Public-Key Encryption with Lazy Parties

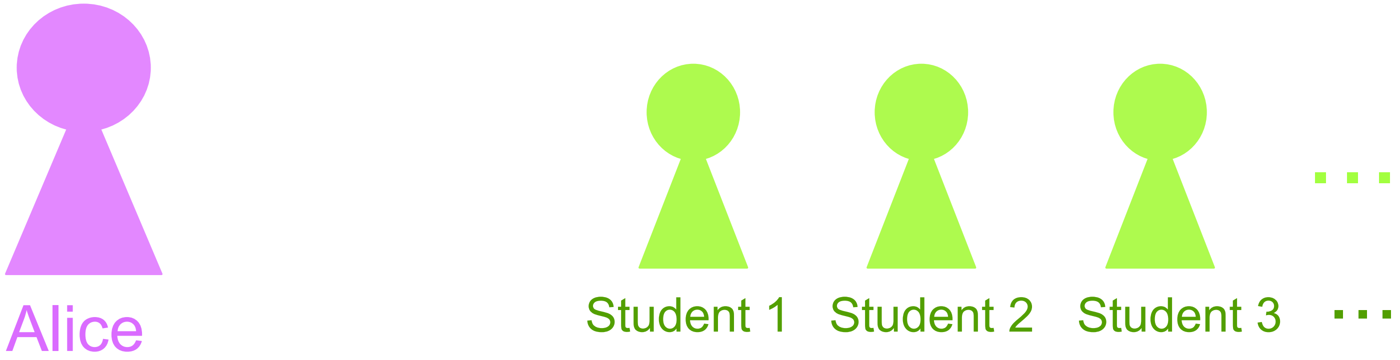
Kenji Yasunaga

Institute of Systems, Information Technologies and Nanotechnologies (ISIT), Japan

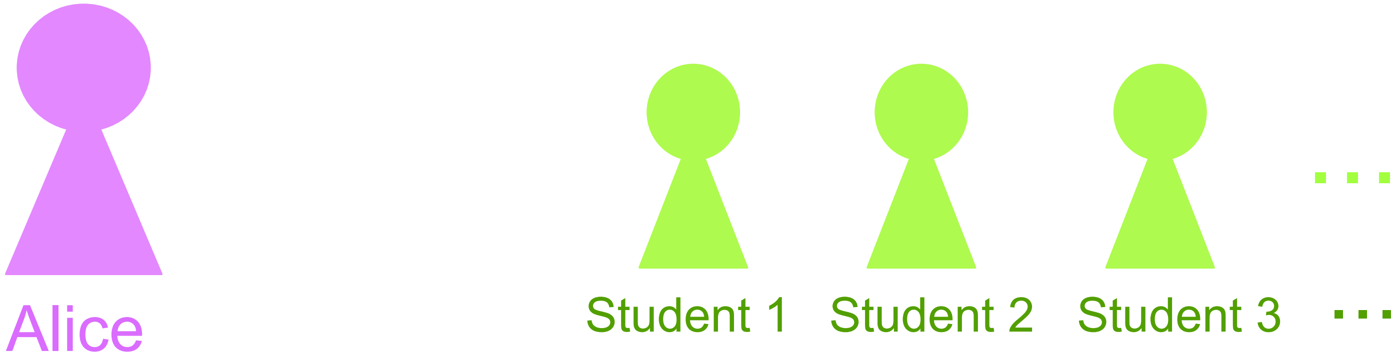
Presented at SCN 2012

IMI Crypto Seminar 2012.12.17

One day in a class,

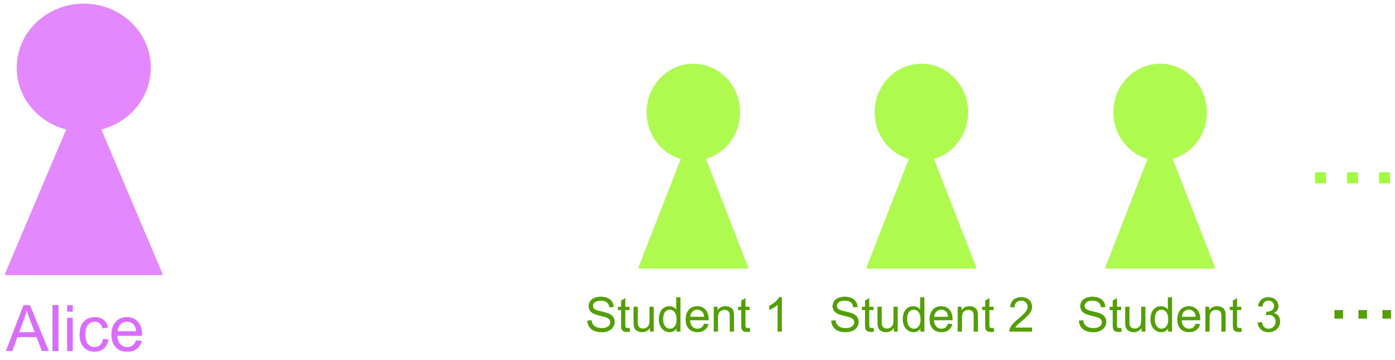


One day in a class,



The class “Introduction to Cryptography”

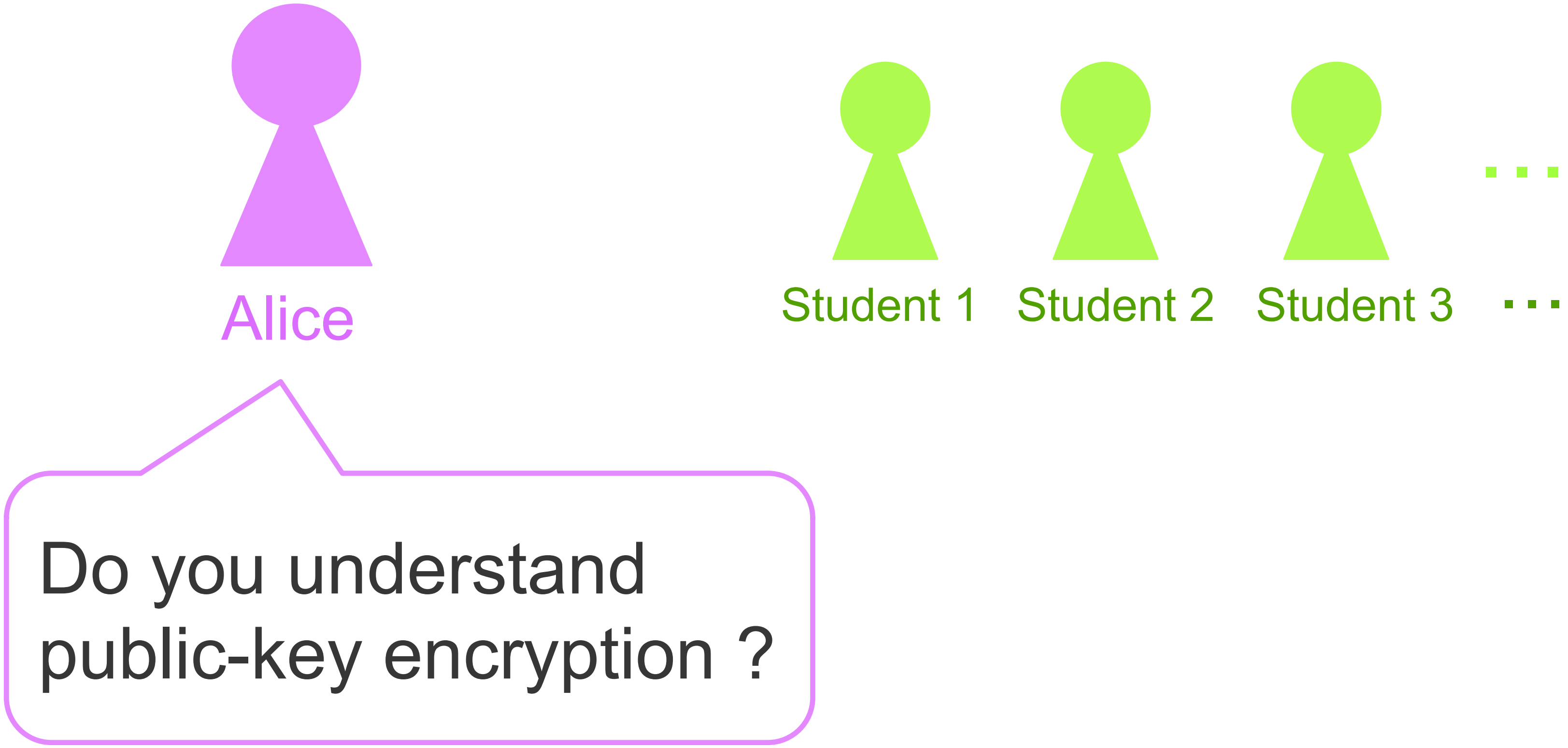
One day in a class,



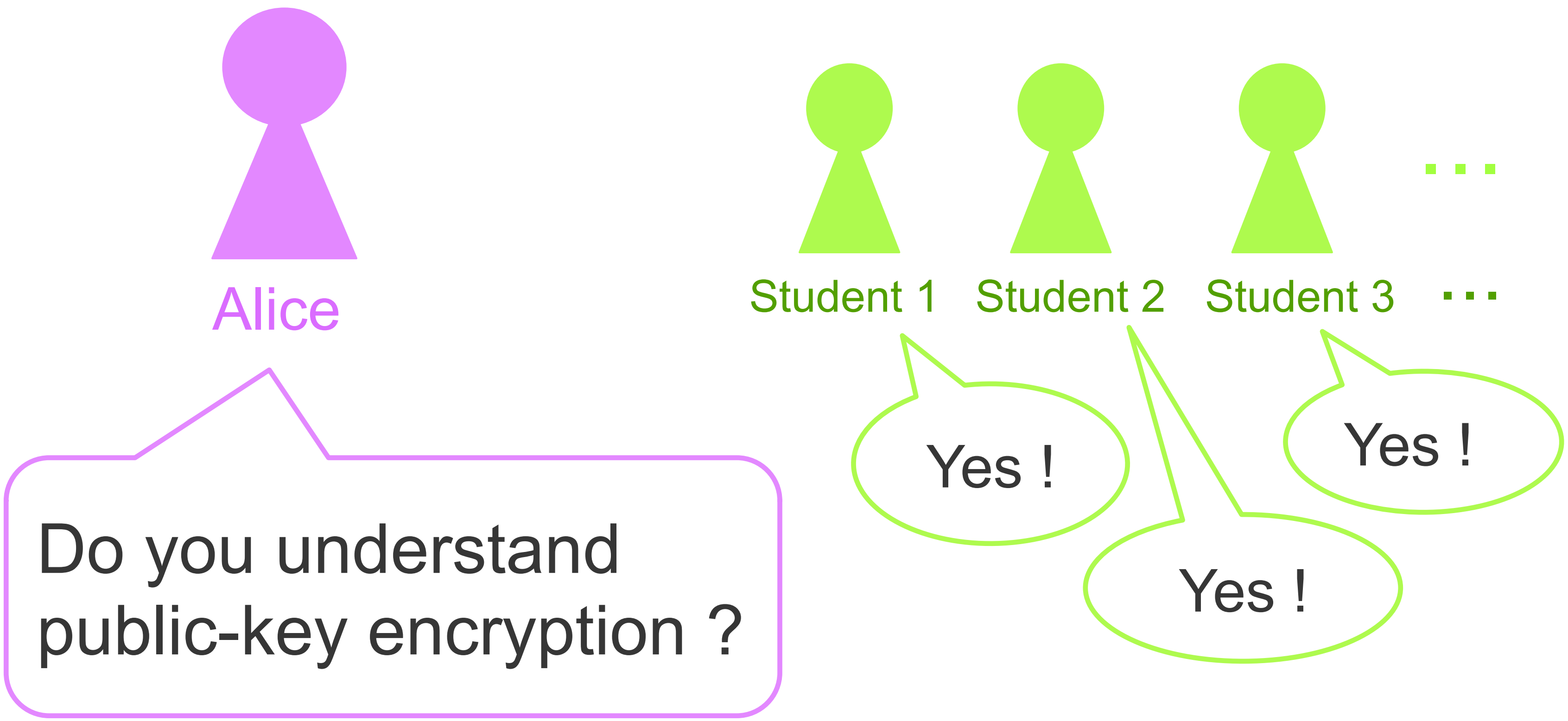
The class "Introduction to Cryptography"

The final exam has finished.

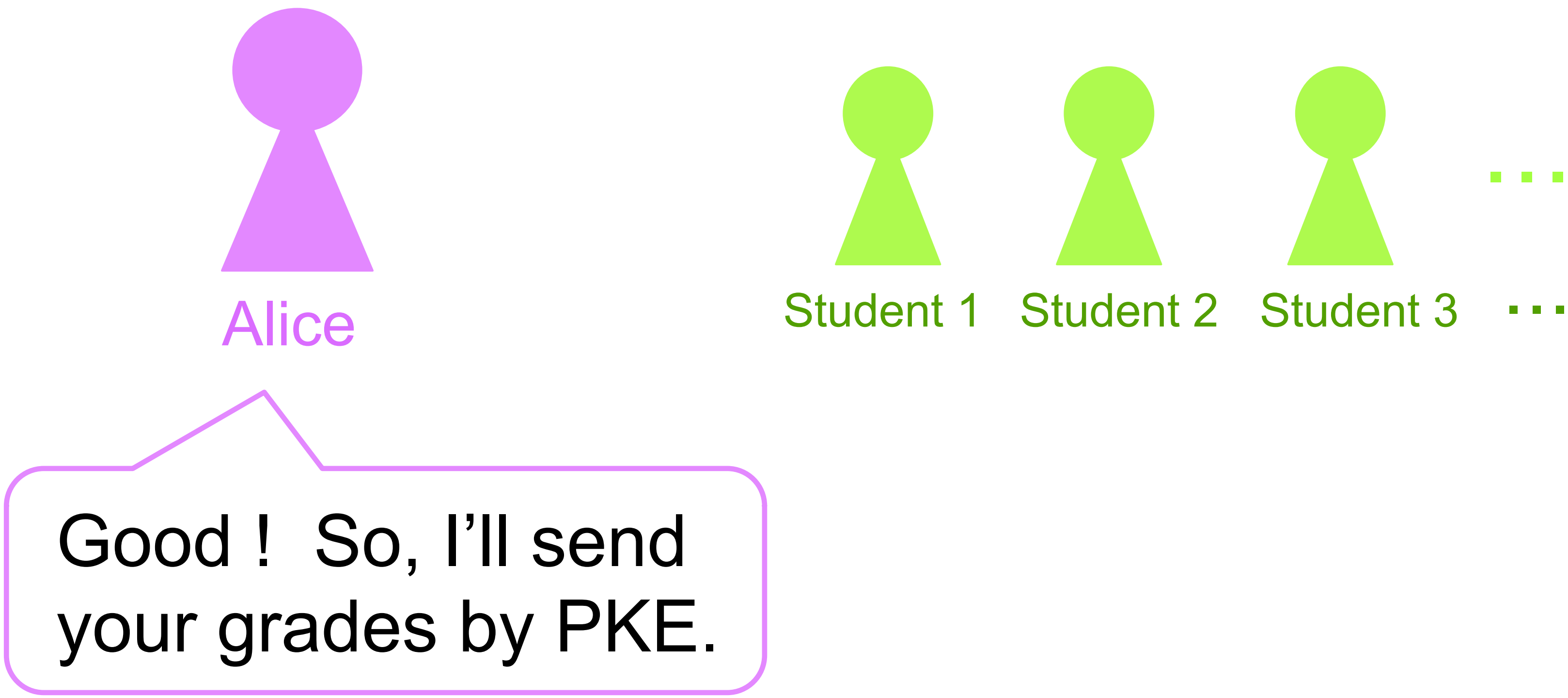
One day in a class,



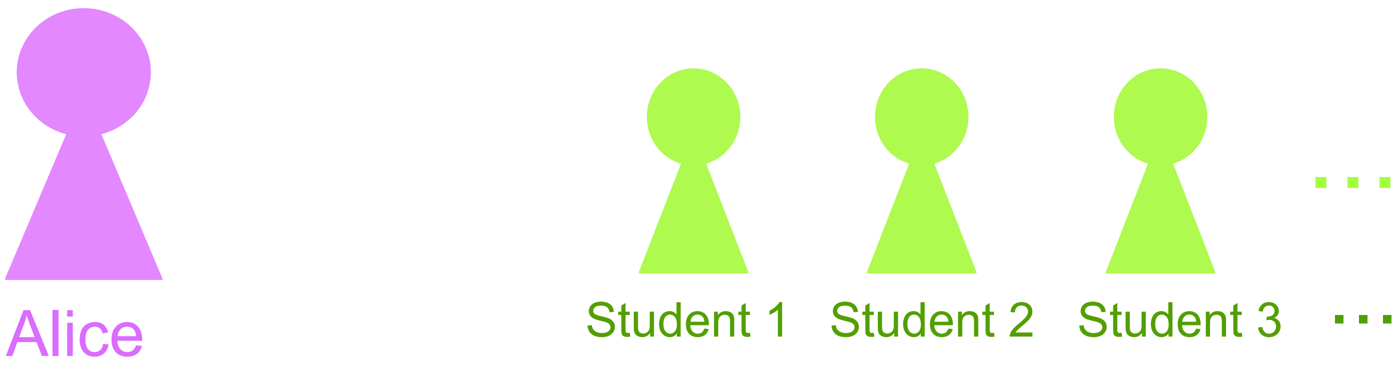
One day in a class,



One day in a class,



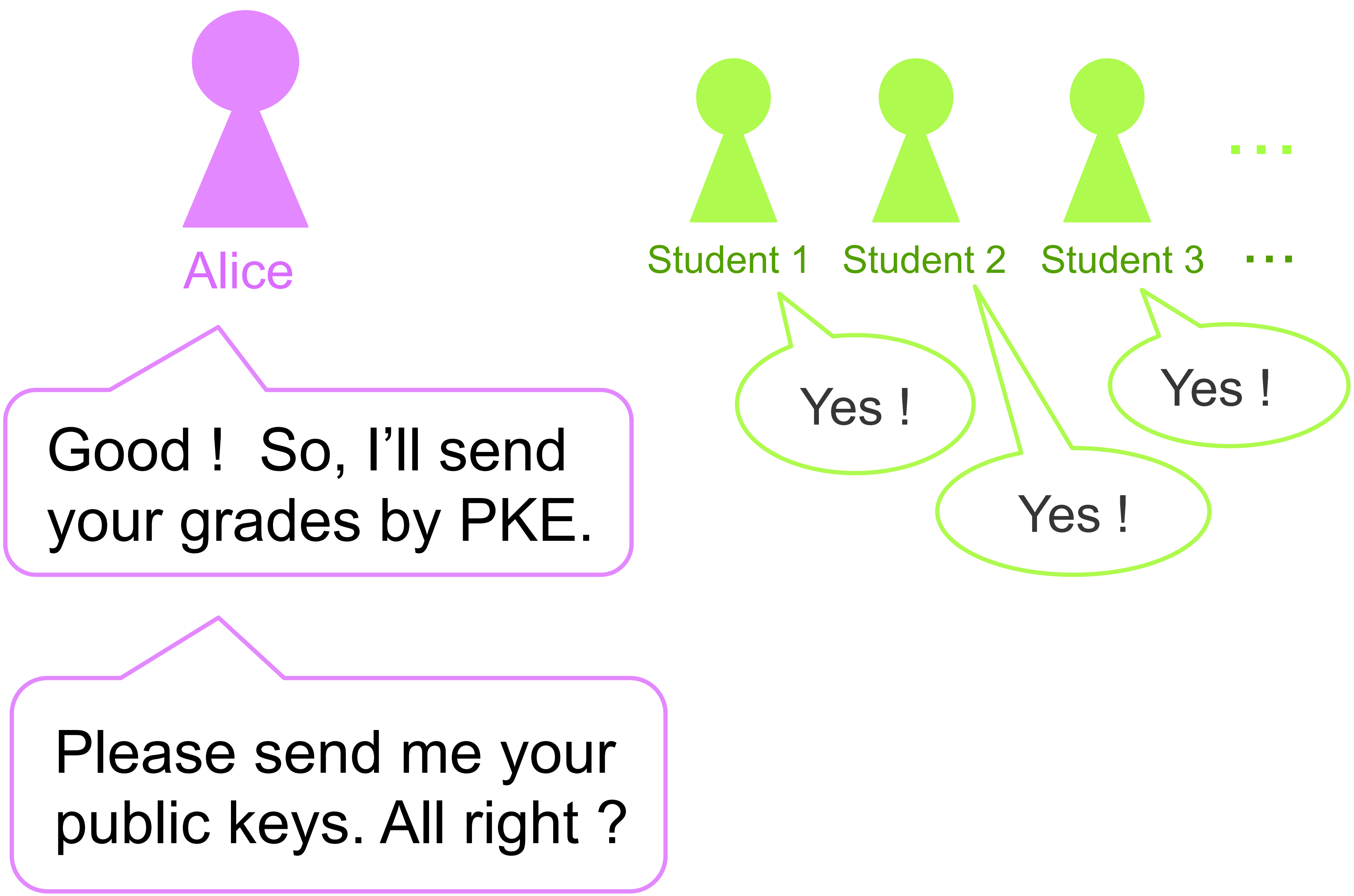
One day in a class,



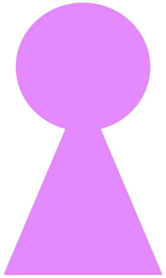
Good ! So, I'll send your grades by PKE.

Please send me your public keys. All right ?

One day in a class,

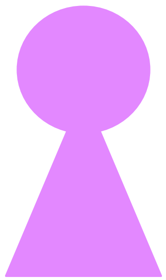


One day in a class,



Alice

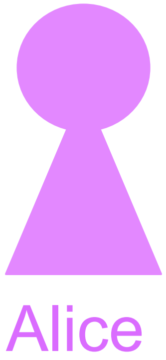
One day in a class,



Alice

Although I said so,
it is troublesome to encrypt all the grades.

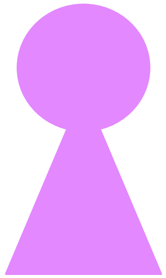
One day in a class,



Although I said so,
it is troublesome to encrypt all the grades.

But, since I promised to use PKE,
I have to do…

One day in a class,



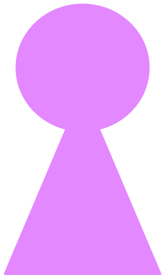
Alice

Although I said so,
it is troublesome to encrypt all the grades.

But, since I promised to use PKE,
I have to do…

What happened ?

What happened is …

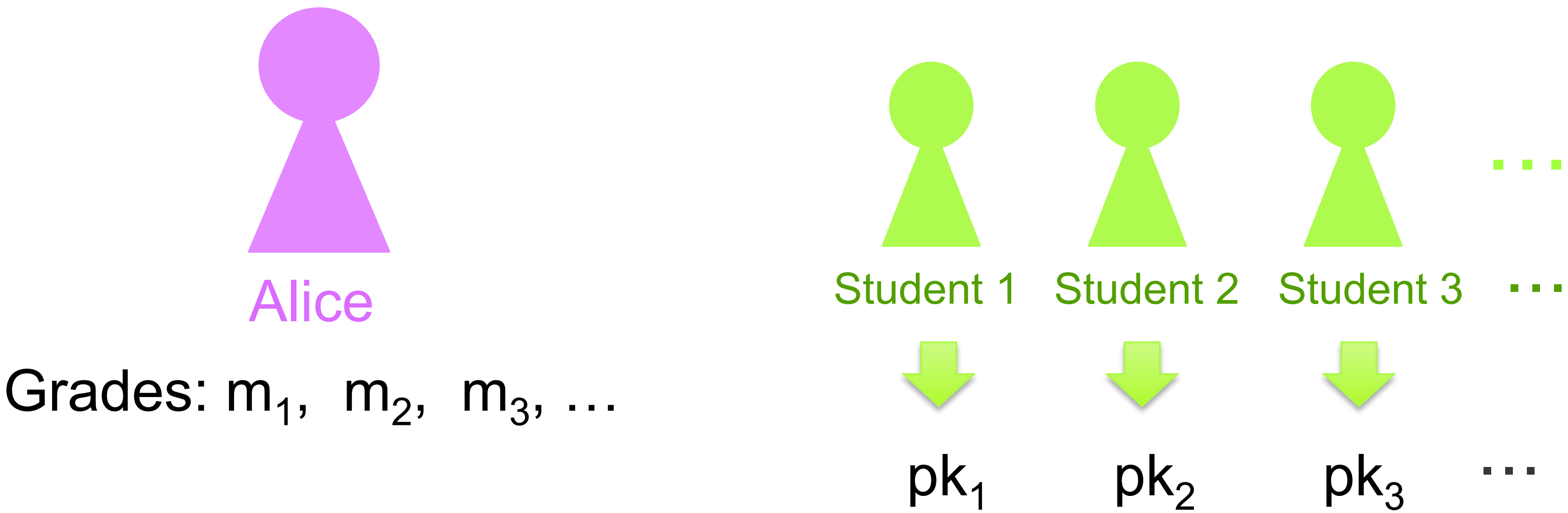


Alice

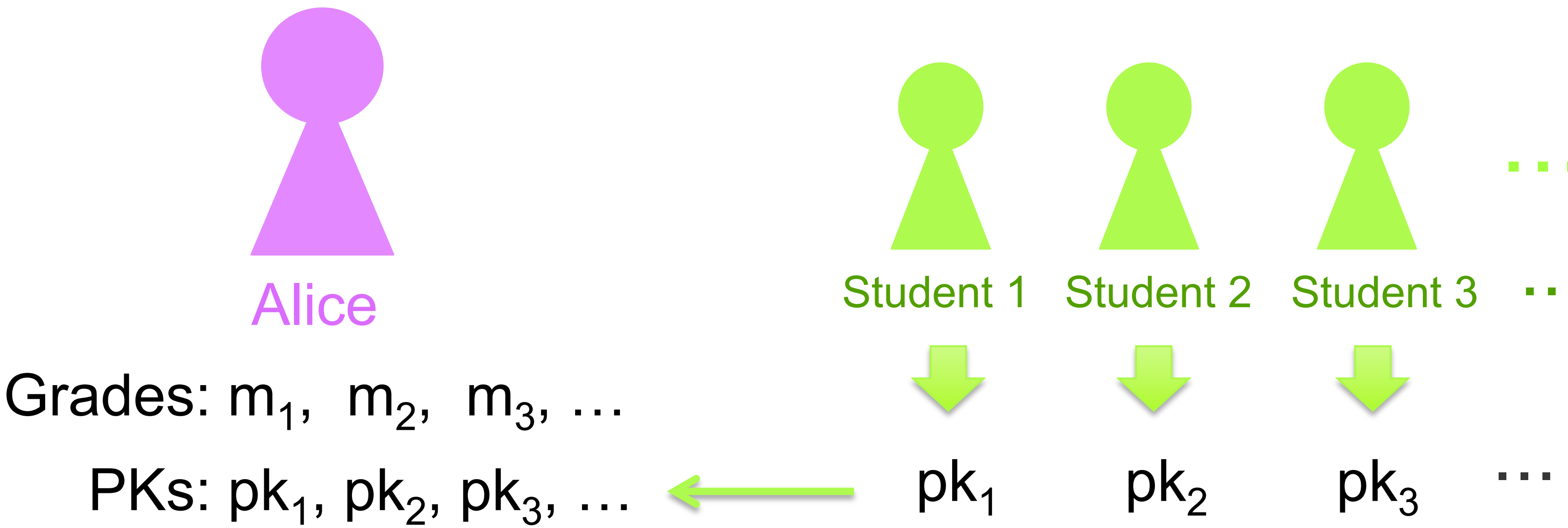
Student 1 Student 2 Student 3 …

Grades: m_1, m_2, m_3, …

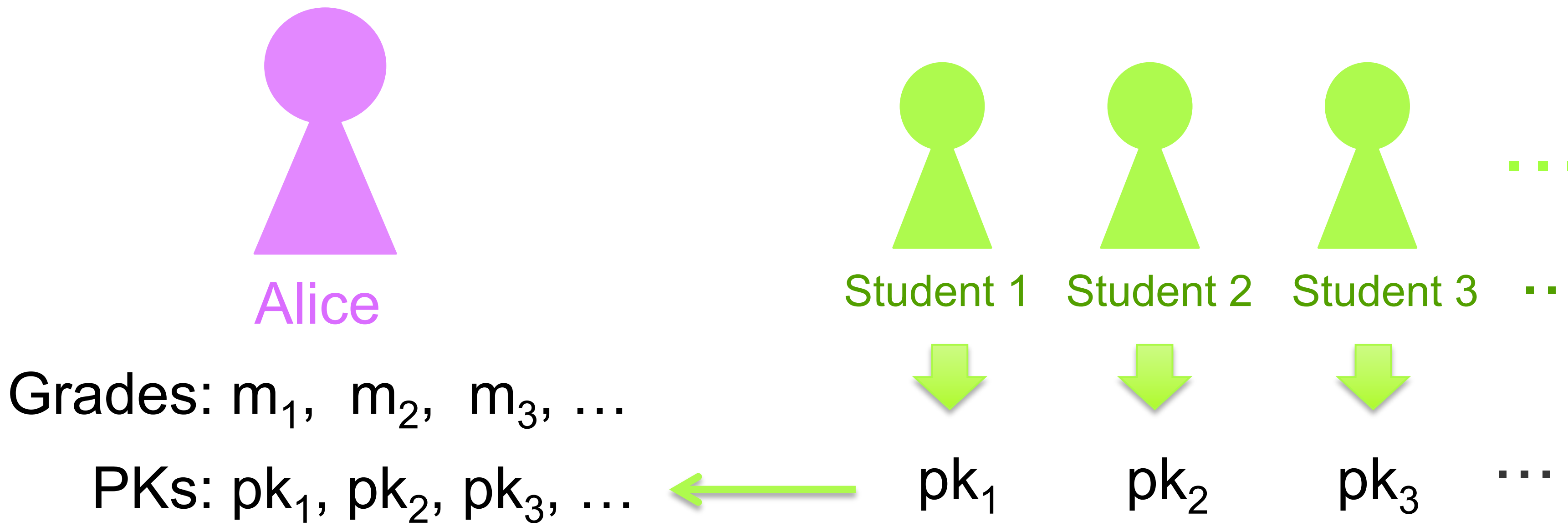
What happened is …



What happened is …

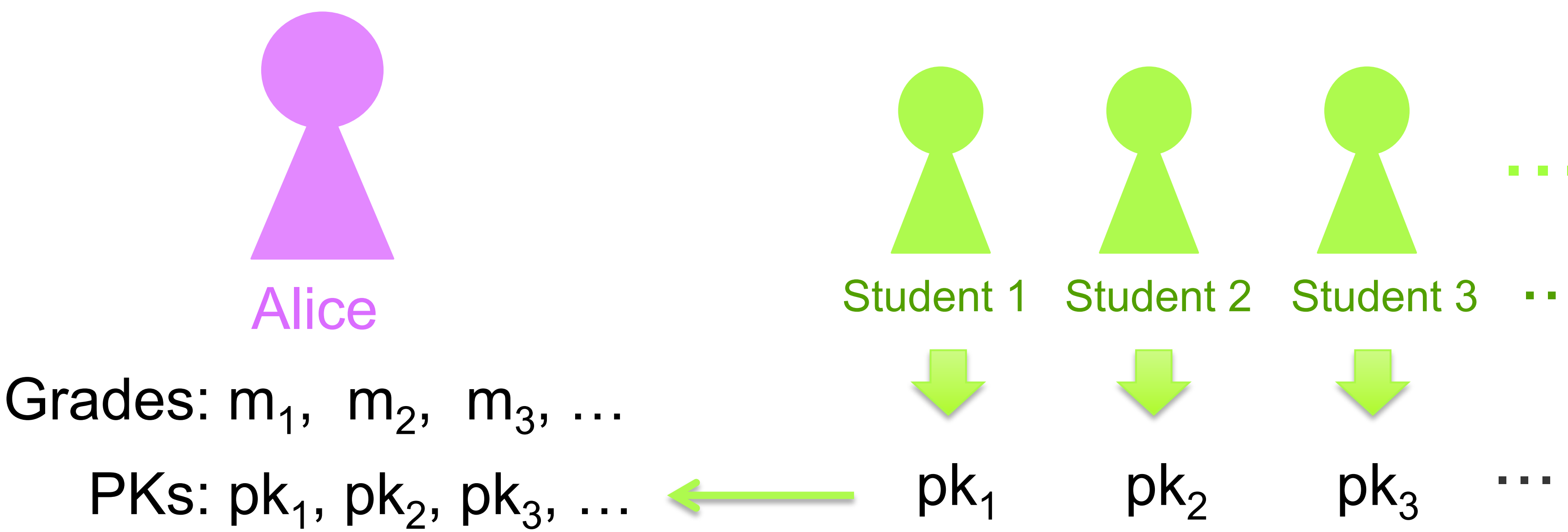


What happened is …



It's troublesome to encrypt honestly…

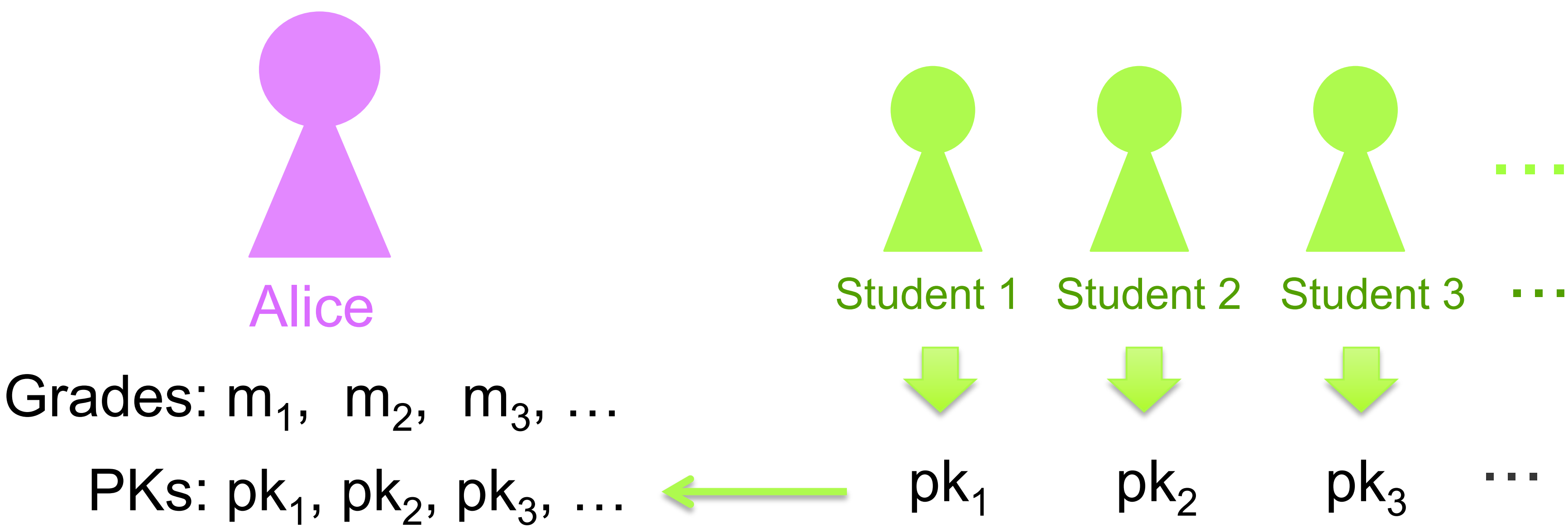
What happened is …



It's troublesome to encrypt honestly…

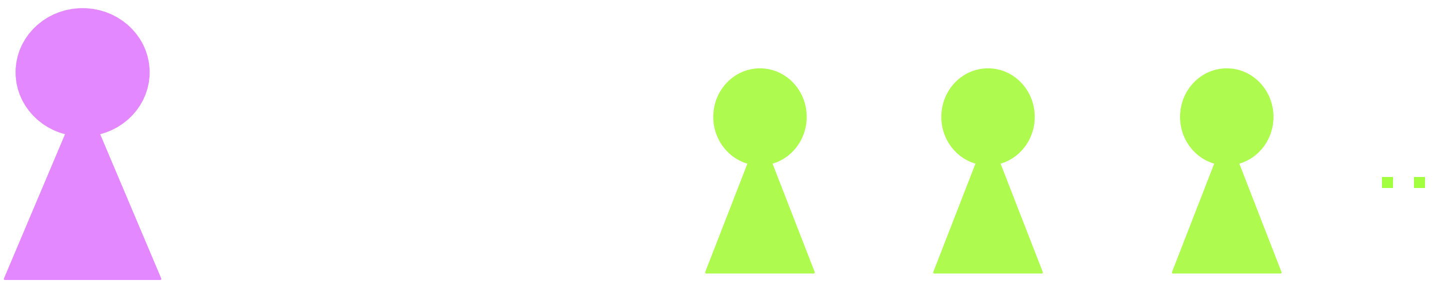
Wait !

What happened is …



The grades are personal information for *students.*
Their security is not my concern.

What happened is …



Alice

Student 1 Student 2 Student 3 ···

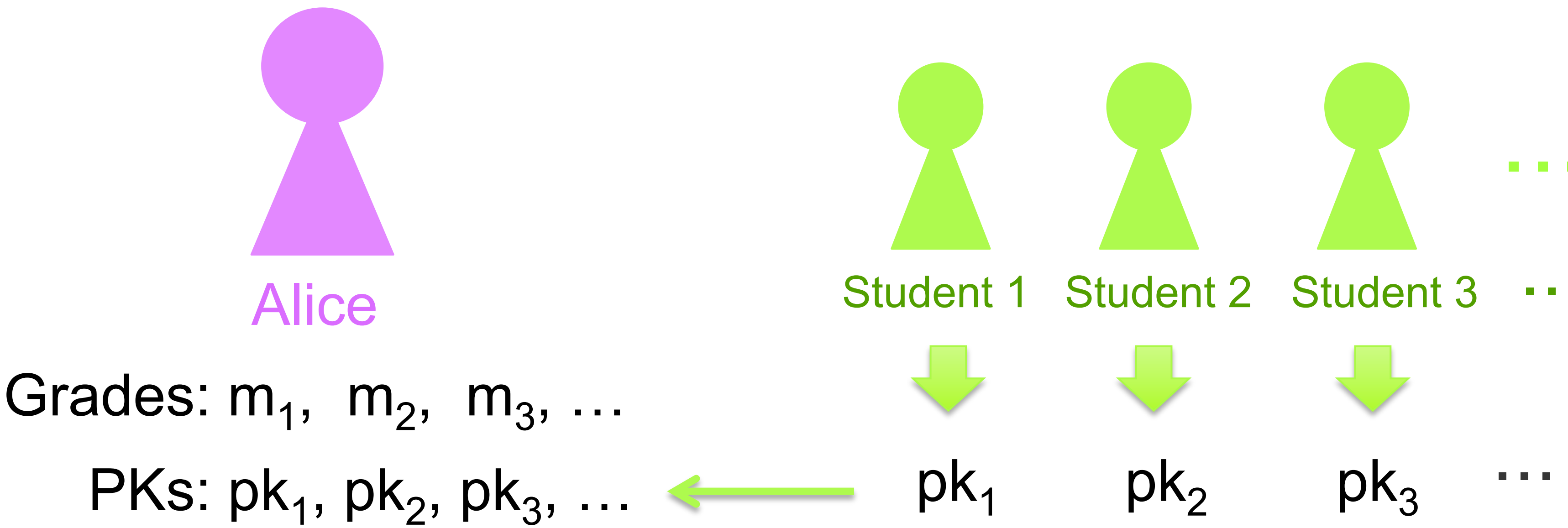
Grades: m_1, m_2, m_3, …

PKs: pk_1, pk_2, pk_3, … ← pk_1 pk_2 pk_3 ···

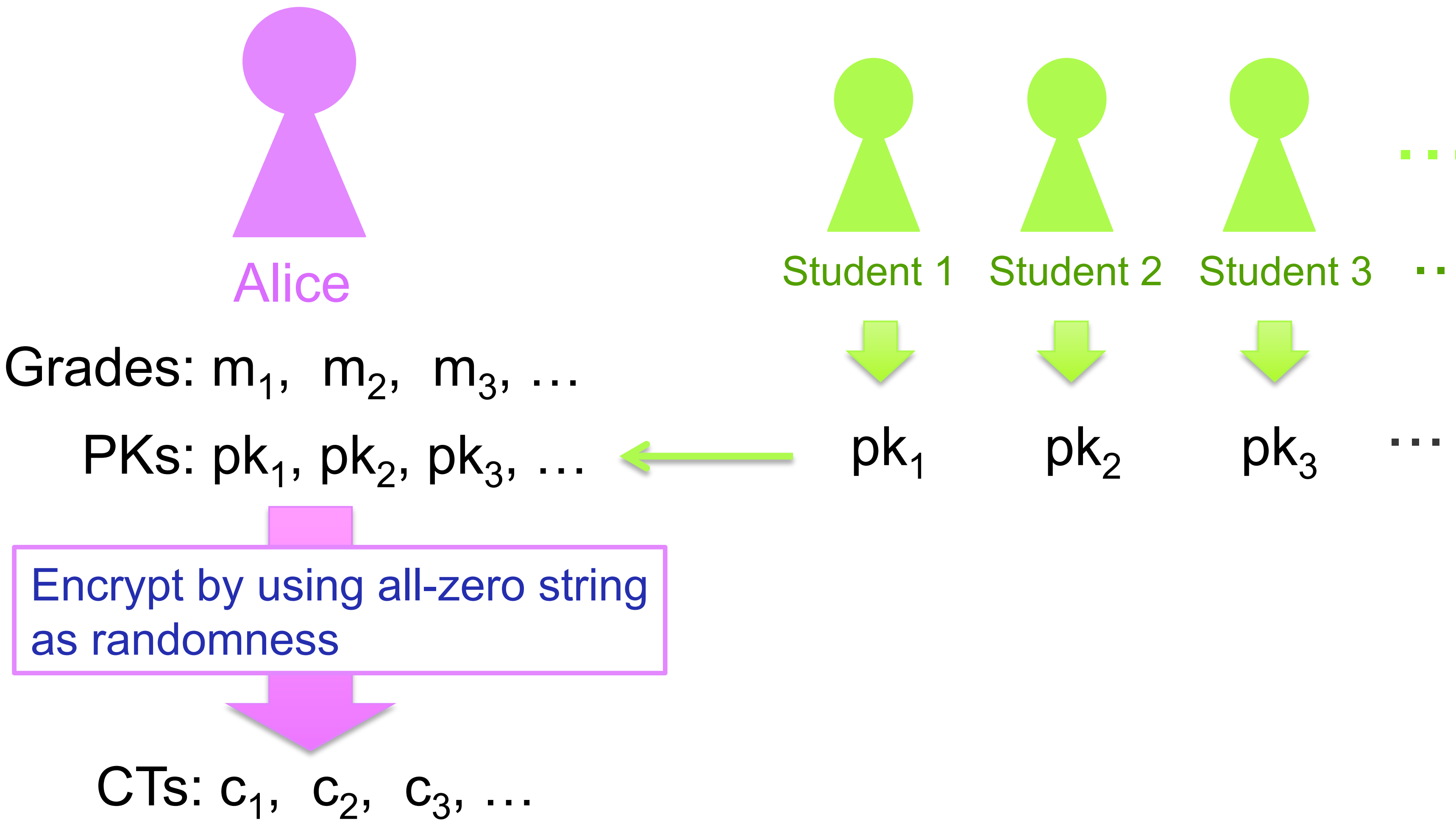
The grades are personal information for *students. Their security is not my concern.*

Want to cut corners…

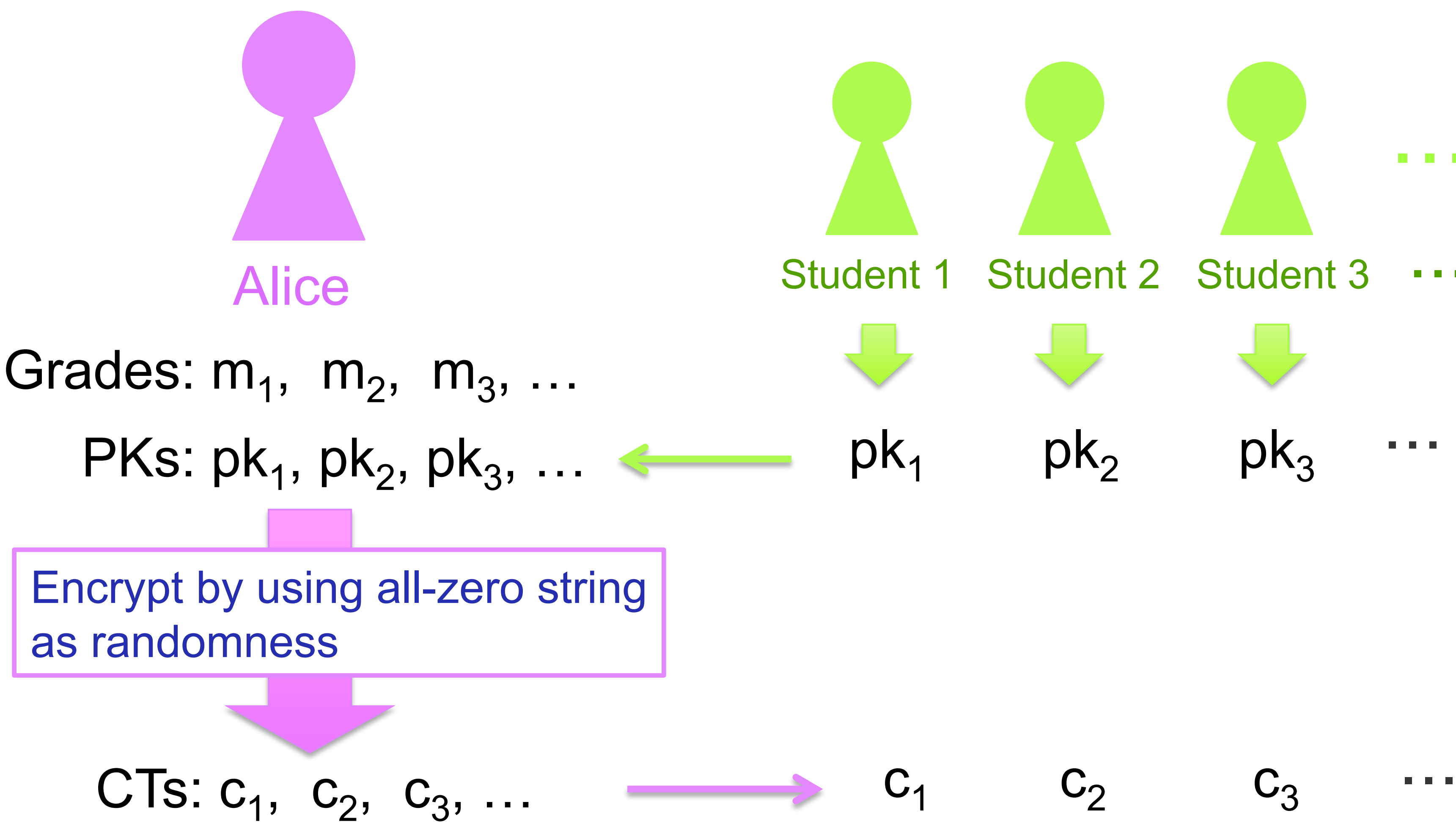
What happened is …



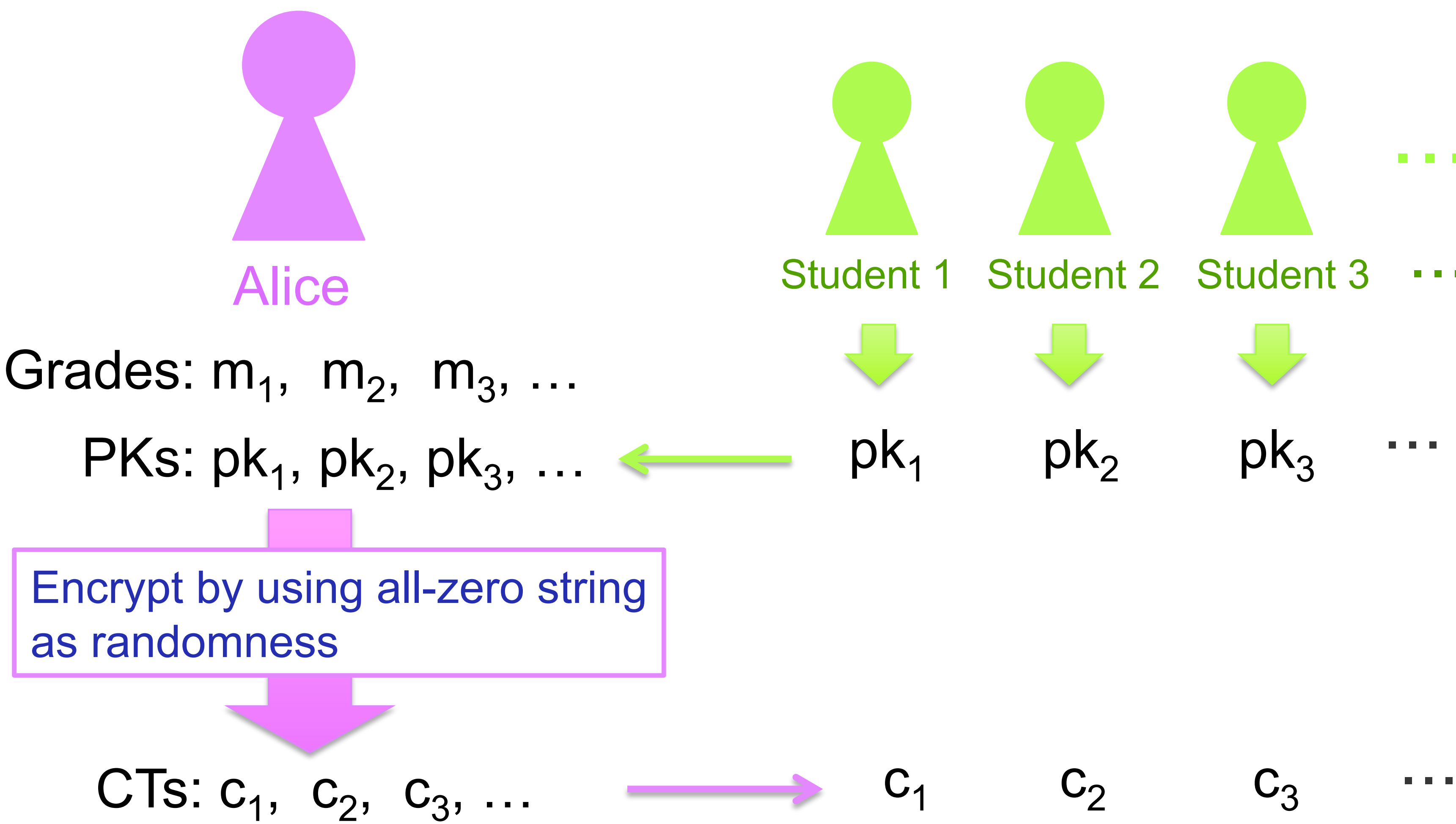
What happened is …



What happened is …



What happened is …



Grades were not sent securely !

The lesson from this example

The lesson from this example

- If some party in cryptographic protocols (PKE)
  1. is not concerned about the security
  2. is not willing to do a costly task (generating randomness)

  → The security can be compromised

The lesson from this example

- If some party in cryptographic protocols (PKE)
  1. is not concerned about the security
  2. is not willing to do a costly task (generating randomness)

  → The security can be compromised
- The reason is that Alice is "lazy"

The lesson from this example

- If some party in cryptographic protocols (PKE)
  1. is not concerned about the security
  2. is not willing to do a costly task (generating randomness)

  → The security can be compromised
- The reason is that Alice is "lazy"
- Traditional crypto did not consider lazy parties

The lesson from this example

- If some party in cryptographic protocols (PKE)
  1. is not concerned about the security
  2. is not willing to do a costly task (generating randomness)

  → The security can be compromised
- The reason is that Alice is "lazy"
- Traditional crypto did not consider lazy parties
- Many people tend to be lazy in the real life…
  → Need secure protocols even for lazy parties

Our results

- Define the security of PKE for lazy parties
  - Lazy parties as rational players

- Construct secure PKEs for lazy parties

Practical motivation

- Lazy parties is an example of protocols that may not work if players behave in their own interests
  - The problem of lazy parties reveals the motivation of using bad randomness in PKE

- Secure PKEs for lazy parties
- → Secure PKEs for which users have an incentive to use good randomness

Lazy parties in PKE

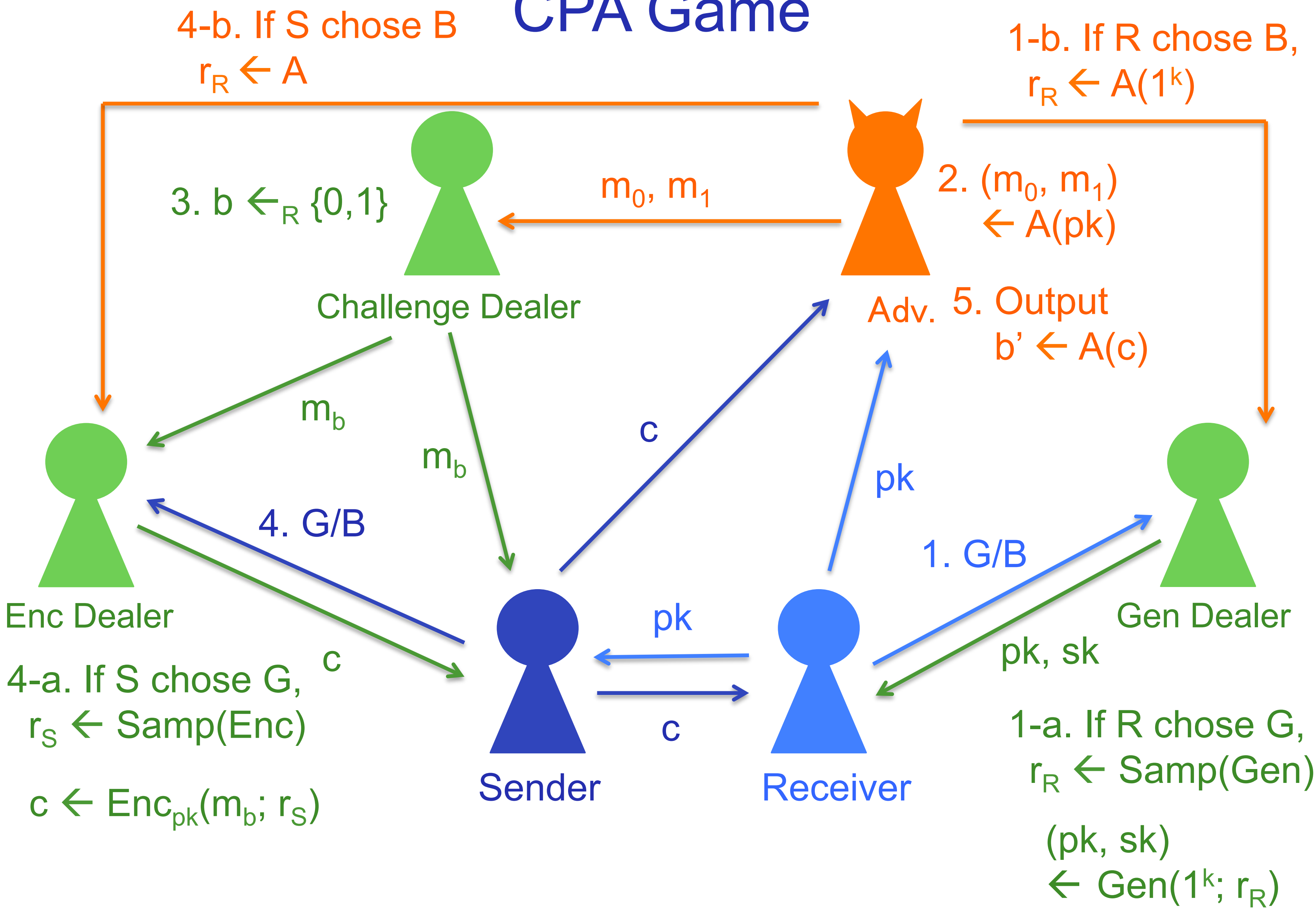
- Sender (S) and Receiver (R) are lazy
- Lazy S (and R)

  (1) wants to securely transmit msgs in M_S (and M_R)

  (2) doesn't want to generate costly randomness

  - Choose
    (a) Costly true randomness (Good randomness) or
    (b) Zero-cost fixed string (Bad randomness)
- Define a game between S, R, and an adversary (Adv)
  - A variant of usual CPA game
  - Lazy parties behave to maximize their payoffs
  - The goal is to design PKE secure for $m \in M_S \cup M_R$

CPA Game



4-b. If S chose B
$r_R \leftarrow A$
1-b. If R chose B,
$r_R \leftarrow A(1^k)$
3. $b \leftarrow_R \{0,1\}$
m_0, m_1
2. (m_0, m_1)
$\leftarrow A(pk)$
Challenge Dealer
Adv.
5. Output
$b' \leftarrow A(c)$
m_b
m_b
c
pk
4. G/B
1. G/B
Enc Dealer
Gen Dealer
pk
pk, sk
c
4-a. If S chose G,
$r_S \leftarrow Samp(Enc)$
$c \leftarrow Enc_{pk}(m_b; r_S)$
c
Sender
Receiver
1-a. If R chose G,
$r_R \leftarrow Samp(Gen)$
(pk, sk)
$\leftarrow Gen(1^k; r_R)$

Remarks on CPA Game

- We define the game more generally
  - Sender may run Gen algorithm
  - Encryption may be interactive
- Output of the Game:
  Out = (Win, Val_S, Val_R, Num_S, Num_R)
  - Win = 1 if b = b', 0 otherwise
  - $Val_w = 1$ if $m \in M_w$, 0 otherwise
  - Num_w : #{ G output by w : $w \in \{S, R\}$ }

Payoff function

- Payoff when the output of CPA game is Out = (Win, Val_S, Val_R, Num_S, Num_R)

  $$u_w(Out) = (-\alpha_w)\bullet Win\bullet Val_w + (-\beta_w)\bullet Num_w$$

  - α_w, $\beta_w > 0$ are real numbers
  - $\alpha_w /2 > q_w \bullet \beta_w$ is assumed. q_w : Maximum of Num_w
    - Costly good randomness is worth for achieving the security

- Payoff when following the pair of strategies (σ_S, σ_R)

  $$U_w(\sigma_S, \sigma_R) = \min E[u_w(Out)]$$

  - min is taken over all Advs, message spaces M_S, M_R

Security of PKE for lazy parties

- For PKE scheme Π, strategies (σ_S, σ_R), $(\Pi, \sigma_S, \sigma_R)$ is CPA secure with (strict) Nash equilibrium

1. If players follow (σ_S, σ_R), then for any adversary, message spaces M_S, M_R,

   $$\Pr[\mathrm{Win} \bullet (\mathrm{Val}_S + \mathrm{Val}_R) \neq 0] \leq 1/2 + \mathrm{negl}(k)$$

2. (σ_S, σ_R) is a (strict) Nash equilibrium

Solution concepts

- (σ_S, σ_R) is a Nash equilibrium :
  - For any $w \in \{S, R\}$ and σ_w',
    $U_w(\sigma_S^*, \sigma_R^*) \leq U_w(\sigma_S, \sigma_R) + negl(k)$
    where $(\sigma_S^*, \sigma_R^*) = (\sigma_S', \sigma_R)$ if $w = S$
    (σ_S, σ_R') otherwise

- (σ_S, σ_R) is a strict Nash equilibrium :
  1. (σ_S, σ_R) is a Nash equilibrium
  2. For any $w \in \{S, R\}$ and $\sigma_w' \neq \sigma_w$,
     $U_w(\sigma_S^*, \sigma_R^*) \leq U_w(\sigma_S, \sigma_R) - 1/k^c$
     where c is a constant

First observation (Impossibility results)

- Sender must generate a secret key
  - A game for distinguishing $m_0, m_1 \in M_R \setminus M_S$

    → S uses Bad randomness

    → Adv can correctly distinguish
    since Adv knows all the inputs to S except m_b

- Encryption must be interactive
  - A game for distinguishing (m_0, m_0) and (m_0, m_1) for $m_0, m_1 \in M_R \setminus M_S$

    → S uses Bad randomness

    → Adv can correctly distinguish
    if two msgs were encrypted by same randomness

Secure PKE for lazy parties (1. Basic setting)

- Two-round PKE Π_{two}
  - Idea: R generates randomness for encryption
    R follows since doesn't know whether $m \in M_R$

Sender | **Receiver**

Key Generation

Sender: $(pk_S, sk_S) \leftarrow Gen(1^k; r_1^S)$

Sender → Receiver: pk_S

Encryption

Receiver: $r_2^R \leftarrow_R U$

Receiver: $c_1 \leftarrow Enc(pk_S, r_2^R; r_3^R)$

Receiver → Sender: c_1

Sender: $r_2^R \leftarrow Dec(sk_S, c_1)$

Sender: $c_2 = m \oplus r_2^R$

Sender → Receiver: c_2

Receiver: $m = c_2 \oplus r_2^R$

Secure PKE for lazy parties (1. Basic setting)

- A problem of Π_{two} :

  If R knows that $m \notin M_R$, R uses Bad randomness

  ($m \in M_S \setminus M_R$ is not sent securely)

Secure PKE for lazy parties (2. R knows additional information)

- R may know whether $m \in M_R$

Secure PKE for lazy parties (2. R knows additional information)

- R may know whether $m \in M_R$

- Three-round PKE Π_{three}

  Idea:
  - Key agreement to share randomness
    - Shared randomness is Good if S or R uses Good
  - Use the shared randomness for encryption

Three-round PKE Π_{three}

Sender | **Receiver**

Key Generation

Receiver: $(pk_R, sk_R) \leftarrow Gen(1^k; r_1^R)$

Receiver → Sender: pk_R

Sender: $(pk_S, sk_S) \leftarrow Gen(1^k; r_1^S)$

Sender → Receiver: pk_S

Encryption

Receiver: $r_2^R \leftarrow_R U$

Receiver: $c_1 \leftarrow Enc(pk_S, r_2^R; r_3^R)$

Receiver → Sender: c_1

Sender: $r_2^R \leftarrow Dec(sk_S, c_1)$

Sender: $r_2^S \leftarrow_R U$

Sender: $r = r_2^R \oplus r_2^S \ (= r_L \circ r_R)$

Sender: $c_2 \leftarrow Enc(pk_R, r_2^S; r_3^S)$

Sender: $c_3 \leftarrow Enc(pk_R, m; r_L)$

Sender → Receiver: c_2, c_3

Receiver: $r_2^S \leftarrow Dec(sk_R, c_2)$

Receiver: $r = r_2^R \oplus r_2^S \ (= r_L \circ r_R)$

Receiver: $m \leftarrow Dec(sk_R, c_3)$

Receiver: $c_4 \leftarrow Enc(pk_S, m; r_R)$

Receiver → Sender: c_4

Non-interactive PKE for lazy parties

Non-interactive PKE for lazy parties

- Additional assumption:
  Players don’t want to reveal their secret keys

Non-interactive PKE for lazy parties

- Additional assumption:
  Players don't want to reveal their secret keys

- Singcryption scheme is secure for lazy parties if signing key (secret key) can be computed from ciphertext and randomness

  → S uses Good to avoid revealing secret key

Conclusions

- "Lazy parties" may compromise the security
  - An example of protocols that may not work if players behave in their own interests
- Our results
  - Define the security of PKE for lazy parties
  - Construct secure PKEs for lazy parties

Conclusions

- "Lazy parties" may compromise the security
  - An example of protocols that may not work if players behave in their own interests
- Our results
  - Define the security of PKE for lazy parties
  - Construct secure PKEs for lazy parties

Thank you

Lazy parties

(1) They are not concerned about the security
in a certain situation

(2) They are unwilling to do a costly task,
although they behave in an honest-looking way

- Costly task:

  Ex. random generation (computation is costly)
  increasing # rounds to finish (time is costly)

- Honest-looking behavior:

  Ex. using all-zero string as randomness

A problem of Π_{three}

- If both S and R knows that $m \in M_S \cap M_R$, it's difficult to determine which of S/R uses Good
  - Exits two different (strict) Nash strategies

A problem of Π_{three}

- If both S and R knows that m $\in M_S \cap M_R$,
  it's difficult to determine which of S/R uses Good
  - Exits two different (strict) Nash strategies

- Solution:
  R uses the all-zero string as randomness in Enc
  if R knows m $\in M_S \cap M_R$
  - All-zero string is a signal to R

The security proof of Π_{three}

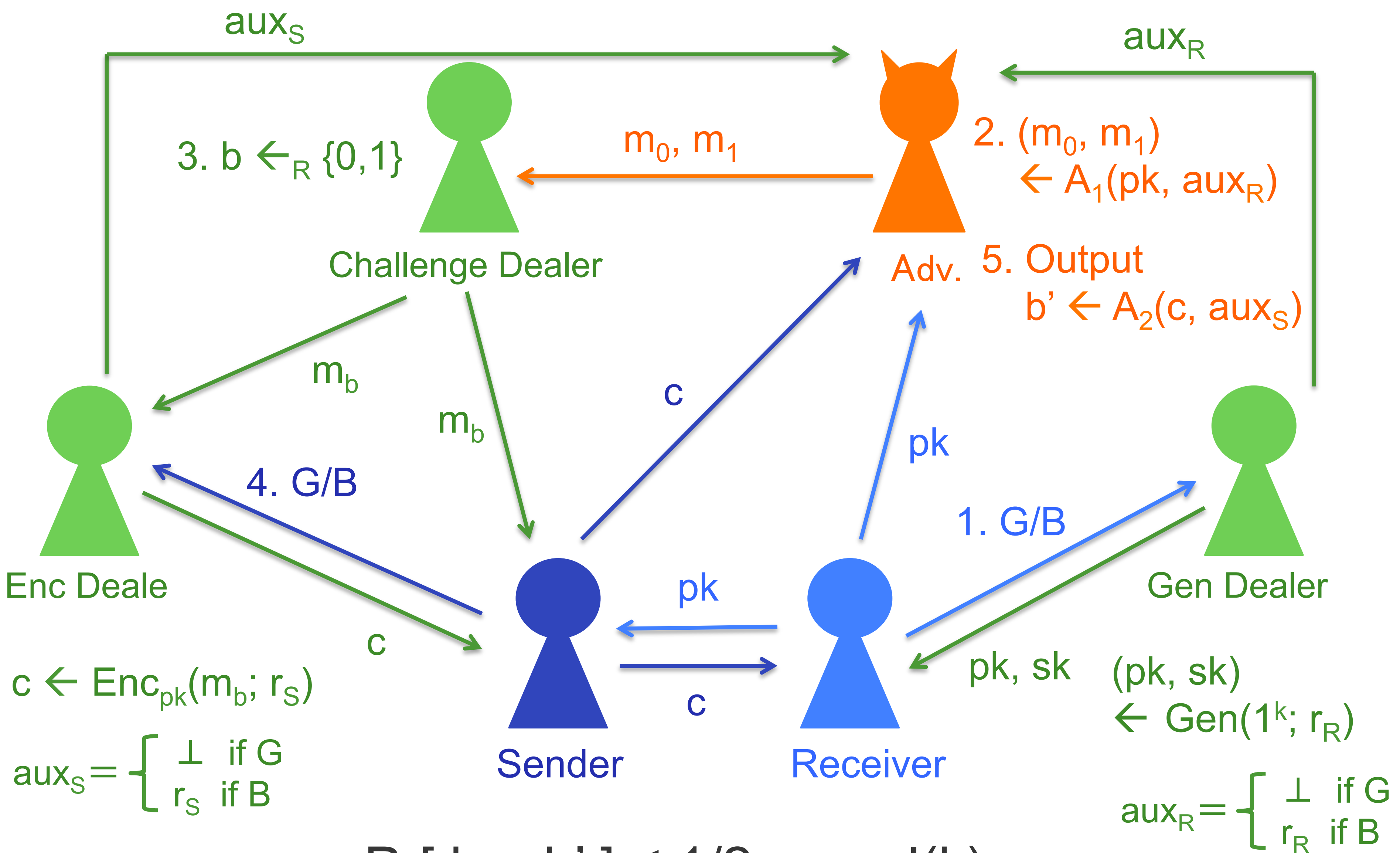
	Key Gen (S)	Key Gen (R)	Enc (S)	Enc (R)	Security
(a)	Good	Good	Good	-	✓
	Good	Good	-	Good	✓
(b)	Bad	-	-	-	No
(c)	-	Bad	-	-	No

(a) $r = r_2^R \oplus r_2^S$ is Good if S or R uses Good

(b) $m \in M_S \setminus M_R$ can be guessed from c_4

(c) $m \in M_R \setminus M_S$ can be guessed from c_3

CPA Game



$$\Pr[\, b = b' \,] \le 1/2 + \mathrm{negl}(k)$$

Impossibility results

- Proposition 1.
  If Sender does not have a secret key,
  then the scheme is not CPA secure with
  Nash equilibrium
  - A game for distinguishing $m_0, m_1 \in M_R \setminus M_S$

    → S uses Bad randomness

    → Adv can correctly distinguish
    since Adv knows all the inputs to S except m_b

→ Sender must generate a secret key