Criptografíía y Seguridad Computacional **2016-01**

Clase 6: 6/04/2016

Profesor: Fernando Krell *Notas: Iván Rubio Venegas*

Recuerdo: Cipher Block Chaining (CBC)

El modo de operación CBC para cifradores de bloque requiere inicializar de forma uniforme un vector IV de un largo n. El cifrado de cada bloque del texto plano se lleva a cabo aplicando la función pseudo aleatoria sobre el XOR entre el bloque anterior cifrado (usando IV si es el primer bloque) y el bloque actual de texto plano. Los siguientes, son los pasos a llevar a cabo:

- Se define $c_0 := IV$
- Por cada i de 1 a l, con l la cantidad de bloques, se define $c_i := F_k(c_{i-1} \oplus m_i)$
- La texto cifrado final es $\langle c_0, c_1, ..., c_l \rangle$

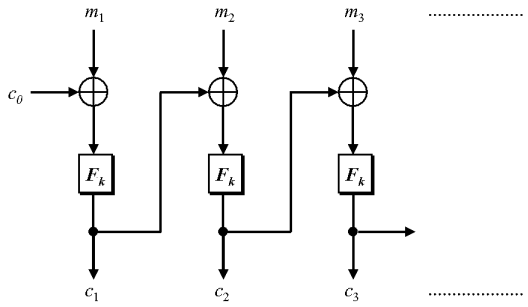

Figura 1: Modo CBC

Padding: PKCS #5. Si faltan b bytes para completar el último bloque, se rellena con b bytes que representan el número b. Cada uno $1 < b \leq l$. Al decifrar un mensaje, si el padding es correcto se entrega el mensaje, si no, se entrega error.

¿Qué puede hacer un adversario activo que ve c? Puede modificarlo y ver la reacción del servidor.

Supongamos que el adversario ve un texto cifrado de 3 bloques IV, c_1 y c_2. Con esta información, lleva a cabo los siguientes pasos:

1. Obtener b. Partiendo por del último byte de c_1, se modifica tal byte de c_1 y se observa si el servidor arroja error, intentamos con el byte que lo precede, y repetimos hasta que el servidor acepta. Luego se define: $m_2 = F_k^{-1}(c_2) \oplus c_1$. Si se modifican bytes correspondientes al padding, el servidor enviará error. Al encontrar el primer byte que arroja, se puede computar b.

2. Teniendo b se obtiene el mensaje m_2 de la siguiente manera. Sea $m_2 = x_1x_2...bbbb...b$, con b repetido b veces, se modifican $c_{n-b}...n$, osea los últimos b bytes del mensaje m_2. De forma tal que, m_2 termine con $b+1$ en vez de b. Es decir el mensaje será $m_2 = x_1...x_{n-b}...b+1, ...b+1$ con $b+1$ repetido b veces .

3. Se modifica el byte x_{n-b-1} varias veces hasta que el resultado sea $b+1$. Se obtiene $b+1 = x_{n-b-1} \oplus \Delta_i$. Siendo Δ_i elegido por el adversario para modificar x_{n-b-1}. Finalmente, aplicando XOR en los dos lados de la ecuación, se obtiene $x_{n-b-1} = b+1 \oplus \Delta_i$

4. Se hace lo mismo para los demás valores de entre x_{n-b-2} y x_n.

Ejercicio: ¿Cómo el adversario puede decifrar m_1 también?

Modelo de seguridad

Seguridad bajo ataques de texto *cifrado* escogido. (Chosen Ciphertext Attacks, CCA)

Definición 1. *Un esquema de encriptación de llave privada es CCA-seguro si para todo adversario PPT $\mathcal{A}$ existe una función negligible* negl*, tal que:*

$$\Pr[\mathsf{Exp}^{ind\text{-}CCA}_{\Pi,\mathcal{A}}(\lambda) = \text{“}A \text{ gana”}\,] \leq \mathsf{negl}(\lambda)$$

Ningún esquema visto hasta ahora es CCA-seguro. Cualquier modificación al texto cifrado da un texto plano relacionado al original, lo cual permite que el adversario utilize su oráculo sobre alguna modificación del texto cifrado recibido c y reconocer si c corresponde a m_0 o a m_1.

$\underline{\mathsf{Exp}^{\mathtt{ind}}_{\Pi,\mathcal{A}}(\lambda)}$**:**

1. $k \leftarrow \mathsf{Gen}(1^\lambda)$
2. $m_0, m_1 \leftarrow \mathcal{A}(1^{\lambda^{\mathsf{Enc}_k(\cdot),\mathsf{Dec}_k(\cdot)}})$
3. $b \xleftarrow{\$} \{0,1\}, c \leftarrow \mathsf{Enc}_k(m_b)$
4. $b' \leftarrow \mathcal{A}(c^{\mathsf{Enc}_k()\mathsf{Dec}_k()}$

output "$\mathcal{A}$ gana" sii $b' = b$ y c no fue consultado a $\mathsf{Dec}_k()$.

Figura 2: Experimento $\mathsf{Exp}^{\mathtt{ind\text{-}CCA}}_{\Pi,\mathcal{A}}(\lambda)$

Cifrados autenticados

Sea $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ un esquema de cifrado, se quiere que tenga privacidad de mensajes (CCA), e integridad de mensajes. Recordemos del concepto de integridad:

Integridad: textos cifrados no pueden ser falsificables

$\underline{\mathsf{Exp}^{\mathtt{ind}}_{\Pi,\mathcal{A}}(\lambda)}$**:**

1. $k \leftarrow \mathsf{Gen}(1^\lambda)$
2. $c \leftarrow \mathcal{A}^{\mathsf{Enc}_k()}$
3. $m \leftarrow \mathsf{Dec}_k(c)$

output "$\mathcal{A}$ gana" si, y sólo si, $m \neq \perp$ y m no fue consultado al oráculo $\mathsf{Enc}_k(\cdot)$.

Figura 3: Experimento $\mathsf{Exp}^{\mathtt{enc\text{-}fals}}_{\Pi,\mathcal{A}}(\lambda)$

Posibles construcciones para un sistema de encriptación que cumpla ambas propiedades:

1. **Cifrar y autenticar (CPA-enc + MAC)**
   Se llevan a cabo de forma independiente, dado un mensaje m:
   $\mathsf{Enc}_k(m) : c \leftarrow \mathsf{Enc}_{k_1}{}^{(\mathsf{cpa})}(m), t \leftarrow \mathsf{Mac}_{k_2}(m)$
   Output: $\langle c, t \rangle$
   No es seguro pues MAC no asegura privacidad.

2. **Autenticar y después cifrar (MAC y después CPA-enc)**
   $\mathsf{Enc}_k(m) : t \leftarrow \mathsf{Mac}_{k_1}(m), c \leftarrow \mathsf{Enc}^{(\mathsf{cpa})}_{k_2}(m||t)$
   $\mathsf{Dec}_k(c)$:

   a) $\mathsf{Dec}^{(\mathsf{cpa})}_{k_2}(c) = m||t$

   b) Si $\mathsf{Vrfy}_{k_1}(m, t) = 1$, output m

   Este sistema tampoco es seguro, pués el algoritmo de descifrado $\mathsf{Dec}^{(\mathsf{cpa})}(c)$ puede arrojar error, y por lo tanto, es vulnerable a un ataque de padding como fue visto en la sección anterior.

3. **Cifrar y depués autenticar (CPA-enc y luego MAC)** $\mathsf{Enc}_k(m) : c \leftarrow \mathsf{Enc}^{(\mathsf{cpa})}_{k_2}(m), t \leftarrow \mathsf{Mac}_{k_1}(c)$

   Este sistema sí es seguro pues al aplicar MAC sobre c no se modifica su valor (es decir se mantiene la propiedad de seguridad de CPA), y se tiene la propiedad de integridad por usar MAC.