Criptografía y Seguridad
Tarea 4

Valeria García Landa
Luis Fernando González Montiel

1. Sea $F : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ una función de cifrado por bloques (una permutación pseudoaleatoria). Considera las siguientes formas de usar F para cifrar un bloque $m \in \{0,1\}^n$ con la clave k:

   - $c = F_k(m)$
   - Se elige $r \leftarrow \{0,1\}^n$ y el cifrado es $c = (r, F_k(r) \oplus m)$.

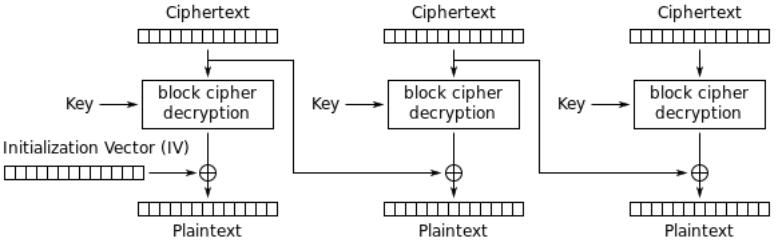
   ¿Cuál de las dos formas consideradas es más segura? ¿Por qué? ¿Preferirías una de estas formas o algún modo de operación de los vistos en clase?

   Comparando ambos modos de operación, el segundo método tiene un extra, pues estamos ocupando una r que se genera pseudoaleatoriamente la cual es de suma importancia, ya que si suponemos que tenemos que cifrar varios bloques, entonces por cada bloque que se cifre, se debe generar una nueva r. En cambio, el primer método pues sólo aplicamos la función F con la llave y eso es todo. Por lo que el segundo método es más seguro que el primero.

   Respecto a los demás modos de operación que vimos CBC parece ser más seguro puesto que cada bloque depende de la información del anterior y al momento de descifrar, pues de igual manera, se necesita la información de todos los bloques, si alguno llegara a faltar, entonces no se obtendría toda la información.

2. Es posible que cuando se manda un criptotexto $c = c_1, c_2, c_3...$ en la transmisión se pierda el bloque c_2, así que se recibe el mensaje $c' = c_1, c_3, c_4...$ Describe el efecto que causa un bloque perdido en el criptotexto, cuando se descifra usando los modos de operación CBC, OFB y CTR.

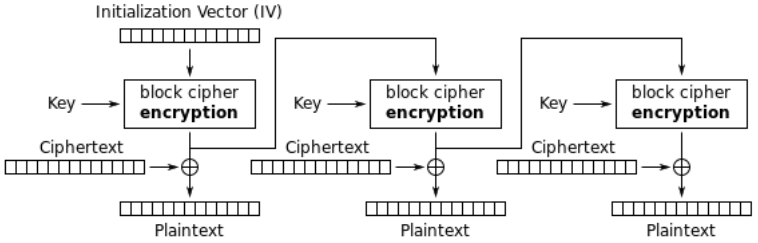
   CBC



   Como vemos en el diagrama para descifrar un mensaje, se recibe como vector de entrada la información del c_{i-1} mensaje cifrado, por lo que al perder la información del bloque c_i, al momento de descifrar el bloque c_{i+1} algo saldrá mal, pues no tenemos la información de entrada que necesitamos, por lo que no se podrá descifrar ese bloque y por lo tanto, no se tendrá el texto claro completo.

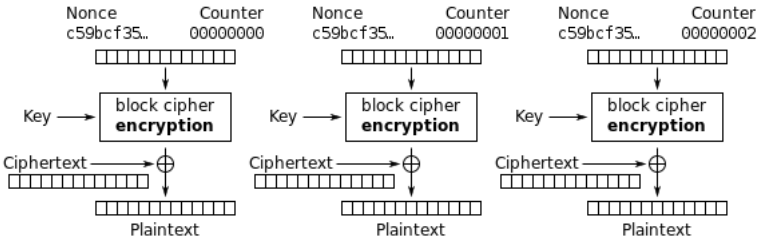
Los siguientes bloques al bloque c_{i+1} se podrán descifrar sin ningún problema, ya que esos bloques no se perdieron.

OFB



En este modo de operación, pasa lo mismo que en CBC pues también para descifrar requiere de la información del bloque anterior, por lo que el texto claro no tendrá la información completa.
Como vemos en el diagrama, el modo de operar varía un poco al anterior. Pero el resultado al final de cuentas se vería afectado de la misma manera.

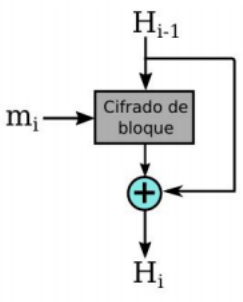
CTR



En este odo de operación, es totalmente distinto ya que para descrifrar un bloque no se necesita de ninguna información previa, es decir, el bloque c_i no depende del bloque c_{i-1}. Por lo que al haber perdido un bloque, cualquiera, no afectará a los demás.

3. Existen varias formas de usar un cifrador de bloques para construir funciones hash.

   - Describe como hacer una función hash usando el método de Davies-Meyer.



Usar como cifrado de bloque: AES, DES, etc; que tenga el mismo tamaño de bloque que de llave.
Tomar la entrada, dividirla en bloques del tamaño aceptado por el cifrado. Suministrar el bloque i

al cifrado y la salida como clave de entrada al cifrado del bloque $i + 1$.

Tenemos un mensaje m de tamaño M al que le queremos calcular su valor de hash H que debe medir $N<M$ bits.
Dividimos M en bloques de tamaño $N : \{m_1, m_2, ..., m_k\}$ con $k = \lceil \frac{M}{N} \rceil$ posiblemente el último bloque quede incompleto. Lo rellenamos con ceros.

m_1 entra a un cifrado de bloque junto con una clave inventada a la que llamamos Vector de Inicialización (Initialization Vector o IV). La salida se ingresa como clave del miso cifrado, con m_2 como entrada. La salida del cifrado del bloque, luego de recibir como clave el cifrado de m_i y como entrada m_{i+1}, se pasa como clave para cifrar m_{i+2}

Específicamente para Davis-Mayer, entra H_{i-1} dando como salida final H_i
Formalmente lo podemos ver como $H_i = H_{i-1} \oplus E_{m_i}(H_i)$

- Recuerda que en DES existen llaves k para las que es fácil encontrar m tal que $DES_k(m) = m$. Muestra cómo usar esta propiedad para encontrar una colisión en la construcción de Davies-Meyer aplicada a DES.

  Usando la propiedad dada, $DES_k(m) = m$, es decir, que los mensajes se quedan fijos, al aplicarle la función de compresión Davies-Meyer sabemos que a la salida del bloque le aplicamos un XOR con la salida del bloque anterior, pero si se da el caso en el que $m_1 \neq m_2$ se cifran y da el caso en el que la salida es la misma y al aplicarle el XOR a ambos mensajes, nos dará la misma salida del bloque. Por lo que es ahí en donde se encuentra una colisión, es decir, donde $DES_k(m) = m$.

5. Extrae el archivo $mis_archivos.zip$, que contiene el directorio $Mis\ archivos$. Desde este directorio ejecuta *juego.py* con Python 3:
   Haz un programa que duncione como *vacuna* para el ransomware, es decir, que revierta los cambios hechos por *juego.py*

   Para este ejercicio se encuentra una carpeta src en donde se encontrará el directorio y el archivo *vacuna.py*
   Una vez parados en el directorio "Mis Archivos"habrá que ejecutar *juego.py* para que los archivos se encripten, hecho esto, copiamos y pegamos en este directorio el archivo *vacuna.py*, para que no se encripte dicho archivo, decidimos ponerla un nivel antes. Ya que se se cambió de nivel, ejecutamos *vacuna.py* para revertir los cambios.