Security reduction of FO transform and variations

Ganyu (Bruce) Xu

University of Waterloo

April, 2024

Fujisaki-Okamoto transformation, 1999

Inputs:

- Public-key encryption scheme: $(\mathsf{KeyGen}, E^{\mathsf{asym}}, D^{\mathsf{asym}})$
- Symmetric cipher $(E^{\mathsf{sym}}, D^{\mathsf{sym}})$
- Key-derivation function [1] $G : \{0,1\}^* \to \mathcal{K}^{\mathsf{sym}}$
- Hash function $H : \{0,1\}^* \to \mathsf{Coin}^{\mathsf{asym}}$

Hybrid scheme's key generation is identical to the PKE's

[1] This is also a hash function and follows the random oracle assumption

FO 1999 routines

Algorithm 1: E^{hy}

Input: $\text{pk}^{\text{hy}}, m \in \mathcal{M}^{\text{sym}}$
Output: $(e \in \mathcal{C}^{\text{asym}}, c \in \mathcal{C}^{\text{sym}})$
$\sigma \xleftarrow{\$} \mathcal{M}^{\text{asym}}$;
$a \leftarrow G(\sigma), c \leftarrow E_a^{\text{sym}}(m)$;
`// PKE encryption accepts` r `as a seed`
$r \leftarrow H(c, \sigma), e \leftarrow E^{\text{asym}}(\text{pk}, \sigma, r)$;
return (e, c);

Algorithm 2: D^{hy}

Input: $\text{pk}, \text{sk}, (e, c)$
$\hat{\sigma} \leftarrow D^{\text{asym}}(\text{sk}, e)$;
$\hat{r} \leftarrow H(c, \hat{\sigma})$;
$\hat{c} \leftarrow E^{\text{asym}}(\text{pk}, \hat{\sigma}, \hat{r})$;
if $\hat{c} \neq c$ **then**
    return $\perp$;
end
$\hat{a} \leftarrow G(\sigma)$;
$\hat{m} \leftarrow D_{\hat{a}}^{\text{sym}}(c)$;
return $\hat{m}$;

Fujisaki-Okamoto transformation, 1999

Security result

Under the random oracle assumption, for every IND-CCA adversary against the hybrid scheme with advantage $\epsilon^{\text{hy}}_{\text{IND-CCA}}$, there exists an OW-CPA adversary against the underlying PKE with advantage $\epsilon^{\text{asym}}_{\text{OW-CPA}}$ and an IND-CPA adversary against the underlying symmetric ciphert with advantage $\epsilon^{\text{sym}}_{\text{IND-CPA}}$ such that

$$\epsilon^{\text{hy}}_{\text{IND-CCA}} \leq q_D 2^{-\gamma} + q_H \epsilon^{\text{asym}}_{\text{OW-CPA}} + \epsilon^{\text{sym}}_{\text{IND-CPA}}$$

Fujisaki-Okamoto transformation, 1999

Proof overview:

- Use $\mathcal{A}^{\text{asym}}_{\text{OW-CPA}}$ and $\mathcal{A}^{\text{sym}}_{\text{IND-CPA}}$ to simulate the IND-CCA game
- Simulate decryption oracle without using secret key

Fujisaki-Okamoto transformation, 1999

To simulate $\mathcal{O}^D(e, c)$ without secret key:

Algorithm 3: Hybrid encryption E^{hy}

Input: $\mathrm{pk}^{\mathrm{hy}}, m \in \mathcal{M}^{\mathrm{sym}}$
Output: $(e \in \mathcal{C}^{\mathrm{asym}}, c \in \mathcal{C}^{\mathrm{sym}})$
$\sigma \xleftarrow{\$} \mathcal{M}^{\mathrm{asym}}$;
$a \leftarrow G(\sigma), c \leftarrow E_a^{\mathrm{sym}}(m)$;
`// PKE encryption accepts` r `as a seed`
$r \leftarrow H(c, \sigma)$, $e \leftarrow E^{\mathrm{asym}}(\mathrm{pk}, \sigma, r)$;
return (e, c);

Decryption oracle without secret key

Algorithm 4: $\mathcal{O}_1^D$: decryption oracle without sk

Input: The query $(\tilde{e}, \tilde{c})$

foreach (σ, c, r) *in H's tape* **do**
    if $c = \tilde{c}$ **then**
        $a \leftarrow G(\sigma)$;
        $m \leftarrow D_a^{\text{sym}}(\tilde{c})$;
        return m;
    end
end
return $\perp$;

Challenge encryption with truly random key/coin

Algorithm 5: Challenge encryption E_*^{hy}

Input: $\mathsf{pk}^{\mathsf{hy}}, m \in \mathcal{M}^{\mathsf{sym}}$

Output: $(e \in \mathcal{C}^{\mathsf{asym}}, c \in \mathcal{C}^{\mathsf{sym}})$

$\sigma \xleftarrow{\$} \mathcal{M}^{\mathsf{asym}}$;

$a \xleftarrow{\$} \mathcal{K}^{\mathsf{sym}}, c \leftarrow E_a^{\mathsf{sym}}(m)$;

`// PKE encryption accepts` r `as a seed`

$r \xleftarrow{\$} \mathsf{Coin}, e \leftarrow E^{\mathsf{asym}}(\mathsf{pk}, \sigma, r)$;

return (e, c);

Game 0: IND-CCA game

Algorithm 6: Vanilla IND-CCA game

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}();$

$(m_0, m_1) \xleftarrow{\$} \mathcal{A}^{\mathsf{hy}}_{\mathsf{IND\text{-}CCA}}(\mathsf{pk}, \mathcal{O}^D);$

$b \xleftarrow{\$} \{0, 1\};$

$c^* \leftarrow E^{\mathsf{hy}}(\mathsf{pk}, m_b);$

$\hat{b} \xleftarrow{\$} \mathcal{A}^{\mathsf{hy}}_{\mathsf{IND\text{-}CCA}}(\mathsf{pk}, \mathcal{O}^D, c^*);$

Adversary wins if $\hat{b} = b$;

Game 1: modify the decryption oracle

Algorithm 7: Game 1

$(\mathrm{pk}, \mathrm{sk}) \xleftarrow{\$} \mathrm{KeyGen}();$

$(m_0, m_1) \xleftarrow{\$} \mathcal{A}^{\mathrm{hy}}_{\mathrm{IND\text{-}CCA}}(\mathrm{pk}, \mathcal{O}^D_1);$

$b \xleftarrow{\$} \{0, 1\};$

$c^* \leftarrow E^{\mathrm{hy}}(\mathrm{pk}, m_b);$

$\hat{b} \xleftarrow{\$} \mathcal{A}^{\mathrm{hy}}_{\mathrm{IND\text{-}CCA}}(\mathrm{pk}, \mathcal{O}^D_1, c^*);$

Adversary wins if $\hat{b} = b$;

Loss of security when $\mathcal{A}$ queries $\mathcal{O}^D$ with valid ciphertexts built without querying H at least once

$$\epsilon_0 - \epsilon_1 \leq q_D 2^{-\gamma}$$

Game 2: use true randomness in challenge encryption

Algorithm 8: Game 2

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}()$;
$(m_0, m_1) \xleftarrow{\$} \mathcal{A}^{\mathsf{hy}}_{\mathsf{IND\text{-}CCA}}(\mathsf{pk}, \mathcal{O}^D_1)$;
$b \xleftarrow{\$} \{0, 1\}$;
$c^* \leftarrow E^{\mathsf{hy}}_*(\mathsf{pk}, m_b)$;
$\hat{b} \xleftarrow{\$} \mathcal{A}^{\mathsf{hy}}_{\mathsf{IND\text{-}CCA}}(\mathsf{pk}, \mathcal{O}^D_1, c^*)$;
Adversary wins if $\hat{b} = b$;

Loss of security when $\mathcal{A}$ queries either G or H with σ^*

$$\epsilon_1 - \epsilon_2 \leq P[\mathsf{QUERY}^*]$$

Simulate game 2 with IND-CPA adversary

Algorithm 9: Symmetric cipher IND-CPA game $(E^{\text{sym}}, D^{\text{sym}})$

$a^* \xleftarrow{\$} \mathcal{K}^{\text{sym}}$;

$(\text{pk}, \text{sk}) \xleftarrow{\$} \text{KeyGen}^{\text{hy}}()$;

$(m_0, m_1) \xleftarrow{\$} \mathcal{A}^{\text{hy}}_{\text{IND-CCA}}(\text{pk}, \mathcal{O}^D_1)$;

$b \xleftarrow{\$} \{0, 1\}$;

$c^* \leftarrow E^{\text{sym}}_{a^*}(m_b)$;

$\sigma^* \xleftarrow{\$} \mathcal{M}^{\text{asym}}, r^* \xleftarrow{\$} \text{Coin}$;

$e^* \leftarrow E^{\text{asym}}(\text{pk}, \sigma^*, r^*)$;

$\hat{b} \xleftarrow{\$} \mathcal{A}^{\text{hy}}_{\text{IND-CCA}}(\text{pk}, \mathcal{O}^D_1, (e^*, c^*))$;

$\mathcal{A}^{\text{sym}}_{\text{IND-CPA}}$ wins if $\hat{b} = b$

$\mathcal{A}^{\text{sym}}_{\text{IND-CPA}}$ perfectly simulates game 2 and wins iff $\mathcal{A}^{\text{hy}}_{\text{IND-CCA}}$ wins

$$\epsilon_2 = \epsilon^{\text{sym}}_{\text{IND-CPA}}$$

Simulate game 2 with OW-CPA adversary

Algorithm 10: OW-CPA game against $(E^{\text{asym}}, D^{\text{asym}})$

$(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}^{\text{asym}}()$;

$\sigma^* \xleftarrow{\$} \mathcal{M}^{\text{asym}}$; $e^* \xleftarrow{\$} E^{\text{asym}}(\text{pk}, \sigma^*)$ `// truly random coin`;

$(m_0, m_1) \xleftarrow{\$} \mathcal{A}^{\text{hy}}_{\text{IND-CCA}}(\text{pk}, \mathcal{O}^D_1)$;

$a^* \xleftarrow{\$} \mathcal{K}^{\text{sym}}$; $b \xleftarrow{\$} \{0, 1\}$; $c^* \leftarrow E^{\text{sym}}_{a^*}(m_b)$;

$\mathcal{A}^{\text{hy}}_{\text{IND-CCA}}(\text{pk}, \mathcal{O}^D_1, (e^*, c^*))$ `// discard the output`;

Sample a random σ from the tape of H or G;

$\mathcal{A}^{\text{asym}}_{\text{OW-CPA}}$ wins if $\sigma = \sigma^*$

$\mathcal{A}^{\text{asym}}_{\text{OW-CPA}}$ wins if $\mathcal{A}^{\text{hy}}_{\text{IND-CCA}}$ queried on σ^* (aka QUERY*) and the randomly chosen σ is the correct one:

$$\epsilon^{\text{asym}}_{\text{OW-CPA}} = P[\text{QUERY}^*] \cdot \frac{1}{q_H}$$

FO 1999, recap

$$\epsilon^{\mathsf{hy}}_{\mathsf{IND\text{-}CCA}} \leq q_D 2^{-\gamma} + q_H \epsilon^{\mathsf{asym}}_{\mathsf{OW\text{-}CPA}} + \epsilon^{\mathsf{sym}}_{\mathsf{IND\text{-}CPA}}$$

- But it's not a KEM?
- Non-tight security

Hofheinz, Hovelmanns, Kiltz, 2017

"A modular analysis of the Fujisaki-Okamoto transformation"

- Tighter security
- No need for SKE
- IND-CCA KEM
- Used by Kyber and McEliece

Modularity

The transformation happens in two steps

1. OW-CPA/IND-CPA PKE to OW-PCVA PKE
2. OW-PCVA PKE to IND-CCA KEM

What is PCVA?

In addition to CPA, the adversary can access two more oracles:

- **Plaintext checking oracle (PCO)** takes a pair of (m, c) and check if they are valid encryption/decryption of each other
- **Ciphertext validation oracle (CVO)** takes a ciphertext c and checks if it is a valid ciphertext

Vanilla PCO, CVO

The vanilla implementations use the secret key to run the decryption routine

Algorithm 11: $\mathcal{O}^{\mathsf{CVO}}$

Input: $\tilde{c}$

$\hat{m} \leftarrow D(\mathsf{sk}, c)$;

if $\hat{m} = \perp$ **then**

  return $\perp$;

end

if $E(pk, \hat{m}) \neq \tilde{c}$ **then**

  return $\perp$;

end

return 1;

Algorithm 12: $\mathcal{O}^{\mathsf{PCO}}$

Input: $(\tilde{m}, \tilde{c})$

if $D(sk, \tilde{c}) \neq \tilde{m}$ **then**

  return $\perp$;

end

if $E(pk, \tilde{m}) \neq \tilde{c}$ **then**

  return $\perp$;

end

return *1*;

The OW-PCVA transformation (E^T, D^T)

Inputs:

- A PKE (E, D)
- A hash function G

Algorithm 13: E^T

Input: pk, m

$r \leftarrow G(m)$;

$c \leftarrow E(\mathsf{pk}, m, r)$;

return c;

Algorithm 14: D^T

Input: $\mathsf{sk}, \mathsf{pk}, c$

$\hat{m} \leftarrow D(\mathsf{sk}, c)$;

$\hat{r} \leftarrow G(\hat{m})$;

if $E(pk, \hat{m}, \hat{r}) \neq c$ **then**

  return $\perp$;

end

return $\hat{m}$;

OW-PCVA security of (E^T, D^T)

Theorem

For every OW-PCVA adversary against the T-transformation (E^T, D^T) with advantage $\epsilon^T_{\text{OW-PCVA}}$ there exists an IND-CPA adversary against the underlying PKE (E, D) with advantage $\epsilon_{\text{IND-CPA}}$ such that

$$\epsilon^T_{\text{OW-PCVA}} \leq q_V 2^{-\gamma} + q_H \delta + \frac{1}{|\mathcal{M}|} + 3\epsilon_{\text{IND-CPA}}$$

OW-PCVA proof overview

Similar strategy to the FO 1999 proof:

- Modify PCO and CVO so that they don't use secret key
- Simulate OW-PCVA game using an IND-CPA adversary

Modified PCO

Instead of checking both encryption and decryption, check only encryption

Algorithm 15: $\mathcal{O}^{\mathsf{PCO}}$

Input: $(\tilde{m}, \tilde{c})$
if $E(pk, \tilde{m}) \neq \tilde{c}$ **then**
  return $\perp$;
end
if $D(sk, \tilde{c}) \neq \tilde{m}$ **then**
  return $\perp$;
end
return *1*;

Algorithm 16: $\mathcal{O}_1^{\mathsf{PCO}}$

Input: $(\tilde{m}, \tilde{c})$
if $E(pk, \tilde{m}) \neq \tilde{c}$ **then**
  return $\perp$;
end
return *1*;

Modified CVO

Instead of running the decryption routine, check the hash oracle G

Algorithm 17: $\mathcal{O}^{\mathrm{CVO}}$

Input: $\tilde{c}$

$\hat{m} \leftarrow D(\mathrm{sk}, \tilde{c})$;
if $\hat{m} = \perp$ **then**
  return $\perp$;
end
if $E(pk, \hat{m}) \neq \tilde{c}$ **then**
  return $\perp$;
end
return 1;

Algorithm 18: $\mathcal{O}_1^{\mathrm{CVO}}$

Input: $\tilde{c}$

if $\exists (m, r) \in G \ s.t. \ E(pk, m) = \tilde{c}$ **then**
  return 1;
end
return $\perp$

Game 0: OW-PCVA game

Algorithm 19: Game 0

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}()$;

$m^* \xleftarrow{\$} \mathcal{M}$; $c^* \leftarrow E^T(\mathsf{pk}, m^*)$;

$\hat{m} \leftarrow \mathcal{A}^{\mathsf{T}}_{\mathsf{OW\text{-}PCVA}}(\mathsf{pk}, c^*, \mathcal{O}^{\mathsf{PCO}}, \mathcal{O}^{\mathsf{CVO}})$;

$\mathcal{A}^{T}_{\mathsf{OW\text{-}PCVA}}$ wins if $\hat{m} = m^*$

Game 1: modify the PCO

Algorithm 20: Game 0

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}();$

$m^* \xleftarrow{\$} \mathcal{M};\ c^* \leftarrow E^T(\mathsf{pk}, m^*);$

$\hat{m} \leftarrow \mathcal{A}^{\mathsf{T}}_{\mathsf{OW\text{-}PCVA}}(\mathsf{pk}, c^*, \mathcal{O}^{\mathsf{PCO}}_1, \mathcal{O}^{\mathsf{CVO}});$

$\mathcal{A}^{T}_{\mathsf{OW\text{-}PCVA}}$ wins if $\hat{m} = m^*$

Remark

Loss of tightness when decryption error [2] happens:

$$\epsilon_0 - \epsilon_1 \leq q_G \delta$$

[2] A PKE is δ-correct if for some fixed keypair and a randomly sampled m, $P[D(\mathsf{sk}, E(\mathsf{pk}, m)) \neq m] \leq \delta$

Game 2: modify the CVO

Algorithm 21: Game 0

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}()$;
$m^* \xleftarrow{\$} \mathcal{M}$; $c^* \leftarrow E^T(\mathsf{pk}, m^*)$;
$\hat{m} \leftarrow \mathcal{A}^{\mathsf{T}}_{\mathsf{OW\text{-}PCVA}}(\mathsf{pk}, c^*, \mathcal{O}_1^{\mathsf{PCO}}, \mathcal{O}_1^{\mathsf{CVO}})$;
$\mathcal{A}^{T}_{\mathsf{OW\text{-}PCVA}}$ wins if $\hat{m} = m^*$

Remark

Loss of tightness when $\mathcal{A}$ queried some $\tilde{c}$ without querying G

$$\epsilon_1 - \epsilon_2 \leq q_V 2^{-\gamma}$$

Game 3: use a truly random coin

Algorithm 22: Game 0

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}();$

$m^* \xleftarrow{\$} \mathcal{M};\; r^* \xleftarrow{\$} \mathsf{Coin};\ c^* \leftarrow E(\mathsf{pk}, m^*, r^*);$

$\hat{m} \leftarrow \mathcal{A}^{\mathsf{T}}_{\mathsf{OW\text{-}PCVA}}(\mathsf{pk}, c^*, \mathcal{O}_1^{\mathsf{PCO}}, \mathcal{O}_1^{\mathsf{CVO}});$

$\mathcal{A}_{\mathsf{OW\text{-}CPA}}$ wins if $\hat{m} = m^*$

Remark

Loss of tightness when $\mathcal{A}$ queries G on m^*

$$\epsilon_2 - \epsilon_3 \leq P[\mathsf{QUERY}^*]$$

Simulate game 3 with OW-CPA adversary

Game 3 can be perfectly simulated by an OW-CPA adversary against the underlying PKE (E, D):

$$\epsilon_3 = \epsilon_{\text{OW-CPA}}$$

The OW-CPA advantage can be directly translated into IND-CPA advantage with the following "well-known results":

Theorem

For every IND-CPA adversary with advantage $\epsilon_{\text{IND-CPA}}$ there exists an OW-CPA adversary with advantage $\epsilon_{\text{OW-CPA}}$ such that

$$\epsilon_{\text{OW-CPA}} = \frac{1}{|\mathcal{M}|} + \epsilon_{\text{IND-CPA}}$$

Simulate game 3 with IND-CPA adversary

We can build $\mathcal{A}_{\text{IND-CPA}}$ that:

- Sample random m_0, m_1
- Check the hash function tape for matching m_b

Algorithm 23: IND-CPA game against (E, D)

$(\text{pk}, \text{sk}) \xleftarrow{\$} \text{KeyGen}();$

$(m_0, m_1) \xleftarrow{\$} \mathcal{M}^a;$

$b \xleftarrow{\$} \{0, 1\};\ c^* = E(\text{pk}, m_b);$

$\hat{m} \leftarrow \mathcal{A}^T_{\text{OW-PCVA}}(\text{pk}, c^*, \mathcal{O}_1^{\text{PCO}}, \mathcal{O}_1^{\text{CVO}});$

$\hat{b} \leftarrow \text{CheckTape}();$

$\mathcal{A}_{\text{IND-CPA}}$ wins if $\hat{b} = b$;

[a] We omit nuance about sampling m_0, m_1 randomly while making sure that they are distinct

CheckTape()

If $\exists (m, r) \in G$ such that $m = m_0$ or $m = m_1$, then set $\hat{b} = 0$ or $\hat{b} = 1$ accordingly.
If no such record exists, return a blind guess $\hat{b} \xleftarrow{\$} \{0, 1\}$

$$P[\hat{b} = b] = P[\text{QUERY}^*] + (1 - P[\text{QUERY}^*])\frac{1}{2}$$

Which implies

$$\epsilon_{\text{IND-CPA}} = \frac{1}{2} P[\text{QUERY}^*]$$

IND-CCA KEM

	explicit rejection	implicit rejection
PKE is IND-CPA	$U^{\perp}$	$U^{\not\perp}$
PKE is OW-CPA	$U_m^{\perp}$	$U_m^{\not\perp}$

Table: KEM transformations

$U^\perp$ implementation

H is another hash function

Algorithm 24: $U^\perp$ Encap

Input: pk

$m \xleftarrow{\$} \mathcal{M}$;

$c \leftarrow E^T(\mathrm{pk}, m)$;

$K \leftarrow H(m, c)$;

return (c, K);

Algorithm 25: $U^\perp$ Decap

Input: sk, c

Output: Shared secret

$m \leftarrow D^T(\mathrm{sk}, c)$;

if $m = \perp$ **then**

  return $\perp$;

end

return $H(m, c)$;

$U^{\perp}$ security

For every IND-CCA adversary against $U^{\perp}$ with advantage $\epsilon^{U^{\perp}}_{\text{IND-CCA}}$, there exists an OW-PCVA adversary against (E^T, D^T) with advantage $\epsilon^{T}_{\text{OW-PCVA}}$ such that

$$\epsilon^{U^{\perp}}_{\text{IND-CCA}} \leq \epsilon^{T}_{\text{OW-PCVA}}$$

Simulate decapsulation oracle

Goal

If the query $\tilde{c}$ is a valid ciphertext that decrypts to $\tilde{m}$, $\mathcal{O}^D$ should return $H(\tilde{m}, \tilde{c})$

Strategy

- Make both H and $\mathcal{O}^D$ stateful
- Use PCO and CVO to "decrypt" and check integrity

Patched hash and decap oracle

$\mathcal{O}_1^D$ keeps track of past queries $(\tilde{c}, \tilde{K})$

Algorithm 26: H_1

Input: $(\tilde{m}, \tilde{c})$
if $\exists(\tilde{m}, \tilde{c}, \tilde{K}) \in H_1$ **then**
  return $\tilde{K}$;
end
$\tilde{K} \xleftarrow{\$} \{0,1\}^*$;
if $\mathcal{O}^{PCO}(\tilde{m}, \tilde{c}) \neq \perp$ **then**
  Append $(\tilde{c}, \tilde{K})$ to $\mathcal{O}^D$
end
return $\tilde{K}$;

Algorithm 27: $\mathcal{O}_1^D$

Input: $\tilde{c}$
if $(\tilde{c}, \tilde{K}) \in \mathcal{O}_1^D$ **then**
  return $\tilde{K}$;
end
if $\mathcal{O}^{CVO}(\tilde{c}) = \perp$ **then**
  return $\perp$;
end
$\tilde{K} \xleftarrow{\$} \{0,1\}^*$;
Append $(\tilde{c}, \tilde{K})$ to $\mathcal{O}^D$;
return $\tilde{K}$;

Patched oracles behave exactly like their vanilla counterparts

Game 0: KEM IND-CCA

Algorithm 28: Game 0: KEM IND-CCA

$(\mathrm{pk}, \mathrm{sk}) \xleftarrow{\$} \mathrm{KeyGen}()$;

$(c^*, K_0) \xleftarrow{\$} E^{U^\perp}(\mathrm{pk})$; $K_1 \xleftarrow{\$} \{0,1\}^*$;

$b \xleftarrow{\$} \{0,1\}$; $K^* \leftarrow K_b$;

$\hat{b} \xleftarrow{\$} \mathcal{A}^{U^\perp}_{\text{IND-CCA}}(\mathrm{pk}, c^*, K^*, \mathcal{O}^D, H)$;

$\mathcal{A}^{U^\perp}_{\text{IND-CCA}}$ wins if $\hat{b} = b$

Game 1: Use patched oracles

Algorithm 29: Game 1: with patched oracles

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}();$

$(c^*, K_0) \xleftarrow{\$} E^{U^\perp}(\mathsf{pk});\ K_1 \xleftarrow{\$} \{0,1\}^*;$

$b \xleftarrow{\$} \{0,1\};\ K^* \leftarrow K_b;$

$\hat{b} \xleftarrow{\$} \mathcal{A}^{U^\perp}_{\text{IND-CCA}}(\mathsf{pk}, c^*, K^*, \mathcal{O}^D_1, H_1);$

$\mathcal{A}^{U^\perp}_{\text{IND-CCA}}$ wins if $\hat{b} = b$

Remark

There is no difference between game 0 and game 1

$$\epsilon_0 = \epsilon_1$$

Game 2: Use truly random K^*

Algorithm 30: Game 2: unwinnable game

$(\mathrm{pk}, \mathrm{sk}) \xleftarrow{\$} \mathrm{KeyGen}()$;

$(c^*, K_0) \xleftarrow{\$} E^{U^\perp}(\mathrm{pk})$;

$K^* \xleftarrow{\$} \{0,1\}^*$;

$b \xleftarrow{\$} \{0,1\}$;

$\hat{b} \xleftarrow{\$} \mathcal{A}^{U^\perp}_{\text{IND-CCA}}(\mathrm{pk}, c^*, K^*, \mathcal{O}^D_1, H_1)$;

$\mathcal{A}^{U^\perp}_{\text{IND-CCA}}$ wins if $\hat{b} = b$

Remark

Game 2 and game 1 diverge when $\mathcal{A}_{\text{IND-CCA}}$ queries H on (m^*, c^*)

$$\epsilon_1 - \epsilon_2 \leq P[\text{QUERY}^*]$$

Also, game 2 is unwinnable: $\epsilon_2 = 0$

Simulate game 2 with OW-PCVA adversary

Algorithm 31: OW-PCVA game

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}();$

$m^* \xleftarrow{\$} \mathcal{M}$; $c^* \leftarrow E^T(\mathsf{pk}, m^*)$;

$K^* \xleftarrow{\$} \{0,1\}^*$;

$\hat{b} \leftarrow \mathcal{A}^{U^\perp}_{\text{IND-CCA}}(\mathsf{pk}, c^*, K^*, \mathcal{O}^D_1, H_1)$;

$\hat{m} \leftarrow \mathsf{CheckTape}()$;

$\mathcal{A}^T_{\text{OW-PCVA}}$ wins if $\hat{m} = m^*$

Remark

$\mathcal{A}^T_{\text{OW-PCVA}}$ wins if $\mathcal{A}^{U^\perp}_{\text{IND-CCA}}$ queries on m^*

$$P[\text{QUERY}^*] = \epsilon^T_{\text{OW-PCVA}}$$