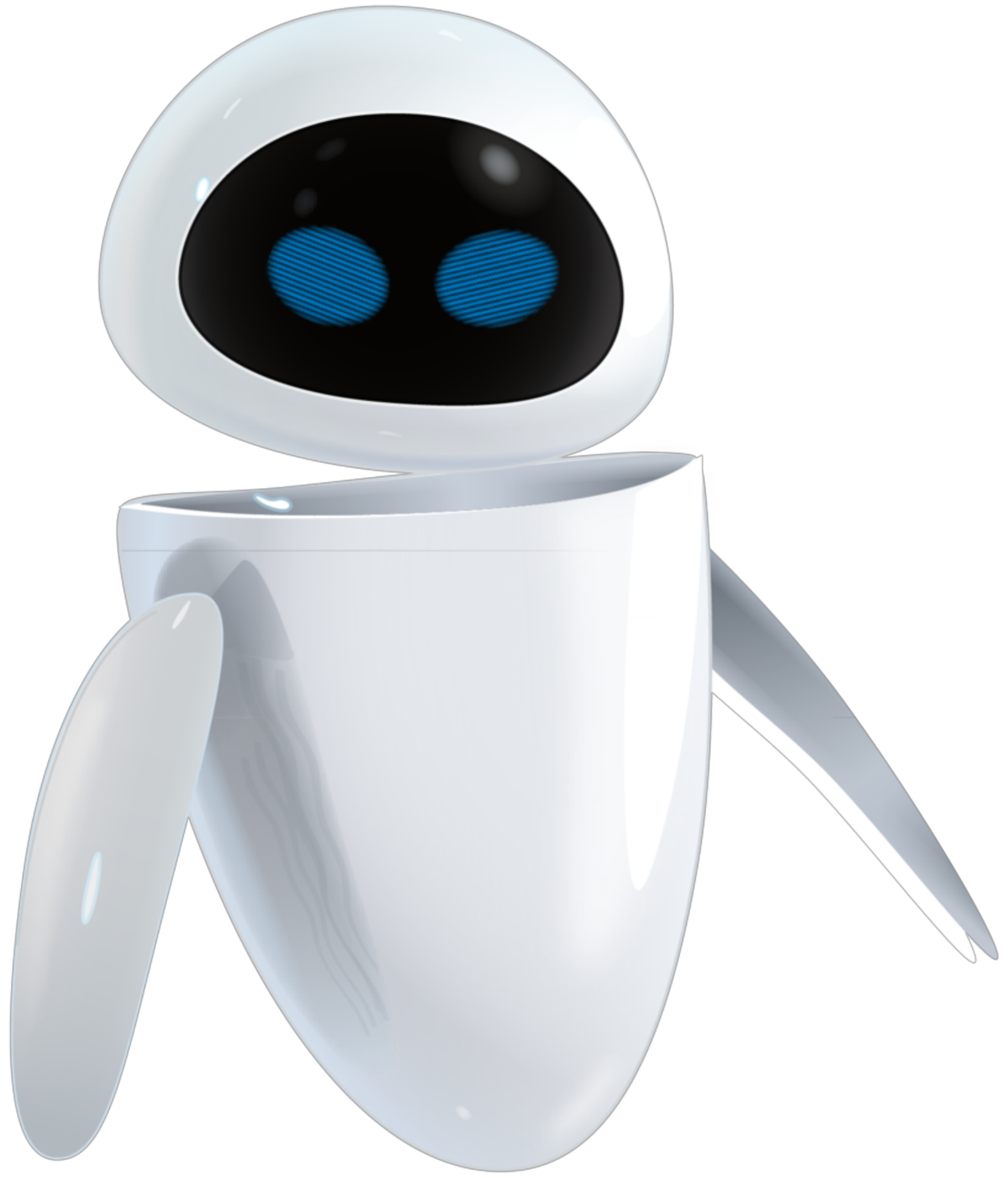
-	-	-	-	-	+	+	+

```
Magma V2.23-1 (STUDENT)   Tue Sep  4 2018 17:25:28    [Seed = 1072351204]
Type ? for help.  Type <Ctrl>-D to quit.

Loading file "NewHopeAttack.mag"
```

```
Magma V2.23-1 (STUDENT)   Tue Sep  4 2018 17:25:28    [Seed = 1072351204]
Type ? for help.  Type <Ctrl>-D to quit.

Loading file "NewHopeAttack.mag"
```

Results

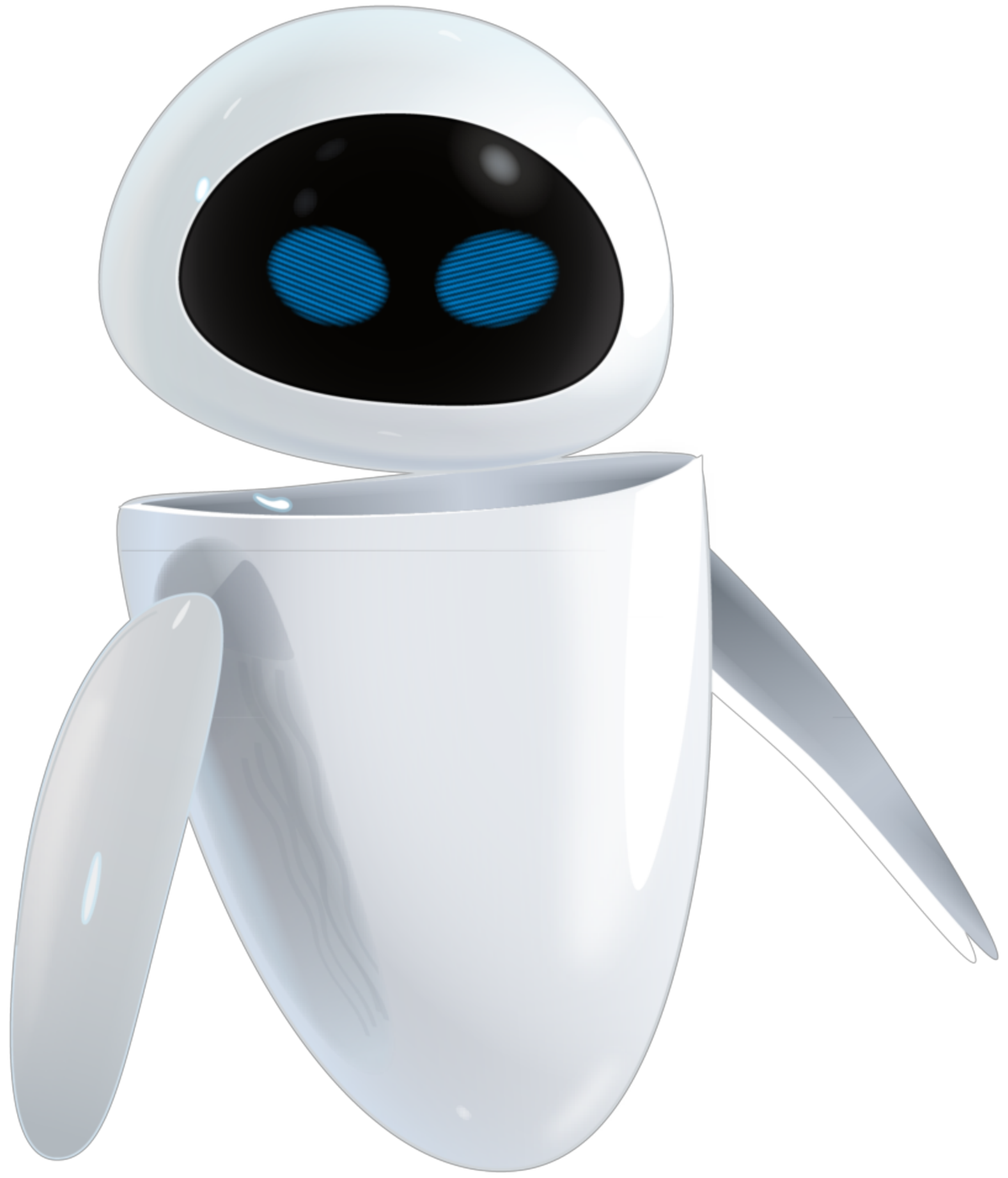
Average queries	Success probability
16 700	95 %

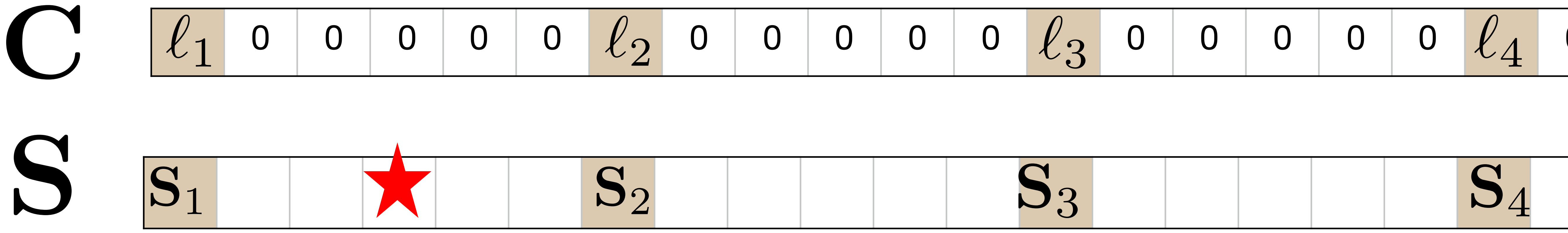


Results

Average queries	Success probability
16 700	95 %

Sometimes the secret has large coefficients the induce errors outside of the target





C	ℓ_1	0	0	0	0	0	ℓ_2	0	0	0	0	0	ℓ_3	0	0	0	0	0	ℓ_4
S	$\mathbf{s}_1$			★			$\mathbf{s}_2$						$\mathbf{s}_3$						$\mathbf{s}_4$

RSAConference2019

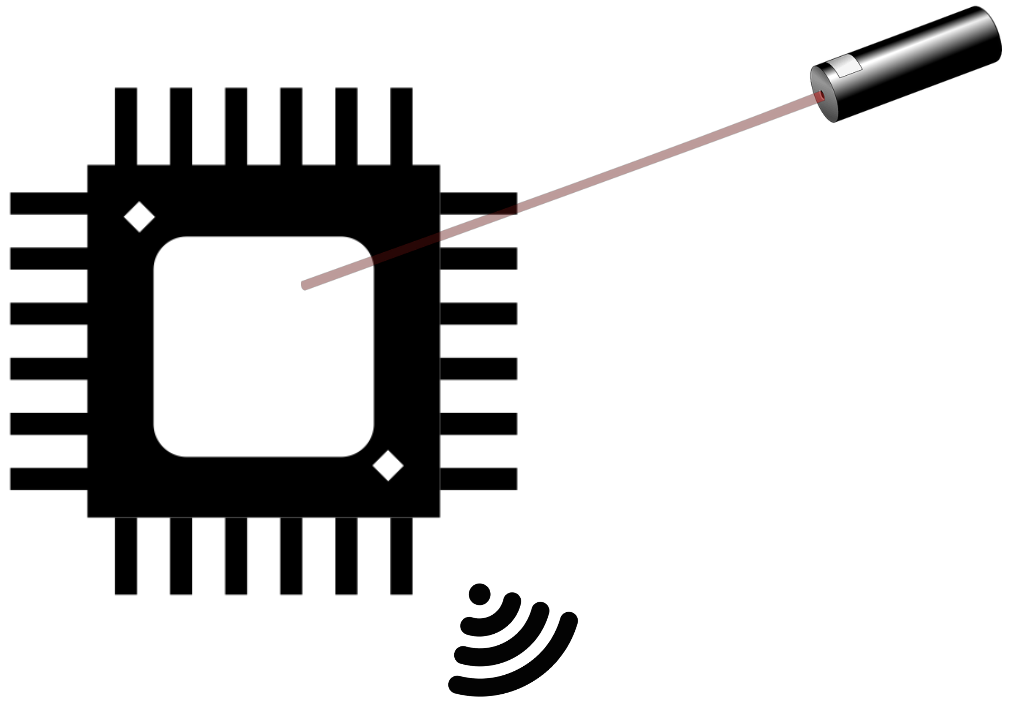
- The attack can be extended to the CCA version with a stronger model

NewHope CCA version

Fujisaki-Okamoto transform (variant): Alice checks Eve’s computation with her seed

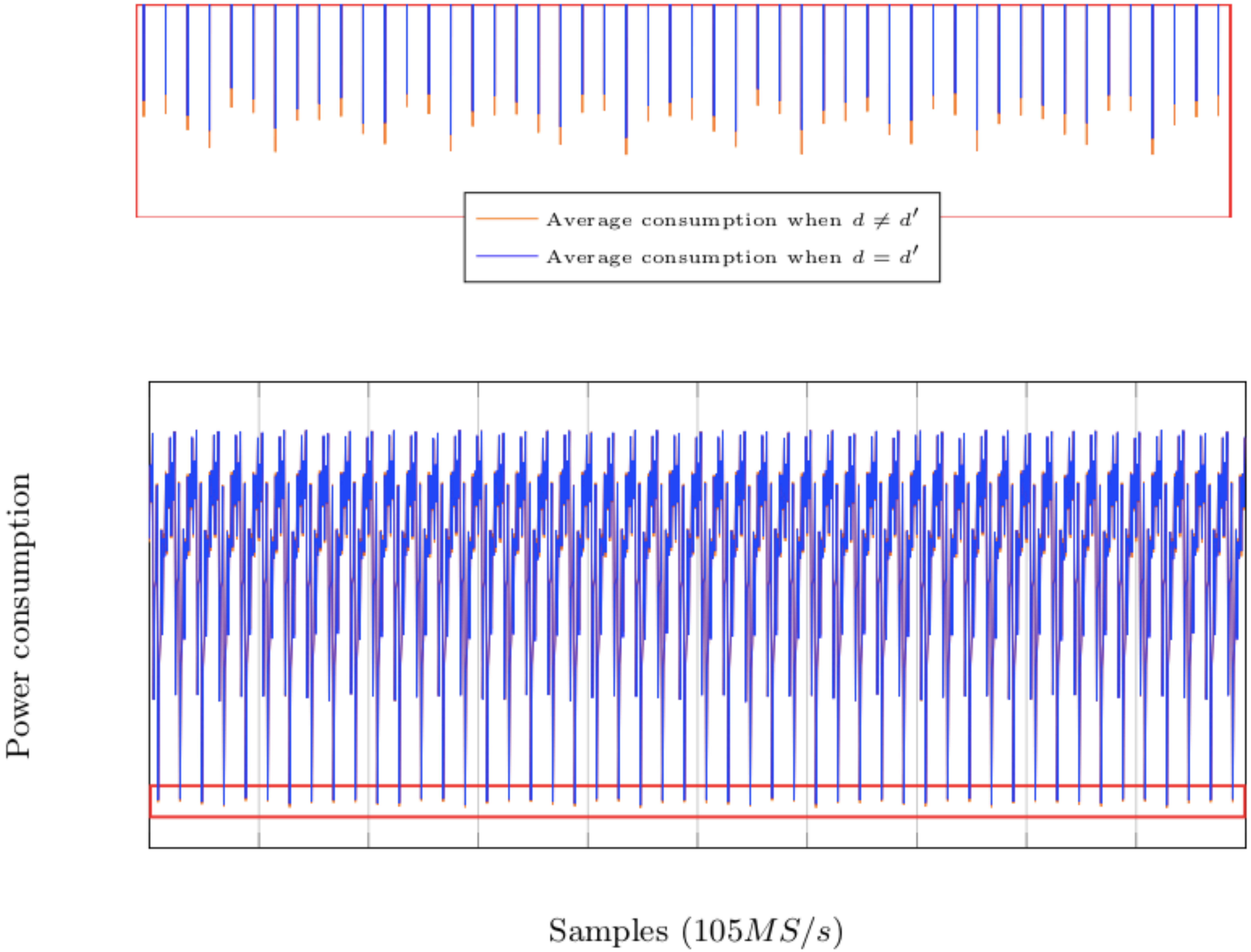
NewHope CCA version

Fujisaki-Okamoto transform (variant): Alice checks Eve’s computation with her seed



Key caching is still insecure with side channels

Single fault attack
Differential power analysis


Average consumption when $d \neq d'$
Average consumption when $d = d'$
Power consumption
Samples $(105MS/s)$

Take away message

Be careful with key caching when implementing lattice based schemes

Refresh the sk/pk pair as often as possible

Thank you for you attention

Full version of the paper

https://eprint.iacr.org/2019/075