CRIPTOGRAFIA DE CLAVE SIMÉTRICA: AES

Daniel Penazzi
Facultad de Matemática, Astronomía y Física
Universidad Nacional de Córdoba, Córdoba, Argentina

Notas de curso para las Jornadas De Criptografía Y Códigos Autocorrectores

Mar del Plata, Noviembre 2006

1. Introducción

En estas notas breves, intentaremos dar una idea general de la criptografía de clave simétrica, en especial criptosistemas de bloque, concentrandonos particularmente en el estandard norteamericano, el AES (Advanced Encryption Standard).

Si bien la criptografía de clave pública/privada es muy interesante y revolucionó la criptografía en los 1970, no es adecuada para encriptamiento de archivos grandes. El núcleo de las aplicaciones criptográficas han sido siempre los algoritmos de clave simétrica, en donde ambas partes deben conocer la clave para poder comunicarse. En particular los algoritmos de bloque siempre han sido muy populares. Quizás el más famoso de ellos fue el DES (Data Encryption Standard), que fue el estandard norteamericano (y de facto mundial) desde los 1970s. A fines de los 1990s, sin embargo, estaba claro que estaba llegando al fin de su vida útil, y el gobierno norteamericano lanzó un llamado internacional para un nuevo estandard. De entre 15 presentaciones, resulto elegido un algoritmo, Rijndael, que se convirtió en el AES (Advance Encryption Standard).

Las razones de la elección de Rijndael como AES son diversas. Entre ellas estuvo el hecho de su estructura simple y matemática, que no dejaba lugar a ninguna duda de que pudiera haber una "trapdoor" oculta; su gran rapidez y versatilidad de implementación, tanto en procesadores de 8 bits, como de 32 bits, asi como en harwdare; y su demostrable seguridad contra los dos ataques más importantes de la década del 1990: el criptoanálisis diferencial y el criptoanálisis lineal.

Para ejemplificar qué se espera de un cifer moderno daremos una breve introducción a los sistemas criptográficos clásicos , y luego describiremos el AES, los ataques diferenciales y lineales, y cómo Rijndael usa elementos de la teoría de códigos y cuerpos finitos para proveerse de inmunidad contra estos ataques.

Finalmente, describiremos otros ataques que tratan de aprovechar la rica estructura matemática de Rijndael.

El conjunto $\{0, 1, ..., n-1\}$ *con la estructura de anillo dada por la suma y el producto módulo* n*, será denotado como* $\mathbb{Z}_n$ *(como no usaremos los números* n*-ádicos, no hay problema de confusión en la notación). El (único módulo isomorfismo) cuerpo finito de* 2^n *elementos será denotado por* $GF(2^n)$*. La suma en* $GF(2^n)$ *será denotada como* $+$*, al igual que la suma de enteros, salvo cuando esto pueda dar lugar a confusión, en cuyo caso se denotará la suma de* $GF(2^n)$ *como* $\oplus$*. El espacio vectorial* $\mathbb{Z}_2^n$ *será identificado con el cuerpo* $GF(2^n)$ *cuando sea conveniente.*

2. Criptografía Antigua

Si una persona A (Alicia) desea mandar un mensaje M a una persona B (Bob) sin que una tercera persona E (Eva) lo pueda entender, pero a traves de un medio en el cual se sospecha que Eva puede escuchar, entonces una parte de la criptografía es la que se encarga de disfrazar el mensaje P (llamado el plaintext*) en*

otro mensaje C, que llamaremos mensaje cifrado o ciphertext*, de tal forma que aunque Eva intercepte C, no pueda reconstruir P, al menos fácilmente, pero que Bob si pueda, en general por poseer una clave secreta que le permita hacerlo.*

Un ataque contra un criptosistema (o cifer, para usar una palabra más corta) es un algoritmo que le permita a Eva reconstruir P a partir de C.

Hay varias clases de ataques, dependiendo de cuanta capacidad tenga Eva:

1) En el ataque más básico ("known ciphertext"), Eva solo es capaz de interceptar los ciphertexts, y quiere reconstruir los plaintexts.

Durante casi toda la historia, los cifers eran jusgados de acuerdo solo con el criterio de que tan capaz era de resistir este tipo de ataques, sin embargo, hay otros ataques:

2) Puede pasar que Eva no solo es capaz de interceptar los ciphertexts sino que tuvo acceso a algunos plaintexts, y lo que quiere es obviamente descifrar aquellos ciphertexts para los cuales no tuvo acceso al plaintext. (ataque de tipo "known plaintext"). (por ejemplo, Eva puede saber que Alicia siempre comienza sus mensajes a Bob de la misma forma, y por lo tanto saber que siginifican las primeras partes del mensaje).

Muchos criptosistemas son muy buenos contra ataques de tipo known ciphertext, pero fallan completamente contra ataques de tipo known plaintext.

3) Eva incluso puede ser capaz de elegir que es lo que Alicia va a encriptar. (ataques de "chosen plaintext"). Por ejemplo, puede dar información a Alicia que Eva esta segura que le retransmitirá a Bob. Un ejemplo de este tipo de ataque le permitio a los estadounidenses averiguar que los Japoneses atacarian Midway.

Veamos algunos cifers historicos:

Uno de los primeros criptosistemas conocidos es el **scytalus***, usado por los griegos pero tambien por otras culturas: consistia en un bastón largo, alrededor del cual se enrollaba una tira para escribir, como una venda. El mensaje a enviar se escribia verticalmente, y al desenrollarse la tira,las letras del mensaje quedaban transpuestas de lugar, siendo imposibles de entender. Un mensajero llevaba esta tira hasta el destinatario, el cual tenia un scytalus igual, y enrollando la tira podia leer el mensaje. Obviamente, solo personas autorizadas como generales o gobernantes poseian una copia del scytalus. Un remanente del scytalus puede verse en la costumbre moderna del bastón de mando.*

Este criptosistema es lo que se llama un **cifer de transposición***: los caracteres individuales del mensaje no son cambiados, sino solo transpuestos. Para la epoca era realmente bueno, pues Eva no podia leer el mensaje sin el scytalus, pero Bob, que tenia una copia, si.*

A lo largo de la historia cifers de transposición cada más complicados fueron usados, pero en general, por si solos no son suficientemente buenos y pueden ser quebrados si se reune una cantidad suficiente de textos.

Otro cifer famoso de la antiguedad es el **Cesar Shift***: En este caso, en vez de transponer las letras del mensaje, Cesar* **reemplazaba** *cada letra por la letra que estaba a distancia 3 hacia la derecha en el alfabeto, con rotación al final, es decir, (en nuestro alfabeto) seria reemplazar la letra A por D, la B por E,..., la X por A, la Y por B y la Z por C. Por ejemplo, VINI VIDI VINCI seria YLPL YLGL YLPFL. Matematicamente hablando, si asignamos el valor A=1, B=2,..., Y=26, Z=0, seria reemplazar la variable x por $x+3 \bmod 27$. Esto no es un cifer de transposición sino lo que se llama un* **cifer de substitución***: el mensaje se parte en*

bloques adecuados (en este caso, bloques de una letra), y cada bloque se substituye por otro. Por ello tambien hablamos de cifers de bloque.

Si bien para la epoca parecia espectacular, es en realidad muy débil. Las debilidades principales son varias:

La primera debilidad es que es un sistema cuya seguridad depende de que no se conozca el sistema. Esto viola una de las reglas cardinales de la criptografía, que es suponer que el adversario tendrá conocimiento, ya sea por soborno, ingenieria reversa o lo que sea, del sistema. Un sistema que se base en seguridad por obscuridad no es seguro. Por ejemplo, si Cesar usaba su sistema para hablar con Pompeyo y con Marco Antonio, y luego Pompeyo se volvia su enemigo, cuando quería hablar con Marco Antonio sin que se enterara Pompeyo, no podía.

En general, los sistemas criptográficos deben depender, ademas del sistema en si, de una CLAVE, que puede ser variada periódicamente, o entre individuos distintos. Asi, si el sistema anterior hubiese tenido una clave, Cesar solo tendria que cambiar la clave para hablar con Marco Antonio.

Una posibilidad seria entonces hacer que en vez de ser siempre una rotación de tres unidades, la rotación dependa de una clave, i.e. $x \mapsto x + k \bmod 27$*, donde* k *es la clave.*

Esto sin embargo no sería muy fuerte, pues solo hay 27 claves posibles, asi que en seguida podria recorrerse todo el espacio de claves (desencriptando hasta obtener algo que tenga sentido) hasta encontrar la correcta. Entonces, no solo debemos tener una clave, sino que el espacio de claves posibles debe ser lo suficientemente grande como para que no sea factible efectuar un ataque de fuerza bruta.

Aun sin contar esto, esta el problema de que la tranformación es lineal, sobre un espacio vectorial de dimensión uno, asi que solo se necesita averiguar el encriptamiento de UNA letra para obtener el encriptamiento de todas las demas. Es decir, este sistema es muy vulnerable a un ataque de tipo known plaintext.

Una posibilidad para corregir estos problemas sería tomar una permutación cualquiera de $\{0, 1, ..., 26\}$*, y no una necesariamente lineal. La clave seria la tranposición.*

Como hay $27! = 10.888.869.450.418.352.160.768.000.000$ *posibilidades, el espacio de claves es ciertamente enorme, y ademas, aún averiguando el valor de una letra no se obtendría el valor de las demas. Igualmente, este sistema seria muy débil contra un ataque de known plaintext, pues si se interceptan suficientes ciphertexts tales que se conozcan sus plaintexts, se averiguará la clave. (es decir, bastaria conocer un mensaje suficientemente largo como para que aparezcan todas o casi todas las letras en el y su ciphertext para poder leer de ahi en más cualquier mensaje).*

Incluso contra un ataque solo de known ciphertext, este sistema es débil. El lenguaje humano tiene ciertas características que permiten atacar este sistema. Por ejemplo, la frecuencia de las letras no es la misma. Entonces, juntando suficiente información, se puede hacer un análisis de frecuencia de las letras del ciphertext. Si el idioma es castellano, las letras más frecuentes son la E y la A, asi que se puede suponer que la letra más frecuente será una de ellas, y empezar a adivinar el mensaje, llenando los huecos. Actualmente un sistema asi es quebrado en pocos segundos en una computadora, si se tienen suficientes ciphertexts.

Esto refleja otro problema: no solo el espacio de claves debe ser grande, sino tambien el tamaño del bloque debe ser grande, para evitar un ataque de análisis de frecuencia.

Para resolver este problema, en el siglo XIX, Inglaterra adoptó un sistema, llamado PLAYFAIR, que era un cifer de bloque donde el bloque consistia de dos letras. En el ejemplo del Cesar, la letra I siempre era

encriptada como L y la V como Y. En un cifer cuyo bloque fuesen dos letras, la I no siempre se encriptaria como L, pues dependeria de la letra que tuviera al lado, y por lo tanto la frecuencia natural de las letras se veria obscurecida. De todos modos, una vez que se inventaron estos sistemas, se empezaron a usar análisis de frecuencias de pares de letras en vez de letras individuales. Una posibilidad seria tratar con bloques más grandes, como de triples de letras o cuadruples. El problema estaba en encontrar un sistema que fuese fácil de usar, sobretodo antes del advenimiento de las computadoras

Un sistema muy conocido, que se creyó indescifrable por mucho tiempo era el sistema **Vigenére***: en este caso el tamaño del bloque podia ser tan grande como se quisiera: si se queria un bloque de, digamos 5 letras, se tomaba una clave de 5 letras, cada letra correspondiendo con un número que seria la clave de rotación en un Cesar Shift. Entonces, a cada letra del bloque se le aplicaba una rotación distinta, repitiendose cuando se llegará al final del bloque. Por ejemplo, se podia tomar un bloque de 5 letras, y hacer que la primera letra se rotara 3 unidades, la segunda 26, la tercera 4, la cuarta 25 unidades y la quinta una unidad. Entonces, VINI VIDI VINCI quedaria YHQG WLCM TJPBM Observar que las tres V se encriptan como Y, W y T, mientras que las seis I se encriptan como H,G,L,M,J y otra vez M. El sistema se creia impregnable pues parece imposible hacer un análisis de frecuencia, más aún si la longitud del bloque dependia de la clave, como era usual. Pero se descubrió que se pueden hacer análisis que revelan la longitud del bloque (básicamente, se asume que la longitud del bloque es n (hay solo una cantidad razonable de n's que se usaban, en general nunca se usaba $n > 20$) y se parte el mensaje en subbloques distintos, tomando n mensajes distintos, siendo el mensaje i el formado por las letras en las posiciones congruentes a i módulo n. Si el n es el correcto y se hace un análisis de frecuencia de los submensajes, la frecuencia debe ser la usual, corrida. Si no, no se obtiene la frecuencia usual sino una curva de frecuencia distinta. Una vez obtenido el correcto n, se analizan los submensajes por separado. Ciertamente se necesita un mensaje más largo, pero con sufciente longitud se puede encontrar la clave. Esto es porque en realidad Vigenére no es un verdadero cifer de bloque: cada parte del bloque debería depender de todas las otras. En este caso, esto no se satisface, pues la letra segunda no depende de la primera, asi que en realidad no tenemos realmente un bloque de 5 letras, sino 5 bloques de una letra, cada uno con clave distinta, los cuales se quiebran por separado.*

Un verdadero cifer de bloque era el criptosistema de **Hill***, en el cual el tamaño del bloque podia ser arbitrariamente grande, pero un tamaño de 4 o 5 letras era razonable. El algoritmo de Hill básicamente tomaba un bloque de, digamos, 5 letras, y luego miraba el bloque como un vector (columna) en $(\mathbb{Z}_{27})^5$ y lo multiplicaba por una matriz 5×5 que fuese inversible sobre $\mathbb{Z}_{27}$. (la clave era la matriz). Este sistema hace que cada letra del bloque dependa de todas las otras, por lo tanto no se puede hacer un análisis por separado, y realmente hay que hacer un análisis de frecuencia de bloques de 5 letras, lo cual no era practicable. Ademas, la clave es de un tamaño suficientemente grande. Es ademas fácil de implementar, tanto a mano como en la computadoras (donde se podrian tomar tamaños aun más grandes) ¿Porque entonces no se usa este sistema? La razon es que, aunque este sistema resiste muy bien un ataque de tipo known ciphertext, es completamente vulnerable a un ataque de tipo known plaintext: al ser lineal, puede ser quebrado juntando pocos mensajes. (en este caso, bastarian 5 plaintexts linealmente independientes para encontrar la matriz que encripta esos plaintexts en los ciphertexts.)*

Resumiendo, los algoritmos de bloque modernos tratan de tener lo siguiente:

1) Una clave de longitud lo suficientemente grande para evitar ataques de fuerza bruta. (al momento,

128 bits se consideran OK)

2) Un bloque lo suficientemente grande para prevenir ataques de frecuencia (otra vez, 128 bits parece OK por el momento)

3) Un algoritmo que NO sea lineal, para prevenir ataques triviales de known plaintext.

4) Un algoritmo que haga que cada bit del ciphertext dependa de todos los bits del plaintext y de la clave.

5) Un algoritmo que sea razonablemente rápido y fácil de implementar.

El problema es la compatibilidad entre estas condiciones, especialmente de las primeras cuatro con la quinta. Por ejemplo, lo ideal seria un cifer que fuese una permutación aleatoria de los bloques de 128 bits, i.e., una substitución gigantesca, pero ¿como implementar esto eficientemente? No se puede guardar en una tabla todas las 2^{128} *posibilidades.*

Shannon a fines de los '40 introdujo los conceptos de confusión y difusión: Puesto que no es posible substituir todo el bloque, uno podria substituir subbloques más pequeños (la "confusión"), pero luego esta confusión local debe "difundirse" de alguna forma a las otras partes del bloque.

Los alemanes en la primera guerra mundial en realidad fueron precursores de esta idea: el sistema ADFGXV usaba primero una substitución de cada letra por un par de letras, tomadas sólo del alfabeto ADFGXV y luego realizaba una permutación de las letras resultantes, con lo cual mezclaba un poco las substituciones individuales entre si. Sin embargo, la mezcla no era muy buena y de hecho el sistema fue quebrado en la misma guerra en cuestión de semanas.

Pero esta idea de mezclar substituciónes con permutaciones podria haber hecho que a alguien se le ocurriera el siguiente algoritmo: tomar por ejemplo un bloque de 5 letras, aplicar una substitución individual a cada letra (pero que sea no lineal, en vez del shift), y luego multiplicar todo el bloque por una matriz de tipo Hill. La matriz se encarga de que todas las letras dependan de las otras, volviendo imposible la partición del bloque en subbloques, y las substituciónes no lineales previenen montar un ataque de ecuaciones lineales.

Si bien que yo sepa nadie implementó ese sistema antes del advenimiento de las computadoras, los mejores sistemas modernos son básicamente este sistema. La diferencia es que en vez de hacer esto una sola vez, se repite esto varias veces. Es decir, los cifers modernos tienen rondas. *Aunque no siempre (por ejemplo los cifers RC5 y RC6 usan otro modelo, que no veremos), en cada ronda hay típicamente tres pasos:*

1) Un paso de mezcla con una clave parcial (clave de ronda)

2) Un paso de substitución, en donde subbloques del bloque se substituyen por otros por medio de transformaciones no lineales (llamadas "S-boxes" por "Substitution Boxes")

3) Un paso de difusión, que es típicamente una transformación lineal de todo el bloque que asegure que las substituciónes locales se difundan al resto del bloque.

Un cifer asi se llama un Substitution-Linear Network (SLN). El AES, que estudiaremos en este curso es exactamente un cifer de este tipo.

El antecesor del AES, el DES, que fue el estandar norteamericano y mundial desde 1976 hasta más o menos el 2000, tenia una estrucutra ligeramente distinta: era un cifer de tipo "Feistel", que es una estructura donde en cada ronda solo se procesan la mitad de los bits. Aun asi, en cada ronda tambien habia un proceso de substitución local mezclado con difusiones lineales. (ver apéndice para una descripción de DES).

La sigla DES proviene de Data Encryption Standard. En los '70s, los EEUU decidieron que necesitaban un estandard criptográfico seguro que se pudiera usar en los bancos y otros lugares civiles. Luego de un

concurso en el cual solo se presentó la IBM, y luego de pasar por la NSA (National Security Agency), se propuso el DES. El DES fue finalmente quebrado en los '90s, usando un ataque de fuerza bruta. La razón es que el tamaño de la clave (56 bits) se consideraba suficientemente grande en los '70s, pero era insuficiente en los '90s. (en realidad, ya en los '70s se criticó el tamaño de la clave: el proyecto original de IBM tenia una clave de 128 bits, pero la NSA la redujo a 56. Se supone que es porque la NSA tenia ya en los '70s computadoras capaces de quebrar una clave de 56 bits pero no una de 128. (Aun peor: durante mucho tiempo, DES para exportación debia tener una clave de solo 40 bits).

Lo que nos interesa conocer por ahora de DES es que usaba unos S-boxes que eran muy misteriosos, pues nadie sabia de donde habian salido estas transformaciones misteriosas. Se sospechaba que podia haber una puerta secreta implantada por la NSA en los S-boxes.

Varios otros cifers fueron inventados en los '70s y '80s, con otros S-boxes y otra estructuras. Finalmente, en 1990, se inventó el ataque de criptoanálisis diferencial (DC) el cual barrió con numerosos cifers. Sin embargo, cuando se lo intentó applicar a DES, se vio que sus S-boxes eran sorprendemente resistentes al DC. ¿Como puede ser, si fueron inventados años antes del ataque? Al parecer, los inventores del DES tambien descubrieron el DC, pero lo mantuvieron secreto por razones de seguridad.Como trabajaban para IBM, no tenian problemas de acceso a tiempo de computadoras, y se pasaron dias enteros creando S-boxes al azar hasta encontrar S-boxes que cumplieran ciertas propiedades que garantizaban resistencia contra este ataque.

En 1992, Matsui inventó el criptoanálisis lineal. (LC). Los S-boxes de DES no son particularmente resistentes a este ataque, y de hecho, hasta que fue quebrado por fuerza bruta, el LC era el mejor ataque contra DES (aunque no muy realista en la practica, necesitandose mas o menos 2^{51} textos cifrados para obtener la clave).

A mediados de los '90s estaba claro que DES ya estaba llegando al fin de su vida útil. Esto quedó demostrado luego de que una serie de ataques de fuerza bruta (probando con todas las claves posibles hasta obtener un texto que tuviera sentido) fueron llevados a cabo en un tiempo razonable. (entre uno y dos dias).

Aun teniendo en cuenta la vulnerabilidad teórica de DES respecto del LC, el mayor problema era la longitud de la clave, asi que se intentaron algunos parches para solucionar esto. Uno de ellos fue el triple DES, que consiste en encriptar el mensaje tres veces, con tres claves distintas. (algunos usaban dos claves distintas solamente, igualando la ultima con la primera). Pero otro problema con DES era su rapidez. Si bien el algoritmo era muy rápido en los '70s, fue diseñado teniendo en cuenta la tecnologia de la epoca. En particuar, no habia PCs, y DOS, Windows, etc, todavia estaban en el futuro.

En particular, DES fue diseñado para ser rápido EN HARDWARE. Por ejemplo, para la difusión usaba permutaciones de bits individuales, que no son problema en Hardware, pero son muy dificiles y lentas de implementar en software que debe correr en maquinas orientadas a 8 bits o, más modernamente, 32 bits. Ademas, dado el poder de las maquinas de 32 bits de los '90s, estaba claro que se deberia poder diseñar cifers que aprovecharan más las capacidades de estas maquinas, y que ademas pudieran aprovechar toda la nueva teoría que se habia ido desarrollando. Por ello, el gobierno de los EEUU decidio llamar a un concurso internacional para seleccionar un nuevo estandard, que seria llamado AES (Advance Encryption Standard). Se presentaron 15 candidatos, de diversas partes del mundo. Luego de una primera ronda, en donde se detectaron fallas en algunos de ellos, o bien a otros se los considero no demasiado rápidos, cinco

candidatos fueron elegidos para una ronda final, y finalmente uno de ellos fue elegido como el nuevo estandar. El candidato era el cifer Rijndael, llamado asi por los nombres de sus creadores, Vincent Rijmen y Joan Daemen, dos profesores belgas. Es de aclarar que Rijndael no ganó por una mayoria absoluta, sino simple (saco el 42% de los votos). El segundo candidato más votado, Serpent, de los creadores del DC, es para la mayoria de la gente, más seguro que Rijndael, y más rápido en Hardware, pero considerablemnte más lento en Software, y fue por eso que no fue elegido. Rijndael era claramente uno de los tres más seguros, y era uno de los más rápidos tanto en software (tanto de 32 bits como de 8 bits) como en hardware. Desde el punto de vista de los matemáticos, es afortunado que haya salido elegido, pues tiene una estructura matemática muy rica.

3. AES

Rijndael tiene la estructura típica de SLN que habiamos hablado antes: 10 rondas, en cada una de las cuales hay un paso de substitución a traves de S-boxes, un paso de difusión por medio de una transformación lineal, y un paso de mezcla con la clave. Antes de las 10 rondas, se hace un mix con otra clave (esto se suele llamar "whitening").

Rijndael en realidad es el nombre de tres cifers similares pero distintos en cuanto al tamaño del bloque. Analizaremos solo la versión del AES, que por los requerimientos pedidos debia ser de un bloque de 128 bits.

La mezcla con la clave se realiza al final de la ronda, y es simplemente una suma del bloque con la clave de ronda, usando la suma de $\mathbb{Z}_2^{128}$. ("XOR" por "eXclusive OR")

En cuanto al paso de substitución, que es lo que se realiza al principio de la ronda, el bloque se divide en 16 subloques de 8 bits cada uno. A cada uno de estos subbloques se le aplica un S-box de 8 bits en 8 bits. Construir un S-box de este tamaño al azar que tenga buenas probabilidades diferenciales y lineales, es una tarea casi imposible. (los autores de Rijndael trataron con más de un millón de posibilidades y no encontraron ninguno). Por ello, los autores de Rijndael apelaron a la matemática para construir deterministicamente un S-box que fuese resistente a estos ataques. (Veremos más adelante en detalle esto).

Esto, en cuanto al paso de substitución. Veamos como es el paso de difusión:

En este paso se le aplica una transformación lineal (sobre $GF(2^8)^{16}$) al bloque entero. La transformación lineal es la composición de dos transformaciones lineales más fáciles de describir, salvo en la ultima ronda, en donde solo se aplica una de ellas. (esto se hace para que la implemantación del desencriptado sea similar a la del encriptamiento).

Estas dos transformaciones lineales tiene el nombre sugestivo de ShiftRow y MixColumn. Esto es porque el bloque de 128 bits, se lo mira, como dijimos, como un bloque de 16 subloques de 8 bits (1 byte) cada uno. Estos 16 bytes se piensan en una matriz 4×4. En RowShift, lo que se hace es simplemente rotar las filas de esta matriz: la primera fila se deja fija, la segunda se rota un byte a la izquierda, la tercera dos bytes, y la cuarta tres bytes:

ShiftRow:

$$\begin{bmatrix} A & B & C & D \\ E & F & G & H \\ I & J & K & L \\ M & N & O & P \end{bmatrix} \mapsto \begin{bmatrix} A & B & C & D \\ F & G & H & E \\ K & L & I & J \\ P & M & N & O \end{bmatrix}$$

En MixColumn (que no se efectúa en la ultima ronda), se le aplica una tranformación lineal sobre $GF(2^8)^4$ a cada columna de esta matriz. La transformación lineal es multiplicación por una matriz M 4×4,

invertible sobre $GF(2^8)^4$. (por lo que podemos ver a MixColumn como la multiplicación por M a izquierda de toda la matriz que representa el bloque).

Esta matriz fue elegida cuidadosamente, y la veremos más adelante.

En resumen, Rijndael es:

KeyMixing

Ronda 1: *S-box*, *ShiftRow*, *MixColumn*, *KeyMixing*

Ronda 2: *S-box*, *ShiftRow*, *MixColumn*, *KeyMixing*

⋮

Ronda 9: *S-box*, *ShiftRow*, *MixColumn*, *KeyMixing*

Ronda 10: *S-box*, *ShiftRow*, *KeyMixing*

Por supuesto, tambien se lo podría pensar de la siguiente forma, que es más adecuada para ciertos analisis:

Ronda 1: *KeyMixing*, *S-box*, *ShiftRow*, *MixColumn*

Ronda 2: *KeyMixing*, *S-box*, *ShiftRow*, *MixColumn*

⋮

Ronda 9: *KeyMixing*, *S-box*, *ShiftRow*, *MixColumn*

Ronda 10: *KeyMixing*, *S-box*, *ShiftRow*

KeyMixing

Desencriptamiento *Para desencriptar simplemente hay que correr todo el cifer a la inversa. Esto es posible porque cada componente de cada ronda es inversible.*

La estructura esta ingeniosamente diseñada para tomar ventaja de las características de los proccesadores tanto de 32 bits como de 8 bits. Ademas, es fácilmetne implementable en una variedad de dispositivos hardware. Esta versatilidad de Rijndael fue una de las razones por las cuales fue elegido el AES. (ver apéndice para una descripción de la implementación eficiente en software de 32 bits)

Para terminar de describir las componentes de Rijndael, debemos describir el S-box y la matriz que se usa en el paso MixColumn. Primero veremos el S-box.

La función del S-box es proveer no linealidad, asi pues, queremos que sea una función "altamante no lineal". Pero que se entiende por "altamente no lineal"? Hay (al menos) dos "medidas"de no linealidad de uso generalizado, pero hay que aclarar antes que cuando se dice "no lineal"en realidad se entiende "no afin", solo que la frase "función altamente no afin"parece que no le gusta a nadie.

1) La primer medida de no linealidad esta altamente ligada a la "derivada booleana"de la función. Esto da origen al concepto de funciones diferencialmente δ-uniformes. Las funciones lineales tienen un δ muy grande, y entonces se mide la no linealidad por cuan chico se puede volver δ.

2) Otra medida natural que se puede usar es simplemente que tan "lejos"esta la función del conjunto de funciones afines, de acuerdo con alguna distancia.

La medida 1) esta muy ligada a la resistencia del S-box contra el ataque de Criptoanalisis Diferencial, mientras que la 2) esta relacionada con su resistencia al ataque de Cripotanalisis Lineal. Veamos entonces primero una vista rápida del ataque de Criptoanálisis Diferencial.

4. Criptoanálisis Diferencial

En el Criptoanálisis Diferencial (DC, por sus siglas en ingles) se tratan de explotar las desviaciones estadísticas entre las **diferencias** *de los datos de entrada y los datos de salida de un S-box. Es un ataque de tipo* chosen plaintext.

Supongamos que tenemos un cifer en donde primero mesclamos con la clave, usando la suma (el XOR) de $\mathbb{Z}_2^n$, luego pasamos por S-boxes y finalmente hacemos una tranformación lineal. Supongamos que tenemos plaintexts P y P' y sea $\alpha = P + P'$. Entonces, después de aplicar la clave, tomemos $X = P + k$ y $X' = P' + k$ ($+$ es la suma en $\mathbb{Z}_2^n$), con lo cual:

$$X + X' = (P + k) + (P' + k) = P + P'$$

de donde se deduce que la diferencia es la misma después de aplicar la clave que antes de hacerlo.

Después de pasar por el S-box, sea $Y = S(X)$ y $Y' = S(X')$. Como el S-box no es una función lineal no vale en general que $Y + Y' = S(X + X')$. Ademas, al no conocerse la clave, no se puede calcular X, aun conociendo P. Sin embargo, con el conocimiento de la **diferencia** *de los valores que ingresan al S-box en cada uno de los casos, y analizando las propiedades del S-box, se puede* estimar *la diferencia de salida más probable, creando una tabla con las* probabilidades *de que si entran dos textos con diferencia α, se obtengan dos textos con diferencia β. Como dijimos arriba, la clave es irrelevante cuando se miran las diferencias, por eso se puede montar este ataque sin conocer las claves.*

Si esta probabilidad es muy alta, se pueden iniciar ataques en donde se encriptan muchos textos con diferencias prearregladas para luego, estimando la diferencia de salida, y rastreandola a lo largo de varias rondas (cada una de las cuales agregará su propia probabilidad, por lo que habrá que multiplicarlas), poder deducir información probables sobre los datos de **entrada** *a la ultima ronda, y de esa forma, comparandola con los datos de salida, deducir información sobre la clave de la ultima ronda, con lo cual se pueden averiguar varios bits de la misma. Una vez obtenidos estos, se calculan los restantes por fuerza bruta.*

Un ataque DC busca entonces la entrada no trivial con mayor valor, para tener la probabilidad más favorable. Por lo tanto, es necesario usar un S-box donde la mayor entrada no trivial sea lo más chica posible.

Las definiciones abajo suelen ser un poco más generales, pero nos concentraremos en el caso que nos interesa.

4.1 Definición :

Dada una función $S : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$, y un elemento $\alpha \in \mathbb{Z}_2^n$ la "derivada booleana" de S se define como $\Delta_\alpha S(x) = S(x+\alpha) + S(x)$.

El nombre "derivada" no es muy correcto, pues no hay ningun tipo de limite involucrado. Es simplemente una diferencia, pero supongo que "derivada" suena mejor y por eso se la llama asi. La notación $D_\alpha S(x)$ tambien es muy común para designar la derivada.

4.2 Definición :

Dados vectores fijos $\alpha, \beta \in \mathbb{Z}_2^n$, la probabilidad diferencial $DP(\alpha, \beta)$ esta definida como:

$$DP(\alpha,\beta) = P(\Delta_\alpha S(x) = \beta) = \frac{\#\{x \in \mathbb{Z}_2^n : S(x) + S(x+\alpha) = \beta\}}{2^n}$$

Esta probabilidad en general no es independiente de α y β.

Si S fuese lineal, se tendria $\Delta_\alpha S(x) = S(\alpha)\ \forall \alpha$, es decir, la derivada seria constante, y se tendría

$$DP(\alpha,\beta) = \begin{cases} 1 & si\ \beta = S(\alpha) \\ 0 & si\ no \end{cases}$$

Como hemos dicho, la función del S-box es proveer no linealidad, asi que querriamos una función que se alejara lo más posible de este comportamiento de las funciones lineales.

Hay entradas triviales, pues $DP(0,\beta) = 0$ si $\beta \neq 0$ y $DP(0,0) = 1$, pues $S(x+0) + S(x) = 0$ por estar en un espacio de característica 2.

Ademas, si queremos que S sea biyectiva, tenemos $DP(\alpha, 0) = 0$ si $\alpha \neq 0$. (pues $x + \alpha \neq x$ y S 1-1 implica $S(x+\alpha) \neq S(x)$, asi que, para todo x, $\Delta_\alpha S(x) \neq 0$ si $\alpha \neq 0$).

Salvo estas probabilidades triviales, el resto de las probabilidades depende mucho de S.

Para analizar mejor esto, se suele tomar la tabla de probabilidades $DP(\alpha, \beta)$ con $\alpha, \beta \neq 0$, y multiplicar cada entrada por 2^n para una lectura más fácil, es decir, se calcula lo que se llama la tabla de diferencias*, cuya entrada (α, β) es:*

$$\#\{x \in \mathbb{Z}_2^n : S(x) + S(x+\alpha) = \beta\}$$

Observemos que, si x es solución de la ecuación $S(x+\alpha) + S(x) = \beta$, entonces $x + \alpha$ tambien será solución de esa ecuación, pues $x+\alpha+\alpha = x$. Asi pues en general, cualquier entrada en la tabla de diferencias debe ser PAR, en particular, cualquier entrada no nula debe ser al menos 2. Dicho de otra forma, la mayor probabilidad diferencial de un S-box es de al menos $\frac{1}{2^{n-1}}$.

En general, tenemos:

4.3 Definición :

Una permutación $S : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$ se dice δ-uniforme si:

$$\#\{x \in \mathbb{Z}_2^n : S(x) + S(x+\alpha) = \beta\} \leq \delta\ \forall \alpha, \beta \neq 0$$

Entonces, estamos diciendo que lo mejor que podriamos esperar seria una función que fuese 2*-uniforme. La pregunta es si existen.*

Como ejemplo, tomemos $n = 3$ *(para* $n = 2$ *todas las funciones de* $\mathbb{Z}_2^2 \mapsto \mathbb{Z}_2^2$ *son lineales o afines).*

Podriamos analizar el Cesar Shift en $\mathbb{Z}_8$*. El Cesar Shift es lineal con respecto a la operación de suma en* $\mathbb{Z}_8$*, pero quizas sea una buena función no lineal con respecto a la operación de* $\mathbb{Z}_2^3$*, que es la operación con respecto a la cual estamos midiendo no linealidad. Para una notación más compacta denotaremos* $(0,0,1) = 1$*,* $(0,1,0) = 2$*,* $(0,1,1) = 3$*, etc. El CesarShift entonces es* $S = 34567012$*, es decir,* $S(0) = 3$*,* $S(1) = 4$*, etc. La tabla de diferencias (eliminando las entradas triviales) da:*

	1	2	3	4	5	6	7
1	0	0	4	0	0	0	4
2	0	4	0	0	0	4	0
3	4	0	0	0	4	0	0
4	0	0	0	8	0	0	0
5	0	0	4	0	0	0	4
6	0	4	0	0	0	4	0
7	4	0	0	0	4	0	0

Vemos que es muy mala. La mayor entrada es 8*, asi que con respecto a la uniformidad, es tan mala como una lineal. (una lineal tendria un 8 en cada fila, esta la tiene en una sola, pero aun asi esto seria un dia de fiesta para un criptoanalista)*

Consideremos ahora el S-box $S = 65273410$*. En este caso, obtenemos:*

	1	2	3	4	5	6	7
1	2	0	2	0	2	0	2
2	0	4	0	4	0	0	0
3	2	0	2	0	2	0	2
4	2	0	2	0	2	0	2
5	0	4	0	0	0	4	0
6	2	0	2	0	2	0	2
7	0	0	0	4	0	4	0

Esta tabla es mucho mejor, siendo 4*-uniforme.*

Pero se puede mejorar: si $S = 64170352$ *obtenemos*

	1	2	3	4	5	6	7
1	0	2	2	0	0	2	2
2	2	0	2	0	2	0	2
3	2	2	0	0	2	2	0
4	0	0	0	2	2	2	2
5	0	2	2	2	2	0	0
6	2	0	2	2	0	2	0
7	2	2	0	2	0	0	2

Observamos que toda entrada no trivial es 0 o 2, es decir, esta función es 2*-uniforme.*

En particular, si $\alpha, \beta \neq 0$*, tenemos que* $DP(\alpha, \beta)$ *es* 0 *o* $\frac{1}{4}$*, bastante alejada de la tabla de una función lineal.*

Como dijimos, un atacante buscará las peores entradas del S-box, es decir, las que alcanzan la cota del "δ*". Luego construira una diferencia global de entrada a la ronda, típicamente haciendo cero todas las*

entradas a los S-boxes menos a uno. Con la ayuda de las DP, estimará la diferencia de salida del S-box. El paso lineal tranforma las diferencias con probabilidad uno, asi que el atacante puede calcular las diferencia de salida de toda la ronda. Luego repite con la siguiente ronda, para construir una cadena de diferencias:

4.4 Definición :

Una característica diferencial a lo largo de r rondas es una sucesión de diferencias: $\Omega = (\alpha_1, \alpha_2, \ldots, \alpha_r)$ con α_i una diferencia al principio de la ronda i.

La probabilidad diferencial de la característica es el producto

$DP(\Omega) = \prod_{i=1} DP(\alpha_i, \alpha_{i+1})$.

(Nota: no hablaremos mucho de esto, pero en realidad eso es cierto solo bajo algo llamado la hipotesis de la independencia de las claves de ronda, en la cual se supone que no hay relación significativa entre las claves de rondas distintas. Esto es algo que debería suceder en cualquier expansión de clave buena)

El atacante entonces busca características con alta probabilidad, y en general se usan características de r rondas para poder atacar $r+1$ rondas. Si $DP(\Omega) = p$, se necesitarán al menos $\frac{1}{p}$ textos para que el ataque tenga éxito

Entonces, para defenderse de un ataque de DC, se desea que la probabilidad p de cualquier característica sea lo suficientemente chica como para que no sea factible juntar $\frac{1}{p}$ textos.

En realidad, estrictamente hablando, no basta con esto. el problema es que varias características podrian juntarse:

4.5 Definición :

Un **diferencial** de r rondas es un par (α, β) tal que existe una característica diferencial $(\alpha_1, \alpha_2, \ldots, \alpha_r)$ con $\alpha = \alpha_1$ y $\beta = \alpha_r$.

Un atacante no tiene acceso a las rondas intermedias, por lo que solo esta interesado en diferenciales, no en características. Pero ¿como los encuentra? En general, debe analizar el cifer ronda por ronda, encontrando caracterisitcas diferenciales que le permitan juntarlas para crear el diferencial, que es el que usará. Si la característica tiene probabilidad alta, el atacante sabe que tendrá éxito. Pero, podria suceder que ninguna característica tenga probabilidad alta, pero que existan (aun cuando el atacante no lo sepa) multiples características que lleven al mismo diferencial, i.e., $(\alpha_1^{(j)}, \alpha_2^{(j)}, \ldots, \alpha_r^{(j)})$, $j = 1, ..., M$ con $\alpha = \alpha_1^{(j)}$ y $\beta = \alpha_r^{(j)}$ para todo j. Con lo cual, la probabilidad de que ocurra el diferencial no es $\prod_{i=1}^{r-1} DP(\alpha_i, \alpha_{i+1})$ sino más bien

$$\sum_{j=1}^{M} \prod_{i=1}^{r-1} DP(\alpha_i^{(j)}, \alpha_{i+1}^{(j)})$$

con lo cual, aun cuando cada probabilidad individual sea baja, si M es lo suficientemente grande, la probabilidad total seria de considerar.

De hecho, esto es lo que ocurre en el AES: veremos que la probabilidad de cualquier característica de 8 rondas tendrá probabilidad de a lo sumo 2^{-300}. Por otro lado, la suma de las probabilidades de todos los diferenciales debe ser uno, asi que en cualquier cifer de 128 bits debe haber al menos un diferencial con probabilidad al menos 2^{-128}, y a menos que todos los diferenciales tuvieran la misma probabilidad, debe haber alguno con probabilidad mayor.

Por supuesto, el problema para un atacante seria encontrarlo, pues no puede guiarse por las características. Ademas, si bien se pueden diseñar a proposito cifers cuyas características diferenciales tengan baja probabilidad pero con diferenciales de alta probabilidad, Rijndael y la mayoria de cifers bien diseñados no tienen una estructura que permita suponer que hay agrupamientos anormales de características. Por lo que en general se acepta que un cifer con bajas probabilidades de características es seguro contra DC.

De cualquier forma, el constructor de un cifer debe construir S-boxes con una δ-uniformidad con δ lo más chico posible.

Vimos más arriba un ejemplo de un S-box de 3 bits que era 2-uniforme. Veremos más abajo que para todo n impar existen permutaciones 2-uniformes de $\mathbb{Z}_2^n$. En el caso n par, esto no se sabe si es cierto o no, y de hecho se cree que no es cierto:

Conjetura (Open Problem) No existen permutaciones $S : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$ que sean 2-uniformes cuando n es par.

(Se conocen varias **funciones** *2-uniformes en el caso n par, pero ninguna de las que se conocen es biyectiva).*

Si la conjetura es cierta, como cada entrada de la tabla de diferencias debe ser par pareceria que lo mejor que podemos pedir es que un S-box sea 4-uniforme en el caso n par. (si queremos que el S-box sea biyectivo).

Diremos entonces que un S-box es óptimo respecto de DC si es 2-uniforme en el caso impar y 4-uniforme en el caso par.

¿Como encontrar S-boxes "optimos"?

Una forma es crearlos al azar hasta encontrar alguno. Esto se puede hacer si n es chico (por ejemplo, para $n = 3, 4$), pero para $n = 8$ por ejemplo, aparentemente no se puede. El team de Rijndael chequeo más de un millon de S-boxes al azar y no encontró ninguno que fuese 4-uniforme. Sin embargo, hay una forma de construir directamente S-boxes óptimos. En 1994, Nyberg dio un método para construir S-boxes óptimos, sin necesidad de crear al azar y luego testear.

4.6 Teorema (Nyberg, primera parte) :

Sea S el S-box dado por $S(x) = x^{-1}$ si $x \neq 0$ y $S(0) = 0$, donde la inversión se hace en el cuerpo finito $GF(2^n)$.

Entonces, S es 2-uniforme si n es impar y 4-uniforme (pero no 2 uniforme) si n es par.

Prueba:

Dados $\alpha, \beta \neq 0$, queremos contar el número de soluciones de la ecuación $S(x)+S(x+\alpha) = \beta$. Observemos primero que si $\beta = \alpha^{-1}$, tenemos las dos soluciones $x = 0, \alpha$, pues $S(0) + S(\alpha) = \alpha^{-1} = \beta$.

Si $\beta \neq \alpha^{-1}$ entonces 0 y α no son soluciones. Independientemente que lo sean o no, queremos contar cuantas otras soluciones hay.

Supongamos entonces $x \neq 0, \alpha$. En ese caso:

$S(x) + S(x + \alpha) = \frac{1}{x} + \frac{1}{x+\alpha} = \frac{x+\alpha+x}{x(x+\alpha)} = \frac{\alpha}{x(x+\alpha)}$

Por lo tanto $S(x) + S(x + \alpha) = \beta$ si y solo si $\alpha = x(x + \alpha)\beta$. El número de x's que satisface esta ecuación viene dado entonces por una ecuación de grado 2, y como estamos en un cuerpo, la ecuación solo puede tener dos soluciones como máximo.

Entonces, el numero total de soluciones es 0 o 2 si $\beta \neq \alpha^{-1}$ y 2 o 4 si $\beta = \alpha^{-1}$.

Con lo cual vemos que estos S-boxes son 4-uniformes.

Veamos que en el caso n par la cota 4 se alcanza (es decir, no son 2-uniforme), pero en el caso n impar no.

En cualquiera de los casos, vemos que la unica posibilidad de que haya cuatro soluciones es si $\beta = \alpha^{-1}$. Supongamos eso, entonces, y analizemos que soluciones hay ademas de las triviales de la ecuación $S(x) + S(x+\alpha) = \alpha^{-1}$. Si $x \neq 0, \alpha$, tenemos por la cuenta de arriba que: $\alpha = x(x+\alpha)\beta = x(x+\alpha)\alpha^{-1}$ por lo que $1 = (x\alpha^{-1})(x\alpha^{-1}+1)$. Llamando $y = x\alpha^{-1}$, tenemos que $1 = y(y+1) = y^2 + y$.

Recordemos que como $(GF(2^n) - \{0\}, .)$ es un grupo, por el teorema de Lagrange tendremos que el orden de cualquier elemento dividirá al orden del grupo, asi que $y^{2^n-1} = 1$, es decir, $y^{2^n} = y$. Ademas, como $(a+b)^2 = a^2 + b^2$ en $GF(2^n)$, tenemos: (elevando al cuadrado repetidamente y usando $y^{2^n} = y$)

$$\begin{aligned}
1 =& y^2 + y \\
1 =& y^4 + y^2 \\
1 =& y^8 + y^4 \\
& \vdots \quad \vdots \\
1 =& y^{2^{n-1}} + y^{2^{n-2}} \\
1 =& y + y^{2^{n-1}}
\end{aligned}$$

lo cual nos da, sumando, que $\overbrace{1 + ... + 1}^{n} = 0$, lo cual solo puede pasar si n es par. Con lo cual vemos que en el caso n impar el elemento "y" no existe y la función es 2-uniforme.

Asi, vemos que para n impar esta función es 2-uniforme.

En el caso par, $n = 2k$, tenemos que $2^n - 1 = 4^k - 1 \equiv_{(3)} 1^k - 1 = 0$, es decir, 3 divide a $2^n - 1$. Ahora bien, recpordemos que el grupo multiplicativo $(GF(2^n) - \{0\}, .)$ es isomorfo al grupo $(\mathbb{Z}_{2^n-1}, +)$, por lo que existe al menos un elemento cuyo orden (multiplicativo) es exactamente $2^n - 1$; es decir, un elemento primitivo. (esto se suele llamar el teorema del elemento primitivo)

Si ρ es un tal elemento, sea $\gamma = \rho^{\frac{2^n-1}{3}}$, que esta bien definido pues 3 divide a $2^n - 1$. Entonces $\gamma^3 = \rho^{2^n-1} = 1$, por lo tanto tenemos que $0 = \gamma^3 + 1 = (\gamma+1)(\gamma^2+\gamma+1)$ y como $\gamma \neq 1$ (pues ρ es primitivo), tenemos que $\gamma^2 + \gamma + 1 = 0$. Tomando entonces $x = \alpha\gamma$, el cual será distinto de 0 y de α (pues $\gamma \neq 1$) tenemos una solución extra de la ecuación. (y, por lo tanto, en realidad dos soluciones extras, pues $\alpha + \alpha\gamma$ tambien será entonces solución). Asi, en el caso n par, tenemos 4 soluciones a la ecuación $S(x) + S(x+\alpha) = \alpha^{-1}$, con lo cual S es 4-uniforme pero no 2-uniforme.

QED.

5. El S-box de AES

El teorema de Nyberg entonces nos permite construir un S-box resistente a DC. Por ello, los creadores de Rijndael decidieron usar este método para construir su S-box. Observemos que, desde el punto de vista diferencial, seria mejor usar un S-box de tamaño impar, pues tienen mejores características. El problema aca es que los procesadores trabajan mejor en conjuntos de bits que sean potencias de 2, como 8 bits, 16 bits o 32 bits. Por lo tanto, se prefirió usar un S-box de 8 bits, por motivos de implementación.

Ahora bien, un S-box puro de Nyberg tiene algunos defectos obvios: entre otras cosas, el 0 y el 1 quedan fijos (i.e., no hay substitución), y por otro lado, si se lo mira como polinomio sobre $GF(2^8)$, el S-box es simplemente $x \mapsto x^{254}$ (como dijimos antes, por el teorema de Lagrange $a^{255} = 1$ para todo $a \neq 0$, y por lo tanto $a^{-1} = a^{254}$ si $a \neq 0$. Como $0^{254} = 0$, tenemos que el S-box de Nyberg es x^{254}). Si el polinomio que representa al S-box cuando se lo mira como un mapa en $GF(2^8)$ tiene pocos terminos, existe la posibilidad de que se puede montar otro tipo de ataques, llamados ataques de interpolación. Por lo tanto, para evitar estos ataques, y la debilidad de tener dos puntos fijos, en Rijndael se modifica la construcción básica de Nyberg: Si denotamos por $N(x)$ a un S-box de Nyberg, entonces el S-box de Rijndael es de la forma $S(x) = AN(x) + b$, donde ahora miramos los elementos de $GF(2^8)$ como vectores columnas sobre $\mathbb{Z}_2^8$, y donde A es una matriz 8×8 invertible sobre $\mathbb{Z}_2$ y b es un vector columna fijo. Eligiendo A y b cuidadosamente, se obtiene un S-box que no tiene puntos fijos, ni puntos fijos opuestos (un a que va al complemento de a), y tal que la descripción del S-box como polinomio sobre $GF(2^8)$ tenga muchos terminos distintos de cero. Con las elecciones que hizo el team de Rijndael, el polinomio tiene 9 terminos con coeficientes distintos de cero. (los correspondientes a los grados 0, 127, 191, 223, 239, 247, 251, 253 y 254). No obstante, con una mejor elección de A y b, se podria haber logrado un polinomio en donde todos los terminos tuvieran coeficientes distintos de cero. La razon por la cual no hicieron esto es que ademas de todas las demas restricciones, los creadores de Rijndael se autoimpusieron otra: querian que no se pudiera sospechar que hubiera ninguna puerta oculta en Rijndael. Por lo tanto, la matriz A la elijieron como la matriz que representa una multiplicación por un polinomio módulo $x^8 + 1$, cuando se mira los elementos de $\mathbb{Z}_2^n$ como polinomios en $\mathbb{Z}_2[x]/(x^8 + 1)$, y el polinomio que elijieron fue el primer polinomio en una tabla publicada años antes.

De todos modos, resulta que esta construcción es más o menos inútil: se puede probar que la multiplicación por A se puede desaclopar del S-box y mirarsela como parte de la transformación lineal, y la suma de b se puede aclopar a la expansion de la clave. Por lo tanto, si bien el S-box de Rijndael es, en la presentación original de Rijmen y Daemen un S-box de Nyberg modificado, se puede mirar a Rinjndael como un SLN con un S-box puro de Nyberg, cambiando la transformación lineal y la expansión de la clave.

6. La matriz de difusión de Rijndael

Habiamos dicho que Rijndael, en el paso MixColumn, usa una matriz con elementos en $GF(2^8)$ para mezclar los elementos de cada columna. Esta matriz fue elejida cuidadosamente, apoyandose en la teoría de códigos de corrección de errores.

6.1 Definición :

Dado un espacio vectorial $\mathbb{F}^k$, y una transformación lineal $L : \mathbb{F}^k \mapsto \mathbb{F}^k$, se define el (differential) **branch number** de L como:

$$\mathcal{B}(L) = \text{Min}\{w(x) + w(L(x)) | x \in \mathbb{F}^k, x \neq 0\}$$

donde w es el peso de Hamming $w(x) = \#\{i \in \{1, ..., k\} | x_i \neq 0\}$.

El branch number de una transformación lineal es una medida de su poder de difusión: si x tiene t entradas distintas de cero, entonces se sabe que $L(x)$ tendrá al menos $\mathcal{B}(L) - t$ entradas distintas de cero.

Obviamente, tomando x de peso 1, como $w(L(x)) \leq k$, tenemos que $w(x) + w(L(x)) \leq k+1$ para esos xs, con lo cual $\mathcal{B}(L) \leq k+1$.

Una transformación lineal L se dice maximalmente difusiva si $\mathcal{B}(L) = k+1$. Analogamente podemos hablar de matrices maximalmente difusivas.

¿como podemos hallar una? ¿Existen?

Recordemos de la teoría de códigos que un código (n, k, δ) es un código lineal de longitud n, dimensión k y distancia (de Hamming) mínima entre palabras no nulas igual a δ.

Recordemos que una matriz de chequeo de un código C es una matriz H tal que $C = Nu(H)$.

El siguiente teorema de teoría de códigos se puede usar pera calcular δ:

6.2 Teorema :

Si H es una matriz de chequeo de un código C, entonces C tiene distancia mínima entre palabras igual a δ si y solo si cualesquiera $\delta - 1$ columnas de H son linealmente independientes pero existen δ columnas linealmente dependientes.

Prueba:

Esto es la simple formula de álgebra lineal $Hx^t = \sum_j H^{(j)} x_j$, donde $H^{(j)}$ es la columna j-ésima de H, por lo que $Hx^t = 0 \iff \sum_j H^{(j)} x_j = 0$, es decir, existen r columnas linealmente dependientes de H si y solo si existe un vector x de peso r tal que $Hx^t = 0$, es decir, sii existe un elemento de peso r en C. Por ser C lineal, δ es el menor peso de Hamming no nulo de las palabras de C.

QED.

6.3 Corolario ("Singleton bound") :

Un código (n, k, δ) satisface $\delta \leq n - k + 1$.

Prueba:

Pues por el teorema anterior, δ es el menor numero de columnas linealmente dependientes de H, por lo tanto, es menor o igual que el rango de H más 1. Pero el rango de H es $n - k$.

QED.

6.4 Definición :

Códigos (n, k, δ) tales que $\delta = n - k + 1$ se llaman códigos MDS (Maximum Distance Separable Codes).

Por ejemplo, los códigos de Reed-Solomon son MDS.

Sea C un código (n, k, δ) sobre $GF(2^m)$ que sea MDS, y sea G una matriz generadora del mismo, reducida por filas. Entonces G es de la forma $G = [I|A]$ con la identidad I $k \times k$ y A $k \times (n-k)$. Diremos que una tal matriz es una matriz MDS.

6.5 Propiedad :

Una matriz es MDS sii el determinante de toda submatriz cuadrada es no nulo.

Prueba:

La condición del lado derecho es verdadera para A si y solo si lo es para A^t, asi que lo probaremos para A^t.

Si $G = [I|A]$, es sabido que una matriz de chequeo es $H = [A^t|I]$. Supongamos que C sea MDS. Sea B una submatriz $r \times r$ de A^t. Entonces debe ser $r \leq n - k$. Tomemos las $n - k - r$ columnas de I con "1"s en filas que no correspondan con filas de B, junto con las r columnas de A^t correspondientes a las columnas de B. Esas $n - k$ columnas de H deben ser linealmente independientes, por ser C MDS. Veamos un dibujo (sin perdida de generalidad, supongamos que son las primeras r filas y r columnas):

$$\begin{bmatrix} B & 0 \\ * & I \end{bmatrix}$$

Reduciendo las entradas en la matriz $*$ usando la identidad, obtenemos una matriz de la forma

$$\begin{bmatrix} B & 0 \\ 0 & I \end{bmatrix}$$

Las columnas de esta matriz deben ser linealmente independientes, lo cual implica que $\det B \neq 0$.

Viceversa, supongamos que toda submatriz cuadrada de A^t es invertible. Tomemos $n - k$ columnas de H. Si esas $n - k$ columnas son las $n - k$ columnas de la derecha, forman la identidad y son linealmente independientes. Suopongamos entonces que en esas $n - k$ columnas hay solo $n - k - r$ correspondientes a las ultimas $n - k$ columnas, y r en las primeras k columnas. Tomemos la matriz B cuyos elementos estan en esas r columnas y en las filas donde no hay ningun "1" en las $n - k - r$ columnas elejidas en la derecha. Como $\det B \neq 0$, sus columnas son linealmente independientes, con lo cual las columnas correspondientes en A^t tambien lo son. Ademas, como las $n - k - r$ columnas restantes tienen unos en filas distintas de las de B, las $n - k$ columnas son linealmente independientes. Esto prueba que el código es MDS.

QED.

6.6 Corolario :

Una matriz es MDS si y solo si su transpuesta lo es.

6.7 Teorema :

Una matriz MDS correspondiente a un código (n, k, δ) con $n = 2k$ es maximálmente difusiva.

Prueba:

Por el corolario anterior, si A es MDS, A^t lo es. Por lo tanto consideremos el código C con matriz generadora $G = [I|A^t]$.

Tomemos un vectores (fila) v que este en el código C. La distancia entre v y 0 es entonces de al menos $\delta = n - k + 1 = 2k - k + 1 = k + 1$. Por ser G generadora, debemos tener que existe u tal que $v = uG$. Entonces, v es de la forma (u, uA^t). Por lo tanto:

$$k + 1 = \delta \leq d_H(v, 0) = d_H(u, 0) + d_H(uA^t, 0) = w(u) + w(uA^t) \qquad (**)$$

Tomando ahora vectores columnas $x = u^t$, $(**)$ dice que $k + 1 \leq w(x) + w(Ax)$ y por lo tanto A es maximalmente difusiva.

QED.

En el caso particular de $k = 4$, tenemos entonces una matriz 4×4 tal que la suma de las diferencias de entrada más las diferencias de salida será de al menos 5.

Como dijimos, una tal matriz podria derivarse de un código de Reed-Solomon. Los autores de Rijndael usaron este método con un antecesor de Rijndael, el cifer Shark, pero para Rijndael prefirieron no usar un código de Reed-Solomon, pues en ese caso los coeficientes de la matriz serian todos distintos y complicados respecto a la implementación de la multiplicación en $GF(2^8)$. Si bien esta multiplicación, en la implementación para procesadores de 32 bits, la harian ellos, y luego se guardaria todo en una tabla, esto no puede hacerse cuando se lo quiere implementar en procesadores de 8 bits (smartcards, por ejemplo) o en hardware. (es por prestar atención a estos pequeños detalles de implementación que Rijndael se convirtio en AES). En vez de ellos, los autores buscaron al azar (usando la propiedad de los determinantes de las submatrices, ver **6.5.** *) hasta encontrar una matriz MDS que tuviera coeficientes simples, de hecho, los coeficientes son todos 1,2 o 3. La matriz es:*

$$M = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Recordemos que podemos representar a $GF(2^8)$ por medio de polinomios de grado menor o igual a 7 con coeficientes en $\mathbb{Z}_2$. Aqui se entiende que "2"= x y "3"= $x+1$ en esa representación. Con lo cual la multiplicación se efectua en forma rápida.

7. Resistencia al DC

Recordemos que en un ataque DC lo que se quiere es introducir una diferencia de entrada α y luego rastrear su evolución más probable a lo largo de las rondas. Ahora bien, para hacer esto, debemos ver su paso por los S-boxes. Entonces debemos dividir α en subbloques que correspondan con los S-boxes. En el caso de Rijndael, tendremos $\alpha = (\alpha_1, ..., \alpha_{16})$. Ahora bien, sabemos que si la diferencia de entrada es 0 entonces la diferencia de salida tambien será cero, pero esta es una diferencia no explotable en un ataque DC. Entonces, hay que fijarse en aquellos i tales que $\alpha_i \neq 0$. Llamaremos a estos S-boxes **"activos"**. *Entonces, en un ataque DC, se deberá usar una diferencia de entrada, calcular la probabilidad del pasaje por cada S-box activo, y seguir rastreando estas diferencias a lo largo de las rondas. Obviamente, mientras más S-boxes sean activos, como cada uno agrega su propia probabilidad, menor será la probabilidad total. Por lo tanto, un atacante busca dos cosas: primero, probabilidades relativamente altas en el S-box, y segundo, diferencias que no provoquen demasiados S-boxes activos, para que la probabilidad se mantenga más o menos alta.*

Por ello, Rijmen y Daemen se aseguraron de introducirles dificultades a un atacante en ambos frentes: usaron un S-box que fuese lo mejor posible desde el punto de vista diferencial, y usaron una matriz con el mayor poder de difusión posible.

Rijmen y Daemen probaron el siguiente:

7.1 Lema :

En cuatro rondas consecutivas de Rijndael hay al menos 25 S-boxes activos.

Con este lema, probaron:

7.2 Teorema :

En Rijndael, ninguna característica diferencial de 4 rondas tiene probabilidad mayor a 2^{-150}.

Prueba:

En cualquier ataque diferencial, por el lema, debe haber al menos 25 S-boxes activos en 4 rondas. Por el teorema de Nyberg, cada uno de esos S-boxes es 4-uniforme, asi que tiene una probabilidad diferencial máxima de $\frac{4}{2^8} = 2^{-6}$. Asi, la probabilidad diferencial total es de a lo sumo $(2^{-6})^{25} = 2^{-150}$.

QED.

Esto significa que para poder montar un ataque DC con alguna probabilidad de éxito contra solamente 4 rondas de Rijndael, se deben disponer de al menos 2^{150} mensajes. Pero como el bloque tiene un tamaño de sólo 128 bits, el TOTAL de mensajes es de solo 2^{128}, y por lo tanto un ataque DC no puede ser montado, ni siquiera contra 4 rondas de Rijndael, mucho menos para las 10 rondas.

Nos queda ver el lema:

Prueba del lema 7.1.

Denotemos por $\rho_{r,j}$ el número de S-boxes activos en la columna j en la ronda r, al principio de la ronda. (o, lo que es lo mismo, justo antes de hacer ShiftRow, porque el paso de substitución no cambia el número de S-boxes activos).

Denotemos por $\sigma_{r,j}$ el número de S-boxes activos en la columna j en la ronda r justo despues de hacer ShiftRow, pero antes de hacer MixColumn.

Observemos que luego de hacer MixColumn, el numero de S-boxes activos no cambia hasta el ShiftRow de la siguiente ronda, es decir, el número de S-boxes activos en la columna j en la ronda r DESPUES de MixColumn es simplemente $\rho_{r+1,j}$.

Entonces, como la matriz M es MDS, tenemos que:

$$\sigma_{r,j} \neq 0 \Rightarrow \sigma_{r,j} + \rho_{r+1,j} \geq 5 \qquad (1)$$

Llamando a una columna activa si tiene al menos un S-box activo, y denotando por Ω_r el número de columnas activas al final de la ronda r, tenemos por (1) que:

$$\sum_{j=1}^{4} \sigma_{r,j} + \rho_{r+1,j} \geq 5\Omega_r \qquad (2)$$

(esto pues si una columna es activa luego de MixColumn, tambien lo era antes de MixColumn)

Denotemos ademas por γ_r el número de S-boxes activos en la ronda r, al principio de la ronda. Entonces, $\gamma_r = \sum_{j=1}^{4} \rho_{r,j}$ Pero como ShiftRow mantiene la cantidad de S-boxes activos, pues solo es una permutación de bytes, entonces tambien es cierto que $\gamma_r = \sum_{j=1}^{4} \sigma_{r,j}$ Podemos entonces reescribir (2) como:

$$\gamma_r + \gamma_{r+1} \geq 5\Omega_r \qquad (3)$$

Denotando por N el total de S-boxes activos en 4 rondas consecutivas, tenemos que $N = \gamma_1 + \gamma_2 + \gamma_3 + \gamma_4$, asi pues, por (3):

$$N \geq 5(\Omega_1 + \Omega_3) \qquad (4)$$

Como MixColumn no cambia el número de columnas activas, Ω_3 es igual al número de columnas activas justo despues de aplicar ShiftRow en la ronda 3. Ahora bien, ShiftRow manda cada entrada de una columna a columnas distintas, por lo tanto si tomamos en la ronda 3 la columna con el mayor número de S-boxes

activos antes de ShiftRow, digamos que es la columna q, cada uno de esos S-boxes activos irá a parar a una columna distinta, por lo que:

$$\Omega_3 \geq \rho_{3,q} = Max_{j=1}^{4} \rho_{3,j} \qquad (5)$$

En cuanto a Ω_1, podemos hacer un razonamiento similar, pues la inversa de ShiftRow también tiene la propiedad de mandar todos los elementos de una columna a columnas distintas. Entonces, tomemos en la ronda 2 la columna con el mayor número de S-boxes activos luego de ShiftRow,digamos que es la columna t. Cada uno de esos S-boxes activos vino de una columna distinta antes de ShiftRow, por lo que el número de columnas activas antes de ShiftRow será al menos esa cantidad de S-boxes activos. Es decir, el número de columnas activas al principio de la ronda 2 es al menos igual a $\sigma_{2,t}$. Pero el número de columnas activas al principio de la ronda 2 es igual al número de columnas activas al final de la ronda 1, es decir, igual a Ω_1. Tenemos entonces que:

$$\Omega_1 \geq \sigma_{2,t} = Max_{j=1}^{4} \sigma_{2,j} \qquad (6)$$

(4),(5)y (6) nos dan:

$$\begin{aligned} N \geq & 5(Max_{j=1}^{4} \rho_{3,j} + Max_{j=1}^{4} \sigma_{2,j}) \\ \geq & 5 Max_{j=1}^{4} (\rho_{3,j} + \sigma_{2,j}) \\ \geq & 5.5 = 25 \qquad \text{(por (1)} \end{aligned}$$

QED.

Como dijimos en la discusion de DC, en teoría no basta con acotar las caracterisiticas, sino que hay que pensar en los diferenciales. Si bien puede pasar que haya diferenciales con probabilidad alta y características con probabilidad baja, esos cifers tienen anomalias en su construcción. Rijndael trata esencialmente a todos los bits de la misma forma, por lo que es razonable suponer que no habra ningún "clustering"anormal de características.

Este argumento heuristico ha sido estudiado intensivamente, y en [PSCY02] se prueba que en cualquier cifer del tipo del AES, cualquier diferencial de al menos 4 rondas tiene probabilidad acotada por:

$$p^{16} + Max\{4p^{17} + 6p^{18} + 4p^{19}, 4p^{18} + 72p^{19} + 438p^{20} + 912p^{21} + 184p^{22}\}$$

(i.e., esencialmente p^{16}) donde p es la mayor probabilidad diferencial del S-box. En el caso de Rijndael, se obtiene una cota de $1,06.2^{-96}$. Analisis posteriores en [PSLL03] han bajado esta probabilidad a $1,144.2^{-111}$, con lo cual se puede concluir que aun desde el punto de vista de diferenciales, Rijndael es seguro.

Pasemos ahora a la segunda noción de "altamente no lineal", que tiene que ver con el ataque de Criptoanalisis Lineal.

8. Criptoanálisis Lineal

Matsui introdujo en 1993 el **Linear Cryptanalysis** *(LC), que explota la baja no linealidad de los S-boxes. LC es un ataque de tipo* known plaintext *que aprovecha las desviaciones estadísticas de las distintas* **relaciones lineales** *entre los datos de entrada al S-box y los de salida: si bien el S-box no es*

lineal, puede haber ciertas relaciones lineales entre bits individuales de los datos de entrada y los de salida. Estas relaciones no serán satisfechas siempre, pero basta con que sean satisfechas con cierta probabilidad.

Por ejemplo, denotemos $Y = S(X)$*, y supongamos que deseamos investigar la relación* $y_2 + y_3 = x_1 + x_2 + x_4$*. (donde* $+$ *es la suma XOR de* $\mathbb{Z}_2$*).*

Si no hay relación en el conjunto de bits seleccionados, entonces la ecuación booleana a que da lugar será satisfecha con probabilidad $\frac{1}{2}$*. El problema podría ocurrir con relaciones que ocurren con probabilidad significativamente mayor a 1/2 o significativamente menor a 1/2.*

Si fuese significativamente mayor, esta claro que se puede explotar. Pero aun si fuese menor se puede explotar, pues supongamos que esa probabilidad sea solo del 10%. Eso significa que existe un 90% de posibilidades de que $y_2 + y_3 = 1 + x_1 + x_2 + x_4$*, con lo cual tenemos una ecuación afin que es posible usar.*

Ademas, esto permite ignorar el efecto de la clave: para ejemplificar, tomemos un texto de $n = 4$ *bits* $P = (P_1, P_2, P_3, P_4)$*. Consideremos la expresión lineal* $L = P_1 + P_2 + P_4$*. Observemos que* L *puede describirse fácilmente como* $L = P \cdot \Gamma$*, donde "*$\cdot$*" es la forma bilineal canónica del espacio vectorial* $\mathbb{Z}_2^4$ *y* Γ *es el vector* $(1, 1, 0, 1)$*, pues:*

$$\begin{aligned} P \cdot \Gamma =& (P_1, P_2, P_3, P_4) \cdot (1, 1, 0, 1) \\ =& P_1.1 + P_2.1 + P_3.0 + P_4.1 \\ =& P_1 + P_2 + P_4 \\ =& L \end{aligned}$$

Si mezclamos con la clave, obtenemos $X = P + k$*, y por la bilinealidad, obtenemos que* $X \cdot \Gamma = (P \cdot \Gamma) + (k \cdot \Gamma)$ *i.e.,* $X \cdot \Gamma$ *y* $P \cdot \Gamma$ *son iguales excepto por el termino* $t = k \cdot \Gamma$*, que es 0 ó 1 y no cambia una vez fijada la clave. Sea* $Y = S(X)$*, y supongamos que nos interesa la expresión* $y_2 + y_3 = Y \cdot \Lambda$*, donde* $\Lambda = (0, 1, 1, 0)$*. Si el atacante busca explotar esta situación querrá que alguna de las probabilidades:* $Prob(Y \cdot \Lambda = X \cdot \Gamma)$ *o* $Prob(Y \cdot \Lambda = 1 + X \cdot \Gamma)$ *sea alta, pero esto es lo mismo que pedir que* $Prob(Y \cdot \Lambda = X \cdot \Gamma)$ *sea substancialmente mayor o substancialmente menor que 1/2, i.e., que el "bias" definido como probabilidad menos 1/2 sea alto en valor absoluto. Como* $Prob(Y \cdot \Lambda = X \cdot \Gamma) = 1 - Prob(Y \cdot \Lambda = t + P \cdot \Gamma)$*, entonces basta con saber que* $Prob(Y \cdot \Lambda = X \cdot \Gamma)$ *se aparta de 1/2 para saber que* $Prob(Y \cdot \Lambda = P \cdot \Gamma)$ *se aparta de 1/2.*

Análogamente al DC , en LC tenemos la tabla de biases (o tabla de aproximaciones lineales):

$$(\Gamma, \Lambda) \mapsto \frac{\#\{x \in \mathbb{Z}_2^n : x \cdot \Gamma = S(x) \cdot \Lambda)\}}{2^n} - \frac{1}{2}$$

Γ, Λ *se llaman tradicionalmente* mascaras*, aunque Rijmen y Daemen le llaman "selection patterns".*

Se desea que el mayor valor absoluto *de la tabla sea lo menor posible. Al igual que en el caso diferencial, los valores de las entradas con* $\Gamma = 0$ *o* $\Lambda = 0$ *son triviales: Si* $\Lambda = \Gamma = 0$*, entonces el sistema es* $0 = 0$*, es decir, todos los* x *lo satisfacen. Si* $\Lambda = 0 \neq \Gamma$*, entonces queremos saber cuantos* x *hay con* $\Gamma \cdot x = 0$*, y esto es claramente la mitad. (es un sistema de UNA ecuación lineal: el núcleo tiene dimension* $n - 1$*). Si* $\Gamma = 0 \neq \Lambda$*, entonces queremos saber cuantas soluciones tiene el sistema* $\Lambda \cdot S(x) = 0$*. Pero el sistema* $\Lambda \cdot y = 0$ *tiene* 2^{n-1} *soluciones, y* S *es biyectiva, asi que hay tambien* 2^{n-1} *soluciones.*

Por lo tanto, nos concentraremos en los casos en los que las mascaras sean no nulas.

Ahora bien, en el caso de DC teniamos probabilidades, las cuales sabemos que se multiplican. Pero aca tenemos biases. ¿que pasa con ellos? Matsui probó un lema que permite rastrear estos biases de ronda a

ronda. Sin embargo, diversos autores (entre ellos los creadores de Rijndael) luego aclararon que es mejor tratar no con los biases, sino con el doble de los biases. En este contexto al doble de los biases le llaman "correlación":

8.1 Definición :

Dadas dos variables booleanas X e Y, la correlación entre X e Y es $c(X,Y) = 2P(X = Y) - 1$.

Como de ahora en más usaremos tanto la suma XOR como la suma usual de enteros, distinguiremos la suma XOR como $\oplus$.

Ahora bien, Matsui probó que si se tiene un bias de b, se necesitan aproximadamente $O(b^{-2})$ textos para que el ataque tenga éxito. Por ello, otros autores mencionaron que para establecer mayor semejanza entre DC y LC, quizas seria mejor usar el cuadrado de los biases como medida. Sumado a la observación de que el uso de las correlaciones es más fácil de aplicar que los biases, esto motivó que el propio Matsui propusiera que se usara como medida el cuadrado de las correlaciones. A esta estadistica se le llamó la probabilidad lineal *(aunque los autores de Rindael le llaman "correlation potential"):*

8.2 Definición :

Dadas variables booleana X, Y; la probabilidad lineal $LP(X{=}Y)$ se define como:

$$LP(X{=}Y) = (2P(X{=}Y) - 1)^2$$

Dada una función $S : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$, y mascaras Λ, Γ definimos $LP(\Gamma, \Lambda)$ como:

$$LP(\Gamma, \Lambda) = LP(x \cdot \Gamma{=}S(x) \cdot \Lambda)$$

Hay que aclarar que la probabilidad lineal NO ES una probabilidad: se la llama asi por analogia con dos propiedades de las probabilidaes: por un lado, la probabilidad lineal es un número entre 1 y 0. Ademas, la probabilidad usual sirve para calcular probabilidades de productos de variables booleanas: si $X_1, ..., X_n$ son variables aleatorias booleanas independientes, entonces $P(X_1...X_n{=}1) = P(X_1{=}1)P(X_2{=}1)...P(X_n{=}1)$. La probabilidad lineal tiene esa propiedad, pero con respecto a la SUMA de variables aleatorias. (¿quizas seria mejor llamarla probabilidad logaritmica?)

8.3 Teorema (Piling-Up lemma de Matsui -en terminos de probabilidades lineales-) :

Denotando la suma XOR por $\oplus$, sean X_i variables aleatorias booleanas independientes. Entonces:

$$LP(X_1 \oplus X_2 \oplus ... \oplus X_n{=}0) = LP(X_1{=}0)LP(X_2{=}0)...LP(X_n{=}0)$$

Prueba:

Usando inducción en n, esta claro que basta probarlo para $n = 2$. Denotemos por $p_i = P(X_i = 0)$, y

por $p = P(X_1 \oplus X_2 = 0)$. Ahora bien, $X_1 \oplus X_2 = 0 \iff X_1 = X_2$, por lo tanto:

$$\begin{aligned}
2p-1 =& 2P(X_1 \oplus X_2 = 0) - 1 \\
=& 2[P(X_1 = 0)P(X_2 = 0) + P(X_1 = 1)P(X_2 = 1)] - 1 \\
=& 2[p_1p_2 + (1-p_1)(1-p_2)] - 1 \\
=& 2[p_1p_2 + 1 - (p_1 + p_2) + p_1p_2] - 1 \\
=& 4p_1p_2 - 2(p_1 + p_2) + 1 \\
=& (2p_1 - 1)(2p_2 - 1)
\end{aligned}$$

Elevando al cuadrado tenemos $LP(X_1 \oplus X_2 = 0) = LP(X_1 = 0)LP(X_2 = 0)$.

QED.

Por lo tanto, se pueden ir rastreando "características lineales" en forma análoga a como se rastrean las "características diferenciales".

9. No Linealidad y Resistencia a LC

Recordemos que habiamos dicho que una medida natural que se puede usar para medir la no linealidad de una función es su distancia de las funciones afines, y que esta medida en realidad mide la resistencia del S-box al ataque LC. Veamos en detalle esto. Para ello, primero debemos introducir una distancia.

9.1 Definición :

Dadas funciones $f, g : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2$ la distancia (de Hamming) entre ellas viene dada por

$$d(f,g) = \#\{x \in \mathbb{Z}_2^n | f(x) \neq g(x)\}$$

Dada una función $F : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^m$, sea $\mathcal{L}(F)$ el conjunto de funciones de $\mathbb{Z}_2^n$ en $\mathbb{Z}_2$ que son combinaciones lineales de las cooordenadas de F. La no linealidad de F se define entonces como $\mathcal{N}(F) = d(\mathcal{L}(F), \mathcal{A}_n)$, donde $\mathcal{A}_n$ es el conjunto de funciones afines de $\mathbb{Z}_2^n \mapsto \mathbb{Z}_2$, es decir:

$$\mathcal{N}(F) = \mathrm{Min}_{y,z \in \mathbb{Z}_2^n, b \in \mathbb{Z}_2} \#\{x \in \mathbb{Z}_2^n | z.F(x) \neq y.x \oplus b\}$$

donde $x.y = x_1y_1 \oplus ... \oplus x_ny_n$ *y similar para* $x.z$.

Observación: Puesto que si $z.F(x) \neq y.x \oplus b$, *entonces* $z.F(x) = y.x \oplus (1 \oplus b)$, *y como el minimo se toma sobre todos los bs, tenemos que:*

$$\mathcal{N}(F) = \mathrm{Min}_{y,z \in \mathbb{Z}_2^n, b \in \mathbb{Z}_2} \#\{x \in \mathbb{Z}_2^n | z.F(x) = y.x \oplus b\}$$

La no linealidad de F *esta definida básicamente en terminos de la no linealidad de las combinaciones lineales de sus coordenadas, asi que veremos más en detalles la no linealidad de funciones de* $\mathbb{Z}_2^n$ *en* $\mathbb{Z}_2$. *La no linealidad de estas funciones está muy relacionada con la transformada de Walsh. (tambien llamada de Walsh-Hadamard):*

9.2 Definición :

La transformada de Walsh de $f : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2$ es una función $\mathcal{W}(f) : \mathbb{Z}_2^n \mapsto \mathbb{R}$ dada por:

$$\mathcal{W}(f)(y) = \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x) \oplus y.x}$$

9.3 Teorema :

Para toda $f : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2$ se tiene:

$$\mathcal{N}(f) = 2^{n-1} - \frac{1}{2}\text{Max}_{y \in \mathbb{Z}_2^n}|\mathcal{W}(f)(y)|$$

Prueba:

Dado $y \in \mathbb{Z}_2^n$, sea $A = d(f, y.x \oplus 1)$ y $B = d(f, y.x)$. Tenemos entonces que $A = \#\{x \in \mathbb{Z}_2^n | f(x) = y.x\}$ y $B = \#\{x \in \mathbb{Z}_2^n | f(x) \neq y.x\}$, por lo tanto $A + B = 2^n$. Ademas:

$$\begin{aligned} A - B =& \#\{x \in \mathbb{Z}_2^n | f(x) = y.x\} - \#\{x \in \mathbb{Z}_2^n | f(x) \neq y.x\} \\ =& \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x) \oplus y.x} \\ =& \mathcal{W}(f)(y) \end{aligned}$$

Por lo tanto, $A = \frac{1}{2}(2^n + \mathcal{W}(f)(y))$ y $B = \frac{1}{2}(2^n - \mathcal{W}(f)(y))$.

Entonces:

$$\begin{aligned} \mathcal{N}(f) =& d(f, \mathcal{A}_n) \\ =& \text{Min}_y \text{Min}\{d(f, y.x), d(f, y.x \oplus 1)\} \\ =& \text{Min}_y \text{Min}\{\frac{1}{2}(2^n - \mathcal{W}(f)(y)), \frac{1}{2}(2^n + \mathcal{W}(f)(y))\} \\ =& \text{Min}_y \frac{1}{2}(2^n - |\mathcal{W}(f)(y)|) \\ =& 2^{n-1} - \text{Max}_y |\mathcal{W}(f)(y)| \end{aligned}$$

QED.

9.4 Teorema :

Si S es un S-box de $\mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$, y Γ, Λ son mascaras, entonces

$$LP(\Gamma, \Lambda) = (\frac{\mathcal{W}(S.\Lambda)(\Gamma)}{2^n})^2$$

Prueba:

Usando el mismo truco que en el teorema anterior, vemos que

$$\#\{x \in \mathbb{Z}_2^n : x \cdot \Gamma = S(x) \cdot \Lambda)\} = \frac{1}{2}(2^n + \mathcal{W}(S.\Lambda)(\Gamma))$$

por lo tanto

$$\begin{aligned} LP(\Gamma, \Lambda) =& (2.\frac{\#\{x \in \mathbb{Z}_2^n : x \cdot \Gamma = S(x) \cdot \Lambda)\}}{2^n} - 1)^2 \\ =& (2.\frac{1}{2}\frac{2^n + \mathcal{W}(S.\Lambda)(\Gamma)}{2^n} - 1)^2 \\ =& (\frac{\mathcal{W}(S.\Lambda)(\Gamma)}{2^n})^2 \end{aligned}$$

QED.

Si llamamos LP_{Max} a la mayor probabilidad lineal, lo cual mide la resistencia a LC (dada por tener un LP_{Max} chico), vemos que esta inversamente relacionada con la no linealidad del S-box: (mientras mas grande la no linealidad, mejor la resistencia):

9.5 Corolario :

Si S es un S-box de $\mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$, entonces

$$\mathcal{N}(S) = 2^{n-1}(1 - \sqrt{LP\text{Max}})$$

Prueba:

De **9.4.** vemos que $|\mathcal{W}(S.\Lambda)(\Gamma)| = 2^n\sqrt{LP(\Gamma,\Lambda)}$. Usando esto en **9.3.** , obtenemos el resultado. QED.

10. Nyberg, segunda parte

Una que necesitaremos es lo siguiente. Recordemos que podemos mirar los elementos del espacio vectorial $\mathbb{Z}_2^n$ como elementos del cuerpo $GF(2^n)$.

10.1 Definición :

La traza $Tr : GF(2^n) \mapsto GF(2^n)$ es la función

$$Tr(x) = x \oplus x^2 \oplus x^4 \oplus x^8 \oplus ... \oplus x^{2^{n-1}}$$

Debido a que $(a \oplus b)^2 = a^2 \oplus b^2$ en $GF(2^n)$ y usando que $x^{2^n} = x$, es elemental ver que $Tr(x)^2 = Tr(x)$, con lo cual deducimos que $Tr(x) \in \{0, 1\}$ para todo x. Entonces, en realidad podemos pensar que $Tr : GF(2^n) \mapsto \mathbb{Z}_2$. Usando otra vez que $(a \oplus b)^2 = a^2 \oplus b^2$, es fácil ver que Tr es una función lineal de $GF(2^n)$ en $\mathbb{Z}_2$, cuando se lo mia a $GF(2^n)$ como $\mathbb{Z}_2$-espacio vectorial.

Ademas, dado $y \in GF(2^n)$ la función $x \mapsto xy$ es $\mathbb{Z}_2$-lineal, por lo tanto $x \mapsto Tr(xy)$ tambien es lineal.

10.2 Teorema :

$f : GF(2^n) \mapsto \mathbb{Z}_2$ es lineal si y solo si existe $y \in GF(2^n)$ tal que $f(x) = Tr(xy)$.

Prueba:

El "si"lo vimos en los comentarios previos al teorema. Para ver el "solo si", observemos primero que Tr no es un mapa trivial, pues al mirarla como función de $GF(2^n)$ en $GF(2^n)$, es un polinomio de grado 2^{n-1}, el cual puede tener a lo sumo 2^{n-1} raices. Por lo tanto, hay (al menos 2^{n-1}) xs tales que $Tr(x) \neq 0$, asi que Tr no es la función identicamente cero.

Pero entonces, supongamos que $Tr(xy) = Tr(xz)\ \forall x$. Por la linealidad, $Tr(x(y \oplus z)) = 0 \forall x$ (1). Si $y \neq z$, entonces existe la inversa $(y \oplus z)^{-1}$ en $GF(2^n)$. Por lo tanto, dado $\tilde{x} \in GF(2^n)$, sea $x = \tilde{x}(y \oplus z)^{-1}$. Por (1), tenemos que $Tr(\tilde{x}) = Tr(x(y \oplus z)) = 0$ para todo $\tilde{x}$, lo cual es absurdo. Asi, vemos que $Tr(xy) = Tr(xz)\ \forall x \Rightarrow y = z$.

Entonces el conjunto de funciones $\{x \mapsto Tr(xy)\}_{y \in GF(2^n)}$ es un conjunto de 2^n transformaciones lineales. Pero toda transformación lineal de $GF(2^n) \mapsto \mathbb{Z}_2^n$ se representa por una matriz $1 \times n$ con elementos en $\mathbb{Z}_2$, asi pues, hay solo 2^n transformaciones lineales de $GF(2^n) \mapsto \mathbb{Z}_2$.Por lo tanto, el conjunto $\{x \mapsto Tr(xy)\}_{y \in GF(2^n)}$ es el conjunto de TODAS las transformaciones lineals de $GF(2^n) \mapsto \mathbb{Z}_2^n$.

QED.

10.3 Corolario :

$$\forall v \in \mathbb{Z}_2^n \exists y \in GF(2^n) \text{ tal que } x.v = Tr(xy) \forall x \in GF(2^n)$$

El teorema de Nyberg decia que el S-box dado por inversión en el cuerpo finito $GF(2^n)$ era óptimo respecto de DC. En realidad, Nyberg tambien probó que ese S-box tambien tiene buenas propiedades respecto de LC.:

10.4 Teorema (Nyberg, segunda parte) :
Sea S el S-box dado por $S(x) = x^{-1}$ si $x \neq 0$ y $S(0) = 0$, donde la inversión se hace en el cuerpo finito $GF(2^n)$.

Entonces, $LP(\Gamma, \Lambda) \leq 2^{-n+2}$.

Prueba:

Por **10.3.** , existen z, y tales que $x \cdot \Gamma = Tr(xz)$ y $S(x) \cdot \Lambda = Tr(S(x)y)$. Por lo tanto, usando la linealidad de Tr, tenemos que $x \cdot \Gamma \oplus S(x) \cdot \Lambda = Tr(xz \oplus S(x)y)$. Ademas, como suponemos que $\Gamma, \Lambda \neq 0$, tenemos que $z, y \neq 0$. Entonces:

$$\begin{aligned}
\mathcal{W}(S.\Lambda)(\Gamma) &= \sum_{x \in GF(2^n)} (-1)^{x \cdot \Gamma \oplus S(x) \cdot \Lambda} \\
&= \sum_{x \in GF(2^n)} (-1)^{Tr(xz \oplus S(x)y)} \\
&= \sum_{x \in GF(2^n)} (-1)^{Tr(xz \oplus x^{-1}y)} \\
&= \sum_{u \in GF(2^n)} (-1)^{Tr(u \oplus u^{-1}zy)} \qquad \text{(substituimos } u = xz\text{)}
\end{aligned}$$

Esta es una suma de Kloosterman, la cual se sabe (cota de Weil-Carlitz-Uchiyama, ver [CU57]) que es menor o igual a $2\sqrt{2^n} = 2^{\frac{n}{2}+1}$. Por lo tanto:

$$\begin{aligned}
LP(\Gamma, \Lambda) =& (\frac{\mathcal{W}(S.\Lambda)(\Gamma)}{2^n})^2 \qquad \text{(por } \mathbf{9.4}\text{.)} \\
\leq& (\frac{2^{\frac{n}{2}+1}}{2^n})^2 \\
=& (2^{-\frac{n}{2}+1})^2 \\
=& 2^{-n+2}
\end{aligned}$$

QED.

En particular, el S-box de Rijndael tiene LPs acotadas por 2^{-6}. Al igual que en DC, se ve que debe haber 25 S-boxes activos (respecto de un ataque de LC) en 4 rondas consecutivas de Rijndael, asi que la probabilidad lineal total debe ser de a lo sumo $c = (2^{-6})^{25} = 2^{-150}$, y por lo tanto, se necesitan aproximadamente 2^{150} textos para intentar el ataque, más de los que hay.

10.5 Observación :

Asi como existe un differential branch number, existe un linear branch number. Para una matriz cualquiera, en general no son iguales. Sin embargo, se puede ver que el lineal branch number de una matriz es igual al

differential branch number de la transpuesta. Por lo tanto, como vimos que una matriz es MDS si y solo si su transpuesta lo es (ver **6.6.** *), tenemos que en el caso de matrices MDS, el linear branch number y el differential branch number coinciden, por lo que la cuenta de arriba sigue siendo válida.*

11. Debilidad Lineal del S-box de Nyberg

Como vimos, el S-box de Nyberg es altamente no lineal, con cualquiera de las dos medidas de no linealidad que usemos. Sin embargo, sorprendentemente se descubrió que las coordenadas del S-box de Rijndael estan linealmente relacionadas entre si!!. Luego de este descubrimiento diversos autores probaron que esta es una caracteristica de cualquier S-box de Nyberg:

11.1 Teorema :

Las componentes booleanas de un S-box de Nyberg estan todas linealmente relacionadas: si denotamos por $S_i(x)$ la i-ésima componente de un S-box de Nyberg, entonces para todo i, j existe una transformación lineal $L_{i,j} : \mathbb{Z}_2^n \mapsto \mathbb{Z}_2^n$ tal que $S_j = S_i \circ L_{i,j}$

Prueba:

Sea e_i el vector con un 1 en la i-ésima coordena y cero en las otras. Entonces existe un $c_i \in GF(2^n)$ tal que $x.e_i = Tr(c_i x)$ para todo x. Por lo tanto:

$$\begin{aligned} S_j(x) =& e_j.S(x) \\ =& Tr(c_j S(x)) \\ =& Tr(c_j x^{-1}) \\ =& Tr(c_i c_i^{-1} c_j x^{-1}) \\ =& Tr(c_i (c_i c_j^{-1} x)^{-1}) \\ =& Tr(c_i S(c_i c_j^{-1} x)) \\ =& e_i.S(c_i c_j^{-1} x) \\ =& S_i(c_i c_j^{-1} x) \end{aligned}$$

y por supuesto $x \mapsto c_i c_j^{-1} x$ es lineal sobre $\mathbb{Z}_2$.

QED.

Todavia no se ha podido explotar esto, pero a la mayoria de los criptógrafos les preocupa una relación tan lineal entre las coordenadas de un S-box supuestamente no lineal.

12. Otros ataques sobre Rijndael

Como hemos visto, Rindael tiene una enorme resistencia contra ataques DC o LC. Tambien fue construido teniendo en cuenta resistencia contra ataques de interpolación, por ejemplo. Ahora bien, si ya 4 rondas son resistentes, ¿para que usar 10? La razon es que sus propios creadores diseñaron un ataque sobre 4 rondas de Rijndael, que puede ser extendido a 6 rondas. Por lo que agregaron 4 rondas de "buffer" contra este ataque. El ataque fue llamado el ataque SQUARE por los autores, porque primero lo aplicaron contra el cifer SQUARE, que fue un antecesor de Rijndael. Este ataque se puede usar contra una variedad de cifers, y más recientemente algunos autores le han llamado el ataque de **criptoanálisis integral**. *No tenemos mucho*

tiempo de explicar este ataque aqui, pero se apoya fuertemente en las propiedades de inversibilidad de cada uno de los pasos de Rijndael. En vez de mandar un par de mensajes con una diferencia preestablecida, este ataque manda 256 textos, que coinciden en todos los S-boxes menos uno, y en ese S-box se mandan todos los mensajes posibles. Luego se observan las propiedades de este conjunto de mensajes, y se deducen propiedades que deberia tener el conjunto de salida, si es que se adivinó el segmento de clave correspondiente al S-box correctamente.

Esto permite adivinar la clave, en vez de trabajando sobre una clave de 128 bits, trabajando sobre 16 subclaves de 8 bits cada una, lo cual es muy rápido de hacer. (i.e., la diferencia entre 2^{128} *y* 16.2^8*), pero no se ha podido extender este ataque a las 10 rondas completas.*

Ataques Algebraicos

Los ataques más prometedores contra Rijndael tienen que ver con su rica estructura matemática.

Mas prometedor es el uso de ataques llamados "algebraicos", básicamente usando bases de Grobner. Es decir, el S-box de Rijndael, por diseño, no tiene relaciones lineales entre sus componentes. Pero por definición, si uno "desprende" la composición afin en la definición del S-box y se queda con la parte "pura" de Nyberg, si le llamamos $y = S(x)$*, se tiene que* $xy = 1$*, es decir, una relación cuadratica entre la entrada y la salida. En el ataque XSL, se genera un sistema de (miles de) ecuaciones (no lineales) de ese tipo, que sus autores proclaman que seria capaz de ser resuelto, al menos en teoria, con solo uno o dos textos encriptados. Por supuesto, ningun sistema conocido actual es capaz de resolver mucho más de a lo sumo unas decenas de tales ecuaciones, asi que por ahora el AES es seguro contra este ataque. Mas aun, no se sabe si en realidad el ataque funciona o no:*

Recordemos que resolver un sistema de ecuaciones lineales es bastante fácil. Sin embargo, resolver un sistema de ecuaciones cuadraticas ya es un problema NP-completo. Ahora bien, supongamos que tenemos un sistema de m *ecuaciones en* n *variables que esta sobredefinido, es decir, con* $m > n$*. En ese caso, si* m *es suficientemente más grande que* n*, se conjetura que el sistema sí puede ser resuelto en tiempo polinomial. Brevemente, cada termino de la forma* $x_i x_j$ *es reeemplazado por una nueva variable,* $y_{i,j}$*, con lo que se obtiene un nuevo sistema, con más variables, pero lineal. Si* m *es suficientemente grande, el sistema lineal es determinado y puede ser resuelto. Luego de ellos, se resuelve el sistema original evaluando las posibilidades para, digamos,* x_i *y propagando con los valores conocidos de* $y_{i,j}$*. Si en el sistema original no habia suficientes ecuaciones, se lo plantea igual, y se resuelve lo que se puede, con lo cual se eliminan algunas variables. Luego se introducen nuevas identidades, como por ejemplo* $y_{i,j}y_{k,r} = y_{i,k}y_{j,r}$ *(pues ambos miembros representan* $x_i x_j x_k x_r$*, lo cual suma nuevas ecuaciones, pero ahora cuadraticas en los* y*, y entonces se relinearíza ESTE sistema, y se continua. Un problema puede ser que ahora las ecuaciones introducidas no sean linealmente independientes. En el algoritmo XL, se crean nuevas ecuaciones con otros metodos algebraicos. (la clave esta en que dependencias algebraicas entre ecuaciones pueden inducir no obstante ecuaciones* linealmente *independientes). No esta claro cual es la complejidad del ataque XL. Ahora bien, en el caso del AES, el sistema de ecuaciones que se deduce es tambien muy ralo, ("sparse") es decir, la mayoria de los coeficientes son cero. Courtois y Pieprzyk en [CP02] modificaron el metodo XL para trabajar sobre sistemas ralos, y obtuviereon el sistema XSL. Ellos estimaron que se necesita un trabajo de complejidad alrededor de* 2^{230} *para quebrar el AES, lo cual es peor que fuerza bruta, pero, cuando aplicaron su ataque a la version de clave de 256 bits de Rijndael, predicen un ataque de complejidad* 2^{255}*, lo cual no es realmente espectacular, pero*

mejora el ataque de fuerza bruta.

Posteriormente, se probó que se puede embeber el AES en un cifer más grande (el BES: big encryption system) en donde todas la operaciones se hacen en $GF(2^8)$ y se obtuvo un sistema de 5248 ecuaciones en 2560 variables del cifer en si más 1408 variables de la clave. Esto sin contar las ecuaciones que se pueden derivar de la expansion de la clave en las claves de ronda, lo cual darian otras 2500 ecuaciones, aunque algunas pueden no ser independiente de las anteriores. Los autores estiman que se podria resolver este sistema para AES (i.e. Rijndael de 128 bits) con un trabajo de complejidad equivalente a solo 2^{100} encripciones, lo cual, si bien impracticable, se considera un "quiebre" teórico de AES. Sin embargo, estas estimaciones dependen bastante de una serie de dudosas estimaciones en el trabajo original de Courtois y Pieprzyk y muchos criptografos en realidad dudan de ellas.

El siguiente intercambio producido en la web entre los autores de Rijndael y los del ataque XSL es interesante:

"El ataque XSL no es más que un sueño" -creadores de Rijndael.

"Es cierto, es Rijndael peor pesadilla" -creadores del XSL.

Para terminar, quiero recordar una frase de un criptólogo famoso pero que no me acuerdo quien es (creo que Shannon). Decia mas o menos asi:

"Criptography is a mixture of mathematics and muddle. Mathematics provide security, but without the muddle, the mathematics can be turned against you".

Rijndael es un excelente cifer matemático, pero quizas no tiene el suficiente "muddle".

13. Apendice: Implementación Eficiente

Implementación eficiente en procesadores modernos *Antes de hablar de esto, dejenme acalarar que la implementación tambien se realiza en forma muy eficiente (en comparación con otros cifers) en procesadores de 8 bits o en hardware, gracias a la simplicidad de los coeficientes de la matriz MDS.*

Pero una de las razones por las cuales Rijndael se convirtió en el AES tiene que ver con su rapidez en procesadores de 32 bits. Esto es porque su estructura esta diseñada matemáticamente para tomar ventaja de estos procesadores.

Para comprender porqué, denotemos el bloque como una matriz A 4×4 sobre $GF(2^8)$.

Recordemos que cada ronda de Rijndael es de la forma: $\kappa \circ \mu \circ \rho \circ \sigma$ donde σ es el pasaje por los S-boxes, ρ es ShiftRow, μ es MixColumn y κ es KeyMixing. La ultima ronda no tiene μ. Como dijimos, en total son 10 rondas, y hay ademas un κ adicional al principio de todo el cifer.

Llamemos $B = \sigma(A)$, $C = \rho(B)$ y $D = \mu(C)$. Sea M la matriz 4×4 MDS usada en el cifer. Observar que como M se aplica a cada columna, esto es lo mismo que decir que $D = MC$. Si numeramos las filas y las columnas desde 0 a 3, entonces $C_{i,j} = B_{i,j+i \bmod 4}$ y $B_{i,j} = S(A_{i,j})$, donde S es el S-box.

Entonces, tenemos que:

$$\begin{aligned} D_{i,j} &= \sum_{m=1}^{4} M_{i,m} C_{m,j} \\ &= \sum_{m=1}^{4} M_{i,m} S(A_{m,j+m \bmod 4}) \end{aligned}$$

Ahora bien, los procesadores de 32 bits tienen registros de, justamente, 32 bits, es decir, pueden guardar 4 bytes. En particular, una transformación $T : GF(2^8) \mapsto GF(2^8)^4$ puede ser eficientemente guardada pues tenemos que guardar simplemente una tabla de 256 registros de 32 bits, lo cual no es un problema.

Entonces, si definimos:

$$T_m(x) = \begin{bmatrix} M_{0,m}S(x) \\ M_{1,m}S(x) \\ M_{2,m}S(x) \\ M_{3,m}S(x) \end{bmatrix} \qquad m = 1, 2, 3, 4$$

estas 4 tranformaciones se pueden guardar eficientemente en procesadores modernos.

Tenemos entonces que la j-esima columna de D es:

$$D^{(j)} = \sum_{m=1}^{4} T_m(A_{m,j+m \bmod 4})$$

Es decir, calcular la matriz D lleva sólo 4 lecturas de tablas y cuatro sumas por cada columna. La matriz final solo requiere un ultmo XOR con la clave. Recordando que la suma de $GF(2^8)$ es la misma que la de $\mathbb{Z}_2^8$, es decir, el XOR bit a bit, la cual es eficientemente implementada en los computadores, el calculo de una ronda de AES es muy rápido, pues solo requiere un total de 16 lecturas de tablas, y 20 XORs.

Implementación de la desencriptación

El metodo anterior funciona muy bien, pero como para desencriptar hay que hacer las operaciones en sentido inverso, ya no podemos implementarlo tan eficientemente. O al menos eso parece, pero los diseñadores del AES pensaron en esto y diseñaron la estuctura para permitir tambien una desencriptación rapida.

El truco aca es que si denotamos por k_i la clave de la ronda i, y k_0 la clave de whitening entonces, recordando que la ultima ronda no tiene MixColumn, tenemos que :

$$AES = (\kappa_{k_{10}} \circ \rho \circ \sigma) \circ (\kappa_{k_9} \circ \mu \circ \rho \circ \sigma) \circ ... \circ (\kappa_{k_1} \circ \mu \circ \rho \circ \sigma) \circ \kappa_{k_0}$$

Pero, primero, σ y ρ claramente conmutan. Por otro lado, $\kappa_k \circ \mu(A) = MA + k = M(A + M^{-1}k) = \mu \circ \kappa_{M^{-1}k}$, es decir, μ y κ tambien "conmutan", si se tiene la precaución de cambiar la clave. Llamemos $\tilde{k}_i = M^{-1}k_i$.

Asi, podemos describir la ronda i como $\mu \circ \kappa_{\tilde{k}_i} \circ \sigma \circ \rho$, por lo que, teniendo en cuenta que $\kappa^{-1} = \kappa$ y agrupando convenientemente:

$$\begin{aligned} AES^{-1} =& \kappa_{k_0} \circ (\rho^{-1} \circ \sigma^{-1} \circ \kappa_{\tilde{k}_1} \circ \mu^{-1}) \circ (\rho^{-1} \circ \sigma^{-1} \circ \kappa_{\tilde{k}_2} \circ \mu^{-1}) \circ ... \\ & ... \circ (\rho^{-1} \circ \sigma^{-1} \circ \kappa_{\tilde{k}_9} \circ \mu^{-1}) \circ (\rho^{-1} \circ \sigma^{-1} \circ \kappa_{\tilde{k}_{10}}) \\ =& (\kappa_{k_0} \circ \rho^{-1} \circ \sigma^{-1}) \circ (\kappa_{\tilde{k}_1} \circ \mu^{-1} \circ \rho^{-1} \circ \sigma^{-1}) \circ ... \circ (\kappa_{\tilde{k}_9} \circ \mu^{-1} \circ \rho^{-1} \circ \sigma^{-1}) \circ \kappa_{\tilde{k}_{10}} \end{aligned}$$

es decir, la misma estructura de AES, solo que cambiando las claves, y reemplazando ρ por ρ^{-1}, que es tambien un ShiftRow, pero para el otro lado, y cambiando el S-box por el S-box inverso, y la multiplicación por M por multiplicación por M^{-1}. Por lo tanto, se puede implementar de la misma forma que el encriptamiento (cambiando las tablas T_m por otra tablas).

Gracias a este diseño, se puede implementar Rijndael muy eficientemente tanto para desencriptar como encriptar. (En realidad, hay una pequeña degradación en la perfomance, pero no debido a la desencriptación

en si, sino al proceso de expandir la clave maestra en las claves de ronda. La expansión es muy rapida para encriptar, pero no asi para las claves necesarias para desencriptar. De todos modos esto se nota solo si se encripta pocos bloques con una misma clave, puesto que se realiza una sola vez por clave).

(Nota: cuando se lo implementa en procesadores de menos de 32 bits, por ejemplo, en smartcards, no se pueden guardar estas tablas. Hay que realizar las operaciones sobre $GF(2^8)$. Pero los coeficientes de la matriz M son todos 1,2 o 3, es decir, representando $GF(2^8)$ mediante polinómios, corresponden con los polinómios $1, x$ y $1+x$, y multiplicación por x se puede implementar muy eficientemente. Sin embargo, los coeficientes de M^{-1} no son tan simples, con lo cual la implementación de la desencriptación es más lenta en estos casos).

14. Apendice: Estructura del DES

La sigla DES proviene de Data Encryption Standard. En los '70s, los EEUU decidieron que necesitaban un estandard criptográfico seguro que se pudiera usar en los bancos y otros lugares civiles. Luego de un concurso en el cual solo se presento la IBM, y luego de pasar por la NSA (National Security Agency), se propuso el DES. El DES fue finalmente quebrado en los '90s, usando un ataque de fuerza bruta. La razon es que el tamaño de la clave (56 bits) se consideraba suficientemente grande en los '70s, pero era insuficiente en los '90s. (en realidad, ya en los '70s se critico el tamaño de la clave: el proyecto original de IBM tenia una clave de 128 bits, pero la NSA la redujo a 56. Se supone que es porque la NSA tenia ya en los '70s computadoras capaces de quebrar una clave de 56 bits pero no una de 128). (Aun peor: durante mucho tiempo, DES para exportación debia tener una clave de solo 40 bits).

No vamos a explicar mucho de DES aca, excepto para decir que tenia 16 rondas, un bloque de 64 bits, y en cada ronda se procesaban solo 32 bits.

Basicamente, el mensaje se partia en dos partes: L y R, cada una de 32 bits. (L por left y R por right). Entonces, en cada ronda se procesaba una sola parte, digamos la R, pasando por una función F que dependia de una clave de ronda. Luego se hacia: $(L, R) := (L \oplus F_k(R), R)$, donde $\oplus$ es la suma en $\mathbb{Z}_2^{32}$. En la siguiente ronda, se hacia lo mismo, pero con otra clave e intercambiando los roles de L y R.

Matematicamente hablando, si denotamos por $\psi_k(L, R) = (L \oplus F_k(R), R)$ y por $\tau(L, R) = (R, L)$, (el "Feistel twist") entonces el cifer era:

$$\psi_{k_{16}} \circ (\tau \circ \psi_{k_{15}}) \circ (\tau \circ \psi_{k_{14}}) \circ .... \circ (\tau \circ \psi_{k_1})$$

donde $k_1, ..., k_{16}$ eran las claves de rondas, que eran básicamente un subconjunto de 48 bits de la clave maestra de 56 bits. (un subconjunto diferente para cada ronda, obviamente).

Observar que la ultima ronda no tenia el Feistel twist. Esto permite usar el mismo algoritmo para desencriptar: si denotamos la expresión anterior por $Feistel[k_1, ..., k_{16}]$, entonces es un ejercicio fácil ver que $Feistel[k_{16}, ..., k_1] \circ Feistel[k_1, ..., k_{16}] = Identidad$, es decir, para desencriptar DES (o cualquier cifer de tipo Feistel) se usa el mismo algoritmo, pero con las claves de ronda en el orden inverso.

Esto vale sea cual sea la F, aun si no es invertible.

En el caso de DES, la F hacia los siguiente:

Los 32 bits se expandian a 48 bits por medio de una tranformación lineal (que básicamente copiaba algunos bits en otros lugares). Luego los 48 bits se mesclaban con los 48 bits de la clave de ronda haciendo

simplemente la suma en $\mathbb{Z}_2^{48}$. Los 48 bits resultantes se partian en 8 subloques de 6 bits cada uno, y a cada uno de estos se le aplicaba una tranformación no lineal de $\mathbb{Z}_2^6$ en $\mathbb{Z}_2^4$. Estas tranformaciones se llamaban S-boxes ("substitution boxes"). Los 8 subloques de 4 bits que quedaban se juntaban de vuelta en 32 bits, y se le aplicaba una transformación lineal a los 32 bits, que era simplemente una matriz de permutación.

Si bien el diseño de DES es muy avanzado, tenia algunas fallas, más alla de su corta clave. Para empezar, la función F no es 1-1. Si bien no es necesario que lo sea, pues esta metida dentro de un cifer Feistel, esto da origen a ciertos ataques estadisticos. Ademas, la tranformación lineal final era solo una matriz de permutación, con lo cual no se producia una mezcla tan buena como la que se podria haber producido con otra transformación lineal más general. De hecho, DES necesita al menos 5 rondas para lograr un buen efecto de mezcla de los bits.

Lo más misteriosos de DES eran los S-boxes, pues en los '70s no se especificó porqué se habian elegido esas tranformaciones y no otras. Luego se reveló que habian sido construidos al azar pero testeando ciertos criterios para dar una cierta protección contra DC.

Bibliografia

[AES]: AES home page: `http://www.nist.gov/aes`

[BS90]: E. Biham, A. Shamir, "Differential Cryptanalysis of DES-Like Cryptosystems", Advances in Cryptology, CRYPTON 90, LNCS 537, Springer-Verlag, 1990, pp 2-21

[BS91]: E. Biham, A. Shamir, "Differential Cryptanalysis of DES-Like Cryptosystems", J. Of Cryptology, no 4, 1991, pp 3-72

[CU57]: L. Carlitz and S. Uchiyama, "Bounds for Exponential Sums", Duke Mathematical Journal v. 24, 1957, pp. 37-41

[CP02]: N. Courtois, J. Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations", Proceedings of AsiaCrypt02, LNCS 2501, Springer-Verlag 2002.

[DKR97]: J. Daemen, L.R.Knudsen, V. Rijmen, "The Block Cipher SQUARE", Fast Software Encryption, LNCS 1267, E. Biham, Ed., Springer-Verlag, 1997, pp 149-165.

[DR99]: J. Daemen, V. Rijmen, "AES Proposal: Rijndael", AES algorithm submission, Septiembre, 1999, disponible en [AES].

[DR02]: J. Daemen, V. Rijmen, "The Design of Rijndael:AES-The Advance Encryption Standard". Springer-Verlag, 2002.

[JK97]: T.Jacobsen, L.R.Knudsen, " The interpolation attack on block ciphers", Fast Software Encription, LNCS 1267, E. Biham, Ed., Springer-Verlag 1997, pp 28-40.

[K95a]: L.R.Knudsen, "Applications of Higher Order Differentials and Partial Differential", K.U. Leuven Workshop on Cryptographic Algorithms, Springer-Verlag, 1995.

[K95b]: L.R.Knudsen, "Truncated and Higher Order Differentials", Fast Software Encryption 94, LNCS 1008, B. Preneel, Ed. Springer-Verlag, 1995, pp 196-211.

[KW02]: Lars Knudsen and David Wagner, "Integral Cryptanalysis (Extended Abstract)" ,
`http://citeseer.nj.nec.com/knudsen02integral.html`

[L94]: X. Lai "Higher Order Derivatives and Differential Cryptanalysis", Communications and Cryptography: Two Sides of One Tapestry, R.E. Blahut et al., eds., Kluwer Academic Publishers, 1994, pp

227-233.

[M90]S. Murphy, "The cryptanalysis of FEAL-4 with 20 chosen plaintexts", Journal of Cryptology, Vol. 2, No. 3, pp. 145-154, 1990.

[M93]: M.Matsui, "Linear Cryptanalyisis Method for DES Cipher", Advances in Cryptology-EUROCRYPT 93, LNCS 765, Springer-Verlag, 1993, pp 386-397

[MR02]: S. Murphy, M. Robshaw. "Essential algebraic structure within the AES", Proceedings of Crypto'02, LNCS 2442, pp 17-38, Springer-Verlag 2002

[N94]: K. Nyberg, "Differentially Uniform Mappings for Cryptography", Advances in Cryptology, EUROCRYPT 93, LNCS 765, Springer-Verlag, 1994, pp 55-64.

[PSCY02] S. Park, S.H. Sung, S. Chee, E-J. Yoon, and J. Lim, "On the security of Rijndael-like structures against differential and linear cryptanalysis", Advances in Cryptology ASIACRYPT 2002, LNCS 2501, pp. 176191, Springer-Verlag, 2002.

[PSLL03] S. Park, S.H. Sung, S. Lee, J. Lim, "Improving the upper bound on the maximum differential and the maximum linear hull probability for SPN structures and AES", Fast Software Encryption (FSE 2003)