IEM-I-03-04

UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERÍA
DPTO. DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO PARA LA ENCRIPTACIÓN DE LA VOZ EN EL SISTEMA PCS SOBRE LA TECNOLOGÍA TDMA (GSM)

Carlos Andrés Donado Coronell
ASESOR: Ing. Rafael Camerano Fuentes

Trabajo de grado presentado
como requisito para obtener
el título de Magíster en
Ingeniería Electrónica

Bogotá D.C., Agosto de 2004

AGRADECIMIENTOS

Agradezco sinceramente a mi asesor de tesis Ing. Rafael Camerano Fuentes por haberme guiado a lo largo de la investigación. Agradezco también al Ing. José Luis Villa por su colaboración en la parte final del trabajo y a Amparo Fúster por haberme facilitado documentación importante sobre el criptoanálisis.

Estoy muy agradecido con mi familia por los consejos aportados y porque me brindó los medios necesarios para el desarrollo de la investigación, con mi novia por su paciencia en tan larga espera y con mi hijo porque desde su inocencia y aparente desentendimiento me iluminó y fortaleció día a día para poder culminar la investigación de la mejor manera.

CONTENIDO

LISTA DE FIGURAS

INTRODUCCIÓN

El Sistema de Comunicación Personal PCS (GSM-1900), de reciente aparición en Colombia, es sin duda un gran avance tecnológico dentro del entorno de las telecomunicaciones. El desempeño de este tipo de sistemas se debe en gran medida a la seguridad en la transmisión de información (voz, datos, imágenes, etc.) que éstos ofrezcan. Este sistema flota sobre un estándar de comunicaciones existente hace algún tiempo (GSM – Sistema Global para las Comunicaciones Móviles) y por tanto basa su seguridad en algoritmos soportados por el mismo y desarrollados algún tiempo atrás (A3: autentificación del usuario; A5: cifrado de la voz y A8: generación de clave de cifrado). Debido a esto y al afán de muchos por acceder a la información privada y existiendo en este tiempo herramientas tecnológicas más potentes a la hora de diseñar ataques criptoanalíticos, esos algoritmos han sido ampliamente superados.

La presente investigación se concentra en la transmisión segura de la voz del usuario en el enlace radio Estación Móvil-Subsistema de Estación Base (MS-BSS) y por tanto se expondrá en detalle el algoritmo de cifrado A5, sus debilidades de acuerdo con los ataques criptoanalíticos efectuados sobre él, la forma de contrarrestar tales ataques y se propondrá un algoritmo que mantenga el rendimiento de aquél pero que además no posea sus debilidades manifiestas.

El algoritmo A5 se emplea para el cifrado del enlace entre el teléfono móvil y la estación base dentro de la red GSM. Existen dos versiones del algoritmo A5: A5/1 o versión fuerte (utilizado en Europa) y A5/2 o versión débil (utilizado fuera de Europa). Aunque el principio de funcionamiento de los dos algoritmos es muy parecido (basado en registros de desplazamiento realimentados), poseen diferencias muy marcadas a la hora del detalle de los mismos. En este caso se abordará la versión débil por corresponder con nuestro entorno práctico.

Una conversación GSM es enviada como una secuencia de tramas cada 4,6 ms. Cada trama contiene 114 bits correspondientes al enlace móvil 1-móvil 2 más 114 bits correspondientes al enlace móvil 2-móvil 1. Cada conversación puede cifrarse por una nueva clave de cifrado Kc. Para cada trama, Kc se mezcla con un número de trama públicamente conocido y el resultado sirve como el estado inicial del algoritmo que produce 228 bits pseudoaleatorios, los cuales se suman bit a bit en operación XOR con los 114+114 bits de texto claro para producir 114+114 bits de texto cifrado (este resultado es el que se transmite en el enlace radio). Cada 4,6 ms se envía una nueva trama inicializada con la misma Kc pero con diferente número de trama. El número de trama es cíclico y está entre 0 y 2'715.647; este número será incrementado al final de cada trama TDMA.

La representación general del sistema es como se muestra en la figura 1.

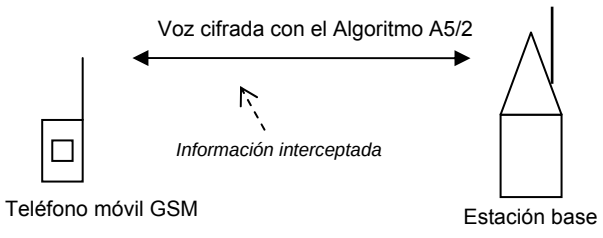

Figura 1. Representación global del sistema de cifrado.

OBJETIVOS

- Estudiar detalladamente las fortalezas y debilidades que presenta actualmente el algoritmo de cifrado para la voz en el sistema PCS.

- Proponer un nuevo algoritmo para el cifrado de la voz en el sistema PCS que mantenga un rendimiento óptimo y además elimine las debilidades que posee el existente actualmente.

- Dejar abierta la posibilidad de aumentar el nivel de seguridad que se maneja actualmente en el sistema PCS para la transmisión de la voz mediante la implementación práctica a futuro del algoritmo desarrollado.

1 ARQUITECTURA DE RED GSM

El sistema GSM se compone de cuatro subsistemas principales:

- Estación móvil (MS): conformada por el equipo móvil (ME) y la tarjeta SIM (módulo de identificación del abonado). Corresponde con la única parte de la red visible al abonado.
- Subsistema de estación base (BSS): conformado por el/los controlador(es) de la estación base (BSC) y el/los transceptor(es) de la estación base (BTS).
- Subsistema de red (NSS): su columna vertebral es el centro de conmutación de servicios móviles (MSC). Este subsistema también maneja bases de datos que registran la posición del usuario (HLR, VLR), permiten la autentificación del usuario (AuC), brindan seguridad y soportan el roaming (EIR).
- Centro de gestión de red (NMC): conformado por el/los centro(s) de operaciones y mantenimiento (OMC).

A continuación se ilustra la arquitectura de la red GSM:

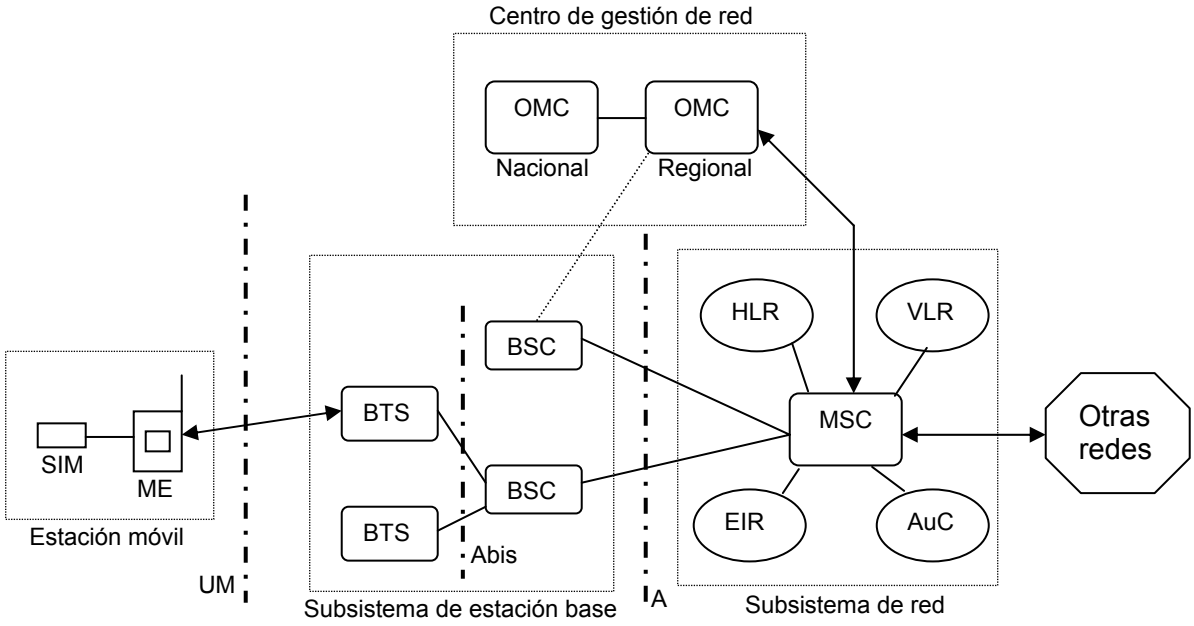

Figura 2. Arquitectura de red GSM.

1.1 ESTACIÓN MÓVIL

El equipo móvil está identificado dentro de cualquier red GSM por un número de 15 cifras denominado identidad internacional del equipo móvil (**IMEI**) que incluye un

código de aprobación GSM, la identidad del fabricante y el número serial del equipo. La tarjeta "inteligente" SIM contiene la identidad internacional del suscriptor móvil (IMSI) usada para identificar al abonado en cualquier red GSM. También contiene información referente a la seguridad y confidencialidad de la información transmitida en la red como el algoritmo de autentificación A3 y su clave, el algoritmo de generación de claves de cifrado A8 y la clave del algoritmo de cifrado A5.

1.2 SUBSISTEMA DE ESTACIÓN BASE

La estación base es la responsable de establecer, mediante la interfaz aérea, las comunicaciones con las estaciones móviles dentro de su área de cobertura. Constituye la interfaz entre la estación móvil y el subsistema de red. Dentro del BSS se encuentran el controlador y los transceptores de estación base comunicados entre sí mediante la interfaz Abis.

Las principales funciones del transceptor de la estación base incluyen la gestión de canales (full/half rate), supervisión de la relación de ondas estacionarias (SWR) en la antena, salto en frecuencia (FH), transmisión discontinua, monitoreo de las señales RF, detección de accesos a la red de las estaciones móviles, realización del proceso criptográfico de los datos transmitidos sobre el canal de radio (determinado por el centro de conmutación de servicios móviles y transmitido al BTS a través del BSC) y gestión de los algoritmos de clave.

Por su parte, el controlador de la estación base constituye la interfaz entre los transceptores de la estación base y el centro de conmutación de servicios móviles perteneciente al subsistema de red. Entre sus funciones principales se pueden mencionar la selección de celdas y canales apropiados para la comunicación, liberación de los canales, gestión del salto en frecuencia (FH), localización de las estaciones móviles, gestión de canales entre el BSC y el MSC, adaptación de la velocidad de transmisión (transcodificación a 64 Kbps), corrección de errores, handover intracelular (cambio de canal), control dinámico de la potencia de transmisión de la estación móvil y del BTS (gracias a las medidas del BTS).

1.3 SUBSISTEMA DE RED

Permite la interconexión (enrutamiento) de la red GSM con otras redes de telefonía fija o móvil y con usuarios en "roaming" a través del MSC. El MSC se diferencia fundamentalmente de una central de la red fija en que debe actualizar constantemente la posición de las estaciones móviles y soportar el traspaso entre celdas (handover) de las llamadas en curso. También se encarga de gestionar el tráfico de uno o más BSS y de la gestión de números variables en tipología de los usuarios. Estas funciones, junto con la autentificación del usuario, confidencialidad y la gestión de la seguridad de la red GSM, las realiza en conjunto con las 4 bases de datos mencionadas y que se describen a continuación:

- Registro de posición base (HLR): contiene información pertinente a los datos de suscripción de los abonados (servicios contratados) y a la localización de los mismos, permitiendo esta última, el encaminamiento de las llamadas que se les dirijan. El HLR está implementado en una estación de trabajo que además incluye otra información adicional como el IMSI (también contenido en la tarjeta SIM), el número RDSI internacional de la estación móvil y las características del equipo utilizado por el usuario.

- Registro de posición visitante (VLR): está asociado con uno o más MSC y se encarga del almacenamiento temporal de los datos (que incluyen posición) de los abonados del correspondiente MSC. Contiene además información particular como la identidad temporal del suscriptor móvil (TMSI) que se asigna en forma dinámica dependiendo del área de localización y que garantiza la confidencialidad del IMSI, el estado de la estación móvil (apagada, ocupada), datos sobre servicios suplementarios del abonado y la identidad del área de localización (LAI) de la asignación móvil (MA) dentro de la(s) celda(s) controlada(s) por el MSC correspondiente.

- Centro de autentificación (AuC): contiene una copia de seguridad de la clave secreta de la tarjeta SIM del abonado a fin de evitar el uso de la red por parte de terceros (identidad del abonado). Verifica la legitimidad de la SIM sin difundir a la red información propia del usuario como el IMSI. El procedimiento de autentificación se realiza cada vez que la MS emite o recibe una llamada, cuando hay actualización de su posición o cuando hace solicitud de servicios suplementarios.

- Registro de identidad de equipo (EIR): contiene listas de los equipos móviles (los identifica mediante su IMEI) que pueden y que no pueden hacer uso de la red GSM. La lista blanca contiene el IMEI de los equipos válidos en la red y el de los operadores internacionales con los que se tiene acuerdo de roaming. La lista gris contiene los equipos no homologados y defectuosos que pueden producir daños susceptibles a la red. Por último, la lista negra contiene los equipos que no pueden acceder a la red porque, por ejemplo, han sido reportados como robados.

1.4 CENTRO DE GESTIÓN DE RED

Corresponde a la máxima jerarquía dentro de la arquitectura GSM. Está conformado por el centro de operaciones y mantenimiento (OMC) el cual tiene acceso remoto a todos los elementos de la red GSM, lo que le permite controlar, supervisar y mantener el correcto funcionamiento del conjunto completo. Entre sus funciones específicas están la facturación de los usuarios, supervisión del tráfico de la red e introducción de variantes en el flujo, administración de los abonados y gestión de la configuración de la red. En redes mayores puede existir un OMC nacional encargado del control general y uno o varios OMC regionales limitados a zonas específicas.

2 INTERFACES DEL SISTEMA

2.1 INTERFAZ RADIO (UM)

La interfaz radio es utilizada por las estaciones móviles para acceder a todos los servicios del sistema GSM utilizando para ello los subsistemas de estación base como punto de conexión con la red.

2.2 INTERFAZ ENTRE EL MSC Y EL BSS (A)

Esta interfaz se utiliza fundamentalmente para el intercambio de información relacionada con las siguientes funciones:
- Gestión del BSS
- Manejo de la llamada
- Gestión de la movilidad

2.3 INTERFAZ ENTRE EL BSC Y EL BTS (A-BIS)

Esta interfaz permite conectar de forma normalizada transceptores y controladores de la estación base.

2.4 INTERFAZ ENTRE EL MSC Y EL VLR ASOCIADO (B)

El registro de posición visitante es la base de datos para gestión y seguimiento de los móviles dentro del área controlada por su MSC asociado (o MSC asociados).

2.5 INTERFAZ ENTRE EL MSC Y EL HLR ASOCIADO (C)

Esta interfaz se utiliza fundamentalmente para las siguientes funciones:
- Al final de una llamada en la que un móvil tiene que ser tarificado, el MSC de ese móvil puede enviar un mensaje de tarificación al HLR.
- Cuando la red fija no puede realizar el procedimiento de interrogación necesario para el establecimiento de una llamada hacia un usuario móvil, el MSC de cabecera debe interrogar al HLR del usuario llamado para conocer su numero de seguimiento.

2.6 INTERFAZ ENTRE EL HLR Y EL VLR (D)

Esta interfaz se utiliza para intercambiar los datos relacionados con la posición de la estación móvil y los datos de suscripción del usuario. A través de esta interfaz el VLR informa al HLR correspondiente la posición de una estación móvil gestionada por este último registro, proporcionándole un número de seguimiento a fin de que pueda

encaminar las llamadas dirigidas hacia esta estación móvil. En el otro sentido el HLR envía al VLR que controla el área donde se encuentra la estación móvil, los datos correspondientes necesarios para soportar los servicios contratados por el usuario. Asimismo, mediante una interfaz similar, el HLR debe informar también al VLR anterior que cancele el registro de localización correspondiente a dicha estación móvil, cuando esta estación móvil se desplaza a un área controlada por otro VLR. Estos intercambios de datos se producen cuando la estación móvil requiere un servicio determinado, cuando el usuario quiere cambiar algunos datos relacionados con su suscripción, o bien cuando los parámetros de la suscripción se modifican por el operador del sistema.

2.7 INTERFAZ ENTRE DOS MSC (E)

Cuando una estación se desplaza del área controlada por un MSC al área de otro MSC, es necesario realizar un procedimiento de traspaso para poder continuar la conversación. En este caso las MSC deben intercambiar datos, a través de la interfaz E, para poder llevar a cabo esta operación.

3 CARACTERÍSTICAS GENERALES DEL SISTEMA PCS

3.1 REUTILIZACIÓN DE FRECUENCIAS

La idea fundamental en la que se basan los sistemas móviles celulares es la reutilización de las frecuencias mediante la división del terreno en celdas continuas que se iluminan desde una estación base con unos determinados canales. La reutilización de frecuencias no es posible en celdas adyacentes, pero sí en otras más alejadas. El número de veces que un canal puede ser reutilizado es mayor cuanto más pequeñas sean las celdas.

De esta manera, la red celular se compone de un conjunto de estaciones base desplegadas por el territorio a cubrir que están interconectadas o conectadas un con un centro de conmutación con acceso a la red telefónica publica, a la RDSI o a otra red celular móvil. La estación base que recibe al móvil con un mayor nivel de potencia es la que queda asignada al mismo.

3.2 EL SISTEMA DE CELDAS

El objetivo de un sistema celular es reutilizar canales, pero al estar estos canales asociados a estaciones base, se deben repetir estaciones base. Una estación base se repite cuando tiene la misma tabla de frecuencias que otra.

Asumiendo condiciones de terreno plano, las estaciones base iluminarían áreas formando triángulos equiláteros. Las bases se despliegan de forma irregular según el terreno, buscando un mínimo de zonas de "sombra". El problema de la red está en determinar la ubicación idónea de las estaciones base para conseguir una mayor cobertura y minimizar las zonas de sombra. Lo normal es que las estaciones base tengan un diagrama de radiación omnidireccional, es decir, que transmitan en todas las direcciones con la misma potencia y frecuencias. No obstante, según la concentración de usuarios y para el mejor aprovechamiento del espectro y de la potencia radiada por las antenas, se puede sectorizar la radiación dirigiendo la potencia hacia una zona específica. El caso más común de sectorización es en el que cada estación base alimenta 3 antenas cada una de las cuales abarca 120º. En la práctica, en zonas muy congestionadas por la demanda de comunicaciones móviles, se instalan seis antenas en cada estación base que suponen seis sectores de 60º, en cuyo centro está la estación base. Dependiendo del movimiento de los móviles, éstos podrían cambiar de asignación de antena siguiendo con la misma estación base.

4 NIVELES DE COMUNICACIÓN

Para el control de las llamadas, la transferencia información y la gestión global del sistema, se requiere que el sistema GSM utilice varios protocolos. Desde la perspectiva de la estación móvil existen cuatro niveles para la comunicación:

a) La interfaz RF (Radio Frecuency) al BTS.
b) El nivel de gestión de recursos de radio (RR, Radio Resources) al BSC.
c) Gestión de la movilidad (MM, Mobility Management).
d) Gestión de las comunicaciones (CM, Communications Management) al registro VLR del MSC.

Además, se utilizan protocolos adicionales para proporcionar servicios de control que se gestionan entre el sistema de conmutación y los componentes de gestión. El canal de transmisión entre la MS y el BTS es el único componente que se repite en las redes celulares GSM, modificado para funcionar sobre diferentes frecuencias en el caso de PCS y reemplazado totalmente en el caso de sistemas de comunicación por satélite. La interfaz entre la MS y el BTS consta de un canal TDMA de salto en frecuencia que se divide en varios subcanales de los cuales unos se utilizan para la transmisión de la información del usuario y el resto se utiliza para los protocolos de control.

5 DESCRIPCIÓN DEL ACCESO AL MEDIO EN GSM-1900MHZ

El estándar GSM se presenta como una plataforma para las comunicaciones inalámbricas, en las bandas de frecuencia de 900 MHz (Sistema de telefonía móvil celular), 1800 MHz (DCS – Sistema celular digital) y **1900 MHz** (PCS – Servicios de comunicación personal), entre otras. En cualquiera de ellas, GSM utiliza la tecnología de acceso al medio por división en frecuencia (**FDMA**) combinada con la de acceso al medio por división en tiempo (**TDMA**). La parte FDMA hace referencia a la división del ancho de banda disponible en diferentes frecuencias (canales) y TDMA se refiere a la subdivisión de cada frecuencia en intervalos de tiempo (gracias a esto varios usuarios pueden utilizar la misma frecuencia).

Con FDMA se utiliza el ancho de banda disponible dividiéndolo en diferentes canales de acuerdo con la frecuencia. Cada uno de estos canales es una subdivisión del recurso total. Con TDMA se utiliza el sub-ancho de banda disponible mediante la asignación de intervalos de tiempo. Cada usuario utiliza el recurso disponible durante un cierto tiempo. Esto aumenta la capacidad de transmisión del sistema pero limita el número de usuarios que usan el recurso en determinado momento. Esto conlleva a la transmisión de información por ráfagas.

GSM-1900 utiliza dos bandas de 60 MHz para transmitir y para recibir información (FDD – Frecuency Division Duplexing) correspondientes al enlace ascendente móvil-base (1850 MHz – 1910 MHz) y al enlace descendente base-móvil (1930 MHz – 1990 MHz). Estas bandas inferior y superior e están divididas (FDMA) cada una en 299 canales (portadoras) de 200KHz de ancho de banda (número de canales de radiofrecuencia absolutos o ARFCN), cada uno subdividido (TDMA) en 8 “slots” o intervalos de tiempo (8 llamadas por canal). La tasa de información enviada por el aire por cada portadora es 270 kbps y como hay 8 canales físicos (intervalos), la información se envía en paquetes por intervalos de tiempo (ráfagas). La tasa efectiva de transmisión por canal físico es entonces 33,75 kbps (270 kbps / 8).

La ventaja de enviar información por ráfagas es el ahorro de energía en la transmisión ya que se emplea 1/8 del tiempo normal para el enlace ascendente y 1/8 para el descendente, consiguiendo con esto mayor duración de las baterías del equipo móvil. Sin embargo, el uso de TDMA tiene el inconveniente que requiere constante sincronización y monitoreo, necesitando mayor robustez. Además existe un problema llamado “alineamiento temporal” el cual consiste en la pérdida de sincronismo desde y hacia la estación móvil al alejarse de la antena, debido a que la señal requerida se va desfasando hasta que termina por salirse de su canal físico, interfiriendo con los canales adyacentes.

5.1 TRAMA TDMA

Una portadora en GSM se compone de 8 canales físicos. Cada uno de estos canales es un intervalo de tiempo dentro de una trama TDMA. La trama TDMA se compone de

8 intervalos de tiempo, cada uno con una duración aproximada de 0,577 ms y por tanto la trama TDMA tiene 4,616 ms (8 x 0,577 ms) de duración. 26 ó 51 tramas forman una multitrama TDMA, 51 ó 26 multitramas forman una supertrama y 2048 supertramas forman una hipertrama TDMA.

Formato de trama TDMA:

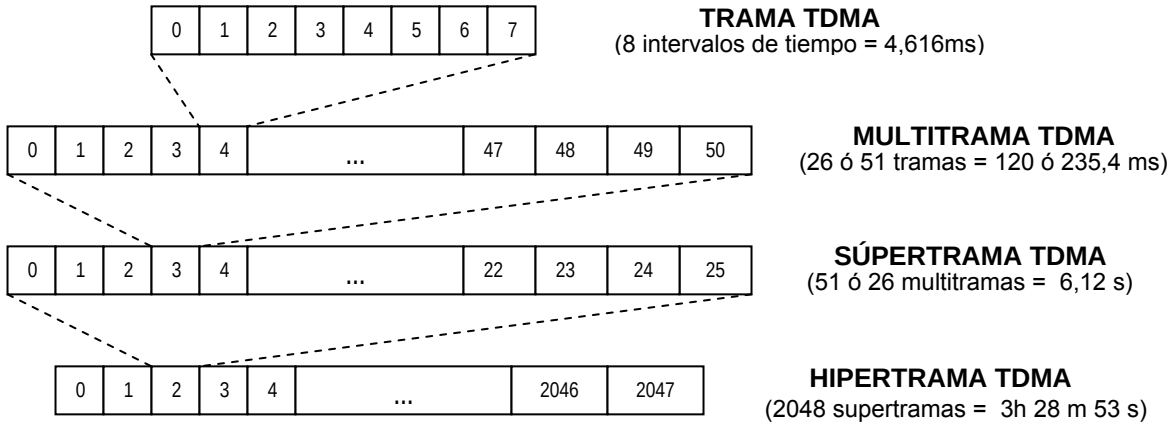

Figura 3. Descripción del canal físico GSM.

Una multitrama puede estar conformada por 26 o por 51 tramas. La primera contiene canales de tráfico y sus canales de control asociados (TCH, SACCH y FACCH) y la segunda contiene exclusivamente canales de control (BCCH, CCCH, SDCCH y SACCH). 51 multitramas de 26 tramas o 26 multitramas de 51 tramas forman una supertrama. Por último, 2048 supertramas forman una hipertrama, la cual se usa como base para la numeración de las tramas TDMA que va, por tanto, desde 0 hasta 2'715.647 (51x26x2048 - 1).

El canal de control asociado lento (SACCH) que envía información acerca de la transmisión, es usado tanto en modo activo (ciclos de 26 multitramas, durante la conversación) como en modo libre (explorando nuevas celdas, ciclos de 51 multitramas). De esta forma se asegura que el uso de la energía sea eficiente en todo momento y que no existan caídas de la señal aún cuando se está hablando.

Adicional a lo anterior, dentro de cada canal físico existen los llamados canales lógicos que incluyen canales de tráfico (TCH) y canales de control (CCH). Los primeros soportan voz digital (canales de tráfico de voz) o datos de usuario (canales de tráfico de datos). Los canales de control permiten la señalización y sincronización entre la estación base (BS) y la estación móvil (MS). Pueden ser unidireccionales para la difusión de información a las estaciones móviles (canales de difusión), usarse para el establecimiento de la conexión entre la MS y la BS (canales de control común) antes de la asignación de un canal de control entre ambas estaciones (canales de control dedicado).

El uso de los anteriores canales dentro del sistema GSM y debido a los principios de operación del mismo, se hace mediante la transmisión de diferentes tipos de ráfagas entre la MS y la BS: ráfaga normal, de acceso, de relleno, de corrección de frecuencia y de sincronización. Dichas ráfagas se componen, en términos generales, de una parte útil y una de guarda. La primera contiene los datos a transmitirse, una secuencia de entrenamiento y una cola de bits. La segunda tiene como propósito evitar que se traslapen las partes útiles de las ráfagas adyacentes (disminuir el problema de alineamiento temporal).

5.2 CANALES LÓGICOS

5.2.1 Canales de tráfico (TCH)

Los canales de tráfico se utilizan para voz y para datos. En los canales de tráfico de voz se manejan velocidades de transmisión de 22,8 kbps (“full rate”), llamados TCH/FS y de 11,4 kbps (“half rate”) denominados TCH/HS. En los canales de tráfico de datos las velocidades especificadas son de 9,6 kbps “full rate” (TCH/F9,6), 4,8 kbps “full rate” (TCH/F4,8), 4,8 kbps “half rate” (TCH/H4,8), ≤2,4 kbps “half rate” (TCH/H2,4) y ≤2,4 kbps “full rate” (TCH/F2,4).

Los canales de tráfico utilizan algunos de los "time slots" (TS) de entre el número 1 o cualquiera del 3 al 7 de la portadora cero y si existen más portadoras en una celda todas ellas se subdividen en canales de tráfico.

5.2.2 Canales de control (CCH)

5.2.2.1 Canales de difusión (BCH)

Los canales de difusión transmiten información desde la estación base hacia todas las estaciones móviles dentro de su área de influencia, ya que la información enviada es vital para la identificación y acceso a la red. Esta conexión es descendente punto a multipunto. De esta manera, permiten que la MS se sintonice con una estación base y se sincronice con la estructura de trama de la celda. Las estaciones base no están sincronizadas entre sí, luego al pasar de una celda a otra, la MS debe contactar nuevamente el BCH de la estación vecina. Estos canales se transmiten en una portadora específica para cada celda, conocida como portadora cero. Dentro de esa portadora, estos canales se ubican en el intervalo de tiempo cero (TS0).

Dentro de los canales de difusión se encuentran el canal de corrección de frecuencia (FCCH), el canal de sincronización (SCH) y el canal de control de difusión (BCCH):

- Canal de corrección de frecuencia: lleva información para la corrección de la frecuencia de la estación móvil y se requiere sólo para la operación del subsistema

radio. Para tal efecto, transmite una señal sinusoidal para indicar que es una portadora de difusión.

- Canal de sincronización: permite que la MS establezca la debida conexión e identifique un BTS. Este canal permite asegurar que la estación base contactada pertenezca a la red contratada por el usuario; además realiza la identificación del número de trama TDMA de la estación base para que la MS se sincronice con ésta. Específicamente contiene dos parámetros codificados: código de identidad del BTS (BSIC - Base Station Identity Code) de 6 bits y el número de trama TDMA reducido (RFN - Reduced Frame Number) de 19 bits. También se requiere sólo para la operación del subsistema radio.

- Canal de control de difusión: este canal transmite información acerca de la localización de la celda, la máxima potencia aceptada por la estación base y las portadoras BCH de celdas vecinas para que la MS realice medidas de potencia en ellas para el handover.

5.2.2.2 Canales de control común (CCCH)

Estos canales gestionan el establecimiento de la comunicación desde y hacia la MS cuando se quiere realizar o recibir una llamada. La conexión en este caso es punto a punto y es ascendente o descendente según sea el canal. La ubicación de estos canales es en la portadora cero, en el TS0 al igual que los canales de difusión. Dentro de estos canales se encuentran:

- Canal de búsqueda (PCH): la MS "escucha" este canal para verificar si alguien quiere establecer contacto con ella. Este canal es usado sólo en enlace descendente.

- Canal de acceso aleatorio (RACH): la MS responde a la petición de búsqueda a través de este canal pidiendo un canal de señalización (SDCCH). También se usa cuando se quiere realizar una llamada. Este canal es usado sólo en enlace ascendente.

- Canal de acceso garantizado (AGCH): informa a la MS la asignación de un canal de señalización o directamente de un canal de tráfico. Este canal es usado sólo en enlace descendente.

- Canal de notificación (NCH): usado sólo en enlace descendente para notificar a las estaciones móviles de las llamadas de voz de grupo y de las llamadas de voz de difusión.

Tanto los BCH como los CCCH utilizan los dos primeros TS de la portadora cero, los cuales se repiten constantemente siguiendo un patrón cíclico de duración 51 TS.

5.2.2.3 Canales de control dedicado (DCCH)

Estos canales permiten la conmutación hacia algún canal de tráfico y además llevan mensajes cortos de texto en ambos sentidos. Adicionalmente, a través de ellos se envía información recogida por la MS hacia la BS para que la red evalúe el traspaso a otra celda ("handover"). El enlace es de punto a punto.

- Canal de control dedicado independiente (SDCCH): es un canal de señalización asignado por la red. Permite el proceso de establecimiento de una llamada ordenándole a la MS conmutar hacia un canal de tráfico, definiéndole una portadora y un intervalo de tiempo. En él también se transmiten mensajes de texto. Este canal se usa tanto en enlace ascendente como descendente y se transmite en la portadora cero en el TS2.

- Canal de control asociado lento (SACCH): durante el enlace ascendente la MS envía a través de él mediciones de potencia y calidad de la señal, procedente de su estación base y de la potencia recibida de las bases vecinas. En el enlace descendente, la MS recibe medidas acerca de cuál potencia de transmisión debe utilizar e instrucciones de avance temporal. Este canal se transmite en la portadora cero en el TS2.

- Canal de control asociado rápido (FACCH): este canal envía información necesaria para el "handover" durante una conversación, lo cual se logra "robando" 20 ms de voz del canal de tráfico. Este canal lógico ocupa parte de un canal de tráfico.

A continuación se especifican estos canales de control dedicado:

- Slow, TCH/F associated, control channel (SACCH/TF)
- Fast, TCH/F associated, control channel (FACCH/F)
- Slow, TCH/H associated, control channel (SACCH/TH)
- Fast, TCH/H associated, control channel (FACCH/H)
- Stand alone dedicated control channel (SDCCH/8)
- Slow, SDCCH/8 associated, control channel (SACCH/C8)
- Stand alone dedicated control channel, combinado con CCCH (SDCCH/4)
- Slow, SDCCH/4 associated, control channel (SACCH/C4)

El último de los canales de control es el canal de difusión de celda (CBCH) usado sólo en enlace descendente y que se usa para el servicio de difusión en celdas de mensajes cortos (SMSCB).

La figura 4 resume la distribución de los canales lógicos.

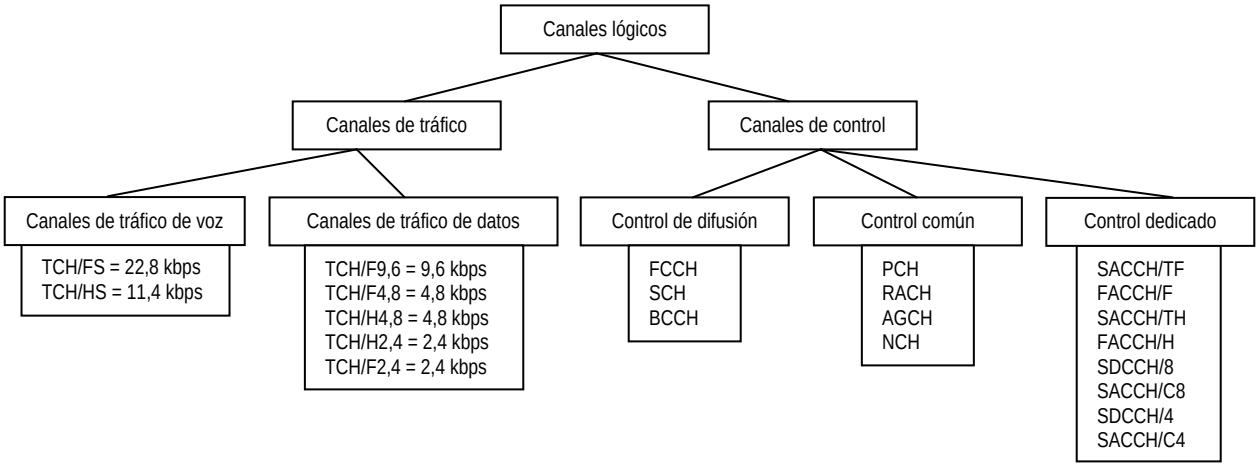

Figura 4. Clasificación de los canales lógicos.

5.3 CANALES DE RADIOFRECUENCIA

El sistema GSM en cualquiera de sus bandas de operación (450, 480, 850, 900, 1800/DCS, 1900/PCS MHz) dispone de un recurso espectral sobre el que debe funcionar. Este recurso físico está dividido tanto en frecuencia como en tiempo. La partición en frecuencia hace referencia a los canales de radiofrecuencia (RFCHs). La partición en tiempo hace referencia a los “time slots” y a las tramas TDMA dentro de cada RFCH. Cada uno de estos canales está numerado dentro del sistema y un subconjunto de los mismos es asignado a cada celda (asignación de celda, CA). Un canal de radiofrecuencia de la CA (conocido como portador BCCH) se usará para transportar información de sincronización y el BCCH. El subconjunto de la asignación de celda asignado a un móvil particular se conoce como asignación móvil (MA).

Los canales de radiofrecuencia del sistema PCS (GSM-1900) están distribuidos en la siguiente banda:

- 1850 MHz a 1910 MHz: enlace ascendente (móvil – base); denominada banda inferior.
- 1930 MHz a 1990 MHz: enlace descendente (base – móvil); denominada banda superior.

El espaciamiento entre portadoras es de 200 KHz y por tanto en los 60 MHz disponibles hay 300 portadoras (en realidad hay 299 portadoras; una banda de guarda de 200 KHz). La frecuencia portadora está designada por el número de canal de radiofrecuencia absoluto (ARFCH). A continuación se especifica la distribución de las portadoras sobre las bandas inferior y superior del sistema PCS-1900 MHz:

	Frecuencia portadora banda inferior		Frecuencia portadora banda superior
PCS 1900 MHz	$f_L = 1850.2 + 0.2 \times n$ [MHz]	$0 \leq n \leq 298$	$f_U = f_L + 80$ [MHz]

Tabla 1. Distribución de los canales de radiofrecuencia en el Sistema PCS.

La figura 5 ilustra la distribución de los canales físicos ("time slots") y la formación de las tramas TDMA dentro de los canales de radiofrecuencia. En el enlace descendente, las tramas TDMA de todos los canales de radiofrecuencia se alinearán en el BTS. Lo mismo aplica para el enlace ascendente. El número de trama es cíclico y está entre 0 y 2'715.647; el mismo será incrementado al final de cada trama TDMA.

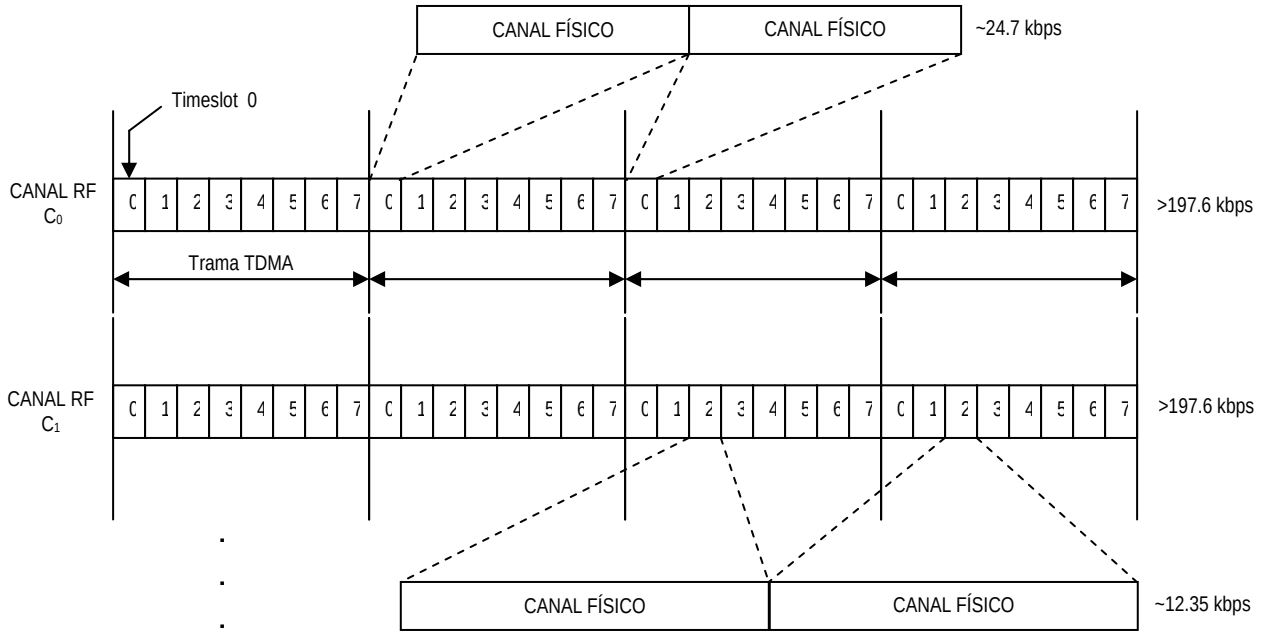

Figura 5. Distribución de los canales físicos y de las tramas TDMA dentro de cada canal de radiofrecuencia.

5.4 CANALES FÍSICOS

Un canal físico utiliza una combinación de multiplexación por división en frecuencia y en tiempo y está definido como una secuencia de canales de radiofrecuencia y de "time slots". Para definir completamente un canal físico particular se requiere hacer la descripción en el dominio de la frecuencia y en el dominio del tiempo.

Respecto a la frecuencia, la secuencia del canal de radiofrecuencia está determinada por una función que en una celda dada, con un conjunto particular de parámetros

generales, con un número de “time slot” específico, con una asignación del canal de radiofrecuencia a la estación móvil (MA) y un índice offset de la estación móvil, mapea el número de trama TDMA en un canal de radiofrecuencia. Por consiguiente, en una celda dada, para un canal físico asignado a un móvil particular, hay una correspondencia única entre el canal de radiofrecuencia y el número de trama TDMA.

Respecto al tiempo, un canal físico dado siempre usará el mismo número de “time slot” en cada trama TDMA. Por consiguiente, una secuencia “time slot” está definida por el número de “time slot” y por la secuencia del número de trama TDMA.

El contenido físico de un “time slot” se denomina “burst” (o ráfaga). Específicamente, es un periodo de la portadora RF que está modulado por una corriente de datos. Un “time slot” está dividido en periodos de 156,25 bits. Un periodo de bit particular dentro de un “time slot” está referenciado por un número de bit (BN), con el primer periodo numerado como 0 y el último ¼ de bit como 156.

En el sistema existen diferentes tipos de “burst”. Una característica de un “burst” es su duración útil. A continuación se enuncian cuatro “burst” completos de 148 bits de duración útil y un “burst” corto de 88 bits de duración útil:

- "Burst" normal (bits útiles 0-147, bits de guarda 148-156): se utiliza para transmitir información de tráfico y canales de control. Lleva los canales TCH, BCCH, PCH, AGCH, SDCCH, SACCH y FACCH.

TB	bits encriptados	F	secuencia de entrenamiento	F	bits encriptados	TB
3	57	1	26	1	57	3

  Los bits encriptados en paquetes de 57 contienen voz o datos más un bit de bandera (F) que señala "modo robado" para indicar que en su lugar va el canal de señalización FACCH. La secuencia de entrenamiento es un patrón de voz conocido por el receptor quien crea un modelo de canal capaz de corregir los errores en la recepción causados por la interferencia entre símbolos (ISI). Los bits de cola (TB) son siempre (0,0,0) e indican al ecualizador que es un punto de arranque / parada.

  Como la duración de un intervalo de tiempo es 0,577 ms transmitidos a 270,833 kbps, implica que hay disponibles 156,25 bits de los cuales hay 148 ocupados. Los restantes 8,25 bits equivalen a 30 μs de período de guarda (GP). Este período permite al transmisor desplazarse por la celda evitando el solapamiento con los otros canales (alineamiento temporal).

- "Burst" de corrección de frecuencia (bits útiles 0-147, bits de guarda 148-156): se utiliza para la sincronización de la frecuencia del móvil.

TB	patrón de bits	TB	GP
3	142	3	8.25

- “Burst” de sincronización (bits útiles 0-147, bits de guarda 148-156): se utiliza para la sincronización de la trama del móvil.

TB 3	bits encriptados 39	secuencia de sincronización 64	bits encriptados 39	TB 3	GP 8,25

- “Burst” de relleno (bits útiles 0-147, bits de guarda 148-156): esta ráfaga la envía el BTS y no contiene información. El formato es idéntico al de una ráfaga normal con un cierto patrón de bits.

TB 3	patrón de bits 58	secuencia de entrenamiento 26	patrón de bits 58	TB 3	GP 8.25

- “Burst” de acceso (bits útiles 0-87, bits de guarda extendida 88-156): esta ráfaga se utiliza para acceder por primera vez a un nuevo BTS después de un handover. El móvil puede encontrarse lejos de la estación base, lo que indica que la ráfaga inicial llegará más tarde, la cual debe ser más corta para evitar el solapamiento.

TB 8	secuencia de sincronización 41	bits encriptados 36	TB 3	GP 68.25

El periodo existente entre “burst” que aparecen en “time slots” consecutivos se llama periodo de guarda. Este periodo es necesario porque se requiere para las estaciones móviles que la transmisión sea atenuada en ese periodo con el necesario cambio de nivel que ocurre durante el mismo. No es necesario que un BTS tenga la capacidad de cambiar de nivel entre “burst” adyacentes, pero se requiere que tenga la capacidad de hacerlo para los “time slots” no usados. En cualquier caso donde la amplitud de la transmisión sea aumentada y disminuida, se puede minimizar la interferencia con otros canales de radiofrecuencia mediante la aplicación de una corriente de bits de modulación apropiada.

5.5 MAPEO DE LOS CANALES LÓGICOS SOBRE LOS CANALES FÍSICOS

Existe un conjunto de parámetros necesarios para describir completamente el mapeo de cualquier canal lógico sobre un canal físico. Estos parámetros se pueden dividir en parámetros generales (característicos a un BTS particular) y en parámetros específicos (característicos de un canal físico dado). Para realizar el mapeo (en frecuencia) desde el número de trama TDMA hasta un canal de radiofrecuencia para un canal particular asignado, se requiere lo siguiente:

- Parámetros generales del BTS, específicos a un BTS y con difusión en el BCCH y SCH:
  - CA: asignación de celda de los canales de radiofrecuencia.
  - FN: número de trama TDMA difundido en el SCH.

- Parámetros específicos del canal definidos en el mensaje de asignación del canal:
  - MA: asignación móvil de los canales de radiofrecuencia la cual define el conjunto de canales de radiofrecuencia a usarse en la secuencia de salto de los móviles. Este parámetro contiene N canales de radiofrecuencia, donde 1≤N≤64.
  - MAIO: asignación móvil del índice offset (0 hasta N-1, 6 bits).
  - HSN: número (generador) de secuencia de salto (0 hasta 63, 6 bits).

El índice a un número de canal de radiofrecuencia absoluto MAI va desde 0 hasta N-1, donde MAI=0 representa el número de canal de radiofrecuencia absoluto (ARFCN) más bajo en la asignación móvil. El ARFCN está en el rango de 0 hasta 7023 y el valor de la frecuencia se puede determinar de la tabla 1 con n = ARFCN.

Para realizar el mapeo en tiempo de los canales lógicos sobre los canales físicos se tienen en cuenta aspectos como el tipo de canal lógico al que se aplica el mapeo, la dirección del mapeo (ascendente o descendente), los “time slots” que permiten o soportan el mapeo, los canales de radiofrecuencia en una asignación de celda sobre los que se pueden mapear los canales lógicos y el tipo de “burst” que se usará en el canal físico.

6 PROCESAMIENTO DE LA VOZ

Debido al carácter natural del sistema GSM la voz a transmitirse se debe digitalizar. El método que se emplea en RDSI y en los sistemas de telefonía actuales para la multiplexación de las líneas de voz sobre troncales de alta velocidad y sobre líneas de fibra óptica es la modulación por codificación de pulsos (PCM). La corriente de salida de PCM es 64 kbps, muy alta para ser factible sobre un enlace de radio.

La red GSM tiene una ruta digital de radio en la que se transfieren bits a través de la interfaz aire. Debido a este formato de bits, la información transferida emplea un espectro de frecuencias considerablemente ancho; cuanto mayor sea la velocidad de bits usada, se necesitará un espectro más ancho. Si el espectro es ancho se reduce la capacidad de la interfaz aire.

Para mantener la capacidad de la interfaz aire, el sistema GSM utiliza un sistema de codificación llamado RPE-LTP (Regular Pulse Excitation - Long Term Prediction), el cual es una forma más efectiva de convertir voz en bits. Con este sistema, la codificación de voz ocupa 13 kbps en lugar de 64 kbps. Básicamente, la información de las muestras previas, la cual no cambia muy rápido, se utiliza para predecir la muestra actual. La representación de la señal se obtiene con los coeficientes de la combinación lineal de las muestras previas más una forma codificada del residuo y la diferencia entre la muestra predicha y la real.

El procesamiento de la voz se refiere a las funciones que el BTS y la MS realizan para garantizar una transferencia de información libre de errores a través del interfaz aire. Estas funciones son la codificación de bloques, la codificación convolucional y el entrelazado ("interleaving"). La codificación de bloques es para la detección de errores, la codificación convolucional para la corrección de errores y el entrelazado es tanto para detección como para corrección.

6.1 CODIFICACIÓN

El algoritmo de codificación de voz utilizado en GSM está basado en un codificador predictivo lineal excitado por impulso rectangular con predicción a largo termino (RPE-LTP). El codificador de voz produce muestras a intervalos de 20 milisegundos a una tasa de bits de 13 kbps, produciendo 260 bits por muestra o trama.

Debido a la interferencia electromagnética natural, la voz codificada o la señal de datos transmitida sobre la interfaz de radio se debe proteger contra los errores. Para lograr esto, GSM utiliza codificación convolucional y entrelazado en bloques. Los algoritmos exactos utilizados difieren para la voz y para las diferentes tasas de datos. El método usado para los bloques de voz se describe a continuación.

El codificador de voz produce un bloque de 260 bits por cada muestra de voz de 20 ms. Algunos bits de ese bloque son más importantes para la calidad de la voz percibida que otros. Estos bits están divididos en tres clases:

- Clase Ia (50 bits): los más sensibles a los errores de bit
- Clase Ib (132 bits): moderadamente sensibles a los errores de bit
- Clase II (78 bits): los menos sensibles a los errores de bit

Los bits de la clase Ia tienen un código de redundancia cíclica de 3 bits agregado para la detección de errores. Si se detecta un error, la trama se considera muy dañada para ser comprensible y se descarta. Ella se reemplaza por una versión ligeramente atenuada de la trama previa recibida correctamente. Estos 53 bits, junto con los 132 bits de clase Ib y una secuencia de cola de 4 bits (un total de 189 bits), se introducen dentro de un codificador convolucional de tasa 1/2 con restricción de longitud de 4. Cada bit de entrada se codifica como dos bits de salida, basado en una combinación de los 4 bits de entrada previos. De ahí que el codificador convolucional saca 378 bits, a los cuales se le suman los restantes 78 bits de clase II bits, los cuales no están protegidos. Así, cada 20 ms se codifica la muestra de voz en 456 bits, dando una tasa de bits de 22,8 kbps.

Para protección adicional contra los errores por ráfagas comunes a la interfaz radio, cada muestra es entrelazada. Los 456 bits de salida del codificador convolucional se dividen en 8 bloques de 57 bits que son entrelazados con el bloque anterior o posterior. A los bloques resultantes de 114 bits se les añaden dos bits (uno por cada 57 bits) que indican si los bits del enlace de voz han sido sustituidos por datos de FACCH. Estos nuevos bloques de 116 bits y son los que forman los "burst" que se transmiten en "time slots" consecutivos. Ya que cada "time slot" transporta dos bloques de 57 bits, cada "burst" transporta tráfico de dos muestras de voz diferentes.

6.2 MODULACIÓN

Según se anotó, el contenido de cada "time slot" se transmite a una tasa de bit total de 270,833 kbps. Esta señal digital es modulada sobre la frecuencia de portadora analógica con el uso del esquema Gaussian-filtered Minimum Shift Keying (GMSK). GMSK supera a otros esquemas de modulación porque mantiene un compromiso entre la eficiencia espectral, la complejidad del transmisor y las emisiones falsas limitadas. La complejidad del transmisor se relaciona con el consumo de potencia, la cual debe minimizarse para la estación móvil. Las emisiones falsas de radio, por fuera del ancho de banda asignado, deben ser controladas estrictamente con el fin de limitar la interferencia con los canales adyacentes y permitir la coexistencia de GSM con los sistemas analógicos antiguos.

6.2.1 Modulación digital en GSM

El esquema de modulación usado en GSM es 0,3 GMSK, donde 0,3 describe el ancho de banda del filtro gausiano con relación a la tasa de bits de la señal (BT=0.3). GMSK es un tipo especial de modulación FM. Los unos y ceros binarios se representan en GSM por desplazamientos en frecuencia de ±67,708 KHz. La velocidad de datos en GSM es de 270,833 kbps, que es exactamente cuatro veces el desplazamiento en frecuencia. Esto minimiza el ancho de banda ocupado por el espectro de modulación y por tanto mejora la capacidad del canal. La señal MSK modulada se pasa a través de un filtro gausiano para atenuar las variaciones rápidas de frecuencia que de otra forma esparcirían energía en los canales adyacentes.

6.2.2 Modulación MSK ("Minimum Shift Keying")

MSK es un tipo especial de FSK ("Frecuency Shift Keying"), con fase continua y un índice de modulación de 0.5. El índice de modulación de una señal FSK es similar al de FM y se define por kFSK = (2DF)/Rb, donde 2DF es el desplazamiento en frecuencia de pico a pico y Rb es la tasa de bits. Un índice de modulación de 0,5 corresponde con el mínimo espacio en frecuencia que permite que dos señales FSK sean ortogonales coherentes, y el nombre MSK implica la mínima separación en frecuencia que permite una detección ortogonal. Dos señales FSK son ortogonales si MSK es una modulación espectralmente eficiente. La modulación MSK posee propiedades como envolvente constante, eficiencia espectral, buena respuesta ante los errores de bits y capacidad de autosincronización.

6.2.3 Modulación GMSK ("Gaussian Minimum Shift Keying")

GMSK es un esquema de modulación binaria simple que se puede ver como derivado de MSK. En GMSK, los lóbulos laterales del espectro de una señal MSK se reducen pasando los datos NRZ modulantes a través de un filtro gausiano de premodulación. El filtro gausiano aplana la trayectoria de fase de la señal MSK y por lo tanto, estabiliza las variaciones de la frecuencia instantánea a través del tiempo. Esto tiene el efecto de reducir considerablemente los niveles de los lóbulos laterales en el espectro transmitido.

El filtrado convierte la señal (donde cada símbolo en banda base ocupa un periodo de tiempo T) en una respuesta donde cada símbolo ocupa varios periodos. Sin embargo, dado que esta conformación de pulsos no cambia el modelo de la trayectoria de la fase, GMSK se puede detectar coherentemente como una señal MSK, o no coherentemente como una señal simple FSK. En la práctica, GMSK es muy atractiva por su excelente eficiencia de potencia y espectral.

7 PROBLEMAS DE TRANSMISIÓN

- Pérdidas debido a la distancia: ocurren cuando la MS se aleja de la BSS, incluso sin obstáculos entre la antena transmisora y la receptora. La potencia entregada por la antena disminuye conforme aumenta la distancia.

- Desvanecimiento: ocurre debido a la existencia de obstáculos físicos como montañas, edificios y árboles. La fuerza de la señal disminuye al pasar por la "sombra" proyectada del objeto. Otro tipo de desvanecimiento es el de "Rayleigh" provocado por el rebote de la señal al recorrer muchos caminos, desfasando la señal al momento de ser recibida y creando pérdidas en la fuerza de la transmisión.

- Desvanecimiento total: ocurre cuando la fuerza de la señal recibida está por debajo de un cierto valor de umbral provocando la pérdida de información. Ese valor de umbral se llama sensibilidad del receptor.

- Alineamiento temporal (o avance temporal): TDMA requiere que la estación móvil transmita sólo en el intervalo de tiempo asignado y que permanezca en silencio el resto del tiempo para no interferir con otras transmisiones que usan el mismo canal. Si la MS se aleja de la BSS, la información tarda más tiempo en llegar y como consecuencia, el móvil demora en responder, haciendo uso del tiempo destinado a otras transmisiones, interfiriendo con ellas.

- Dispersión en el tiempo: es otro de los problemas que aparece en la transmisión digital cuya consecuencia es la interferencia entre símbolos (ISI). El receptor se confunde al recibir simultáneamente un 0 y un 1, que aunque han sido enviados por separado y secuencialmente, el segundo ha tomado una ruta más rápida que el primero, llegando ambos al mismo tiempo.

8 PROPIEDADES DEL SISTEMA

8.1 ALINEAMIENTO ADAPTATIVO EN EL TIEMPO

En el sistema GSM, como en la mayoría de los sistemas celulares, la MS obtiene su temporización de las señales recibidas de la BS. En concreto, la MS transmite su "burst" 3 intervalos de tiempo (3 x 0,577 ms) después que los "bursts" hayan sido recibidos de la BS. No obstante, dado que la temporización depende de lo que tarde en propagarse la señal, que a su vez depende de la distancia entre BS y MS y que el siguiente "burst" recibido en la BS (procedente de otra MS con diferente distancia a la BS) puede superponerse, deben ser tomadas algunas acciones al respecto.

La BS determina el adelanto en la temporización de la transmisión que el móvil debe tener para que sus "bursts" lleguen en el intervalo de tiempo correcto. Este adelanto de temporización es inicialmente calculado por la BS sobre la base del "burst" de acceso recibido en el RACH (que tiene un periodo de guarda de 68,25 bits) y puede ser de 0 a 63 periodos de bit de avance lo que equivale a una separación máxima de 35 Km entre MS y BS.

La BS controla en modo de operación normal con el TCH establecido, el retardo de la señal procedente de la MS, enviando órdenes de corrección en el SACCH y logrando que el error del retardo sea menor que 2 μs (aproximadamente medio periodo de bit).

8.2 CONTROL DE POTENCIA

El control de potencia en el sistema GSM puede ser utilizado tanto en la MS como en la BS y su finalidad principal es la de reducir la interferencia cocanal mientras se trabaja con una potencia transmisora adecuada para mantener la calidad de la señal de voz a través del enlace radioeléctrico. Este control de potencia es obligatorio para las MS mientras que no lo es para las BS. La MS deber ser capaz de variar su potencia de transmisión desde su valor máximo hasta 20 mW en pasos de 2dB.

Para el acceso inicial de una MS en una celda del RACH, dicha MS debe usar o su valor máximo definido por la clase de MS que es o el valor máximo permitido en esa celda si este es menor. Tras esto, la BS calcula el nivel de potencia en radiofrecuencia que debe usar la MS y se lo señala mediante 4 bits que para tal efecto hay dedicados en el SACCH. El cambio de potencia en la MS se realiza a una velocidad de 2 dB cada 60 ms y la MS confirma a la BS el nivel de potencia que utiliza en el SACCH.

8.3 HANDOVER

La MS tiene establecido el proceso de comunicación con la BS que le proporciona mejor enlace. Como la MS se mueve, la BS con la que existe el mejor enlace puede variar, por lo que la MS debe ser reasignada a una nueva BS y su llamada re-enrutada

adecuadamente. Esta necesidad es solucionada mediante el proceso de "handover" que determina la asignación de la MS o de la BS y que por tanto determina el tamaño de las celdas mediante los valores de umbrales de decisión utilizados y determina la calidad del enlace radioeléctrico.
Para controlar el proceso de "handover", el sistema posee información de la calidad del enlace radioeléctrico existente y del de los enlaces alternativos con las BSS vecinas. Para esto, aunque las MS sólo están activas en 2 de los 8 intervalos de tiempo de una trama, tienen la posibilidad de explorar, en los 6 restantes, las transmisiones del BCCH de las BS vecinas. Una vez que la MS tiene información de la calidad de su enlace con la BS utilizada y con las vecinas, transmite a la red la información de las 6 BS con mayor intensidad de señal recibida, donde es tomada la decisión de handover.

Por último, respecto de la BS con la que está enlazada, la MS mide no sólo la intensidad de señal recibida, sino también la calidad de la misma en tasa de error de canal. El medir los dos parámetros permite al sistema conocer si la degradación de un enlace radioeléctrico se debe a falta de señal o a interferencia cocanal. Cuando ocurre esto último, existe el procedimiento de handover intracelular que consiste en cambiar el canal en el que se realiza la comunicación dentro de una misma BS.

8.4 TRANSMISIÓN DISCONTINUA

Como la mayoría de las comunicaciones en sistemas móviles son de voz y éstas son realmente activas menos de la mitad del tiempo, GSM usa la transmisión discontinua (DTX) apoyándose en detectores de actividad vocal (VAD), transmitiendo aquellos intervalos de voz considerados activos. Esto tiene dos ventajas: la señal cocanal de interferencia se reduce a 3dB y la duración de la batería de la MS aumenta considerablemente.

Los intervalos en los que no se transmite voz se rellenan mediante ruido leve. El algoritmo para extraer ese ruido es enviado periódicamente al extremo receptor de la comunicación en los periodos de silencio. El procedimiento DTX es obligatorio para las MS y opcional para las BS.

8.5 SALTO LENTO EN FRECUENCIA

La propagación en sistemas móviles en los que se da multitrayecto produce un desvanecimiento muy característico de la señal. Este efecto negativo se corrige en parte, mediante el salto lento en frecuencia (SFH). La secuencia de "bursts" que forman el TCH es asignada cíclicamente a diferentes frecuencias definidas para una BS. Las señales de temporización disponibles en la BS y en la MS son usadas para mantener a los transmisores y receptores en sincronismo dentro de una secuencia de salto en frecuencia definida.

Una ventaja adicional del SFH es que la interferencia cocanal está más dispersa entre todas las MS, ya que ellas pasarían por la frecuencia en la que existe la interferencia siendo ligeramente afectadas pero ninguna de forma continua. En el sistema GSM la duración de cada salto coincide con la de la trama TDMA (4,616 ms), provocando una frecuencia de salto es de 217 saltos/s.

9 CRIPTOSISTEMAS

Un criptosistema es un conjunto de elementos y funciones dispuestos de tal forma que, con base en un conjunto de reglas sintácticas y semánticas, permite emitir a través de un canal de comunicación un mensaje en modo cifrado y obtener posteriormente el mensaje claro correspondiente en el lado del receptor. Para ese efecto, los criptosistemas se valen de una o varias claves de cifrado, unos procesos de transformaciones de cifrado y los correspondientes procesos de transformaciones de descifrado.

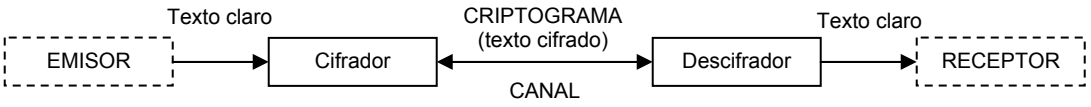

Figura 6. Funcionamiento de un criptosistema.

El elemento más importante de todo el criptosistema es el cifrador, el cual utiliza un algoritmo de cifrado para convertir el texto claro en un criptograma. Usualmente, para hacer esto, el cifrador depende de un parámetro exterior llamado **clave de cifrado** que es aplicado a una función matemática que se asume como irreversible en principio, es decir, no es posible invertir la función a no ser que se disponga de la clave de descifrado (en algunos sistemas la clave de cifrado y de descifrado son las mismas). De esta forma, cualquier conocedor de la clave (y por supuesto de la función), será capaz de descifrar el criptograma y nadie que no conozca dicha clave puede ser capaz de descifrarlo, aún en el caso en que conozca la función utilizada.

9.1 CLASIFICACIÓN DE LOS CRIPTOSISTEMAS

Los criptosistemas, de acuerdo con la disponibilidad de la clave de cifrado / descifrado, se clasifican en criptosistemas de clave pública y criptosistemas de clave secreta.

9.1.1 Criptosistemas de clave pública

En los criptosistemas de clave pública la clave de cifrado se hace de conocimiento general; sin embargo, no ocurre lo mismo con la clave de descifrado (clave privada), que se mantiene en secreto. Ambas claves son dependientes, pero del conocimiento de la pública no es posible deducir la privada sin ningún otro dato. La existencia de ambas claves diferentes, para cifrar y descifrar, hace que también se conozca a estos criptosistemas como asimétricos.

Cuando un receptor desea recibir una información cifrada, debe hacer llegar a todos los potenciales emisores su clave pública, para que éstos cifren los mensajes con

dicha clave. De este modo, el único que podrá descifrar el mensaje será el legítimo receptor mediante su clave privada.

9.1.2 Criptosistemas de clave secreta

Los criptosistemas de clave secreta (también conocidos como criptosistemas simétricos) son aquéllos en los que la clave de cifrado puede ser calculada a partir de la de descifrado y viceversa. En la mayoría de estos sistemas ambas claves coinciden, y por tanto deben mantenerse como un secreto entre emisor y receptor. Si un atacante descubre la clave utilizada en la comunicación, ha roto el criptosistema.

Hasta hace algún tiempo, la invulnerabilidad de este tipo de sistemas dependía de este mantenimiento en secreto de la clave de cifrado. Este hecho presentaba una gran desventaja, ya que había que enviar, aparte del criptograma, la clave de cifrado del emisor al receptor para que éste fuera capaz de descifrar el mensaje.

Los criptosistemas de clave secreta se dividen en dos grandes grupos: los **cifradores de flujo**, que son aquéllos que pueden cifrar un sólo bit de texto claro al mismo tiempo y por tanto su cifrado se produce bit a bit, y los cifradores de bloque, que cifran un bloque de bits como una única unidad. Por la naturaleza de la investigación, se expondrá a continuación el primer grupo.

9.1.2.1 Cifradores de flujo

Los cifradores de flujo se basan en considerar todo el mensaje como un flujo de caracteres o bits y cifrar cada elemento de ese mensaje con un elemento equivalente de una secuencia cifrante. Si la longitud de la secuencia cifrante es menor que el mensaje, el cifrador será periódico. Según el tipo de cifrador, se obtienen distintos periodos para la secuencia cifrante.

La representación básica de un cifrador de flujo es la siguiente:

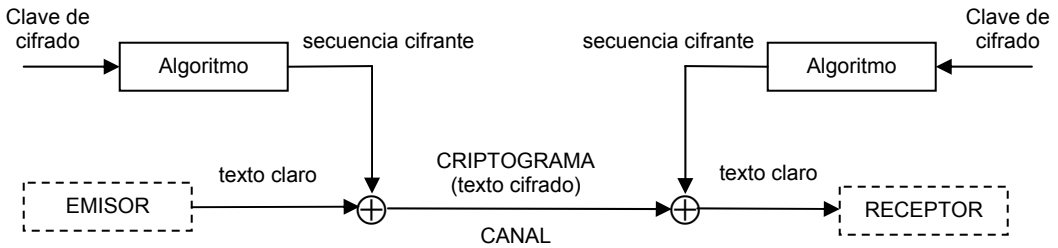

Figura 7. Representación de un cifrador de flujo.

El algoritmo empleado para la obtención de la secuencia cifrante es determinístico y por tanto ésta no es completamente aleatoria (es pseudoaleatoria). Esto resta seguridad al cifrador de flujo, pero permite su implementación práctica. La clave de cifrado usada en ambos lados del cifrador es la misma (clave secreta) la cual se procura que llegue de un lado a otro de manera confidencial.

Debido a que la secuencia cifrante es pseudoaleatoria, el cifrador ve disminuida su seguridad. No obstante, existen algunas características básicas que debe tener la secuencia generada para aproximarse a una secuencia completamente aleatoria y permitir así el empleo de estos cifradores de flujo en los sistemas de telecomunicaciones actuales.

Período: el período de la secuencia cifrante sería ideal si fuese tan largo como el texto en claro que vamos a cifrar. Como esto no puede darse en muchos casos, se usan de menor longitud, lo que provoca que el cifrador sea periódico. Para la práctica se buscan periodos lo más largos posibles.

Postulados de Golomb:

- En cada periodo de la secuencia binaria no debe haber grandes diferencias entre el número de "unos" y el número de "ceros".
- En cada período de la secuencia binaria, la mitad de las subsecuencias de "unos" debe tener longitud 1, una cuarta parte longitud 2, una octava parte longitud 3, etc. En general, hay $(½)^i$ *s*ubsecuencias de longitud "i". Lo mismo ocurre con las subsecuencias de "ceros". Pero además, debe haber el mismo número total de subsecuencias de "unos" y de "ceros".
- La autocorrelación debe ser constante para cualquier número de desplazamientos (retardos) hacia la izquierda o hacia la derecha de la secuencia binaria generada. Entre más pequeño sea ese valor constante de autocorrelación, estamos más cerca de la secuencia pseudoaleatoria ideal. Por supuesto, si el número de desplazamientos es múltiplo del periodo de la secuencia, la función de autocorrelación valdrá 1 (todos los bits tienen correlación máxima consigo mismos). Dicho de otra forma, la autocorrelación compara el valor del pico máximo (tras el desplazamiento de la secuencia) con el valor en el origen.

  Matemáticamente se establece: la función de autocorrelación normalizada (reemplazar "0s" por "-1s" y dejar los "1s" igual) de dicha secuencia binaria debe ser bivalor en un período. El bivalor se expresa como:

$$B(h) = \frac{1}{P}\sum_{n=1}^{P}[xn \ . \ x(n+h)] = \begin{cases} 1 \text{ si } h = 0 \\ k \text{ si } 0 < h < P \end{cases}$$

donde "P" es el periodo de la secuencia, "n" es la posición del elemento en la secuencia y "x" es el valor de dicho elemento (1 ó -1).

Imprevisibilidad: la secuencia cifrante debe ser totalmente imprevisible. Esto significa que si se tiene un conjunto de elementos generados por el cifrador, la probabilidad de encontrar el siguiente elemento debe ser inferior al 50%.

Test espectral: al aplicar una transformada de Fourier sobre la secuencia generada por el algoritmo se debe obtener un pico máximo en el origen y un valor muy pequeño en el resto de las componentes frecuenciales. En el caso normalizado, un valor de 1 en el origen y un valor muy pequeño en el resto, lo que se aproxima al espectro de un ruido blanco. Esto como consecuencia de que los aportes de los múltiples armónicos para la reconstrucción de la secuencia pseudoaleatoria original deben ser muchos en cantidad pero insignificantes en amplitud.

Dentro de los generadores de secuencias cifrantes existen tres categorías bien diferenciadas: generadores de concurrencia lineal, registros de desplazamiento realimentados y combinadores no lineales. Por la naturaleza de la investigación, se expondrá a continuación la segunda categoría.

9.2 REGISTROS DE DESPLAZAMIENTO REALIMENTADOS LINEALMENTE (LFSR)

Los registros de desplazamiento realimentados linealmente es uno de los sistemas más importantes para la generación de secuencias pseudoaleatorias. Para mantener una secuencia periódica de salida se tiene que tener una referencia del bit que se pierde. Matemáticamente este sistema utiliza una función de realimentación de la siguiente forma:

$$g(x) = 1 + c_1x + c_2x^2 + \ldots + c_{n-1}x^{n-1} + c_nx^n$$

Los coeficientes c_i valdrán 1 si la etapa está conectada a la puerta XOR y cero en caso contrario (la última etapa tiene que estar conectada a la puerta XOR para mantener la periodicidad).

La clave serán los valores iniciales que asignemos a los registros. Hay que tener en cuenta que el valor cero en todos los registros no es válido ya que como se utiliza una compuerta XOR, el mensaje saliente sería igual al texto claro (no habría cifrado). Debido a esto, el período de un sistema de este tipo para n etapas es 2^n-1.

Dentro de los registros de desplazamiento realimentados linealmente se encuentran aquéllos con polinomio de realimentación primitivo, cuya secuencia generada presenta las siguientes características:

- La longitud de la secuencia no depende del estado inicial.
- El período es 2^n –1.

Un polinomio de este tipo es $g(x) = 1 + x + x^4$ que representado gráficamente sería:

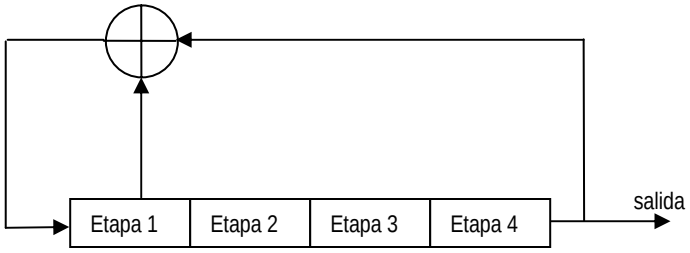

Figura 8. LFSR con polinomio de realimentación primitivo.

Este tipo de generadores con polinomio primitivo son los que resultan más interesantes en los sistemas de cifrado, ya que son los que ofrecen una secuencia de período máxima. Sin embargo, los LFSR con polinomio primitivo tienen algunos posibles puntos débiles. Si se tiene una secuencia de 2^n dígitos consecutivos, es decir, se sabe por todos los estados que pasa el generador, es muy posible calcular la clave inicial del sistema junto con los coeficientes c_i. Para evitar estas debilidades se pueden combinar uno o varios generadores LFSR de este tipo mediante una función no lineal, con lo que se consigue que el conjunto sea bastante confiable. De hecho, éste es el principio que aplica el generador de secuencia A5/2.

9.3 GENERADORES DE SECUENCIA CIFRANTE BASADOS EN LA COMBINACIÓN NO LINEAL DE VARIOS REGISTROS DE DESPLAZAMIENTO

Una configuración común de este tipo de generadores consiste en utilizar uno de los LFSR para controlar al resto. Este LFSR podría servir como temporizador para controlar el desplazamiento de los restantes o bien como selector para escoger a cualquiera de los otros como salida del sistema.

Para este tipo de configuración, si los polinomios de realimentación de los LFSR son primitivos con grados n1, n2, ... , ni, el periodo de la secuencia cifrante que se genera es P = mcm (2^{n1}-1, 2^{n2}-1, ... , 2^{ni}-1).

10 SEGURIDAD EN GSM-1900

A continuación se abordarán los algoritmos de autentificación (A3), de cifrado (A5) y de generación de clave de cifrado (A8), haciéndose énfasis sobre el segundo. Específicamente se expondrá la versión A5/2 de dicho algoritmo o versión débil, la cual se emplea fundamentalmente fuera del continente europeo.

10.1 PROCESO DE AUTENTIFICACIÓN (ALGORITMO A3)

Este proceso consta de dos fases:

1ª. Un código PIN (Personal Identification Number) que protege a la tarjeta SIM (Subscriber Identity Module). El PIN es chequeado por la SIM localmente (no va al enlace radio).

2ª. La red GSM envía un número aleatorio RAND de 128 bits el cual es mezclado con un parámetro secreto denominado Ki mediante un algoritmo conocido como A3. Esto produce un resultado de 32 bits denominado SRES (Signed Response) que se devuelve a la red GSM para su verificación.

En síntesis, el MSC envía simultáneamente a la MS y al AuC la secuencia RAND. En ambos lados se calcula la secuencia SRES y si son idénticas, el proceso de autentificación ha tenido éxito.

El procedimiento anterior se puede representar gráficamente de la siguiente forma:

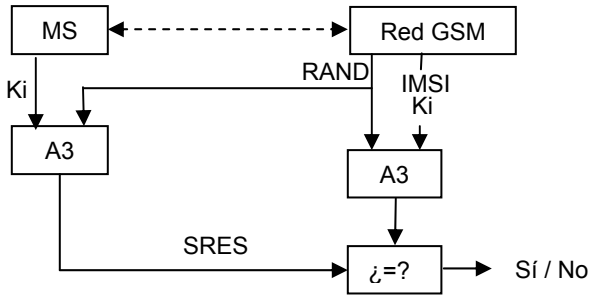

Figura 9. Proceso de autentificación (Algoritmo A3).

La clave de autentificación del suscriptor Ki es asignada junto con el IMSI (International Mobile Subscriber Identity) al momento de la suscripción. La actualización de los vectores RAND y SRES la realiza el HLR y el AuC quienes envían el dato al BSS/MSC/VLR.

El algoritmo A3 es considerado un asunto de los operadores GSM PLMN (Public Land Mobile Network). Este algoritmo permite la autentificación de la identidad de un suscriptor móvil. El algoritmo A3 está contenido en la tarjeta SIM (del lado de la MS) y está implementado en el AuC (del lado de la red).

10.1.1 PARÁMETROS DEL ALGORITMO A3

- De entrada: - RAND = 128 bits
  - Ki = 128 bits
- De salida: - SRES = 32 bits

De acuerdo con las prestaciones de la red GSM, el algoritmo A3 debe ejecutarse en menos de 500 ms.

10.1.2 MÉTODO DE IDENTIFICACIÓN

Para identificar al suscriptor móvil se usa un TMSI (Temporary Mobile Subscriber Identity) el cual tiene significado en un área de localización dada y por tanto debe estar acompañado de un LAI (Location Area Identification) para evitar ambigüedades.

Las bases de datos (como el VLR) mantienen la relación entre TMSI e IMSI. Cuando se recibe un TMSI con un LAI que no corresponde con el VLR actual, el IMSI de la MS se debe solicitar al VLR encargado del área de localización indicada si se conoce su dirección, de otra manera, se debe solicitar a la propia MS.

Se debe asignar un nuevo TMSI al menos por cada procedimiento de actualización de posición. Cuando se asigna un nuevo TMSI a una MS, éste debe ser transmitido en modo cifrado. La MS debe almacenar su TMSI actual en una memoria no volátil junto con el LAI, de tal modo que estos datos no se pierdan cuando la MS se apague.

10.2 CIFRADO DE LA VOZ (ALGORITMO A5)

Para el proceso de cifrado de la voz, lo primero es que la MS y la red fijen una clave de cifrado (Kc) mutua. Para esto se ejecuta el siguiente procedimiento:

- Se inicia con el procedimiento de autentificación y se puede cambiar tantas veces según criterio del operador de red.
- Ocurre en un canal de control dedicado aún no encriptado una vez se identifica la MS en la red (con el TMSI o IMSI).

- Es un cálculo que realiza la MS localmente con el uso del # RAND, la clave Ki (propios del proceso de autentificación) y el algoritmo A8.
- Kc se calcula junto con la SRES.
- Kc es almacenada en la estación móvil hasta su actualización en la próxima autentificación.

La representación gráfica del procedimiento anterior se muestra a continuación:

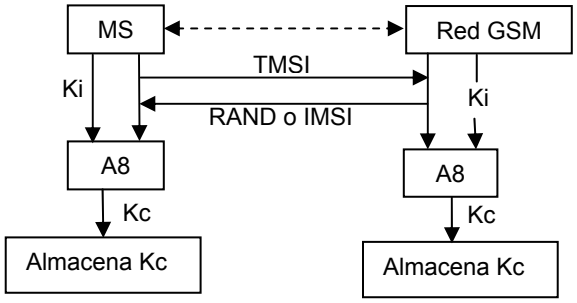

Figura 10. Establecimiento de la clave Kc (para el cifrado de la voz).

El algoritmo A5 realiza la protección de los datos de usuario y de la información de señalización en la capa física sobre los canales dedicados (TCH y DCCH). El algoritmo A5 está implementado tanto en la MS como en el BSS. En la descripción del lado del BSS se asume que un algoritmo A5 está implementado por cada canal físico. El cifrado ocurre antes de la modulación y después del entrelazado, el descifrado ocurre después de la demodulación simétricamente. Tanto el cifrado como el descifrado necesitan el algoritmo A5.

10.2.1 INICIO DE LOS PROCESOS DE CIFRADO Y DESCIFRADO (en el DCCH y TCH)

Estos procesos tienen lugar en un DCCH bajo el control de la red un tiempo después de haberse completado el proceso de autentificación (si lo hay) o después que la clave de cifrado Kc esté disponible en el BSS. No se puede enviar información que necesite protección antes del arranque de los procesos de cifrado y descifrado. El procedimiento es el siguiente:

1. El BSS inicia el proceso de descifrado.
2. El BSS envía un mensaje en texto en claro a la MS para que arranque el proceso de cifrado (“comience el cifrado”).
3. La MS arranca con los procesos de cifrado y descifrado después que ha recibido correctamente el mensaje “comience el cifrado”.
4. Finalmente, el BSS arranca con el cifrado una vez descifra correctamente una trama o mensaje procedente de la MS.

Cuando un TCH es asignado para la transmisión de datos de usuario, la clave usada es aquélla establecida durante la sesión DCCH previa. Los procesos de cifrado y descifrado arrancan inmediatamente.

10.2.2 BOSQUEJO DEL PROCESO CIFRADO / DESCIFRADO

Debido a la técnica TDMA que usa el sistema, los datos útiles (también llamados texto claro) se organizan en bloques de 114 bits. Luego, cada bloque es incorporado dentro de un “burst” normal y transmitido durante un “time slot”. Los bits de información útil dentro de un bloque están numerados e0 hasta e56 y e59 hasta e115 (se ignoran los bits de bandera e57 y e58). “Slots” sucesivos para un canal físico dado están separados al menos por la duración de una trama, aproximadamente 4,615ms.

Para el cifrado, el algoritmo A5 produce cada 4,615 ms una secuencia de 114 bits cifrado / descifrado (llamada bloque) la cual se combina mediante una suma en módulo 2 con los 114 bits del bloque de texto claro. El primer bit cifrado / descifrado producido por A5 se suma con e0, el segundo con e1 y así sucesivamente. El bloque resultante de 114 bits se aplica luego al reconstructor de "bursts".

Para cada “slot”, el descifrado es logrado en el lado de la MS con el bloque 1 de 114 bits producido por A5 y el cifrado se logra con el bloque 2. Como consecuencia, en el lado de la red, el bloque 1 se usa para cifrar y el bloque 2 para descifrar. De ahí que el algoritmo A5 debe producir dos bloques de 114 bits cada 4,615 ms.

La sincronización está garantizada mediante el manejo del algoritmo A5 por una variable de tiempo explícita COUNT, derivada del número de trama TDMA. De ahí que cada bloque de 114 bits producido por A5 depende sólo de la numeración de la trama TDMA y de la clave de cifrado Kc. COUNT está expresado en 22 bits.

La representación de un procedimiento cifrado / descifrado es como se muestra a continuación (el proceso inverso es simétrico):

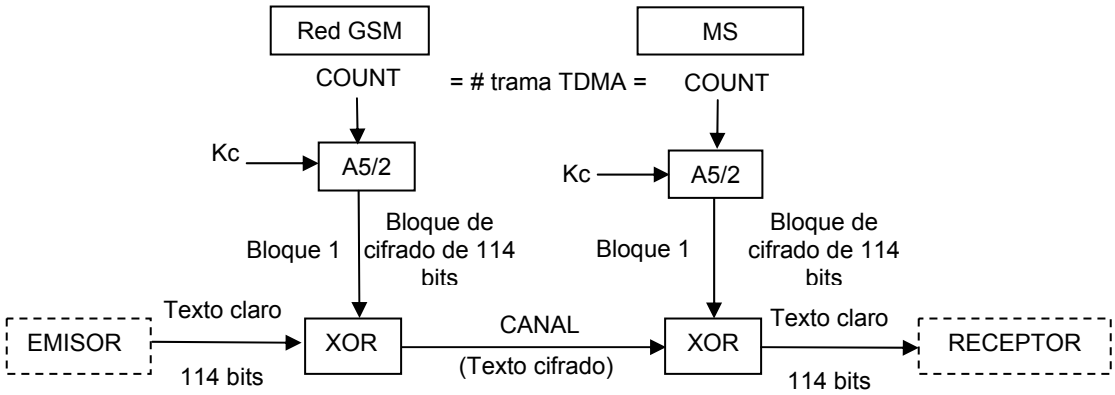

Figura 11. Proceso de cifrado / descifrado mediante el algoritmo A5/2.

10.2.3 PARÁMETROS DEL ALGORITMO A5

- De entrada: - COUNT = 22 bits
  - Kc = 64 bits

- De salida: - Bloque 1 = 114 bits
  - Bloque 2 = 114 bits

NOTA: Si la longitud de Kc es menor a 64 bits, se asume que Kc real corresponde a los bits más significativos de sí misma y que los bits menos significativos son llevados a cero. Para propósitos de señalización y prueba, Kc es considerada como 64 bits no estructurados.

10.2.4 NEGOCIACIÓN DEL ALGORITMO A5

La red debe comparar las capacidades y preferencias de cifrado y cualquier requerimiento especial de la suscripción de la MS con los indicados por ella y actuar de acuerdo con las siguientes reglas:

1) Si la MS y la red no tienen versiones en común del algoritmo A5 y la red no está preparada para usar una conexión sin cifrado, entonces la conexión será liberada.
2) Si la MS y la red tienen al menos una versión del algoritmo A5 en común, entonces la red seleccionará una de dichas versiones para la conexión.
3) Si la MS y la red no tiene versiones en común del algoritmo A5 y la red desea establecer una conexión sin cifrado, entonces se usará una conexión sin cifrado.

10.3 GENERACIÓN DE CLAVE DE CIFRADO (ALGORITMO A8)

Para la obtención de la clave de cifrado Kc se utiliza el algoritmo A8 (según se anotó).

10.3.1 PARÁMETROS DEL ALGORITMO A8

- De entrada: - RAND = 128 bits
  - Ki = 128 bits

- De salida: - Kc = 64 bits

Debido a que la máxima longitud de la clave de cifrado Kc es establecida por el grupo GSM/MoU, el algoritmo A8 debe producir esa longitud para Kc y extenderla si fuera necesario a los 64 bits, donde los bits menos significativos son forzados a cero. Se asume que Kc está contenida en los bits más significativos.

10.4 ENTIDADES DEL SISTEMA GSM DONDE SE ALMACENA INFORMACIÓN DE SEGURIDAD

- HLR: almacena (si se requiere) conjuntos de Kc, RAND y SRES asociados con cada IMSI.
- VLR: almacena conjuntos de Kc, RAND y SRES asociados con cada IMSI. Además, el CKSN, LAI y TMSI son almacenados con la Kc que se asume como válida. Cuando se genera un nuevo TMSI, tanto el viejo como el nuevo TMSI son almacenados. Cuando el viejo TMSI deja de ser válido, se remueve de la base de datos.
- MSC: el algoritmo de encriptación se almacena en el MSC/BSS. Relacionado con una llamada, el MSC almacena la clave de cifrado Kc y el CKSN asociado con la identidad del móvil involucrado en esa llamada.
- MS: la estación móvil almacena permanentemente:
  - Algoritmo de autentificación A3.
  - Algoritmo de encriptación A5.
  - Algoritmo de generación de clave de cifrado A8.
  - Clave de autentificación del suscriptor individual Ki.
  - Clave de cifrado Kc.
  - Número de secuencia de clave de cifrado.
  - TMSI

  La estación móvil genera y almacena:
  - Clave de cifrado Kc.

  La estación móvil recibe y almacena:
  - Número de secuencia de clave de cifrado.
  - TMSI
  - LAI
- AuC: aquí se implementa:
  - Algoritmo(s) de autentificación A3.
  - Algoritmo(s) de generación de clave de cifrado A8.

  Las claves de autentificación secretas individuales Ki de cada suscriptor se almacenan en un centro de autentificación.

10.5 ESPECIFICACIONES DE LOS ALGORITMOS RELACIONADOS CON LA SEGURIDAD

- A3: Algoritmo de autentificación
- A5: Algoritmo de cifrado / descifrado
- A8: Algoritmo generador de clave de cifrado

El algoritmo A5 debe ser común a todas las PLMN GSM y a todas las estaciones móviles (en particular, para permitir el roaming).

Los algoritmos A3 y A8 están a discreción del operador de cada PLMN. Sólo los formatos de sus entradas y salidas deben ser especificados. También es deseable que los tiempos de procesamiento de estos algoritmos permanezcan por debajo de un valor máximo.

10.6 DESCRIPCIÓN DEL ALGORITMO A5/2 (O GENERADOR BINARIO DE SECUENCIA CIFRANTE A5/2)

Una conversación GSM es enviada como una secuencia de tramas cada 4,6 ms. Cada trama contiene 114 bits que representan la comunicación digitalizada A-B y 114 bits que representan la comunicación digitalizada B-A. Cada conversación puede cifrarse por una nueva clave de cifrado Kc. Para cada trama, Kc se mezcla con un contador de trama (o número de trama) públicamente conocido y el resultado sirve como el estado inicial del generador que produce 228 bits pseudoaleatorios. Estos bits se combinan en operación XOR con los 114+114 bits de texto claro para producir 114+114 bits de texto cifrado. Cada 4,6 ms se envía una nueva trama inicializada con la misma Kc pero con diferente número de trama.

El algoritmo A5/2 está conformado de la siguiente manera:

- Cuatro registros de desplazamiento realimentados linealmente (LFSR) denotados por R1, R2, R3 y R4.
  - R1 tiene una longitud de 19 etapas numeradas de izquierda a derecha y polinomio de realimentación primitivo $g_1(x) = 1 + x^{14} + x^{17} + x^{18} + x^{19}$. La salida de este registro es el contenido binario de la etapa 19. Los contenidos de las etapas 13, 15 (complementada) y 16 son las entradas a una copia de la función mayoría F asociada a R1.
  - R2 tiene una longitud de 22 etapas numeradas de izquierda a derecha y polinomio de realimentación primitivo $g_2(x) = 1 + x^{21} + x^{22}$. La salida del registro es el contenido binario de la etapa 22. Los contenidos de las etapas 10, 14, y 17 (complementada) son las entradas a una copia de la función mayoría F asociada a R2.
  - R3 tiene una longitud de 23 etapas numeradas de izquierda a derecha y polinomio de realimentación primitivo $g_3(x) = 1 + x^{8} + x^{21} + x^{22} + x^{23}$. La salida del registro se toma como el contenido binario de la etapa 23. Los contenidos de las etapas 14 (complementada), 17 y 19 son las entradas a una copia de la función mayoría F asociada a R3.
  - R4 tiene una longitud de 17 etapas numeradas de izquierda a derecha y polinomio de realimentación primitivo $g_4(x) = 1 + x^{12} + x^{17}$. Los contenidos de las etapas 4, 7 y 11 son las entradas a una copia de la función mayoría F asociada a R4. La salida de esta copia de la función mayoría determina cuáles registros (R1, R2 o R3) van a desplazarse durante el ciclo actual de operación.

- Cuatro copias de la función mayoría F asociadas a los cuatro registros, cuya forma algebraica es:

$$F = X1X2 \oplus X1X3 \oplus X2X3$$

- Tres puertas XOR de dos entradas que suman la salida de la función mayoría del registro R4 con los bits 11, 4 y 8 respectivamente del mismo registro. Posteriormente, las salidas de estas puertas denotadas por S1, S2 y S3 aparecen complementadas. Estas salidas complementadas activan el desplazamiento del correspondiente registro.

- Una puerta XOR de seis entradas que suma los bits de salida de cada registro más los bits de salida de las funciones mayoría asociadas a R1, R2 y R3, dando lugar a S4 o bit de salida del generador.

La figura 12 presenta la estructura interna del algoritmo A5/2.

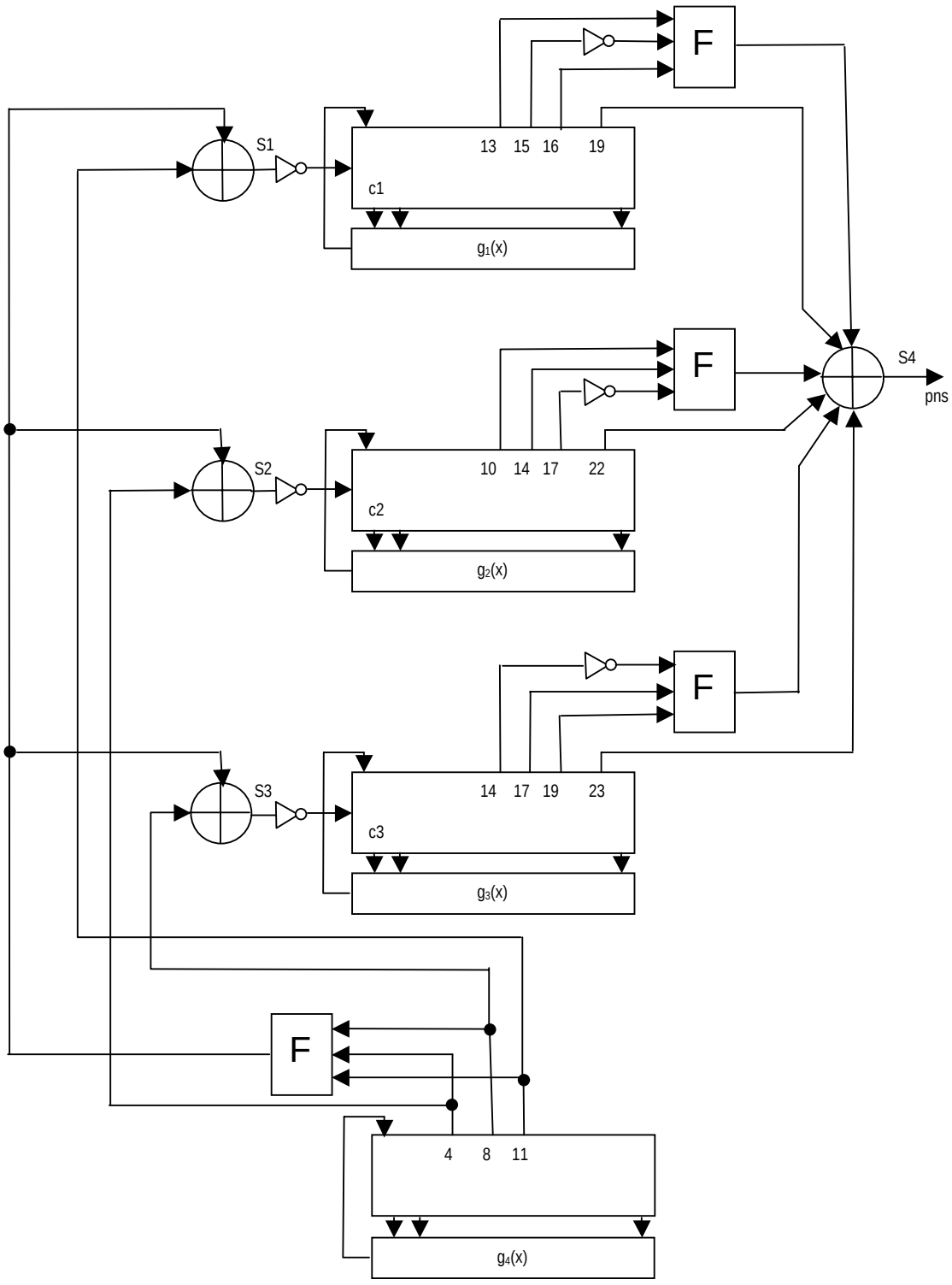

Figura 12. Estructura interna del algoritmo A5/2.

10.6.1 PROCESO DE CIFRADO

El cifrado de cada trama de conversación con el algoritmo A5/2 incluye un proceso de inicialización que va a determinar el estado inicial de cada uno de los registros. En la determinación de dichos estados intervienen la clave de cifrado Kc (64 bits) y el número de trama f (22 bits). El proceso de inicialización se lleva a cabo de la siguiente manera:

1. Se inicializan a cero los cuatro registros R1, R2, R3 y R4.

2. Desplazando regularmente 64 veces los cuatro registros se cargan los 64 bits de la clave Kc dentro de cada registro, sumando mediante una operación XOR, el correspondiente bit de clave con el bit de realimentación de cada uno de los registros. El resultado de esta suma será el nuevo bit de realimentación del correspondiente Ri. Los 8 bytes de clave se cargan empezando por el bit menos significativo.

3. Desplazando regularmente 22 veces los cuatro registros, se cargan ahora los 22 bits del número de trama f dentro de cada registro, sumando mediante una operación XOR, el correspondiente bit del número trama con el bit de realimentación de cada uno de los registros. El resultado de esta suma será el nuevo bit de realimentación del correspondiente Ri. Los 22 bits del número de trama se cargan empezando por el bit menos significativo. Una vez cargados los 22 bits, se colocan a 1 los siguientes bits: bit 16 del registro R1, bit 17 del registro R2 y bit 19 del registro R3.

4. Se llevan a cabo 100 ciclos de operación del algoritmo A5/2 que no dan lugar a bits de salida pero sí modifican el contenido de los cuatro registros. La salida aparece retrasada un ciclo de operación.

Una vez terminado el proceso de inicialización, cada ciclo del algoritmo A5/2 produce un bit de salida de acuerdo con el siguiente procedimiento:

1. Se calcula el valor de la función mayoría **F** asociada con R4 tomando como entradas los bits 4, 8 y 11 del estado actual del mismo registro.

2. Se calcula el valor de S1, S2, S3 y sus correspondientes valores complementados a partir de los bits 11, 4 y 8 respectivamente del estado actual del registro R4 y de la salida de la función mayoría asociada con R4. Si algún $\overline{Si}$ es igual a 1 el correspondiente registro se desplaza, en caso contrario no se desplaza (*"stop/go clocking"*).

3. Se calcula el valor del bit de salida S4 como la suma XOR entre el valor del bit 19 de R1, el bit 22 de R2 y el bit 23 de R3 y las salidas de las copias de la función mayoría **F** asociadas a R1, R2 y R3.

4. Se retrasa el bit de salida en un ciclo de operación del algoritmo, es decir, el bit de salida actual es el bit calculado en el ciclo anterior.

5. Se desplaza el registro R4 para modificar el "stop/go clocking" de los otros tres registros.

Con este procedimiento se generan los primeros 114 bits de la trama (móvil 1-móvil 2). Posteriormente, con el mismo procedimiento, se generan los restantes 114 bits (móvil 2-móvil 1) para completar así los elementos necesarios para el cifrado de una trama.

10.7 ATAQUES CRIPTOANALÍTICOS AL ALGORITMO A5/2

En la literatura abierta se encuentran actualmente disponibles algunos ataques criptoanalíticos al algoritmo A5/2, los cuales están basados fundamentalmente en las debilidades que éste presenta. Una vez el algoritmo A5/2 fue construido mediante ingeniería inversa, sufrió su primer ataque criptoanalítico (Goldberg et al.). Éste es un ataque en el que se conoce el texto claro y que requiere la diferencia en dicho texto de dos tramas GSM que están separadas exactamente 2^{11} tramas, es decir, aproximadamente 6 segundos de conversación. La complejidad en el tiempo promedio de este ataque es aproximadamente 2^{16} productos punto de vectores de 114 bits.

El siguiente ataque realizado al algoritmo A5/2 (Fúster A. et al.) sugiere considerar el estado interno inicial del cifrador (los registros de desplazamiento) como variables, escribir cada bit de salida del algoritmo como una función cuadrática de estas variables y linealizar los términos cuadráticos. En este ataque se demostró que la salida del A5/2 se puede predecir con una probabilidad muy alta después de conocer algunos cientos de bits de salida. Sin embargo, este ataque no descubre la clave inicial del algoritmo, sino que basa su eficacia en la explotación de los puntos débiles inherentes a este tipo de generadores tales como las frecuentes reinicializaciones, el bajo número de bits desechados antes de la generación de la secuencia de salida y la mala distribución de las realimentaciones de los LFSR, aspectos que hacen posible descubrir las relaciones lineales existentes entre los bits producidos por este generador.

El tercer ataque conocido (Barkan et al.) es un ataque puramente algebraico sobre el texto claro conocido del A5/2 que determina la clave inicial. Este ataque aprovecha el bajo orden algebraico de la función de salida del A5/2, la cual se representa como una función multivariable cuadrática del estado inicial de los registros. Posteriormente se plantea un sistema sobre-definido de ecuaciones cuadráticas que expresa el proceso de generación de la clave y por último se resuelven las ecuaciones. En resumen, con el conocimiento del estado interno inicial de los cuatro registros y del número de trama inicial, es posible recuperar la clave de cifrado mediante operaciones algebraicas sencillas. Esto es posible principalmente porque el proceso de inicialización es lineal en dicha clave y en el número de trama. Por tanto, el ataque se enfoca en revelar el estado interno inicial de los registros.

El primer ataque mencionado necesita el conocimiento de al menos 6 segundos de conversación para poder ser efectivo. Sumado a esto está la dificultad de necesitarse la operación XOR de dos tramas separadas exactamente ese tiempo, lo cual hace difícil una implementación práctica. Aunque el segundo y el tercer ataque basan el criptoanálisis del algoritmo en la determinación de parámetros completamente diferentes, ambos están fundamentados en las frecuentes reinicializaciones del algoritmo como principal desventaja. Este aspecto, de mayor relevancia en el segundo de los ataques, abre camino para el planteamiento de un nuevo algoritmo que evite ser vulnerado de la misma forma que lo ha sido el existente actualmente en el sistema PCS. Para dar claridad a la forma práctica como puede ser superado el algoritmo A5/2, se expone brevemente a continuación el ataque realizado por Amparo Fúster y Slobodan Petrovic en su artículo "Criptoanálisis del algoritmo A5/2 para telefonía móvil" [1].

10.7.1 ATAQUE FÚSTER-PETROVIC

Este ataque utiliza 4 tramas (no necesariamente separadas una distancia fija) de la secuencia de salida del generador para la reconstrucción de los restantes bits de la misma. El procedimiento está basado en una parametrización de subclaves en donde el registro R4 juega el papel de subclave. Para cada uno de los posibles estados de este registro, se inicializa el algoritmo de generación de la secuencia, planteando el correspondiente sistema de ecuaciones no lineales (que después se linealiza sustituyendo los productos de variables simples por nuevas variables) cuyas incógnitas son las componentes de los estados iniciales de los restantes registros. A cada nuevo pulso de reloj se plantea una nueva ecuación que se añade al sistema considerado. Sin embargo, la probabilidad de obtener un número de ecuaciones linealmente independientes igual al número de incógnitas del sistema es muy pequeña. De ahí que, en lugar de resolver el sistema de ecuaciones planteado, se puedan encontrar relaciones lineales entre los bits desconocidos de la secuencia de salida que se quiere reconstruir y el conjunto de bits interceptados (por tanto conocidos) de dicha secuencia.

Para la reconstrucción de las relaciones lineales entre los bits de la secuencia de salida se hace uso de diversos puntos débiles inherentes al algoritmo A5/2 como la reinicialización del algoritmo a cada nueva trama, el bajo número de bits desechados tras la reinicialización y antes de producir los 228 bits de salida y una mala distribución de las realimentaciones de los LFSR.

El algoritmo de ataque planteado por Fúster-Petrovic está dentro del siguiente contexto: para cada uno de los posibles estados iniciales del registro R4 y tras la introducción de la clave secreta, se plantea el sistema de ecuaciones no lineales en términos de sus incógnitas x_1, x_2, ..., x_{64}, donde x_1, ..., x_{19} son las variables que corresponden al estado del registro R1, x_{20}, ..., x_{41} son las variables que corresponden al estado del registro R2 y x_{42}, ..., x_{64} son las variables que corresponden al estado del

registro R3. Sea $[F_i]_{i = 1,...,4}$ el vector de números de trama correspondientes a las tramas conocidas por el criptoanalista. El nivel de ruido del canal está ya tenido en cuenta mediante la elección de un determinado umbral T elegido según el valor del BER (Bit Error Ratio). Con estas consideraciones, el algoritmo se puede resumir de la siguiente forma:

Entradas: 4 tramas de la secuencia de salida del generador A5/2 y los correspondientes números de trama; valor umbral T elegido de acuerdo con el BER del canal.
Salida: bits reconstruidos de la secuencia de salida.

PASOS
1. Se inicializan (en cero):
   - valor numérico del estado inicial del registro R4 (s)
   - índice del número de trama (i)
   - número de ecuaciones linealmente independientes en el sistema (m)

2. Se elige el s-ésimo estado del registro R4 y se inicializa una variable que cuenta el número de discrepancias (d) entre la secuencia producida por el generador y la secuencia observada.

3. Se incrementa en 1 la variable correspondiente al índice del número de trama. Se completa el proceso de inicialización empezando a partir del estado s-ésimo del registro R4. Se guarda constancia de las dependencias lineales.

4. Si d>T entonces s=s+1 y se va la paso 2. Si se ha llegado al final de la trama, entonces se va al paso 3; en caso contrario el algoritmo A5/2 lleva a cabo un nuevo ciclo de operación generando un nuevo bit de secuencia, guardando constancia de las dependencias y planteando la ecuación que relaciona estas dependencias con el correspondiente bit se salida.

5. Se linealiza la ecuación obtenida, sustituyendo los términos no lineales por las nuevas variables compuestas y guardando sus correspondientes índices. Se agrega esta ecuación linealizada al sistema.

6. Se comprueba el rango del sistema de ecuaciones linealizado. Si el rango actual del sistema es igual al rango anterior y el bit de salida es conocido, entonces se comprueba si este bit coincide con el bit calculado. Si no coincide, entonces d=d+1, se vuelve al estado anterior del sistema y posteriormente el paso 4. Si el rango actual del sistema es igual al rango anterior y el bit de salida es desconocido, entonces se ejecuta una de las siguientes dos opciones: se calcula el bit desconocido, se vuelve al estado anterior del sistema y luego se va al paso 4 o se hace lo mismo pero sin calcular el bit desconocido. Esto último depende del valor numérico del mayor índice de fila del sistema de ecuaciones planteado.

Este ataque está basado en el estudio, para cada uno de los posibles estados iniciales del registro R4, de un sistema de ecuaciones linealizado cuyas variables son las componentes de los estados iniciales de los restantes registros. A cada pulso de reloj se añade una nueva ecuación y se calcula el rango del sistema. Si no hay incremento en el rango ni se trata de un sistema degenerado, entonces se puede calcular fácilmente el correspondiente bit de la secuencia de salida.

El procedimiento anterior es posible gracias a los puntos débiles del algoritmo A5/2 mencionados. Para restarle validez a este ataque, se analizan a continuación dichos puntos débiles y se propone posteriormente un algoritmo que carece de los mismos y que mantiene el rendimiento del A5/2 existente. Según lo expuesto, la mayor debilidad de este algoritmo está en las frecuentes reinicializaciones (a cada nueva trama) y por tanto ésta se tratará con especial atención.

11 DISEÑO DEL ALGORITMO A5/2+

11.1 PUNTOS DÉBILES DEL ALGORITMO A5/2

La principal desventaja de este algoritmo radica en la escasa longitud de los registros de desplazamiento, en especial la del registro R4 (tan solo de 17 bits) el cual controla por sí solo el desplazamiento de los otros tres registros. Esto permite desarrollar un procedimiento para generar y comprobar la secuencia de control de reloj que gobierna el desplazamiento de los registros R1, R2 y R3. Específicamente, para cada uno de los posibles estados iniciales de R4, se puede generar la correspondiente secuencia de control de reloj. Cada una de estas secuencias podría tener una longitud indefinida pero en la práctica sólo se utilizan 328 (100 + 228) elementos de la misma, pues como ya se ha dicho, tras generar 228 bits de una trama, el algoritmo se reinicializa con un nuevo número de trama. Este aspecto indica el primer punto débil del algoritmo A5/2: FRECUENTES REINICIALIZACIONES. Además, según se mencionó, antes de la generación de la secuencia de salida tan sólo se eliminan 100 bits (100 ciclos de operación), lo cual supone el segundo punto débil del algoritmo: BAJO NÚMERO DE BITS DESECHADOS. El último punto débil hace referencia a la MALA DISTRIBUCIÓN DE LAS REALIMENTACIONES DE LOS REGISTROS (a la derecha de las entradas a las funciones mayoría). Estos tres aspectos hacen posible descubrir las relaciones lineales existentes entre los sucesivos bits de la secuencia de salida [1].

11.1.1 REINICIALIZACIONES DEL GENERADOR

La numeración de las tramas en GSM va desde 0 hasta 2’715.647. Cada una de estas tramas contiene 8 canales físicos (“time slots”). El algoritmo utiliza el número de trama correspondiente junto con la clave Kc y genera 114 bits en sentido móvil 1-2 y posteriormente genera 114 bits en sentido móvil 2-1. Para la próxima trama a cifrar, se usa un nuevo número de trama (la misma clave de cifrado), el cual es cíclico en el intervalo dado. Esto permite cifrar de manera diferente la misma información enviada durante una conversación particular. La importancia del número de trama como entrada del algoritmo A5/2 radica en la criptosincronización que éste permite. Su magnitud es adecuada para efectos criptográficos, pero la gran desventaja aquí es la frecuencia de reinicialización del algoritmo (con un nuevo número de trama cada vez). Sumado a lo anterior está el hecho que el número de trama es públicamente conocido.

Debido al carácter de privacidad propio de este tipo de sistemas, una de las formas más comunes de atacarlos es mediante la ingeniería inversa (reconstruir la entrada a partir de una salida interceptada). El algoritmo A5/2 es susceptible de ser invertido eficientemente aunque con algunas dificultades. Al igual que en el algoritmo A5/1, la función de transición de estado del algoritmo A5/2 no es invertible en modo único. En el primero, por ejemplo, la regla de control del reloj mediante la función mayoría implica que hasta 4 estados pueden converger a un estado común en un ciclo de reloj mientras que algunos estados no tienen predecesores.

Según se explica en [2], es posible correr el algoritmo A5/1 hacia atrás explorando el árbol de posibles estados predecesores y retrocediendo de los extremos sin salida. El número promedio de predecesores de cada nodo es 1 y por tanto el número esperado de vértices en los primeros “k” niveles de cada árbol crece linealmente con k. Como consecuencia, si se encuentra un estado común en el disco y los datos, se puede obtener un pequeño número de candidatos para el estado inicial de la trama. Esto también es aplicable al algoritmo A5/2.

El aspecto que permite el corrimiento hacia atrás del algoritmo A5/2 es que debido a las frecuentes reinicializaciones del mismo, existe una muy corta distancia desde los estados intermedios hasta los estados iniciales, facilitando su inversión y por ende la reconstrucción de la secuencia de salida.

11.2 PLANTEAMIENTO DE LAS MEJORAS DEL ALGORITMO A5/2

Una conversación en GSM consiste en la transmisión de tramas cada 4,6 ms, cada una de las cuales consta de 228 bits. Esto significa que el algoritmo A5/2 se reinicializa (durante la misma conversación) cada 4,6 ms con un nuevo número de trama. La duración de una conversación promedio en el sistema GSM-1900 es de 1 min (operador OLA / red en la hora pico - mayo de 2004), lo que implica el envío de 13044 tramas aproximadamente y ese mismo número de reinicializaciones del algoritmo.

La reinicialización del algoritmo A5/2 implica una variación en la transmisión de una trama a la otra (misma conversación) a partir del paso 3 del procedimiento de inicialización del algoritmo. Esto como consecuencia del empleo de la misma clave Kc durante la misma conversación y el establecimiento en cero de los registros de desplazamiento al comienzo de dicho procedimiento. Para ilustrar mejor esta situación, se muestra en la figura 13 el diagrama de flujo del algoritmo A5/2. Como se observa, una vez generados los primeros 228 bits correspondientes a la primera trama, el algoritmo se inicializa colocando los cuatro registros en cero.

Este aspecto fue de vital importancia en el diseño del nuevo algoritmo (A5/2+), el cual implica cambios a nivel netamente de software soportados sin ningún problema por la infraestructura dispuesta en la red GSM.

11.3 CRITERIOS PARA EL PROCESO DE DISEÑO DE UN ALGORITMO CRIPTOGRÁFICO (3GPP)

La protección que ofrece un algoritmo de cifrado se debe evaluar siempre bajo la suposición que el atacante conoce todos los detalles del mismo y del sistema dentro del cual está implementado. Lo único que el atacante no conoce es la calve. Por supuesto, mantener un algoritmo en secreto brinda una protección adicional, aunque se ha demostrado a lo largo de los años que la robustez de los sistemas criptográficos no está en la privacidad (tarde que temprano terminan siendo descifrados) sino en el

diseño y la evaluación pormenorizada después de someter al algoritmo al escrutinio público.

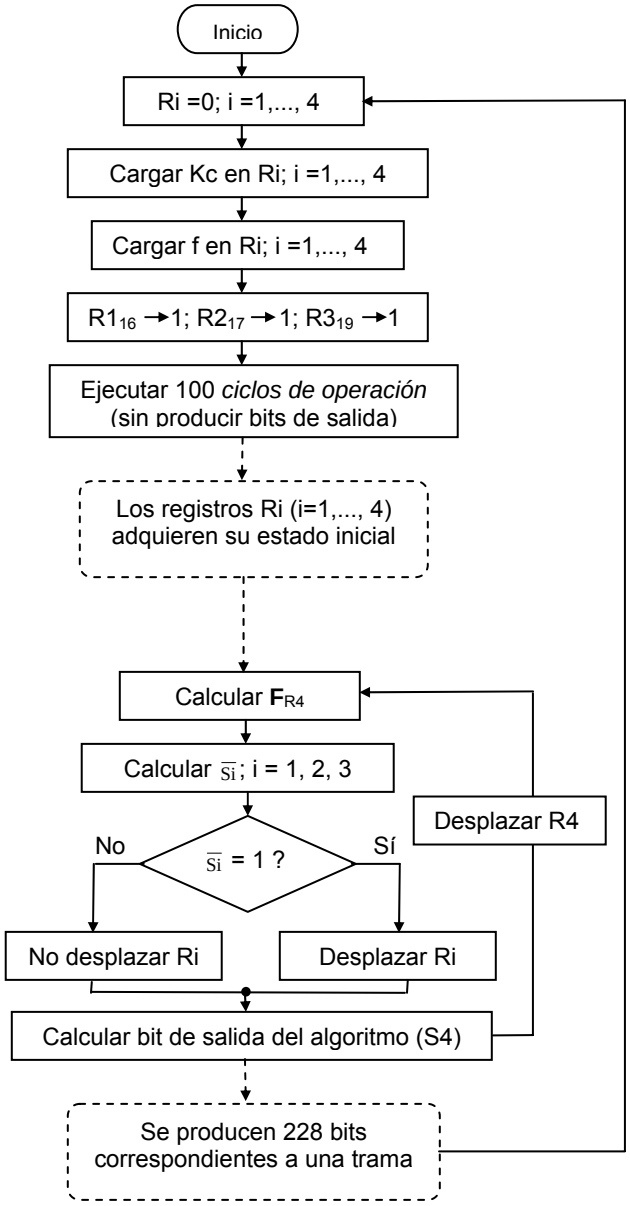

Figura 13. Diagrama de flujo del algoritmo A5/2 (existente).

Los posibles escenarios para el proceso de diseño de un algoritmo sugeridos por la entidad reguladora de los sistemas de telecomunicaciones de tercera generación (3GPP) son:

1. Utilizar un algoritmo público que llene los requerimientos.
2. Proponer un algoritmo confidencial: se selecciona un algoritmo público para crear un algoritmo confidencial.
3. Solicitar propuestas para el algoritmo: se reciben propuestas de algoritmos diseñados por grupos o personas y se evalúan esos diseños.
4. Comisionar un grupo especial para el diseño del algoritmo.

Por el carácter de la presente investigación, se hizo obligatorio acoger el segundo escenario. A pesar de que el algoritmo A5/2 en principio se mantuvo en secreto, con el pasar de lo años los criptoanalistas lo pusieron al descubierto mediante ingeniería inversa. Una implementación pedagógica del A5/2 fue dada a conocer en 1999 por Marc Briceño, Ian Goldberg y David Wagner. En ésta se comprueba el correcto funcionamiento del A5/2 revertido por ellos, comprobando su salida contra un vector de prueba conocido obtenido de la red real (ver anexo A).

De acuerdo con el segundo escenario para el proceso de diseño de un nuevo algoritmo, la entidad 3GPP propone utilizar un algoritmo público y modificarlo de manera confidencial y específica sin necesidad de hacer un completo análisis del mismo.

Procedimiento:

1. Se identifica y selecciona un cierto algoritmo público que puede utilizarse como base para el algoritmo confidencial.
2. Se hacen las modificaciones al algoritmo público de manera confidencial. Se realiza un breve análisis para verificar que las modificaciones no afectan la seguridad de dicho algoritmo.

De acuerdo con esta metodología se hace la propuesta del algoritmo A5/2+.

11.4 CARACTERÍSTICAS DEL DISEÑO (ALGORITMO A5/2+)

Siguiendo las observaciones descritas, se propone un nuevo algoritmo basado en las siguientes modificaciones:

1. Después de la transmisión de la primera trama, los registros no se llevan a cero para el cifrado de las tramas siguientes (durante la misma conversación). Se utiliza el estado actual de los mismos una vez cifrada la trama previa. El algoritmo se reinicializa nuevamente desde el paso 1 (que ahora sufre una modificación de acuerdo con la quinta característica) cada cinco segundos de conversación (ver cuarta característica).
   RAZÓN: a pesar de su reinicialización y antes de completar cinco segundos de conversación, para la transmisión de las tramas posteriores a la primera el algoritmo no empieza su funcionamiento desde un único estado (registros en cero). Con esto se pudo eliminar la desventaja consistente en que los estados

intermedios del algoritmo estuvieran siempre a una distancia corta del estado inicial del mismo.

2. La clave de cifrado Kc se carga en los registros en la primera trama de la conversación y siempre que el algoritmo se reinicie desde el paso 1 (darle un valor inicial a los registros de desplazamiento) de acuerdo con la cuarta característica. Antes que se completen cinco segundos de conversación, el procedimiento de inicialización para el cifrado de las tramas posteriores a la primera empieza ahora en el paso 3 (cargar el número de trama f en los registros).
   RAZÓN: no es necesario cargar la clave Kc en tramas posteriores a la primera y antes de cinco segundos de conversación, ya que desde la transmisión de ésta los registros comienzan su transición de un estado a otro nunca regresando a los dos únicos estados iniciales, evitando así la cercanía de los estados intermedios con los iniciales. Con esta característica también se disminuyó el tiempo de ejecución del procedimiento de inicialización del algoritmo diseñado respecto del algoritmo existente.

3. Los 100 ciclos de operación ahora sólo se ejecutan para la primera trama de la conversación y siempre que el algoritmo se reinicie desde el paso 1 de acuerdo con la cuarta característica.
   RAZÓN: los 100 ciclos se ejecutan para la modificación del contenido de los registros y así darle mayor imprevisibilidad al estado inicial del algoritmo. Debido a las dos primeras características introducidas, éstos son útiles para la primera trama de la conversación y siempre que el algoritmo se reinicie desde el paso 1, ya que posterior a la reinicialización, los LFSR están en constante transición de un estado a otro. Con esta característica también se disminuyó el tiempo de ejecución del procedimiento de inicialización del algoritmo diseñado respecto del existente.

4. Se introdujo una variable que cuenta el número de bits de cifrado generados con el algoritmo A5/2 + durante una conversación y que se inicializa en cero después de obtener 247836 de los mismos (cada 5 segundos de conversación). Cada vez que se genere este número de bits, el algoritmo se reinicializa desde el paso 1 (darle un valor numérico al estado inicial de los LFSR).
   RAZÓN: la reinicialización del algoritmo desde el paso 1 permite mantener el sincronismo entre la estación móvil y la red en el seguimiento del algoritmo de seguridad implementado. Con esto se logra la corrección de los posibles errores (por ruido, interferencia, radiaciones, etc.) que pueden presentarse en la ejecución del algoritmo al interior de la red y que provocarían la divergencia en las salidas de los bloques de cifrado / descifrado en ambos lados del sistema (estación móvil-estación base), provocando a su vez la imposibilidad de descifrar un mensaje ya cifrado
   .
5. Los registros no siempre se inicializan en cero para la primera trama y a cada cinco segundos de conversación, sino que dependiendo de la numeración de la trama TDMA transmitida y de la clave de cifrado Kc, los registros adquieren su estado inicial de acuerdo con el siguiente procedimiento:

- Si al combinar en operación XOR el número de trama TDMA con los 22 bits más significativos de Kc resulta un valor par, los registros se reinicializan en cero.
- Si al combinar en operación XOR el número de trama TDMA con los 22 bits más significativos de Kc resulta un valor impar, los registros se reinicializan en un valor predefinido por el operador de red GSM (denotado por V).

RAZÓN: el llevar los registros a cero siempre que se ejecuta un procedimiento de reinicialización del algoritmo desde el paso 1 (primera trama y a cada cinco segundos de conversación) le restaba seguridad al mismo contra los ataques descritos, ya que según el detalle de los mismos [1], el conocimiento de menos de cuatro tramas de la secuencia de salida es suficiente para reconstruir el resto de la secuencia generada por el algoritmo A5/2 existente. Sin embargo, aunque con las características planteadas hasta ahora se necesitarían por lo menos 15 segundos de conversación para la obtención de las cuatro tramas con las que se haría el ataque (ya que el algoritmo reinicializa sus registros en cero cada 5 segundos), se buscó darle robustez a la seguridad del sistema mediante la introducción de esta característica, bien sea que ese tiempo de "escucha clandestina" no es lo suficientemente alto. Gracias a que esta característica propone la utilización de un parámetro desconocido por quienes interceptan la señal (Kc), se logró restarle aún más importancia a los ataques mencionados. Además, entra a jugar un papel importante el criterio del operador de red al poder ajustar el estado inicial de los registros (tras la reinicialización) de acuerdo con su evaluación.

El diagrama de flujo del algoritmo A5/2+ se muestra en la figura 14.

11.5 RESUMEN COMPARATIVO (A5/2 VS. A5/2 +)

Los algoritmos A5/2 y A5/2+ presentan diferencias muy claras en su funcionamiento, no así en su estructura interna. Éste fue un aspecto de vital importancia en el diseño del algoritmo A5/2+, dada la imposibilidad de entrar a modificar la constitución propia de un algoritmo consolidado dentro de la red GSM hace algún tiempo. La diferencia básica de los algoritmos según se muestra en la figura 15, es que para el A5/2 los registros siempre se ponen en cero (un único estado) a cada nueva trama, mientras que en el A5/2+ los registros utilizan el estado que adquirieron en el cifrado de la trama anterior antes que se completen 5 segundos de conversación y posterior a esto se inicializan en dos posibles estados determinados por Kc y f. Además, aunque no se muestra, en aquellas tramas en las que no se inicializan los registros, el A5/2+ omite cargar Kc en los LFSR y elimina la ejecución de los 100 ciclos de operación para la modificación del contenido de los mismos (aspectos importantes en el tiempo de ejecución del algoritmo).

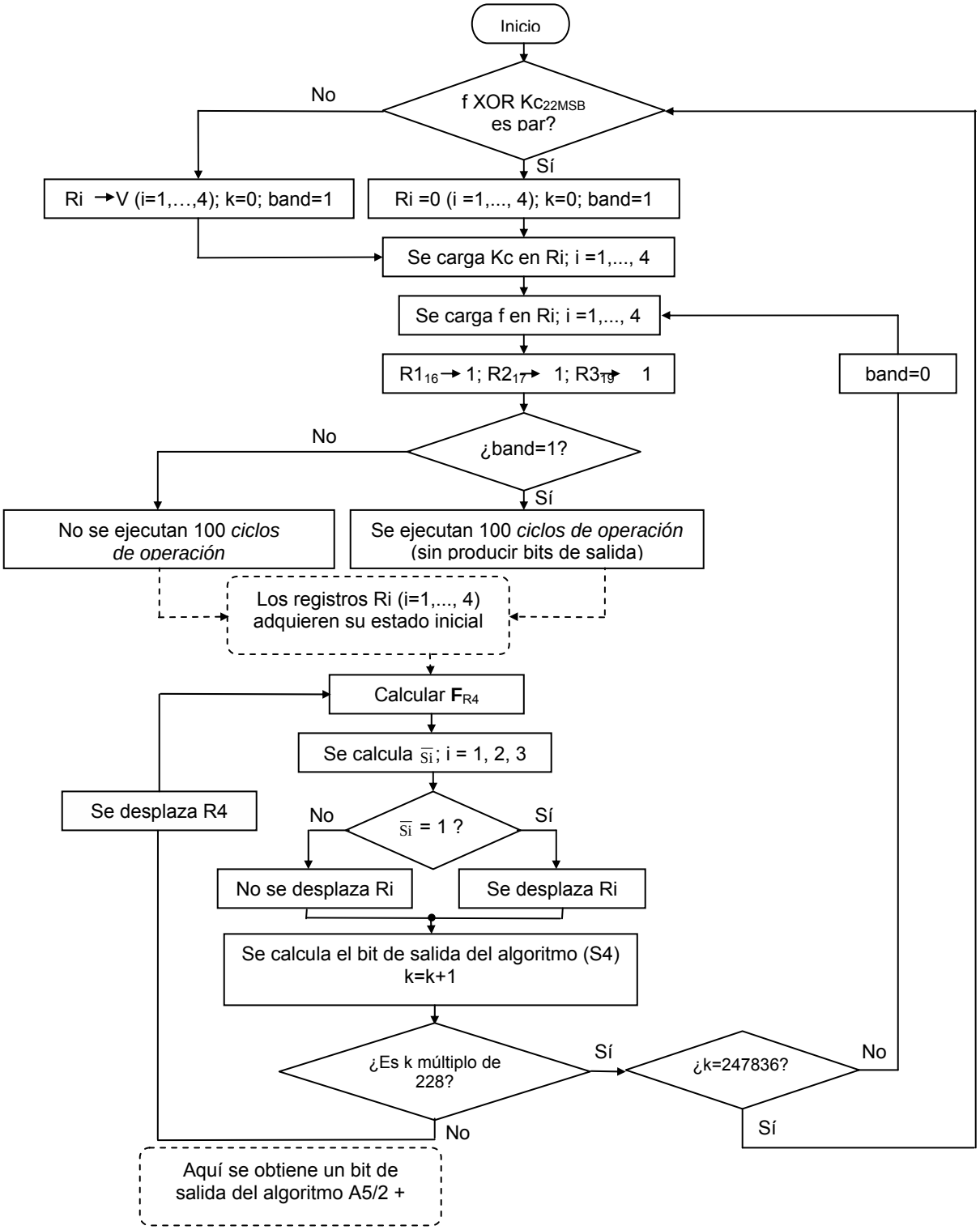

Figura 14. Diagrama de flujo del algoritmo A5/2 +.

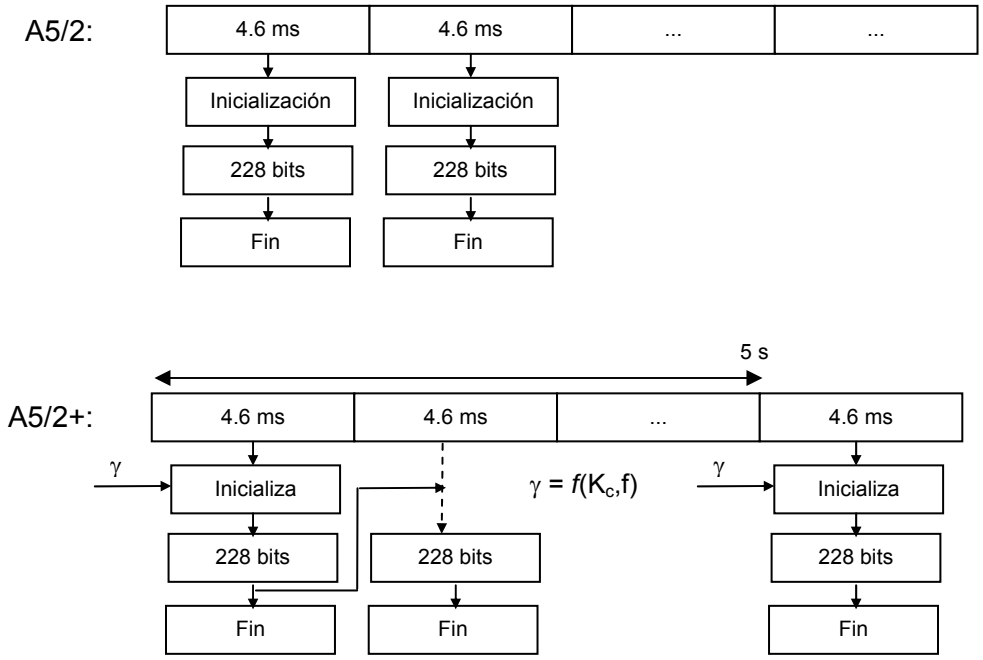

Figura 15. Resumen comparativo A5/2 vs. A5/2+

11.6 EVALUACIÓN DEL ALGORITMO A5/2+

Lo primero en la evaluación del correcto funcionamiento de un generador de secuencia binaria pseudoaleatoria es la comprobación de una serie de postulados que garantizan la seguridad del mismo dentro del contexto de la criptografía. Es decir, mediante los siguientes postulados se comprueba que la secuencia binaria generada se aproxima a una secuencia completamente aleatoria. A continuación se presentan los resultados arrojados por el algoritmo A5/2+ en dicha evaluación:

1. El periodo de la secuencia binaria generada por este tipo algoritmos debe ser lo más largo posible. Como los polinomios de realimentación de los cuatro registros son primitivos, el periodo de la secuencia generada por el algoritmo A5/2 + está dado por:

$$P = \text{mcm}\,(2^{n1}-1,\ 2^{n2}-1,\ 2^{n3}-1,\ 2^{n4}-1) = 2{,}4178\text{x}10^{24}$$

   donde los "ni" corresponden a las longitudes de los "Ri" respectivamente para i=1,...,4 (n1=19, n2=22, n3=23, n4=17).

   Aunque lo ideal sería que la secuencia generada tuviera un periodo tan largo como el texto claro, asegurar esto en la práctica es imposible. Por esta razón, los

sistemas reales trabajan bajo el principio de tener secuencias de cifrado con los periodos más largos posibles. En este caso se cumple con el requisito gracias a la configuración de cuatro LFSR con polinomios de realimentación primitivos. Con el periodo obtenido es posible cifrar $1{,}06x10^{22}$ tramas, correspondientes a $4{,}88x10^{19}$ segundos de conversación (suficiente en la práctica).

2. Postulados de Golomb.

2.1 En cada periodo de la secuencia binaria no debe haber grandes diferencias entre el número de "unos" y el número de "ceros".

Debido a la magnitud del periodo de la secuencia cifrante, se optó por tomar un subconjunto de la misma lo suficientemente grande tal que la estadística de éste representara el periodo completo. Para una cantidad de bits de cifrado superior a 2'500.000 generada por el algoritmo A5/2+, la estadística obtenida indicó que el 48% fueron "unos" y el 52% fueron "ceros", ratificando el postulado.

2.2 En cada período de la secuencia binaria debe haber $(½)^i$ subsecuencias de longitud “i” tanto para "unos" como para "ceros". Pero además, debe haber el mismo número total de subsecuencias de "unos" y de "ceros".

Para el mismo número de bits que en el caso anterior, el generador produjo 714202 subsecuencias de "unos" y 714448 subsecuencias de "ceros". Del número total de subsecuencias de "unos" 357263 fueron de longitud 1 (aproximadamente la mitad del total), 178580 de longitud 2 (aproximadamente una cuarta parte del total) y 89220 de longitud 3 (aproximadamente una octava parte del total). El resto de subsecuencias seguía esta misma línea. De manera semejante se comportaron las subsecuencias de "ceros", lo cual ratificó el postulado

2.3 La función de autocorrelación de la secuencia binaria debe ser bivalor en un período.

En las figuras 16 y 17 se muestran las funciones de autocorrelación para las secuencias binarias pseudoaleatorias generadas por ambos algoritmos y para un desplazamiento máximo de la secuencia de 1000 posiciones.

Como se observa, las funciones de autocorrelación de ambos algoritmos muestran resultados muy similares y óptimos de acuerdo con el tercer postulado de Golomb. Como se aprecia en las gráficas, se efectuó un máximo desplazamiento de la secuencia binaria de 1000 posiciones (bits), para el cual los valores de autocorrelación permanecieron insignificantes excepto para el origen. Es decir, al desplazar la secuencia generada un número determinado de posiciones (hacia la derecha en este caso), el número de coincidencias entre bits es prácticamente nulo (correlación cero), mientras que sin desplazamiento de la secuencia, las coincidencias son todas, exhibiendo correlación máxima en el origen.

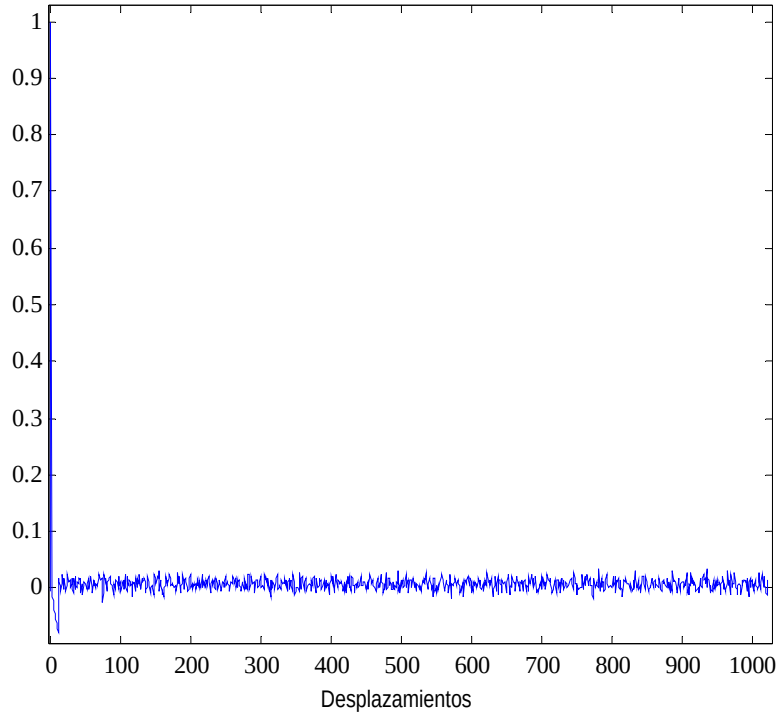

Figura 16. Función de autocorrelación para la secuencia generada por el algoritmo A5/2.

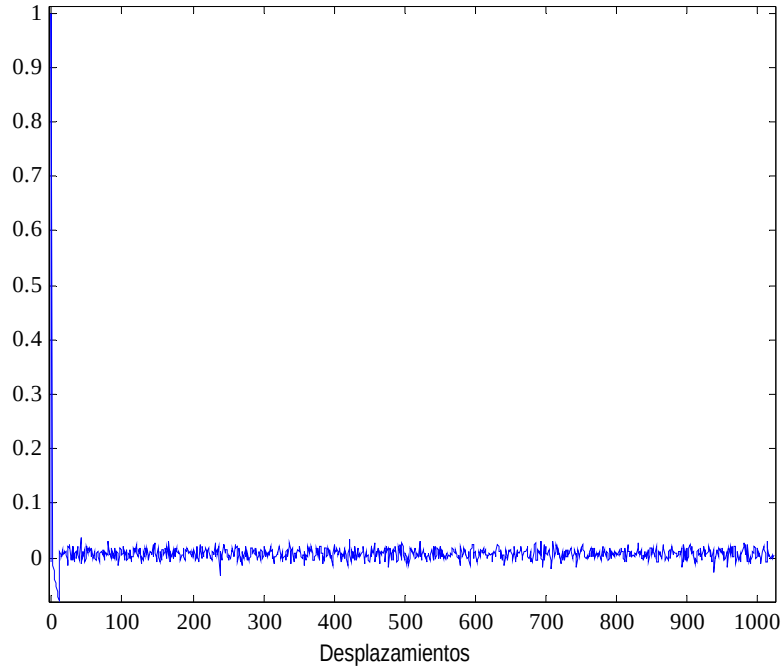

Figura 17. Función de autocorrelación para la secuencia generada por el algoritmo A5/2+.

3. Al aplicar una transformada de Fourier sobre la secuencia generada por el algoritmo se debe obtener un pico máximo en el origen y un valor muy pequeño en el resto de las componentes frecuenciales.

   Según se observa en las figuras 18 y 19, los espectros de ambos algoritmos satisfacen a cabalidad este postulado. Para la componente de frecuencia nula (nivel dc) la gráfica muestra un pico máximo y el resto de las componentes exhiben valores mínimos. Esto es consecuencia de la correcta pseudoaleatoriedad de la secuencia generada por ambos algoritmos que conlleva a que los aportes de los múltiples armónicos para la reconstrucción de la secuencia original sean muchos en cantidad pero insignificantes en amplitud.

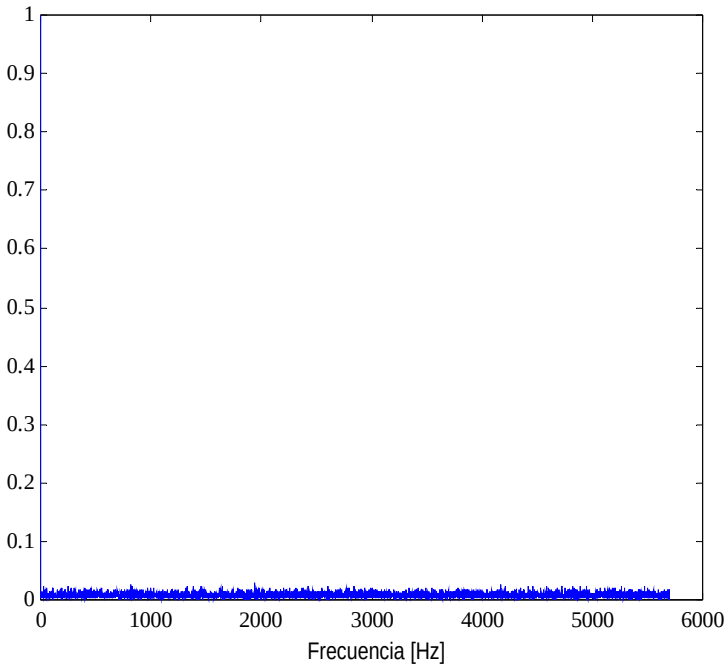

Figura 18. Espectro en frecuencia de la secuencia pseudoaleatoria generada por el algoritmo A5/2.

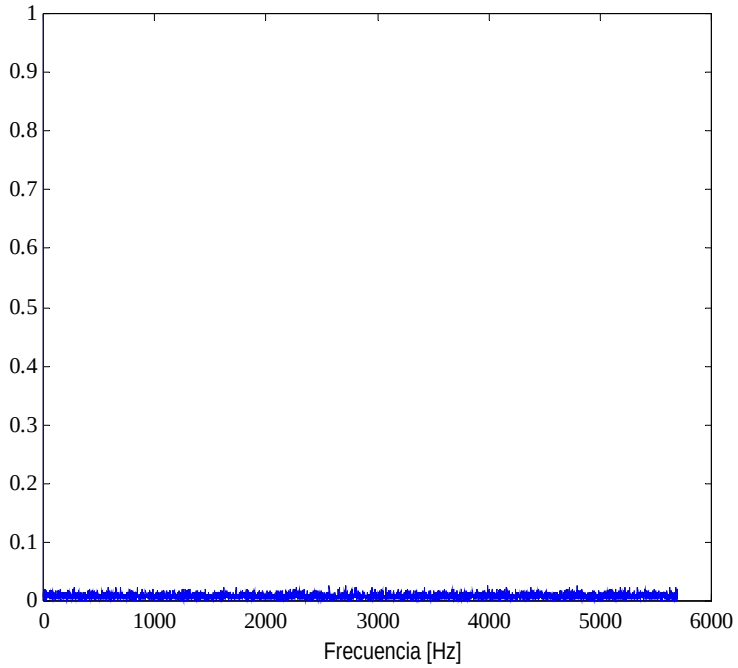

Figura 19. Espectro en frecuencia de la secuencia pseudoaleatoria generada por el algoritmo A5/2+.

Con la comprobación de los postulados anteriores se demostró el correcto funcionamiento del generador A5/2+ respecto a la pseudoaleatoriedad de la secuencia generada. Esto posibilita su uso dentro del sistema de seguridad de la red GSM.

Respecto al desempeño del algoritmo A5/2+ dentro de la red GSM, se evaluó el tiempo de generación de bits de cifrado de cada uno de los dos algoritmos. Evidentemente, para la generación de los 228 bits correspondientes a la primera trama los tiempos son muy parecidos. El aspecto de relevancia aquí es el tiempo necesario para la generación de los bits de cifrado de las tramas posteriores a la primera absoluta y a la primera después de cada cinco segundos de conversación. Gracias a las características 2 y 3 del diseño, este tiempo es menor en el algoritmo A5/2+. Para una cantidad de 8'000.000 de bits aproximadamente (35087 tramas), los tiempos arrojados por ambos algoritmos fueron:

A5/2 = 25,1 seg
A5/2+ = 18,4 seg

Como se observa, éste es inferior en el algoritmo diseñado, lo cual permite mejorar el rendimiento del sistema, ya que según las especificaciones GSM, el algoritmo debe generar una trama cada 4,6 ms. Esto podría ser favorable al algoritmo A5/2+ a la hora de la inclusión de nuevos procesos en los que se necesiten tiempos adicionales.

12 CONCLUSIONES

Se ha hecho el diseño y se ha implementado en Matlab y en lenguaje de programación C un algoritmo para el cifrado de la voz en el enlace estación móvil – estación base dentro de la red GSM. El punto de partida del diseño fue el análisis del algoritmo A5/2 existente, el cual ha sido objeto de innumerables ataques criptográficos en los últimos años, los cuales han posibilitado las escuchas indebidas de las conversaciones privadas de los usuarios dentro del sistema.

Debido a la existencia de una infraestructura de red consolidada en GSM, se hizo el diseño teniendo presente la no exigencia de modificaciones físicas en la misma, bien sea que ella se encuentra avalada por los estándares internacionales desde hace tiempo. Por esta razón, el nuevo algoritmo también incluyó variables como los registros de desplazamiento realimentados linealmente con los mismos polinomios primitivos, la clave de cifrado Kc y el número de trama TDMA además que se mantuvieron invariables sus longitudes. Las nuevas variables introducidas, dada su simplicidad, tampoco exigieron cambios físicos a la red GSM.

Por lo anterior, el algoritmo A5/2+ basó su diseño, más que en la creación de nuevos procesos y variables, en el uso eficiente de los procesos predeterminados dentro de la red GSM, teniendo presente los ataques hechos hasta el momento a la seguridad del sistema. Para este algoritmo, por ejemplo, el número de trama TDMA y la clave de cifrado Kc aumentan en importancia al ser las variables decisorias desde el inicio del algoritmo. Además, se buscó disminuir la vulnerabilidad del algoritmo diseñado al mismo tiempo que se posibilitó la participación dinámica del operador de red GSM en el funcionamiento del mismo, mediante el establecimiento de los registros de desplazamiento (tras la reinicialización) a un valor establecido de acuerdo con su criterio.

Se pueden sintetizar las bondades del algoritmo A5/2+ en los siguientes 4 puntos:

- El algoritmo A5/2+ elimina las desventajas del A5/2 y por tanto deja de ser susceptible a los ataques criptoanalíticos existentes.
- La secuencia binaria pseudoaleatoria generada por el algoritmo A5/2+ es correcta dentro del contexto criptográfico y posibilita su uso a nivel de la seguridad de los sistemas de telecomunicaciones
- El algoritmo A5/2+ le da mayor importancia al número de trama TDMA y a la clave de cifrado Kc que el A5/2. La clave Kc (secreta) es la base de estos generadores binarios y al aumentar en importancia fortalece al sistema.
- El tiempo de ejecución del algoritmo A5/2+ es, en promedio, menor que el del A5/2. Además de mantener su rendimiento (respecto a la secuencia binaria generada) y eliminar sus debilidades, lo mejora en esta característica.

Aunque con la evaluación efectuada al desempeño del algoritmo A5/2+ es suficiente para catalogarlo como un gran aporte a la seguridad de la telefonía móvil, posibles estudios futuros podrían incluir la implementaron práctica dentro de la red GSM a fin de evaluar su rendimiento real. De acuerdo con éste, se podría replantear el tiempo de reinicialización del algoritmo desde el paso 1 a un valor diferente a 5 segundos.

13 BIBLIOGRAFÍA

[1] PETROVIC Slobodan, FÚSTER Amparo. Artículo “Criptoanálisis del Algoritmo A5/2 para telefonía móvil”. Instituto de Física Aplicada C.S.I.C. Madrid, España. http://eprint.iacr.org

[2] BIRYUKOV Alex, SHAMIR Adi, WAGNER David. Artículo “Real Time Cryptanalyisis of A5/1 on a PC”.

[3] www.etsi.org

GSM 02.07: Digital cellular telecommunications system (Phase 2+); Mobile Stations (MS) features.

GSM 02.09: Security aspects.

GSM 03.03: Digital cellular telecommunications system (Phase 2+); Numbering, addressing and identification.

GSM 03.20: Digital cellular telecommunications system (Phase 2+); Security related network functions.

GSM 04.01: Digital cellular telecommunications system (Phase 2+); Mobile Station - Base Station System (MS - BSS) interface; General aspects and principles.

GSM 04.02: Digital cellular telecommunications system (Phase 2+); GSM Public Land Mobile Network (PLMN) access reference configuration.

GSM 04.03: Digital cellular telecommunications system (Phase 2+); Mobile Station - Base Station System (MS - BSS) Interface Channel Structures and Access Capabilities.

GSM 05.01: Digital cellular telecommunications system (Phase 2+); Physical layer on the radio path; General description.

GSM 05.02: Digital cellular telecommunications system (Phase 2+); Multiplexing and multiple access on the radio path.

GSM 05.03: Digital cellular telecommunications system (Phase 2+); Channel coding.

GSM 05.05: Digital cellular telecommunications system (Phase 2+); Radio transmission and reception.

GSM 05.10: Digital cellular telecommunications system (Phase 2+); Radio subsystem synchronisation.

[4] www.3gpp.org

3GPP 33.901: Technical Report: 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Criteria for cryptographic algorithm design process; (Release 4).

3GPP 33.102:Technical Report: 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security Architecture (Release 6).

3GPP 33.105:Technical Report: 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Criptographic Algorthm Requirements (Release 4).

3GPP 35. 201:Technical Report: 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: *f8* and *f9* Specification.

[5] BARKAN Elad, BIHAM Eli, KELLER Nathan. Instant Ciphertext_Only Cryptanalysis of GSM Encrypted Communication. Technion - Computer Science Department, Technical Report, Israel 2003.
[6] ZENG K.C., YANG C.H y RAO T.R. Artículo: "On the Linear Consistency Test (LCT) in Cryptanalysis with Applications". Crypto '89. Lecture Notes in Computer Science 435, Berlin 1989.
[7] BALSTON D., MACARIO R.; Cellular Radio Systems. Boston; London: Artech House, c1993.
[8] AREITIO, J. "Síntesis de Generadores de Secuencias Pseudoaeatorias basados en LFSRs". Conectrónica. Noviembre 1999.
[9] KERNIGHAN Brian, RITCHIE Dennis. El Lenguaje de Programación C. México: Prentice-Hall Hispanoamericana, c1991. 2a ed.
[10] Páginas en Internet:
http://www.melodiasmoviles.com/documentacion/red-gsm.php
http://anas.worldonline.es/wonder3f/infogsm.htm
http://neutron.ing.ucv.ve/revista-e/No2/Gsmseuly.htm
http://acd.asoc.euitt.upm.es/~irebol/cifradores/cf.html
http://acd.asoc.euitt.upm.es/~irebol/cifradores/pagapli.html
http://www.argo.es/~jcea/artic/lfsr.htm

ANEXO A. IMPLEMENTACIÓN PEDAGÓGICA DEL A5/2 EN LENGUAJE C (Marc Briceño, Ian Goldberg y David Wagner)

El siguiente programa es una implementación del algoritmo A5/2 para el cifrado de la voz empleado en el sistema GSM. El programa es una comprobación de la correcta obtención del código del algoritmo A5/2 mediante ingeniería inversa.

```
/*
 * A pedagogical implementation of the GSM A5/1 and A5/2 "voice privacy"
 * encryption algorithms.
 *
 * Copyright (C) 1998-1999: Marc Briceno, Ian Goldberg, and David Wagner
 *
 * The source code below is optimized for instructional value and clarity.
 *
 * This software may be export-controlled by US law.
 *
 * -- Marc Briceno     <marc@scard.org>
 *    Voice:          +1 (925) 798-4042
 */

#include <stdio.h>
#include <stdlib.h>
#define A5_2

/* Masks for the shift registers */
#define R1MASK  0x07FFFF /* 19 bits, numbered 0..18 */
#define R2MASK  0x3FFFFF /* 22 bits, numbered 0..21 */
#define R3MASK  0x7FFFFF /* 23 bits, numbered 0..22 */
#ifdef A5_2
#define R4MASK  0x01FFFF /* 17 bits, numbered 0..16 */
#endif /* A5_2 */

#ifndef A5_2
/* Middle bit of each of the three shift registers, for clock control */
#define R1MID   0x000100 /* bit 8 */
#define R2MID   0x000400 /* bit 10 */
#define R3MID   0x000400 /* bit 10 */
#else /* A5_2 */
/* A bit of R4 that controls each of the shift registers */
#define R4TAP1  0x000400 /* bit 10 */
#define R4TAP2  0x000008 /* bit 3 */
#define R4TAP3  0x000080 /* bit 7 */
#endif /* A5_2 */

/* Feedback taps, for clocking the shift registers.
 * These correspond to the primitive polynomials
 * x^19 + x^18 + x^17 + x^14 + 1, x^22 + x^21 + 1,
```

```
 * x^23 + x^22 + x^21 + x^8 + 1, and x^17 + x^12 + 1. */

#define R1TAPS  0x072000 /* bits 18,17,16,13 */
#define R2TAPS  0x300000 /* bits 21,20 */
#define R3TAPS  0x700080 /* bits 22,21,20,7 */
#ifdef A5_2
#define R4TAPS  0x010800 /* bits 16,11 */
#endif /* A5_2 */

typedef unsigned char byte;
typedef unsigned long word;
typedef word bit;

/* Calculate the parity of a 32-bit word, i.e. the sum of its bits modulo 2 */
bit parity(word x) {
	x ^= x>>16;
	x ^= x>>8;
	x ^= x>>4;
	x ^= x>>2;
	x ^= x>>1;
	return x&1;
}

/* Clock one shift register.  For A5/2, when the last bit of the frame
 * is loaded in, one particular bit of each register is forced to '1';
 * that bit is passed in as the last argument. */
#ifndef A5_2
word clockone(word reg, word mask, word taps) {
#else /* A5_2 */
word clockone(word reg, word mask, word taps, word loaded_bit) {
#endif /* A5_2 */
	word t = reg & taps;
	reg = (reg << 1) & mask;
	reg |= parity(t);
#ifdef A5_2
	reg |= loaded_bit;
#endif /* A5_2 */
	return reg;
}

/* The three shift registers.  They're in global variables to make the code
 * easier to understand.
 * A better implementation would not use global variables. */
word R1, R2, R3;
#ifdef A5_2
word R4;
#endif /* A5_2 */
```

```
/* Return 1 iff at least two of the parameter words are non-zero. */
bit majority(word w1, word w2, word w3) {
	int sum = (w1 != 0) + (w2 != 0) + (w3 != 0);
	if (sum >= 2)
		return 1;
	else
		return 0;
}

/* Clock two or three of R1,R2,R3, with clock control
 * according to their middle bits.
 * Specifically, we clock Ri whenever Ri's middle bit
 * agrees with the majority value of the three middle bits.  For A5/2,
 * use particular bits of R4 instead of the middle bits.  Also, for A5/2,
 * always clock R4.
 * If allP == 1, clock all three of R1,R2,R3, ignoring their middle bits.
 * This is only used for key setup.  If loaded == 1, then this is the last
 * bit of the frame number, and if we're doing A5/2, we have to set a
 * particular bit in each of the four registers. */
void clock(int allP, int loaded) {
#ifndef A5_2
	bit maj = majority(R1&R1MID, R2&R2MID, R3&R3MID);
	if (allP || (((R1&R1MID)!=0) == maj))
		R1 = clockone(R1, R1MASK, R1TAPS);
	if (allP || (((R2&R2MID)!=0) == maj))
		R2 = clockone(R2, R2MASK, R2TAPS);
	if (allP || (((R3&R3MID)!=0) == maj))
		R3 = clockone(R3, R3MASK, R3TAPS);
#else /* A5_2 */
	bit maj = majority(R4&R4TAP1, R4&R4TAP2, R4&R4TAP3);
	if (allP || (((R4&R4TAP1)!=0) == maj))
		R1 = clockone(R1, R1MASK, R1TAPS, loaded<<15);
	if (allP || (((R4&R4TAP2)!=0) == maj))
		R2 = clockone(R2, R2MASK, R2TAPS, loaded<<16);
	if (allP || (((R4&R4TAP3)!=0) == maj))
		R3 = clockone(R3, R3MASK, R3TAPS, loaded<<18);
	R4 = clockone(R4, R4MASK, R4TAPS, loaded<<10);
#endif /* A5_2 */
}

/* Generate an output bit from the current state.
 * You grab a bit from each register via the output generation taps;
 * then you XOR the resulting three bits.  For A5/2, in addition to
 * the top bit of each of R1,R2,R3, also XOR in a majority function
 * of three particular bits of the register (one of them complemented)
 * to make it non-linear.  Also, for A5/2, delay the output by one
 * clock cycle for some reason. */
bit getbit() {
```

```
	bit topbits = (((R1 >> 18) ^ (R2 >> 21) ^ (R3 >> 22)) & 0x01);
#ifndef A5_2
	return topbits;
#else /* A5_2 */
	static bit delaybit = 0;
	bit nowbit = delaybit;
	delaybit = (
	   topbits
	   ^ majority(R1&0x8000, (~R1)&0x4000, R1&0x1000)
	   ^ majority((~R2)&0x10000, R2&0x2000, R2&0x200)
	   ^ majority(R3&0x40000, R3&0x10000, (~R3)&0x2000)
	   );
	return nowbit;
#endif /* A5_2 */
}

/* Do the A5 key setup.  This routine accepts a 64-bit key and
 * a 22-bit frame number. */
void keysetup(byte key[8], word frame) {
	int i;
	bit keybit, framebit;

/* Zero out the shift registers. */
	R1 = R2 = R3 = 0;
#ifdef A5_2
	R4 = 0;
#endif /* A5_2 */

	/* Load the key into the shift registers,
	 * LSB of first byte of key array first,
	 * clocking each register once for every
	 * key bit loaded.  (The usual clock
	 * control rule is temporarily disabled.) */
	for (i=0; i<64; i++) {
		clock(1,0); /* always clock */
		keybit = (key[i/8] >> (i&7)) & 1; /* The i-th bit of the key */
		R1 ^= keybit; R2 ^= keybit; R3 ^= keybit;
#ifdef A5_2
		R4 ^= keybit;
#endif /* A5_2 */
	}

	/* Load the frame number into the shift registers, LSB first,
	 * clocking each register once for every key bit loaded.
	 * (The usual clock control rule is still disabled.)
	 * For A5/2, signal when the last bit is being clocked in. */
	for (i=0; i<22; i++) {
		clock(1,i==21); /* always clock */
```

```
            framebit = (frame >> i) & 1; /* The i-th bit of the frame # */
            R1 ^= framebit; R2 ^= framebit; R3 ^= framebit;
#ifdef A5_2
            R4 ^= framebit;
#endif /* A5_2 */
        }

        /* Run the shift registers for 100 clocks
         * to mix the keying material and frame number
         * together with output generation disabled,
         * so that there is sufficient avalanche.
         * We re-enable the majority-based clock control
         * rule from now on. */
        for (i=0; i<100; i++) {
            clock(0,0);
        }
        /* For A5/2, we have to load the delayed output bit.  This does _not_
         * change the state of the registers.  For A5/1, this is a no-op. */
        getbit();

        /* Now the key is properly set up. */
}

/* Generate output.  We generate 228 bits of
 * keystream output.  The first 114 bits is for
 * the A->B frame; the next 114 bits is for the
 * B->A frame.  You allocate a 15-byte buffer
 * for each direction, and this function fills
 * it in. */
void run(byte AtoBkeystream[], byte BtoAkeystream[]) {
        int i;

/* Zero out the output buffers. */
        for (i=0; i<=113/8; i++)
            AtoBkeystream[i] = BtoAkeystream[i] = 0;

/* Generate 114 bits of keystream for the
         * A->B direction.  Store it, MSB first. */
        for (i=0; i<114; i++) {
            clock(0,0);
            AtoBkeystream[i/8] |= getbit() << (7-(i&7));
        }

/* Generate 114 bits of keystream for the
         * B->A direction.  Store it, MSB first. */
        for (i=0; i<114; i++) {
            clock(0,0);
            BtoAkeystream[i/8] |= getbit() << (7-(i&7));
```

```
    }
}

/* Test the code by comparing it against
 * a known-good test vector. */
void test() {
#ifndef A5_2
    byte key[8] = {0x12, 0x23, 0x45, 0x67, 0x89, 0xAB, 0xCD, 0xEF};
    word frame = 0x134;
    byte goodAtoB[15] = { 0x53, 0x4E, 0xAA, 0x58, 0x2F, 0xE8, 0x15,
                0x1A, 0xB6, 0xE1, 0x85, 0x5A, 0x72, 0x8C, 0x00 };
    byte goodBtoA[15] = { 0x24, 0xFD, 0x35, 0xA3, 0x5D, 0x5F, 0xB6,
                0x52, 0x6D, 0x32, 0xF9, 0x06, 0xDF, 0x1A, 0xC0 };
#else /* A5_2 */
    byte key[8] = {0x00, 0xfc, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff};
    word frame = 0x21;
    byte goodAtoB[15] = { 0xf4, 0x51, 0x2c, 0xac, 0x13, 0x59, 0x37,
                0x64, 0x46, 0x0b, 0x72, 0x2d, 0xad, 0xd5, 0x00 };
    byte goodBtoA[15] = { 0x48, 0x00, 0xd4, 0x32, 0x8e, 0x16, 0xa1,
                0x4d, 0xcd, 0x7b, 0x97, 0x22, 0x26, 0x51, 0x00 };
#endif /* A5_2 */
    byte AtoB[15], BtoA[15];
    int i, failed=0;

    keysetup(key, frame);
    run(AtoB, BtoA);

    /* Compare against the test vector. */
    for (i=0; i<15; i++)
        if (AtoB[i] != goodAtoB[i])
            failed = 1;
    for (i=0; i<15; i++)
        if (BtoA[i] != goodBtoA[i])
            failed = 1;

    /* Print some debugging output. */
    printf("key: 0x");
    for (i=0; i<8; i++)
        printf("%02X", key[i]);
    printf("\n");
    printf("frame number: 0x%06X\n\n", (unsigned int)frame);
    printf("known good output:\n");
    printf(" A->B: 0x");
    for (i=0; i<15; i++)
        printf("%02X", goodAtoB[i]);
    printf("  B->A: 0x");
    for (i=0; i<15; i++)
        printf("%02X", goodBtoA[i]);
```

```
        printf("\n");
        printf("\nobserved output:\n");
        printf(" A->B: 0x");
        for (i=0; i<15; i++)
                printf("%02X", AtoB[i]);
        printf("  B->A: 0x");
        for (i=0; i<15; i++)
                printf("%02X", BtoA[i]);
        printf("\n");

        if (!failed) {
                printf("\nSelf-check succeeded: everything looks ok.\n\n");
                system("PAUSE");
        } else {
                /* Problems!  The test vectors didn't compare*/
                printf("\nI don't know why this broke; contact the authors.\n");
                system("PAUSE");
        }
}
int main(void) {
        test();
        return 0;
}
```

ANEXO B. IMPLEMENTACIÓN EN MATLAB DEL ALGORITMO A5/2+

A continuación se muestra la programación en Matlab del algoritmo A5/2+. El programa tiene como salida la secuencia binaria pseudoaleatoria generada por el algoritmo. En el siguiente anexo se obtienen las estadísticas de dicha secuencia.

```
%--------------------------------------------------------------------------
% A5/2+
% Implementacion del algoritmo de encriptacion A5/2+
% Carlos Donado Coronell - 2004
%--------------------------------------------------------------------------
function [pns,R1,R2,R3,R4,F_ant]=A5_2p(F,K,R1,R2,R3,R4,F_ant)
%
%                       [pns,R1,R2,R3,R4]=A5_2(F,K,R1,R2,R3,R4,F_ant)
% Esta funcion implementa una version del algoritmo A5/2+, donde
%       F :     es el numero de trama
%       K :     es la clave de cifrado
%   R1, R2, R3, R4 son los LFSR
%   F_ant: numero de trama anterior, y puede ser inicializada en vacio
%   pns es un conjunto de 228 bits pseudoaleatorios
%   Si ningun parametro es asignado a la funcion, F = 0x21, K = 0x00FCFFFFFFFFFF;
%   R1 = R2 = R3 = R4 = 0, F_ant = [];
%   Esta version esta basada en el trabajo de grado "Diseno e Implementacion de un
Algoritmo
%   para la encriptacion de la voz en el sistema PCS sobre la tecnologia
%   TDMA(GSM)".

%
% Version implementada y probada por Carlos Donado Coronell - 2004
%

if (nargin<1)
  temp1=dec2bin(hex2dec('21'),22);
  temp1=fliplr(temp1);
  for i=1:length(temp1)
    F(i)=str2num(temp1(i));
  end;
end;

if (nargin<2)|isempty(K)
  temp1=[dec2bin(hex2dec('ff'),8) dec2bin(hex2dec('ff'),8) dec2bin(hex2dec('ff'),8),...
      dec2bin(hex2dec('ff'),8) dec2bin(hex2dec('ff'),8) dec2bin(hex2dec('ff'),8),...
      dec2bin(hex2dec('fc'),8) dec2bin(hex2dec('00'),8)];
  temp1=fliplr(temp1);
  for i=1:length(temp1)
    K(i)=str2num(temp1(i));
  end;
end;
```

```
if (nargin<3) | isempty(R1) | isempty(R2) | isempty(R3) | isempty(R4)
  R1=zeros(1,19);
  R2=zeros(1,22);
  R3=zeros(1,23);
  R4=zeros(1,17);
end

if (nargin<7)
  F_ant=[];
end;

%%Inicializacion
flag=0;

if isempty(F_ant)
  flag=1;
  F_ant=F;
        if (not(mod(sum([F(1) K(43)]),2)))
                R1=zeros(1,19);
                R2=zeros(1,22);
                R3=zeros(1,23);
        R4=zeros(1,17);
        else
        R1=zeros(1,19); for i=2:2:19, R1(i)=1; end;
        R2=zeros(1,22); for i=2:2:22, R2(i)=1; end;
        R3=zeros(1,23); for i=2:2:23, R3(i)=1; end;
        R4=zeros(1,17); for i=2:2:17, R1(i)=1; end;
  end;
end

st1=[]; for ki=1:length(F), st1=[st1,num2str(F(ki))]; end; st1=fliplr(st1); Fd=bin2dec(st1);
st1=[]; for ki=1:length(F_ant), st1=[st1,num2str(F_ant(ki))]; end; st1=fliplr(st1);
F_antd=bin2dec(st1);

if (F_antd > Fd)
  F_ant=zeros(1,22);
  st1=[]; for ki=1:length(F_ant), st1=[st1,num2str(F_ant(ki))]; end; st1=fliplr(st1);
F_antd=bin2dec(st1);
end

if (Fd-F_antd)>1087
  flag=1;
  F_ant=F;
        if (not(mod(sum([F(1) K(43)]),2)))
                R1=zeros(1,19);
                R2=zeros(1,22);
                R3=zeros(1,23);
```

```
        R4=zeros(1,17);
        else
        R1=zeros(1,19); for i=2:2:19, R1(i)=1; end;
        R2=zeros(1,22); for i=2:2:22, R2(i)=1; end;
        R3=zeros(1,23); for i=2:2:23, R3(i)=1; end;
        R4=zeros(1,17); for i=2:2:17, R1(i)=1; end;
  end;
end

if (flag==1)
  for i=1:length(K)              % length(K) debe ser 64
        R1=register(R1,mod(sum([K(i) R1(14) R1(17) R1(18) R1(19)]),2),1);
        R2=register(R2,mod(sum([K(i) R2(21) R2(22)]),2),1);
        R3=register(R3,mod(sum([K(i) R3(8) R3(21) R3(22) R3(23)]),2),1);
        R4=register(R4,mod(sum([K(i) R4(12) R4(17)]),2),1);
  end
end


for i=1:length(F)              % length(F) debe ser 22
  R1=register(R1,mod(sum([F(i) R1(14) R1(17) R1(18) R1(19)]),2),1);
  R2=register(R2,mod(sum([F(i) R2(21) R2(22)]),2),1);
  R3=register(R3,mod(sum([F(i) R3(8) R3(21) R3(22) R3(23)]),2),1);
  R4=register(R4,mod(sum([F(i) R4(12) R4(17)]),2),1);
end
R1(16)=1;
R2(17)=1;
R3(19)=1;

if (flag==1)
        for i=1:100
        c=mayoria(R4(11),R4(4),R4(8));
        c1=mod(sum([1 c R4(11)]),2);
        bt=mod(sum([R1(14) R1(17) R1(18) R1(19)]),2);
        R1=register(R1,bt,c1);
%       st1=[]; for ki=1:length(R1), st1=[st1,num2str(R1(ki))]; end; st1=fliplr(st1);
bin2dec(st1)
                bt=mod(sum([R2(21) R2(22)]),2);
                c2=mod(sum([1 c R4(4)]),2);
        R2=register(R2,bt,c2);
%       st1=[]; for ki=1:length(R2), st1=[st1,num2str(R2(ki))]; end; st1=fliplr(st1);
bin2dec(st1)
                bt=mod(sum([R3(8) R3(21) R3(22) R3(23)]),2);
                c3=mod(sum([1 c R4(8)]),2);
        R3=register(R3,bt,c3);
%       st1=[]; for ki=1:length(R3), st1=[st1,num2str(R3(ki))]; end; st1=fliplr(st1);
bin2dec(st1)
        bt=mod(sum([R4(12) R4(17)]),2);
```

```
        R4=register(R4,bt,1);
%       st1=[]; for ki=1:length(R4), st1=[st1,num2str(R4(ki))]; end; st1=fliplr(st1);
bin2dec(st1)
        end;
end;

%  Operacion Normal
for i=1:228
  pns(i)=mod(sum([mayoria(R1(13),not(R1(15)),R1(16))
mayoria(R2(10),R2(14),not(R2(17))),...
      mayoria(not(R3(14)),R3(17),R3(19)) R1(19) R2(22) R3(23)]),2);
         c=mayoria(R4(11),R4(4),R4(8));
         c1=mod(sum([1 c R4(11)]),2);
         R1=register(R1,mod(sum([R1(14) R1(17) R1(18) R1(19)]),2),c1);
         c2=mod(sum([1 c R4(4)]),2);
         R2=register(R2,mod(sum([R2(21) R2(22)]),2),c2);
         c3=mod(sum([1 c R4(8)]),2);
         R3=register(R3,mod(sum([R3(8) R3(21) R3(22) R3(23)]),2),c3);
  R4=register(R4,mod(sum([R4(12) R4(17)]),2),1);
end


function Ro=register(Ri,bi,c)
% Ro = register(Ri,c,bi)
% if c=1, el registro se desplaza

if c==1
  Ri(2:length(Ri))=Ri(1:length(Ri)-1);
  Ri(1)=bi;
end
Ro=Ri;

function bit=mayoria(b1,b2,b3)

bit = mod(sum(b1*b2+b1*b3+b2*b3),2);
```

ANEXO B.1. PROGRAMA EN MATLAB PARA OBTENER LA ESTADÍSTICA DE LA SECUENCIA BINARIA GENERADA POR EL ALGORITMO A5/2+

Con la siguiente rutina se ejecuta el algoritmo A5/2+. Da como resultado la gráfica de autocorrelación, el espectro en frecuencia y la distribución de "unos" y de "ceros" de la secuencia binaria pseudoaleatoria generada por el algoritmo.

```
%Archivo de prueba para el algoritmo A5_2+
F1d=hex2dec('21');
pns_t=[];
R1=[];
R2=[];
R3=[];
R4=[];
F_ant=[];
for i=1:50                    %50 corridas del algoritmo
        temp1=dec2bin(F1d,22);
        temp1=fliplr(temp1);
  for i=1:length(temp1)
    F(i)=str2num(temp1(i));
  end;
  [pns,R1,R2,R3,R4,F_ant]=A5_2p(F,[],R1,R2,R3,R4,F_ant);
  F1d=F1d+1;
  if (F1d==2715648),
    F1d=0;
  end;
  pns_t=[pns_t pns];
end;
%Analisis de Datos de Salida
%Analisis frecuencia
fft1p=fft(pns_t);
figure(1);
plot(abs(fft1p(1:length(fft1p)/2+1))/max(abs(fft1p)))
title('Analisis frecuencial de la secuencia binaria pseudoaleatoria del A5/2+')
%Analisis de correlacion
figure(2);
[lRp,Rp,CLp]=cra([pns_t' pns_t'],2^10);
plot(Rp(length(Rp)/2-1:length(Rp),1),Rp(length(Rp)/2-1:length(Rp),2)/max(Rp(:,2)))
title('Analisis de correlacion de la secuencia binaria pseudoaleatoria del A5/2+')
xlabel('Desplazamientos')
figure(3)
clf
total_unos=sum(pns_t)/length(pns_t)*100;
total_ceros=(length(pns_t)-sum(pns_t))/length(pns_t)*100;
bar([0,1], [total_ceros,total_unos])
title('Total de ceros y de unos en la secuencia de encriptacion (%)')
ylabel('%')
text(0,total_ceros+1,[num2str(total_ceros),'%'])
text(1,total_unos+1,[num2str(total_unos),'%'])
```

ANEXO C. IMPLEMENTACIÓN EN LENGUAJE C DEL ALGORITMO A5/2+

Con el siguiente programa se cifra mediante el algoritmo A5/2+ el mensaje contenido en un archivo particular. El mensaje cifrado y la secuencia binaria de cifrado generada por el A5/2+ se almacenan en otros dos archivos. Por último, el mensaje descifrado utilizando el mismo algoritmo se muestra en un tercer archivo.

```
/*
 *-------------------------------------------------------
 * Implementación del algoritmo de encriptación A5/2+
 * Carlos Donado Coronell - 2004
 *-------------------------------------------------------
 *
 *  Esta versión está basada en el trabajo de grado "Diseño e implementación de un
algoritmo
 *  para la encriptación de la voz en el sistema PCS sobre la tecnología TDMA(GSM)"
 *  Versión original implementada en Matlab 5.2
 *
 *  Este programa encripta mediante el algoritmo A5/2+ los datos de texto
almacenados
 *  en el archivo c:\temp\carlos~1\file1.txt; el mensaje cifrado lo almacena en el
archivo
 *  c:\cifplus.txt; la secuencia binaria de cifrado la almacena en el archivo c:\cifout.txt
 *  y por último, el mensaje descifrado con el mismo algoritmo lo almacena en el
archivo
 *  c:\fileplus.txt.
 *
 *  Número de trama f = 0x134
 *  Clave de cifrado Kc = 0x EFCDAB8967452312
 */


#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

/* Máscara para los cuatro registros de desplazamiento */
#define R1MASK  0x07FFFF /* 19 bits, numerados 0..18 */
#define R2MASK  0x3FFFFF /* 22 bits, numerados 0..21 */
#define R3MASK  0x7FFFFF /* 23 bits, numerados 0..22 */
#define R4MASK  0x01FFFF /* 17 bits, numerados 0..16 */

/* Máscara para obtener un bit particular de R4 que controla el
 * desplazamiento de los registros 1, 2 y 3 */
#define R4TAP1  0x000400 /* bit 10 */
#define R4TAP2  0x000008 /* bit 3 */
#define R4TAP3  0x000080 /* bit 7 */
```

```
/* Polinomios de realimentación primitivos utilizados en el
 * desplazamiento de los cuatro registros.
 * x^19 + x^18 + x^17 + x^14 + 1, x^22 + x^21 + 1,
 * x^23 + x^22 + x^21 + x^8 + 1 y x^17 + x^12 + 1. */

#define R1TAPS  0x072000 /* bits 18,17,16,13 de R1 */
#define R2TAPS  0x300000 /* bits 21,20 de R2 */
#define R3TAPS  0x700080 /* bits 22,21,20,7 de R3 */
#define R4TAPS  0x010800 /* bits 16,11 de R4 */

/* Tamaño máximo para secuencias */

#define TAM_MAX 500

/* Definición de tipos de variables*/

typedef unsigned char byte;
typedef unsigned long word;
typedef word bit;

/* Se definen los cuatro registros de desplazamiento
 * en variables globales y el número de trama anterior.*/
word R1, R2, R3, R4;
word frame_ant=0xFFFFFF;

/* FUNCIÓN parity: cálculo de la paridad de una palabra de 32 bits, es decir,
 * la suma de sus bits en módulo 2. Si la paridad es impar, regresa 1. Se
 * utiliza para hacer la operación de los polinomios primitivos. */
bit parity(word x) {
        x ^= x>>16;
        x ^= x>>8;
        x ^= x>>4;
        x ^= x>>2;
        x ^= x>>1;
        return x&1;
}

/* FUNCIÓN clockone: desplaza y realimenta uno de los registros de
 * desplazamiento. Una vez cargado el último bit del número de trama, se
 * lleva a '1' un bit particular de los registros 1,2 y 3 */

word clockone(word reg, word mask, word taps, word loaded_bit) {
        word t = reg & taps;
        reg = (reg << 1) & mask;
        reg |= parity(t);
        reg |= loaded_bit;
        return reg;
```

```
}

/* FUNCIÓN majority: regresa 1 si y sólo si al menos dos de los parámetros de
 * entrada son diferentes de cero. Corresponde a la función mayoría. */

bit majority(word w1, word w2, word w3) {
        int sum = (w1 != 0) + (w2 != 0) + (w3 != 0);
        if (sum >= 2)
                return 1;
        else
                return 0;
}

/* FUNCIÓN clock: realimenta dos o tres de los registros entre R1,R2 y R3
 * de acuerdo con un bit particular del registro R4 y con la función mayoría
 * asociada a éste. El registro R4 siempre se realimenta.
 * Si allP == 1, se realimentan los tres registros R1,R2,R3, ignorando la regla
 * anterior. Esto sólo se usa para cargar la clave. Si loaded == 1, entonces se
 * está en el último bit del número de trama y se tiene que llevar a '1' un bit
 * particular de cada uno de los tres primeros registros. */
void clockm(int allP, int loaded) {
        bit maj = majority(R4&R4TAP1, R4&R4TAP2, R4&R4TAP3);
        if (allP || (((R4&R4TAP1)!=0) == maj))
                R1 = clockone(R1, R1MASK, R1TAPS, loaded<<15);
        if (allP || (((R4&R4TAP2)!=0) == maj))
                R2 = clockone(R2, R2MASK, R2TAPS, loaded<<16);
        if (allP || (((R4&R4TAP3)!=0) == maj))
                R3 = clockone(R3, R3MASK, R3TAPS, loaded<<18);
        R4 = clockone(R4, R4MASK, R4TAPS, loaded<<10);
}

/* FUNCIÓN getbit: genera un bit de salida con el estado actual de los
 * registros. Se obtiene efectuando la operación XOR entre el bit superior
 * de cada uno de los registros 1, 2 y 3 y las salidas de las funciones
 * mayoría asociadas con ellos. Además, la salida está retrasada un ciclo
 * de operación. */
bit getbit() {
        bit topbits = (((R1 >> 18) ^ (R2 >> 21) ^ (R3 >> 22)) & 0x01);
        static bit delaybit = 0;
        bit nowbit = delaybit;
        delaybit = (
          topbits
          ^ majority(R1&0x8000, (~R1)&0x4000, R1&0x1000)
          ^ majority((~R2)&0x10000, R2&0x2000, R2&0x200)
          ^ majority(R3&0x40000, R3&0x10000, (~R3)&0x2000)
          );
        return nowbit;
}
```

```
/* FUNCIÓN keysetup: carga en los cuatro registros la clave de cifrado Kc
(64 bits) y el número de trama f (22 bits). */
void keysetup(byte key[8], word frame) {
        int i;
        int flag=0;
        bit keybit, framebit;


        //-----------------------------------------------------------------//
        //              Característica número 1            //
        //                    Característica número 4     //
        //                Característica número 5         //
        //-----------------------------------------------------------------//
        if (frame_ant==0xFFFFFF)
        {
                flag=1;
                frame_ant=frame;
                if (!(((frame&1)^(key[5]>>2))&1)) //XOR entre f y los 22 MSB de Kc
                        R1 = R2 = R3 = R4 = 0;
                else
                        R1 = R2 = R3 = R4 = 2796202;
        }
        if (frame_ant>frame)
                frame_ant=0;
        if (frame-frame_ant>1087)     //Si han transcurrido 5 seg de conversación (1087
tramas)
        {
                flag=1;
                frame_ant=frame;
                if (!(((frame&1)^(key[5]>>2))&1))
                        R1 = R2 = R3 = R4 = 0;
                else
                        R1 = R2 = R3 = R4 = 2796202;
        }

        /* Carga la clave Kc en los cuatro registros de desplazamiento
         * empezando por el bit menos significativo y realimentando cada
         * registro una vez por cada bit cargado. Hace lo anterior si se ha
         * reinicializado el algoritmo. La regla de control de
         * reloj está temporalmente deshabilitada. */
         //-----------------------------------------------------------//
         //              Característica número 2            //
         //-----------------------------------------------------------//
         if (flag == 1)
         {
                for (i=0; i<64; i++) {
```

```
                        clockm(1,0);                                    /* siempre
realimenta */
                        keybit = (key[i/8] >> (i&7)) & 1; /* i-ésimo bit de la clave */
                        R1 ^= keybit; R2 ^= keybit; R3 ^= keybit; R4 ^= keybit;
                }
                flag=0;
        }

        /* Carga el número de trama f en los cuatro registros de desplazamiento
         * empezando por el bit menos significativo y realimentando cada
         * registro una vez por cada bit cargado. La regla de control de
         * reloj está temporalmente deshabilitada. Además, señaliza cuando se
         * está realimentando el último bit. */
        for (i=0; i<22; i++) {
                clockm(1,i==21); /* siempre realimenta */
                framebit = (frame >> i) & 1; /* i-ésimo bit del número de trama */
                R1 ^= framebit; R2 ^= framebit; R3 ^= framebit; R4 ^= framebit;
    }

        /* Se ejecutan 100 ciclos de operación del algoritmo para mezclar
         * la clave Kc y el número de trama f dentro de los registros sin
         * producir bits de salida. Hace lo anterior si se ha
         * reinicializado el algoritmo. Se vuelve a habilitar la regla de control
         * del reloj basada en la función mayoría desde ahora en adelante. */
         //-------------------------------------------------------------//
         //                  Característica número 3        //
         //-------------------------------------------------------------//
        if (flag==1)
        {
                flag=0;
                for (i=0; i<100; i++) {
                        clockm(0,0);
                }
        }
        /* Se debe cargar el bit de salida retrasado, lo cual no cambia el
         * estado de los registros. */
        getbit();
}

/* FUNCIÓN run: genera la salida del algoritmo. Se generan 228 bits del flujo de
 * salida; los primeros 114 bits de la trama corresponden a la dirección A->B;
 * los restantes 114 bits corresponden a la dirección B->A.  Se utiliza un buffer
 * de 15 bytes para cada dirección. */
void run(byte AtoBkeystream[], byte BtoAkeystream[]) {
        int i;

        /* Se inicializan en cero los buffers de salida. */
        for (i=0; i<=113/8; i++)
```

```
                AtoBkeystream[i] = BtoAkeystream[i] = 0;

        /* Se generan 114 bits del flujo correspondientes a la dirección A->B.
         * Se almacena primero el bit más significativo. */
        for (i=0; i<114; i++) {
                clockm(0,0);
                AtoBkeystream[i/8] |= getbit() << (7-(i&7));
        }

        /* Se generan 114 bits del flujo correspondientes a la dirección B->A.
         * Se almacena primero el bit más significativo. */
        for (i=0; i<114; i++) {
                clockm(0,0);
                BtoAkeystream[i/8] |= getbit() << (7-(i&7));
        }
}

/* FUNCIÓN test: prueba el funcionamiento de las secuencias de cifrado. */
void test() {
        char mensaje [TAM_MAX];
        char cifrado [TAM_MAX];
        char descifrado [TAM_MAX];
        char respuesta;
        byte key[8] = {0x12, 0x23, 0x45, 0x67, 0x89, 0xAB, 0xCD, 0xEF};
        word frame = 0x134;
        byte AtoB[15], BtoA[15], secuencia_cif[15], secuencia_cifr[15];
        int i, k,k_r, longitud;
        byte bit_men, bit_sec;

        for(i=0; i<TAM_MAX; i++)
                cifrado[i]=descifrado[i]=0x00;
        keysetup(key, frame);
        run(AtoB, BtoA);

        //-----------------------------------------------------------------//
        printf("\n\nSi desea la secuencia de cifrado A-B digite 1; \n");
        printf("si desea la secuencia de cifrado B-A digite 2:\n");
        fflush(stdin);
        respuesta=getchar();

        if(respuesta=='1')
        {
                memcpy(secuencia_cif, AtoB, 15);
                memcpy(secuencia_cifr, AtoB, 15);
        }
        else if (respuesta=='2')
        {
                memcpy(secuencia_cif, BtoA, 15);
```

```
            memcpy(secuencia_cifr, BtoA, 15);
    }
    //-----------Abre para lectura el archivo que va a ser encriptado//

    FILE *in, *out, *interout, *cifout;
    if ((in = fopen("c:\\temp\\carlos~1\\file1.txt", "rt"))== NULL)
    {
            fprintf(stderr, "Cannot open input file.\n");
    }
    if ((out = fopen("\\fileplus.txt", "wt")) == NULL)
    {
            fprintf(stderr, "Cannot open output file.\n");
    }
    if ((interout = fopen("\\cifplus.txt", "wt")) == NULL)
    {
            fprintf(stderr, "Cannot open output file.\n");
    }
            if ((cifout = fopen("\\cifout.txt", "wt")) == NULL)
    {
            fprintf(stderr, "Cannot open output file.\n");
    }

    //-----------Inicialización del reloj de ejecución del programa--//
    clock_t start, end;
    start = clock();

    k=0;
    k_r=0;
while (!feof(in))
{
    mensaje[0]=fgetc(in);
    if (mensaje[0] != -1)
    {
            longitud=8;             //Encriptamos cada caracter del archivo
            cifrado[0]=NULL;
            descifrado[0]=NULL;
            for(i=0; i<longitud; i++){
                    bit_men=(mensaje[i/8]>>(7-(i&7)))&1;
                    if(k==114)
                    {
                            k=0;
                            frame++;
                            if (frame > (51*26*2048-1))
                                    frame=0x00;
                            keysetup(key, frame);
                            run(AtoB, BtoA);
                            if(respuesta=='1')
                                    memcpy(secuencia_cifr, AtoB, 15);
```

```
                        else if (respuesta=='2')
                                memcpy(secuencia_cifr, BtoA, 15);
                        }
                        bit_sec=(secuencia_cifr[k/8]>>(7-(k&7)))&1;
                        fprintf(cifout,"%d",bit_sec);
                        k++;
                        cifrado[i/8] |= ((bit_sec^bit_men)&1)<<(7-(i&7));
                }
                fputc(cifrado[0],interout);
                for(i=0; i<longitud; i++){
                        bit_men=(cifrado[i/8]>>(7-(i&7)))&1;
                        if (k_r==114)
                        {
                                k_r=0;
                                if(respuesta=='1')
                                        memcpy(secuencia_cif, AtoB, 15);
                                else if (respuesta=='2')
                                        memcpy(secuencia_cif, BtoA, 15);
                        }
                        bit_sec=(secuencia_cif[k_r/8]>>(7-(k_r&7)))&1;
                        k_r++;
                        descifrado[i/8] |= ((bit_sec^bit_men)&1)<<(7-(i&7));
                }
                fputc(descifrado[0],out);
        }
        }
        end=clock();
        //printf("El Tiempo de computo fue: %f\n", (end - start)/CLK_TCK);
        printf ("\n\n\nEl mensaje del archivo file1.txt (C:temp_carlos~1) ha sido
cifrado.\n\n");
        printf ("El mensaje cifrado se guarda en el archivo cifplus.txt (C:\).\n\n");
        printf ("La secuencia binaria de cifrado se almacena en el archivo cifout.txt
(C:\).\n\n");
        printf ("El mensaje descifrado utilizando la misma secuencia binaria se
almacena en \n");
        printf ("el archivo fileplus.txt (C:\)\n\n");
        printf ("POR LOS SIMBOLOS GENERADOS, SE SUGIERE OBSERVAR EL MENSAJE
CIFRADO\n");
        printf ("EN EL ENTORNO DE MS-DOS\n\n\n\n");
        fclose(in);
        fclose(out);
        fclose(interout);
}
/* PROGRAMA PRINCIPAL*/
int main(void) {
        test();
        return 0;
}
```