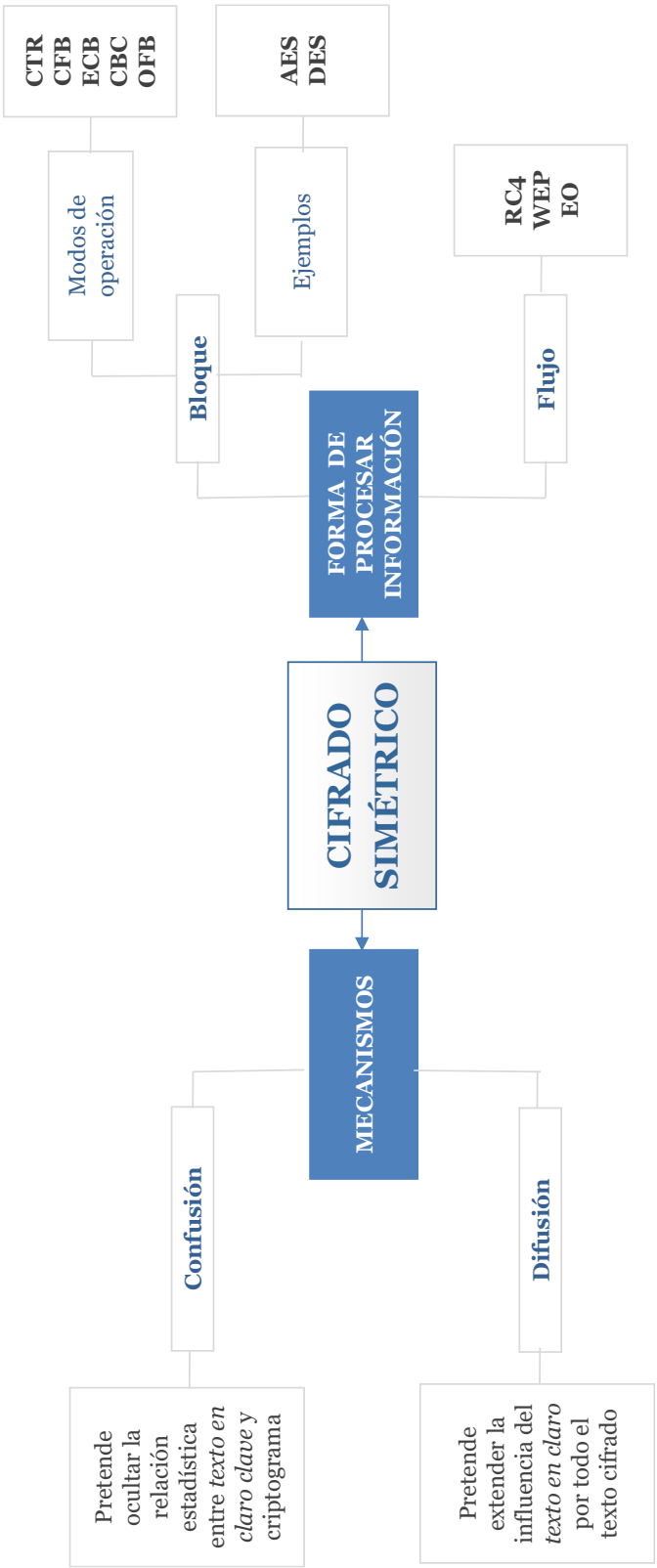
Criptografía simétrica

TEMA 3

Esquema


CIFRADO SIMÉTRICO
FORMA DE PROCESAR INFORMACIÓN
Bloque
Modos de operación
CTR
CFB
ECB
CBC
OFB
Ejemplos
AES
DES
Flujo
RC4
WEP
EO
MECANISMOS
Confusión
Pretende ocultar la relación estadística entre *texto en claro clave* y criptograma
Difusión
Pretende extender la influencia del *texto en claro* por todo el texto cifrado

Ideas clave

3.1. ¿Cómo estudiar este tema?

El estudio de este tema se realiza a través de los contenidos desarrollados en las **Ideas clave** expuestas a continuación.

El objetivo principal del tema es aprender las características principales y tipos de algoritmos criptográficos simétricos.

Según estudiamos en el tema anterior la **criptografía** puede definirse como el «estudio de las técnicas matemáticas necesarias para proporcionar aspectos de seguridad al tratamiento de información, tales como **privacidad**, **integridad**, **autenticación o no repudio**».

Concretamente, la **privacidad** de la información hace referencia al hecho de mantener secreto el contenido de cierta información para aquellas partes no autorizadas. Habitualmente, la privacidad se obtiene a través de técnicas de *cifrado*.

Por otro lado, la **integridad** de la información asegura que ésta no ha sido modificada en forma alguna de forma no autorizada. La manipulación puede consistir en la inserción de información adicional, el borrado o la sustitución de la misma. La integridad puede asegurarse mediante unas primitivas criptográficas llamadas funciones *hash*.

La **autenticación** asegura que la información proviene realmente de las partes autorizadas, y no de una tercera que está tratando de suplantarlas. Para poder realizar esta comprobación, dichas partes deben ser previamente identificadas. Por tanto, la **identificación** y la autenticación pueden ser consideradas equivalentes y aplicadas, respectivamente, a entidades e información. La autenticación se obtiene con el empleo de las denominadas *firmas digitales*.

Por último, el último de los aspectos básicos para poder hablar de seguridad de la información consiste en el **no repudio**. Éste hace referencia a la imposibilidad de las partes implicadas de negar que hayan emitido cierto mensaje o que hayan participado en una comunicación determinada.

Habitualmente es necesario el concurso de una tercera parte, en la que el resto deposita su confianza, y que son, por tanto, denominadas *terceras partes confiables* (TTP, de su acrónimo inglés *Third Trusted Parties*).

Puedes encontrar un resumen de todos estos aspectos en **Figura 1**.

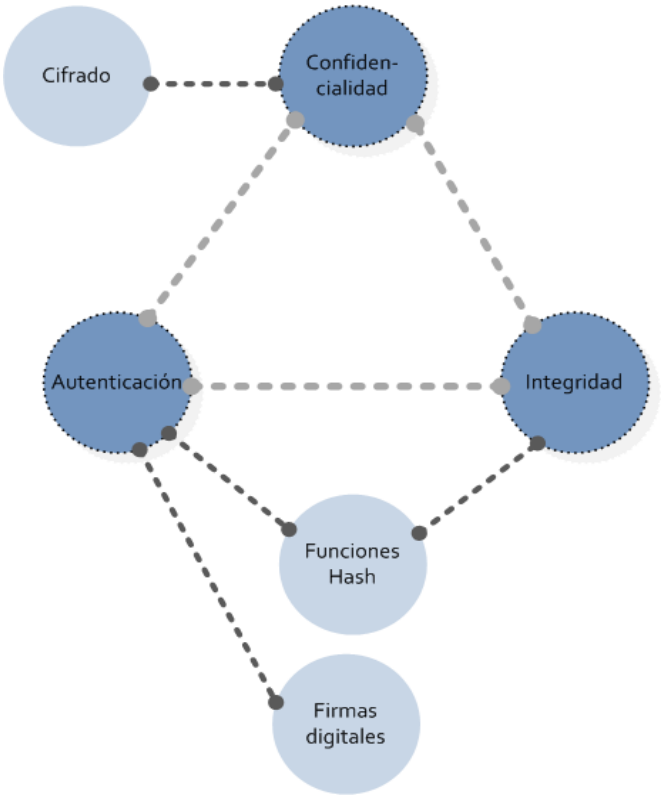


Figura 1. Aspectos de seguridad proporcionados por las técnicas criptográficas

3.2. Introducción y terminología básica

¿Cuántas veces accedes a una web al cabo del día?, a bien seguro que muchas, e incluso más de las veces que crees lo haces a servicios HTTPS (banca, redes sociales, etc.). ¿Qué clase de algoritmos usan?, ¿con todo tipo de navegadores? Paralelo a esto, y casi tan cotidiano como visitar sitios web, está el envío de correos electrónicos, usando también algoritmos criptográficos, como parte de la creación de canales seguros de comunicación. La criptografía simétrica ha llegado para no marcharse.

Comenzaremos definiendo una serie de conceptos que necesitaremos a lo largo del resto del tema. Estudia la **Figura 2** y repasa cada uno de los elementos que en ella se exponen.

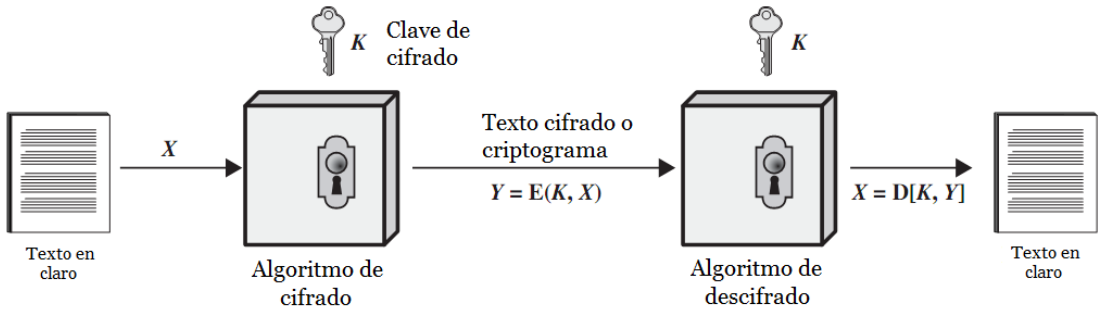


Figura 2. Proceso básico de cifrado/descifrado simétrico

Tipos de algoritmos de cifrado

Los **algoritmos de cifrado** pueden clasificarse en función de varios criterios, como el tipo particular de transformación aplicada al texto en claro o las características de las claves utilizadas. Una primera clasificación en base a este último criterio, el más habitual, es la siguiente:

- **Métodos simétricos o de clave secreta**: son aquellos en los que la clave de cifrado y descifrado coinciden. Como es lógico, esta clave *debe ser mantenida en secreto*, lo que supone que emisor y receptor deben acordarla previamente a la comunicación. O bien, que existe un *centro de distribución de claves* que ha hecho llegar dicha clave a ambos por un canal seguro.
- **Métodos asimétricos o de clave pública**: son aquellos en los que la clave de cifrado y descifrado son diferentes. Normalmente la clave de cifrado es públicamente conocida, mientras que la de descifrado es secreta y conocida únicamente por su propietario.

En este tema estudiaremos con detalle el primer grupo, dejando para el siguiente los importantes algoritmos de clave pública.

La siguiente figura resume estos dos enfoques y proporciona algunos ejemplos sobre algoritmos concretos pertenecientes a cada tipo.

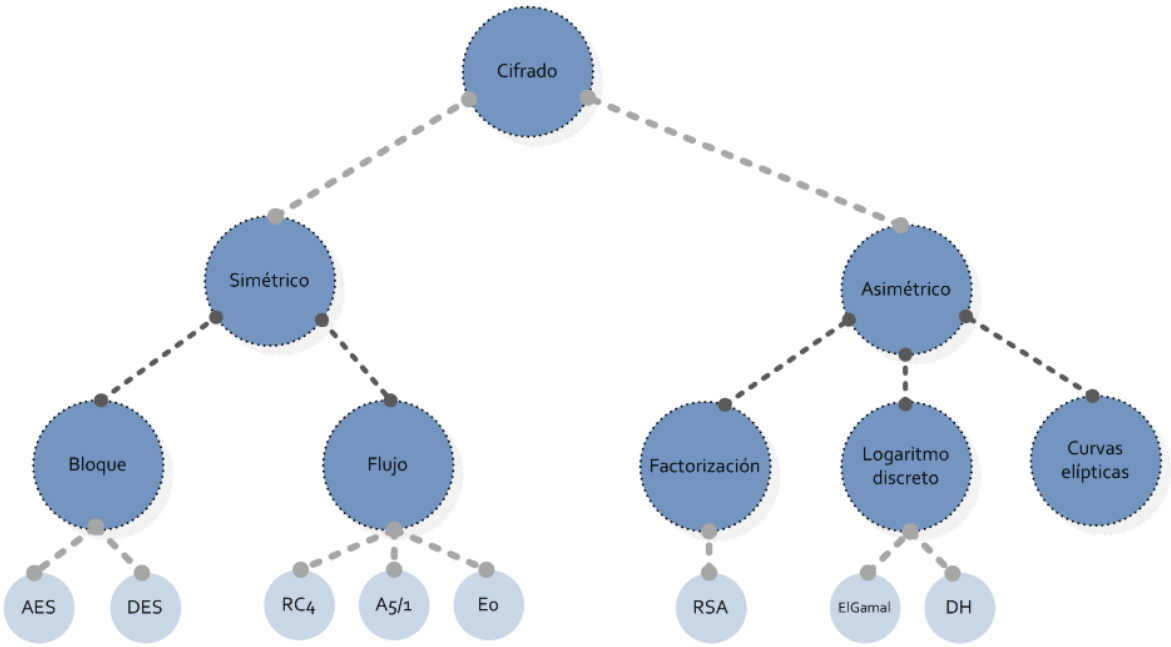


Figura 3. Tipos básicos de técnicas de cifrado

Criptografía simétrica vs asimétrica

Llegados a este punto podríamos preguntarnos, **¿por qué existen dos tipos de algoritmos de cifrado?** Si ambos sirven para cifrar, **¿por qué no utilizar solo un tipo de ellos?** Como suele ocurrir, cada tipo tiene sus ventajas e inconvenientes, por lo que no existe un tipo de algoritmo ideal para todas las situaciones.

Por ejemplo, a continuación se enumeran algunos de los inconvenientes inherentes al uso de criptografía simétrica, el tipo que nos ocupa en este tema:

- **Gestión de claves problemática:** en una comunicación entre *n* usuarios, el número de claves necesarias crece en exponencialmente con un factor igual a n^2. En la práctica, este hecho imposibilita su uso como único método de cifrado en una gran organización.
- **Distribución de claves problemática:** no existe posibilidad de enviar, de forma segura y eficiente, una clave a través de un medio o canal inseguro.
- **No dispone de firma digital**: Aunque sí es posible autenticar el mensaje mediante ciertas primitivas simétricas, denominadas funciones hash con clave, no es posible firmar digitalmente el mensaje, con las ventajas que aporta en este aspecto los algoritmos de clave pública.

Además, estos inconvenientes resultan los recíprocos de la asimétrica, por lo que éstos serán sus ventajas y viceversa. En la siguiente figura se resumen estas características:

Figura 4. Comparativa criptografía simétrica vs asimétrica

3.3. Cifrado simétrico

Ante estos inconvenientes de la criptografía simétrica, podríamos entonces hacernos una pregunta natural: ¿qué ventajas tiene, entonces, el uso de cifrado simétrico? La respuesta, en esencia, es su **velocidad**. La velocidad del cifrado simétrico es muy alta, mucho más que en el caso asimétrico.

Esta velocidad es consecuencia de que **los algoritmos simétricos son**, en general, **mucho más simples y necesitan menor capacidad de cómputo que los asimétricos**. Esto les permite ejecutarse en máquinas con muy poca capacidad de proceso, como microcontroladores y pequeños microprocesadores. A día de hoy muy pocos microcontroladores pueden realizar operaciones de cifrado utilizando algoritmos de clave pública.

Otra ventaja importante en determinados escenarios es que **las claves simétricas son**, en general, **más cortas en longitud que sus equivalentes asimétricos** (128 bits frente a 1024, por ejemplo), por lo que en dispositivos con muy poco memoria puede ser un aspecto importante.

Principios básicos de diseño: confusión y difusión

En general, todos los algoritmos de cifrado, simétricos y asimétricos, funcionan transformando el texto en claro a través de dos mecanismos básicos: **confusión** y **difusión**.

En esencia, la **confusión** pretende ocultar la relación existente entre texto en claro, texto cifrado y la clave. Un buen mecanismo de confusión hará muy difícil extraer relaciones estadísticas entre ellos La confusión del texto en claro puede conseguirse con una simple tabla de sustitución, que toma una entrada de *m* bits y los sustituye por otros *n* bits de acuerdo a la transformación definida en una tabla. Estas tablas se conocen con el nombre de ***cajas S*** (S de cajas de Sustitución) en criptografía. Puedes encontrar un ejemplo de una *caja S* de 8x6 en la **Figura 5**.

El algoritmo de cifrado *DES*, por ejemplo, emplea ocho *cajas S* de 6x4 bits. Normalmente, cuanto mayor sea el tamaño de las *cajas S*, más resistente será el algoritmo que las utilice. Sin embargo, el problema de los valores concretos que componen una *caja S* no es en absoluto trivial y es un tema activo de investigación en la comunidad criptográfica.

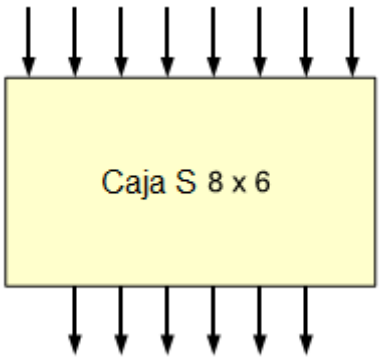


Figura 5. Caja S de 8 entradas y 6 salidas

Por otro lado, el objetivo de la **difusión** es **«repartir» la influencia de cada bit del texto en claro entre el texto cifrado**. En otras palabras, se trata de que cada bit del texto cifrado dependa todo lo posible del mayor número posible de bits del texto en claro. La difusión puede conseguirse de muchas maneras pero un mecanismo ilustrativo puede ser transformar un mensaje de entrada $M = m_1, m_2,\ldots, m_n$ con una operación de suma modular:

$$y_n = \left(\sum_{i=1}^{k} m_{n+i} \right) \bmod 26$$

De esta forma tan simple, la estructura estadística del texto en claro queda «difuminada» en el texto cifrado. Por supuesto, este es un mecanismo extremadamente simple, mostrado sólo a ejemplo ilustrativo, que no aporta seguridad real. En los algoritmos reales, como veremos a continuación, la difusión suele obtenerse a través de **operaciones de permutación**.

Ahora bien, para obtener una seguridad completa tendríamos que definir *cajas S* diferentes para cada par <clave, texto en claro>. Obviamente esto es completamente inviable, por lo que en la práctica se intercalan alternativamente operaciones de confusión (sustituciones simples, con tablas pequeñas) y de difusión (permutaciones). Estas estructuras se denominan **redes de sustitución-permutación**, o redes S-P.

Cifradores simétricos de bloque

A su vez, en función de cómo tratan la información durante el proceso de cifrado, los algoritmos de cifrado simétrico se clasifican en cifradores de **bloque** y de **flujo**. Vuelve a consultar la **Figura 3** y observa la **rama de cifrado simétrico**.

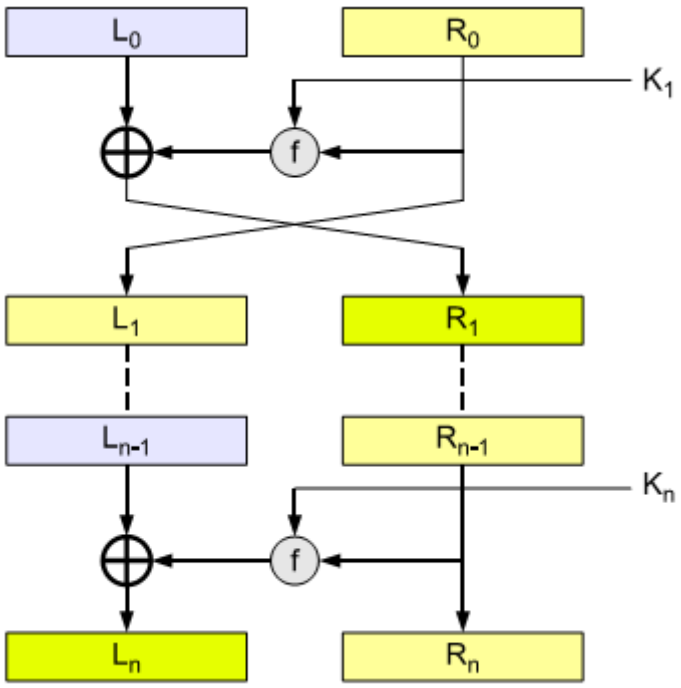


Figura 6. Red de Feistel típica

En este tipo de cifrado, el mensaje se agrupa en bloques, por lo general de 8 ó 16 bytes (64 ó 128 bits), antes de aplicar ciertas operaciones iterativamente a cada uno de los bloques. A su vez, estos bloques de longitud *n* suelen dividirse en dos mitades, *L* y *R* (de Left y Right). Entonces el proceso de cifrado suele definirse aplicando repetidamente una serie de rondas, donde la salida de cada una de ellas se utiliza como entrada para la siguiente (véase la **Figura 6**).

Este tipo de estructura se denomina de forma genérica **red de Feistel**, y se emplea en muchos de los algoritmos de bloque más utilizados y conocidos, como *DES*, *FEAL*, *CAST*, *Blowfish*, etc. Su propiedad más destacable es que para invertir la función de cifrado, es decir, para descifrar, basta aplicar el mismo esquema con las subclaves K_i en orden inverso. Esta característica simplifica mucho su implementación tanto en software como hardware.

La mayoría de algoritmos de cifrado simétrico son de bloque. Algunos muy conocidos son *DES* (utilizado todavía en muchas aplicaciones bancarias, aunque ha sido sustituido oficialmente por el nuevo estándar *AES*), *IDEA*, *CAST* (utilizado en el correo electrónico) o el *3DES* (utilizado en comercio electrónico).

Tamaño de bloque

Otro aspecto importante de los cifradores de bloque es el tamaño del mismo. **¿Cuál es el tamaño de bloque óptimo?** En realidad, no existe una única respuesta. Si el bloque fuese muy pequeño, por ejemplo de uno o dos bytes, se facilitaría un ataque estadístico del cifrador. Se trataría en realidad, de un **cifrado por monogramas o digramas muy débil**.

Por otro lado, si el bloque fuese muy grande, por ejemplo de cientos de bytes, el sistema sería lento en el tratamiento del *texto en claro* y no tendría un buen rendimiento. Los valores habituales de 64 y 128 bits son un término medio que satisface ambas condiciones, una especia de valor de compromiso.

Algoritmo	Bloque (bits)	Clave (bits)	Rondas
DES	64	56	16
3DES	64	112	16
CAST	64	64	8
Blowfish	64	variable	16
AES	128	128 o más	flexible
Twofish	128	variable	variable
RC5	Variable	Variable	variable

Tabla 1. Tabla de algoritmos de cifrado en bloque

3.4. Modos de operación

Los **modos de operación** hacen referencia a cómo se tratan los bloques de información durante el proceso de cifrado. Básicamente, existen **cuatro grandes modos**, resumidos en la **Figura 7**. En los últimos años se ha propuesto y comenzado a utilizar también un nuevo modo, llamado modo CTR (de *CounTeR mode*, en inglés), debido a su simplicidad y alta velocidad de cifrado.

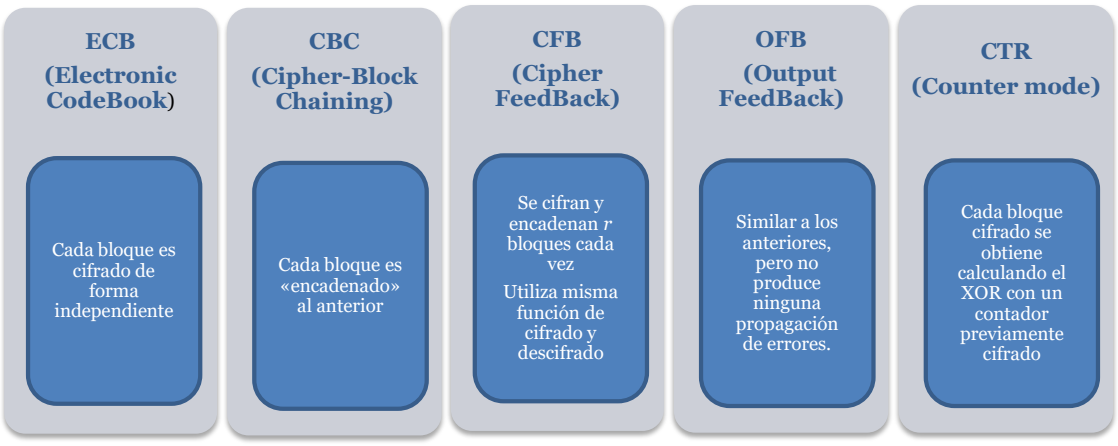


Figura 7. Resumen de los modos de operación típicos del cifrado en bloque

Modo ECB – Electronic Codebook

Este modo es el más sencillo ya que consiste simplemente en cifrar cada bloque de forma independiente de los demás, tal y como puede verse en la **Figura 8¡Error! No se encuentra el origen de la referencia.**.

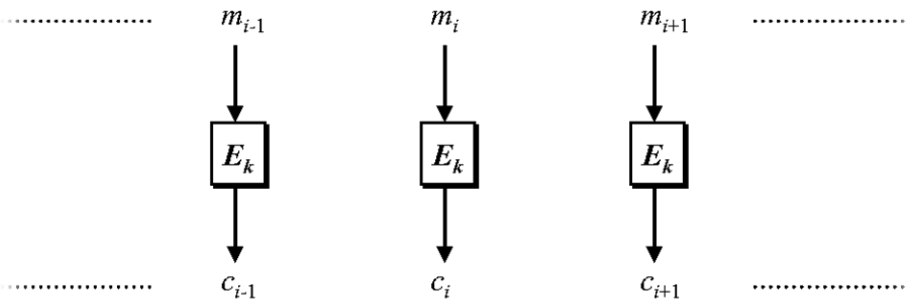


Figura 8. Modo ECB. Cada bloque de entrada es cifrado de manera independiente

Las **desventajas** de este enfoque resultan claras:

- La ventaja principal es que **no «oculta» los patrones que pueda haber en los datos en claro**, pues bloques idénticos de *texto en claro* producirán siempre iguales textos cifrados. Un ejemplo claro de qué tipo de problemas puede acarrear este hecho puede verse en la **Figura 9**. Es fácil observar cómo en la imagen cifrada con el **modo ECB** aún pueden distinguirse las zonas donde existe imagen sobre el fondo blanco. Esto es, en cualquier caso, inadmisible para un cifrador moderno de calidad y es aún más grave dependiendo del contenido concreto de la imagen (como un mapa, por ejemplo).
- Por otro lado, un atacante activo **puede manipular los bloques cifrados, modificando el orden, insertando nuevos o eliminado algunos de ellos**.

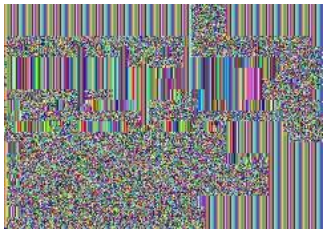
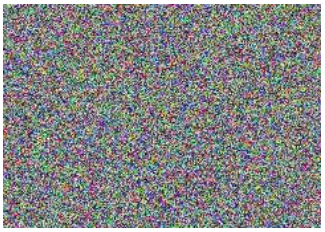
UNIVERSIDAD INTERNACIONAL DE LA RIOJA		
Imagen en claro	**Imagen cifrada con el modo ECB**	**Imagen cifrada con cualquier**

Figura 9. El cifrado ECB no es adecuado para determinados usos

En resumen, este modo **resulta inseguro** para la mayoría de aplicaciones y la recomendación es que no se use ya en la actualidad.

Modo CBC – Cipher Block Chaining

Para evitar los problemas anteriores, el modo CBC **encadena los bloques unos a otros**, de forma que **el cifrado se «propaga» por todos ellos**. Como consecuencia, el cambio en el orden de los bloques cifrados (intencionado o no) hace que el proceso de descifrado falle.

Ya que el primer bloque no tiene uno previo al que encadenarse, se utiliza un **vector de inicialización**. Éste es un valor que no necesita ser secreto, pero que debe ser tan aleatorio como sea posible. En la **Figura 10** puede observarse un esquema de este modo de funcionamiento.

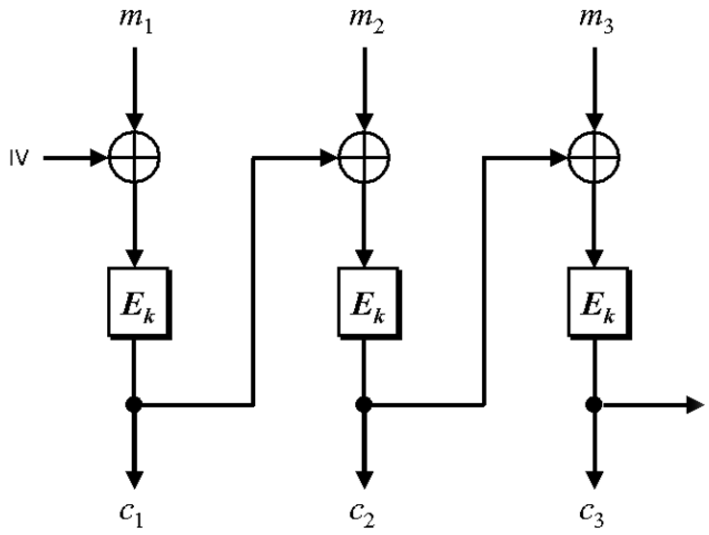


Figura 10. Modo CBC. Cada bloque está vinculado al anterior

Modo CFB – Cipher Feedback

Por último, el modo CFB trata de solventar todos los inconvenientes de los modos anteriores y lleva a cabo una **realimentación de los bloques implicados**. Por tanto, **permite unidades de datos más pequeñas que bloques**, por lo general un byte.

Al igual que el modo CBC, utiliza un **vector de inicialización**, pero proveniente de un registro de desplazamiento de 64 bits. Enmascara también el mensaje, pero en este caso la propagación de un error se limita a un solo bloque. La **Figura 11** resume su esquema de funcionamiento.

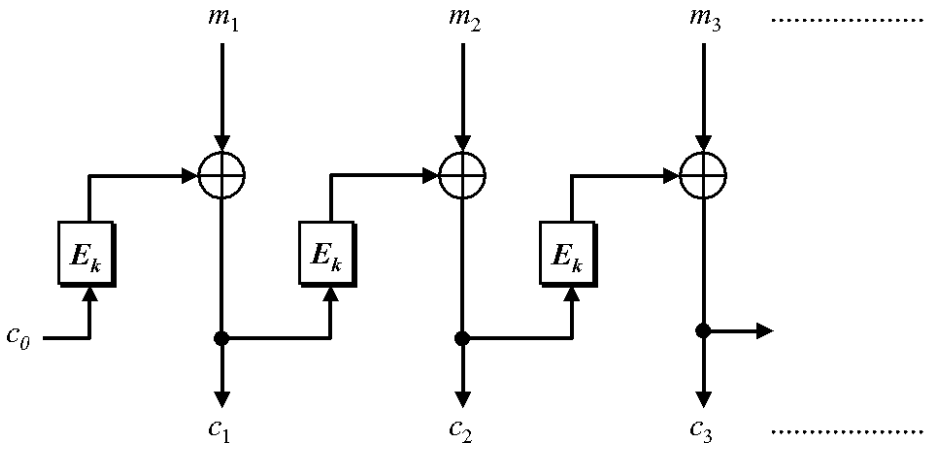


Figura 11. Modo CFB. Los bloques se realimentan unos a otros

Modo CTR– Counter mode

Aunque este modo fue propuesto por Diffie Hellman en nada menos que 1979, nunca había tenido popularidad ni había sido utilizado de forma masiva, como otros modos. Esto ha cambiado, por fin, en los últimos años, cuando criptógrafos reconocidos han insistido en sus numerosas ventajas y alta seguridad que proporciona.

En esencia, el modo utiliza un valor, similar el vector de inicialización del resto de modos, pero que en este caso suele llamarse **contador**. El proceso de cifrado es muy sencillo: se toma este valor, se cifra con la *clave K* correspondiente y al resultado se le aplica la *función XOR* (suma binaria) junto con el *texto en claro* para obtener el primer bloque cifrado. Después se incrementa el valor del contador y se repite el proceso con el resto de bloques del mensaje (sin realimentación de los mismos). En la **Figura 12** puedes encontrar un esquema del funcionamiento.

Normalmente el contador se inicializa a algún valor aleatorio y a partir de ahí, se incrementa para cada bloque. La única condición a cumplir es que sea **diferente para todos los mensajes cifrados con la misma clave**. Este aspecto es esencial y constituye quizás la única desventaja o debilidad que puede achacarse a este modo: si

se usa un mismo valor para el contador múltiples veces, entonces la confidencialidad de todos los bloques cifrados con dicho valor puede verse comprometida.
Sin embargo, las ventajas del modo sobrepasan con creces este inconveniente (que, en cualquier caso, también sufren el resto de modos con el tratamiento del vector de inicialización). Algunas de estas ventajas son una **gran eficiencia de implementación tanto en hardware como software**, que permite descifrar un bloque concreto del mensaje cifrado sin tener que descifrar todos los anteriores y que es, al menos, tan seguro como el resto de modos.

En resumen, **hoy en día los expertos recomiendan utilizar únicamente dos modos de operación, CBC y CTR, para la mayoría de aplicaciones**.

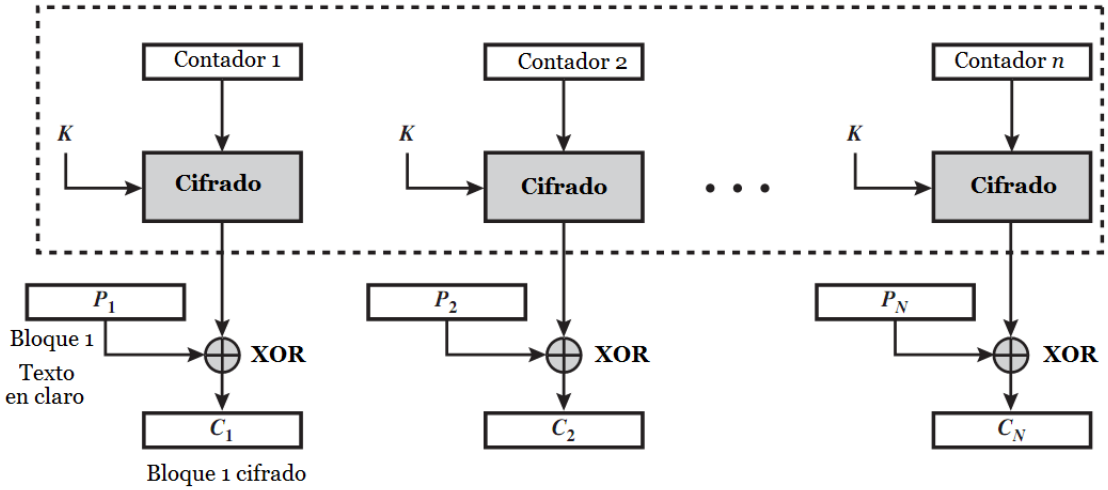


Figura 12. Modo de operación CTR

3.5. DES

DES (Data Encryption Standard) es sin duda el algoritmo criptográfico más conocido de la historia. Hasta su creación oficial en el año 1975, la criptografía era materia reservada y secreta absolutamente controlada por los servicios secretos (la NSA en EEUU) y el ejército.

DES, de origen civil, sirvió en cierta forma para acabar con esa hegemonía. La historia es casi novelesca y muy interesante: comienza en EEUU en 1971 con Horst Feistel, de origen alemán, que trabajaba en ese momento para las Fuerzas Áreas. Feistel, cansado de que la NSA dirigiera todo su trabajo, decidió incorporarse a *IBM*, donde podría

desarrollar su idea de un ***criptosistema basado en bloques***. Su motivación era su preocupación por la privacidad personal, problema que conocía de «primera mano».

National Security Agency: es la famosa agencia de espionaje norteamericana, encargada de la seguridad en las comunicaciones militares y gubernamentales, así como de interceptar y penetrar en las comunicaciones de ejércitos y gobiernos de otros países. Fue creada a finales de 1952, con el objeto de reunir en un solo órgano los servicios de inteligencia de los tres cuerpos del ejército americano. Emplea más personas que la CIA y sus instalaciones ocupan un espacio similar al del Pentágono.

El primer prototipo se llamó **Demon**, debido a que el lenguaje de programación utilizado para codificarlo no admitía nombres largos como «Demostration». Luego, un compañero de *IBM* lo cambió por ***Lucifer***, un juego de palabras en inglés con las palabras *cipher* y *Lucypher*. Rápidamente, IBM solicitó y obtuvo varias patentes sobre *Lucifer*. Enseguida llegó la primera aplicación «seria» de Lucifer. Lloyds utilizó el sistema para su red de cajeros en Londres.

Ese fue el punto de inflexión, pues en *IBM* descubrieron que Lucifer tenía aplicaciones reales (y rentables). Así que IBM puso a trabajar a todo un equipo completo y creó un departamento de banca. Enseguida llegaron los problemas: a principios de 1974 la *NSA* decidió que ya «les había dejado jugar bastante» y pidió a *IBM* que fueran a verles:

«Queremos controlar la implementación. Ustedes lo desarrollan en secreto y nosotros controlamos el proceso y sugerimos los cambios. No queremos que se venda, sobre todo a ciertos países. Le daremos una lista y las licencias de exportación dependerán de que el cliente, autorizado por nosotros, haya firmado un documento en el que se comprometa a no revenderlo. A cambio, nuestros criptoanalistas analizarán el algoritmo. Si tiene algún punto débil, será detectado.»

IBM decidió aceptar la oferta de la NSA, porque apenas había información pública disponible sobre cómo construir un sistema criptográfico de seguridad militar. Walt Tuchman, nuevo responsable del proyecto, declaró:

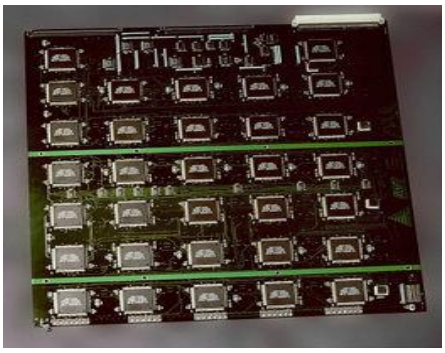
«Todo era materia reservada. Sabíamos, gracias a nuestros conocimientos en matemáticas, qué era lo que hacía que una cifra fuera difícil de descifrar. Pero necesitábamos una seguridad extra, que sólo la NSA nos podía dar.»

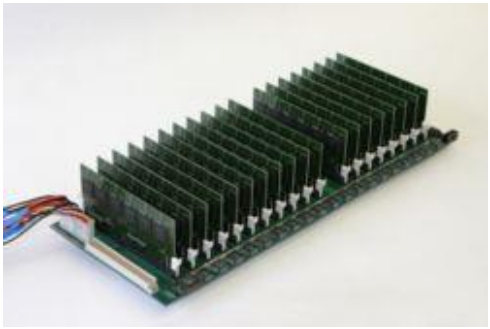
Este fue el inicio de todas las suspicacias que han rodeado a *DES* durante todas su historia. Los hechos ciertos son que:

- ***Lucifer* tenía unas *cajas S* particulares creadas por Feistel.** Cuando el algoritmo volvió de la NSA en una de sus muchas revisiones, las cajas S habían cambiado completamente. Por supuesto, sin explicaciones.
- **Feistel ideó *Lucifer* con una longitud de clave de 128 bits.** La NSA obligó a rebajar esta longitud hasta 56 bits. ¿Por qué?

Parece claro que la NSA rebajó la longitud de clave hasta el número de bits «que ellos podían romper fácilmente». Aunque se ha especulado mucho sobre que la particular disposición de las *cajas S* era una puerta trasera de la NSA, es poco probable. La NSA ha podido descifrar siempre los mensajes cifrados con DES, simplemente, por «fuerza bruta».

En cualquier caso, DES se considera inseguro hoy en día y en su lugar, debería utilizarse AES, el nuevo estándar, en cualquier aplicación. Existen he hecho, máquinas específicamente diseñadas para ello pueden romper un texto cifrado con DES en unas horas pocas horas (habitualmente menos de 24), pudiendo llegar a probar más de 90 billones de claves por segundo.





3DES

Debido a DES sigue en el corazón de muchos sistemas financieros y de comercio electrónico que no son fáciles (ni baratos) de cambiar, se desarrolló una solución: una especie de **DES fortalecido**. Ideado también para evitar los problemas de longitud de clave insuficientes, el algoritmo ***3DES* (Triple-DES)** es básicamente, **tres cifrados simples de DES encadenados, cada uno realizado con una clave diferente**.

De esta forma, se obtiene una longitud de clave teórica de 168 bits (56 * 3 bits) pero, en la práctica, esta se reduce a solo 112 bits, pues es vulnerable a un ataque denominado *meet-in-the-middle*. Aunque, evidentemente, resulta más lento que el DES, es mucho más seguro, pues el principal problema de este es su escasa longitud de clave. Resulta además fácil de implementar, pues reutiliza gran parte del código existente.

Como ya se ha comentado, es ampliamente superado en todos los aspectos técnicos por AES, pero todavía utilizado ampliamente en el sector bancario (por ejemplo, en el estándar EMV de medios de pago).

Modos de cifrado de 3DES

La variante más simple opera de la siguiente forma:

$$C = DES(k_3;DES(k_2;DES(k_1;M)))$$

Donde *M* es el mensaje, *C* el criptograma y k_1, k_2 y k_3 claves DES normales de 56 bits. Esta variante se conoce como *EEE*, porque las tres operaciones son de cifrado.

Sin embargo, para simplificar la interoperabilidad con DES, se suele reemplazar el paso intermedio por una operación de descifrado:

$$C = DES(k_3;DES^{-1}(k_2;DES(k_1;M)))$$

Este modo se conoce como EDE y permite representar una operación de cifrado DES con un 3DES EDE y claves $k_1 = k_2 = k_3 =$ k. El descifrado como paso intermedio no afecta a la seguridad del algoritmo (¡siempre que $k_1 \neq k_2 \neq k_3$!). El esquema de funcionamiento de este último modo se representa en la **Figura 13**.

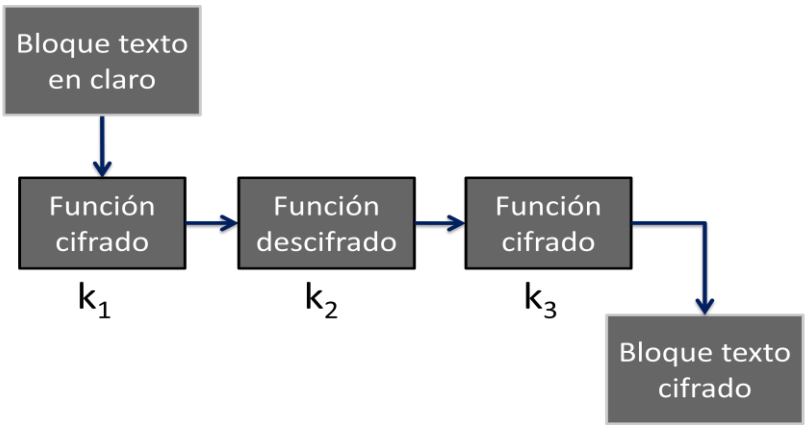


Figura 13. 3DES en modo de funcionamiento EDE

3.6. AES - Advanced Encryption Standard

Como hemos visto, DES ha sido el estándar de cifrado durante algo más de dos décadas, lo que supone una eternidad en el ámbito de la criptografía y la computación. Aprobado como estándar en 1976, finalmente los avances técnicos y teóricos en el campo, junto con las dudas que siempre existieron sobre la participación de la NSA en su diseño, llevaron finalmente a que en 1997 el NIST (*National Institute of Standards and Technology*) no renovara la certificación de DES y convocara un concurso público para diseñar y elegir un nuevo algoritmo que lo sustituyese.

El proceso finalizó en octubre del año 2000, cuando se designó al **algoritmo belga Rijndael** como **nuevo estándar de cifrado del siglo XXI**. Sus principales **características** son las siguientes:

- **No es de tipo Feistel**, sino que utiliza álgebra de cuerpos finitos [concretamente un cuerpo de Galois, GF (2^8)], aunque sí utiliza *cajas S*.
- **Diseñado para ser eficiente en microprocesadores de cualquier ancho de palabra**, desde 8 bits, usados en tarjetas inteligentes o microcontroladores, hasta CPUs de 64 bits.
- **Tamaño de clave variable**, de 128, 192 y 256 bits, o cualquier otro múltiplo de 4 bytes.
- **Tamaño del bloque de texto también variable.** Normalmente 128 bits, aunque puede usarse cualquier múltiplo de 4 bytes.

- **Número de rondas flexible según necesidades del usuario.** Es decir, el algoritmo permite ajustar el número de seguridad global que ofrece, de forma que pueda obtenerse un buen equilibrio entre seguridad y consumo energético o tiempo de computación, por ejemplo.

Estructura de AES

Como hemos visto, un aspecto muy característico de AES es que, a diferencia de DES y otros muchos algoritmos de bloque, **no está diseñado en torno a una red Feistel**. Sus diseñadores definieron cada ronda de forma que hicieran uso de **cuatro funciones invertibles basadas en aritmética modular en cuerpos finitos, o cuerpos de Galois**, con el objetivo de proporcionar una **resistencia máxima frente al criptoanálisis lineal y diferencial** (definiremos estos ataques en la sección 3.8).
Cada ronda está formada, a su vez, por **tres capas**, donde **cada una de las cuales tiene un propósito concreto y definido**:

- La primera capa, una mezcla utilizando una función lineal, proporciona un alto nivel de difusión conforme el texto en claro circula y se transforma a lo largo de las distintas rondas.
- La siguiente capa es no lineal, y consiste esencialmente en la aplicación concurrente de varias *cajas S* con propiedades óptimas de no linealidad.
- La última capa, un *simple XOR*, suma el estado intermedio y la subclave correspondiente a cada ronda.

Los detalles matemáticos no son sencillos (aunque tampoco complicados) y resultan muy interesantes para quien desee entender en profundidad **cómo es un cifrador moderno**.

Recuerda que una **función lineal** es aquella cuya salida puede expresarse como la suma de las salidas de dos de sus argumentos. Formalmente, se dice que f es una función lineal si $f(x+y) = f(x) + f(y)$. También debe cumplir la propiedad de homogeneidad: $f(\alpha x) = \alpha f(x)$.

Seguridad de AES

A día de hoy **no existen ataques exitosos contra este algoritmo y su uso se considera seguro** (siempre que se utilice una clave superior a 128 bits). También se ha comprobado que **es resistente a criptoanálisis tanto lineal como diferencial**.

En efecto, el método más eficiente conocido hasta la fecha para recuperar la clave a partir de un par *texto cifrado–texto claro* es la fuerza bruta o búsqueda exhaustiva, es decir, probar todas y cada una de las posibles claves. En el **epígrafe 3.8** encontrarás información sobre cuánto se tardaría en llevar a cabo este proceso.

Finalmente, es interesante comentar que AES ha hecho que por fin, la NSA haga «las paces» con la criptografía civil, y es que en 2003 AES fue el algoritmo elegido por la NSA para cifrar su propia información clasificada como secreto y alto secreto.

3.7. Cifrado en flujo

El **cifrado en flujo**, en contraposición al cifrado en bloque, procesa el mensaje de entrada bit a bit. La operación de cifrado en sí misma es muy simple, pues normalmente se trata de una simple *función XOR*, entre una secuencia binaria pseudo-aleatoria y el propio mensaje en claro. Dicha secuencia se conoce con el nombre de **secuencia cifrante.**

Esta secuencia se denomina ***pseudoaleatoria*** porque estrictamente hablando, un ordenador determinista clásico no puede generar números físicamente aleatorios sin utilizar un hardware especial. ¿Cómo se solventa este problema entonces? Pues **utilizando algoritmos que generan matemáticamente secuencias** que, si bien no son completamente aleatorias, sí tienen unas propiedades estadísticas excelentes.

Una vez se dispone de dicha secuencia, que obviamente debe ser al menos tan larga como el mensaje a cifrar, simplemente se realiza un XOR entre cada byte de dicha secuencia y del *texto en claro*. Observa la **Figura 14** para entender el esquema general de funcionamiento.

Por ejemplo, si el byte correspondiente de la **secuencia cifrante es `01101100`** y el del ***texto en claro* es `01000001`** (que corresponde a la letra *A* en código ASCII), el byte del criptograma será:

```
   01101100    texto en claro
⊕  01000001    secuencia cifrante
   00101101    criptograma
```

El proceso de descifrado es también extremadamente simple, pues únicamente se debe volver a aplicar la función XOR, esta vez entre el criptograma y la misma secuencia cifrante.

En resumen, **los cifradores de flujo están compuestos esencialmente por**:

- **Un *algoritmo de cifrado* basado en una simple función XOR.** El algoritmo de descifrado es exactamente el mismo, debido a la siguiente propiedad de esta función: `A (texto en claro) ⊕ S (secuencia cifrante) = C (criptograma) -> C ⊕ S = A.`
- **Una *secuencia cifrante* que se obtiene a partir una clave secreta K y un algoritmo generador determinístico.**

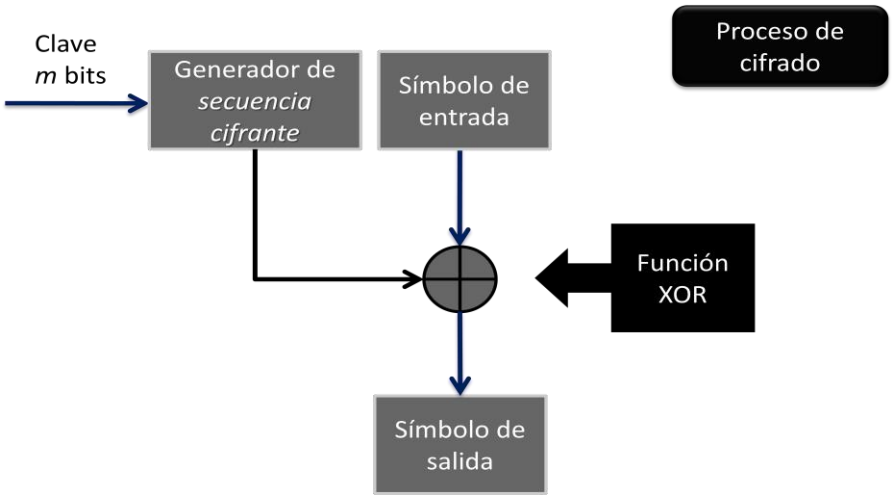


Figura 14. Esquema del proceso de cifrado en flujo

Propiedades de la secuencia cifrante

Como hemos comentado en el punto anterior, para que un cifrador en flujo sea seguro, la **secuencia cifrante** debe cumplir una serie de condiciones:

- **Período**: la clave deberá ser tanto o más larga que el mensaje. En la práctica se utiliza una *semilla K*, de entre 128 y 256 bits, de forma que se generen períodos superiores a 10^{35} elementos.
- **Propiedades estadísticas**: la distribución de bits de unos (1s) y ceros (0s) deberá ser uniforme para que represente a una secuencia pseudoaleatoria.
- **Longitud de la clave**: como se observa en la **Figura 15**, el generador de secuencia es alimentado con una clave secreta K de m bits, que se utiliza a modo de semilla. Al igual que con los cifradores de bloque, esta clave debe ser de longitud suficiente para evitar los ataques por fuerza bruta (con la tecnología actual, un mínimo de 128 bits y recomendable de 256 bits).

Si la secuencia cifrante cumple estas condiciones, un **cifrador en flujo puede ser tan seguro como un cifrador en bloque con la misma longitud de clave**. ¿Cuáles son las ventajas del cifrado en flujo, entonces?

La principal es sin duda, que a igual de condiciones el cifrado en flujo suele ser mucho más rápido que el de bloque. Consulta la **Tabla 2** para hacerte una idea de las diferencias reales de velocidades. El código y la complejidad de las implementaciones es normalmente mucho menor también.

Cifrador	Longitud de clave	Velocidad de cifrado (Mbps)
DES	56	9
3DES	168	3
AES	128	19
RC4	**Variable**	**45**

Tabla 2. Comparativa de velocidad de cifrado entre distintos algoritmos

Sin embargo, no todo son ventajas. En general, utilizar un algoritmo en flujo en un sistema criptográfico real es más complicado, y necesita de unos conocimientos profundos en el área. Por ejemplo, una gran ventaja de los algoritmos de bloque es que **se pueden reutilizar las claves**; es decir, se pueden cifrar dos bloques de datos distintos con la misma clave. Sin embargo, si se hace lo mismo con un cifrador en flujo, esto es, se cifran dos textos en claro con la misma secuencia cifrante, criptoanalizar en ese caso el sistema es prácticamente trivial. Para evitar este inconveniente, los cifrados en flujo **deben incorporar un mecanismo denominada vector de inicialización**, que modifica el estado inicial de generador de secuencia para cada paquete de datos a cifrar. Esto posibilita utilizar los cifrados en flujo en redes de conmutación de paquetes, como Internet.

A pesar de sus inconvenientes, existen cifradores en flujo muy conocidos y utilizados, como **RC4**, **WEP (**que sorprendentemente sigue en uso en muchas redes inalámbricas, a pesar de poderse romper en unas horas) o **E0**, utilizado en el protocolo *Bluetooth*.

RC4

RC4 es un **algoritmo de cifrado en flujo** muy conocido y de amplia difusión, diseñado por Ron Rivest (RSA) en 1987. Su principal característica, como buen cifrador de flujo, **es que es extremadamente rápido**, por lo que en protocolos como *SSL*, *WEP* o *WPA*.

A día de hoy se considera inseguro, pues han sido varias vulnerabilidades que le afectan y no se recomienda su uso en ningún sistema de nuevo diseño.

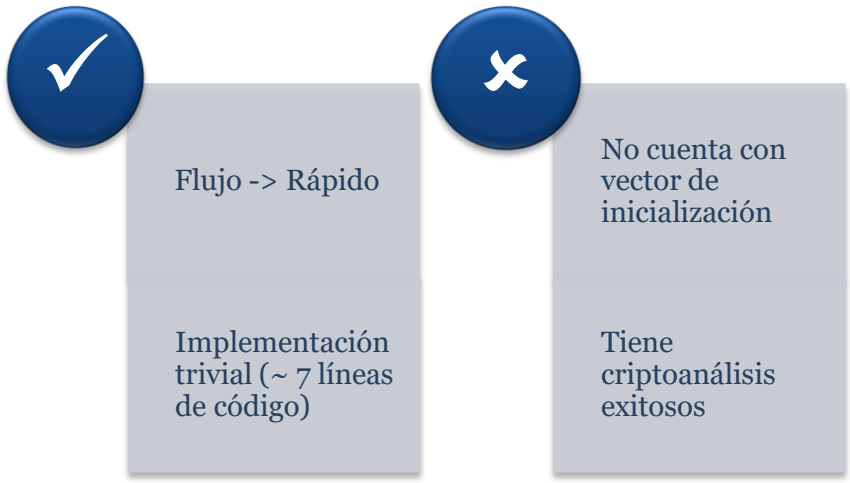


Figura 16. Resumen de características más importantes de RC4

3.8. Criptoanálisis del cifrado simétrico

Como ya se ha explicado anteriormente, ***el criptoanálisis consiste básicamente en comprometer la seguridad de un criptosistema***. Esto se puede conseguir de muchas formas, pero el objetivo final es siempre el mismo: **poder descifrar un mensaje sin conocer la clave de cifrado, o bien poder descubrir dicha clave para poder descifrar todos los mensajes a continuación**.

El **proceso de criptoanálisis** comienza habitualmente con la recolección del máximo número de datos cifrados con una misma clave. El método que se utilice para obtenerlos no afecta al proceso posterior, y pueden variar desde una simple escucha del canal (método pasivo), hasta la generación automática de criptogramas (método activo).

Un ejemplo concreto de este último tipo de métodos es el ***protocolo WEP*** que, como hemos visto, utiliza *RC4* y contiene un fallo de diseño que lo hace vulnerable. El método para atacarlo comienza utilizando un método activo, inyectar tráfico en la red inalámbrica correspondiente, para generar una respuesta del *router* y obtener, así, suficientes pares <texto en claro-criptograma> para iniciar el análisis.

Dependiendo de qué información dispone o puede conseguir el criptoanalista, un ataque de criptoanálisis se clasifica en los siguientes tipos: ***de texto claro conocido o escogido***, o ***de texto cifrado conocido o escogido***.

A continuación veremos algunos tipos en más profundidad.

Criptoanálisis de texto claro escogido

Uno de los tipos más interesantes es el de **texto claro escogido**, en el que el atacante puede seleccionar qué *textos en claro* quiere cifrar para obtener los criptogramas correspondientes. Esta situación es más habitual de lo que parece, suele presentarse cuando se tiene acceso al dispositivo de cifrado (por ejemplo, con las tarjetas SIM de los teléfonos móviles). En este caso, el número de datos necesarios para el análisis desciende drásticamente y, si el sistema es débil o contiene un fallo de diseño, pueden ser suficientes unos cientos de pares <texto claro-criptograma> para recuperar la clave secreta empleada en su cifrado.

Ataques por fuerza bruta

Si todos los métodos anteriores, más elegantes y elaborados, fallan, siempre nos queda el método de la **fuerza bruta**. Como su nombre indica, se trata simplemente de probar de forma exhaustiva todas las posibles claves (a dicho conjunto se le denomina **espacio de claves**) hasta dar con la correcta. Si estás pensando en ponerte a programar para poner a prueba este método te aconsejo que consultes antes la siguiente **Tabla 3** para encontrar una estimación de los tiempos necesarios para que este tipo de ataques tengan éxito.

Durante muchos años se ha considerado dentro de la comunidad criptográfica, que el espacio de claves de los criptosistemas modernos hacían los métodos basados en la fuerza bruta, inviables. Sin embargo, el imparable crecimiento en la potencia de cálculo y la amenaza de los ordenadores cuánticos, ha hecho que longitudes de clave que hace unos años se consideraban seguras (como 64 bits), puedan romperse hoy en día por simple fuerza bruta.

Longitud de la clave	Tiempo necesario para romper la clave
40 bits	2 segundos
48 bits	9 minutos
56 bits	40 horas
64 bits	14 meses
72 bits	305 años
80 bits	78.250 años
96 bits	5.127.160.311 años
112 bits	336.013.578.167.538 años
128 bits	22.020.985.858.787.784.059 años
192 bits	$1.872 * 10^{37}$ años
256 bits	$9.1 \cdot 10^{50}$ años

Tabla 3. Tabla comparativa de tiempos estimados para romper claves simétricas de distintas longitudes por fuerza bruta

Afortunadamente, **la complejidad de atacar por fuerza bruta una clave crece de forma exponencial (y no lineal)** conforme aumenta la longitud de la misma, por lo que añadir un simple bit a la clave duplica el tamaño del espacio de claves. Esto hace que crezca rápidamente el tiempo necesario para «barrer» dicho espacio completamente, hasta el punto de que por ejemplo, para claves simétricas de 256 bits no habría energía suficiente en el Universo para que pudiera los ordenadores necesarios pudieran completar su trabajo. Consulta la **Tabla** para conocer otros valores en función de la longitud de la clave atacada. Si te cuesta comprender los números más grandes, no te preocupes: entender y ser capaces de comparar números tan grandes es imposible para la mente humana. Baste decir que el universo existe sólo desde hace 14.000 millones de años ($1.4 \cdot 10^{10}$). Esto significa que, en la práctica, es imposible romper, por fuerza bruta, una clave simétrica de longitud superior a 128 bits.

Sin embargo, como hemos comentando, cuando se «rompe» un algoritmo de cifrado casi nunca es debido a la fuerza bruta, sino a encontrar fallos en el diseño que no fueron advertidos por sus autores. Dos de más potentes de criptoanálisis que existen actualmente son el **análisis diferencial** y el **análisis lineal**. Sus detalles son complejos y están claramente fuera del ámbito de un curso introductorio a la criptografía, pero sí es conveniente que conozcas sus nombres.

Lo + recomendado

No dejes de leer…

Los detalles del espionaje de la NSA

La NSA ha adquirido gran notoriedad tras conocerse sus actividades de espionaje a través del caso Snowden y las escuchas a líderes aliados. Este informe detalla estas actividades y explora la menos conocida, la colaboración público-privada que hace posible esas actividades.

Accede al documento a través del aula virtual o desde la siguiente dirección web:
http://www.realinstitutoelcano.org/wps/wcm/connect/7366288041c9aefda642ae709b5c3216/ARI41-2013-THIBER-NSA-espionaje-publico-privada-Snowden.pdf?MOD=AJPERES&CACHEID=7366288041c9aefda642ae709b5c3216

No dejes de ver…

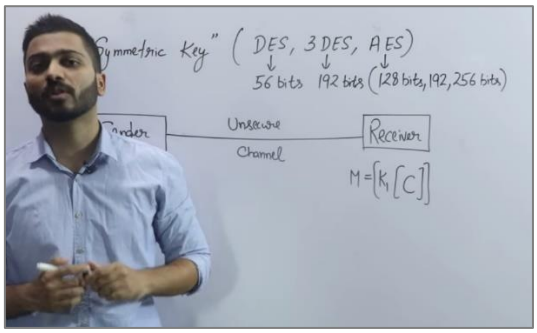
Lección sobre los sistemas de cifrado de clave simétrica

Este interesante documental de la *Intypedia*, una iniciativa de enseñanza de la criptografía liderada por el profesor Ramió de la Universidad Politécnica de Madrid, realiza una interesante introducción a la criptografía de clave simétrica y resume muchos de los conceptos que hemos estudiado en este tema.

Accede al video a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=46Pwz2V-t8Q

Lección sobre los sistemas de cifrado de clave simétrica

Interesante lección acerca de los sistemas de cifrado de clave simétrica.



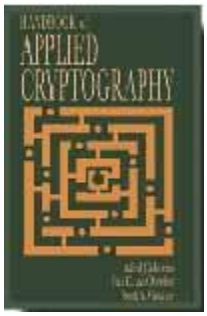
Accede al vídeo a través del aula virtual o desde la siguiente dirección web:
https://www.youtube.com/watch?v=6AmmQiOWoXM&t=1s

+ Información

A fondo

Handbook of applied cryptography

Menezes, A. &, Vanstone, O. (1996). *Handbook of applied cryptography*. CRC Press.



Este un libro clásico, profundo, con el que tarde o temprano se topa todo estudiante de criptografía. Los autores han cedido sus derechos y pueden descargarte gratuitamente una copia de todos sus capítulos en la siguiente dirección:

Accede al documento a través del aula virtual o desde la siguiente dirección web:
http://cacr.uwaterloo.ca/hac/about/toc3.pdf

Estudio sobre la criptografía utilizada en la app móvil Telegram

Este estudio analiza las decisiones de diseño realizadas en la aplicación para móviles *Telegram*, y las critica justificando las razones. Es un buen ejemplo de una aplicación práctica y que muestra la importancia para cualquier ingeniero informático de tener una base mínima de formación criptográfica.

Otro estudio sobre el mismo tema, aunque más profundo y profesional, es el realizado por el INTECO español.

Accede al documento a través del aula virtual o desde la siguiente dirección web:
http://cert.inteco.es/extfrontinteco/img/File/intecocert/EstudiosInformes/INT_telegram_ES.pdf

Recursos externos

OpenSSL

Kit criptográfica.

Accede a la página través del aula virtual o desde la siguiente dirección web:

http://www.openssl.org

Accede al manual a través del aula virtual o desde la siguiente dirección web:

http://www.iit.upcomillas.es/palacios/seguridad/openssl.pdf

Test

1. ¿Cómo sería el texto cifrado de un texto claro puesto en tres filas y ocho columnas? Siendo el texto claro: «LA_ESCITALA_FUE_LA_PRIMERA»

A. LEILFPMAASTAULRECAEAIR_
B. LEILFPMAASTAULRE_CEAIR_
C. LEILFPMAASTAULRECAEAIR
D. Ninguna de las anteriores.

2. En los algoritmos de cifrado simétrico:

A. La clave de cifrado debe permanecer secreta, y es diferente a la de descifrado
B. La clave de cifrado y descifrado son iguales, pero no es necesario que permanezcan secretas
C. La clave de cifrado y descifrado son la mismas, y deben permanecer secretas para mantener la seguridad del sistema
D. Las claves son iguales, pero sólo la del emisor del mensaje debe permanecer secreta

3. Uno de los grandes problemas del cifrado simétrico es:

A. La gestión y distribución de claves
B. La baja velocidad de cifrado de los algoritmos de bloque frente a los de flujo
C. Es necesario que las claves aumenten mucho de tamaño para evitar los ataques de fuerza bruta
D. Debido a la gran capacidad de cómputo que exige, no puede funcionar en dispositivos pequeños como microcontroladores

4. En general, todo algoritmo de cifrado, simétrico o asimétrico, utiliza dos mecanismos básicos, confusión y difusión. Estos conceptos consisten en:

A. La confusión trata de repartir la influencia del texto en claro a lo largo de todo el criptograma
B. La confusión trata de ocultar la relación estadística entre texto en claro, clave de cifrado y criptograma. Suele conseguirse a través de cajas de sustitución
C. La confusión trata de ocultar la relación estadística entre texto en claro, clave de cifrado y criptograma. Suele conseguirse a través de permutaciones
D. La confusión trata de ocultar la relación estadística entre texto en claro, clave de cifrado y criptograma. Suele conseguirse a través de estructuras denominadas redes de Feistel

5. AES y DES son dos conocidos algoritmos de cifrado simétrico, pero…

A. AES es inseguro y no debería ser utilizado. DES es el nuevo estándar
B. DES es inseguro y no debería ser utilizado. AES es el nuevo estándar
C. AES es un algoritmo de cifrado en flujo, por lo que es más rápido que DES
D. DES es un algoritmo de cifrado en flujo, por lo que es más rápido que AES

6. Los modos de operación de los cifradores simétricos de bloque recomendados hoy en día son:

A. ECB y CTR
B. CBC y CTR
C. CFB y CTR
D. ECB y CBC

7. El modo de operación ECB puede resultar inseguro en determinadas situaciones. ¿Por qué?

A. No existe encadenamiento entre bloques, por lo que no oculta adecuadamente las relaciones existentes en los datos
B. Existe encadenamiento de bloques, pero un error de transmisión en uno de ellos se propagaría a todos los demás
C. No existe encadenamiento de bloques, por lo que un único error en la transmisión haría que no se pudiera descifrar el resto del mensaje
D. Su diseño es complejo, por lo que son necesarios conocimientos profundos; esto puede provocar errores en su implementación

8. El modo de operación más rápida es:

A. ECB
B. CBC
C. CFB
D. CTR

9. El ataque por fuerza bruta consiste en tratar de:

A. Encontrar fallos en el diseño del cifrador probando todas las posibles debilidades existentes

B. Encontrar la clave de cifrado probando todas las posibles combinaciones

C. Aplicar los principios del criptoanálisis diferencial para encontrar un fallo en el diseño

D. Aplicar los principios del texto claro escogido, para encontrar fallos en el diseño del cifrador

10. La longitud mínima que se considera segura hoy en día para una clave simétrica es:

A. 64 bits

B. 96 bits

C. 128 bits

D. 256 bits