A survey of IND-CCA constructions

Ganyu (Bruce) Xu (g66xu)

CO 789, Winter 2024

1 Introduction

Indistinguishability under (adaptive) chosen-ciphertext attack (IND-CCA2) is widely recognized as the desirable security notion for public-key cryptography. However, directly achieving IND-CCA2 security is difficult. Previous attempts at deploying public-key cryptography in production, such as the usage of RSA PKCS1 v1.5 in early versions of SSl/TLS, were found to be vulnerable to adaptive chosen-ciphertext attacks.

Instead of directly constructing IND-CCA2 secure cryptosystem from NP-hard problems, recent works approached this problem by proposing generic transformation that take cryptographic primitives of lesser strengths (e.g. OW-CPA, IND-CPA) and produce encryption schemes that lose only a negligible amount of security. One such transformation was proposed by Fujisaki and Okamoto in 1999 and later improved by Hofheinz, Hovelmann, and Kiltz in 2017. With simple construction and robust security reduction, the FO transformation is adopted by submissions to NIST's post-quantum cryptography competition (notably by Kyber, which has been standardized in FIPS 203 in 2024).

In this paper, we will review the constructions of the Fujisaki-Okamoto transformation and its variations. We will also review their security results, including the techniques used in the security reduction. Finally, we will discuss open problems and propose some optimization.

2 Preliminaries

2.1 Spread and correctness

The spread of measures the randomness of ciphertext in a probabilistic encryption scheme, where the randomness of the encryption routine is determined by a random coin $r \xleftarrow{\$} \text{Coin}$ from the coin space.

Definition 2.1 (γ-spread)**.** *The spread γ of an asymmetric encryption scheme* (KeyGen, E, D) *for some fixed public key pk and plaintext m is defined by*

$$\gamma(pk, m) = -\log \|E_{pk}(m)\|_\infty$$

Where $\|E_{pk}(m)\|_\infty$ is the min-entropy of the ciphertext with respect to the coin:

$$\|E_{pk}(m)\|_\infty = \max_{c \in \mathcal{C}} P[E_{pk}(m) = c]$$

Definition 2.2 (correctness)**.** *A public-key encrpytion scheme* (KeyGen, E, D) *is δ-correct if across all possible keypairs and messages, the probability of decryption error is at most δ:*

$$\max_{m \in \mathcal{M}, (pk, sk) \xleftarrow{\$} \text{KeyGen}()} P[D_{sk}(E_{pk}(m)) \neq m] \leq \delta$$

2.2 Difference lemma

The difference lemma is used extensively in the sequence of games to estimate the loss of security when one game is transformed into another

Lemma 1 (Difference lemma)**.** *Let A, B, F be events defined on the same probability space. If $P[A \cap \neg F] = P[B \cap \neg F]$, then $P[A] - P[B] \leq P[F]$*

Proof.

$$\begin{aligned}
P[A] - P[B] &= (P[A \cap F] + P[A \cap \neg F]) - (P[B \cap F] + P[B \cap \neg F]) \\
&= (P[A \cap F] - P[B \cap F]) + (P[A \cap \neg F] - P[B \cap \neg F]) \\
&= P[A \cap F] - P[B \cap F] \\
&\leq P[A \cap F] \\
&\leq P[F]
\end{aligned}$$

□

In the context of security reduction using sequence of game, the difference lemma is invoked with event A and B set to "adversary winning one game" and "adversary winning another game", and event F set to the event that would cause the two games two diverge. If the diverging event F does not happen, then the two games are identical to each other, so the advantage of the adversary remains the same across the two games; if the diverging event F happens, then the advantage of the adversary does not change by more than the probability of the diverging event F.

2.3 IND-CPA to OW-CPA

It is trivially true that if a public-key encryption scheme is not one-way secure, then it is not indistinguishable. The converse if also almost true, with the caveat that the message space must be sufficiently large:

Lemma 2. *Let* (KeyGen, E, D) *be a public-key encryption scheme with a finite message space $\mathcal{M}$. For every IND-CPA adversary A, there exists an OW-CPA adversary B such that*

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

3 Related works

Key encapsulation mechanisms (KEM) usually have weaker security requirements than public-key encryption schemes (PKE) thanks to the fact that properly designed KEMs obfuscate the plaintext through a key-derivation function (which is usually a hash function and thus a random oracle under the RO assumption) in the decryption routine. As a result, we can implement a simple IND-CCA2 KEM using an IND-CCA2 PKE:

Algorithm 1: KeyGen

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

Algorithm 2: Encap

```
Input: pk
$m \xleftarrow{\$} \mathcal{M}$;
$c \xleftarrow{\$} E(\text{pk}, m)$;
$K \leftarrow H(m, c)$;
return $(c, K)$;
```

Algorithm 3: Decap

```
Input: $\text{sk}, c \in \mathcal{C}$
$\hat{m} \leftarrow D(\text{sk}, c)$;
$\hat{K} \leftarrow H(\hat{m}, c)$;
return $\hat{K}$;
```

It is straightforward to simulate the KEM game using a PKE adversary. The PKE adversary samples two random messages $(m_0, m_1) \xleftarrow{\$} \mathcal{M}$ and obtains the challenge encryption for one of them $c^* \leftarrow E(\text{pk}, m_b)$. The PKE adversary then computes $K^* \leftarrow H(c^*, m_0)$ and passes (c^*, K^*) as the challenge encapsulation to the KEM adversary. If c^* is an encryption of m_0, then K^* is pseudorandom, otherwise under the random oralce model, K^* is truly random from the perspective of the KEM adversary. Therefore, $\epsilon_{\text{PKE}} = \epsilon_{\text{KEM}}$.

Using IND-CCA2 PKE as an IND-CCA2 KEM has been in production for a long time. For example, TLS 1.2 [2] offers this implementation for key exchange using RSA-OAEP[1] as the underlying IND-CCA2 PKE. The key exchange protocol proceeds as follows:

1. Server sends its public key through the X.509 certificate
2. Client samples a master secret, encrypts the secret using server's public key, then sends the ciphertext to the server
3. Server decrypts the ciphertext to obtain the master secret

This approach to key exchange has since been deprecated, and RSA is no longer supported for key exchange starting with TLS 1.3. In addition to the difficulty of implementing RSA correctly and securely (RSA is notoriously hard to implement without side-channel vulnerabilities), using the server's long-term keypair for establishing common secrets fails to provide forward secrecy, meaning that if the long-term key is compromised, all prior communications can be trivially decrypted.

While the modern approach is to use ephemeral Diffie-Hellman (with either prime field or elliptic curve), it is difficult to adapt this approach to the post-quantum setting. Instead, post-quantum key exchange usually builds an IND-CCA2 KEM, then perform the key exchange using ephemeral keypairs. At each session's handshake, the client generates the a new keypair and sends the public key to the server. The server uses the public key to run the encapsulation routine and sends back the ciphertext. Finally, the client runs the decapsulation routine on the ciphertext to recover the shared secret.

4 Fujisaki-Okamoto transformation and variations

In 1999, Fujisaki and Okamoto [3] proposed a generic transformation that outputs an IND-CCA2-secure public-key encryption scheme using an IND-CPA symmetric cipher, an OW-CPA asymmetric cipher, and two hash functions. Later on in 2017, Hofheinz, Hovelmann, and Kiltz [4] improved on the security of the Fujisaki-Transformation and adapted the original approach to constructing IND-CCA2 KEMs. In this section, we will give a quick overview of the individual transformations and their security results. We will then discuss the techniques used in the security reduction. Finally, we will provide a sketch of the actual proof.

4.1 An overview of transformed routines and security results

The original 1999 Fujisaki-Okamoto transformation takes as input a symmetric cipher $(E^{\text{sym}}, D^{\text{sym}})$, an asymmetric cipher $(\text{KeyGen}, E^{\text{asym}}, D^{\text{asym}})$, a key derivation function (aka a hash function) $G : \mathcal{M}^{\text{asym}} \to \mathcal{K}^{\text{sym}}$, and a hash functino $H : \{0,1\}^* \to \text{Coin}^{\text{asym}}$.

Specifically, the asymmetric cipher is assumed to be probabilistic, but with pseudorandomness seeded on some coin $r \in \text{Coin}^{\text{asym}}$. When E^{asym} is called without specifying a value for coin, it is assumed that a coin is uniformly sampled from the coin space and passed in as the pseudorandom seed:

$$c \xleftarrow{\$} E^{\text{asym}}(\text{pk}, m) \Leftrightarrow c \leftarrow E^{\text{asym}}(\text{pk}, m, r \xleftarrow{\$} \text{Coin})$$

The output of the transformation is a public-key encryption scheme $(\text{KeyGen}, E^{\text{hy}}, D^{\text{hy}})$. Note that the key generation routine is exactly the same as the input asymmetric cipher's routine, so no distinction of notation is needed. The encryption and decryption routines are as follows:

```
Algorithm 4: Hybrid encryption $E^{\text{hy}}$
  Input: $\text{pk}, m \in \mathcal{M}^{\text{sym}}$
  $\sigma \xleftarrow{\$} \mathcal{M}^{\text{asym}}$;
  $a \leftarrow G(\sigma)$; $c \leftarrow E^{\text{sym}}_a(m)$;
  $r \leftarrow H(\sigma, c)$; $e \leftarrow E^{\text{asym}}(\text{pk}, m, r)$;
  return $(c, e)$;
```

```
Algorithm 5: Hybrid decryption $D^{\text{hy}}$
  Input: $\text{sk}, c \in \mathcal{C}^{\text{sym}}, e \in \mathcal{C}^{\text{asym}}$
  $\hat{\sigma} \leftarrow D^{\text{asym}}(\text{sk}, e)$;
  $\hat{r} \leftarrow H(\hat{\sigma}, c)$;
  if $E^{asym}(pk, \hat{\sigma}, \hat{r}) \neq e$ then
      return $\perp$; // the "re-encryption"
  end
  $\hat{a} \leftarrow G(\hat{\sigma})$;
  $\hat{m} \leftarrow D^{\text{sym}}_{\hat{a}}(c)$;
  return $\hat{m}$;
```

Theorem 4.1 (Security result for $(E^{\text{hy}}, D^{\text{hy}})$)**.** *If the input asymmetric cipher has γ spread, then for every IND-CPA adversary $\mathcal{A}^{sym}_{IND\text{-}CPA}$ against the underlying symmetric cipher with advantage ϵ^{sym} and OW-CPA adversary $\mathcal{A}^{asym}_{OW\text{-}CPA}$ against the input asymmetric cipher with advantage ϵ^{asym}, there exists an IND-CCA2 adversary with against the hybrid scheme making q_H hash queries and q_D decryption queries such that:*

$$\epsilon^{hy} \leq q_D 2^{-\gamma} + \epsilon^{sym} + q_H \epsilon^{asym}$$

Hofheinz, Hovenmann, and Kiltz pointed out in their 2017 paper a number of drawbacks of the original FO transformation:

1. Decryption error, which is especially possible with lattice-based schemes, is not accounted for
2. the $q_H \epsilon^{\text{asym}}$ term causes security to be non-tight, which have implications when choosing the security parameters

In their proposals, IND-CCA2 security is achieved through a two-step transformation. First, an OW-CPA asymmetric scheme is transformed into an OW-PCVA asymmetric scheme. The OW-PCVA scheme is then transformed into an IND-CCA2 KEM. Here PCVA refers to the adversary's ability to access some plaintext-checking oracle and some ciphertext validation oracle. Although PCVA is a non-standard security definition, it is entirely contained within the two-step transformation and serves only as an intermediary step.

Neither the OW-PCVA nor the IND-CCA transformation makes use of any symmetric cipher, so we ditch the distinction. The OW-PCVA transformation takes as input an asymmetric cipher (E, D) and a hash function $G : \mathcal{M} \to \text{Coin}$. The transformed encryption and decryption routines (E^T, D^T) are as follows:

```
Algorithm 6: OW-PCVA $E^T$
  Input: pk, $m \in \mathcal{M}$
$r \leftarrow G(m)$;
$c \leftarrow E(\text{pk}, m, r)$;
return $c$;
```

```
Algorithm 7: OW-PCVA $D^T$
  Input: sk, $c \in \mathcal{C}$
$\hat{m} \leftarrow D(\text{sk}, c)$;
$\hat{r} \leftarrow G(\hat{m})$;
if $E(pk, \hat{m}, \hat{r}) \neq c$ then
    return $\perp$;
end
return $\hat{m}$;
```

Depending on whether the implementation chooses implicit or explicit rejection of invalid ciphertext for the final KEM, the second transformation has slightly different key generation and decryption routine. The explicit rejection KEM is denoted by $U^{\perp}$, and its key generation is identical to the input asymmetric cipher.

```
Algorithm 8: $U^{\perp}$ Encap
  Input: pk
$m \xleftarrow{\$} \mathcal{M}$;
$c \leftarrow E^T(\text{pk}, m)$;
$K \leftarrow H(m, c)$ // H is a hash function;
return $(c, K)$
```

```
Algorithm 9: $U^{\perp}$ Decap
  Input: $c \in \mathcal{C}$, sk
$\hat{m} \leftarrow D^T(\text{sk}, c)$;
if $m = \perp$ then
    return $\perp$;
end
return $H(\hat{m}, c)$
```

On the other hand, the implicit rejection KEM, denoted by $U^{\not\perp}$, generate a random message $m' \xleftarrow{\$} \mathcal{M}$ as the fake input to H when the ciphertext is invalid

```
Algorithm 10: KeyGen
pk, sk $\xleftarrow{\$}$ KeyGen();
$m' \xleftarrow{\$} \mathcal{M}$;
return $(pk, (sk, m'))$;
```

```
Algorithm 11: Encap
$m \xleftarrow{\$} \mathcal{M}$;
$c \leftarrow E^T(\text{pk}, m)$;
$K \leftarrow H(m, c)$;
return $(c, K)$
```

```
Algorithm 12: Decap
$\hat{m} \leftarrow D^T(\text{sk}, c)$;
if $m = \perp$ then
    return $H(m', c)$;
end
return $H(\hat{m}, c)$
```

The security result is similarly expressed in two theorems:

Theorem 4.2. *If the input asymmetric scheme is δ-correct and γ-spread, then for every OW-PCVA adversary that issues at most q_G hash queries and q_V ciphertext validation queries with advantage ϵ^T, there exists an IND-CPA adversary against the underlying asymmetric scheme with advantage ϵ such that*

$$\epsilon^T \leq q_G \cdot \delta + q_V \cdot 2^{-\gamma} + \frac{2q_G + 1}{|\mathcal{M}|} + 3\epsilon$$

Theorem 4.3. *For every IND-CCA2 adversary against the $U^{\perp}$ KEM with advantage $\epsilon^{U^{\perp}}$, there exists an OW-PCVA adversary against the input scheme (E^T, D^T) with advantage ϵ^T such that*

$$\epsilon^{U^{\perp}} \leq \epsilon^T$$

Theorem 4.4. *For every IND-CCA2 adversary against the $U^{\not\perp}$ KEM that issues q_H hash queries with advantage $\epsilon^{U^{\not\perp}}$, there exists an OW-PCVA adversary against the input scheme (E^T, D^T) with advantage ϵ^T such that*

$$\epsilon^{U^{\not\perp}} \leq \epsilon^T + \frac{q_H}{|\mathcal{M}|}$$

4.2 A discussion of techniques

4.3 A sketch of proof

5 Future works on generic IND-CCA transformation

References

[1] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology—EUROCRYPT'94: Workshop on the Theory and Application of Cryptographic Techniques Perugia, Italy, May 9–12, 1994 Proceedings 13*, pages 92–111. Springer, 1995.

[2] Tim Dierks and Eric Rescorla. Rfc 5246: The transport layer security (tls) protocol version 1.2, 2008.

[3] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual international cryptology conference*, pages 537–554. Springer, 1999.

[4] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the fujisaki-okamoto transformation. In *Theory of Cryptography Conference*, pages 341–371. Springer, 2017.