Capítulo 6

Simulaciones y resultados

En este capítulo se incluyen simulaciones del funcionamiento del WEP y la técnica FCICT utilizando la interfase visual creada en MATLAB. La primera simulación consiste en la encriptación de una cadena de caracteres dada, utilizando el algoritmo del WEP. Posteriormente se realizan tres simulaciones de la FCICT utilizando la técnica de reemplazo y variando el valor de *m* para observar el efecto que tiene en la cantidad de caracteres insertados en un texto cifrado. Después se presenta una simulación de la técnica FCICT utilizando el algoritmo de compresión de Huffman. Finalmente se muestran los resultados con tablas comparativas que muestran las diferencias entre los distintos algoritmos usados. En el apéndice C se muestran las imágenes de la interfase visual correspondientes a cada simulación.

6.1 Introducción

El funcionamiento ideal de la técnica FCICT con todos los componentes del WEP se muestra en la Figura 5.1. Para implementar dicho procedimiento se tendría que incluir una cadena de 40 o 104 bits simulando una llave compartida entre una estación transmisora y otra receptora. Además se debería incluir una cadena de 24 bits como vector de inicialización, mismo que debería cambiar cada vez que se transmitiera un conjunto de datos de 2312 bytes entre ambas estaciones, simulando los datos correspondientes a una trama 802.11. Así mismo se debería incluir el algoritmo CRC32 para verificar la integridad de la información encriptada.

Cabe mencionar que los programas elaborados en esta tesis no incluyen algunas etapas del funcionamiento general del WEP, tales como el CRC, el vector de inicialización y la llave compartida. El CRC no se realiza porque el objetivo de la técnica FCICT se enfoca principalmente en contrarrestar las debilidades del cifrado del WEP causado por los vectores de inicialización, más allá de verificar la integridad de los datos enviados. Por otra parte, la llave *K* genera la tabla de estados que genera los bytes pseudo-aleatorios para generar la secuencia de llaves, esto suple a la llave compartida y al vector de inicialización, ya que al tratarse de una simulación, no se manejan paquetes de datos 802.11 entre dos estaciones. Hay que recordar que los vectores de inicialización cambian en cada trama transmitida; en la simulación se considera únicamente una cadena de caracteres que se encripta y decripta con el algoritmo RC4, aunque no existe un vector de inicialización y una llave compartida específica, el procedimiento es el mismo.

6.2 Encriptación y decriptación de información del WEP.

En este ejemplo se utiliza el texto “computadora” que será encriptado con el algoritmo RC4. Se determinó una longitud de 45 bytes de la llave *K* para iniciar una tabla de estados, misma que genera una lista de bytes pseudo-aleatorios que forman la secuencia de llaves que se utilizarán para encriptar el texto original. Una vez determinada la longitud de la llave, el algoritmo RC4 utiliza este dato para iniciar el proceso que genera las llaves de encriptación (una para cada byte de texto transmitido), las cuales se ven en la Figura 6.1.

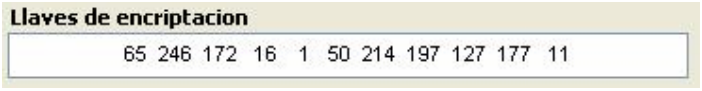

Figura 6.1 Llaves de encriptación.

Estas llaves corresponden a los bytes pseudo-aleatorios expresados en forma decimal. Con el fin de mostrar de forma sencilla el funcionamiento de los algoritmos, todos los resultados se expresarán en forma decimal. Para generar el texto cifrado, se realiza la operación XOR entre las llaves calculadas y el texto “computadora” que corresponde a los bytes expresados en código ASCII mostrados en la Figura 6.2.

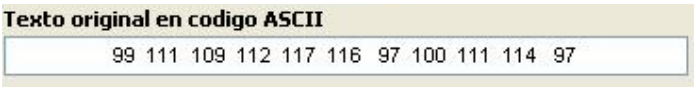

Figura 6.2 Texto original representado en código ASCII.

Los bytes que forman el texto cifrado resultante se muestran en la Figura 6.3. Este texto cifrado se envía con el vector de inicialización al receptor, el cual usará el algoritmo RC4 para calcular las llaves que decripten el mensaje haciendo la operación XOR entre el texto cifrado y las llaves calculadas. El resultado de esta operación muestra los bytes mostrados en la Figura 6.2, que corresponden al texto “computadora”, por lo que el mensaje fue decriptado como se esperaba.

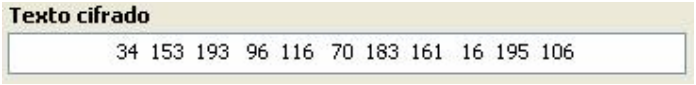

Figura 6.3 Texto cifrado RC4.

En la práctica, debido a que el vector de inicialización viaja del transmisor al receptor sin ningún tipo de encriptación, el intruso puede monitorear la red para obtener el texto cifrado y el vector de inicialización, y así poder ejecutar algún tipo de ataque de los mencionados en el capítulo 3, con el fin de decriptar la información de interés para el intruso.

6.3 Encriptación y decriptación de información usando la técnica FCICT con técnica de reemplazo y valor m igual a 1.

Para comparar esta técnica con el WEP tradicional, se utilizó la misma cadena de texto que el ejemplo anterior, así como el mismo valor para la longitud de la llave *K*. Además, para la simulación de la técnica FCICT se establece también el valor de *fakekey* explicado anteriormente, que para este ejemplo se eligió de 160, aunque puede tomar cualquier valor entre 1 y 256.

Al igual que en la simulación del WEP, se genera un texto cifrado obtenido al realizar la operación XOR entre las llaves de encriptación generadas por el algoritmo RC4 y el texto a transmitir. El cifrado resultante es idéntico al obtenido en la simulación # 1 (ver Figura 6.3) ya que se utilizó una misma longitud de llave *K* y la misma cadena de caracteres a encriptar.

Una vez que el RC4 generó el texto cifrado, empieza el proceso de inserción de caracteres falsos. La técnica FCICT utiliza una condición para determinar en qué partes del texto cifrado se insertará un caracter falso. Esta condición está determinada por la ecuación 5.1, donde $m = 1$. Como se mencionó anteriormente, a medida que decrece el valor de *m*, aumenta el número de caracteres falsos a insertar; por lo que para este ejemplo, se espera que el algoritmo inserte el mayor número de caracteres falsos posibles en el texto cifrado.

Después de que el algoritmo insertó los caracteres falsos determinados por la variable *fakechar*, el cifrado resultante quedó conformado por los bytes mostrados en la Figura 6.4.

Figura 6.4 Cifrado FCICT.

En la parte de decriptación, el receptor recibe el texto cifrado que incluye los caracteres falsos, los identifica, y los reemplaza por aquellos caracteres que forman parte del cifrado original del RC4. En la Tabla 6.1 se muestran los caracteres falsos insertados que hacen diferente a un cifrado de otro.

Cifrado RC4	34	153	193	96	116	70	183	161	16	195	106
Cifrado FCICT	34	153	19	246	116	206	183	239	56	73	106

Caracteres falsos insertados

Tabla 6.1. Diferencias entre el cifrado RC4 y el cifrado FCICT.

Posteriormente, el receptor realiza la operación XOR entre el cifrado RC4 recuperado y las llaves de encriptación calculadas, para obtener el mensaje original satisfactoriamente, que en esta simulación corresponde al texto “computadora”.

En la práctica, un intruso que esté monitoreando la red puede obtener, al igual que en el primer caso, el vector de inicialización puesto que se sigue enviando sin encriptación alguna, pero el texto cifrado que recupere el intruso será diferente al generado por el algoritmo RC4. Esto hará que el intruso no pueda recuperar los datos originales que circulan por la red, ya que la mayoría de los ataques se basan en operaciones XOR entre textos cifrados RC4, llaves de encriptación, y mensajes originales obtenidos. En este caso, la simulación muestra que un intruso recupera la cadena de caracteres mostrada en la Figura 6.5 el cual es diferente al texto

“computadora” que recuperó el receptor indicado, por lo que no importa que un intruso obtenga un vector de inicialización y un texto cifrado, ya que al realizar la operación XOR entre la secuencia de llaves y los datos encriptados, no recuperará la información original.

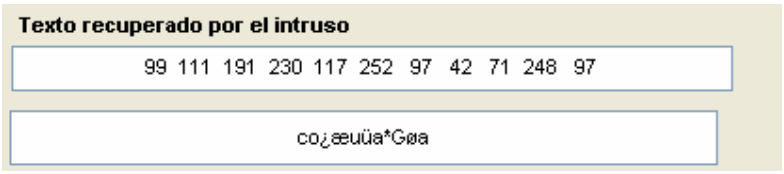

Figura 6.5. Texto recuperado por un intruso.

A continuación se presentan dos simulaciones más para ver los efectos que tiene el valor de *m* en la inserción de caracteres falsos en un texto cifrado.

6.4 Encriptación y decriptación de información usando la técnica FCICT con técnica de reemplazo y valor m igual a 4.

Para esta simulación, así como para las siguientes simulaciones, el texto a transmitir, y los valores de la llave *K* y *fakekey*, serán igual al caso anterior, con el fin de no alterar el cifrado RC4 y observar únicamente como varía la cantidad de caracteres insertados en el mismo al variar el valor de *m*, que en esta simulación es de 4. De acuerdo al algoritmo FCICT, un valor más grande de *m* debe insertar menos caracteres falsos en el cifrado. Para comprobarlo se ejecutó el programa, obteniendo el cifrado FCICT mostrado en la Figura 6.6.

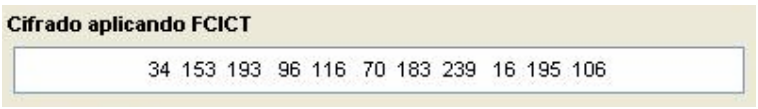

Figura 6.6. Cifrado FCICT.

Como se puede observar en la Tabla 6.2, en el cifrado FCICT con valor *m* = 1, se insertaron 6 caracteres falsos, mientras que al aumentar el valor de *m* a 4, únicamente se insertó un caracter falso, por lo que aumenta la probabilidad de que un intruso detecte el carácter falso.

Cifrado RC4	34	153	193	96	116	70	183	161	16	195	106
Cifrado FCICT con *m* = 1	34	153	19	246	116	206	183	239	56	73	106
Cifrado FCICT con *m* = 4	34	153	193	96	116	70	183	239	16	195	106

Caracteres falsos insertados

Tabla 6.2. Comparativa entre cifrados FCICT con valores de *m*: 1 y 4.

En este caso, la simulación muestra que el receptor decriptó la información con éxito obteniendo el texto “computadora”, mientras que un intruso recupera el texto mostrado en la Figura 6.7.

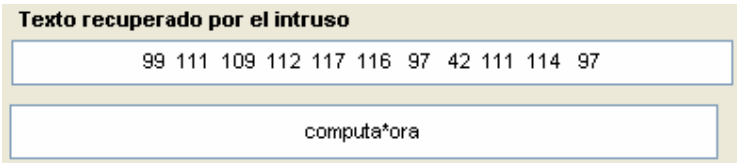

Figura 6.7. Texto recuperado por el intruso.

Aun cuando el texto recuperado por el intruso tiene mayor cantidad de caracteres similares al texto recuperado por el receptor, sigue obteniendo una cadena de caracteres diferentes, por lo que la información que envía el transmisor únicamente la puede decriptar de forma exitosa el receptor indicado.

6.5 Encriptación y decriptación de información usando la técnica FCICT con técnica de reemplazo y valor m igual a 8.

Por último, se eligió *m* = 8, que es el máximo valor que soporta este algoritmo. Usando este valor se ejecutó el programa y se obtuvo el cifrado FCICT mostrado en la Figura 6.8.

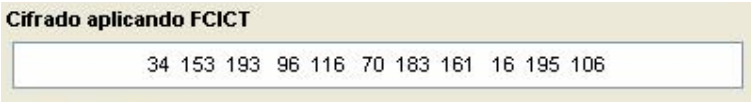

Figura 6.8. Cifrado FCICT.

En la Tabla 6.3 se pueden observar las diferencias entre los cifrados FCICT al variar el valor de *m*.

Cifrado RC4	34	153	193	96	116	70	183	161	16	195	106
Cifrado FCICT con *m* = 1	34	153	19	246	116	206	183	239	56	73	106
Cifrado FCICT con *m* = 4	34	153	193	96	116	70	183	239	16	195	106
Cifrado FCICT con *m* = 8	34	153	193	96	116	70	183	161	16	195	106

Caracteres falsos insertados

Tabla 6.3. Comparativa entre cifrados FCICT con valores de *m*: 1, 4 y 8.

Se puede notar que en el caso extremo del valor máximo de *m*, no se insertó ningún carácter falso; es decir, que el cifrado RC4 es idéntico al FCICT con valor *m* = 8, por lo que un intruso recuperaría la misma información que obtendría el receptor. En la Figura 6.9 se puede notar que en la simulación el intruso recuperó la misma cadena que recuperó el intruso, es decir el texto “computadora”.

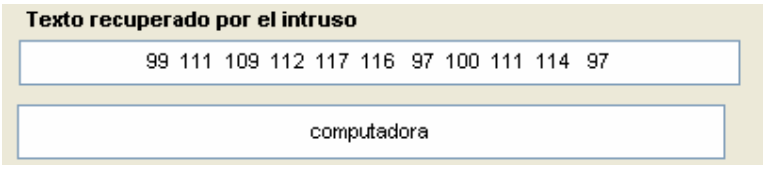

Figura 6.9. Texto recuperado por el intruso.

Esto muestra claramente que el valor ideal para *m* es 1, ya que inserta el mayor número de caracteres falsos posibles en el texto cifrado por el WEP, lo cual disminuye la probabilidad de que un intruso identifique los caracteres falsos y decripte la información que circula por la red.

6.6 Encriptación y decriptación de información usando la técnica FCICT con algoritmo de compresión de Huffman.

En esta simulación se mostrará el funcionamiento de la técnica FCICT utilizando el algoritmo de compresión de Huffman para la inserción de caracteres falsos. Para esta simulación, volvimos a tomar el texto a transmitir, y los valores de la llave *K* y *fakekey*, de las simulaciones anteriores, con el fin de no alterar el cifrado RC4 y observar la diferencia entre la técnica de reemplazo y el algoritmo de compresión de Huffman.

Habiendo comprobado el efecto del valor de *m*, para esta simulación se regresó al valor de 1. Con estos datos se ejecutó la simulación correspondiente a la técnica FCICT con el algoritmo de compresión de Huffman y se obtuvo el texto cifrado RC4 así como el texto cifrado FCICT, ambos se muestran en la Figura 6.10.

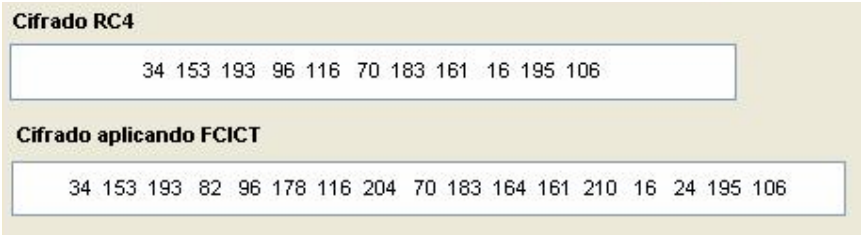

Figura 6.10. Comparación entre cifrados RC4 y FCICT usando el algoritmo de compresión de Huffman.

Se puede notar claramente que la longitud del texto cifrado FCICT es mayor a la del texto cifrado RC4. Este último corresponde al tamaño del texto “computadora” encriptado, que es de 11 bytes. El tamaño del cifrado FCICT resultante es de 17 bytes por lo que se insertaron 6 caracteres falsos en el cifrado RC4. El receptor puede identificar y descartar estos caracteres falsos para formar el cifrado RC4, y así decriptar la infomación con las llaves calculadas a partir del vector de inicialización.

En la simulación se muestra que en el caso de que un intruso obtenga el texto cifrado por la técnica FCICT con el algoritmo de compresión de Huffman, tendría que identificar primeramente, al igual que con la técnica de reemplazo, los caracteres falsos insertados en el texto cifrado para descartarlos posteriormente, ya que la longitud del texto encriptado por la FCICT es mayor a la longitud de la secuencia de llaves utilizadas para la decriptación (ver Figura 6.11).

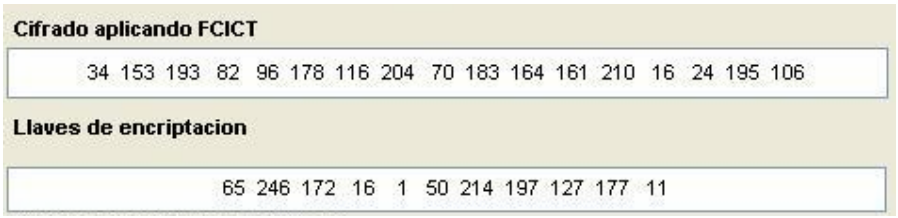

Figura 6.11. Texto cifrado y secuencia de llaves recuperados por un intruso.

Ya se ha mencionado que esta situación complicaría la labor a un intruso, ya que tendría que calcular primeramente la llave falsa para así poder determinar qué caracteres son falsos en el texto cifrado. Y cualquier error en la detección de un solo caracter falso llevaría a un cálculo incorrecto de las llaves para decriptar la información.

De las cuatro simulaciones correspondientes a la técnica FCICT se puede observar que ya sea utilizando la técnica de reemplazo o el algoritmo de compresión de Huffman, este algoritmo agrega una etapa al WEP tradicional modificando el texto encriptado por el mismo y así lograr un WEP mejorado.