Análisis del posible uso del Broadcast Encryption en la televisión de pago y en otras aplicaciones

AUTOR

Javier Ferrer Aguilar

DIRECTOR DE PROYECTO

Carles Padró

UPC

Abril 2009

Índice general

1 Introducción 1

1.1 Contexto 1

1.2 Propósitos y objetivos 2

1.3 Limitaciones 3

1.4 Estructura de la memoria 4

2 Sistemas de Acceso Condicional (CAS) 6

2.1 Arquitectura común de los CAS 7

2.1.1 Jerarquía de claves 9

2.1.2 Distribución de claves 10

2.1.3 Interoperabilidad CA 11

2.1.3.1 SimulCrypt 11

2.1.3.2 MultiCrypt 12

2.1.3.3 Comparación 13

2.2 DVB 13

2.2.1 Common Scrambling Algorithm (CSA) 14

2.2.1.1 Cifrador en flujo 15

2.2.1.2 Cifrador en bloque 16

2.3 Algunos sistemas CA propietarios 16

2.3.1 VideoGuard 16

2.3.2 Nagravision 17

2.3.3 Irdeto Access 19

2.3.4 Conax 19

2.3.5 Viaccess 20

2.4 Piratería 21

2.4.1 Mercado de la piratería de la televisión de pago en la UE 21

2.4.2 Atractivo del Mercado de la piratería de la televisión de pago 22

2.4.3 Efectos económicos de la piratería de la televisión de pago 22

2.4.3.1 En los proveedores de CAS 22

2.4.3.2 En los proveedores de televisión de pago ... 23
2.4.4 Clasificación de los piratas en diferentes grupos ... 23
2.4.5 Ataques ... 24
2.4.5.1 Comunidad Black Hat ... 25
2.4.5.2 Ataques sin tarjeta ... 25
2.4.5.3 Tarjeta compartida (Card Sharing) ... 26
2.4.6 Medidas anti-piratería utilizadas ... 27
2.4.7 Medidas anti-piratería propuestas ... 28
2.4.8 Traitor tracing ... 29

3 Broadcast Encryption ... 31
3.1 Introducción ... 31
3.2 Algoritmos Stateless ... 33
3.2.1 Esquemas en forma de árbol ... 34
3.2.1.1 Complete Subtree (CS) ... 35
3.2.1.2 Subset Difference (SD) ... 38
3.2.1.3 Layered Subset Difference (LSD) ... 43
3.2.1.4 Seguridad ... 45
3.2.2 Esquemas con estructura lineal ... 46
3.2.2.1 Basic Chain Scheme (BCS) ... 47
3.2.2.2 C-basic Chain Scheme (CBCS) ... 49
3.2.2.3 Seguridad ... 51
3.2.2.4 Skipping Chain Scheme (SCS) ... 51
3.3 Resumen comparativo de los BES estudiados ... 56

4 Análisis ... 58
4.1 Seguridad de los CAS ... 58
4.1.1 Servicios de seguridad ... 58
4.1.2 Clasificación de los mecanismos de seguridad ... 61
4.1.3 Algoritmos criptográficos usados en la distribución de claves ... 62
4.1.4 Comunicación receptor – *smart card* ... 68
4.1.4.1 Análisis del método propuesto ... 72
4.1.5 Rupturas de CAS (caso concreto de Nagra 2) ... 73
4.2 Comparativa entre el CAS y los BES ... 78

4.2.1 Simulación 79
4.2.2 Resultados 80
4.3 Implantación del BE en los CAS actuales 91
4.3.1 Primer esquema propuesto 91
4.3.2 Segundo esquema propuesto 93

5 Advanced Access Content System (AACS) 98
5.1 Antecesores 99
5.2 Cifrado y descifrado de medios pre-grabados 99
5.3 Autenticación de la Unidad y el host mediante acuerdo de clave 105
5.3.1 Protocolo de autenticación mutua y protocolo de acuerdo de clave 106
5.3.2 Descripción del esquema de autenticación del AACS Unidad-Host 106
5.3.3 Análisis del esquema de autenticación del AACS unidad-host 107
5.4 La seguridad del AACS en el mundo real 108

6 Conclusiones 110

7 Glosario de términos 113

8 Referencias 115

9 Anexos 121
9.1 Algoritmos de cifrado 121
9.1.1 DES (Data Encryption Standard) 121
9.1.2 Triple DES 123
9.1.3 IDEA (Internacional Data Encryption Algorithm) 124
9.1.4 AES (Advanced Encryption Standard) 126
9.1.5 RSA (Rivest Shamir Adleman) 130
9.2 Modos de Operación para Algoritmos de Cifrado por Bloques 132
9.2.1 Modo ECB 132
9.2.2 Modo CBC 132
9.2.3 Modo CFB 133
9.3 Simulación 134

9.4 Ejemplo simple del descifrado de un EMM de Nagravision 2 140
9.5 Esquema de autenticación AACS unidad-host..... 149

Índice de figuras

Figura 1: Esquema de la difusión de televisión vía satélite 2
Figura 2: Logos de algunos de los estándares del DVB 7
Figura 3: Componentes comunes en los sistemas DVB CA 8
Figura 4: Jerarquía de claves de tres niveles 10
Figura 5: SimulCrypt 12
Figura 6: Módulo de Interfaz Común para tarjetas de PC 13
Figura 7: Datos a través del Scrambler 15
Figura 8: El cifrador en flujo 16
Figura 9: Arquitectura de la tarjeta compartida 27
Figura 10: Ataque mediante tarjeta compartida 27
Figura 11: Árbol binario completo 35
Figura 12: Esquema Complete Subtree 36
Figura 13: Asignación de vector binario al nodo 20 37
Figura 14: Esquema Subset Difference 38
Figura 15: Steiner Tree 39
Figura 16: Nodos clave para el nodo raíz 1 41
Figura 17: Nodos clave para el nodo raíz 2 41
Figura 18: Nodos clave para el nodo raíz 5 41
Figura 19: Nodos clave para el nodo raíz 11 41
Figura 20: Niveles especiales y capas en un árbol binario de 16 usuarios 43
Figura 21: Reconstrucción de subconjuntos SD con subconjuntos LSD 44
Figura 22: Creación de los intervalos 48
Figura 23: Creación de los intervalos-C 50
Figura 24: intervalos-6 perforados-1 52
Figura 25: Cadena de claves de intervalos -20 perforados-10 53
Figura 26: Cadena de claves de único sentido desde $K_{1,1}$ con p=3, c=C=5 54
Figura 27: Cooperación smart card y receptor 68
Figura 28: Esquema comunicación receptor – smart card 71
Figura 29: TO de los sistemas para *n*=1024 81
Figura 30: Detalle del TO de los sistemas para *n*=1024 82

Figura 31: TO para diferentes n (10% de revocados) 83
Figura 32: Detalle del TO para diferentes n (10% de revocados) 84
Figura 33: TO del esquema C-basic Chain Scheme para diferentes C 84
Figura 34: TO del esquema Skipping Chain Scheme para diferentes C y p 86
Figura 35: CC de los sistemas para *n*=1024 87
Figura 36: CC para diferentes n (10% de revocados) 88
Figura 37: SS de los sistemas en función de n 89
Figura 38: Primer esquema propuesto 92
Figura 39: Flujo de información enviado por parte del centro emisor 95
Figura 41: Segundo esquema propuesto 96
Figura 42: Cifrado y descifrado de la información pregrabada 100
Figura 43: Función AES-G 101
Figura 44: Descifrado del contenido 103
Figura 45: Revocación de contenido de información pregrabada 104
Figura 46: Esquema de la función f del algoritmo DES 122
Figura 47: Calculo de las K_i para el algoritmo DES 122
Figura 48: Subclaves empleadas en el algoritmo IDEA 125
Figura 49: Modo CBC (A: Cifrado, B: Descifrado) 132
Figura 50: Complete Subtree TO (*n*=1024) 134
Figura 51: Subset Difference TO (*n*=1024) 134
Figura 52: Layered Subset Difference TO (*n*=1024) 135
Figura 53: Basic Chain Scheme TO (*n*=1024) 135
Figura 54: C-basic Chain Scheme TO (*n*=1024) 136
Figura 55: Skipping Chain Scheme TO (*n*=1024) 136
Figura 56: Media y cota superior teórica del TO del CS (*n*=1024) 137
Figura 57: Media y cota superior teórica del TO del SD (*n*=1024) 137
Figura 58: Media y cota superior teórica del TO del LSD (*n*=1024) 138
Figura 59: Media y cota superior teórica del TO del BCS (*n*=1024) 138
Figura 60: Media y cota superior teórica del TO del CBCS (*n*=1024) 139

Índice de tablas

Tabla 1: Generación de claves por parte de los usuarios de la Figura 14 42
Tabla 2: Cuadro resumen de los esquemas de BE 57
Tabla 3: Equivalencia comparativa de longitud de clave entre algoritmos (seguridad) 66
Tabla 4: Ejemplo de matriz de estado con Nb=5 (160 bits). 127
Tabla 5: Ejemplo de clave con Nk=4 (128 bits) 127
Tabla 6: Número de rondas para AES en función de los tamaños de clave y bloque 128
Tabla 7: Valores de ci según el tamaño de bloque de Nb 129

1 Introducción

1.1 Contexto

Existen múltiples plataformas de difusión de la televisión de pago. Estas plataformas pueden ser el satélite, el cable, la fibra óptica, la antena terrestre, etc. En todas ellas, desde ya hace un tiempo, la señal de televisión se emite en formato digital.

Una vez se comprobó que la señal de televisión digital ofrecía un mejor rendimiento en cuanto a ancho de banda consumido y calidad de la imagen, entre otros, todos los proveedores de televisión de pago optaron por migrar la tecnología al formato digital. Incluso ahora, la señal de televisión en abierto (pública) está siendo emitida en formato digital y el formato analógico está a punto de desaparecer.

La televisión analógica no permitía grandes medidas de codificación mientras que la televisión digital sí. Fue precisamente en el momento de transición del formato analógico al digital cuando se precisó de nuevos estándares para que los sistemas de codificación evolucionasen en una misma dirección. La entidad que surgió con el fin de cubrir estas necesidades fue el *Digital Video Broadcasting* (DVB).

Una vez establecidos los estándares, las compañías que se dedicaban a la protección de la señal de televisión analógica, desarrollaron cada una por su cuenta, nuevos sistemas de protección de la señal de televisión digital. Estos nuevos sistemas eran diferentes entre ellos, pero cumplían un esquema básico común. Este esquema es el definido por el DVB.

A los sistemas que se encargan de proteger la señal de televisión se les llama Sistemas de Acceso Condicional (CAS). Son los encargados de permitir al acceso al contenido de la señal a aquellos usuarios autorizados y de bloquear el acceso al contenido a aquellos usuarios que no lo estén. Para conseguir tal fin hacen uso de elementos criptográficos.

Pese a las grandes mejoras que permite el uso de la televisión digital a la hora de garantizar la seguridad de la señal, los Sistemas de Acceso Condicional no han parado de ser modificados e incluso rediseñados por culpa de las diferentes vulnerabilidades detectadas. La piratería dedicada grandes esfuerzos a explotar y descubrir dichas vulnerabilidades, por lo que ha hecho y hace mucho daño a la industria de la televisión de pago. Por este motivo, las compañías invierten mucho dinero en investigación y desarrollo de nuevos sistemas.

Desde el 1992, por parte de la rama de la investigación más académica, se viene trabajando en un nuevo concepto de esquema de difusión global de la información de forma

segura a través de medios inseguros. A este tipo de esquema se le llamó *Broadcast Encryption* (BE) y desde entonces han aparecido múltiples esquemas propuestos. Estos esquemas poseen grandes cualidades y algunos de ellos se adaptan especialmente bien a las necesidades de los sistemas de difusión de televisión de pago.

Un claro ejemplo de la madurez alcanzada por el *Broadcast Encryption* es que ya está siendo usado en una aplicación comercial, el *Advanced Access Content System* (AACS). El AACS es el sistema encargado de proteger el acceso a los contenidos de los discos ópticos de alta densidad, tales como los discos HD DVD y los discos Blu-ray. Una parte importante de la seguridad de este sistema reside en el esquema de *Broadcast Encryption*.

1.2 Propósitos y objetivos

En este proyecto se pretende realizar un estudio de los sistemas de difusión de televisión de pago que hay en funcionamiento actualmente y en la medida de lo posible, proponer un nuevo sistema que haga uso de los esquemas de *Broadcast Encryption* para aumentar tanto la seguridad como la eficiencia del sistema.

El proyecto se va a centrar únicamente en aquellos sistemas de televisión de pago en los cuales el canal es de único sentido, es decir, que el receptor no es capaz de mantener una comunicación bidireccional directa con el centro emisor en relación a la solicitud de claves, derechos de visión, alteración del hardware, etc.

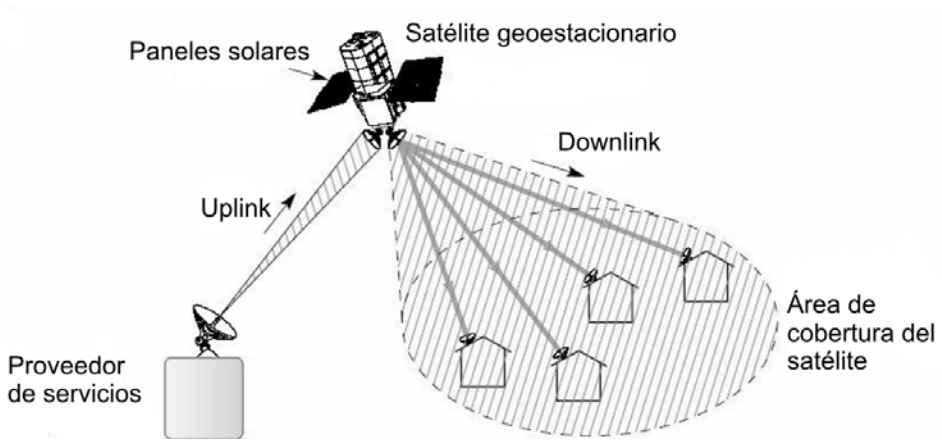

Figura 1: Esquema de la difusión de televisión vía satélite

Un claro ejemplo de estos sistemas es la televisión de pago vía satélite, en el que el único canal accesible por los usuarios es el canal Downlink del satélite. Es por este canal por donde se reciben las señales de televisión (imagen, sonido, mensajes del sistema y datos). Dado que no se va a mantener una comunicación bidireccional con el centro emisor, los receptores tienen que ser capaces de acceder a los contenidos de las señales recibidas mediante una "configuración inicial de fábrica".

Para poder proponer un nuevo sistema de difusión de televisión de pago, se va a estudiar el Sistema de Acceso Condicional tal y como el DVB lo contempla, así como el concepto del *Broadcast Encryption*, junto con algunos de los algoritmos de este tipo ya desarrollados. Dadas las ventajas que proporciona el uso de los algoritmos de *Broadcast Encryption* en la difusión de televisión de pago comparado con los servicios actuales de Acceso Condicional (CA), el sistema propuesto, alternativo al actual, poseerá como núcleo un esquema de *Broadcast Encryption*.

Por otro lado, también se va a llevar a cabo un análisis comparativo de los Sistemas de Acceso Condicional con los diferentes esquemas de *Broadcast Encryption* que se mencionan a lo largo del *Tema 3*. Este análisis proporcionará una idea bastante realista de las características de cada sistema. Por lo que se podrán extraer las fortalezas y las debilidades de cada uno de los sistemas frente a los otros.

1.3 Limitaciones

Actualmente existen un gran número de empresas que proporcionan servicios de Acceso Condicional para la televisión de pago. Como ya se ha mencionado, cada una de ellas ha desarrollado su propio Sistema de Acceso Condicional, por lo que poseen diferentes características a pesar estar basados en un estándar común.

Además de que cada Sistema de Acceso Condicional es distinto a los demás, se tiene que tener en cuenta que éstos son secretos. Resulta realmente complicado conocer con seguridad el esquema completo de uno de estos sistemas, ya que los detalles de éstos no están publicados. La poca información que se puede obtener de ellos es a través de estudios/descubrimientos hechos por usuarios de foros de Internet. No resulta fácil saber si uno de estos descubrimientos publicado en los foros es cierto o falso, pero lo que si se puede decir es que dan origen a un montón de ideas por parte de otros usuarios de los foros. Después

de ver esto uno se da cuenta que existe un gran número de personas, con un gran nivel de formación, que "trabaja" para desentramar los misterios detrás de este secretismo.

Es por culpa de este secretismo por el que no se ha podido entrar en detalles de los diferentes sistemas usados. Originalmente, se pretendía realizar un análisis de la situación actual de los Sistemas de Acceso Condicional a partir de los distintos esquemas y elementos criptográficos utilizados, para posteriormente, comentar las mejoras que podría introducir el uso del *Broadcast Encryption* en estos sistemas. No obstante, al no ser pública esta información, se ha tenido que realizar el análisis del modelo general y para cosas más detalladas se ha usado información de la segunda versión del Sistema de Acceso Condicional usado por Digital+, el llamado Nagravision 2. Se trata de un Sistema de Acceso Condicional que ya no está operativo en España, pero que fue el objetivo de muchos ataques realizados, en parte gracias al conocimiento de muchos detalles del sistema.

1.4 Estructura de la memoria

La memoria del proyecto está estructurada en cuatro grandes bloques en función de la temática tratada. Cada uno de estos bloques hace referencia a un tema de la memoria (*Temas 2, 3, 4* y *5*). A continuación se puede leer un breve resumen del contenido de cada uno de ellos.

Tema 2

Se pretende dar una visión general de los CAS. Se proporciona información acerca de los elementos en común de éstos, así como también se mencionan a algunos de los proveedores más relevantes actualmente. También se hace una reflexión de como se encuentra la piratería así como los ataques más comunes a los CAS y herramientas que permiten que estos ataques sean fructíferos.

Tema 3

En este tema se hace una introducción al tema del *Broadcast Encryption*. Posteriormente se explican una serie de algoritmos diferenciándolos entre algoritmos de *Broadcast Encryption* que poseen estructura de árbol y los que poseen estructura lineal. Para finalizar se realiza una comparación de todos los algoritmos explicados en el tema.

Tema 4

Se realiza un análisis de la seguridad de los CAS a modo general. Para ello se hace una pequeña introducción a los sistemas de seguridad usados así como los mecanismos de seguridad empleados. También se comentan algunos de los algoritmos criptográficos más usados por los CAS para la difusión de las claves cifradas, junto a sus ventajas y desventajas frente a los otros. Dado que en la seguridad del sistema también interviene la comunicación entre el receptor y la *smart card*, en este tema se proporciona un apartado en el que se propone un esquema de comunicación entre estas partes. No se trata del esquema usado mayoritariamente, pero plantea unas mejoras interesantes dadas las especificaciones del sistema.

Para dar una visión más real de lo que está sucediendo en cuanto a la seguridad de los CAS, se proporciona información de cómo se pudo romper el Sistema de Acceso Condicional Nagra2 (Nagravision 2) a partir de las debilidades detectadas. Con este ejemplo se pretende demostrar que a pesar de que los algoritmos criptográficos utilizados por los CAS son seguros, si estos no se utilizan bajo las especificaciones que garantizan su seguridad, como por ejemplo la longitud de las claves, éstos no cumplirán con su propósito.

En este tema también se pretende proporcionar los motivos que puedan propiciar una migración global del Sistema de Acceso Condicional a un esquema de *Broadcast Encryption*, o como mínimo proporcionar algunas ideas de cómo introducir el *Broadcast Encryption* en los Sistemas de Acceso Condicional actuales. Para ello, se ha desarrollado un programa que simula los esquemas de *Broadcast Encryption* comentados en el proyecto. Este programa de simulación calcula los valores de los parámetros *Transmission Overhead* (*TO*), *Storage Space* (*SS*) y *Computacional Cost* (*CC*) para cada uno de los esquemas y para un CAS genérico. También se proponen dos esquemas de difusión de televisión de pago que utilizan el *Broadcast Encryption*.

Tema 5

Se trata un tema que se sale un poco de la línea principal del proyecto. No obstante, en este tema se presenta una aplicación llevada a la práctica en la que el uso de un algoritmo de *Broadcast Encryption* desempeña un papel muy importante en lo referente a la protección de contenidos. Esta aplicación demuestra que los algoritmos de *Broadcast Encryption* ya están lo suficientemente maduros como para introducirse en aplicaciones comerciales.

2 Sistemas de Acceso Condicional (CAS)

El Sistema de Acceso Condicional o CAS, es un sistema que previene o como mínimo limita el acceso incontrolado a los contenidos de la televisión de pago codificando las señales. Sólo permiten el acceso al contenido a aquellos usuarios que posean cierta información especial, generalmente una clave de descodificación.

Fue en Estados Unidos en 1983 donde se implementó el primer sistema de Acceso Condicional, el VIDEOCYPHER. Posteriormente los europeos adaptaron este sistema a su sistema de codificación de TV analógico D-MAC (Multiplexado de componentes analógicos) con el nombre de EUROCRYPT. Ante la lentitud de la implementación del sistema MAC, debido sobretodo a la duda de si este sistema se iba a convertir en estándar, algunos radiodifusores optaron por implementar sistemas de Acceso Condicional en las transmisiones analógicas en PAL.

Más adelante, en 1993 se fundó el *Digital Video Broadcasting* (DVB) y éste desarrolló las especificaciones del Sistema de Acceso Condicional para la televisión de pago vía satélite, entre otros [76]. La televisión digital propició la aparición de una serie de servicios adicionales, como por ejemplo el *pay-per-view* (PPV). En el PPV el abonado paga por producto consumido, una película, un documental de historia, un partido de fútbol, etc.

A pesar de que exista un estándar para los CAS [36], cada proveedor de Sistema de Acceso Condicional tiene desarrollado un sistema propio, por lo que los operadores de televisión y los usuarios finales están bloqueados por estos Sistemas de Acceso Condicional propietarios. Los vendedores de sistemas de Acceso Condicional también controlan a los fabricantes de receptores con su tecnología propietaria. De este modo, aquellos suscriptores que deseen conseguir nuevos servicios tienen que comprarse un nuevo receptor que esté preparado para soportar dichos servicios.

La televisión digital puede ser distribuida mediante satélites, cable e infraestructuras terrestres. Otro medio para la difusión de televisión digital que está aumentando en popularidad últimamente es a través de las redes basadas en IP.

Para la difusión vía satélite, en Europa se utilizan los estándares DVB-S y DVB-S2 [74] proporcionados por la organización DVB (*Digital Video Broadcasting*). Mientras que para el caso de la difusión terrestre se utiliza el estándar DVB-T y para el de la televisión por cable el DVB-C.

Figura 2: Logos de algunos de los estándares del DVB

El Sistema de Acceso Condicional juega un papel muy importante en la distribución de la televisión de pago. Como ya se ha comentado, el propósito principal de un Sistema de Acceso Condicional es el de proporcionar un mecanismo de seguridad que permita acceder a un cierto programa basándose en el pago del contenido consumido. Se utiliza la codificación y el cifrado para prevenir el acceso de usuarios no autorizados a contenidos y servicios protegidos. La codificación se utiliza para el cifrado y protección de señales de programa con ciertos algoritmos, como el *Common Scrambling Algorithm* (CSA), desarrollado por el grupo DVB [76]. El cifrado se utiliza para la protección de la clave de codificación, llamada palabra de control (*Control Word*, CW).

2.1 Arquitectura común de los CAS

El DVB define una estructura común para todos los CAS que persigan la compatibilidad DVB [36]. En esta estructura el CAS está formado por cinco componentes importantes: el sistema de administración de suscriptor (*Subscriber Management System* o SMS), el sistema de autorización del suscriptor (*Subscriber Authorization System* o SAS), el codificador (*Scrambler*), el descodificador (*Descrambler*) y la *smart card* (tarjeta inteligente). Las tres primeras están localizadas en el servidor y las otras dos en el receptor. En la *Figura 3* se detalla donde se ubican estos cinco componentes dentro de la estructura y más adelante se proporciona una pequeña explicación de cual es el papel que desempeñan cada uno de ellos.

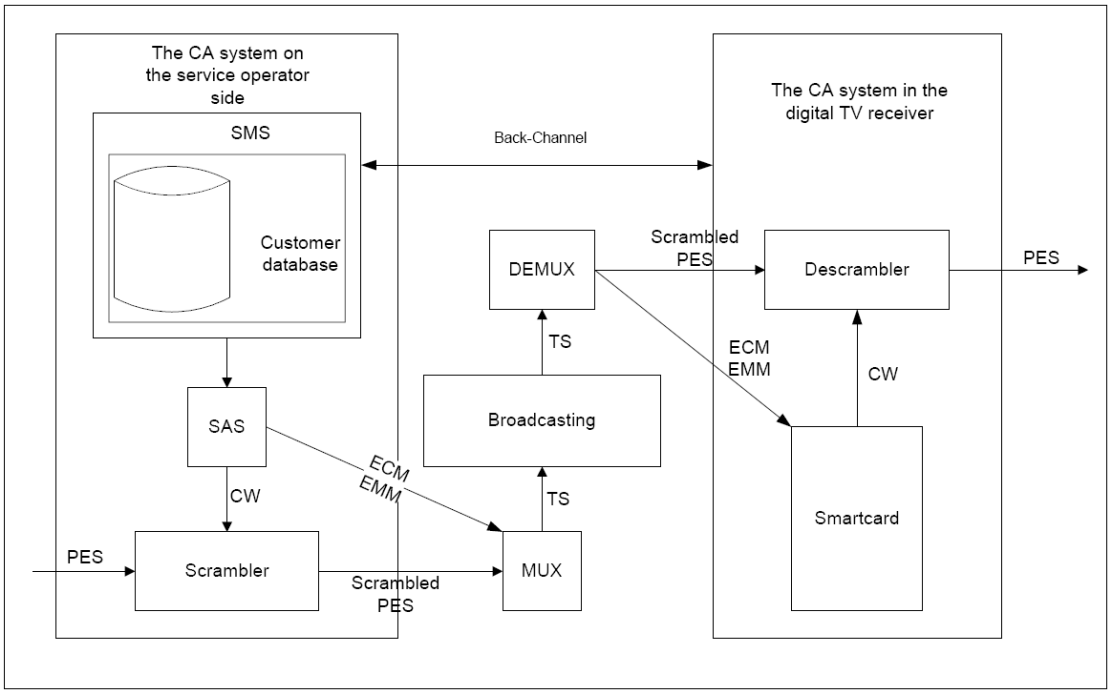

Figura 3: Componentes comunes en los sistemas DVB CA

Subscriber Management System

Según el EBU, el SMS es un centro de negocio donde se tienen en cuenta las *smart cards* y los pagos de los suscriptores. El SMS contiene cierta información importante como ahora el número de serie de los receptores y los servicios a los que está suscrito el cliente.

Subscriber Authorization System

El SAS es responsable de crear, organizar y proporcionar el flujo de datos de control, *Entitlement Management Message* (EMM) y el *Entitlement Control Message* (ECM) de acuerdo con el SMS. Más adelante se proporcionará más información acerca del EMM y del ECM.

Codificador

Este dispositivo utiliza un potente algoritmo para que el contenido sea ilegible, al contenido también se le conoce como *Packetized Elementary Stream* (PES). En los sistemas DVB el codificador implementa el CSA (*Common Scrambling Algorithm*).

Descodificador

Este dispositivo utiliza un potente algoritmo para que el PES vuelva a ser legible. En los sistemas DVB el descodificador implementa el *Common Descrambler Algorithm*.

Smart card

La *smart card* se utiliza para descifrar señales difundidas. El SMS valida las *smart cards* preprogramándolas, antes de entregárselas a los usuarios, con claves para autorizar el acceso al contenido de ciertos programas o servicios.

Codificación y descodificación

El PES es codificado mediante un algoritmo lo suficientemente potente para proporcionar seguridad durante un cierto periodo de tiempo. El codificador puede utilizar la CW como la clave de cifrado, o bien, como la semilla de inicialización de un generador pseudoaleatorio.

2.1.1 Jerarquía de claves

En general, los CAS tienen una jerarquía de claves de tres niveles. Esta jerarquía de claves de tres niveles fue propuesta en unas recomendaciones de la ITU en 1992. La variable aleatoria CW (*Control Word*), es la clave que se encuentra en la parte baja de la jerarquía. La clave CW se utiliza como semilla de un generador pseudoaleatorio (PRG) para generar una secuencia pseudoaleatoria o bien como clave en si, la cual será utilizada para codificar el PES. Dado que los sistemas de televisión de pago utilizan cifrado simétrico, la CW se utiliza tanto para codificar como para descodificar. Por motivos de seguridad, la CW debe ser actualizada cada 5-20 segundos.

En el nivel intermedio de la jerarquía de claves está la clave de autorización (*Authorization Key* o AK). Sirve para cifrar la CW y luego generar el ECM (*Entitlement Control Message*). La AK es única para cada canal y a veces se la llama clave de servicio por parte de los vendedores de CA. La AK se actualiza cada cierto periodo de tiempo, el cual depende de la política del operador. Este periodo de tiempo puede variar entre semanas y meses.

Cada vez que se actualiza la CW, sólo es necesario que se difunda un único ECM en el que se encuentra esta nueva CW. Este procedimiento conlleva una utilización del ancho de banda muy reducida, ya que de este modo, un único ECM sirve para todos los usuarios y no es necesario enviar un ECM distinto para cada uno de los usuarios del sistema.

En la parte alta de la jerarquía de claves está la *Master Private Key* (MPK) que se utiliza para cifrar la AK junto con otra información que otorga ciertos derechos al usuario a quien va destinado. El EMM se genera a parir de la clave cifrada y de esta información. La MPK es única para cada suscriptor y está almacenada en la *smart card*. Por este motivo, los EMM van destinados a usuarios individuales y serán necesarios enviar tantos EMM como usuarios autorizados hayan en el sistema. Como consecuencia, el cambio de la AK supondrá un alto consumo del ancho de banda.

Como se puede comprobar en la *Figura 4*, el EMM, ECM y el TS (*Transport Stream*) codificado serán multiplexados y posteriormente enviados al suscriptor a través de los canales de difusión. El receptor, mediante la *smart card*, descifrará el EMM y a continuación el ECM. Una vez la *smart card* haya descifrado el ECM, el receptor obtendrá la CW y éste será capaz de descodificar el TS codificado para así obtener el TS original.

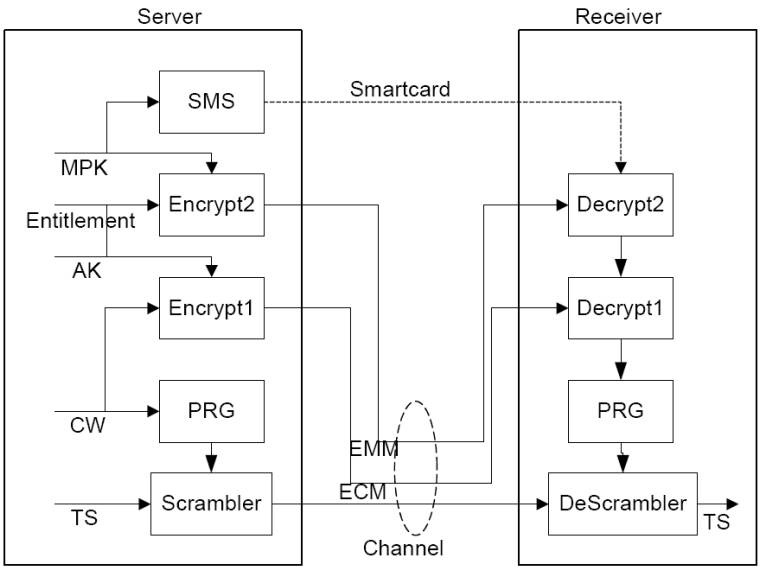

Figura 4: Jerarquía de claves de tres niveles

2.1.2 Distribución de claves

Para controlar el acceso a las CW el estándar DVB define dos tipos de mensajes, los ya mencionados EMM y ECM [36]. No obstante, el contenido de estos mensajes no queda especificado en el estándar, por lo que cada proveedor de Sistemas de Acceso Condicional define estos mensajes de diferente forma.

El ECM es un mensaje global, ya que se trata de un mensaje idéntico para todos los clientes de un mismo CAS y mismo canal. La autenticidad del ECM y la confidencialidad de la CW contenida en él suelen estar protegidas. El generador de ECM necesita una CW por entrada. El ECM resultante tendrá una longitud máxima de 256 bytes y se incluirá en el TS (*Transport Stream*). Para permitir un acceso rápido al servicio, un ECM es inyectado en este flujo elemental alrededor de 10 veces por segundo. Es fácil comprobar que el ancho de banda requerido para el ECM es de 20 kbps, un valor pequeño al lado de los varios Mbps necesarios para la transmisión del video codificado de un único canal.

El EMM se utiliza para proporcionarle derechos a un cliente del sistema. Estos derechos permiten al cliente el accedeso a una parte específica del contenido. La diferencia fundamental entre el ECM y el EMM es que el EMM está destinado a un único cliente o grupo fijo de clientes del sistema. Por razones de procesado por parte de las *smart cards*, el tamaño de los EMM también está restringido a 256 bytes. Después de que un consumidor ordene un servicio de pago a su proveedor de televisión, un generador de EMM introduce los derechos y claves pertinentes en un EMM. Posteriormente este EMM será inyectado en el TS.

El tráfico de los EMM será sensiblemente grande para sistemas con muchos suscriptores. En la jerarquía de claves de tres niveles, la AK contenida en el EMM se debe cifrar con la MPK de cada uno de los suscriptores cada vez que ésta se actualice. Suponiendo que hay S suscriptores y T canales, habrán aproximadamente S·T paquetes para ser cifrados y transmitidos por parte del centro emisor.

2.1.3 Interoperabilidad CA

El proyecto DVB ha respaldado dos estándares diferenciados para los Sistemas de Acceso Condicional [36]. El nombre de estos dos estándares es: *SimulCrypt* y *MultiCrypt*. Estos dos estándares son completamente válidos para ser usados tanto en las transmisiones vía satélite como también, para las transmisiones por cable y transmisiones terrestres.

2.1.3.1 SimulCrypt

SimulCrypt permite trabajar con múltiples mensajes que son generados por diferentes CAS. En [44] se encuentra descrito el estándar de forma detallo. El contenido MPEG (*Motion*

Picture Experts Group) se codifica mediante el CSA a partir de la CW. Cada sistema CA genera sus propios mensajes de control (EMM y ECM) cumpliendo con el estándar del DVB. De este modo, el contenido codificado y los mensajes de cada sistema se multiplexan en un mismo MPEG-TS y se difunden a los receptores. El receptor filtra el mensaje necesario para poder recuperar la clave y posteriormente descodificar el TS.

La filosofía del sistema es que en una misma área geográfica sólo sea necesario tener una *smart card*, un único módulo CA y un único descodificador para recibir un servicio local.

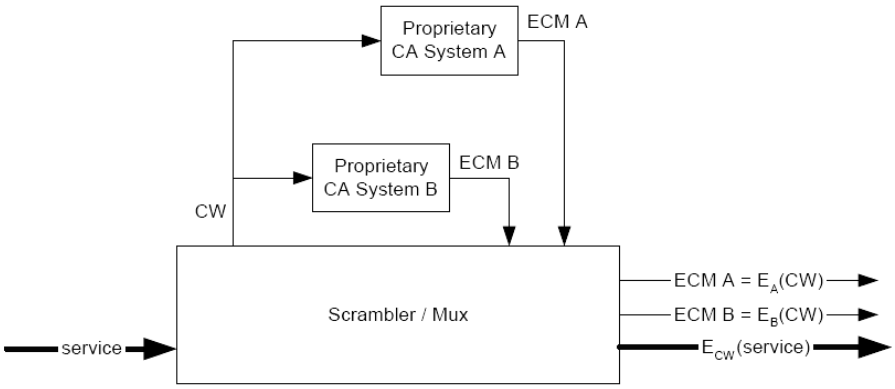

Figura 5: SimulCrypt

2.1.3.2 MultiCrypt

MultiCrypt es un sistema abierto que permite que un mismo receptor funcione con múltiples CAS mediante un Interfaz Común (CI) definido por el DVB. Este funcionamiento posibilita la competencia entre proveedores de CAS y operadores de televisión de pago. El interfaz común es un interfaz estandarizado para conectar al receptor un módulo extraíble, generalmente un módulo PCMCIA. El módulo PCMCIA implementa las funciones específicas para un CAS particular. Para poder visualizar un programa codificado por un CAS que no esté instalado en el receptor, el usuario deberá intercambiar los módulos manualmente. Existen algunos receptores que permiten tener varias PCMCIA al mismo tiempo y de este modo el usuario se ahorra el tener que irlas cambiando.

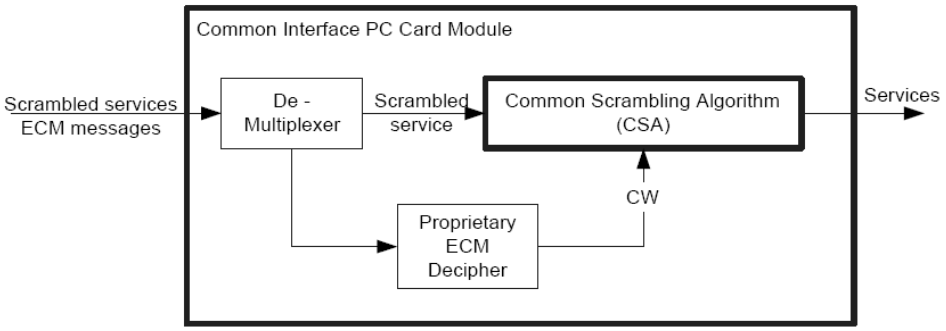

Figura 6: Módulo de Interfaz Común para tarjetas de PC

2.1.3.3 Comparación

Con *SimulCrypt* se requieren acuerdos comerciales entre diferentes proveedores de servicio para conseguir la interoperabilidad entre diferentes CAS. Esto crea una barrera a las nuevas compañías que pretenden entrar en el mercado de la televisión de pago. Otra desventaja de *SimulCrypt* es el incremento en el uso del ancho de banda, ya que los mensajes CA se multiplicarán, por lo que se creará un problema de escalabilidad. Por otro lado, la seguridad de todo el sistema *SimulCrypt* será igual de fuerte que el más débil de los CAS. La ventaja principal de *SimulCrypt* es que no hay un coste extra para los usuarios finales. Mientras que la principal desventaja de *MultiCrypt*, es precisamente el coste extra para los usuarios finales, ya que los equipos son más caros y se requiere módulos extras.

2.2 DVB

El proyecto *Digital Video Broadcasting* (DVB) es una iniciativa europea que ha dado lugar a un número significativo de los requisitos comerciales, las especificaciones técnicas y directrices para el desarrollo y el despliegue de la televisión digital.

El proyecto DVB ha sido muy productivo desde sus inicios en 1993. El DVB ha llevado a cabo el desarrollo de un número significativo de las especificaciones técnicas y directrices dentro de diversos campos como por ejemplo en el de la transmisión, el del multiplexado, el de la interconectividad, el del Acceso Condicional, etc. [76].

2.2.1 Common Scrambling Algorithm (CSA)

El CSA es un estándar desarrollado por DVB [76]. Este algoritmo se utiliza en la mayoría de CAS alrededor del mundo. Se utiliza para codificar el flujo MPEG y hacerlo ininteligible si no se posee la semilla de 8 bytes (CW), necesaria para la codificación.

El CSA es capaz de prevenir de ataques piratas por un cierto periodo de tiempo, habitualmente este tiempo de vida es el esperado para un CAS. Los detalles de su implementación sólo estaban disponibles para los fabricantes de equipos y siempre bajo un acuerdo con el ETSI de no revelación de contenidos. No obstante, desde el 2002, numerosos detalles del algoritmo salieron a la luz, como ahora las permutaciones, las cajas-S (*S-boxes*), algunos cálculos de bits, etc. En [46] y [55] se pueden observar algunos de los detalles de este algoritmo.

El Algoritmo de Codificación (*Scrambling Algorithm*) puede operar como un cifrador en bloque o en flujo de cascada, para los cuales se usa la misma clave CW de 64 bits. El flujo de entrada se divide en bloques de 8 bytes. En el caso que la longitud del último bloque sea menor que 8 bytes, a este bloque se le llamará Residuo.

Como se puede observar en la *Figura 7*, la secuencia de 8 bytes se cifra en modo *Chiper Block Chaining* (CBC) de orden inverso. El número inicial aleatorio llamado Vector de Inicialización (IV) se usa para generar el último bloque de 8 bytes (IB_{n-1}) y el bloque de texto cifrado (IB_{n-1}) generado se usa como número aleatorio del penúltimo bloque de 8 bytes (DB_{n-2}), etc. Se puede observar como el residuo (R) no se toca durante el cifrado CBC de orden inverso. La última salida de la cadena de bloques IB_0 se utiliza para la inicialización del cifrador en flujo. Las salidas de los cifradores en flujo (de CB_1 a CB_{n-1}) se usan para realizar un XOR con los bloques cifrados (IB_i, donde $1 \leq i \leq n-1$).

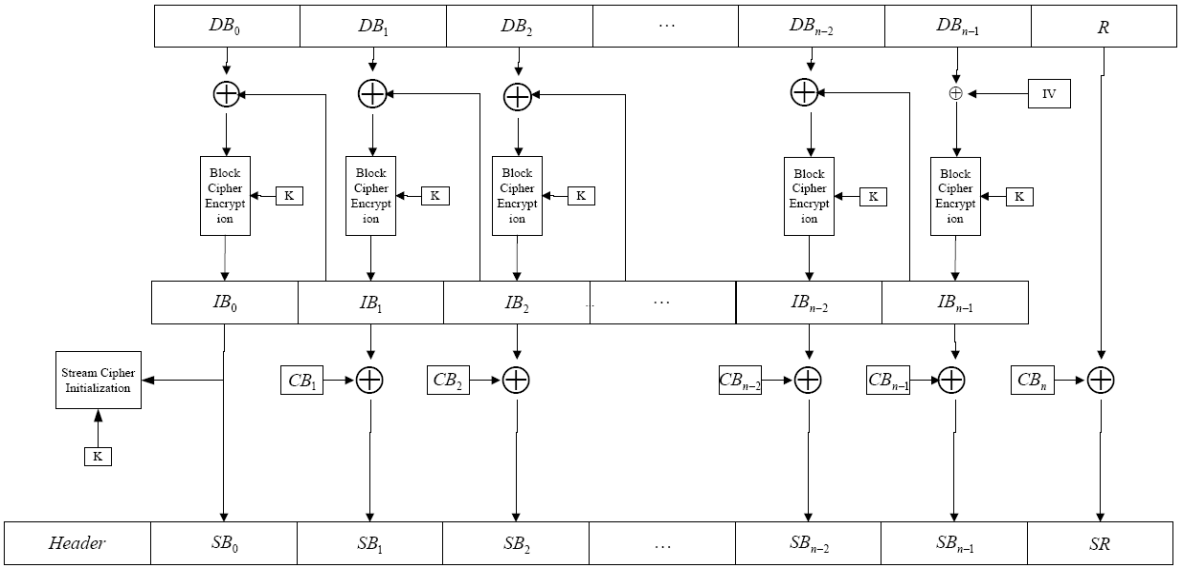

Figura 7: Datos a través del Scrambler

A continuación se proporciona cierta información para poder comprender la *Figura 7*.

- K: Clave de 64 bits. Utilizada tanto para el cifrado en bloque y en flujo.
- SB_i: Bloque i-ésimo de 8 bytes del flujo de salida "codificado" (*Scrambled Block*).
- CB_i: Bloque i-ésimo de 8 bytes del cifrado en flujo de salida.
- IB_i: Bloque i-ésimo intermedio de 8 bytes.
- DB_i: Bloque i-ésimo de 8 bytes de los datos de entrada.
- *R*: Residuo.
- *SR*: Residuo scrambleado.
- IV: Vector de Inicialización usado en el cifrado en bloque.

2.2.1.1 Cifrador en flujo

El cifrador en flujo está formado por dos registros de desplazamiento retroalimentados (*Feedback-Shift-Register*, FSR) y un combinador con memoria. En la *Figura 8* se puede observar el esquema del cifrador en flujo. Los registros p, q y c son registros de bit, todos los otros registros son de 4 bits.

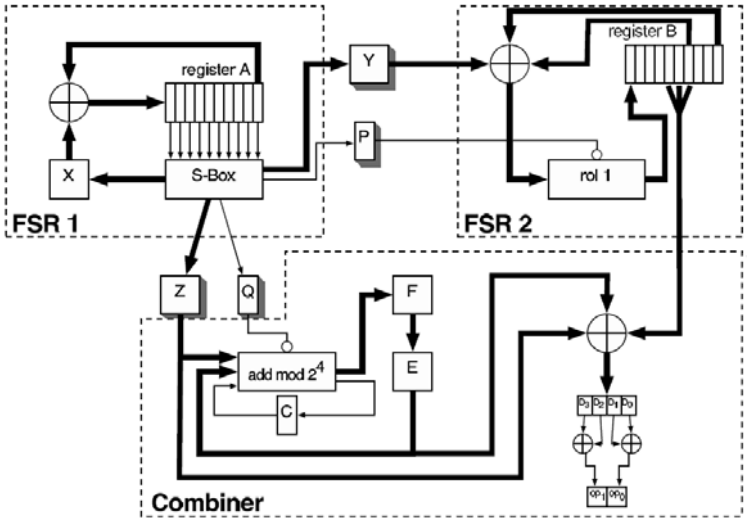

Figura 8: El cifrador en flujo

El cifrador en flujo puede operar en dos modos posibles. El primero de estos es el modo de inicialización, en el cual el estado inicial del cifrador está cargado. El segundo es el modo generador, en el cual el cifrador produce dos bits por ciclo de reloj.

2.2.1.2 Cifrador en bloque

El CSA emplea un cifrador en bloque iterativo que trabaja con bloques de información de 64 bits y con una clave de 64 bits (K). En cada ronda del cifrador se toma un vector de 8 bytes y una entrada de un byte de la clave expandida, como salida se obtiene un vector de 8 bytes. Esta ronda de transformación se aplica 56 veces.

2.3 Algunos sistemas CA propietarios

2.3.1 VideoGuard

VideoGuard es un sistema de acceso condicional (CAS) desarrollado por NDS. Se utiliza prácticamente de forma exclusiva para televisión digital vía satélite (DVB-S). Existen diferentes variantes de la protección dependiendo de las necesidades del proveedor. Se trata de uno de los sistemas más extendidos por todo el mundo y sus implementaciones más usadas

están en DirecTV y BSkyB's Sky Digital, que adoptó este sistema en 1998 para reemplazar al sistema *crackeado* VideoCrypt.

Se utiliza tanto para paquetes estándar de suscripción como para películas y eventos de pago por visión. Los permisos de visión se pueden descargar a la *smart card* a través del "aire" o bien mediante un posible MODEM en el receptor.

Es imposible conseguir un módulo de acceso condicional de VideoGuard con licencia para receptores de terceras partes, o conseguir rutinas de VideoGuard para software del módulo de acceso condicional. A diferencia de este, los CAMs y el software para estos están disponibles para los otros sistemas de acceso condicional (Irdeto, Nagravision y Viaccess).

A pesar de ello, en el 2005 se consiguió, mediante ingeniería inversa, un CAM actualizable que permitía ver los canales autorizados a una tarjeta legítima. Un emulador del software de CAM para la DreamBox (decodificador de satélite basado en Linux) puede hacer lo mismo. Sin embargo, muchas *smart cards* están ligadas a un número de serie específico de receptor.

Algunas de las emisoras que utilizan este sistema son las ya mencionadas DirecTV (en Estados Unidos) y BSkyB's Sky Digital (en Inglaterra e Irlanda), además de Hot (Israel), Yes (Israel), Viasat (Escandinavia), Sky Italia, Sky Network Television (Nueva Zelanda), Foxtel (Australia), Tata Sky & Hathway Cable(dvb-c) (India), D Smart (Turquía), TotalTV (Serbia), ONO (España), Canadian Broadcasting Corporation (Canadá) and Telewizja Polska S.A. (Polonia).

2.3.2 Nagravision

Nagravision, también conocido como Nagra, es el sistema de codificación de señal MPEG basada en la aritmética modular que utilizan plataformas de televisión vía satélite de pago tanto digitales como analógicas como Digital+ (España), Premiere (Alemania), Polsat (Polonia) o TV Cabo (Portugal), entre otras muchas.

Nagravision es un proyecto de la empresa suiza de seguridad informática Kudelski Corporation, especializada en cifrado y descifrado de señal MPEG en acceso condicional a ella, base principal de todos los proveedores de televisión por satélite. El proyecto Nagravisión ha tenido varias versiones y numerosos parches de seguridad; conocidas como Nagra 1, Nagra 2 (modificación de Nagra 1) y Nagra 3 (como se conoce popularmente) han permitido a la empresa superar los inconvenientes de la piratería, que ha dejado en evidencia a la compañía en numerosas ocasiones del pasado.

Nagravision 1 era utilizado por Canal+ terrestre, canal que fue reemplazado por Cuatro desde el pasado noviembre de 2005. En los primeros años de emisión de la plataforma Canal Satélite Digital se utilizó el sistema Nagravisión 2. Tanto los sistemas Nagravisión 1 y Nagravisión 2 tenían numerosos fallos de seguridad graves, que permitieron que los sistemas fuesen pirateados y visionados ilegalmente. Debido a esto TV Cabo (Portugal) decidió adaptar sus sistemas a Nagra 3 a finales de 2007, solucionando hasta la fecha el problema del visionado ilegal, así como Digital + (empresa fusión de Vía Digital y Canal Satélite Digital) que desde el 4 de diciembre 2007 usa el sistema nuevo Nagravisión 3. Actualmente existen proveedores como Premiere (Alemania) que continúan utilizando Nagra 2, aunque ya están probando sus equipos con Nagra 3, en vista a los exitosos resultados obtenidos por sus colegas europeos.

La aritmética modular es la base de todo el sistema Nagra y el otro pilar, es el uso de tarjetas (*smart card*) para el acceso condicional a la señal, punto clave del sistema y donde la piratería centra todos sus esfuerzos, por ser el centro neurálgico de todo el sistema.

Las tarjetas tienen una memoria EEPROM y un chip microcontrolador que permite el control de los datos. En Nagra la EEPROM se divide en dos zonas una de datos y otra de código y en esta última el proveedor puede grabar código ejecutable por la tarjeta para corregir errores.

Para explotar los fallos de seguridad del sistema Nagra 1, se utilizaron unas tarjetas emuladoras. Situándolas en el lugar de las originales se conseguía controlar la decodificación condicional, mediante el uso de las excepciones de software que proporciona el sistema Nagra, haciendo posible ejecutar cualquier instrucción en el decodificador original de la empresa. Las claves se introducían en los chips de las tarjetas consiguiendo visualizar la señal.

Durante la decadencia del sistema Nagra 1, se encargó a Kudelski Corporation el desarrollo de un sistema nuevo para mejorar la seguridad del acceso condicional, pero al mismo tiempo también se exigió la compatibilidad de los equipos anteriores; al principio dio lugar a una época durante la cual Nagra 2 parecía impenetrable, aunque luego se comprobó que en realidad era una modificación del sistema Nagra 1, en vez de un sistema nuevo.

Nagravisión 2 consiste en un aumento de la seguridad del sistema anterior y la modificación de los errores que tenia Nagra 1, en principio todo era positivo, pero la compatibilidad exigida no permitió cambiar el concepto global de la arquitectura de trabajo. De esta forma los hackers situaron sus estudios en la misma dirección donde dejaron su anterior investigación y los resultados fueron más que sorprendentes.

Existen muy pocos datos sobre este nuevo sistema, Nagravision 3, propuesto por la compañía Kudelski, ya que aun nadie ha sido capaz de penetrarlo. No obstante, parece ser que hay algunos datos claros, Nagra 3 utiliza un sistema de claves de longitud doble que Nagra 2, por lo que estaríamos hablando de treinta y dos pares de dígitos hexadecimales, lo que supone un cifrado de un nivel extremadamente alto, siendo este un dato a favor de la seguridad del sistema pero también en contra, porque aceptar esta suposición indicaría que el sistema comete el mismo error que Nagra 2 y se basa en el sistema anterior.

2.3.3 Irdeto Access

Irdeto proporciona soluciones para la protección de contenidos digitales en televisión digital, IPTV y entornos móviles. La compañía posee dos sedes, una en Hoofddorp cerca de Ámsterdam y en Pekín. Irdeto proporciona tanto sistemas de acceso condicional basados en *smart card* como basados en software para operadoras de televisión de pago por cable, satélite, terrestre e IPTV.

Irdeto posee clientes en todos los continentes excepto América del Sur, siendo en Europa y en Asia donde poseen su mayor volumen de clientes. Algunos de estos clientes son: Austar (Australia), Telia (Suecia), Isat BV (Holanda), iN-Di (Bélgica), ORF (Austria), Showtime Arabia, Al Jazeera Sports, DStv (Sudáfrica), Globecast (Norteamérica), Guangdong Cable TV (China), Shanghai Interactive TV (China) SkyWave TV (China) entre muchos otros.

2.3.4 Conax

Conax es una companyía que se dedica a diseñar Sistemas de Acceso Condicional para la televisión digital y su sede está en Oslo, Noruega. Además de ésta, posee subsedes en Estados Unidos, la India y Alemania y oficinas de soporte y ventas en Rusia, Singapur, China, Corea del Sur, Brasil y Canadá.

Conax, mediante una solución no propietaria y abierta, permite la posibilidad de libertad de elección de la tecnología, esto se consigue mediante una política de licencias completamente abierta, justa y no discriminativa para los fabricantes de receptores a la hora de incluir soporte para el CAS Conax en los receptores. También debe cumplir el formato de transmisión del DVB y el de OpenCable, así como el del DVB SimulCrypt tanto en el lado de emisión como el de recepción (STB).

Por otro lado, usa un módulo de seguridad extraíble que cumple la norma ISO 7816 compatible con la *smart card.* También posee un suministro de módulos de interfaz común de acceso condicional y CableCards.

2.3.5 Viaccess

Viaccess es un sistema de cifrado para la televisión digital, desarrollado por France Telecom. Existen hasta 5 versiones en funcionamiento hoy en día. Fue desarrollado como la versión digital del sistema EuroCrypt, utilizado con el sistema híbrido de MAC.

A menudo se refiere a la primera versión como Viaccess 1 y a las tres posteriores, aunque diferentes, como Viaccess 2. PC2.3 y PC2.4 son conocidas por no ser efectivas, muchos receptores pueden ser "parcheados" para descifrar las señales Viaccess sin pagar, no obstante, PC2.5 y PC2.6 son seguras. Fueron introducidas en el 2004 y a finales del 2005 respectivamente. PC3.0 fue introducido a mediados del 2007.

Existen dos modificaciones del Viaccess PC2.3. La primera de ellas, se conoce como TPS Crypt y a pesar de estar comprometida, ha sido modificada para utilizar claves AES (Advanced Encryption System). Estas claves AES al principio se actualizaban una vez al día, sin embargo, por culpa de las visualizaciones no autorizadas se introdujo un segundo sistema TPS Crypt. En este segundo sistema las claves cambiaban cada 12 minutos. Estas claves eran transmitidas a través del sistema de televisión abierta TPS y por este motivo sólo los receptores TPS podían recibir las claves AES. Mediante una monitorización y análisis de las claves por parte de grupos de hackers se ha conseguido generar una lista de claves donde la predicción de las claves AES ha sido satisfactoria.

La segunda modificación de Viaccess se llama ThalesCrypt y es utilizado por Canal Satellite France para proteger los contenidos.

Algunos de los proveedores que utilizan Viaccess son los siguientes: TPS (Francia), Canal Satellite France (Francia), Boxer TV Access (Suecia) y RTV (Eslovenia) entre otros.

2.4 Piratería

La piratería, que viola los derechos de autor y las leyes de la propiedad, es una de las principales preocupaciones de hoy en día en entornos altamente técnicos. Como consecuencia, ha aparecido un interés para el desarrollo de contramedidas técnicas eficaces para proteger este tipo de violaciones. En la mayoría de los escenarios de radiodifusión, un proveedor de datos proporciona claves a los usuarios privilegiados. Sin embargo, estos usuarios privilegiados pueden hacer estas claves públicas para cualquiera. En este escenario, a estos usuarios se les llama piratas, o también traidores. Los piratas hacen negocio rompiendo las medidas de seguridad del sistema condicional y venden dispositivos que permiten visionar los contenidos a usuarios no autorizados.

2.4.1 Mercado de la piratería de la televisión de pago en la UE

La piratería de la televisión de pago afecta significativamente a varios proveedores que proporcionan televisión de pago de forma legal. En el 2001, según [50], el ratio de usuarios que pirateaban la televisión de pago era el 30% de los suscriptores legales, en la Unión Europea. Dado que los suscriptores legales entonces ascendían a 33 millones, eso quiere decir que había alrededor de 10 millones de tarjetas piratas.

El mercado de la piratería de la televisión de pago se caracteriza por la gran demanda de productos, grandes márgenes de beneficio en los productos de piratería y "pérdida" del miedo a las consecuencias legales por parte de los fabricantes piratas y los consumidores.

Los esquemas de CAS se vulneran principalmente explotando los errores de implementación en las *smart cards*, errores en los módulos CA y los débiles protocolos de seguridad de los CAS. Además, numerosas páginas Web y comunidades virtuales describen con detalle como descodificar los esquemas de codificación CAS y obtener las claves de descodificación. Los programas tanto para hardware como para software, así como *smart cards* en blanco y dispositivos de programación se pueden conseguir de forma legal.

2.4.2 Atractivo del Mercado de la piratería de la televisión de pago

Los grandes márgenes de beneficio atraen a los piratas de la televisión de pago. Los piratas habilidosos facturan a los espectadores mensualmente alrededor de 16 € (datos extraídos de [41]), proporcionándoles las actualizaciones de las claves de acceso.

Comparando las facturas legales con los costes de producción de tarjetas piratas, se estima que los fabricantes de tarjetas piratas obtienen un margen mayor al 30% durante el primer año de suscripción. Los piratas consiguen beneficios con sólo vender 8 tarjetas a 30 € cada una (datos extraídos de [41]). Si, por otro lado, sólo se dedican a la venta de tarjetas, sin suscripción a la actualización de las claves de decodificación, sólo necesitan vender 2 tarjetas para amortizar los costes de producción.

A pesar de los grandes márgenes de beneficio para los vendedores de televisión de pago y la baja tasa de suscripción de los consumidores, los productos piratas también tienen desventajas. Los consumidores de televisión de pago ilegales a menudo se encuentran con interrupciones en la programación como, por ejemplo, en los cambios de las claves de decodificación. Entonces precisan de una actualización de las claves para volver a capturar la señal. Estas interrupciones pueden provocar que estos espectadores ilegales formalicen su suscripción legalmente para evitar el complicado y costoso proceso de actualización de claves.

2.4.3 Efectos económicos de la piratería de la televisión de pago

La piratería de la televisión de pago redirecciona los ingresos de los proveedores de la televisión de pago a los piratas. Además, los ingresos de escritores, guionistas, artistas, actores, músicos y otros productores de contenidos han decrecido por culpa de la piratería debido a la reducción de la rivalidad y demanda de contenidos de televisión de pago.

2.4.3.1 En los proveedores de CAS

La existencia de la piratería de televisión de pago provoca que los proveedores de CAS inviertan sustancialmente en la integración de mecanismos antipiratería complejos en sus sistemas y actualizando continuamente su sistema de seguridad.

La demanda de receptores ilegales, los cuales pueden ser fácilmente manipulados, está aumentando en el mercado. Esto provoca que los proveedores de CAS tengan que integrar sus sistemas en los receptores.

Indirectamente, los proveedores de CAS también sufren pérdidas cuando aumenta su mala reputación y como consecuencia pierden a sus clientes. Para que esto no suceda, la mayoría de los proveedores de CAS proporcionan un seguro contra la piratería, en el cual están dispuestos a cambiar todas las *smart cards* de forma gratuita en el caso que las *smart cards* hayan sido hacheadas por piratas.

2.4.3.2 En los proveedores de televisión de pago

El uso de productos de piratería sólo reduce los ingresos y la demanda si asumimos que los piratas pagarían alternativamente por la suscripción.

La piratería de la televisión de pago no afecta significativamente en el número de suscripciones, sino en el tipo de suscripción. La suscripción Premium ha decaído ya que los suscriptores mediante una suscripción básica y la piratería, pueden tener acceso a todos los canales. Esto ha provocado el cierre de canales, así como el paro de futuros canales Premium planeados.

Si los CAS no son capaces de reducir la piratería hasta un nivel aceptable, la única medida técnica que les quedará a los proveedores de CAS será el intercambio regular de *smart cards* o sino una actualización completa del sistema. Obviamente, ambas medidas son muy costosas, pero no obstante necesarias.

2.4.4 Clasificación de los piratas en diferentes grupos

Se pueden distinguir tres grupos de piratas de televisión de pago: piratas profesionales, fabricantes locales de tarjetas de pago y piratas de la industria del hogar.

Los piratas profesionales utilizan equipamiento técnico y producen un gran número de tarjetas pirata. Ellos mismo producen y venden las tarjetas de forma muy profesional.

Los fabricantes locales, por otro lado, utilizan métodos de producción algo menos industriales, no obstante también causan estragos a la industria de la televisión de pago. Éstos, tanto producen las tarjetas desde cero como utilizan tarjetas en blanco disponibles

comercialmente. Existen manuales para manipular las *smart cards*, para construir dispositivos programables y para descifrar claves que pueden ser descargados de Internet. Las URLs están consideradas ilegales y para continuar existiendo cambian con frecuencia.

Además de vender tarjetas piratas, los piratas fabricantes locales generan beneficios vendiendo *smart cards* en blanco, dispositivos de programación o sistemas de recepción de satélite completos, incluido la *smart card*. Por otro lado, el crimen organizado a menudo utiliza a los fabricantes locales como canales de distribución para tarjetas piratas fabricadas profesionalmente.

Los piratas de la industria del hogar "crackean" los sistemas de códigos únicamente por satisfacción personal o como reto intelectual. Técnicamente los experimentados espectadores de la televisión de pago actúan como piratas por afición, utilizando el mismo hardware que los piratas profesionales y la información de los sitios Web de hackers. Estas tarjetas piratas fabricadas por uno mismo se utilizan en las propias salas de estar de los piratas o las venden o ofrecen por un módico precio a los amigos, vecinos o familiares. Los piratas aficionados suelen recibir copias piratas de software, música, CDs o DVDs. Por el hecho de no publicar ni venden sus logros, éstos tienen un efecto pequeño dentro de la industria. A excepción de algunos profesionales, los piratas dependen de los sitios Web de hackers para obtener las claves, el software, el hardware y los manuales. Estas webs de hackers, sin soporte comercial, proporcionan el principal punto de intercambio para muchos usuarios, convirtiéndose así en la piedra roseta de la piratería no profesional.

2.4.5 Ataques

Existe una gran variedad de métodos para sortear la seguridad en función del grado de sofisticación. El método escogido por el atacante depende de múltiples factores clave; que servicio está siendo seleccionado, su nivel de seguridad y la habilidad del usuario. Algunos ataques simples están disponibles en los equipos, pero otros precisan de un alto grado de conocimiento para llevarlos a cabo.

2.4.5.1 Comunidad Black Hat

La comunidad alrededor de esta tecnología se está ampliando y crece de forma rápida. Existen foros para múltiples motivos; zona de principiantes donde las preguntas pueden ser contestadas, zonas de usuarios avanzados para la programación de nuevos ataques, sección de consejos para discutir sobre nuevos equipos (receptores, antenas, CAMs y tarjetas) y discusiones abiertas relacionadas con noticias y eventos. Un aspecto a tener en cuenta es la naturaleza secreta de algunas de las comunidades más establecidas. Éstas no paran de mover algunos de sus recursos de Internet de una localización virtual a otra. Después de que un miembro haya proporcionado suficientes consejos o que haya demostrado sus habilidades, éste será elevado a grupos más selectos para discutir sobre avances realizados. Debido a la naturaleza de Internet resulta muy complicado erradicar estas comunidades, tal y como sucede con otras muchas comunidades ilícitas de Internet de otros tipos.

2.4.5.2 Ataques sin tarjeta

La forma más básica de ataque es el ataque individual sin tarjeta. Algunos CASs sólo cuentan con elementos *smart card* y otros sólo con CAM, pero se asume que los CAS disponen de ambos elementos. Los usuarios deben disponer de ambos elementos activos en sus receptores abiertos para descodificar la señal DVB.

No obstante, la emulación existe y puede proporcionar la funcionalidad de los CAMs y de las *smart cards* mediante software. En el momento en el que se consigue crear una imagen y añadirla al receptor abierto, es preciso habilitar la emulación y el débil sistema CAS podrá ser visionado. La experimentación práctica ha demostrado que un gran número de sistemas CAS utilizados han sido rotos fácilmente.

Los sistemas CAS más avanzados son resistentes a este tipo de ataque y precisan de métodos de ataque más complejos. Un ejemplo de estos métodos es el ataque que se realiza compartiendo tarjetas (*Card Sharing*). Este método se describe en el siguiente apartado. Un aspecto muy importante es la facilidad con la que un ataque se puede llevar a cabo.

Como demostración de la simplicidad de realizar un ataque sin tarjeta, el receptor abierto se puede reconfigurar para romper aquellos servicios débilmente codificados con no mucho más que las herramientas que proporciona la comunidad black hat. Para llevar a cabo este tipo de ataques sólo se precisa del software recomendado por la comunidad black hat, una

DreamBox (receptor abierto comercial), un PC, una antena parabólica y una tarjeta sintonizadora de televisión.

A continuación se detalla brevemente el procedimiento. Primero se conecta la DreamBox al PC y se le carga una imagen existente. Después de esta operación la imagen original de la DreamBox habrá sido sustituida por una imagen comprometida. La lista de configuración de satélites se habrá cargado y estará conectado a un satélite decidido previamente. Después de haber activado ciertas funcionalidades, los programas DVB codificados estarán disponibles sin ningún tipo de suscripción.

2.4.5.3 Tarjeta compartida (Card Sharing)

El ataque mediante tarjeta compartida es altamente sofisticado en comparación con el ataque sin tarjeta. En este método se le hace un *bypass* a la seguridad del CAS de forma lucrativa. El ataque consiste en permitir a un usuario que tenga una tarjeta legítima, operar con un gran número de usuarios ilegítimos y proporcionarles acceso no autorizado a todos ellos a los contenidos protegidos. Este tipo de ataque se centra en los usuarios de receptores abiertos y está en constante evolución.

El usuario con la tarjeta suscrita hace correr un Card Server en su receptor abierto reconfigurado y escucha las comunicaciones de los clientes en un determinado puerto. Los EMMs y los ECMs son enviados por los clientes al servidor para que éste los descodifique por turnos. El servidor devuelve a los clientes el CW utilizado para la codificación de los contenidos. El servidor realiza una autenticación unilateral de los clientes utilizando la tarjeta legítima y a continuación intercambia ECMs y EMMs permitiendo el acceso a los programas DVB a los clientes sin suscripción. El único factor limitante de la red es que el servidor de tarjetas proporciona una hebra a cada cliente. Esto quiere decir que el servidor sólo puede soportar tantos clientes como la tarjeta permita. Las tarjetas están limitadas a nivel de recursos, como ahora computacionalmente, a nivel de almacenamiento y de comunicaciones. Esto provoca que el número de posibles usuarios autentificados por una tarjeta sea finito.

El conocimiento para poder realizar este ataque está disponible a través de la comunidad black hat y la reconfiguración de los receptores abiertos es sencilla para los clientes y un poco más complicada para los servidores. El alcance del ataque depende de la implementación del CAS.

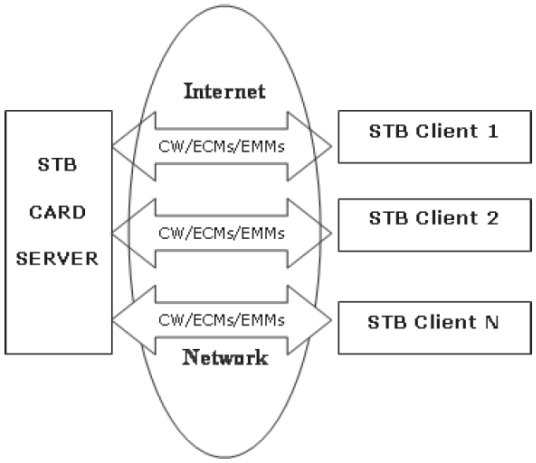

Figura 9: Arquitectura de la tarjeta compartida

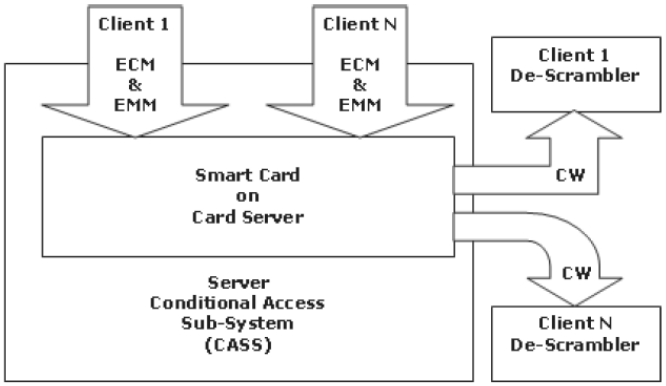

Figura 10: Ataque mediante tarjeta compartida

El desarrollo de ataques a los sistemas DVB es muy complicado de realizar y requiere de un alto grado de conocimiento que no es fácilmente alcanzable. No obstante, las comunidades black hat proporcionan herramientas que permiten a los usuarios dejar de lado los detalles técnicos. Éste es uno de los factores claves del amplio incremento del uso de receptores abiertos de una forma ilegal, además, también influye el hecho que un atacante pueda experimentar con relativa certeza de que no va a ser cogido.

2.4.6 Medidas anti-piratería utilizadas

El elemento principal de las medidas anti-piratería es el Sistema de Acceso Condicional. Éste se presenta como la medida más importante a la hora de determinar el logro o el fracaso de un ataque. Por otro lado, a otro nivel está la utilización de la *smart card*. La

smart card es un módulo externo del receptor que el usuario fácilmente la puede introducir o extraer de éste. La *smart card* posee la información privada necesaria para poder acceder al sistema. La capacidad de la *smart card* de realizar operaciones permite aumentar la seguridad del sistema ya que se presenta como un módulo robusto frente a ataques físicos y resulta muy complicado hacerse con la información que contiene.

No obstante, por muchos elementos de seguridad que presente la *smart card* la historia hace pensar que cabe la posibilidad de que un atacante con los medios necesarios sea capaz de obtener el contenido de la *smart card* (claves, algoritmos de cifrado...). Como consecuencia, ese atacante podrá acceder a todos los contenidos a los que la *smart card* en cuestión tenía acceso. Para contrarrestar este echo, los proveedores de servicios de acceso condicional permiten renovar la población de *smart cards* en posesión de los usuarios y de este modo, cambiar la información privada almacenada en las *smart cards* de todo el sistema.

Existe otro motivo por el que se lleva a cavo el cambio de *smart cards* a parte del ya mencionado. El cambio de *smart cards* también puede venir motivado por el hecho que se realicen pequeñas modificaciones en el sistema, cosa que pondrá las cosas más difíciles a los atacantes, ya que se pueden ver modificados la longitud de las claves, los algoritmos de cifrado, nuevos códigos de comandos así como modificación de los ya existente, entre otros. Todo esto, se puede llevar a cabo con un coste relativamente bajo, ya que los receptores de los usuarios continúan siendo los mismo y lo único que hay que hacer es comprar, configurar y distribuir las nuevas *smart cards*.

Además de las medidas ya comentadas, existen otras más agresivas que son el envío por parte del centro emisor de comandos no soportados por las *smart cards* piratas y los cierres de páginas Web en las que se proporciona información para poder acceder a los contenidos de la televisión de pago de forma ilícita.

2.4.7 Medidas anti-piratería propuestas

Como medida a tomar para evitar el problema del card sharing se propone poner énfasis en el correcto funcionamiento de la tarjeta. En esencia, que la tarjeta tenga conocimiento del comportamiento a seguir y que en cuanto se viole este comportamiento, que deniegue los servicios de acceso.

La tarjeta necesita tener idea de que procesos de instrucciones espera recibir en un periodo de tiempo concreto. Analizando el comportamiento de los procesos en la tarjeta o

aprendiendo sobre el tiempo de estos, la tarjeta podrá detectar el comportamiento de procesos anómalos y prevenir o detener el ataque. Se necesita que la *smart card* tenga memoria de estados y que sea capaz de procesar transiciones de estado para soportar dicha funcionalidad.

En relación a las estadísticas del tiempo de ejecución, se debe considerar que a lo largo de un determinado periodo de tiempo se han de producir un número concreto de procesos. Por ejemplo, el proceso normal de autenticación debe ser de una vez cada dos minutos. En el caso del ataque mediante card sharing, en el cual un determinado número de receptores sin tarjeta se comunican con una tarjeta central, se debe acceder a la tarjeta de forma muy regular para poder autentificar los mensajes, por lo que este comportamiento puede ser detectado y como consecuencia se puede prevenir. No obstante, el microcontrolador de la *smart card* depende de una señal de reloj externa y por este motivo, la *smart card* no dispone de una fuente interna para poder "rastrear" el tiempo independientemente del receptor.

En un ataque mediante card sharing, todos los mensajes ECM y EMM son enviados desde el grupo de usuarios ilegales a una *smart card*. Es obvio el ver que esta *smart card*, en el caso que estén conectados a ella 100 usuarios, estará procesando 100 mensajes más de lo normal. Por eso, se propone que la tarjeta esté habilitada a tratar un determinado número de mensajes ECM por hora. En caso de sobrepasarlo, la tarjeta deberá negar el servicio. Una propuesta interesante sería que la tarjeta realizase configuraciones de forma dinámica. De este modo, la tarjeta puede determinar un punto de referencia para su uso y en caso de desviarse considerablemente detectar el comportamiento anómalo.

La mayoría de las contramedidas propuestas han sido extraídas de [42].

2.4.8 Traitor tracing

El traitor tracing es una prevención de copia y detección de "fugas" del sistema. El método consiste en que el distribuidor añade un valor único para cada ejemplar determinado. Cuando una copia de la misma se filtra al público, el distribuidor puede comprobar el valor en él y determinar la fuente.

Los esquemas de traitor tracing se utilizan cuando un proveedor desea enviar información de forma segura a múltiples usuarios autorizados. Pero algunos usuarios autorizados, a los que se les llama traidores, ayudan a los usuarios no autorizados para que puedan acceder a dicha información.

Para ayudar a los piratas, los traidores pueden obtener simplemente la información haciendo uso de sus claves de descifrado y dar la información a los piratas. Dado que existen ciertas restricciones a la hora de dar la información, los traidores pueden proceder de una forma alternativa y es entregando las claves de descifrado en vez de la información. Mediante este conocimiento, los piratas actúan como un usuario autorizado y extraen la información clara de forma sencilla de la información cifrada.

Los esquemas de traitor tracing ayudan en tres aspectos en la prevención de la piratería: disuaden a los usuarios de cooperar con piratas, identifican a los piratas y permiten emprender acciones legales contra estos y también pueden ser usados para inhabilitar a usuarios piratas activos.

El uso de los esquemas de traitor tracing desmotivan a los usuarios a que ayuden a los piratas y especialmente a que compartan sus claves para que sean utilizadas en descodificadores ilegales. En particular, si el proceso de obtención de una clave personal por parte de un usuario requiere de una identificación física o registro, será más difícil la obtención de un gran número de claves por parte de los piratas. Consecuentemente, los esquemas de traitor tracing pueden identificar las fuentes de las claves que son utilizadas en los decodificadores piratas y con este simple hecho disuadir a los usuarios de ayudar a los piratas.

Cuando se encuentra un descodificador pirata y se identifica la fuente de las claves, se deberían emprender acciones legales contra esta fuente.

3 Broadcast Encryption

3.1 Introducción

En 1991, Shimons Berkovits introdujo el concepto de *Broadcast Encryption* y posteriormente, en 1993, fue formalizado por Amos Fiat y Moni Naor [1]. El tema de *Broadcast Encryption* ha adquirido especial relevancia y se han presentado varios algoritmos desde entonces, tal y como se puede comprovar en [9], [16], [25], [27] y [30].

El concepto *Broadcast* significa transmitir información a un conjunto de receptores. Un ejemplo de transmisiones *broadcast* es la radio, para ello se utiliza el aire como medio de propagación y todo aquel que disponga de un receptor de radio, es capaz de escuchar la transmisión. Este tipo de comunicaciones suele ser de un solo sentido, ya que los receptores no son capaces de enviar nada al centro transmisor.

Existen ciertas aplicaciones, como ahora la televisión de pago, las comunicaciones vía satélite o la protección de contenidos de medios de almacenamiento como por ejemplo el *Blu-ray Disc*, en las que se desea "asegurar" el contenido del mensaje transmitido, de tal forma que sólo pueda ser leído por aquellos usuarios autorizados. El *Broadcast Encryption* soluciona el problema de cómo llevar a cabo transmisiones seguras a un grupo autorizado de usuarios a través de un medio inseguro. El grupo de usuarios puede variar de forma dinámica, por ejemplo, cuando un cliente inicia o finaliza su suscripción a un canal de la televisión de pago.

En la teoría del *Broadcast Encryption*, a los usuarios que están autorizados para leer un mensaje se les llama usuarios privilegiados y a los que no, se les llama usuarios revocados. Los términos receptor y usuario se suelen usar indistintamente, la única diferencia radica en que todos los receptores pueden acceder al mensaje transmitido, pero sólo los usuarios privilegiados pueden acceder a su contenido.

El mensaje transmitido se suele dividir en dos partes, la cabecera y el cuerpo. El cuerpo está formado por el contenido protegido y la cabecera contiene la información necesaria para acceder al contenido. La parte más importante a la hora de analizar los algoritmos es la cabecera, ya que es donde se ubica la información referente a las claves y a los grupos.

El esquema más simple de *Broadcast Encryption* es aquel que se le asigna una clave simétrica secreta a cada usuario privilegiado y se cifra el mensaje con cada una de estas claves.

A continuación, se transmiten todos los mensajes cifrados. Claramente se trata de un esquema muy ineficiente en términos de tiempo de procesado y tamaño del mensaje transmitido. No obstante, se le puede introducir una pequeña modificación para ahorrar ancho de banda y tiempo de procesado. La solución pasa por cifrar el mensaje una única vez con una clave *k* y a su vez cifrar la clave *k* con cada una de las claves simétricas de los usuarios privilegiados.

Mediante este esquema se consigue que cada usuario tenga que almacenar una sola clave y que la longitud de transmisión sea proporcional al tamaño del grupo de usuarios privilegiados. Se trata de un esquema práctico sólo cuando el grupo de usuarios privilegiados es pequeño.

Además de éste, existe otro esquema sencillo que consiste en proporcionar una clave para cada posible grupo de usuarios. De esta forma, todo usuario deberá tener almacenadas las claves de los grupos a los que pertenezca. Dado que cada usuario podrá pertenecer a 2^{n-1} grupos, donde *n* es el número de usuarios totales, éstos deberán almacenar 2^{n-1} claves.

También se trata de un esquema ineficiente, pero esta vez en términos de capacidad de almacenamiento por parte de los receptores, ya que éstos deben almacenar un gran número de claves. Este esquema resulta imposible de implementar cuando *n* es elevada.

Los objetivos de los esquemas son: reducir el tiempo de procesado (*Computational Cost* o CC), tanto por parte del centro emisor como por el de los receptores, reducir el tamaño del mensaje transmitido (*Transmission Overhead* o TO) y reducir la capacidad de almacenamiento en los receptores (*Storage Space* o SS) [1]. El tamaño del mensaje transmitido (TO) depende tanto del número de mensajes cifrados que contiene, como del tamaño de la cabecera del mensaje. En la cabecera se suele encontrar información crítica como claves cifradas e información sobre que receptor puede descifrar el mensaje.

El algoritmo de *Broadcast Encryption* cifra el mensaje de tal forma que múltiples usuarios sean capaces de descifrarlo. Esto se puede realizar de muchas maneras dependiendo del algoritmo utilizado. Por ejemplo, para enviar un mensaje a múltiples usuarios es necesario hacer un grupo con todos ellos, es decir, que éstos posean cierta información común.

El algoritmo utilizado para agrupar a los suscriptores es muy importante para determinar la eficiencia de todo el sistema de *Broadcast Encryption*. Éste puede ser dinámico o preestablecido. En un sistema dinámico la jerarquía de grupo de los suscriptores es modificable en todo momento, puede aparecer un nuevo usuario privilegiado o revocarse a un usuario del sistema. Mientras que en el sistema preestablecido, la jerarquía de grupo está predefinida y no varía.

La mayoría de esquemas de *Broadcast Encryption* realizan los siguientes pasos:

Inicialización: Se le asigna a cada usuario algún tipo de información secreta, como por ejemplo un conjunto de claves, que permite al usuario descifrar el mensaje cifrado.

Cifrado: El centro emisor usa un algoritmo de *Broadcast Encryption* que toma como entrada el mensaje y el conjunto de usuarios revocados y a continuación, obtiene el mensaje cifrado.

Descifrado: El usuario puede usar su información secreta para descifrar el mensaje cifrado y obtener así el mensaje claro. La única condición que se debe cumplir es que el usuario pertenezca al conjunto de usuarios privilegiados.

Llegado este punto, es importante introducir el concepto de algoritmo *stateless* y algoritmo *stateful*. Aquellos sistemas que implementan algoritmos *stateless* no son capaces de actualizar la información que posee el usuario y que se utiliza para el descifrado. La información es almacenada en los receptores o en las *smart cards* en el proceso de fabricación. Por otro lado, los receptores/sistemas que implementan algoritmos *stateful* actualizan la información almacenada que se utiliza para el descifrado del contenido transmitido, cuando se añade o revoca a un usuario al sistema. Dado que en este proyecto se estudian aplicaciones de difusión de contenidos de un único sentido, sólo se mencionarán los algoritmos del tipo *stateless*.

3.2 Algoritmos Stateless

Los esquemas de tipo *Subset Cover* se presentaron por primera vez en [27]. Los que se van a presentar a continuación son del tipo *stateless*, que como ya se ha comentado anteriormente, son aquellos que sólo necesitan realizar la inicialización del receptor una única vez. Después, los receptores son capaces de descifrar cualquier mensaje si éstos están autorizados. Esto es posible sin tener que actualizar la información almacenada.

Estos algoritmos llevan a cabo una agrupación predefinida de los usuarios N. A cada conjunto de usuarios se le llama S (*subset*), no es necesario que los conjuntos sean disjuntos, por lo que un usuario puede ser miembro de más de uno. Para poder diferenciar entre conjuntos, a cada uno se le asigna un índice. Al conjunto de subconjuntos que contienen a todos los usuarios privilegiados U y a ninguno de los usuarios revocados R, se le llama *cover*,

C. Los subconjuntos en los *covers* son siempre disjuntos, lo que significa que un usuario pertenecerá como mucho a un subconjunto dentro del *cover*. A continuación se muestra con una notación matemática lo que se acaba de explicar, donde S_{i_x} es un subconjunto de usuarios privilegiados.

$$C = \left\{S_{i_1}, S_{i_2}, ...S_{i_m}\right\}$$

$$N \setminus R = \bigcup_{j=1}^{m} S_{i_j}$$

A cada subconjunto se le asocia una clave secreta, *L*. Esta clave sólo será conocida por aquellos usuarios que pertenezcan al subconjunto en cuestión.

Todos estos algoritmos precisan de dos funciones básicas de cifrado. A la primera función se le llama $F_K(M)$ y se utiliza para cifrar el mensaje M, el cual es el contenido del mensaje transmitido. Este mensaje puede ser de una longitud considerable, por lo que F deberá ser bastante rápida. Una buena opción es elegir F como un cifrador en flujo.

La segunda función es $E_{L_i}(K)$ y se usa para cifrar la clave de sesión secreta K. Estas claves son cortas y de longitud fija, por lo que se usará una función de cifrado E más lenta. Por lo general, el receptor sólo deberá llevar a cabo un descifrado. Una buena opción es elegir E como un cifrador en bloque.

En este tema se van a describir 6 tipos diferentes de esquemas de *Broadcast Encryption*. Los tres primeros son esquemas de revocado basados en una estructura en forma de árbol y los tres últimos están basados en una estructura lineal.

3.2.1 Esquemas en forma de árbol

En los esquemas con estructura en forma de árbol, se define *N* como un conjunto de *n* usuarios, donde $n = 2^h$. En estos esquemas, cada usuario corresponde a un nodo hoja en el árbol binario de altura *h*. En el caso de que $n \neq 2^h$, se debe usar el árbol binario completo más pequeño con más de *n* hojas. El árbol binario completo posee $2n - 1$ nodos. En la *Figura 11* se puede observar un árbol binario completo con 16 hojas (nodos del 16 al 31), donde cada hoja hace referencia a un usuario.

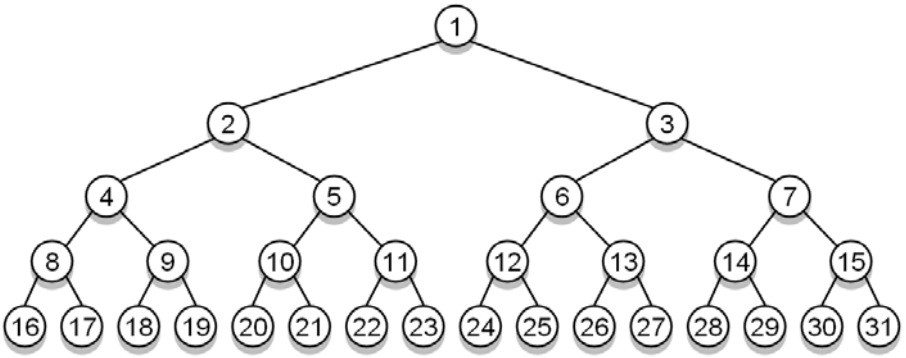

Figura 11: Árbol binario completo

3.2.1.1 Complete Subtree (CS)

En este algoritmo los subconjuntos se pueden representar gráficamente como agrupaciones de nodos en forma de árbol, tal y como se ha comentado anteriormente. El proceso de selección de los subconjuntos adecuados en el algoritmo *Complete Subtree* es bastante sencillo. Se trata de buscar a lo largo de todo el árbol aquellos nodos, lo más próximos al nodo raíz posible, que tengan por descendientes únicamente a usuarios privilegiados. Se puede ver un ejemplo del resultado de este procedimiento en la *Figura 12*.

Inicialización

Para empezar, el centro emisor debe generar claves aleatorias para cada subconjunto del árbol. A cada usuario se le proporcionará esta información a través de un canal privado. Por ejemplo, esta información será entregada al usuario mediante una *smart card* o codificándola en el receptor en el momento de su fabricación.

El receptor ha de ser capaz de deducir todas las claves secretas de los subconjuntos a los que pertenece a partir de la información secreta que el centro emisor le proporcionó en su día. En el esquema *Complete Subtree* la información secreta son todas aquellas claves de los subconjuntos a los que el usuario pertenece, en otras palabras, todos los subconjuntos desde el receptor hasta el nodo raíz. Por lo que la longitud de la información secreta es $O(\log(n)+1)$.

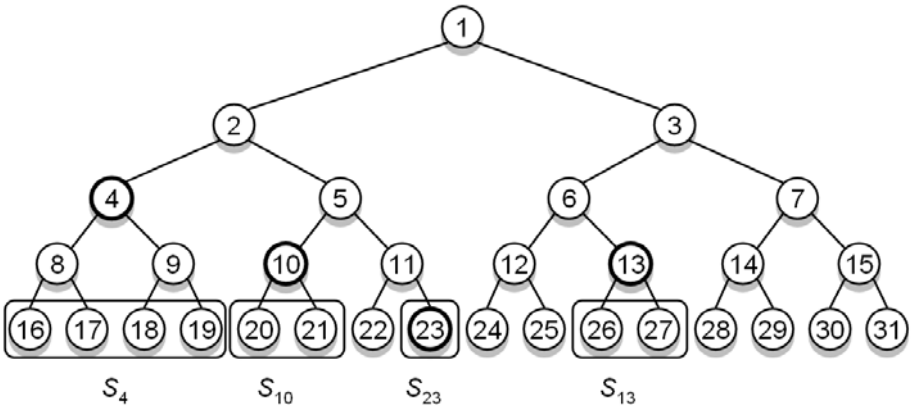

Figura 12: Esquema Complete Subtree

En la figura anterior, se puede comprobar un ejemplo de como se realizaría la agrupación de los usuarios privilegiados mediante el algoritmo *Complete Subtree*. El conjunto de usuarios revocados para el caso de la figura es el {22, 24, 25, 28, 29, 30, 31}. Los subconjuntos de usuarios privilegiados resultantes para el conjunto de usuarios revocados mencionado son $\{S_4, S_{10}, S_{23}, S_{13}\}$, tal y como se puede ver en la *Figura 12*.

Cifrado

1. La finalidad de cifrar el mensaje radica en que sólo aquellos usuarios privilegiados sean capaces de leerlo. Estos usuarios son miembros de un subconjunto del *cover* y por eso disponen de la clave para ese subconjunto. El centro emisor usa esas claves para cifrar el mensaje de la siguiente manera:
2. Elige una clave de sesión aleatoria K y cifra el mensaje utilizando la función de cifrado $F_K(M)$.
3. Para cada subconjunto del *cover*, cifra la clave de sesión usando la función de cifrado $E_{L_i}(K)$, donde L_i es la clave secreta asociada al subconjunto.
4. Añade las claves cifradas junto a sus índices y el mensaje cifrado a la cabecera y al cuerpo del mensaje transmitido respectivamente.

$$\left\langle \left[i_1, i_2, ..., i_m; E_{L_{i_1}}(K), E_{L_{i_2}}(K), ..., E_{L_{i_m}}(K) \right], F_K(M) \right\rangle$$

Si se toma el caso de la *Figura 12*, el mensaje a transmitir quedaría de la siguiente forma:

$$\left\langle \left[i_4, i_{10}, i_{13}, i_{23}; E_{L_4}(K), E_{L_{10}}(K), E_{L_{13}}(K), E_{L_{23}}(K) \right], F_K(M) \right\rangle$$

Descifrado

El descifrado es un proceso sencillo, ya que en el receptor se tienen almacenadas todas las claves necesarias para el descifrado. El único problema reside en buscar si el usuario pertenece a algún subconjunto específico de los de la cabecera del mensaje transmitido. Cuando se encuentra el subconjunto, la clave correspondiente puede ser recuperada de la información secreta en un cierto tiempo. El descifrado está compuesto por los siguientes pasos:

1. Buscar en la cabecera del mensaje transmitido el subconjunto S_i al que el usuario pertenece.
2. Recuperar la clave del subconjunto L_i de la información secreta.
3. Descifrar la clave de sesión K usando $E_{L_i}^{-1}\left(E_{L_i}(K)\right) = K$.
4. Descifrar el mensaje usando $F_K^{-1}\left(F_K(M)\right) = M$.

Una forma de comprobar si el usuario pertenece al subconjunto, es revisando desde el usuario hasta el nodo raíz, si alguno de los subconjuntos está en la cabecera. No obstante, existe otra forma más eficiente para ello. Se debe representar el índice mediante un vector binario, en el cual el nodo de la izquierda se representa mediante un 0 y el de la derecha mediante un 1. Por ejemplo, el subconjunto con el índice 20 se representará con el vector binario 0100, tal y como se puede comprobar en la *Figura 13*. De esta forma, si un usuario pertenece al subconjunto, simplemente deberá comprobar que el vector binario del subconjunto es el prefijo en el vector binario del usuario.

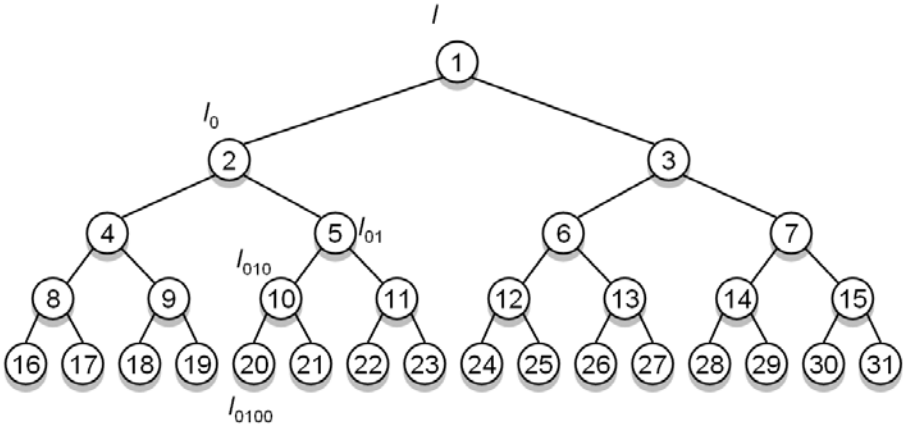

Figura 13: Asignación de vector binario al nodo 20

Revocación

Un usuario individual puede ser revocado con $O(\log(n))$ cifrados. Si se tienen que revocar a *r* usuarios, no se podrán usar ninguna de las claves que poseen almacenadas estos *r* usuarios. Por lo tanto, después de eliminar estas claves, el árbol de claves se dividirá en como mucho *r* subárboles, cada uno de ellos con $O(n/r)$ nodos hoja. De ahí que en el peor de los casos, sean necesarios $O(r\log(n/r))$ cifrados para $n-r$ usuarios privilegiados.

3.2.1.2 Subset Difference (SD)

Este algoritmo es muy similar al *Complete Subtree*, con la diferencia de que se trata de un algoritmo más eficiente a la hora de describir el *subset cover*. Esto se debe a que un usuario puede pertenecer a un número más elevado de subconjuntos. El *Subset Difference* se hizo público en el 2001 en [27].

En el algoritmo *Subset Difference*, los subconjuntos se definen de tal manera que sean el conjunto diferencia entre dos subconjuntos de los usuarios del árbol. El subconjunto $S_{i,j}$ se define como todos los usuarios que son descendientes del nodo *i*, pero que no lo son del nodo *j*. También se puede escribir como $S_{i,j} = S_i \setminus S_j$, donde S_x es el conjunto de nodos hoja del subárbol en el que *x* es la raíz. Si $i = j$, $S_{i,j}$ es un conjunto vacío, por ello, a este conjunto no se le asigna ninguna clave. En la *Figura 14* hay un ejemplo de lo que se ha comentado, se trata del mismo conjunto de usuarios privilegiados que el mostrado en la *Figura 12*. Como se puede observar, en el caso del *Subset Difference* es posible agrupar al conjunto de usuarios privilegiados en cuestión, en 2 subconjuntos, mientras que para el caso del *Complete Subtree* eran necesarios 4.

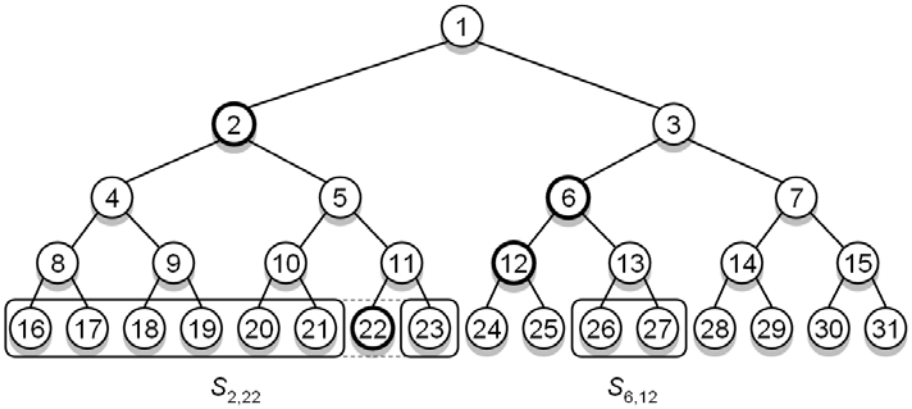

Figura 14: Esquema Subset Difference

En la *Figura 15* se muestra un ejemplo en el cual hay 7 nodos hoja revocados y el *Steiner tree ST*(*R*) de estos usuarios revocados está resaltado. El *Steiner tree* es aquel árbol de cobertura mínima que contiene a determinados elementos. A partir de éste y usando el algoritmo, se encuentra que el *subset cover* para los nodos hoja que hay es $\{S_{2,22}, S_{6,12}\}$, tal y como se observa en la *Figura 14*.

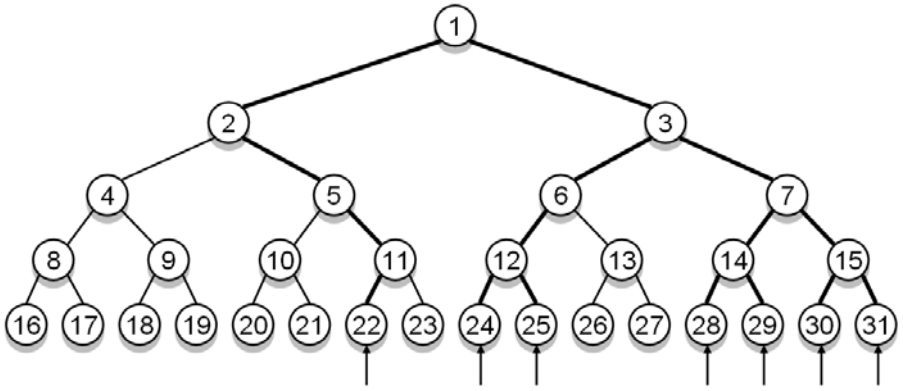

Figura 15: Steiner Tree

La consecuencia directa de que un usuario pertenezca a un mayor número de subconjuntos es la necesidad de almacenar un mayor número de claves. De este modo, este algoritmo es más eficiente que el anterior en lo referente a la longitud del mensaje transmitido, ya que la reduce, pero por el contrario, es necesaria una mayor capacidad de almacenamiento por parte de los receptores. Para solventar el problema de las claves, el algoritmo puede usar un mecanismo de asignación de claves computacional y de este modo, reducir el número de claves a almacenar en el receptor. Este mecanismo se explicará más adelante.

Inicialización

Del mismo modo que para el caso del *Complete Subtree*, en el *Subset Difference* el centro emisor debe generar claves aleatorias para cada subconjunto del árbol. A cada usuario se le proporcionará esta información a través de un canal privado.

El receptor ha de ser capaz de deducir todas las claves secretas de los subconjuntos a los que pertenece a partir de la información secreta que el centro emisor le proporcionó en su día. En el esquema *Subset Difference* la información secreta es simplemente el conjunto de claves de los subconjuntos a los que el usuario pertenece, en otras palabras, todos los subconjuntos desde el receptor hasta el nodo raíz. Por lo que la longitud de la información secreta es $O(\log^2(n))$, mucho mayor que la del esquema *Complete Subtree*.

Cifrado

El centro emisor realiza los siguientes pasos para el cifrado del mensaje:

1. Elige una clave de sesión aleatoria K y cifra el mensaje utilizando la función de cifrado $F_K(M)$.
2. Para cada subconjunto $S_{i,j}$, cifra la clave de sesión usando la función de cifrado $E_{L_{i,j}}(K)$, donde $L_{i,j}$ es la clave secreta asociada al subconjunto.
3. Añade las claves cifradas junto a sus índices y el mensaje cifrado a la cabecera y al cuerpo del mensaje transmitido respectivamente.

Si se toma el caso de la *Figura 14*, el mensaje a transmitir quedaría de la siguiente forma:

$$\left\langle \left[i_{2,22}, i_{6,12}; E_{L_{2,22}}(K), E_{L_{6,12}}(K) \right], F_K(M) \right\rangle$$

Descifrado

El descifrado está compuesto por los siguientes pasos:

1. Buscar en la cabecera del mensaje transmitido el subconjunto $S_{i,j}$ al que el usuario pertenece.
2. Recuperar la clave $L_{i,j}$ asociada al subconjunto $S_{i,j}$ de la información secreta.
3. Descifrar la clave de sesión K usando $E^{-1}_{L_{i,j}}\left(E_{L_{i,j}}(K)\right) = K$.
4. Descifrar el mensaje usando $F_K^{-1}\left(F_K(M)\right) = M$.

Cada subconjunto en el SD tiene asociado una clave $L_{i,j}$. Como ya se ha comentado, el algoritmo SD necesita un mayor número de claves almacenadas en el receptor comparado con el algoritmo CS. Este problema se soluciona utilizando una estructura de etiquetas *I* sin tener que almacenar las claves en el receptor. Las etiquetas se utilizan para generar todas las subclaves del nodo etiquetado I_i usando un generador de secuencias pseudoaleatorias, o más detalladamente una función *hash* de único sentido.

La longitud de la salida del algoritmo *hash* de único sentido es más larga que la de la entrada. En el SD al algoritmo *hash* de único sentido se le define como $G: \{0,1\}^n \rightarrow \{0,1\}^{3n}$. Dicho de otro modo, la longitud de salida del algoritmo es 3 veces mayor a la longitud de la

entrada. La salida se divide en 3 partes: la parte de la izquierda, la parte central y la parte de la derecha. A la función $G(I)$ se la puede ver como tres funciones separadas $G_L(I), G_M(I)$ y $G_R(I)$ que generan la parte izquierda, central y derecha de la salida dada una determinada entrada.

A la parte izquierda y derecha de la salida se las llama "etiquetas intermedias" $I_{i,j}$ y son la información secreta almacenada por los receptores. Dado que la función G es irreversible, la etiqueta del nodo original no puede ser derivada de las etiquetas intermedias. De este modo, sólo aquellos usuarios que pertenezcan al subconjunto en cuestión podrán generar su clave.

Una condición necesaria de la función G es que dada una entrada aleatoria, ningún adversario pueda distinguir la salida de G de una secuencia aleatoria de misma longitud.

Gracias a este sistema de generación de claves, cada usuario deberá almacenar como mínimo una clave derivada de cada nodo que esté en el camino entre el nodo hoja correspondiente al usuario en cuestión y el nodo raíz, pero nunca ha de almacenar la clave de uno de los nodos que formen parte del camino.

En el caso de que un usuario se hiciese con la clave de uno de los nodos que conforman el camino entre usuario y raíz, el usuario sería capaz de calcular claves de conjuntos a los que el usuario no pertenece. En las figuras de más adelante se pueden ver los nodos, coloreados en gris, de los que el usuario que estuviese en el nodo hoja número 22 debería almacenar sus claves. Los nodos coloreados en gris corresponden a los nodos hermanos de aquellos que forman parte del camino que va del usuario al nodo raíz.

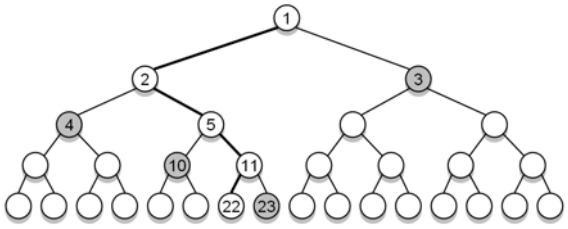

Figura 16: Nodos clave para el nodo raíz 1

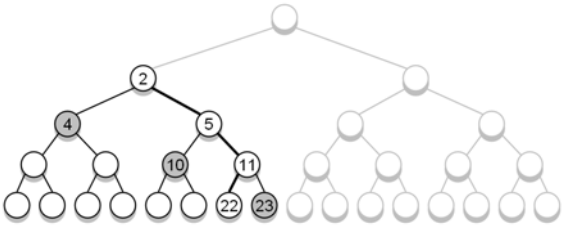

Figura 17: Nodos clave para el nodo raíz 2

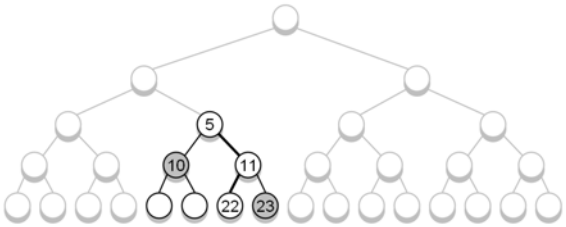

Figura 18: Nodos clave para el nodo raíz 5

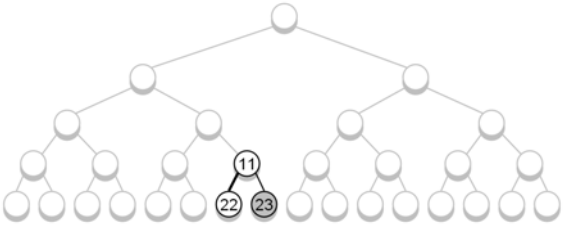

Figura 19: Nodos clave para el nodo raíz 11

Como se puede observar, para un conjunto de 16 usuarios, cada uno de ellos deberá almacenar un total de 10 claves, a partir de las cuales se pueden calcular todas las otras claves de los subconjuntos a los que el usuario pueda ser miembro. Estas claves serán $\{I_{1,3}, I_{1,4}, I_{1,10}, I_{1,23}, I_{2,4}, I_{2,10}, I_{2,23}, I_{5,10}, I_{5,23}, I_{11,23}\}$ para el nodo hoja 22. De forma más general, un usuario cualquiera deberá almacenar $\sum_{1}^{h} x$ o lo que es lo mismo $\frac{h^2+h}{2}$, donde $h = \log(n)$.

Teniendo en cuenta el ejemplo de la *Figura 14* y lo explicado hasta ahora, para generar las claves $\{L_{2,22}, L_{6,12}\}$ de los *subset* $\{S_{2,22}, S_{6,12}\}$ se deben seguir los pasos que quedan reflejados a continuación:

Usuarios	Clave almacenada	Clave generada
16 al 19	$I_{2,5}$	$L_{2,22} = G_M(G_L(G_R(I_{2,5})))$
20 y 21	$I_{2,11}$	$L_{2,22} = G_M(G_L(I_{2,11}))$
23	$I_{2,22}$	$L_{2,22} = G_M(I_{2,22})$
26 y 27	$I_{6,12}$	$L_{6,12} = G_M(I_{6,12})$

Tabla 1: Generación de claves por parte de los usuarios de la Figura 14

Revocación

A continuación se detalla una estrategia que se puede seguir para revocar a *r* usuarios.

1. Si $r = 1$, se usa la clave $S_{O,u} = S_O \setminus S_u$, en donde *O* hace referencia al nodo raíz y *u* al usuario revocado.
2. En otro caso, se coge a dos usuarios revocados x_1 y x_2 cuyo último ancestro común v no posea ningún otro descendiente revocado.
3. Llamaremos *l* al hijo de *v* que es a la vez ancestro de x_1. Por otro lado, llamaremos *k* al hijo de *v* que es ancestro de x_2. Se usan las claves de S_{l,x_1} y S_{k,x_2}. Si $l = x_1$ (o $k = x_2$), S_{l,x_1} (o S_{k,x_2}) está vacío y por ello no se añade ninguna clave.
4. Eliminar a todos los descendientes de *v* y marcar a *v* para ser eliminado. Volver al paso (1)

Como mucho 2 claves serán seleccionadas en el paso (3) y el número de nodos revocados disminuirá en uno en cada iteración, como mucho se usarán $2r-1$ claves para r usuarios revocados. No obstante, se puede decir que el número medio de claves para revocar a r usuarios es $1.25r$.

3.2.1.3 Layered Subset Difference (LSD)

El algoritmo LSD fue publicado por D. Halevy y A. Shamir en el 2002 [9]. Se trata de una extensión del algoritmo SD. El objetivo principal del algoritmo LSD es reducir el número de claves que los usuarios deben almacenar. Esto se consigue introduciendo niveles especiales en la estructura árbol del algoritmo SD.

En el LSD se definen algunos de los $\log(n)+1$ niveles como especiales, más concretamente, son niveles especiales aquellos que cumplan $k\cdot\sqrt{\log(n)}$, con $k=1,...,\sqrt{\log(n)}$. Se debe tener en cuenta que el nodo raíz se encuentra en el nivel 0. Lo que quiere decir que hay $\sqrt{\log(n)}+1$ niveles especiales equiespaciados por $\sqrt{\log(n)}$ niveles. A estos grupos de niveles consecutivos, incluido el nivel especial inferior, se los conoce como capa.

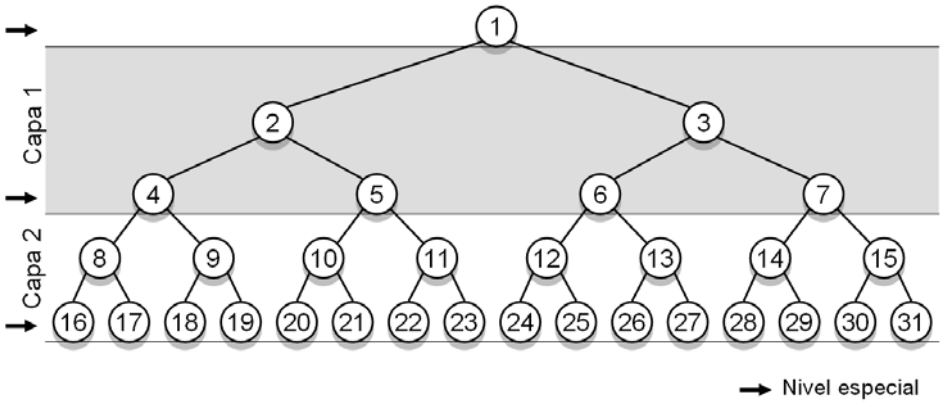

Figura 20: Niveles especiales y capas en un árbol binario de 16 usuarios

Un conjunto $S_{i,j}$ en el protocolo LSD es útil, si al menos cumple una de las siguientes condiciones:

1. que *i* pertenezca a un nivel especial.
2. que *i* y *j* estén situados entre los mismos niveles especiales, o lo que es lo mismo, que estén en la misma capa.

Cuando se ejecuta el algoritmo LSD, primero se debe crear un conjunto de subconjuntos que contengan a todos los usuarios privilegiados, del mismo modo que en el algoritmo SD. Posteriormente se deberán convertir en subconjuntos del tipo LSD, es decir que sean $S_{i,j}$ útiles. Para ello es necesario que se cumpla alguna de las dos condiciones comentadas anteriormente.

Si se tiene un subconjunto $S_{i,j}$ en el algoritmo SD, en el cual *i* no pertenece a un nivel especial y además *i* y *j* no se encuentran en la misma capa, se debe cambiar a 2 subconjuntos $S_{i,k}$ y $S_{k,j}$, en los cuales *k* deberá estar en el camino de *i* a *j* y pertenecer al nivel especial más próximo a *i*. De este modo, se define $S_{i,j} = S_{i,k} \cup S_{k,j}$. Se puede observar un ejemplo de lo comentado en la *Figura 21*.

Por otro lado, en el caso que *i* se encuentre en un nivel especial o bien que *i* y *j* se encuentren en la misma capa, se deberá mantener el subconjunto $S_{i,j}$ obtenido mediante el algoritmo SD.

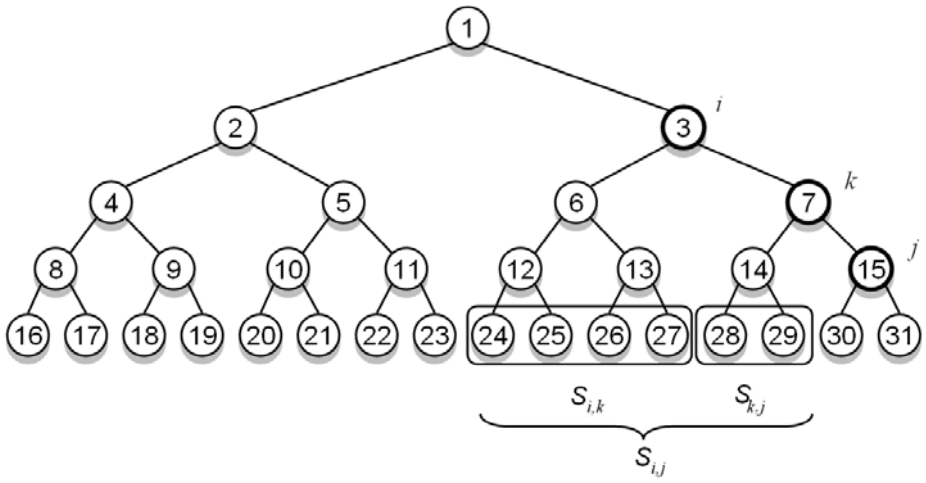

Figura 21: Reconstrucción de subconjuntos SD con subconjuntos LSD

De este modo, cualquier subconjunto SD que no cumpla las dos condiciones de subconjunto útil, se deberá expresar como la unión de dos subconjuntos útiles. Por lo que el número de cifrados se verá multiplicado por dos en el peor de los casos. No obstante, el número de claves que cada usuario deberá almacenar se verá reducido considerablemente.

Para calcular el número de claves que cada usuario debe almacenar podemos distinguir entre dos tipos de conjuntos, los locales y los especiales.

Conjuntos locales: Cada capa contiene $\sqrt{\log(n)}$ niveles, de este modo el número de parejas de *i*,*j* que pertenecen a una capa es $O\left(\left(\sqrt{\log(n)}\right)^2\right) = O(\log(n))$. Dado que hay $\sqrt{\log(n)}$ posibles capas, el número de parejas *i*,*j* de este tipo es $O\left(\log^{3/2}(n)\right)$.

Conjuntos especiales: Cada *i* que se encuentre en un nivel especial se puede asociar se puede asociar con cualquier *j* que se encuentre en uno de los $O(\log(n))$ niveles por debajo de éste. Dado que hay $\sqrt{\log(n)}$ posibles *i*, el número de claves de este tipo también es $O\left(\log^{3/2}(n)\right)$.

Como consecuencia, el número total de claves que un usuario debe conocer se reduce de $O(\log^2(n))$ a $O\left(\log^{3/2}(n)\right)$.

Es relativamente sencillo demostrar que la elección de $\sqrt{\log(n)}$ como la distancia entre dos niveles especiales consecutivos es optima en relación a todas las particiones equidistantes. Para cualquier elección de la distancia *s* entre dos niveles especiales consecutivos, el número de claves que cada usuario debe recordar es $O\left(s^2 \cdot (\log(n)/s) + (\log(n) \cdot (\log(n)/s))\right)$. La primera derivada de esta expresión (como función de *s*) es $\log(n) - \left(\log^2(n)/s^2\right)$, la cual es igual a 0 para $s = \sqrt{\log(n)}$. Dado que la segunda derivada es positiva, la elección de este valor reduce la complejidad del almacenamiento de este esquema.

3.2.1.4 Seguridad

Se dice que la seguridad de los algoritmos anteriores se ha roto si un usuario no privilegiado es capaz de descifrar el mensaje transmitido. Un adversario puede recolectar la información secreta de todos los usuarios revocados con la intención de romper el esquema. Pero aún teniendo en cuenta este tipo de ataque, resulta improbable que el atacante sea capaz de distinguir el cifrado de un texto plano de otro aleatorio, teniendo en cuenta que se haya hecho una elección correcta de los algoritmos F y E.

Para garantizar la seguridad, los algoritmos anteriores deben cumplir una propiedad llamada clave indistinguible. Quiere decir que la clave L del subconjunto no se puede distinguir de una clave aleatoria de cualquiera de los usuarios revocados.

Esta propiedad se cumple de forma clara para el caso del CS, ya que la clave se elige de forma aleatoria. Pero para el caso del SD, y de forma análoga para el LSD, la cosa no es tan evidente. Si el lector lo desea puede comprobar la demostración en el artículo [27].

3.2.2 Esquemas con estructura lineal

En este tipo de esquemas los usuarios se representan como puntos alineados y en orden. Además, reducen el tamaño de la transmisión hasta r. No obstante, para ello, precisan de gran capacidad de almacenamiento por parte de cada usuario. Estos esquemas de *Boadcast Encryption* se pueden consultar en [30].

La idea de estos esquemas es la siguiente, se define L como una línea con n puntos en ella, donde n es el número total de usuarios. En el esquema, a cada usuario se le identifica con un índice entero $k \in [0,n\text{-}1]$ y se representa con u_k en la línea L. El centro, primero asigna la clave de usuario $K(u_k)$ a cada usuario u_k. Considerando a L como el conjunto de n usuarios, se define $S_{(Scheme)}$ como el conjunto de todos los subconjuntos de L que cumplen una cierta condición para el esquema en cuestión.

El centro asigna a cada subconjunto $S \in S_{(Scheme)}$ una clave K, llamada clave del subconjunto S que puede ser obtenida por cualquier usuario privilegiado de S usando su clave privada. Para cada sesión, el centro encuentra los subconjuntos disjuntos S_1, S_2,…, S_m en $S_{(Scheme)}$, cuya unión contiene a todos los usuarios privilegiados, manteniendo m lo más pequeña posible. Para finalizar, el centro cifra la clave de sesión SK con la clave del subconjunto S_μ para cada μ=1,…, m. Estos m cifrados junto a la información de los S_μ's, forman la cabecera. El número m se suele definir como *Transmision Overhead* (*TO*).

3.2.2.1 Basic Chain Scheme (BCS)

Se definen u_0, u_1,…, u_{n-1} como los usuarios, en donde n es el número total de usuarios y r es el número de usuarios revocados. El intervalo se define a partir de los usuarios de los extremos u_i y u_j, se identifica con $I_{i,j}$. En el *Basic Chain Scheme*, $S_{(basic)}$ es el conjunto de todos estos $I_{i,j}$, donde i y j deben cumplir $0 \le i \le j \le n-1$. A cada intervalo $I_{i,j}$ se le asigna una clave de intervalo $K_{i,j}$ que será usada para el cifrado y el descifrado de la clave de sesión por los usuarios de $I_{i,j}$. El número de claves de usuario que debe almacenar un usuario u_k es $(k+1)\cdot(n-k)$ para k=0, 1, …, n-1 y por eso, el número medio de claves de usuario por usuario es $\frac{(n+1)\cdot(n+2)}{6}$, que es bastante elevado.

Generación de claves

Para reducir el número de claves que deben ser almacenadas, se introduce la permutación de sentido único $h:\{0,1\}^* \to \{0,1\}^*$. A continuación, se eligen n claves aleatorias $K_{0,0}, K_{1,1}, ...K_{n-1,n-1}$ y se construye una cadena de claves para cada $K_{i,i}$ del siguiente modo:

$$K_{i,i}, K_{i,i+1} = h(K_{i,i}), K_{i,i+2} = h(K_{i,i+1}) = h^2(K_{i,i}), ..., K_{i,n-1} = h^{n-1-i}(K_{i,i})$$

Cada usuario u_k deberá poseer el siguiente conjunto de claves para poder generar cualquier clave del intervalo al que pueda pertenecer:

$$K(u_k) = \{K_{0,k}, K_{1,k}, ...K_{k,k}\}$$

Para cada u_k, siendo $k = 0, 1, 2, ..., n-1$. La clave de intervalo $K_{i,j}$ sólo podrá ser calculada por aquellos u_k que cumplan $i \le k \le j$ y por ningún otro. Esta condición se debe cumplir incluso en el caso de que todos aquellos usuarios que no pertenezcan al intervalo cooperasen entre ellos para encontrar la clave.

Cifrado

Para cada sesión, el centro marca a los usuarios revocados en la línea L y posteriormente los elimina. Mediante este procedimiento obtiene los intervalos disjuntos llamados $I_{i_1,j_1},...,I_{i_m,j_m}$ formados por los usuarios privilegiados, tal y como se observa en la

Figura 22. La unión de estos intervalos contiene a todos los usuarios privilegiados. De este modo, el centro transmitirá lo siguiente:

$$\left\langle \left[(i_1, j_1), \ldots, (i_m, j_m); \varepsilon_{K_{i_1,j_1}}(SK), \ldots, \varepsilon_{K_{i_m,j_m}}(SK) \right]; \varepsilon_{SK}(M) \right\rangle$$

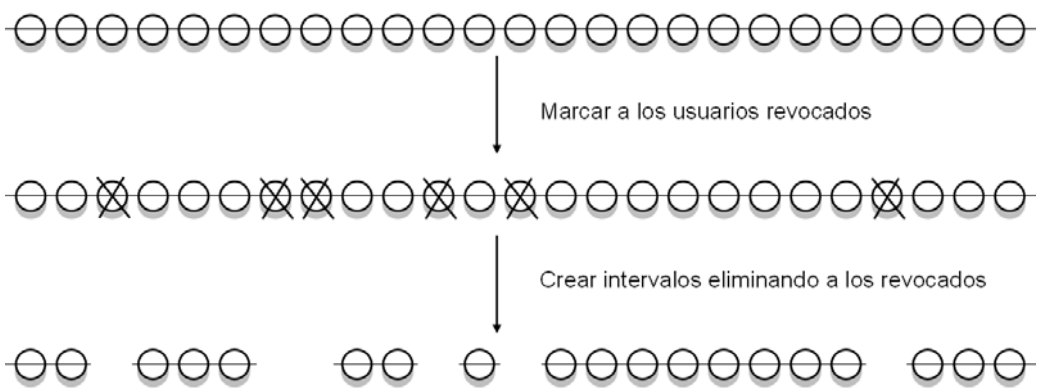

Figura 22: Creación de los intervalos

Descifrado

Una vez recibido el mensaje cifrado, cada usuario privilegiado u_k lo primero que debe hacer es localizar el intervalo $I_{i,j}$ al que pertenece, es decir, busca *i,j* tales que $i \le k \le j$ y a continuación calcular $K_{i,j} = h^{j-k}(K_{i,k})$ a partir de la clave $K_{i,k}$ que posee. Posteriormente u_k descifra $\varepsilon_{K_{i,j}}(SK)$ y $\varepsilon_{SK}(M)$ para obtener la clave de sesión *SK* y el mensaje *M* respectivamente. Se puede comprobar como un usuario revocado no puede computar la clave de sesión, ya que no pertenece a ningún intervalo de los de la cabecera.

Características

Cuando se revocan a *r* usuarios en el *Basic Chain Scheme*, el número máximo de posibles intervalos disjuntos en $S_{(\text{basic})}$ que contengan a todos los usuarios privilegiados es *r*+1, de este modo $TO_{(basic)} = r + 1$.

Cada usuario u_k deberá almacenar $(k+1)$ claves de usuario, por lo que el tamaño de almacenamiento en media es $\frac{n+1}{2}$ y en el peor de los casos $SS_{(basic)} = n$. El centro emisor deberá almacenar *n* claves $K_{0,0}, K_{1,1}, \ldots K_{n-1,n-1}$.

Finalmente, el coste de computación (CC) es como mucho *n*-1 computaciones de la permutación de único sentido *h*.

$$CC_{(basic)} = n-1$$

3.2.2.2 C-basic Chain Scheme (CBCS)

Aunque el *Basic Chain Scheme* reduce el *TO* hasta *r*, el tamaño de almacenamiento necesario para cada usuario es demasiado grande. No obstante, existe una forma de reducir el *SS* creando uniones de intervalos.

Se define *C* como un entero positivo predeterminado y $S_{(C\text{-basic})}$ como el conjunto de todos los intervalos de la forma $I_{i,j} \in S_{(\text{basic})}$ que cumplan $j-i+1 \le C$, donde $j-i+1$ es la longitud de $I_{i,j}$. A estos intervalos se les llama intervalos-*C*.

Generación de claves

La generación de claves del *C-basic Chain Scheme* es exactamente igual que la del *basic Chain Scheme* excepto en la longitud máxima de las cadenas de claves que en este caso es *C*. Por ejemplo, el centro construye la cadena de claves

$$K_{i,i}, K_{i,i+1} = h(K_{i,i}), K_{i,i+2} = h(K_{i,i+1}) = h^2(K_{i,i}), ..., K_{i,i+C-1} = h^{C-1}(K_{i,i})$$

para cada $0 \le i \le n-C$ y

$$K(u_k) \begin{cases} \{K_{k-C+1,k}, K_{k-C+2,k}, ..., K_{k,k}\} & si\ k \ge C \\ \{K_{1,k}, K_{2,k}, ..., K_{k,k}\} & en\ otro\ caso \end{cases}$$

para u_k, con $k = 0, 1, 2, ..., n-1$

Cifrado y descifrado

Para cada sesión, el centro encuentra los intervalos disjuntos como en el *Basic Chain Scheme*. Si todos los intervalos son de longitud menor o igual que *C*, usa los intervalos para el cifrado. Si la longitud de los intervalos es mayor que *C* entonces el centro emisor fragmenta los intervalos $I_{i,j}$ con $j-i+1 > C$ en subintervalos

$$I_{i,i+C-1}, I_{i+C,i+2C-1}, \ ..., I_{i+(q-1)C,i+qC-1}, I_{i+qC,j}$$

donde $q = \left\lfloor \frac{j-i+1}{C} \right\rfloor$ y el último subintervalo $I_{i+qC,j}$ debe ser excluido si $q = \frac{j-i+1}{C}$.

En otras palabras, el centro fragmenta $I_{i,j}$ en q intervalos de longitud C, empezando por u_i y el subintervalo número (q+1) formado por los usuarios restantes, si los hay. Este proceso se ilustra en la *Figura 23*.

Una vez el centro obtiene los intervalos-C $I_{i_1,j_1},...,I_{i_m,j_m} \in S_{(\text{C-basic})}$, la unión cubrirá a todos los usuarios privilegiados. El cifrado y el descifrado son idénticos a los del *Basic Chain Scheme*. Aquellos usuarios que no pertenezcan a ningún intervalo-C serán usuarios revocados y nunca podrán acceder a la clave de sesión *SK*.

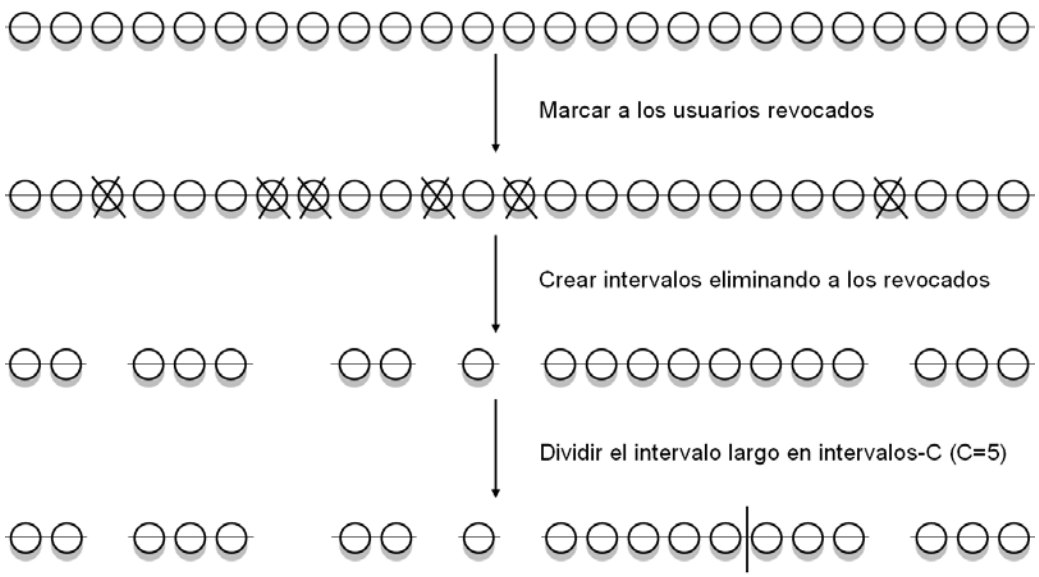

Figura 23: Creación de los intervalos-C

Características

En el esquema C-basic, cada usuario u_k necesita guardar como máximo C claves y por ello $SS_{(C-basic)} = C$, mientras que para el centro sigue siendo necesario almacenar n claves, $K_{0,0}, K_{1,1}, ...K_{n-1,n-1}$, donde n es el número total de usuarios.

El coste de computación es en el peor de los casos de C-1 computaciones, por lo que $CC_{(C-basic)} = C - 1$. Por otro lado, para n usuarios, r de ellos revocados el *Transmission Overhead* es:

$$TO_{(C-basic)} = r + \left\lceil \frac{n-2r}{C} \right\rceil$$

3.2.2.3 Seguridad

Un usuario u_k puede calcular la clave $K_{i,j}$ si y sólo si $u_k \in I_{i,j}$, en donde $i \le k \le j$. En el caso que $u_k \notin I_{i,j}$, esto significaría que $k < i$ o $j < k$. Si $k < i$, entonces u_k no podría calcular ningún valor de la siguiente cadena de claves:

$$K_{i,i}, K_{i,i+1} = h(K_{i,i}), K_{i,i+2} = h(K_{i,i+1}) = h^2(K_{i,i}), ..., K_{i,j} = h^{j-i}(K_{i,i})$$

Si $j < k$, entonces u_k no podrá calcular $K_{i,j}$ debido a que *h* es de único sentido.

La única forma de calcular la clave $K_{i,j}$ del intervalo $I_{i,j}$ es consiguiendo una de las claves de la siguiente cadena de claves: $K_{i,i}, K_{i,i+1}, K_{i,i+2}, ..., K_{i,j-1}, K_{i,j}$.

Para cada k, $i \le k \le j$, la clave $K_{i,k}$ sólo la pueden calcular los usuarios $u_i ... u_k$. Por eso, ningún otro usuario que no sea $u_i ... u_k$ podría calcular cualquiera de las claves de la cadena de claves anterior. Esto implica que aunque todos los usuarios revocados cooperen, éstos no podrían recuperar la clave de sesión.

3.2.2.4 Skipping Chain Scheme (SCS)

Antes de poder explicar el algoritmo *Skipping Chain Scheme* es necesario explicar el concepto de intervalos perforados. Primero de todo es necesario definir *p*, *c* y *C = lc* como enteros positivos, donde *l* es también un entero. *C* es la constante introducida en el *C-basic Chain Scheme*.

El intervalo-*c* perforado-*p* es un subconjunto de *c* o menos usuarios consecutivos empezando y acabando en un usuario privilegiado y a su vez, conteniendo *p* o menos usuarios revocados. $S_{(c,\ p-skip)}$ es el conjunto de todos los intervalos perforados-*p*.

$$S_{(C;c,\ p-skip)} = S_{(C-basic)} \bigcup S_{(c,\ p-skip)}$$

En cada sesión, los intervalos disjuntos en $S_{(C;c,\ p-skip)}$, que cubren a todos los usuarios privilegiados, se determinan bajo las siguientes normas:

El primer intervalo empieza por el primer usuario privilegiado de la izquierda.

Cada intervalo empieza y acaba en un usuario privilegiado.

Un intervalo con al menos un usuario revocado debe contener como mucho *c* usuarios, incluidos los *p* revocados.

Cada intervalo contiene el máximo número posible de usuarios, incluidos los revocados.

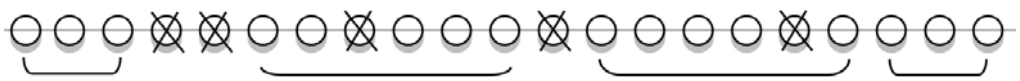

Figura 24: intervalos-6 perforados-1

La *Figura 24* muestra como se hacen los intervalos-*c* perforados-*p* con un ejemplo en donde *p*=1 y *c*=6.

Los intervalos en $S_{(C;c,\ p-skip)}$ empezando por u_i y acabando en u_j con $u_{x_1},...,u_{x_q}$ usuarios revocados se denota con $I_{i,j;x_1,...,x_q}$ o $I_{i,j;X}$ donde $X=\{x_1,...,x_q\}$ y donde:

$$\begin{cases} 1 \le j-i+1 \le C & si\ X=0, o\ equivalentemente\ q=0 \\ 1 \le j-i+1 \le c & en\ otro\ caso \end{cases}$$

$0 \le q \le p$ y $i < x_1 < ... < x_q < j$ si $q \neq 0$. Cuando $X=0$, se escribe $I_{i,j}$.

Una vez comprendido el concepto de intervalos-*c* perforados-*p*, se propone el esquema *Skipping Chain* con los parámetros *C*, *c* y *p* para el *Broadcast Encryption*. Éste se denota con (*C*; *c*, *p*-skip). En el esquema *Skipping Chain*, se asigna una única clave por intervalo en $S_{(C;c,\ p-skip)}$, esta clave sólo podrá ser derivada por los usuarios que pertenezcan al intervalo en cuestión. Para conseguirlo, se debe construir una cadena de claves que tenga en cuanta a los usuarios revocados.

Generación de claves

Siendo $h_t : \{0,1\}^* \to \{0,1\}^*$ es una permutación de único sentido para cada $t=1,2,...,p$. Para asignar una clave para cada intervalo en $S_{(C;c,\ p-skip)}$, se eligen *n* claves $K_{1,1}, K_{2,2}, ...K_{n,n}$, de forma aleatoria, para luego entregarlas a $u_1,u_2,...,u_n$, respectivamente. Partiendo de cada $K_{i,i}$, se construyen las cadenas de claves para todos los intervalos posibles en $S_{(C;c,\ p-skip)}$ empezando por u_i. Se define $I \in S_{(C;c,\ p-skip)}$ como un intervalo. La cadena de claves para *I* se construye de forma inductiva siguiendo las siguientes reglas:

- La cadena empieza por $K_{i,i}$
- Para cualquier usuario privilegiado $u_k \in I$, si el usuario $u_{k+1} \in I$, implicará que también es un usuario privilegiado, por lo que sólo se aplica h_0 en la clave de u_k para obtener la clave de u_{k+1} .
- Si se revocan a los siguientes t usuarios y el usuario $u_{k+t+1} \in I$ es un usuario privilegiado, entonces se saltan a estos usuarios revocados y se aplica h_t a la clave de u_k para obtener la clave de u_{k+t+1}, donde $1 \le t \le p$.

La *Figura 25* muestra como se construye la cadena de claves de un determinado intervalo perforado (con p=10, c=20).

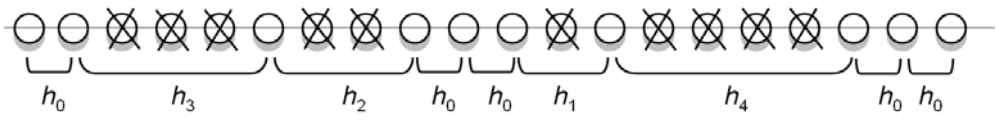

Figura 25: Cadena de claves de intervalos -20 perforados-10

En la cadena de claves, para $I = I_{i,j;x_1,\ldots,x_q}$, la clave de un usuario privilegiado $u_k \in I$ se denota con $K_{i,k;\,x_1,\ldots,x_t}$, donde $i < x_1 < \ldots < x_t < k < x_{t+1} < \ldots < x_q$ y $0 \le t \le q \le p$. Cuando t=0, simplemente se escribe $K_{i,k}$.

Por ejemplo:

$$K_{5,11} = h_0^6\left(K_{5,5}\right);\ K_{5,11;\,7} = h_0^3 h_1 h_0\left(K_{5,5}\right);\ K_{4,11;\,5,6,7,9,10} = h_2 h_3\left(K_{4,4}\right);\ K_{3,11;\,4,5,7,8} = h_0^2 h_2^2\left(K_{3,3}\right);$$
$$K_{3,11;\,4,5,6,7,9} = h_0 h_1 h_4\left(K_{3,3}\right);\ldots$$

El centro asigna estas claves a los usuarios, por lo que el usuario u_k recibe su clave de usuario $K\left(u_k\right)$ que consta de todas las posibles $K_{i,k;\,x_1,\ldots,x_t}$, donde

$$\begin{cases} i < x_1 < x_2 < \ldots < x_t < k \quad y \quad 3 \le k - i + 1 \le c \quad si\ 0 < t \le p \\ i \le p \quad y \quad 1 \le k - i + 1 \le C \end{cases}$$

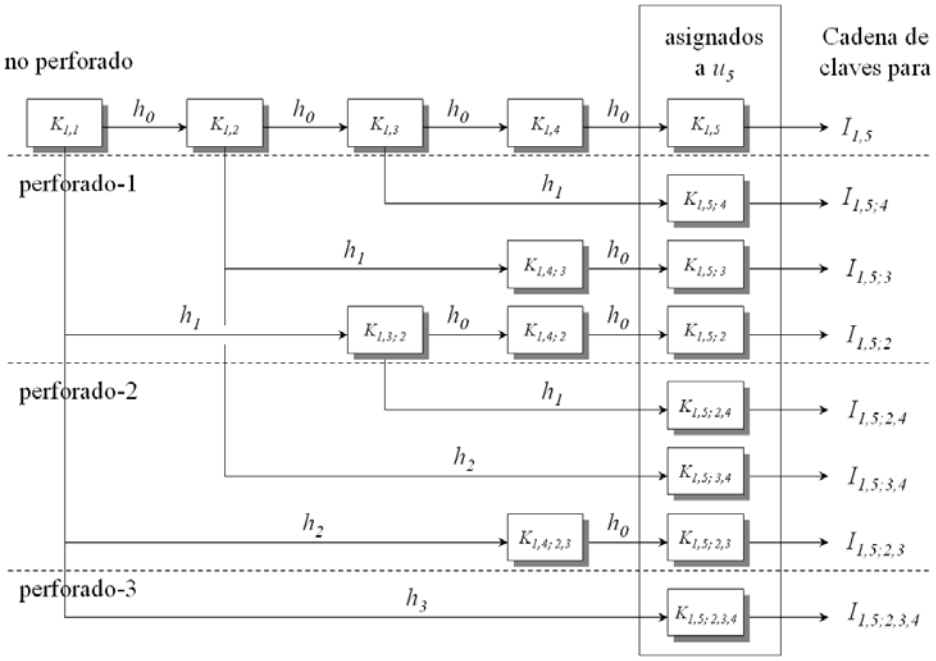

Figura 26: Cadena de claves de único sentido desde $K_{1,1}$ con p=3, c=C=5

La *Figura 26* muestra la asignación de claves de u_5 en el *Skipping Chain Scheme* con p=3 y c=C=5.

Cifrado

Para cada sesión, el centro divide L en intervalos disjuntos $I_1, I_2, ..., I_m \in S_{(C;c,\ p-skip)}$, cuya unión contiene a todos los usuarios privilegiados. Este procedimiento se realiza siguiendo las reglas descritas en la explicación de los intervalos perforados. Se define $I = I_{i,j;x_1,...,x_q}$ como una de las I_μ. La última clave $K_{i,j;\,x_1,...,x_q}$ de la cadena de claves para I es la clave del intervalo de I. Consecuentemente, se denota la clave del intervalo de I_μ con K_μ para cada $\mu = 1, 2, ..., m$. Posteriormente el centro transmite:

$$\left\langle \left[\text{info}_1, \text{info}_2, ..., \text{info}_\text{m}; \varepsilon_{K_1}(SK), \varepsilon_{K_2}(SK), ..., \varepsilon_{K_m}(SK) \right]; \varepsilon_{SK}(M) \right\rangle$$

donde info_μ es información de I_μ. La información de $I = I_{i,j;x_1,...,x_q}$ consiste en $i, \gamma_0, \gamma_1, ..., \gamma_q$ donde $\gamma_0 = j - i + 1$ y $\gamma_t = x_t - i$ para cada $t = 1, 2, ..., q$. La posición inicial i se puede representar por $\log n$ bits y las γ's por como mucho $\log C$ bits. Por lo que el tamaño de

toda la info$_\mu$'s es $m(\log n+(p+1)\log C)$, el cual se ignorará cuando se compute el *TO*, ya que es pequeño comparado con el tamaño de todos los $\varepsilon_{K_\mu}(SK)$.

Descifrado

Una vez se recibe el mensaje cifrado, cada usuario privilegiado u_k busca el intervalo al cual pertenece usando la información de la cabecera. Siendo el intervalo $I=I_{i,j;x_1,...,x_q}\in S_{(C;c,p-skip)}$, donde $i\le k\le j$ y $k\neq x_1,...,x_q$. El usuario u_k podrá obtener $K_{i,j;\,x_1,...,x_q}$ de la siguiente forma:

Encontrar t para la cual $x_t<k<x_{t+1}$, donde $0\le t\le q$. Si $t=0$ y $t=q$, significa que no hay ningún usuario revocado antes ni después de u_k, respectivamente.

Elegir $K_{i,k;\,x_1,...,x_t}$ de la clave de usuarios asignada.

Empezando por $K_{i,k;\,x_1,...,x_t}$, se aplica la permutación de único sentido h_i bajo la norma descrita en la generación de claves, hasta que el segundo suscriptor llegue a *j*.

La clave resultante es $K_{i,j;\,x_1,...,x_q}$.

Con esto, u_k es capaz de descifrar $\varepsilon_{K_{i,j;x_1,...,x_q}}(SK)$ y $\varepsilon_{SK}(M)$, para así obtener la clave de sesión *SK* y el mensaje *M* respectivamente.

Características

A continuación se van a comentar los valores de los parámetros TO, CC y SS del *Skipping Chain Scheme*, en donde *C*=*lc*, *l*≥2 y *c*≥4. No obstante, no se van a justificar dichos valores. En el caso de que el lector lo desee, puede consultar la justificación en [30].

En general,

$$TO_{(C;c,p-skip)}=\begin{cases}\left(1-\dfrac{c}{C}\right)r+\dfrac{N}{C} & si\ 0\le r\le\dfrac{N}{c}\\ \left(\dfrac{1}{p+1}-\dfrac{p(p+2)}{D_p}\right)r+\dfrac{p(p+1)}{D_p}N & si\ \dfrac{N}{c}\le r\le\dfrac{(p+1)N}{p+2}\end{cases}$$

donde $D_p=(p+1)^2c-(p+1)(p+2)$.

El coste computacional es fácil de obtener y es como mucho de $C-1$ cálculos de la permutación de único sentido.

$$CC_{(C;c,p-skip)} = C - 1$$

El espacio de almacenamiento necesario para cada usuario es independiente de N y de r, como se puede ver a continuación:

$$SS_{(C;c,p-skip)} = \sum_{k=1}^{p} \binom{c-1}{k+1} + C$$

El esquema *Skipping Chain* obtiene unos resultados muy buenos en términos de *TO* cuando *r* es grande. No obstante, cuando *r* es muy pequeño, éste presenta un *TO* superior al del SD. Esto se debe a que los intervalos grandes (>*C*) constituidos sólo por usuarios privilegiados, requiere de múltiples intervalos en $S_{(C;c,\ p-skip)}$ para cubrir a todos los usuarios privilegiados. En realidad, el esquema *C-basic Chain* tiene el mismo problema.

Existe un esquema llamado *Cascade Chain* que resuelve este problema introduciendo unas estructuras de capas y cadenas de claves en cascada. El esquema *Cascade Chain* también está basado en el esquema *C-basic Chain*, pero a diferencia de éste, reduce de forma considerable el *TO* cuando *r* es muy pequeño.

Dado el aumento considerable de complejidad de este algoritmo respecto a los tres anteriores y que su explicación podría resultar demasiado extensa, se ha optado por no introducirlo en el proyecto. No obstante, si el lector lo desea puede consultar los detalles sobre este esquema, así como de otro llamado *Combined Chain Scheme* en [30].

3.3 Resumen comparativo de los BES estudiados

En la

Tabla 2 se muestra de forma resumida las características de eficiencia de los esquemas mencionados en este tema. Como se ha podido ver, los aspectos más importantes a la hora de evaluar la eficiencia de los esquemas de *Broadcast Encryption* son: la longitud del mensaje (TO), el espacio de almacenamiento de claves en el receptor (SS), el espacio necesario para el almacenamiento de claves en el centro emisor y el coste computacional (CC) tanto por parte del receptor como del centro emisor.

En la siguiente tabla sólo se muestran los valores que hacen referencia a la parte del receptor. Esto es así ya que a pesar de que los valores de la parte del centro emisor son interesantes, éstos son menos importantes que los de la parte del receptor.

Método	Transmission Overhead (TO)	Espacio de almacenamiento (SS)	Coste Computacional (CC)
Complete Subtree (CS)	$O\left(r\log\left(\frac{n}{r}\right)\right)$	$O(\log(n))$	$O(\log(\log(n)))$
Subset Difference (SD)	$O(r)$	$O\left(\frac{1}{2}\log^2(n)\right)$	$O(\log(n))$
Layer Subset Difference (LSD)	$O(r)$	$O(\log^{3/2}(n))$	$O(\log(n))$
Basic Chain Scheme (BCS)	$O(r)$	$O(n)$	$O(n)$
C-basic Chain Scheme (CBCS)	$O\left(r+\left\lceil\frac{n-2r}{C}\right\rceil\right)$	$O(C)$	$O(C)$
Skipping Chain Scheme (SCS)	(*)	$\sum_{k=1}^{p}\binom{c-1}{k+1}+C$	$O(C)$

Tabla 2: Cuadro resumen de los esquemas de BE

$$(*)\begin{cases} O\left(\left(1-\frac{c}{C}\right)r+\frac{N}{C}\right) & si\ 0\le r\le\frac{N}{c} \\ O\left(\left(\frac{1}{p+1}-\frac{p(p+2)}{D_p}\right)r+\frac{p(p+1)}{D_p}N\right) & si\ \frac{N}{c}\le r\le\frac{(p+1)N}{p+2} \end{cases}$$

4 Análisis

4.1 Seguridad de los CAS

4.1.1 Servicios de seguridad

Para hacer frente a las amenazas de seguridad de un sistema, hay definidos una serie de servicios para proteger los sistemas de procesado de datos y de transferencia de información. Estos servicios hacen uso de uno o varios mecanismos de seguridad. Una clasificación útil de los servicios de seguridad que proporcionan los Sistemas de Acceso Condicional, es la siguiente: confidencialidad, autenticación, integridad y control de acceso. No existe un único mecanismo capaz de proveer todos los servicios citados, pero la mayoría de ellos hacen uso de técnicas criptográficas basadas en el cifrado de la información.

A continuación se va a dar una pequeña definición de lo que se entiende por cada uno de estos servicios de seguridad y además se van a determinar los elementos de los Sistemas de Acceso Condicional que se encargan de proporcionar cada uno de los servicios.

Confidencialidad

Requiere que la información sea accesible únicamente por los usuarios autorizados. En algunos casos, no sólo hay que proteger el contenido de los mensajes, confidencialidad del mensaje, sino también las identidades de los receptores, confidencialidad del tráfico de mensajes.

En los Sistemas de Acceso Condicional de televisión de pago, debido a que la información se transmite por un medio al que todo el mundo tiene acceso, ésta se debe proteger de alguna forma. Esta forma es mediante la codificación y el cifrado. La codificación permite que el contenido transmitido sólo pueda ser visionado por aquellos que dispongan de la clave de codificación/descodificación (CW). Los algoritmos de cifrado son muchos y varían en función del Sistema de Acceso Condicional. No obstante, el cifrado es el encargado de garantizan la confidencialidad de los mensajes que contienen las claves simétricas transmitidas (CW y AK).

El cifrado garantiza que la información no sea inteligible para individuos, entidades o procesos no autorizados. Consiste en transformar un texto en claro mediante un proceso de cifrado en un texto cifrado, gracias a una información secreta o clave de cifrado. Cuando se

emplea la misma clave en las operaciones de cifrado y descifrado, como es el caso de la codificación y cifrado de las señales de televisión de pago, se dice que el criptosistema es simétrico. Estos sistemas son mucho más rápidos que los de clave pública, resultando apropiados para funciones de cifrado de grandes volúmenes de datos.

Cuando se utiliza una pareja de claves para los procesos de cifrado y descifrado, se dice que el criptosistema es asimétrico o de clave pública. Una clave, la privada, se mantiene secreta, mientras que la segunda clave, la pública, puede ser conocida por todos. De forma general, las claves públicas se utilizan para cifrar y las privadas, para descifrar. El sistema tiene la propiedad de que a partir del conocimiento de la clave pública no es posible determinar la clave privada.

Autenticación

Requiere una identificación correcta del origen del mensaje, asegurando que la entidad no es falsa. Se distinguen dos tipos: de entidad, que asegura la identidad de las entidades participantes en la comunicación y de origen de información, que asegura que una unidad de información proviene de cierta entidad, siendo la firma digital el mecanismo más extendido.

El mecanismo de la firma digital implica el cifrado, por medio de la clave secreta del emisor, de una cadena comprimida de datos que se va a transferir. La firma digital se envía junto con los datos ordinarios. Este mensaje se procesa en el receptor, para verificar su integridad.

Para proteger la autenticidad de un mensaje de acceso condicional, se puede usar el esquema de firma digital basado en criptografía de clave pública. El hecho de utilizar la firma digital permite poder validar la autenticidad del origen de la información y lo único que hay que hacer, es proteger la autenticidad de la clave pública del proveedor de servicios. De hecho, en los sistemas de acceso condicional, la única entidad que está capacitada de generar la firma de un mensaje es el proveedor de televisión de pago, por lo que se deberá almacenar su clave pública en la ROM de la *smart card*.

En realidad, el proveedor de televisión de pago puede usar claves diferentes en función del tipo de mensaje. En este caso, en la *smart card* se deberán almacenar tantas claves públicas como claves use el proveedor para las firmas digitales de los mensajes.

De este modo, cuando a un usuario le llega un mensaje, coge la firma digital de éste y la descifra. En el caso que sea correcta, quiere decir que el mensaje ha sido emitido por el proveedor de televisión de pago. Por otro lado, si no es correcta, quiere decir que una entidad maliciosa ha inyectado la información y no se debe proceder con el descifrado del mensaje.

Integridad

Requiere que la información sólo pueda ser modificada por las entidades autorizadas. La modificación incluye escritura, cambio, borrado, creación y reactuación de los mensajes transmitidos. La integridad de datos asegura que los datos recibidos no han sido modificados de ninguna manera, por ejemplo mediante una función hash o con una firma, mientras que la integridad de secuencia de datos asegura que la secuencia de los bloques o unidades de datos recibidas no ha sido alterada y que no hay unidades repetidas o perdidas, por ejemplo mediante *time-stamps* (sellos de tiempo).

Dado que los mensajes del sistema de acceso condicional están firmados, se puede decir que la integridad de éstos está garantizada, ya que la firma digital consiste en el resultado de introducir el contenido del mensaje en una función hash y posteriormente cifrarlo mediante la clave privada de la entidad, tal y como se ha mencionado anteriormente. Cuando un receptor recibe el mensaje, lo único que debe hacer es procesar la información tal y como lo ha hecho anteriormente el emisor. Si después de descifrar la firma mediante la clave pública coinciden, se puede decir que los datos no han sido modificados. De modo que, por lo visto hasta el momento, la firma digital de los mensajes proporciona tanto autenticidad como integridad.

Control de acceso

Requiere que el acceso a los recursos (información, capacidad de cálculo, nodos de comunicaciones, entidades físicas, etc.) sea controlado y limitado por el sistema destino, mediante el uso de contraseñas o llaves hardware, por ejemplo, protegiéndolos frente a usos no autorizados o manipulación.

Los elementos que proporcionan el control de acceso en los sistemas de acceso condicional son las tres claves. Empezando por arriba, la MPK (*Master Private Key*), almacenada en la *smart card*, es la clave mediante la cual se puede obtener la AK (*Authorization Key*), en caso de estar autorizado. A su vez, mediante la AK se puede obtener la CW (*Control Word*), que a su vez, mediante ésta se puede acceder al contenido difundido.

4.1.2 Clasificación de los mecanismos de seguridad

Los mecanismos básicos pueden agruparse de varias formas para proporcionar los servicios previamente mencionados. Conviene resaltar que los mecanismos poseen tres componentes principales:

- Una información secreta, como claves y contraseñas, conocidas por las entidades autorizadas.
- Un conjunto de algoritmos, para llevar a cabo el cifrado, descifrado, hash y generación de números aleatorios.
- Un conjunto de procedimientos, que definen cómo se usarán los algoritmos, quién envía qué a quién y cuándo.

Asimismo es importante notar que los sistemas de seguridad requieren una gestión de seguridad. La gestión comprende dos campos bien amplios:

Seguridad en la generación, localización y distribución de la información secreta, de modo que sólo pueda ser accedida por aquellas entidades autorizadas. Hasta hace relativamente poco tiempo, los proveedores de CAS centraban todos sus esfuerzos en este campo, pero su incapacidad de tener éxito, les ha obligado a buscar la solución en el siguiente campo.

La política de los servicios y mecanismos de seguridad para detectar infracciones de seguridad y emprender acciones correctivas. Se trata de una vertiente en fase de desarrollo en cuanto a su uso en los CAS. No obstante, la aparición de múltiples mecanismos como ahora el "Watermarking", el "Fingerprinting" o el "Traitor tracing" entre otros, hace presagiar que no nos encontramos demasiado lejos de que estos mecanismos empiecen a dar sus frutos en un futuro próximo.

A pesar de que el tema política de servicios y mecanismos de seguridad para detectar infracciones de seguridad es muy interesante, este proyecto se centra más en el campo anterior y se dejará éste de lado, a pesar de que ya existen estudios/aplicaciones de Traitor tracing basadas en *Broadcast Encryption*.

4.1.3 Algoritmos criptográficos usados en la distribución de claves

En este apartado se va a hablar sobre algunos de los algoritmos criptográficos más usados en lo referente a la distribución de claves. El objetivo de este tema no es proporcionar información detallada sobre el funcionamiento de estos algoritmos. Por este motivo, si el lector desea, puede encontrar los detalles del funcionamiento de los algoritmos citados en [Anexo 9.1]. Lo que se pretende mediante este apartado es determinar para que se usan estos algoritmos, cuando se usan y un poco sus puntos fuertes.

Antes de empezar a hablar sobre los algoritmos criptográficos, se debe comentar que estos se pueden clasificar en dos grandes grupos: los simétricos y los asimétricos. Se dice que un algoritmo es simétrico o asimétrico en función de si la clave de cifrado es igual a la de descifrado o no. Los algoritmos simétricos utilizan la misma clave tanto para cifrar como para descifrar. Por otro lado, los algoritmos asimétricos utilizan una clave para cifrar y otra diferente para descifrar.

Tanto unos como los otros poseen ventajas y desventajas. Es por este motivo, por el cual será más adecuado usar un algoritmo u otro en función de las circunstancias. Así a grandes rasgos, las principales ventajas de los algoritmos simétricos son la seguridad y la velocidad. Por otro lado, la principal ventaja de los algoritmos asimétricos es que no poseen el problema de cómo hacer llegar la clave a la otra parte implicada.

Los algoritmos asimétricos, al disponer de una pareja de claves, una para cifrar y otra para descifrar, lo que hacen es mantener en secreto la clave de descifrado mientras se hace pública la clave de cifrado. Por este motivo, a esta clave se la llama clave pública. De este modo, todo aquel que desee enviar información de forma segura cifrará el mensaje con esta clave y solo aquel que disponga de la clave secreta (clave de descifrado) será capaz de obtener el mensaje en claro.

DES (Data Encryption Standard)

Las principales ventajas del algoritmo DES son que el cifrado y descifrado son idénticos, su ejecución es con subclaves en orden inverso. Además es muy fácil de implementar, permitiendo un ahorro de área (en hardware). Por otro lado es relativamente rápido y “seguro” bajo antiguos estándares.

Sus principales desventajas son por un lado que posee algunas propiedades curiosas como su lentitud bajo estándares actuales (especialmente en software). Por ejemplo, las operaciones a nivel bit (Expansión), las operaciones con “semi-registros” (XOR con subkey),

las operaciones con "semi-bytes" (Funciones S) y las permutaciones inicial y final limitadas a bloques de 64 bits. Además, el tamaño de clave es insuficiente, 2^{56} posibilidades, en base 10 son 72.057.594.037.927.936 posibles claves ($7{,}2 \cdot 10^{16}$). Mediante un ataque de fuerza bruta serían necesarias, en media, probar con 2^{55} claves, o lo que es lo mismo, probar $3{,}6 \cdot 10^{16}$ claves.

Algunos hechos interesantes de este algoritmo son que en 1977 se propuso a nivel teórico, un equipo para romper DES en un día. El coste del equipo se estimó en 20 millones de dólares aproximadamente. Más adelante en 1993, Wiener propuso otro equipo teórico que sería capaz de descifrar un mensaje en 7 horas, el coste aproximado de éste estaba estimado en 1 millón de dólares. No fue hasta el 1998 cuando Electronic Frontier Foundation (EFF) diseñó un equipo con hardware especializado: 1536 ASICs, con un coste de 250.000 dólares que rompió el DES en 2 días. De la última proeza que se tiene constancia es del 2007 en las Universidades de Bochum y Kiel, las cuales diseñaron un equipo hardware altamente flexible: 120 FPGAs (Xilinx Spartan3), con un coste de únicamente 10.000 dólares. Este equipo es capaz de romper el DES en 6,4 días en media.

Dado que a día de hoy el algoritmo DES ha dejado de ser "seguro" los CAS han dejado de usarlo en el cifrado de las claves que se distribuyen en los mensajes ECM y EMM. A pesar de que seguiría siendo útil en el cifrado de la CW del mensaje ECM, ya que ésta tiene un periodo de vida de escasos segundos. Eso sí, siempre y cuando la AK variase cada relativamente poco tiempo, ya que un atacante podría centrar su ataque en buscar la clave (AK) que descifra la CW y no buscar la propia CW. De este modo, una vez conseguida la AK que descifra la CW, el atacante sería capaz de visualizar el contenido hasta el siguiente cambio de AK. Por eso, si AK cambia con mucha frecuencia el ataque no dará sus frutos, ya que cuando el atacante consiga la clave con la que se cifró la CW, ésta ya no estará vigente, es decir, habrá cambiado.

El hecho que la AK cambie cada mucho tiempo provoca que el cifrado mediante DES sea poco recomendable. No obstante, la evolución de este algoritmo, el Triple DES, que apareció con el fin de hacer el espacio de claves superior, sí que se presenta como una opción válida.

Triple DES (3DES, TDES, 2DES...)

Cuando se descubrió que una clave de 56 bits no era suficiente para evitar un ataque de fuerza bruta, el Triple DES fue elegido como forma de agrandar el largo de la clave sin necesidad de cambiar de algoritmo de cifrado. Este método de cifrado es inmune al ataque por

encuentro a medio camino. Además, permite trabajar de tres formas distintas, proporcionando la posibilidad de trabajar con tres longitudes de clave efectivas diferentes (56/112/168 bits).

Este algoritmo se basa en la concatenación de tres módulos del algoritmo DES básico. De este modo se puede usar una clave para cada módulo. A estas claves las llamaré k_1, k_2 y k_3. Cada una de estas claves debe tener una longitud efectiva de 56 bits. El procedimiento es el siguiente: el primer módulo cifra el mensaje con la clave k_1, el segundo módulo descifra el mensaje cifrado con la clave k_2 y finalmente el tercer módulo lo vuelve a cifrar con k_3.

Como he comentado antes, existen 3 modos de funcionamiento. Estos 3 modos están definidos en función del uso de las claves. Se le llama TDES o 3DES cuando $k_1 \neq k_2 \neq k_3$, cosa que hace que longitud efectiva de la clave sea de 168 bits (56 bits de k_1+56 bits de k_2+56 bits de k_3). Si $k_1 = k_3 \neq k_2$, entonces se le llama 2DES y la longitud efectiva es de 112 bits (56 bits de k_1 +56 bits de k_2). Por otro lado, si se cumple $k_1 = k_2 = k_3$, entonces estamos frente al DES clásico, ya que el cifrado del primer módulo se compensa con el descifrado del segundo. La longitud efectiva de la clave de este modo es igual a la del DES, 56 bits.

Tanto el hecho de trabajar en modo 2DES como en modo 3DES en los CAS, permiten que las claves cifradas transmitidas en los mensajes EMM y ECM estén protegidas para su periodo de vida (tiempo durante el cual una clave se mantiene sin variaciones). Su principal inconveniente es que para realizar un cifrado/descifrado de este tipo, es necesario el triple de tiempo que para el cifrado/descifrado mediante DES.

IDEA (Internacional Data Encryption Algorithm)

También se trata de un algoritmo de cifrado en bloque y simétrico como el DES, por lo que utiliza la misma clave tanto para el cifrado como para el descifrado. Este algoritmo tuvo una gran acogida entre los profesionales de la seguridad, no tanta como en su día el DES, pero la utilización de una clave de 128 bits le ayudó a tener muchos usuarios. El IDEA tuvo especial relevancia en aplicaciones como el cifrado de correos electrónicos.

El ataque de fuerza bruta puro y duro a este algoritmo resulta impracticable, ya que como bien se sabe el espacio de claves de este algoritmo es de 128 bits. O lo que es lo mismo en base 10, 340.282.366.920.938.463.463.374.607.431.768.211.456 claves ($\sim 3{,}4 \cdot 10^{38}$).

Para recuperar una clave del espacio de claves mediante fuerza bruta, en media se debe buscar para la mitad del espacio de claves, es decir, para 127 bits. O lo que es lo mismo en base 10, para 170.141.183.460.469.231.731.687.303.715.884.105.728 claves ($\sim 1{,}7 \cdot 10^{38}$).

Después de ver los números que se manejan del espacio de claves, resulta obvio que con la capacidad de cálculo actual, el tiempo necesario para encontrar las claves que se

utilizan en los CAS en el caso que se utilizase este algoritmo para cifrarlas sería mucho mayor al periodo de vida de las propias claves. Por lo que el algoritmo de cifrado IDEA se considera válido, igual que el Triple DES, para su uso en el cifrado de las claves de los CAS.

AES (Advanced Encryption Standard)

El AES también es un algoritmo de cifrado simétrico y por bloques. Se trata del algoritmo más moderno de los mencionados en el proyecto. Está llamado a ser el sustituto del DES, en cuanto al volumen de su uso. En muchas aplicaciones ya está sustituyendo al Triple DES.

Las principales ventajas son por un lado, que la longitud del bloque de entrada es más grande que el de los algoritmos DES e IDEA, pasa de ser 64 a 128 bits. Además está diseñado para realizar más operaciones de forma paralela a diferentes niveles. Las operaciones se manejan a nivel de bytes, cosa que es altamente eficiente en la mayoría de plataformas, especialmente a nivel de software. Por otro lado, presenta una gran flexibilidad en el nivel de seguridad, ya que permite claves entre 128/192/256 bits.

Como desventaja destacable se encuentran que el cifrado y descifrado no son idénticas: usan funciones LUT inversas y diferentes operaciones MixColumns.

El hecho que en los CAS los receptores sólo tengan que descifrar los mensajes (claves) cifrados, provoca que la desventaja que se ha comentado, el proceso de cifrado y descifrados no son idénticos, no influya demasiado, ya que la principal tarea de los receptores es la de descifrar y no la de cifrar. También el AES se presenta como un algoritmo válido para el cifrado de las claves, ya que permite la posibilidad de proporcionar mayor seguridad, permitiendo longitudes de clave superiores a las de los otros algoritmos. Pero no es esta característica la que lo hace más apto, ya que la seguridad que proporcionan los otros ya es suficiente, sino que se trata de un algoritmo más eficiente y rápido que permite sobrecargar a la *smart card* de los receptores en menor medida que los otros algoritmos mencionados.

RSA (Rivest Shamir Adleman)

El RSA tiene este nombre porque sus creadores lo bautizaron con sus apellidos. Se trata de un algoritmo de cifrado por bloques asimétrico. La seguridad de este algoritmo reside en la dificultad a la hora de factorizar números muy grandes. A día de hoy, la factorización se presenta como un aspecto complicado de tratar, pero puede que el día de mañana se encuentre una forma de factorizar números grandes y entonces dejará de ser seguro. Además, hay 3 cosas que ya están sucediendo y continuarán pasando que afectan a este tema que son: están

apareciendo avances en la técnica de factorización, el poder de cálculo de los equipos está aumentando y el coste de estos equipos está disminuyendo. Hay que tener en cuenta que el hecho que la potencia de cálculo es un arma de doble filo, ya que a pesar de que sirve para el ataque de este algoritmo, también sirve para crear números primos más grandes.

El ataque de fuerza bruta de este algoritmo se basa en la factorización. Un atacante tiene acceso al la clave pública, o lo que es lo mismo, el atacante posee *e* y *n*. El atacante lo que busca es la clave privada *d*. Para obtener *d*, se tiene que factorizar *n*. Existen un gran número de algoritmos diseñados para la factorización, algunos de ellos muy eficientes que han permitido factorizar números de hasta 512 bits en 5 meses con múltiples máquinas trabajando de forma distribuida. Aunque se dice que mediante un coste de un millón de dólares aproximadamente, se es capaz de fatorizar un número de 1024 bits en un año.

A continuación se muestra una tabla donde se detalla, en función de si el algoritmo utilizado es simétrico o asimétrico, la longitud de las claves necesarias para proporcionar el mismo nivel de seguridad. Este cálculo se ha realizado para el caso que se use el algoritmo NFS (Number Field Sieve) para la factorización, uno de los más rápidos.

Simétrico	Asimétrico
56 bits	384 bits
64 bits	512 bits
80 bits	768 bits
112 bits	1792 bits
128 bits	2304 bits

Tabla 3: Equivalencia comparativa de longitud de clave entre algoritmos (seguridad)

A pesar de que el algoritmo RSA es especialmente útil para el inetrcambio de claves, dada su condición de algoritmo de clave asimétrica, su aplicación con este fin en los CAS no resulta tan práctica. Los CAS basan su funcionamiento en una jerarquía de claves y es esta jerarquía de claves la que permite la distribución de las claves de niveles más bajos sin necesidad de algoritmos de cifrado asimétricos, ya que hay una clave que se mantiene siempre constante, la MPK. Lo que quiero decir con esto es que el uso del RSA no es necesario para la distribución de claves, pero que no sea necesario no quiere decir que no se pueda usar. Claro que se puede usar, pero dadas sus características resulta poco recomendable, ya que las operaciones que debe desempeñar este algoritmo son costosas (a nivel computacional) y lentas.

A pesar de no ser demasiado práctico para la distribución de claves en los CAS, el RSA resulta sumamente útil en el tema de las firmas digitales de los mensajes. Se trata de una tarea que sólo la puede desempeñar este algoritmo de entre los citados en este apartado. De este modo, en los CAS, el algoritmo RSA se usa para firmar los mensajes, es decir, se cifra la función resumen del mensaje mediante el algoritmo RSA con la clave privada del centro emisor. Así, los receptores serán capaces de comprobar la integridad de los datos y autenticidad del origen, descifrando la función resumen con la clave pública y comprobándola con la función resumen generada por el receptor.

Tipos de seguridad

Dado que se ha estado hablando de seguridad en los algoritmos anteriores, se ha considerado oportuno proporcionar a continuación una breve explicación de los siguientes tipos de seguridad. Antes de todo, hay que decir que la seguridad de un sistema criptográfico depende de un parámetro denominado fuerza, que define el grado de dificultad de romper un sistema.

Se dice que un algoritmo proporciona *seguridad incondicional* cuando un criptosistema es seguro con independencia de los medios disponibles (infinita potencia de cálculo). Si un criptosistema es incondicionalmente seguro no se puede romper. Se demostró la existencia de criptosistemas de seguridad incondicional, por ejemplo el cifrado de Vernann, aunque en la práctica no tienen utilidad, ya que la longitud de la clave es tan larga como el propio mensaje.

Los criptosistemas más modernos se basan en la *seguridad computacional*. La idea es considerar un criptosistema computacionalmente seguro cuando aunque haya un algoritmo que rompa el sistema éste requiera un tiempo de computación tan grande que sea inviable llevarlo a la práctica. Un claro ejemplo de estos algoritmos son los de clave pública basados en problemas de alta complejidad de cálculo, como es el caso del RSA.

El término de *seguridad probable* hace referencia a aquellos sistemas que si se pude demostrar que existe una cota inferior (en media) en los pasos que se deben realizar para llevar a cabo cualquier ataque. La seguridad probable hace referencia a ataques que estén limitados por la cantidad de datos o por el tiempo que se dispone. Unos claros ejemplos de algoritmos de éste tipo son el DES y el IDEA.

4.1.4 Comunicación receptor – *smart card*

En este apartado se va a hablar sobre las comunicaciones entre la *smart card* y el receptor, se puede consultar más sobre este tema en [56]. Partiendo del funcionamiento descrito en el *Tema* 2, en la *Figura 27* se puede observar como si la CW no se cifra cuando se envía al receptor, un atacante puede redireccionar la CW hacia otro descodificador del mismo tipo para descodificar el programa directamente, escuchando la comunicación de la interfaz de la *smart card*, la cual es insegura. En el caso que la CW sea cifrada con una clave de sesión SK, ésta es importante que varíe de forma dinámica para mantener la comunicación segura. No obstante, en muchos de los sistemas actuales, la SK es siempre un parámetro fijo, el cual reduce significativamente la seguridad, ya que los piratas pueden llevar a cabo ataques de texto claro conocido simplemente escuchando la comunicación. Además, sin autenticación mutua, los piratas pueden duplicar una *smart card* o redireccionar el mensaje de comunicación entre una *smart card* legítima y un receptor hacia otro receptor, para darle derechos de recibir el programa en cuestión a un usuario no autorizado, el cual causará unas pérdidas considerables al proveedor de servicios. Por este motivo, son necesarias unas SK dinámicas y autenticación mutua entre el receptor y la *smart card*.

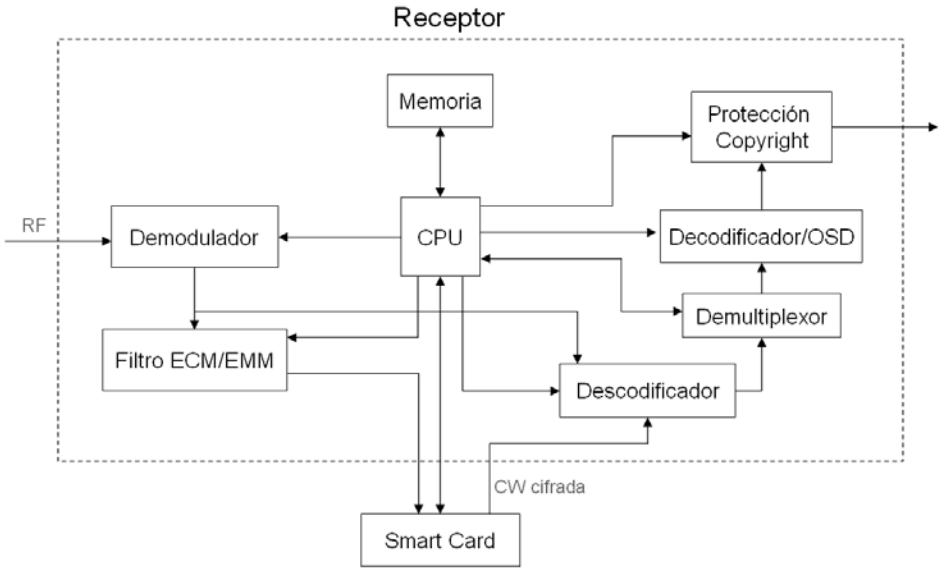

Figura 27: Cooperación smart card y receptor

Una de las formas para poder conseguir que dos o más entidades se pongan de acuerdo en la elección de una SK es mediante el protocolo de intercambio de claves, a pesar de que las comunicaciones se realicen a través de un canal inseguro. Esta SK dinámica usada en el esquema permitirá mejorar la seguridad del sistema.

Para realizar la explicación de las fases se ha utilizado la siguiente notación:

- ID_c : Identidad (número) de la *smart card*.
- ID_s : Identidad (número) del receptor.
- x_s : Clave secreta del receptor.
- $h(\cdot)$: Función hash de único sentido con output de 128 bits.
- $H(\cdot)$: Otra función hash de único sentido con output de 128 bits, diferente de $h(\cdot)$.
- $E_{SK}(\cdot)/D_{SK}(\cdot)$: Algoritmo de cifrado/descifrado de clave simétrica *SK*.
- T' : Identificador de tiempo del receptor en un determinado instante.
- T : Identificador de tiempo de la smart card en un determinado instante.
- ΔT : Tiempo transcurrido entre el envío y la recepción de un dato entre las partes.
- $\oplus$: Operador a nivel de bit, OR exclusivo.

Fase de registro

Cuando un usuario U desea suscribirse, calcula $IP = H(ID_c, PW)$ y envía ID_c e IP al SMS. El SMS calcula $C = IP \oplus H(ID_c, x_s)$ y almacena IP, C, ID_s, $h(\cdot)$, $H(\cdot)$ y MPK con otra información de cuenta de usuario en la *smart card*.

Fase de login

Si el usuario U desea recibir los programas a los que está suscrito, debe introducir la SC en el receptor e introducir el ID_c y el PW. La *smart card* realizará las siguientes operaciones:

1. Comprobará la igualdad de IP y de $H(ID_c, PW)$. Si no son iguales, la smart card rechaza el login.
2. Si se acepta la solicitud de login, la smart card generará un número aleatorio *r* con una longitud de 512 bits, posteriormente calculará C_2 y CID_c.

$$C_2 = h(T \oplus C \oplus IP)$$

$$CID_c = ID_c \oplus h(ID_s)$$

3. Posteriormente enviará (CID_c, C_2, r, T) al receptor.

Fase de autenticación mutua

El receptor y la *smart card* deben seguir los siguientes pasos para realizar la autenticación mutua:

1. Una vez el receptor recibe (CID_c, C_2, r, T), comprueba si $(T'-T) < \Delta T$. Si $(T'-T) \geq \Delta T$, el receptor parará la comunicación. En caso contrario, el receptor obtiene el ID_c, calcula $C_2^{'}$ y comprueba si C_2 y $C_2^{'}$ coinciden.

$$ID_c = CID_c \oplus h(ID_s)$$

$$C_2^{'} = h(T \oplus H(ID_s, x_s))$$

2. Si C_2 es igual a $C_2^{'}$, el receptor elige un número aleatorio e y calcula e' y C_3 del siguiente modo (donde T_1 es el identificador de tiempo del receptor en ese preciso instante):

$$e' = h(ID_s) \oplus e \oplus T_1$$

$$C_3 = h(C_2^{'} \oplus ID_s \oplus T_1 \oplus e')$$

3. El receptor envía (C_3, T_1, e) a la *smart card*.
4. La *smart card* calcula $C_3^{'}$ y comprueba la igualdad de C_3 y $C_3^{'}$.

$$e' = h(ID_s) \oplus e \oplus T_1$$

$$C_3^{'} = h(C_2^{'} \oplus ID_s \oplus T_1 \oplus e')$$

5. Si no se cumple la igualdad de C_3 y $C_3^{'}$, la smart card rechaza la comunicación. En el caso que se cumpla la igualdad, la smart card aceptará al receptor.

Fase de acuerdo de claves

Si se lleva a cabo la mutua autenticación de forma satisfactoria, entonces ambas partes usarán la siguiente ecuación para la clave de sesión común:

$$SK = H(r, e, ID_c, ID_s)$$

Fase de la transmisión de la CW

Después de descifrar la CW, la *smart card* usa la SK para cifrar la CW, $\mathrm{CW}' = E_{SK}(\mathrm{CW})$ y envía la CW' al receptor para descodificar el programa. El receptor descifra la CW', $\mathrm{CW} = D_{SK}(\mathrm{CW}')$.

Fase de cambio de password

El usuario puede cambiar el password de la *smart card* de PW a PW' libremente. Lo único que debe hacer es introducir el ID_c y el PW en la *smart card*. La smart card comprobará la igualdad $IP = H(ID_c, PW)$ y si se cumple, el usuario podrá introducir el nuevo password PW' en la *smart card*. La *smart card* calculará entonces $IP' = H(ID_c, PW')$ y $R' = IP' \oplus R \oplus IP$. A continuación, sustituirá R por R' e IP por IP'.

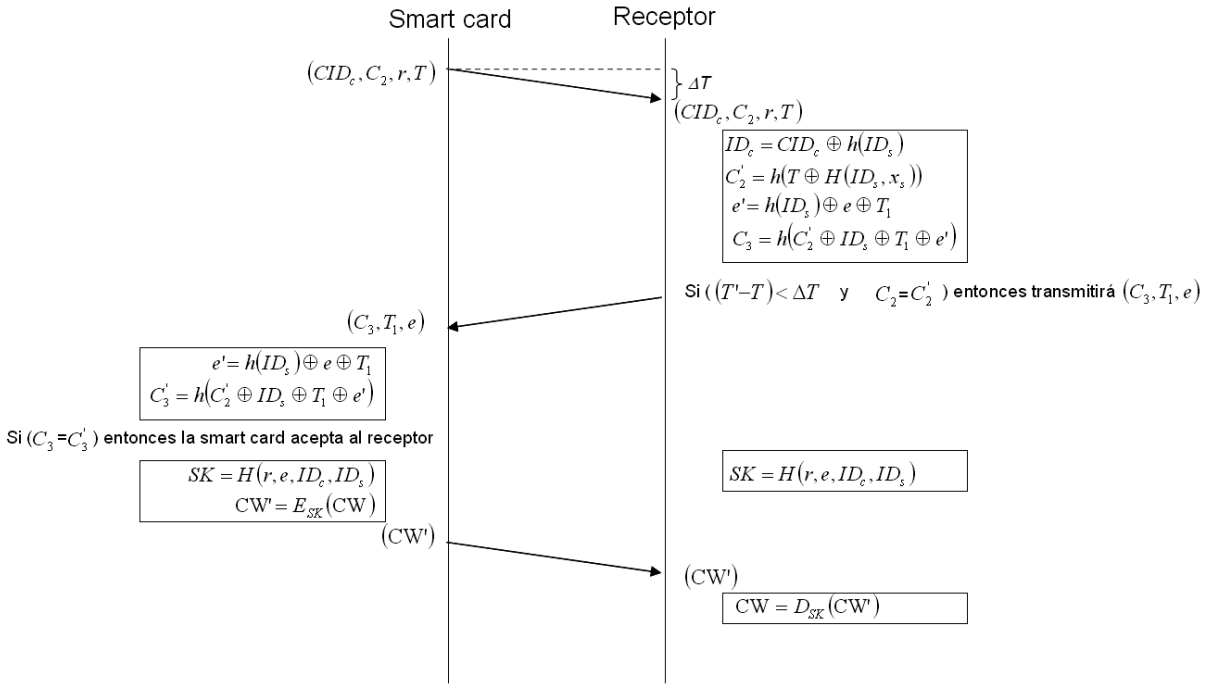

Figura 28: Esquema comunicación receptor – smart card

Además del esquema propuesto, existen otros métodos más o menos seguros para obtener el mismo resultado. Los esquemas más comunes suelen guardar parte de la información almacenada en la *smart card*, cifrada mediante algoritmos de clave simétrica (DES, AES, IDEA…). La cual no se puede conocer a menos que se posea la misma clave que se usó para cifrarla, por ejemplo, esta clave podría ser PW. Por otro lado, para llevar a cabo la autenticación mutua, se suelen usar algoritmos de clave asimétrica, ya que con el simple hecho de que la *smart card* cifre la información con la clave pública del receptor (previamente almacenada en la *smart card*) se consigue que sólo el receptor autorizado sea capaz de descifrarla.

El motivo por el que he explicado un esquema y no otro, es por el simple hecho que el esquema comentado no posee ningún tipo de operación exponencial, las cuales aportan mucha carga computacional, aspecto en el cual las *smart cards* no van sobradas. No obstante, hay que comentar que si están preparadas para ello. Si se reduce el coste computacional derivado de la autenticación mutua y del acuerdo de claves, se dispondrá de mayor capacidad de recursos por parte de la *smart card* para el descifrado de las claves de la señal de televisión.

4.1.4.1 Análisis del método propuesto

El sistema propuesto tiene varias ventajas, como las que se comentan a continuación:

Resistencia contra el ataque de repetición: En el sistema propuesto, clave de sesión (SK) se genera a partir de la salida de una función *hash* de único sentido, cuya entrada es la concatenación de r, e, ID_c e ID_s. Los cuatro valores, r, e, ID_c e ID_s sólo son conocidos por parte de la *smart card* y del receptor. Dado que el número aleatorio r es la información semilla para cada clave de sesión, el ataque de repetición y el de suplantación no son posibles. Por otro lado, dado que la clave de sesión es diferente en cada comunicación, es difícil atacar el algoritmo de cifrado para obtener la CW mediante el ataque de texto plano conocido.

Seguridad contra el problema de la clonación de *smart cards*: El esquema propuesto es también resistente al problema de la clonación de *smart cards*. Cuando un atacante usa su *smart card* clonada en otro receptor, el ID del receptor (ID_s) no corresponde con el almacenado en la *smart card* clonada y del mismo modo, la función *hash* de la *smart card* clonada y la del receptor son diferentes. Por lo tanto, es imposible utilizar esa *smart card* clonada en otro receptor.

Seguridad contra el redireccionamiento de la comunicación del mensaje: Cuando un atacante redirecciona la comunicación de una *smart card* del mensaje a otro receptor, este nuevo receptor no dispone de la información necesaria, la SK, para poder descifrar la CW.

Eficiencia computacional: Debido a que las *smart cards* tienen una capacidad de cálculo muy limitada, la carga de trabajo de éstas es un aspecto a tener en cuenta. Como se puede observar, en el esquema sólo se usan operaciones de OR exclusiva y de este modo la carga computacional es muy pequeña, frente a otras posibles operaciones como ya se ha comentado anteriormente.

4.1.5 Rupturas de CAS (caso concreto de Nagra 2)

En este apartado se pretende dar una visión real de un Sistema de Acceso Condicional. El CAS elegido ha sido Nagra2 de Nagravision. Se trata de un CAS que ha estado en funcionamiento hasta hace "relativamente poco tiempo" en la plataforma Digital+. Una vez conocido el funcionamiento del sistema se procederá a dar una explicación de cómo una serie de gente muy avispada ha conseguido romper el sistema. Toda la información ha sido extraída de foros Web en la que los diferentes usuarios proponían sus teorías y logros, así como intentaban contestar las dudas de otros usuarios [70], [71], [72] y [73].

Como ya he comentado el Nagra2 es un Sistema de Acceso Condiciona que ya no está vigente en España, ya que Digital+ lo ha sustituido por Nagra3. Hasta la fecha, Nagra3 permanece fuerte contra los ataques, ya que todavía es desconocido su funcionamiento, no obstante, lo interesante será ver lo que dura.

Antes de comentar como se rompe el sistema Nagra2, se va a comentar cierta información acerca de los ECM y de los EMM en el sistema Nagra2, así como recordar algunos de los conceptos ya comentados con el fin de poder entender más fácilmente la explicación que hay más adelante acerca de la ruptura del sistema.

EMM

Los EMM se utilizan para comunicar al equipo del abonado (decodificador/*smart card*) cual va a ser la próxima AK. Los EMM se envían cuando el proveedor lo considera oportuno, y los periodos suelen oscilar, dependiendo la época, entre meses y días, o incluso horas. Los pasos por los que pasa la clave AK cuando el proveedor procede a formar el mensaje EMM son:

1. Se cifra la clave AK en claro mediante el algoritmo 3DES (clave opertiva). Ha esta clave cifrada se la llama Ikey
2. Se le aplica un proceso de "camuflado" la Ikey
3. Se cifra con RSA (e,N)
4. Se cifra con IDEA (Idea key)
5. Se vuelve a cifrar con RSA (e,N)

El descifrado de los EMM culmina con la obtención de la AK, que va a servir para descifrar la clave de codificación (CW) del video y el audio.

ECM

Los ECM se utilizan para comunicar al equipo del abonado (decodificador/*smart card*) cual es la clave que se ha utilizado para codificar la señal de video y audio. Los ECM se envían a intervalos muy cortos de tiempo (varios por segundos). Los ECM viajan cifrados con RSA (e,N), IDEA (Idea key) y RSA (e,N), aunque sus claves son distintas a las que se utilizan para los EMM.

La *smart card* descifra el ECM y devuelve al Firm (decodificador), la clave CW cifrada con 3DES (SessionKey).

Ruptura del sistema

Durante los últimos días de vida de Nagra2 en Digital+, especialmente entre marzo y diciembre del 2007 se usó la Fuerza Bruta para romper el cifrado de las claves operativas. Esto fue posible porque en cuanto el operador se dio cuenta que los hackers habían conseguido romper la clave utilizada para cifrar 3DES, la Clave Operativa, a partir de la Ikey 06/46, se vieron obligados a improvisar de forma precipitada, un método de camuflaje de las Ikeys, de forma que el descifrado 3DES fallase en el intento de obtener la AK. La fortaleza de dicho método se basaba en comunicar a las *smart cards* vía los EMM el método utilizado para camuflar la Ikey, y al mismo tiempo, cifrar dicha comunicación con una nueva clave.

En poco tiempo, los hackers lograron obtener de nuevo la clave operativa. Lo curioso de este asunto, es que no necesitaron utilizar sus técnicas de criptoanálisis y dumpeo de *smart cards*. El sistema que utilizaron para obtener la clave operativa, fue un sistema que ya había sido desechado en otras ocasiones, por inviable: La Fuerza Bruta.

Intentar romper por medio de Fuerza Bruta el cifrado RSA/IDEA/3DES con claves de 16 bytes (128 bits), requiere un tiempo de procesamiento tan grande que lo hace inútil a efectos prácticos. No obstante, este sistema poseía una importante debilidad que podía ser explotada usando la Fuerza Bruta. A partir de entonces, los programas basados en Fuerza Bruta proliferaron en poco tiempo y con notable éxito.

Para aquellos que no lo sepan, el concepto de Fuerza Bruta significa ir probando hasta que consigues un éxito. El procedimiento es el siguiente, se envía al decodificador una clave para comprobar si la misma nos permite ver el contenido de la señal. Si se ve, entonces se ha

acertado y el ejercicio ha terminado. Pero si la pantalla sigue negra, se intenta con otra clave. Este ejercicio se debe repetir hasta conseguir un éxito.

Para poder llevar un ataque de Fuerza Bruta eficiente es necesario un procedimiento que ayude a mejorar las previsiones sobre cuales serán las claves más probables, es decir, aquellas que produzcan éxito. A este procedimiento se le llama procedimiento "generador de claves candidatas". Existe un segundo procedimiento que es el que se encarga de decir si la clave candidata utilizada produce éxito, a este procedimiento se le llama "banco de prueba" o "certificador de candidatas". Con sólo estos dos procedimientos, se garantiza que se logrará encontrar la clave correcta.

Llegados a este punto, uno puede pensar que dado los algoritmos y las longitudes de las claves utilizadas, el periodo de tiempo necesario para que se produzca un éxito mediante este ataque sea demasiado grande. Sólo con calcular el numero de posibles claves candidatas, queda claro que todavía no se dispone de maquinas suficiente veloces para generar/probar las claves en un tiempo "razonable". Es por esta razón, que hasta marzo del 2007 a nadie se le pasó por la cabeza utilizar la Fuerza Bruta para averiguar la clave. No obstante, el nuevo método de camuflaje incorporado en los EMM a partir de marzo del 2007, provocó una tremenda debilidad ante posibles ataques de Fuerza Bruta sin necesidad de grandes análisis.

A la hora de llevar a cabo un ataque de Fuerza Bruta a este sistema se han de tener claros los siguientes parámetros:

- Número de claves candidatas que se deben generar para asegurar que el éxito se produzca.
- Velocidad del "generador de candidatas"
- Velocidad del "certificador"

Es obvio, que para desarrollar un sistema de Fuerza Bruta, orientado a explotar la debilidad de los métodos de camuflaje de la Ikey, se requiere un buen conocimiento de los algoritmos de cifrado usados (RSA, IDEA, IDEACBC, 2TDES) así como las correspondientes claves que se utilizan. Lo cierto es que se conocen en profundidad todos los algoritmos, y casi todas las claves utilizadas.

A la vista de los resultados reales y utilizando técnicas de criptoanálisis, se averiguó el funcionamiento de, al menos, seis métodos de camuflaje utilizados por el operador:

1. Suma 1 a un número variable de bytes de la Ikey, comenzando en un byte elegido.
2. Resta 1 a un número variable de bytes de la Ikey, comenzando en un byte elegido.
3. Invierte los *nibbles* de un número variable de bytes de la Ikey, comenzando en un byte elegido.
4. Hace XOR con un número variable de bytes de la Ikey, comenzando en un byte elegido, con datos de la zona UROM00, a partir de una dirección elegida.
5. Hace Suma con un número variable de bytes de la Ikey, comenzando en un byte elegido, con datos de la zona UROM00, a partir de una dirección elegida.
6. Hace Resta con un número variable de bytes de la Ikey, comenzando en un byte elegido, con datos de la zona UROM00, a partir de una dirección elegida.

Para no complicar innecesariamente la explicación, se intentará describir las funciones básicas que realiza un programa de los que están diseñados para encontrar las claves. Los datos de entrada, que el programa necesita son:

a) Un ECM.

b) La IKey camuflada, obtenida a partir de los EMM por algún programa.

Como ya he comentado, el programa debería realizar su trabajo utilizando dos procedimientos: Generación de candidatas y Certificación.

Generador de claves candidatas

El procedimiento Generador consiste en los siguientes pasos:

1. Tomando como base la Ikey Camuflada, aplicar:
2. El método 1
3. Este método obtiene un numero variable de Ikeys Candidatas (depende del numero de bytes a camuflar "n")
4. Generar una Ikey Candidata, la cifra con 3DES y obtiene una clave candidata
5. La clave candidata se envía al Procedimiento de Certificación, que mediante cálculos, decide si dicha clave permitirá que el firmware descodifique las tramas de video, codificadas MPEG2 (éxito) o NO (fracaso).

Si la clave candidata no es la "buena", el programa deberá continuar analizando la siguiente candidata volviendo al paso 3. En el caso que después de haber terminado con el método 1, el programa no haya obtenido éxito, este deberá elegir el siguiente método, y realiza los mismos cálculos. Así con todos los métodos, hasta obtener éxito. Si no se obtiene éxito con ninguno de los métodos conocidos, entonces se puede afirmar que el proveedor esta utilizando un nuevo método de camuflaje, que el programa desconoce.

Procedimiento de Certificación

Para poder explicar/entender este procedimiento hay que recordar los siguientes conocimientos. Como ya se ha dicho en el *Tema 2*, la finalidad del ECM es transportar la clave CW para que el firmware pueda decodificar las tramas de video/audio que van codificadas. Es conveniente recordar que dicha CW, se modifica cada pocos segundos por el proveedor, con la finalidad de dificultar el conocimiento de la misma. El mensaje ECM lleva 8 bloques de 8 bytes cifrados, entre los que se encuentran la CW, la Firma, y los datos necesarios para que la tarjeta autorice al abonado para ver el canal que tiene sintonizado.

La CW es la clave que el firmware debe usar para decodificar las tramas de audio/video, y la Firma es un hash de 8 bytes, obtenido con una clave especial, también conocida. La Firma ocupa los primeros 8 bytes cifrados del bloque de 64 bytes.

Los ocho bloques mencionados (64 bytes) están cifrados a tres niveles. Los algoritmos y claves necesarios para poder descifrarlos son los siguientes:

- Algoritmo: RSA, clave de descifrado (d,N) ambas conocidas d=03 y N=Modulo ECM RSA
- Algoritmo: IDEACBC, cifrado y descifrado con la AK que no conocemos. .
- Algoritmo: RSA, clave de descifrado (d,N) ambas conocidas d=03 y N=Modulo ECM RSA

Entre los algoritmos mencionados, existen una serie de procesos de ajustes de bits y *byteflops*, que no se mencionan para no complicar la explicación, pero hay que dejar claro que existen.

Una vez aplicado el algoritmo 3, se dispone del bloque de 64 bytes totalmente en claro. Los primeros 8 bytes del cual son la Firma Descifrada y a continuación la CW junto a otra información diversa.

Una vez descifrado el ECM, la tarjeta debe verificar su autenticidad, para asegurarse de que realmente ha sido enviada por el proveedor autorizado, y no inyectada por un posible

atacante. La verificación de autenticidad se realiza calculando la Firma con un procedimiento distinto al de descifrado. Consiste en usar el algoritmo de cifrado IDEACBC y una clave especial para este procedimiento (también conocida). Si una vez realizado este procedimiento, la Firma Calculada no coincide con la Firma Descifrada, se abandona el proceso y el abonado no es autorizado a ver el canal sintonizado.

Una vez conocido el proceso de descifrado del ECM, y la comprobación de la firma, se está en condiciones de montar el procedimiento para Certificar si la clave candidata recibida del Generador, es la correcta o no.

Una vez se recibe la clave candidata procedente del Generador se procede al descifrado del ECM tal como se indica en la explicación anterior. Con lo cual, al final del mismo, y si todo es correcto (algoritmos y claves utilizadas) el programa conoce la Firma Descifrada. Hay que recordar que el descifrado de IDEACBC se realiza usando como clave la clave candidata.

A continuación se procede a Calcular la Firma, tal como lo hace la tarjeta (algoritmo IDEACBC con la clave especial conocida). Si la firma calculada coincide con la firma descifrada, obtenida al final del descifrado del ECM, quiere decir que todas las claves utilizadas eran correctas y por tanto la clave candidata también lo era. Si por el contrario, las firmas no coinciden, deberá continuar, verificando la siguiente clave candidata, cedida por el Generador.

Si el programa no logra obtener éxito una vez analizadas todas las claves obtenidas por explotación de TODOS los métodos conocidos, puede pasar que el proveedor esté utilizando un nuevo método de camuflaje que el programador desconoce.

4.2 Comparativa entre el CAS y los BES

Con el fin de obtener una idea más visual de las características de los sistemas, tanto del Sistema de Acceso Condicional como de los esquemas *Broadcast Encryption*, se ha decidido desarrollar un programa en C que simule el funcionamiento de los diferentes sistemas. Este programa permite realizar el cálculo aproximado del TO, el CC y el SS medio para cada sistema. Hay que destacar que los datos de los parámetros de cada sistema proporcionados hasta el momento, son, en la mayoría de los casos, cotas superiores. Estas cotas superiores son difícilmente alcanzables a no ser que se cumplan una serie de

condiciones específicas. Por este motivo se ha decidido calcular los valores medios, ya que de este modo se consigue obtener un resultado que dista menos con los posibles valores reales de su posible aplicación.

No obstante, no hay que olvidarse de los valores de los parámetros presentados hasta el momento. A pesar de ser cotas superiores con una probabilidad muy pequeña de que sucedan, los diseños de implantación de estos sistemas deben estar preparados para soportarlas. A la hora de diseñar los esquemas en los que se implanten estos sistemas, se deben tener en cuenta las cotas superiores para que el sistema no se sature en el remoto caso de que estas se alcancen.

4.2.1 Simulación

Como ya se ha comentado, el programa de simulación ha sido desarrollado en lenguaje C. Su principal finalidad es proporcionar los valores aproximados de la media del TO, el CC y el SS de un sistema de acceso condicional genérico y de los esquemas de *Broadcast Encryption* presentados en el *Tema 3*. Los cálculos se hacen para un determinado conjunto de n usuarios, tanto revocados como privilegiados.

Con el fin de facilitar el trabajo se ha decidido que n sea del tipo 2^h, donde h es un número entero positivo. Esta elección se ha hecho esencialmente por el motivo que los esquemas de *Broadcast Encryption* en forma de árbol necesitan una estructura binaria completa.

El programa se puede dividir en tres partes. Una primera parte que se encarga de generar las poblaciones de usuarios de forma aleatoria, una segunda parte que se encarga de llevara a cabo las "agrupaciones" de los usuarios y una última parte de cálculo y publicación de los resultados.

Primera parte: Generación de usuarios

Se encarga de generar diferentes variaciones de usuarios. Con el fin de obtener más o menos precisión en los resultados, el programa permite generar un número determinado de variaciones para cada población de usuarios privilegiados. Estas poblaciones de usuarios privilegiados van desde 0 hasta n. De este modo, si se desean obtener valores con mucha precisión se deberá elegir un número elevado de variaciones para el mayor número de las poblaciones de x usuarios privilegiados posibles, pudiendo ser x=0, 1, 2, 3,…, n. Como

consecuencia, el tiempo necesario para que el programa haga los cálculos incrementará considerablemente.

Segunda parte: Agrupación de subconjuntos

A esta parte del programa la he llamado agrupación de subconjuntos porque como su nombre indica se encarga de construir el grupo de subconjuntos de usuarios que cubre la totalidad de la población de usuarios privilegiados para cada unos de los esquemas. Los algoritmos que se encargan de realizar/construir las agrupaciones de los subconjuntos para cada uno de los esquemas han sido desarrollados por mi mismo. Para su desarrollo me he basado en la información de la teoría (*Tema 3*).

Tercera parte: Cálculo y publicación de resultados

Esta última parte del programa se encarga del cálculo y publicación de los valores de los parámetros de los esquemas obtenidos. Una vez construida la agrupación de subconjuntos de usuarios privilegiados de cada uno de los esquemas, el cálculo de los parámetros resulta más o menos complicado en función del parámetro a calcular y del esquema utilizado. Por ejemplo, el cálculo del TO para cualquiera de los esquemas resulta muy sencillo una vez se ha llevado a cabo la agrupación de subconjuntos, ya que simplemente hace referencia al número total de subconjuntos de usuarios privilegiados, mientras que el cálculo del CC del esquema *Layered Subset Difference*, por decir uno, resulta algo más complicado de implementar.

Una vez se tiene el resultado de los parámetros para cada una de las variaciones calculadas de las poblaciones de usuarios privilegiados, se realiza la media para cada población de x usuarios privilegiados. Posteriormente, estos resultados se almacenan en ficheros para poder realizar un posterior análisis, ya sea gráfico o numérico.

4.2.2 Resultados

En este apartado se va ha realizar un análisis gráfico de los resultados obtenidos mediante el programa desarrollado. Para ello se han realizado algunos gráficos que facilitan el análisis comparativo entre esquemas a partir de sus parámetros.

TO

En la *Figura 29* se pueden observar los valores medios del TO para cada uno de los esquemas para un total de 1024 usuarios, entre revocados y privilegiados. Como se puede observar a primera vista, el número de cifrados que debe realizar el centro emisor en el caso del Sistema de Acceso Condicional (CAS) para los mensajes EMM, es directamente proporcional al número de usuarios privilegiados. Para obtener el TO real del CAS, a este valor se le debería sumar 1. Este 1 hace referencia a los mensajes ECM.

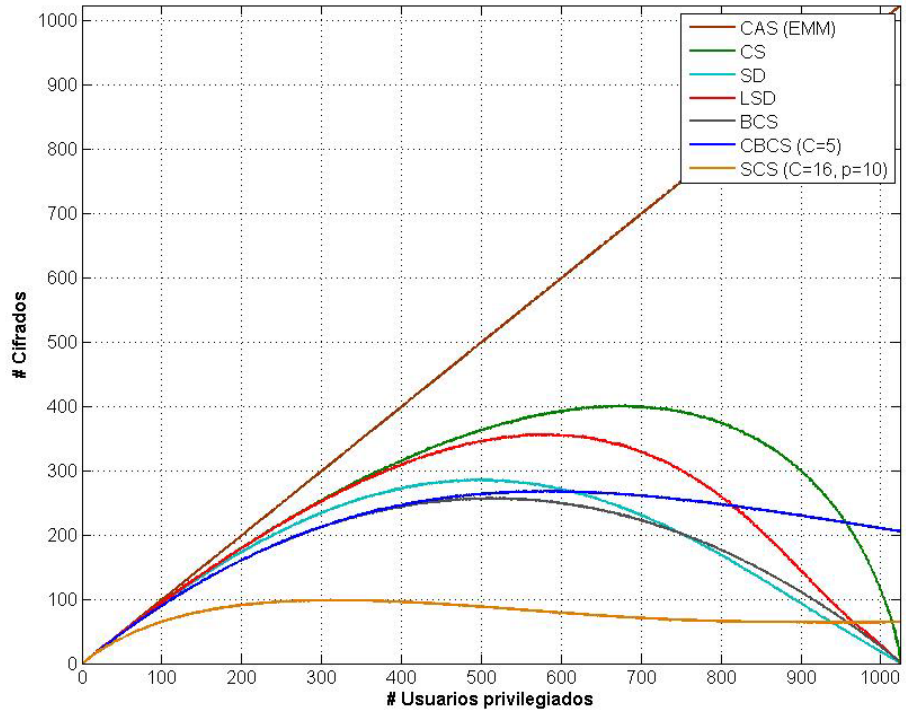

Figura 29: TO de los sistemas para *n*=1024

Para poder realizar un análisis con algo más de detalle, en la siguiente figura se muestra una ampliación de la zona de los esquemas de *Broadcast Encryption*. Como se puede observar, los valores del TO son muy parecidos para todos los sistemas, a excepción del *Skipping Chain Scheme* que dada la elección de C=16 y p=10 es el más diferente, hasta que aproximadamente el 10% del total de usuarios son privilegiados. A partir de ese punto hasta que el 20% del total son usuarios privilegiados, los valores se empiezan a distanciar en función del tipo de sistema. De este modo, se observa que los valores de TO son similares en este tramo, por un lado los que tienen estructura en forma de árbol y por el otro los que tienen estructura lineal.

En el tramo que va desde 50% hasta el 70% de usuarios privilegiados, se pueden observar los máximos de los esquemas, a excepción del Sistema de Acceso Condicional y del *Skipping Chain Scheme*. Se trata del tramo en donde existe mayor diferencia entre esquemas.

A partir del máximo, los esquemas decrecen hasta llegar a 100% de usuarios privilegiados. No obstante, hay que destacar que los esquemas parametrizables, el *C-basic Chain Scheme* y el *Skipping Chain Scheme*, no decrecen hasta 1 como el resto de esquemas, por el mero hecho de la parametrización. Estos valores de parametrización permiten reducir el CC y el SS pero al mismo tiempo penaliza al TO cuando el número de usuarios revocados es pequeño. Provocando que el TO sea superior al resto de los esquemas cuando *r* (número de usuarios revocados) es pequeño.

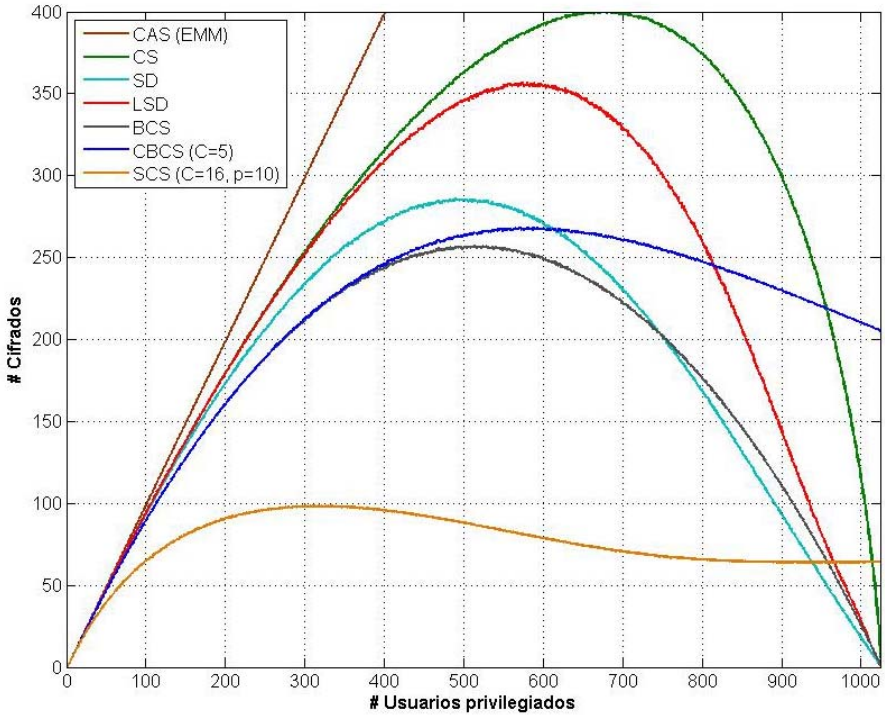

Figura 30: Detalle del TO de los sistemas para *n*=1024

En la *Figura 31* se puede observar el comportamiento del TO para cada uno de los sistemas a medida que *n* va aumentando. Como se ha podido ver en los gráficos anteriores, el TO varía en función de los usuarios revocados, por lo que para hacer el siguiente gráfico, se ha cogido el valor de TO de cada *n* en el que había un 10% de usuarios revocados. El resultado es el siguiente:

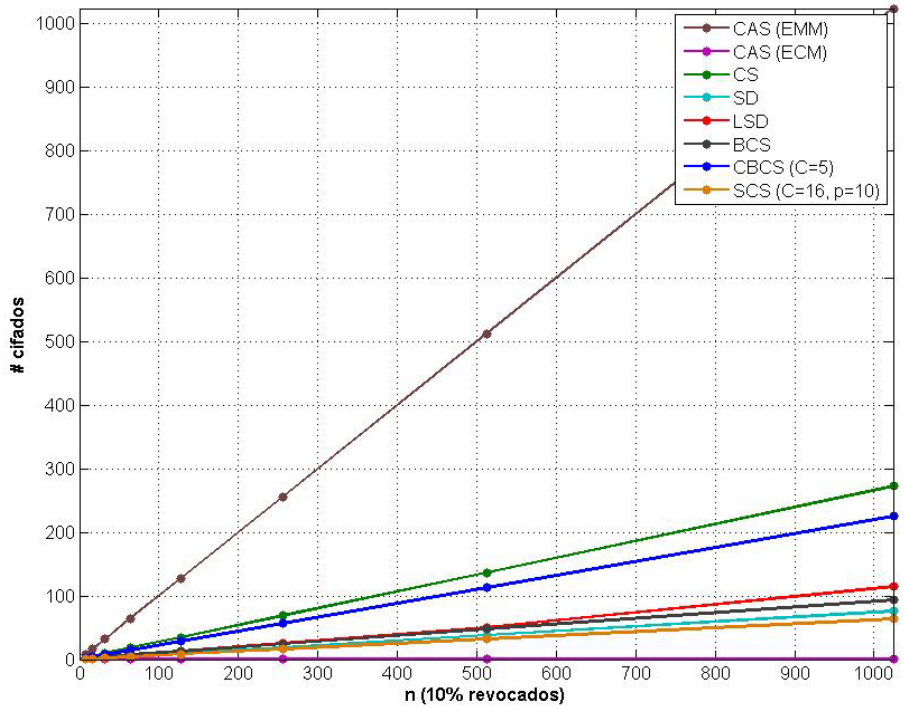

Figura 31: TO para diferentes n (10% de revocados)

Se puede observar como el comportamiento es lineal y crece a medida que *n* aumenta. En este caso sí que se ha representado el TO que aporta el envío de mensajes ECM en los Sistemas de Acceso Condicional, pero como se puede observar, este no aporta nada de información, ya que este es constante e igual a 1.

En la siguiente figura, se ha realizado una ampliación, dejando de lado la evolución de TO de los mensajes EMM que ya es conocido, para poder observar la evolución del TO del esquema *Layered Subset Difference*. Como se puede ver, no acaba de ser lineal. Esto se debe al tema de las capas. Su gráfico será lineal a tramos, es decir, los tramos vendrán especificados por aquellas *h* consecutivas, tal que $\sqrt{h}$ sea un entero. Recuérdese que $n = 2^h$ y que el *Layered Subset Difference* define $\sqrt{\log(n)}$ niveles especiales y capas de $\sqrt{\log(n)}$ niveles. Por este motivo se observa que el tramo que va de n=16 (h=4) a n=512 (h=9) es lineal. El siguiente tramo será lineal entre n=512 (h=9) y n= 65536 (h=16).

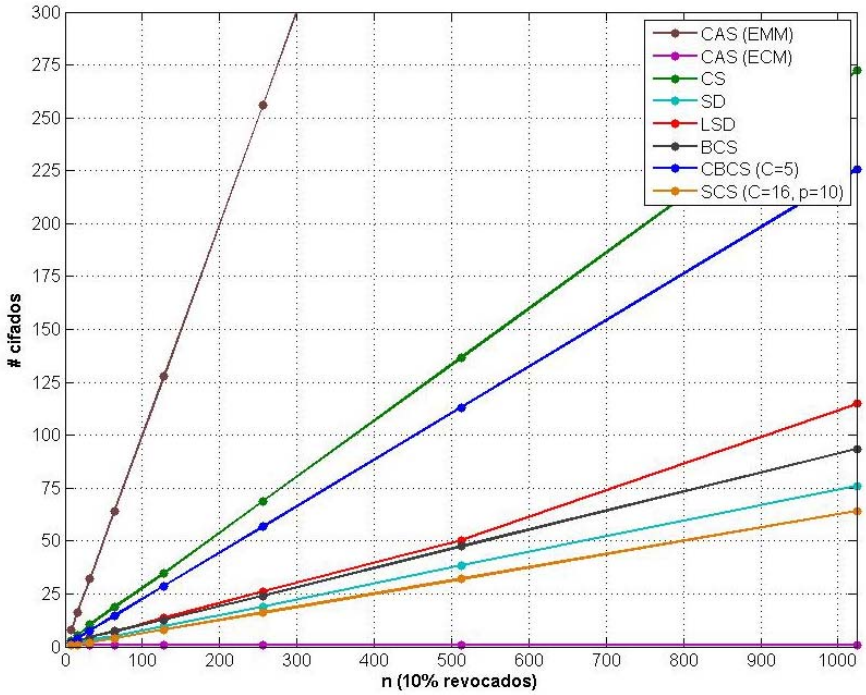

Figura 32: Detalle del TO para diferentes n (10% de revocados)

Como ya se ha comentado anteriormente los esquemas *C-basic Chain Scheme* y *Skipping Chain Scheme* son los únicos que son parametrizables. No obstante, hasta el momento sólo se ha mostrado su evolución para unos valores de C y p concretos. En las siguientes figuras se puede observar la evolución del TO para diferentes valores de C y p.

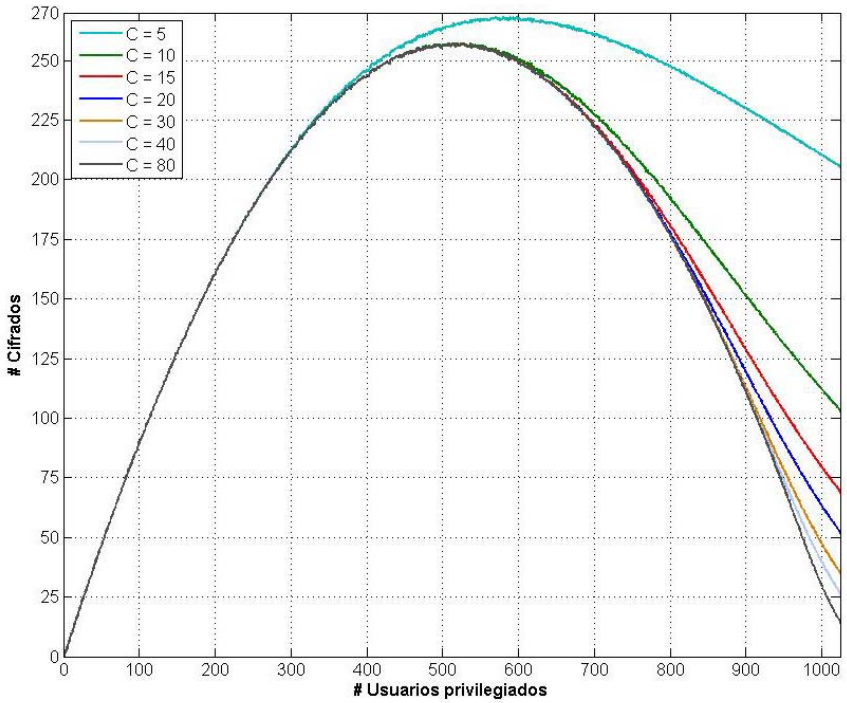

Figura 33: TO del esquema C-basic Chain Scheme para diferentes C

En la figura anterior, se puede observar, como era de esperar, que a medida que C es más grande, la curva del TO del *C-basic Chain Scheme* se aproxima a la del *Basic Chain Scheme.* También se puede observar como cuando C es pequeña, un ligero aumento de esta corresponde a una mejora sustancial en el TO, mientras que cuando la C es grande, esto no ocurre. Un ejemplo de esto se observa en la figura con C=5 y C=10 para el caso de C pequeña y C=40 y C=80 para el de C grande.

Del mismo modo que para la figura anterior se ha querido demostrar el TO en función del valor de C escogido para el esquema *C-basic Chain Scheme*, en la siguiente figura se pretende demostrar de forma visual como afecta la elección de C y p en el esquema *Skipping Chain Scheme*.

Se puede ver como el valor del TO de las opciones en las que las p son iguales, pero con diferente C, es muy similar hasta aproximadamente el 20% de usuarios privilegiados. Por otro lado, a medida que el número de usuarios revocados disminuye, las curvas, como era de esperar, tienden a agruparse en función de la C. Otro resultado interesante que se puede observar es el de que el TO es menor cuanto mayor sea p. Esto es así, ya que los intervalos están definidos a partir de C y de p. Los intervalos pueden tener como máximo C usuarios, entre los cuales habrá como máximo p revocados. De este modo, pueden haber intervalos con menos de C usuarios, pero con p revocados e intervalos con C usuarios pero con menos de p revocados.

Para poder reducir el TO interesa que haya el menor número posible de intervalos. Por lo que interesa que siempre se cumpla que los intervalos sean de C usuarios. La forma de conseguirlo es que se permita un gran número de usuarios revocados en estos. Es por esto, que cuanto mayor sea p menor será el TO, dentro de unos límites ya que p puede ser como mucho igual a C-2.

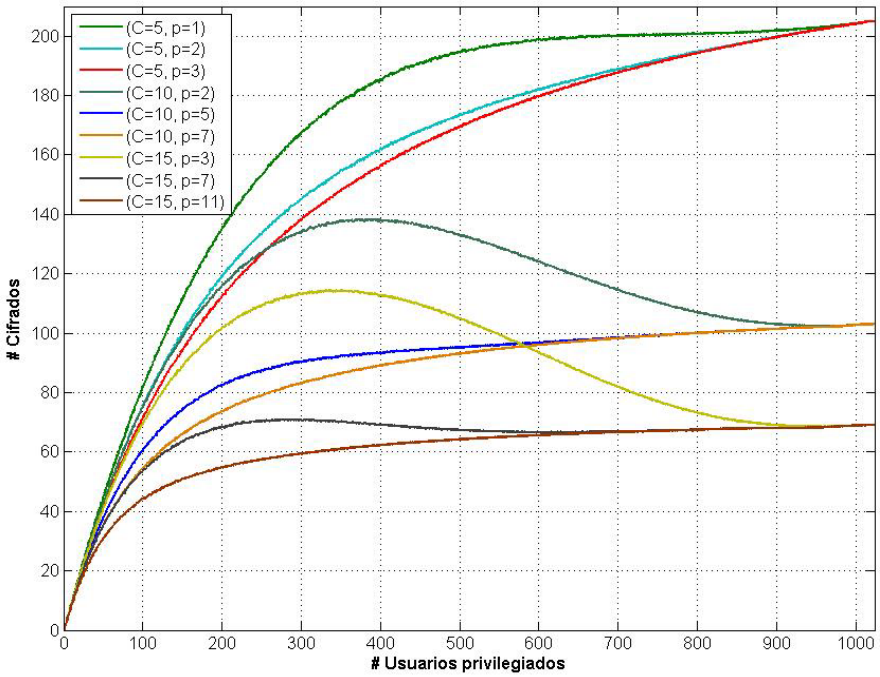

Figura 34: TO del esquema Skipping Chain Scheme para diferentes C y p

CC

En este apartado se va a observar el Coste Computacional necesario para cada sistema. Para empezar se tiene el Sistema de Acceso Condicional, en la siguiente figura se puede observar que este es constante e igual a 1. La verdad es que en realidad es igual a 2, ya que para obtener la CW primero se debe descifrar la clave AK a partir de la MPK y posteriormente, a partir de la AK obtenida, descifrar la CW, pero bien es cierto que la clave AK varía con muy poca frecuencia y por este motivo he decidido obviar esta operación a la hora de calcular el coste computacional.

En la siguiente figura, destacan dos esquemas, que son el *Complete Subtree* y el *Basic Chain Scheme*. El *Complete Subtree* porque el coste computacional es igual a 0 y el *Basic Chain Scheme* porque cuando los usuarios revocados son pequeños el coste computacional se sale del área del gráfico, en realidad este valor en media tiende a 512 para n=1024 cuando el número de usuarios revocados tiende a 0. Por este motivo, se trata de una mala elección de esquema si se prevé que el sistema donde vaya a ser usado pueda trabajar en esta situación.

Otro esquema que destaca a primera vista es el *Skipping Chain Scheme*, ya que entre el 20% y el 85% de usuarios privilegiados es el que precisa de mayor Coste Computacional, en parte por culpa de los valores de los parámetros C y p elegidos. El hecho de que sea el esquema con mejor TO es gracias a que penalice al CC, al SS o bien a ambos.

En la figura también se puede observar la diferencia de Coste Computacional entre el esquema *Subset Difference* y el *Layered Subset Difference*. Como se ha visto en el apartado anterior, el hecho que el *Layered Subset Difference* utilice las capas y los niveles especiales provoca un aumento de TO respecto al del *Subset Difference*, pero como se puede observar ahora, este hecho provoca que el Coste Computacional del *Layered Subset Difference* sea menor que el del *Subset Difference*. Más adelante se verá que también provoca que el número de claves que son necesarias que se almacenen sea menor.

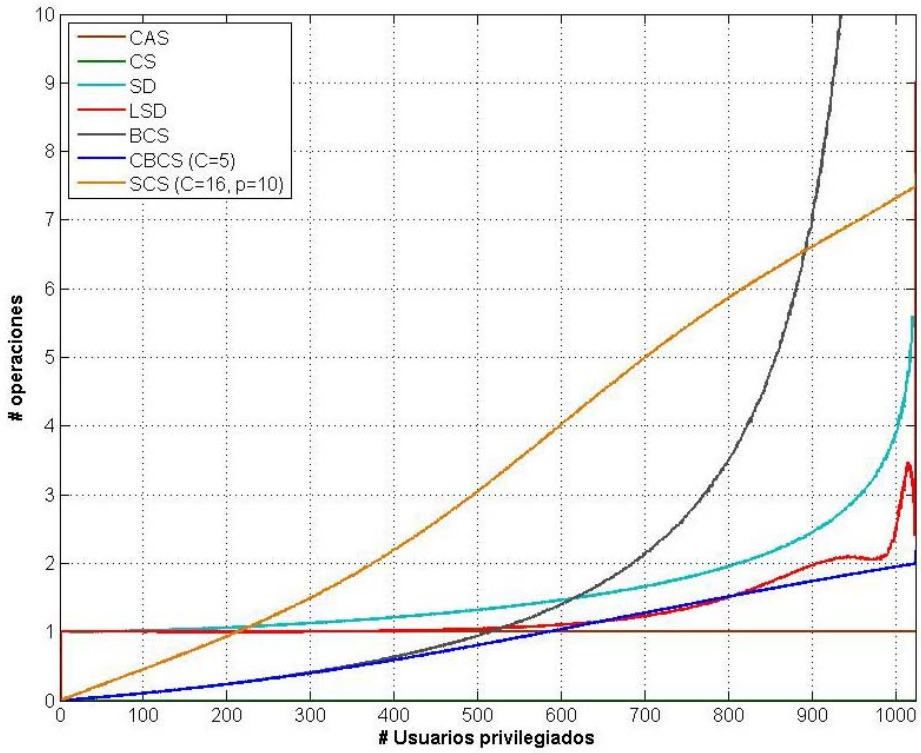

Figura 35: CC de los sistemas para *n*=1024

En la siguiente figura queda patente la evolución del Coste Computacional a medida que aumenta *n*, *n*=8, 16, 32,…, 1024. Del mismo modo que para el TO, en este caso se ha elegido el valor del Coste Computacional de cada *n* con un total del 10% de usuarios revocados.

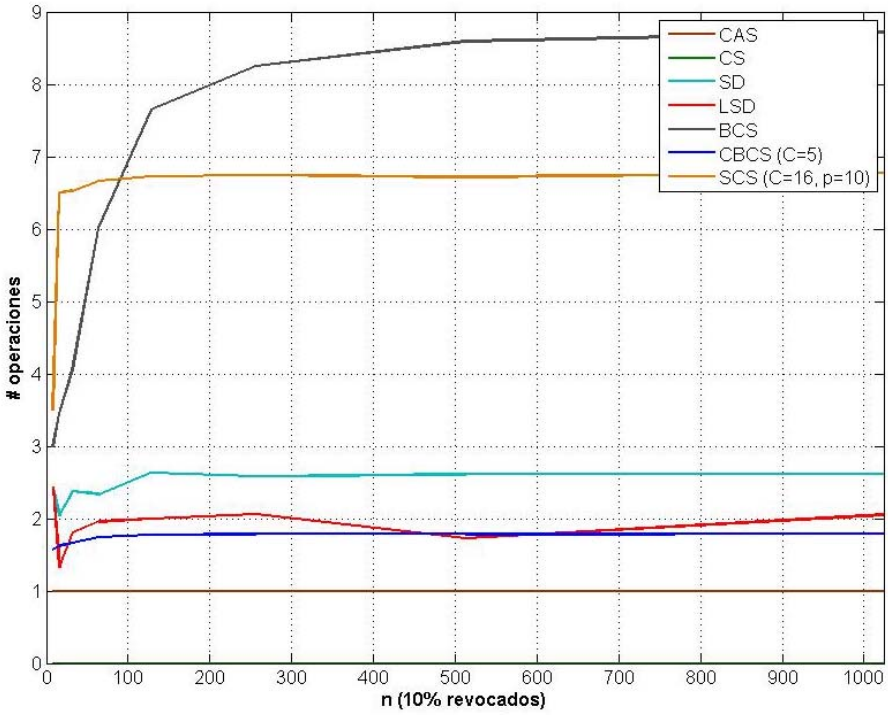

Figura 36: CC para diferentes n (10% de revocados)

SS

Gracias a su sistema jerárquico de claves, los usuarios de los Sistemas de Acceso Condicional sólo necesitan almacenar una clave, mientras que para el resto de esquemas el número de claves que se deben almacenar viene determinado por el número de posibles usuarios *n*, a excepción de los esquemas *C- basic Chain Scheme* y *Skipping Chain Scheme*, que no dependen de *n* y si de los parámetros C y p elegidos.

Los usuarios de los sistemas que deben almacenar un mayor número de claves en el caso de *n*=1024 usuarios son los del *Skipping Chain Scheme* y los del *Basic Chain Scheme*. Resulta bastante evidente, después de observar la fórmula que hace referencia al almacenamiento de claves del esquema *Skipping Chain Scheme* de la *Tabla 2*, que este número va a ser bastante elevado para determinados valores de C y de p, ya que para su cálculo interviene un sumatorio de diferentes combinaciones. No obstante, hay que destacar que este resultado es constante al ser C y p fijos, por lo que cuando *n* es lo suficientemente grande, los usuarios de este esquema deberán almacenar un número menor de claves que los de otros esquemas. En este caso, C=16 y p=10, los usuarios de este esquema deben almacenar 32192 claves.

Los usuarios del esquema *Basic Chain Scheme* deben almacenar de media *n*/2 claves. Se trata de un esquema un tanto desequilibrado, ya que los primeros usuarios de la estructura

lineal tienen que almacenar un número muy reducido de claves, mientras que los del final de la estructura tienen que almacenar muchas, tantas como usuarios tengan por delante en la estructura lineal más uno.

Dado que los valores de almacenamiento de claves para los esquemas *Skipping Chain Scheme* y *Basic Chain Scheme* difieren mucho del resto de sistemas, se ha decidido no mostrarlo en la siguiente figura.

Mirando la figura se puede observar como el tercer valor más grande de almacenamiento de claves es el del esquema *Subset Difference*. Recuérdese que ni el *Skipping Chain Scheme* ni el *Basic Chain Scheme* han sido representados, pero que éstos son de 32192 y $n/2$ claves respectivamente. El valor del almacenamiento de claves del esquema *Subset Difference* es un resultado esperable, ya que posee un TO y CC considerablemente buenos respecto a los otros esquemas y por algún lado se tenía que ver penalizado.

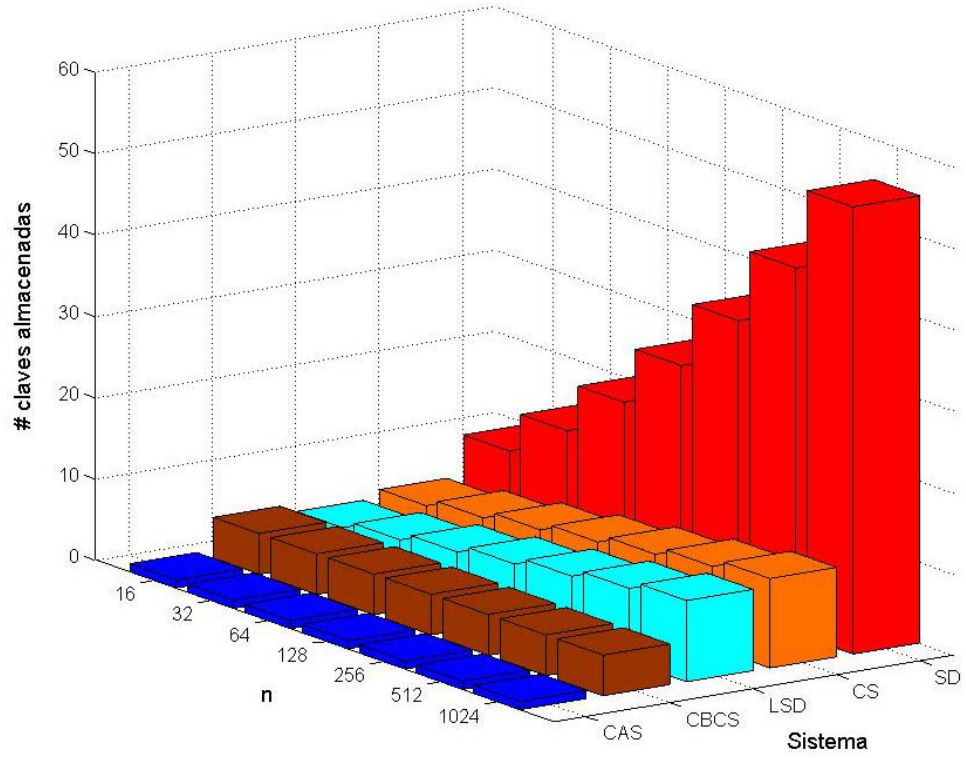

Figura 37: SS de los sistemas en función de n

Como ya se ha dicho, el número de claves que debe almacenar un usuario en cada uno de los sistemas, a excepción del Sistema de Acceso Condicional, el *C- basic Chain Scheme* y el *Skipping Chain Scheme* depende de *n*. Las expresiones para calcular estos valores se pueden consultar en la

Tabla 2.

A modo de resumen, a partir de los valores obtenidos mediante las simulaciones, ha quedado patente que ninguno de los sistemas es mejor que los demás en la totalidad de los tres parámetros TO, CC y SS. Es más, el hecho que un sistema sea mejor que los demás en alguno de los tres parámetros es porque está penalizando a alguno de los otros dos, incluso puede suceder que a los dos. Es por este motivo que se deberá escoger un esquema u otro en función de las características y necesidades del sistema donde se vaya a implementar.

A priori, se puede decir que el Sistema de Acceso Condicional no es del todo adecuado para estar implantado en la televisión de pago vía satélite, ya que este consume un gran ancho de banda debido a su valor tan elevado de TO (EMM). El ancho de banda en estos sistemas es fijo para cada canal, por lo que los proveedores de acceso condicional, para no saturar al sistema, reducen este TO cambiando la AK cada mucho tiempo, con la consiguiente debilidad a nivel de seguridad que esto implica.

Para un sistema de televisión de pago, lo más ideal sería un esquema que mantuviese más o menos un equilibrio razonable entre los tres parámetros que se adecue a las necesidades del ancho de banda y de las capacidades de la *smart card* de cálculo y de almacenamiento. A primera vista, este esquema podría ser el *Skipping Chain Scheme*, el cual posee un TO muy bueno y el hecho de poseer los parámetros C y p le permiten adaptarse al tipo de sistema variando sus valores. No obstante, posee un inconveniente y es que en poblaciones muy grandes de usuarios n, cuando r es pequeño el TO es muy grande a no ser que C también lo sea.

Un mismo Sistema de Acceso Condicional debe proporcionar seguridad a un gran número de canales. Dentro de la televisión de pago hay que distinguir entre dos modalidades, el *pay-per-view* (PPV) y el *pay-per-channel* (PPC). Existe una gran diferencia entre ambas modalidades. En el PPV el usuario paga por ver un cierto contenido, partido de fútbol, película, documental, etc., mientras que en el PPC el usuario paga por ver un canal. El tiempo que el usuario está autorizado para ver el contenido en el caso del PPV es del orden de 2 horas (lo que dure el programa), mientras el del PPC es del orden de meses.

Existen un gran número de canales a los que los usuarios están suscritos en la modalidad PPC. Estos canales suelen entrar dentro de las suscripciones básicas y por lo tanto, la práctica totalidad de usuarios del sistema tienen acceso a ellos. En estos canales sería impracticable usar el esquema *Skipping Chain* ya que para obtener un TO razonable, sería necesario un C muy elevado y consecuentemente un SS y CC demasiado grandes. Por este

motivo, el esquema que considero más adecuado para ser implementado en el sistema de televisión de pago es el *Layered Subset Difference*. A pesar de poseer un TO algo elevado, posee unos valores muy buenos de CC y SS. Además, ofrece un TO bastante bueno en sistemas en los que hay pocos usuarios revocados, interesante para el caso de PPC de canales básicos.

4.3 Implantación del BE en los CAS actuales

En este apartado, como su título indica, se van a proponer dos esquemas utilizando el BE en los CAS actuales. En el primer esquema propuesto, se usa el BE para la distribución de las claves AK, mientras que el segundo esquema propuesto se combinan ambas ideas, CAS y BE.

4.3.1 Primer esquema propuesto

Como se ha podido observar, mediante el uso de alguno de los esquemas de BE presentados en este proyecto se puede conseguir reducir el TO del sistema de acceso condicional actual fruto del envío de las claves AK cifradas. Al utilizar alguno de los esquemas de BE se consigue que las claves AK cifradas, en vez de ser enviadas a un único usuario, sean enviadas a un conjunto de usuarios.

El sistema funcionará exactamente igual a como funcionaba anteriormente, la única diferencia es que los usuarios tendrán varias claves y deberán saber cual de ellas deben utilizar para descifrar la AK en el caso que estén autorizados. Los mensajes EMM pasarán de ser transmisiones unicast a transmisiones broadcast, ya que los mensajes EMM serán comunes para un conjunto de usuarios.

A continuación se presenta una figura que muestra como quedaría el esquema propuesto:

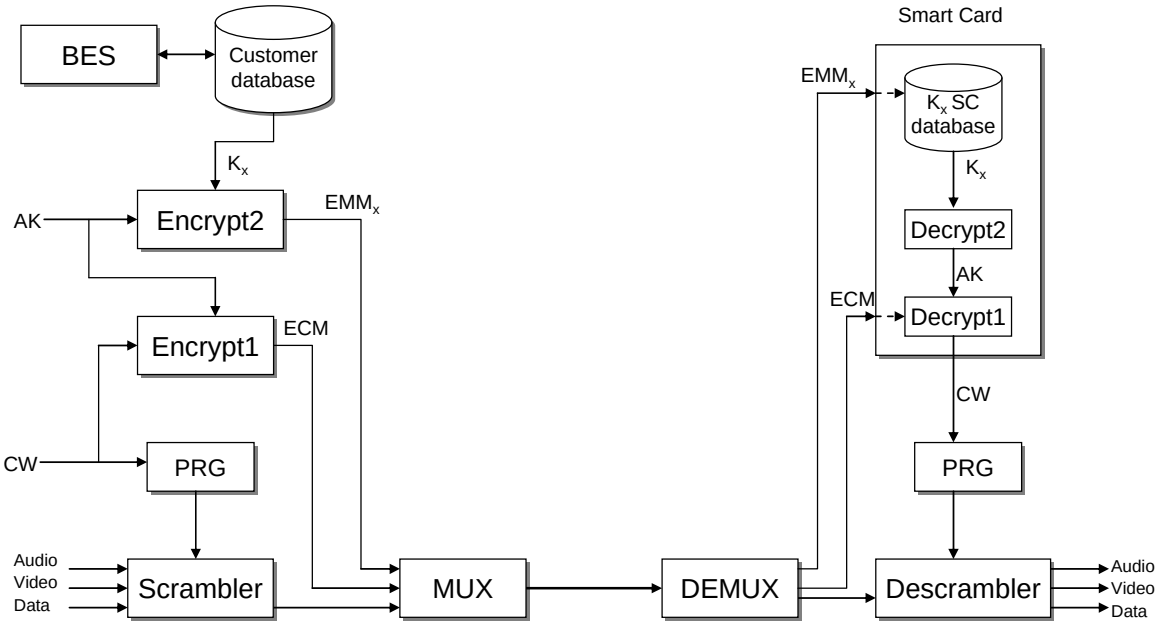

Figura 38: Primer esquema propuesto

Como se puede observar, es muy parecido al sistema de acceso condicional estándar presentado en el *Tema 2*, *Figura 3* y *Figura 4*, con la pequeña diferencia que se le ha introducido un módulo que le he llamado BES (*Broadcast Encryption Scheme*). Este módulo es el encargado de construir los subconjuntos de los usuarios privilegiados a partir de la información que le proporciona el proveedor de que usuarios están revocados y cuales no. Una vez calculado el conjunto de subconjuntos que engloban a todos los usuarios privilegiados, se extrae la clave de cada subconjunto (K_x) y se cifra la clave AK con ésta. Cada EMM_x será enviado a los usuarios privilegiados de cada uno de los subconjuntos (x).

A favor:

Mediante este esquema se consigue reducir considerablemente el número de mensajes EMM que el proveedor de servicios debe enviar a los usuarios. De este modo, al ser más pequeño el número de mensajes EMM a ser enviados, el proveedor es capaz de modificar la AK con más frecuencia, con la consecuente mejora de seguridad que ello implica.

Lo más importante de todo es que esta mejora se ha introducido bajo el sistema ya existente y llevarlo a la práctica no costaría "demasiado", ya que es totalmente compatible. Simplemente se tendría que implantar el esquema de *Broadcast Encryption* en el centro emisor y cambiar todas las *smart cards*.

En contra:

El hecho que se hayan reducido el número de mensajes EMM que tienen que ser enviados por parte del centro emisor, provoca que sea necesario almacenar un conjunto de claves en la *smart card* y que ésta sea capaz de calcular claves a partir de otras ya almacenadas. Por este motivo, serán necesarias *smart cards* más potentes y con mayor capacidad de las que se necesitaban hasta el momento y por lo tanto, más caras.

La utilización de este esquema junto al correcto uso de los elementos criptográficos, deberían ser suficientes para garantizar la seguridad de todo el sistema.

4.3.2 Segundo esquema propuesto

En este segundo esquema se intenta huir un poco del estándar del DVB, ya que éste presenta una jerarquía de claves y de mensajes que, a priori, no son necesarios si se hace uso de los esquemas de *Broadcast Encryption*. Este segundo esquema propuesto es una combinación de ambos sistemas, *Broadcast Encryption* y Acceso Condicional. De este modo, se convertirá en un esquema que dejará de cumplir las especificaciones del estándar DVB y por otro lado, el propio mensaje BE dejará de ser como el de la teoría.

La idea es coger el mensaje BE y separar el cuerpo del mensaje de la cabecera. Recordar que en el cuerpo del mensaje se encuentra cifrada la información, que se desea difundir, y en la cabecera se encuentra la información necesaria para poder descifrarla.

$$\left\langle \left[\text{info}_1,...,\text{info}_\text{m} ; \varepsilon_{K_1}(SK),...,\varepsilon_{K_m}(SK) \right]; \varepsilon_{SK}(M) \right\rangle$$

Comparando el número de claves que se utilizan para enviar la información en los dos sistemas, se puede observar que en el esquema de BE se utiliza una clave menos que en los CAS. Las 3 claves utilizadas en los CAS son: la MPK, la AK y la CW. Mientras que las 2 claves utilizadas en los esquemas de BE son: la K_x y la *SK*.

Observando los dos sistemas, no es difícil darse cuenta que tanto la clave CW del CAS como la clave *SK* de esquema BE tienen la misma finalidad, se trata de una única clave mediante la cual se cifra/descifrar la información difundida. Por otro lado, la equivalencia entre la MPK y la AK con la K_x es un poco más complicada de ver. La K_x es una clave

privada a nivel de subconjunto, tal y como la MPK a nivel de usuario. Por este motivo, se encuentran almacenadas en el dispositivo del usuario, ya sea en la *smart card* o en el propio receptor. Así como la MPK es única, la K_x es diferente para cada subconjunto de usuarios. De este modo, el usuario debe almacenar tantas K_x como posibles subconjuntos haya a los que pueda pertenecer. Por otro lado, para ahorrarse la comunicación unicast de volúmenes de información elevados, los CAS utilizan la AK. Se trata de una clave única para todo el sistema pero que varía con el tiempo. La funcionalidad de la AK, a pesar de ser dinámica y común para todo el sistema, es análoga a la K_x, ya que como ésta, se usa para cifrar/descifrar la clave de cifrado del contenido difundido.

Vistas las equivalencias entre las claves la idea es construir un sistema que haga uso del BE partiendo del estándar de sistema de acceso condicional. Para ello, lo primero de todo es darse cuenta de los inconvenientes que presenta el envío del mensaje de BE. Por un lado se tiene la longitud de la cabecera y por el otro los retardos en la reproducción del contenido.

Una longitud de cabecera excesivamente larga puede provocar un incorrecto funcionamiento del sistema, esto sucede si el ancho de banda necesario para difundir los mensajes BE es mayor que el disponible. Este hecho provocaría la necesidad de reducir la calidad de la información (video/audio) difundida, con tal de ceñirse al ancho de banda disponible. A continuación se comenta un posible ejemplo numérico que describe esta situación.

Supongamos que la tasa del flujo de datos de audio/video es de 5000 kbps. Para poder cumplir con esta tasa, cada segundo se envían 5 mensajes de BE que contienen 1000 kb de audio/video codificados. Suponiendo que el ancho de banda del canal sea de 6000 kbps y que la cabecera de cada uno de los mensajes es de 150 kb, no habría ningún problema, ya que el ancho de banda consumido sería de 5750 kbps. Pero en el caso que la longitud de la cabecera aumentase y fuese superior a los 200 kb el sistema no lo soportaría, ya que el ancho de banda necesario para transmitir los mensajes sería superior al disponible.

Existen 2 posibles soluciones a este problema, por un lado reducir la tasa de los datos de audio/video, con su consecuente reducción de calidad en la imagen y en el sonido y la otra solución sería reducir el número de mensajes BE a transmitir por segundo. La consecuencia directa de reducir el número de mensajes de BE por segundo es el posterior retardo en la reproducción, siempre y cuando el descifrado se realice una vez recibido la totalidad del mensaje. En el caso que el descifrado se haga a medida que se van recibiendo los datos cifrados, este proceso no introducirá un retardo provocado por la longitud del mensaje, pero si que se vería afectado aquel usuario privilegiado que selecciona el canal en cuestión en un

determinado instante posterior a la recepción de la cabecera del mensaje, este usuario no visualizará el contenido hasta la recepción del siguiente mensaje BE.

Basándome en el funcionamiento de los CAS, dados los problemas que surgen a la hora de difundir el mensaje BE como tal, se plantea la posibilidad de tratar a la cabecera del mensaje de forma análoga a los mensajes EMM/ECM de los CAS y al cuerpo del mensaje BE como la contenido codificado. De este modo, tanto la cabecera como el cuerpo del mensaje serán enviados por separado y será el receptor el encargado de construir el mensaje BE en destino. El hecho de deshacer el mensaje BE permite trabajar de un mayor número de formas ajustándose a las necesidades del sistema. Por ejemplo, la cabecera se puede fragmentar creando así mensajes de acreditación más pequeños, pudiendo enviarse estos mensajes, que contienen la clave *SK*, por anticipado, es decir, enviar las claves de cifrado/descifrado de tal forma que cuando llegue el contenido cifrado al receptor, todos los usuarios ya posean su clave de descifrado. En la *Figura 39* se muestra un poco la idea. En este caso la cabecera ha sido dividida en tres fragmentos y estos fragmentos se envían antes de enviar el contenido cifrado con la *SK* que contienen los fragmentos.

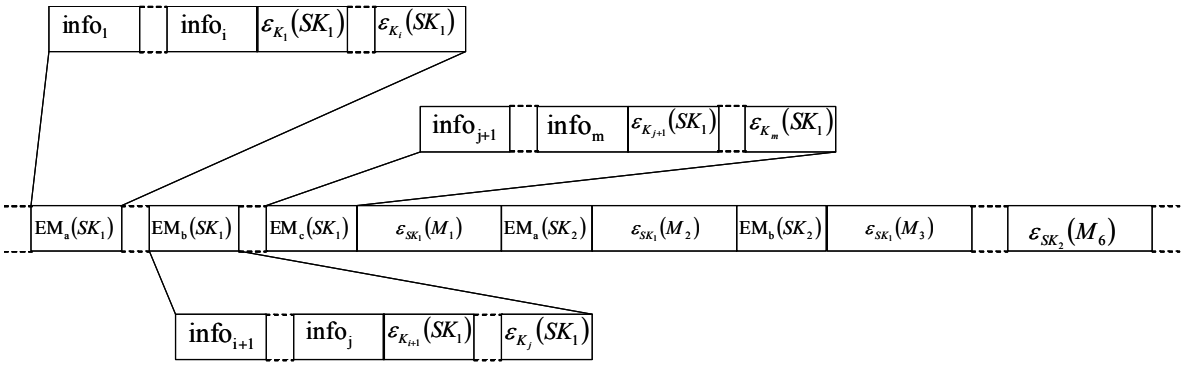

Figura 39: Flujo de información enviado por parte del centro emisor

Como se puede observar, los $EM_x(SK_i)$ son mensajes que contienen fragmentos de la cabecera del mensaje de *Broadcast Encryption* y son enviados antes de enviar el contenido M_j cifrado con SK_i.

A continuación se presenta la figura del esquema propuesto:

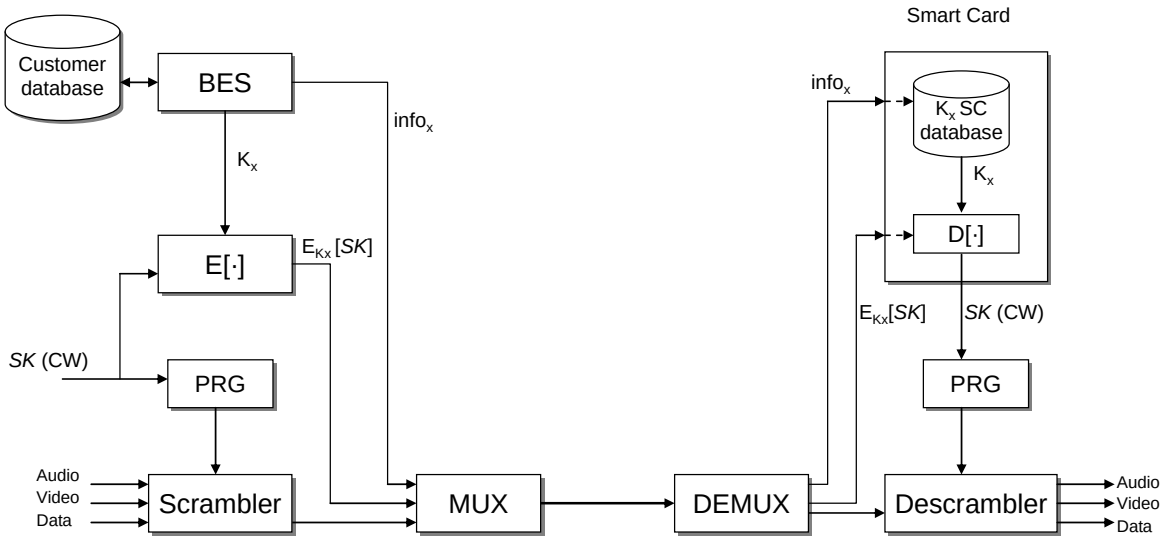

Figura 40: Segundo esquema propuesto

A favor:

El uso de alguno de los esquemas de *Broadcast Encryption* en este sistema permite reducir el TO del CAS. El hecho de fragmentar la cabecera de los mensajes de BE, como ya se ha dicho, permite una mayor versatilidad en el funcionamiento del sistema. Nótese que pueden haber tantos fragmentos de contenido codificados con la misma clave como se desee. No es necesario enviar la clave *SK* cifrada para cada fragmento de contenido enviado, siempre y cuando la *SK* no varíe.

En contra:

Resulta evidente que mediante este esquema se han dejado de cumplir con las especificaciones del estándar DVB, los nuevos mensajes enviados por el centro emisor no son ni ECM ni EMM. Por este motivo, muy probablemente no se podrán utilizar los receptores que hay en el mercado, por lo que sería necesario reabastecer a todos los usuarios de nuevos receptores en el caso que se decidiese implantar el esquema.

Además de cambiarle los receptores a los usuarios, también sería necesario cambiarles las *smart cards*, ya que tal y como sucedía para el es primer esquema propuesto, las *smart cards* de este esquema deben tener una mayor capacidad de almacenamiento y de cálculo. A parte de que cada usuario deberá tener un conjunto de nuevas claves almacenadas en éstas que sólo podrán ser almacenadas por el proveedor en el proceso de fabricación.

Por otro lado, a nivel de seguridad, este nuevo esquema no permitirá cambiar la *SK* con tanta frecuencia como lo hacía el CAS, debido al TO del esquema de *Broadcast*

Encryprion. Es por este motivo que para garantizar la seguridad entre el cambio de una *SK* y otra, será necesario que la longitud de la *SK* sea superior a la CW del CAS. No obstante, hay que tener en cuenta que el intervalo de vida de la *SK* será mucho inferior al de la clave AK del CAS.

5 Advanced Access Content System (AACS)

Proteger el contenido multimedia como el video y el audio de la piratería es un problema considerable para la industria. El contenido multimedia debe ser empaquetado y distribuido a los usuarios finales, de tal modo que ningún otro usuario que no pertenezca al conjunto de usuarios privilegiados definidos por el propietario del contenido, sea capaz de usarlo. Al sistema diseñado con este propósito se le conoce con el nombre de *content distribution system* (sistema de distribución de contenidos). Además, el sistema de distribución de contenidos debe asegurarse de que los usuarios privilegiados no posean la habilidad de distribuir el contenido sin permiso.

En este tema se va a explicar uno de los sistemas en el que el propietario del contenido realiza la distribución del mismo a través de medios de almacenamiento físicos, más concretamente discos ópticos (CD, DVD, HD-DVD y Blu-Ray). Existen un gran número de sistemas criptográficos desarrollados para la distribución de contenidos en medios físicos, pero en este tema me voy a centrar en el Advanced Access Content System (AACS) [77].

El propósito de dedicar un tema entero a este sistema es proporcionar la visión general de un sistema en el que se ha implementado un esquema de *Broadcast Encryption* en una aplicación comercial y cuales han sido sus resultados. La difusión de Televisión de Pago y la de contenidos multimedia mediante medios ópticos tienen un mismo objetivo. Éste objetivo común es que el contenido de éstos sólo pueda ser usado por aquellos usuarios a los cuales el centro emisor les haya dado permiso. La principal dificultad es que tanto una parte de los receptores de Televisión de Pago como los reproductores de medios ópticos son de tipo *stateless* y de ahí la importancia del uso de sistemas de *Broadcast Encryption*.

El AACS es un sistema de protección anticopia para los discos de alta definición. Se basa en meter la clave de descifrado de la película dentro de mismo disco que la contiene, pero para acceder a ella se requiere un lector óptico autorizado y un reproductor software autorizado. De este modo, en teoría, no se puede construir simplemente un “ripeador” que te copie el contenido al disco duro y ya está, ya que ese “ripeador” no estará autorizado y no podrá obtener la clave para leer el contenido.

El Advanced Access Content System (AACS) fue desarrollado por ocho compañías [77]: Disney, IBM, Intel, Matsushita (Panasonic), Microsoft, Sony, Toshiba y Warner Brothers.

A pesar de que el AACS se pueda usar tanto para los contenidos pre-grabados en el medio físico, como para contenidos que van a ser grabados, en este tema me centraré en el de contenidos pregrabados.

5.1 Antecesores

El AACS es el sucesor del Content Scramble System (CSS) [65] y del Content Protection for Recordable/Pre-Recordable Media (CPRM/CPPM).

El CSS es un sistema de distribución de contenidos físicos usado en los DVD. Fue introducido por la Asociación de Control de Copias (CCA) de DVD en el año 1996. El CSS sigue la estructura básica de un sistema de distribución de contenidos físicos, en el que el estudio de cine proporciona el contenido y las normas de uso al "replicador" con licencia para la distribución. El CSS está basado en un cifrador de flujo propietario que fue roto en el 1999, poco después de que el formato DVD empezase a cobrar popularidad. El CSS se consiguió romper ya que éste no proporciona la propiedad de revocar una determinada clave del sistema que haya sido comprometida.

El CPRM/CPPM es un sistema de distribución de contenidos físicos que se usa principalmente para la protección de DVD-audio, aunque también se usa en tarjetas de memorias flash, como es el caso de las tarjetas Secure Digital. El CPRM/CPPM fue creado en el año 2000 por 4C Entity, LLC, el cual es el cosorcio fundado por IBM, Matsushita y Toshiba. Se construyó para ser más robusto que su predecesor, incorporando la revocación de claves de dispositivos comprometidos y capacidades de *traitor tracing*. El ACCS incorpora el mismo esquema de *traitor tracing* que el CPRM/CPPM, permitiendo localizar a un dispositivo comprometido a pesar de que la clave comprometida no se haya hecho pública.

5.2 Cifrado y descifrado de medios pre-grabados

Se refiere a medios pre-grabados a todos aquellos medios que se distribuyen con el contenido ya cargado, aunque en las implementaciones actuales, generalmente hacen referencia a los discos de video pre-grabado HD-DVD y Blu-Ray, a juegos o software. Por

otro lado, los medios grabables suelen hacer referencia al conjunto de discos HD-DVD y Blu-Ray en blanco en los que los consumidores pueden grabar su propia información. Los detalles del AACS para el cifrado/descifrado de contenidos pre-grabados están publicados por las 8 empresas ya mencionadas [60], [61] y [62].

El "Replicador" es la parte que lleva a cabo el cifrado del contenido, mientras que el descifrado lo realiza el dispositivo reproductor del cliente. La *Figura 41* muestra un esquema de los procesos de cifrado y descifrado del contenido.

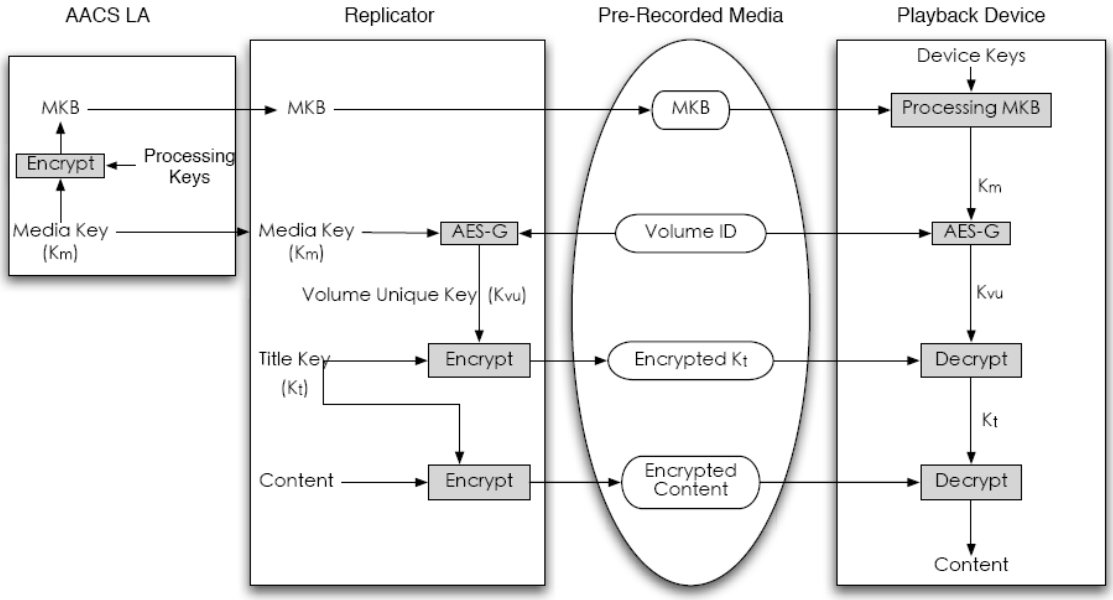

Figura 41: Cifrado y descifrado de la información pregrabada

Antes de explicar los procedimientos de cifrado y descifrado, se introducirán algunos términos y elementos criptográficos usados en el AACS.

Título: El propietario del contenido, por ejemplo PIXAR, proporciona el contenido a un "Replicador" con licencia en forma de uno o más Títulos. Por ejemplo, una película, tal como WALL-E.

Title Key (K_t): El "Replicador" genera aleatoriamente una Title Key de 128 bits para cada Título. Bajo el criterio del "Replicador", la Title Key se puede usar para el cifrado de todas las instancias de un Título dado, o usar diferentes Title Keys para distintas instancias.

Volume ID: Los "Replicadores" también generan aleatoriamente un identificador de 128 bits para cada Título, llamado Volum ID. Éste es posteriormente grabado en un área especial del disco llamada "burst cut area". Esta área especial, tanto en los discos grabables como ya grabados, es de solo lectura para las grabadoras de los consumidores. El propósito del Volume ID es el de proteger el contenido contra las copias bit a bit. Del mismo modo que para la Title Key, el mismo Volume ID puede ser usado para todas las instancias de un Título dado, o usar diferentes Volume IDs para distintas instancias.

Media Key (K_m): La Media Key es proporcionada por la Autoridad de Licencias del AACS (la AACS LA), y se usa junto al Identificador de Volumen para calcular la Clave de Volumen Única. Como la mayoría de claves del AACS, la Media Key tiene una longitud de 128 bits.

Volume Unique Key (K_{vu}): La Volume Unique Key se usa para el cifrado de la Title Key. Se trata de la salida de una función de único sentido, AES-G, la cual toma como entradas al Identificador de Volumen y a la Media Key.

AES: El Advanced Encryption Standard (AES) es el encargado de cifrar/descifrar los bloques de información en el AACS. El contenido es cifrado/descifrado usando el AES en modo CBC [*Anexo* 9.2.2]. El conjunto de claves se cifran/descifran usando el AES en modo EBC. Dado que la mayoría de claves son de 128 bits, sólo será necesario un cifrado/descifrado de las claves de cifrado/descifrado.

AES-G: El AES-G es una función de único sentido basada en AES que a partir de dos entradas de 128 bits genera una salida del mismo tamaño. El AES-G se describe en la *Figura 42*.

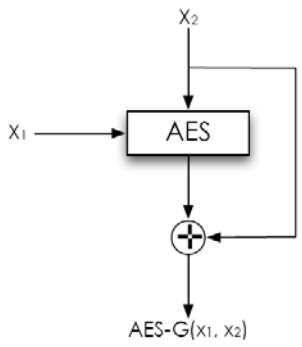

Figura 42: Función AES-G

Device Key: La Device Key tiene el mismo propósito que las etiquetas en el esquema Subset Difference. Cada una tiene una longitud de 128 bits y se encuentra almacenada en el dispositivo reproductor.

Processing Key: Cada Processing Key tiene una longitud de 128 bits y se puede derivar de la clave del dispositivo. Las claves de procesado se usan para cifrar la Media Key.

Media Key Block (MKB): El AACS LA es el encargado de proporcionar el Media Key Block y contiene información sobre dispositivos revocados. La Media Key se cifra con una o más claves de procesado (Processing Keys) y se sitúa en el Media Key Block junto a la información que indica las claves de procesado empleadas en el cifrado, es equivalente a la cabecera del mensaje de los esquemas de *Broadcast Encryption* vistos en temas anteriores.

Cifrado del contenido:

El propietario del contenido entrega el contenido a un "replicador" con licencia, el cual genera aleatoriamente una Title Key secreta para el contenido. Posteriormente, el contenido se cifra con la Title Key usando el AES en modo CBC.

Para cada Título (o conjunto de Títulos), el AACS LA proporciona un MKB y una Media Key al "Replicador". Las Media Keys cifradas se encuentran incluidas en el MKB y sólo los usuarios privilegiados son capaces de descifrar una de ellas para así conseguir la Media Key. Para que sólo los usuarios privilegiados puedan descifrar la Media Key, el AACS LA usa el esquema *Subset Difference* para cifrarla.

El replicador genera aleatoriamente un Volume ID para el contenido. A continuación, usa el Volume ID junto a la Media Key para calcular la Volume Unique Key mediante la función AES-G. La Title Key se cifra con la Volume Unique Key usando el AES.

Finalmente, el "replicador" graba el MKB, el Volume ID, la Title Key cifrada y el contenido cifrado en el disco, junto a un conjunto de información no relacionado con el cifrado de contenidos. Hay que destacar que el Volume ID se graba en el "Burst cutting area", lugar del disco que es de solo lectura para los lectores/grabadores comerciales.

Descifrado del contenido:

Si el dispositivo reproductor no está revocado, se tienen que usar las Device Keys para obtener a partir de éstas la Media Key descifrada. Las Media Keys cifradas se encuentran grabadas en el MKB.

Posteriormente, el dispositivo reproductor lee el Volume ID del disco y lo usa junto con la Media Key como entradas de la función AES-G para calcular la Volume Unique Key. Una vez obtenida la Volume Unique Key, se puede obtener la Title Key descifrándola con ésta. Finalmente, el dispositivo reproductor es capaz de descifra el contenido mediante la Title Key.

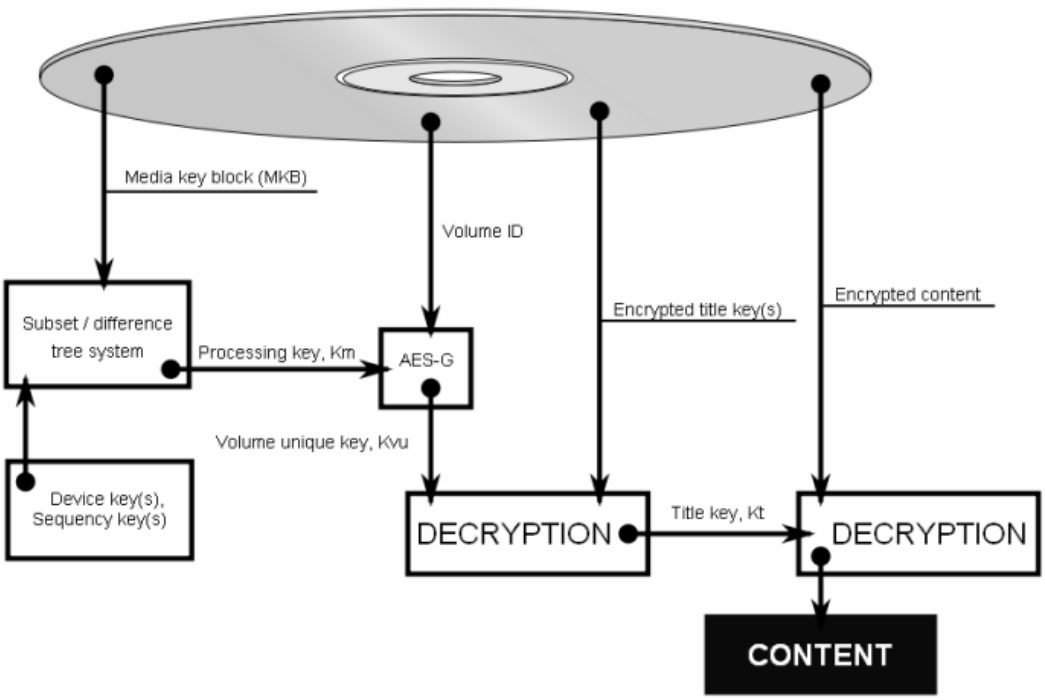

Figura 43: Descifrado del contenido

Revocación del contenido

Existe una forma de realizar la autenticación des del disco al dispositivo reproductor que consigue la revocación del contenido. El mecanismo de revocación de contenidos añade una capa de control sobre el contenido. Este permite que la AACS LA pueda revocar ciertos títulos, prevenir de alteraciones maliciosas de contenidos lícitos y deshabilitar la reproducción de títulos no autorizados. En la *Figura 44* se muestra un diagrama del mecanismo de revocación de contenidos.

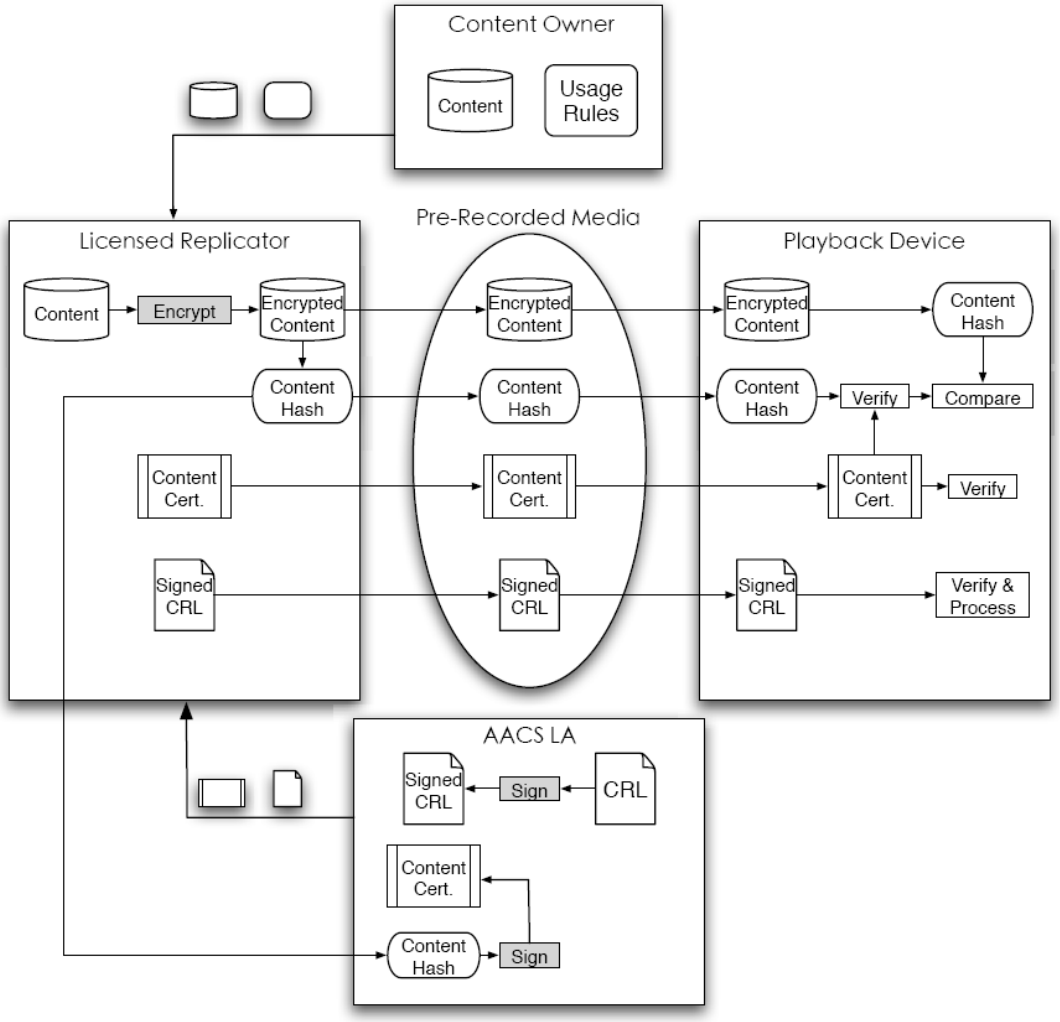

Figura 44: Revocación de contenido de información pregrabada

El propietario del título proporciona el título y las reglas de uso a un "replicador" con licencia. El "replicador" realiza el cifrado del contenido tal y como se ha descrito anteriormente. A continuación, se calcula el *hash* del contenido cifrado. Posteriormente, se envía el *hash* a la AACS LA. El "replicador" liga criptográficamente la Title Key y las reglas del uso para prevenir de cualquier tipo de alteración maliciosa de estas reglas.

Una vez la AACS LA recibe el *hash* del contenido, ésta firma el contenido usando la clave pública de la AACS LA para hacer el certificado del contenido y posteriormente enviárselo al "replicador". Junto con el certificado, la AACS LA también envía una lista de revocación de contenidos (CRL) puesta al día. La CRL contiene una lista de los títulos revocados que han sido firmados y que poseen certificados válidos, pero que no deberán ser reproducidos por los reproductores comerciales. El "replicador" graba el CRL, el *hash* del contenido y el certificado en el disco.

Cuando se inserta el disco pregrabado en el reproductor, el dispositivo verifica el certificado del contenido y la CRL. Si alguna de las dos verificaciones falla, el dispositivo debe parar el procesado del disco. El dispositivo calcula un *hash* del contenido cifrado del mismo modo que lo hizo el "replicador". Posteriormente, el *hash* es comparado con el almacenado en el disco y si no coinciden, el procesado del disco se debe abortar. Cada dispositivo comercial debe poseer al menos 128 kilobytes de almacenamiento no volátil con el propósito de almacenar la más reciente CRL. El dispositivo comprueba si el contenido insertado en el dispositivo está listado en la CRL, si lo está, el dispositivo debe terminar de acceder al contenido.

5.3 Autenticación de la Unidad y el host mediante acuerdo de clave

Para diseñar un esquema de protección de los medios de información que sea capaz de ejecutarse en plataformas abiertas como PCs, los diseñadores tienen que asegurarse de que no sea susceptible al ataque de "virtualización del dispositivo". Un dispositivo virtual puede simular un dispositivo físico en todos los aspectos a fin de que la CPU sea engañada y crea que existe un dispositivo cuando en realidad no lo hay. Para implementar un ataque de dispositivo virtual a un sistema de medios de información, tales como los sistemas de reproducción de DVD, el atacante puede construir software que implementa una unidad de DVD virtual. El contenido de los discos ópticos se traslada al disco duro del ordenador como una imagen del disco. El atacante puede entonces reproducir este "disco DVD" a través de la unidad de DVD virtual en un software reproductor de DVD legítimo.

El atacante puede duplicar la imagen del disco en múltiples copias y difundirlas ilegalmente, a pesar de que él nunca conozca el contenido del DVD. Con el fin de defenderse contra este ataque, la unidad tiene que tener la capacidad de demostrar que el host (por ejemplo, el software de reproducción) es una unidad legítima. Esto se puede conseguir por medio de un protocolo criptográfico de autenticación.

El esquema de autenticación AACS unidad-host realiza una autenticación mutua, que significa que no sólo la unidad demuestra su identidad legítima al host, sino que también el host tiene que demostrar su identidad a la unidad. Después de que la unidad y el host completan con éxito el período de sesiones de protocolo de autenticación, se establece entre ellos una clave secreta compartida. Por lo tanto, el protocolo de autenticación mutua del

AACS unidad-host se combina con un protocolo de acuerdo de clave. La clave secreta compartida se utiliza para propósitos de autenticación de mensaje.

5.3.1 Protocolo de autenticación mutua y protocolo de acuerdo de clave

En un protocolo de autenticación mutua, las dos entidades participantes necesitan demostrar sus identidades los unos a los otros. Si una entidad ha demostrado con éxito su identidad a la otra entidad, la otra entidad está obligada a "aceptarle". Un período de sesiones de protocolo de autenticación mutua se completa con éxito si ambos participantes se aceptan hasta el final del período de sesiones. Los Protocolos de autenticación mutua se pueden diseñar utilizando ya sea esquemas criptográficos de clave simétrica o asimétrica.

Después de que las dos entidades se hayan autenticado la una a la otra, lo más normal es que quieran comunicarse entre ellas. Por lo que tiene sentido combinar un protocolo de acuerdo de clave o protocolo de intercambio de claves con uno de autenticación mutua, porque una clave secreta compartida proporciona confidencialidad y / o integridad de los datos a ambas entidades. En un protocolo de acuerdo de clave, ambas entidades contribuyen con información que se utiliza para obtener una clave secreta compartida. Un protocolo de acuerdo clave utiliza la mayoría de las veces un esquema de clave asimétrica. En un protocolo de intercambio de claves, la clave secreta compartida no es necesariamente producto de la información aportada por las entidades participantes, y es difícil de conseguir un protocolo de intercambio de claves seguro.

5.3.2 Descripción del esquema de autenticación del AACS Unidad-Host

Cuando se usa el AACS en un sistema basado en PC, la unidad y el host son entidades separadas, tanto la unidad y el host poseen certificados expedidos por parte de la AACS LA. Estos certificados, denominados certificado de la unidad y certificado del host, contienen campos donde se indica la capacidad del dispositivo, un identificador único, la clave pública del dispositivo y una firma de la AACS LA, que verifica la integridad del certificado firmado con una clave privada de la AACS LA. Tanto la unidad como el host tienen la correspondiente clave pública de la AACS LA para la verificación de la firma. Se puede encontrar una descripción completa del formato de certificado de en [59].

Cada vez que se coloca un nuevo medio en la unidad, se produce el proceso de autenticación entre la unidad y el host. Esto es necesario porque el nuevo disco puede contener listas de revocación actualizadas. Cada disco compatible contiene unaestructura de datos llamada Media Key Block (MKB), que posee la información necesaria para obtener las claves para descifrar el contenido. También contiene la última lista de revocación de la unidad (DRL) y lista de revocación de host (HRL), que respectivamente, contienen una lista de identificadores de las unidades revocadas y una lista de identificadores de los hosts revocados. Una unidad sólo puede comunicarse con un host que no haya sido revocado, y del mismo modo un host sólo podrá comunicarse con una unidad que no haya sido revocada.

En el *Anexo 9.5* se puede consultar el flujo del esquema de autenticación de la AACS unidad-host.

Después de completar con éxito el algoritmo de autenticación de la unidad y el host, ambas establecen una clave de bus compartida basada en el protocolo de acuerdo clave de Diffie - Hellman de curva elíptica. Es interesante observar que, si bien esta clave podría ser utilizada para cifrar los mensajes entre la unidad y el host, no se utiliza realmente para este fin. En lugar de ello, la clave de bus se utiliza únicamente para la autenticación de mensajes mediante la inclusión de un MAC para cualquier mensaje que viaje entre la unidad y el host.

5.3.3 Análisis del esquema de autenticación del AACS unidad-host

En este esquema, se pueden destacar principalmente dos debilidades. La primera de ellas se encuentra en los primeros cuatro pasos del esquema de autenticación. Supongamos que la DRL en el MKB es más reciente que la almacenada en la DRL host. Un usuario malicioso, puede cambiar el número de versión MKB a un una edad avanzada, y enviar la modificación de MKB' al host. Esta modificación podría no ser detectada durante el procedimiento de autenticación, ya que, según las especificaciones, el host primero comprueba el número de versión de la MKB, y si el número de la versión es más antiguo que su DRL, salta al paso 2, que implica la verificación de la firma de la DRL en el MKB.

Si la unidad ya ha sido revocada, se podría alterar maliciosamente el número de la versión del MKB, con el fin de no dejar que el receptor actualice su DRL, de modo que pueda mantener la interacción con el host.

La segunda debilidad es la siguiente: Supongamos que A y B son dos entidades participantes honestas tratando de establecer una clave secreta a través de un protocolo de acuerdo de clave y X es una entidad maliciosa. En este ataque, X suplanta a las entidades honestas, es decir, A cree estar compartiendo una clave con X, cuando en realidad lo está haciendo con la otra entidad honesta B y ésta, cree que está compartiendo la clave con A. Por lo tanto, al final del protocolo, X puede actuar en el nombre de B para interactuar con A.

Este ataque podría ser explotado en la práctica. Por ejemplo, supongamos que se revoca a la unidad A. A continuación, se puede emplear este ataque preguntando a la unidad B, que no está revocada, para suplantarla. Dado que el host sólo ve el certificado de la unidad B, el procedimiento de la autenticación debería concluir con éxito. De esta manera, la unidad A todavía puede interactuar con el host después del procedimiento de autenticación.

5.4 La seguridad del AACS en el mundo real

Puede parecer que el protocolo de cifrado usado por el AACS es robusto, pero entonces, ¿Por qué se han pirateado tantas películas en HD-DVD y en Blu-ray? La verdad es que los atacantes han sido capaces de recuperar algunas de las claves usadas en el proceso de cifrado del contenido. Des del 27 de diciembre del 2006 [66], los atacantes han sido capaces de localizar la Title Key para cualquier título llevando a cabo el volcado de memoria (RAM) del software de decodificación WinDVD y PowerDVD, basados en Microsoft Windows. Una vez la Title Key está comprometida, cualquiera que tenga el título correspondiente puede descifrar el contenido cifrado libremente. El 11 de febrero del 2007 [68], los atacantes recuperaron dos claves de procesado tanto de los discos HD-DVD como de los Blu-ray en otro volcado de memoria del WinDVD. El conocimiento de estas dos claves permite descifrar todos los discos publicados antes de la revocación. El 24 de febrero del 2007 [67], los atacantes consiguieron extraer algunas de las claves del dispositivo almacenadas en WinDVD.

El 16 de abril del 2007, la AACS LA anunció que las claves de dispositivo para el software de descodificación fueron finalmente revocadas. La AACS LA culpó a la implementación pobre por parte del software para garantizar el compromiso en vez de alguna debilidad propia del sistema. No obstante, la filtración de únicamente dos claves de procesado ha permitido que se haya podido copiar libremente todas las películas en formato HD-DVD y Blu-ray publicadas antes de abril del 2007.

Desafortunadamente para la AACS LA y para los estudios de cine, el 23 de Mayo del 2007 se posteó en una Web de Internet una nueva clave de procesado. Esta nueva clave no se revocó hasta el 7 de septiembre del 2007. El motivo que provocó que la AACS LA tardase tanto tiempo en revocar la clave se debe a que los fabricantes necesitan rediseñar sus productos con el fin de que los atacantes no vuelvan a tener éxito. Si el fabricante simplemente repone las nuevas claves, el atacante conseguirá extraer las claves realizando el mismo procedimiento que aplicó para extraer las antiguas. Con el fin de tener suficiente tiempo para actualizar su producto, el AACS Interim Adopter Agreement dice que los fabricantes pueden recibir una notificación 90 días antes de que sus claves sean revocadas.

Las malas noticias para la AACS LA no acabaron ahí, ya que SlySoft, una compañía basada en Antigua, tiene el producto *AnyDVD HD* que es capaz de sortear de forma efectiva el AACS y piratear la mayoría de títulos en HD-DVD y Blu-ray publicados hasta la fecha. AnyDVD HD es un driver basado en Microsoft Windows que trabaja en segundo plano y proporciona el descifrado de información en alta definición. De este modo puede cooperar con *CloneDVD*, el cual es otro producto de SlySoft. AnyDVD usa las claves de procesado y de dispositivo, que todavía no han sido revocadas, para descifrar los títulos en HD-DVD y Blu-ray. La AACS LA probablemente está en proceso de averiguar cuales son esas claves comprometidas. De nuevo, sin que el esquema AACS haya sido roto, los atacantes siguen siendo capaces de extraer las claves de dispositivo y de procesado.

Si el ciclo de revocación de las claves es siempre mayor al tiempo necesario por parte del atacante de extraer las claves secretas, la AACS tendrá la batalla perdida. En realidad, si el atacante sólo necesita algunos meses para extraer las claves, la efectividad del AACS para proteger la propiedad intelectual se verá claramente menguada en relación a la longitud del ciclo de revocado, ya que sólo serán necesarios pocos meses para que la copia pirateada aparezca posteriormente a la publicación de la auténtica.

Las claves de título son como las claves de sesión, una clave de sesión es responsable para una sesión en concreto, que en el caso del AACS representa a toda la distribución de una película. El que una clave de sesión esté comprometida, sólo debería afectar a una única sesión, mientras que las otras sesiones no se deberían ver afectadas, siempre y cuando sus claves de sesión no se vean comprometidas. Debido a la naturaleza *stateless* de los reproductores en AACS, las claves de procesado y del dispositivo pueden tener un periodo de vida más largo que el del propio dispositivo. Es deseable que aunque una clave de sesión este comprometida, esto no comprometa también a las claves de dispositivo y de procesado.

6 Conclusiones

Los múltiples trabajos y estudios sobre el *Broadcast Encryption* hacen presagiar que no estamos lejos de una posible implantación de estos esquemas en los Sistemas de Acceso Condicional de la televisión de pago. Los actuales Sistemas de Acceso Condicional, por lo general, acostumbran a ser seguros durante sus primeros días de vida. La historia ha demostrado que tarde o temprano una buena parte de estos sistemas acaban siendo vulnerados.

Gran parte de la culpa que estos sistemas dejen de proporcionar seguridad frente a cierto grupo de usuarios, reside en los propios proveedores de los servicios. Éstos, acostumbran a calificar a su sistema de seguro, pero eso sí, siempre manteniendo los detalles de éste en secreto. Lo ideal, tal y como sucede con los algoritmos criptográficos, sería que los detalles del sistema de acceso condicional fuesen públicos. De este modo, se podría medir el grado de seguridad que ofrece el sistema como tal y no el grado de seguridad fruto del desconocimiento de su funcionamiento.

El no hacer pública las especificaciones del sistema refuerza la idea de que el sistema sea inseguro bajo el conocimiento de ciertos detalles. Gran parte de los sistemas rotos hasta la fecha poseían debilidades de este tipo. Un claro ejemplo son los procesos de camuflaje y el uso repetitivo de las ciertas claves.

Otro aspecto de seguridad a tener en cuenta en los sistemas actuales de acceso condicional es el empleo de los mecanismos criptográficos de la forma adecuada. Es decir, siguiendo los procedimientos que garantizan la seguridad de los algoritmos. Un posible ejemplo es la longitud de la clave. La longitud de la clave (secreta o pública) de un algoritmo debe ser lo suficientemente larga para garantizar la seguridad del contenido cifrado durante un intervalo de tiempo adecuado.

Las dimensiones y limitaciones de los sistemas juegan un papel en contra de la seguridad de éstos. Existen dos claras limitaciones de los sistemas, que son por un lado la poca capacidad de cálculo de las *smart cards* y por el otro el ancho de banda del canal de televisión.

Las *smart cards* poseen un pequeño procesador de datos en su interior que es el que les permite realizar los cálculos, pero éste posee unas características bastante limitadas, por lo que resulta muy inadecuado para cierto tipo de cálculos, como es el caso de las operaciones exponenciales. Es por este motivo, por el cual los algoritmos criptográficos elegidos para ser

usados en los sistemas de acceso condicional deben poseer unas características concretas que les permita ser usados por las *smart cards*.

Por otro lado, existe una relación directa entre el número de usuarios del sistema y el ancho de banda utilizado por éste. Esto se debe a las transmisiones unicast de los mensajes EMM. El hecho de que estos mensajes consuman tanto ancho de banda provoca que la clave AK no varíe con tanta frecuencia como la que se desearía.

La variación de la clave AK cada pocas horas ayudaría a que un gran número de usuarios se quedasen a oscuras (no podrían acceder al contenido). Actualmente, buena parte de la población de usuarios ilícitos obtiene la clave AK a través de Webs de Internet en donde se publican éstas. Esto es posible gracias a que las claves AK actuales suelen tener una vigencia de muchos días, incluso meses. El hecho de variar la clave AK cada pocas horas dificultaría este procedimiento.

Los esquemas de *Broadcast Encryption* se presentan como una clara ayuda para reducir el ancho de banda consumido, fruto del envío de los mensajes EMM. La idea principal de estos esquemas es buscar el mayor número posible de usuarios que compartan una misma clave, para que de este modo el envío de las claves de descifrado ocupe el menor ancho de banda. Eso sí, en esta búsqueda de la clave común para un subconjunto de usuarios también intervienen otros factores como son el espacio de almacenamiento de claves y el coste computacional necesario para obtener ciertas claves.

Los tres factores (TO, CC y SS) están relacionados, por lo que la mejora de uno de los factores conlleva a empeorar alguno de los otros dos. Lo ideal es encontrar un esquema que mantenga el equilibrio entre los tres factores y que estos sean lo más pequeños posibles. Pero como esto resulta muy complicado, hay que buscar, entre los esquemas de *Broadcast Encryption* ya existentes, aquel que se adapte mejor a las especificaciones del sistema.

Las necesidades del sistema de acceso condicional de la televisión de pago son, principalmente, un ancho de banda reducido ligado a una capacidad de cálculo limitada por parte de las *smart cards*. Una elección del esquema de *Broadcast Encryption* adecuado sería el *Layered Subset Diference*, el cual a pesar de no ser el más adecuado en cuanto al TO, presenta unos valores bastante aceptables de CC y SS.

Se han propuesto 2 formas para implantar el esquema de *Broadcast Encryption* en los servicios de televisión. La primera es aprovechando el sistema de acceso condicional ya existente. En esta opción se ha buscado mantener la compatibilidad con el modelo de sistema de acceso condicional utilizado por el DVB. La segunda forma es diseñando un nuevo sistema que no es compatible con los estándares del modelo funcional del sistema de acceso

condicional del DVB. Ambas propuestas presentan ventajas y desventajas respecto al modelo de sistema de acceso condicional actual.

Aunque parezca que todavía se está lejos de la implantación de los esquemas de *Broadcast Encryption* en sistemas comerciales dada su "reciente" aparición, lo cierto es que ya ha habido un sistema comercial que ha usado el esquema de *Broadcast Encryption Subset Difference* con éxito. Este sistema es el AACS. Éste no ha proporcionado toda la seguridad que se esperaba de él en los dispositivos de almacenamiento ópticos de alta definición. Pero hay que destacar que las debilidades existentes son el fruto de otras partes del sistema y a día de hoy, la parte del sistema que se ocupa el esquema de *Broadcast Encryption Subset Difference* permanece segura.

7 Glosario de términos

3DES	3- Data Encryption Standard
AACS	Advanced Access Content System
AACS LA	Advanced Access Content System Licensing Authority
AES	Advanced Encryption Standard
AK	Authorization Key
BCS	Basic Chain Scheme
BE	Broadcast Encryption
BES	Broadcast Encryption Scheme
CA	Conditional Access
CAM	Conditional Access Module
CAS	Conditional Access System
CBC	Chiper Block Chaining
CBCS	C-basic Chain Scheme
CC	Computational Cost
CCA	Copy Control Association
CI	Common Interface
CPRM/ CPPM	Content Protection for Recordable/Pre-Recordable Media
CRL	Certificate Revocation List
CS	Complete Subtree
CSA	Common Scrambling Algorithm
CSS	Content Scramble System
CW	Control Word
DES	Data Encryption Standard
DRL	Device Revocation List
DVB	Digital Video Broadcasting
DVB-C	Digital Video Broadcasting – Cable
DVB-S	Digital Video Broadcasting - Satellite
DVB-S2	Digital Video Broadcasting - Satellite - Second Generation
DVB-T	Digital Video Broadcasting – Terrestrial
EBU	European Broadcasting Union

ECM	Entitlement Control Message
EEPROM	Electrically-Erasable Programmable Read-Only Memory
EMM	Entitlement Management Message
ES	Elementary Stream
ETSI	European Telecommunications Standards Institute
FSR	Feedback-Shift-Register
HD-DVD	High-Definition/Density Digital Versatile Disc
HRL	Host Revocation List
IDEA	Internacional Data Encryption Algorithm
IP	Internet Protocol
LSD	Layered Subset Difference
MKB	Media Key Block
MPEG	Moving Picture Experts Group
MPK	Master Private Key
PES	Packetized Elementary Stream
PRG	Pseudo Random Generator
RAM	Random Access Memory
ROM	Read Only Memory
RSA	Rivest Shamir Adleman
SAS	Subscriber Authorization System
SCS	Skipping Chain Scheme
SD	Subset Difference
SMS	Subscriber Management System
SS	Storage Space
ST	Steiner Tree
STB	Set-Top-Box
TDES	Triple- Data Encryption Standard
TO	Transmission Overhead
TS	Transport Stream
URL	Uniform Resource Locator

8 Referencias

[1] A. Fiat, M. Naor, Broadcast encryption, Advances in Cryptology, CRYPTO'93, LNCS 773, pages 480-491, Springer-Verlag, 1994.

[2] T. Hesselius, T. Savela, A Java Framework for Broadcast Encryption Algorithms, Master Thesis Report, LiTH-ISY-EX-3563-2004, 2004.

[3] N. Molavi, X. Zhao, *A Security Study of Digital TV Distribution Systems*, Master Thesis, Department of Computer and Systems Sciences Royal Institute of Technology, June 2005.

[4] G. Kreitz, *Optimization of broadcast encryption schemes*, Master's thesis, Royal Institute of Technology, 2005.

[5] M. Johansson, *Practical Evaluation of Revocation Schemes*, Master's thesis, Royal Institute of Technology, 2004.

[6] N. Attrapadung, H. Imai, Practical Broadcast Encryption from Graph-Theoretic Techniques and Subset-Incremental-Chain Structure, IEICE Trans. Fundamentals, vol.E90–A, pages 187-203, No.1 January 2007.

[7] Obied, *Broadcast Encryption*, April 2005, http://pages.cpsc.ucalgary.ca/~amaobied/papers/be.pdf.

[8] D. Boneh, A. Sahai, and B. Waters, *Fully collusion resistant traitor tracing with short ciphertexts and private keys*, In Eurocrypt '06, pages 573-592, Springer-Verlag, 2006.

[9] D. Halevy, A. Shamir, *The LSD broadcast encryption scheme*, Lecture Notes in Computer Science, vol. 2442, pages 47–60, 2002.

[10] Anderson, F. Claesson, J. Löfvenberg, *Effects of user adjacency in the subset difference scheme for broadcast encryption*, RadioVetenskap och Kommunikation, Linköping, 2005.

[11] E.C. Park, I.F. Blake, *On the mean number of encryptions for tree-based broadcast encryption schemes*. Journal of Discrete Algorithms, vol. 4, pages 215–238, 2006.

[12] Y. Dodis, N. Fazio, *Public key broadcast encryption for stateless receivers*, In Proceedings of the Digital Rights Management Workshop 2002, LNCS vol. 2696, pages 61–80, Springer, 2002.

[13] Adelsbach, U. Greveler, *A Broadcast Encryption Scheme with Free-Riders but Unconditional Security*, First International Conference on Digital Rights Management:

Technology, Issues, Challenges and Systems Sydney, Australia, 31 October - 2 November 2005.

[14] D. Boneh, B. Waters, *A fully collusion resistant broadcast, trace, and revoke system*, In Proceedings of ACM Computer and Communications Security (CCS), 2006.

[15] T. Asano, K Kamio, *A Lightweight Tree Based One-Key Broadcast Encryption Scheme*, IEICE Trans. Fundamentals, Vol.E89–A, No.7 July 2006.

[16] J. Horwitz, A Survey of Broadcast Encryption, Manuscript, 2003.

[17] How, J. Chee, J. Amundsen, *Authenticated Broadcast Encryption Scheme*, Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops - Vol. 01, pages 512-518, IEEE Computer Society 2007.

[18] J. Lotspiech, S. Nusser, F. Pestoni, *Broadcast Encryption's Bright Future*, IEEE Computer vol. 35, pages 57-63, 2002.

[19] M. J. Mihaljevic, *Broadcast encryption schemes based on the sectioned key tree*, Lecture Notes in Computer Science, vol. 2836, pages 158–169, 2003.

[20] M. Ramkumar, Broadcast Encryption Using Probabilistic Key Distribution and Applications. Journal of Computers, vol. 1, No. 3, June 2006.

[21] Boneh, C. Gentry, B. Waters, *Collusion resistant broadcast encryption with short ciphertexts and private keys*, CRYPTO 2005, LNCS vol. 3621, pages 258–275, 2005.

[22] Z. Ramzan, D. P. Woodruff, *Fast Algorithms for the Free Riders Problem in Broadcast Encryption*, IACR ePrint Archive, http://eprint.iacr.org, 2006.

[23] J.Y. Hwang, D.H. Lee, J. Lim, *Generic transformation for scalable broadcast encryption schemes*, CRYPTO 2005, LNCS vol. 3621, pages 276–292, 2005.

[24] M. Abdalla, Y. Shavitt, A. Wool, *Key management for restricted multicast using broadcast encryption*, IEEE/ACM Trans. Networking, vol. 8, pages 443–454, Aug. 2000.

[25] N. S. Jho, E. S. Yoo, J. H. Cheon, M. H. Kim, *New Broadcast Encryption Scheme Using Tree-Based Circle*. Digital Rights Management Workshop 2005, pages 37-44, 2005.

[26] Barth, D. Boneh, B. Waters, *Privacy in encrypted content distribution using private broadcast encryption*, In Financial Cryptography '06, Springer, LNCS, 2006.

[27] Naor, M. Naor, J. Lotspiech, *Revocation and Tracing Schemes for Stateless Receivers*, Advances in Cryptology - CRYPTO '01, vol. 2139 of Lecture Notes in Computer Science, pages 41-62, Springer Verlag, 2001.

[28] P.Wang, P. Ning, D.S. Reeves, *Storage-efficient stateless group key revocation*, ISC 2004, LNCS vol. 3225, pages 25–38, 2004.

[29] H. Hwang, P. J. Lee. *Efficient Broadcast Encryption Scheme with Log-Key Storage*, Financial Cryptography and Data Security - FC 2006, LNCS vol. 4107, pages 281-295, 2006 .

[30] J. H. Cheon, N. S. Jho, M. H. Kim, E. S. Yoo, *Skipping, Cascade, and Combined Chain Schemes for Broadcast Encryption*, IEEE Transactions on Information Theory, Vol. 54, pages 5155-5171, Nov. 2008.

[31] Huang , S. Shieh, F. Ho, J. Wang, *Efficient Key Distribution Schemes for Secure Media Delivery in Pay-TV Systems*, IEEE Transactions on Multimedia, Vol. 6, Issue 5, pages 760 - 76, Oct. 2004.

[32] Kirkels, M. Maas, P. Roelse, *A Security Architecture for Pay-Per-View Business Models in Conditional Access Systems*, DRM '07: Proceedings of the 2007 ACM workshop on Digital Rights Management, pages 1-9, 2007.

[33] Q. Xie, S. B. Zheng, X. J. Yu, A smartcard conditional access interface scheme for conditional access subsystem separation in digital TV broadcasting, Journal of Zhejiang University SCIENCE A, pages 1008-1017, 2006.

[34] Liu, W. Zhang, T. Jiang, *A Scalable Key Distribution Scheme for Conditional Access System in Digital Pay-TV System*, IEEE Transactions on Consumer Electronics, vol. 50, No. 2 , pages 632–637, 2004.

[35] M. Zhu, M. Zhang, X. Chen, D. Zhang, Z. Huang. *A Hierarchical Key Distribution Scheme for Conditional Access System in DTV Broadcasting*, 2006 International Conference on Computational Intelligence and Security, vol. 2, pages 1532 - 1535, 2006.

[36] EBU Project Group B/CA, *Functional model of a conditional access system*, EBU Technical Review, pages 64-77, Winter 1995.

[37] Snamba, *Scrambling (Conditional Access System)*, Broadcast Technology No.12, Autumn 2002.

[38] M. Zheng, S. B. Zheng, *A common smart-card-based conditional access system for digital set-top box*, IEEE Trans. Cons. Elec., vol. 50, No. 2, pages 601-605, 2004.

[39] H. S. Koo, I. K. Lee, J. M. Kim, S. W. Ra, *A Novel Rekey Management Scheme in Digital Broadcasting Network*, APNOMS 2006, LNCS vol. 4238, pages 283 – 292, 2006.

[40] Q. Xie, S. B. Zheng, X. J. Yu, *A smart-card-based conditional access subsystem separation scheme for digital TV broadcasting*, IEEE Trans. Cons. Elec., vol. 51, No. 3, pages 925-932, 2005.

[41] Loebbecke, M. Fischer, *Business opportunities and risks from pay-TV piracy: The case of Europe*, in Proc. 11th Americas Conf. Information Systems, Omaha, NE, , pages 462–471, 2005.

[42] L. Francis, W. G. Sirett, K. Mayes, K. Markantonakis, C*ountermeasures for Attacks on Satellite TV Cards using Open Receivers*, Conferences in Research and Practice in Information Technology, vol. 44, 2005.

[43] Data Encryption Standard – Cipher Block Chaining Packet Encryption Specification, Engineering Committee, Digital Video Subcommittee, SCTE 52 2008.

[44] Digital Video Broadcasting (DVB); DVB SimulCrypt; Head-end architecture and synchronization. ETSI TS 101 197 V1.2.1 (2002-02).

[45] S. Y. Wang, C. S. Laih, *Efficient Key Distribution for Access Control in Pay-TV Systems*, IEEE Transactions On Multimedia, vol. 10, No. 3, pages 480-492, April 2008.

[46] K. Wirt, *Fault attack on the DVB Common Scrambling Algorithm*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 3481, pages 577-584, 2005.

[47] M. Tunstall, K. Markantonakis, K. Mayes, *Inhibiting Card Sharing Attacks*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 4266, pages 239-251, 2006.

[48] K. Tu, C. S. Laih, S. H. Toung, *On key distribution management for condition access system on Pay-TV system*, IEEE Int. Symp. Consumer Electronics (ISCE'98) , vol. 45, pages 151–159, 1998.

[49] T. Jiang, S. Zeng, B. Lin, *Key distribution based on hierarchical access control for conditional access system in DTV broadcast*, IEEE Trans. on Consumer Electronics, vol. 50, No. 1, pages 225-230, Feb. 2004.

[50] Loebbecke, M. Fischer, *Pay TV Piracy and Its Effects on Pay TV Provision*, Journal of Media Business Studies, vol. 2, pages 17-34, 2005.

[51] W. Kanjanarin, T. Amornraksa, *Scrambling and key distribution scheme for digital television*, In Proc. 9th IEEE International Conference on Networks, pages 140–145, 2001.

[52] *Study on the impact of the conditional access directive*, Study prepared on behalf of the European Commission Directorate General for Internal Market & Services, December 2007.

[53] StuntGuy, *The NagraVision hacking FAQ*, Revision: 12282000. http://www.mtl2600.org/bev/ESFAQ.TXT

[54] J. Anstensen, An investigation of the possibility of defining a new conditional access application programming interface for digital television receivers, Master's Thesis in Linköpings tekniska högskola, on 17th June, 2002.

[55] R. P. Weinmann, K. Wirt, *Analysis of the dvb common scrambling algorithm*, In Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, CMS 2004, Proceedings, Kluwer Academic Publishers, 2004.

[56] H. Kim, *Secure Communication in Digital TV Broadcasting*, IJCSNS International Journal of Computer Science and Network Security, vol. 8, No. 9, September 2008.

[57] K. Henry, J. Sui, G. Zhong, An Overview of the Advanced Access Content System (AACS), Technical Report, 2007.

[58] Advanced Access Content System: Technical Overview (Informative), July 21, 2004.

[59] Advanced Access Content System: Introduction and Common Cryptographic Elements, Revision 0.91 February 17, 2006, http://www.aacsla.com/specifications/specs091/AACS_Spec_Common_0.91.pdf

[60] Advanced Access Content System: Pre-recorded Video Book, Revision 0.92 November 29, 2007. http://www.aacsla.com/specifications/specs091/AACS_Spec_Prerecorded_0.91.pdf

[61] Advanced Access Content System: Blu-ray Disc Pre-recorded Book, Revision 0.92 December 5, 2007. http://www.aacsla.com/specifications/specs091/AACS_Spec_BD_Prerecorded_0.91.pdf

[62] Advanced Access Content System: HD DVD and DVD Pre-recorded Book, Revision 0.912 August 15, 2006. http://www.aacsla.com/specifications/specs091/AACS_Spec_HD_DVD_and_DVD_Prerecorded_0.91.pdf

[63] Content Protection for Optical Media. A Comparison of Self-Protecting Digital Content and AACS, Independent Security Evaluators May 3, 2005.

[64] J. Sui, A Security Analysis of Some Physical Content Distribution Systems, Master Thesis Waterloo, Ontario, Canada, 2008.

[65] Kesden, *Content Scrambling System (CSS)*, Carnegie Mellon University, 2000. http://www.cs.cmu.edu/~dst/DeCSS/Kesden/index.html

[66] Muslix64, BackupHDDVD, A Tool to Decrypt AACS Protected Movies, Doom9's Forum. Acceso Julio 2008. http://forum.doom9.org/showthread.php?t=119871

[67] Vampire, *WinDVD 8 Device Key Found!*, Doom9's Forum. Acceso Julio 2008. http://forum.doom9.org/showthread.php?t=122664

[68] arnezami, Processing Key, Media Key and Volume ID found!!!, Doom9's Forum. Acceso Julio 2008. http://forum.doom9.org/showthread.php?t=121866&page=6

[69] arnezami, *Understanding AACS (including Subset-Difference)*, Doom9's Forum. Acceso Julio 2008. http://forum.doom9.org/showthread.php?t=122363

[70] jareno2000, *Proceso de decriptación EMM*, Foro Zacky Files. Acceso Junio 2008. http://foros.zackyfiles.com/showthread.php?page=8&t=475820

[71] calpe19, *Proceso de decriptación EMM*, Foro Zacky Files. Acceso Junio 2008. http://foros.zackyfiles.com/showthread.php?page=8&t=475820

[72] jareno2000, El uso de Fuerza Bruta para romper sistemas de cifrado, RDI-SAT-Forums. Acceso Junio 2008. http://forum.d--n--s.com/showthread.php?t=40527&page=4

[73] jareno2000, Fundamentos de RSA y conocimientos de los EMM y los ECM, RDI-SAT-Forums. Acceso Junio 2008. http://www.rdi-board.com/showpost.php?p=353490&postcount=23

[74] Morello, V. Mignone, *DVB-S2— ready for lift off*, EBU Technical Review, October 2004.

[75] J. Lucena, *Criptografía y Seguridad en Computadores*, Cuarta Edición. Versión 0.7.51, 11 de junio de 2008. http://wwwdi.ujaen.es/~mlucena/bin/cysec4.zip

[76] http://www.dvb.org/

[77] http://www.aacsla.com/home

[78] http://www.jquirke.com.au/index.php/2007/02/12/aacss-subset-cover-scheme-explained/

[79] J. Pastor, M. A. Sarasa, J. L. Salazar, *Criptografía Digital: Fundamentos y Aicaciones*, Prensas Univers. Zaragoza, Rustica, 1998.

9 Anexos

9.1 Algoritmos de cifrado

9.1.1 DES (Data Encryption Standard)

El DES (Data Encryption Standard) es un algoritmo de cifrado, cuyo uso se ha propagado ampliamente por todo el mundo. Se basa en el algoritmo LUCIFER, que había sido desarrollado por IBM a principios de los setenta, y fue adoptado como estándar por el Gobierno de los EE.UU. para comunicaciones no clasificadas en 1976. En realidad la NSA lo diseñó para ser implementado por hardware, creyendo que los detalles iban a ser mantenidos en secreto, pero la Oficina Nacional de Estandarización publicó su especificación con suficiente detalle como para que cualquiera pudiera implementarlo por software.

Hoy en día, DES se considera inseguro para muchas aplicaciones. Esto se debe principalmente a que el tamaño de clave de 56 bits es corto; las claves de DES se han roto en menos de 24 horas. Existen también resultados analíticos que demuestran debilidades teóricas en su cifrado, aunque son inviables en la práctica. Se cree que el algoritmo es seguro en la práctica en su variante de Triple DES, aunque existan ataques teóricos.

El DES es el algoritmo prototipo del cifrado por bloques, toma un texto en claro de una longitud fija de bits y lo transforma mediante una serie de complicadas operaciones en otro texto cifrado de la misma longitud. En el caso de DES el tamaño del bloque es de 64 bits. DES utiliza también una clave criptográfica para modificar la transformación, de modo que el descifrado sólo puede ser realizado por aquellos que conozcan la clave concreta utilizada en el cifrado. La clave mide 64 bits, aunque en realidad, sólo 56 de ellos son empleados por el algoritmo. Los ocho bits restantes se utilizan únicamente para comprobar la paridad, y después son descartados. Por tanto, la longitud de clave efectiva en DES es de 56 bits, y así es como se suele especificar.

Al igual que otros cifrados de bloque, DES debe ser utilizado en el modo de operación de cifrado de bloque si se aplica a un mensaje mayor de 64 bits.

El DES es una Red de Feistel de 16 rondas, más dos permutaciones, una que se aplica al principio (P_i) y otra que se aplica al final (P_f), tales que $P_i = P_f^{-1}$.

La función f (*Figura 45*) se compone de una permutación de expansión (E), que convierte el bloque de 32 bits correspondiente en uno de 48. Después realiza un OR-exclusivo

con el valor K_i, también de 48 bits, aplica ocho S-Cajas de 6x4 bits, y efectúa una nueva permutación *P*.

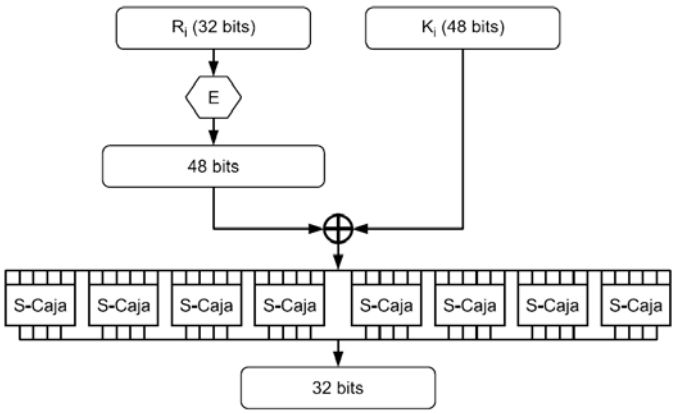

Figura 45: Esquema de la función f del algoritmo DES

Se calcula un total de 16 valores de K_i (*Figura 46*), uno para cada ronda, efectuando primero una permutación inicial *EP1* sobre la clave de 64 bits, llevando a cabo desplazamientos a la izquierda de cada una de las dos mitades, de 28 bits resultantes, y realizando finalmente una elección permutada (*EP2*) de 48 bits en cada ronda, que será la K_i. Los desplazamientos a la izquierda son de dos bits, salvo para las rondas 1, 2, 9 y 16, en las que se desplaza sólo un bit. Nótese que aunque la clave para el algoritmo DES tiene en principio 64 bits, se ignoran ocho de ellos, un bit de paridad por cada byte de la clave, por lo que en la práctica se usan sólo 56 bits.

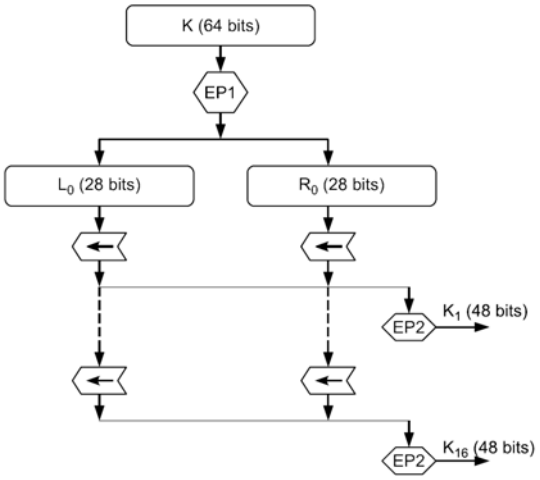

Figura 46: Calculo de las K_i para el algoritmo DES

Para descifrar basta con usar el mismo algoritmo (ya que $P_i = P_f^{-1}$) empleando las K_i en orden inverso.

Desde hace algunos años, el algoritmo ha sido sustituido por el nuevo AES (Advanced Encryption Standard).

9.1.2 Triple DES

El Triple DES es el algoritmo que hace triple cifrado del DES. También es conocido como TDES o 3DES y fue desarrollado por IBM en 1978.

No llega a ser un cifrado múltiple, porque no son independientes todas las subclases. Este hecho se basa en que DES tiene la característica matemática de no ser un grupo, lo que implica que si se cifra el mismo bloque dos veces con dos claves diferentes se aumenta el tamaño efectivo de la clave.

La variante más simple del Triple DES funciona de la siguiente manera:

$$C = E_{DES}^{k_3}\left(D_{DES}^{k_2}\left(E_{DES}^{k_1}(M)\right)\right)$$

donde M es el mensaje a cifrar y k_1, k_2 y k_3 las respectivas claves DES.

Cuando se descubrió que una clave de 56 bits no era suficiente para evitar un ataque de fuerza bruta, TDES fue elegido como forma de agrandar el largo de la clave sin necesidad de cambiar de algoritmo de cifrado. Este método de cifrado es inmune al ataque por encuentro a medio camino, doblando la longitud efectiva de la clave (112 bits), pero en cambio es preciso triplicar el número de operaciones de cifrado, haciendo este método de cifrado muchísimo más seguro que el DES. Por tanto, la longitud de la clave usada será de 192 bits, aunque como se ha dicho su eficacia solo sea de 112 bits.

El Triple DES está desapareciendo lentamente, siendo reemplazado por el algoritmo AES. Sin embargo, la mayoría de las tarjeta de crédito y otros medios de pago electrónico tienen como estándar el algoritmo Triple DES (anteriormente usaban el DES). Por el diseño DES y por lo tanto TDES son algoritmos lentos. AES puede llegar a ser hasta 6 veces más rápido y hasta el día de la fecha no se encontró ninguna vulnerabilidad.

9.1.3 IDEA (Internacional Data Encryption Algorithm)

El algoritmo IDEA (Internacional Data Encryption Algorithm) es un cifrador por bloques que fue descrito por primera vez en 1992. Fue un algoritmo propuesto como reemplazo del DES (Data Encryption Standard). Trabaja con bloques de 64 bits de longitud y emplea una clave de 128 bits. Como en el caso de DES, se usa el mismo algoritmo tanto para cifrar como para descifrar.

Como ocurre con todos los algoritmos simétricos de cifrado por bloques, IDEA se basa en los conceptos de confusión y difusión, haciendo uso de las siguientes operaciones elementales (todas ellas fáciles de implementar):

- XOR
- Suma módulo 216
- Producto módulo 216 + 1

El algoritmo IDEA consta de ocho rondas. Dividiremos el bloque X a codificar, de 64 bits, en cuatro partes X_1, X_2, X_3 y X_4 de 16 bits. Para la interpretación entera de dichos registros se empleará el criterio big endian, lo cual significa que el primer byte es el más significativo. Denominaremos Z_i a cada una de las 52 subclaves de 16 bits que vamos a necesitar. Las operaciones que llevaremos a cabo en cada ronda son las siguientes:

1. Multiplicar X_1 por Z_1
2. Sumar X_2 con Z_2
3. Sumar X_3 con Z_3
4. Multiplicar X_4 por Z_4
5. Hacer un XOR entre los resultados del paso 1 y el paso 3
6. Hacer un XOR entre los resultados del paso 2 y el paso 4
7. Multiplicar el resultado del paso 5 por Z_5.
8. Sumar los resultados de los pasos 6 y 7
9. Multiplicar el resultado del paso 8 por Z_6
10. Sumar los resultados de los pasos 7 y 9
11. Hacer un XOR entre los resultados de los pasos 1 y 9
12. Hacer un XOR entre los resultados de los pasos 3 y 9
13. Hacer un XOR entre los resultados de los pasos 2 y 10
14. Hacer un XOR entre los resultados de los pasos 4 y 10

La salida de cada iteración serán los cuatro sub-bloques obtenidos en los pasos 11, 12, 13 y 14, que serán la entrada del siguiente ciclo, en el que emplearemos las siguientes seis subclaves, hasta un total de 48. Al final de todo intercambiaremos los dos bloques centrales (en realidad con eso deshacemos el intercambio que llevamos a cabo en los pasos 12 y 13).

Después de la octava iteración, se realiza la siguiente transformación:

1. Multiplicar X_1 por Z_{49}
2. Sumar X_2 con Z_{50}
3. Sumar X_3 con Z_{51}
4. Multiplicar X_4 por Z_{52}.

Las primeras ocho subclaves se calculan dividiendo la clave de entrada en bloques de 16 bits. Las siguientes ocho se calculan rotando la clave de entrada 25 bits a la izquierda y volviendo a dividirla, y así sucesivamente.

Las subclaves necesarias para descifrar se obtienen cambiando de orden las Z_i y calculando sus inversas para la suma o la multiplicación, según la *Figura 47*. Puesto que $2^{16}+1$ es un número primo, nunca podremos obtener cero como producto de dos números, por lo que no necesitamos representar dicho valor. Cuando estemos calculando productos, utilizaremos el cero para expresar el número 2^{16}, un uno seguido de 16 ceros. Esta representación es coherente puesto que los registros que se emplean internamente en el algoritmo poseen únicamente 16 bits.

Ronda	Subclaves de Cifrado						Subclaves de Descifrado					
1	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_{49}^{-1}	$-Z_{50}$	$-Z_{51}$	Z_{52}^{-1}	Z_{47}	Z_{48}
2	Z_7	Z_8	Z_9	Z_{10}	Z_{11}	Z_{12}	Z_{43}^{-1}	$-Z_{45}$	$-Z_{44}$	Z_{46}^{-1}	Z_{41}	Z_{42}
3	Z_{13}	Z_{14}	Z_{15}	Z_{16}	Z_{17}	Z_{18}	Z_{37}^{-1}	$-Z_{39}$	$-Z_{38}$	Z_{40}^{-1}	Z_{35}	Z_{36}
4	Z_{19}	Z_{20}	Z_{21}	Z_{22}	Z_{23}	Z_{24}	Z_{31}^{-1}	$-Z_{33}$	$-Z_{32}$	Z_{34}^{-1}	Z_{29}	Z_{30}
5	Z_{25}	Z_{26}	Z_{27}	Z_{28}	Z_{29}	Z_{30}	Z_{25}^{-1}	$-Z_{27}$	$-Z_{26}$	Z_{28}^{-1}	Z_{23}	Z_{24}
6	Z_{31}	Z_{32}	Z_{33}	Z_{34}	Z_{35}	Z_{36}	Z_{19}^{-1}	$-Z_{21}$	$-Z_{20}$	Z_{22}^{-1}	Z_{17}	Z_{18}
7	Z_{37}	Z_{38}	Z_{39}	Z_{40}	Z_{41}	Z_{42}	Z_{13}^{-1}	$-Z_{15}$	$-Z_{14}$	Z_{16}^{-1}	Z_{11}	Z_{12}
8	Z_{43}	Z_{44}	Z_{45}	Z_{46}	Z_{47}	Z_{48}	Z_7^{-1}	$-Z_9$	$-Z_8$	Z_{10}^{-1}	Z_5	Z_6
Final	Z_{49}	Z_{50}	Z_{51}	Z_{52}			Z_1^{-1}	$-Z_2$	$-Z_3$	Z_4^{-1}		

Figura 47: Subclaves empleadas en el algoritmo IDEA

9.1.4 AES (Advanced Encryption Standard)

El AES (Advanced Encryption Standard), también conocido como Rijndael, es un esquema de cifrado por bloques adoptado como un estándar de cifrado por el gobierno de los Estados Unidos. Se espera que sea usado en el mundo entero y analizado exhaustivamente, como fue el caso de su predecesor, el Data Encryption Standard (DES). El AES se transformó en un estándar efectivo el 26 de mayo de 2002. Desde 2006, el AES es uno de los algoritmos más populares usados en criptografía simétrica.

Al contrario que su predecesor (DES), Rijndael es una red de sustitución-permutación, no una red de Feistel. AES es rápido tanto en software como en hardware, es relativamente fácil de implementar, y requiere poca memoria. Como nuevo estándar de cifrado, se está utilizando actualmente a gran escala.

En el lugar de la red de Feistel se ha definido cada ronda como una composición de cuatro funciones invertibles diferentes, formando tres capas, diseñadas para proporcionar resistencia frente a criptoanálisis lineal y diferencial. Cada una de las funciones tiene un propósito preciso:

- La capa de mezcla lineal, funciones DesplazarFila y MezclarColumnas, permite obtener un alto nivel de difusión a lo largo de varias rondas.
- La capa no lineal, función ByteSub, consiste en la aplicación paralela de s-cajas con propiedades óptimas de no linealidad.
- La capa de adición de clave es un simple or-exclusivo entre el estado intermedio y la subclave correspondiente a cada ronda.

Elementos del AES

AES es un sistema de cifrado por bloques, diseñado para manejar longitudes de clave y de bloque variables, ambas comprendidas entre los 128 y los 256 bits. Realiza varias de sus operaciones internas a nivel de byte, interpretando éstos como elementos de un cuerpo de Galois $GF(2^8)$. El resto de operaciones se efectúan en términos de registros de 32 bits. Sin embargo, en algunos casos, una secuencia de 32 bits se toma como un polinomio de grado inferior a 4, cuyos coeficientes son a su vez polinomios en $GF(2^8)$.

Si bien, como ya se ha dicho, este algoritmo soporta diferentes tamaños de bloque y de clave, en el estándar adoptado por el Gobierno Estadounidense en noviembre de 2001, se

especifica una longitud fija de bloque de 128 bits ($N_b = 4$, como se verá más adelante), y la longitud de clave a escoger entre 128, 192 y 256 bits.

AES es un algoritmo que se basa en aplicar un número determinado de rondas a un valor intermedio que se denomina *estado*. Dicho estado puede representarse mediante una matriz rectangular de bytes, que posee cuatro filas, y N_b columnas. Así, por ejemplo, si nuestro bloque tiene 160 bits (*Tabla 4*), N_b será igual a 5.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$
a1,0	a1,1	a1,2	a1,3	a1,4
a2,0	a2,1	a2,2	a2,3	a2,4
a3,0	a3,1	a32	a3,3	a3,4

Tabla 4: Ejemplo de matriz de estado con Nb=5 (160 bits).

k0,0	k0,1	k0,2	k0,3
k1,0	k1,1	k1,2	k1,3
k2,0	k2,1	k2,2	k2,3
k3,0	k3,1	k32	k3,3

Tabla 5: Ejemplo de clave con Nk=4 (128 bits).

La clave tiene una estructura análoga a la del *estado*, y se representará mediante una tabla con cuatro filas y N_k columnas. Si nuestra clave tiene, por ejemplo, 128 bits, N_k será igual a 4 (*Tabla 5*).

En algunos casos, tanto el estado como la clave se consideran como vectores de registros de 32 bits, estando cada registro constituido por los bytes de la columna correspondiente, ordenados de arriba a abajo.

El bloque que se pretende cifrar o descifrar se traslada directamente byte a byte sobre la matriz de estado, siguiendo la secuencia $a_{0,0}$, $a_{1,0}$, $a_{2,0}$, $a_{3,0}$, $a_{0,1}$ …, y análogamente, los bytes de la clave se copian sobre la matriz de clave en el mismo orden, a saber, $k_{0,0}$, $k_{1,0}$, $k_{2,0}$, $k_{3,0}$, $k_{0,1}$...

Siendo *B* el bloque que queremos cifrar, y *S* la matriz de estado, el algoritmo AES con *n* rondas queda como sigue:

- Calcular K_0, K_1, . . . K_n subclaves a partir de la clave K.
- $S \leftarrow B \oplus K_0$

- Para i = 1 hasta n hacer
    - Aplicar ronda i–ésima del algoritmo con la subclave K_i.

Puesto que cada ronda es una sucesión de funciones invertibles, el algoritmo de descifrado consistirá en aplicar las inversas de cada una de las funciones en el orden contrario, y utilizar los mismos K_i que en el cifrado, sólo que comenzando por el último.

	Nb = 4 (128 bits)	Nb = 6 (192 bits)	Nb = 8 (256 bits)
Nb = 4 (128 bits)	10	12	14
Nb = 6 (192 bits)	12	12	14
Nb = 8 (256 bits)	14	14	14

Tabla 6: Número de rondas para AES en función de los tamaños de clave y bloque

Rondas AES

Puesto que AES permite emplear diferentes longitudes tanto de bloque como de clave, el número de rondas requerido en cada caso es variable. En la *Tabla 6* se especifica cuántas rondas son necesarias en función de N_b y N_k.

Siendo S la matriz de estado, y K_i la subclave correspondiente a la ronda i–ésima, cada una de las rondas posee la siguiente estructura:

- S ← ByteSub(S)
- S ← DesplazarFila(S)
- S ← MezclarColumnas(S)
- $S \leftarrow K_i \oplus S$

La última ronda es igual a las anteriores, pero eliminando el paso 3.

Función *ByteSub*

La transformación *ByteSub* es una sustitución no lineal que se aplica a cada byte de la matriz de estado, mediante una s-caja 8x8 invertible, que se obtiene componiendo dos transformaciones:

Cada byte es considerado como un elemento del $GF(2^8)$ que genera el polinomio irreducible $m(x) = x^8 + x^4 + x^3 + x + 1$, y sustituido por su inversa multiplicativa. El valor cero queda inalterado.

El siguiente paso consiste en aplicar la siguiente transformación afín en *GF*(2), siendo $x_0, x_1, \ldots, x_7$ los bits del byte correspondiente, e $y_0, y_1, \ldots, y_7$ los del resultado:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

La función inversa de *ByteSub* sería la aplicación de la inversa de la s-caja correspondiente a cada byte de la matriz de estado.

Función DesplazarFila

Esta transformación consiste en desplazar a la izquierda cíclicamente las filas de la matriz de estado. Cada fila f_i se desplaza un número de posiciones c_i diferente. Mientras que c_0 siempre es igual a cero (esta fila siempre permanece inalterada), el resto de valores viene en función de N_b.

Nb	c1	c2	c3
4	1	2	3
6	1	2	3
8	1	3	4

Tabla 7: Valores de ci según el tamaño de bloque de Nb

La función inversa de *DesplazarFila* será, obviamente, un desplazamiento de las filas de la matriz de estado el mismo número de posiciones que en el cuadro anterior, pero a la derecha.

Función MezclarColumnas

Para aplicar esta función cada columna del vector de estado se considera un polinomio cuyos coeficientes pertenecen a *GF*(2^8), es decir, son también polinomios y se multiplica módulo $x^4 + 1$ por:

$$c(x) = 03x^3 + 01x^2 + 01x + 02$$

donde 03 es el valor hexadecimal que se obtiene concatenando los coeficientes binarios del polinomio correspondiente en $GF(2^8)$, en este caso 00000011, o sea, $x + 1$, y así sucesivamente.

La inversa de *MezclarColumnas* se obtiene multiplicando cada columna de la matriz de estado por el polinomio:

$$d(x) = 0Bx^3 + 0Dx^2 + 09x + 0E$$

9.1.5 RSA (Rivest Shamir Adleman)

El sistema criptográfico de clave pública RSA es un algoritmo asimétrico de cifrado de bloques, que utiliza una clave pública, la cual se distribuye (en forma autenticada preferentemente), y otra privada, la cual es guardada en secreto por su propietario. Debe su nombre a sus tres inventores: Ronald Rivest, Adi Shamir y Leonard Adleman, y estuvo bajo patente de los Laboratorios RSA hasta el 20 de septiembre de 2000, por lo que su uso comercial estuvo restringido hasta esa fecha.

RSA se basa en la dificultad para factorizar grandes números. Las claves pública y privada se calculan a partir de un número que se obtiene como producto de dos primos grandes. El atacante se enfrentará, si quiere recuperar un texto claro a partir del criptograma y la clave pública, a un problema de factorización o tendrá que resolver un logaritmo discreto.

Para generar un par de claves (K_P , K_p), en primer lugar se eligen aleatoriamente dos números primos grandes, p y q. Después se calcula el producto $n = pq$.

Escogeremos ahora un número e primo relativo con $(p-1)(q-1)$. (e, n) será la clave pública. Nótese que e debe tener inversa módulo $(p-1)(q-1)$, por lo que existirá un número d tal que

$$de \equiv 1 \pmod{(p-1)(q-1)}$$

es decir, que d es la inversa de e módulo $(p-1)(q-1)$. (d, n) será la clave privada. Esta inversa puede calcularse fácilmente empleando el Algoritmo Extendido de Euclides.

Nótese que si desconocemos los factores de n, este cálculo resulta prácticamente imposible.

La operación de *cifrado* se lleva a cabo según la expresión:

$$c = m^e \pmod{n}$$

mientras que el *descifrado* se hará de la siguiente forma:

$$m = c^d \pmod{n}$$

ya que

$$c^d = \left(m^e\right)^d = m^{ed} = m^{k(p-1)(q-1)+1} = \left(m^k\right)^{(p-1)(q-1)} m = m$$

Actualmente se considera segura una clave RSA con una longitud de n de al menos 1024 bits, si bien se recomienda el uso de claves no inferiores a 2048 bits. Hasta hace relativamente poco se recomendaban 512 bits, pero en mayo de 1999, Adi Shamir presentó el denominado dispositivo *Twinkle*, un ingenio capaz de factorizar números de manera muy rápida, aprovechando los últimos avances en la optimización de algoritmos específicos para esta tarea. Este dispositivo, aún no construido, podría ser incorporado en ordenadores de bajo coste y pondría en serio peligro los mensajes cifrados con claves de 512 bits o menos.

Teniendo en cuenta los avances de la tecnología, y suponiendo que el algoritmo RSA no sea roto analíticamente, deberemos escoger la longitud de la clave en función del tiempo que queramos que nuestra información permanezca en secreto. Efectivamente, una clave de 1024 bits parece a todas luces demasiado corta como para proteger información por más de unos pocos años.

9.2 Modos de Operación para Algoritmos de Cifrado por Bloques

9.2.1 Modo ECB

El modo ECB (*Electronic Codebook*) es el método más sencillo y directo de aplicar un algoritmo de cifrado por bloques. Simplemente se subdivide la cadena que se quiere codificar en bloques del tamaño adecuado y se cifran todos ellos empleando la misma clave.

9.2.2 Modo CBC

El modo CBC (*Cipher Book Chaining Mode*) incorpora un mecanismo de retroalimentación en el cifrado por bloques. Esto significa que el cifrado de bloques anteriores condiciona la codificación del actual, por lo que será imposible sustituir un bloque individual en el mensaje cifrado, ya que esta sustitución afectará a dos bloques en el mensaje descifrado resultante. Esto se consigue efectuando una operación *XOR* entre el bloque del mensaje que queremos codificar y el último criptograma obtenido. Para cifrar el primer bloque, se emplea el denominado vector de inicalización (V.I.), que deberá ser conocido por ambos interlocutores.

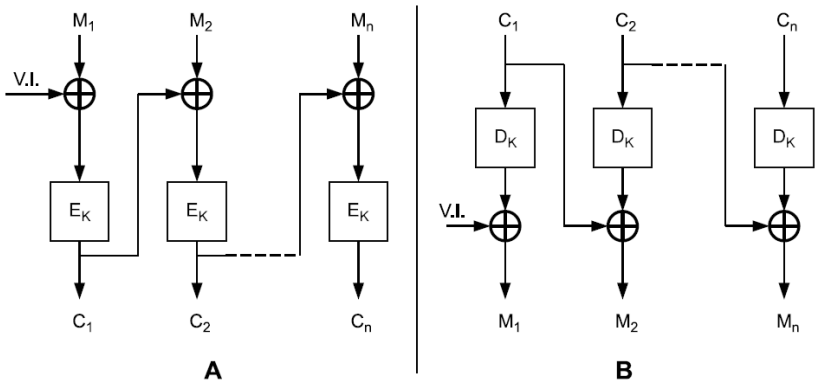

Figura 48: Modo CBC (A: Cifrado, B: Descifrado)

9.2.3 Modo CFB

El modo de operación CFB (*Cipher-Feedback*), cada bloque es cifrado y luego combinado, mediante la operación *XOR*, con el siguiente bloque del mensaje original. Al igual que en el modo CBC, se emplea un vector de inicialización a la hora de codificar el primer bloque. Con este método, si se produce un error en la transmisión, ésta se vuelve a sincronizar de forma automática, a partir del segundo bloque consecutivo que llegue de forma correcta.

9.3 Simulación

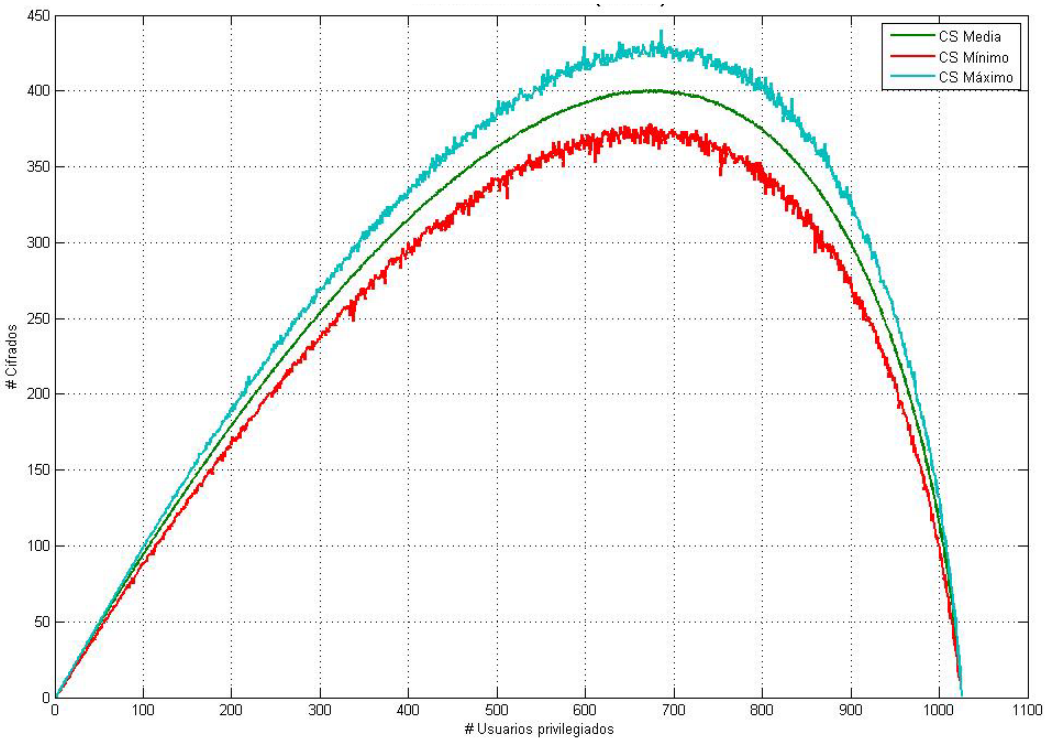

Figura 49: Complete Subtree TO (*n*=1024)

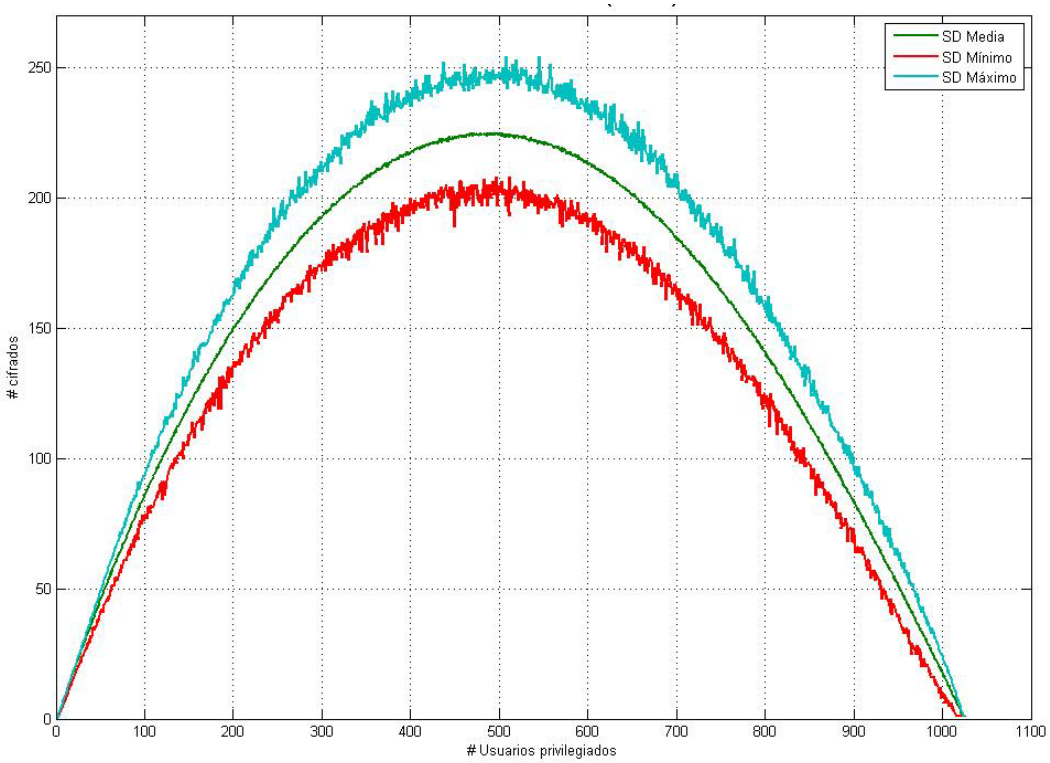

Figura 50: Subset Difference TO (*n*=1024)

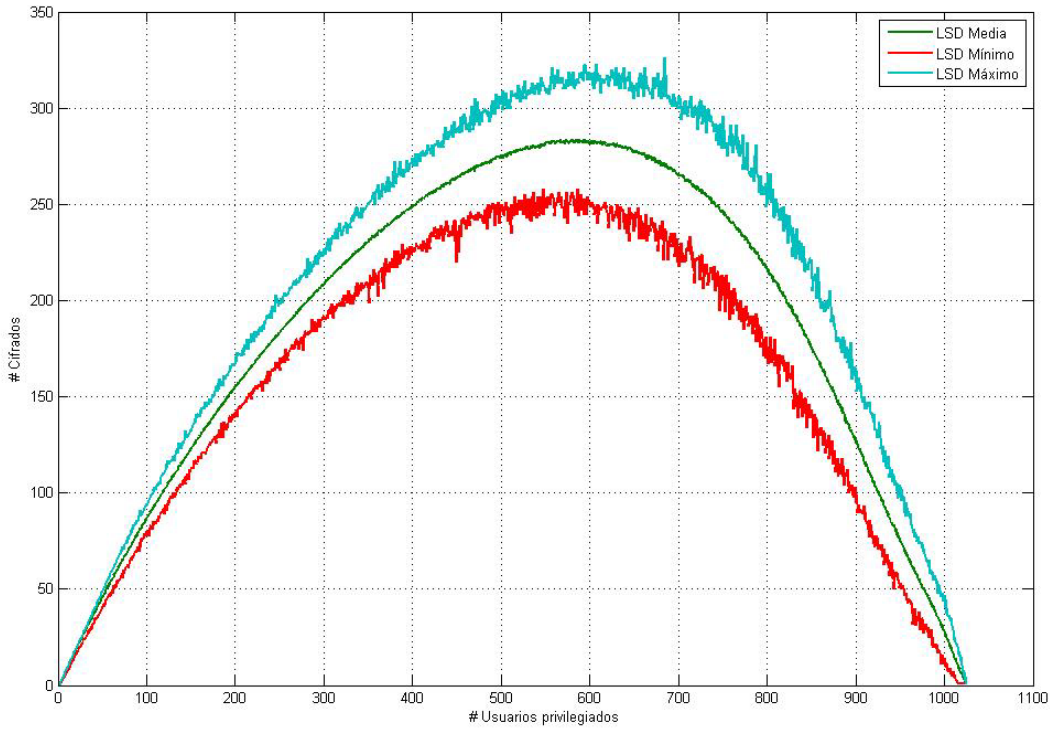

Figura 51: Layered Subset Difference TO (*n*=1024)

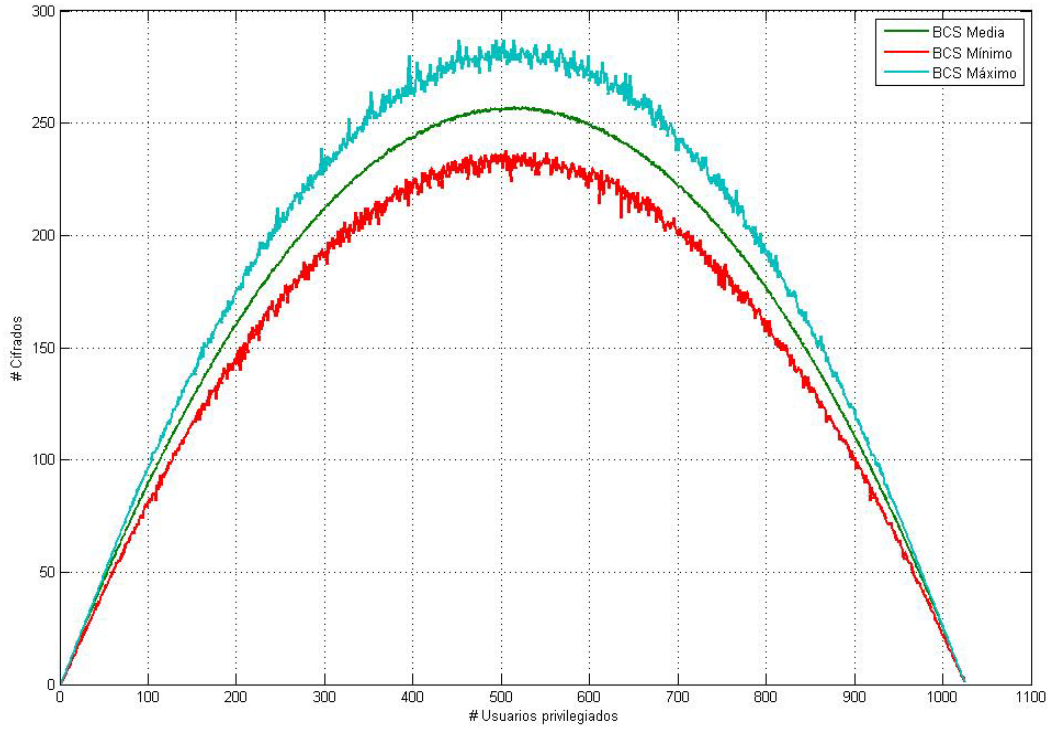

Figura 52: Basic Chain Scheme TO (*n*=1024)

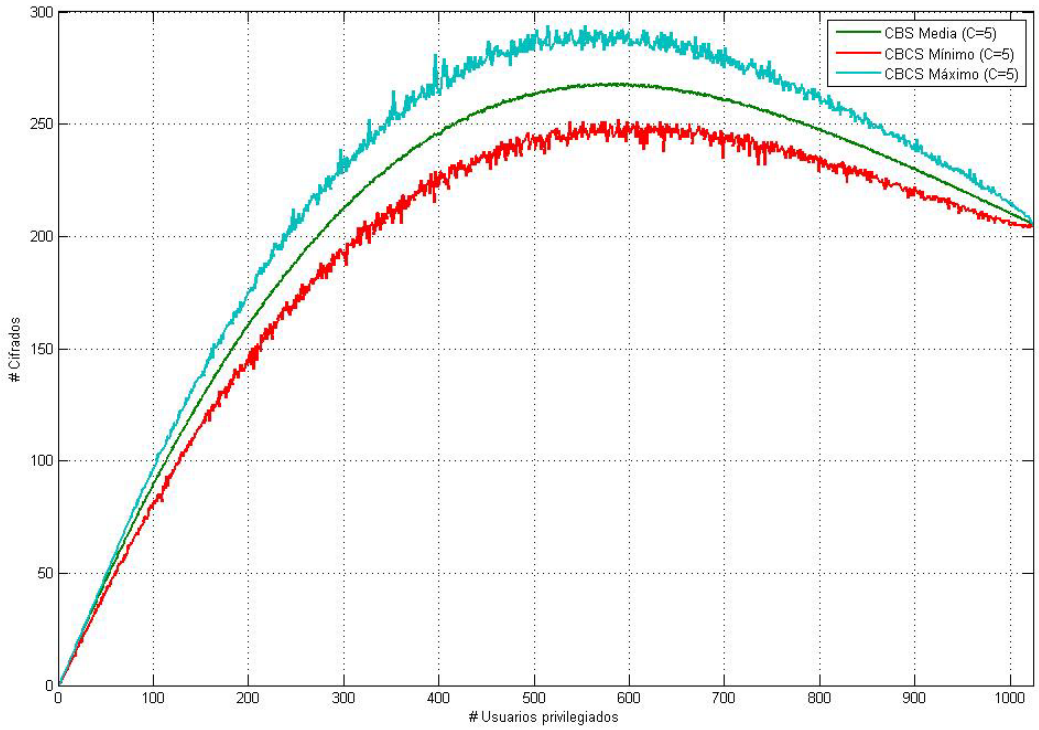

Figura 53: C-basic Chain Scheme TO (*n*=1024)

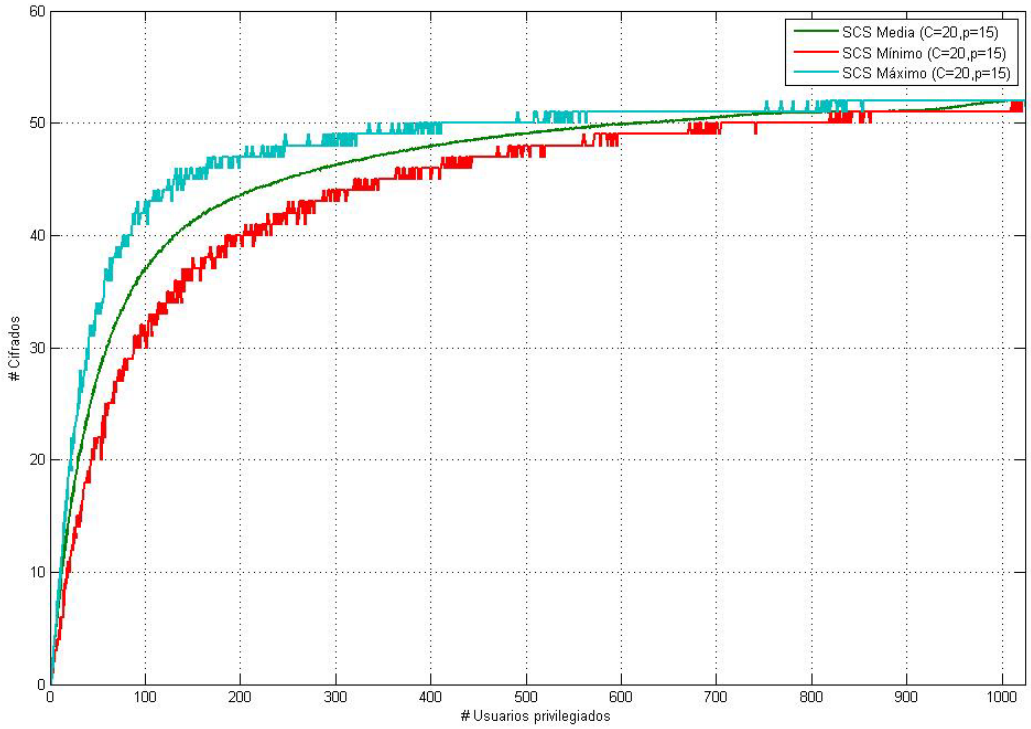

Figura 54: Skipping Chain Scheme TO (*n*=1024)

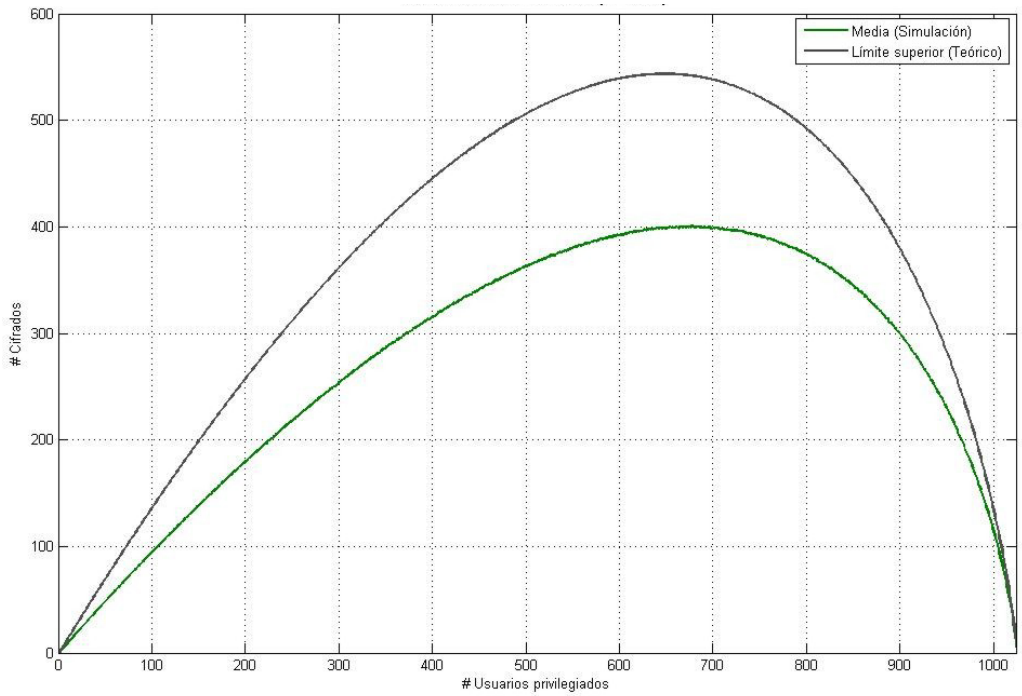

Figura 55: Media y cota superior teórica del TO del CS (*n*=1024)

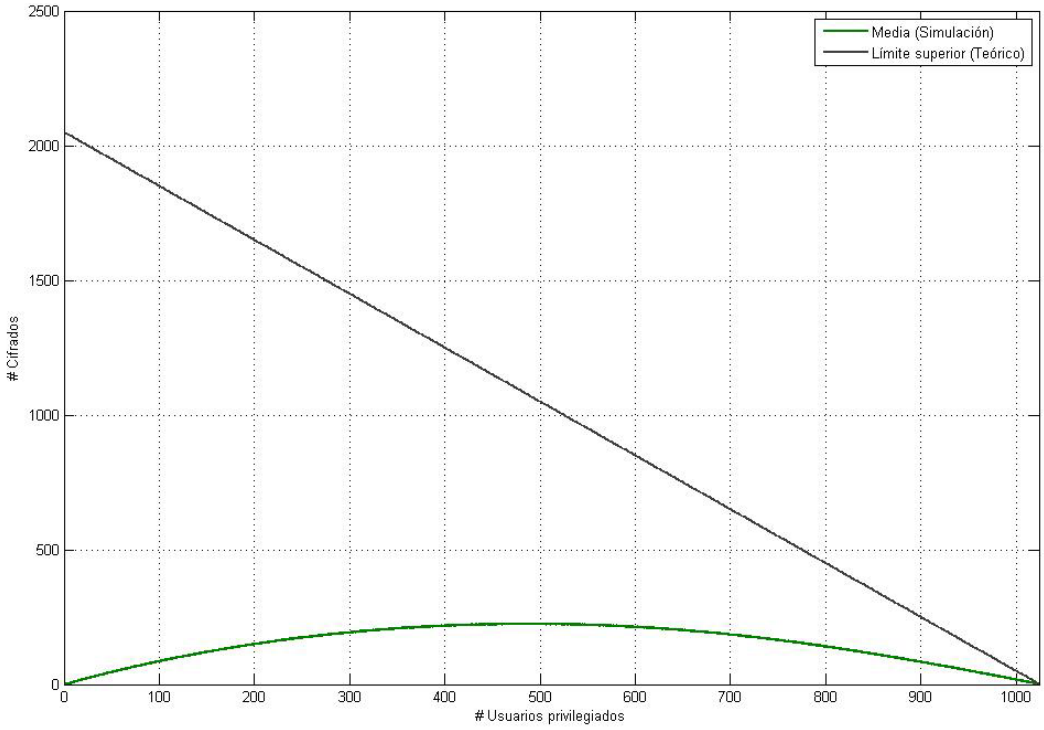

Figura 56: Media y cota superior teórica del TO del SD (*n*=1024)

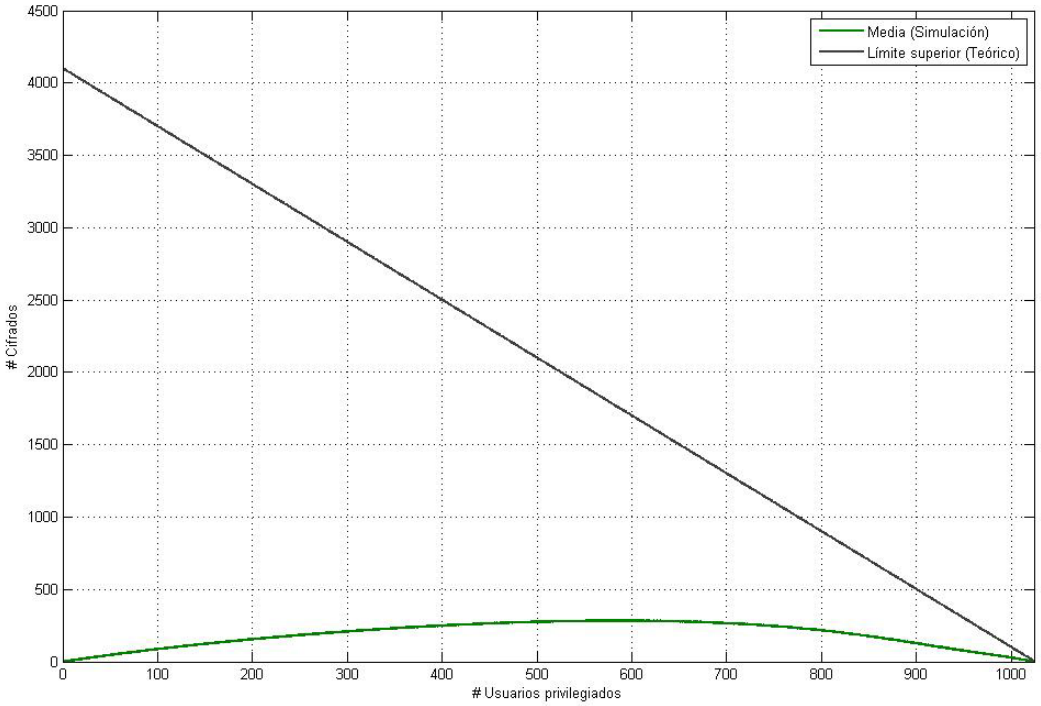

Figura 57: Media y cota superior teórica del TO del LSD (*n*=1024)

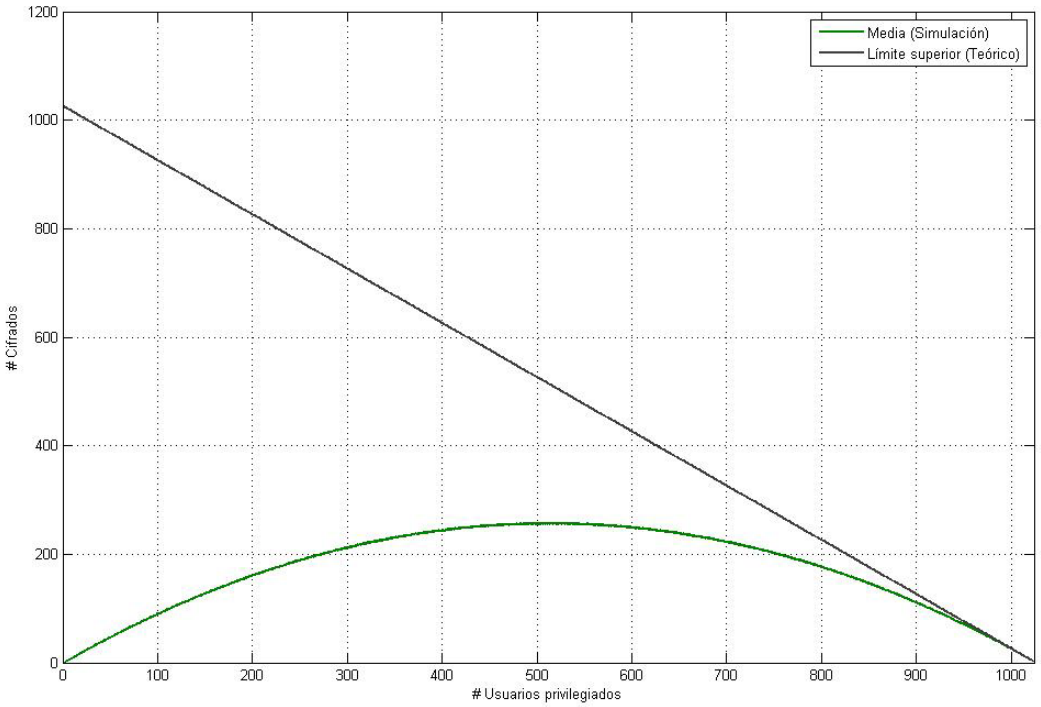

Figura 58: Media y cota superior teórica del TO del BCS (*n*=1024)

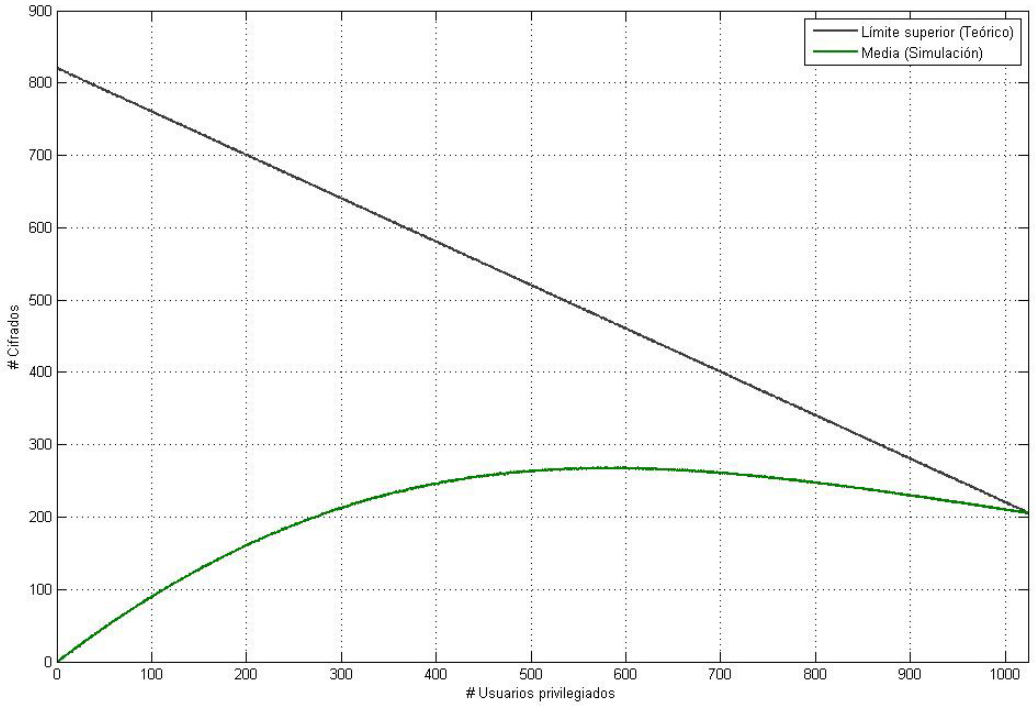

Figura 59: Media y cota superior teórica del TO del CBCS (*n*=1024)

9.4 Ejemplo simple del descifrado de un EMM de Nagravision 2

Instrucción original:
21006DA0CA00006704654001920018F56C533A32C5526B31504793F1189C9DAE6447CD
140F9ACB7963B662BB16BB53AF780EC4A340217877385B02A9090519F58EF9FEBE332
69E23DF46239CF8BDDCAD43B6A67F72D5F44938F1C9BD40D0457B5030FA557A0267
8F9F1BDC8D8FE282

Clave de selección...: 42 18 52 10 03 18 (EMM-G) 4001 Conjunto de
claves 2 (RAM IKEY AC3)
CRL.....: 3E

Tamaño del mensaje a descrifrar: 96 bytes

Paso 1: RSA de 768 Bits

M	F56C533A32C5526B31504793F1189C9DAE6447CD140F9ACB7963 B662BB16BB53AF780EC4A340217877385B02A9090519F58EF9FE BE33269E23DF46239CF8BDDCAD43B6A67F72D5F44938F1C9BD4 0D0457B5030FA557A02678F9F1BDC8D8FE282
N	87BABC043F6656A504011FF45D1F3778CEA37CFF305821CF4ED0 8B7F8D65CA97CE86C9B6CB045B465A69E6DF7CBAFF95CE272E 0EF7C86EE6AC8B290F269E36DDC792F7A9C25EC607172D91692F A39447DFEA17C4912BDFB4529F2B817849CFE5
e	03
M^e mod N	A92B741C80DC49DDD0C81897C13110E9126F033657EE7F80C28D EA91134D48CC21E49D0E0B40F265FDCEA803A59224EA5DA3BB 74FBDB8BAD3BCFBC66E5CE0E0FD80C37A3D36E4F5F42450DD F50BF2BEDADD499175C2E6464C9F4326608367115

Paso 2: Para ajustar el byte que finaliza el RSA previo

Para ajustar el último byte I del mensaje de acuerdo con el bit 7 del tercer byte de la clave de selección

Tercer byte de la clave de selección..: 18 (00011000 en binario)

Último byte del mensaje.....: 15 (00010101 en binario)

15 OR (18 and 80) = 15

Esto se realiza por lo siguiente porque el RSA es un algoritmo en el que interviene la operación "modulo" y por la naturaleza de esta operación nunca puede haber un resultado que sea mayor que el modulo.
Al no poder tener resultados mayores que el modulo, tampoco podemos cifrar datos mayores que el modulo. Ya que el descifrado jamás seria correcto porque daría algo menor que el modulo. Para solucionar esto lo que hacen es quitar siempre el bit más significativo, convirtiendo así siempre los datos en menores que el modulo

Un ejemplo:
Tenemos estas claves para aun RSA: E=03 D=735B y N=AEAF(1010 1110 1010 1111) y queremos cifrar estos datos que son mayores que el modulo datos=D036 (1101 0000 0011 0110). Si cifráramos sin más obtendríamos: datos^D mod N = 74DC
Si ahora desciframos: 74DC^E mod N = 2187(0010 0001 1000 0111) que nada tiene que ver con el resultado original.
La forma correcta es hacer esto es la siguiente:
1-Quitar el bit mas significativo de los datos..0101 0000 0011 0110=5036. (Ahora son mas pequeños que el modulo)
2-Ahora ciframos 5036^D mod N = 2CFD
3-Transmitimos ese resultado al receptor y le decimos de alguna manera si el bit que quitamos al principio valía 1 o no.
4-Ahora desciframos 2CFD^E mod N = 5036 y sabemos que el bit mas significativo valía 1, así que lo ponemos a uno y nos queda 9036 que eran los datos originales.

En los ECM y EMM pasa esto mismo, el cuerpo del ECM son los datos cifrados a los que les habían quitado ese bit. Y ese bit se dice si era 1 o 0 utilizando un bit del select key. Por eso, después de descifrar el RSA tenemos que mirar ese bit del select key y si era 1 ponerlo, para así obtener los datos originales.
Esto hay que tenerlo en cuenta siempre que se realice un RSA, sin embargo cuando trabajamos con un ECM o EMM solo se comprueba en el primer RSA. Esto es así porque ellos se aseguran de que tras el segundo RSA ese bit deba ser siempre cero y por tanto no necesitan decir si era 1 o no. Ya se sabe que es 0. Siempre el primer byte del resultado del segundo RSA se corresponde con el primer byte de la firma, y el algoritmo con el que se calculo esa firma se asegura de que ese bit sea 0 siempre.

Resultado:

M	A92B741C80DC49DDD0C81897C13110E9126F033657EE7F80C28DEA91134 D48CC21E49D0E0B40F265FDCEA803A59224EA5DA3BB74FBDB8BAD3B CFBC66E5CE0E0FD80C37A3D36E4F5F42450DDF50BF2BEDADD499175C2 E6464C9F4326608367115

Paso 3: IDEACBC de 12 Transformaciones

Clave IDEACBC: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA

Empezamos con el último bloque, supeditamos este bloque al algoritmo IDEA por medio de la clave IDEACBC. El resultado XOR con el byte previo sucesivamente hasta alcanzar el primer byte que está relacionado solamente a la clave IDEA, sin el XOR ya que no es bloque precedente.

Bloque 12.....: C9 F4 32 66 08 36 71 15
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: D2 B7 08 FA 2D 28 0D 79
Bloque cifrado XOR Bloque 11 = 7F 63 91 ED 71 06 69 1D -- Nuevo bloque 12

Bloque 11.....: AD D4 99 17 5C 2E 64 64
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA

Bloque cifrado.....: D3 37 3A 93 CA 65 1A B2
Bloque cifrado XOR Bloque 10 = 91 72 37 4C 9A DA 31 5F -- Nuevo bloque 11

Bloque 10.....: 42 45 0D DF 50 BF 2B ED
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 40 26 30 78 BF E8 38 E3
Bloque cifrado XOR Bloque 9 = 98 2A 07 DB 6C 86 77 BC -- Nuevo bloque 10

Bloque 9.....: D8 0C 37 A3 D3 6E 4F 5F
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 2A 52 1A AC D9 79 57 5A
Bloque cifrado XOR Bloque 8 = 11 9D A6 CA 3C B7 59 55 -- Nuevo bloque 9

Bloque 8.....: 3B CF BC 66 E5 CE 0E 0F
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 1F 5C 37 E3 7C 98 30 72
Bloque cifrado XOR Bloque 7 = 42 FF 8C 97 87 43 BB DF -- Nuevo bloque 8

Bloque 7.....: 5D A3 BB 74 FB DB 8B AD
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 25 F5 E4 0E 76 32 23 A8
Bloque cifrado XOR Bloque 6 = D8 3B 4C 0D D3 A0 07 42 -- Nuevo bloque 7

Bloque 6.....: FD CE A8 03 A5 92 24 EA
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 91 16 6A E0 63 8D 32 BA
Bloque cifrado XOR Bloque 5 = B0 F2 F7 EE 68 CD C0 DF -- Nuevo bloque 6

Bloque 5.....: 21 E4 9D 0E 0B 40 F2 65
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 85 EC 84 E9 5D 55 A8 AD
Bloque cifrado XOR Bloque 4 = 47 61 6E 78 4E 18 E0 61 -- Nuevo bloque 5

Bloque 4.....: C2 8D EA 91 13 4D 48 CC

Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 3E C0 B6 77 1D 58 AC 02
Bloque cifrado XOR Bloque 3 = 2C AF B5 41 4A B6 D3 82 -- Nuevo bloque 4

Bloque 3.....: 12 6F 03 36 57 EE 7F 80
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: B5 91 8C 7D 34 BA 91 5F
Bloque cifrado XOR Bloque 2 = 65 59 94 EA F5 8B 81 B6 -- Nuevo bloque 3

Bloque 2.....: D0 C8 18 97 C1 31 10 E9
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: A5 94 45 FC 71 98 4E 30
Bloque cifrado XOR Bloque 1 = 0C BF 31 E0 F1 44 07 ED -- Nuevo bloque 2

Bloque 1.....: A9 2B 74 1C 80 DC 49 DD
Calculado con IDEACBC..: 20 37 BC 03 3B 5A 48 66 3B 99 1B F4 EE 80 81 BA
Bloque cifrado.....: 08 1C A4 54 95 AC 9E 0D -- Nuevo bloque 1

Resultado:

M	081CA45495AC9E0D0CBF31E0F14407ED655994EAF58B81B62CFB5414AB6D38247616E784E18E061B0F2F7EE68CDC0DFD83B4C0DD3A0074242FF8C978743BBDF119DA6CA3CB75955982A07DB6C8677BC9172374C9ADA315F7F6391ED7106691D

Paso 4: RSA de 768 Bits

M	081CA45495AC9E0D0CBF31E0F14407ED655994EAF58B81B62CAFB5414AB6D38247616E784E18E061B0F2F7EE68CDC0DFD83B4C0DD3A0074242FF8C978743BBDF119DA6CA3CB75955982A07DB6C8677BC9172374C9ADA315F7F6391ED7106691D
N	87BABC043F6656A504011FF45D1F3778CEA37CFF305821CF4ED08B7F8D65CA97CE86C9B6CB045B465A69E6DF7CBAFF95CE272E0E

	F7C86EE6AC8B290F269E36DDC792F7A9C25EC607172D91692FA39447DFEA17C4912BDFB4529F2B817849CFE5
e	03
M^e mod N	71B0DD03FE7BBC4640011380000115537789834101420110060800 10EFD8DC51244CEF438241D2168FFACA2B42011046080010B997A360E6E6AD87F8457974C06038970000000000000000000000000000000000000000000000000000000000000000

Resultado:

M	71B0DD03FE7BBC4640011380000115537789834101420110060800 10EFD8DC51244CEF438241D2168FFACA2B42011046080010B997A360E6E6AD87F8457974C06038970000000000000000000000000000000000000000000000000000000000000000

Paso 5: Verificar firma

Firma clave IDEACBC: 0E F8 C0 AA A9 40 26 E3 97 B2 75 82 85 2F EE 77

Empezar con el segundo bloque, relacionar con el algoritmo IDEA por medio de la clave IDEACBC. Haz un XOR entre este bloque y el resultado calculado obtenido. A continuación calcula una nueva clave IDEACBC, la primera parte resulta ser la primera y el resultado del cálculo obtenido la segunda parte.
De esta manera hasta que I termine bloque, para que I termine desconectamos el bit 7 del primer byte de las señales it si está encendido.

Bloque 2.....: 40 01 13 80 00 01 15 53
Clave IDEACBC....: 0E F8 C0 AA A9 40 26 E3 97 B2 75 82 85 2F EE 77
Bloque cifrado.....: 1B FE B6 E1 A0 C4 EE 78
(Bloque 2) XOR (Bloque cifrado): 5B FF A5 61 A0 C5 FB 2B – Segundo octeto de la nueva clave IDEACBC
Nueva clave IDEACBC: 97 B2 75 82 85 2F EE 77 5B FF A5 61 A0 C5 FB 2B

Bloque 3.....: 77 89 83 41 01 42 01 10

Clave IDEACBC....: 97 B2 75 82 85 2F EE 77 5B FF A5 61 A0 C5 FB 2B

Bloque cifrado.....: 2E 60 7F B5 18 A2 6A 75

(Bloque 3) XOR (Bloque cifrado): 59 E9 FC F4 19 E0 6B 65 -- Segundo octeto de la nueva clave IDEACBC

Nueva clave IDEACBC: 5B FF A5 61 A0 C5 FB 2B 59 E9 FC F4 19 E0 6B 65

Bloque 4.....: 06 08 00 10 EF D8 DC 51

Clave IDEACBC....: 5B FF A5 61 A0 C5 FB 2B 59 E9 FC F4 19 E0 6B 65

Bloque cifrado.....: C5 88 73 18 1D 50 9A 26

(Bloque 4) XOR (Bloque cifrado): C3 80 73 08 F2 88 46 77 -- Segundo octeto de la nueva clave IDEACBC

Nueva clave IDEACBC: 59 E9 FC F4 19 E0 6B 65 C3 80 73 08 F2 88 46 77

Bloque 5.....: 24 4C EF 43 82 41 D2 16

Clave IDEACBC....: 59 E9 FC F4 19 E0 6B 65 C3 80 73 08 F2 88 46 77

Bloque cifrado.....: 68 04 5B 51 6B 40 CB 80

(Bloque 5) XOR (Bloque cifrado): 4C 48 B4 12 E9 01 19 96 -- Segundo octeto de la nueva clave IDEACBC

Nueva clave IDEACBC: C3 80 73 08 F2 88 46 77 4C 48 B4 12 E9 01 19 96

Bloque 6.....: 8F FA CA 2B 42 01 10 46

Clave IDEACBC....: C3 80 73 08 F2 88 46 77 4C 48 B4 12 E9 01 19 96

Bloque cifrado.....: 59 C7 E9 4F DB 89 46 33

(Bloque 6) XOR (Bloque cifrado): D6 3D 23 64 99 88 56 75 -- Segundo octeto de la nueva clave IDEACBC

Nueva clave IDEACBC: 4C 48 B4 12 E9 01 19 96 D6 3D 23 64 99 88 56 75

Bloque 7.....: 08 00 10 B9 97 A3 60 E6

Clave IDEACBC....: 4C 48 B4 12 E9 01 19 96 D6 3D 23 64 99 88 56 75

Bloque cifrado.....: EE 4E 38 00 C3 7E 47 5B

(Bloque 7) XOR (Bloque cifrado): E6 4E 28 B9 54 DD 27 BD -- Segundo octeto de la nueva clave IDEACBC

Nueva clave IDEACBC: D6 3D 23 64 99 88 56 75 E6 4E 28 B9 54 DD 27 BD

Bloque 8.....: E6 AD 87 F8 45 79 74 C0
Clave IDEACBC....: D6 3D 23 64 99 88 56 75 E6 4E 28 B9 54 DD 27 BD
Bloque cifrado.....: B3 46 73 3B DC F6 8D 2E
(Bloque 8) XOR (Bloque cifrado): 55 EB F4 C3 99 8F F9 EE -- Segundo octeto de la nueva clave IDEACBC
Nueva clave IDEACBC: E6 4E 28 B9 54 DD 27 BD 55 EB F4 C3 99 8F F9 EE

Bloque 9.....: 60 38 97 00 00 00 00 00
Clave IDEACBC....: E6 4E 28 B9 54 DD 27 BD 55 EB F4 C3 99 8F F9 EE
Bloque cifrado.....: D1 C5 4A 86 F9 17 0C AB
(Bloque 9) XOR (Bloque cifrado): B1 FD DD 86 F9 17 0C AB -- Segundo octeto de la nueva clave IDEACBC
Nueva clave IDEACBC: 55 EB F4 C3 99 8F F9 EE B1 FD DD 86 F9 17 0C AB

Bloque 10.....: 00 00 00 00 00 00 00 00
Clave IDEACBC....: 55 EB F4 C3 99 8F F9 EE B1 FD DD 86 F9 17 0C AB
Bloque cifrado.....: 96 04 24 77 AF 96 B5 78
(Bloque 10) XOR (Bloque cifrado): 96 04 24 77 AF 96 B5 78 -- Segundo octeto de la nueva clave IDEACBC
Nueva clave IDEACBC: B1 FD DD 86 F9 17 0C AB 96 04 24 77 AF 96 B5 78

Bloque 11.....: 00 00 00 00 00 00 00 00
Clave IDEACBC....: B1 FD DD 86 F9 17 0C AB 96 04 24 77 AF 96 B5 78
Bloque cifrado.....: C2 1E 0E 2C 86 F9 D5 1E
(Bloque 11) XOR (Bloque cifrado): C2 1E 0E 2C 86 F9 D5 1E -- Segundo octeto de la nueva clave IDEACBC
Nueva clave IDEACBC: 96 04 24 77 AF 96 B5 78 C2 1E 0E 2C 86 F9 D5 1E

Bloque 12.....: 00 00 00 00 00 00 00 00
Clave IDEACBC....: 96 04 24 77 AF 96 B5 78 C2 1E 0E 2C 86 F9 D5 1E
Bloque cifrado.....: 71 B0 DD 03 FE 7B BC 46
(Bloque 12) XOR (Bloque cifrado): 71 B0 DD 03 FE 7B BC 46 -- It signs without adjusting

Paso 6:

Para ajustar el primer byte de la firma
Primer byte de la firma...: 71 (01110001 en binario)
71 AND 7F = 71

Firma original..: 71B0DD03FE7BBC46
Firma calculada.: 71B0DD03FE7BBC46

Firma 71B0DD03FE7BBC46
4001 Proveedor (********* (España))
13800001 Marca de fecha de creación (01/09/2005 0:00:02)
15537789 Marca de fecha de expiración (12/12/2006 17:00:02)

Analizando EMM

Cmd 83 (Cambio de ProvID para el resto de cmd)
Proveedor 4101

Cmd 42 (Actualización de la nueva clave)
Datos 011006080010 Tamaño de los datos (16 en decimal)
clave Idea 06 EFD8DC51244CEF438241D2168FFACA2B ---------- clave original en tarjeta original EN eMU 00 & 01

Cmd 42 (Actualización de la nueva clave)
Datos 011046080010 Tamaño de los datos (16 en decimal)
clave Idea 46 B997A360E6E6AD87F8457974C0603897 -------- clave original en tarjeta original EN eMU 00 & 01

9.5 Esquema de autenticación AACS unidad-host

A continuación se muestra una descripción detallada del protocolo de autenticación del AACS unidad-host. Se puede consultar dicho esquema en [59].

	Unidad		**Host**
1.		MKB →	Compara la versión del DRL almacenado con el DRL del MKB. Si el DRL del MKB no es más nuevo, entonces se usa el DRL almacenado.
2.			Si el DRL del MKB es más nuevo, entonces se verifica la firma del DRL , en el caso de que sea válida se usa el DRL del MKB, en caso contrario se aborta.
3.	Compara la versión de HRL almacenado con el HRL del MKB. Si el HRL del MKB no es más nuevo, entonces se usa el almacenado.		
4.	Si el HRL del MKB es más nuevo, entonces se verifica la firma del HRL , en el caso de que sea válida se usa el HRL del MKB, en caso contrario se aborta.		
5.		AGI D →	
6.			Genera 160 bits para Hn
7.		← H_n, H_{cert}	

8. Verificar el tipo y la longitud del certificado del host. Aborta en caso de error
9. Verifica la firma del certificado del host. Aborta en caso de error.
10. Comprueba el HRL y abortar en caso de encontrar el ID del host
11. Solicita Dn y certificado de la unidad
12. Genera 160 bits para Dn
13. D_n, D_{cert} →
14. Verifica el tipo y la longitud del certificado de la unidad. Abortar en caso de error
15. Verifica la firma del certificado de la unidad. Abortar en caso de error.
16. Comprueba el DRL y abortar en caso de encontrar el ID de la unidad
17. Solicita el punto Dv de la curva elíptica y su firma
18. Genera un número aleatorio Dk de 160 bits
19. Calcula Dv = DkG donde G es el punto de la base de la curva elíptica
20. Calcula Dsig como la firma de Hn || Dv usando la clave privada de la unidad
21. D_v, D_{sig} →

22.			Verifica Dsig y aborta en caso de error
23.			Generar un número aleatorio Hk de 160 bits
24.			Calcula Hv = HkG
25.			Calcula Hsig como la firma de Dn || Hv usando la clave privada del host
26.		H_v, H_{sig} ←	
27.	Verifica Hsig y aborta en caso de error		
28.	Calcula la clave Bus Bk como los 128 bits menos significativos de la coordenada-x(DkHv)		
29.			Calcula la clave Bus Bk como los 128 bits menos significativos de la coordenada-x(HkDv)