Capítulo 5

Técnica de Inserción de Caracteres Falsos y Compresión (FCICT)

En este capítulo se presenta y analiza el funcionamiento de la técnica FCICT que es una alternativa para eliminar algunas de las debilidades del WEP, diseñada por Nishanth Chandran y K. R. Bhavana en el año 2005. Primeramente se presenta una introducción a la técnica FCICT explicando de forma general su funcionamiento; posteriormente se explican detalladamente los procesos de encriptación y decriptación de la misma. En la parte final del capítulo se explica cómo este algoritmo elimina las debilidades del WEP.

5.1 Funcionamiento de la FCICT.

La técnica FCICT se aplica como un proceso adicional al cifrado del WEP para lograr así un WEP mejorado. El mecanismo FCICT consiste de dos partes. En la primer parte, se insertan caracteres de acuerdo al cumplimiento de una condición que depende del valor de una llave y de los datos involucrados. En la segunda parte, se comprime el texto por el algoritmo de compresión de Huffman o con una técnica de reemplazo más simple.

En las siguientes secciones se presenta cada una de las etapas involucradas en la técnica FCICT, que muestran el uso del algoritmo RC4 para generar la secuencia de llaves con la que se encriptará la información y los procesos de encriptación y decriptación de datos.

5.1.1 Generación de la secuencia de llaves

En esta parte se utiliza el algoritmo RC4 para generar una secuencia de números pseudo aleatorios que servirán como llaves de encriptación. Primeramente se utiliza una llave de longitud variable *K* que puede ser de 1 a 256 bytes, que será utilizada para inicializar un vector *S* de 256 bytes. Antes de inicializarlo, el vector *S* está formado por valores de 0 a 255 en orden ascendente.

Se crea un vector temporal *T* que obtiene el valor para sus primeros *n* elementos, copiando los *n* bytes de *K*. Después *K* se repite tantas veces como sea necesario para llenar todo el vector *T*. Posteriormente, se utiliza *T* para producir la permutación inicial de *S*.

Una vez que se inicializó S, la llave *K* no se utiliza más. El siguiente paso es generar una secuencia de números pseudo aleatorios que servirán como llaves de encriptación. Empezando el proceso desde S_0 hasta S_{255}, cada valor de *S* se intercambia con otro del mismo vector de acuerdo a un esquema dictado por la configuración actual de *S*.

Al finalizar el ciclo, es decir cuando se intercambió el valor en la posición 255 de *S*, este valor se toma como una llave. Debido a que se requiere generar una llave de encriptación para cada byte del texto plano a encriptar, este ciclo se repite hasta generar el número de llaves necesario. El texto cifrado (encriptado) resultante, se obtiene al realizar la operación lógica XOR entre cada byte correspondiente a una llave generada y un byte correspondiente al texto plano.

Hasta este punto ya se tiene un texto encriptado de la misma forma en la que encripta el protocolo WEP. El siguiente paso es insertar caracteres falsos en dicho texto. Esto se puede realizar utilizando una técnica de reemplazo de caracteres o usando el algoritmo de compresión de Huffman.

5.1.2 Encriptación FCICT utilizando la técnica de reemplazo.

Este procedimiento toma el texto cifrado RC4 (*R)* y una llave denominada *fakekey*, que sirve para generar una permutación aleatoria de números del 0 al 255. Esta llave se genera utilizando una llave de longitud variable *K* de la misma forma en que se generó *S* en el algoritmo RC4.

Como primer paso, se utiliza *R* para inicializar, con los mismos valores, a *C* que es el texto cifrado que incluirá los caracteres falsos. Para cada carácter de *R*, exceptuando el primero y el último, se realizan las siguientes operaciones:

- *fakecheck* es una variable cuyos valores se usan para verificar el cumplimiento de una condición que está determinada por:

$$fakecheck = (R_i + R_{i-1} + fakekey[i \bmod 256]) \bmod 2^m \qquad (5.1)$$

donde R_i *y* R_{i-1} son dos bytes adyacentes del cifrado RC4. La expresión *fakekey[i mod 256]* corresponde a un valor pseudo aleatorio generado dentro del vector *fakekey* apuntado por *i mod 256*. Debido a que el cálculo de *fakecheck* se realiza por cada byte del texto cifrado, se utiliza la función *mod* que impide que

el apuntador del vector *fakekey* sobrepase el valor de 256 en el caso de que el texto cifrado tenga un número mayor de bytes.

- Si *fakecheck* es igual a cero, entonces se inserta un caracter falso en el texto.

  La suma de los valores expresados en la ecuación 5.1 se utiliza para realizar la operación módulo con el valor 2^m. El valor *m* puede ser un entero entre 1 y 8. Para valores pequeños de *m*, aumenta la probabilidad de que *fakecheck* sea igual a cero. Para valores grandes de *m*, disminuye esta probabilidad. Esto hace que la probabilidad de un intruso para encontrar los caracteres falsos, sea directamente proporcional al valor elegido para *m*.

- El caracter falso a insertar, se determina por la ecuación:

$$fakechar = (R_i \oplus R_{i-1} \oplus fakekey[i \bmod 256]) \quad (5.2)$$

  donde $\oplus$ denota la operación lógica XOR.

La inserción de este carácter falso causaría un incremento en la longitud del cifrado, la cual sería mayor a la longitud de la secuencia de llaves calculadas, por lo que el receptor no podría decriptar la información. Para que el cifrado FCICT y la secuencia de llaves tengan la misma longitud, se utiliza la técnica de reemplazo, en la cual el carácter falso se “mezcla” con un caracter del cifrado RC4 (WEP). Así se obtiene un solo valor determinado por:

$$C_{i+1} = (fakechar + R_{i+1}) \bmod 256 \quad (5.3)$$

Una vez generado el texto cifrado *C* con los caracteres falsos insertados, este se envía al receptor con el vector de inicialización (ver Figura 5.1) el cual utilizará el receptor para calcular las llaves que servirán para decriptar el texto cifrado y obtener el texto original.

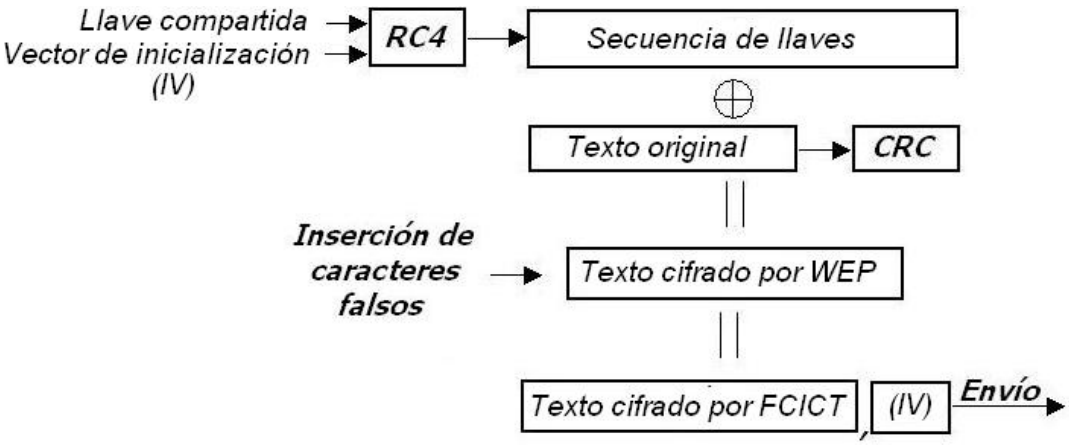

Figura 5.1 Encriptación FCICT.

La inserción de caracteres falsos utilizando la técnica de reemplazo se muestra en la Figura 5.2.

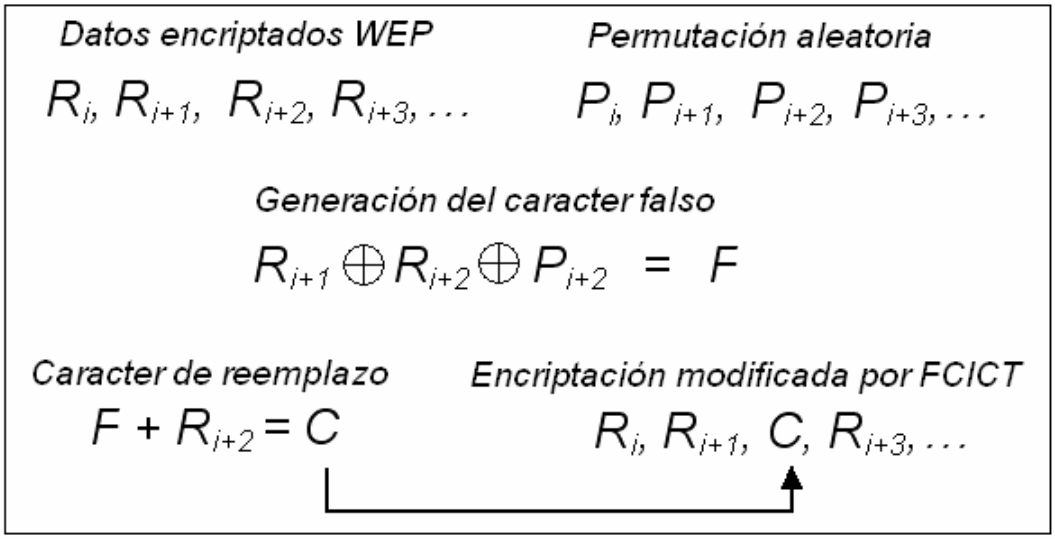

Figura 5.2. Inserción de caracteres falsos utilizando la técnica de reemplazo.

5.1.3 Decriptación FCICT utilizando la técnica de reemplazo.

El receptor de los datos utiliza el mismo algoritmo que el transmisor, para identificar la posición donde se insertaron los caracteres falsos. Esta posición depende del valor de *fakekey* y de la ecuación 5.1.

Al identificar que un caracter es falso, el receptor cambia su valor por uno que corresponda al cifrado RC4. Este nuevo valor está determinado por la siguiente ecuación:

$$C_{i+1} = (2^8 + C_{i+1} - fakechar) \mod 256 \qquad (5.4)$$

$C_{i+1} = (2^8 + C_{i+1} - fakechar) \; mod \; 256$

En la Figura 5.3 se muestra la forma en la que se reemplaza el caracter falso por uno correspondiente al cifrado RC4 (WEP).

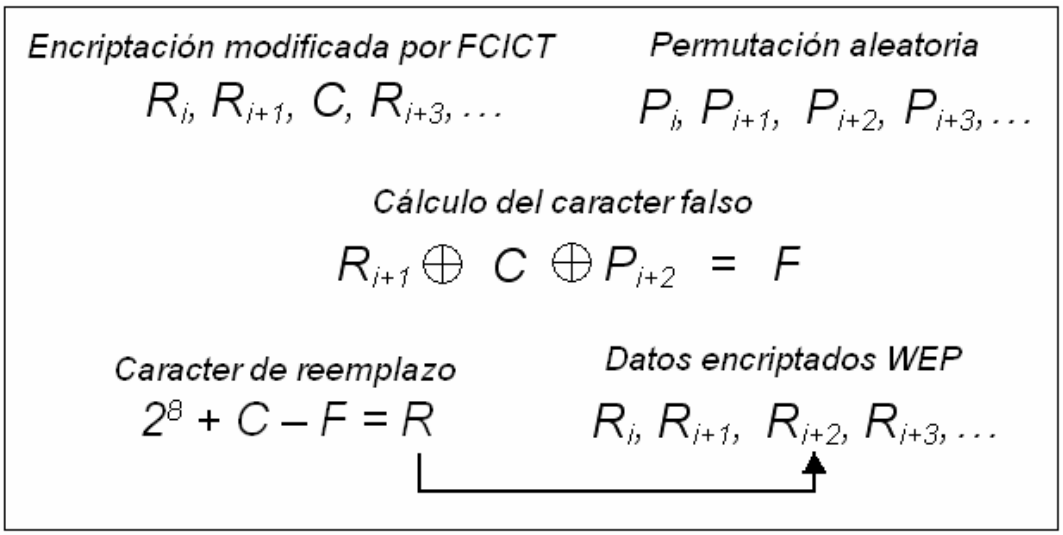

Figura 5.3. Reemplazo del caracter falso en la decriptación por técnica de reemplazo.

Ya que el receptor reemplazó los caracteres falsos detectados por los caracteres que corresponden al cifrado RC4, realiza la operación XOR entre el texto cifrado y las llaves de decriptación para obtener el mensaje original.

5.1.4 Encriptación FCICT utilizando el algoritmo de compresión de Huffman.

En este caso también se utiliza la llave *fakekey* y el texto cifrado RC4 sobre el cual se insertarán los caracteres falsos. La condición para insertar un carácter falso así como la ecuación que determina el carácter falso a insertar, son iguales a las de la encriptación que utiliza técnica de reemplazo.

La única diferencia en la encriptación FCICT usando el algoritmo de compresión de Huffman, es que los caracteres falsos calculados, se insertan directamente en el algoritmo RC4 provocando que el texto cifrado (después de la inserción) tenga una longitud mayor a la de la secuencia de llaves. Por esto el receptor necesita determinar los caracteres falsos, descartarlos del texto cifrado, y decriptarlo usando el RC4. La inserción de caracteres en este caso ocurre como se muestra en la Figura 5.4.

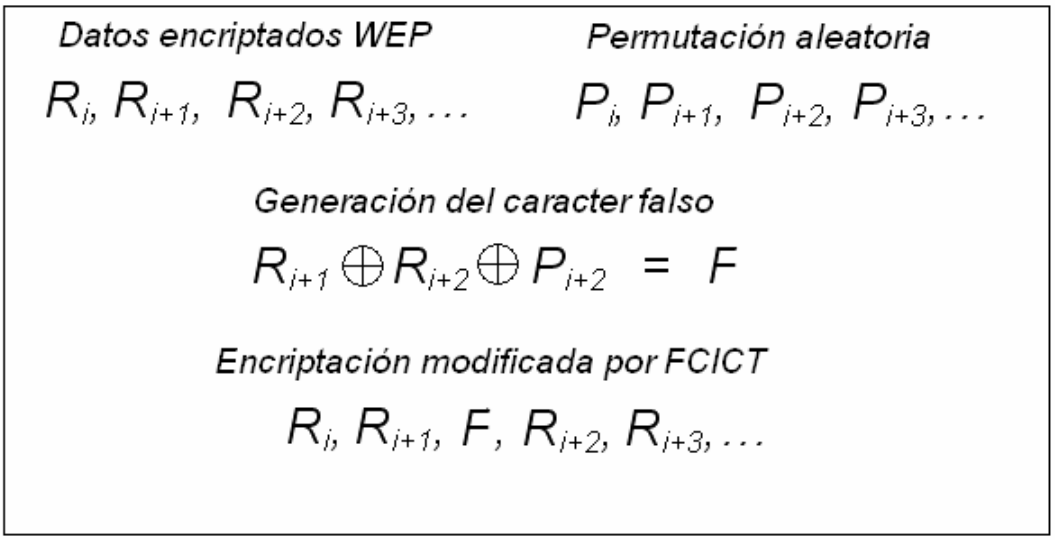

Figura 5.4. Inserción de caracteres falsos por algoritmo de compresión de Huffman.

5.1.5 Decriptación FCICT utilizando el algoritmo de compresión de Huffman.

Al igual que en la decriptación con la técnica de reemplazo, el receptor determina los caracteres falsos usando las mismas ecuaciones. En este caso, el receptor no necesita calcular el valor de reemplazo para obtener el cifrado RC4, simplemente descarta los caracteres falsos detectados del texto cifrado.

Para detectar los caracteres falsos se utiliza la ecuación 5.1. Para cada byte del texto cifrado FCICT recibido, se verifica si el valor de *fakecheck* es igual a cero. Si es así, este valor se descarta ya que es un caracter falso y se procede a verificar de la misma forma los bytes siguientes del cifrado FCICT (ver Figura 5.5).

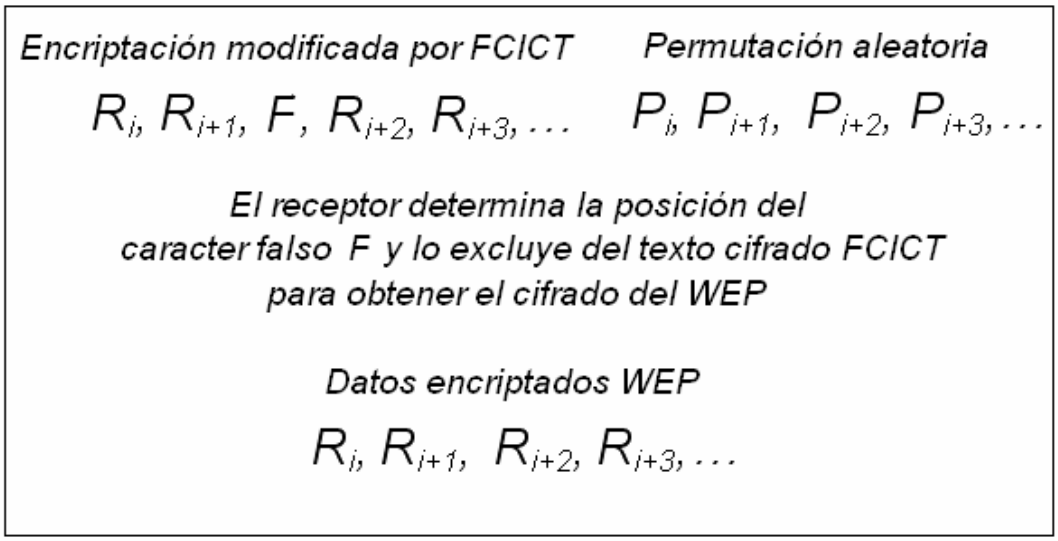

Figura 5.5. Exclusión de caracteres falsos por algoritmo de compresión de Huffman.

Al descartar todos los caracteres falsos, la longitud del texto cifrado disminuye, y debe ser igual a la longitud de la secuencia de llaves para poder hacer la operación XOR entre cada byte y recuperar el texto plano.

5.2 Ventajas de la técnica FCICT sobre el WEP.

Analizando la seguridad de este esquema, muestra que el algoritmo WEP trabajando con la técnica FCICT es casi invulnerable a ataques por fuerza bruta. Ya que un ataque de este tipo realizado sobre este algoritmo necesitaría un cálculo de 256! para obtener el valor de *fakekey*. Además, un solo error en la identificación de un caracter falso, llevaría a obtener una serie de llaves incorrectas para la decriptación [CHA05]. Por lo que un intruso no lograría decriptar la información original.

Un ataque típico en el WEP consiste en interceptar varios textos cifrados hasta recuperar dos que hayan sido encriptados con el mismo vector de inicialización, para después realizar la operación XOR entre un cifrado y un texto plano obtenido y recuperar las secuencias de llaves.

Este ataque no se realizaría de una forma tan simple en datos encriptados con la técnica FCICT. Ya que los textos cifrados obtenidos por un intruso contiene bytes que sólo el receptor puede identificar como falsos y descartarlos. Un ataque de este tipo llevaría al intruso a obtener secuencias de llaves erróneas con las cuales no podría decriptar la información original.

Utilizando la técnica de reemplazo, el caracter original encriptado se modifica en la posición donde *fakecheck* cumpla la condición. Un intruso tendría que llegar primero a este valor antes de iniciar la decriptación. Si este caracter se descarta por el intruso, se perderían caracteres originales del texto plano. Para un valor de $m = 1$, ser perdería aproximadamente el 50% de los caracteres del texto plano.