Randomness Leakage in the KEM/DEM Framework

Hitoshi Namiki (Ricoh)

Keisuke Tanaka (Tokyo Inst. of Tech.)

Kenji Yasunaga (Tokyo Inst. of Tech. → ISIT)

ProvSec 2011

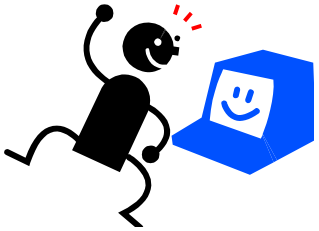
Leakage-Resilient Cryptography

- Prove the security even if some secret information leaks (by side-channel attacks)

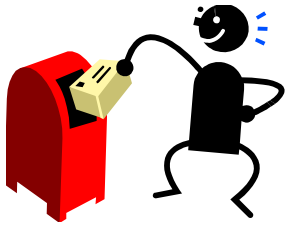
Leakage-Resilient Cryptography

- Prove the security even if some secret information leaks (by side-channel attacks)
  - Stream Cipher [DP08][Pie09]
  - Public-Key Encryption [AGV09][NS09] [ADW09][AND+10][BG10][DHL+10] …
  - Signature [ADW09][KV09][FKP10][MTV+11] [BSW11] …
  - etc.

Leakage-Resilient PKE



c

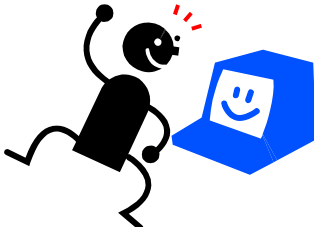


$m = Dec_{sk}(c)$

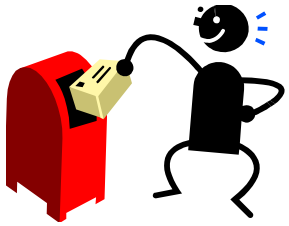
$c = Enc_{pk}(m;r)$

Leakage-Resilient PKE

- Leakage of secret key [AGV09][NS09][ADW09] …



c

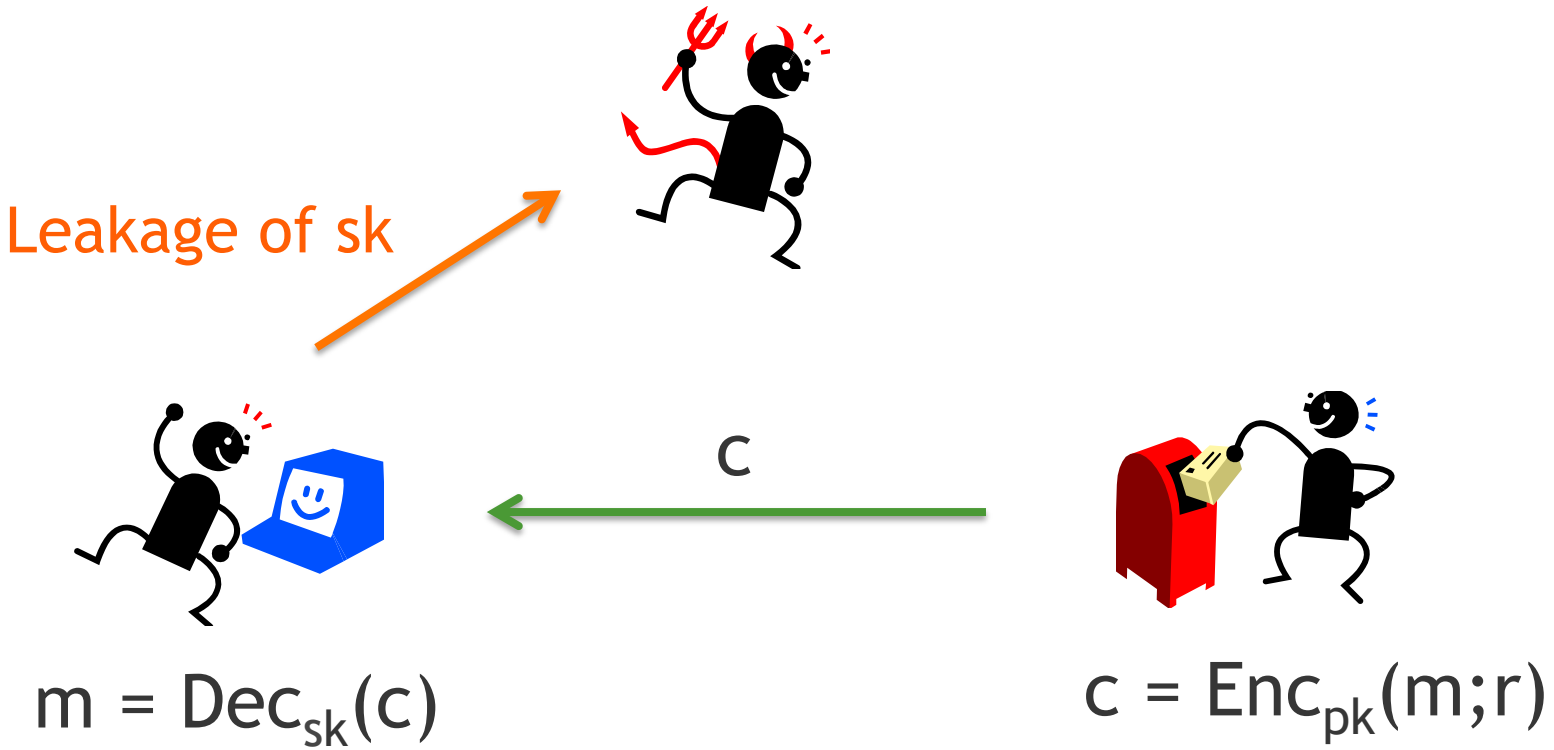


$m = Dec_{sk}(c)$

$c = Enc_{pk}(m;r)$

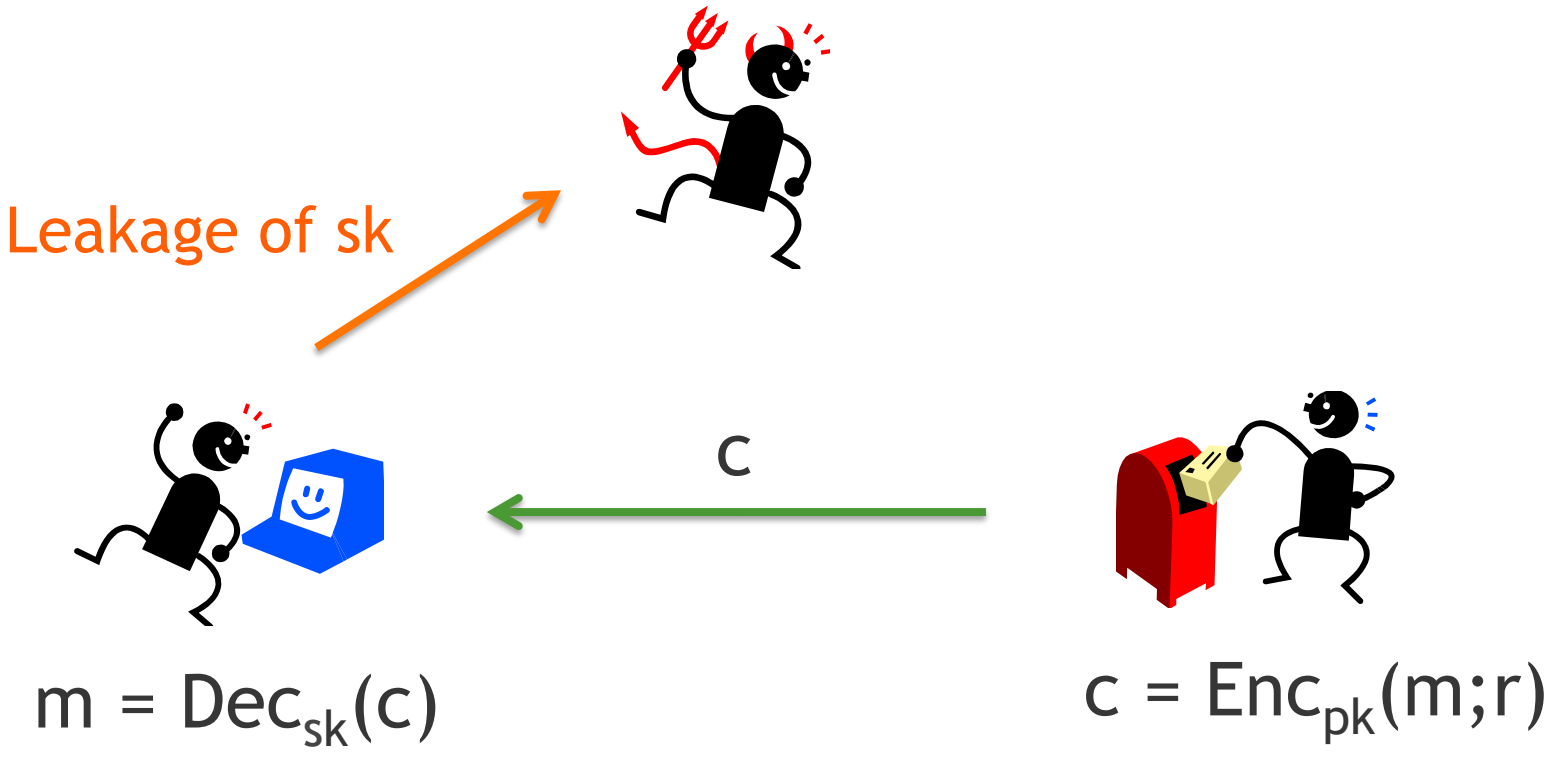
Leakage-Resilient PKE

- Leakage of secret key [AGV09][NS09][ADW09] …



Leakage-Resilient PKE

- Leakage of secret key [AGV09][NS09][ADW09] …
  - Restriction: Amount of leakage is bounded



This Work

This Work

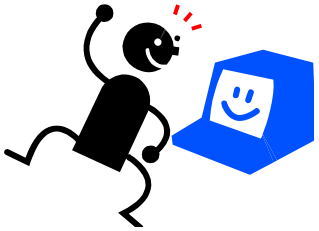
- Leakage of randomness in encryption

This Work

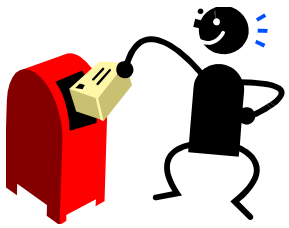
- Leakage of randomness in encryption
  - Restriction: Amount of leakage is bounded

This Work

- Leakage of randomness in encryption
  - Restriction: Amount of leakage is bounded



c



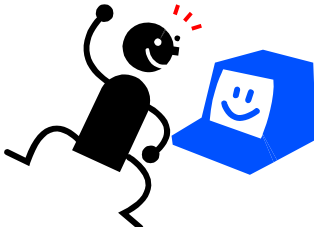
$m = Dec_{sk}(c)$

$c = Enc_{pk}(m;r)$

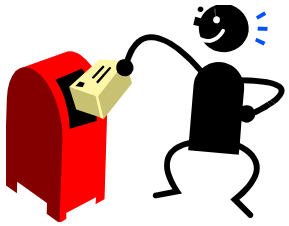
This Work

- Leakage of randomness in encryption
  - Restriction: Amount of leakage is bounded

Leakage of r



c



$m = Dec_{sk}(c)$

$c = Enc_{pk}(m;r)$

Our Results

Our Results

- No secure randomness-LR scheme
  if leaks after public key is published

Our Results

- No secure randomness-LR scheme if leaks after public key is published
  - Even if 1-bit leaks

Our Results

- No secure randomness-LR scheme if leaks after public key is published
  - Even if 1-bit leaks
  - Contrast to key-LR scheme (Secure schemes [AGV09][NS09]…)

Our Results

- No secure randomness-LR scheme if leaks after public key is published
  - Even if 1-bit leaks
  - Contrast to key-LR scheme (Secure schemes [AGV09][NS09]…)
- Secure randomness-LR KEM/DEM scheme even if leaks after public key is published

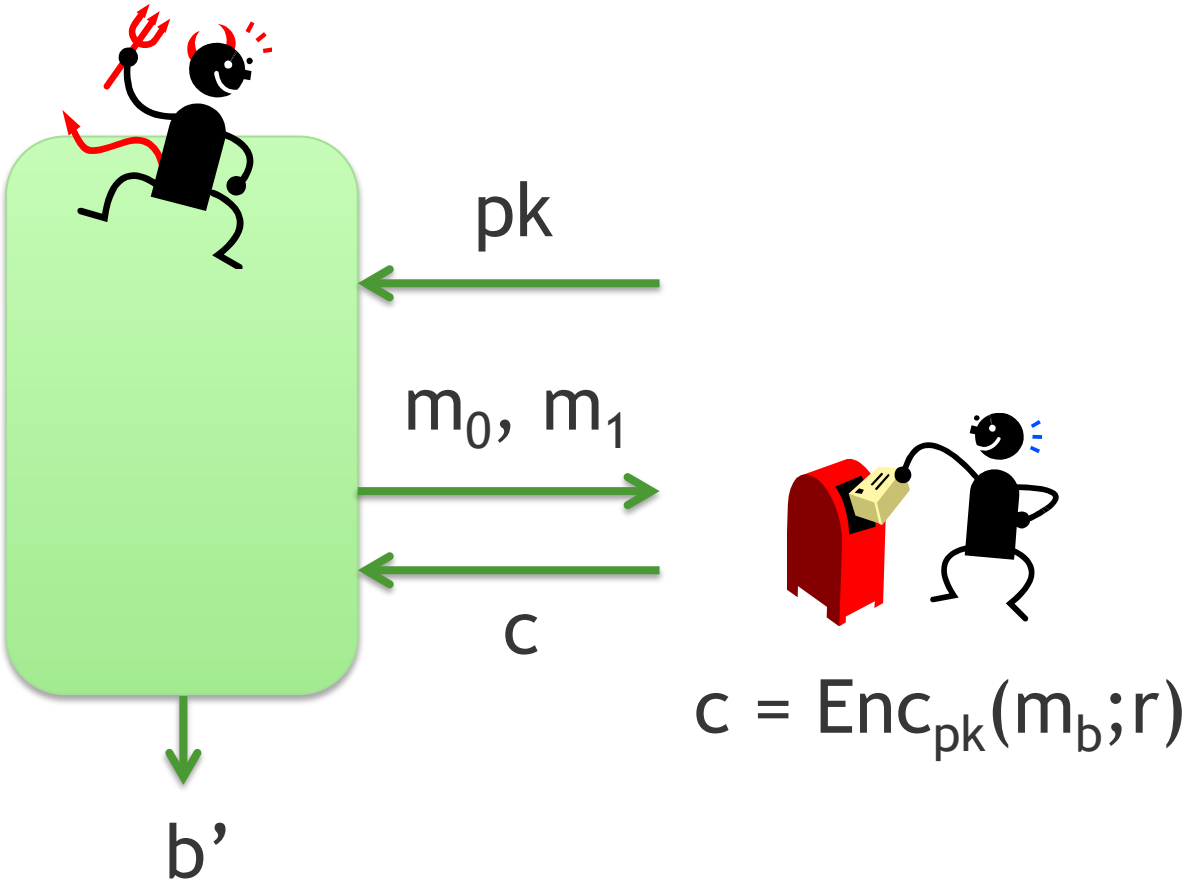
Our Results

- No secure randomness-LR scheme if leaks after public key is published
  - Even if 1-bit leaks
  - Contrast to key-LR scheme (Secure schemes [AGV09][NS09]…)
- Secure randomness-LR KEM/DEM scheme even if leaks after public key is published
  - Relax the leakage model (describe later)

Our Results

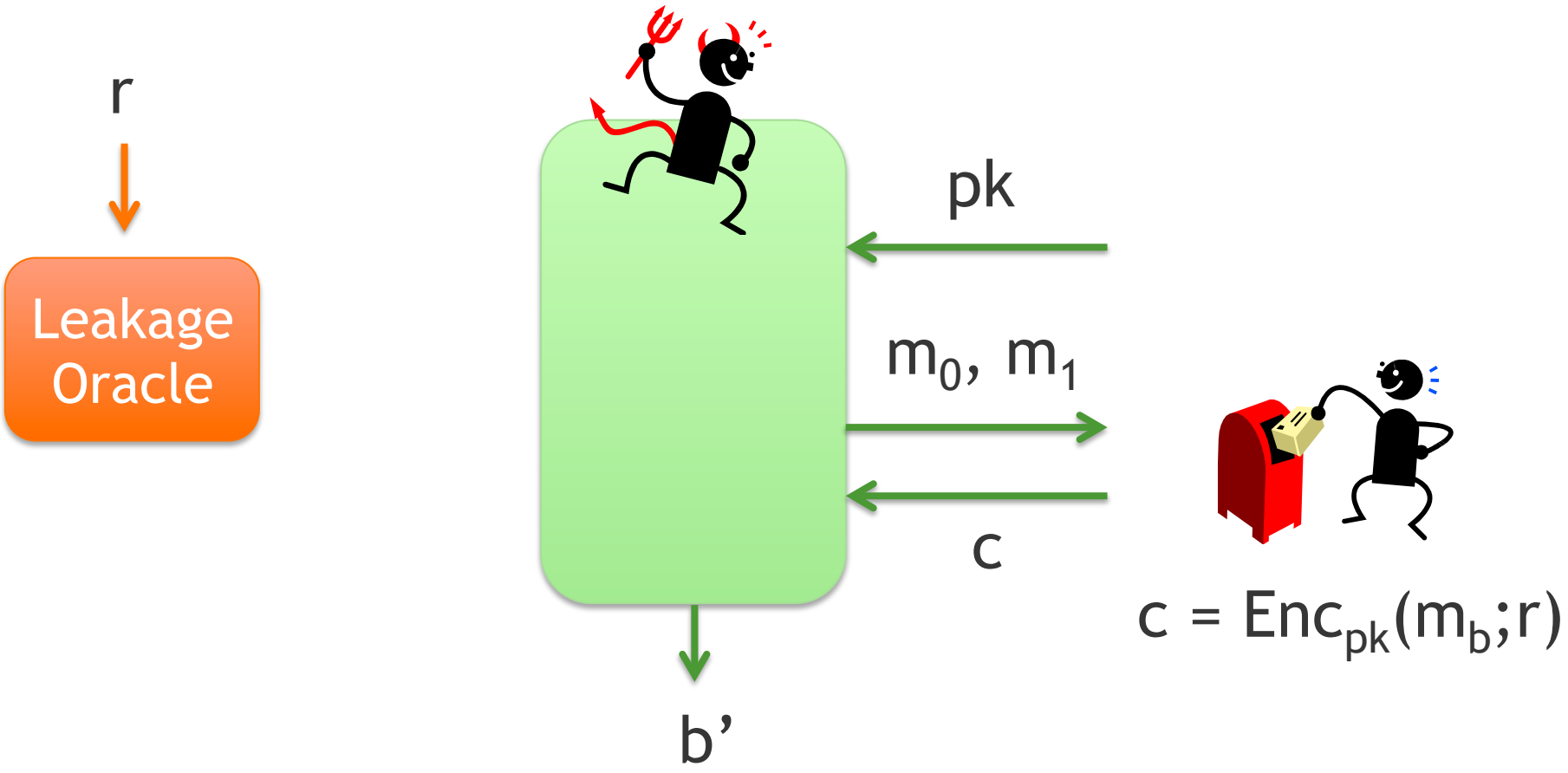
- No secure randomness-LR scheme if leaks after public key is published
  - Even if 1-bit leaks
  - Contrast to key-LR scheme (Secure schemes [AGV09][NS09]…)
- Secure randomness-LR KEM/DEM scheme even if leaks after public key is published
  - Relax the leakage model (describe later)
  - Leakage rate = 1 – o(1) (DDH assumption)

Model of Randomness Leakage



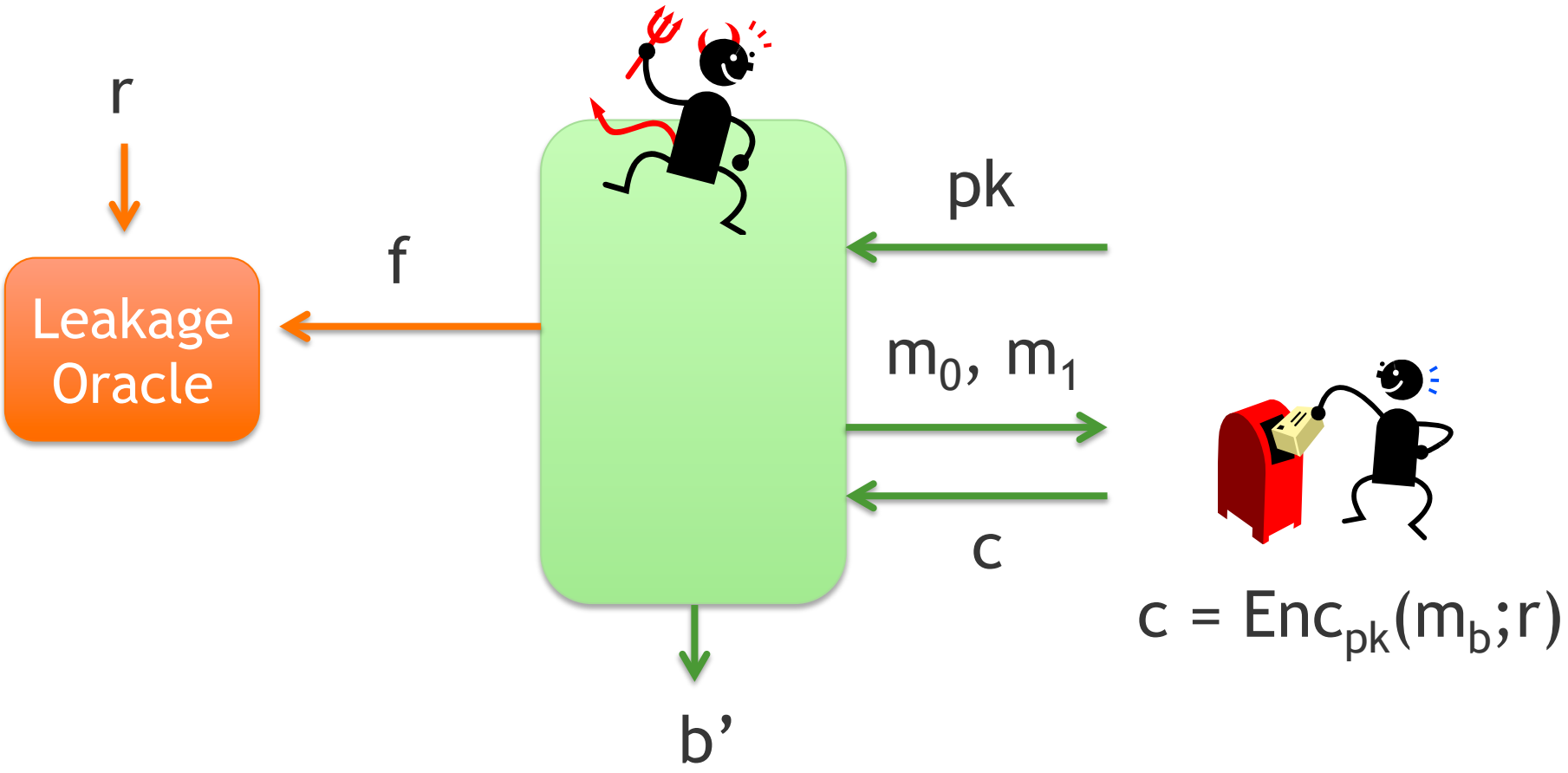
$$Pr[\ b' = b\ ] \leq 1/2 + negl(n)$$

Model of Randomness Leakage



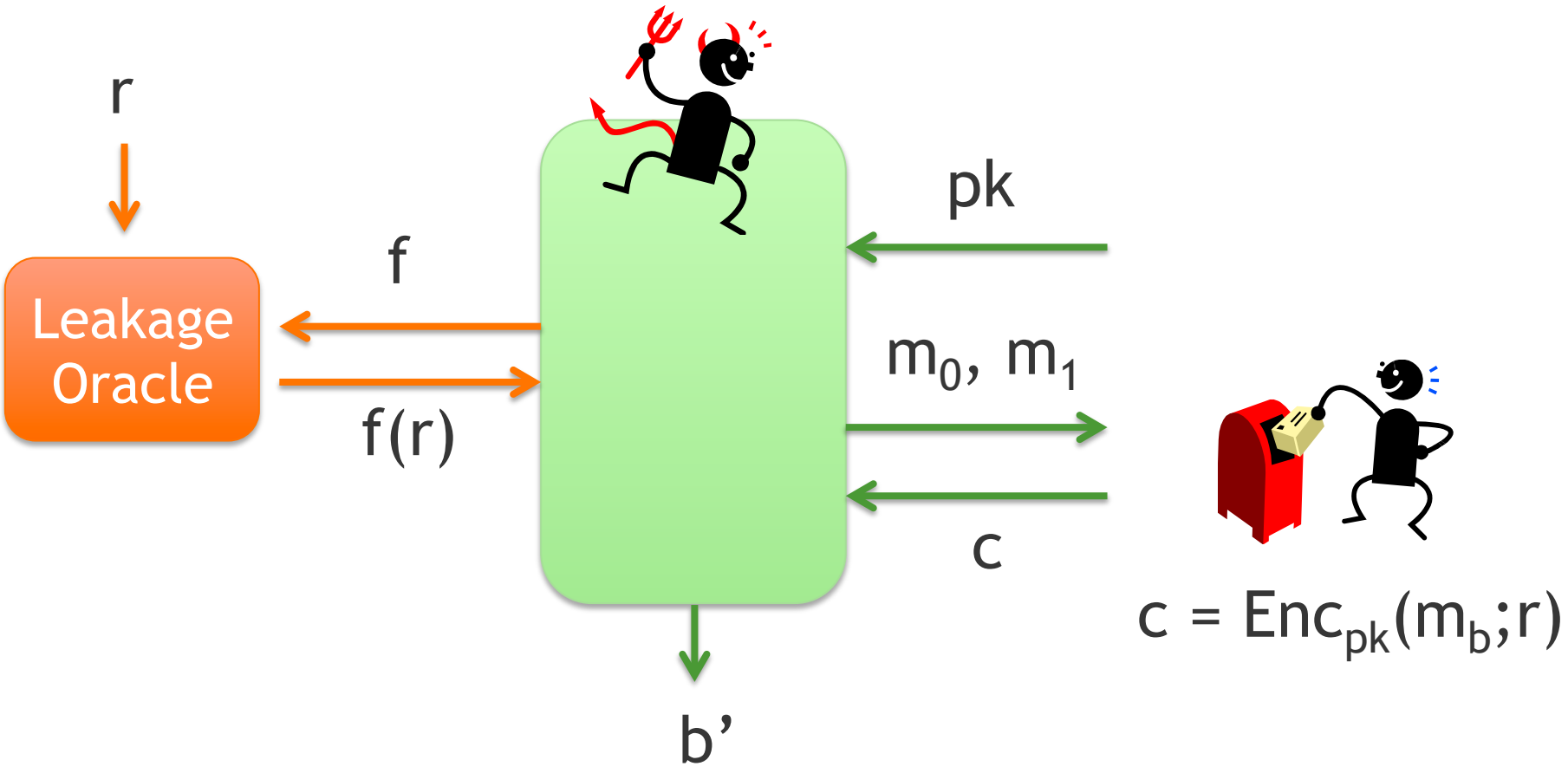
$$Pr[\ b' = b\ ] \leq 1/2 + negl(n)$$

Model of Randomness Leakage



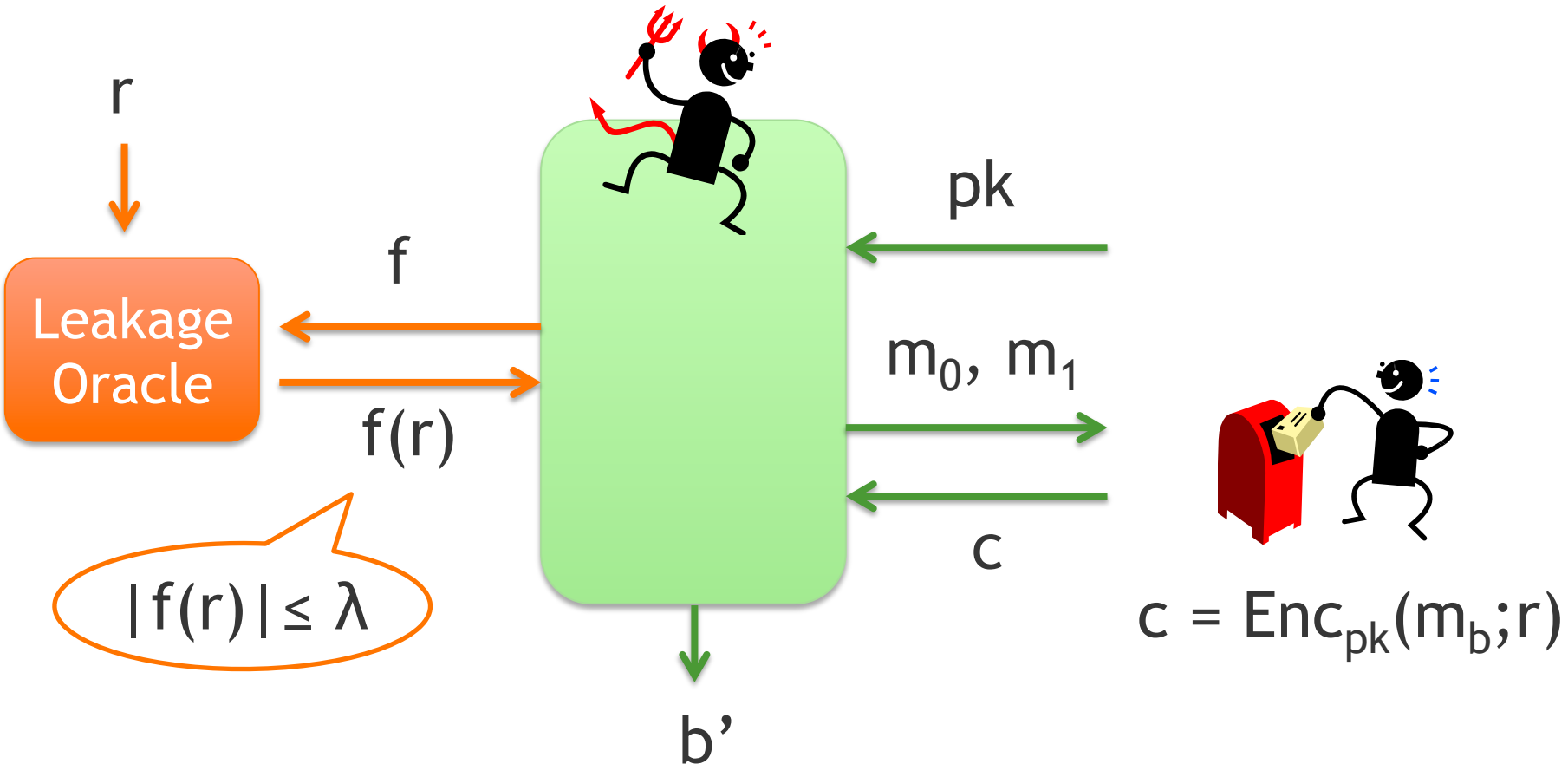
$$Pr[\ b' = b\ ] \leq 1/2 + negl(n)$$

Model of Randomness Leakage



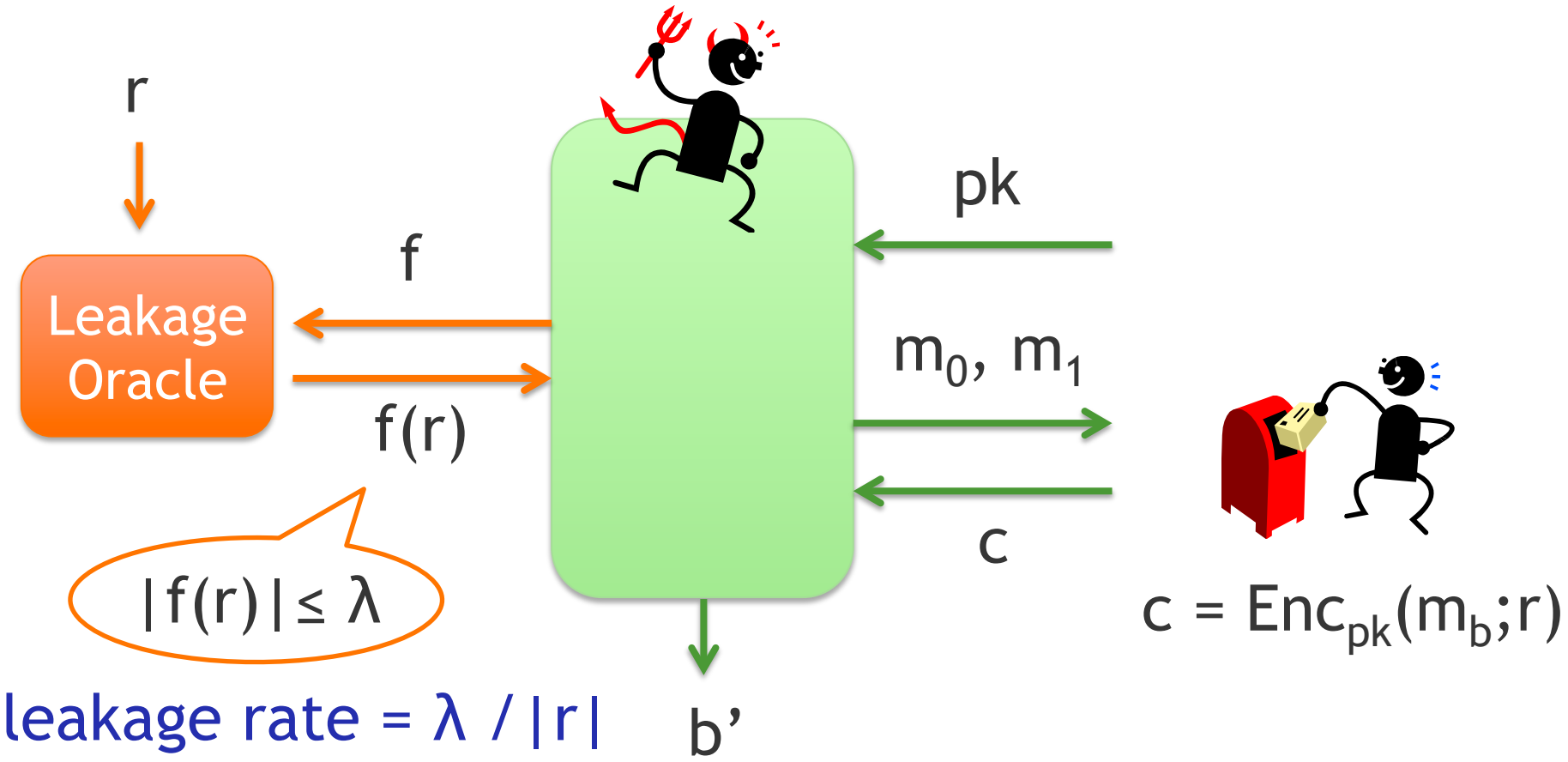
$$Pr[\ b' = b\ ] \leq 1/2 + negl(n)$$

Model of Randomness Leakage



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

Model of Randomness Leakage



leakage rate = $\lambda / |r|$

$Pr[\ b' = b\ ] \leq 1/2 + negl(n)$

Randomness leakage after public key is published

Theorem.
No secure randomness-LR scheme exists
if randomness leaks after public key is published

Randomness leakage after public key is published

Theorem.
No secure randomness-LR scheme exists
if randomness leaks after public key is published

Proof:

- Adversary's strategy:
  - Set $f(r) := \{$ i-th bit of $\mathrm{Enc}_{pk}(m_0; r) \}$ for random i
  - If $f(r) \neq \{$ i-th bit of c $\}$, output 1, o.w. a random guess
- When b = 0, Pr[b = b'] = 1/2
- When b = 1, Pr[b = b'] $\geq 1/2 + 1/|c|$
  since $f(r) \neq \{$ i-th bit of c $\}$ w.p. at least $1/|c|$

Randomness leakage after public key is published

- Randomness leakage is more serious than key leakage !!
  - 1-bit leakage → insecurity
  - Secure key-LR scheme [AGV09][NS09]…

Randomness leakage after public key is published

- Randomness leakage is more serious than key leakage !!
  - 1-bit leakage → insecurity
  - Secure key-LR scheme [AGV09][NS09]…

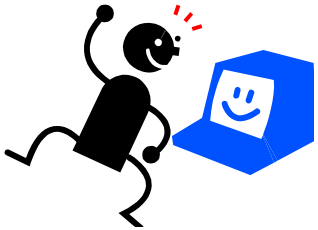
- Relax the leakage model
  - Fit for KEM/DEM framework

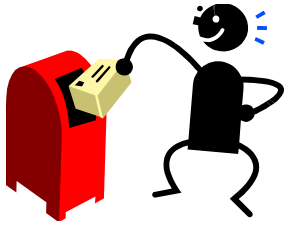
KEM/DEM Framework

- KEM ≈ PKE for random messages
  - Random message is used as secret key of DEM

KEM/DEM Framework

- KEM ≈ PKE for random messages
  - Random message is used as secret key of DEM



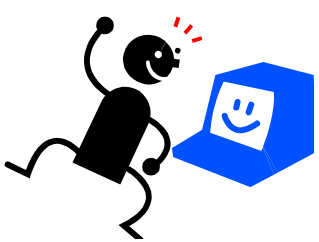


$(c_1, K) = KEM.Enc_{pk}(r_1)$

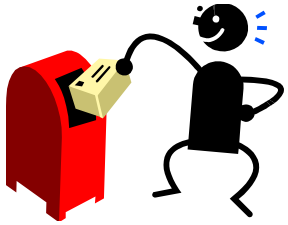
$c_2 = DEM.Enc_K(m;r_2)$

KEM/DEM Framework

- KEM ≈ PKE for random messages
  - Random message is used as secret key of DEM



c_1, c_2

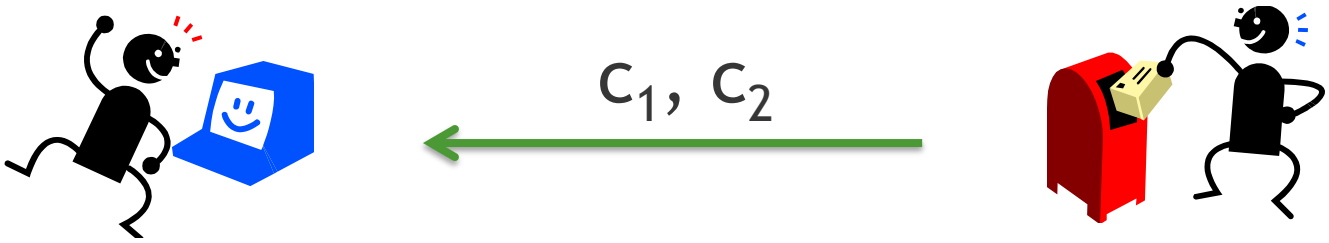


$(c_1, K) = KEM.Enc_{pk}(r_1)$

$c_2 = DEM.Enc_K(m;r_2)$

KEM/DEM Framework

- KEM ≈ PKE for random messages
  - Random message is used as secret key of DEM



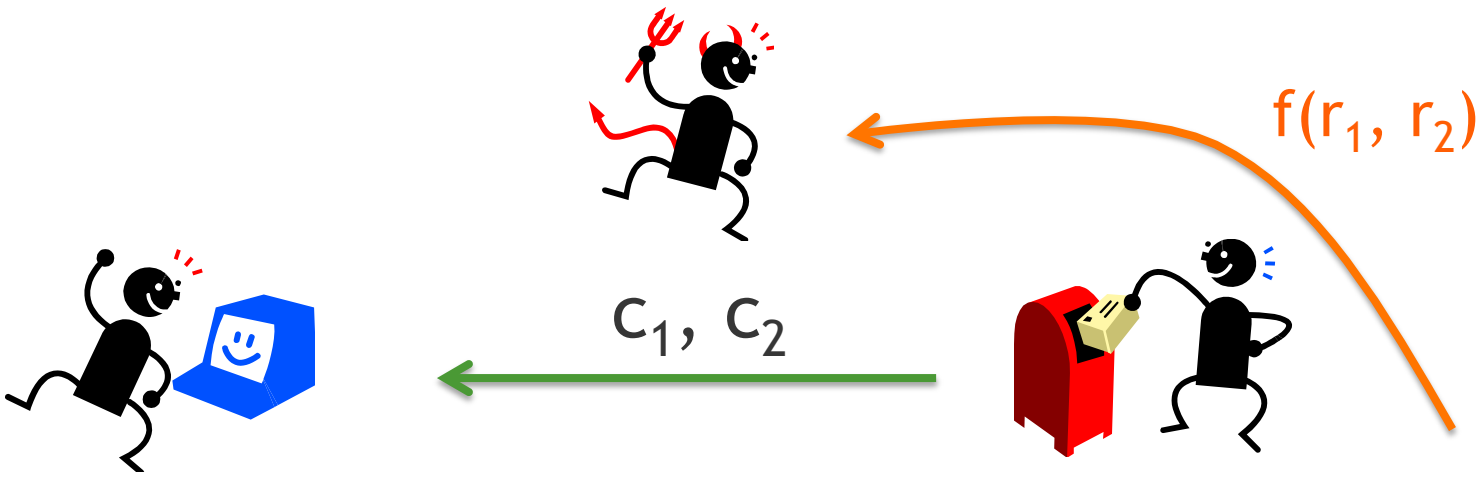
$K = KEM.Dec_{sk}(c_1)$

$m = DEM.Dec_K(c_2)$

$(c_1, K) = KEM.Enc_{pk}(r_1)$

$c_2 = DEM.Enc_K(m;r_2)$

Randomness Leakage in KEM/DEM



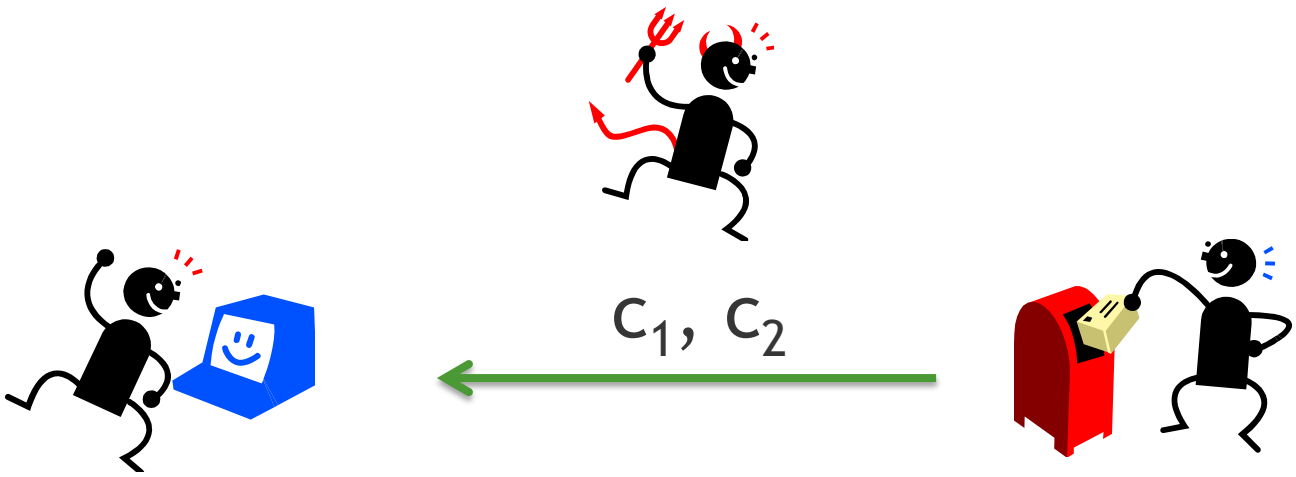
$K = KEM.Dec_{sk}(c_1)$

$m = DEM.Dec_K(c_2)$

$(c_1, K) = KEM.Enc_{pk}(r_1)$

$c_2 = DEM.Enc_K(m; r_2)$

Randomness Leakage in KEM/DEM



$K = KEM.Dec_{sk}(c_1)$

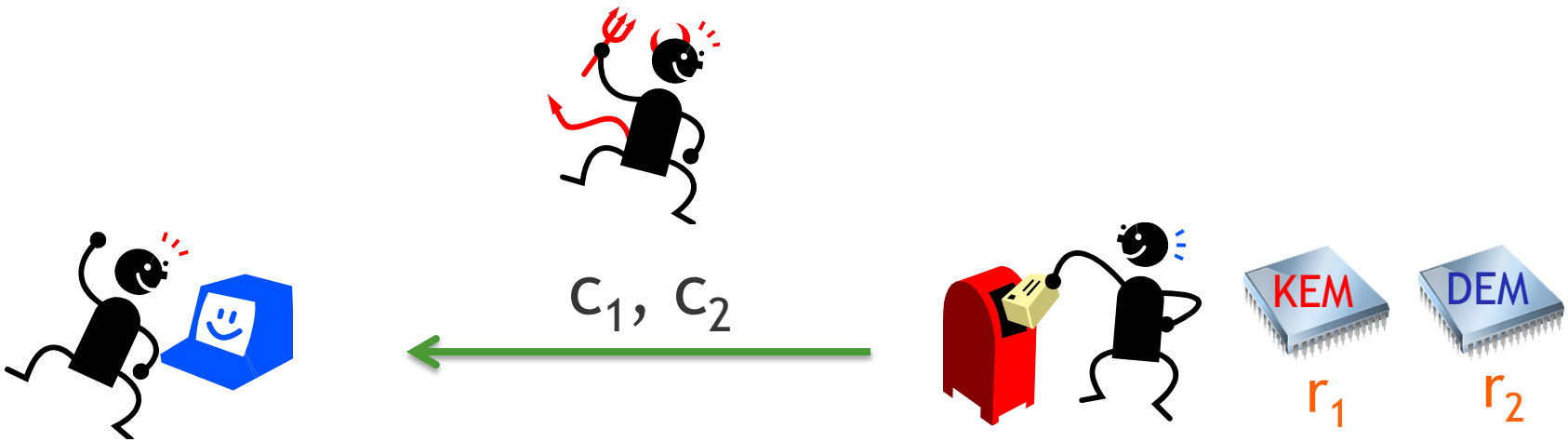
$m = DEM.Dec_K(c_2)$

$(c_1, K) = KEM.Enc_{pk}(r_1)$

$c_2 = DEM.Enc_K(m; r_2)$

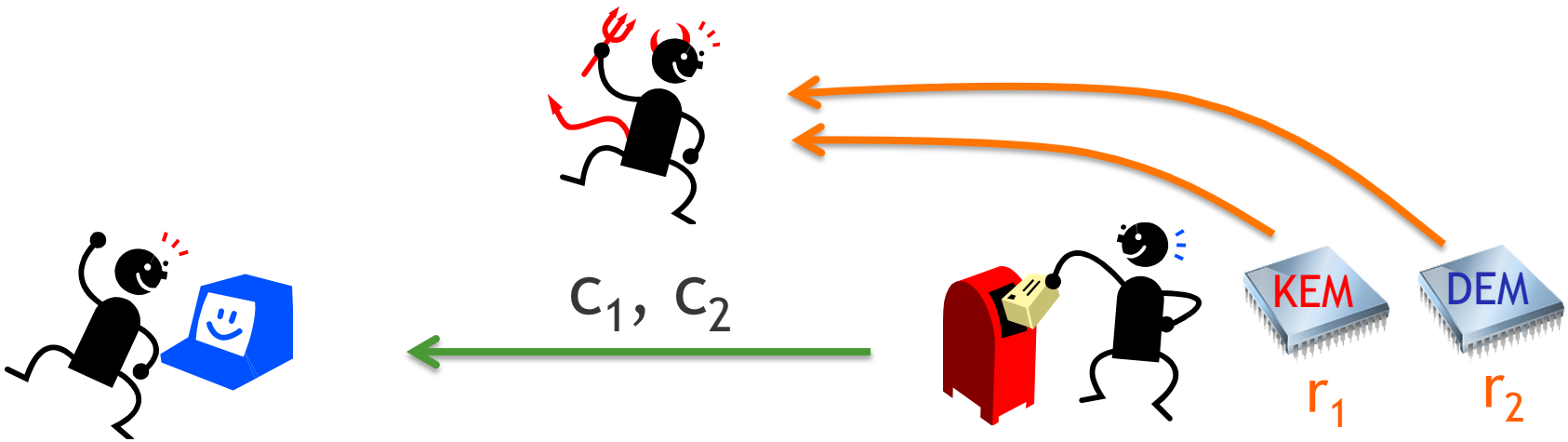
- Relaxation: Rand. for KEM/DEM leaks independently

Randomness Leakage in KEM/DEM



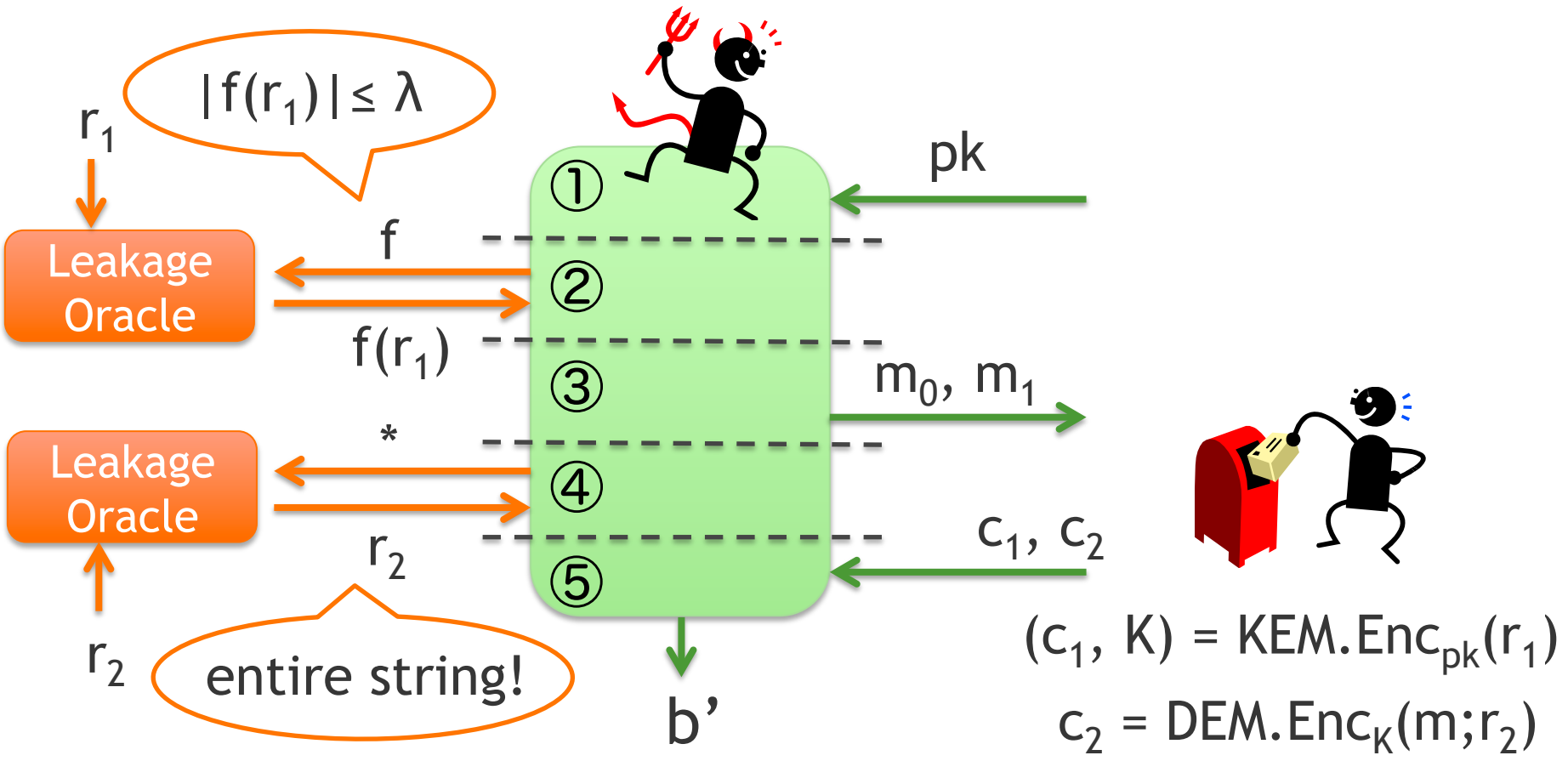
- Relaxation: Rand. for KEM/DEM leaks independently
  - The situation that KEM/DEM are implemented by different chips

Randomness Leakage in KEM/DEM

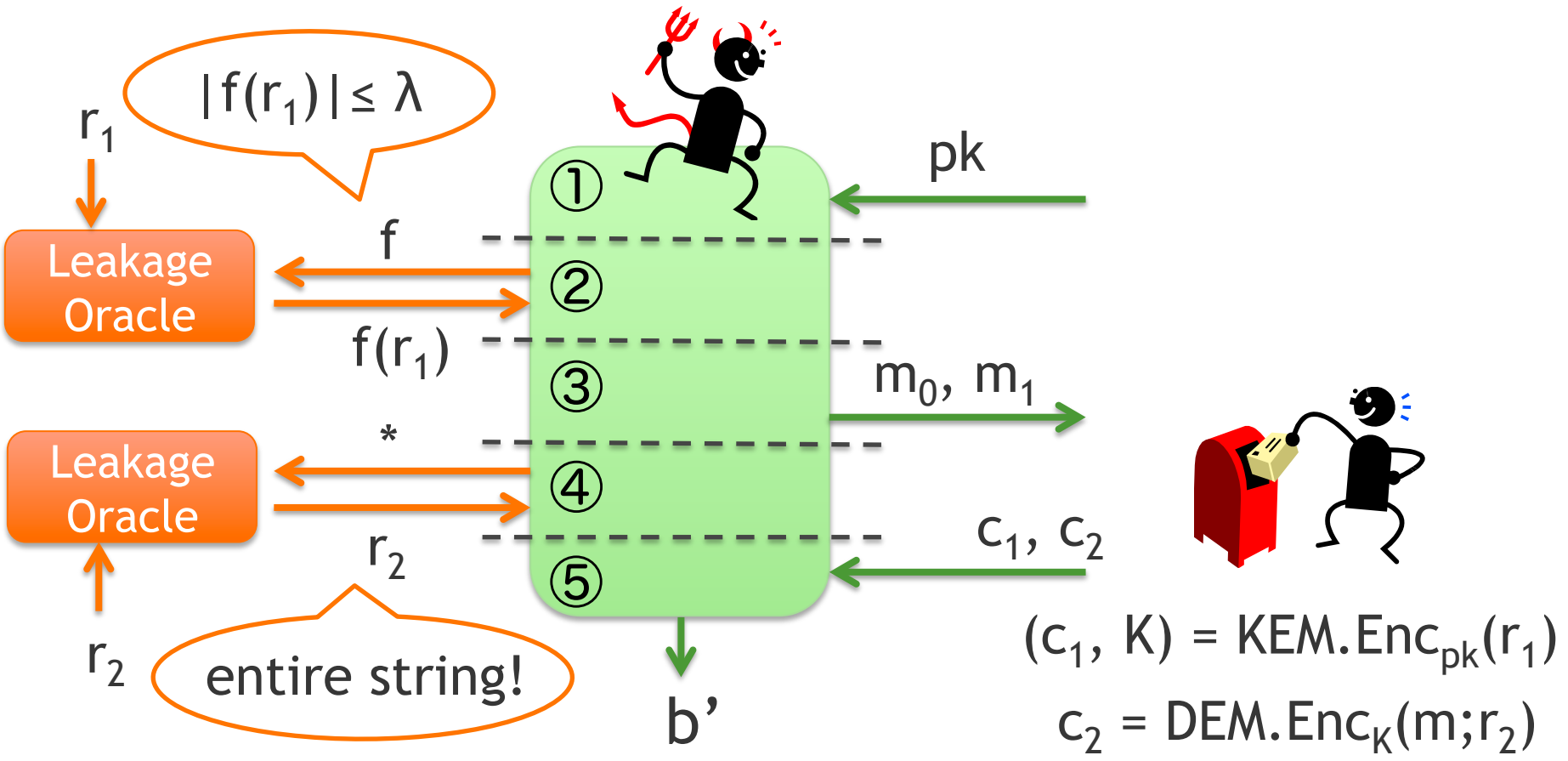


- Relaxation: Rand. for KEM/DEM leaks independently
  - The situation that KEM/DEM are implemented by different chips

Randomness Leakage in KEM/DEM



Randomness Leakage in KEM/DEM



- Remark:
  (1) Rand. for KEM/DEM leaks independently
  + (2) Messages are independent of DEM leakage

Secure randomness-LR KEM/DEM scheme

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme
  Exchange the roles of key and randomness !!

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme
  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen :

Enc :

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme
  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen : 1^n → param

sample → sk

param,sk → pk

PK = (param, pk), SK = sk

Enc :

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme
  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen : 1^n → param

sample → sk

param,sk → pk

PK = (param, pk), SK = sk

Enc : sample → r_1

param,r_1 → c_1 } KEM

→ K }

m, K → c_2 } DEM

C = (c_1, c_2)

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen : 1^n → param

sample → sk

param,sk → pk

PK = (param, pk), SK = sk

Enc : sample → r_1

param,r_1 → c_1 } KEM

→ K }

m, K → c_2 } DEM

C = (c_1, c_2)

Rand.-LR KEM/DEM scheme:

Gen :

PK = SK =

Enc :

C =

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen : 1^n → param

sample → sk

param,sk → pk

PK = (param, pk), SK = sk

Enc : sample → r_1

param,r_1 → c_1

→ K

m, K → c_2

C = (c_1, c_2)

Rand.-LR KEM/DEM scheme:

Gen :

PK = SK =

Enc :

C =

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

  Exchange the roles of key and randomness !!

Key-LR KEM/DEM scheme:

Gen : 1^n → param

sample → sk

param,sk → pk

PK = (param, pk), SK = sk

Enc : sample → r_1

param,r_1 → c_1

→ K

m, K → c_2

C = (c_1, c_2)

Rand.-LR KEM/DEM scheme:

Gen :

PK = SK =

Enc : sample → sk

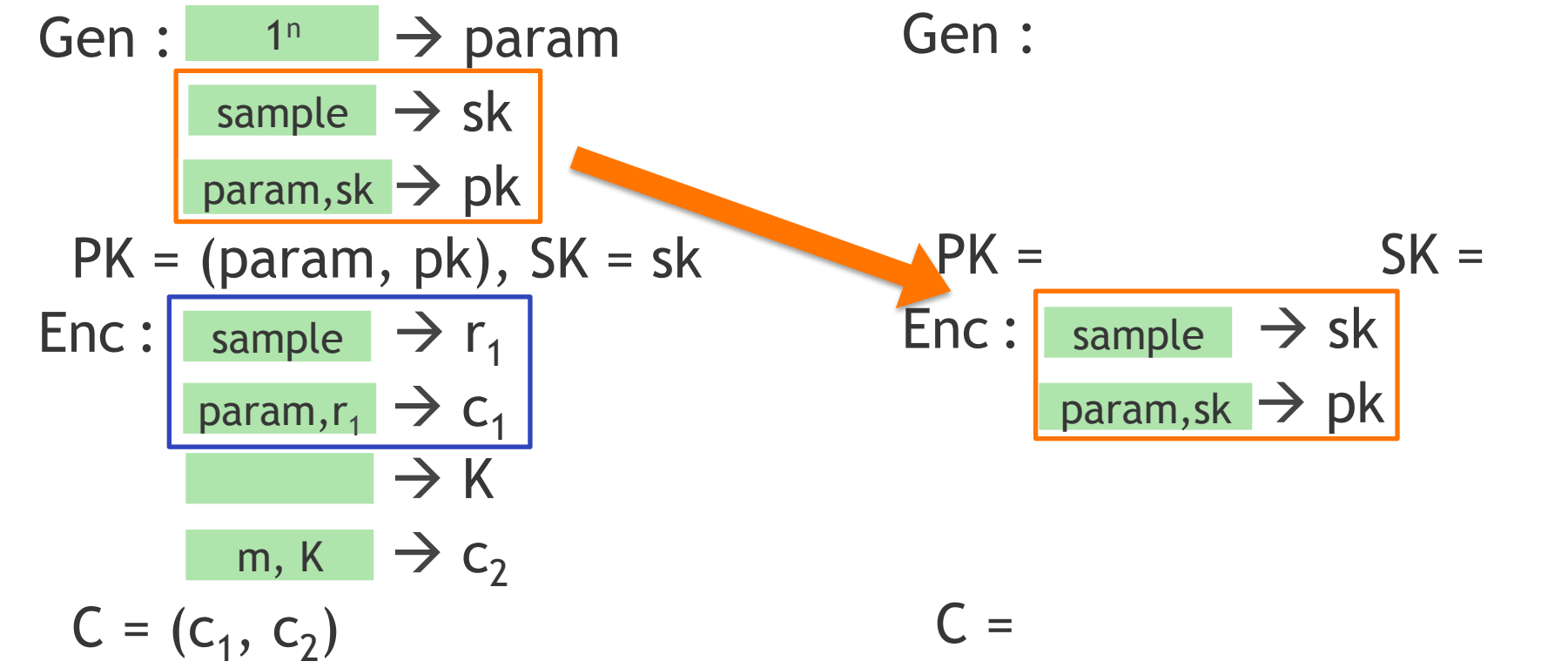
param,sk → pk

C =

Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

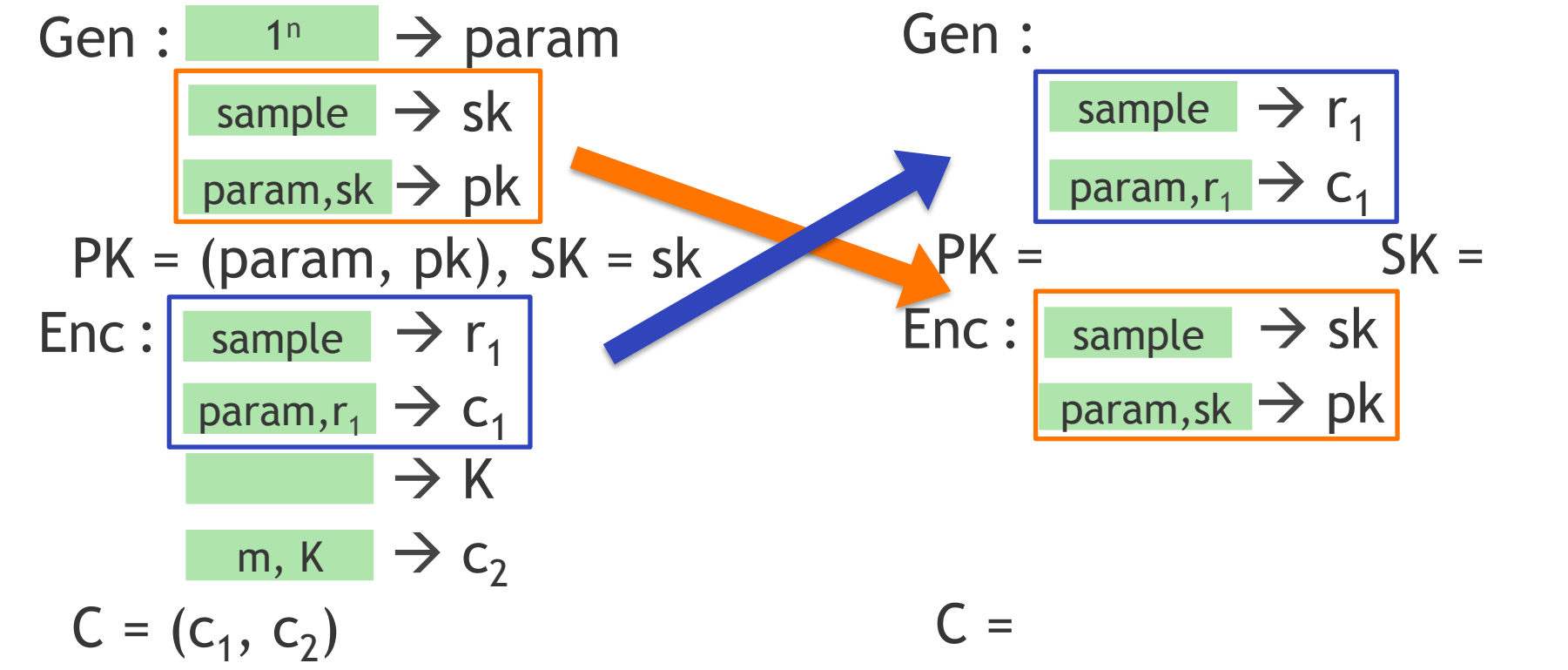
  Exchange the roles of key and randomness !!



Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

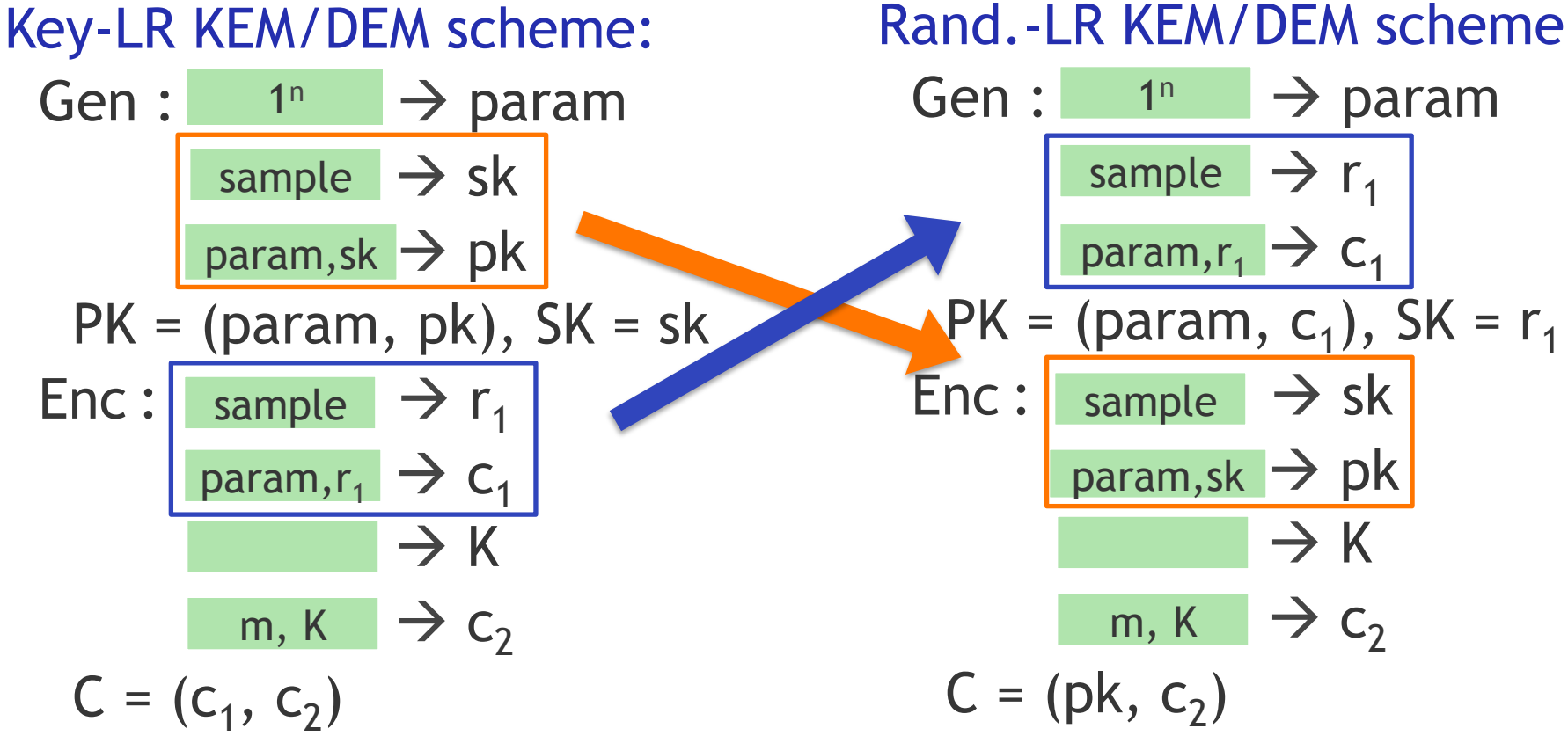
  Exchange the roles of key and randomness !!



Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

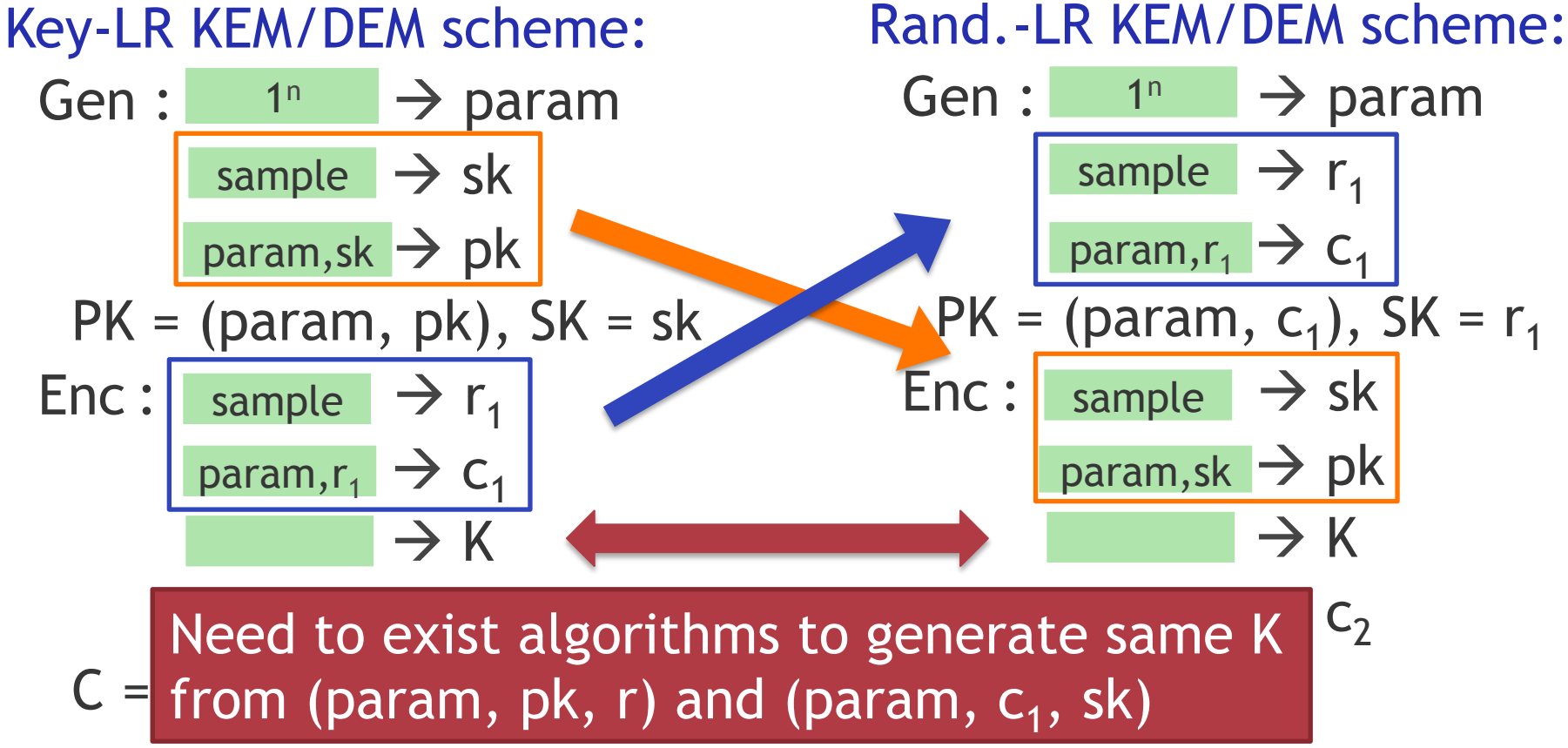
  Exchange the roles of key and randomness !!



Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

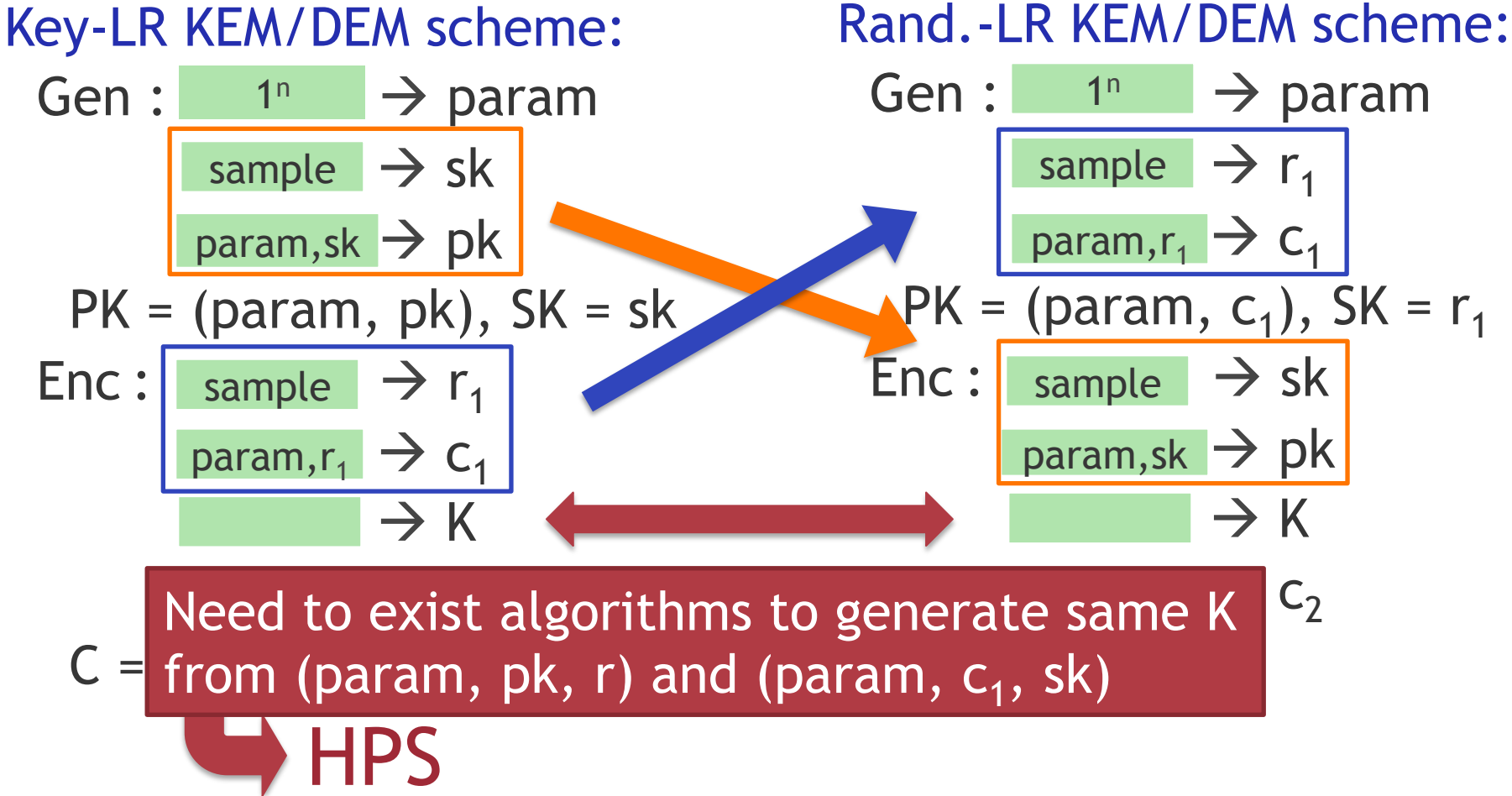
  Exchange the roles of key and randomness !!



Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

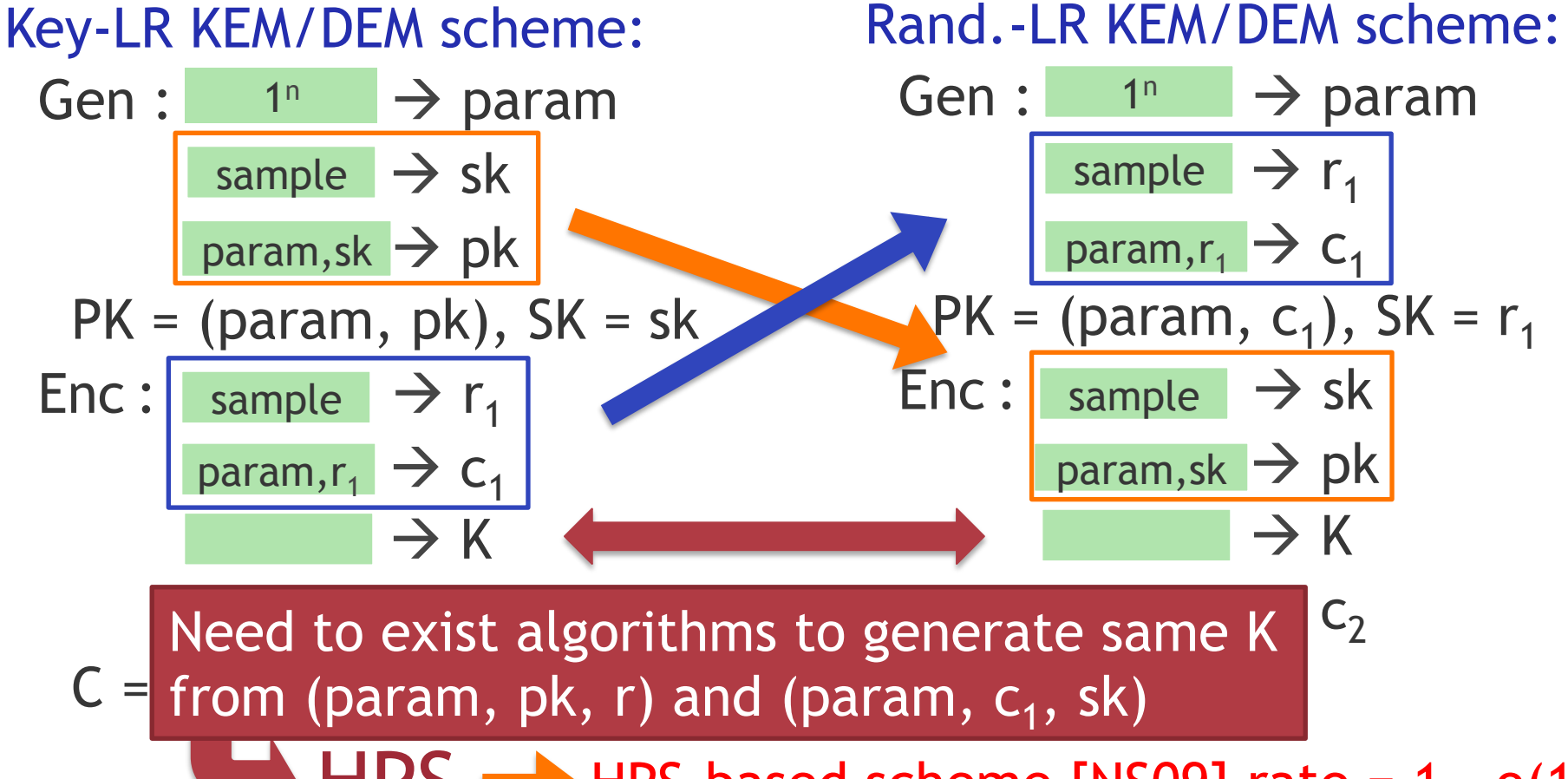
  Exchange the roles of key and randomness !!



Secure randomness-LR KEM/DEM scheme

- Idea: key-LR scheme → randomness-LR scheme

Exchange the roles of key and randomness !!



Conclusions

- Leakage of randomness in PKE
- Our results
  - No secure scheme if leakage occurs after PK is published
  - Secure KEM/DEM scheme even if leakage occurs after PK is published
    - Restriction: Rand. in KEM/DEM leaks independently + message is independent of DEM leakage
    - Idea: key-LR → randomness-LR
    - Leakage rate: 1 – o(1) from key-LR scheme [NS09]

Thank you

Related work

- Hedged public-key encryption [BBN+09]
  - Adversary can choose a joint distribution of message and randomness (with enough entropy)
    - If uniform randomness → CPA-security
      otherwise → weaker security
  - Can be seen as randomness-LR PKE
    - Randomness leakage = Choice of distribution
    - Corresponding to randomness leakage before public key is published
      - Message must be independent of public key

Secure randomness-LR KEM/DEM scheme

Key-LR scheme [NS09]:

Gen : $g_1, g_2 \in_R G$ → param

$x_1, x_2 \in_R Z_p$ → sk

$h = g_1^{x1} g_2^{x2}$ → pk

PK = (param, pk), SK = sk

Enc : $r \in_R Z_p$ → r

g_1^r, g_2^r → c_1

h^r → K

$s \in_R \{0,1\}^t$

$Ext(K,s)+m$ → c_2

C = (c_1, c_2)

Our scheme:

Gen : $g_1, g_2 \in_R G$ → param

$r \in_R Z_p$ → r

g_1^r, g_2^r → c_1

PK = (param, c_1), SK = r

Enc : $x_1, x_2 \in_R Z_p$ → sk

$h = g_1^{x1} g_2^{x2}$ → pk

$(g_1^r)^{x1}(g_2^r)^{x2}$ → K

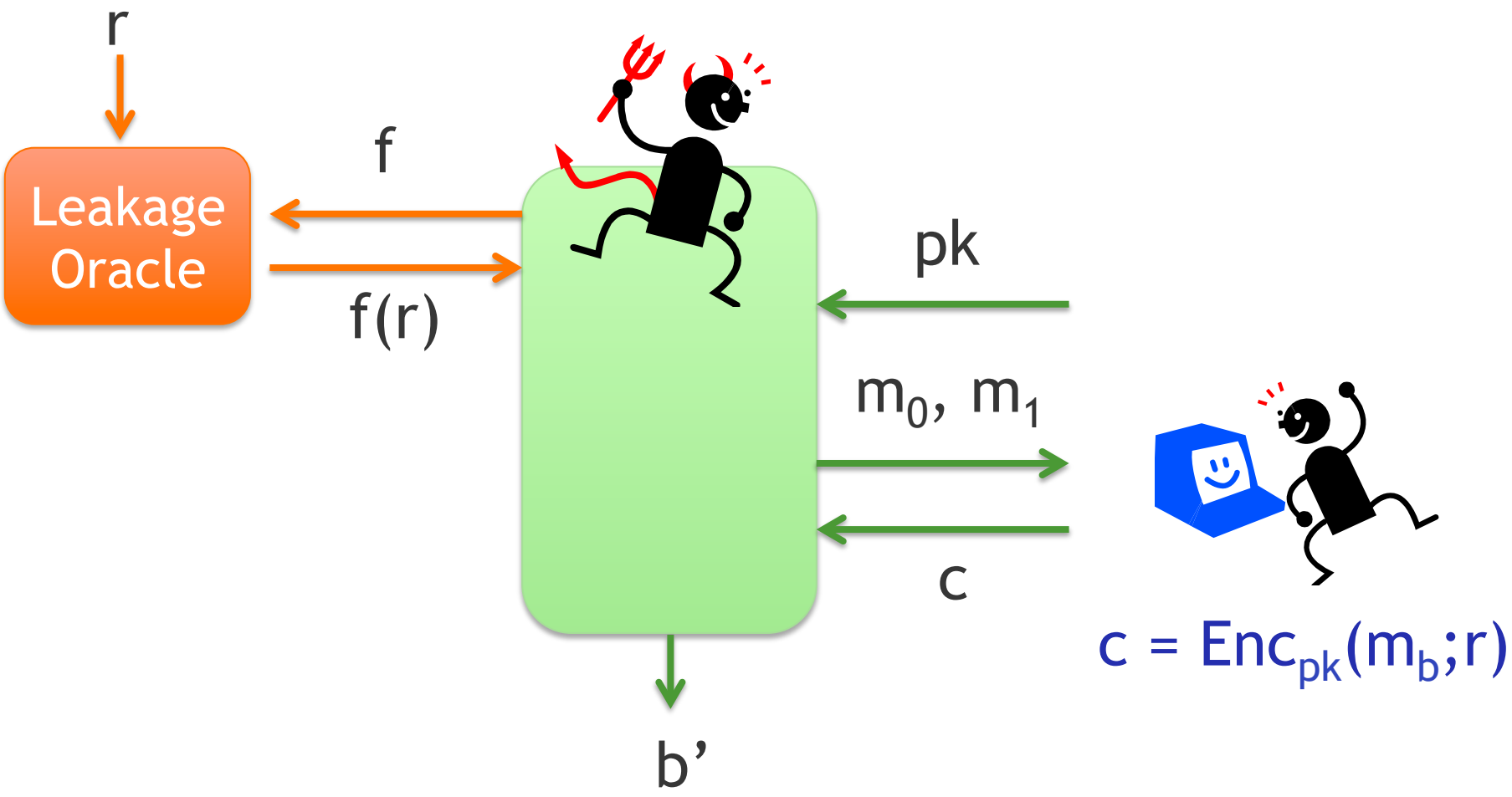
$s \in_R \{0,1\}^t$

$Ext(K, s)+m$ → c_2

C = (pk, c_2)

- ElGamal-based scheme of [NS09] → leak rate = 1/2
  - HPS-based scheme of [NS09] → leak rate = 1 - o(1)

Leakage occurs before public key is published



Leakage occurs before public key is published

- $\pi' = (Gen', Enc', Dec')$
  $Gen'(1^n) : s \leftarrow U_t$, $(pk, sk) \leftarrow Gen(1^n)$,
  $pk' = (pk, s)$, $sk' = sk$
  $Enc'_{pk'}(m) : r \leftarrow U_k$, $c = Enc_{pk}(m; Ext(r,s))$
  $Dec'_{sk'}(c) = Dec_{sk}(c)$

- Theorem.
  If $\pi = (Gen, Enc, Dec)$ is CPA-secure,
  then π' is randomness-LR secure
  - Proof: $Ext(r,s)$ is (almost) uniform
    even if $f(r)$ leaks
  - Remark: Only one-message (or bounded-poly-many message) security