

Clase 18: cifrado híbrido, la metodología KEM-DEM

Fernando Virdia, versión: 0.1, mayo 2025

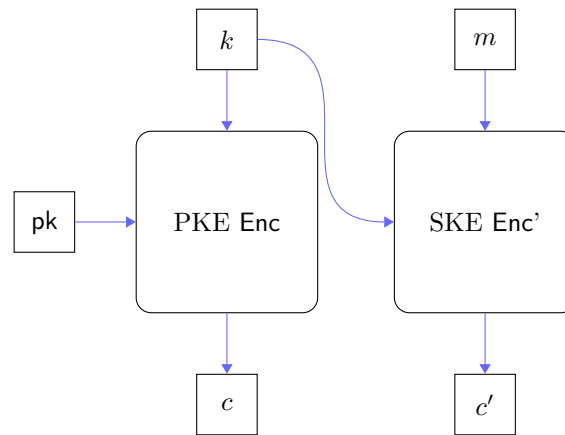
9.2 Cifrado híbrido

Es interesante observar que en practica, el costo de un cifrado a clave publica es generalmente mas alto del de un cifrado simétrico. Lanzando `openssl speed rsa aes` en mi laptop:

- En 10s, 6852 descifrados y 344141 cifrados de RSA-3072 (“128 bits de seguridad”), mensaje de ≈ 3000 bits.
- En 10s, 941746 cifrados o descifrados de AES-128 CBC-mode, mensaje de ≈ 131072 bits.

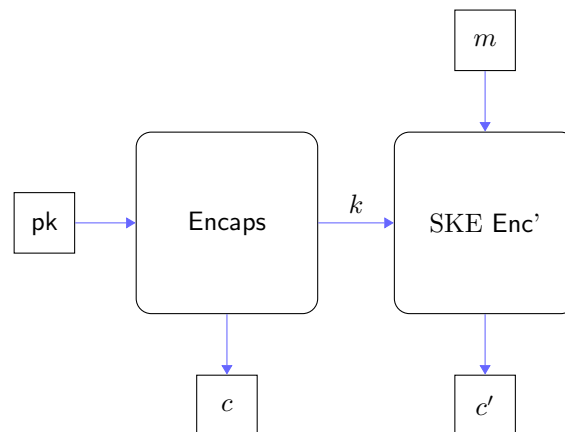
Por este motivo, aunque es técnicamente posible usar sistemas como ElGamal para mensajes arbitrariamente largos, En la practica PKE y KEX ambos utilizan una primera fase a clave publica para generar claves para un SKES. En el caso de PKE, este método se llama de cifrado híbrido (“hybrid encryption”).

Por ejemplo, sean $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ un PKES, $\Pi' = (\text{Gen}', \text{Enc}', \text{Dec}')$ un SKES. Podemos cifrar una clave k de Π' utilizando Π , y luego cifrar un mensaje arbitrariamente largo utilizando $\text{Enc}'(k, \cdot)$. Transmitiendo ambos cifrados, es posible recuperar el mensaje.



Una técnica hoy mas común es remplazar el PKES con un mecanismo de encapsulamiento de claves (KEM, “key-encapsulation mechanism”).

Se puede pensar un KEM como un sistema de cifrado PKES de “mensajes al azar”. Esta restricción generalmente le permite a un KEM ser mas compacto y eficiente que un PKES que ofrece similar seguridad.



Un sistema híbrido que utiliza un KEM se lo llama también KEM-DEM, donde el cifrado simétrico tiene el rol de mecanismo de encapsulamiento de datos (DEM, “data-encapsulation mechanism”). Para obtener PKE que otorgue seguridad IND-CCA, vamos a utilizar una construcción KEM-DEM.

Definición 36 (KEM, mecanismo de encapsulado de claves). Sean $\mathcal{K}_p$, $\mathcal{K}_s$, $\mathcal{M}$, $\mathcal{C}$ conjuntos. Decimos que $\Pi = (\text{Gen}, \text{Encaps}, \text{Decaps})$ forman un mecanismo de encapsulamiento de claves (“key-encapsulation mechanism”, KEM) con espacio de claves publicas $\mathcal{K}_p$, de claves privadas $\mathcal{K}_s$, de claves a encapsular $\mathcal{M}$, y de cifrados o “encapsulados” $\mathcal{C}$, si

- Gen es un algoritmo aleatorio, $\text{Gen}: \emptyset \rightarrow \mathcal{K}_s \times \mathcal{K}_p$ que retorna una tupla (sk, pk) ;
- Encaps es un algoritmo aleatorio, $\text{Encaps}: \mathcal{K}_p \rightarrow \mathcal{M} \times \mathcal{C}$ que retorna una clave k y un encapsulamiento c de k ;
- Decaps es un algoritmo determinista $\text{Dec}: \mathcal{K}_s \times \mathcal{C} \rightarrow \mathcal{M}$, que decapsula c , retornando k ;

donde Gen , Encaps y Decaps son “eficientes”. Decimos que Π es δ -correcto si

$$\Pr_{\substack{(sk, pk) \sim \text{Gen}(), \\ \$ \text{ en } \text{Encaps}}}[(k, c) \leftarrow \text{Encaps}(pk, \$) : \text{Decaps}(sk, c) \neq k] \leq \delta.$$

Comentario 59. En el resto del curso, consideramos solamente KEMs (0-)correctos.

Ahora podemos definir una noción de “KEM IND-CCA”.

Definición 37 (OT-IND-CCA KEM). Sea $\Pi = (\text{Gen}, \text{Encaps}, \text{Decaps})$ un KEM.

$\text{Exp}^{\text{KEM-OT-CCA}}(\mathcal{A}, b)$	$D(c)$
1: $(\text{sk}, \text{pk}) \leftarrow \text{Gen}()$	1: if $c = c^*$:
2: $(k_0, c^*) \leftarrow \text{Encaps}(pk)$	2: return $\perp$
3: $k_1 \xleftarrow{\$} \mathcal{M}$	3: $k \leftarrow \text{Decaps}(\text{sk}, c)$
4: $b' \leftarrow \mathcal{A}^D(pk, k_b, c^*)$	4: return k
5: return b'	

Decimos que Π otorga “one-time” (ε, t, q_d) -indistinguibilidad bajo ataques de cifrado elegido (KEM IND-CCA, “indistinguishability under chosen-ciphertext attacks”), si cualquier adversario $\mathcal{A}$ que corre en tiempo $\leq t$, y realiza $\leq q_d$ queries a D , tiene ventaja

$$\text{Adv}(\text{Exp}^{\text{KEM-OT-CCA}}, \mathcal{A}) := \left| \Pr[\text{Exp}^{\text{KEM-OT-CCA}}(\mathcal{A}, 0) \Rightarrow 1] - \Pr[\text{Exp}^{\text{KEM-OT-CCA}}(\mathcal{A}, 1) \Rightarrow 1] \right| \leq \varepsilon.$$

Utilizando un KEM que otorga OT-IND-CCA KEM y un DEM que otorga AE, es posible obtener un PKE que otorga OT-IND-CCA, y por lo tanto IND-CCA (c.f. Lemma 18).

Teorema 6 (OT-IND-CCA KEM + AE $\Rightarrow$ OT-IND-CCA PKE, informal). Sea $\Pi = (\text{Gen}, \text{Encaps}, \text{Decaps})$ un KEM, y $\Pi' = (\text{Gen}', \text{Enc}', \text{Dec}')$ un SKES, tales que el espacio $\Pi.\mathcal{M}$ de claves a encapsular de Π es igual al espacio de claves $\Pi'.\mathcal{K}$ de Π' , $\Pi.\mathcal{M} = \Pi'.\mathcal{K}$, y que $\text{Gen}'() = “k \xleftarrow{\$} \Pi'.\mathcal{K}”$. Sea $\Pi^{hy} = (\text{Gen}^{hy}, \text{Enc}^{hy}, \text{Dec}^{hy})$ el PKES obtenido de manera KEM-DEM,

$\text{Gen}^{hy}()$	$\text{Enc}^{hy}(pk, m)$	$\text{Dec}^{hy}(\text{sk}, c = (c_0, c_1))$
1: $(\text{sk}, \text{pk}) \leftarrow \text{Gen}()$	1: $k, c_0 \leftarrow \text{Encaps}(pk)$	1: $k \leftarrow \text{Decaps}(\text{sk}, c_0)$
2: return (sk, pk)	2: $c_1 \leftarrow \text{Enc}'(k, m)$	2: if $k = \perp$:
	3: $c \leftarrow (c_0, c_1)$	3: return $\perp$
	4: return c	4: $m' \leftarrow \text{Dec}'(k, c_1)$
		5: return m'

Si Π otorga KEM OT-IND-CCA y Π' otorga AE, entonces Π^{hy} otorga PKE OT-IND-CCA.

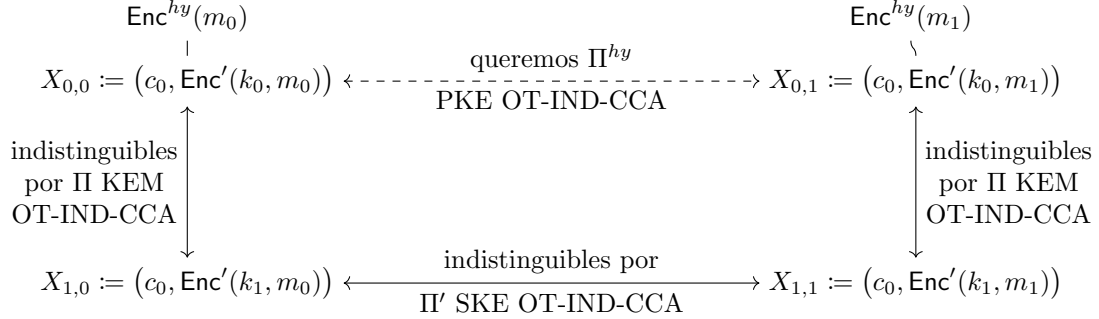
Boceto de demostración.

Para demostrar el teorema, los pasos son

1. demostrar que AE $\Rightarrow$ una noción simétrica de OT-IND-CCA.
2. demostrar que Π^{hy} otorga PKE OT-IND-CCA,.

Una demostración de Item 1 se encuentra en [BS23, Theorem 9.1]. Para demostrar Item 2, la idea es de asumir un adversario $\mathcal{A}$ que juega una variante Exp de $\text{Exp}^{\text{OT-IND-CCA}}$ contra Π^{hy} con dos parámetros, $\text{Exp}(\mathcal{A}, b, \hat{b})$. $\mathcal{A}$ tiene que adivinar b dado un cifrado

$$(c_0, \text{Enc}'(k_{\hat{b}}, m_b)), \text{ donde } (k_0, c_0) \leftarrow \text{Encaps}(\text{pk}) \text{ y } k_1 \xleftarrow{\$} \Pi.\mathcal{K} = \Pi'.\mathcal{K}.$$



Notamos que:

Si $\hat{b} = 0$, entonces $(c_0, \text{Enc}'(k_0, m_b)) = \text{Enc}^{hy}(m_b)$, por lo que para demostrar el resultado, queremos una cota superior para la distancia entre los $\text{Exp}(\mathcal{A}, 0, 0)$ y $\text{Exp}(\mathcal{A}, 1, 0)$.

$X_{b,\hat{b}}$ es una función de la variable aleatoria $Y_{\hat{b}} := (c_0, k_{\hat{b}})$ (la función es $f_{m_b}(x, y) \mapsto (x, \text{Enc}'(y, m_b))$).

Por Π ser un IND-CCA KEM, $Y_{b,0}$ y $Y_{b,1}$ son indistinguibles. Por *data processing inequality* (Theorem 2), también lo tienen que ser $X_{b,0}$ y $X_{b,1}$.

$\text{Enc}'(k_1, m_0)$ y $\text{Enc}'(k_1, m_1)$ son indistinguibles por ser Π' un OT-IND-CCA SKES.

La semejanza de $X_{0,0}$ con $X_{0,1}$, de $X_{0,1}$ con $X_{1,1}$ y $X_{1,1}$ con $X_{1,0}$ nos da el resultado. $\square$

9.3 Transformada de Fujisaki-Okamoto

Nos falta una sola componente: un KEM que otorgue OT-IND-CCA.

Para construir un KEM OT-IND-CCA, vamos a utilizar una variante de la “transformada de Fujisaki-Okamoto” (FO, “Fujisaki-Okamoto transform”). Esta nos permite combinar un PKES que otorga IND-CPA, con dos funciones de hash, obteniendo un KEM OT-IND-CCA!

Hay muchas variantes de FO [HHK17], y esta no debe considerarse la única manera o la manera “correcta”.

Queremos un KEM que resista ataques bajo queries de decapsulado. La idea es volver las queries de decapsulado inútiles. Supongamos que $\mathcal{A}$ pide una query $D(c)$ al oráculo D de decapsulado. Puede haber generado c de diferentes maneras. Llamando $c \leftarrow \text{Encaps}(\text{pk})$, que internamente llama $\text{Enc}(\text{pk}, m)$. En este caso el oráculo D es inútil, $\mathcal{A}$ conoce m ! Modificando algún $c \mapsto c + \Delta$ en tránsito “inventándose” algún $c \in \mathcal{C}$. Vamos a diseñar un PKE donde “inventarse” $c \in \mathcal{C}$ tal que $\text{Dec}(c) \neq \perp$ sea difícil, y tal que modificaciones tipo $c + \Delta$ sean detectables.

Definición 38 (PKE Rígido). Sea $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ un PKES. Decimos que Π es rígido si $\forall(\text{sk}, \text{pk}) \leftarrow \text{Gen}(), \forall c \in \mathcal{C}$

$$\text{Dec}(\text{sk}, c) = \perp \vee \text{Enc}(\text{pk}, \text{Dec}(\text{sk}, c)) = c.$$

Comentario 60. Para que un PKE Π sea rígido, necesita ser determinista! De otra manera, el valor de $\text{Enc}(\text{pk}, \text{Dec}(\text{sk}, c))$ sería una variable aleatoria.

- Podemos obtener un PKE determinista usando cifrando aleatorio $\text{Enc}(\text{pk}, m; r)$ donde seteamos las monedas $r = G(m)$, con $G: \mathcal{M} \rightarrow \mathcal{R}$ una función de hash criptográfica.
- Al ser determinista, Π no puede otorgar PKE IND-CPA. En su lugar, vamos a suponer que Π otorga una noción más débil, one-wayness: dados $\text{pk}, \text{Enc}(\text{pk}, m)$, recuperar m es difícil.

Lemma 20. Sea Π un PKE rígido y correcto. Entonces $c = \text{Enc}(\text{pk}, m) \iff \text{Dec}(\text{sk}, c) = m$.

Demostración. Supongamos que $c = \text{Enc}(\text{pk}, m)$. Por correctness, $\text{Dec}(\text{sk}, c) = m$. Supongamos $\text{Dec}(\text{sk}, c) = m$. Por rigidez, $\text{Enc}(\text{pk}, \text{Dec}(\text{sk}, c)) = \text{Enc}(\text{pk}, m) = c$. $\square$

Definición 39 (Un PKE rigido). Sea $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ un PKES. Sea $G: \mathcal{M} \rightarrow \mathcal{R}$ una función de hash criptográfica. Construimos un PKES rigido $\Pi' = T[\Pi, G] = (\text{Gen}', \text{Enc}', \text{Dec}')$ donde

$\text{Gen}'()$	$\text{Enc}'(\text{pk}, m)$	$\text{Dec}'(\text{sk} = (\text{sk}_\Pi, \text{pk}), c)$
1: $(\text{sk}_\Pi, \text{pk}) \leftarrow \text{Gen}'()$	1: $c \leftarrow \text{Enc}(\text{pk}, m; G(m))$	1: $m' \leftarrow \text{Dec}(\text{sk}_\Pi, c)$
2: $\text{sk} \leftarrow (\text{sk}_\Pi, \text{pk})$	2: return c	2: if $m' = \perp$ or $\text{Enc}(\text{pk}, m'; G(m')) \neq c$:
3: return (sk, pk)		3: return $\perp$
		4: return m'

Lemma 21. El PKES en Definición 39 es rigido.

Demostración. Por definición de Dec' , $m' = \text{Dec}(\text{sk}, c)$ es retornado si y solo si $\text{Enc}'(m') = c$ o $m' = \perp$. Por lo tanto $\text{Enc}'(\text{pk}, \text{Dec}'(\text{sk}, c)) = c \vee \text{Dec}'(\text{sk}, c) = \perp$. $\square$

Con un PKE rigido, podemos definir el KEM de Fujisaki-Okamoto.

Definición 40 (FO KEM). Sea $\mathcal{M} = \{0, 1\}^\ell$ un conjunto finito. Sean $\Pi^{\text{Rig}} = (\text{Gen}^{\text{Rig}}, \text{Enc}^{\text{Rig}}, \text{Dec}^{\text{Rig}})$ un PKES rigido. Sea $H: \Pi^{\text{Rig}} \cdot \mathcal{M} \rightarrow \mathcal{M}$ una función de hash criptográfica. Definimos el KEM $\Pi' = U_m^\perp[\Pi^{\text{Rig}}, H] = (\text{Gen}, \text{Encaps}, \text{Decaps})$ con espacio de claves a encapsular $\mathcal{M}$,

$\text{Gen}()$	$\text{Encaps}(\text{pk})$	$\text{Decaps}(\text{sk}, c)$
1: $(\text{sk}, \text{pk}) \leftarrow \text{Gen}^{\text{Rig}}()$	1: $m \xleftarrow{\$} \Pi^{\text{Rig}} \cdot \mathcal{M}$	1: $m' := \text{Dec}^{\text{Rig}}(\text{sk}, c)$
2: return (sk, pk)	2: $c := \text{Enc}^{\text{Rig}}(\text{pk}, m)$	2: if $m' = \perp$:
	3: $k := H(m)$	3: return $\perp$
	4: return (k, c)	4: $k := H(m')$
		5: return k

Comentario 61. Notamos la necesidad de usar una función de hash $H: \Pi \cdot \mathcal{M} \rightarrow \mathcal{M}$ dentro del KEM de Fujisaki-Okamoto. Esto es porque dado que Π es rigido, también es determinista. El juego KEM IND-CCA le otorga al adversario un encapsulado $c := \text{Enc}^{\text{Rig}}(\text{pk}, m)$ y una clave k que puede ser $H(m)$ o aleatoria. Si usáramos m directamente como la clave k , entonces $\mathcal{A}$ podría testear distinguir k como no-aleatoria simplemente verificando si $c = \text{Enc}^{\text{Rig}}(\text{pk}, m)$ o no.

Para que el KEM FO sea IND-CCA, vamos a necesitar construir nuestro PKE rigido a partir de un PKE que otorga “alta entropía” de cifrado, o “ γ -spreadness”.

Definición 41 (γ -spreadness). Sea $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ un PKES con cifrado no determinista. Decimos que Π es “ γ -spread” si $\forall (\text{sk}, \text{pk}) \leftarrow \text{Gen}(), m \in \mathcal{M}, c \in \mathcal{C}$,

$$\Pr[\text{Enc}(\text{pk}, m) = c] \leq 2^{-\gamma},$$

donde la probabilidad es sobre las monedas de Enc .

Comentario 62. La mayoría de los IND-CPA PKES otorgan spreadness. Si no es así, cualquier IND-CPA PKE puede ser modificado para otorgar spreadness de manera sencilla [BS23, Exercise 12.10]. ElGamal PKE es naturalmente γ -spread con $\gamma = 1/|\mathbb{G}|$.

Teorema 7 (OW correct γ -spread PKE + ROM $\Rightarrow$ OT-IND-CCA KEM, informal). Sea $\mathcal{M} = \{0, 1\}^\ell$. Sea Π un PKES non-determinista correcto y γ -spread. Sea $G: \Pi \cdot \mathcal{M} \rightarrow \Pi \cdot \mathcal{R}$ una función de hash criptográfica. Sea $H: \Pi \cdot \mathcal{M} \rightarrow \mathcal{M}$ una función de hash criptográfica independiente de G . Sea $\Pi^{\text{Rig}} = T[\Pi, G]$ el PKE rigido definido en Definición 39. Sea $\Pi' = U_m^\perp[\Pi^{\text{Rig}}, H]$ el KEM definido en Definición 40. Si Π^{Rig} es one-way (OW), y modelamos G y H como oráculos aleatorios, entonces Π' es OT-IND-CCA.

Boceto de demostración. Una demostración formal de esta variante FO puede obtenerse combinando Teoremas 3.2 y 3.3 en [HHK17], o [HHMS25].

La idea es la siguiente. Sea $\mathcal{A}$ un adversario KEM OT-IND-CCA. Construimos un KEM donde intuitivamente un oráculo de decapsulado no le sirve, en cuanto el PKE utilizado internamente es rigido. En la demostración, vamos a construir un adversario $\mathcal{B}$ contra el OW PKE. $\mathcal{B}$ va a ejecutar $\mathcal{A}$, y cada vez que $\mathcal{A}$ pide una query al oráculo de decapsulado, $\mathcal{B}$ va a generar la respuesta *sin tener*

acceso a sk . Sea $\Pi^{\text{Rig}} = (\text{Gen}^{\text{Rig}}, \text{Enc}^{\text{Rig}}, \text{Dec}^{\text{Rig}})$ el PKE rigido. Dada una query $D(c)$, sabemos que $\text{Dec}^{\text{Rig}}(\text{sk}, c) = \perp$, o $\text{Enc}^{\text{Rig}}(\text{pk}, \text{Dec}^{\text{Rig}}(\text{sk}, c)) = c$.

Si c fue generado usando $\text{Encaps}(\text{pk})$, internamente Enc^{Rig} ha computado $G(m)$ como monedas para el cifrado. Dado que G es un oraculo aleatorio, en lugar de utilizar la funcion de hash, $\mathcal{A}$ calcula $G(m)$ a traves de una query a una “API” que implementa G . $\mathcal{B}$ va a inspeccionar cada query hecha por $\mathcal{A}$ a G , y va a guardarla en una hash-map Q_G indexada con por $c := \text{Enc}^{\text{Rig}}(\text{pk}, m)$: $Q_G[c] = m$. Si $\mathcal{A}$ pide $D(c)$, $\mathcal{B}$ entonces puede retornar $Q_G[c] = m = \text{Dec}^{\text{Rig}}(\text{sk}, c)$ correctamente, sin conocer sk . Si $\mathcal{A}$ no ha generado c a traves de $\text{Encaps}(\text{pk})$, supongamos que $\text{Dec}^{\text{Rig}}(\text{sk}, c) = m' \neq \perp$. Por γ -spreadness, $\Pr[\text{Enc}(\text{pk}, m') = c] \leq 2^{-\gamma}$. Es improbable ($p \leq 2^{-\gamma}$) generar el cifrado $c = \text{Enc}^{\text{Rig}}(\text{pk}, m')$ de m' que descifra correctamente bajo Dec^{Rig} , sin haber calculado $G(m')$. Por rigidez y correctness, $m' = \text{Dec}^{\text{Rig}}(\text{sk}, c) \Leftrightarrow c = \text{Enc}^{\text{Rig}}(\text{pk}, m')$, por lo que m' va a estar guardado bajo el correcto $Q_G[c]$, y no bajo otro cifrado. Por lo cual nuevamente $\mathcal{B}$ puede responder $D(c)$ utilizando $Q_G[c]$, fallando encontrar $Q_G[c]$ con probabilidad $\leq 2^{-\gamma}$. Si finalmente Q_G no contiene un index c , entonces $\mathcal{B}$ responde $D(c)$ con $\perp$.

Al recibir un challenge (pk, c^*) , $\mathcal{B}$ va a generar $k_1 \leftarrow U(\mathcal{M})$, y ejecutar $\mathcal{A}(\text{pk}, k_1, c^*)$. La simulacion de D es correcta con probabilidad alta ($\geq 1 - 2^{-\gamma}$), por lo tanto $\mathcal{A}$ va a lograr distinguir que $k_1 \neq H(m^* := \text{Dec}^{\text{Rig}}(\text{sk}, c^*))$. Pero al ser H un oraculo aleatorio, la distribucion de k_1 y la de $H(m^*)$ son identicas, a menos de no haber calculado $H(m^*)$ explicitamente! Por lo que la unica manera de distinguir, es de hacer en algun momento una query de m^* a la API de H . Habiendo $\mathcal{B}$ monitoreado queries a H , guardando las queries en una lista Q_H construida como Q_G , $\mathcal{B}$ va a encontrar m^* bajo $Q_H[c^*]$ cuando $\mathcal{A}$ termine de ejecutar. Por lo tanto $\mathcal{B}$ rompe OW PKE, contradiciendo nuestra suposicion de que esto es dificil. $\square$

Comentario 63. Con los cifrados del curso, Podríamos realizar un IND-CCA PKE combinando un FO KEM basado sobre ElGamal PKE, junto a un AE obtenido como EtM. Un cifrado parecido y que también otorga IND-CCA es parte del estándar ISO/IEC 18033-2 para cifrado de claves publicas, bajo el nombre de DHIES (ECIES si $\mathbb{G}$ es una curva elíptica). El estándar FIPS ML-KEM para cifrados post-cuánticos, llamado también Kyber, es un OT-IND-CCA KEM obtenido de un tipo de postulado similar a DDH, a través de una variante de FO parecida a la que acabamos de ver.