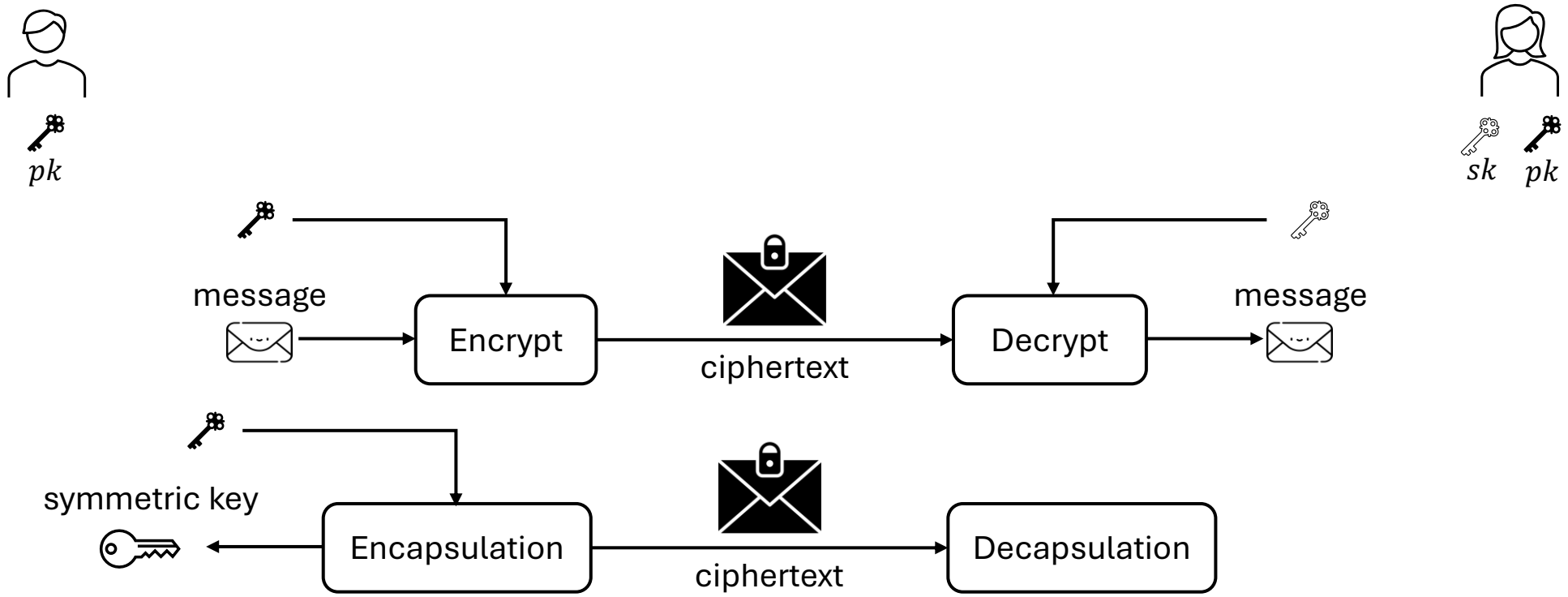
Cryptography Engineering

- Lecture 5 (Nov 19, 2025)
- Today's notes:
  - IND-CCA security
  - Fujisaki-Okamoto transformation

KEM and PKE

- Key Encapsulation Mechanism (KEM) v.s. Public-key Encryption (PKE)



- PKE:
- KEM:

Indistinguishability

- How do we define **Confidentiality** (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key **cannot learn any information** from a ciphertext

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define **“cannot learn any information”?**

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk **cannot fully recover** its plaintext.

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk cannot fully recover its plaintext.
  - Counterexample: **Cannot fully recover ≠ Learn nothing.**

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk cannot fully recover its plaintext.
  - Counterexample: Cannot fully recover ≠ Learn nothing.
  - A useful notion as an **underlying building block**, **but not sufficient for defining Confidentiality.**

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk cannot fully recover its plaintext.
  - Counterexample: Cannot fully recover ≠ Learn nothing.
  - A useful notion as an underlying building block, but not sufficient for defining Confidentiality.

- Standard definition: **Indistinguishability**
  - $\forall m_0, m_1, (pk, m_0, m_1, \mathrm{Enc}(pk, m_0))$ is indistinguishable from $(pk, m_0, m_1, \mathrm{Enc}(pk, m_1))$

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk cannot fully recover its plaintext.
  - Counterexample: Cannot fully recover ≠ Learn nothing.
  - A useful notion as an underlying building block, but not sufficient for defining Confidentiality.

- Standard definition: **Indistinguishability**
  - $\forall m_0, m_1, (pk, m_0, m_1, \mathrm{Enc}(pk, m_0))$ is indistinguishable from $(pk, m_0, m_1, \mathrm{Enc}(pk, m_1))$
  - Here confidentiality follows by letting m_1 as a meaningless plaintext (e.g., $m_1 = 00 \ldots 00$).

Indistinguishability

- How do we define Confidentiality (Vertraulichkeit) of an encryption scheme?
  - Intuition: Any party without the secret key cannot learn any information from a ciphertext
  - But how should we define “cannot learn any information”?

- An incomplete definition: One-wayness
  - Given a ciphertext, any party without sk cannot fully recover its plaintext.
  - Counterexample: Cannot fully recover ≠ Learn nothing.
  - A useful notion as an underlying building block, but not sufficient for defining Confidentiality.

- Standard definition: **Indistinguishability**
  - $\forall m_0, m_1, (pk, m_0, m_1, \mathrm{Enc}(pk, m_0))$ is indistinguishable from $(pk, m_0, m_1, \mathrm{Enc}(pk, m_1))$
  - Here confidentiality follows by letting m_1 as a meaningless plaintext (e.g., $m_1 = 00 \ldots 00$).
  - **In the real world: Computationally indistinguishability**

IND against Chosen-Plaintext Attacks

- When defining security, we also need to consider the adversary's capabilities:
  - Stronger capability => Capture stronger security, more complex attacking scenarios…

IND against Chosen-Plaintext Attacks

- When defining security, we also need to consider the adversary's capabilities:
  - Stronger capability => Capture stronger security, more complex attacking scenarios…

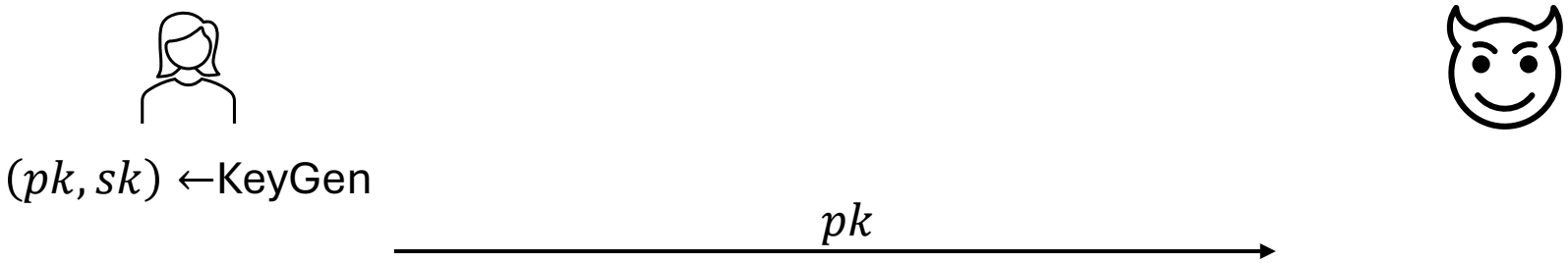
- IND-CPA: Indistinguishability against **Chosen-Plaintext Attacks**
  - The adversary has the public key, so it can use the pk **to encrypt any plaintext it wants**…
  - Basic security of a public-key encryption scheme

IND against Chosen-Plaintext Attacks

- When defining security, we also need to consider the adversary's capabilities:
  - Stronger capability => Capture stronger security, more complex attacking scenarios…
- IND-CPA: Indistinguishability against Chosen-Plaintext Attacks
  - The adversary has the public key, so it can use the pk to encrypt any plaintext it wants…
  - Basic security of a public-key encryption scheme
- ElGamal encryption/DHIES scheme are IND-CPA secure under Diffie-Hellman assumptions.

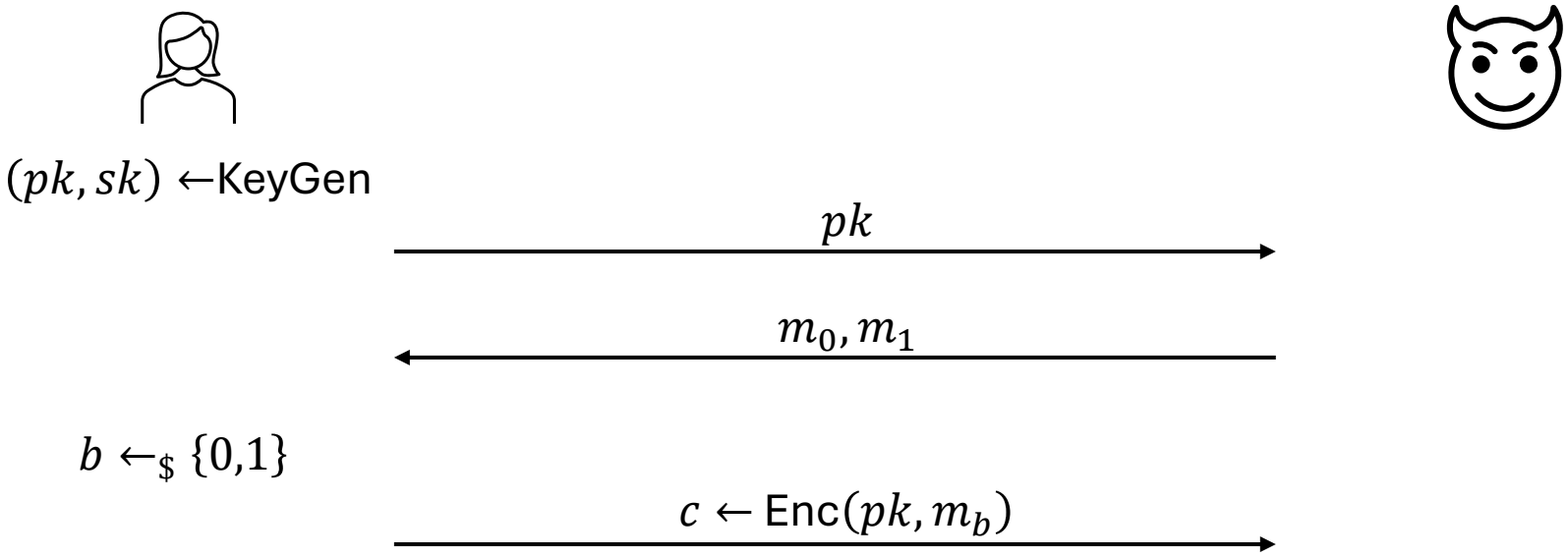
IND against Chosen-Plaintext Attacks

- IND-CPA security game:



IND against Chosen-Plaintext Attacks

- IND-CPA security game:



IND against Chosen-Plaintext Attacks

- IND-CPA security game:

$(pk, sk) \leftarrow$KeyGen

pk →

← m_0, m_1

$b \leftarrow_{\$} \{0,1\}$

$c \leftarrow \text{Enc}(pk, m_b)$ →

← b'

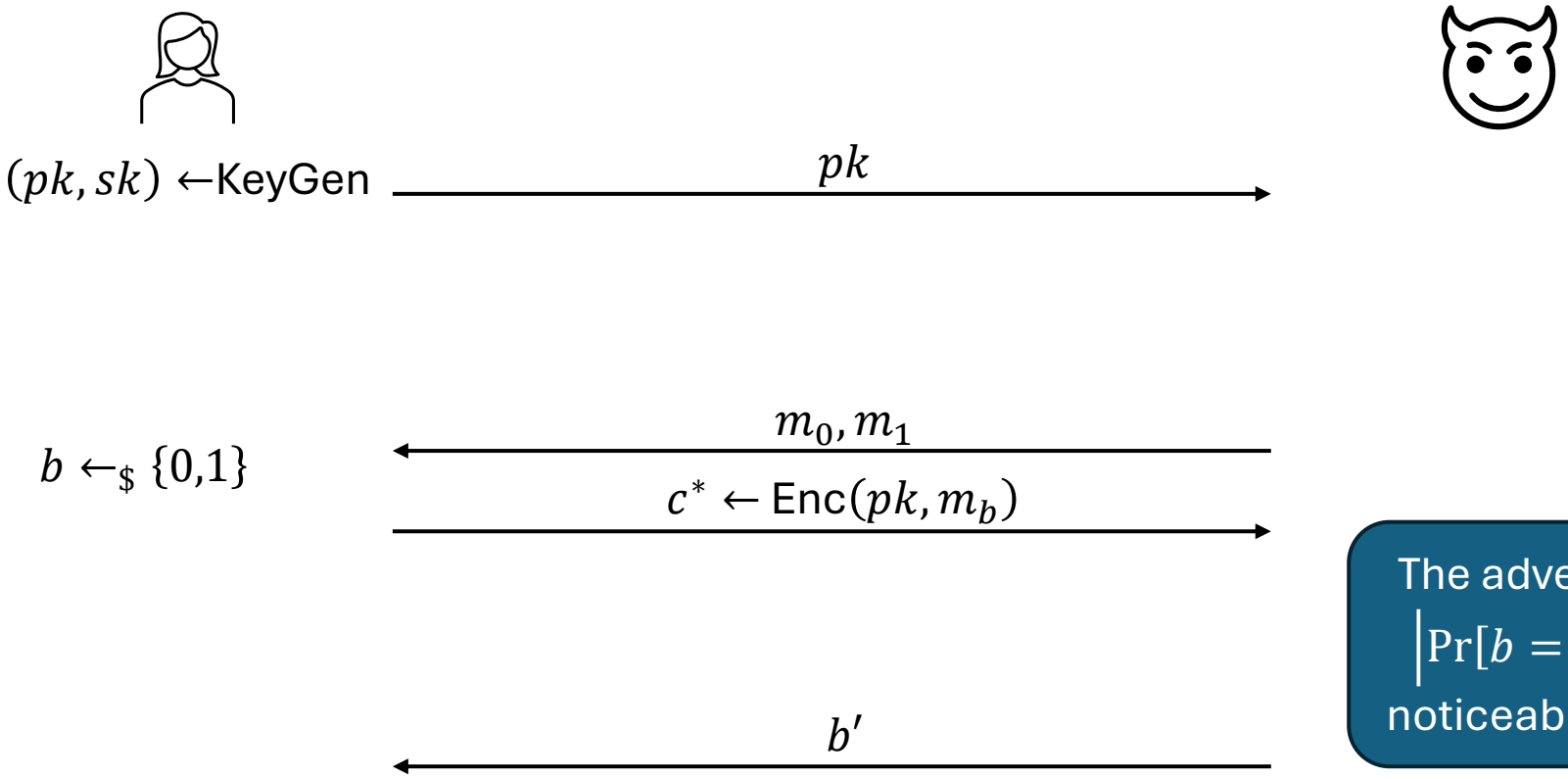
The adversary wins if $\left|\Pr[b = b'] - \frac{1}{2}\right|$ is a noticeable probability

IND against Chosen-Ciphertext Attacks

- IND-CCA: Indistinguishability against **Chosen-Ciphertext Attacks**
  - The adversary has the public key, so it can use the pk to encrypt any plaintext it wants…
  - The adversary can also ask the sk owner **to encrypt ciphertexts chosen by the adversary**

IND against Chosen-Ciphertext Attacks

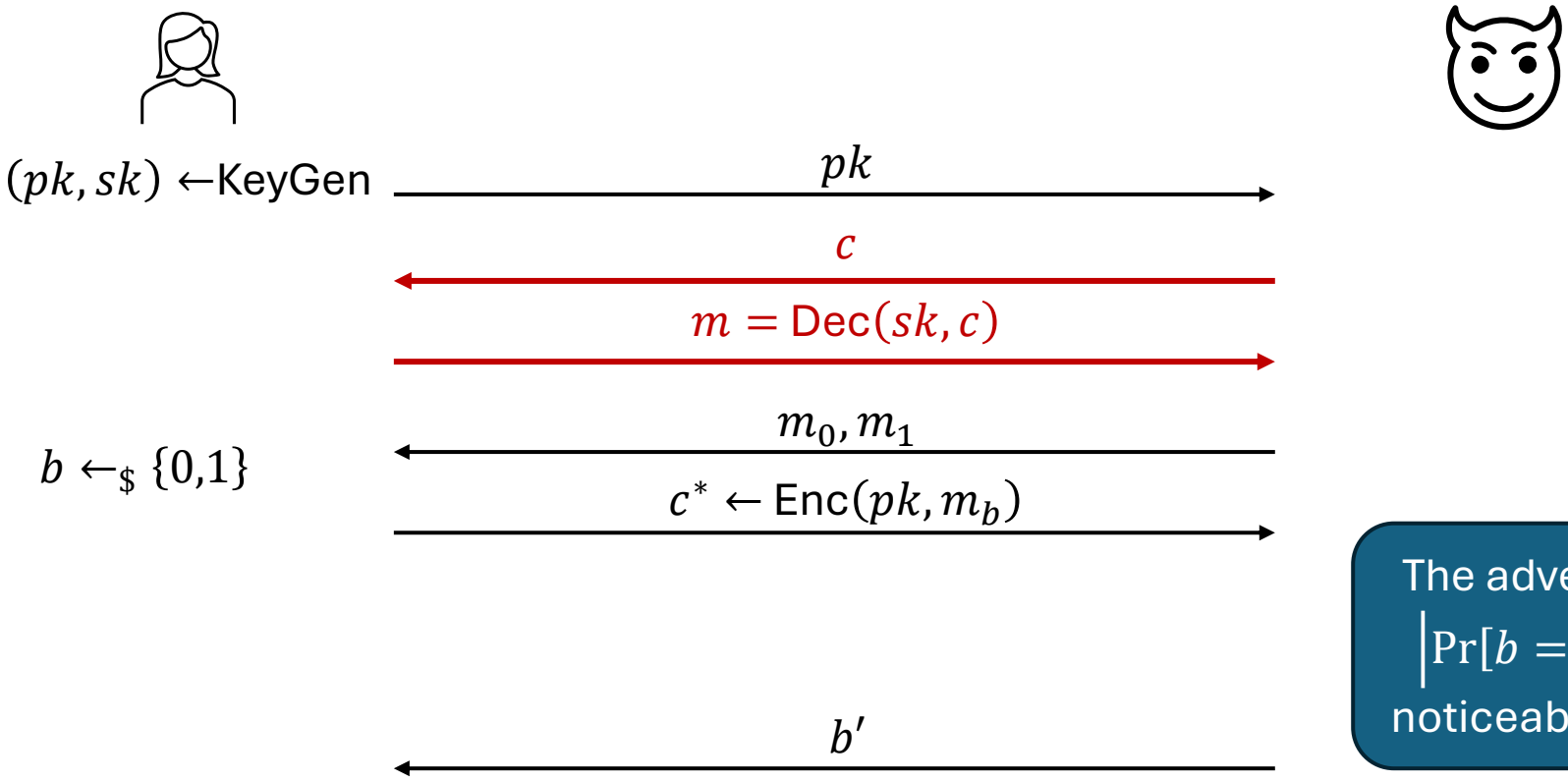
- IND-CPA security game:



The adversary wins if $\left|\Pr[b = b'] - \frac{1}{2}\right|$ is a noticeable probability

IND against Chosen-Ciphertext Attacks

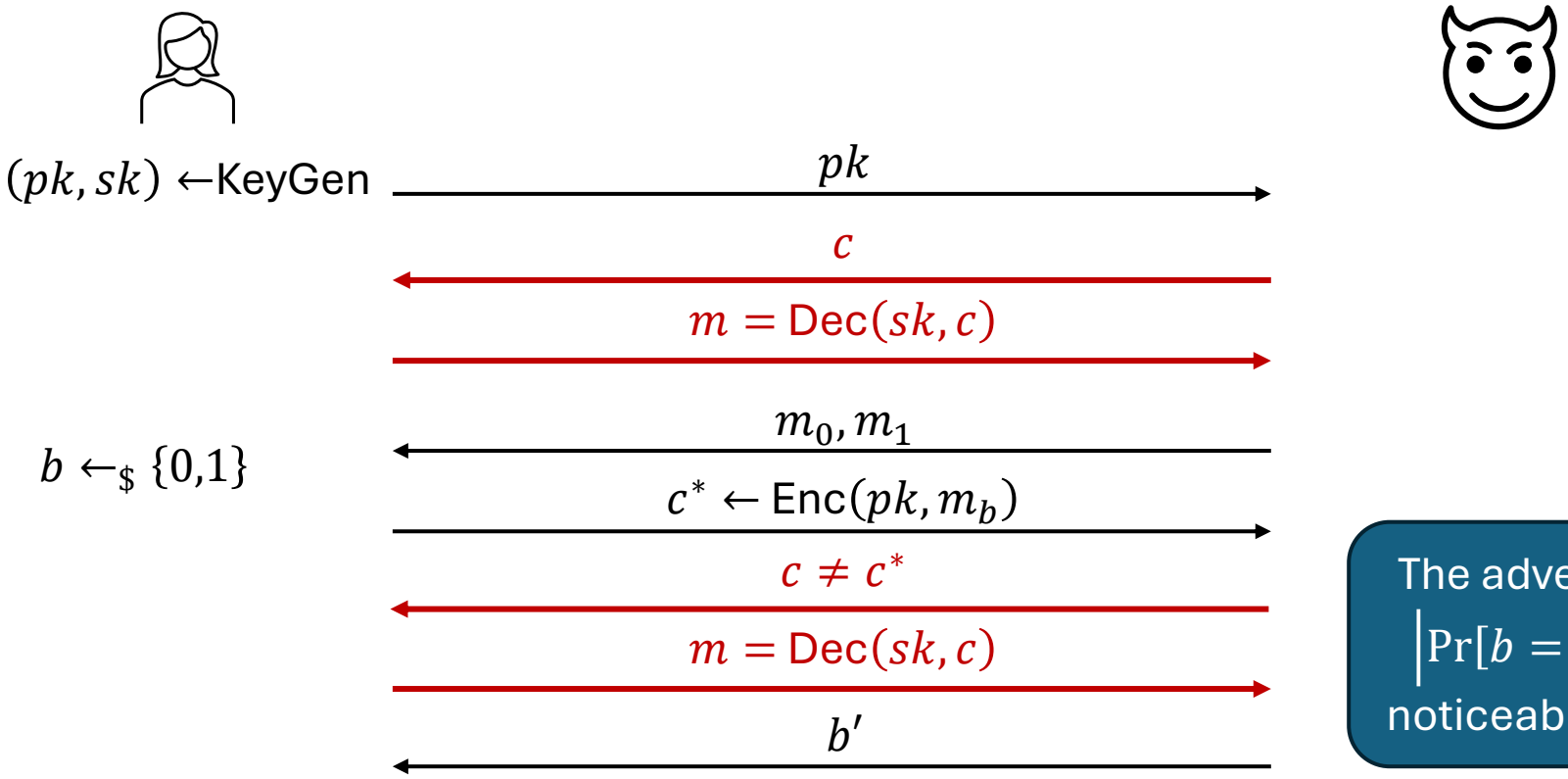
- IND-**CCA** security game:



The adversary wins if $\left|\Pr[b = b'] - \frac{1}{2}\right|$ is a noticeable probability

IND against Chosen-Ciphertext Attacks

- IND-**CCA** security game:



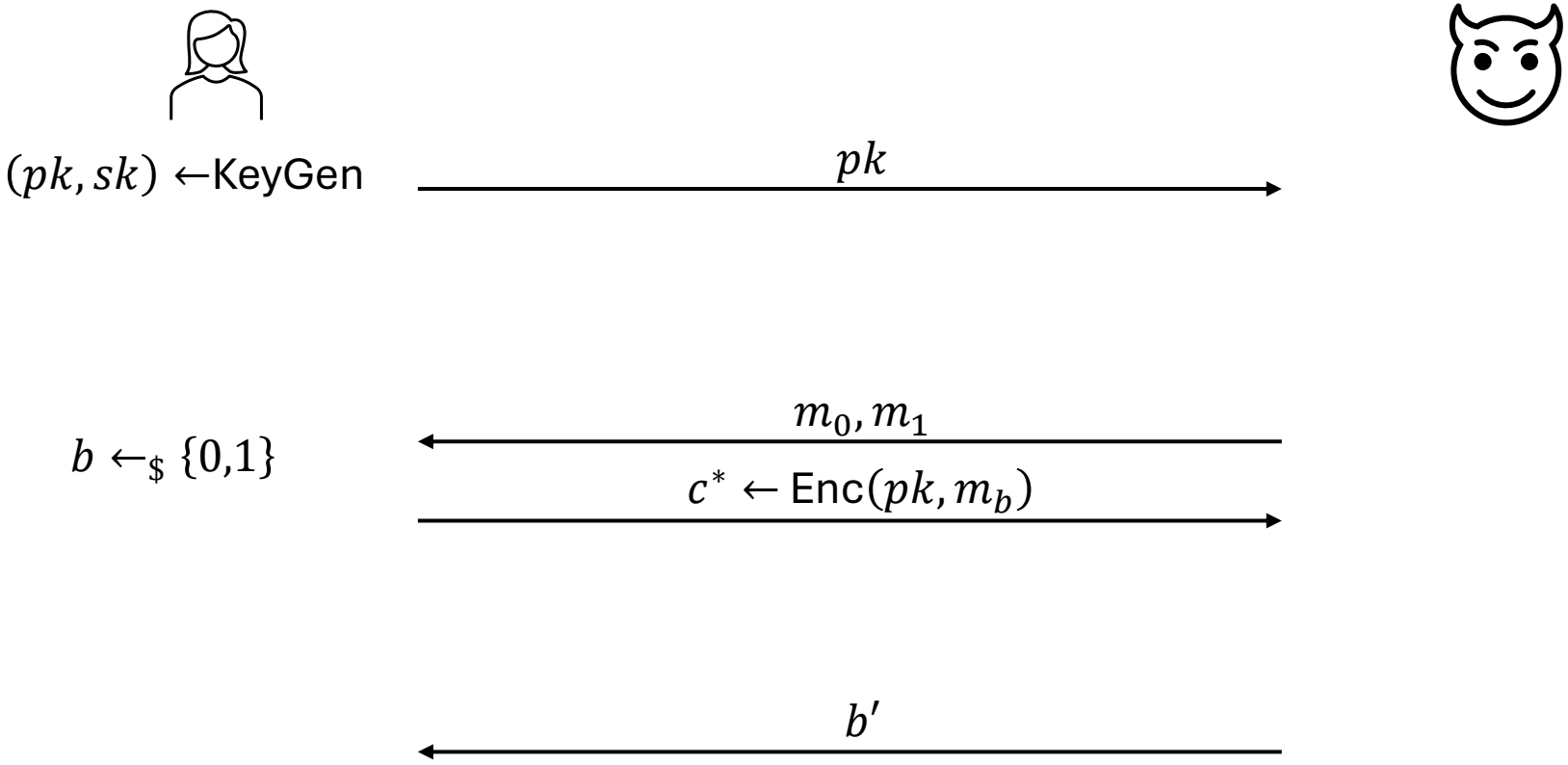
The adversary wins if $\left|\Pr[b = b'] - \frac{1}{2}\right|$ is a noticeable probability

IND against Chosen-Ciphertext Attacks

- IND-CCA: Indistinguishability against **Chosen-Ciphertext Attacks**
  - The adversary has the public key, so it can use the pk to encrypt any plaintext it wants…
  - The adversary can also ask the sk owner to encrypt ciphertexts chosen by the adversary
  - Standard requirement in modern PKE/KEM designs
  - More motivations: https://www.csa.iisc.ac.in/~arpita/Cryptography15/Scribe4M.pdf

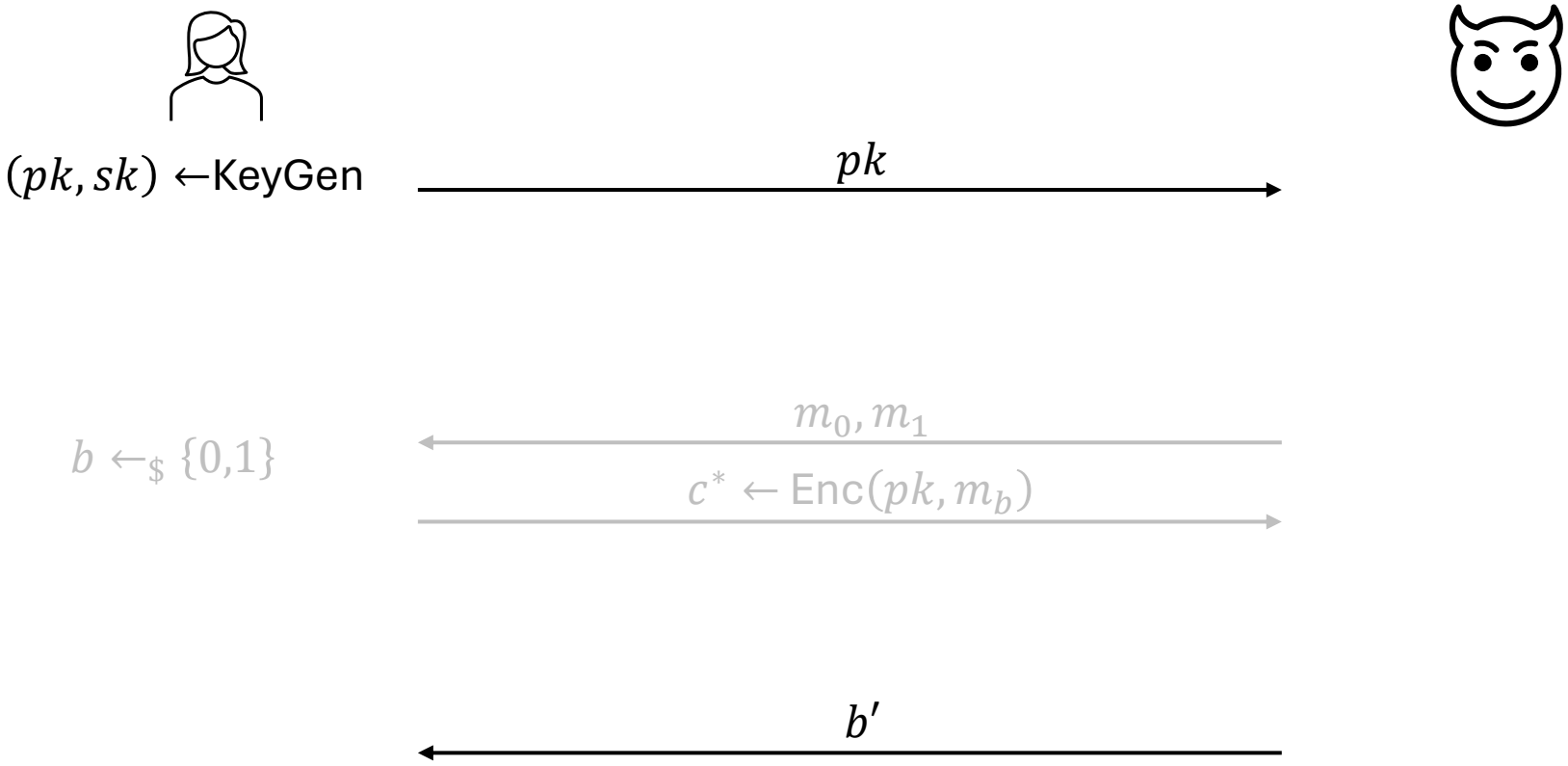
IND-CPA & IND-CCA security for KEM

- IND-CPA security game for PKE:



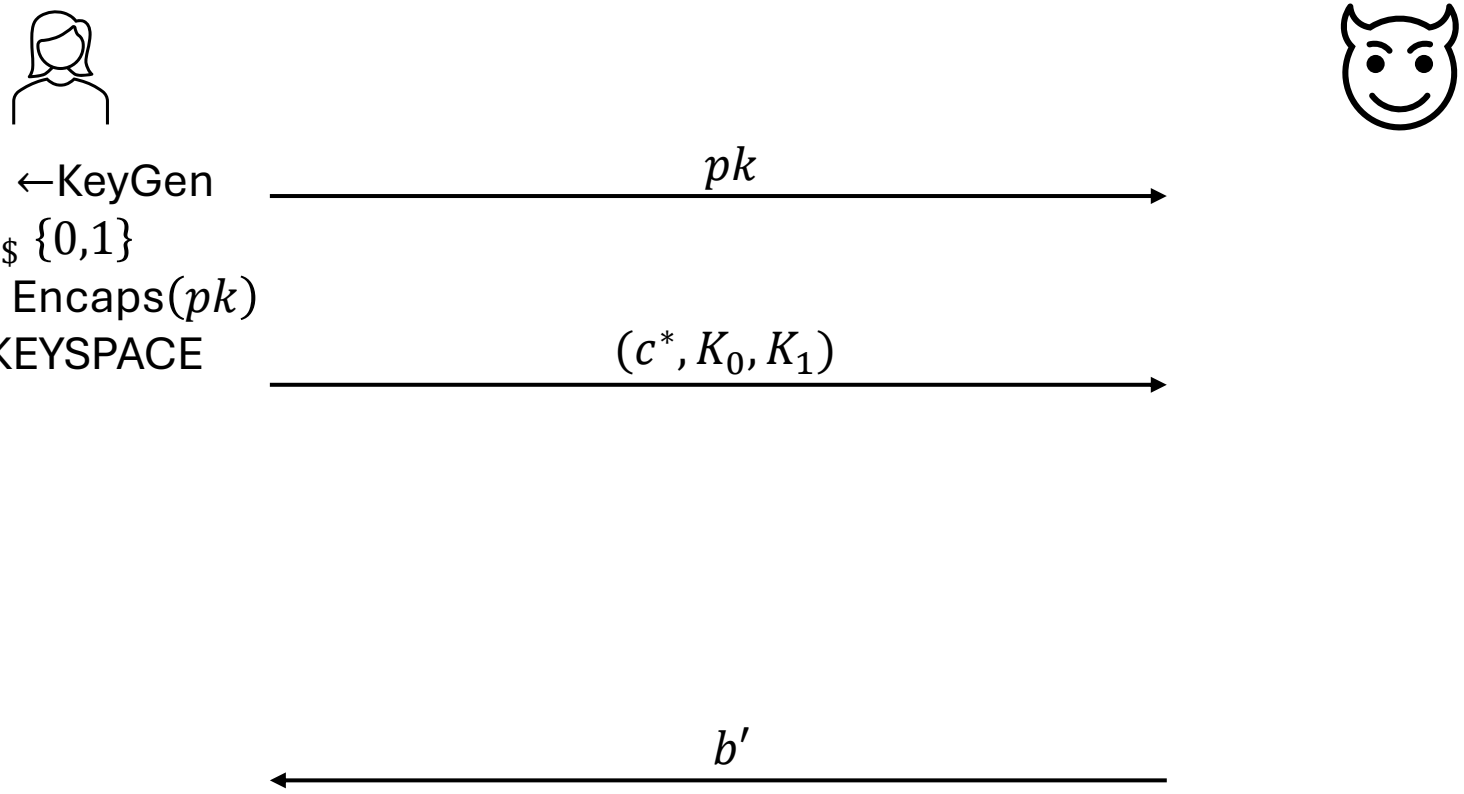
IND-CPA & IND-CCA security for KEM

- IND-CPA security game for **KEM**:



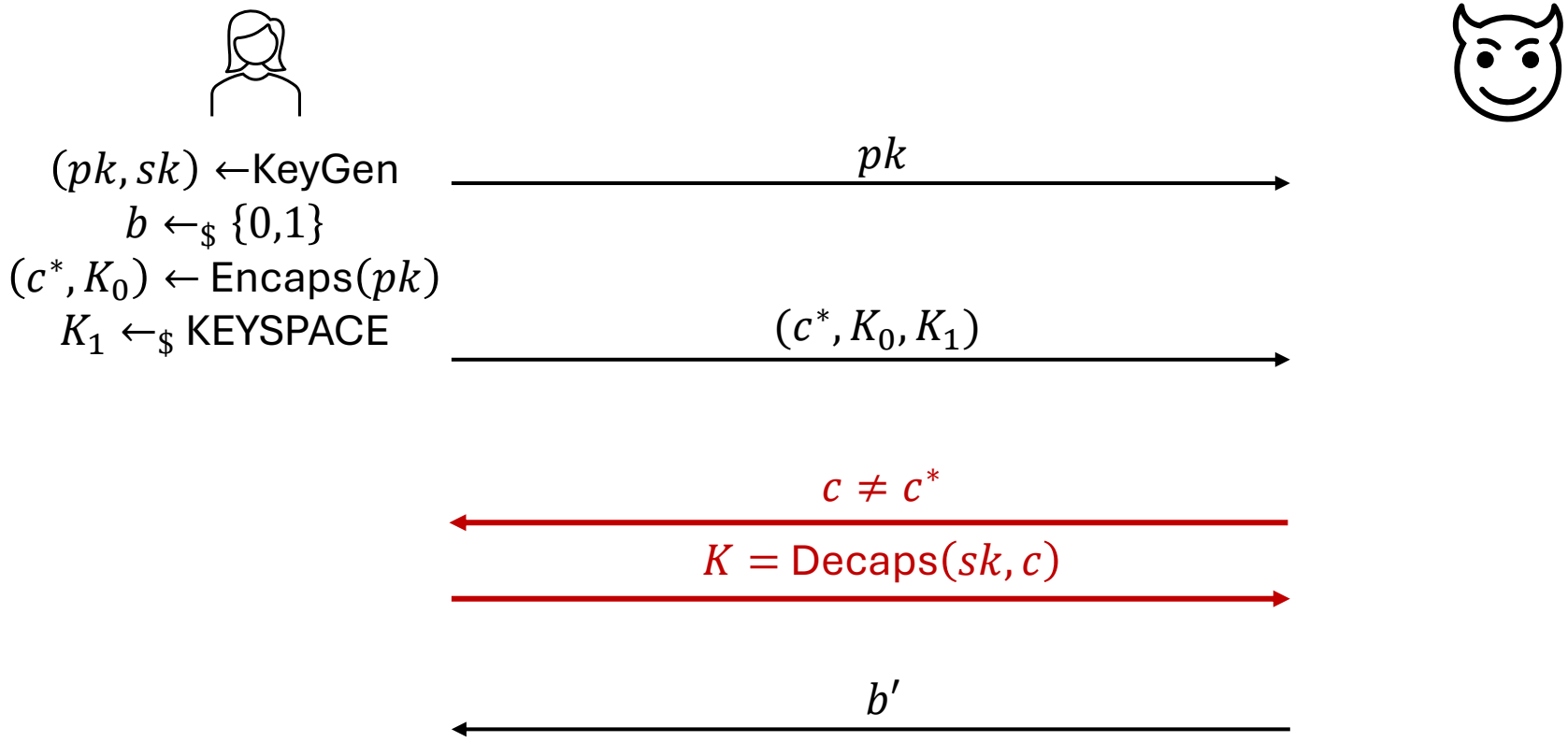
IND-CPA & IND-CCA security for KEM

- IND-CPA security game for **KEM**:



IND-CPA & IND-CCA security for KEM

- IND-**CCA** security game for **KEM**:



IND-CPA & IND-CCA security for KEM

- IND-**CCA** security game for **KEM**:

$(pk, sk) \leftarrow$KeyGen

$b \leftarrow_{\$} \{0,1\}$

$(c^*, K_0) \leftarrow \text{Encaps}(pk)$

$K_1 \leftarrow_{\$}$ KEYSPACE

pk →

(c^*, K_0, K_1) →

$c \neq c^*$ ←

$K = \text{Decaps}(sk, c)$ →

b' ←

The adversary wins if $\left|\Pr[b = b'] - \frac{1}{2}\right|$ is a noticeable probability

Fujisaki-Okamoto Transformation

- IND-CCA security is a standard requirement in modern PKE/KEM designs
  - But how can we achieve it?

Fujisaki-Okamoto Transformation

- IND-CCA security is a standard requirement in modern PKE/KEM designs
  - But how can we achieve it?

- **Fujisaki-Okamoto Transformation**
  - Convert IND-CPA PKE into IND-CCA PKE/KEM
  - In this lecture, we focus on a “IND-CPA PKE to IND-CCA KEM” variant

Fujisaki-Okamoto Transformation

- Let PKE = (KG, Enc, Dec) be a public-key encryption scheme
- Let H, G be two hash functions (Quick question: How to instantiate them using SHA256)
- We construct an FOKEM scheme based on PKE

Fujisaki-Okamoto Transformation

- Let PKE = (KG, Enc, Dec) be a public-key encryption scheme
- Let H, G be two hash functions (Quick question: How to instantiate them using SHA256)
- We construct an FOKEM scheme based on PKE

KeyGen:
1. $(pk, sk) \leftarrow$ KG
2. $prk \leftarrow \{0,1\}^L$
3. $pk := pk$
4. $sk := (prk, sk)$

Fujisaki-Okamoto Transformation

- Let PKE = (KG, Enc, Dec) be a public-key encryption scheme
- Let H, G be two hash functions (Quick question: How to instantiate them using SHA256)
- We construct an FOKEM scheme based on PKE

KeyGen:
1. $(pk, sk) \leftarrow$ KG
2. $prk \leftarrow \{0,1\}^L$
3. $pk := pk$
4. $sk := (prk, sk)$

Encaps$(pk = pk)$:
1. $m \leftarrow_{\$}$ MsgSpace
2. $r := G(pk, m)$
   // randomness for PKE
3. $c :=$ Enc$(pk, m; r)$
4. $K := H(pk, m, c)$
5. $c := c$
6. return (c, K)

Fujisaki-Okamoto Transformation

- Let PKE = (KG, Enc, Dec) be a public-key encryption scheme
- Let H, G be two hash functions (Quick question: How to instantiate them using SHA256)
- We construct an FOKEM scheme based on PKE

KeyGen:
1. $(pk, sk) \leftarrow$ KG
2. $prk \leftarrow \{0,1\}^L$
3. $pk := pk$
4. $sk := (prk, sk)$

Encaps$(pk = pk)$:
1. $m \leftarrow_{\$}$ MsgSpace
2. $r := G(pk, m)$
   // randomness for PKE
3. $c :=$ Enc$(pk, m; r)$
4. $K := H(pk, m, c)$
5. $c := c$
6. return (c, K)

Decaps$(sk = (prk, sk), c = c)$:
1. $m =$ Dec(sk, c)
2. $r := G(pk, m)$
   // Recover randomness
3. $c' :=$ Enc$(pk, m; r)$
   // Re-encryption check
4. If $c == c'$: return $H(pk, m, c)$
5. Else: return $H(pk, prk, c)$

Fujisaki-Okamoto Transformation

- Let PKE = (KG, Enc, Dec) be a public-key encryption scheme
- Let H, G be two hash functions (Quick question: How to instantiate them using SHA256)
- We construct an FOKEM scheme based on PKE.

- ElGamal encryption is **not IND-CCA secure**, but can we “save it” via FO transform?

Exercises

- Study the IND-CCA security of FO[ElGamal encryption], i.e., replace the PKE in the FOKEM construction with ElGamal encryption:
  - Write the IND-CCA game for ElGamal encryption scheme.
  - Show an IND-CCA attack against ElGamal encryption
    - *Try modifying* $c^* = (g^r, g^{xr} \cdot m_b)$ *to* $c' = (g^r, g^{xr} \cdot m_b \cdot m')$ *with some known* m'*, what happens if you submit* c' *for decrypting?*
  - Write the IND-CCA game for FO[ElGamal encryption].
  - Show why your attack fails when using FO.

- **Coding task:** Implement the Fujisaki-Okamoto transformation to convert your ElGamal encryption into an IND-CCA secure KEM.